Chapter 4: Relational Model

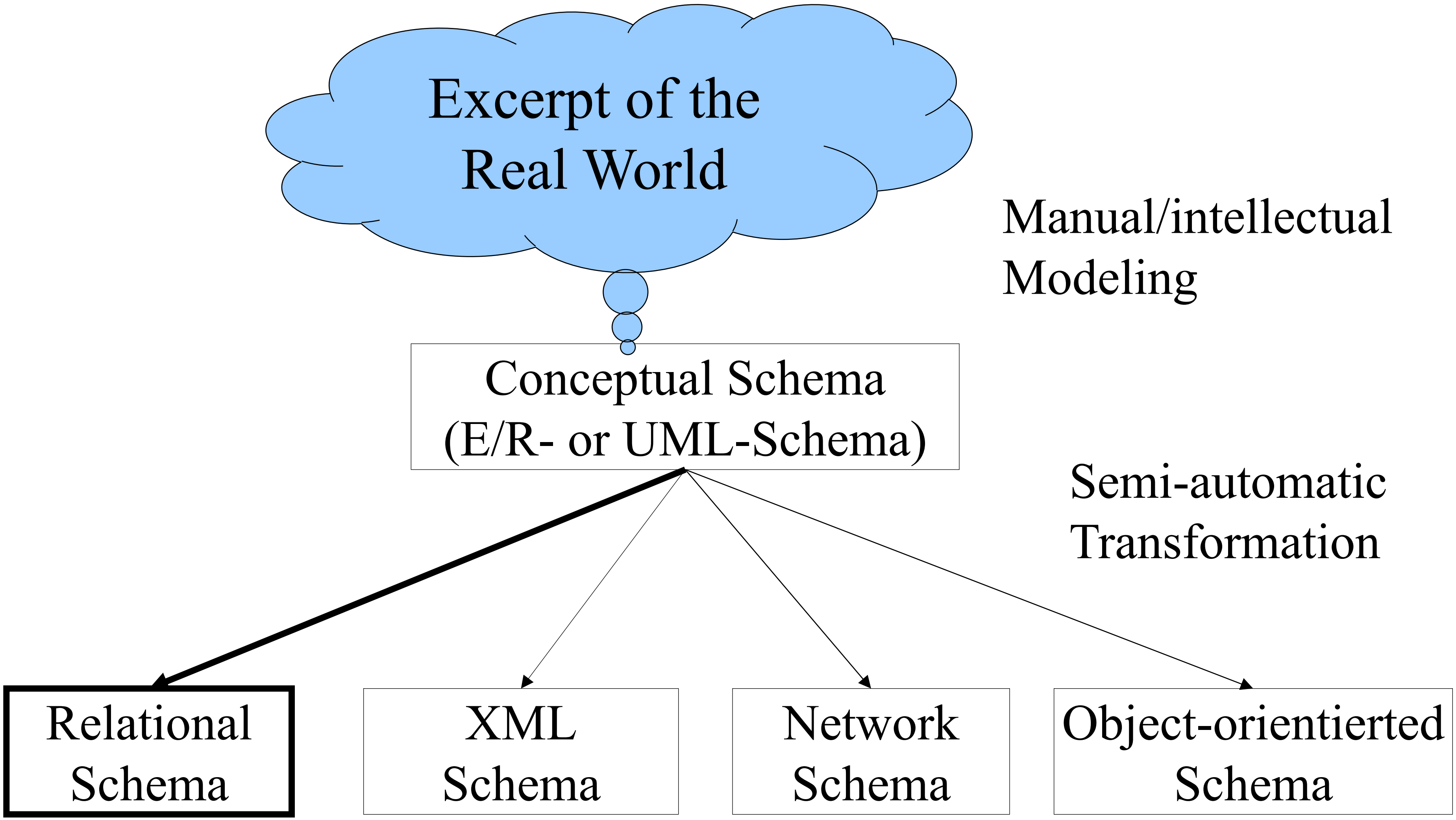
Content:

- Relational model
- How to transform ER diagrams into a relational model

Next:

