Datenbanken II A: DB-Entwurf

Chapter 1: ER-Diagrams I: Entities, Attributes, Relationships

Prof. Dr. Stefan Brass

Martin-Luther-Universität Halle-Wittenberg

Wintersemester 2024/25

http://www.informatik.uni-halle.de/~brass/dd24/

Objectives

After completing this chapter, you should be able to:

- enumerate the ER-constructs supported by Oracle SQL Developer Data Modeler.
- draw ER-diagrams in the graphical syntax of Oracle SQL Developer Data Modeler ("Barker Notation").
- You should also be able to read such diagrams.
- explain the difference between the global DB schema and the views contained in single diagrams.

Contents

Entity-Relationship-Model (1)

- The Entity-Relationship-Model is called a "semantic data model", because it more closely resembles the real world than e.g. the relational model.
  - In the ER-model, persons are modelled. In the relational model, only their names/numbers.
  - In the ER-model, there is a distinction between entities and relationships. In the relational model, both are represented by relations.

    This expressiveness is not needed to satisfy the information requirements of the applications. But it makes the correspondence between the schema and the real world clearer (like a comment).

Entity-Relationship-Model (2)

- Proposed by Peter Pin-Shan Chen (1976).
- There is a useful graphical notation which helps to establish a better overview; to "see" the structure of the data.

  The graphical notation also helps to communicate with the future users.
- Many variations and extensions of the ER-Model have been proposed.

  We use here the notation proposed by Richard Barker (his book appeared 1989/90). The easy extensibility of the ER-model (because it is not bound to a DBMS) is also one of the strengths of the model.

Entity-Relationship-Model (3)

- There are specialized graphical editors and other design tools.

  E.g. Oracle Designer and Oracle SQL Developer Data Modeler are CASE tools for DB-applications, and one component of them is a graphical editor for ER-diagrams. (CASE: Computer-Aided Software Engineering.)
  Alternatives: Sybase PowerDesigner, CA ERwin, ...

- The ER-model is a standard tool for conceptual DB design. However, recently, object-oriented methods are also used.

  In Software Engineering, UML (the Unified Modeling Language) has become a standard. One can use UML class diagrams for DB design. They are based on the ER-model, but they have no keys (only as an extension). Keys a a central database concept.

Basic ER-Model Concepts (1)

Entities:

- Objects in the miniworld about which information has to be stored. E.g. persons, books, courses.

  It does not matter whether entities have a physical existence (and can be touched) or only a conceptual existence.

- At each instant, the miniworld that has to be modelled can contain only a finite number of entities.

  E.g. "all numbers" (infinitely many) cannot be entities.

- It must be possible to distinguish entities from each other, i.e. they must have some identity.

  So ants in a heap do not qualify as entities.

Basic ER-Model Concepts (2)

Data Type Elements:

- Values from some possibly infinite set, which can be stored and printed.

  E.g. strings, numbers, dates, lengths, pictures. A person cannot be stored (entity), but his/her name can be stored (data type element).

- Most current DBMS have some predefined set of data types which they support.

- It is possible to use non-standard types in ER-schemas (complicates the later logical design).

  The ER-Design should be independent of a specific DBMS.

Basic ER-Model Concepts (3)

Attribute:

- A property or characteristic of an entity.

  Depending on the specific version of the ER-model, also relationships can have attributes (see below).

- E.g. the title of this course is “Database Design”.
- The value of an attribute is an element of a data type like string, integer, date: It has a printable representation.
- Attributes can be optional, i.e. permit null values.

  Their semantics is then a partial function from entities to data type values.

Basic ER-Model Concepts (4)

Relationship:

- Relation between pairs of entities ("binary relationship").

  Some ER-notations allow relationships involving more than two entities.

  My experience shows that this often leads to errors.

- E.g. I (a person) teach "Database Design" (a course).
- The word "Relationship" is also used as an abbreviation for "Relationship-Type" (see below).

  It should be clear from the context what is meant.

Basic ER-Model Concepts (5)

Entity-Type:

- Set of similar entities (with respect to the information which has to be stored about them), i.e. entities which have the same attributes.
- E.g. all faculty members of this university.

Relationship-Type:

- Set of similar relationships.
- E.g. “X teaches course Y”.

ER-Model Variants

- Variants of the ER-Model differ in:
  - The selection of ER modeling constructs.

    See next page for the ER constructs supported in Oracle Designer.
  - The notation used for these constructs.

    E.g. softboxes are used for entities, and the "crowsfoot"/"chicken feet" notation for cardinalities.
  - The possibility to model also behaviour: Methods/Operations supported by the entities.

    This is typical for object-oriented approaches. Oracle Designer has other tools for modeling this (Process Diagrams, Dataflow Diagrams).

Supported ER-Constructs (1)

Oracle Designer supports the following ER-constructs:

- Binary relationships including recursive ones.
- The most important cardinalities.

  0 and 1 as minimum, 1 and $*$ as maximum. The cardinality $(1, *)$ on both sides of the relationship is excluded (insertion would be too difficult). In the repository, arbitrary min-max cardinalities can be specified, but they are not shown in the diagrams.
- Optional attributes (null values).
- Domains for attributes.

  Domains are similar to user-defined data types, however, they are only names for the standard datatypes of the DBMS. See below.

Supported ER-Constructs (2)

- Constraints on attribute values.
- Keys of entity types.
- Weak entities.

  I.e. using a relationship as part of the key of entities.
- Disjoint and total specialization.
- Mutually exclusive relationships.
- Non-transferable relationships.
- Various additional information about entities.

  E.g. synonyms, expected sizes, comments, further documentation.

Supported ER-Constructs (3)

Oracle Designer does not support these ER-constructs:

- Ternary etc. relationships.

  As explained below, one can always replace relationships by "association entities" and binary relationships.

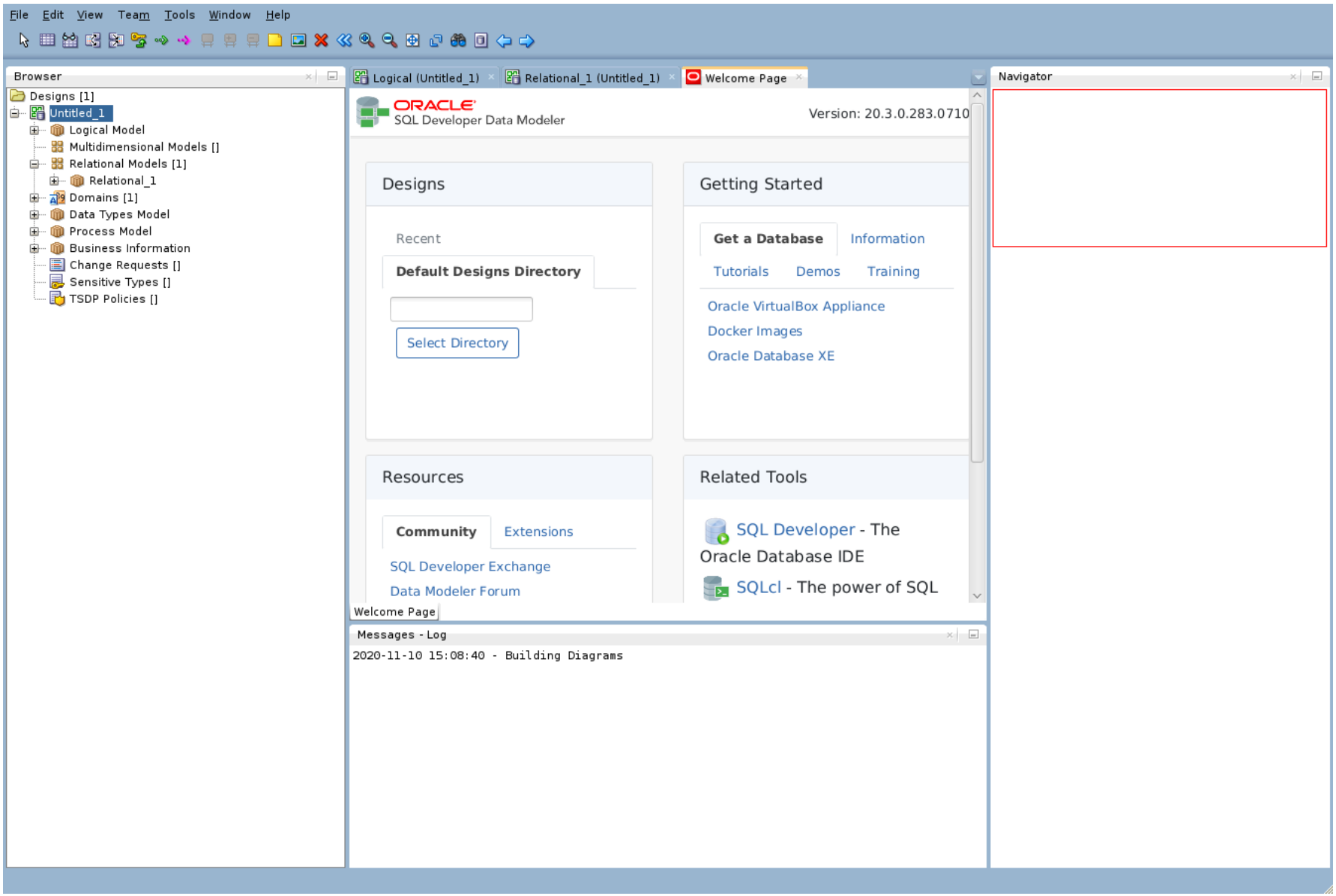
- Relationship attributes.

  Also in this case, one must turn the relationship into an (association) entity with two binary relationships without attributes.

- Multivalued/structured attributes.

  Multivalued attributes can be handled with weak entities. For structured attributes, the components can be declared as attributes.

- Non-disjoint specialization.


File Edit View Team Tools Window Help
Browser
Designs [1]
Untitled_1
Logical Model
Multidimensional Models []
Relational Models [1]
Relational_1
Domains [1]
Data Types Model
Process Model
Business Information
Change Requests []
Sensitive Types []
TSDP Policies []
Logical (Untitled_1)
Relational_1 (Untitled_1)
Welcome Page
ORACLE
SQL Developer Data Modeler
Version: 20.3.0.283.0710
Designs
Recent
Default Designs Directory
Select Directory
Getting Started
Get a Database
Information
Tutorials
Demos
Training
Oracle VirtualBox Appliance
Docker Images
Oracle Database XE
Resources
Community
Extensions
SQL Developer Exchange
Data Modeler Forum
Related Tools
SQL Developer - The Oracle Database IDE
SQLcl - The power of SQL
Welcome Page
Messages - Log
2020-11-10 15:08:40 - Building Diagrams
Navigator

Oracle SQL Developer Data Modeler

- When one starts the Oracle SQL Developer Data Modeler, there is an "Untitled" Project already.
- One uses "save as" to give the project a name.

  This is a bit uncommon. From other development tools, one would expect that there is a menu item "new project".
  If there is an error in saving, one probably cannot write `/opt/datamodeler/datamodeler/types/defaultdomains.xml`.
  This file does not exist at the beginning. One "quick&dirty" solution is to make the directory writable by the current user.

- The Entity-Relationship-Schema is called the "Logical Model" in the Data Modeler.
- One creates an entity type by selecting the "entity with gear" symbol in the palette below the menu and then clicking into the logical diagram.

Contents

Entities and Attributes (1)

- The Barker notation uses a "softbox" (rectangle with rounded corners) for entity types.
- Attribute names are written into this box:

Oracle Designer (old)

Oracle Data Modeler (new)

- Attributes which are mandatory (not null) are marked with *, optional attributes with ○.

Entities and Attributes (2)

- Primary key attributes are marked with #.

  Attributes marked with # together constitute the primary key.

  Primary key attributes are automatically mandatory. Therefore, in Oracle Designer no additional "∗" was shown. This was changed in the Data Modeler. The Data Modeler displays an "∪" in the "key column" for all attributes that participate in non-primary keys ("unique"). These might allow null values.

- One can customize what is displayed, e.g. it is possible not to show attributes.

  This is useful e.g. in order to get an overview of a large schema.

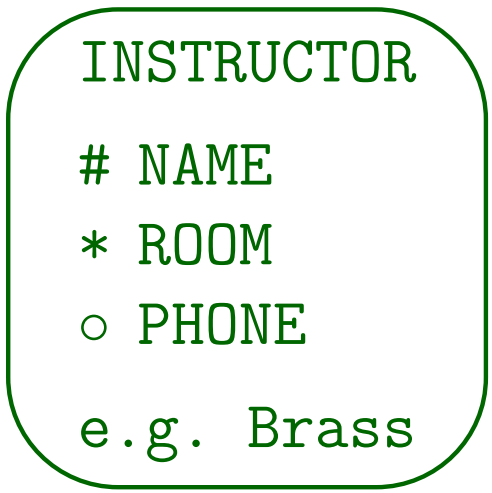
- More information is stored about entities which is not graphically displayed.

  One can open a dialog box with additional entity properties.

- Entity type names must be unique in the schema.

Entities and Attributes (3)

- In earlier versions, one could mention an example entity at the bottom of the entity type box:

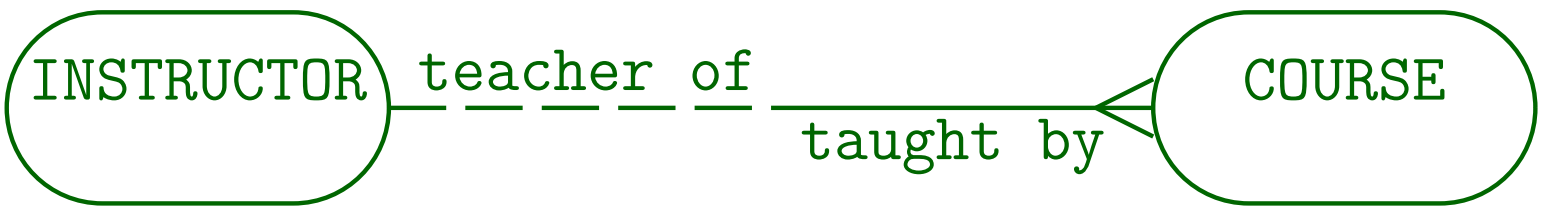


- This is a nice idea, but it is no longer supported.

