



Lecture 13 of 42

Entity-Relational Data Modeling Notes: PS3 Notes, E-R Applications (Projects)

Friday, 22 September 2006

William H. Hsu
Department of Computing and Information Sciences, KSU

KSOL course page: <http://snipurl.com/va60>
Course web site: <http://www.kddresearch.org/Courses/Fall-2006/CIS560>
Instructor home page: <http://www.cis.ksu.edu/~bhsu>

Reading for Next Class:
Rest of Chapter 6, Silberschatz *et al.*, 5th edition



Modeling

- A *database* can be modeled as:
 - * a collection of entities,
 - * relationship among entities.
- An **entity** is an object that exists and is distinguishable from other objects.
 - * Example: specific person, company, event, plant
- Entities have *attributes*
 - * Example: people have *names* and *addresses*
- An **entity set** is a set of entities of the same type that share the same properties.
 - * Example: set of all persons, companies, trees, holidays





Entity Sets *customer* and *loan*

customer_id	customer_name	customer_street	customer_city	loan_number	amount
321-12-3123	Jones	Main	Harrison	L-17	1000
019-28-3746	Smith	North	Rye	L-23	2000
677-89-9011	Hayes	Main	Harrison	L-15	1500
555-55-5555	Jackson	Dupont	Woodside	L-14	1500
244-66-8800	Curry	North	Rye	L-19	500
963-96-3963	Williams	Nassau	Princeton	L-11	900
335-57-7991	Adams	Spring	Pittsfield	L-16	1300

customer *loan*



Relationship Sets

- A **relationship** is an association among several entities

Example:

Hayes depositor A-102
customer entity relationship set *account* entity

- A **relationship set** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

$$\{(e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n\}$$

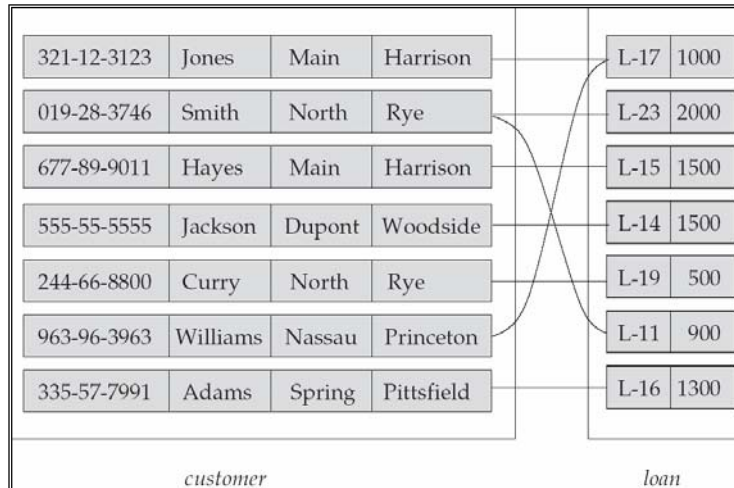
where (e_1, e_2, \dots, e_n) is a relationship

* Example:

$(\text{Hayes}, \text{A-102}) \in \text{depositor}$

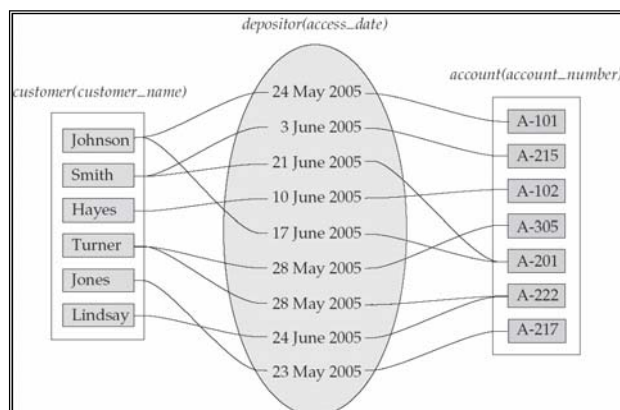


Relationship Set *borrower*



Relationship Sets (Cont.)

- An **attribute** can also be property of a relationship set.
- For instance, the *depositor* relationship set between entity sets *customer* and *account* may have the attribute *access-date*





Degree of a Relationship Set

- Refers to number of entity sets that participate in a relationship set.
- Relationship sets that involve two entity sets are **binary** (or degree two). Generally, most relationship sets in a database system are binary.
- Relationship sets may involve more than two entity sets.
 - ▶ Example: Suppose employees of a bank may have jobs (responsibilities) at multiple branches, with different jobs at different branches. Then there is a ternary relationship set between entity sets *employee*, *job*, and *branch*
- Relationships between more than two entity sets are rare. Most relationships are binary. (More on this later.)



