E/R Modeling

Examples

Overview of Database Design

* *Conceptual design*: *ER Model* is used at this stage, which is then translated to a relational schema.
* *Schema Refinement*: *(Normalization)* Check relational schema for redundancies and related anomalies.
* *Physical Database Design and Tuning*: Consider typical workloads and further refine the database design.

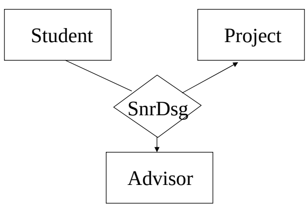
Conceptual Design

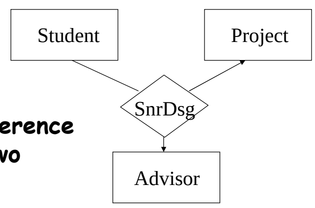
- What are the *entities* and *relationships* in the enterprise?
- What information about these entities and relationships should we store in the database?
- What are the *integrity constraints* that hold?
- Represent this information pictorially in *ER diagrams*, then map ER diagram into a relational schema.

ER Diagrams -- the basics

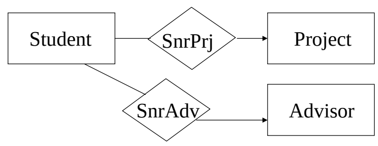
- Attributes: things typically used as column names, e.g. Name, Age, Height. They are drawn as ovals: Name
- Entities: real world "objects", e.g. Students, Courses, Routes, Climbers. Entity sets are drawn as rectangles: Courses
- Relationships: relationships between entities, e.g. a student enrolls in a course, a climber climbs a route, etc. Relationship sets are drawn as diamonds: Enrolls

N-ary relationships



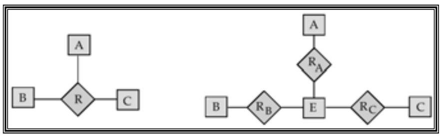


What is the difference between these two representations?



Converting Non-Binary Relationships to Binary Form

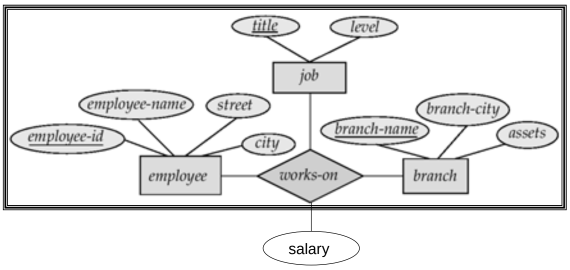
- In general, any non-binary relationship can be represented using binary relationships by creating an artificial entity set.
  - Replace *R* between entity sets A, B and C by an entity set *E*, and three relationship sets:
    1. R_A, relating *E* and *A*
    2. R_B, relating *E* and *B*
    3. R_C, relating *E* and *C*
  - Create a special identifying attribute for *E*
  - Add any attributes of *R* to *E*
  - For each relationship (a_i , b_i , c_i) in *R,* create
    1. a new entity e_i in the entity set *E*
    2. add (e_i , a_i) to R_A
    3. add (e_i , b_i) to R_B
    4. add (e_i , c_i) to R_C



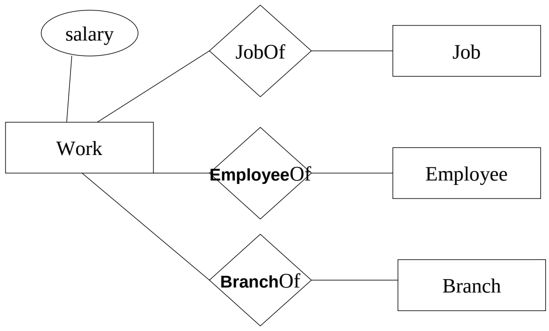
Converting Non-Binary Relationships (Cont.)

