SCIENCE
PASSION
TECHNOLOGY

Data Management
02 Conceptual Design

Matthias Boehm

Graz University of Technology, Austria
Computer Science and Biomedical Engineering
Institute of Interactive Systems and Data Science
BMK endowed chair for Data Management

Last update: Mar 11, 2022

ISDS

Announcements/Org

- **#1 Video Recording**
  - Link in **TeachCenter** & **TUbe** (lectures will be public)
  - Hybrid: HSi13 / https://tugraz.webex.com/meet/m.boehm
- **#2 Course Registrations SS22**
  - Data Management (lectures/exercises): **471 (4)**
  - Databases (combined lectures/exercises): **70 (0)**

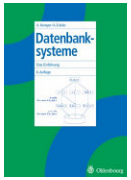
Total: **541**

- **#3 Exercise 1 Published**
  - Task description published last Tuesday (discussed today)
  - **Deadline: Mar 29, 11.59pm** in TeachCenter
- **#4 SIGMOD Programming Context 2022 (Apr 30)**
  - **Task: entity resolution blocking** (recall, runtime limit)
  - http://sigmod2022contest.eastus.cloudapp.azure.com/index.shtml
  - **Organized by:** Georgia Tech / University of Modena → Awards: XXX USD (MS)

Extra-curricular Activity

Agenda

- **DB Design Lifecycle**
- **ER Model and Diagrams**
- **Exercise 01 – Data Modeling**

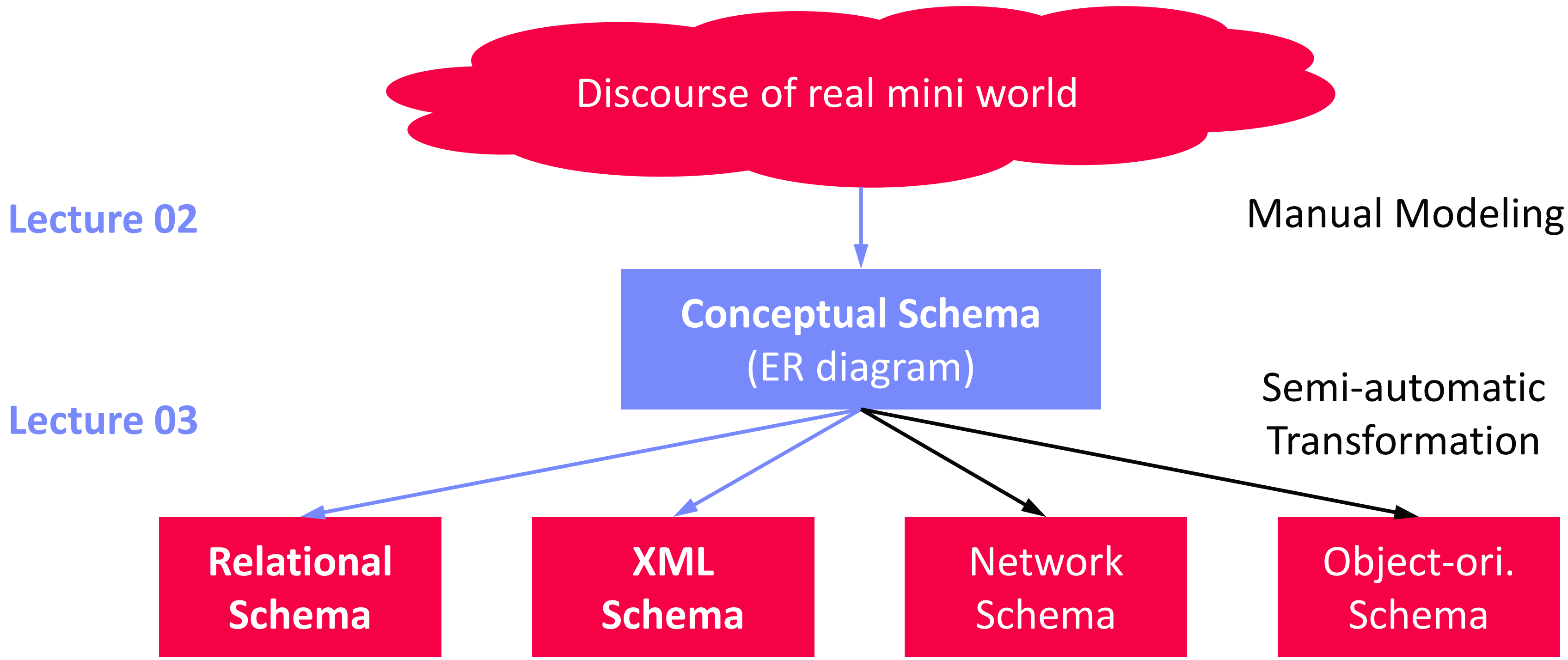


[**Credit:** Alfons Kemper, André Eickler: Datenbanksysteme - Eine Einführung, 10. Auflage. De Gruyter Studium, de Gruyter Oldenbourg 2015, ISBN 978-3-11-044375-2, pp. 1-879]

DB Design Lifecycle

Data Modeling

- **Data Model**
  - Concepts for describing data objects and their relationships (meta model)
  - **Schema:** Description (structure, semantics) of specific data collection



Data Models

- **Conceptual Data Models**
  - **Entity-Relationship Model (ERM)**, focus on data, ~1975
  - Unified Modeling Language (UML), focus on data and behavior, ~1990

- **Logical Data Models**
  - **Relational** (Object/Relational)

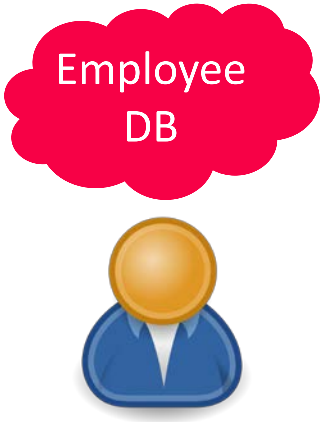
  - Key-Value
  - Document (XML, JSON)
  - Graph
  - Time Series
  - Matrix/Tensor

  Partly covered in part B

  - Object-oriented
  - Network
  - Hierarchical

  Mostly obsolete

DB Design Lifecycle Phases



- **#1 Requirements engineering**
  - Collect and analyze data and application requirements

  ➔ **Specification documents**
- **#2 Conceptual Design** (lecture 02, exercise 1)
  - Model data semantics and structure, independent of logical data model

  ➔ **ER model / diagram**
- **#3 Logical Design** (lecture 03, exercise 1)
  - Model data with implementation primitives of concrete data model

  ➔ **e.g., relational schema** + integrity constraints, views, permissions, etc
- **#4 Physical Design** (lecture 07, exercise 3)
  - Model **user-level data organization** in a specific DBMS (and data model)
  - Account for deployment environment and performance requirements

Relevance in Practice

- **Analogy ERM-UML**
  - **Model-driven development** (self-documenting, but quickly outdated)
  - **But:** Once data is loaded, data model and schema harder to change

- **Observation: Full-fledged ER modeling rarely used in practice**
  - Often the logical schema (relational schema) is directly created, maintained and used for documentation
  - **Reasons:** redundancy, indirection, single target (relational)
  - Simplified ER modeling used for brainstorming and early ideas

- **Goals**
  - **Understanding of proper database design** from conceptual to physical schema
  - ER modeling as a helpful **tool in database design**
  - Schema transformation and normalization as blueprint for **good designs**

Tool Support

- **#1 Visual Design Tools**
  - Draw ER diagrams in any presentation software (e.g., MS PowerPoint, LibreOffice)
  - Many desktop or web-based tools support ER diagrams directly (e.g., MS Visio, creately.com)

- **#2 Design Tools w/ Code Generation**
  - Draw and validate ER diagrams
  - Generate relational schemas as SQL DDL scripts
  - **Examples:** SAP (Sybase) PowerDesigner, MS Visual Studio plugins (SQL server), etc.

