Entity-Relationship Modelling

- Entities
- Attributes
- Relationships
- Mapping Cardinality
- Keys
- Reduction of an E-R Diagram to Tables

Entity Sets

- A “enterprise” can be modeled as a collection of:
  - entities, and
  - relationships among those entities.
- An *entity* is an object that is distinguishable from other objects.
  - A specific person, company, automobile, etc.
- Entities have *attributes:*
  - People have *names* and *addresses*
  - An entity and its’ attributes are represented by a tuple

    (342-97-4873, Smith, Main, Orlando)
- An *entity set* is a set of entities of the same type, i.e., that share the same properties and attributes.
  - Set of all students, set of all companies, set of all automobiles

Entity Sets customer *and* loan

- Entity sets:
  - The set of all *customers* of the Bank
  - The set of all *loans* at the bank

- Entities:
  - Customers of the Bank – Bob Jones, Sue Smith, Mark Hayes, etc.
  - Loans at the Bank – L17, L15, L23, etc.

- Attributes*:*
  - Bob Jones has an ID# (321-12-3231), a street address (475 Main Street), a city of residence (Orlando), and a last name (Jones).
  - Loan L17 has an amount ($4537), a date when the loan was taken out (12/15/2009), and a loan number (L17).

Attributes

- The set of permitted values for an attribute is call the *domain* of that attribute.

- Attributes can be:
  - Simple (i.e., atomic) – height in inches, weight in ounces, last-name
  - Composite – name, address, date-of-birth
  - Single-valued – height in inches, date-of-birth, name (any of the above)
  - Multi-valued – phone-numbers, dependents, hobbies
  - Derived – “age” is derived, or rather, computed from “date-of-birth”

Attributes, Cont.

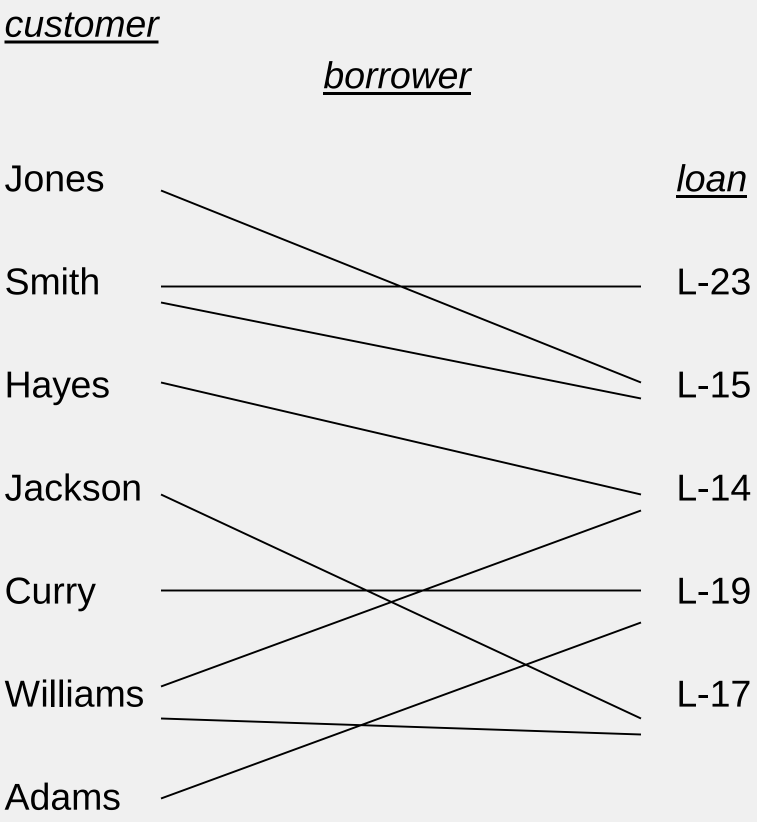
- Keep in mind that modeling is NOT design!
  - during modeling we are focused on what the relevant data is, and not whether or how it will be stored in the database.
  - age vs. date-of-birth
- This approach is:
  - consistent with most text-books
  - somewhat *inconsistent* with industry

Relationship Sets

- A *relationship* is an association among two (or more) entities
  - *Hayes* is a *depositor* for account *A-102*
  - The relationship is denoted by a tuple (Hayes, A-102)
- A *relationship set* is a set of relationships, all of the same type.

Attributes, Cont.

- Relationships can be visualized graphically:



customer

borrower

loan

Jones

Smith

Hayes

Jackson

Curry

Williams

Adams

L-23

L-15

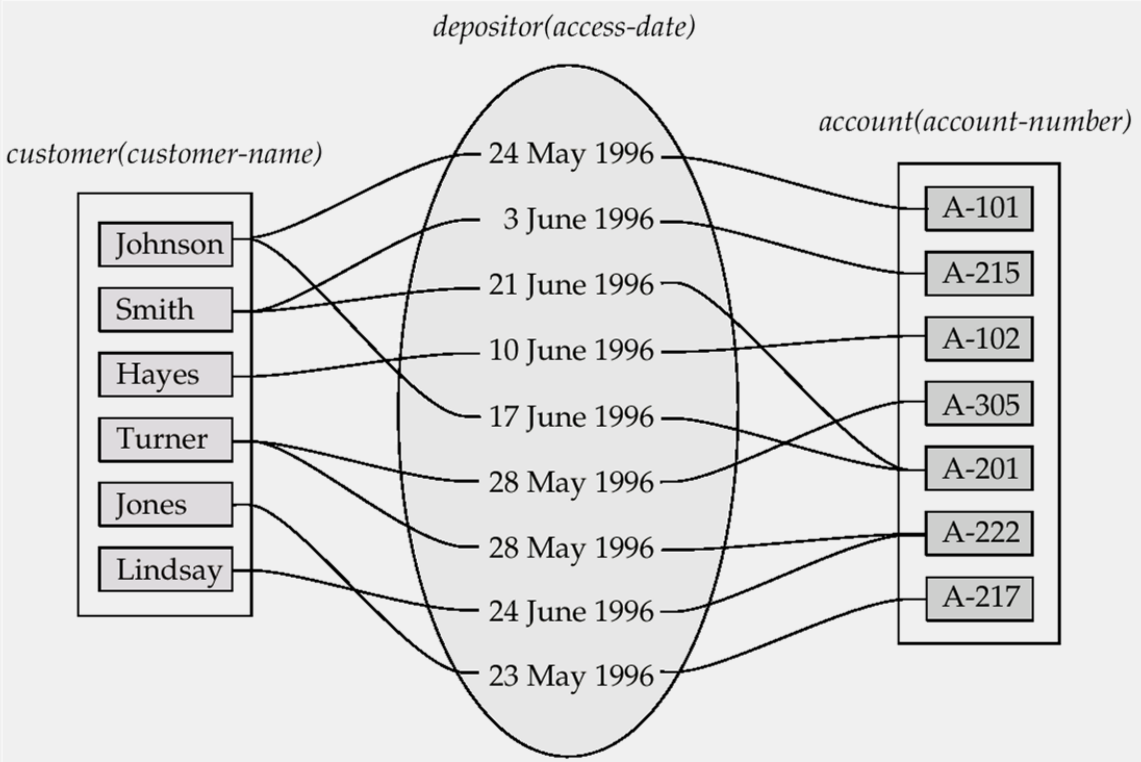
L-14

L-19

L-17

Relationship Sets (Cont.)

- An attribute can also be property of a relationship set.



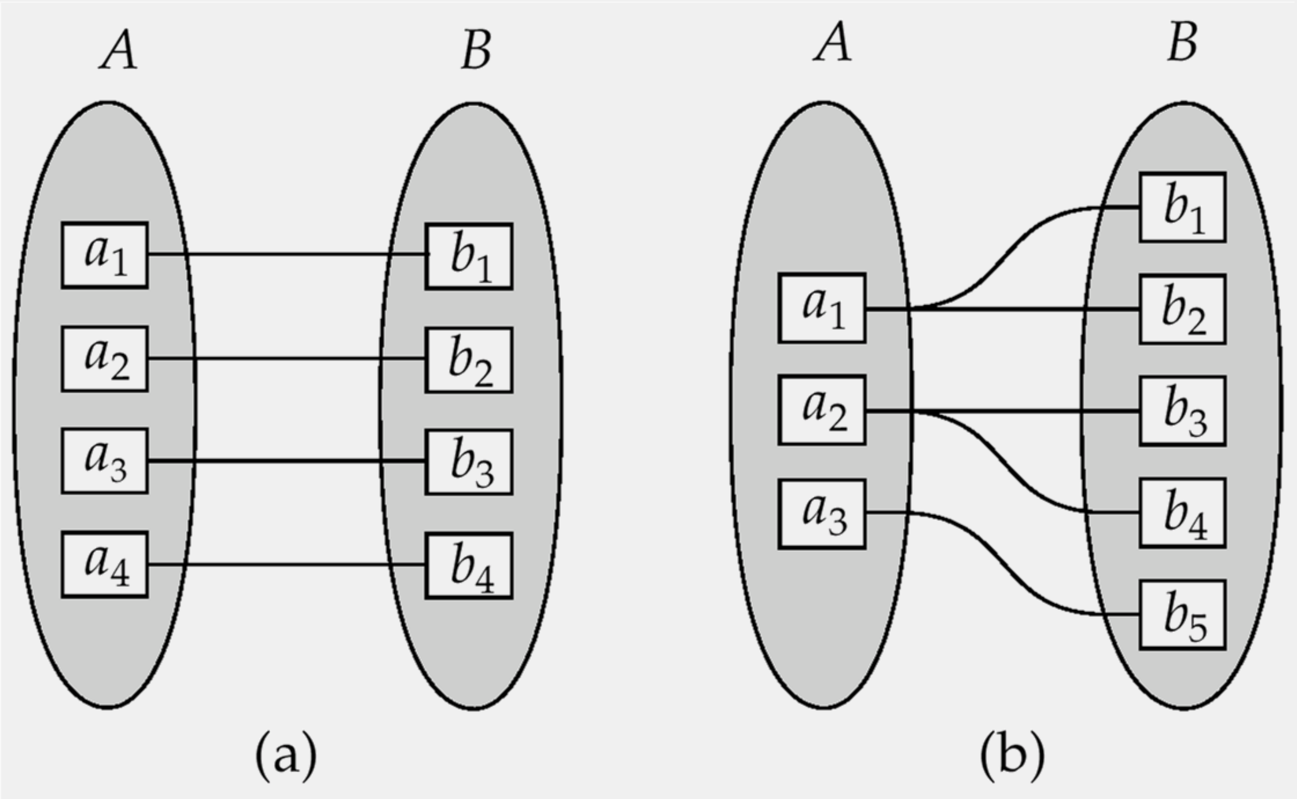
Relationship Sets with Attributes, Cont.

- Another example of a relationship set having attributes:
  - Entities: *Student* and *Course*
  - Relationship: *Has-Taken*
- Where does the attribute *grade* go?

