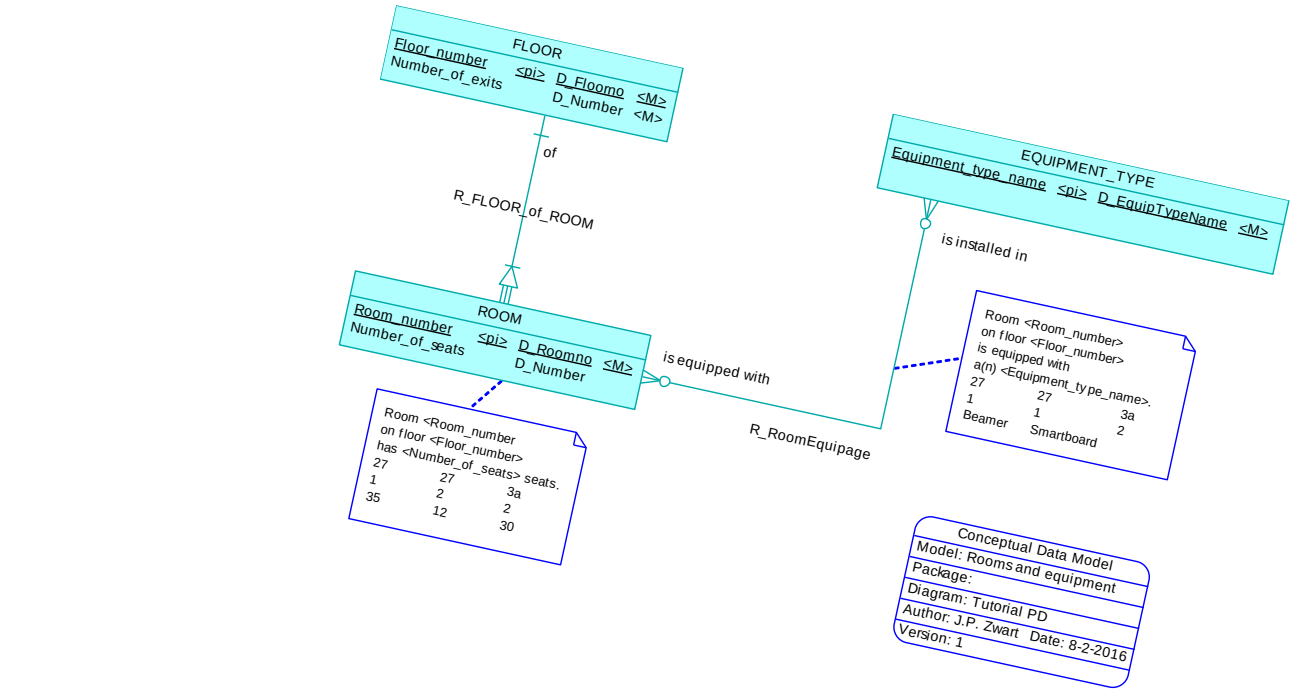
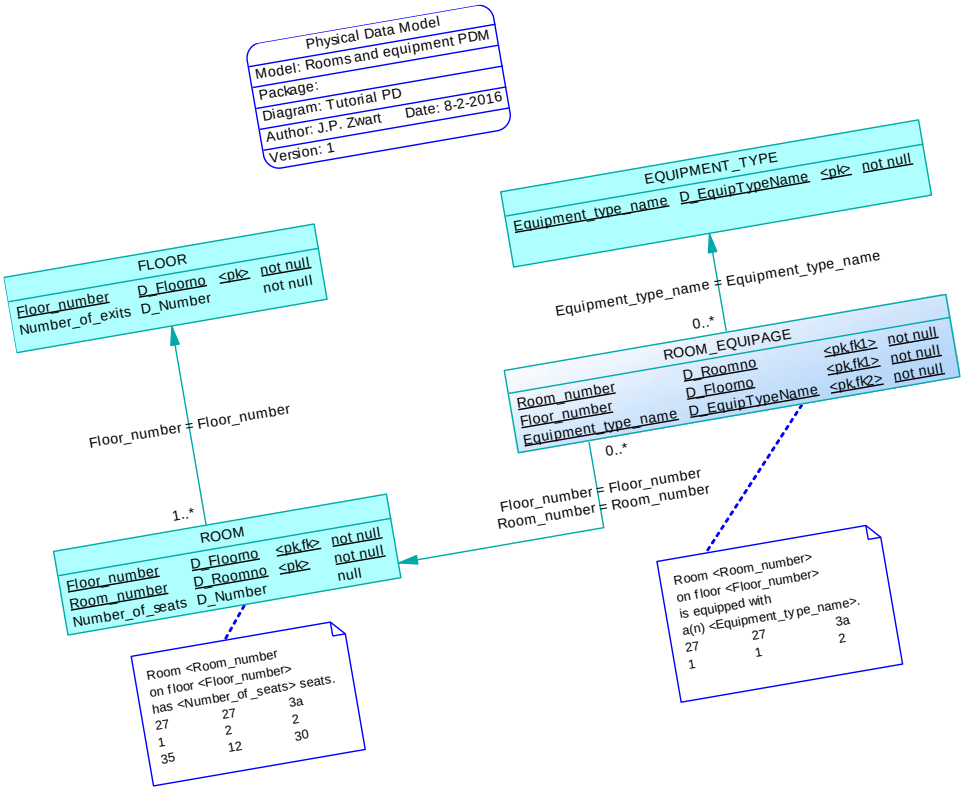

FLOOR
Floor_number <pi> D_Floorno <M>
Number_of_exits D_Number <M>
of
R_FLOOR_of_ROOM
EQUIPMENT_TYPE
Equipment_type_name <pi> D_EquipTypeName <M>
is installed in
ROOM
Room_number <pi> D_Roomno <M>
Number_of_seats D_Number
is equipped with
R_RoomEquipage
Room <Room_number>
on floor <Floor_number>
is equipped with
a(n) <Equipment_type_name>.
27 27 3a
1 1 2
Beamer Smartboard
Room <Room_number
on floor <Floor_number>
has <Number_of_seats> seats.
27 27 3a
1 2 2
35 12 30
Conceptual Data Model
Model: Rooms and equipment
Package:
Diagram: Tutorial PD
Author: J.P. Zwart Date: 8-2-2016
Version: 1


Physical Data Model
Model: Rooms and equipment PDM
Package:
Diagram: Tutorial PD
Author: J.P. Zwart Date: 8-2-2016
Version: 1
EQUIPMENT_TYPE
Equipment_type_name D_EquipTypeName <pk> not null
FLOOR
Floor_number D_Floorno <pk> not null
Number_of_exits D_Number not null
Equipment_type_name = Equipment_type_name
0..*
ROOM_EQUIPAGE
Room_number D_Roomno <pk,fk1> not null
Floor_number D_Floorno <pk,fk1> not null
Equipment_type_name D_EquipTypeName <pk,fk2> not null
Floor_number = Floor_number
1..*
0..*
Floor_number = Floor_number
Room_number = Room_number
ROOM
Floor_number D_Floorno <pk,fk> not null
Room_number D_Roomno <pk> not null
Number_of_seats D_Number null
Room <Room_number>
on floor <Floor_number>
is equipped with
a(n) <Equipment_type_name>.
27 27 3a
1 1 2
Room <Room_number
on floor <Floor_number>
has <Number_of_seats> seats.
27 27 3a
1 2 2
35 12 30

General information

Title	ISE-DMDD Tutorial PowerDesigner
Academic year	2017-2018
Training Programs	Information Science, profile Software Development (SD) Information Science, profile Enterprise Software Solutions (ESS) Information Science, profile Information Management & Consultancy (IMC)
Author	Jan Pieter Zwart

The English texts were written by Jan-Pieter Zwart, who also made all the figures.