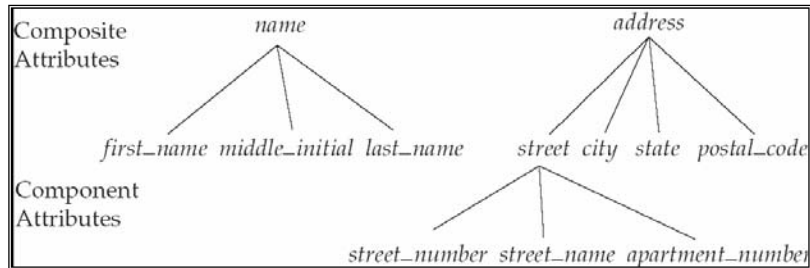
Attributes

- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set.
Example:
$$\text{customer} = (\text{customer_id}, \text{customer_name}, \text{customer_street}, \text{customer_city})$$
$$\text{loan} = (\text{loan_number}, \text{amount})$$
- **Domain** – the set of permitted values for each attribute
- Attribute types:
 - * *Simple and composite attributes.*
 - * *Single-valued and multi-valued attributes*
 - ⇒ Example: multivalued attribute: *phone_numbers*
 - * *Derived attributes*
 - ⇒ Can be computed from other attributes
 - ⇒ Example: age, given date_of_birth





Composite Attributes



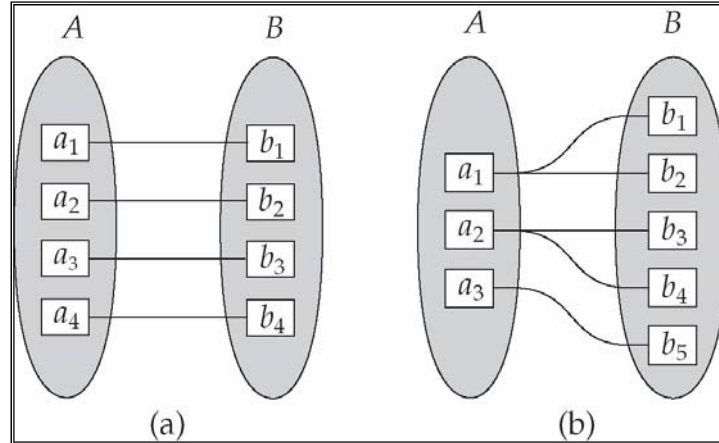
Mapping Cardinality Constraints

- Express the number of entities to which another entity can be associated via a relationship set.
- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the following types:
 - * One to one
 - * One to many
 - * Many to one
 - * Many to many





Mapping Cardinalities



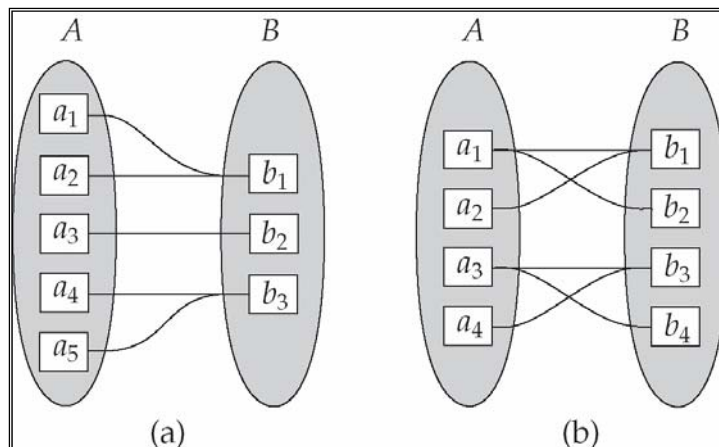
One to one

One to many

Note: Some elements in A and B may not be mapped to any elements in the other set



Mapping Cardinalities



Many to one

Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set





Keys

- A **super key** of an entity set is a set of one or more attributes whose values uniquely determine each entity.
- A **candidate key** of an entity set is a minimal super key
 - * *Customer_id* is candidate key of *customer*
 - * *account_number* is candidate key of *account*
- Although several candidate keys may exist, one of the candidate keys is selected to be the **primary key**.



