COS 597A:
Principles of
Database and Information Systems

Entity-relationship (ER) model

Database Design Phases

1. characterize user needs
2. conceptual design
   - ➢ structure of data
     - * Entity-relationship model
   - – nature of functionality
     - questions
     - modifications
3. implement in database system
   - – logical design
   - – physical design

Entity-relationship model

- ➢ Goal: Capture semantics of information objects
- ➢ Goal: Capture complexity of relationships between objects
- Used first for database modeling but now expanded use

History

- Developed 1976 by Peter Chen
  after relational model
- Chen felt relational model not rich enough
  - – relational model: everything a (mathematical) relation on collection of domains D_i
    - e.g. name from domain of strings
    - Relation subset of $D_1 \times D_2 \times \ldots \times D_k$ (k-ary)
  - – ER model differentiate between objects described by attributes and relationships among objects

ER model basics

- An *entity* is a distinct object in the "real" world
  - – person, book, movie character, disease, …
  - – ***conceptual***
- *Attributes* are basic properties of entities
  - – some defs. don't allow substructure for attributes
    - name ↔ first name, last name
- An *entity* is described/defined by its attributes
  - – entity is *tuple* (or set) *of attributes*
  - – attribute is function: entity set → domain of attribute values

ER model basics II

- A *relationship* is a *tuple* of *entities*
  - – entities are thus related
  - – relationship has some meaning
    - (PU store, "cute tiger" baby shirt)
- A relationship set is a set of relationships of the same type
  - – "same type" = same component types
    - {stores} X {items for sale}
- A relationship can have its own *attributes*
  - – different from entity attributes
  - – descriptive only
    - cannot use to distinguish two tuples of a relationship set
    - (PU store, "cute tiger" baby shirt), "in stock?"

Example

- Entity course with attributes:
  department, number, semester
- Entity student with attributes:
  first name, last name, ID number
- Relationship "take" relating:
  A student to a course

- Both entities and relationships are tuples but at different granularities
- We choose which are entities and which are relationships
- We choose attributes that best describe entities
- We choose semantics of a relationship between entities

Board Example

Entity *Books*: (title, ISBN#, edition, date)

Entity *Authors*:
(name, gender, birth date, place of birth, date of death)

Entity *Publishers*: (name, country, address)

Relationship *written by*: (***books, authors***)

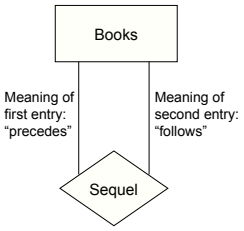
Relationship *published by*:
(***books, publishers***, in print)

Another relationship

Relationship *sequel*:
(*books, books*)

Any book can be on either side of relationship

If sequence longer than 2, middle book appears in tuple on each side



Example tuples for *Lord of the Rings*:
(*Fellowship of the Ring, Twin Towers*)
(*Twin Towers, Return of the King*)

Summary: Entity

- An entity is an element of $A_1 \times A_2 \times \ldots \times A_k$ where A_i is the domain of values of the i^{th} attribute of the entity (for entity with k attributes)
- $A_1 \times A_2 \times \ldots \times A_k$ it the type of the entity
- Set of entities of same type – entity set

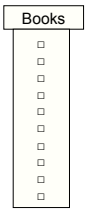
Summary: Relationship

- A relationship is an element of $E_1 \times E_2 \times \ldots \times E_m \times A_1 \times A_2 \times \ldots \times A_p$ where $E_1, \ldots, E_m$ are entity types (for relationship between m entities) and A_i is the domain of values of the i^{th} attribute of the entity for an entity with p attributes
- $E_1 \times E_2 \times \ldots \times E_m \times A_1 \times A_2 \times \ldots \times A_p$ is the type of the relationship
- Set of relationships of same type – relationship set
- Use of attributes is constrained so that for relationships $(e_1, \ldots e_m, a_1, \ldots, a_p)$ and $(e'_1, \ldots e'_m, a'_1, \ldots, a'_p)$, if $(e_1, \ldots e_m) = (e'_1, \ldots e'_m)$ then $(a_1, \ldots, a_p) = (a'_1, \ldots, a'_p)$

Constraints

Identifying entities

Key: a minimal* set of attributes whose values uniquely identify each entity in an entity set



▫ represents a book

* No subset will do the job

Identifying entities

Key: a minimal set of attributes whose values uniquely identify each entity in an entity set

Candidate Key: any key

Primary key: a candidate key defined to be primary by person who defines entity

Superkey: any set of attributes that contains a candidate key

Denote primary key by underlining attributes

Entity *Books*: (title, ISBN#, edition, date)

Entity *Authors*:
(name, gender, birth date, place of birth, date of death)

Entity *Publishers*: (name, country, address)

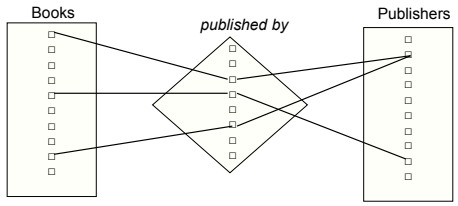
Constraints on entities

- Declaring a candidate key constrains values of attributes
- Example: ISBN# as key
  - No book without an ISBN#
  - No two books with same ISBN#

What about constraints on relationships?