Table of Contents

1 Use of this Reader

This reader provides a step-by-step tutorial to learn to use PowerDesigner 16.0. Some details may be different in later or older versions of PowerDesigner (16.6 is the current version).

It will be shown how to create a Conceptual Data Model (CDM), or ERM diagram, and how to derive from this CDM a Physical Data Model (PDM), or logical relational database schema. From this PDM, an SQL DDL script can be generated, to be fed into a Relational Database Management System (RDBMS), such as SQL Server, to create the (empty) database with.

The route in the opposite direction is frequently traveled in practice as well: read in an existing relational database structure, and reverse engineer the so obtained PDM into a CDM. This is done to make additions and/or changes in this reverse-engineered CDM, and then to regenerate the (altered) database structure. This will be done in case studies in the DMDD course, and will not be elaborated on in this tutorial.

This reader does not provide a systematic way to design a good CDM. See the reader Data Modeling and Relational Database Structures for this. It is assumed here that a good design has already been made, so the focus is on how to realize this design in PowerDesigner 16.0.

In addition, mandatory display settings for diagrams in the course DMDD are presented, and directions for the style of modeling are given as well.

2 Making a Conceptual Data Model (CDM)

2.1 The example CDM

We will build the CDM shown in figure 2.1. This is an Entity-Relationship Model (ERM) with three **Entity Type**s (ETs) FLOOR, ROOM and EQUIPMENT_TYPE, together with their **attribute**s (Atts) and their mutual **Relationship Type**s (RTs).

ET FLOOR has two attributes. The first is its **identifying** attribute Floor_number (the '<pi>' symbol indicates that this Att is the primary identifier, as does the underlining), to which the **domain** FLOOR_NO applies. The second is Number_of_exits, to which domain NUMBER applies. Both these Atts are **mandatory**, indicated by the symbol '<M>', which means that every entity of ET FLOOR must have a value in both these attributes.

ET ROOM also has two attributes: its identifying Att Room_number with domein ROOM_NO, which is mandatory, and the Att Number_of_seats with domain NUMBER. The third entity type EQUIPMENT_TYPE has only one identifying mandatory attribute Equipment_type_name with domain EQUIP_TYPE_NAME.

The data types that apply to the domains are not shown: data types are not considered to be a part of a conceptual data model. They are important at the physical data level however, and can already be specified at the conceptual level in PowerDesigner, which we will indeed do, but we will not show them in the CDM.

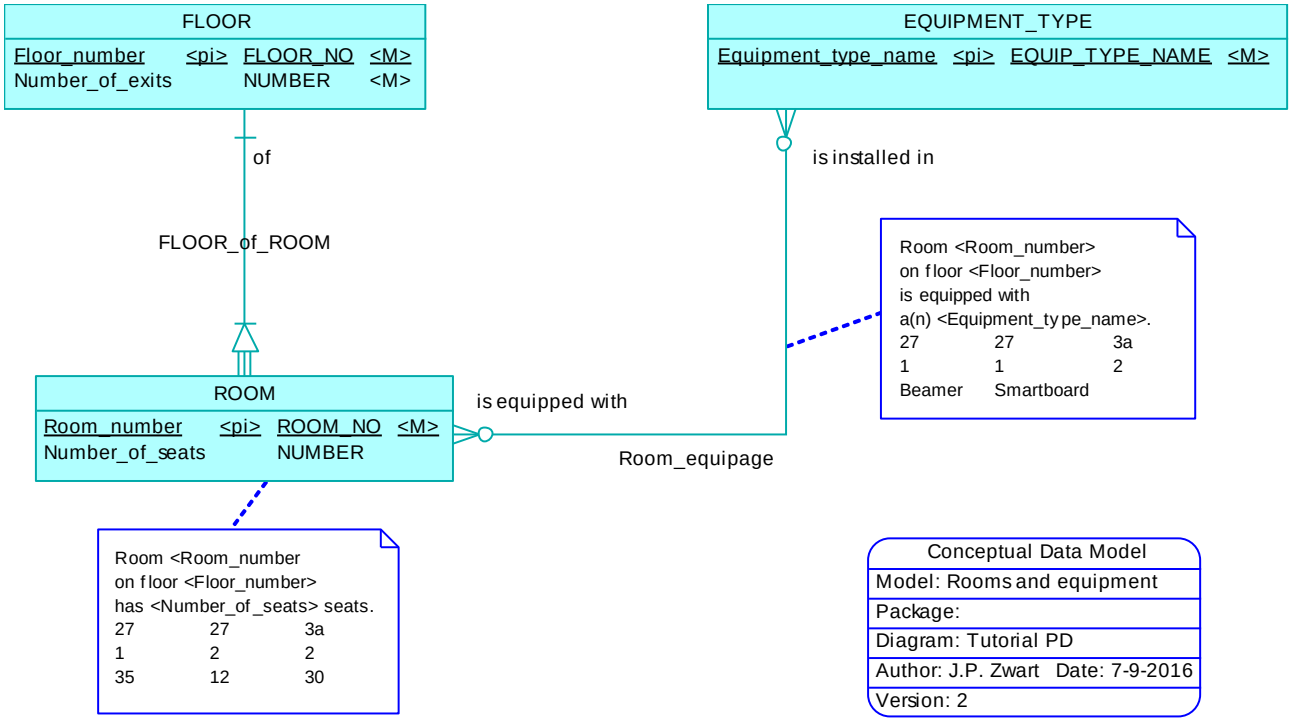


Figure 2.1 CDM for rooms with equipments

There is a relationship type Room_equipage between ETs ROOM and EQUIPMENT_TYPE. The texts 'is installed in' and 'is equipped with' serve to make the meaning of the RT clear. Such a clarifying text is called a 'role' in PowerDesigner. **Always supply at least one role for each RT.** This is a many-to-many RT: a room can be equipped with several types of equipments (beamer, whiteboard, smartboard, …), and a smartboard can be installed in several rooms. This is indicated by the **maximum cardinality** MANY (indicated by a crow's foot) at both sides of the RT. A room doesn't have to have a type of equipment installed, and there can be a type of equipment that is not installed in any room (like a robot). This is indicated by the **minimum cardinality** ZERO (indicated by the small '0') at both sides of the RT.