Mapping Cardinalities

- The *mapping cardinality* of a relationship set expresses the number of entities to which one entity can be associated via the relationship set.
  - One to one – US residents & social security #’s
  - One to many – academic advisors (assuming at most one major)
  - Many to one – same as one-to-many
  - Many to many – depositors

Mapping Cardinalities

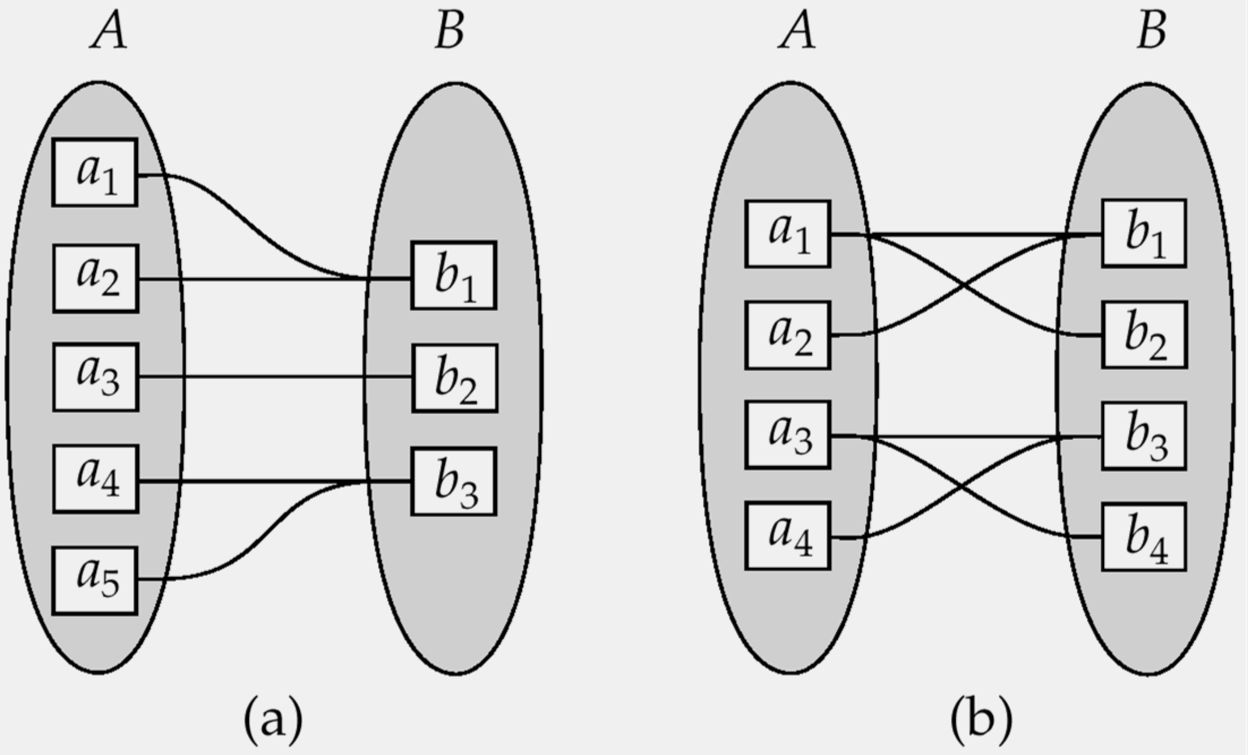


One to one

One to many

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

Mapping Cardinalities



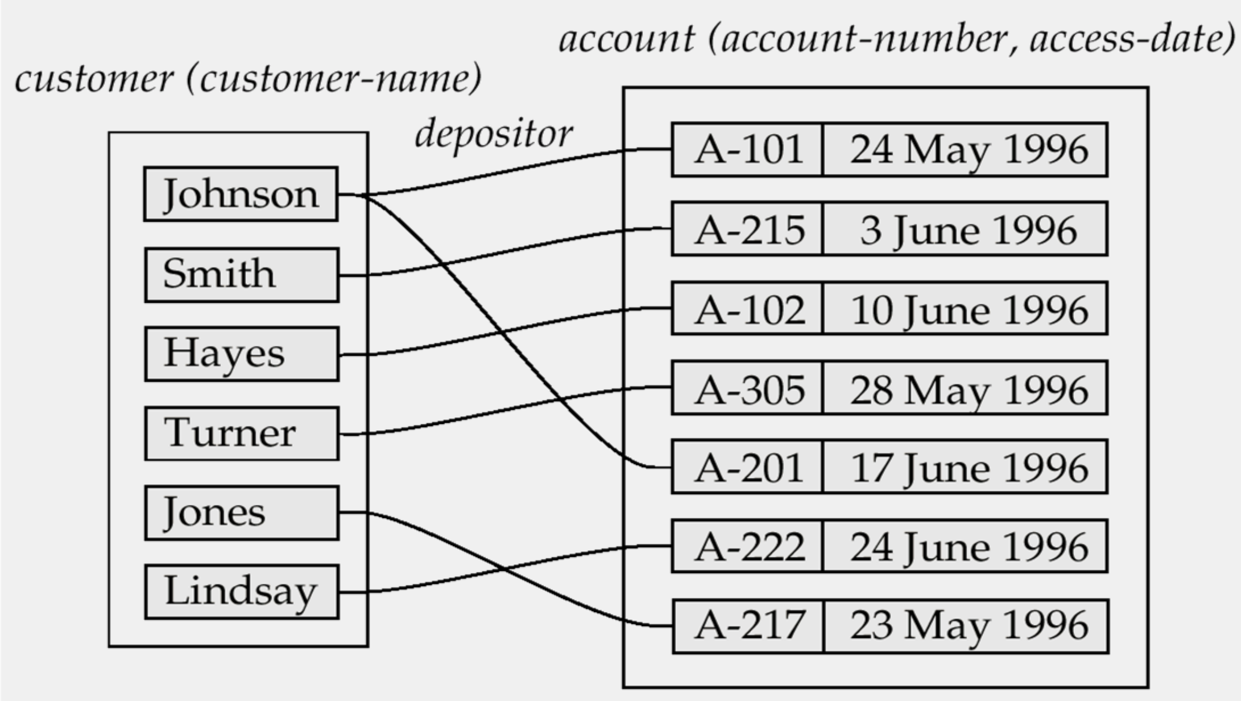
Many to one

Many to many

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

Mapping Cardinalities affect Attribute Placement

- In the banking enterprise, *access-date* could be an attribute of account instead of a relationship attribute if each account can have only one customer, i.e., if the relationship is one-to-many.



customer (customer-name)

depositor

account (account-number, access-date)

Johnson
Smith
Hayes
Turner
Jones
Lindsay

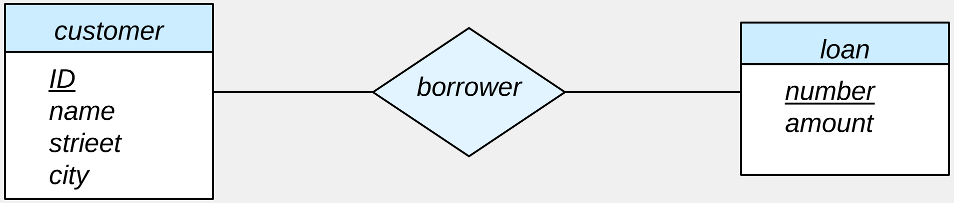
A-101	24 May 1996
A-215	3 June 1996
A-102	10 June 1996
A-305	28 May 1996
A-201	17 June 1996
A-222	24 June 1996
A-217	23 May 1996

E-R Diagrams

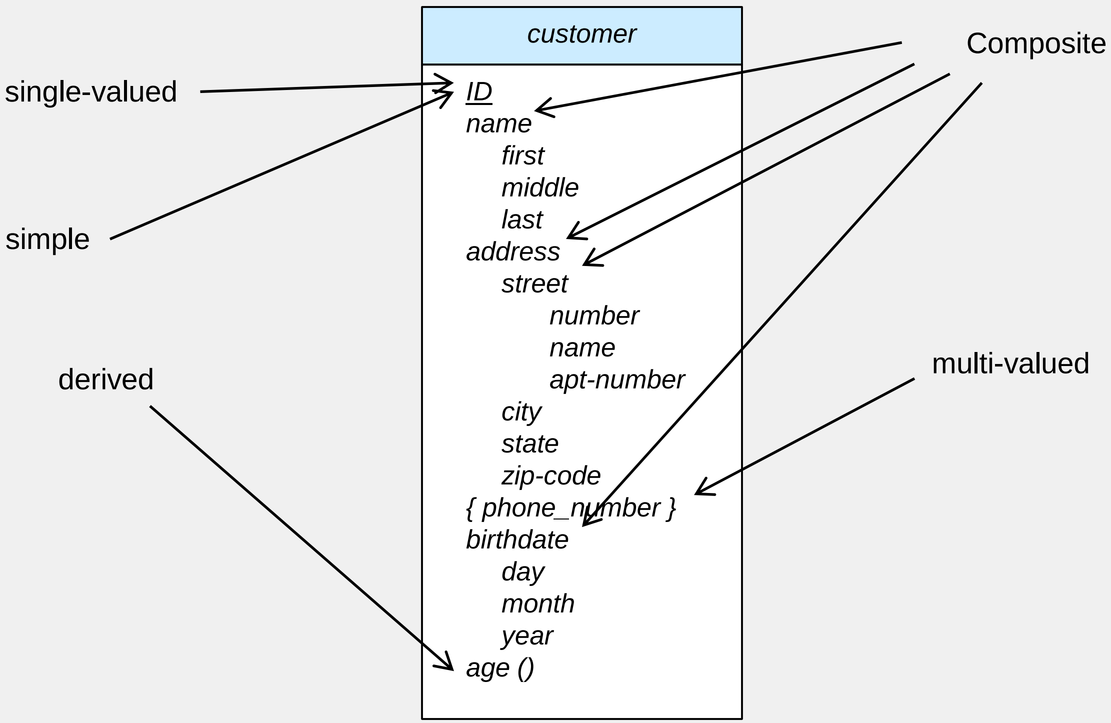
- Several ER diagramming techniques have been proposed over the years:
  - Chen's notation - 1976
  - IDEF1X (NIST) - 1993
  - Crow's feet (Barker, Palmer, Ellis, et al.) – 1981
  - UML (Booch, Jacobson and Rumbaugh) – 1990's
  - Others…
- The authors current version is somewhat UML-like, but previously used Chen's notation.

E-R Diagrams

- Rectangles - entity sets
- Diamonds - relationship sets
- Lines - connect entity sets to relationship sets.
- Underlined Attributes – primary key attributes

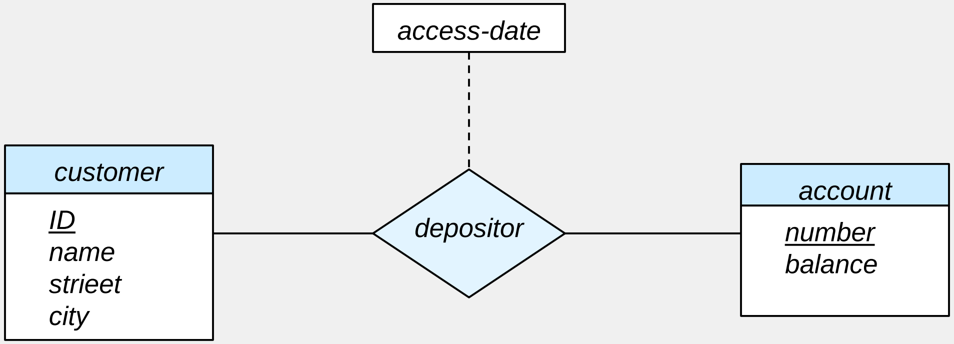


E-R Diagram with Composite, Multi-valued, and Derived Attributes



- Notes:
  - An ER diagram is typically accompanied by a document that defines all the terms
  - Much harder to do than it appears (e.g., what is an “orbit” for a satellite?)
  - In many applications the terms are much more ambiguous (e.g., function designators)

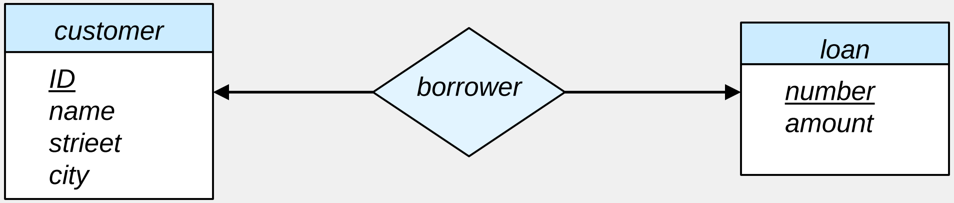
Relationship Sets with Attributes



Cardinality Constraints

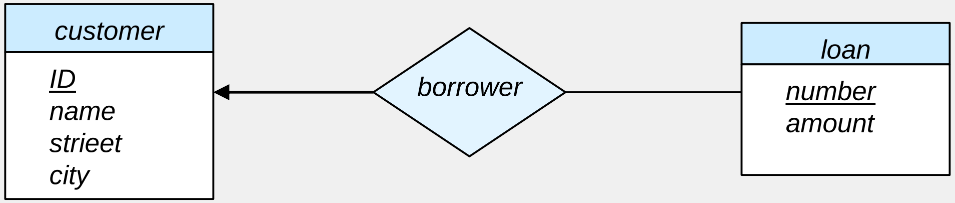
- Mapping cardinality is indicated by drawing a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship and the entity.