- Transform relational model into database schema

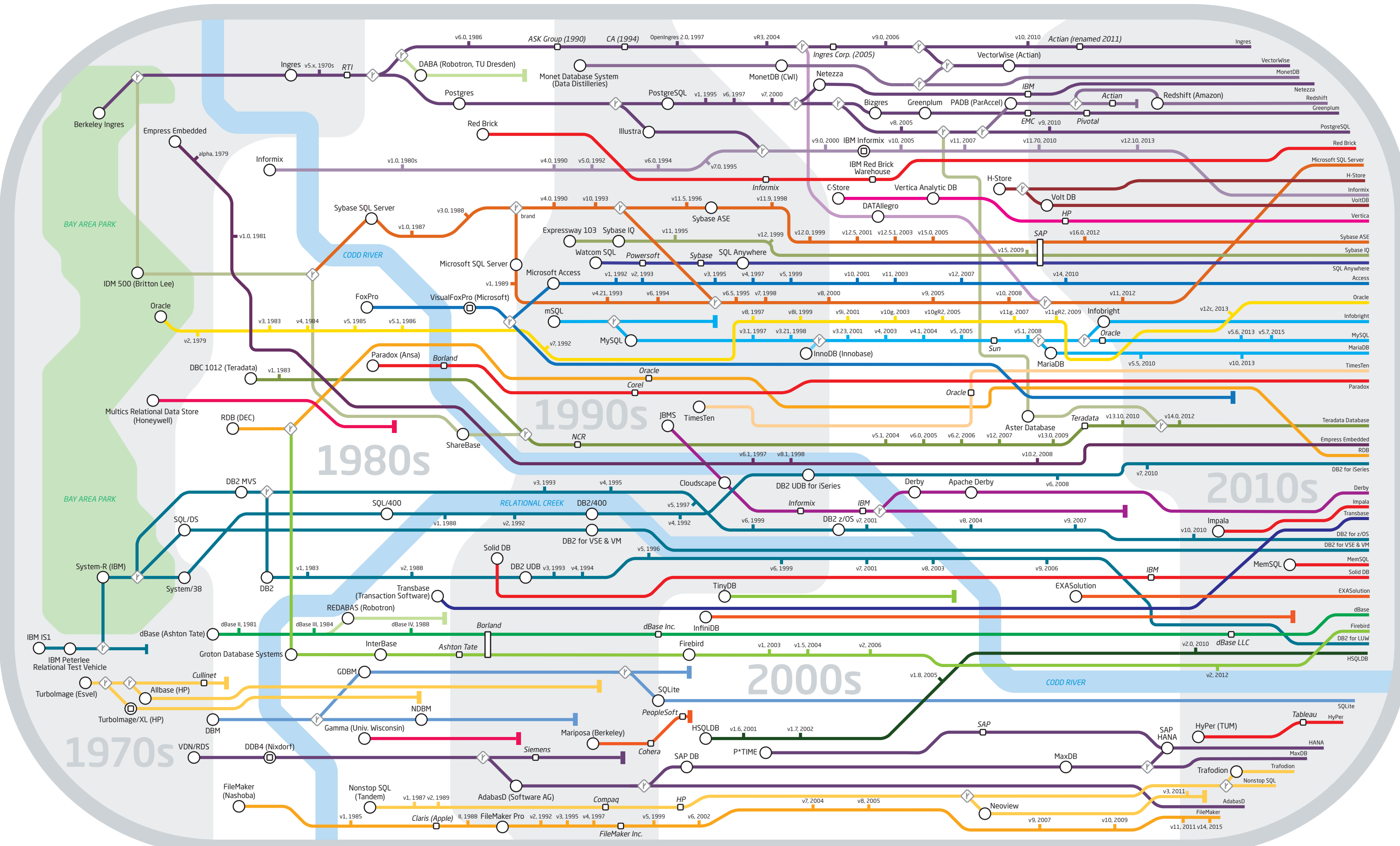
Data modeling



Development of relational DBMS

- Codasyl, beginning 1960: network data model
- IMS, mid 1960: hierarchical data model
- **Ted Codd, 1970: foundation relational data model**
- **System R, mid 1970: relational DBMS (research prototype)**
- Genealogy poster: www.hpi.de/naumann/projects/rdbms-genealogy.html

Genealogy of Relational Database Management Systems



Key to lines and symbols

DBMS name (Company) | *Acquisition* | Versions (v9, 2006) | Discontinued | Branch (intellectual and/or code) | Crossing lines have no special semantics

HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

Felix Naumann, Jana Bauckmann, Claudia Exeler, Jan-Peer Rudolph, Fabian Tschirschnitz
Contact - Hasso Plattner Institut, University of Potsdam, felix.naumann@hpi.de
Design - Alexander Sandt Grafik-Design, Hamburg

Version 6.0 - October 2018
https://hpi.de/naumann/projects/rdbms-genealogy.html

Development of relational DBMS

Commercial relational DBMS,
Focus OLTP

- Oracle V2, end 1970
- Ingres (Berkeley), end 1970 → PostgreSQL
- SQL/DS, beginning 1980: IBM → DB2
- MS SQL Server, 1990 (out of Sybase)
- MySQL, end 1990

→ since end 1990: Object relational extensions

OLTP vs. OLAP

OLTP (Online transaction processing)

- Mostly: short-running, write-heavy transactions
- Mission critical
- Example: Order in online warehouse

OLAP (Online analytical processing)

- Mostly: long-running, read-only transactions
- Decision support systems
- Example: Income losses caused by returns for products on sale in the last year.

Hybrid Systems

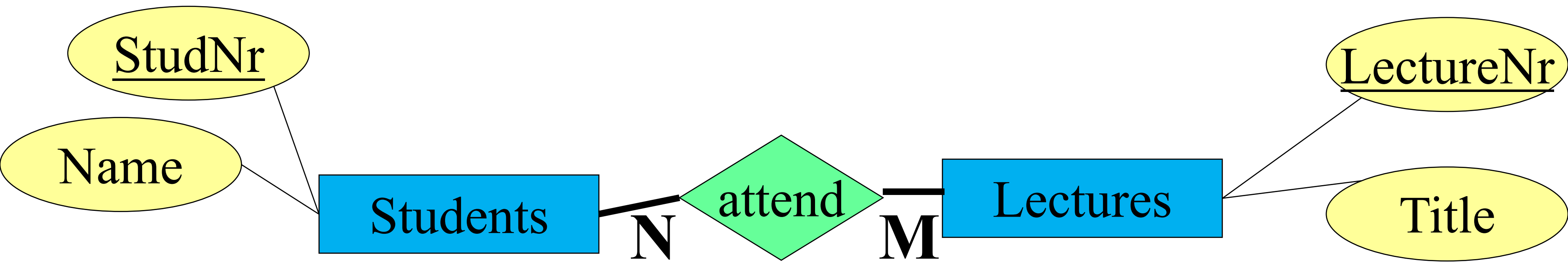
- Optimized for OLTP and OLAP
- Example: HyPer, Umbra, SAP HANA (main memory DBMS, multi processor architecture)

Development of relational DBMS

Database extensions:

- Object orientied DBMS (since end 1980)
- XML DBMS (since end 1990)
- Main memory DBMS (since end 1990)
- Analytical DBMS (Focus OLAP) (since beginning 2000)
- Next (2020+): massive amounts of SSDs? Persistent Memory?

Relational Data Model



Students	
StudNr	Name
26120	Fichte
25403	Jonas
...	...

attend	
StudNr	LectureNr
25403	5022
26120	5001
...	...

Lectures	
LectureNr	Title
5001	Grundzüge
5022	Glaube und Wissen
...	...

Example Relation

PhoneBook		
Name	**Street**	**Phone#**
Mickey Mouse	Main Street	4711
Minnie Mouse	Broadway	94725
Donald Duck	Broadway	95672
...	...	...

- **Instance:** current state of the relation or data base
- **Candidate key:** minimal set of attributes whose values uniquely identify a tuple
- **Primary key:** underlined
  - One of the candidate keys is chosen as primary key
  - Has a special meaning when referencing tuples

Foundation relational model

Let D_1, D_2, ..., D_n be **domains**

Relation: $R \subseteq D_1 \times \ldots \times D_n$

e.g.: Phone book $\subseteq$ *string x string x integer*

Tuple: $t \in R$

e.g.: *t* = („Mickey Mouse“, „Main Street“, 4711)

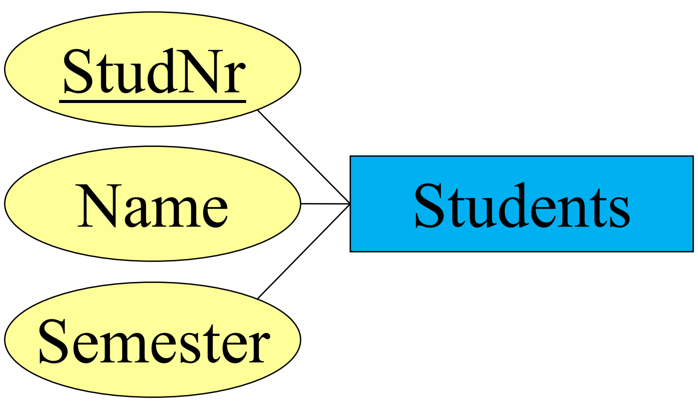
Schema: *determines the* structure of the stored data