- Constraints are statements about structure

How many entities have link? How many links per entity?

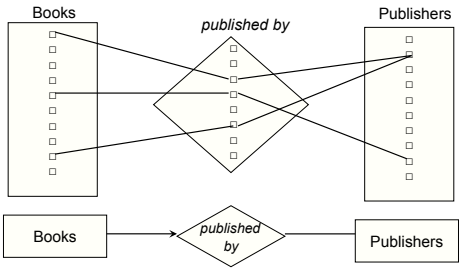


Two major constraints for relationships

- Key constraint on an entity type participating in a relationship type:
  - Each entity of the entity type appears in at most one tuple of the relationship type
    - Equivalently, the value of the key-constrained entity determines the values of the other entities in a relationship tuple
- Total Participation constraint of an entity type participating in a relationship type:
  - Every entity in the entity set of the entity type appears in a tuple of the relationship

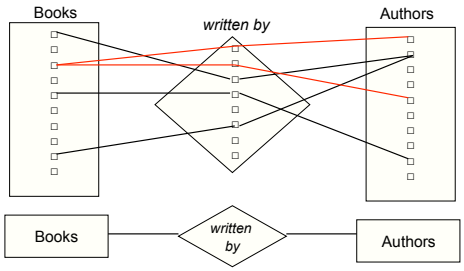
Example: key constraint on books for *published by*? YES

But remember NOT by inspection!!



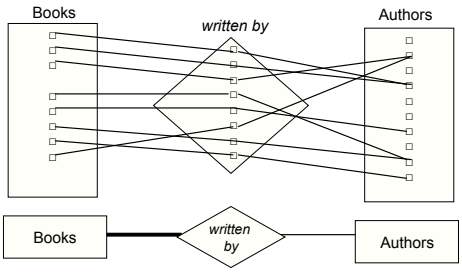
Example: key constraint on books for *written by*? NO

But remember NOT by inspection!!



Example: total participation constraint on books for *written by*? YES

all books authored by someone



Fill in rest of constraints on board diagram

- watch what constraint means
- watch intended use of database

Multiple Key Constraints

- can have binary relationship where both entity sets satisfy key constraint
  - i.e. both entity sets keys for relationship

Total participation constraints?

Multiple Key Constraints

- can have binary relationship where both entity sets satisfy key constraint
  - i.e. both entity sets keys for relationship

Constraints cannot denote in basic ER model

- Domain attribute constraints within entity
  - Need to test values of attributes not simply membership properties in sets
  - Example:
    Attribute *NJ driver*: yes/no flag
    Attribute *age*: number
    Constraint "if age <17 then *NJ driver* == "no"

Constraints cannot denote in basic ER model

- Functional constraints

  Example:

  person entity with 6 attributes:
  first name, last name, street address, state, area code, 7-digit phone number.

  Constraint:

  if *area code* of person 1 = *area code* of person 2
  then *state* of person 1 = *state* of person 2

  Equivalently, *area code* determines *state*

Constraints cannot denote in basic ER model

- Functional constraints

  General form:

  Let A and B be subsets of attributes for an entity type.
  For any entities e_j and e_k of the type:

  If the values of attributes in set A for tuple e_j equal the values of attributes in set A for tuple e_k

  Then the values of attributes in set B for tuple e_j equal the values of attributes in set B for tuple e_k

Constraints cannot denote in basic ER model

- Functional constraints

  More complicated example:

  customer entity with 8 attributes:
  height, weight, arm length, leg length, color preference, jacket size, pant size, shirt size

  Constraint:

  Height, weight, arm length determine *shirt size*
  Height, weight, leg length determine *pant size*

n-ary relationships, n>2

Example- scheme:
tutorial offering = Tutorials X Instructors
X Conferences

- Do with binary relationships?
- Capturing constraints. Careful!

Richness of ER model

Richness of ER model

1. Tuples can be composed of tuples – hierarchy:
   - Relationship tuples composed of entity tuples
   - By aggregation, relationship tuples can contain relationship tuples
2. Inheritance for entities:
   - An entity type can be further refined