- One-to-one relationship:
  - A customer would be associated with at most one loan
  - A loan would be associated with at most one customer



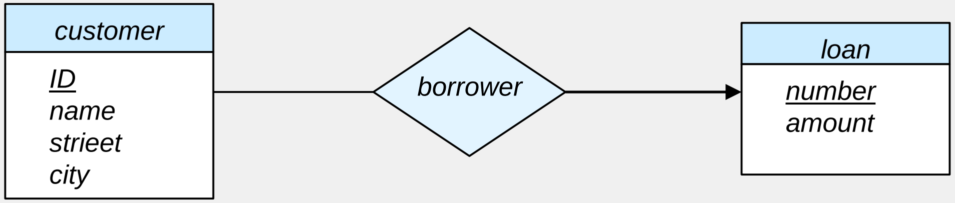
One-To-Many Relationship

- One-to-many relationship - a customer is associated with zero or more one loans, and a loan is associated with at most one customer.



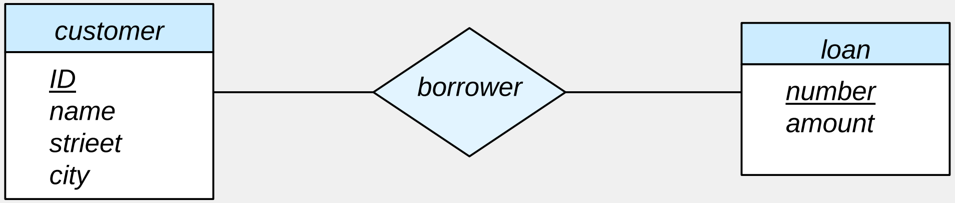
Many-To-One Relationships

- Many-to-one relationship - a loan is associated with zero or more customers, and a customer is associated with at most one loan.



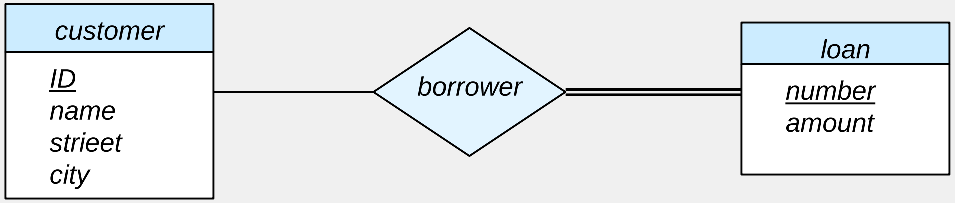
Many-To-Many Relationship

- Many-to-many relationship - a customer is associated with zero or more loans, and a loan is associated with zero of more customers.



Participation of an Entity Set in a Relationship Set

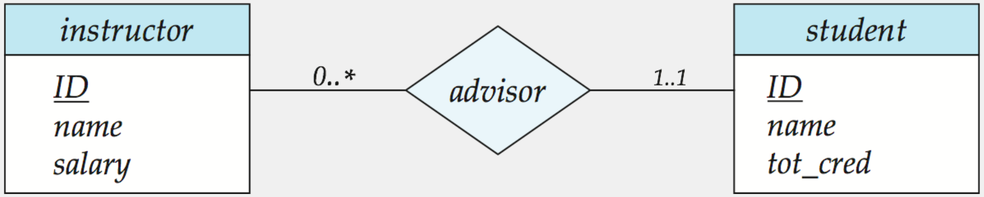
- Total participation - every entity in an entity set must participate in the relationship set; indicated by a double-line and a double-diamond.



- If participation in a relationship is optional for some entities then that entity set is said to have *partial participation* in the relationship.

Alternative Notation for Cardinality Limits

- Another version of the notation uses specific cardinality limits:



Keys

- A *super key* of an entity set is a set of one or more attributes that uniquely identify each entity in the entity set.
- 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*.
  - All others are referred to as *secondary keys*
  - Book says selection of the primary key is arbitrary, but this is not true.
- Later we will also discuss *foreign keys* and *search keys*.

Other Examples of Keys

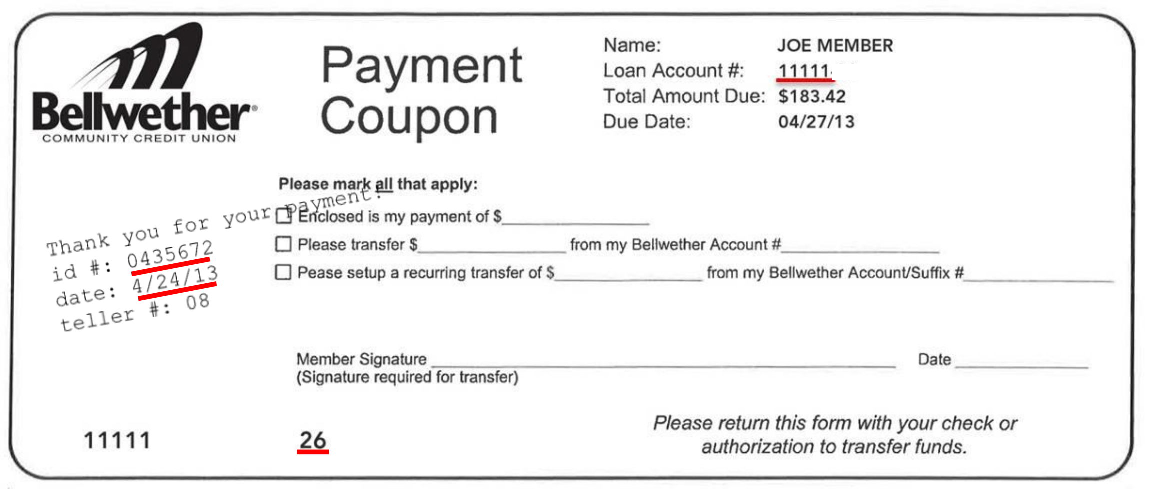
- Student = (SS#, Name, Date-of-Birth, Status)

SS#	- Super key and candidate key
SS#, Name, DOB	- Super key, but not candidate key
SS#, Status	- Super key, but not candidate key
Name	- Neither

Other Examples of Keys

- Payment = (Payment#, Loan#, Date-Made, ID#)

ID#	- Super key, and candidate key
Payment#, Loan#	- Super key, and candidate key
ID#, Date-Made	- Super key, but not candidate key
Date-Made	- Neither



Bellwether
COMMUNITY CREDIT UNION

Payment Coupon

Name: JOE MEMBER
Loan Account #: 11111
Total Amount Due: $183.42
Due Date: 04/27/13

Please mark all that apply:
□ Enclosed is my payment of $______________
□ Please transfer $______________ from my Bellwether Account #______________
□ Pease setup a recurring transfer of $______________ from my Bellwether Account/Suffix #______________

Thank you for your payment!
id #: 0435672
date: 4/24/13
teller #: 08

Member Signature ____________________________ Date ____________
(Signature required for transfer)

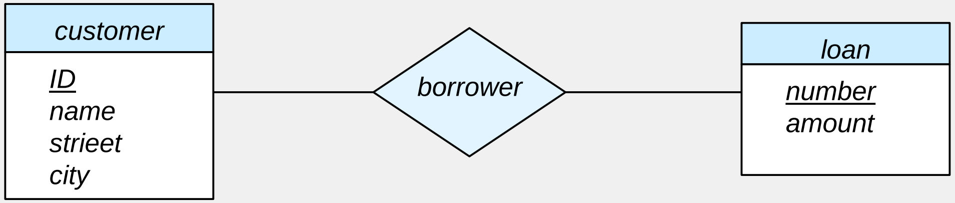
11111 26

Please return this form with your check or authorization to transfer funds.

- ID# is frequently referred to ask a *pseudo-key*.

Keys in an ER Diagram

- A primary key of an entity set is specified in an ER diagram by underlining the key attributes.



Keys for Relationship Sets

- Much like an entity set, a relationship set can also have a super key.
- 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*
- The mapping cardinality of a relationship set will determine, in part, what the candidate keys of the relationship set are.
- If the relationship is one-to-one, then using just one of the primary keys of the participating entity sets is, in fact, minimal, and hence forms a candidate key for the relationship set.

Design Options

- Frequently there are many ways to model a given situation.
- Use of an entity vs. an attribute:
  - Is *Telephone-Number* an attribute or an entity?
- Use of an entity vs. a relationship:
  - Is *Loan* an entity or a relationship?
- Placement of relationship attributes

Lets Try an Example!

Construct an ER diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents.

Lets Try an Example!

From the 6th edition…

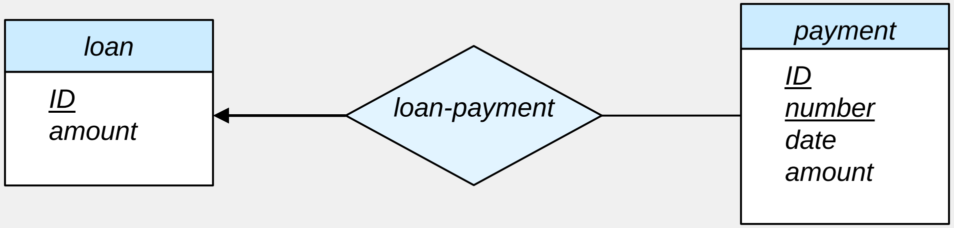
Construct an ER diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents. Each insurance policy covers one or more cars, and has one or more premium payments associated with it. Each payment is for a particular period of time, and has an associated due date, and the date when the payment was received.