e.g.: *Phone book:* {[*Name: string, Street: string, phone#: integer*]}

Transformation: Entities

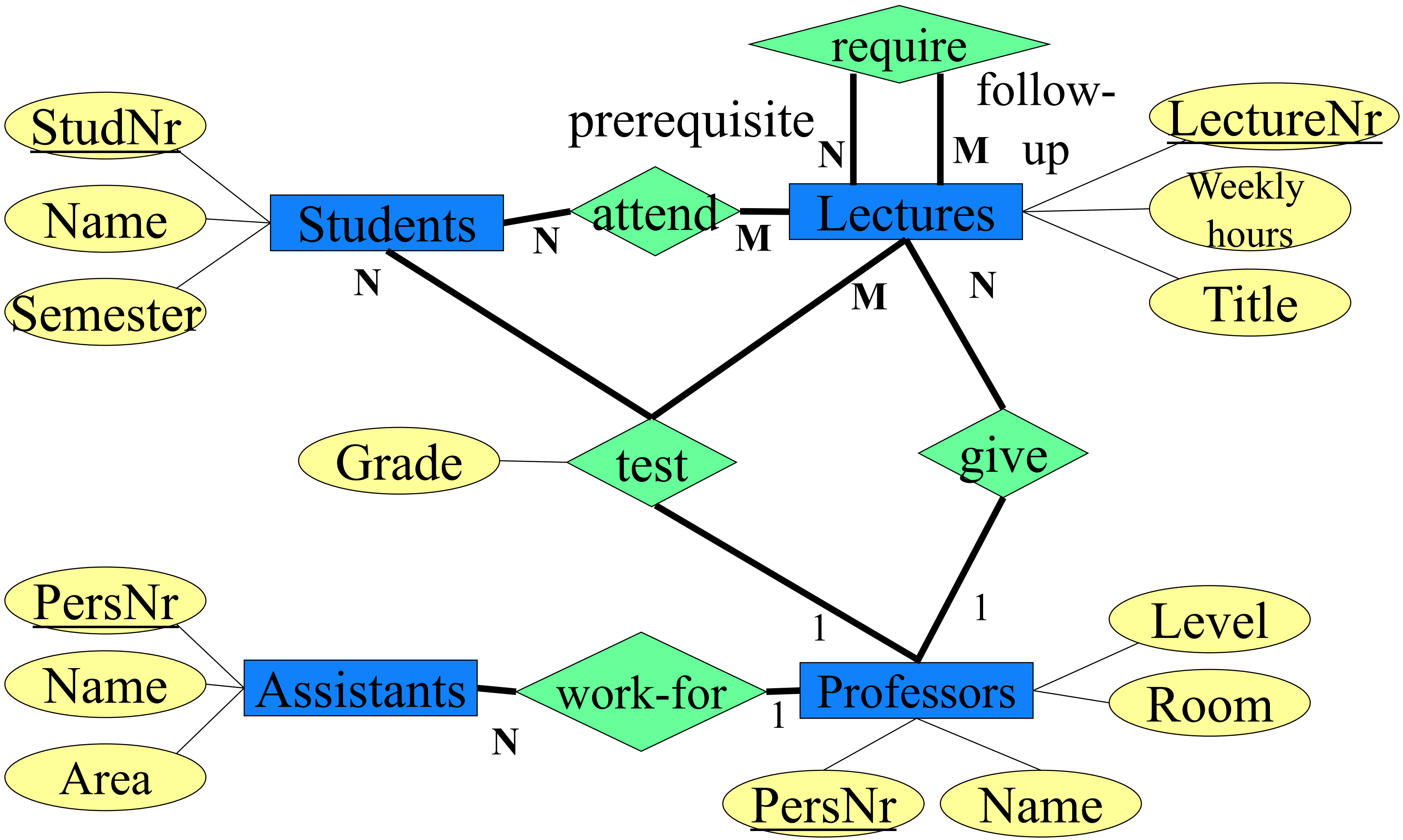
Entity sets to relations:

- Entity set E with attributes A_i out of domains D_i ($1 \leq i \leq k$)
  ⇨ k-ary relation $E(A_1{:}D_1, \ldots, A_k{:}D_k)$.
- Keep key attributes



⇨ **Students**: {[<u>StudNr: integer</u>,
Name: *string*,
Semester: *integer*]}

University Schema



Relational depiction of entity sets

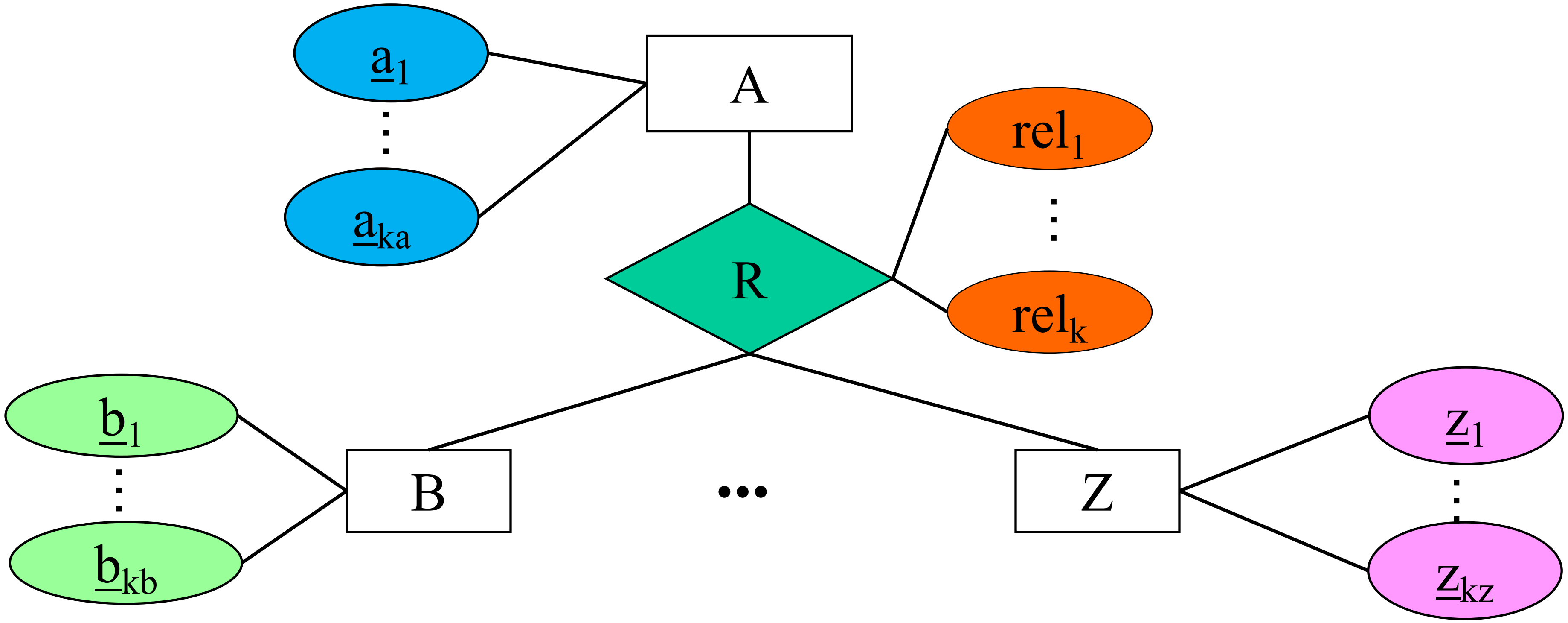
Students: {[StudNr:integer, *Name*: *string*, *Semester*: *integer*]}

