Database Management System (DBMS)
B.Sc. CSIT
Fourth Semester

Unit 3.
Data Modeling using the Entity-Relationship Model
Session 1

Unit Outline

Unit 3. Data Modeling using the Entity-Relational Model

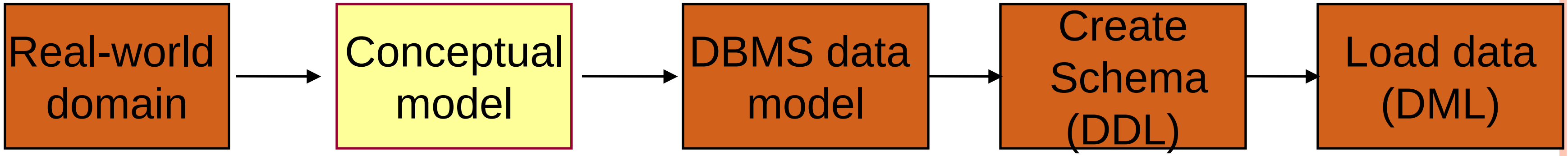
6 hours

Using High-Level Conceptual Data Models for Database Design; Entity Types, Entity Sets, Attributes, and Keys; Relationship Types, Relationship Sets, Roles, and Structural Constraints; Weak Entity Types; ER Diagrams, Naming Conventions, and Design Issues; Relationship Types of Degree Higher Than Two; Subclasses, Super Classes, and Inheritance; Specialization and Generalization; Constraints and Characteristics of Specialization and Generalization

Database Design

Steps in building a database for an application:

1. Understand real-world domain being captured
2. Specify it using a database conceptual model (E/R,OO)
3. Translate specification to model of DBMS (relational)
4. Create schema using DBMS commands (DDL)
5. Load data (DML)



Entity-Relationship Model (E/R)

Conceptual (high-level, semantic) data models:

Provide concepts that are close to the way many users perceive data. (Also called entity-based or object-based data models.)

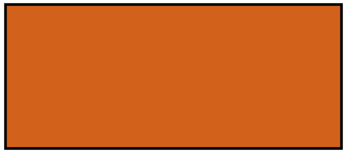
The **Entity-Relationship model** (ER) is a high-level description of the structure of the DB

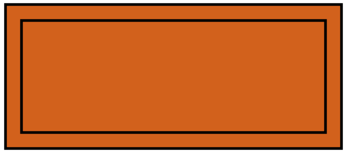
The **Entity-Relationship Diagram** (ERD) is a graphical model for representing the conceptual model for the data

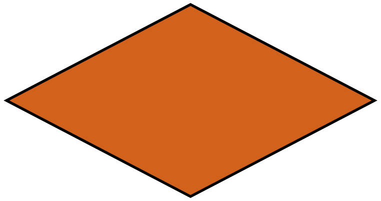
A E/R models the DB using three element types:

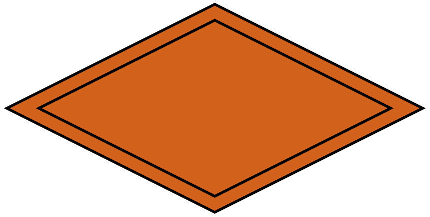
- Entities
- Attributes
- Relationships

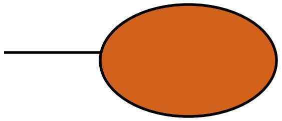
SYMBOLS USED IN ER DIAGRAMS(1)

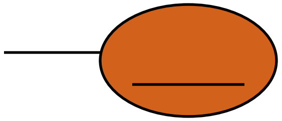

ENTITY

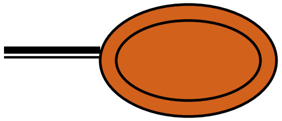

WEAK ENTITY

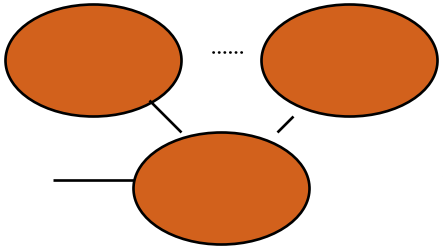

RELATIONSHIP

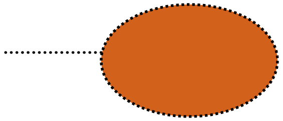

IDENTIFYING
RELATIONSHIP


ATTRIBUTE

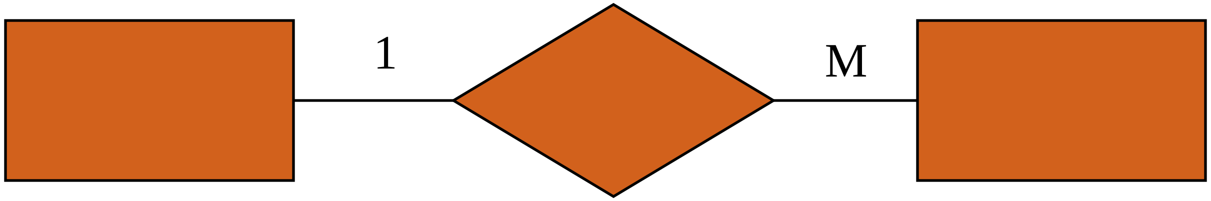

KEY ATTRIBUTE


MULTI-VALUED

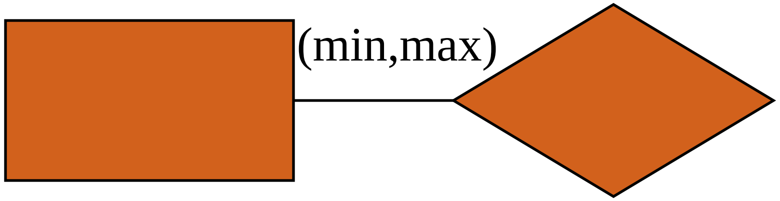

COMPOSITE


DERIVED

SYMBOLS USED IN ER DIAGRAMS(2)