  Of course, one can and should mention an example in the description of the entity type (not shown in the diagram).

Relationships (1)

- Relationships are marked by lines between the entity boxes (no diamond).
- The form of the line (dashed or solid) and the line end (simple or crowsfoot) describe the cardinalities:



- This is very illustrative: One instructor can teach many courses, but each course is only taught by one instructor (see below).

Relationships (2)

- Relationships have two names (seen from each of the ends): The "from name" and the "to name".

  In the Data Modeler: "name on source", "name on target".

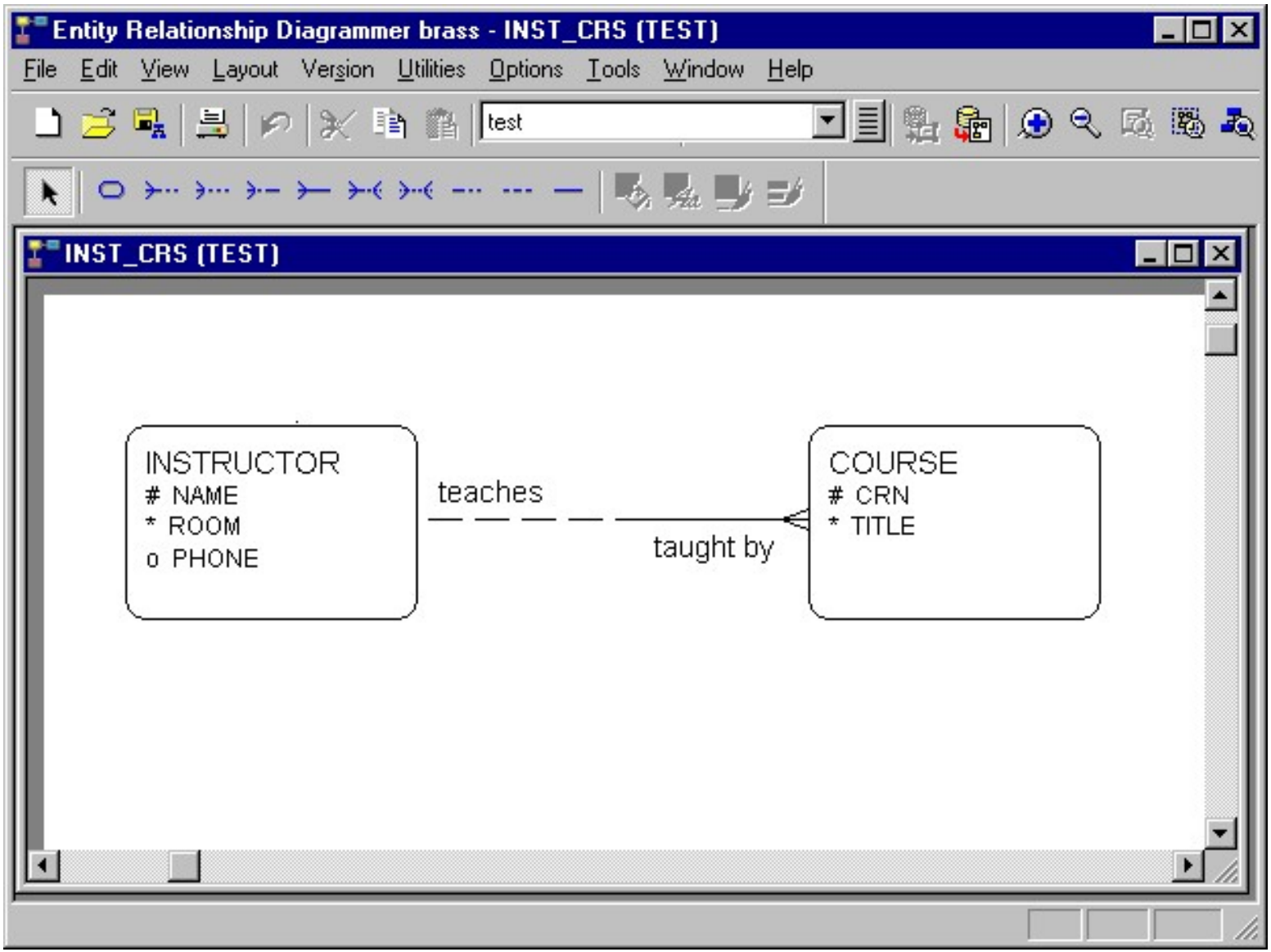
- Just as a relationship between people, relationships are "both way": This is reflected in the two names.

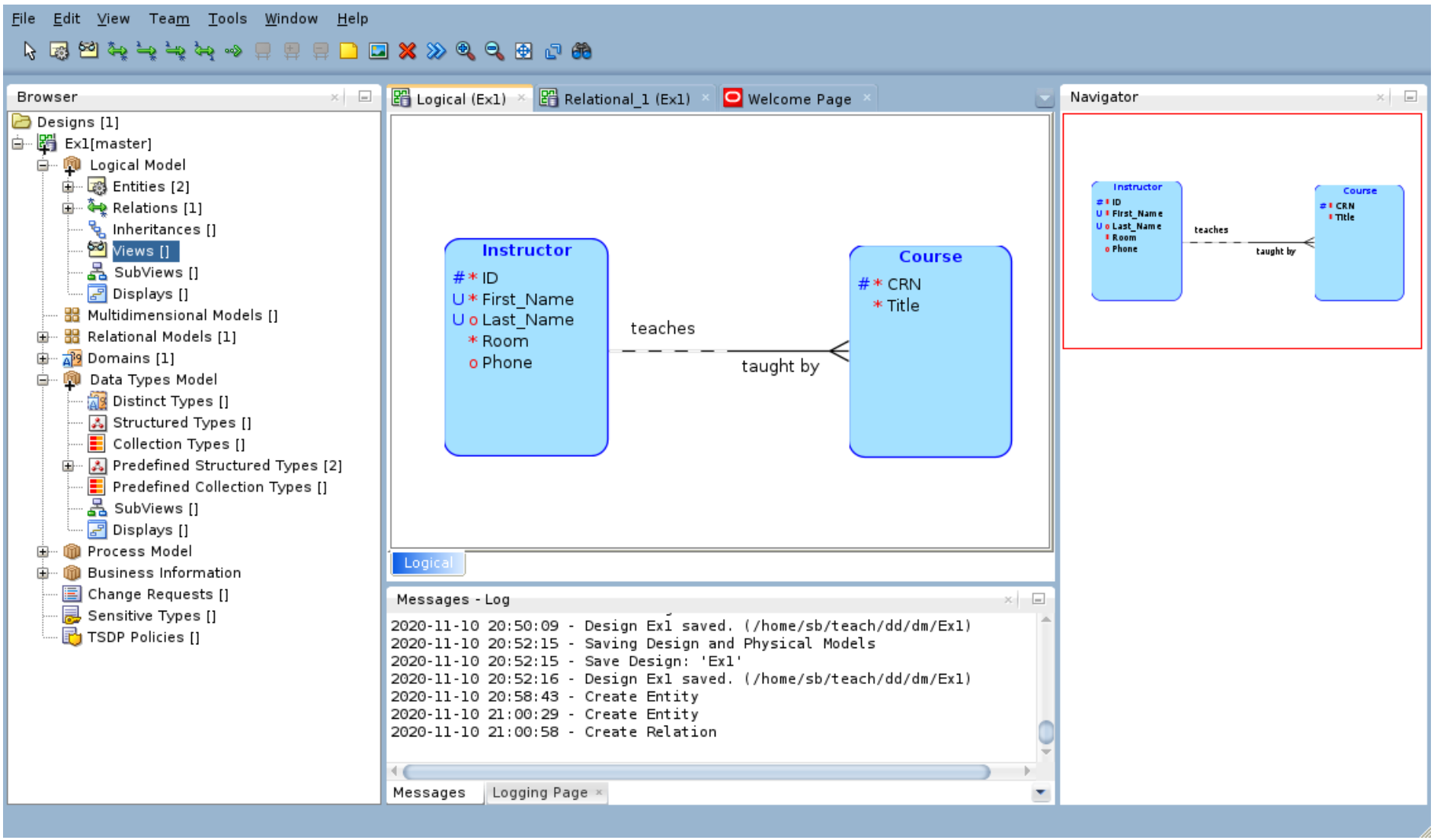
- The same names can be used for different relationships (they do not have to be globally unique).

  Only between the same two entity types there cannot be two relationships that agree in both, to and from name.

- In the Data Modeler, there is also a "relationship name".

  By default, no names are displayed for relationships. One can show the "name on source" and "name on target" by right clicking on the display background and select "Show→Labels".

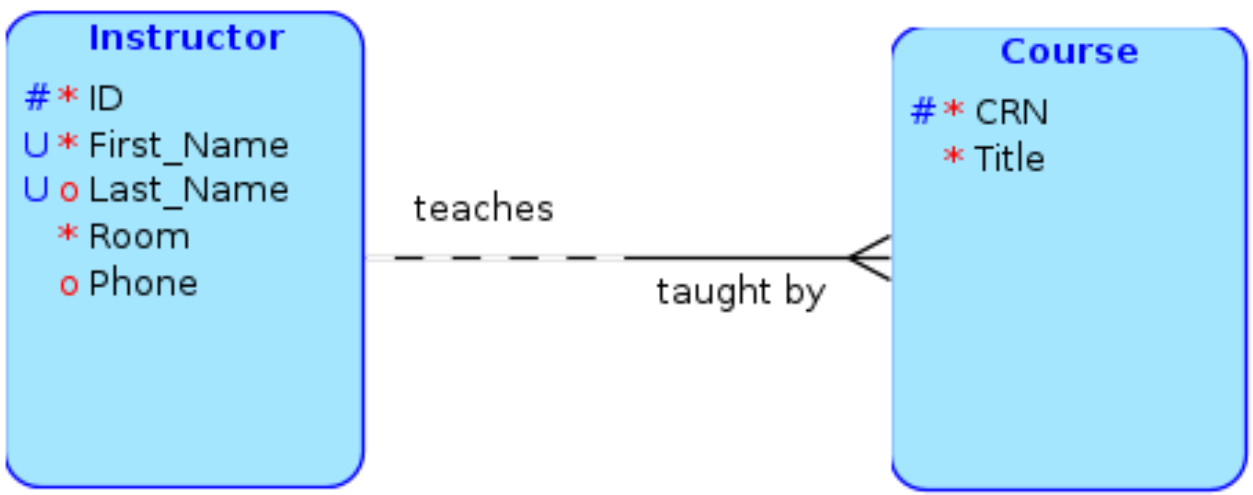

Entity Relationship Diagrammer brass - INST_CRS (TEST)
File Edit View Layout Version Utilities Options Tools Window Help
test
INST_CRS (TEST)
INSTRUCTOR
NAME
* ROOM
o PHONE
teaches
taught by
COURSE
CRN
* TITLE


File Edit View Team Tools Window Help
Browser
Designs [1]
Ex1[master]
Logical Model
Entities [2]
Relations [1]
Inheritances []
Views []
SubViews []
Displays []
Multidimensional Models []
Relational Models [1]
Domains [1]
Data Types Model
Distinct Types []
Structured Types []
Collection Types []
Predefined Structured Types [2]
Predefined Collection Types []
SubViews []
Displays []
Process Model
Business Information
Change Requests []
Sensitive Types []
TSDP Policies []
Logical (Ex1)
Relational_1 (Ex1)
Welcome Page
Instructor
* ID
U * First_Name
U o Last_Name
* Room
o Phone
teaches
taught by
Course
* CRN
* Title
Logical
Navigator
Messages - Log
2020-11-10 20:50:09 - Design Ex1 saved. (/home/sb/teach/dd/dm/Ex1)
2020-11-10 20:52:15 - Saving Design and Physical Models
2020-11-10 20:52:15 - Save Design: 'Ex1'
2020-11-10 20:52:16 - Design Ex1 saved. (/home/sb/teach/dd/dm/Ex1)
2020-11-10 20:58:43 - Create Entity
2020-11-10 21:00:29 - Create Entity
2020-11-10 21:00:58 - Create Relation
Messages
Logging Page

Export des Diagrams

- One can export the diagram as image in various formats (PNG, PDF, SVG oder HTML/SVG).

  Right click on diagram background, choose “Print Diagram”.



Notation for Cardinalities (1)

- A classic ER-notation is the following:

Here entity types are shown as rectangles, and relationships as diamonds. Attributes would normally be shown in "ovals" attached to the rectangles or diamonds. However, this takes a lot of space.

- The cardinality restrictions on the relationships are specified as intervals for the number of connected relationships for a single entity: "(min,max)-notation".
  - An instructor entity can be related to any number of course entities (between 0 and arbitrarily many).
  - A course entity must be related to exactly one instructor entity (minimally 1 and maximally 1).

Notation for Cardinalities (2)

- As maximum cardinalities, only `1` and `*` are common. The maximum cardinalities on both sides classify a relationship as
  - Many-to-many (N:M): "`*`" on both sides.
  - One-to-many (1:N): "`*`" and "`1`".

    "*" on one side, "1" on the other side.
  - One-to-one (1:1): "`1`" on both sides.
- The example is "one-to-many" from instructor to course, i.e. one instructor can teach many courses.

