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: Oct 11, 2020
Announcements/Org

- **#1 Video Recording**
  - Link in TeachCenter & TUbe (lectures will be public)

- **#2 Course Registrations SS20**
  - Data Management (lectures/exercises): 143/144
  - Databases (combined lectures/exercises): 68

- **#3 Exercise 1**
  - Task Description published last night (discussed in today’s lecture)
  - Deadline: Nov 03 in TeachCenter

- **#4 Learning Analytics (LA)**
  - LA Research Study by Carla Souta Barreiros

"Participate and make a positive impact"
Agenda

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

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

Diagram:
- Discourse of real mini world
- Lecture 02
- Lecture 03
- Conceptual Schema (ER diagram)
  - Relational Schema
  - XML Schema
  - Network Schema
  - Object-ori. Schema
- Manual Modeling
- Semi-automatic Transformation
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
  - Object-oriented
  - Network
  - Hierarchical

  - **Partly covered in part B**
  - **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  - Model-driven development (self-documenting, but quickly outdated)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. *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

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

- **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

---

**Employee**

**works ⊆ A × B**

**First Name**

**works in**

**Weak entities**

**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

- **1:1 (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.

Partial Function
Employee $\not\rightarrow$ 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)
- (an employee can manage N projects)
- (a project is managed by 1 employee)
- (an employee can manage N projects)

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

- **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

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

\[ n \cdot (n + 1) \]
\[ \frac{2} \]
(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)
    - (1, 70)
    - (0, *)
  - (each employee in exactly one department)
    - (1, 1)
    - (0, 1)

Diagram:
- Department
- Dept-Emp
- Employee
  - Chen notation (for comparison)
  - 1
  - N

Diagram notation:
- (min₁, max₁) (min₂, max₂)
(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 $E_2$ whose existence depends on the other entities $E_1$
  - Visualized as a special rectangle with double border
  - Primary key is contains primary key of $E_1$
  - Relationship between strong and weak entity types $1:N$ (sometimes $1:1$)

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

```
Employee --- 1 --- dep. --- Dependent
```
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*

![Entity-Relationship (ER) Model and Diagrams](image)

- **Multiplicity**
  - 1 Project and 1 Supplier $\rightarrow$ supply $P$ parts
  - 1 Project and 1 Part $\rightarrow$ supplied by $N$ suppliers (*1 instead of N?*)
  - 1 Supplier and 1 Part $\rightarrow$ supply for $M$ projects
Recursive Relationships

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

- **Examples**

  ![Diagram](image)

  - **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]

Entity-Relationship (ER) Model and Diagrams

Department

Employee

Dependent

Supplier

Part

Project

Dept-Emp

Emp-Dep.

Dept-Emp

Emp-Dep.

Department

Employee

Dependent

Supplier

Part

Project

Dept-Emp

Emp-Dep.

Dept-Emp

Emp-Dep.

N-ary relationship type

Recursive relationship type

Weak entity type

Works in

Manages

N-ary relationship type
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
Excurcus: 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. [...]”

- Chinese characters representing real-world entities

- Composition of two Chinese characters

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

- **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: Oct 12, 2020
Deadline: Nov 03, 2020
Exercises: The Movies Dataset

- **Dataset**
  - Derived (extracted, cleaned) from The Movies Dataset for movies year ≥ 2011
  - Note: Still in process of data cleaning
  - Clone or download your copy from https://github.com/tugraz-isds/datasets.git
  - Find CSV files in <datasets>/movies

- **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)
Overview Exercise 1 Tasks

- **Task 1.1: ER Modeling (movies, cast, ratings)**
  - Create an ER diagram in Modified Chen (MC) notation
  - [https://github.com/tugraz-isds/datasets/tree/master/movies](https://github.com/tugraz-isds/datasets/tree/master/movies)

- **Task 1.2: Mapping ER Diagram into Relational Model**
  - Create a relational schema in 3NF for the ER diagram from Task 1.1
  - **FDs:** LangCode → Language, CountryCode → Country, multi-valued attributes
  - a) text-based schema, OR b) SQL DDL script

- **Task 1.3: Relational Normalization**
  - Explain why the schema from Task 1.2 is in third normal form (3NF)

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

[https://mboehm7.github.io/teaching/ws2021_dbs/01_ExerciseModeling.pdf]
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 Oct 12, 2020; deadline: Nov 03, 2020
  - **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 19]