



# Lecture 14: Entity Relationship Modelling

## o The Entity-Relationship Model

- o Entities
- o Relationships
- o Attributes

## o Constraining the instances

- o Cardinalities
- o Identifiers
- o Generalization



# The Entity Relationship Model

## o Entity-Relationship Schema

- o Describes data requirements for a new information system
- o Direct, easy-to-understand graphical notation
- o Translates readily to relational schema for database design
  - o But more abstract than relational schema
  - o E.g. can represent an entity without knowing its properties
- o comparable to UML class diagrams

## o Entities:

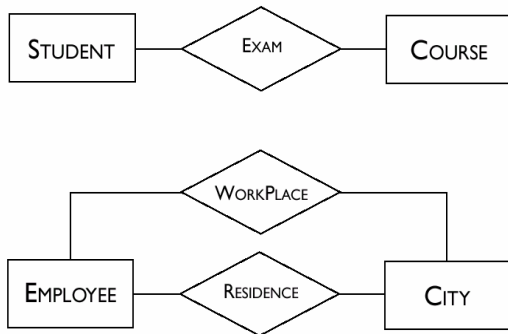
- o classes of objects with properties in common and an autonomous existence
  - o E.g. City, Department, Employee, Purchase and Sale
- o An instance of an entity is an object in the class represented by the entity
  - o E.g. Stockholm, Helsinki, are examples of instances of the entity City

## o Relationships:

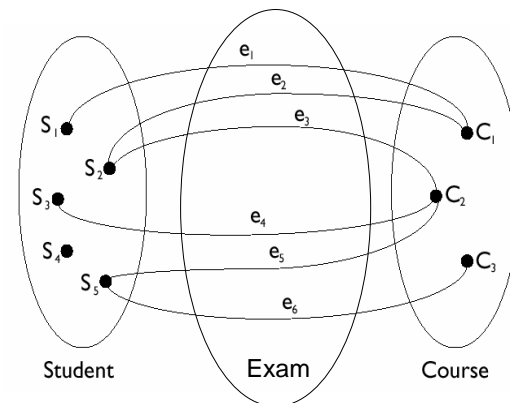
- o logical links between two or more entities.
  - o E.g. Residence is a relationship that can exist between the City and Employee
- o An instance of a relationship is an n-tuple of instances of entities
  - o E.g. the pair (Johansen, Stockholm), is an instance in the relationship Residence.



# Examples



# Example Instances for Exam



University of Toronto Department of Computer Science

## What Does An E-R Diagram Really Mean?

```

    graph LR
      Course[Course] --- Meets{Meets} --- Room[Room]
  
```

- Course and Room are entities.
  - Their instances are particular courses (eg CSC340F) and rooms (eg MB128)
- Meets is a relationship.
  - Its instances describe particular meetings.
  - Each meeting has exactly one associated course and room

Course instances Meets instances Room instances

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 5

University of Toronto Department of Computer Science

## Recursive Relationships

- an entity can have relationships with itself...
 

```

          graph TD
            EMPLOYEE[EMPLOYEE] --- COLLEAGUE{COLLEAGUE} --- EMPLOYEE
        
```
- If the relationship is not symmetric...
  - ...need to indicate the two roles that the entity plays in the relationship.

```

          graph TD
            SOVEREIGN[SOVEREIGN] --- SUCCESSION{SUCCESSION} --- SOVEREIGN
            SUCCESSION --- Predecessor[Predecessor]
            SUCCESSION --- Successor[Successor]
        
```

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 6

University of Toronto Department of Computer Science

## Ternary Relationships

```

    graph LR
      SUPPLIER[SUPPLIER] --- SUPPLY{SUPPLY} --- PRODUCT[PRODUCT]
      SUPPLY --- DEPARTMENT[DEPARTMENT]
  
```

Supplier Product Department

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 7

University of Toronto Department of Computer Science

## AND/XOR Relationships

```

    graph LR
      Order[Order] --- XOR{XOR} --- Part[Part]
      Order --- XOR --- Service[Service]
  
```

"Each Order either contains a part or requests a service, but not both"

```

    graph LR
      Order[Order] --- AND{AND} --- Shipment[Shipment]
      Order --- AND --- Invoice[Invoice]
  
```

"For any given order, whenever there is at least one invoice there is also at least one shipment and vice versa"

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 8

University of Toronto Department of Computer Science

## Attributes

⇒ associates with each instance of an entity (or relationship) a value belonging to a set (the domain of the attribute).  
 ↳ The domain determines the admissible values for the attribute.

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 9

University of Toronto Department of Computer Science

## Composite Attributes

⇒ These group attributes of the same entity or relationship that have closely connected meanings or uses.

