



Software & Systems
Engineering

Semantik der UML 2.0

A Methodological Approach for Object-relational Database
Design using UML

Bearbeiter: Han Jiang

Betreuer: Stefan Wagner



Gliederung

- F Datenbank Geschichte
 - Objekt-Relationale Datenbanken

- F UML
 - Klassendiagramme

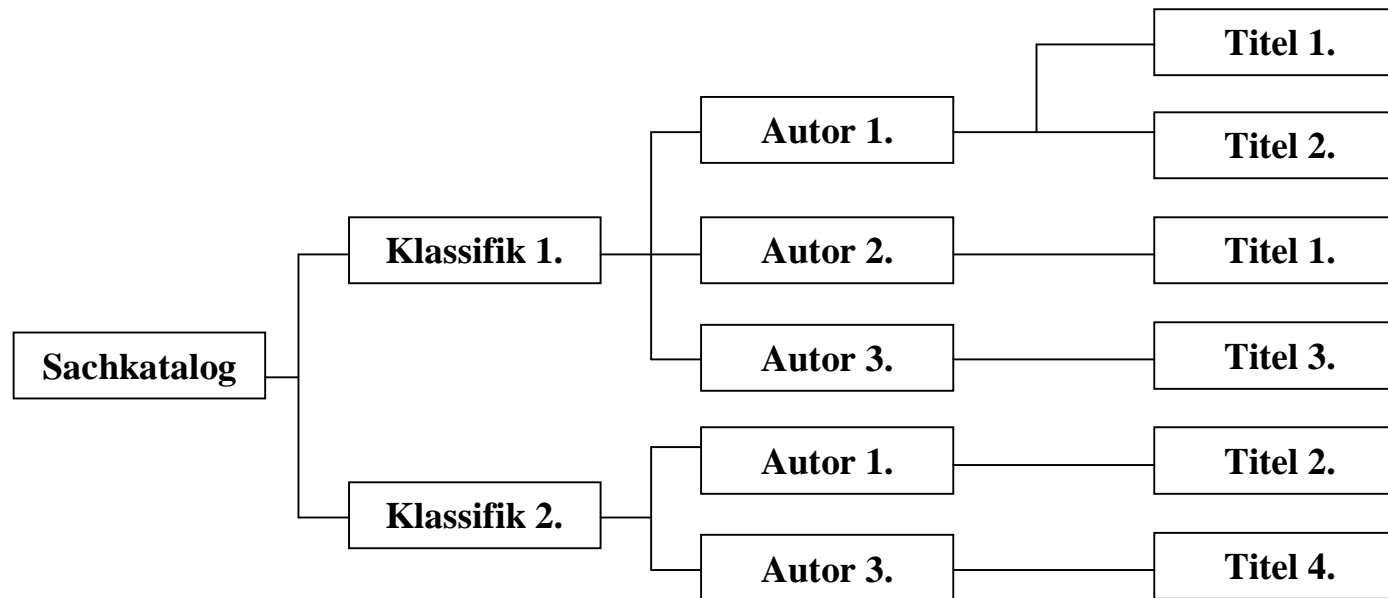
- F Objekt-Relationales Datenbank-Design mit der UML

- F Zusammenfassung



Datenbank Geschichte

F Hierarchisches Datenmodell

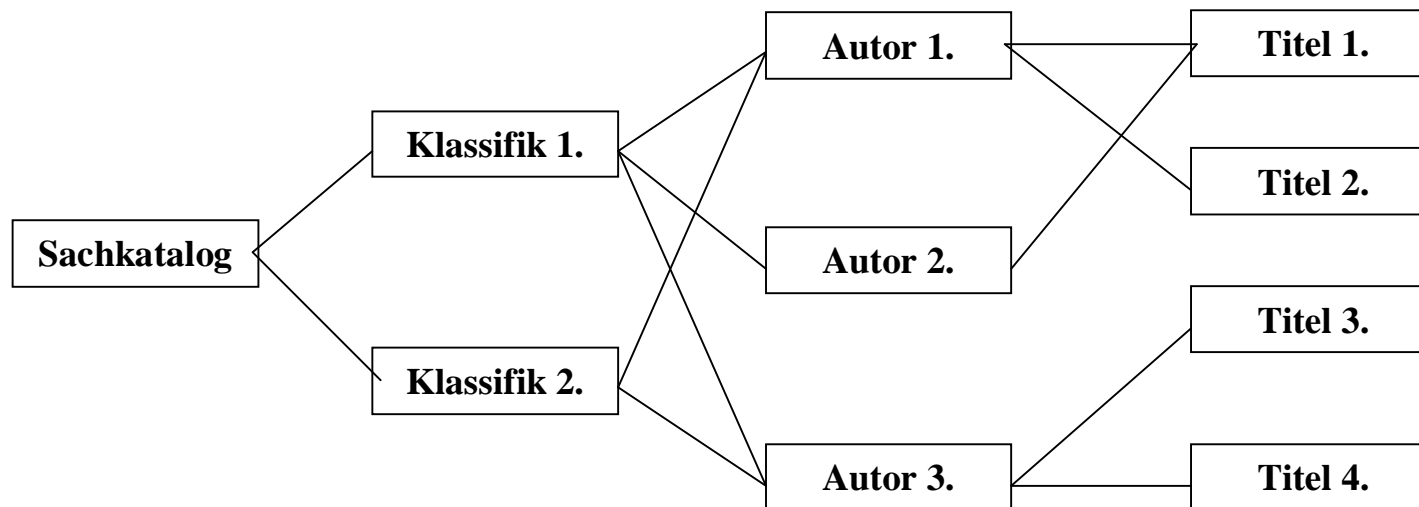


- Nachteile:
1. ein einzelner Dateneintrag kann mehrfach erscheinen.
 2. Verknüpfung über mehrere Ebene unmöglich



Datenbank Geschichte

F Netzwerkmodell



Nachteile: 1. die Struktur sehr schnell ziemlich undurchsichtig.