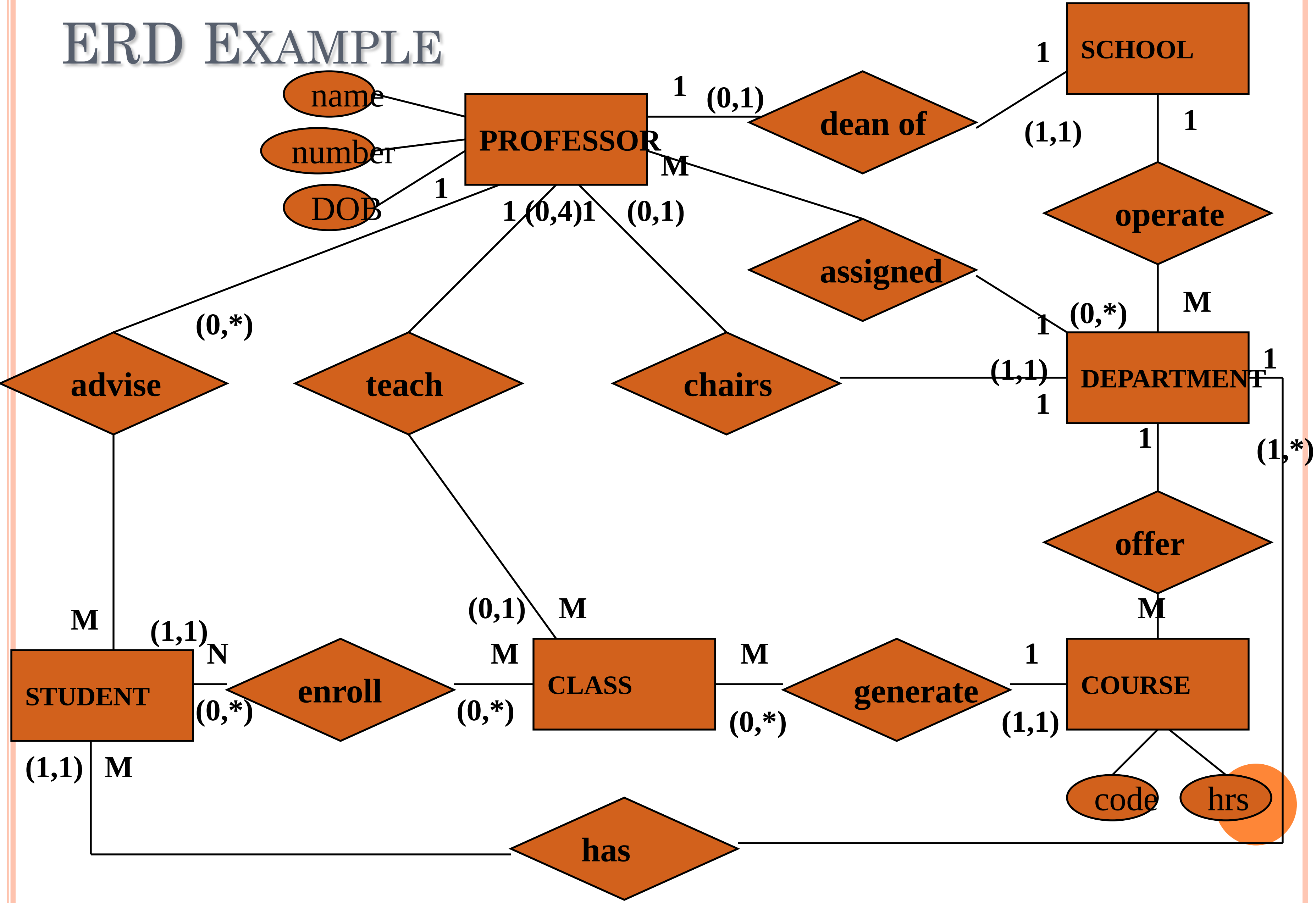


CARDINALITY RATIO



PARTICIPATION CONSTRAINTS

ERD Example


SCHOOL
1
PROFESSOR
name
number
DOB
1
(0,1)
dean of
(1,1)
1
operate
M
1
1
(0,4)
1
(0,1)
assigned
1
(0,*)
M
(0,*)
advise
teach
chairs
(1,1)
DEPARTMENT
1
1
1
(1,*)
offer
M
M
(1,1)
N
(0,1)
M
M
M
1
STUDENT
enroll
CLASS
generate
COURSE
(0,*)
(0,*)
(0,*)
(1,1)
(1,1)
M
code
hrs
has

WHAT IS A DATABASE APPLICATION?

- Refers to :

  A particular database **+** Associated programs to implement the quires and updates that has a user-friendly interfaces (GUIs).
- Example: Bank database + programs to keep track of the deposit & withdrawal.
- Conceptual modeling is important in designing a successful database application.

Using High-Level Conceptual Data Models of Database Design

- Requirements collection and analysis
- Conceptual schema
- Logical design
- Physical design

ENTITIES & ATTRIBUTES

- **Entity:**

The basic object that the ER model represent is an entity which is a thing in the real world.

  - Physical existence : like a person, employee
  - Conceptual existence : like job, course

- **Attribute:**

The particular properties that describes the entity.

e.g.: an employee is described by : name, age, address, salary , job.

* An entity will have a value for each of its attributes.