There also is a RT FLOOR_of_ROOM between ROOM and FLOOR. This models on which floor a room is situated. The **dependency triangle** indicates that ROOM is a **weak entity type**: it needs the <pi> of FLOOR as a part of its own identifier. Both the minimum and maximum cardinalities at the side of FLOOR are therefore ONE, because every entity of ET ROOM must be on exactly one floor. A floor can contain at least one and at most several rooms, so the other two cardinalities are ONE and MANY. Such dependent RTs always have a short formal role, like 'of' or 'in'.

2.2 Getting started practically

2.2.1 Windows, Toolbox, setting the notation style

Start PowerDesigner. In the 'Welcome to PowerDesigner' window, click 'Create Model', and in the following 'New Model' window, double-click 'Conceptual Data'. You will now see a window with tab 'Diagram_1'.

There are several notation styles for a Conceptual Data Model (CDM), such as Barker, IDEF1X, etc. We will use the 'Entity/Relationship' style. To set this style, right-click in an empty space in the 'Diagram_1' window, and from the pop-up menu choose option 'Model Options…'. In the 'Model Options' window that now appears, find the 'Notation' drop-down list and choose the 'Entity/Relationship' option. Click OK to return to 'Diagram_1'.

At the right-hand side of your screen, there shpuld be a list 'Toolbox'. If you don't see the Toolbox, in the main menu click 'View' → 'Toolbox'. Any other windows can be minimized or closed (and reopened with main menu option 'View').

2.2.2 Display preferences for CDM

Let's set the default display preferences first. **These settings are mandatory for all CDMs you want to communicate with your teachers.** In the 'Diagram_1' window, right-click in an empty space and choose 'Display Preferences…' → 'Entity'. Set the options on tab 'Content' as shown in figure 2.2, and click the 'Set As Default' button. On the tab 'Format', click the 'Modify' button, and in the 'Symbol Format' window, on the 'Size', mark checkbox 'Auto adjust to text'. Click 'OK', then 'Set as Default'.

Repeat the step above for: 'Display Preferences…' → 'Relationship'. Set the options on tab 'Content' as shown in figure 2.3 and click the 'Set As Default' button.

Make sure you always use these settings.