→ **Note: For the exercises, please use basic drawing tools**
(existing tools use slightly diverging notations)

Entity-Relationship (ER) Model and Diagrams

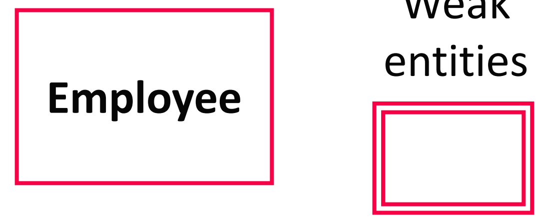
[Peter P. Chen: The Entity-Relationship Model - Toward a Unified View of Data. **ACM Trans. Database Syst. 1(1) 1976**]

[Peter P. Chen: The Entity-Relationship Model: Toward a Unified View of Data. **VLDB 1975**]

ER Diagram Components (Chen Notation)

- **Entity Type** (noun)
  - Entities are objects of the real world
  - An entity type (or **entity set**) represents a collection of entities

Weak entities

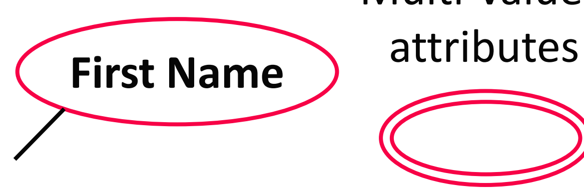


- **Relationship Type** (verb)
  - Relationships are concrete associations of entities
  - Relationship type (or **relationship set**) or relationship of entity types

$works \subseteq A \times B$

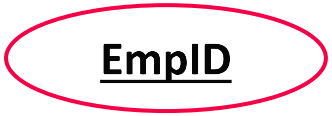
- **Attribute**
  - Entities or relationships are characterized by attribute-value pairs
  - Attribute types (or value sets) describe entity and relationship types
  - Extended attributes: composite, multi-valued, derived

Multi-valued attributes

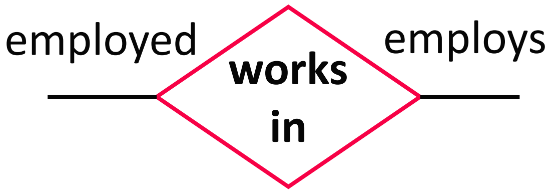


ER Diagram Components (Chen Notation), cont.

- **Keys**
  - Attributes that uniquely identify an entity
  - Every entity type must have such a key
  - Natural or surrogate (artificial) keys

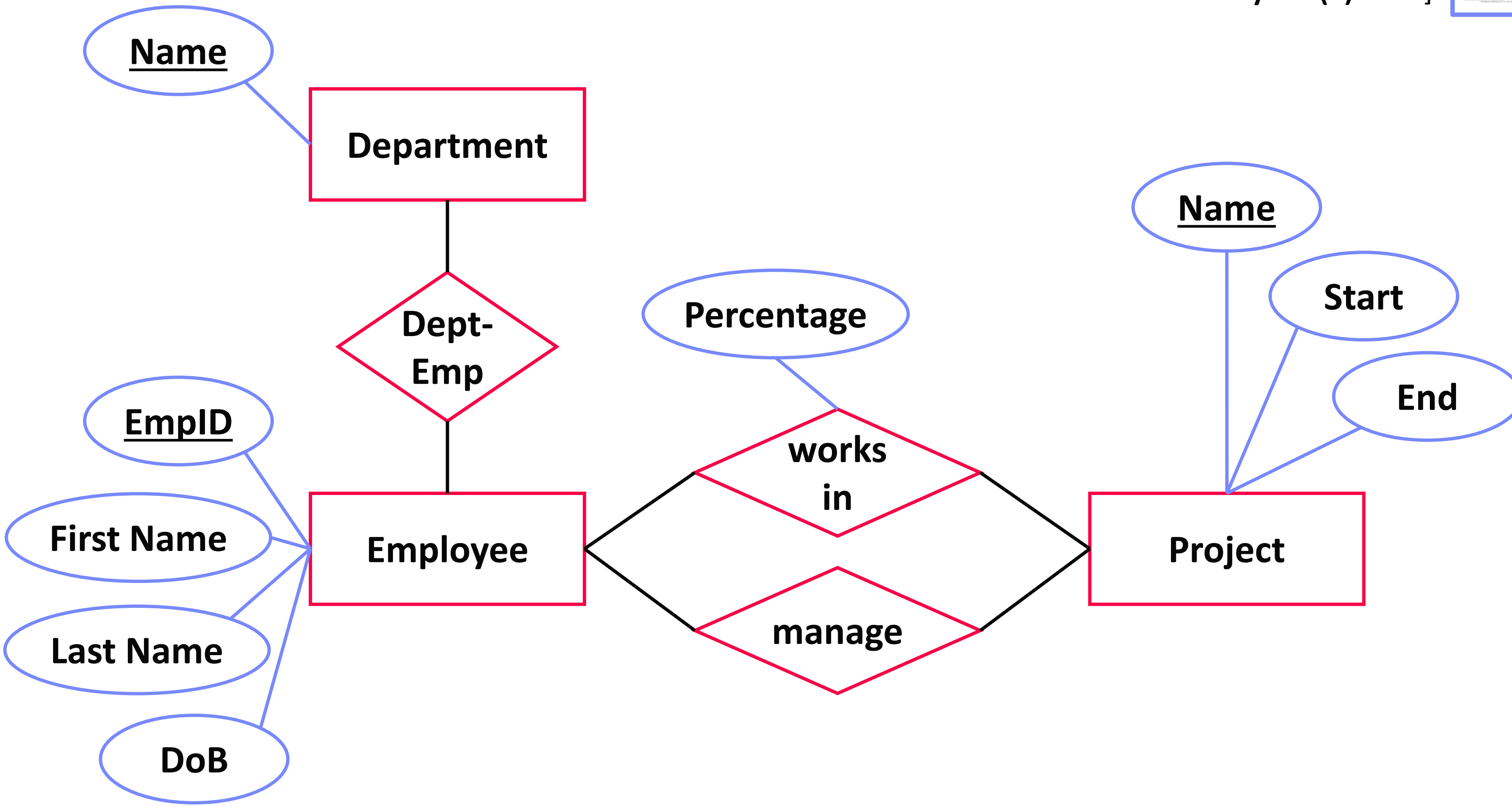


- **Role**
  - Optional description of relationship types
  - Useful for recursive relationships



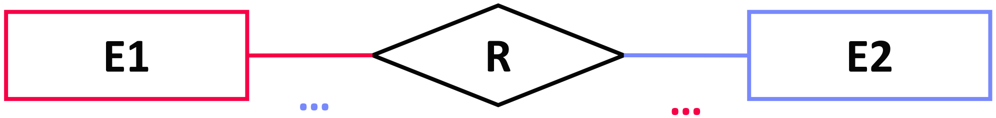
An EmployeeDB Example

[Peter P. Chen: The Entity-Relationship Model - Toward a Unified View of Data. **ACM Trans. Database Syst. 1(1) 1976**]