Datenbank Geschichte

F Relationales Datenmodell

Studenten	
MatrNr	Name
26120	Fichte
25403	Jonas
...	...

hören	
MatrNr	VolNr
25403	5022
26120	5001
...	...

Vorlesung	
VolNr	VolNr
5001	Grundzuege
5002	Glaube und Wissen
...	...

- Vorteile:
1. die Struktur der Datenbank kann verändert werden.
 2. virtuelle Tabellen mit unterschiedlicher logischen Struktur



Datenbank Geschichte

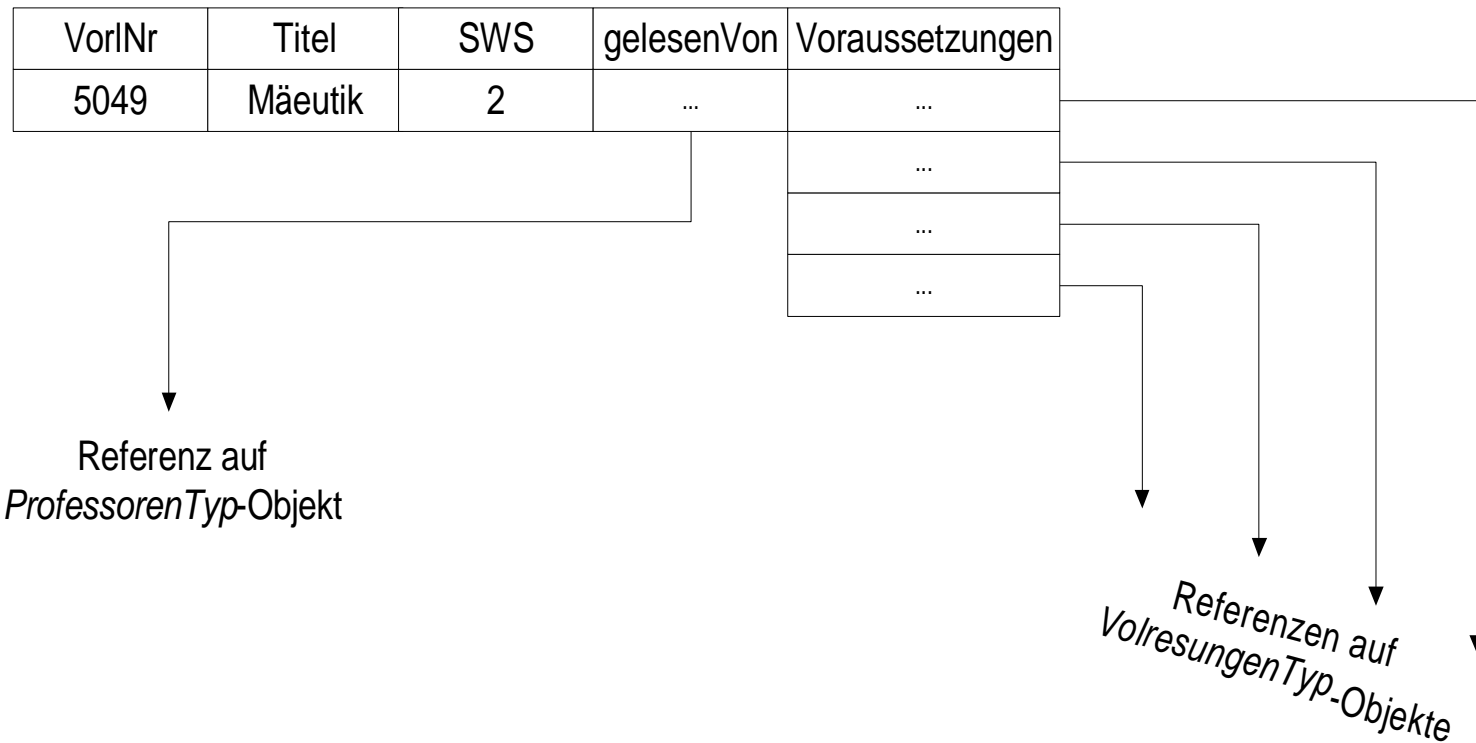
F Was ist neu in objekt-relationalen Datenbanken?

- Große Objekte(LOB),
- Mengewertige Attribute (z.B: Array),
- Geschachtelte Relationen,
- Typedeklarationen,
- Referenzen,
- Objektidentität
- ...



Datenbank Geschichte

F Objekt-Relationales Datenmodell





UML Einführung

F Was ist das UML?

eine Sprache zur Spezifikation, Visualisierung, Konstruktion und Dokumentation von Modellen für Softwaresysteme.