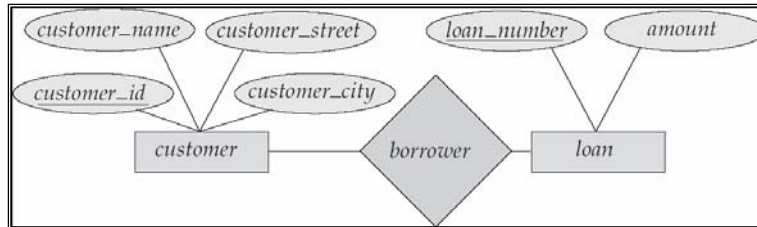
Keys for Relationship Sets

- The combination of primary keys of the participating entity sets forms a super key of a relationship set.
 - * *(customer_id, account_number)* is the super key of *depositor*
 - * **NOTE:** *this means a pair of entity sets can have at most one relationship in a particular relationship set.*
 - ⇒ Example: if we wish to track all *access_dates* to each account by each customer, we cannot assume a relationship for each access. We can use a multivalued attribute though
- Must consider the mapping cardinality of the relationship set when deciding the what are the candidate keys
- Need to consider semantics of relationship set in selecting the *primary key* in case of more than one candidate key





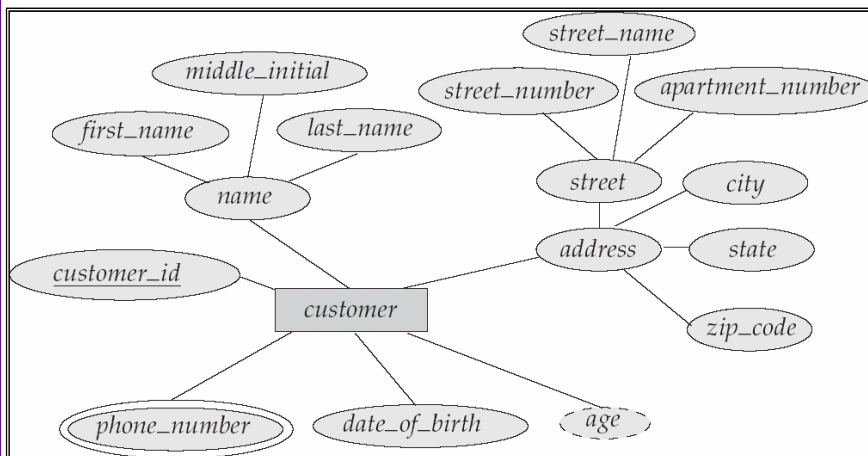
E-R Diagrams



- Rectangles represent entity sets.
- Diamonds represent relationship sets.
- Lines link attributes to entity sets and entity sets to relationship sets.
- Ellipses represent attributes
 - Double ellipses represent multivalued attributes.
 - Dashed ellipses denote derived attributes.
- Underline indicates primary key attributes (will study later)

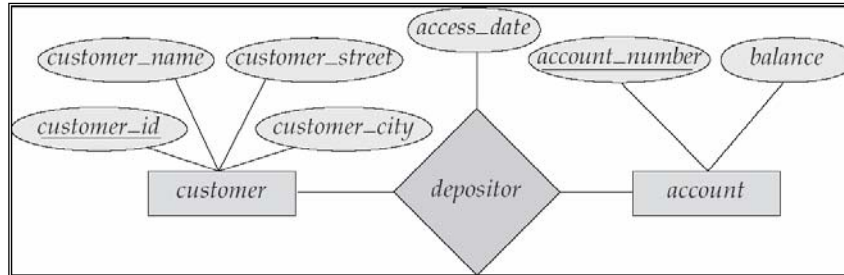


E-R Diagram With Composite, Multivalued, and Derived Attributes



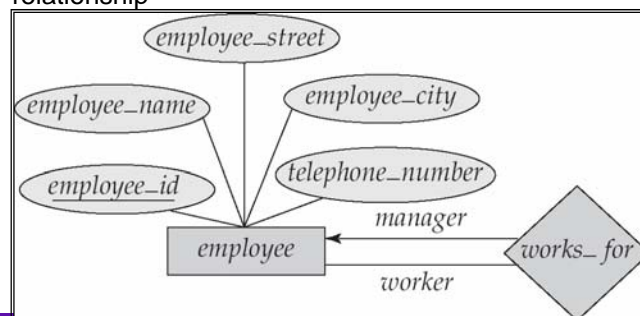


Relationship Sets with Attributes



Roles

- Entity sets of a relationship need not be distinct
- The labels “manager” and “worker” are called **roles**; they specify how employee entities interact via the works_for relationship set.
- Roles are indicated in E-R diagrams by labeling the lines that connect diamonds to rectangles.
- Role labels are optional, and are used to clarify semantics of the relationship