Lectures: {[LectureNr:integer, *Title*: *string*, *WeeklyHours*: *integer*]}

Professors: {[PersNr:integer, *Name*: *string*, *Level*: *string*, *Room*: *integer*]}

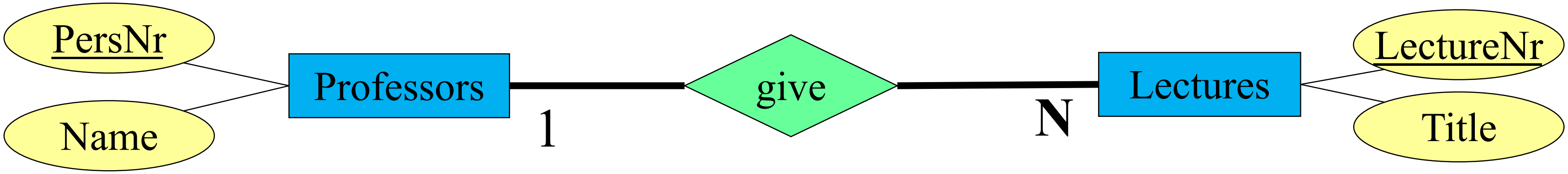
Assistants: {[PersNr:integer, *Name*: *string*, *Area*: *string*]}

Transformation: Relationships Attributes



$R: \{[\underbrace{a_1,\dots,a_{ka}}_{\text{Keys of A}}, \underbrace{b_1,\dots,b_{kb}}_{\text{Keys of B}}, \dots, \underbrace{z_1,\dots,z_{kz}}_{\text{Keys of Z}}, \underbrace{rel_1,\dots,rel_k}_{\text{Attributes of R}}]\}$

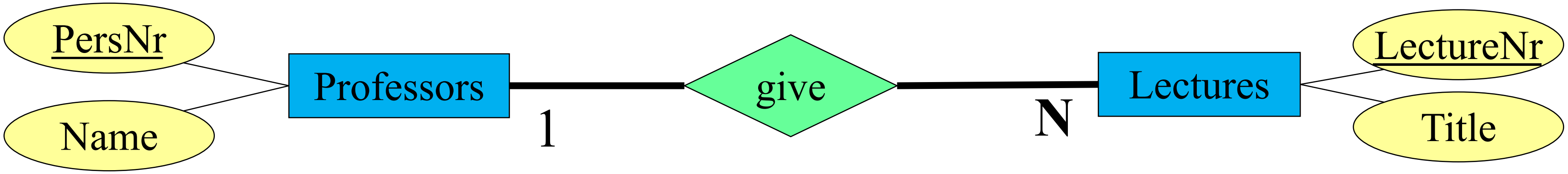
Foreign Keys



give (1:N): {[PersNr: integer, LectureNr: integer]}

- **Candidate key:** minimal set of attributes that uniquely identify a tuple
- **Primary key:** underlined
  - One of the candidate keys is chosen as primary key
  - Has a special meaning when referencing tuples
- **Foreign key:** set of attributes that refer to the primary key of another relation:
  - *PersNr* is a foreign key in *give* that references the *PersNr* in *Professors*
  - *LectureNr* is a foreign key in *give* that references the *LectureNr* in *Lectures*
  - *LectureNr* is also the primary key of *give*

1:N Relationship



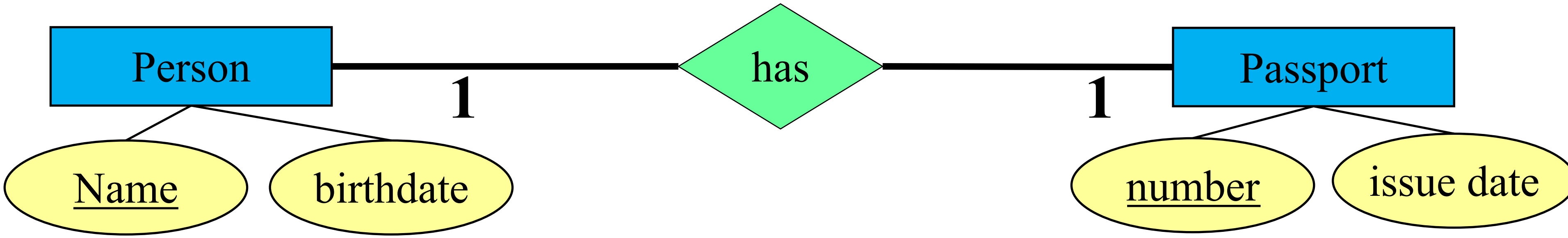
Professors		
PersNr	Name	Level
2125	Sokrates	C4
2133	Popper	C4
...	...	...

give	
PersNr	LectureNr
2133	5001
2125	5041
2125	4052
...	...

Lectures	
LectureNr	Title
5001	Databases
5041	Ethik
4052	Logik
...	...

give (1:N): {[lectureNr: int, persNr: int]}

1:1 Relationship



Person	
Name	birthdate
Sam	1970
Jack	1965
…	…

has	
Name	LectureNr
Sam	8128
Jack	512
…	…

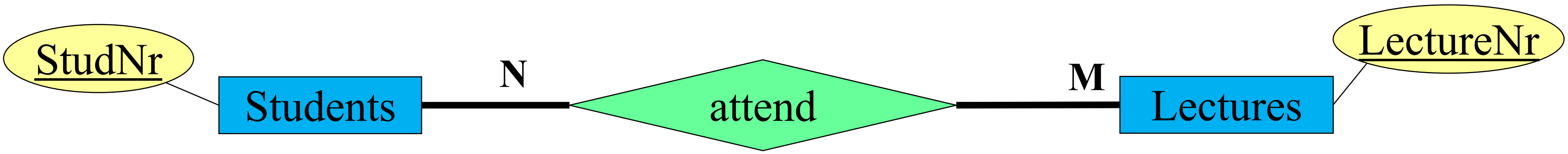
Lectures	
number	issue date
512	March 2018
8128	Jan 2020
…	…