✓ **Wichtig: UML ist kein Methode !**



UML Einführung

F Wozu wird UML gebraucht ?

- Software ist zunehmend komplexer.
 - Windows 2000 bis 40 Millionen Linien Programmcode
 - Ein Programmierer kann den ganzen Code nicht allein verwalten.
- Es ist schwer für Entwickler den Code zu verstehen, den er nicht selbst geschrieben hat.
- Wir brauchen eine einfache Repräsentation für komplexe Systeme.



UML Einführung

F Die Modellelemente der UML werden nach Diagrammtypen gegliedert :

- Anwendungsfalldiagramm
- **Klassendiagramm**
- Aktivitätsdiagramm
- Kollaborationsdiagramm
- Sequenzdiagramm
- Zustandsdiagramm
- Komponentendiagramm
- Einsatzdiagramm



Klassendiagramme

F Definition von Klassendiagrammen

- Diagramm, das eine Menge von statischen Modellelementen zeigt, wie Klassen und ihre Beziehungen.
- Ein Klassendiagramm dient der Darstellung der statischen Entwurfssicht eines Systems.

F Beschreibung

Die Struktur eines Systems mit Hilfe von

- Klassen
- Beziehungen zwischen Klassen



Notation von Klassendiagrammen

Klasse	Notation			
Normale Klasse	<table border="1"><tr><td>Klassenname</td></tr><tr><td>Attribute1 : Typ = Initialwert Attribute2 : Typ = Initialwert</td></tr><tr><td>operation1() operation1()</td></tr></table>	Klassenname	Attribute1 : Typ = Initialwert Attribute2 : Typ = Initialwert	operation1() operation1()
Klassenname				
Attribute1 : Typ = Initialwert Attribute2 : Typ = Initialwert				
operation1() operation1()				
Metaklasse	<table border="1"><tr><td><<metaclass>> Klassenname</td></tr></table>	<<metaclass>> Klassenname		
<<metaclass>> Klassenname				
Parametrisierbare Klasse	<table border="1"><tr><td>ParametrisierbareKlasse</td></tr><tr><td>Attribute1 : Typ = Initialwert</td></tr><tr><td>operation1()</td></tr></table> <p>Parameter</p>	ParametrisierbareKlasse	Attribute1 : Typ = Initialwert	operation1()
ParametrisierbareKlasse				
Attribute1 : Typ = Initialwert				
operation1()				
Schnittstellklasse	<table border="1"><tr><td><<interface>> Klassenname</td></tr></table>	<<interface>> Klassenname		
<<interface>> Klassenname				
Abstrakte Klasse	<table border="1"><tr><td>Klassenname {abstract}</td></tr></table>	Klassenname {abstract}		
Klassenname {abstract}				



Notation von Klassendiagrammen

F Attribute: [Visibility] [/]Name[:type][multiplicity][=default] [{property-string}]

Typen	Beispiele
/abgeleitetes Attribut	/alter:Integer {alter = today – gebDatum}
<u>Klassenattribut</u>	-counter:Integer= 0 {counter>=0}
+publicAttribut	+pi:Double = 3.1415926 {readonly}
#protectedAttribut	#name: String=„unbekannt“
-privateAttribut	
~packageattribut	



Notation von Klassendiagrammen

F Operation: [Visibility]name([Parameter-list]):[return-result][{properties}]

Operationen	Beispiele
operation() {abstract}	anzeigen{abstract}
<u>KlassenOperation()</u>	+getCounter():Integer{veraltet}
+publicOperation()	
#protectedOperation()	#addRufNr(rufNr:String)
-privateOperation	
~packageOperation()	



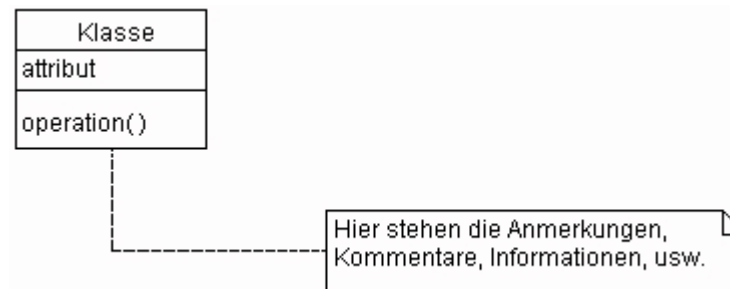
Notation von Klassendiagrammen

F Properties:

- Zusicherung: beschreibt ein Bedingung oder Integritätsregel.
Beispiel: Length: int = 0 { length >= 0}
- Merkmal: Modellelemente weitere Eigenschaften hinzufügen.
Beispiel: {abstract} , {Version=3,11}

F Notizen:

Kommentare zu einem Diagramm oder einem oder mehreren beliebigen Modellelementen Notation:

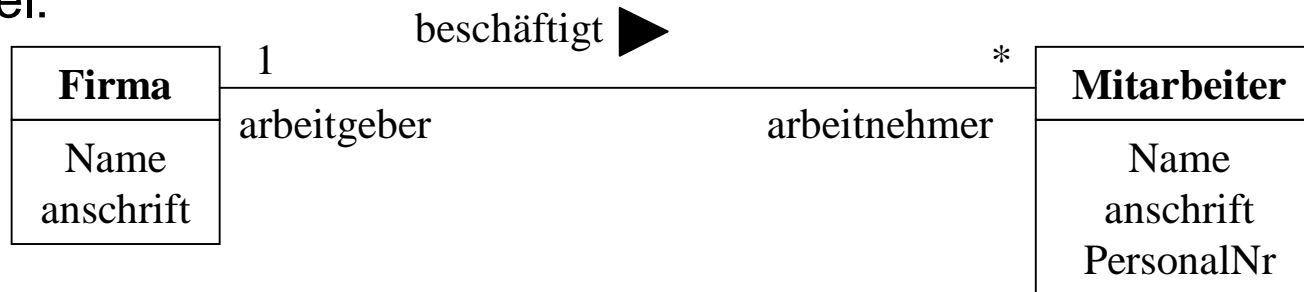




Notation von Beziehungen

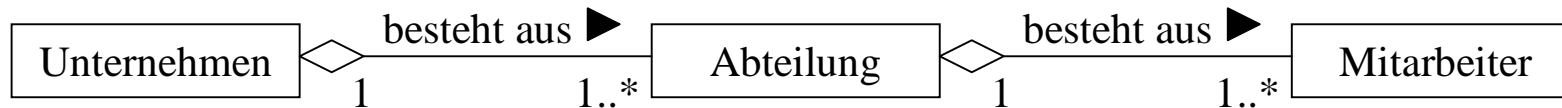
F Assoziation: die Relation zwischen Klassen.

Beispiel:



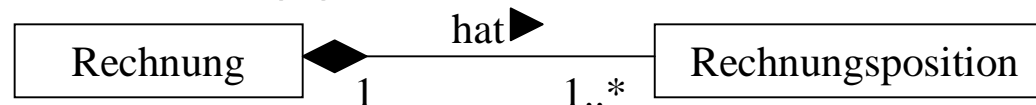
F Aggregation: eine Assoziation, um Ganzes-Teile-Hierarchie darzustellen.

Beispiel:



F Komposition: eine strenge Form der Aggregation, bei der die Teile vom Ganzen existenzabhängig sind.

Beispiel:

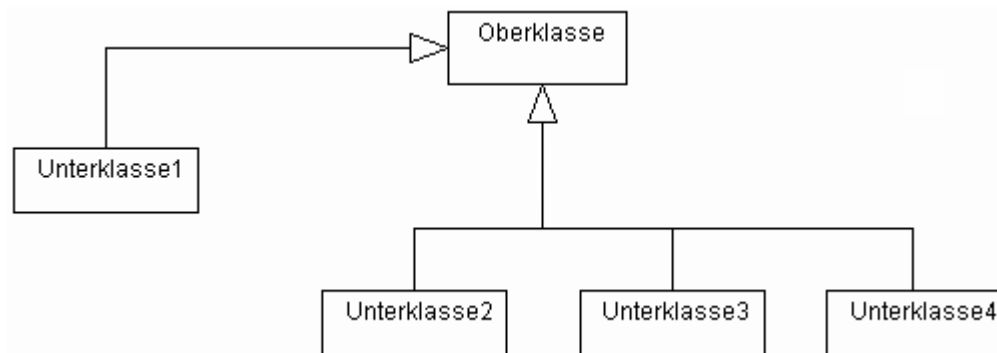




Notation von Beziehungen

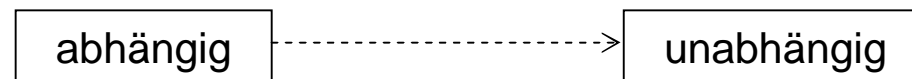
F Vererbung: Relation zwischen Oberklasse und Unterklasse.

Notation:



F Abhängigkeit: Beziehung zwischen zwei Modellelementen.
Eine Änderung in einem Element bewirkt andere Änderung.

Notation:





UML-Profile

F Definition:

Vordefinierter Satz von Constraints, Tagged Values und Stereotypen, um das UML Metamodell für eine spezielle Umgebung, einen Anwendungsbereich oder einen Prozess anzupassen.

- Stereotyp
 - Ereiterungsmechanismus
- Tagged Value
 - Schlüsselwort-Wert Paar, das die Semantik einzelner Modellelemente erweitert.
- Constraint
 - Präzisiert oder beschränkt die Semantik eines beliebigen UML Modellelements






Objekt-Relationales Datenbank- Design mit der UML

F Vorteile:

- Objekt-Relationale Datenbank darstellbar
- das ganze System, inklusive des Datenbank-Schemas, kann auf dem gleichen Weg formuliert werden.
- UML ist eine erweiterbare Sprache



Stereotypen für das Datenbank-Design

	Database Element	UML Element	Stereotype	Icon
Architectural	Database	Component	<<Database>>	
	Schema	Package	<<Schema>>	
Conceptual	Persistent class	Class	<<Persistent>>	
	Multivalued Attribute	Attribute	<<MA>>	
	Calculated Attribute	Attribute	<<DA>>	
	Composed Attribute	Attribute	<<CA>>	
	Identifier	Attribute	<<ID>>	
Logical	Table	Class	<<Table>>	
	View	Class	<<View>>	
	Column	Attributes	<<Column>>	
	Primary Key	Attributes	<<PK>>	
	Foreign Key	Attributes	<<FK>>	
	NOT NULL Constraint	Attributes	<<NOT NULL>>	
	Unique Constraint	Attributes	<<Unique>>	
	Trigger	Constraint	<<Trigger>>	
	CHECK Constraint	Constraint	<<Check>>	
	Stored Procedure	Class	<<Stored Procedure>>	
Physical	Tablespace	Component	<<Tablespace>>	
	Index	Class	<<Index>>	