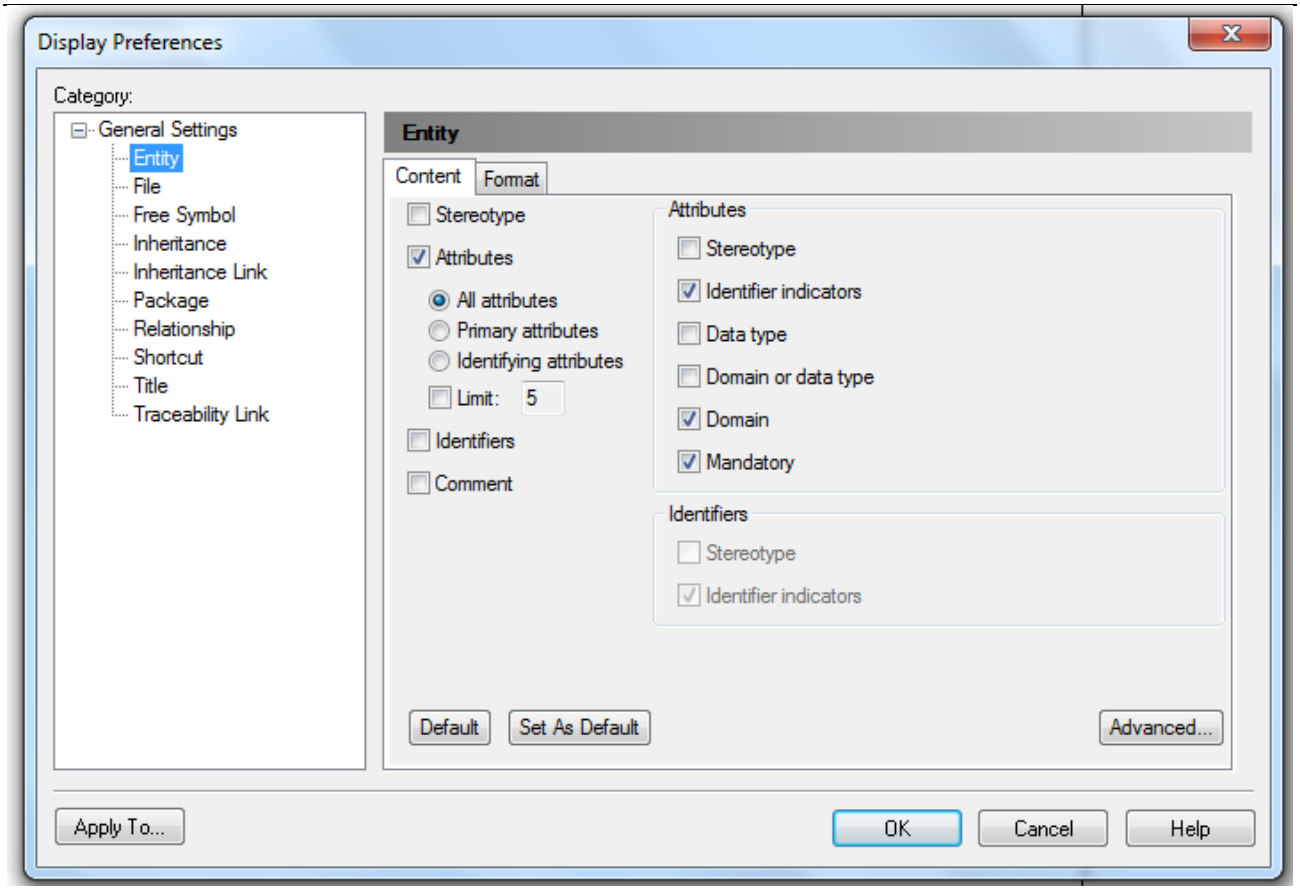


Figure 2.2 Mandatory display settings for ETs

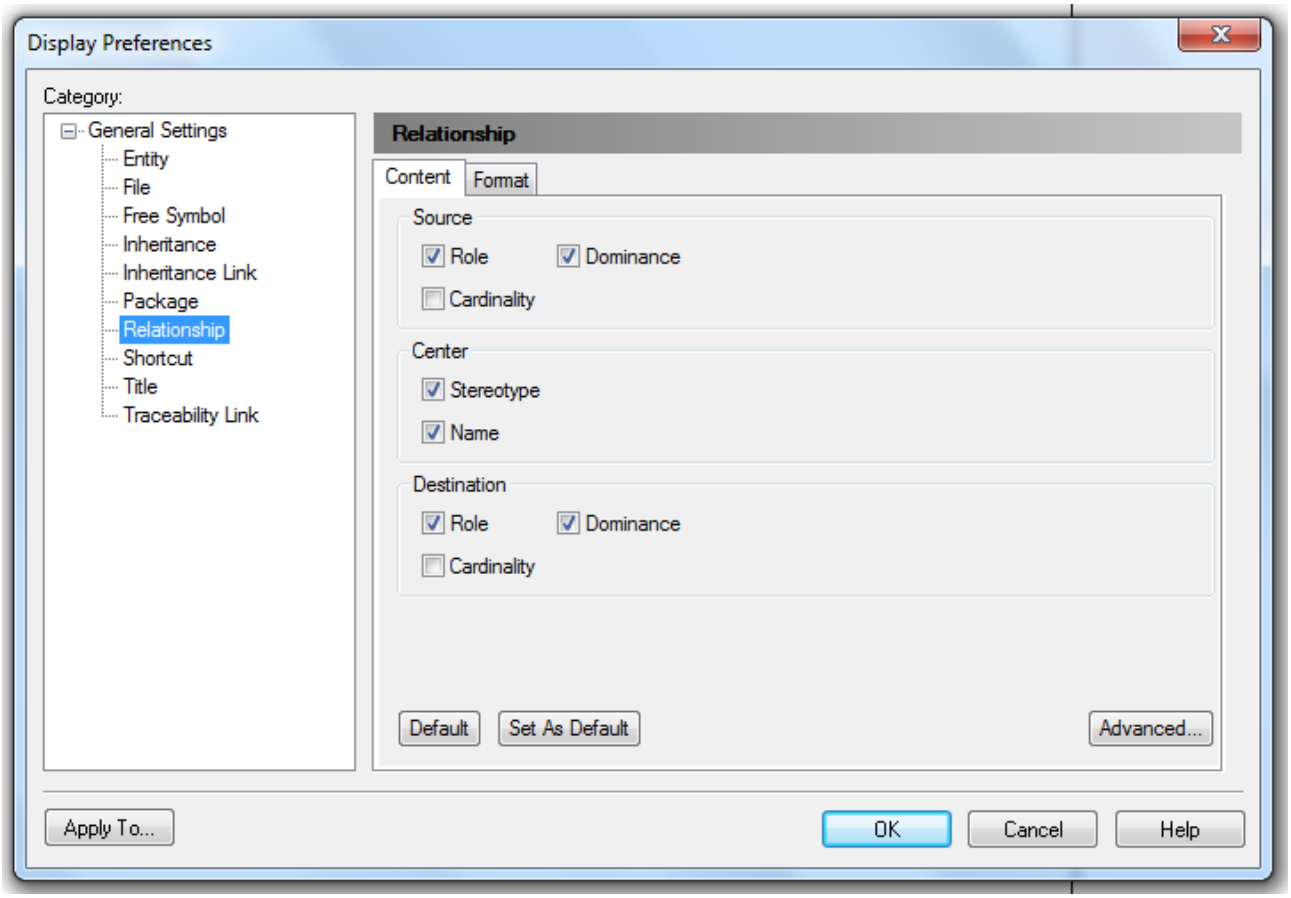


Figure 2.3 Mandatory display settings for RTs

2.2.3 CDM Title

From the Toolbox, in the 'Free Symbols' section, click 'Title'. Move the cursor to the 'Diagram 1' window, and click once in an empty spot. A title (rounded rectangle named 'Conceptual Data Model') appears. Right-click to deactivate the Toolbox, then double-click the title and fill in the 'Model Properties' window (Name, Author, Version, …). **Always add a relevant title to all your CDMs.**

2.2.4 Adding ETs, Atts and domains

First, we'll create the entity types (ETs) with their attributes (Atts) and domains.

In the Toolbox, in the 'Conceptual Diagram' section, click on 'Entity'. Move the cursor to the CDM window (it was called 'Diagram_1', but you have hopefully changed this to a more meaningful name in the Title) and click three times in separate spots: three empty ETs appear. Right-click anywhere to deactivate the Toolbox.

Go to the first ET and double-click it. A window 'Entity Properties' appears. We will fill this in to create ET FLOOR:

- On tab 'General', enter the following:
  - Name: FLOOR (Code: is used by PowerDesigner internally, defaults to the name).
  - Comment and Number can be skipped.
  - Checkbox 'Generate' should be marked already (please check).
- Open tab 'Attributes'. You can customize the columns you see there using the 'Customize Columns and Filter' button (4th icon from the right). Make sure you can see the following columns: 'Name', 'Domain', 'Primary' ('P') and 'Mandatory' ('M'), as is shown in figure 2.4. Any other order of these four columns, and any other additional columns will do as well, but these are the ones we will use most, and it is convenient to arrange them as is shown.