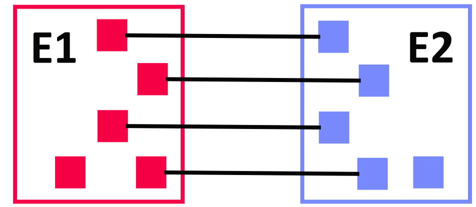
Multiplicity/Cardinality in Chen Notation

1 .. [0,1]
N ... [0,1,N]

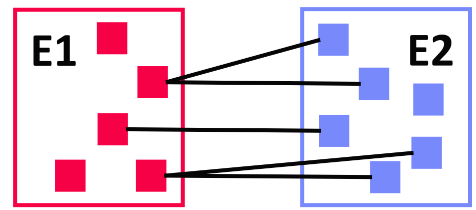


$R \subseteq E1 \times E2$

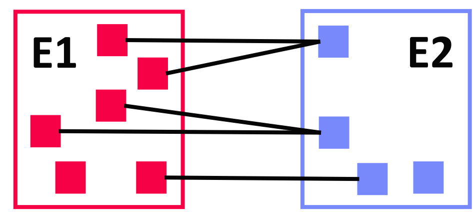
- **1:1 (one-to-one)**
  - Each e1 relates to at most one e2
  - Each e2 relates to at most one e1



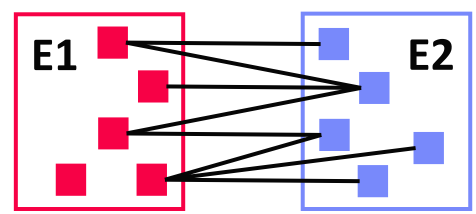
- **1:N (one-to-many)**
  - Each e1 relates to many e2 (0,1,…N)
  - Each e2 relates to at most one e1



- **N:1 (many-to-one)**
  - Symmetric to 1:N



- **N:M (many-to-many)**
  - Each e1 relates to many e2 (0,1,…M)
  - Each e2 related to many e1 (0,1,…N)



An EmployeeDB Example, cont.

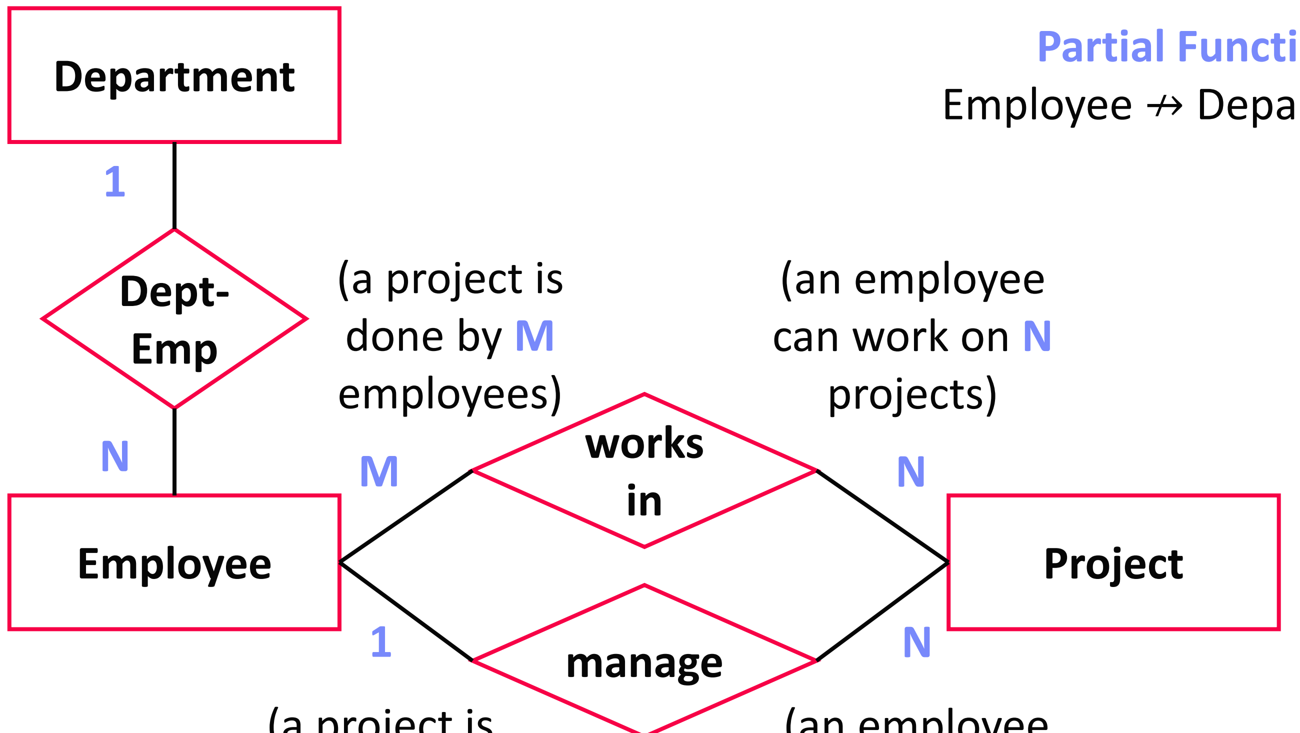
[Peter P. Chen: The Entity-Relationship Model - Toward a Unified View of Data. **ACM Trans. Database Syst. 1(1) 1976**]

Partial Function

Employee ↛ Department

(an employee belongs to 1 department)

(a department contains N employees)



(a project is done by M employees)

(an employee can work on N projects)

(a project is managed by 1 employee)

(an employee can manage N projects)

Multiplicity in Modified Chen Notation

- **Extension:** C ("choice"/"can") to model 0 or 1, while 1 means exactly 1 and M means at least 1.

4 alternatives (1, C, M, MC)
→ 4*4 = 16 combinations
(symmetric combinations omitted)

$$\boxed{\begin{matrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{matrix}} \qquad \frac{n \cdot (n+1)}{2}$$

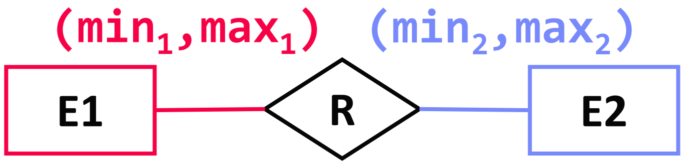
- **1:1** – [1] to [1]
- **1:C** – [1] to [0 or 1]
- **1:M** – [1] to [at least 1]
- **1:MC** – [1] to [arbitrary many]

- **C:C** – [0 or 1] to [0 or 1] → **see 1:1 in Chen**
- **C:M** – [0 or 1] to [at least 1]
- **C:MC** – [0 or 1] to [arbitrary many] → **see 1:N in Chen**

- **M:M** – [at least 1] to [at least 1]
- **M:MC** – [at least 1] to [arbitrary many]