Das Objekt-relationale Modell: SQL1999 and Oracle8i

F SQL1999

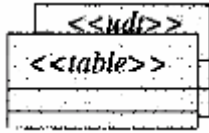
- Kein 1NF
- Daten type
 - Strukturierter Typ: Typed table & value type
 - OID (Objekt ID) & ref (Referenz)
 - ROW type
 - Ein kollektion type: ARRAY
- Vererbung unterstützung

F Oracle8i

- Kein 1NF
- Daten type
 - Strukturierter Typ: Typed table & value type
 - OID & ref
 - Kein ROW type
 - Zwei kinder von kollektionen: VARRAY und verschachtelte Tabelle
- Keine unterstützung von Vererbung (erst oracle9i)



UML-Erweiterung für Objekt- relationales Datenbank-Design

SQL:1999 Stereotypes	
<p>Structured Type Metamodel Klasse: Klasse Beschreibung: <<UDT>> erlaubt die Repräsentation von neuer Datentyp. Es entspricht dem Strukturierten Typ in SQL:1999</p> <p>Icon: Nein Zusicherung: kann nur für die Definition von <i>Value Type</i> verwendet werden Tagged values: Nein</p>	<p>Typed Table Metamodel Klasse: Klasse Beschreibung: definiert als <<Object Type>>. Es soll als eine Tabelle von einem Datentyp definiert werden.</p> <p>Icon: </p> <p>Zusicherung: impliziert die Definition einer Tabelle eines strukturierten Datentyp. Tagged values: Nein</p>

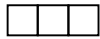



UML-Erweiterung für Objekt- relationales Datenbank-Design

SQL:1999 Stereotypes	
<p>Knows Metamodel Klasse: Assoziation Beschreibung: <<knows>> verbindet eine Klasse mit einem benutzerdefinierten Datentyp <<UDT>>. Eine uni-direktionale Relation. Die Richtung wird durch einen Pfeil angegeben und endet bei definiertem Datentyp. Icon: Nein Zusicherung: nur Verbindung einer Klasse mit <<UDT>> Type Tagged values: Nein</p>	<p>REF Type Metamodel Klasse: Attribute Beschreibung: repräsentiert ein Link zu einer <<Object Type>> Klasse Icon: ● → Zusicherung: Ein <<REF>> Attribut kann nur ein <<Object type>> referenzieren Tagged values: die <<Object type>> Klasse</p>



UML-Erweiterung für Objekt- relationales Datenbank-Design

SQL:1999 Stereotypes	
<p>Array Metamodel Klasse: Attribute Beschreibung: repräsentiert einen indizierte Kollektionstypen mit fester Länge.</p> <p>Icon: </p> <p>Zusicherung: beliebiger Typ außer <<Array>> Tagged values: der Basistyp von array die Länge der Elemente</p>	<p>Row Type Metamodel Klasse: Attribute Beschreibung: repräsentiert ein komponiertes Attribute mit fester Anzahl von Elementen. Die können verschiedene Datentypen haben.</p> <p>Icon: </p> <p>Zusicherung: keine Methode Tagged values: Der Name der Elemente und Datentypen</p>

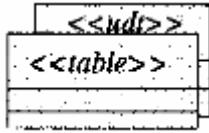


UML-Erweiterung für Objekt- relationales Datenbank-Design

SQL:1999 Stereotypes	
<p>Redefined Method Metamodel Klasse: Methode Beschreibung: <<redef>> ist eine vererbte Methode, die von der Unterklasse noch einmal implementiert wird.</p> <p>Icon: Nein Zusicherung: Nein Tagged values: Parameter, Type und Rückgabewert</p>	<p>Deferred method Metamodel Klasse: Methode Beschreibung: ein <<def>> Methode. Implementierung erste in der unterer Klasse.</p> <p>Icon: Nein Zusicherung: muss in obere Klasse definiert werden. Tagged values: Parameter, type und Rückgabewert</p>

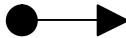


UML-Erweiterung für Objekt-relationales Datenbank-Design

Oracle8i Stereotypes	
<p>Object Type Metamodel Klasse: Klasse Beschreibung: <<UDT>> erlaubt die Repräsentation von neuer Datentyp. Es entspricht dem Strukturierten Typ in SQL:1999</p> <p>Icon: Nein Zusicherung: kann nur für die Definition von <i>Value Type</i> verwendet werden Tagged values: Nein</p>	<p>Object Table Metamodel Klasse: Klasse Beschreibung: definiert als <<Object Type>>. Es soll als eine Tabelle von einem Datentyp definiert werden.</p> <p>Icon: </p> <p>Zusicherung: impliziert die Definition einer Tabelle eines strukturierten Datentyp. Tagged values: Nein</p>

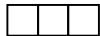
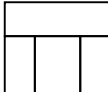