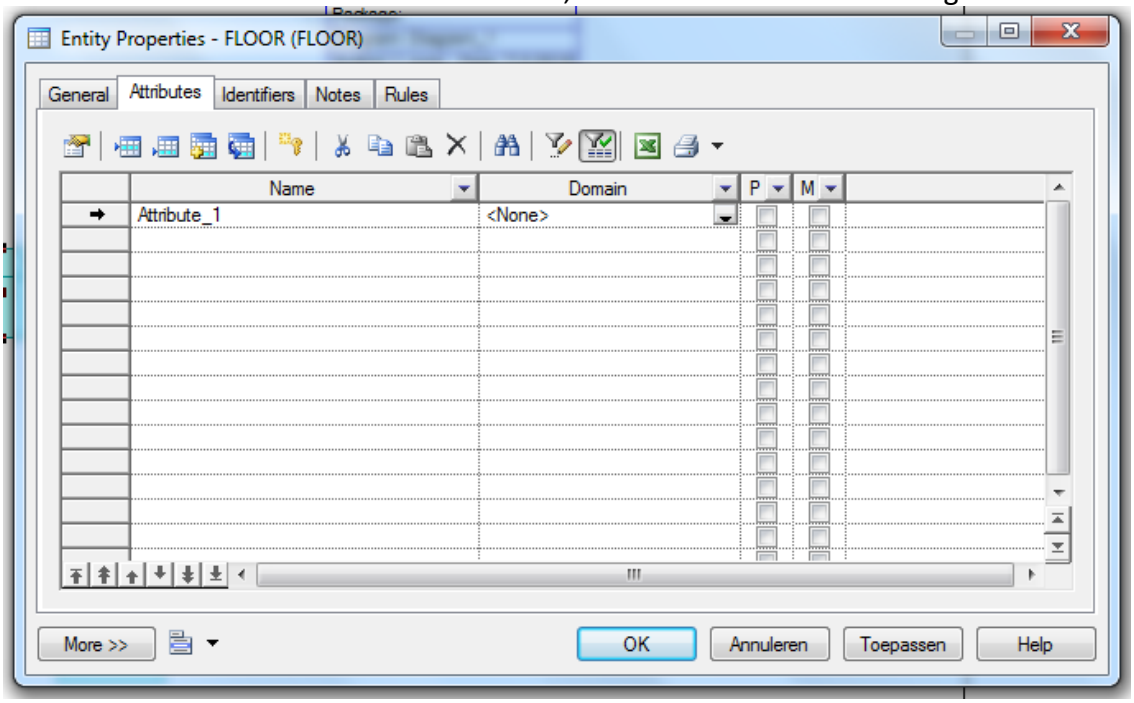


Figure 2.4 Most used columns on tab Attributes

- Place the cursor in column 'Name', top row, and click. Override the automatically generated name 'Attribute_1' with Floor_number'.
- Column 'Domain' now contains the text '<None>'. We'll have to create a new domain for this Att, because no domain has been made yet. This can be done in several ways, and we'll explore one of them here. In the main menu, click 'Model' → 'Domains…'. The 'List of Domains' window appears. In the top row, column 'Name', enter 'FLOOR_NO'.
- If you like, you can assign a data type to this domain as well: in column 'Data Type', click in the top row, and from the drop-down list there (not the one in the column name) choose 'Short integer'. You can adjust the column width etc. for better reading. Note: Adding a data type is not needed at the conceptual level, but if you do it here, PowerDesigner will take the data type along when we generate a logical relational model (PDM) and an SQL DDL script (otherwise you can add the data type later in the PDM).
- Click 'OK' to close the window and save the domain.
- In the 'Entity Properties' window, tab 'Attributes', row 'Floor_number', click in column 'Domain', and from the drop-down list there (not the one in the column title) choose the domain we just made (FLOOR_NO).
- Since this attribute is the <pi> of ET FLOOR, mark the checkbox in the 'P'-column. The checkbox in the 'M'-column is then marked automatically as well.
- Add the second attribute of FLOOR in the same way: 'Number_of_exits', new domain 'NUMBER', data type 'Short integer', checkbox 'P' empty, checkbox 'M' marked. Note: the two domains we made are different even though they have the same data type.
- Click OK to close the window and save the ET with its Atts and domains. FLOOR should now look exactly as is figure 2.1.
- Save the diagram somewhere (File → Save as…).

- Now we can do the other two ETs in the same way. See figure 2.1 for the desired result. Choose plausible data types (like 'character 4' for ROOM_NO). Check all other details and make sure your version of the three ETs is identical to that in figure 2.1. Save the diagram.

2.2.5 Adding RTs

First, we'll add the RT between ROOM and FLOOR.

- Place FLOOR above ROOM (by dragging the ETs), with a few centimeters distance in between. In the Toolbox, in the 'Conceptual Diagram' section, click on 'Relationship'. Move the cursor to hover over FLOOR, then click-and-hold-down, and drag the cursor to ROOM. Release the mouse button when the dursor hovers over ROOM. A new RT between FLOOR and ROOM appears, with maximum cardinality ONE at the side of FLOOR, and MANY at the side of ROOM. This is a general property of RTs in PowerDesigner: add them by dragging FROM the parent ET TO the child ET, and the maximum cardinalities are as they should be (but they are easily changed otherwise). Right-click to deactivate the Toolbox.
- Double-click the new RT; the window 'Relationship Properties' appears.
- Tab 'General': enter the name: FLOOR_of_ROOM.
- Tab 'Cardinalities':
    - Section 'Cardinalities': this is for the **maximum** cardinalities.
        - Radio button 'One-Many' should be chosen (if not, do so now).
        - Dominant Role: leave blank (it is only applicable to 'One-One' RTs).

  - Section 'FLOOR to ROOM': this is for the **minimum** cardinality at the side of ROOM, and the role at the side of FLOOR (don't blame us, we didn't design this!).
    - In the field 'Role name': type 'of', and check in the small diagram at the top of the window that this text indeed appears at the side of FLOOR.
    - The minimum cardinality at the side of ROOM should be ONE: at least one room on each floor. To accomplish this: mark the checkbox 'Mandatory' and check in the small diagram at the top of the window if the change is indeed made and the cardinality is correct.
  - Section 'ROOM to FLOOR': this is for the **minimum** cardinality at the side of FLOOR, and the role at the side of ROOM (the other way around from the previous section).
    - Field 'Role name': leave empty here (**always enter at least one role for each RT**, and we did enter one at the other side).
    - ROOM needs the <pi> of FLOOR as a part of its own identification: ROOM is a **weak ET**. To add a dependency triangle: mark the checkbox 'Dependent', and check in the small diagram at the top of the window that the triangle has indeed appeared.
    - The minimum cardinality at the side of FLOOR should be ONE: since the floor is a part of the identifier of a room, exactly one floor must be known for each room. Because the checkbox 'Dependent' is now marked, PowerDesigner has already done so.
- Click 'OK' to save the RT and return to the diagram window. Save the diagram.

Experiment a little with the layout of the RT, its name, and the role. If you click on the RT, two small black square handles appear that can be dragged to change the place the RT is attached to the ET. If you Ctrl-click the RT, an extra handle appears at that spot so you can make bends in the RT. The name and the role can be dragged to different places. You'll find there are restrictions on how far you can drag them. Though limited, this offers possibilities to improve the layout of your diagram. Consult the help files for further information about this (and don't get too frustrated if PD spoils it all again when you change something somewhere else). In general, do layouting only as a last step.

Finally, we'll add the last RT in the same way.

- Place EQUIPMENT_TYPE to the right of FLOOR, at the same height.
- Toolbox: Relationship. Drag from ROOM diagonally up to EQUIPMENT_TYPE. The RT should have a horizontal line, a bend and a vertical line (if not, Ctrl-click it and drag the handles so you'll have a layout approximately like in figure 2.1).
- Right-click to deactivate the Toolbox, and double-click the new RT.
- Name: Room_equipage.
- Maximum cardinalities: Many-to-Many.
- Minimum cardinalities: ZERO on both sides.
- Roles: add 'is equipped with' and 'is installed in' in the correct fields.

Save the diagram.

2.2.6 Adding predicates and example populations

To practice adding the semantics and the instances to the type level diagram, we'll put in two predicates with example populations (though this model is so small and clear it isn't really needed).

First predicate:

- From the Toolbox, section 'Standard', click 'Note', and click twice in the diagram (two empty note symbols appear). Right-click to deactivate the Toolbox.
- Double-click one of the note symbols. The window 'Text' appears.
- At the bottom, activate radio button 'RTF'.
- In the main space, type (use hard returns to force new lines):

```
Room <Room_number>
on floor <Floor_number>
has <Number_of_seats> seats.
```

- Select all text, and choose a convenient font (click the 'A' icon, choose 'Arial', 'regular', and '7' (this is what I use for the diagrams)).
- On new lines, enter the values (using **Ctrl-Tab** to separate the values horizontally):

```
27    27    3a
1     2     2
35    12    30
```

- Adjust the size of the note symbol so it fits tightly around the text, and place it in a convenient spot close to ET ROOM.
- From the Toolbox, section 'Standard', click 'Link/Traceability Link'.
- Move the cursor to the just finished note symbol, click-and-hold-down, and drag to ET ROOM. A dotted line appears that links the note and the ET, even if you drag them around the canvas (try it out).
- Save the diagram.