- **MC:MC** – [arbitrary many] to [arbitrary many] → **see M:N in Chen**

(min,max)-Notation

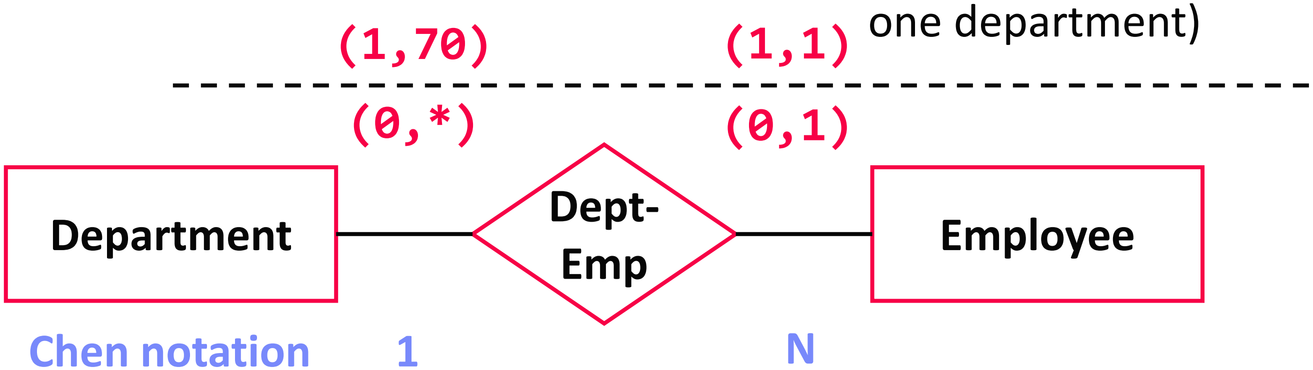


- **Alternative Cardinality Notation**
  - Indicate concrete min/max constraints
    (each entity is part of at least/at most x relationships)
  - Chen and (min,max) notation generally incomparable
  - Wildcard * indicates arbitrary many (i.e., N)

- **Examples**

(each department has 1 – 70 employees)

(each employee in exactly one department)

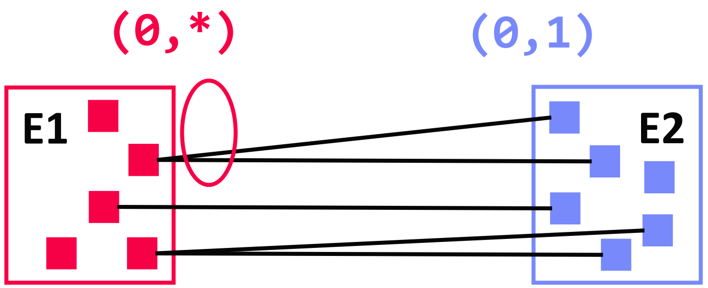


Chen notation
(for comparison)

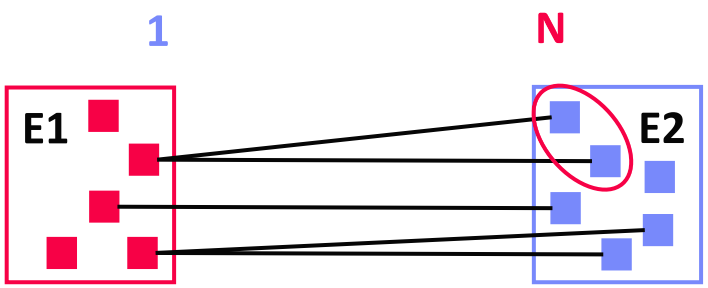
(min,max)-Notation, cont.

- **Problem: Where do these conflicting notations come from?**

- **Understanding (min, max)-Notation**
  - **Focus on relationships!**
  - Describes number of outgoing relationships for each entity

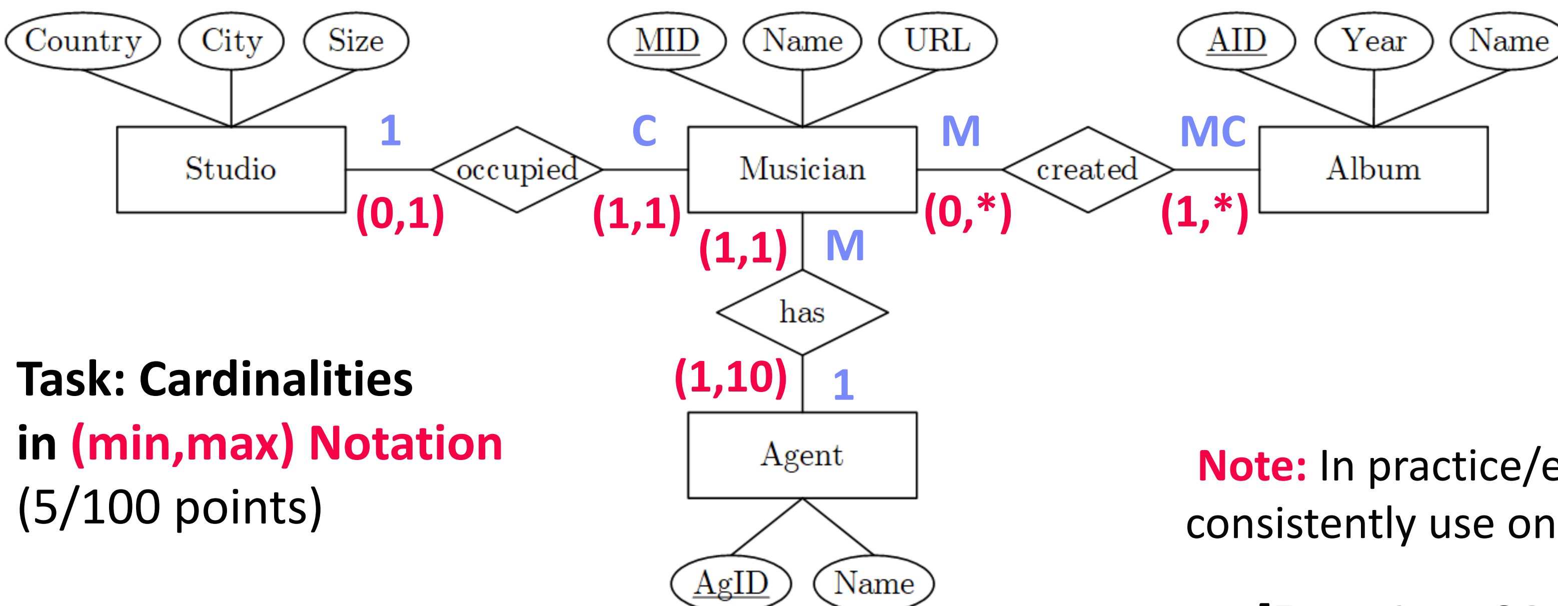


- **Understanding Chen- / Modified-Chen-Notation**
  - **Focus on entities!**
  - Describes number of target entities (over relationships) for each entity



BREAK (and Test Yourself)

- **Task: Cardinalities in Modified-Chen Notation** (prev. exam 6/100 points)
  - A musician might have created none or arbitrary many albums, and any album is created by at least one musician.
  - Every musician has exactly one agent, and an agent might be responsible for one to ten musicians.
  - Every musician occupies exactly one studio, and musicians never share a studio.