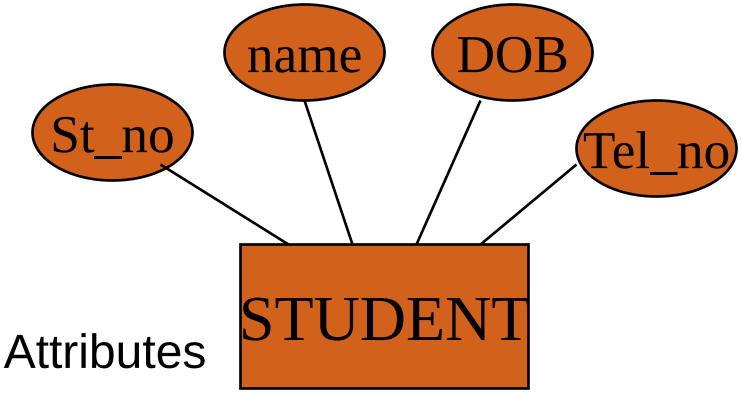
ENTITIES & ATTRIBUTES

Entities

STUDENT

COURSE

Attributes



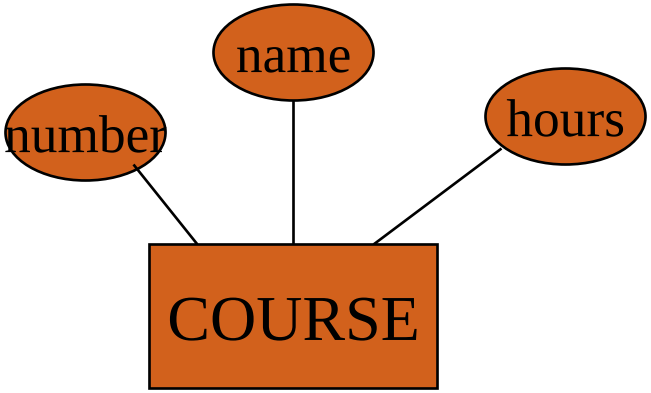
St_no

name

DOB

Tel_no

STUDENT



number

name

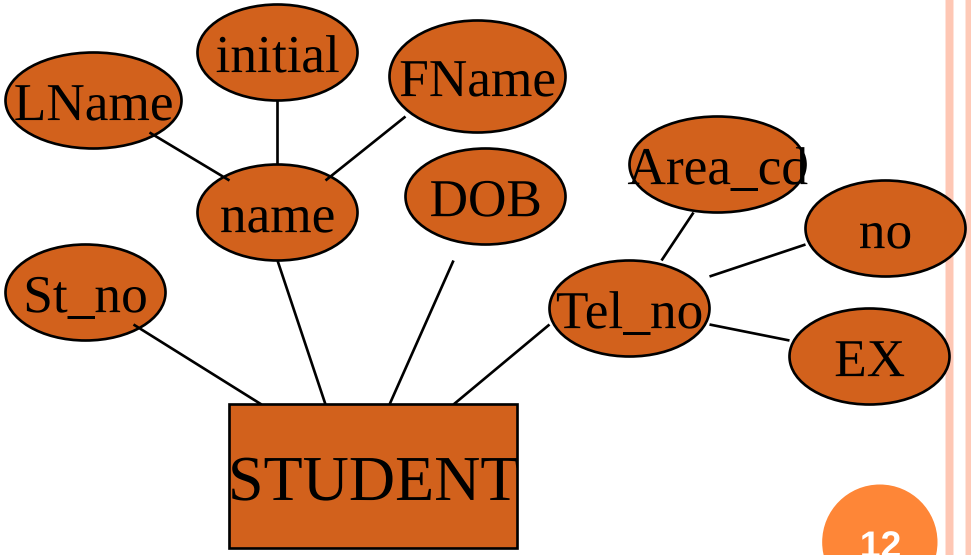
hours

COURSE

Attribute Types
Composite v Simple

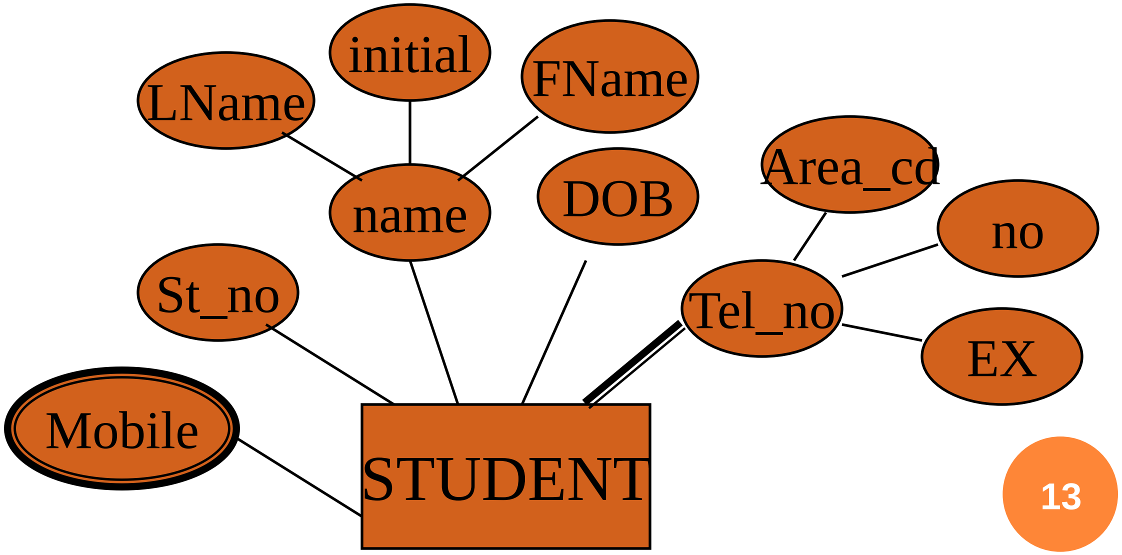
- Composite attributes: divided into smaller subparts e.g. name divided into first, initial ,last.
- Simple attributes: attributes that are not divisible e.g. salary.

In some cases composite values do not need to be divisible => use them as a single unit.

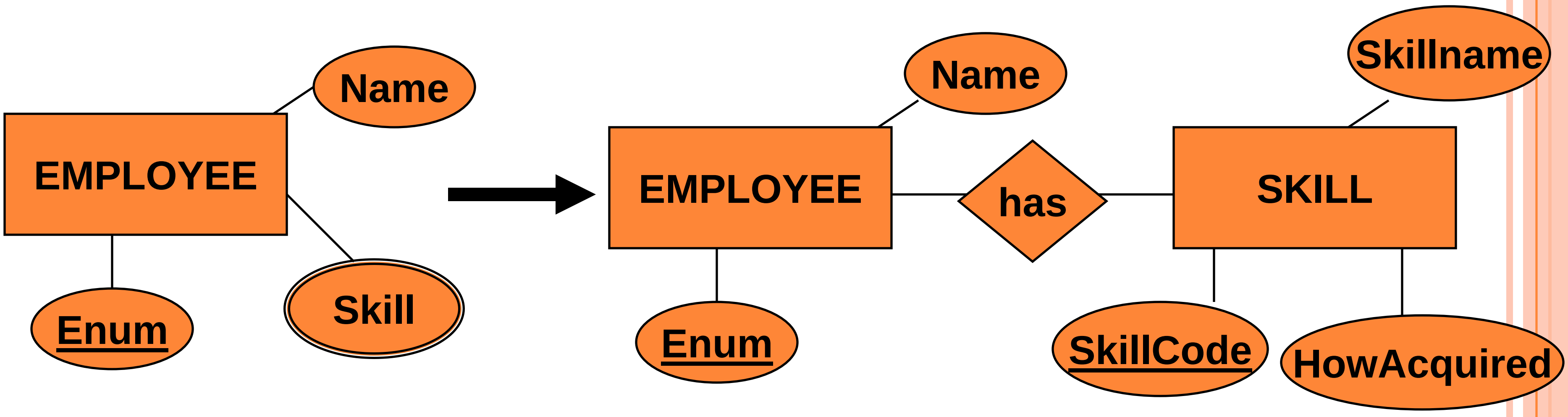


Single Valued v Multivalued

- Single value : most attributes are of this type. E.g. age
- Multivalued: set of values for the same entity. E.g. car colors, previous degrees.



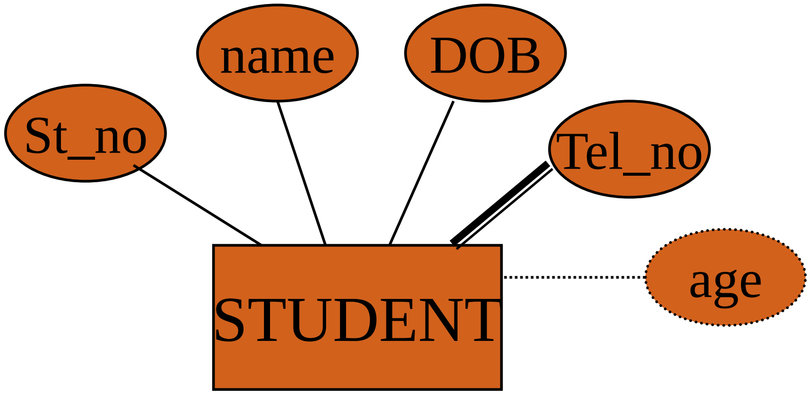
Multivalued Attributes



- It is desirable to decompose multivalued attribute to a separate entity. It might be a 1:N or M:N relationship.

Stored v Derived Attributes

- Stored attributes: regular attributes.
- Derived attributes: are calculated from attributes of one or several related entities.
- E.g. calculating the salary from the hours worked and number of hours.



MORE ATTRIBUTE TYPES

- Null values: some entities may not have an applicable values for an attribute => a special value is created called Null. E.g. in the employee information the field of mobile number can have a null value because some employees do not have mobile phones.
- Complex attributes: composite and multivalued attributes can be nested. E.g. a person may have many certificate, each certificate has the subfields name, year and level. Therefore it is represented this way : {certificate (name, year, level)}

  {} represents multivalued () represents composite attributes.

Entities

- Entity Type: is a collection of occurrences of entities that have common properties. E.g. employees.
- Entity Instance: is a single occurrence of an entity type. E.g. the student named Aseel Mohammed.
- Entity Set: a collection of all entities of a particular entity type in the database at any point in time. E.g. Employee refers to both the type of entity + the current set of all employees entity in the database.