Examples and Exercises

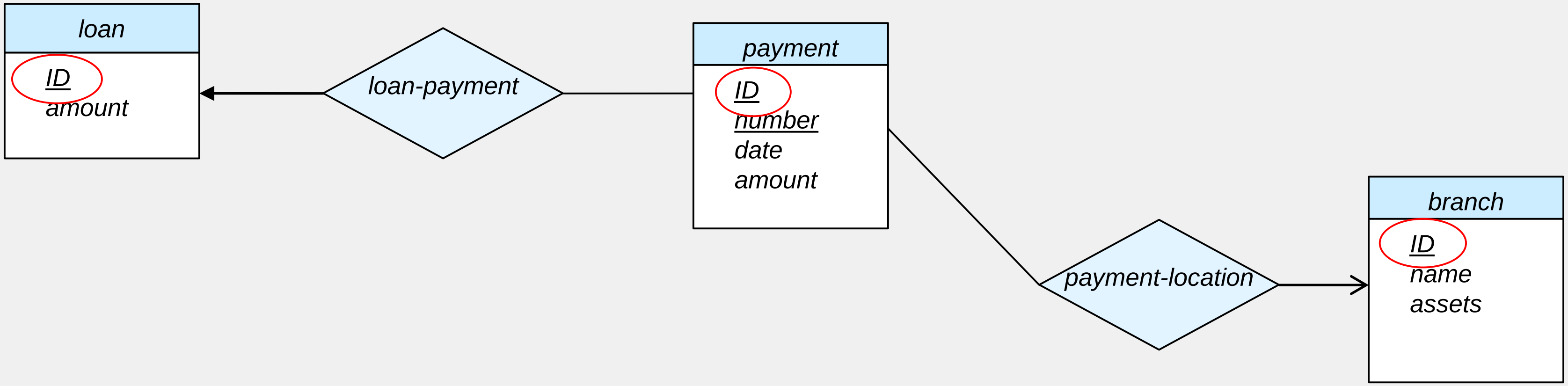
- Look for other ER diagramming exercises in the book, at the end of the chapter, or online.
- Google image search “ER diagram example”
- One example:
  - http://commons.wikimedia.org/wiki/File:ER_Diagram_MMORPG.png

Strong vs. Weak Entity Sets

- Recall the loan and payment entity sets:



- Now consider the following: (notice the ambiguity)

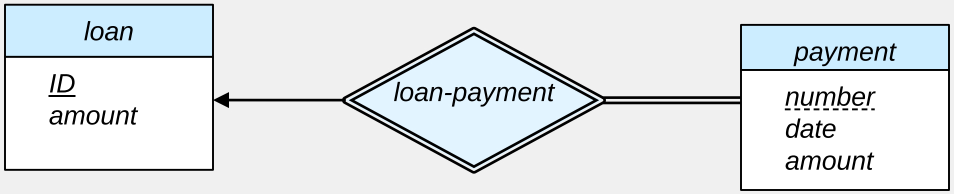


Strong vs. Weak Entity Sets

- For most entity sets, a primary key can specified in terms of its immediate attributes.
- Such an entity set is referred to as a *strong entity set*.
- For some entities, however, it is helpful to specify its' primary key, at least in part, in terms of some other entities' attributes.
- Such an entity set is referred to as a *weak entity set*.
- A weak entity set is typically associated with an *identifying entity set* (which is usually strong) via a total, one-to-many relationship.

Strong vs. Weak Entity Sets

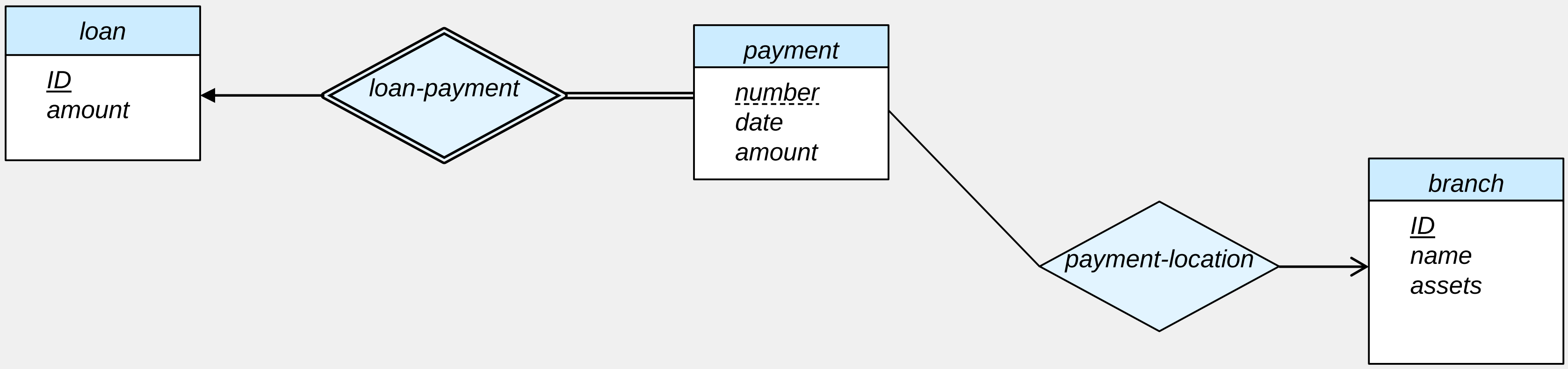
- Expressed as strong & weak entity sets:



- The relationship between the strong and weak entity sets is specified with a double-diamond.

Strong vs. Weak Entity Sets

- Note how this resolves the ambiguity:

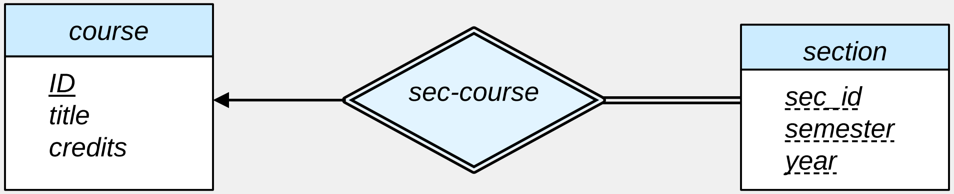


Strong vs. Weak Entity Sets

- In such a case, the (weak) entity typically has a subset of attributes, called a *discriminator (or partial key)*, that distinguishes among all entities of the weak entity set associated with one identifying entity.
- The discriminator is underlined with a dashed line.
- A primary key for the weak entity set consists of two parts:
  - The primary key of the associated identifying entity set
  - The weak entity set's discriminator
- Primary key for *payment* is (*loan-number, payment-number*)

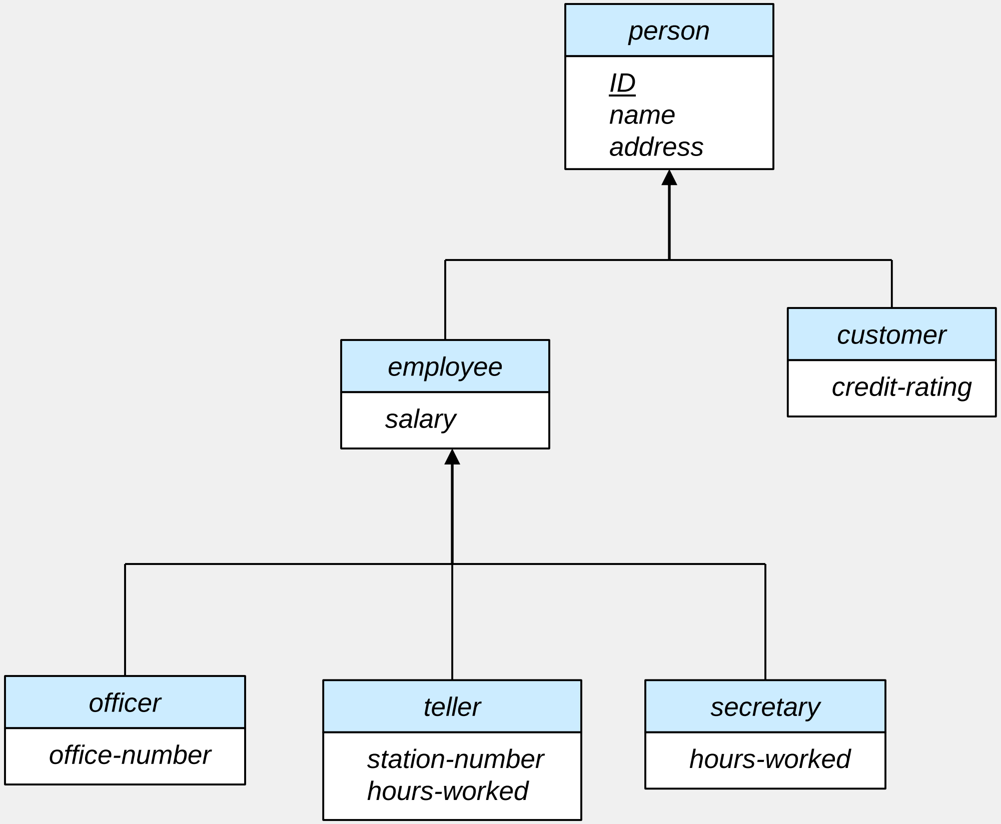
Another Example

- In the university enterprise, a *course* is a strong entity and a *section* can be modeled as a weak entity.



- The discriminator of *section* would be *sec-id, semester* and *year.*

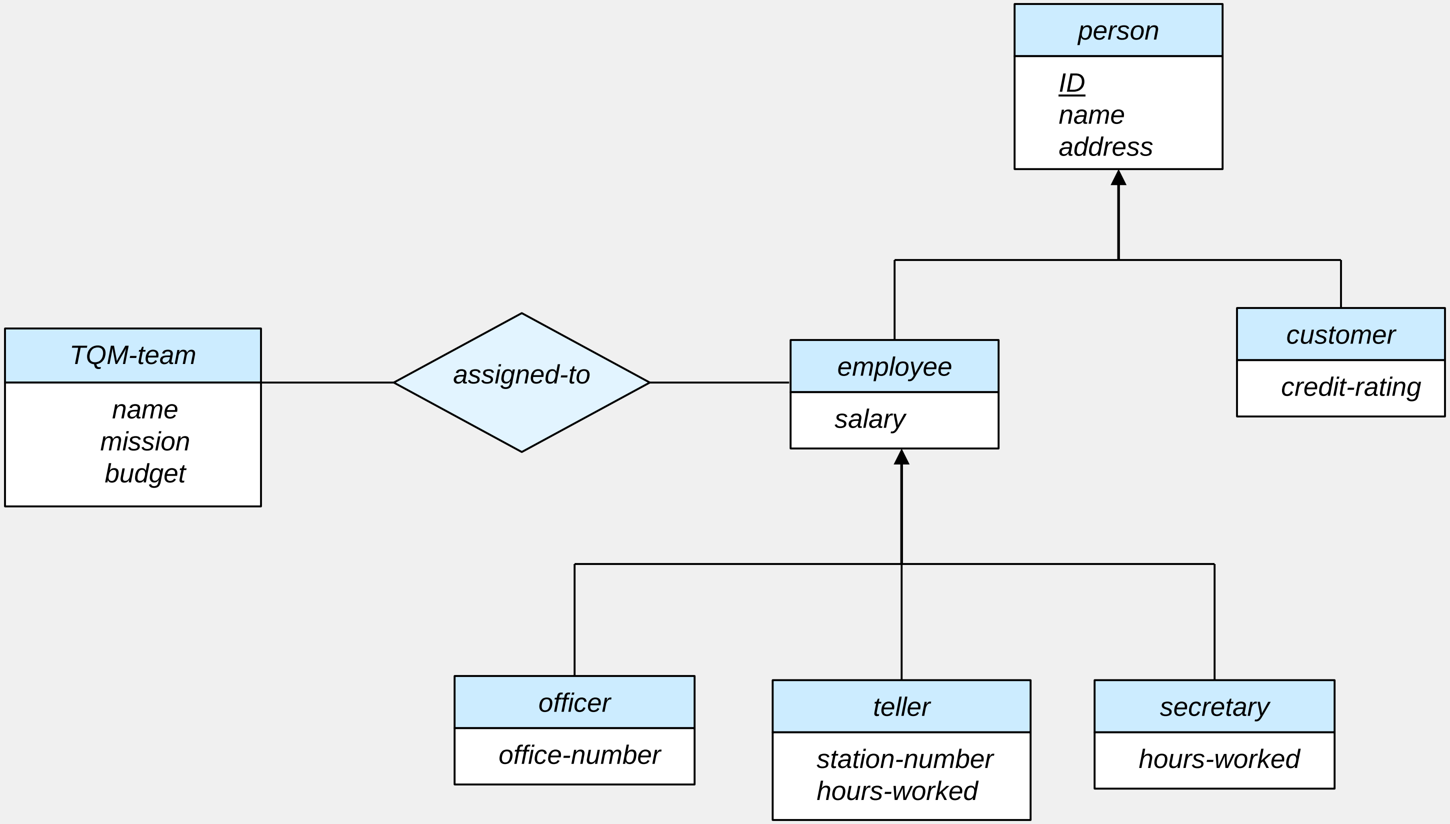
Specialization and Generalization - Inheritance



Specialization

- Inheritance relationships also referred to as a *superclass-subclass* relationships.
- Lower-level entity sets:
  - Have attributes that do not apply to the higher-level entity set.
  - Participate in relationships that do not apply to the higher-level entity set, e.g., airline employees, pilots, crew, agents, etc., but only pilots are certified certified to fly certain aircraft types.
- Lower-level entity sets are said to *inherit* all the attributes and relationships from the higher-level entity sets to which they are linked.

