COMP5138
Relational Database Management Systems

Lecture 2:
Conceptual Database Design

Today’s Agenda

• Introduction
• Entity Relationship Model
• Case Study
• Unified Model Language
### Database Design Sequence

| Requirements Analysis | Understand…  
|-----------------------|-------------  
|                       | what data is to be stored  
|                       | what applications must be built  
|                       | what operations are most frequent  
| Conceptual Design     | Develop…  
|                       | high-level description of the data closely matching how users think of the data  
|                       | Works as communication vehicle  
| Logical Design        | Convert…  
|                       | conceptual design into a logical database schema  
| Physical Design       | Convert…  
|                       | logical schema into a physical schema for a specific DBMS and tuned for application  

### Conceptual Data Model

- **Goal:** Specification of database schema  
- **Methodology:**  
  - **Conceptual Design:** A technique for understanding and capturing business information requirements **graphically**  
    - depicts the associations among different categories of data within a business or information system.  
    - Convert conceptual database design to DDL  
  - Conceptual Database Design does **not** imply how data is implemented, created, modified, used, or deleted.  
    - Works as communication vehicle  
  - An conceptual data model is **model & database independent**
Conceptual Data Models

- Entity-Relationship Model (ERM)
- Object-oriented Data Models
  - Unified Modelling Language (UML)
  - OMT, Booch, ...
  - Object Role Modelling (ORM)
  - Semantic Object Model (SOM)
  - Semantic Data Models (SDM)
  - KL-ONE etc.

Today’s Agenda

- Introduction
- Entity Relationship Model
- Case Study
- Unified Model Language
Entity Relationship Model

- Data model for conceptual database design:
  - High-level graphical representation of what data needs to be contained in the system and how they are related.
  - Tools: ERwin, Sybase Power Designer, ...
  - Templates for all kinds of drawing tools, e.g., Microsoft Visio

- First designed by Peter Chen in 1976
  - Several variations have since appeared
  - Here: enhanced or extended E-R model

E-R Model: enterprise is viewed as a set of
  - Entities
  - Relationships among entities

Entities

- **Entity**: an object that is involved in the enterprise and is distinguishable from other objects.
  - Example: John Doe, unit COMP5138, account 4711

- **Entity Set** (also: entity type): set of similar objects
  - members share the same properties / characteristics
  - Example: students, courses, accounts
  - Note: entity sets need not to be disjoint (e.g. person who is manager)

- **Attribute**: describes one aspect of an entity type
  - Example: people have names and addresses
  - depicted by an ellipses
**Entity Set**

- Entity Set is described by a set of attributes
  - Descriptive properties possessed by all members of an entity set
  - Example: Person has ID, Name, Address, Hobbies

- **Domain**: possible values of an attribute
  - In contrast to relational model values can be complex / set-oriented!
    - Simple and composite attributes.
    - Single-valued and multi-valued attributes
  - Example see next slide

- **Key**: minimal set of attributes that uniquely identifies an entity in the set (several such candidate keys possible)
  - One chosen as Primary Key (PK)

- **Entity Schema**: entity set name, attributes (+domains), PK

---

**Graphical Representation in E-R Diagram**

**Symbols:**
- Entity Sets represented by a rectangle
- Attributes depicted by ellipses
  - Double ellipses for multi-valued attributes
  - Keys are underlined

**Examples:**
- Entities Sets
  - `Books`
  - `Employees`

- Attributes
  - `isbn`
  - `title`
  - `authors`
  - `tfn`
  - `first`
  - `last`
  - `name`

- Relations
  - Books.authors is a multi-valued attribute;
  - Employees.name is a composite attribute.

*The University of Sydney*
Relationships

- **Relationship**: relates two or more entities
  - number of entities is also known as the **degree** of the relationship
  - Example: John is enrolled in COMP5138

- **Relationship Set** (*R.ship Type*): set of similar relationships
  - Formally: a relation among \( n \geq 2 \) entities, each from entity sets:
    \[ \{(e_1, e_2, \ldots, e_n) | e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\} \]
  - Example: Students (entity set) related to UnitOfStudies (entity set) by EnrolledIn (relationship set).