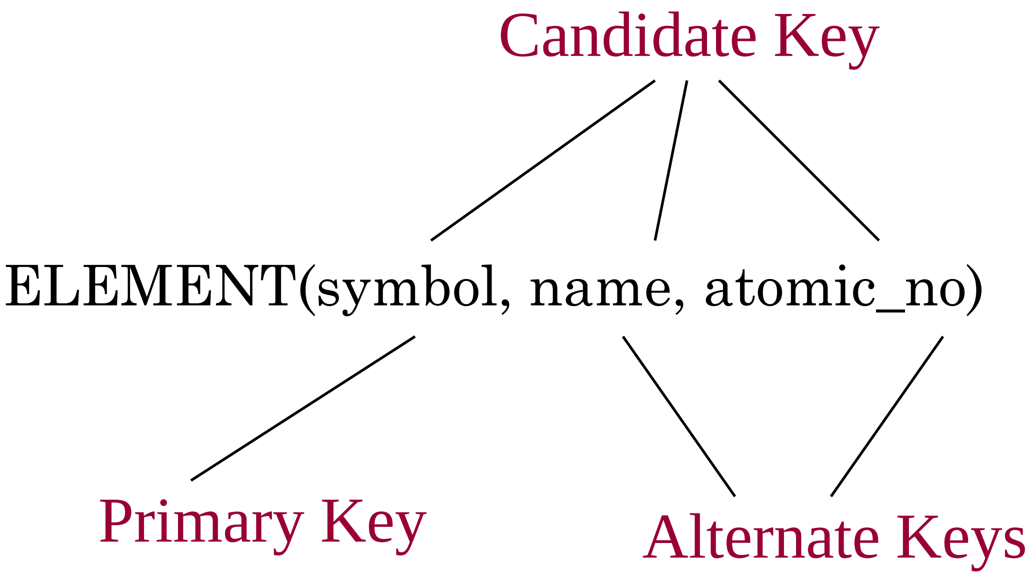
KEY ATTRIBUTES OF AN ENTITY TYPE

- Key attribute: is a key that identifies each entity uniquely, and has a distinct value . E.g. the name attribute in the company entity is unique.
- Candidate Key (CK): a set of attributes that uniquely identify an instance of an entity. E.g. Students can be identified by SSN or ID.
- Composite Key: is a key made from more than one attribute. E.g. the flight can be uniquely identified by flight number and flight date.
- Alternate Key (AK): is a candidate key that is NOT selected to be the primary key.

KEYS EXAMPLE

ELEMENT(symbol, name, atomic_no)

Keys Example



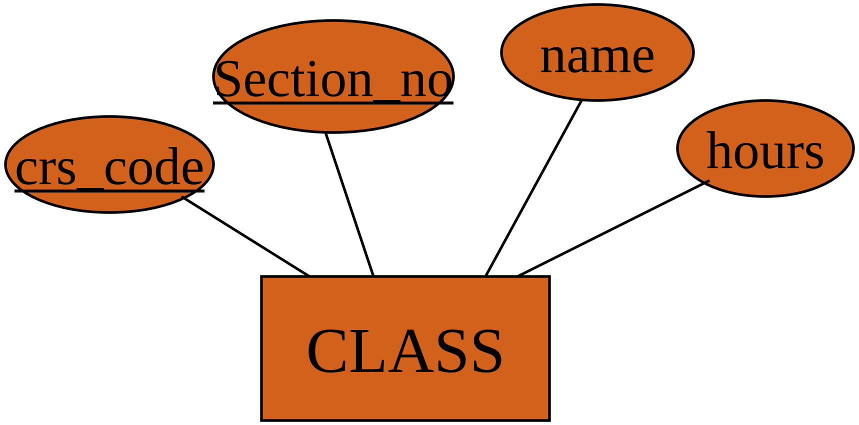
Candidate Key

ELEMENT(symbol, name, atomic_no)

Primary Key

Alternate Keys

Composite Key in ERD



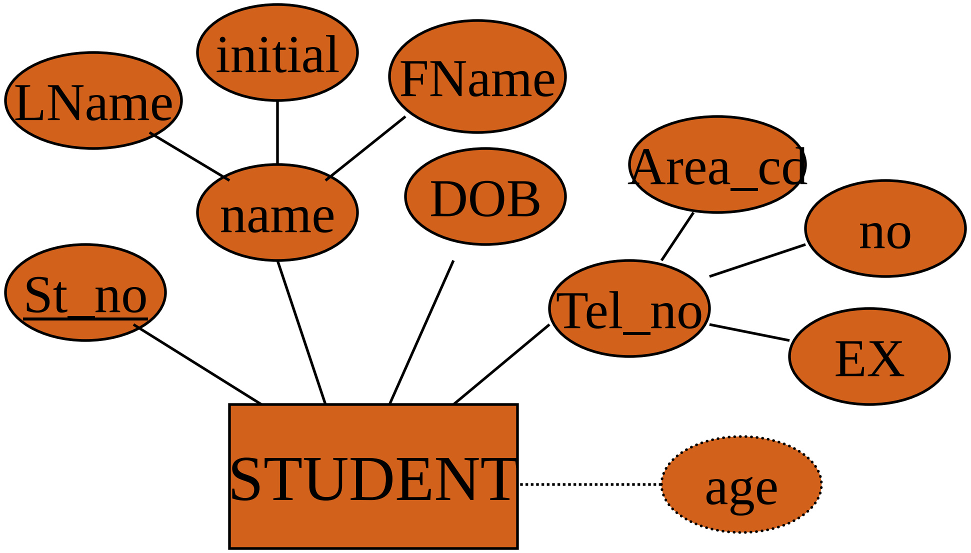
Primary Key Attribute (PK)

- Can be a single attribute or composite attribute.
- Can be called identifier.
- Weak entity may have no key.
- Has the following criteria:
    - Should not change its value
    - Not null
    - Avoid intelligent keys.
    - Substitute large composite keys with surrogate keys (system generated keys for unique numbers).

Choice of PK

Choice of Primary Key (PK) is based on:

- Attribute length
- Number of attributes required
- Certainty of uniqueness



Strong & Weak Entity Types

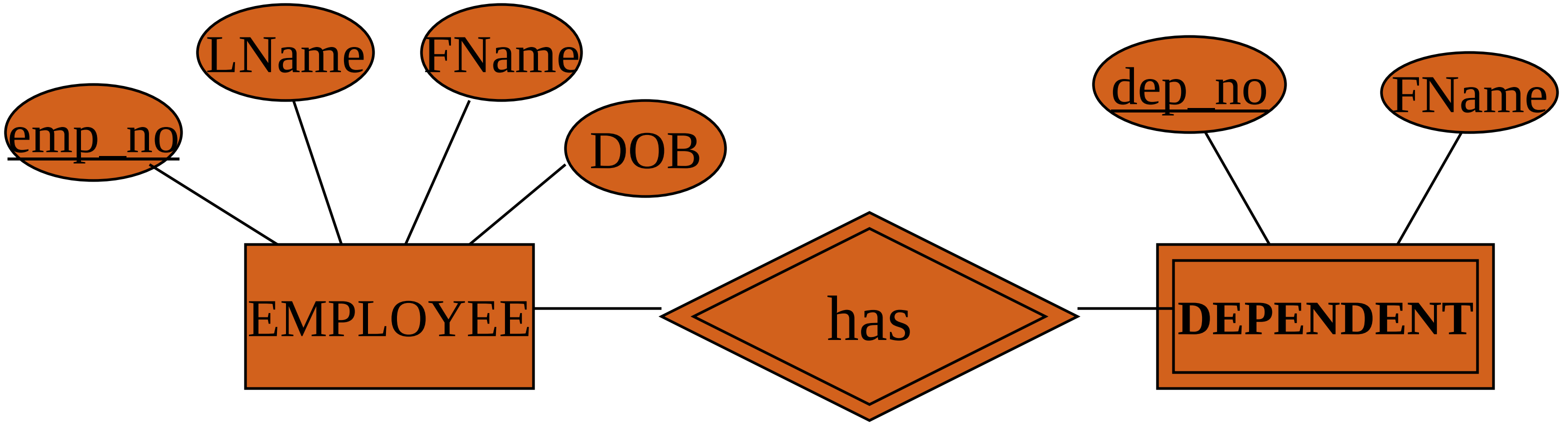
A **strong entity type** is NOT existence-dependent on some other entity type. It has a PK.

A **weak entity type** is an entity type that is existence-dependent on some other entity type. It does not have a PK.

WEAK ENTITY TYPE

- The existence of a weak entity type depends on the existence of a strong entity set; it must relate to the strong entity type via a relationship type called **identifying relationship**.

- The PK of a weak entity set is formed by the PK of its strong entity type, plus a weak entity type discriminator attribute (Partial key).