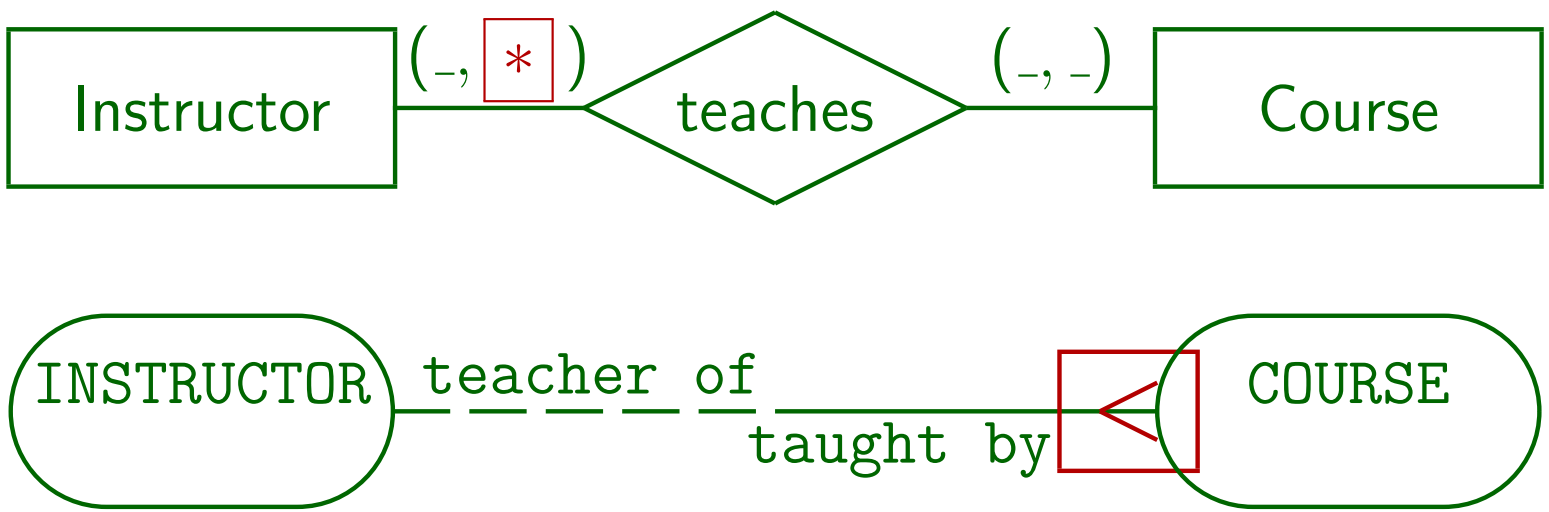
  The maximum cardinality "*" is written near the instructor (i.e. the "one" side): The (min,max)-cardinalities on the instructor side describe the outgoing edges from a single instructor (number of courses).

Notation for Cardinalities (3)

- As minimum cardinalities, only 0 and 1 are common:
  - The minimum cardinality "0" means optional (or partial) participation in the relationship:
    Not every instructor must teach a course.
  - The minimum cardinality "1" means mandatory (or total) participation in the relationship:
    Every course must be taught by an instructor.
- A relationship can be optional on both sides, optional on one side and mandatory on the other, or mandatory on both sides (difficult for insertions).

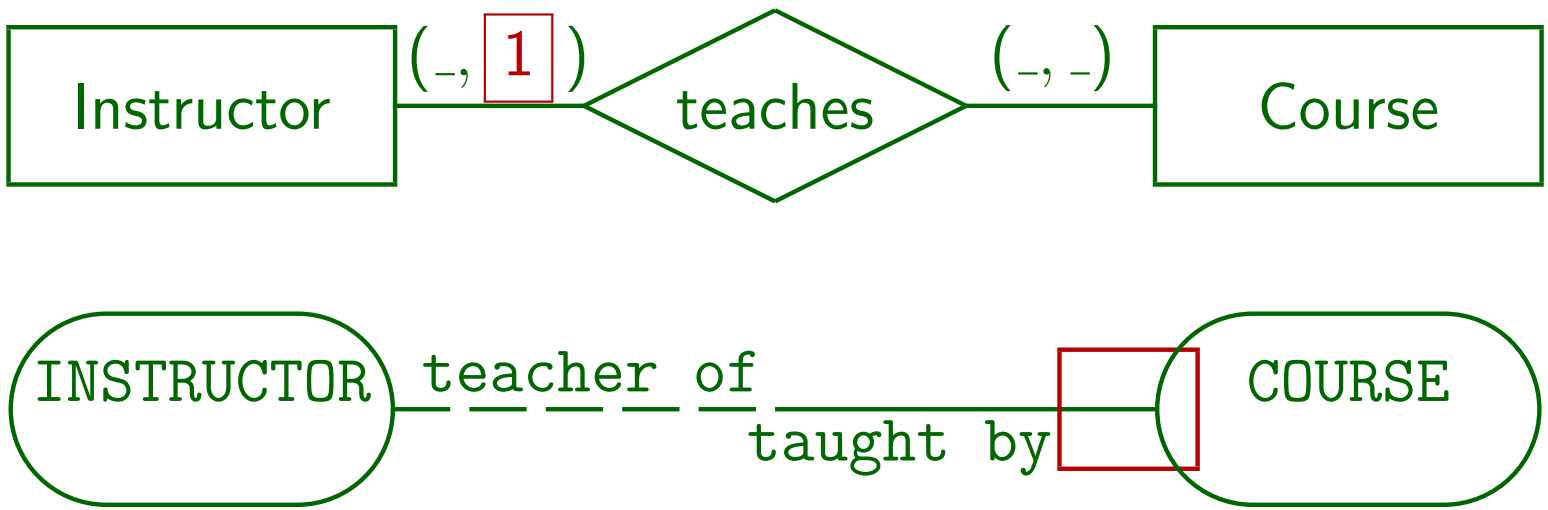
Notation for Cardinalities (4)

- The Barker notation can represent the common maximum cardinalities (1 and $*$).
- For the maximum cardinality "$*$" on the instructor side, a crowsfoot is drawn on the course side:



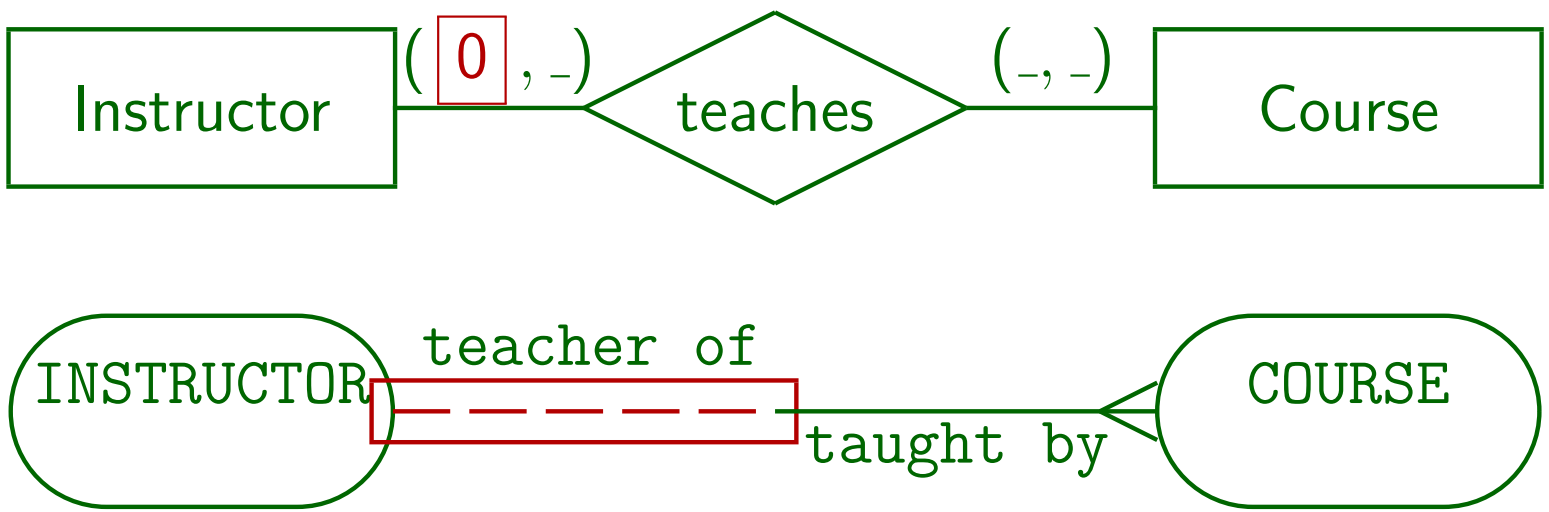
Notation for Cardinalities (5)

- If the maximum cardinality should be "1" on the instructor side (each instructor can teach only one course), no crowsfoot is drawn on the course side:



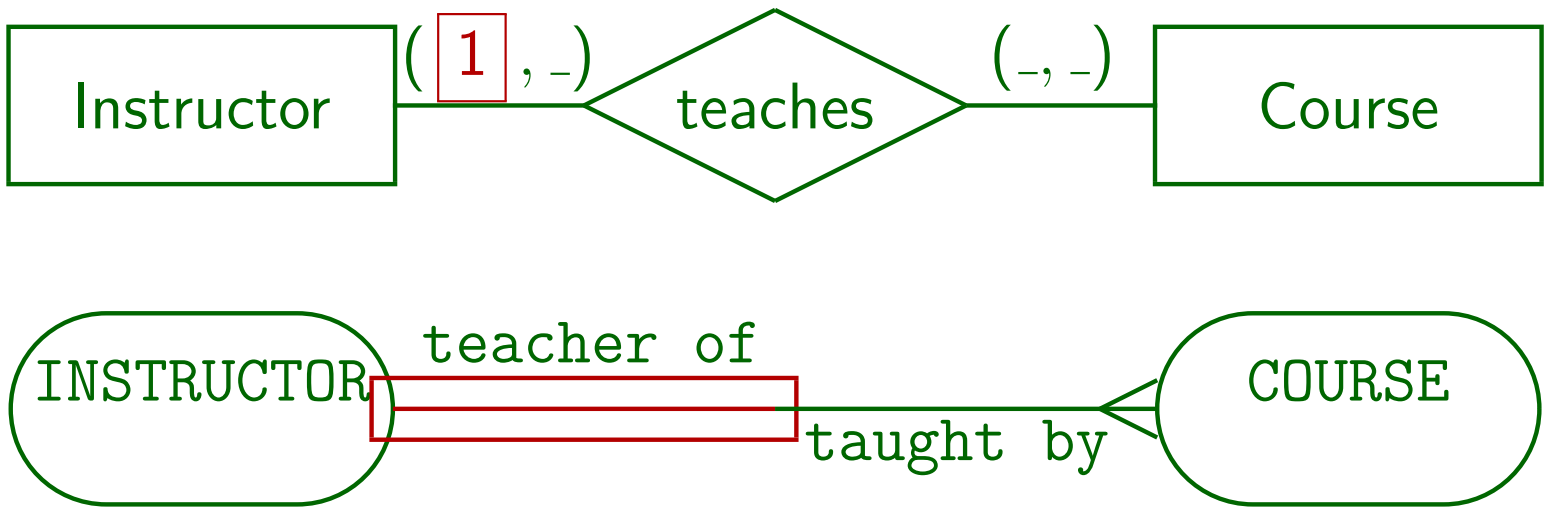
Notation for Cardinalities (6)

- The Barker notation can represent the common minimum cardinalities (0 and 1).
- For the minimum cardinality "0" on the instructor side, a dashed line is drawn on the instructor side:



Notation for Cardinalities (7)

- For the minimum cardinality "1" on the instructor side (each instructor must teach at least one course), a solid line is drawn on the instructor side:



Checking Cardinalities (1)

- If the names of the relationship ("teacher of", "taught by") are chosen rigorously, natural language sentences that explain the cardinalities can be automatically generated.
  - "Each (and every) INSTRUCTOR may be teacher of one or more COURSES."
  - "Each (and every) COURSE must be taught by one and only one INSTRUCTOR (ever)."

    Phrases in parentheses only emphasize, but don't change the meaning. They can be left out.

Checking Cardinalities (2)

- "May be" indicates optional participation, "must be" is used for mandatory participation.

  Oracle Designer knows the plural form of every entity type, as required for generating these sentences. Some design reports that the "Repository Reports" utility produces contain such sentences.

- Note that both sentences are needed to completely describe the relationship.

- However, it is sometimes difficult to choose relationship names that fit into this pattern.

  They must consist of a noun (role) and a preposition. For verbs like "teaches" a slightly different pattern would be needed.

Checking Cardinalities (3)

- Such sentences can be shown to domain experts (future users) who cannot read ER-diagrams.
- This is a way of validating the database schema (checking it for correctness).

  Keys and other constraints could be validated in the same way.

- Actually, one can even produce a question form and let the user check whether it is true:

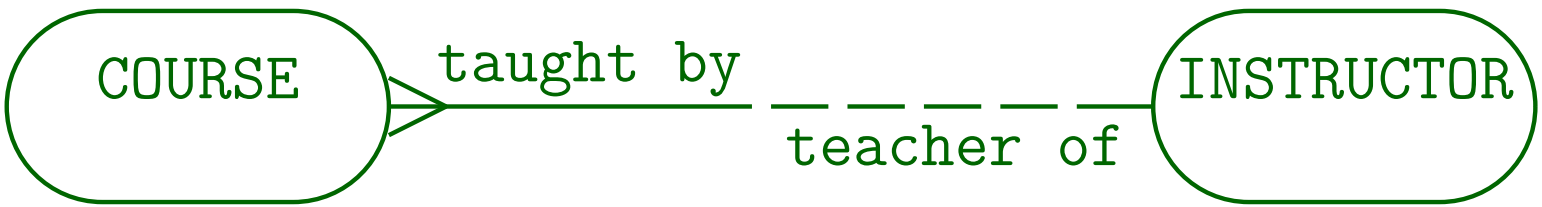
  Each and every course must be taught by one and only one instructor, is that true?