has (1:1): {[name: string, number: int]}

or

has (1:1): {[name: string, number: int]}

N:M Relationship



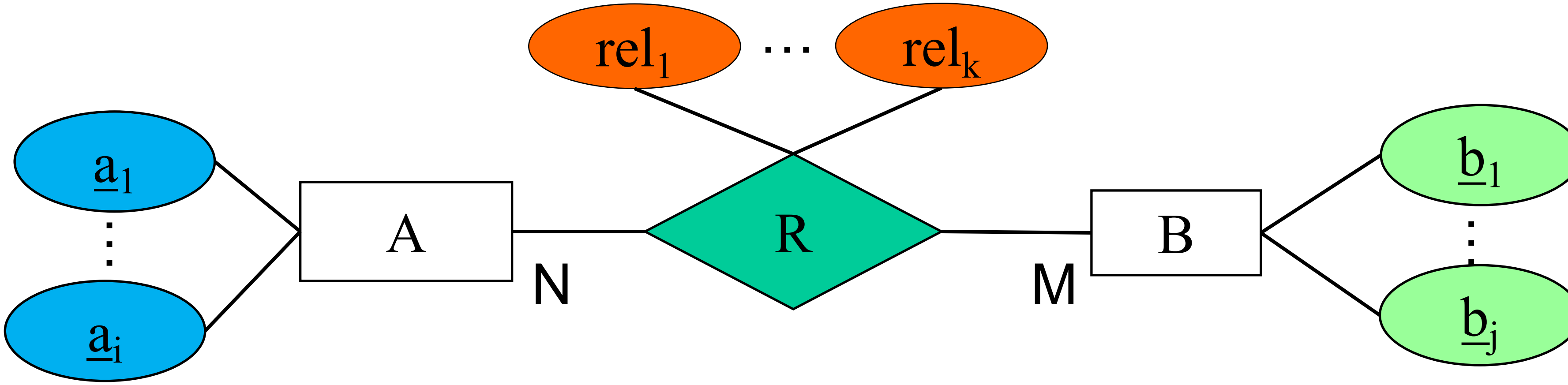
Students	
StudNr	...
26120	...
27550	...
...	...

attend	
StudNr	LectureNr
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
...	...

Lectures	
LectureNr	...
5001	...
4052	...
...	...

attend (N:M): {[studNr: int, lectureNr: int]}

Transformation: Relationships Keys



N:M $\Rightarrow$ R: $\{[\underline{a_1,\ldots,a_i,\ b_1,\ldots,b_j},\ rel_1,\ldots,rel_k]\}$

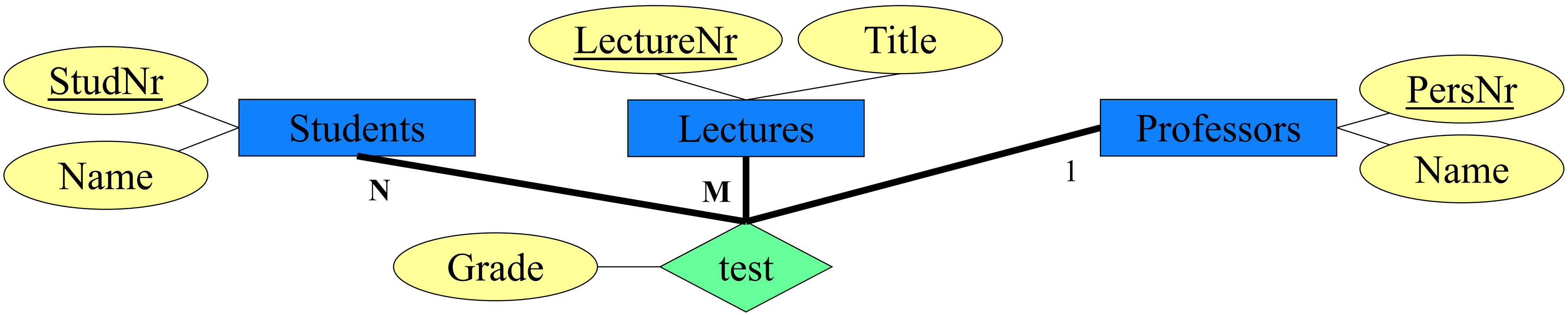
N:1 $\Rightarrow$ R: $\{[\underline{a_1,\ldots,a_i},\ b_1,\ldots,b_j,\ rel_1,\ldots,rel_k]\}$

1:M $\Rightarrow$ R: $\{[a_1,\ldots,a_i,\ \underline{b_1,\ldots,b_j},\ rel_1,\ldots,rel_k]\}$

1:1 $\Rightarrow$ R: $\{[\underline{a_1,\ldots,a_i},\ b_1,\ldots,b_j,\ rel_1,\ldots,rel_k]\}$

or $\Rightarrow$ R: $\{[a_1,\ldots,a_i,\ \underline{b_1,\ldots,b_j},\ rel_1,\ldots,rel_k]\}$

Ternary Relations



Students	
StudNr	Name
26120	Fichte
27550	Schopenhauer
...	...

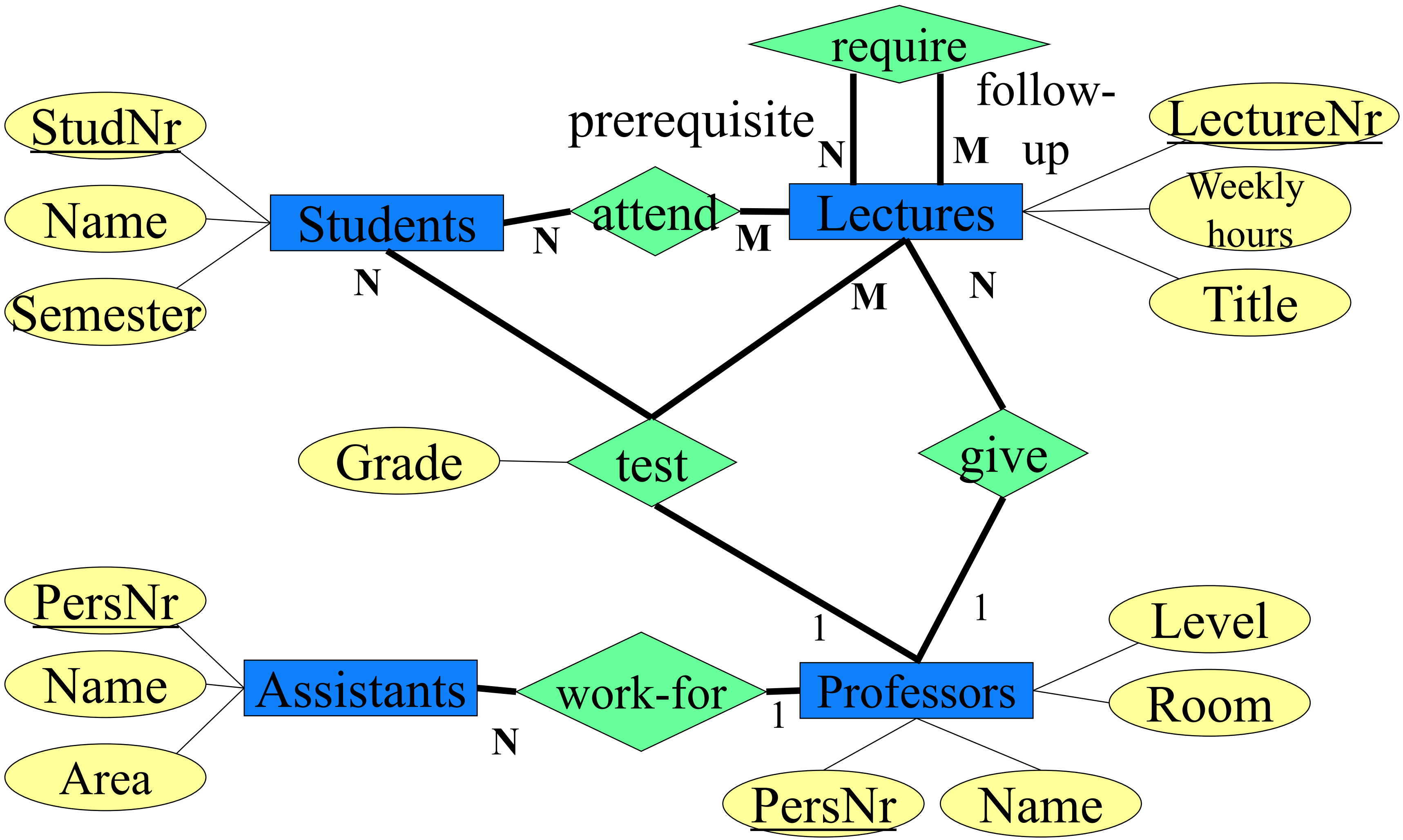
Lectures	
LectureNr	Title
5001	Grundzüge
4052	Logik
...	...