VALUES SETS (DOMAINS) OF ATTRIBUTES

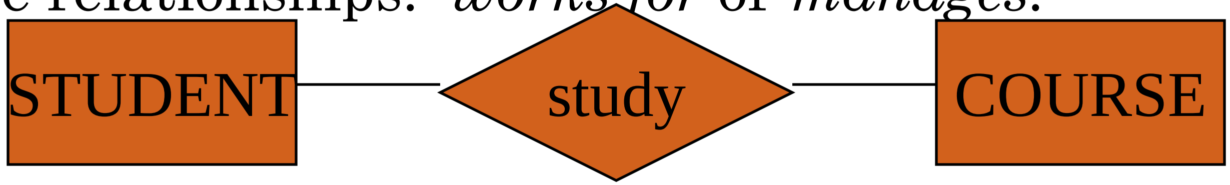
- Domain: the set of values that might be assigned to the attribute for each individual entity.

  e.g. ages of employees can be (16-70)

  e.g. names is a set of alphabetical characters only
- Not displayed in ER diagram.

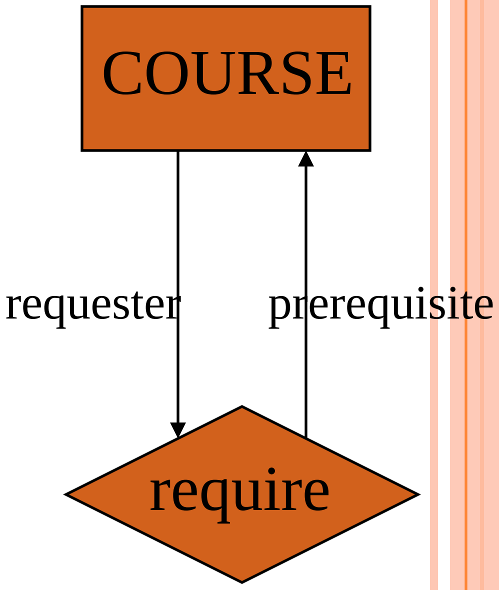
RELATIONSHIPS

- Relationship Type: R among n entity types E1,E2,…En defines a set of associations among entities from these types.
- Each entity type participates in the relationship type R.
- Relationships may have attributes.
- Degree of relationship: number of entities participating in the relation.
- More than one relationship can exist with the same entity types. E.g. an employee and department may have the relationships: *works for* or *manages*.



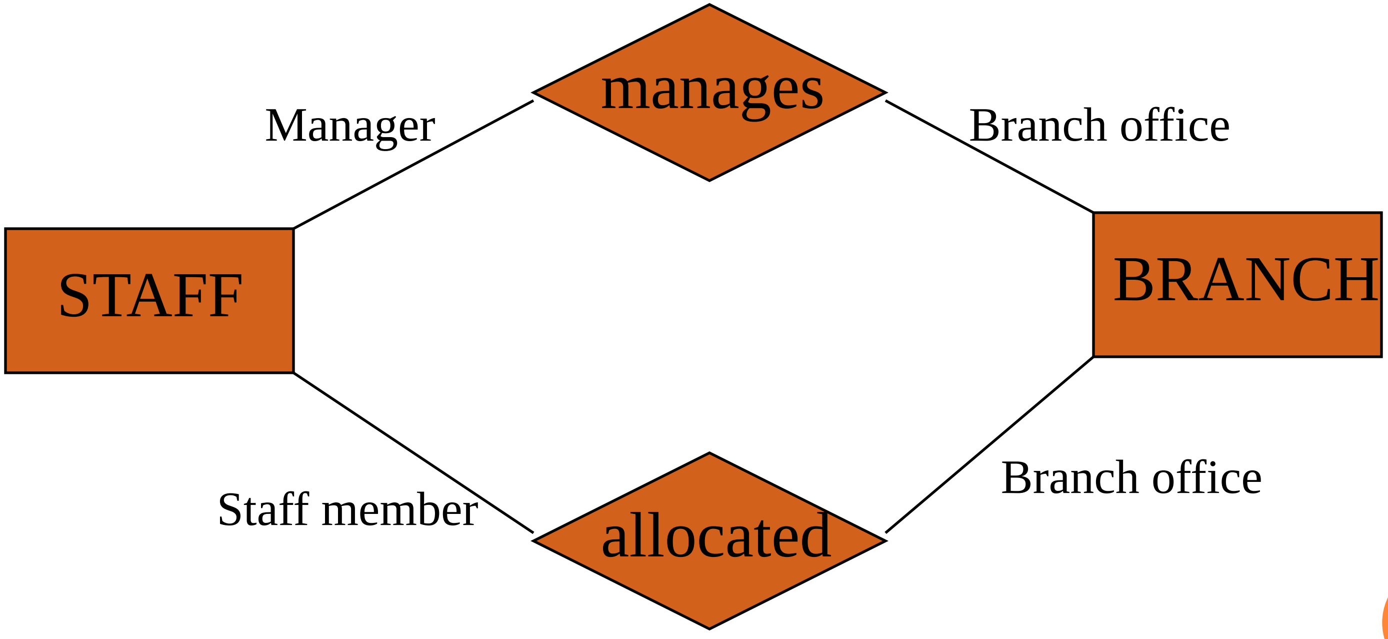
ROLES

Role indicates the purpose that each participating entity type plays in a relationship (e.g. prerequisite, requester)



Roles

Role can be used when two entities are associated through more than one relationship to classify the purpose of each relationship



Degree of Relationship Type

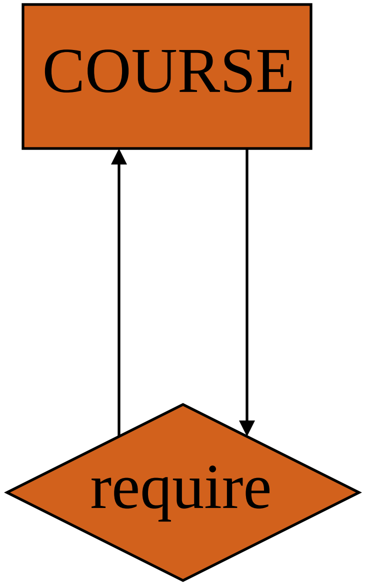
Degree of relationship refers to number of participating entity types in a relationship.

- A relationship of degree one (1 entity types) is **unary (recursive).**
- A relationship of degree two (2 entity types) are **binary.**
- A relationship of degree three (3 entity types) are **ternary.**

RECURSIVE RELATIONSHIP

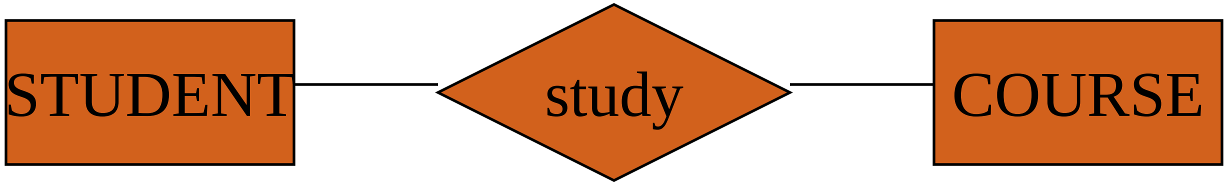
Recursive relationship is a relationship type where the same entity type participates more than once in a different role. It is a **unary relationship.**