UML-Erweiterung für Objekt- relationales Datenbank-Design

Oracle8i Stereotypes	
<p>Knows Metamodel Klasse: Assoziation Beschreibung: verbindet eine Klasse mit einem benutzerdefinierten Datentyp <<UDT>>, <<Array>> oder <<NT>>. Ein uni-direktionale Relation. Die Richtung wird durch einen Pfeil angegeben und endet beim definierte Datentyp</p> <p>Icon: Nein Zusicherung: nur Verbindung eine Klasse mit <<UDT>> Type Tagged values: Nein</p>	<p>REF Type Beschreibung: repräsentiert ein Link zu <<Object Type>> Klasse</p> <p>Icon: </p> <p>Zusicherung: Ein <<REF>> Attribut kann nur ein <<Object type>> referenzieren Tagged values: die <<Object type>> Klasse</p>



UML-Erweiterung für Objekt- relationales Datenbank-Design

Oracle8i Stereotypes	
<p>VARRAY Metamodel Klasse: Attribute/Klasse Beschreibung: repräsentiert eine indizierte und fest lange Kollektion Type.</p> <p>Icon: </p> <p>Zusicherung: beliebiger Typ außer <<Array>> und <<NT>> Tagged values: der Basistyp von array die Länge der Elemente</p>	<p>Nested Table Metamodel Klasse: Klasse Beschreibung: <<NT>> repräsentiert einen nicht-indizierten Kollektionstyp ohne feste Länge</p> <p>Icon: </p> <p>Zusicherung: beliebige Type außer <<Array>> oder <<NT>> Tagged values: Der Basistyp der verschachtelten Tabelle</p>



Vorgehen

F Analyse

- mit UML, um das konzeptuelle Schema zu entwerfen

F Design

- Standard Design: unabhängig von Produkt (SQL:1999)
- Spezielles Design: design für spezielles Produkt (Z.B. Oracle8i)