  ☐ yes ☐ no

Cardinalities (1)

- The toolbar has nine different relationship types.
  The first is “many to one (mandatory to optional)”:

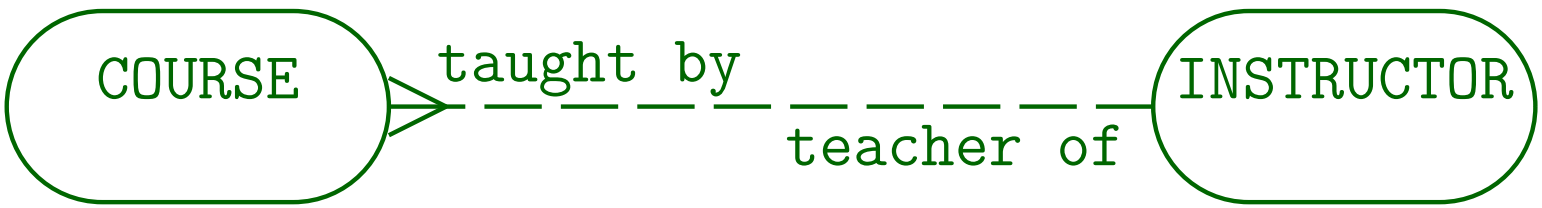
- In Barker Notation:



Cardinalities (2)

- The next is "many to one (optional to optional)":

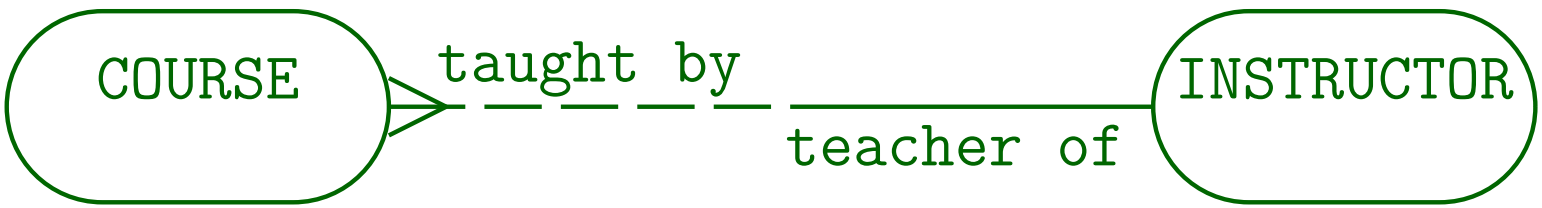
- In this case, a course has not necessarily a teacher assigned.
- In Barker notation:



Cardinalities (3)

- Many to one (optional to mandatory):

- In Barker Notation:



Here an instructor must teach at least one course, and can teach any number of courses. A course does not require an instructor, but can have at most one.

Cardinalities (4)

- Many to one (mandatory to mandatory):

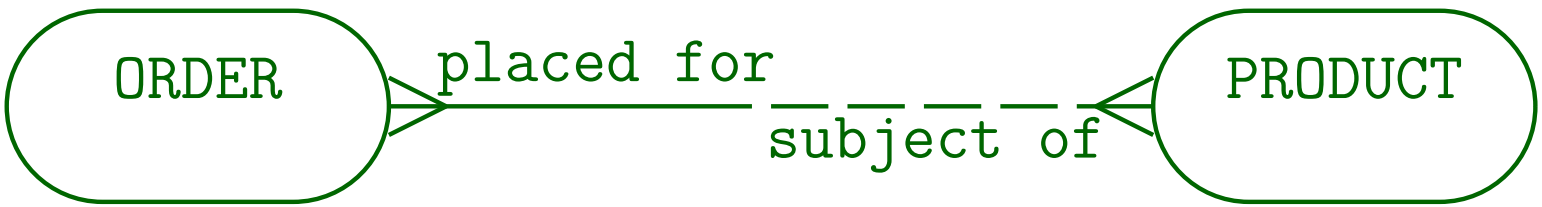
- In Barker notation:

Every invoice item belongs to exactly one invoice. An invoice can consist of several items, but must consist of at least one.

Cardinalities (5)

- Many to many (mandatory to optional):

- In Barker notation:

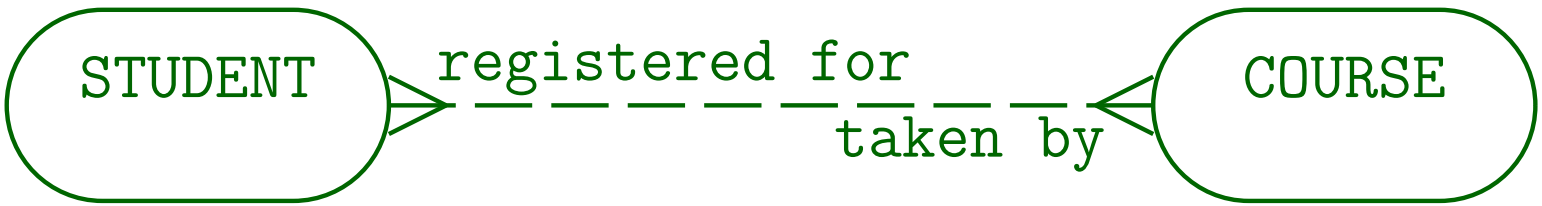


Every purchase order must be for at least one product, but can be for many products. One product can be ordered in many purchase oders. There can be new products that are not yet ordered.

Cardinalities (6)

- Many to many (optional to optional):

- In Barker notation:



- This is the most general relationship:

  A student can take any number of courses (including zero), a course can be taken by any number of students (again including zero).

Cardinalities (7)

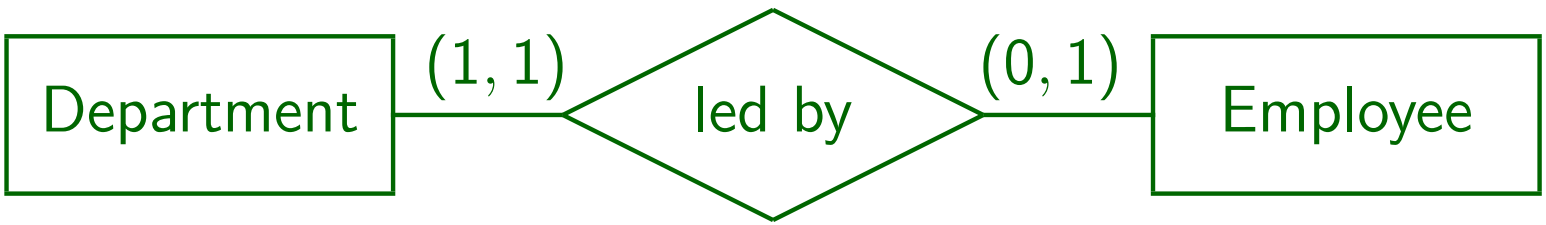
- Many to many (mandatory to mandatory):

- This is not supported in Oracle Designer.
- It would be very difficult to insert entities.

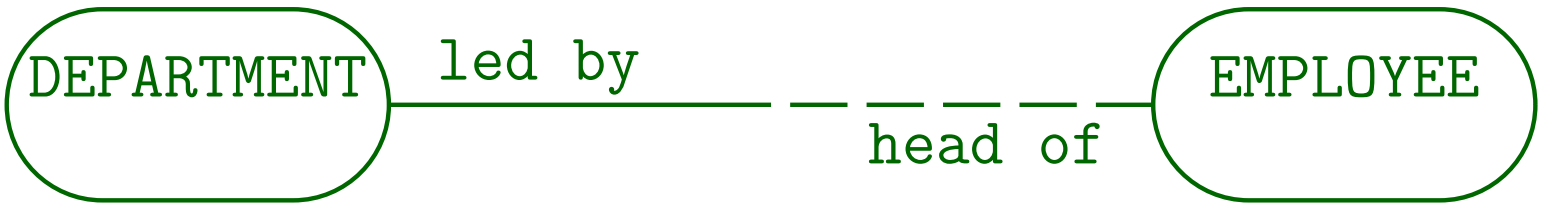
  A student cannot be inserted without a course, and a course cannot be inserted without a student. In general, when one defines cardinalities, one should think about elementary transactions. Which insertions/deletions must happen together such that the cardinality requirements are satisfied at the end of the transaction? If the transaction is too complicated, the cardinality requirements should be relaxed.

Cardinalities (8)

- One to one (mandatory to optional):



- In Barker notation:

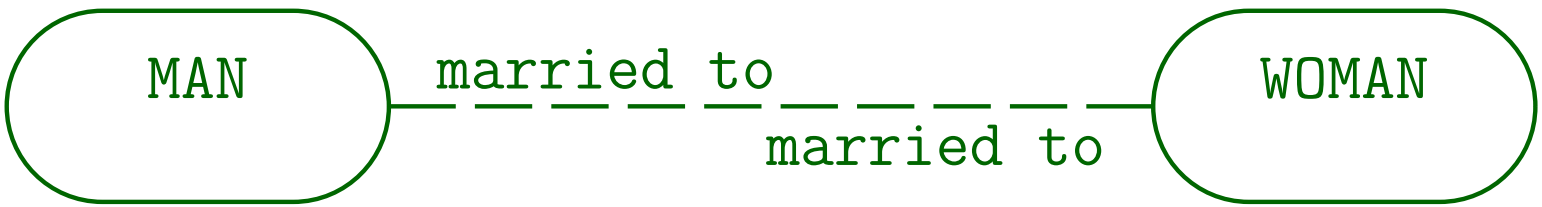


Every department is led by exactly one employee, an employee can be head of at most one department.

Cardinalities (9)

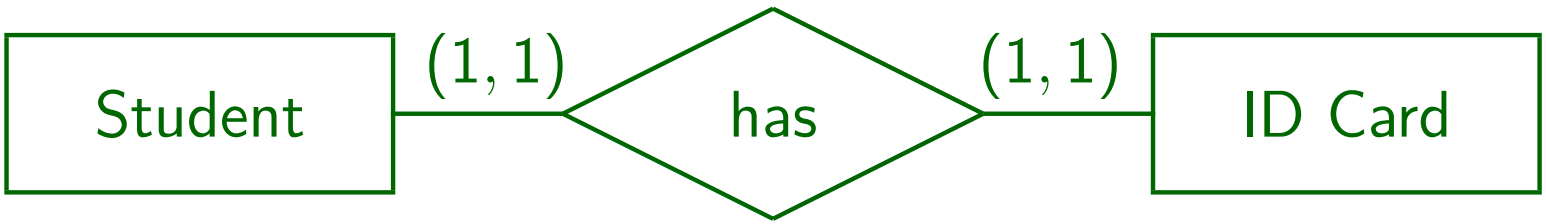
- One to one (optional to optional):

- In Barker notation:

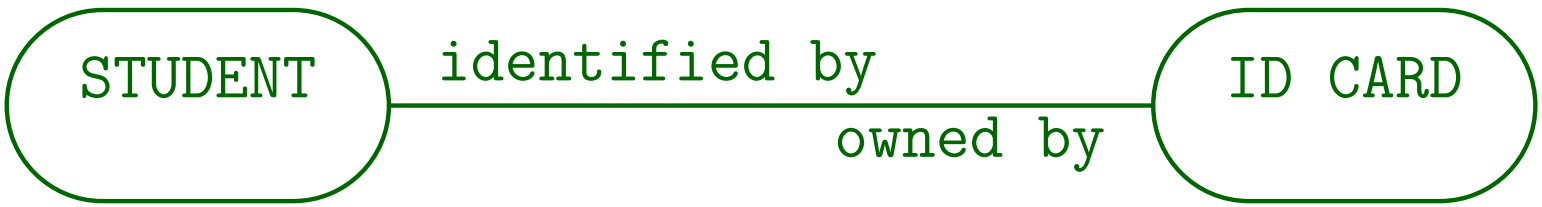


Cardinalities (10)

- One to one (mandatory to mandatory):



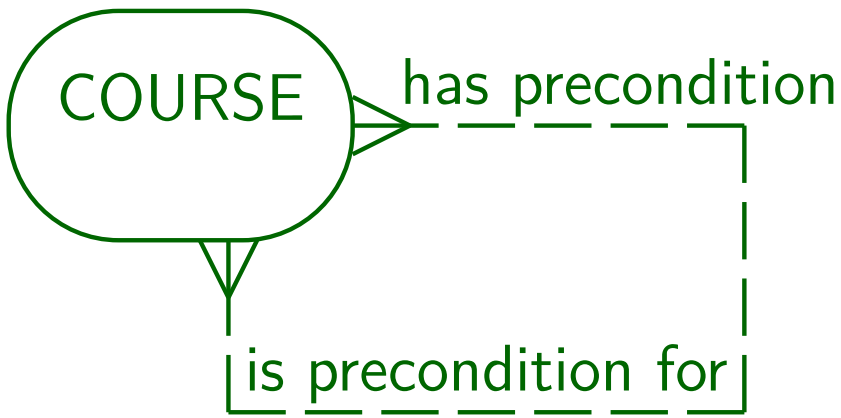
- In Barker notation:



- Uncommon. Consider merging the two entities.

Recursive Relationships

- Barker Notation does support recursive relationships (between two entities of the same type).



- The tool actually displays recursive relationships with a three-quarter circle ("swine ear").