- Has the degree 1



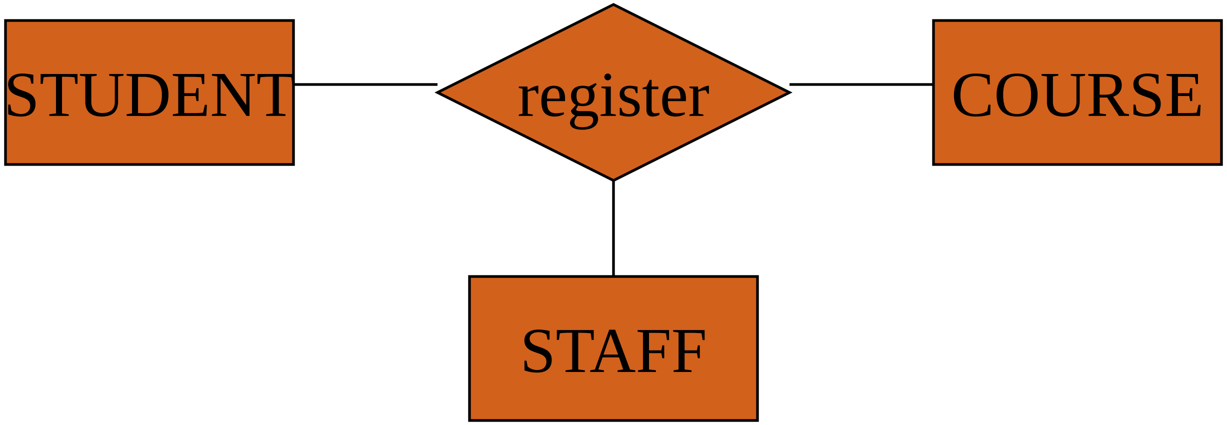
Degree of Relationship Type

A relationship of degree two (2 entity types) are **binary.**



Degree of Relationship Type

A relationship of degree three (3 entity types) are **ternary.** (e.g. registration of a student in a course by a staff)



Attributes of Relationship types

A relationship type can have attributes; for example, HoursPerWeek of WORKS_ON; its value for each relationship instance describes the number of hours per week that an EMPLOYEE works on a PROJECT.

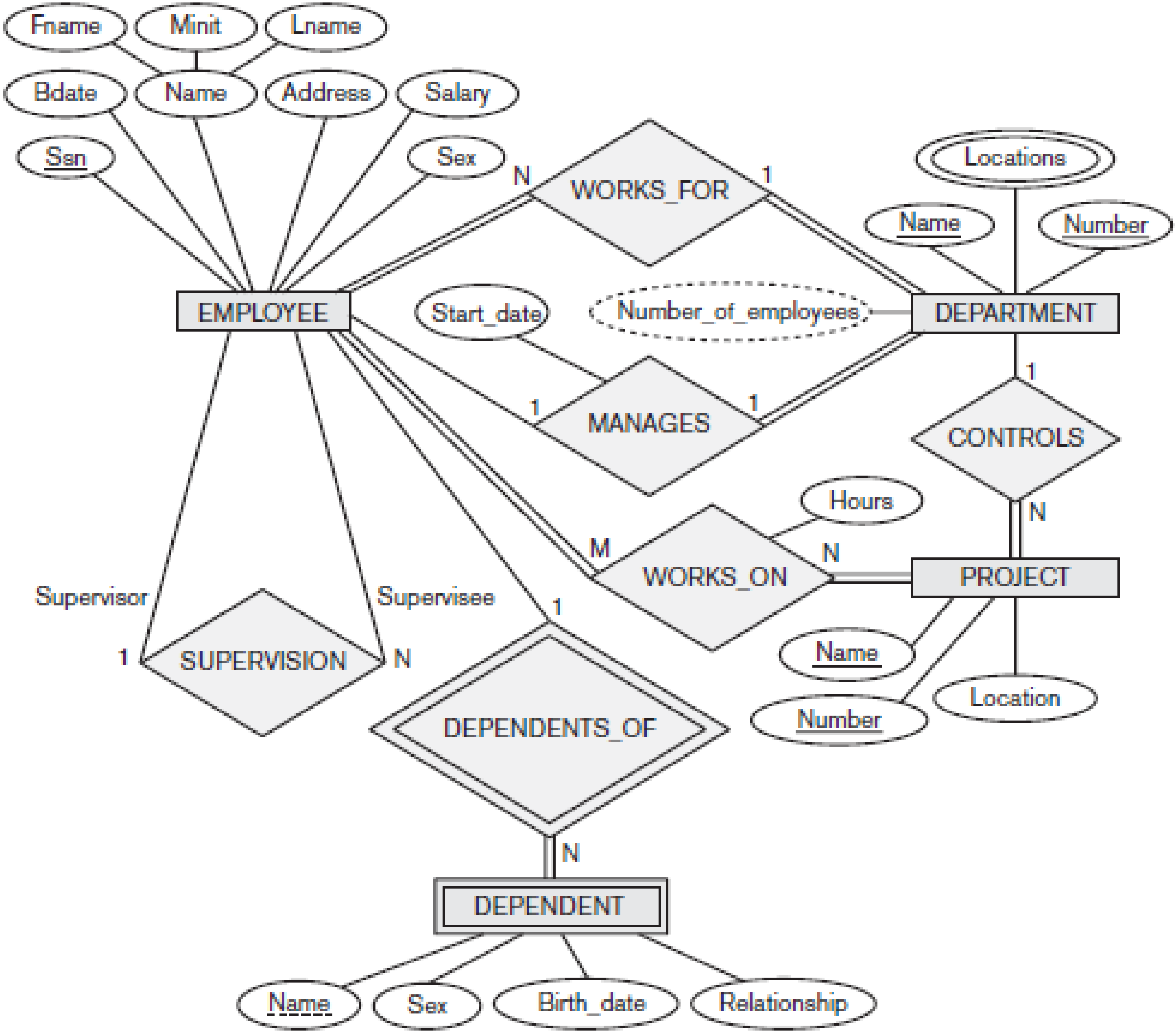


Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.

Constraints on Relationships

Constraints on Relationship Types (Also known as ratio constraints)

- – Maximum Cardinality
  - One-to-one (1:1)
  - One-to-many (1:N) or Many-to-one (N:1)
  - Many-to-many
- – Minimum Cardinality (also called participation constraint or existence dependency constraints)
  - zero (optional participation, not existence-dependent)
  - one or more (mandatory, existence-dependent)

CARDINALITIES

Cardinality ratio expresses the number of relationships an entity can participate in.

Most useful in describing binary relationship types.

For a binary relationship type the mapping cardinality must be one of the following types:

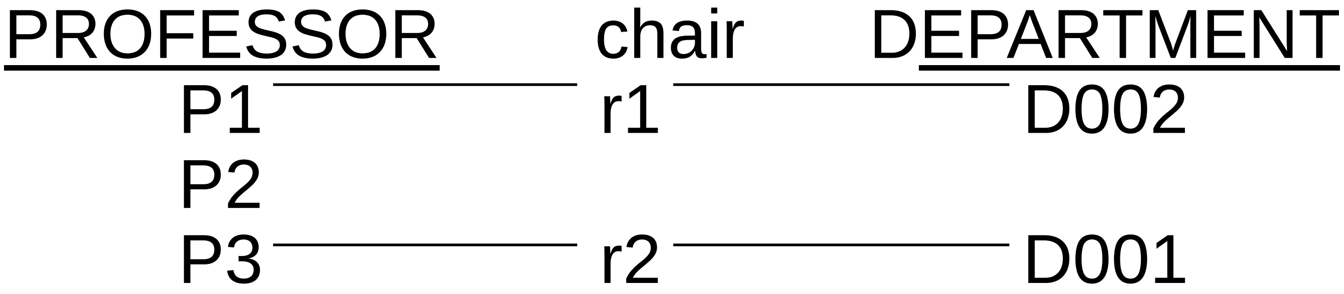
– One to one (1:1)

