Entity–Relationship Data Model

Tutorial

SWEN304/SWEN439

Trimester 1, 2021

Lecturer: Dr Hui Ma

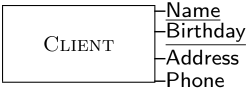
Engineering and Computer Science

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Entity-Relationship Model - Terminology

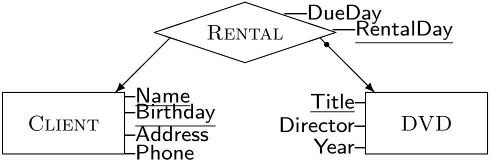
- The target of the database is regarded as consisting of **entities** and **relationships**
- An **Entity type** is used to describe a set of entities with the same set of attributes
- A **Key** is needed to **uniquely identify** the entities in an entity set
- An entity type E is defined by a finite, non-empty set of attribute $attr(E)$, and a key $id(E)$
- write $E = (attr(E), id(E))$
  with $attr(E)$ the set of attributes and $id(E)$ the key of E
- to indicate an attribute A with a domain D we write $A : D$, e.g. Name: STRING (sometime we omit)
- **Example:** $\textsc{Client} = (\{\text{Name, Birthday, Address, Phone}\}, \{\text{Name, Birthday}\})$

  An entity type $\textsc{Client}$ with a set of attributes $\{$Name, Birthday, Address, Phone$\}$, and a key $\{$Name, Birthday$\}$



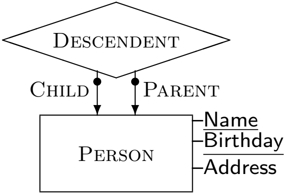
Entity-Relationship Model - Terminology

- **Relationships** are associations between entities, or objects that are derived from entities
- **Relationships type** is used to model a set of relationships that are described by the same set of components and attributes
- A **Keys** is needed to **uniquely identify** the relationships in a relationship set
- A Relationship type is defined by a set of **components** $comp(R)$, a set of **attributes** $attr(R)$, and a key $id(R) \subseteq comp(R) \cup attr(R)$ (*a subset of its components and attributes*)
- write $R = (comp(R), attr(R), id(R))$
- **Example:**
  $\textsc{Rental} = (\{\textsc{Client}, \text{ DVD}\}, \{\text{RentalDay, DueDay}\}, \{\text{DVD}, \text{RentalDay}\})$
  $\textsc{Rental}$ has a component set $\{\textsc{Client}, \text{ DVD}\}$, an attribute set $\{\text{RentalDay, DueDay}\}$, and a key $\{\text{DVD}, \text{RentalDay}\}$



Entity-Relationship Model - Terminology

- A relationship type is called **recursive** if it contains the same object type (Entity type or Relationship Type) more than once as a component
- **Role names** are used to distinguish the different occurrences of the same entity type in a recursive relationship type (in our example $\textsc{Child}$ and $\textsc{Parent}$)
- **Example:** $\textsc{Descendent} = (\{\text{Parent} : \textsc{Person}, \text{Child} : \textsc{Person}\}, \emptyset, \{\text{Parent} : \textsc{Person}, \text{Child} : \textsc{Person}\})$



Higher-order Relationship Types

- A relationship type R is of **order 1** if all its components are entity types
  - **Example**: $\textsc{Rental}$ is of order 1, $\textsc{Client}$ is of order 0
- A relationship type R is of **order k** if its components have maximum order $k - 1$
  - **Example**: relationship type GraduateStudent is of order 2
- For the sake of convenience, entity types and relationship types together are called **object types**
  - Entities and relationships are called objects
  - Entity sets and relationship sets are called object sets
  - Entity types may be regarded as object types of order 0
  - Object types of order k are just relationship types of order k

Cluster

- sometimes it is necessary to model **alternatives**
- A **cluster** C represents an alternative among a collection of object types
  - let C be an alternative between given object types $C_1, \ldots, C_m$
  - for simplicity, the cluster can be denoted as $C = C_1 \oplus \cdots \oplus C_m$
  - we call $C_1, \ldots, C_m$ the **components** of the cluster C

- **Example:** $\textsc{Employee} = \textsc{Lecturer} \oplus \textsc{Tutor} \oplus \textsc{General\_Staff}$
- Every object of cluster type C is an object of exactly one of the object types in the alternative
- A cluster is of **order** k if its components have maximum order $k-1$
- **Example**: cluster Hires is of order 3.

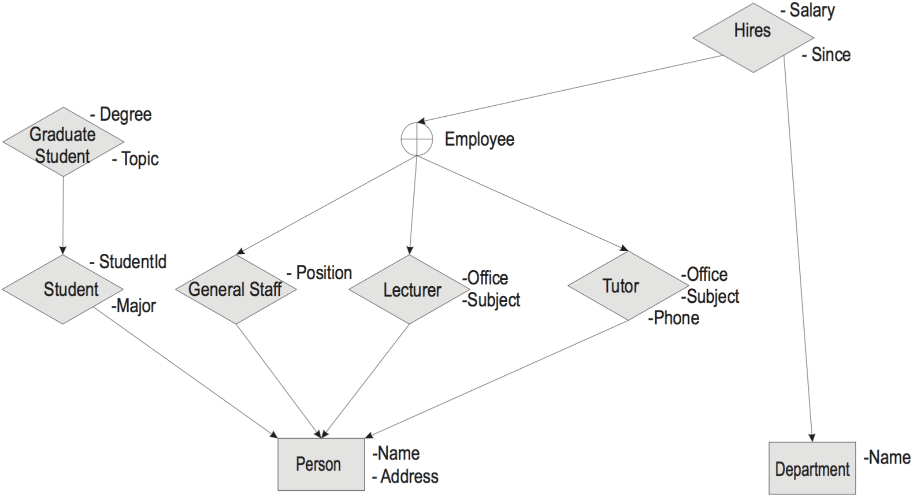
Extended Entity-Relationship Schema and Diagram

- An **extended entity-relationship schema** (**extended ER schema**, for short) is a finite set $\mathcal{S}$ of object types (entity types, relationship types, clusters), such that for every object type O in $\mathcal{S}$ all its components belong to $\mathcal{S}$, too

- An **instance** $\mathcal{S}^t$ of an extended ER schema $\mathcal{S}$ assigns each object type O an object set O^t, such that for each object o in the object set O^t and for every component O' of O the object $o(O')$ belongs to the object set O'^t

- A **extended entity-relationship diagram** (**extended ER diagram**, for short) is a directed graph with a node for every entity type or relationship type
  - attributes in the ER diagram by attaching them to their entity or relationship type
  - keys in the ER diagram by underlining key attributes and by putting dots on key components
  - attribute domains in the ER diagram by adding them to the attributes
  - **Convention**: draw entity types as rectangles, draw relationship types as diamonds, draw clusters as $\oplus$

EER Diagram: An Example


Hires
- Salary
- Since
Graduate Student
- Degree
- Topic
Employee
Student
- StudentId
-Major
General Staff
- Position
Lecturer
-Office
-Subject
Tutor
-Office
-Subject
-Phone
Person
-Name
- Address
Department
-Name