Second predicate:

- In the same way, edit the other note symbol.
- Predicate:

```
Room <Room_number>
on floor <Floor_number>
is equipped with
a(n) <Equipment_type_name>.
```

- Population:

```
27        27            3a
1         1             2
Beamer    Smartboard    Beamer
```

- This predicate + population belong to a RT, not a <pi> + Att pair. The Link will however not connect to a RT, so we must use a line:
  - From the Toolbox, section 'Free Symbols', click on 'Line'. Click-and-hold-down somewhere in a free spot on the diagram, drag a distance, and release the mouse button. A line has appeared.
  - Right-click to deactivate the Toolbox, then right-click the line, and from the pop-up menu choose 'Format…'. In the 'Symbol Format' window, tab 'Line Style', field 'Style', choose the small dotted line (this looks the same as the links on the diagram). Click 'OK'.
  - Drag the line handles to the RT (aim carefully) and the ET (the line will be visible on the ET, but we'll fix that in the next step).
  - Right-click the line, and from the pop-up menu choose 'Order' → 'Send to Back'.

Done! Optimize the layout and save the diagram. It should be identical to figure 2.1.

3 Deriving a Physical Data Model (PDM)

3.1 The terms used in PowerDesigner

Usually the following terms are used by data modelers:

- Conceptual data model: models the structure of data without referring to any way of implementation, not even whether the database will be relational, object oriented, or otherwise. In practice however, ERM is quite close to relational.
- Logical data model: models the structure of a database in a technique-specific way (usually according to the Relational Model: tables, columns, references, etc.), but independent of the implementation platform (SQL Server, Oracle, …) and without considering technical issues like performance.
- Physical data model: a database structure that does consider a specific platform (using its specific features) and does consider technical optimization (indexing, dummy keys, etc).

PowerDesigner however deviates from this in a somewhat confusing way (see the appendix for further explanations). It uses the terms:

- CDM (Conceptual Data Model): this does indeed correspond with a conceptual data model as meant above. So far so good.
- LDM (Logical Data Model): does not correspond with a logical data model as meant above. It is a strange mix of ERM and Relational, incorporating a weakness of the Relational model (foreign key columns) into ERM, while still keeping the RTs. We recommend to avoid this.
- PDM (Physical Data Model): this corresponds with a logical data model as meant above.
- DDL script: this corresponds with the automatically generated part of a physical data model as meant above.)

So we will derive a PDM, but not an LDM, from the CDM we made in chapter 2 of this tutorial.

3.2 Generating a PDM from a CDM

3.2.1 Setting the generation options

- Open the CDM we made in chapter 2. From the main menu choose 'Tools' → 'Generate Physical Data Model'. A window 'PDM Generation Options' appears.
- Tab 'General':
  - Radio button 'Generate new Physical Model' should be activated
  - Field 'DBMS' offers a large set of platforms. Choose one (for a short generated standard SQL DDL script choose 'ANSI Level 2', for SQL Server choose the latest version supported by PowerDesigner (SQL Server 2008).
  - Field 'Name': give a clear name, like 'Tutorial PDM'.
- Tab 'Detail'
  - 'Reference', 'Update rule': Cascade.
  - 'Reference', 'Delete rule': Restrict.
- Click 'OK'. A The generated PDM appears in a new tab (the CDM is still in another tab, and a result list has appeared in a third tab). The Output window opens and should contain the message: 'The model has been successfullly generated…'. Save the PDM file somewhere.

3.2.2 Display preferences for PDM

We need to set default display references for a PDM (as we did on section 2.2.2 for a CDM). **These settings are mandatory for all PDMs you want to communicate with your teachers.** In the PDM window, right-click in an empty space and choose 'Display Preferences…' → 'Table'. Set the options on tab 'Content' as shown in figure 3.1, and click the 'Set As Default' button. On the tab 'Format', click the 'Modify' button, and in the 'Symbol Format' window, on the 'Size', mark checkbox 'Auto adjust to text'. Click 'OK', then 'Set as Default'.

Repeat the step above for: 'Display Preferences…' → 'Reference'. Set the options on tab 'Content' as shown in figure 3.2 and click the 'Set As Default' button. These settings are necessary because

- Without the join condition it is not always possible to tell from where to where a reference points, not even if you know the <pk>s and <fk>s.
- Without the cardinalities at the foot of the references, it is impossible to tell whether it is an ordinary foreign key reference (0..*) or a mandatory-child reference (1..*).

Make sure you always use these settings.