– Many to one (M:1)

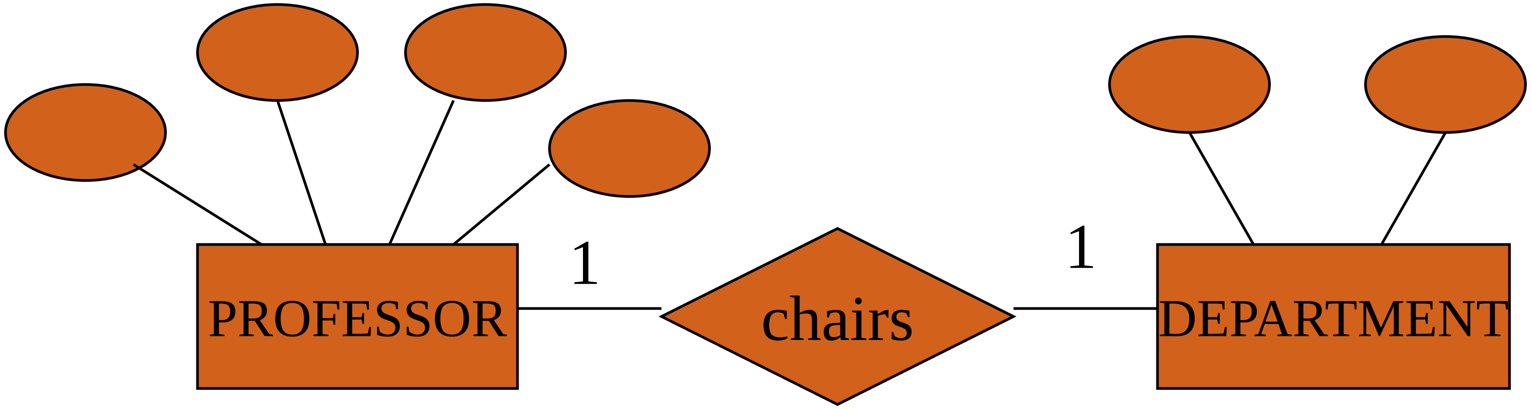
– One to many(1:M)

– Many to many (M:N)

ONE-TO-ONE RELATIONSHIP

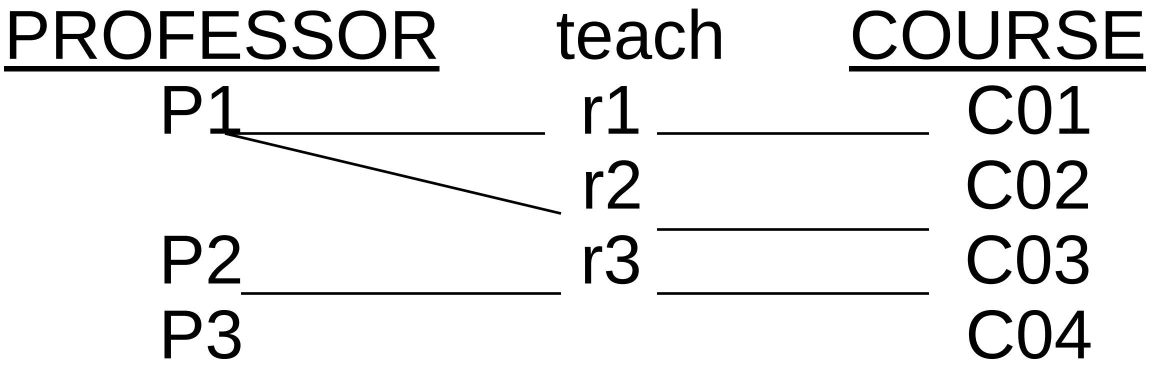


PROFESSOR	chair	DEPARTMENT
P1	r1	D002
P2		
P3	r2	D001



A professor chairs at most one department; and a department is chaired by only one professor

One-To-Many Relationship



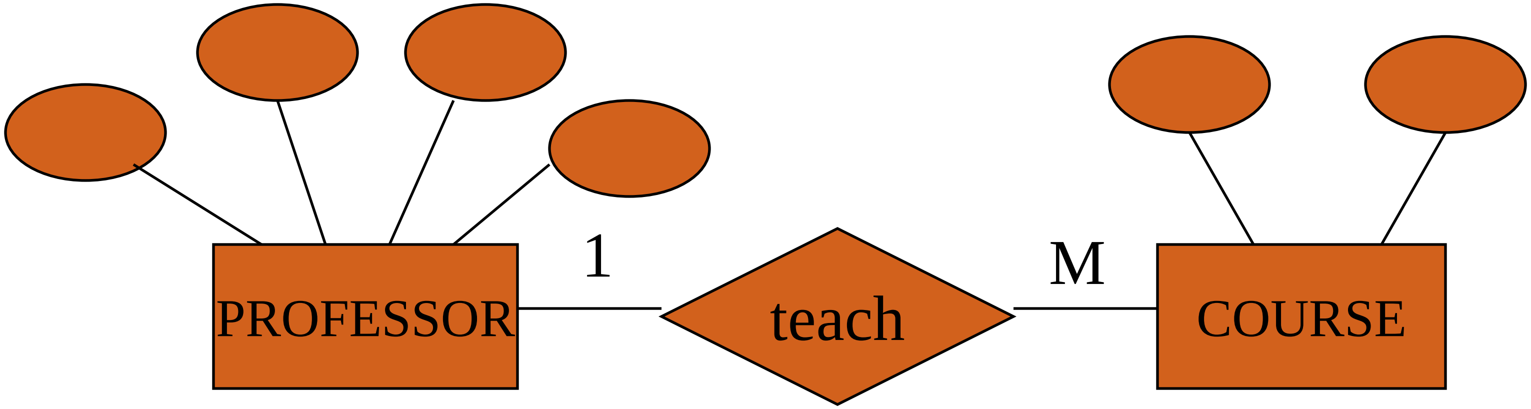
PROFESSOR teach COURSE

P1 r1 C01

r2 C02

P2 r3 C03

P3 C04



PROFESSOR

1

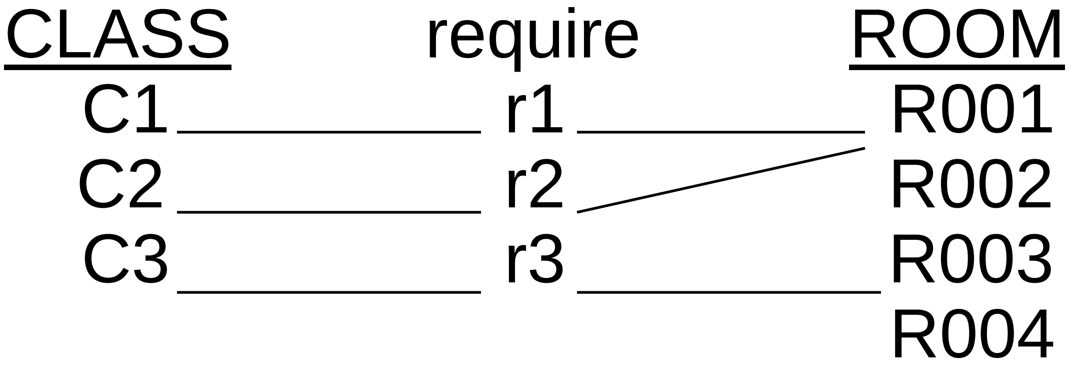
teach

M

COURSE

A course is taught by at most one professor; a professor teaches many courses.