Professors	
PersNr	Name
2125	Sokrates
2133	Popper
...	...

test			
StudNr	LectureNr	PersNr	Grade
26120	5001	2125	1.0
26120	4052	2133	2.3
27550	5001	2133	1.7
...	...	...	...

test (N:M:1): {[studNr: int, lectureNr: int, persNr: int, grade: decimal]}

University Schema



Relationships of our example schema

attend (N:M): {[<u>StudNr: integer, LectureNr: integer</u>]}

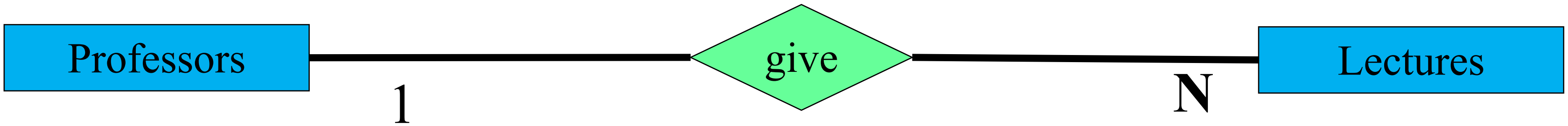
give (1:N): {[PersNr: integer, <u>LectureNr: integer</u>]}

work_for (N:1): {[<u>AssistantsPersNr: integer</u>,
ProfPersNr: integer]}

require (N:M): {[<u>Predecessor: integer, Successor: integer</u>]}

test (N:M:1*):* {[<u>StudNr: integer, LectureNr: integer</u>,
PersNr: integer, Grade: decimal]}

Refined relational schema



1:N-Relationship (simple):

Professors : {[*PersNr*, *Name*, *Level*, *Room*]}

Lectures : {[*LectureNr*, *Title*, *WeeklyHours*]}

give: {[*PersNr* , *LectureNr*]}

Refinement by subsumption:

Professors : {[*PersNr*, *Name*, *Level*, *Room*]}

Lectures : {[*LectureNr*, *Title*, *WeeklyHours*, ***given_by***]}

Rule:

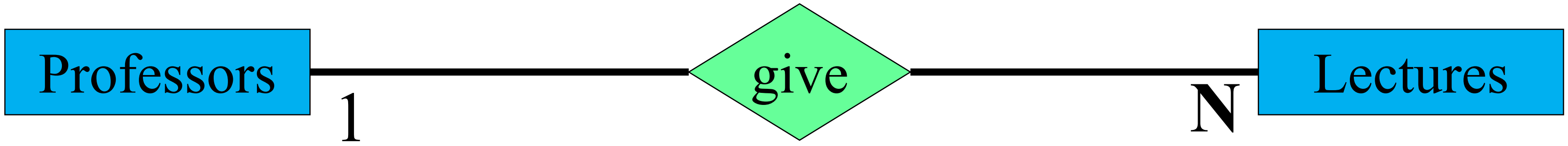
Binary relations (**not** entities) with the same primary key as an entity can be subsumed

... but only those and no others!

Instance of *Professors* and Lectures

Professors			
PersNr	Name	Level	Room
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Lectures			
Lecture Nr	Title	Weekly Hours	given_by
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
...	...	...	...



Why not the other way round?

Professors				
PersNr	Name	Level	Room	gives
2125	Sokrates	C4	226	5041
2125	Sokrates	C4	226	5049
2125	Sokrates	C4	226	4052
...	...	...	...	...
2134	Augustinus	C3	309	5022
2136	Curie	C4	36	??

Lectures		
Lecture Nr	Title	Weekly Hours
5001	Grundzüge	4
5041	Ethik	4
5043	Erkenntnistheorie	3
5049	Mäeutik	2
4052	Logik	4
5052	Wissenschaftstheorie	3
...	...	...

Consequences→Anomalies

Professors				
PersNr	Name	Level	Room	gives
2125	Sokrates	C4	226	5041
2125	Sokrates	C4	226	5049
2125	Sokrates	C4	226	4052
...	...	...	...	...
2134	Augustinus	C3	309	5022
2136	Curie	C4	36	??

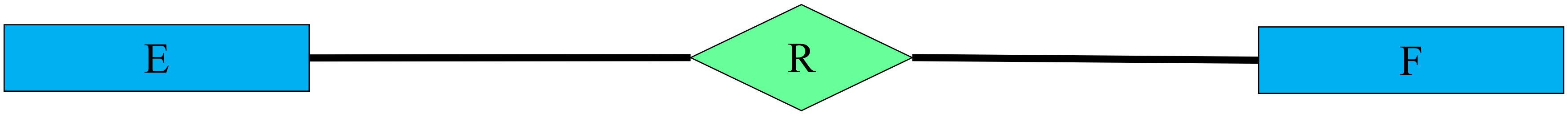
Lectures		
Lecture Nr	Title	Weekly Hours
5001	Grundzüge	4
5041	Ethik	4
5043	Erkenntnistheorie	3
5049	Mäeutik	2
4052	Logik	4
5052	Wissenschaftstheorie	3
...	...	...