Contents

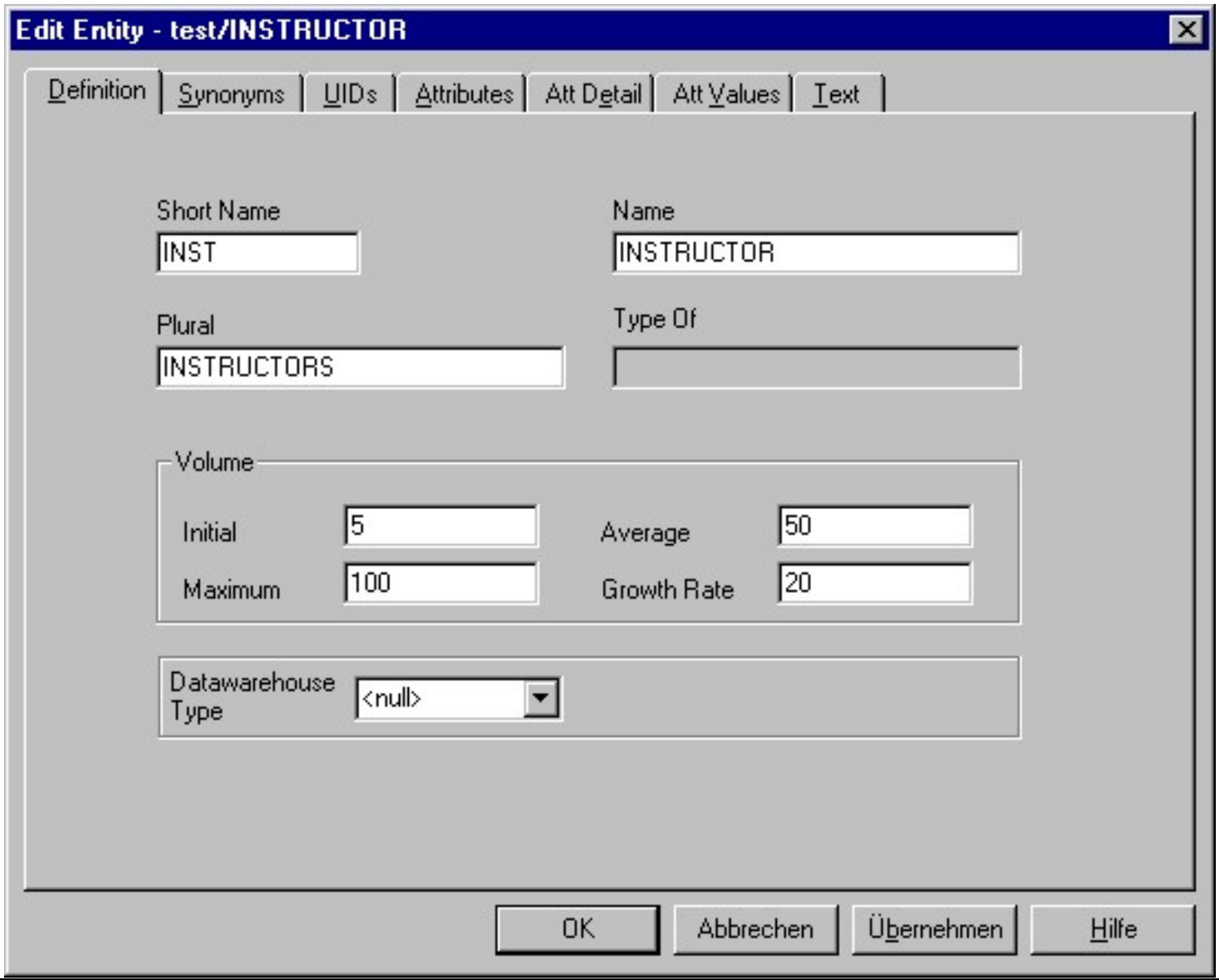
Entity Properties (1)

- By double clicking on an entity in a diagram, one opens the "Edit Entity" dialog box.
- It gives access to the properties of the entity, its attributes (including constraints for attribute values), unique identifiers (keys), and synonyms.
- In this way, much more information can be stored about the entity type than what is actually shown on the diagram.

  It is possible to customize what is shown in the diagram, e.g. all attributes, only mandatory attributes, only the primary key attributes, or no attributes.

Entity Properties (2)

- The dialog box has several tabs/pages:
  - Definition (name, plural, short name, etc.)
  - Synonyms (alternative names)
  - UIDs (keys and weak entity identification)
  - Attributes (list of all attributes)
  - Attribute Detail (one page for each attribute)
  - Attribute Values (constraints)
  - Text (documentation, notes).


Edit Entity - test/INSTRUCTOR
Definition
Synonyms
UIDs
Attributes
Att Detail
Att Values
Text
Short Name
INST
Name
INSTRUCTOR
Plural
INSTRUCTORS
Type Of
Volume
Initial
5
Average
50
Maximum
100
Growth Rate
20
Datawarehouse Type
<null>
OK
Abbrechen
Übernehmen
Hilfe

Entity Properties (4)

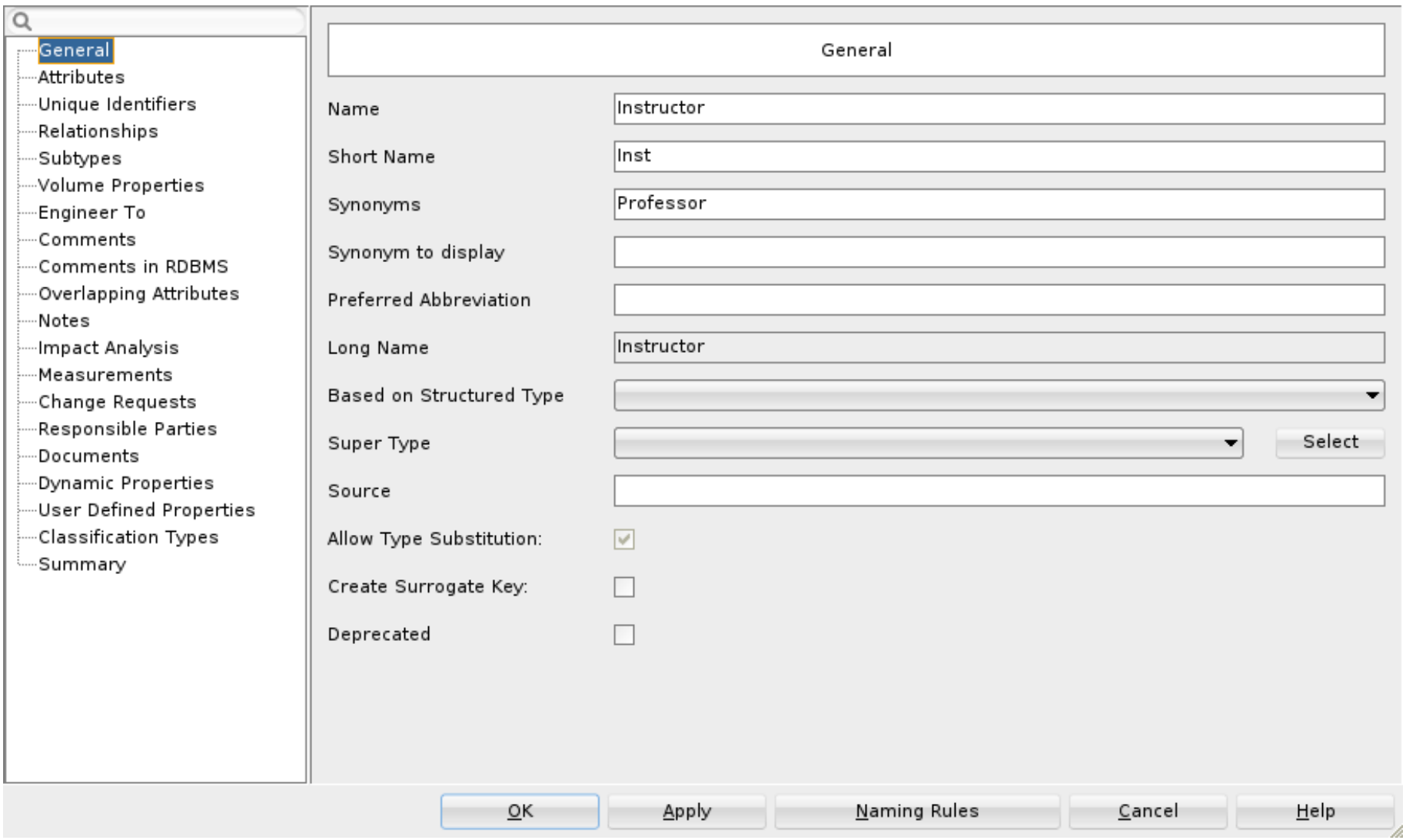
"Definition" Page:

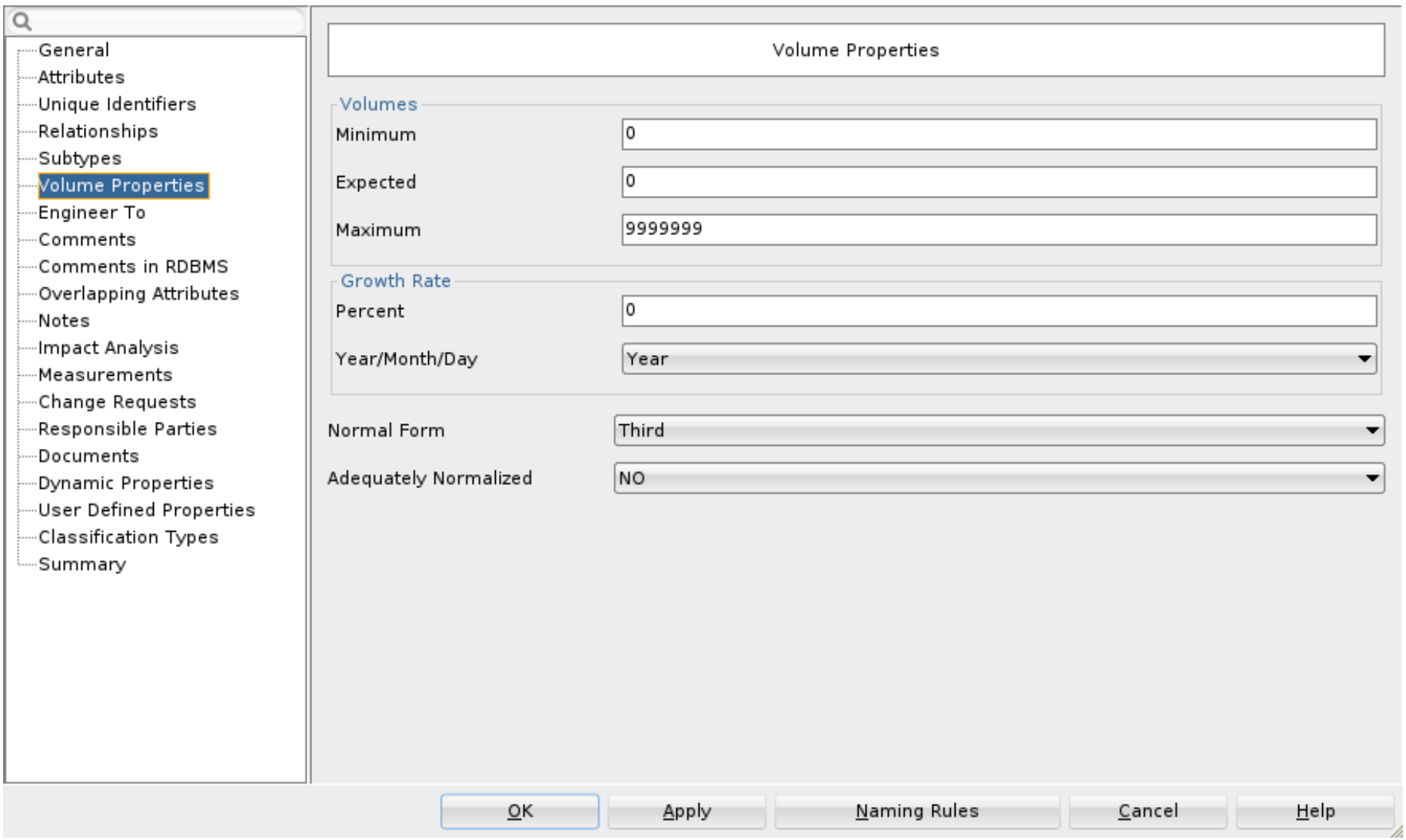
- Names of an entity (Short name, Name, Plural).
- Super class: "type of" (if this is a subclass).
- Expected number of entities of this type.

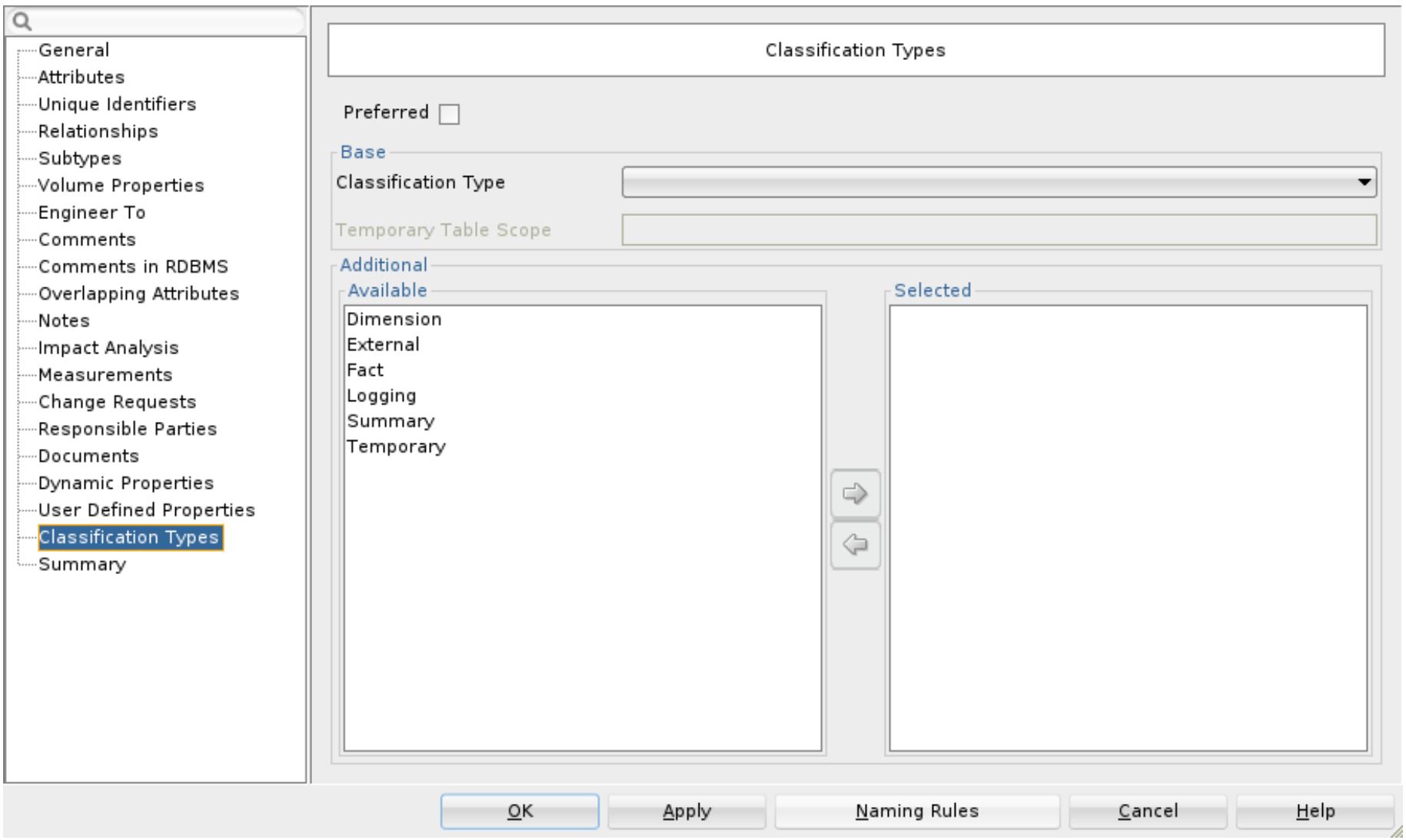
  This information is important for physical design.
  The initial, average, and maximum volume (number of entities) can be specified, as well as the annual growth rate (in percent). The meaning is a bit unclear, e.g. whether maximum is increased by the annual growth rate, and over which time interval the average is taken.