- **Distinction**:
  - relation (relational model) - set of tuples
  - relationship (E-R Model) – describes relationship between entities
  - Both entity sets and relationship sets (E-R model) may be represented as relations (in the relational model)

Relationship Attributes & Roles

- **Relationship-Attribute**
  Relationships can also have additional properties
  - E.g., John enrolles in COMP5138 in the second semester 2005
  - John and COMP5138 are related
  - Semester 2 2005 describes the relationship - value of the Semester attribute of the EnrolledIn relationship set

- **Relationship-Role**
  Each participating entity can be named with an explicit role.
  - E.g. John is value of **Student** role, COMP5138 value of **Subject** role
  - useful for relationship that relate elements of same entity type
  - Example: Supervises ( Employee:Manager, Employee )
Graphical Representation of Relationships in E-R Diagrams

Symbol:

- Diamonds represent relationship sets
- Lines link attributes to entity sets and entity sets to relationship sets.
- Roles are edges labeled with role names

Example

Relationship Degree

- Degree of a Relationship: # of entity types involved
  - Unary Relationship (Recursive)
  - Binary Relationship (80%)
Schema of Relationship Sets

- The combination of the primary keys of the participating entity types forms a super key of a relationship.
  - Example: (student_Id, UoS_number) is the super key of Enrolls
  - One must consider the mapping cardinality of the relationship when deciding what are the candidate keys

- Relationship Set Schema:
  - Relationship name
  - Role names (or: names of participating entity sets)
  - Relationship attributes and their types
  - key

Key Constraints

- If, for a particular participant entity type, each entity participates in at most one relationship, the corresponding role is a key of relationship type
  - E.g., Employee role is unique in WorksIn
  - but there may be employees who are working in no department at all
  - also called: one-to-many or 1:N relationship

- Representation in E-R diagram: arrow

- Example: An employee works in at most one department.
Participation Constraint

- If every entity participates in \textit{at least one} relationship, a participation constraint holds:
  - A participation constraint of entity type $E$ having role in relationship type $R$ states that for $e$ in $E$ there is an $r$ in $R$ such that $(r) = e$.
  - Also called a \textbf{total participation} of $E$ in $R$.
- A participation that is not total is said to be \textbf{partial}.
- Representation in E-R diagram: thick line.
- Example: every employee works in at least one department.

Participation and Key Constraint

- If every entity participates in \textit{exactly one} relationship, both a participation and a key constraint hold.
  - Again: 1:N relationship.
- Representation in E-R diagrams: thick arrow.
- Example: Every employee works in exactly one department.
Cardinality Constraints

- Generalisation of key and participation constraints
- A **cardinality constraint** for the participation of an entity set $E$ in a relationship $R$ specifies how often an entity of set $E$ participates in $R$ at least (minimum cardinality) and at most (maximum cardinality).
  - In an ER-diagram we annotate the edge between an entity set $E$ and relationship $R$ with $min..max$, where $min$ is the minimum cardinality and $max$ the maximum cardinality. If no maximal cardinality is specified, we set '*' as max number ("don’t care").
- Example: Every employee works in 1 to 3 departments.
  - Note: This is an extension to the textbook’s notation!

Comparison of Notations

"Every employee can manage several departments."  "Every Dept. is managed by exactly one employee"

**With Key & Participation Constraint:** (cf. textbook, page 33)

- partial participation
- multiple side

**With Cardinality Constraints:** (cf. Kifer/Bernstein, pages 76/77 and 82/83)

- partial participation
- multiple side

"Crow's-foot" notation: (e.g. Hoffer, ed7, pages 95,116-122)

Be aware that the Crow's foot notation in inverse to our notation from the textbook!
Weak Entities

- **Weak entity set**: An entity set that does not have a primary key.
  - Its existence depends on the existence of one or more *identifying entity sets*
  - It must relate to the identifying entity set via a total, one-to-many *identifying relationship set* from the identifying to the weak entity set
  - Examples: *child* from parents, *payment* of a loan

- The **discriminator** (or *partial key*) of a weak entity set is the set of attributes that distinguishes among all the entities of a weak entity set related to the same owning entity.

- The primary key of a weak entity set is formed by the primary key of the strong entity set(s) on which the weak entity set is existence dependent, plus the weak entity set’s discriminator.

**Representation of Weak Entity Set**

Symbols:
- We depict a weak entity set by double rectangles.
- Identifying relationship depicted using a double diamond
- Underline the discriminator of a weak entity set with a dashed line

Example:
- `loanNumber`, `amount` (Loan entity set)
- `paymentNumber`: discriminator of the payment entity set
- Primary key for payment: `(loanNumber, paymentNumber)`
Enhanced E-R Model

- ER model in its original form did not support
  - SPECIALIZATION/ GENERALIZATION
  - ABSTRACTIONS

- This led to the development of ‘Enhanced’ ER model
  - Includes all modeling concepts of basic ER
  - Additional some object-oriented concepts:
    subclasses/superclasses, specialization/generalization, categories, attribute inheritance
  - The resulting model is sometimes called the enhanced-ER or Extended ER (E2R or EER) model
    - used to model applications more completely and accurately if needed

- If we will talk about E-R model, we always mean EER model

Generalisation / Specialisation

- Arranging of entity types in a type hierarchy.
  - Determine entity types whose set of properties are actual a subset of another entity type.