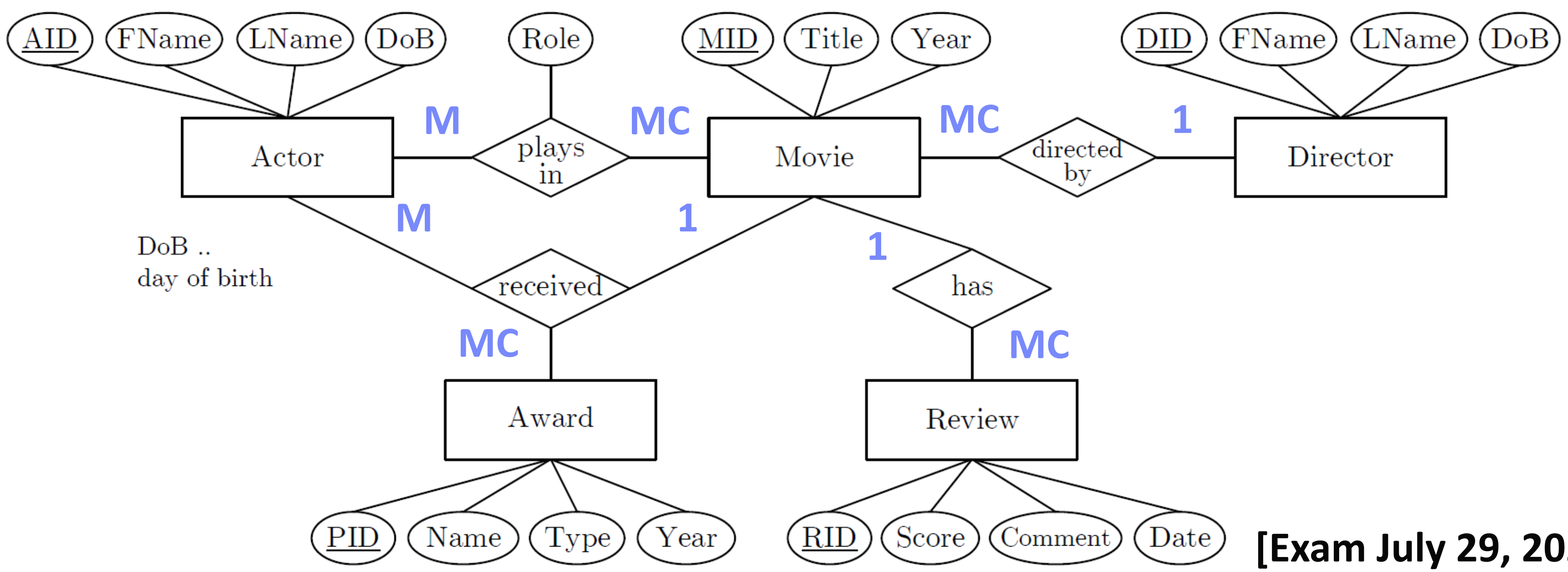
- **Task: Cardinalities in (min,max) Notation** (5/100 points)

Note: In practice/exams, consistently use only one

[Exam June 24, 2019]

BREAK (and Test Yourself), cont.

- **Task: Cardinalities in Modified-Chen Notation** (prev. exam 9/100 points)
  - An actor may play roles in an arbitrary number of movies (incl. none), every movie has a cast of at least one but potentially many actors
  - A movie is directed by 1 director, directors produce arbitrary many movies
  - A movie review refers to 1 movie, but there can be 0-many reviews per movie
  - Actors (incl a single actor) may receive multiple awards for a single movie. An actor can receive only 1 per movie. Awards to 1-many actors are possible.



[Exam July 29, 2020]

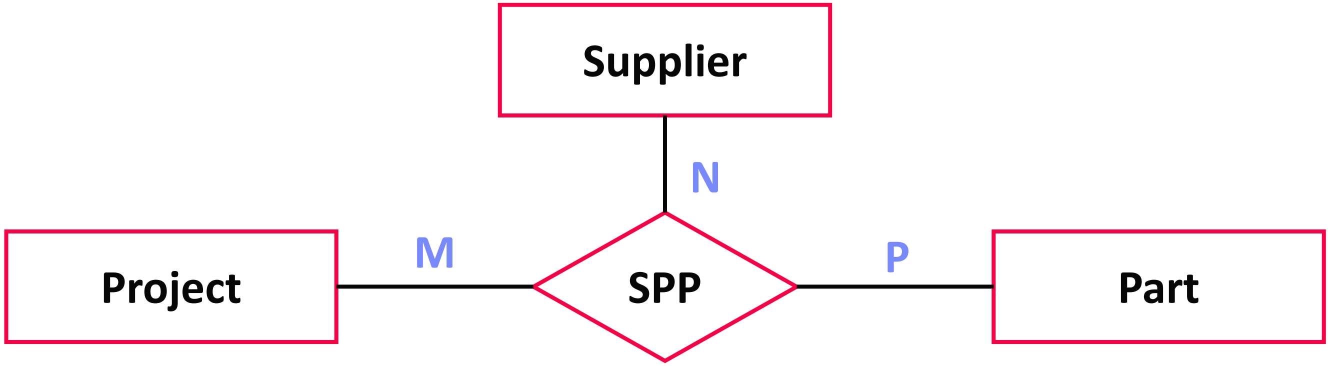
Weak Entity Types

- **Existence Dependencies**
  - Entities **E2** whose existence depends on the other entities **E1**
  - Visualized as a special rectangle with double border
  - Primary key of **E2** contains primary key of **E1**
  - Relationship between strong and weak entity types **1:N** (sometimes **1:1**)

- **Examples**
  - Dependents of an employee (spouse, children)
  - Rooms of a building

N-ary Relationships

- **Use of n-ary relationships**
  - Relationship type among multiple entity types
  - N-ary relationship can be converted to binary relationships
  - Design choice: **simplicity** and **consistency constraints**

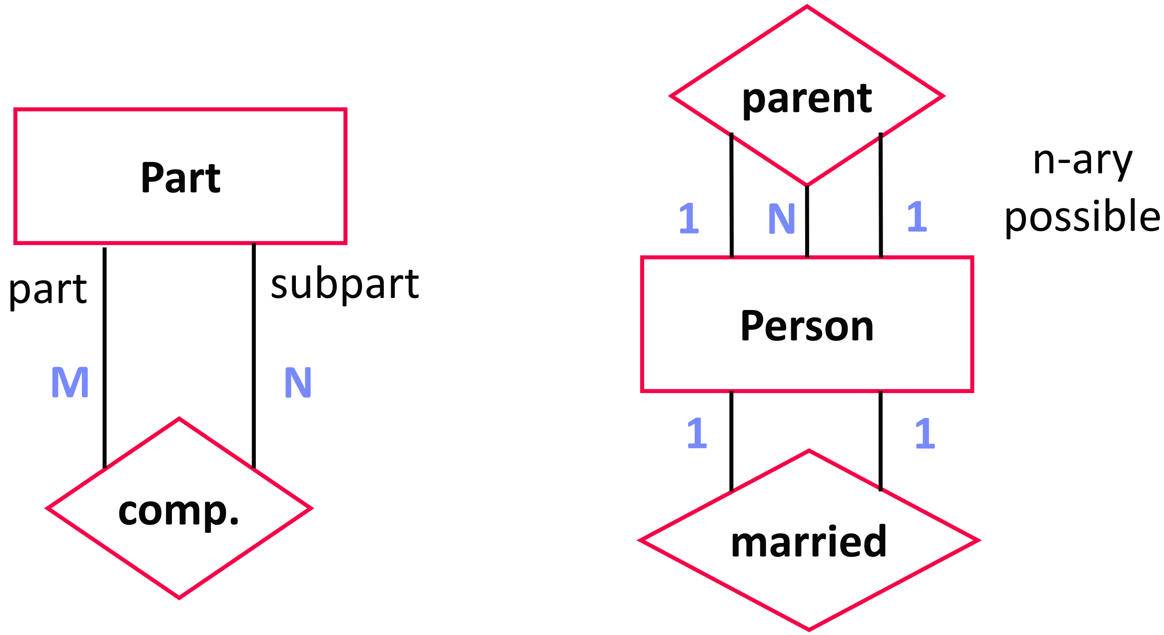


- **Multiplicity**
  - 1 Project and 1 Supplier → supply P parts
  - 1 Project and 1 Part → supplied by N suppliers (**1 instead of N?**)
  - 1 Supplier and 1 Part → supply for M projects

Recursive Relationships

- **Definition**
  - Recursive relationships are relations between entities of the same type
  - Use roles to differentiate cardinalities