Specialization and Generalization - Inheritance



Specialization vs. Generalization

- Top-down design process; we designate sub-groupings within an entity set that are distinctive from other entities in the set.
- Bottom-up design process: combine a number of entity sets that share the same features into a higher-level entity set.
- The terms specialization and generalization are used interchangeably, for the obvious reasons.

Specialization and Generalization, Cont.

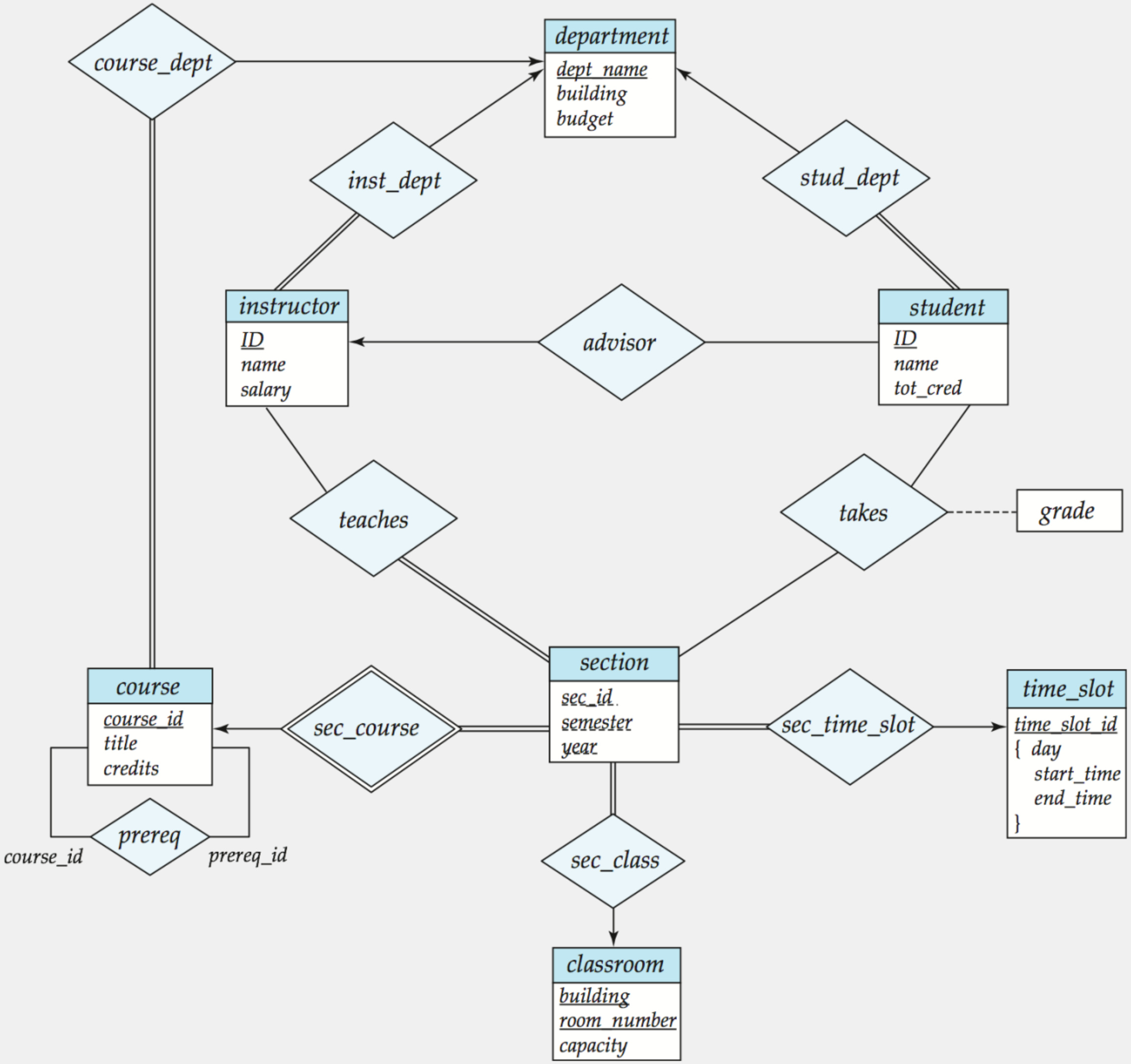
- A specialization/generalization relationship can be:
  - disjoint vs. overlapping – notated in any number of ways.
  - total vs. partial – notated as before.

- Multiple specializations of an entity set are possible:
  - *permanent-employee* vs. *temporary-employee*
  - In addition to *officer* vs. *secretary* vs. *teller*

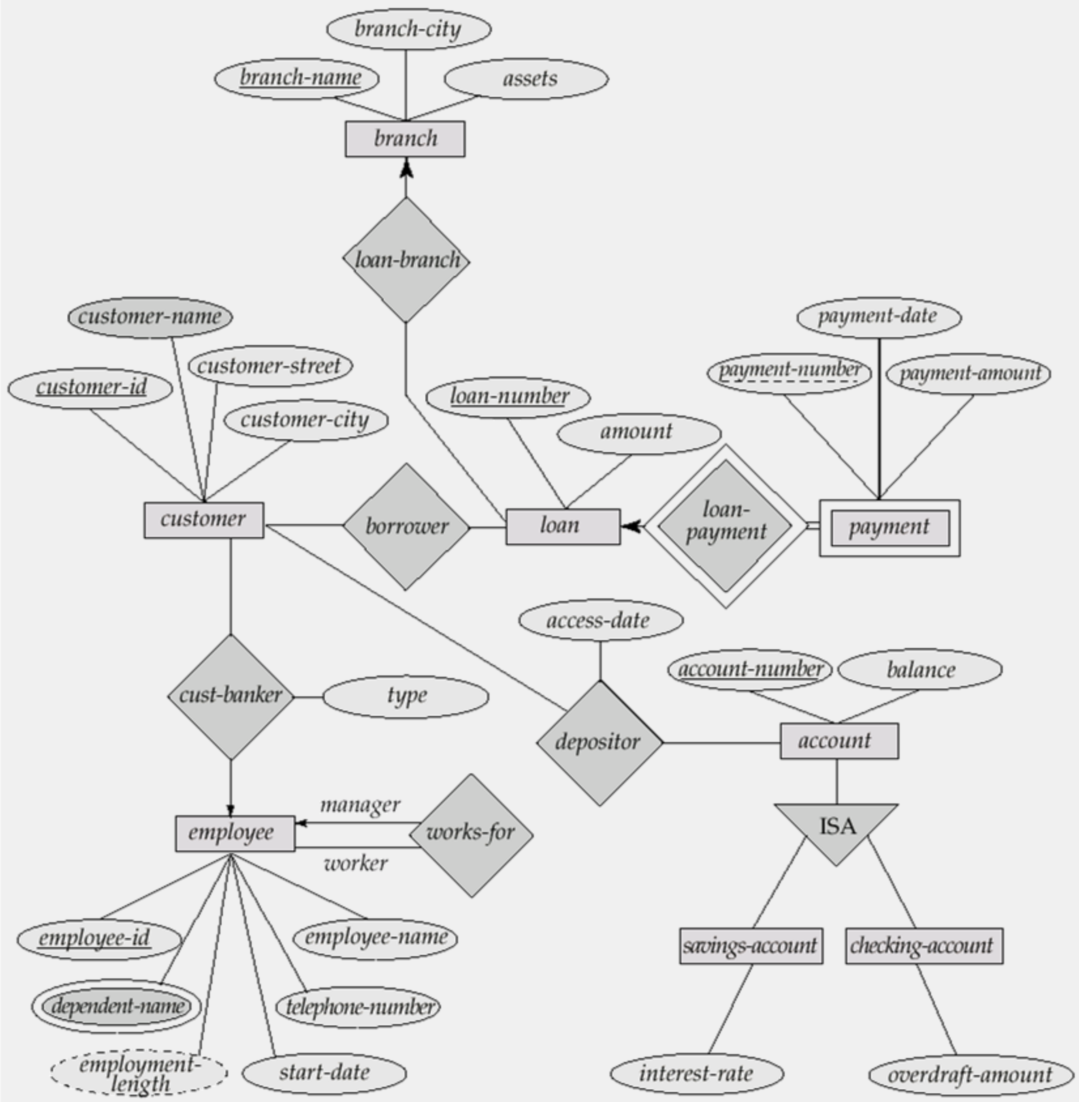
- Each particular employee would be a member of:
  - one of *permanent-employee* or *temporary-employee*, and
  - one of *officer*, *secretary*, or *teller*