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 10

University of Toronto Department of Computer Science

## Schema with Attributes

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 11

University of Toronto Department of Computer Science

## Cardinalities

⇒ Cardinalities constrain participation in relationships  
 ↳ maximum and minimum number of relationship instances in which an entity instance can participate.  
 ↳ E.g.

⇒ cardinality is any pair of non-negative integers (a,b)  
 ↳ such that a=b.  
 ↳ If a=0 then entity participation in a relationship is optional  
 ↳ If a=1 then entity participation in a relationship is mandatory.  
 ↳ If b=1 each instance of the entity is associated at most with a single instance of the relationship  
 ↳ If b="N" then each instance of the entity is associated with an arbitrary number of instances of the relationship.

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 12

University of Toronto Department of Computer Science

## Cardinality Example

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 13

University of Toronto Department of Computer Science

## Instantiating ER diagrams

⇒ An ER diagram specifies what states are possible in the world being modeled

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 14

University of Toronto Department of Computer Science

## Illegal Instantiations

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 15

University of Toronto Department of Computer Science

## Cardinalities of Attributes

⇒ Attributes can also have cardinalities

- ↳ To describe the minimum and maximum number of values of the attribute associated with each instance of an entity or a relationship.
- ↳ The default is (1,1)
- ↳ Optional attributes have cardinality (0,1)

⇒ Multi-valued attribute cardinalities are problematic

- ↳ Usually better modelled with additional entities linked by one-to-many (or many-to-many) relationships

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 16

University of Toronto Department of Computer Science

## Identifiers (also known as "keys")

- How to uniquely identify instances of an entity?
  - An identifier may be formed by one or more attributes of the entity itself
  - If attributes of an entity are not sufficient to identify instances unambiguously, other entities can be involved in the identification
  - A relationships is identified using identifiers for all the entities it relates
    - E.g. the identifier for the relationship (Person-) Owns(-Car) is a combination of the Person and Car identifiers.

*internal, single-attribute* (points to Registration in AUTOMOBILE)

*external, multi-attribute* (points to Registration, Year, Surname in STUDENT)

*internal, multi-attribute* (points to Surname, FirstName, Address in PERSON)

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 17

University of Toronto Department of Computer Science

## Notes on Identifiers

- Identifiers and cardinality:
  - An identifier can involve one or more attributes, provided that each has (1,1) cardinality
  - An external identifier can involve one or more entities, provided that each is a member of a relationship to which the entity to identify participates with cardinality (1,1)
- Cycles
  - An external identifier can involve an entity that is in its turn identified externally, as long as cycles are not generated;
- Multiple identifiers
  - Each entity must have at least one (internal or external) identifier
  - An entity can have more than one identifier
    - Note: if there is more than one identifier, then the attributes and entities involved in an identification can be optional (minimum cardinality equal to 0).

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 18

University of Toronto Department of Computer Science

## Schema with Identifiers

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 19

University of Toronto Department of Computer Science

## Modeling an Application with Identifiers

- Identifiers provide an important modelling tool
  - E.g. Assume we want a database storing information about lecture meetings.
    - If we use the identifier <course name, day, hour> for the Meeting entity.
      - This says there can only be one meeting at any one time for a given course name, day, hour; we can't have two sections of the same course meeting at the same day+hour.
    - If we use only <course name> as identifier for Meeting.
      - This says that there can only be one meeting per given course name (unreasonable!)
    - If we use <course instructor, room> as identifier for Meeting
      - we are stating that there can only be one meeting for a given instructor+room combination, so an instructor must have all her meetings in different rooms!
    - If we use <course instructor> by itself as identifier for Meeting
      - We are stating that each instructor participates in at most one meeting (unreasonable!)

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al. "Database Systems" McGraw Hill, 1999 20

University of Toronto Department of Computer Science

## Generalizations

⇒ Show "is-a" relationships between entities

⇒ Inheritance:

- ↳ Every instance of a child entity is also an instance of the parent entity
- ↳ Every property of the parent entity (attribute, identifier, relationship or other generalization) is also a property of a child entity

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al., "Database Systems" McGraw Hill, 1999 21

University of Toronto Department of Computer Science

## Types of Generalizations

⇒ Total generalizations:

- ↳ ...every instance of the parent entity is an instance of one of its children
- ↳ Shown as a solid arrow
- ↳ (otherwise: Partial, shown as an unfilled arrow)

⇒ Exclusive generalizations:

- ↳ ...every instance of the parent entity is at most an instance of one of its children
- ↳ (otherwise: overlapping)

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al., "Database Systems" McGraw Hill, 1999 22

University of Toronto Department of Computer Science

## The E-R Meta-Model (as an E-R Diagram)

© Easterbrook 2004 This lecture adapted from chapter 5 of Arzeni et al., "Database Systems" McGraw Hill, 1999 23