- Also need to translate constraints
  - Translating all constraints may not be possible
  - There may be instances in the translated schema that cannot correspond to any instance of *R*
    - *Exercise: add constraints to the relationships R_A, R_B and R_C to ensure that a newly created entity corresponds to exactly one entity in each of entity sets A, B and C*
  - We can avoid creating an identifying attribute by making E a weak entity set (described shortly) identified by the three relationship sets

E-R Diagram with a Ternary Relationship



Converting Multi-way Relationships to Binary

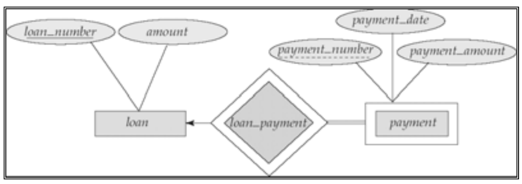


Weak Entity Sets

- An entity set that does not have a primary key is referred to as a **weak entity set**.
- The existence of a weak entity set depends on the existence of a **identifying entity set**
  - it must relate to the identifying entity set via a total, one-to-many relationship set from the identifying to the weak entity set
  - Identifying relationship depicted using a double diamond
- 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.
- The primary key of a weak entity set is formed by the primary key of the strong entity set on which the weak entity set is existence dependent, plus the weak entity set's discriminator.

Weak Entity Sets (Cont.)

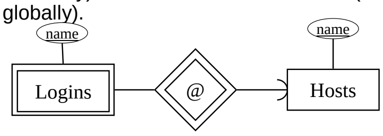
- We depict a weak entity set by double rectangles.
- We underline the discriminator of a weak entity set with a dashed line.
- payment_number – discriminator of the *payment* entity set
- Primary key for *payment* – (*loan_number, payment_number*)



Example: Logins (Email Addresses)

Login name = user name + host name, e.g., `ark@soe.ucsc.edu`.

- A "login" entity corresponds to a user name on a particular host, but the passwd table doesn't record the host, just the user name, e.g., `ark`.
- Key for a login = the user name at the host (which is unique for that host only) + the IP address of the host (which is unique globally).

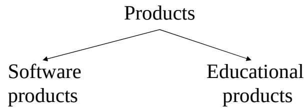


- Design issue: Under what circumstances could we simply make login-name and host-name be attributes of logins, and dispense with the weak E.S.?

Modeling Subclasses

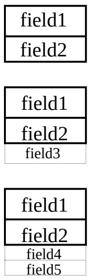
The world is inherently hierarchical. Some entities are special cases of others

- We need a notion of subclass.
- This is supported naturally in object-oriented formalisms.

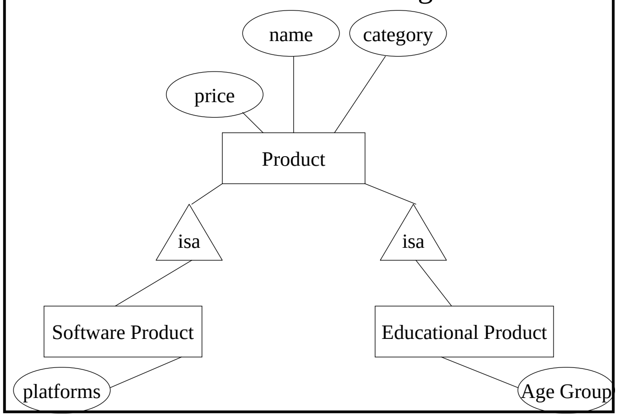


Understanding Subclasses

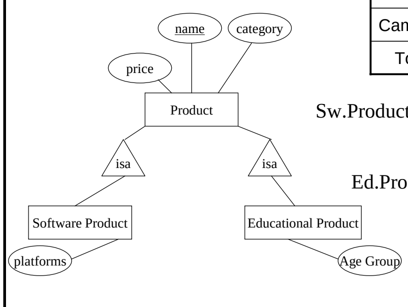
- Think in terms of records:
  - Product
  - SoftwareProduct
  - EducationalProduct



Subclasses in E/R Diagrams



Subclasses to Relations



Product

Name	Price	Category
Gizmo	99	gadget
Camera	49	photo
Toy	39	gadget

Sw.Product

Name	platforms
Gizmo	unix

Ed.Product

Name	Age Group
Gizmo	todler
Toy	retired