- Multiple inheritance

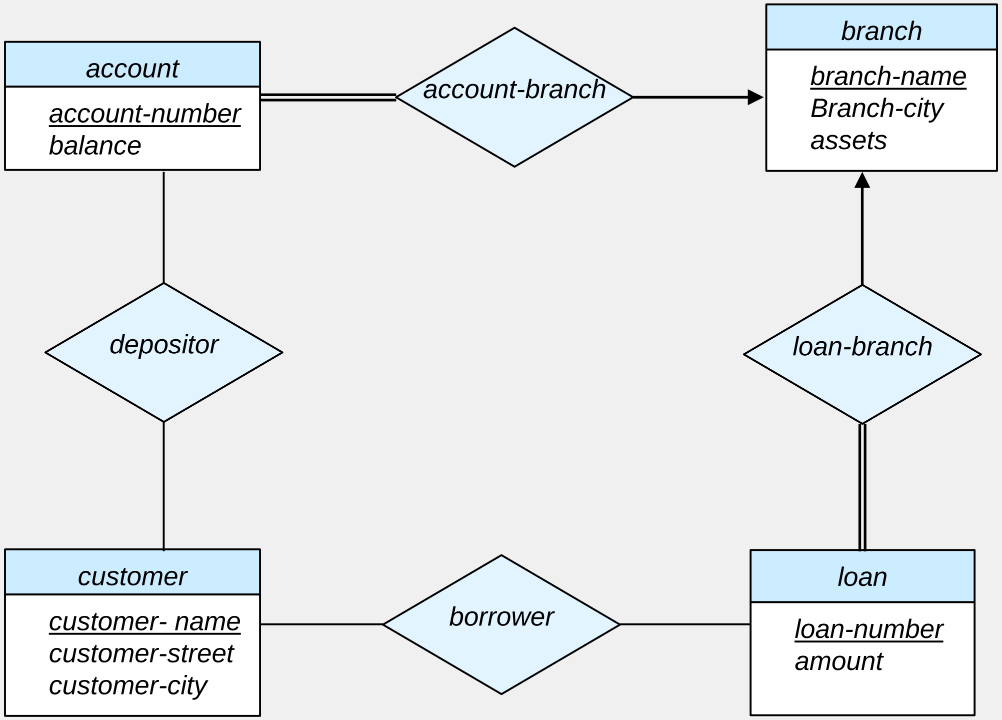
E-R Diagram for a University



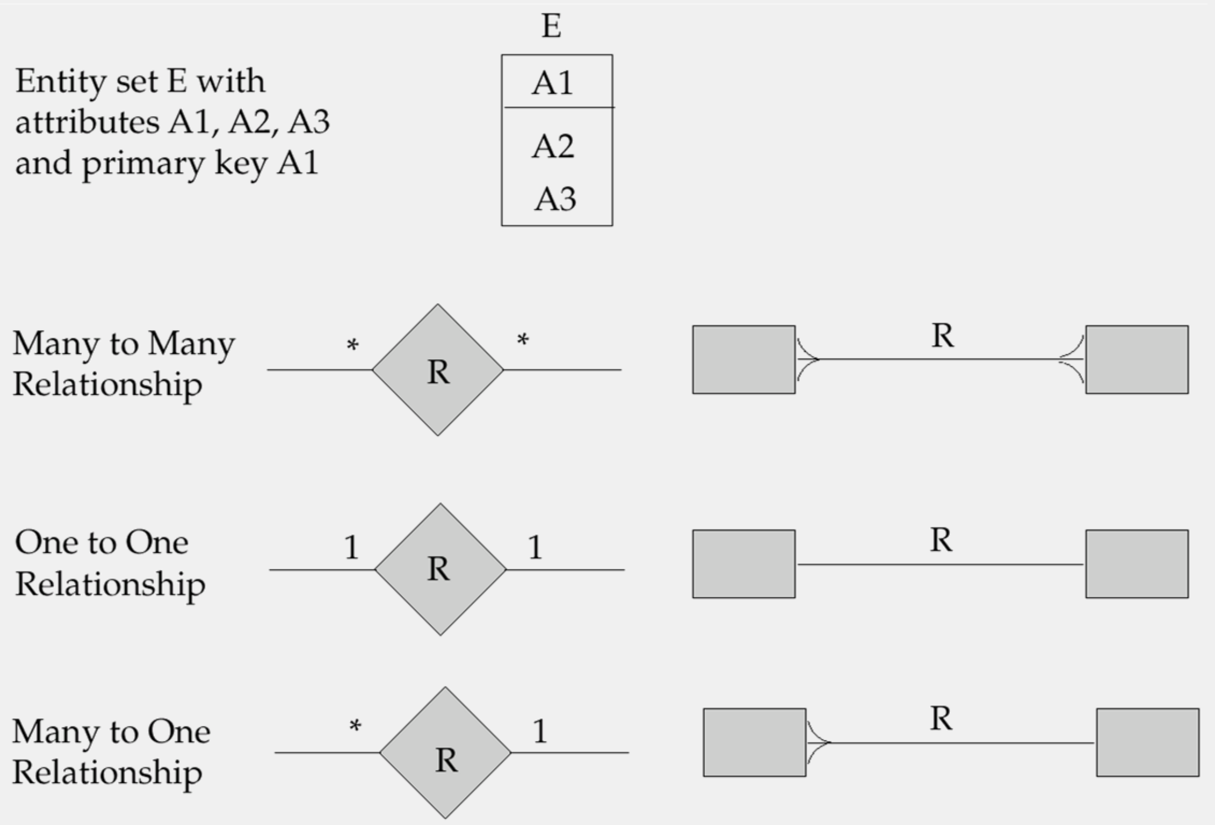
E-R Diagram for a Banking Enterprise (Chen's Notation)



E-R Diagram for the Banking Enterprise



Alternative E-R Notations



Entity set E with attributes A1, A2, A3 and primary key A1

Many to Many Relationship

One to One Relationship

Many to One Relationship

ER Diagramming Practice

- Practice developing ER diagrams:
  - see exercises at the end of the chapter on ER diagrams
  - use your imagination!

- Possible enterprises to model:
  - airline or airport
  - hospital
  - Insurance company
  - library
  - retailor – clothing, food, equipment
  - your favorite government agency

Reduction of an E-R Schema to Tables

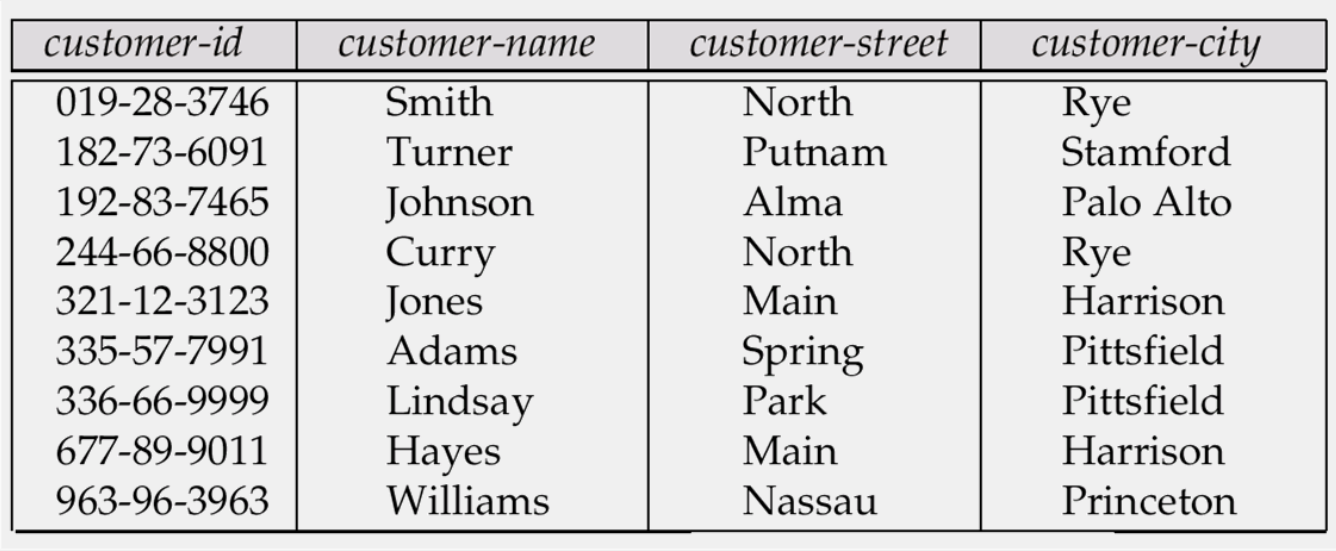
- Converting an E-R diagram to a relational database:
  - Each entity set is converted to its' own table.
  - Each relationship *can be* (but may not be) converted to its' own table.

- Each table has a number of columns, which generally corresponding to the attributes in the corresponding entity or relationship set.

- The resulting tables can be modified in a variety of ways to support performance, space, or other requirements.

Representing Entity Sets as Tables

- A strong entity set reduces to a table with the same attributes.



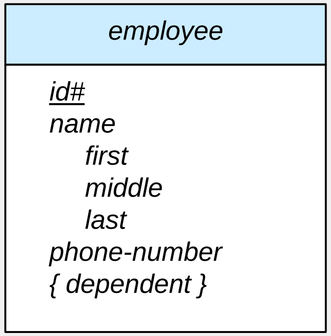
customer-id	*customer-name*	*customer-street*	*customer-city*
019-28-3746	Smith	North	Rye
182-73-6091	Turner	Putnam	Stamford
192-83-7465	Johnson	Alma	Palo Alto
244-66-8800	Curry	North	Rye
321-12-3123	Jones	Main	Harrison
335-57-7991	Adams	Spring	Pittsfield
336-66-9999	Lindsay	Park	Pittsfield
677-89-9011	Hayes	Main	Harrison
963-96-3963	Williams	Nassau	Princeton

Composite and Multi-valued Attributes

- Composite attributes are broken up.
- A multi-valued attribute *M* of entity *E* is represented by a new table with the following attributes:
  - The primary key of *E*
  - An attribute corresponding to multi-valued attribute *M*

Composite and Multi-valued Attributes

- Example:



employee

id#
name
first
middle
last
phone-number
{ dependent }

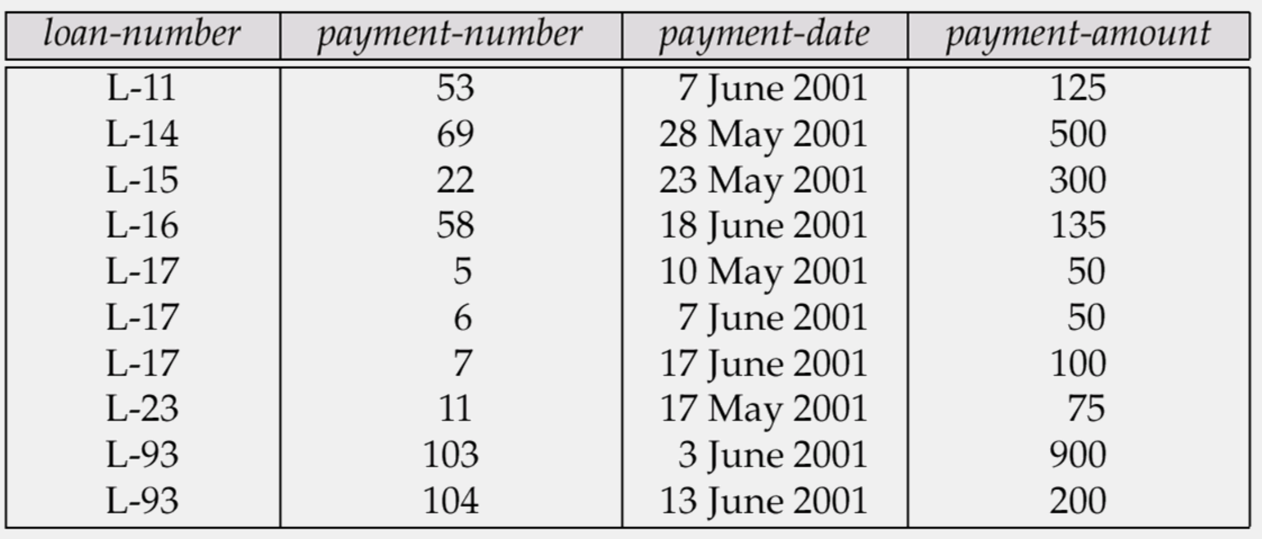
- Tables:

employee (id#, first-name, middle-name, last-name, phone-number)

dependent (id#, dname)

Representing Weak Entity Sets

- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set.

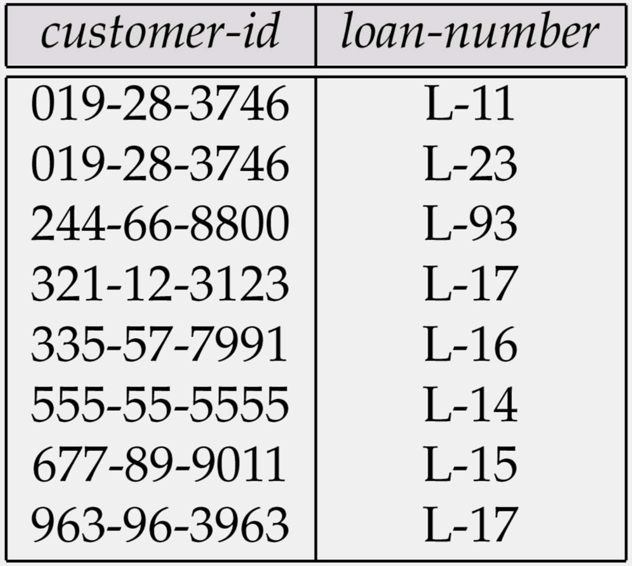


loan-number	*payment-number*	*payment-date*	*payment-amount*
L-11	53	7 June 2001	125
L-14	69	28 May 2001	500
L-15	22	23 May 2001	300
L-16	58	18 June 2001	135
L-17	5	10 May 2001	50
L-17	6	7 June 2001	50
L-17	7	17 June 2001	100
L-23	11	17 May 2001	75
L-93	103	3 June 2001	900
L-93	104	13 June 2001	200

Representing Relationship Sets as Tables

- A many-to-many relationship set is represented as a table with columns for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.