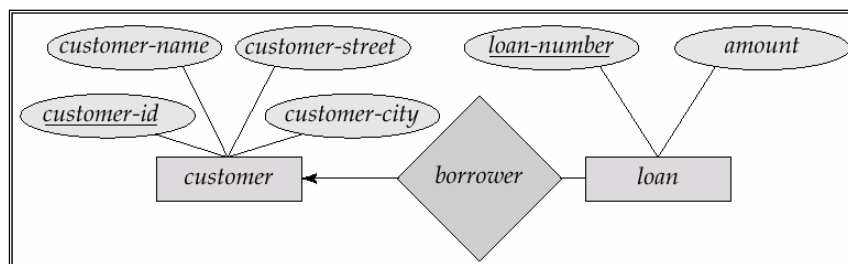
Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (\rightarrow), signifying “one,” or an undirected line ($-$), signifying “many,” between the relationship set and the entity set.
- One-to-one relationship:
 - * A customer is associated with at most one loan via the relationship *borrower*
 - * A loan is associated with at most one customer via *borrower*



One-To-Many Relationship

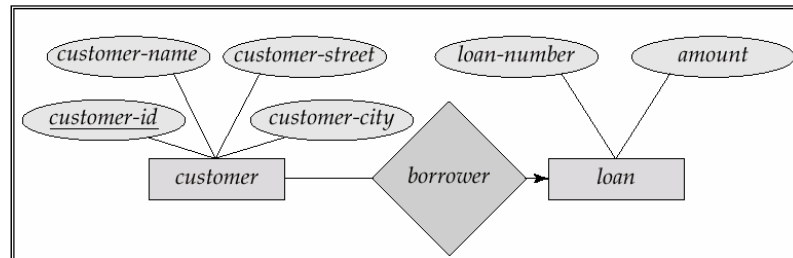
- In the one-to-many relationship a loan is associated with at most one customer via *borrower*, a customer is associated with several (including 0) loans via *borrower*





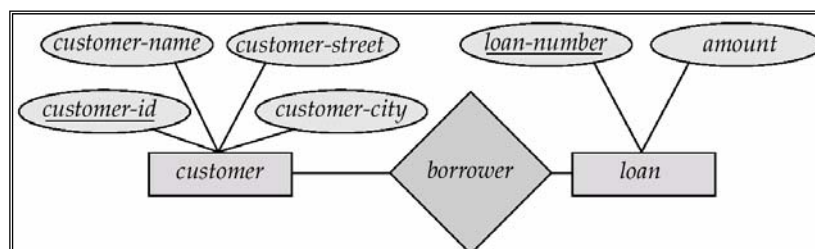
Many-To-One Relationships

- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*



Many-To-Many Relationship

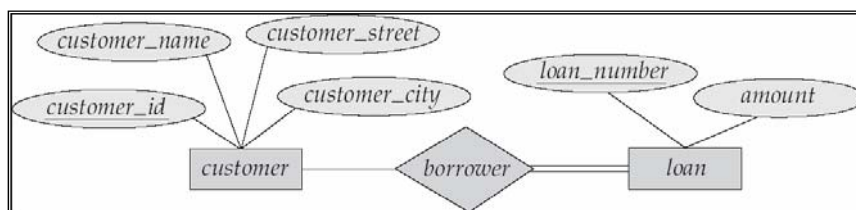
- A customer is associated with several (possibly 0) loans via *borrower*
- A loan is associated with several (possibly 0) customers via *borrower*





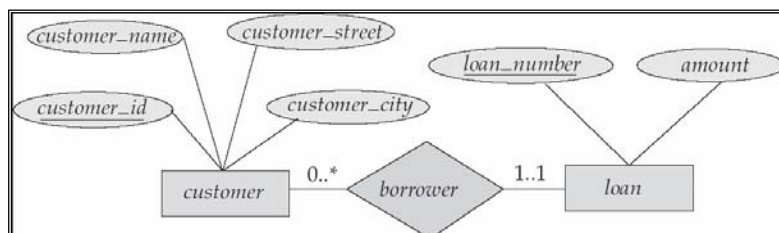
Participation of an Entity Set in a Relationship Set

- Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
 - E.g. participation of loan in borrower is total
 - ▶ every loan must have a customer associated to it via borrower
- Partial participation: some entities may not participate in any relationship in the relationship set
 - Example: participation of customer in borrower is partial



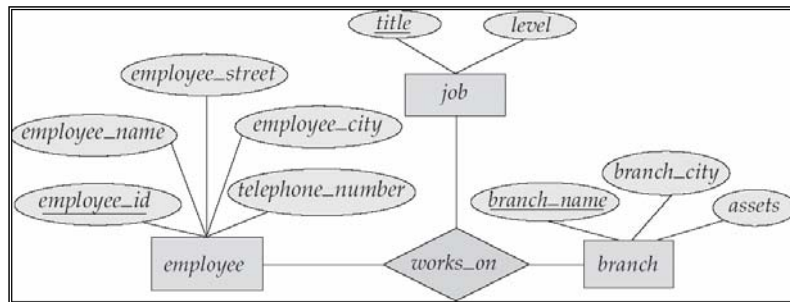
Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints





E-R Diagram with a Ternary Relationship



Cardinality Constraints on Ternary Relationship

- We allow at most one arrow out of a ternary (or greater degree) relationship to indicate a cardinality constraint
- E.g. an arrow from *works_on* to *job* indicates each employee works on at most one job at any branch.
- If there is more than one arrow, there are two ways of defining the meaning.
 - * E.g a ternary relationship *R* between *A*, *B* and *C* with arrows to *B* and *C* could mean
 1. each *A* entity is associated with a unique entity from *B* and *C* or
 2. each pair of entities from (*A*, *B*) is associated with a unique *C* entity, and each pair (*A*, *C*) is associated with a unique *B*
 - * Each alternative has been used in different formalisms
 - * To avoid confusion we outlaw more than one arrow



Design Issues

- **Use of entity sets vs. attributes**
Choice mainly depends on the structure of the enterprise being modeled, and on the semantics associated with the attribute in question.
- **Use of entity sets vs. relationship sets**
Possible guideline is to designate a relationship set to describe an action that occurs between entities
- **Binary versus n-ary relationship sets**
Although it is possible to replace any nonbinary (n -ary, for $n > 2$) relationship set by a number of distinct binary relationship sets, a n -ary relationship set shows more clearly that several entities participate in a single relationship.
- **Placement of relationship attributes**



Binary Vs. Non-Binary Relationships

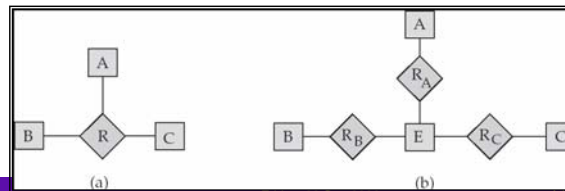
- Some relationships that appear to be non-binary may be better represented using binary relationships
 - * E.g. A ternary relationship *parents*, relating a child to his/her father and mother, is best replaced by two binary relationships, *father* and *mother*
 - ⇒ Using two binary relationships allows partial information (e.g. only mother being know)
 - * But there are some relationships that are naturally non-binary
 - ⇒ Example: *works_on*





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



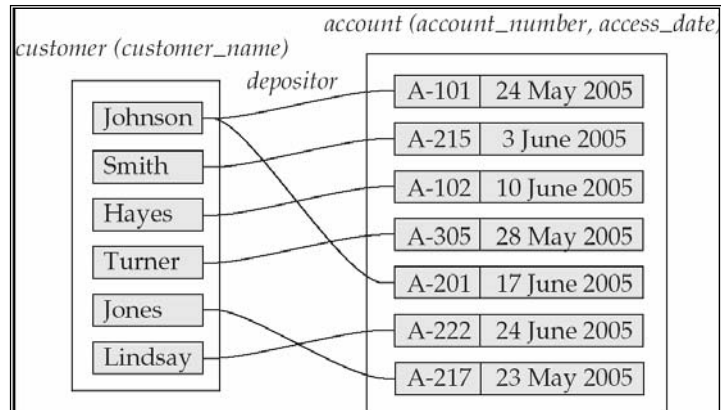
Converting Non-Binary Relationships (Cont.)

- 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



Mapping Cardinalities affect ER Design

- Can make access-date an attribute of account, instead of a relationship attribute, if each account can have only one customer
 - That is, the relationship from account to customer is many to one, or equivalently, customer to account is one to many



ER Domains: Term Projects for 560



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 an **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.