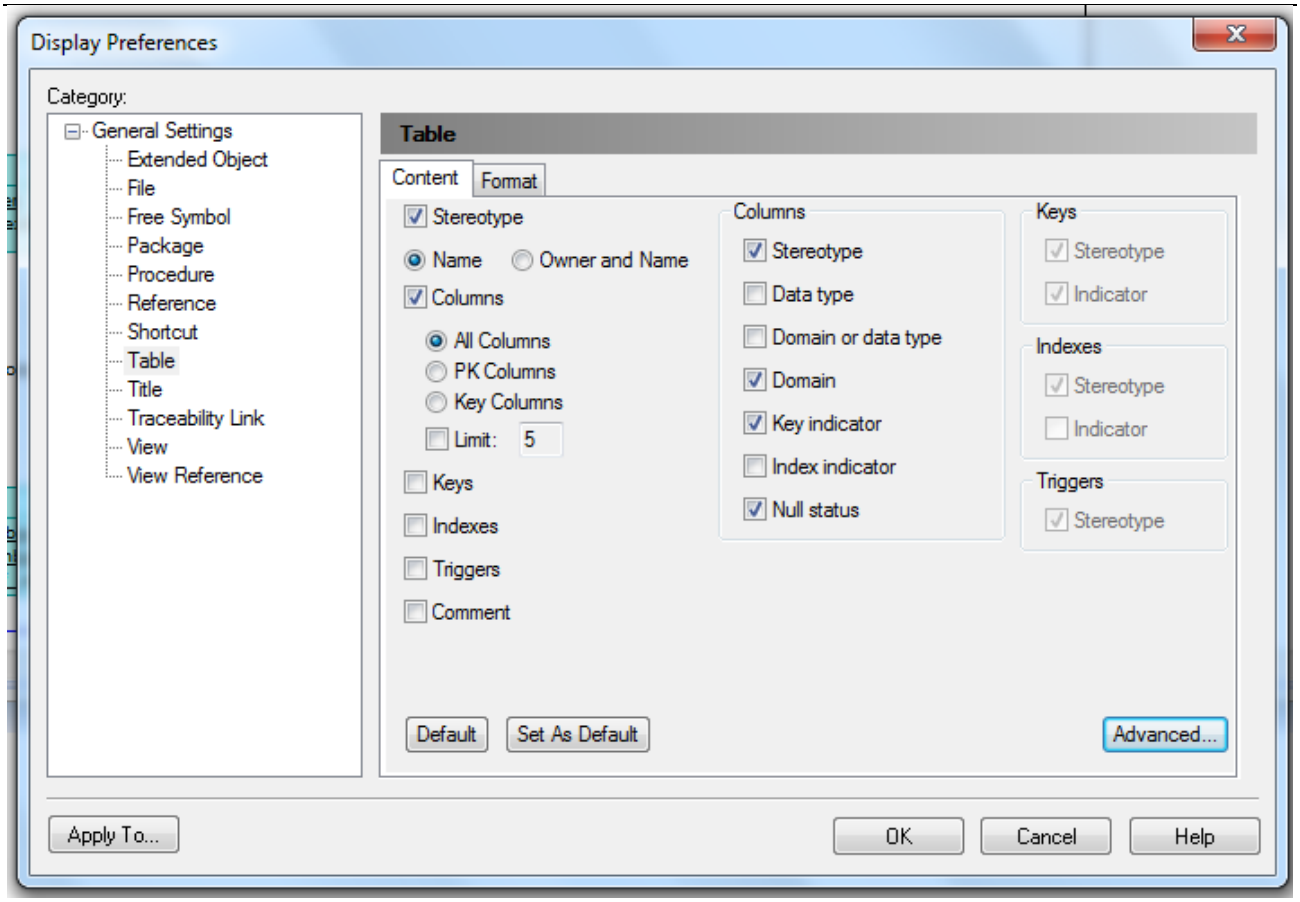


Figure 3.1 Mandatory display settings for table

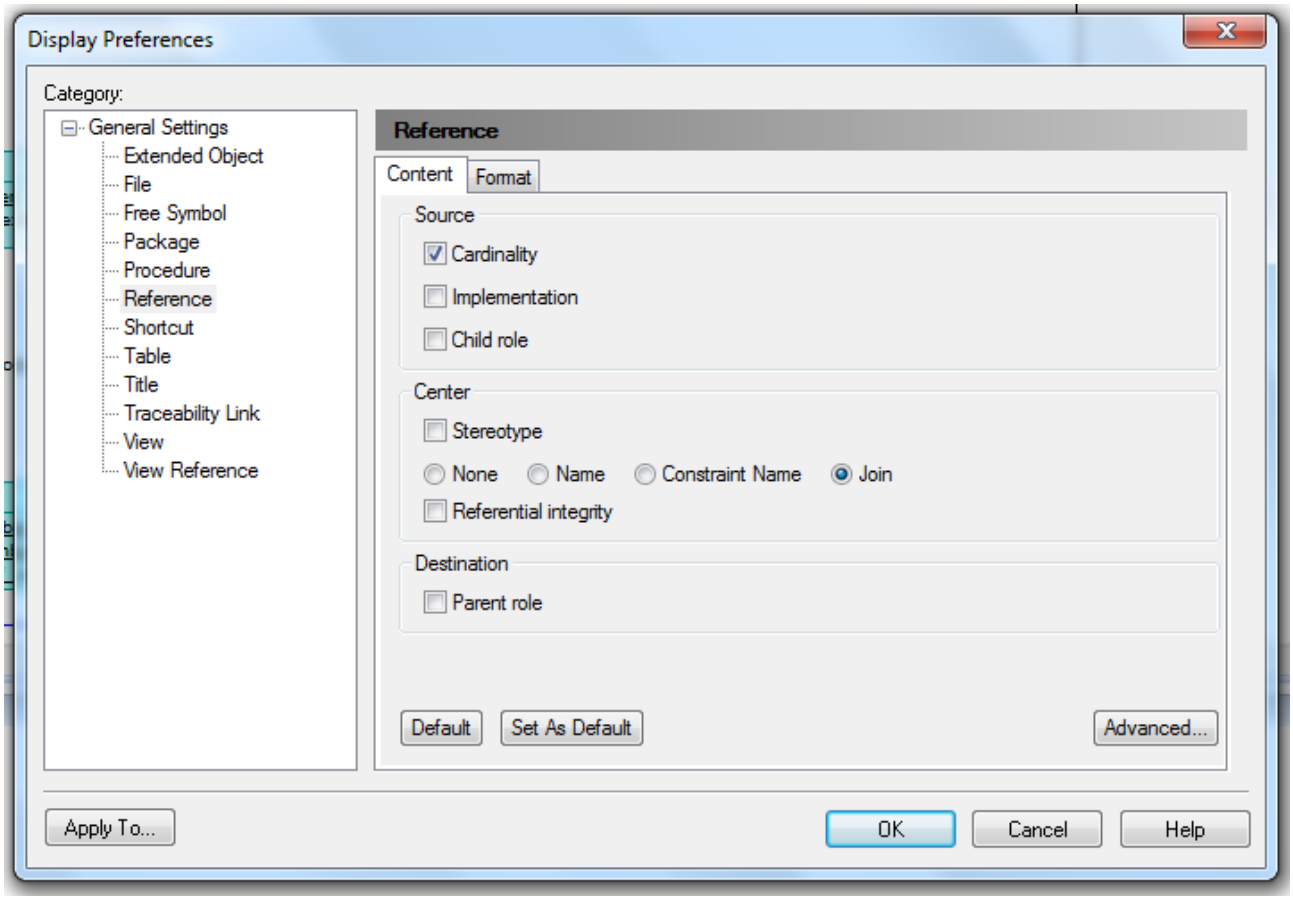


Figure 3.2 Mandatory display settings for reference

3.2.3 The generated PDM

If you do a little layouting, and change a few awkward generated names (replace the name of the old RT R_RoomEquipage with a table name like ROOM_EQUIPAGE for example) the PDM should look like the one in figure 3.3.