- Example: the *borrower* relationship set:



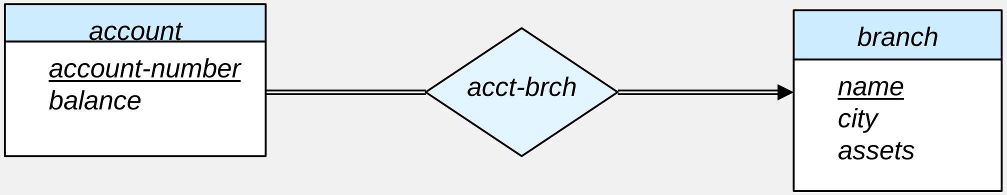
customer-id	*loan-number*
019-28-3746	L-11
019-28-3746	L-23
244-66-8800	L-93
321-12-3123	L-17
335-57-7991	L-16
555-55-5555	L-14
677-89-9011	L-15
963-96-3963	L-17

Representing Relationship Sets as Tables, cont.

- A many-to-one relationship set can be represented just like a many-to-many relationship.

- Technically this is not necessary, and in some cases it does not result in a good design.

- Example:



Relationship Set (total, many-to-one from account to branch)

Representing Relationship Sets as Tables, cont.

- The preceding could be converted to 3 tables directly, or as follows:

  account (*account-number*, *balance*, *branch-name*)

  branch (*branch-name*, *branch-city*, *assets*)

- Since the above relationship is total, this makes sense.
  - by the way, eliminating an unnecessary table is frequently considered…cool…

- On the other hand, suppose:
  - the relationship is partial
  - lots of accounts
  - most accounts don't have branches

- Consider a query that looks up account #'s for a given branch-name.
  - In this case, 3 tables are potentially better (why?).

Representing Relationship Sets as Tables, cont.

- For one-to-one relationship sets, the extra attribute can be added to either of the tables corresponding to the two entity sets.
- Other relationship attributes would be treated similarly.
- Note that either of the above could introduce null values if the relationship is not total.

Representing Specialization as Tables

- Note: This discussion assumes a 2-level inheritance hierarchy.
  - Exercise: Generalize it to an arbitrarily deep hierarchy.
- Method 1:
  - Form a table for the higher level entity set.
  - Form a table for each lower level entity set, including the primary key of the higher level entity set and local attributes.

table	attributes
person	*name, street, city*
customer	*name, credit-rating*
employee	*name, salary*

  - One Drawback: getting information about specific entities requires accessing two tables

Representing Specialization as Tables cont.

- Method 2:
  - Form a table for each entity set with all local and inherited attributes

table	table attributes
person	*name, street, city*
customer	*name, street, city, credit-rating*
employee	*name, street, city, salary*

- This method has obvious redundancies.
  - Particularly bad for persons who are both customers and employees.
- If specialization is total, the table for the generalized entity is redundant.
  - Temptation is to delete the *person* table; still might be needed for foreign key constraints.

Representing Specialization as Tables

- Method 3: (not presented in the book)
  - Form one table for the higher level entity set
  - This table has one column for every attribute in every subclass.

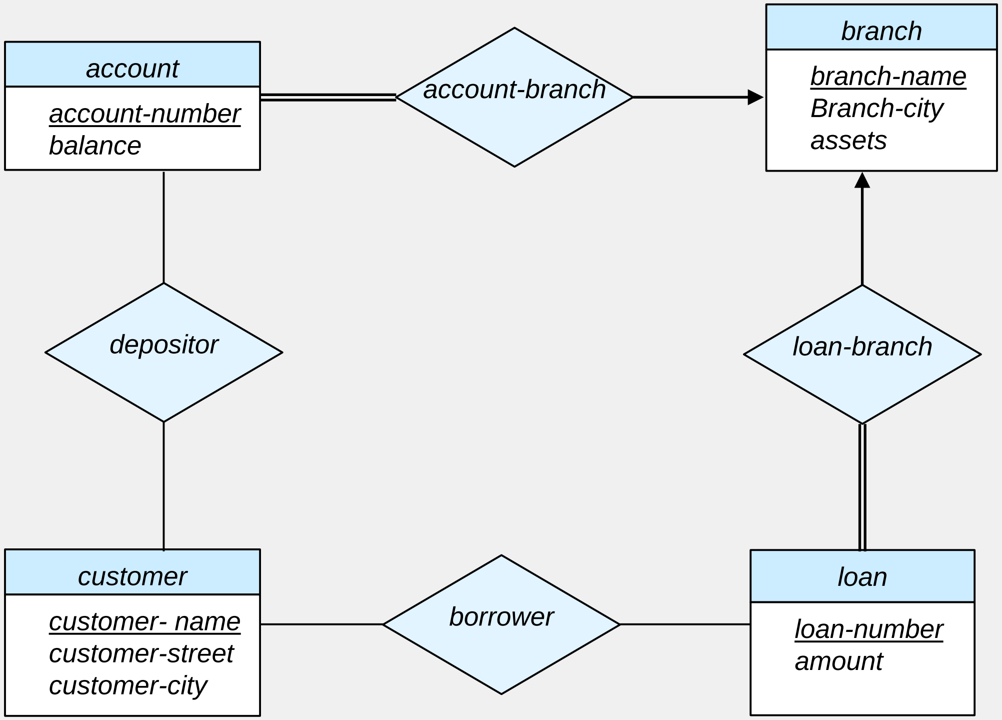
table	attributes
person	*name, street, city, credit-rating, salary*

  - Optionally, include a *type* attribute that indicates which subclass the stored entity belongs to.

- Obvious drawback - contains multiple nullable attributes.

- Sometimes referred to as a “junk drawer”

E-R Diagram for the Banking Enterprise



Relational Schemes for the Banking Enterprise

- The following relational schemes result:

branch (branch-name, branch-city, assets)

customer (customer-name, customer-street, customer-city)

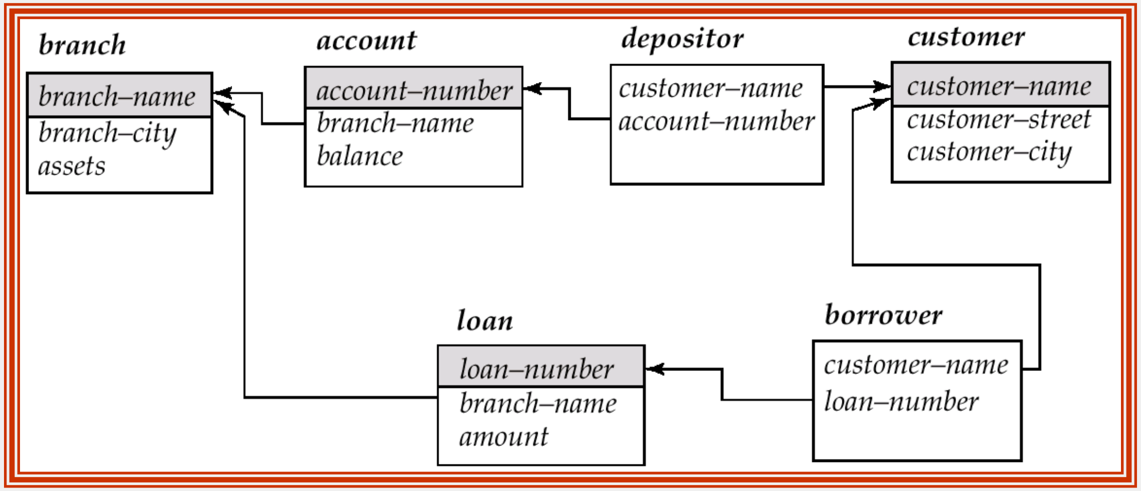
account (account-number, branch-name, balance)

loan (loan-number, branch-name, amount)

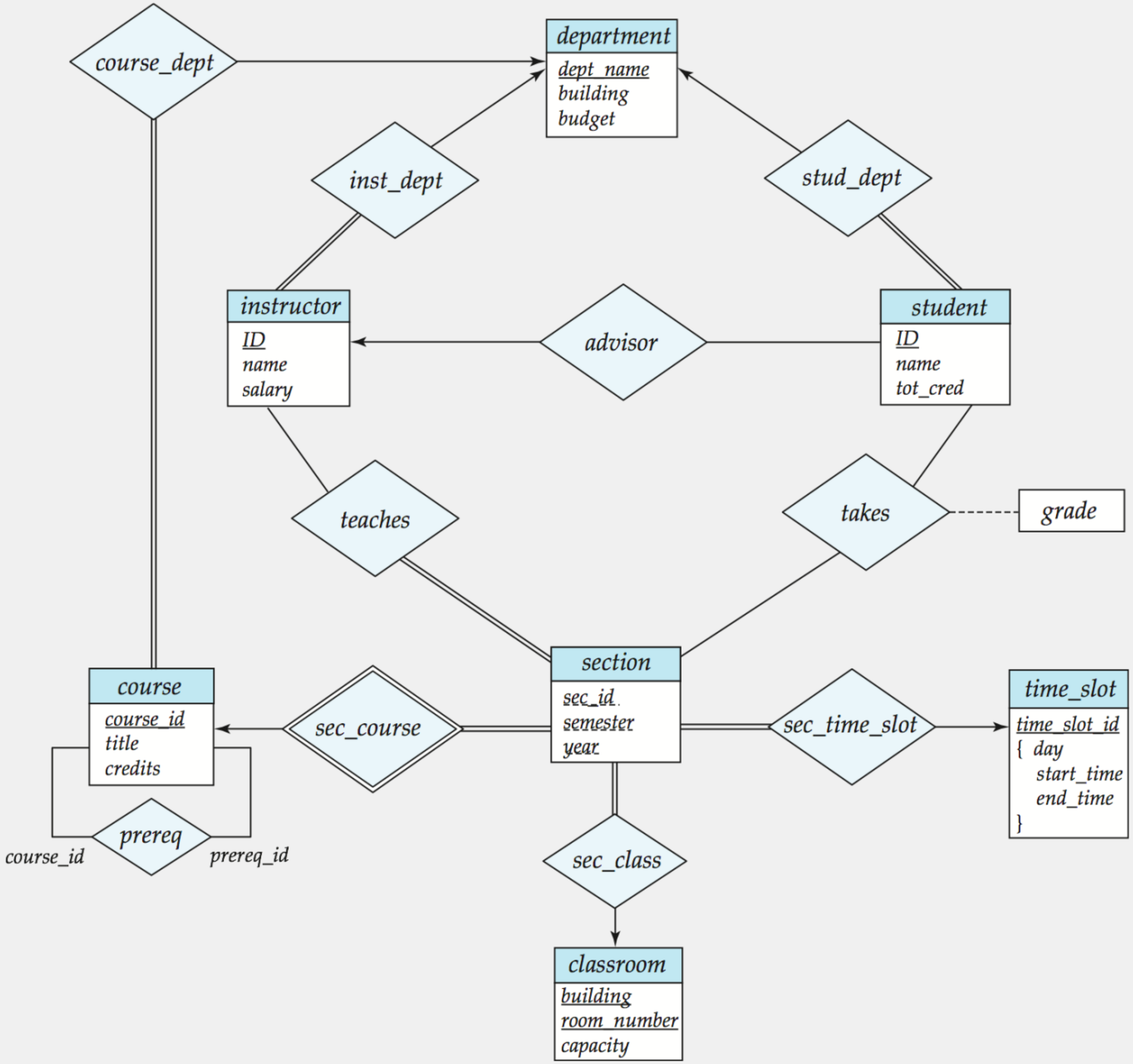
depositor (customer-name, account-number)

borrower (customer-name, loan-number)

Schema Diagram for the Banking Enterprise



E-R Diagram for a University



Relational Schemes for a University

Classroom (building, room-number, capacity)

Department (dept-name, building, budget)

Course (course-id, title, dept-name, credits)

Instructor (ID, name, depart-name, salary)

Section (course-id, sec-id, semester, year, building, room-number, time-slot-id)

Teaches (ID, course-id, sec-id, semester, year)

Student (ID, name, dept-name, tot-cred)