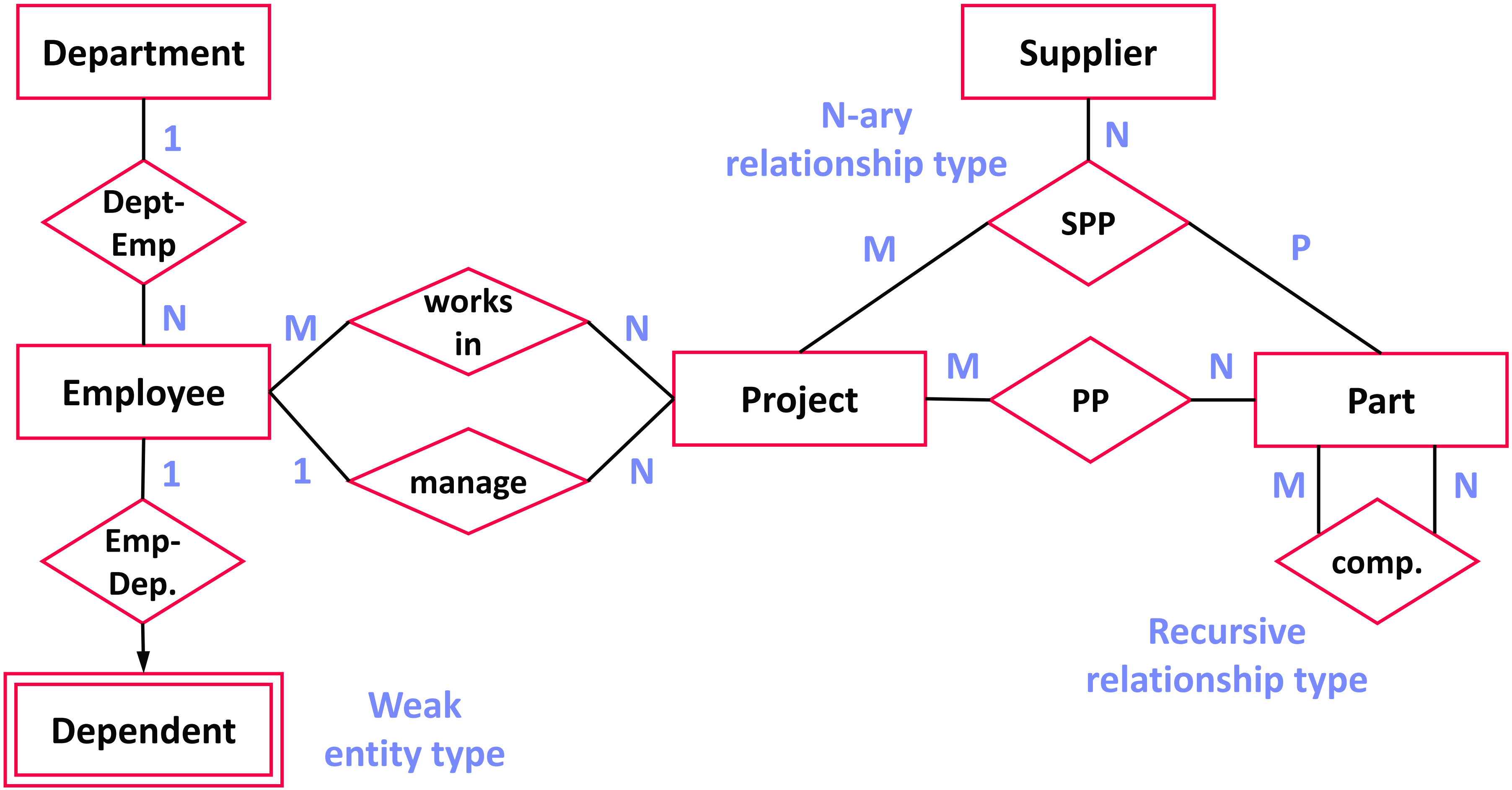
- **Examples**



- **Beware of [at least 1] constraints in recursive relationships** (e.g., (min,max)-notation, or MC notation)

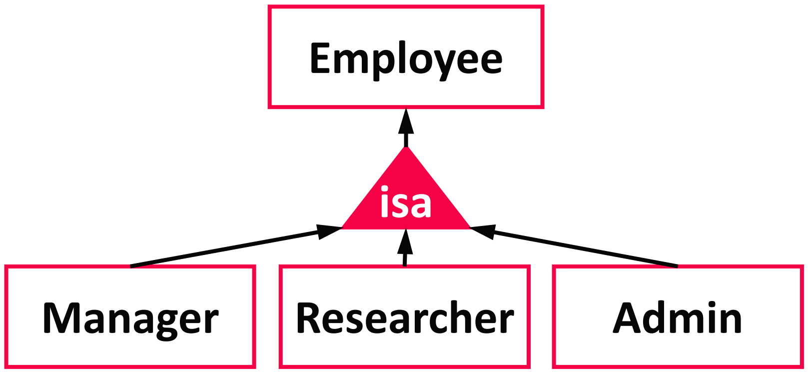
An EmployeeDB Example, cont.

[Peter P. Chen: The Entity-Relationship Model - Toward a Unified View of Data. **ACM Trans. Database Syst. 1(1) 1976**]

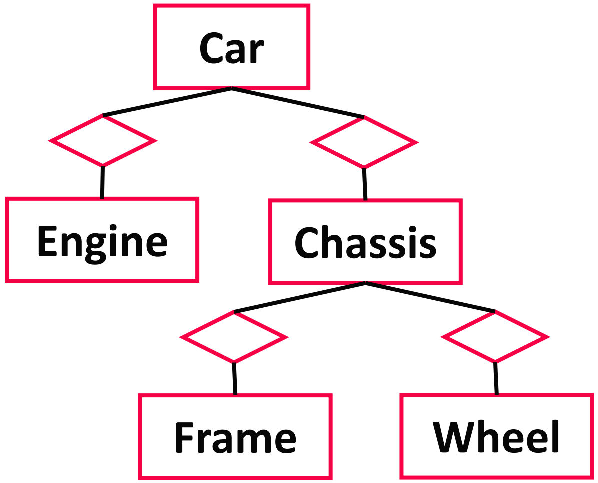


Specialization and Aggregation

- **Specialization via Subclasses**
  - **Tree of specialized entity types** (no multi-inheritance)
  - Graphical symbol: triangle (or hexagon, or subset)
  - Each entity of subclass is entity of superclass, but not vice versa

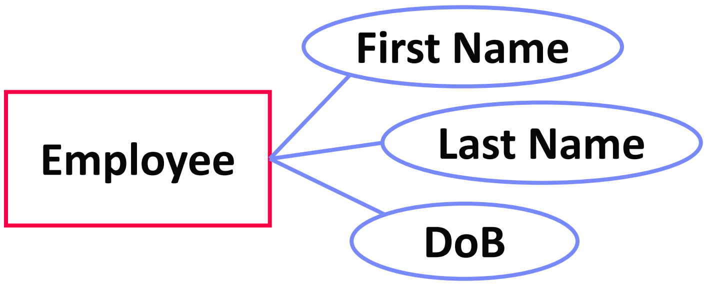


- **Aggregation** (composition, not specialization)
  - **#1: Recursive relationship types**, or
  - **#2: Explicit tree of entity** and relationship types
  - Design choice: number of types known and finite, and heterogeneous attributes
- **Beware: Simplicity is key**