F Implementierung

- Physikalisches Design aufstellen



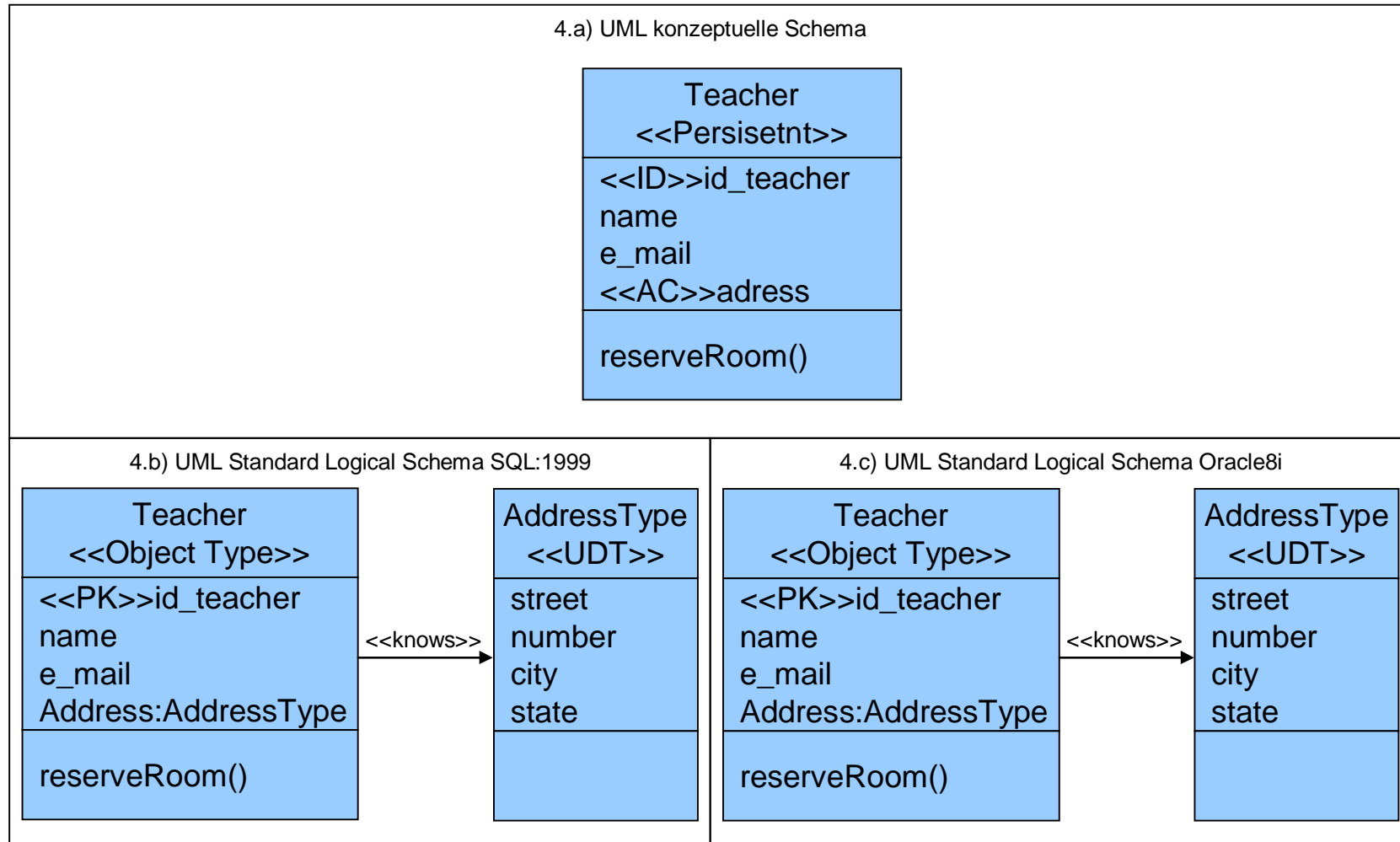
Transformationsrichtlinien

F Transformation für Klassen und Attributen

UML	SQL:1999	Oracle8i
Klasse	Structured Type	Object Type
Klasse Erweiterung	Typed Table	Table of Object Type
Attribute :		
Mehrwertige	ARRAY	VARRAY/nested table
komponierte	ROW/Structured Type	Object Type
abgeleitete	Trigger/Method	Trigger/Method



Beispiel für Klassentransformation





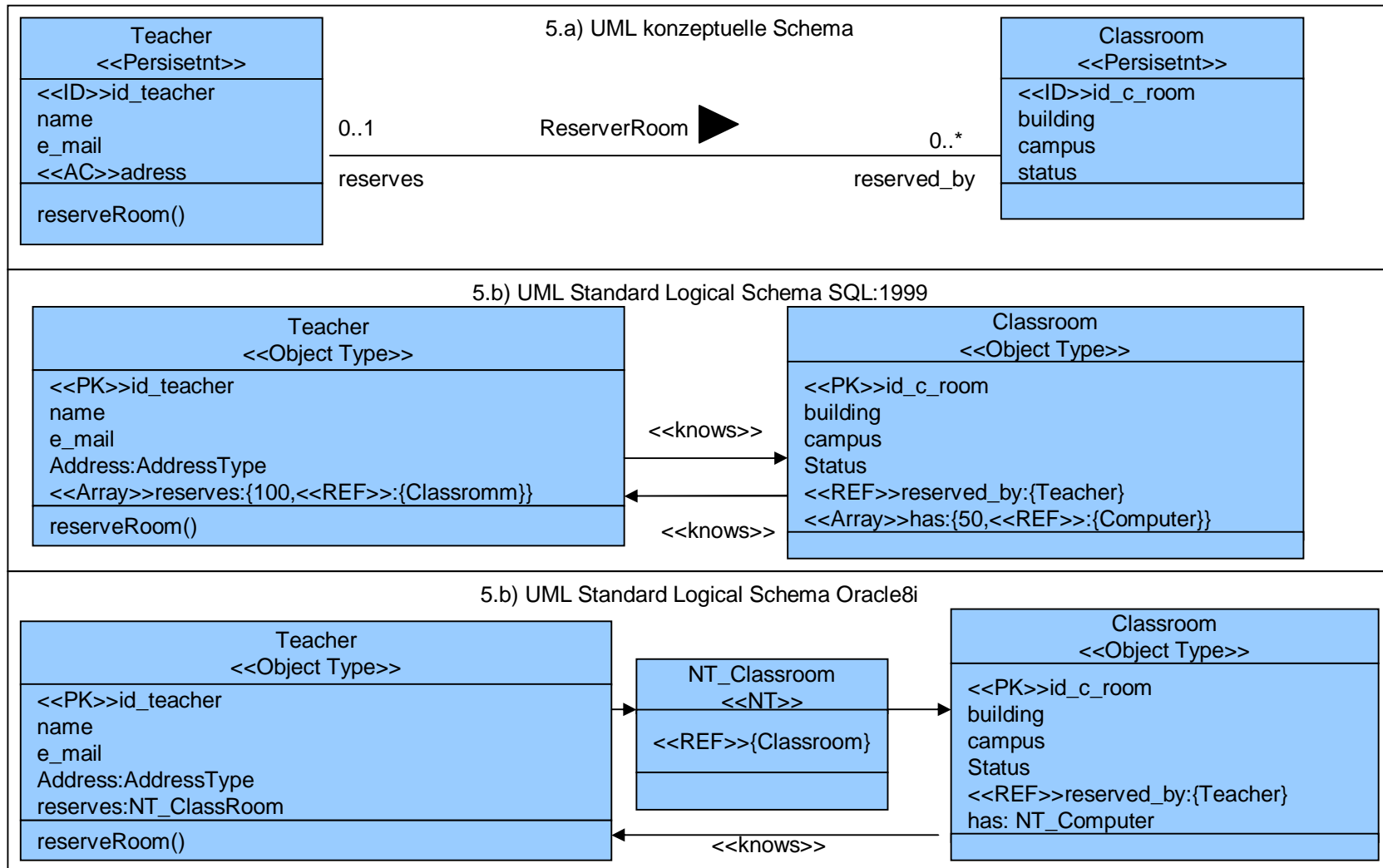
Transformationsrichtlinien

F Transformation für Assoziation (Bidirektional)

UML	SQL:1999	Oracle8i
eins-zu-eins	REF/REF	REF/REF
eins-zu-viel	REF/ARRAY	REF/VARRAY REF/verschachtelte Tabelle
viel-zu-viel	ARRAY/ARRAY	VARRAY/VARRAY verschachtelte Tabelle/verschaltelte Tabelle