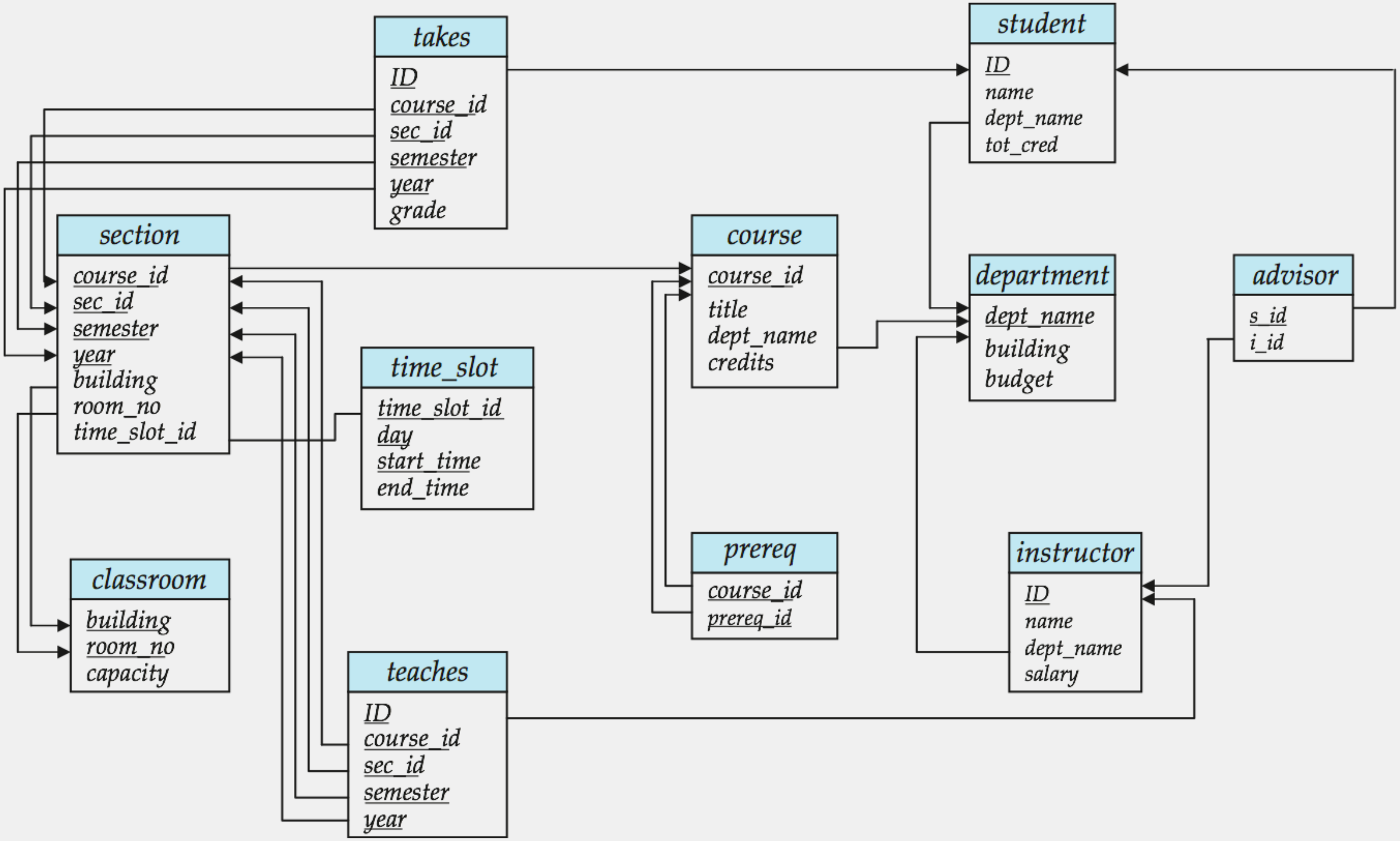
Takes (ID, course-id, sec-id, semester, year, grade)

Advisor (s-ID, i-ID)

Time-slot (time-slot-id, day, start-time, end-time)

Prereq (course-id, prereq-id)

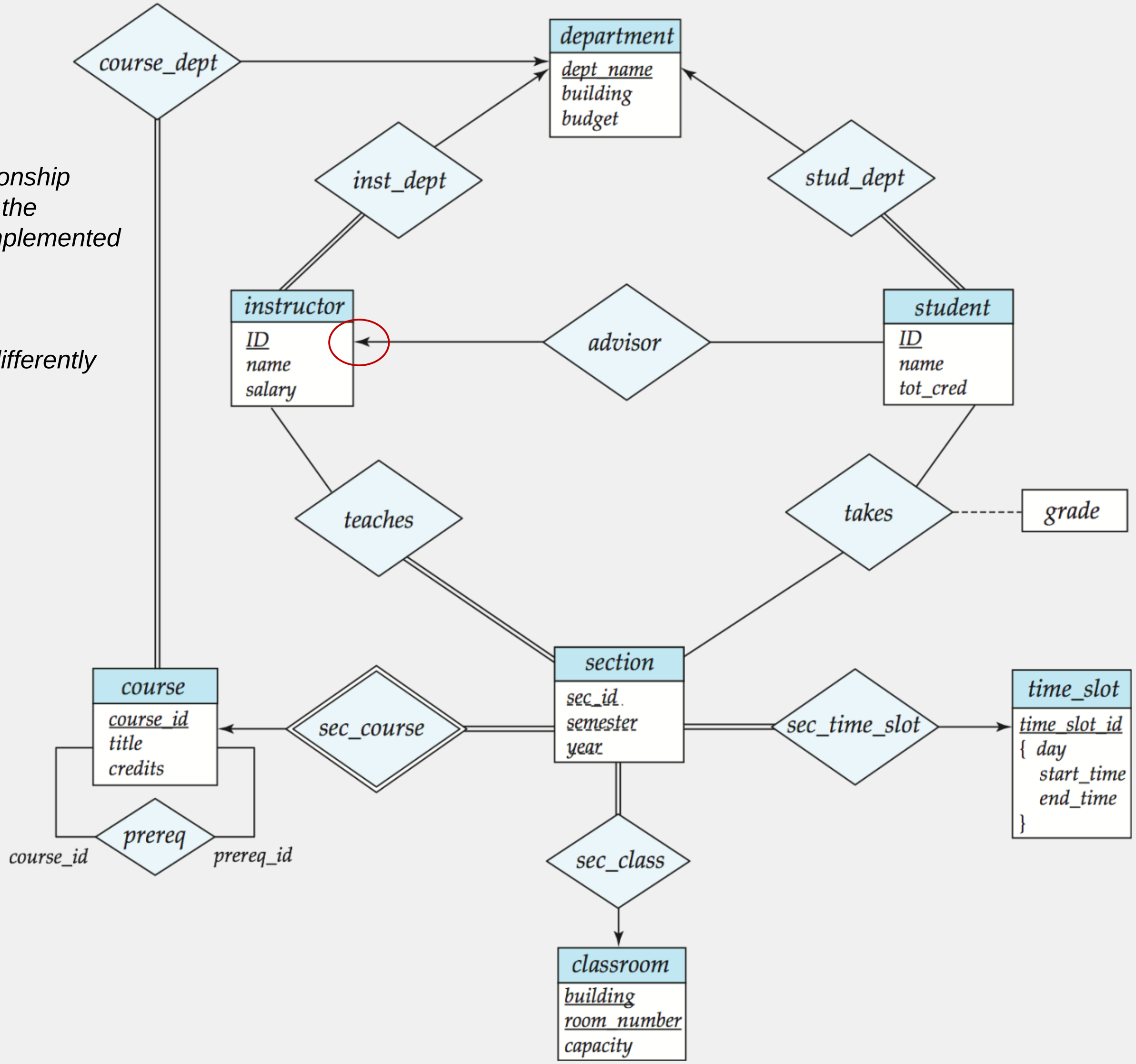
Schema Diagram for a University



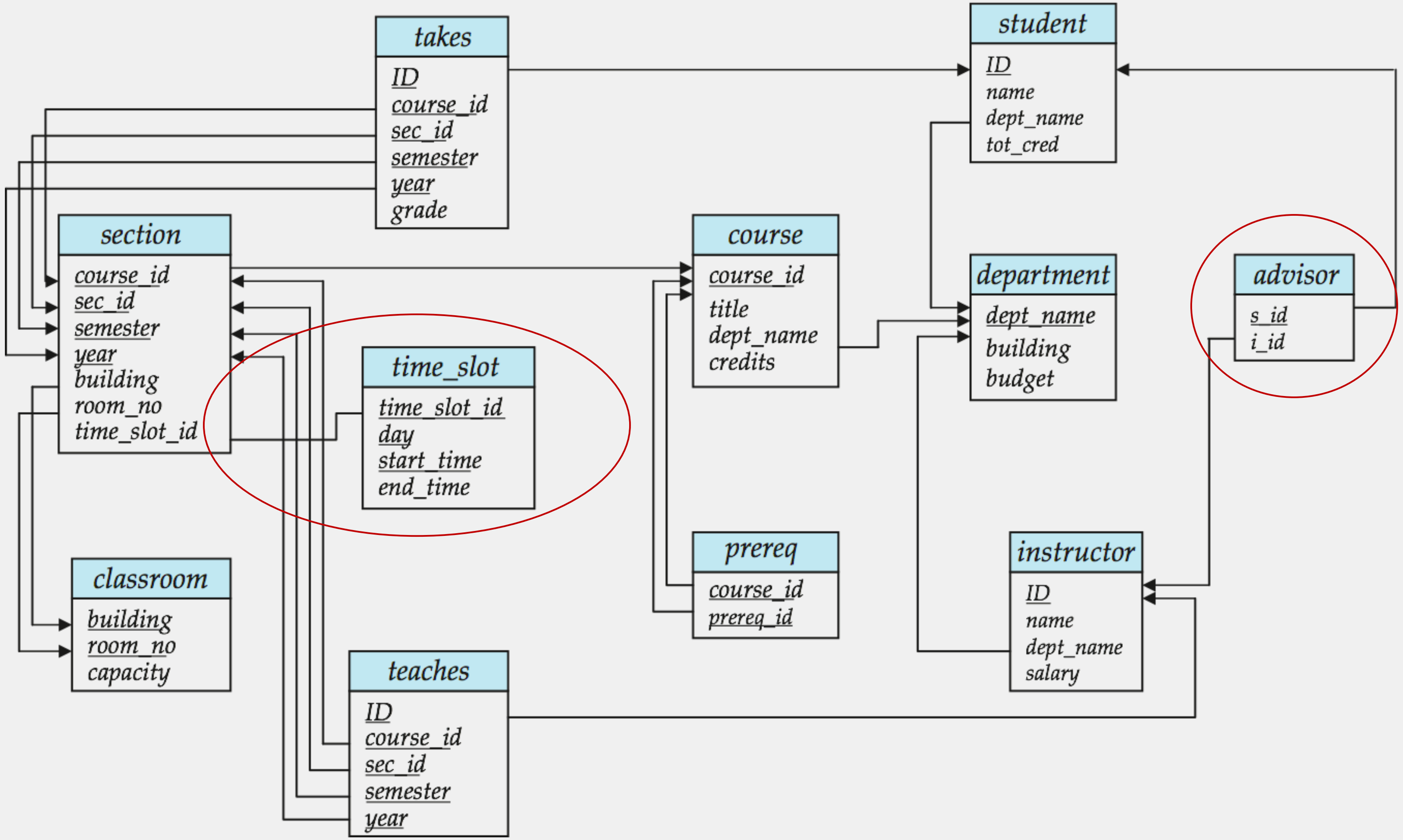
E-R Diagram for a University

Note the advisor relationship Is one-to-many, but in the schema it has been implemented as its' own table..

Similarly, time—slot is implemented slightly differently than described…



Schema Diagram for a University



The picture can't be displayed.

End of Chapter 6