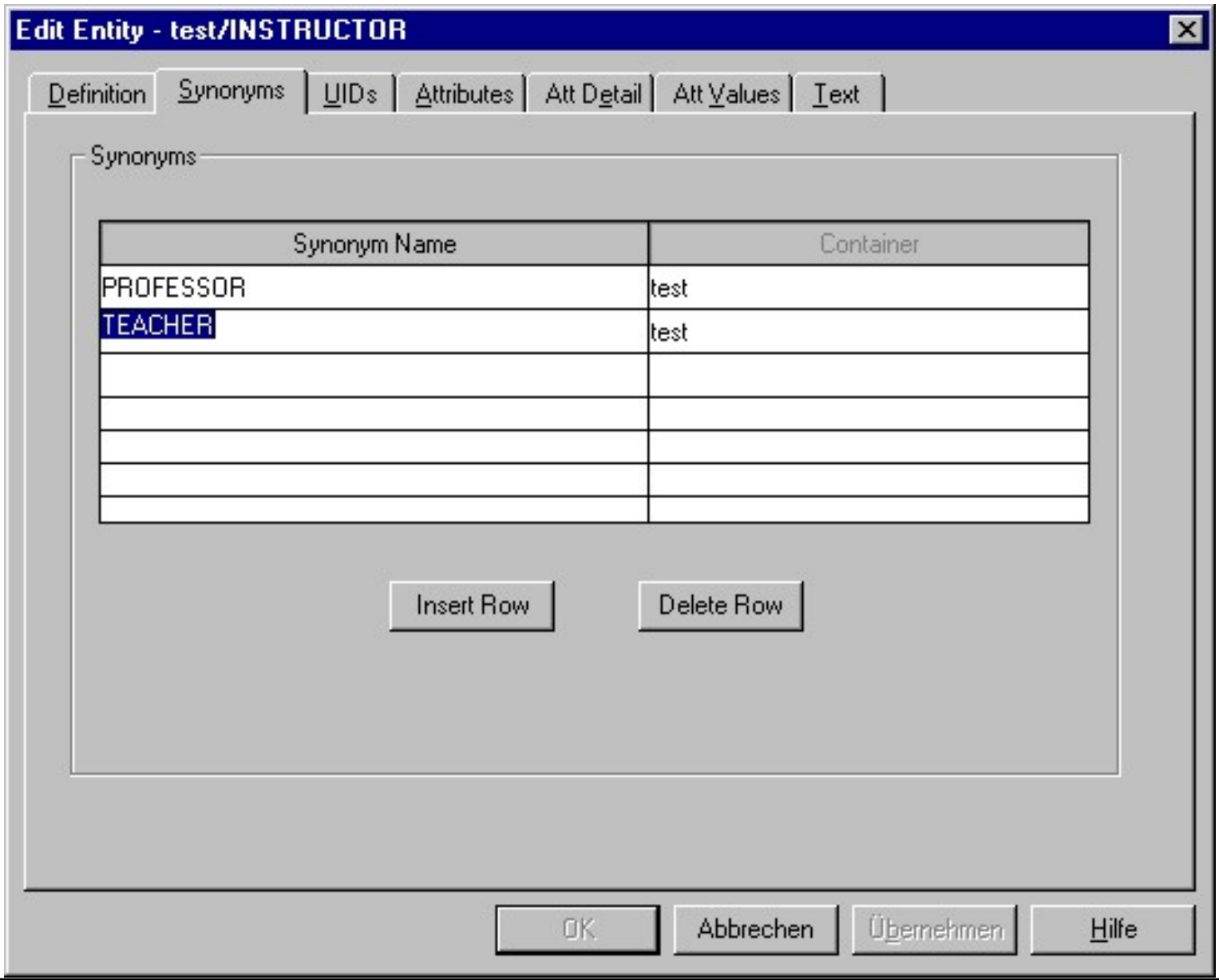
- Datawarehouse type (if DW application).

  "Fact" vs. "Dimension" tables (see below).


General
Attributes
Unique Identifiers
Relationships
Subtypes
Volume Properties
Engineer To
Comments
Comments in RDBMS
Overlapping Attributes
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
User Defined Properties
Classification Types
Summary
General
Name
Instructor
Short Name
Inst
Synonyms
Professor
Synonym to display
Preferred Abbreviation
Long Name
Instructor
Based on Structured Type
Super Type
Select
Source
Allow Type Substitution:
Create Surrogate Key:
Deprecated
OK
Apply
Naming Rules
Cancel
Help


General
Attributes
Unique Identifiers
Relationships
Subtypes
Volume Properties
Engineer To
Comments
Comments in RDBMS
Overlapping Attributes
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
User Defined Properties
Classification Types
Summary
Volume Properties
Volumes
Minimum
0
Expected
0
Maximum
9999999
Growth Rate
Percent
0
Year/Month/Day
Year
Normal Form
Third
Adequately Normalized
NO
OK
Apply
Naming Rules
Cancel
Help

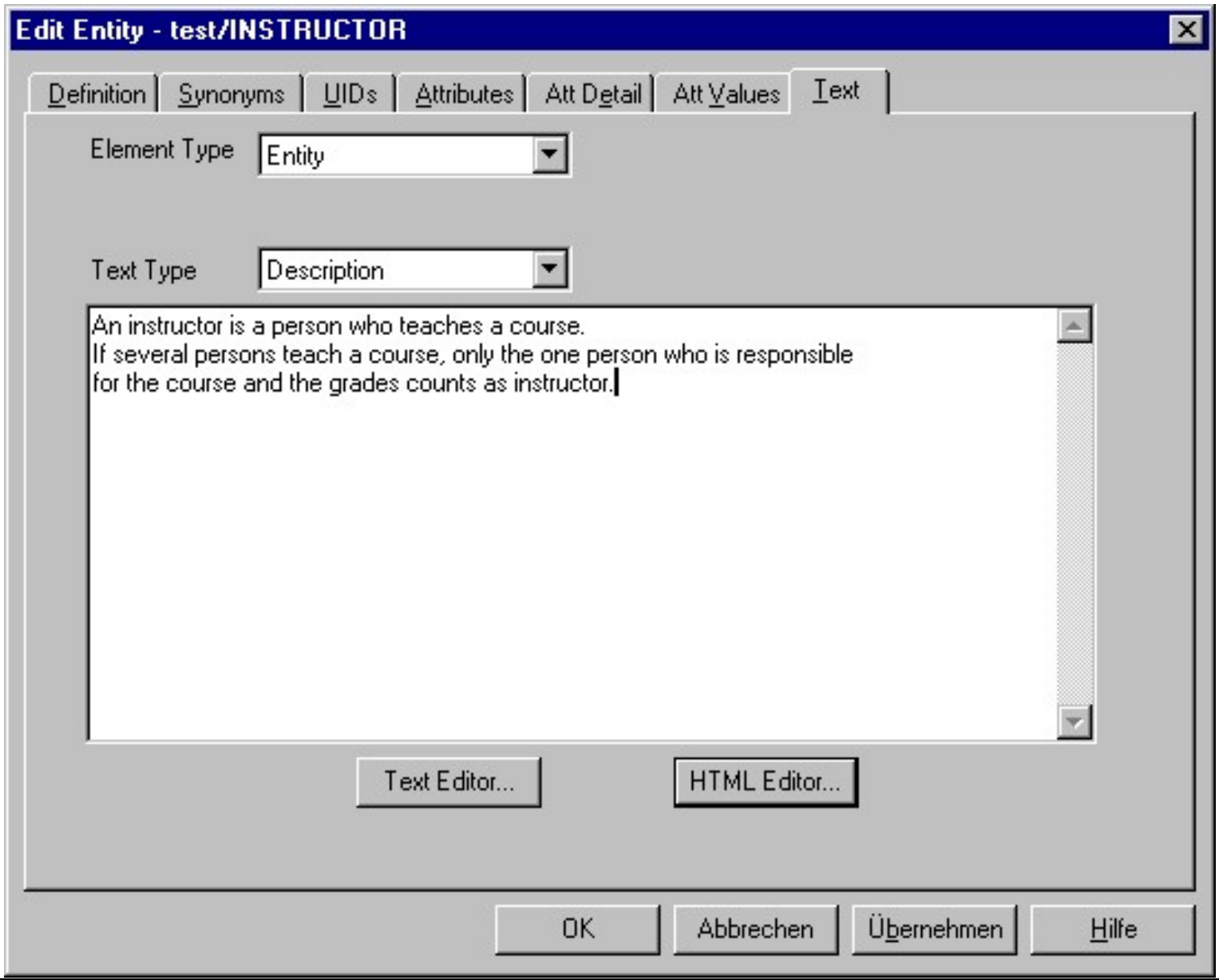

General
Attributes
Unique Identifiers
Relationships
Subtypes
Volume Properties
Engineer To
Comments
Comments in RDBMS
Overlapping Attributes
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
User Defined Properties
Classification Types
Summary
Classification Types
Preferred
Base
Classification Type
Temporary Table Scope
Additional
Available
Dimension
External
Fact
Logging
Summary
Temporary
Selected
OK
Apply
Naming Rules
Cancel
Help


Edit Entity - test/INSTRUCTOR
Definition
Synonyms
UIDs
Attributes
Att Detail
Att Values
Text
Synonyms
Synonym Name
Container
PROFESSOR
test
TEACHER
test
Insert Row
Delete Row
OK
Abbrechen
Übernehmen
Hilfe

Entity Properties (9)

"Synonyms" Page:

- Alternative names for the entity can be defined.
- It is an important task in database design to check whether things named differently by different users are really the same concept.
- The Oracle DBMS permits to define synonyms for tables and other database objects (not in SQL-92).
- However, it is also possible to treat synonyms only as part of the design documentation, and not to reflect them in the final relational schema.


Edit Entity - test/INSTRUCTOR
Definition
Synonyms
UIDs
Attributes
Att Detail
Att Values
Text
Element Type
Entity
Text Type
Description
An instructor is a person who teaches a course.
If several persons teach a course, only the one person who is responsible
for the course and the grades counts as instructor.
Text Editor...
HTML Editor...
OK
Abbrechen
Übernehmen
Hilfe

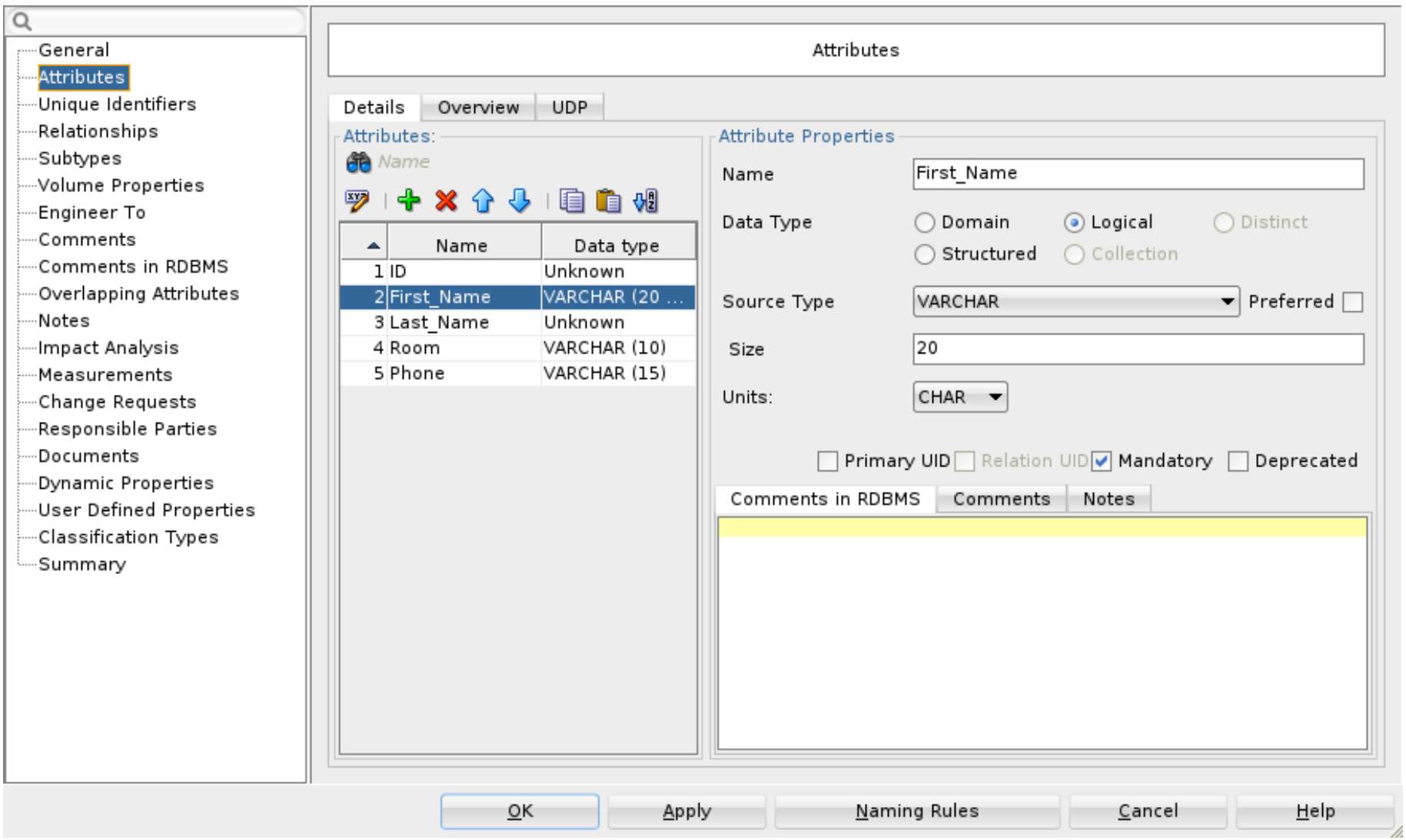
Entity Properties (11)

"Text" Page:

- Textual descriptions/definitions of entities and attributes can be stored (in ASCII or HTML).
- Also "Notes" about entities and attributes can be stored, and the system can be extended to allow other text types.
- These texts will be part of the design documentation which can be generated by the "Repository Reports" Utility.