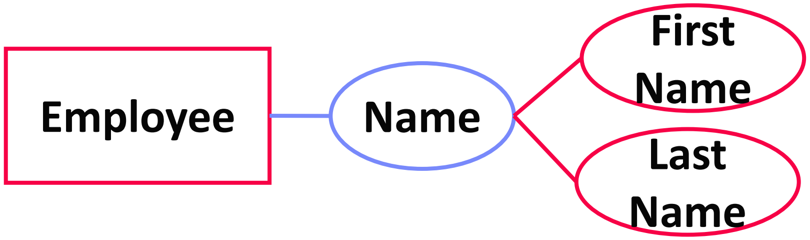


Types of Attributes

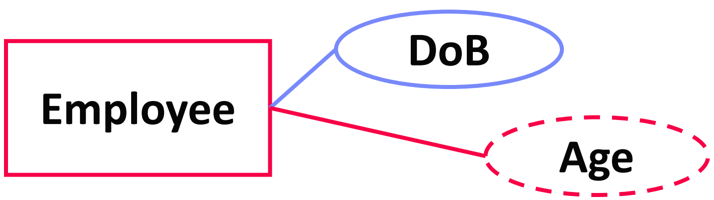
- **Atomic Attributes**
  - Basic, single-valued attributes



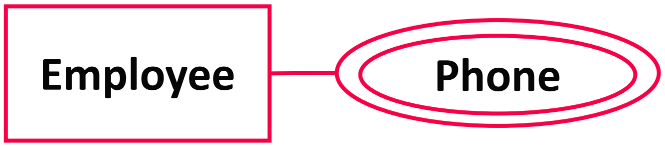
- **Composite Attributes**
  - Attributes as structured data types
  - Can be represented as a hierarchy



- **Derived Attributes**
  - Attributes derived from other data
  - Examples: Number of employees in dep, employee age, employee yearly salary



- **Multi-valued Attributes**
  - Attributes with list of homogeneous entries

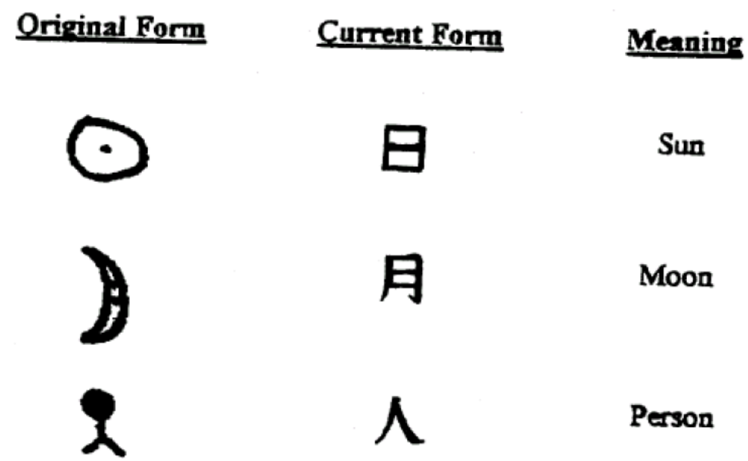


Excursus: Influence of Chinese Characters?

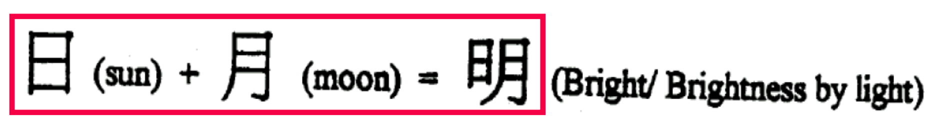
"What does the Chinese character construction principles have to do with ER modeling? The answer is: both Chinese characters and the ER model are trying to model the world – trying to use graphics to represent the entities in the real world. […]"

[Peter Pin-Shan Chen: Entity-Relationship Modeling: Historical Events, Future Trends, and Lessons Learned. **Software Pioneers 2002**]

- **Chinese characters representing real-world entities**



- **Composition of two Chinese characters**



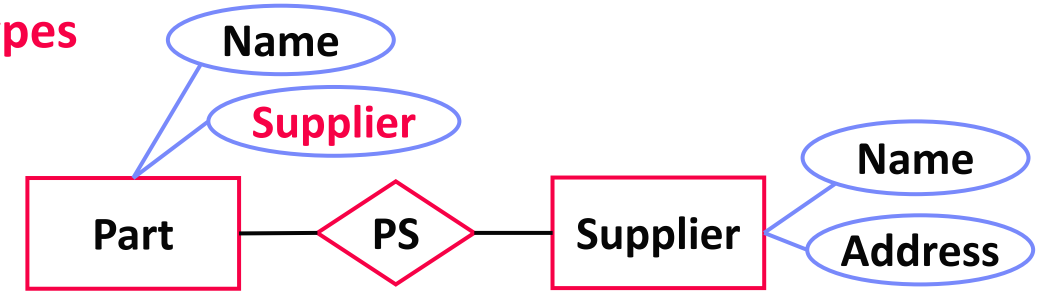
Design Decisions

Avoid redundancy
Avoid unnecessary complexity

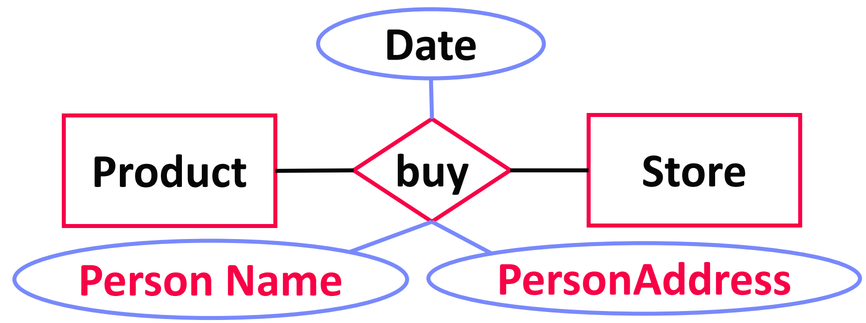
- **Meta-Level:**
  - Which notations to use (Chen, Modified Chen, (min,max)-notation)?
- **Entities**
  - What are the entity types (entity vs relationship vs attribute)?
  - What are the attributes of each entity type?
  - What are key attributes (one or many)?
  - What are weak entities (with partial keys)?
- **Relationships**
  - What are the relationship types between entities (binary, n-ary)?
  - What are the attributes of each relationship type?
  - What are the cardinalities?
- **Attributes**
  - What are composite, multi-valued, or derived attributes?

Design Decisions – Examples of Poor Choices

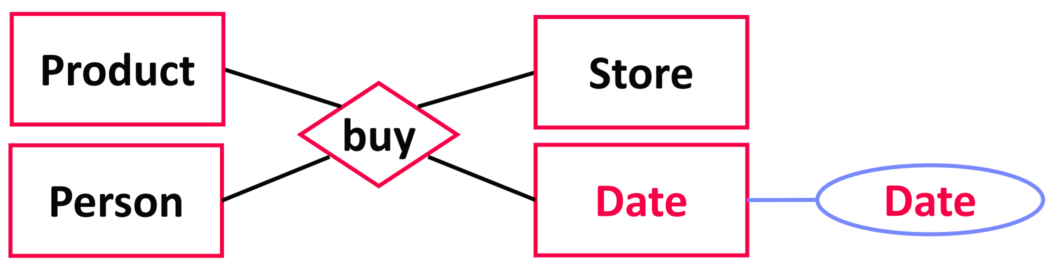
- **#1 Overuse of weak entity types**
- **#2 Redundant attributes**
  - **Redundant supplier name** in Part and Supplier



- **#3 Repeated information**
  - **Missing person entity type** → redundancy per purchase



- **#4 Unnecessary Complexity**
  - **Unnecessary entity type Date**
  - Avoid single-attribute entity types unless in many relationships