- Definition Generalisation / Specialisation / Inheritance:
  Two entity types $E$ and $F$ are in an ISA-relationship (“$F$ is a $E$”), if
  (1) the set of attributes of $F$ is a superset of the set of attributes of $E$, and
  (2) the entity set $F$ is a subset of the entity set of $E$ (“each $f$ is an $e$”)

- One says that $F$ is a specialisation of $E$ ($F$ is subclass) and $E$ is a generalisation of $F$ ($E$ is superclass).
  - Example: Freshman is a subclass of Student

- Attribute inheritance – a lower-level entity type inherits all the attributes and relationship participations of its supertype.
- Depicted by a triangle component labeled ISA
Superclass/Subclass Example

represents two ISA relationships

Constraints on ISA Hierarchies

• We can specify overlap and covering constraints for ISA hierarchies:
  • Overlap Constraints
    – Disjoint
      • an entity can belong to only one lower-level entity set
      • Noted in E-R diagram by writing disjoint next to the ISA triangle
    – Overlapping  (the default - opposite to textbook!)
      • an entity can belong to more than one lower-level entity set
  • Covering Constraints
    – Total
      • an entity must belong to one of the lower-level entity sets
      • Denoted with a thick line between the ISA-triangle and the superclass
    – Partial  (the default)
      • an entity need not belong to one of the lower-level entity sets
Today’s Agenda

• Introduction

• Entity Relationship Model

• Case Study

• Unified Model Language

Case Study

• Textbook, Section 2.8: Case Study “The Internet Book-Shop”
Today’s Agenda

• Introduction
• Entity Relationship Model
• Case Study
• Unified Model Language

UML

• UML: Unified Modeling Language
• UML has many components to graphically model different aspects of an entire software system
  – Use Case
  – Activity
  – Statechart
  – Class
• UML Class Diagrams correspond to E-R Diagram, but there are several differences.
UML Class Diagram Notation

1. Entity sets and attributes

```
| customer-name | customer-street | customer-id | customer-city |
```

```
customer
```

2. Relationships

```
E1 role1 R role2 E2
```

```
a1 a2
```

```
E1 role1 R role2 E2
```

```
E1 role1 R role2 E2
```

UML Class Diagrams

- Entity sets are shown as boxes, and attributes are shown within the box, rather than as separate ellipses in E-R diagrams.
- Binary relationship sets are represented in UML by just drawing a line connecting the entity sets. The relationship set name is written adjacent to the line.
- The role played by an entity set in a relationship set may also be specified by writing the role name on the line, adjacent to the entity set.
- The relationship set name may alternatively be written in a box, along with attributes of the relationship set, and the box is connected, using a dotted line, to the line depicting the relationship set.
- Non-binary relationships drawn using diamonds, just as in ER diagrams.
Cardinality constraints are specified in the form \( l..h \), where \( l \) denotes the minimum and \( h \) the maximum number of relationships an entity can participate in.

- the positioning of the constraints is OPPOSITE to the positioning of cardinality constraints in our E-R notation
  - it is the same as the positioning of constraints in crow's foot notation.
- The constraint \( 0..* \) on the E2 side and \( 0..1 \) on the E1 side means that each E2 entity can participate in at most one relationship, whereas each E1 entity can participate in many relationships; in other words, the relationship is many to one from E2 to E1.
- Single values, such as 1 or * may be written on edges; The single value 1 on an edge is treated as equivalent to 1..1, while * is equivalent to 0..*.

3. Cardinality constraints

4. Generalization and Specialization
Summary

• Conceptual Modeling
• ERD
  – Entity
  – Attributes
    • Single/Multivalued, Simple/Composite attributes
    • Keys
  – Relationship
    • Key, Participation and Cardinality Constraints
  – Weak Entities
  – ISA-Hierarchies: Generalization/Specialization (Inheritance)
• UML
  – Class Diagram

Next Week

• Introduction to Relational Data Model
  – Foundations
  – DDL part of SQL

• Logical Database Design
  – Transforming E-R Diagrams into Relations

• Relational Algebra
  – A formal query language for the relational data model

• Textbook
  – Chapters 3 and Chapter 4 (without section 4.3)