In Oracle, comments can also be stored in the data dictionary.

Contents


General
Attributes
Unique Identifiers
Relationships
Subtypes
Volume Properties
Engineer To
Comments
Comments in RDBMS
Overlapping Attributes
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
User Defined Properties
Classification Types
Summary
Attributes
Details
Overview
UDP
Attributes:
Name
Name
Data type
1 ID Unknown
2 First_Name VARCHAR (20 ...
3 Last_Name Unknown
4 Room VARCHAR (10)
5 Phone VARCHAR (15)
Attribute Properties
Name
First_Name
Data Type
Domain
Logical
Distinct
Structured
Collection
Source Type
VARCHAR
Preferred
Size
20
Units:
CHAR
Primary UID
Relation UID
Mandatory
Deprecated
Comments in RDBMS
Comments
Notes
OK
Apply
Naming Rules
Cancel
Help

Attributes (1)

- Here the entity attributes can be declared with the following information:
  - Name
  - Sequence number to define the order in which the attributes will be displayed (see below).
  - Domain, Data Type/Format (see below).
  - Is this attribute optional (i.e. possibly null)?
  - Is this attribute part of the primary key?
  - A short comment on the attribute.

Attributes (2)

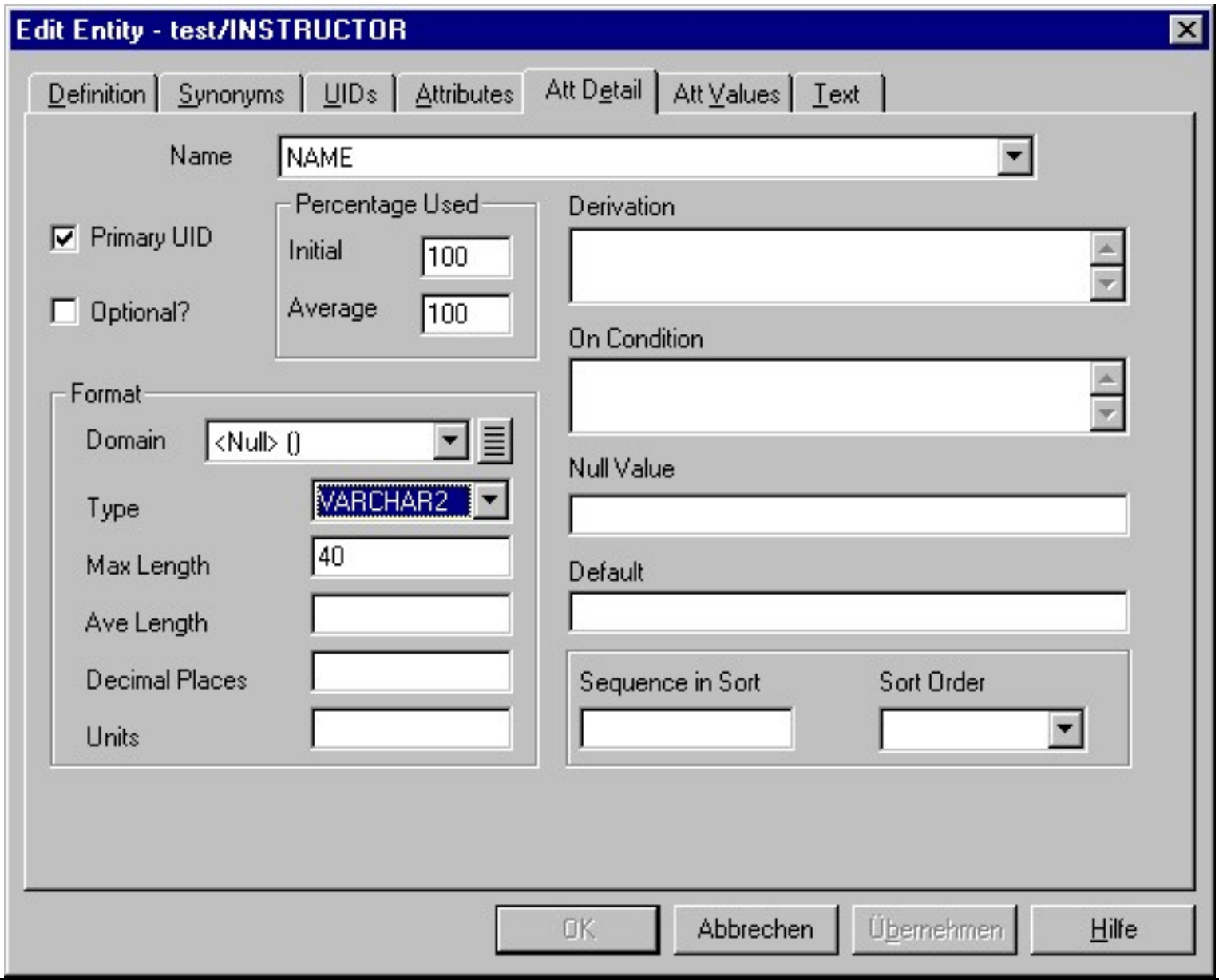
- Possible attribute formats (data types) are
  `CHAR`, `DATE`, `IMAGE`, `INTEGER`, `MONEY`, `NUMBER`, `PHOTOGRAPH`, `SOUND`, `TEXT`, `TIME`, `TIMESTAMP`, `VARCHAR2`, `VIDEO`.
- Not all of these data types exist in Oracle or all the other supported goal systems.
- It is a task of the logical design phase to map the data types used in the conceptual schema to data types supported in the selected DBMS.

  E.g. `IMAGE`, `VIDEO`, and `SOUND` would be mapped to a binary large object in Oracle (`BLOB`). The database design transformer contains mappings for various systems.

Attributes (3)

- Some types (e.g. `CHAR`, `VARCHAR2`, `NUMBER`) require a maximal length, some (e.g. `NUMBER`) also the number of decimal places after the point (precision, "dec").
- Instead of defining the data types for every attribute separately, one should use domains (see below).
- If the sequence number is left blank, one gets the default attribute sequence: (1) primary key attributes, (2) mandatory attributes, (3) optional attributes. Each group is alphabetically sorted.

  The alphabetical order is usually not what is intended.


Edit Entity - test/INSTRUCTOR
Definition
Synonyms
UIDs
Attributes
Att Detail
Att Values
Text
Name
NAME
Primary UID
Optional?
Percentage Used
Initial
100
Average
100
Derivation
On Condition
Format
Domain
<Null> ()
Type
VARCHAR2
Max Length
40
Ave Length
Decimal Places
Units
Null Value
Default
Sequence in Sort
Sort Order
OK
Abbrechen
Übernehmen
Hilfe

Attributes (5)

- Information on the "Attribute Detail" page:
  - Physical design information: average length and percentage of entities having a non-null value.
  - Units for the attribute (e.g. "kg", "cm", "in").
  - A derivation formula/algorithm if this attribute is derived.

    No syntax checks are done on e.g. the derivation formula (any text can be entered, not only SQL).
  - A condition when this attribute can be used.

    E.g. "Flight Hours" is defined if/only if (?) "Job" is pilot. E.g. only associate and full professors can have a value in the column `TENURE_SINCE`.

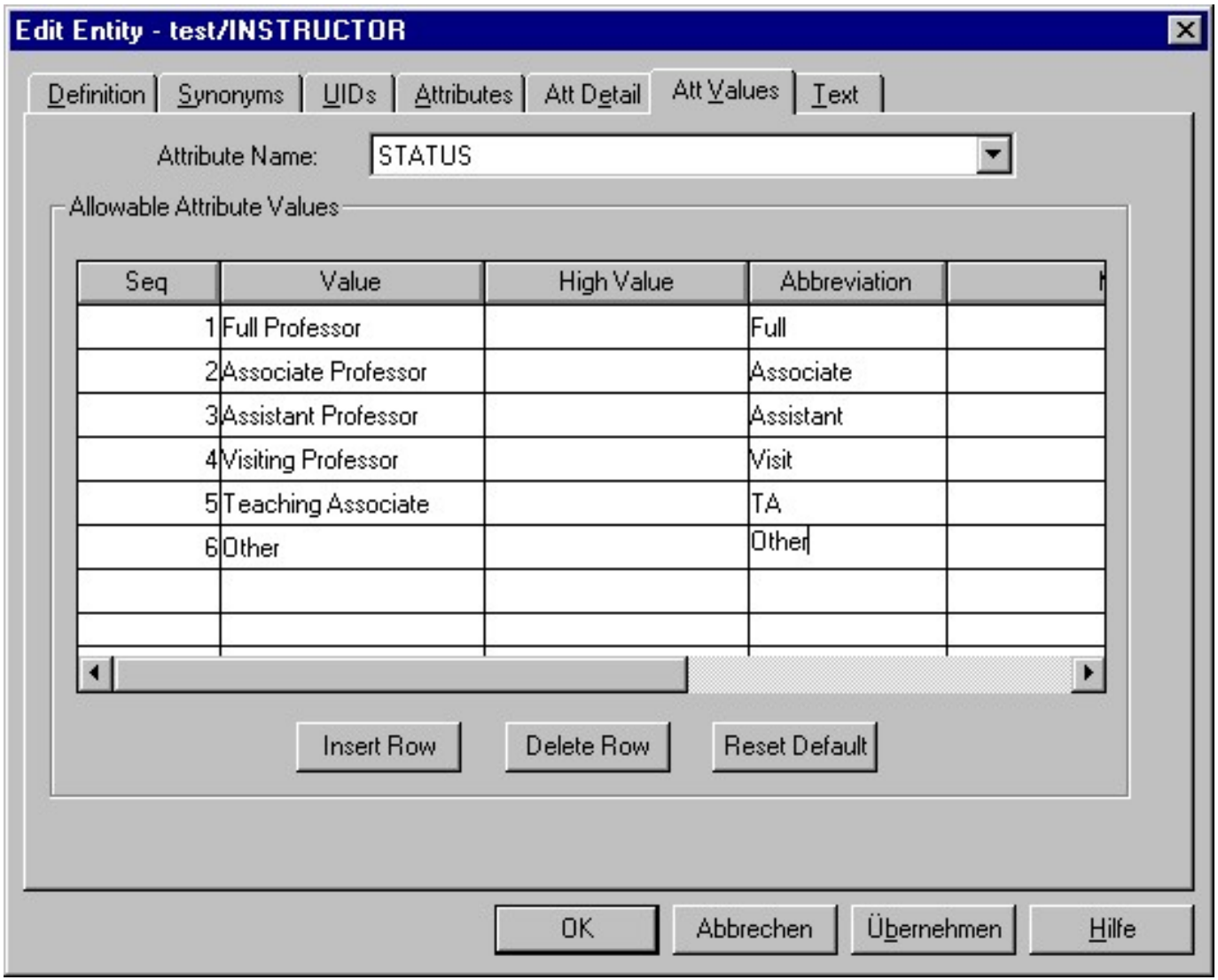
Attributes (6)

- Information on the "Attribute Detail" page:
  - A representation for a null value if the DBMS does not support null values.

    This would be strange for a modern DBMS.
  - A default value (to simplify data entry).

    A default value can be specified in the `CREATE TABLE` statement.

    Oracle Designer did not check the default value against the type.
  - If the entities/rows should be sorted by this attribute, the relative priority of this sort criterion and order (asc/desc).