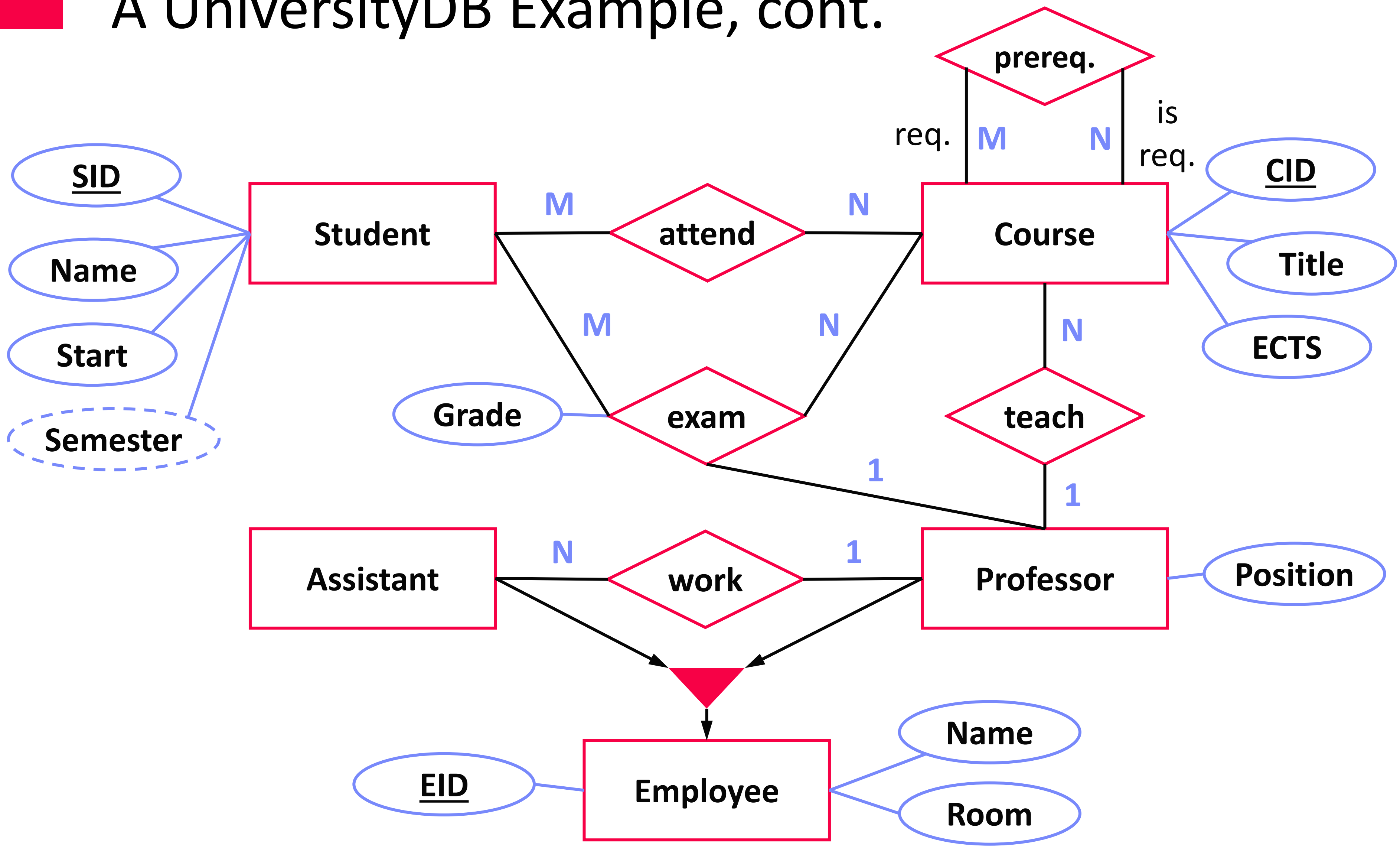


A UniversityDB Example

- **Discourse of Real Mini World**
  - **Students** (with SID, name, and semester) attend **courses** (CID, title, ECTS), and take graded exams per course
  - **Professors** teach courses and have positions, **assistants** work for professors
  - A course may have another course as prerequisites
  - Both professors and assistants are university **employees** (EID, name, and room number); professors also have a position

- **Task: Create an ER diagram in Chen notation**
  - Include entity types, relationship types, attributes, and generalizations
  - Mark primary keys, roles for recursive relationships, and derived attributes

A UniversityDB Example, cont.



Exercise 01 – Data Modeling

Published: **Mar 08, 2022**

Deadline: **Mar 29, 2022**

Exercises: Graz Districts

New

www.data.gv.at

- **Dataset**
  - Graz districts, streets, schools, universities, population counts by age and country (to be cleaned and prepared → Ex 02)
  - Clone or download your copy from https://github.com/tugraz-isds/datasets.git
  - Find CSV files in <datasets>/districts_graz

- **Exercises**
  - **01** Data modeling (relational schema)
  - **02** Data ingestion and SQL query processing
  - **03** Physical design tuning, query processing, and transaction processing
  - **04** Large-scale data analysis (distributed query processing and ML model training – anomalies?)

Overview Exercise 1 Tasks

[https://mboehm7.github.io/teaching/ss22_dbs/01_ExerciseModeling.pdf]

- **Task 1.1: ER Modeling (14/25)**
  - Graz districts / census: 17 districts, 1627 streets, addresses, schools/universities, persons, countries
  - Create an ER diagram in Modified Chen (MC) notation
  - **Partial Result:** `ERDiagram.pdf`

- **Task 1.2: Mapping ER Diagram into Relational Model (11/25)**
  - Create a relational schema in 3NF for the ER diagram from Task 1.1
  - a) text-based schema, **OR** b) SQL DDL script
  - **Partial Result:** `Schema.txt` or `CreateSchema.sql`

```
<Table>(<Attribute 1>:<type>(PK), <Attribute 2>:<type>, ..., <Attribute n>(FK))
```

- **Expected result** (for all three subtasks)
  - **`DBExercise01_<studentID>.zip`**

Don't get your own studentID wrong

Overview Exercise 1 – Discourse

- Graz has 17 *districts* (e.g., `6., Jakomini`), each characterized by a unique DistrictID, one or multiple postal codes, a population count (as of 01/2022), an area (in km^2), and a population density (population count / area).
- Every *street* (e.g., `1850, Inffeldgasse, 6`) belongs to exactly one district, and has a unique Graz street code, and a street name.
- An *address* in Graz is characterized by its street, a street number, and an apartment number (where the apartment number might be N/A).
- *Schools* and *universities* have a unique name, an address, a phone number, and a type of educational institution.
- Every registered *person*—characterized by a first name, a last name, a gender (female, male, diverse), a day of birth (DoB), and age—has exactly one primary residence address, and an arbitrary number of secondary residence addresses. Furthermore, every person has a citizenship of at least one country; and every *country* is described by a unique ID, a unique three-letter country code, and a country name.

(person counts potentially different from population count)

Summary and Q&A

- **Summary**
  - DB Design lifecycle from requirements to physical design
  - Entity-Relationship (ER) Model and Diagrams
- **Importance of Good Database Design**
  - Poor database design ➔ **development and maintenance costs**, as well as performance problems
  - Once data is loaded, **schema changes very difficult** (data model, or conceptual and logical schema)
- **Exercise 1: Data Modeling**
  - Published: Mar 08, 2022; deadline: Mar 29, 2022
  - **Recommendation:** start with task 1.1 this week; ask questions in upcoming lectures or on news group
- **Next lecture: 03 Data Models and Normalization** [Oct 14]