Beispiel für Assoziationstransformation





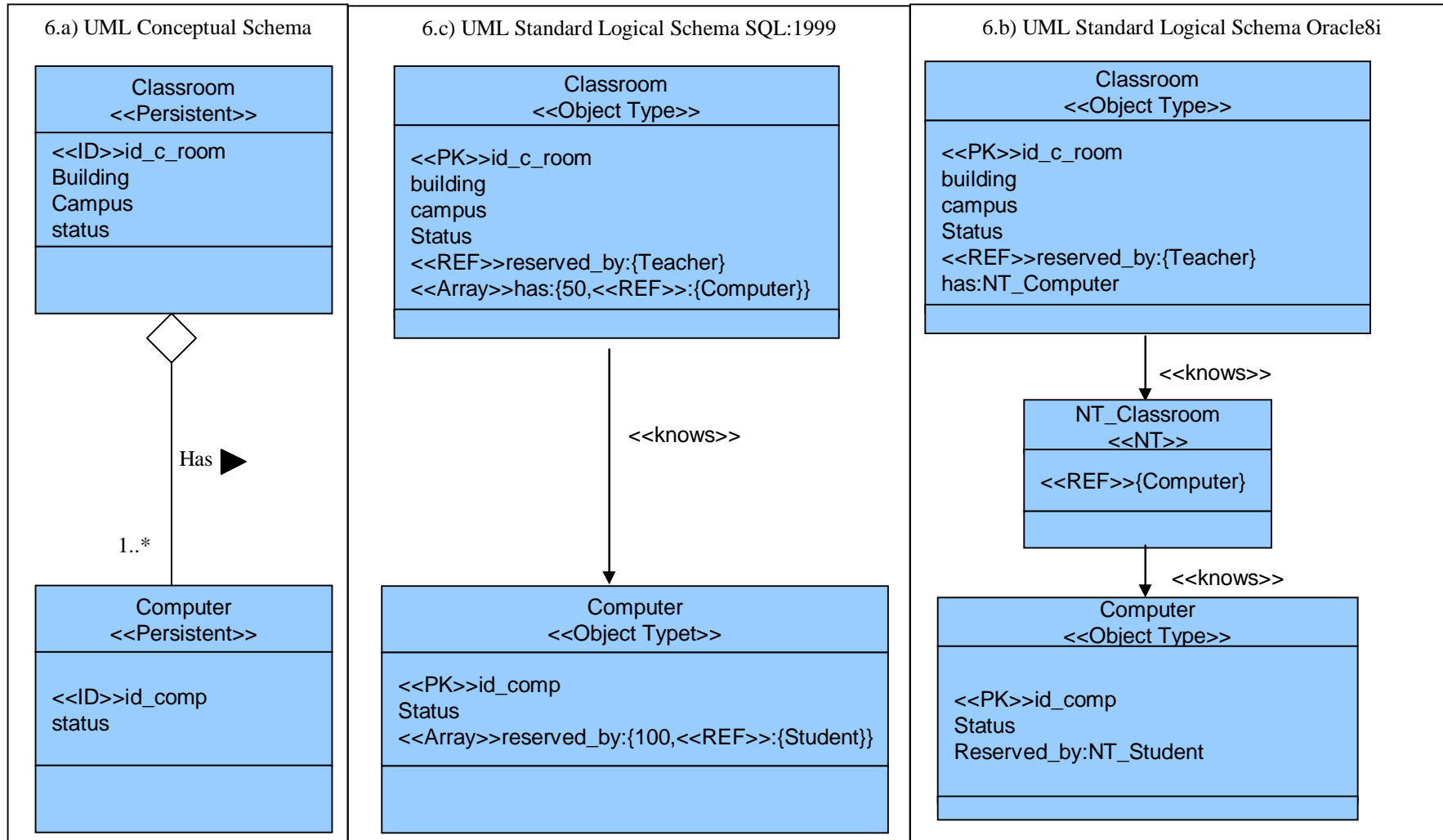
Transformationsrichtlinie

F Transformation für Aggregation

UML	SQL:1999	Oracle8i
Aggregation	ARRAY	verschachtelte Tabelle VARRAY von Referenzen
Komposition	ARRAY	verschachtelte Tabelle VARRAY von Objekten/Ref.



Beispiel für Aggregation Transformation





Zusammenfassung

- F** Objekt-relationale Datenbank ist zunehmend benutzt.
- F** Neues Vorgehen für objekt-relationale Datenbank design besteht aus drei Phasen.
- F** UML wegen der Erweiterbarkeit ist besonders geeignet für Datenbank-Design.



Literatur

- F A Methodological Approach for Object-Relational Database Design using UML , *Esperanza Marcos, Belen Vela, Jose Maria Cavero*
- F Objekt-orientierte Software-entwicklung
Analyse und Design mit der UML 2.0 , *Oldenbourg , Bernd Oestereich*
- F Objektorientierte Softwaretechnik
mit UML, Entwurfsmustern und Java , *Pearson , Bernd Brügge, Allen H.Dutoit*
- F UML Einführung <http://ivs.cs.uni-magdeburg.de/~dumke/UML/>
- F UML Bible, *Wiley Publishing , Tom Pender*



Vielen Dank
für Ihre Aufmerksamkeit!