Aggregation

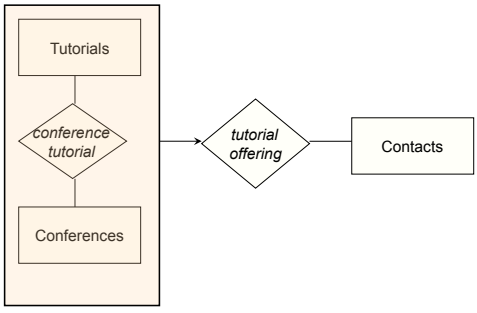
tutorial offering = Tutorials X Conferences X Contacts

tutorial offering = [Tutorials X Conferences] X Contacts

because wanted Tutorials X Conferences as key

express constraint

Diagram

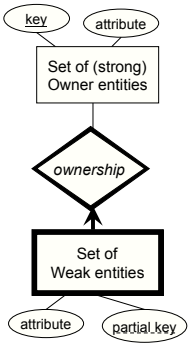


Aggregation

Examples when need for semantics?

to board 

Weak entity



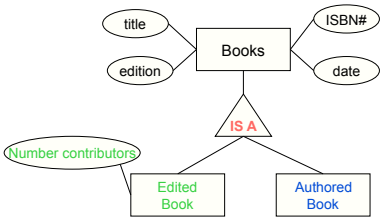
Weak entity only completely identified when related to owner entity

- Weak entity set has no key
- Weak entity has partial key (aka discriminator)
- Owner key + partial key = key for weak entity
- Weak entity fully participates in ownership relationship
- Weak entity is key for *ownership* relationship
  - key constraint on weak entity in ownership relationship

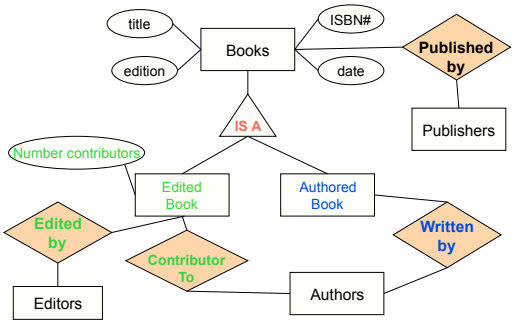
Richness of ER model

1. Tuples can be composed of tuples – hierarchy:
   - Relationship tuples composed of entity tuples
   - By aggregation, relationship tuples can contain relationship tuples
2. Inheritance for entities:
   - An entity type can be further refined

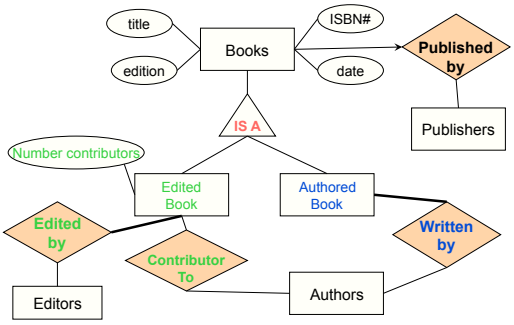
Inheritance hierarchy for entities



Using hierarchy with relationships



Key and total participation constraints possible

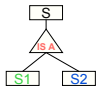


We just declared constraints:

1. A book is published by at most one publisher (key constraint)
2. Every authored book is written by some author (total participation constraint)
3. Every edited book is edited by some editor (total participation constraint)

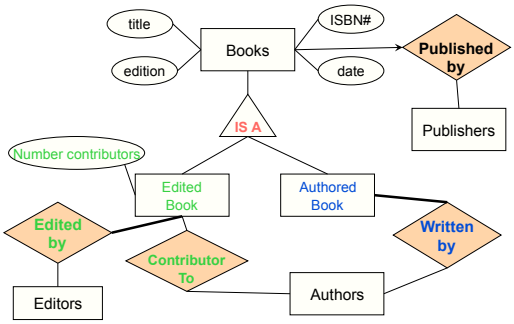
Constraints on "is a" decomposition

- Covering
  If union of subclasses = superclass
  S= S1 U S2



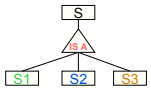
- Overlapping
  If subclasses can overlap
  S1 ∩ S2 may ≠ Ø

Overlapping? Covering?

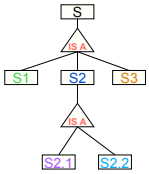


GENERAL "IS A"

- Can have > 2 subclasses
  for one "IS A"

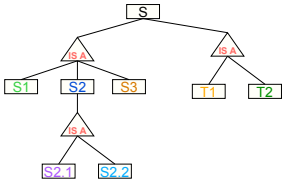


- Subclasses can be further
  refined with "IS A"
  – Deep hierarchy possible

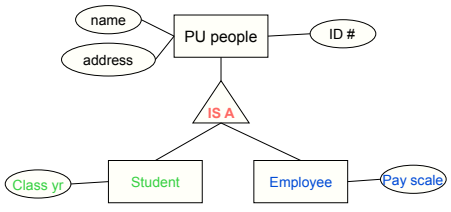


GENERAL "IS A"

- Can have more that one
  "IS A" refinement for an
  entity



Another example



Summary

- ER model is rich model for describing objects and their relationships
- Can capture:
  - constraints
  - Aggregations of objects into more complex objects
  - Different decompositions of entity types
  - Deep hierarchy of decomposition
- Model used as first effort to organize information and make precise what is needed