Edit Entity - test/INSTRUCTOR
Definition
Synonyms
UIDs
Attributes
Att Detail
Att Values
Text
Attribute Name:
STATUS
Allowable Attribute Values
Seq
Value
High Value
Abbreviation
1 Full Professor Full
2 Associate Professor Associate
3 Assistant Professor Assistant
4 Visiting Professor Visit
5 Teaching Associate TA
6 Other Other
Insert Row
Delete Row
Reset Default
OK
Abbrechen
Übernehmen
Hilfe

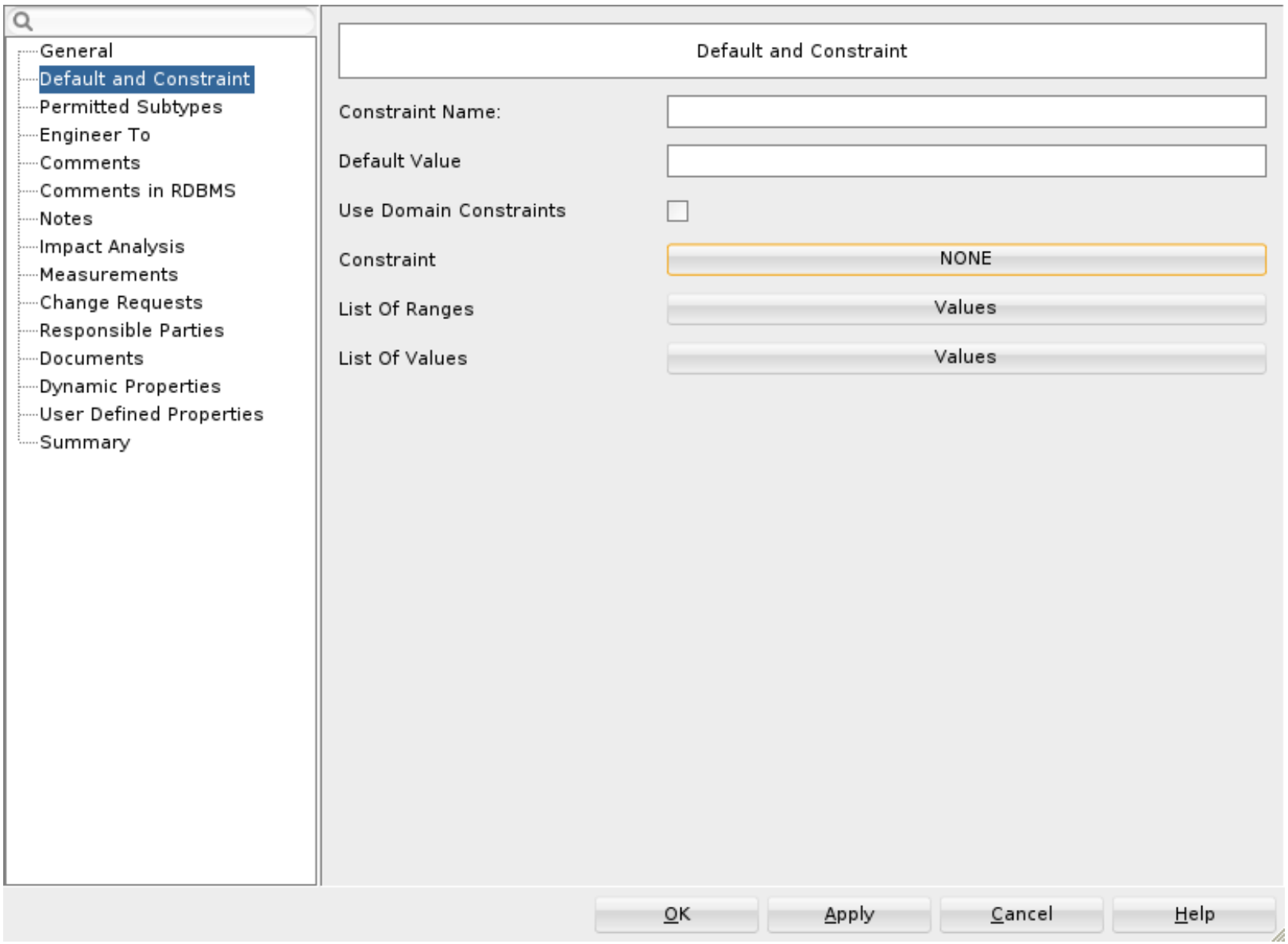
Attributes (8)

"Attribute Values" Page:

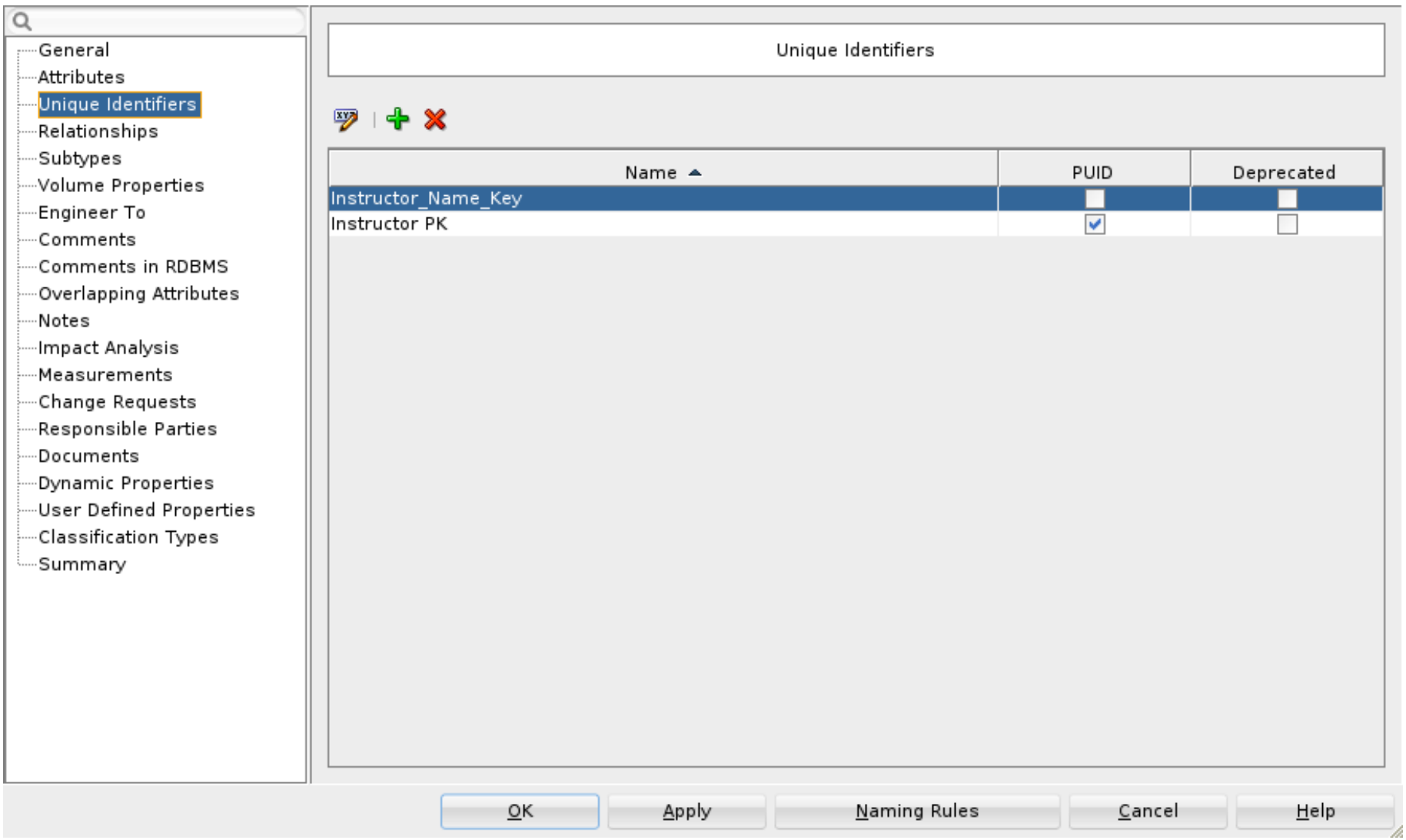
- On this page, restrictions for the values of an attribute can be defined (e.g. for "enumeration type" attributes).
- One can define all possible values of an attribute:
  - Value
  - Sequence number

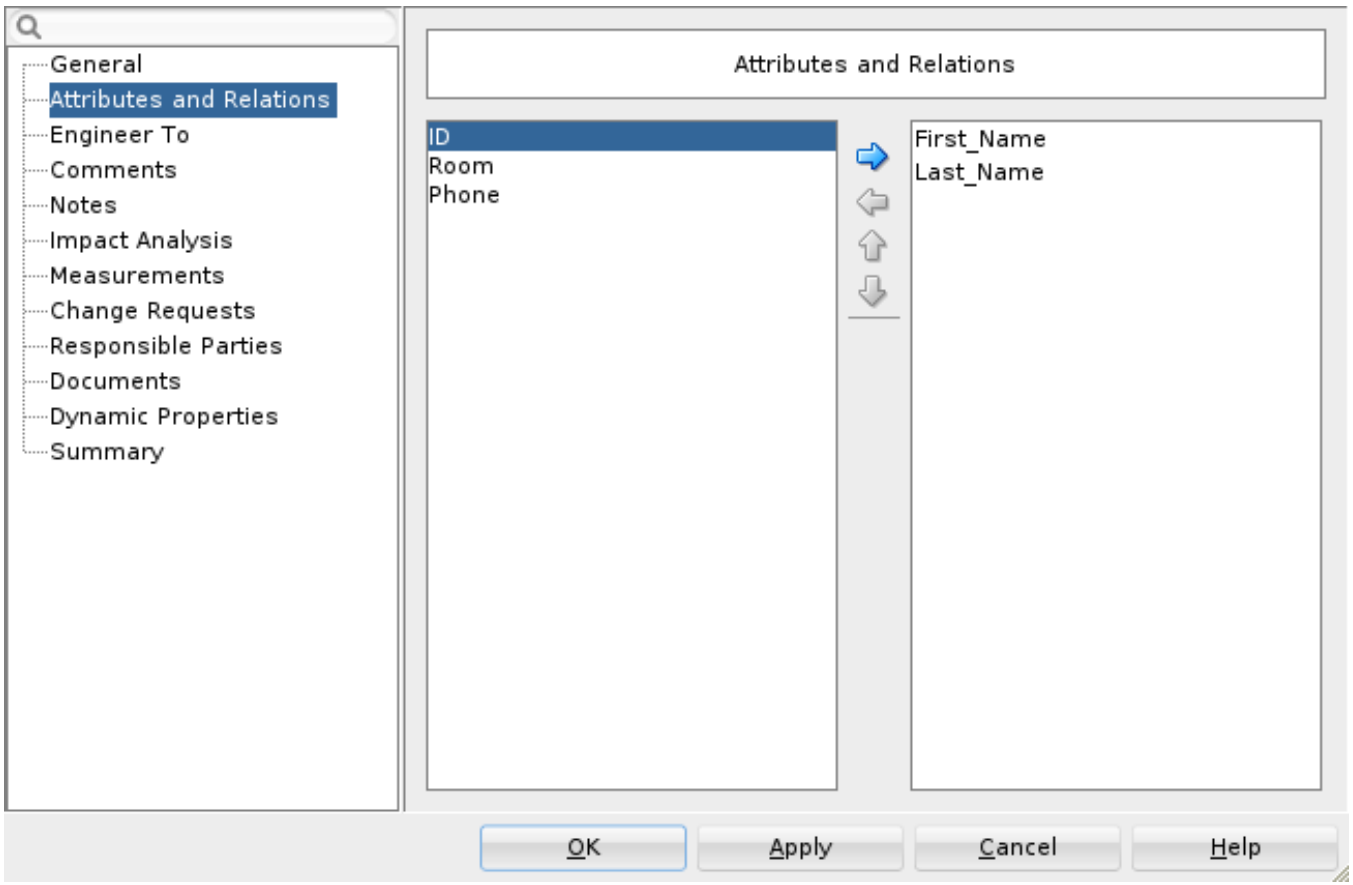
    E.g. for printed documentation, menus in application programs.
  - Abbreviation
  - Meaning (help text)

    Already in the ER-design, information is collected that later can be used for the generation of application programs (forms for inserting data).
- Alternatively, one can define an interval of legal values.


General
Default and Constraint
Permitted Subtypes
Engineer To
Comments
Comments in RDBMS
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
User Defined Properties
Summary
Default and Constraint
Constraint Name:
Default Value
Use Domain Constraints
Constraint
NONE
List Of Ranges
Values
List Of Values
Values
OK
Apply
Cancel
Help

Contents


General
Attributes
Unique Identifiers
Relationships
Subtypes
Volume Properties
Engineer To
Comments
Comments in RDBMS
Overlapping Attributes
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
User Defined Properties
Classification Types
Summary
Unique Identifiers
Name
PUID
Deprecated
Instructor_Name_Key
Instructor PK
OK
Apply
Naming Rules
Cancel
Help


General
Attributes and Relations
Engineer To
Comments
Notes
Impact Analysis
Measurements
Change Requests
Responsible Parties
Documents
Dynamic Properties
Summary
Attributes and Relations
ID
Room
Phone
First_Name
Last_Name
OK
Apply
Cancel
Help

UIDs/Keys (1)

"UIDs" Page:

- On this page keys (unique identifiers) can be defined.
- More than one key can be declared, but exactly one must be marked as primary key.

  Primary key information entered on this page is automatically reflected on the "Attributes" pages.

- The Designer does not prevent that a primary key attribute is optional (which is illegal in SQL).
- Each key/unique identifier must be named.

UIDs/Keys (2)

- Not only attributes, but also relationships can be used as a means for identification.

  Entity types that use this are also called weak entity types, see below.

- E.g. if instructors had a relationship to departments, and the UID consists of this relationship and the instructor name, there can be instructors with the same name in different departments.

  The foreign key that contains the ID of the department together with the instructor name becomes a composed key of the instructor. This works only for relationships with a $(1, 1)$-cardinality, e.g. on the many side of a one-to-many relationship. The Designer does not check this.

References

- Barker: CASE*Method, Entity Relationship Modelling.
  Addison-Wesley, 1990, ISBN 0-201-41696-4, ca. $61.
- Koletzke/Dorsey: Oracle Designer Handbook, 2nd Edition.
  ORACLE Press, 1998, ISBN 0-07-882417-6, ca. $40.
- A. Lulushi: Inside Oracle Designer/2000.
  Prentice Hall, 1998, ISBN 0-13-849753-2, ca. $50.
- Oracle/Martin Wykes: Designer/2000, Release 2.1.1, Tutorial.
  Part No. Z23274-02, Oracle, 1998.
- Heli Helskyaho: Oracle SQL Developer Data Modeler for Database Design Mastery.
  McGraw Hill Education / Oracle Press, 2015, ISBN 0071850090, 336 pages.
- Teorey: Database Modeling & Design, 3rd Edition.
  Morgan Kaufmann, 1999, ISBN 1-55860-500-2, ca. $32.
- Elmasri/Navathe: Fundamentals of Database Systems, 2nd Ed.,
  Appendix A, “Alternative Diagrammatic Notations”.
- Rauh/Stickel: Konzeptuelle Datenmodellierung (in German), Teubner, 1997.