Update anomaly: What if Sokrates moves?

Delete anomaly: What if „Glaube und Wissen“ is dropped?

Insert anomaly: Curie is new and does not yet give lectures?

Refined transformation rules



Relationship Refinement (cont‘d):

1:1-Relationship:

Relationship R between 2 entities E and F.
⇨ no Relation R, instead primary key of E in Relation in F or vice versa.

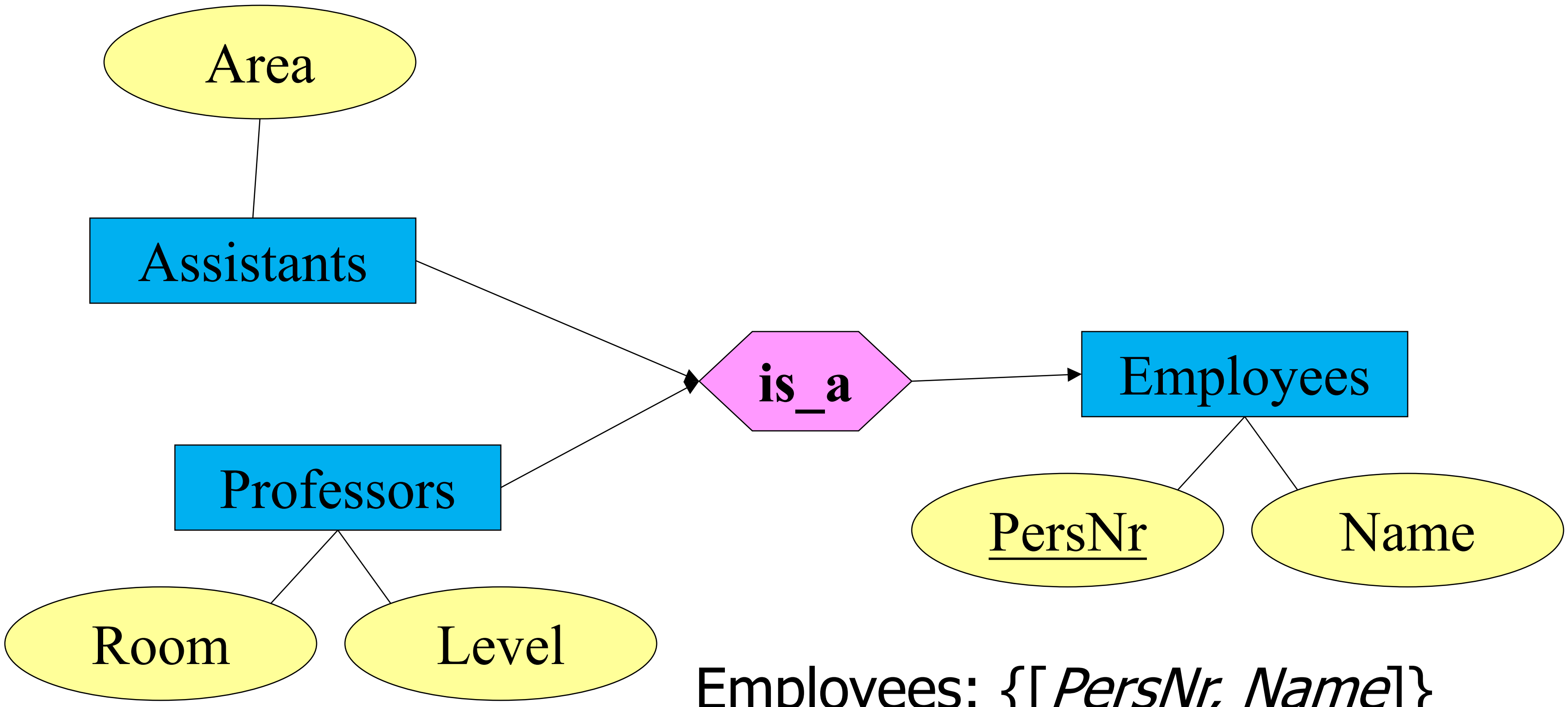
1:N-Relationship:

Relationship R between 2 entities E and F.
⇨ no Relation R, instead primary key of E in relation F as foreign key.
In case R has own attributes incorporate them into F.

Any other Relationship or Entity:

No refinement !!!