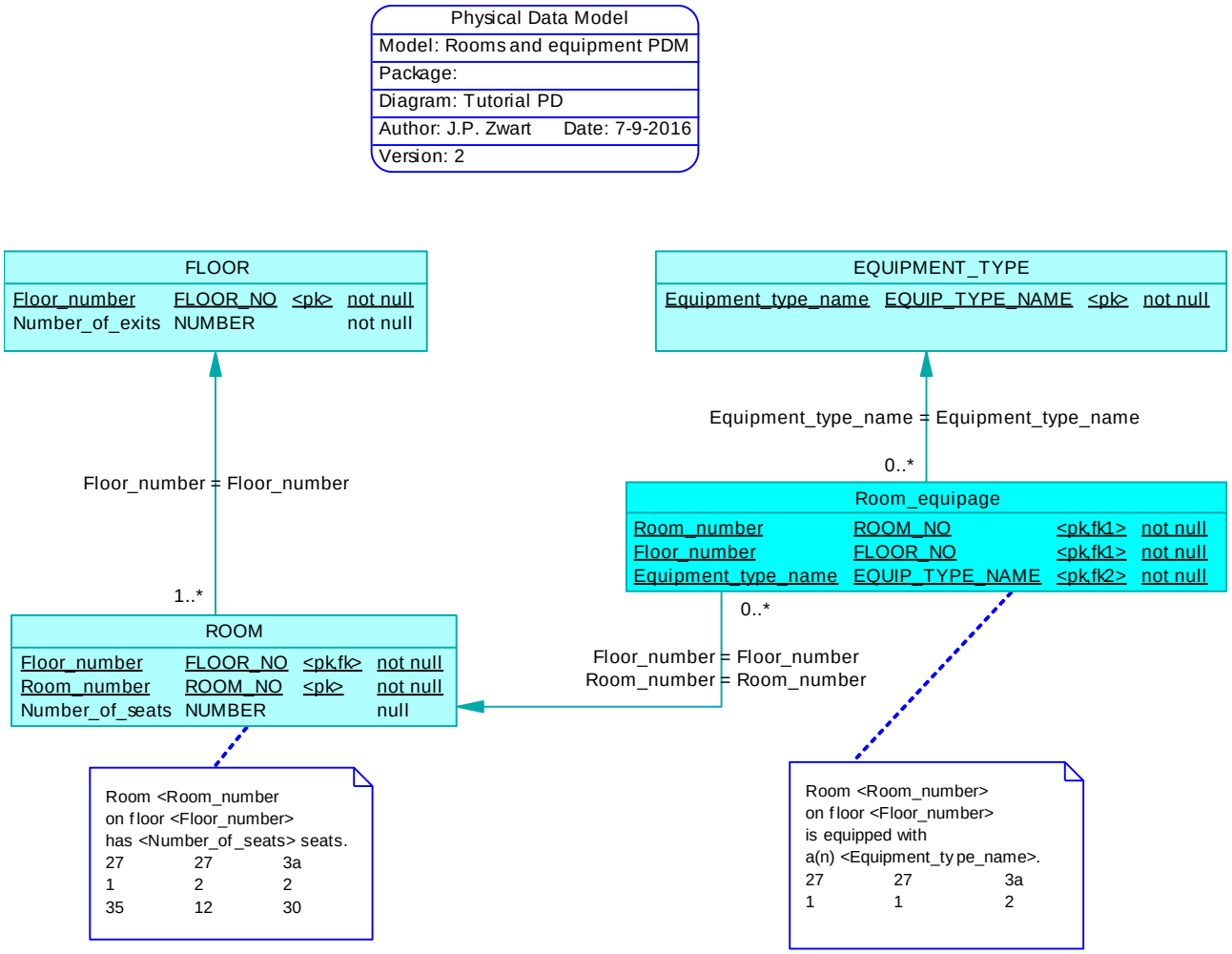


Figure 3.3 Generated PDM

Note the following points:

- There is an extra table Room_equipage, that comes from the many-to-many RT Room_equipage in the CDM.
- <pk> stands for primary key, <fk> for foreign key (with a sequence number where needed).
- A PDM looks very much like a CDM, but it is important to keep the distinction well in mind. Never mix the levels up: an ET is not a table, a RT is not a reference, etc.
- In general, a little manual tweaking of the generated PDM is required (better names for tables or columns, adjustment of the predicates to these new names, adding or changing data types, etc.).
- The predicates and example population can be carried over with hardly any change. This is another advantage of fact-oriented modeling: fact types are independent of the modeling technique used.
- The fact type about the room equipage is now no longer modeled as a RT, but as an 'ordinary' table, so the note symbol can be connected to the table using a Link instead of a loose line. It is a weakness of ERM that fact types are modeled either as <pi>+Att pairs or as RTs, instead of as a single construct (in this respect the Relational Model is actually better).

3.3 Generating a DDL script

Finally we'll generate DDL scripts from the PDM.

- From the main menu choose 'Database' → 'Generate Database'. A window 'Database Generation' appears.
- Tab 'General':
    - Note that 'DBMS' cannot be changed here: if you want another platform, go back to the CDM and generate the PDM for that platform first.
    - Field 'Directory': enter a convenient directory name to save the script file in.
    - Field 'File Name': enter a clear name for the script file.
- Click 'OK'.

Study the generated SQL script closely to find out which model parts are or are not taken over into the script file, depending on the chosen platform. One usual suspect is:

- Mandatory child references: often only the 'ordinary' FK-reference is taken along, and the mandatory child reference pointing in the opposite direction is 'lost' and must be programmed by hand.

Import the script into the platform of your choice, and admire your fully automatically generated empty tables.

Having worked through this tutorial, you should be able to start using PowerDesigner professionally. You're welcome to explore the many features not discussed here (for example: main menu 'Database' → 'Generate Test Data'). Have fun!

Optional extra exercise

Make an alternative version of the CDM in figure 2.1: model the room equipage not as a RT Room_equipage, but as an empty ET ROOM_EQUIPAGE, which has two dependent RTs: one to ROOM and another to EQUIPMENT_TYPE (with what cardinalities?). Generate the PDM from this second CDM and compare it closely with the previous PDM. It should be exactly the same (except for the name of the fourth table, which is spelled differently with capital letters). You might even add an attribute now to ROOM_EQUIPAGE (like Number, to contain the number of that equipment type installed in that room (2 whiteboards for instance)). Note: this extension is not possible in the first CDM, which would have to be changed to the second to accommodate the new attribute (which can be done automatically: right-click the many-to-many RT and choose 'Change to Entity').

4 Appendix:
Conceptual, Logical and Physical Level, relation to CDM, LDM, PDM.

Level	Characteristics	PowerDesigner (PD)

Conceptual Level	• Conceptual data model
	• Part of Functional Design (see theme 5)
	• Describes only structure and semantics of data
	  ○ Fact Oriented modeling: FCO-IM, ORM,….
	  ○ **ERM**
	  ○ UML class diagrams
	• Can be transformed automatically to different database models on logical level (relational, data warehouse star schema, object oriented DB, functional program….)
	• Meta data also free of redundancy
	• Independent of database models
		Conceptual Data model (CDM)
		
		PD generates an LDM (DON'T!) or PDM (DO) based on CDM.

???	Strange mix of what the rest of the world calls conceptual and logical (worst of both worlds) • Does have relationship types and cardinalities • But metadata not free of redundancy because of <fi>	**Logical Data model (LDM)** **Strongly advised against!**
Logical Level	• Logical Data Model • Part of Technical Design (see theme 5) • A choice for one database model (e.g. relational) • Independent of specific database management system (DBMS) (SQL Server, Oracle…) • Usually according to recognized standard (e.g. SQL '92) and with a limited number of data types • Metadata not free of redundancy	**Physical Data model (PDM)** PD generates scripts for various DBMS

Physical Level	• Physical Data model
	• Part of Technical Design (see theme 5)
	• Choice for one DBMS, e.g. SQL Server
	• Uses specific possibilities or limitations of this DBMS
	  ○ More practical data types
	  ○ DBMS specific SQL commands
	  ○ Which constraints to program / not to program yourself?
		---- 	