Relational Modelling of the Generalization

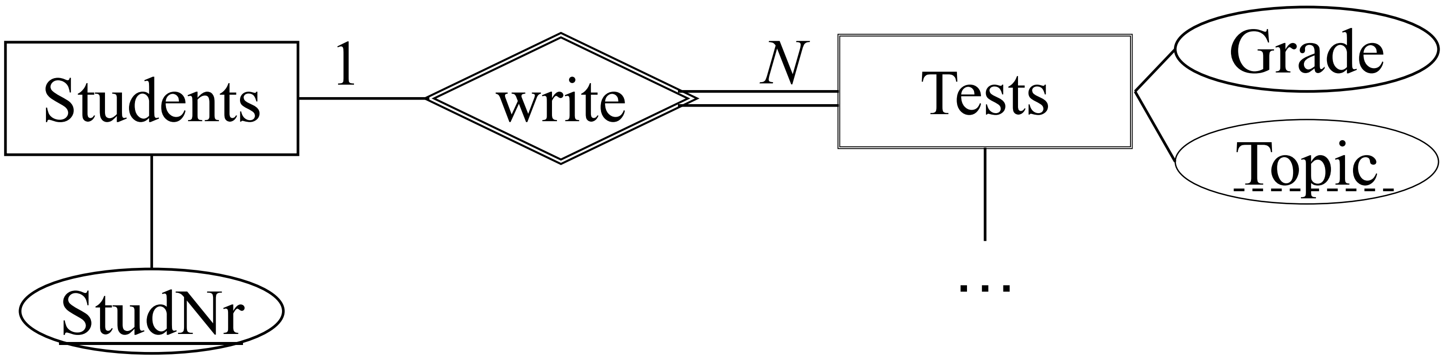


Employees: {[*PersNr*, *Name*]}

Professors: {[*PersNr*, *Level*, *Room*]}

Assistants: {[*PersNr*, *Area*]}

Relational Modelling of weak Entities



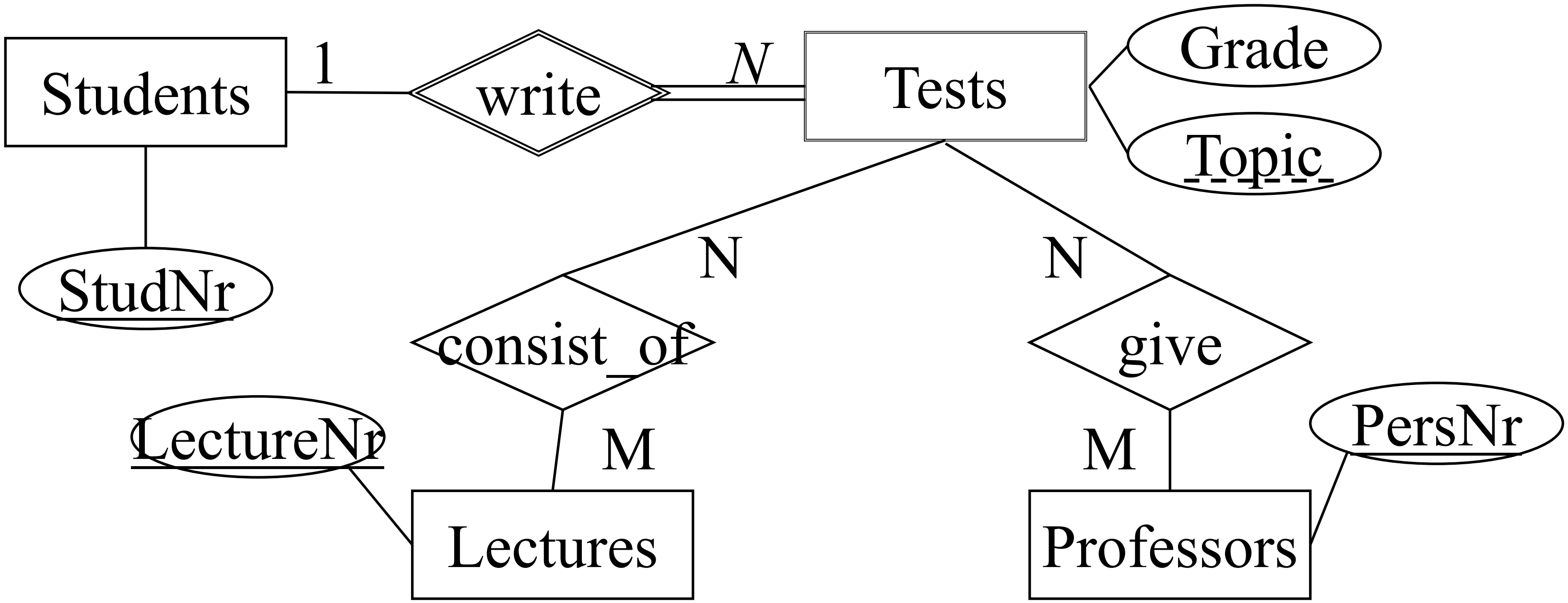
Entity sets Students, Tests:

Students: {[StudNr: integer, …]}

Tests: {[StudNr: integer, Topic: string, Grade: integer]}

The relation between a weak and strong entity can always be refined

Relational Modelling of weak Entities



consist_of and ***give*:**

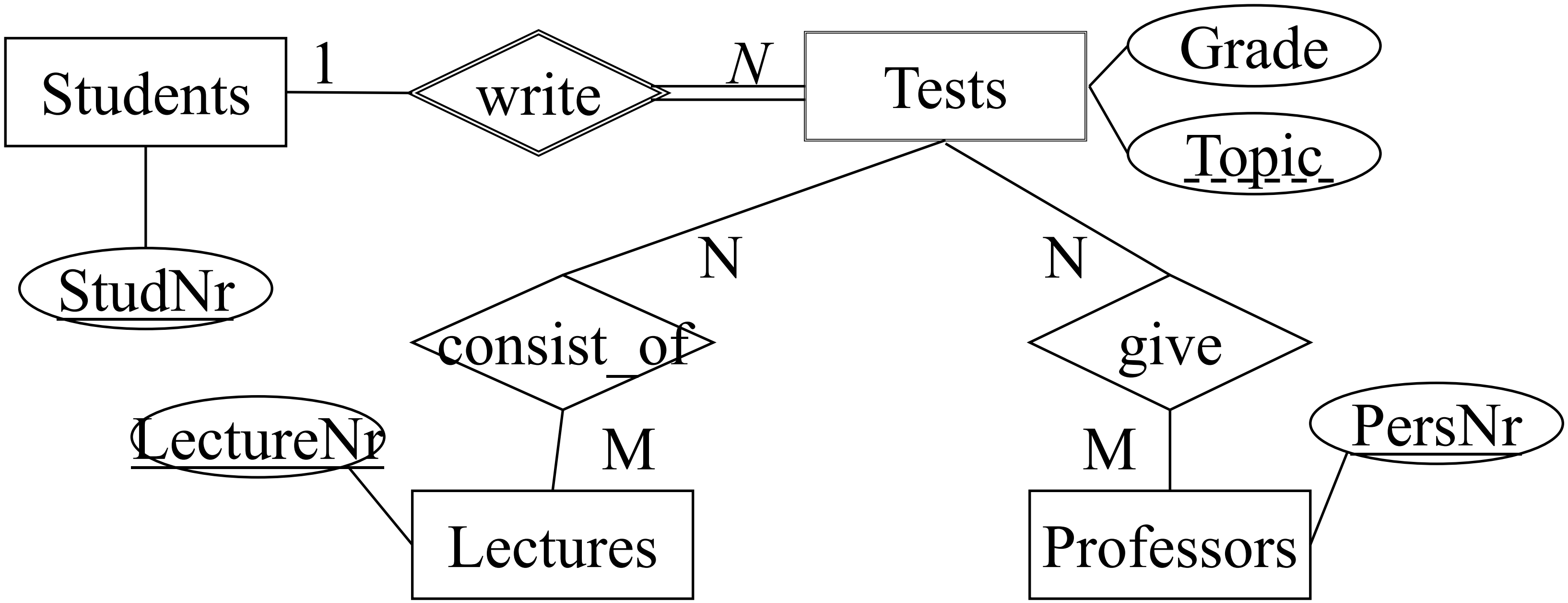
Need primary key of the relation **Tests:**

-> **studNr** and **topic**

Must be used as a foreign key in the relations:

-> **consist_of** and **give**

Relational Modelling of weak Entities



consist_of: {[StudNr: integer, Part: string, LectureNr: integer]}

give: {[StudNr: integer, Part: string, PersNr: integer]}

Professors

PersNr	Name	Level	Room
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Students

StudNr	Name	Semester
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
26830	Aristoxenos	8
27550	Schopenhauer	6
28106	Carnap	3
29120	Theophrastos	2
29555	Feuerbach	2

Lectures

Lecture Nr	Title	Weekly Hours	Given_by
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

attend

StudNr	LectureNr
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
25403	5022
29555	5022
29555	5001

require

Predecessor	Successor
5001	5041
5001	5043
5001	5049
5041	5216
5043	5052
5041	5052
5052	5259

Assistants

PersNr	Name	Area	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenbewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

test

StudNr	LectureNr	PersNr	Grade
28106	5001	2126	1
25403	5041	2125	2
27550	4630	2137	2