Many-To-one Relationship



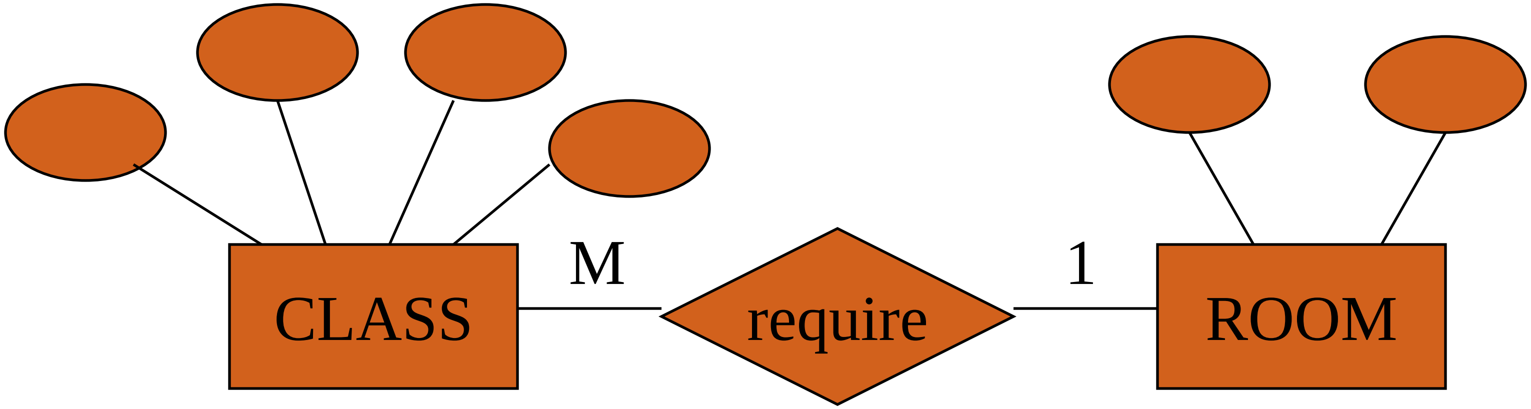
CLASS require ROOM

C1 r1 R001

C2 r2 R002

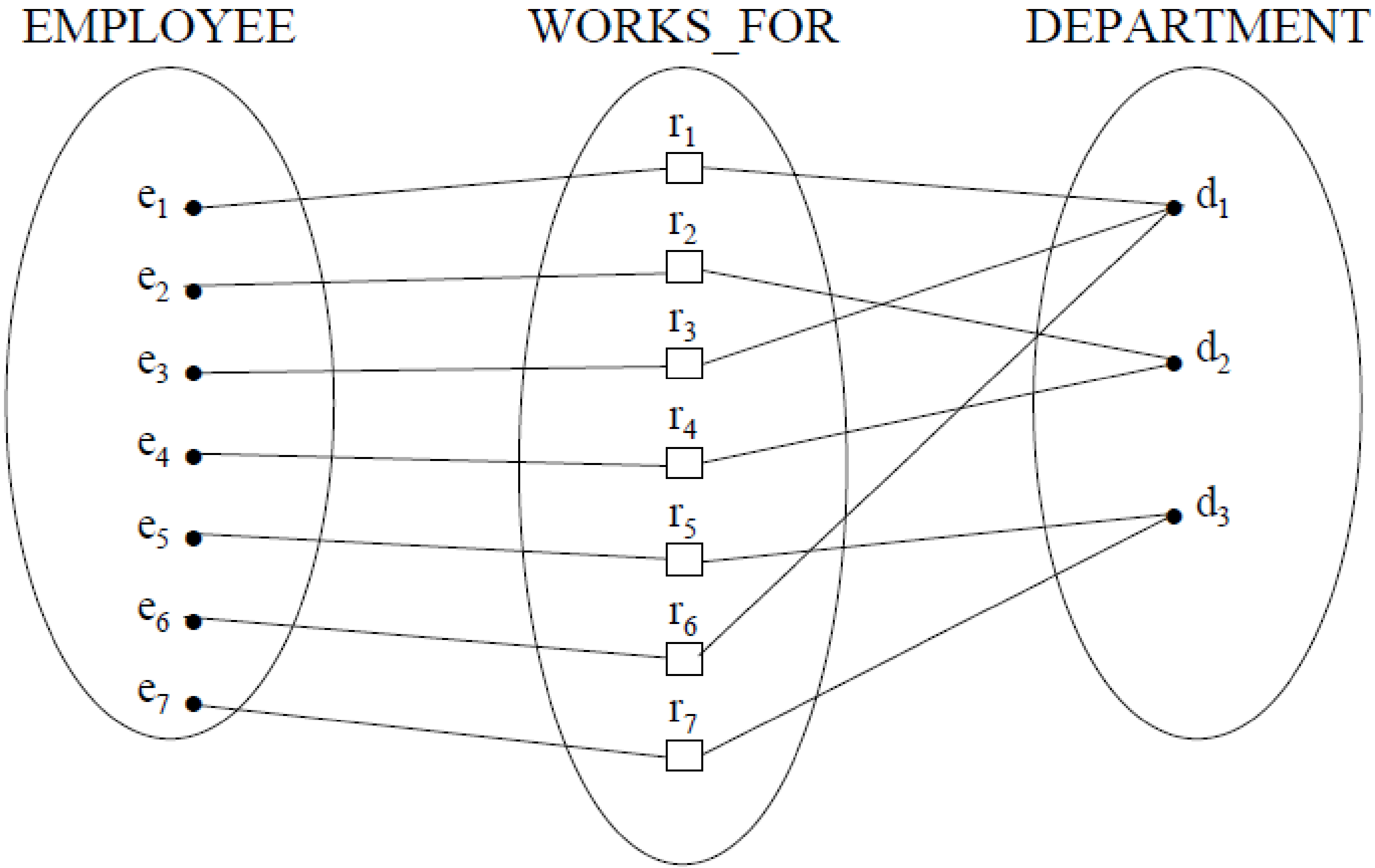
C3 r3 R003

R004

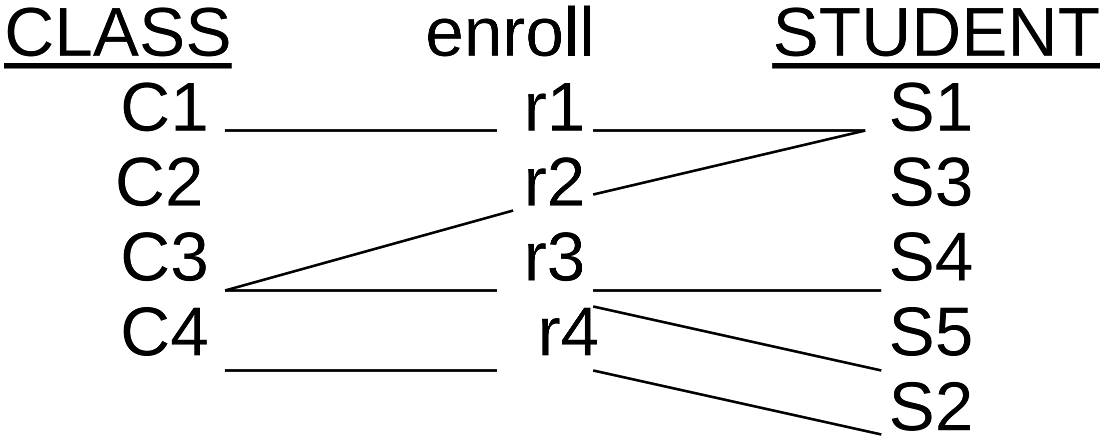


A class requires one room; while a room can be scheduled for many classes

Many-to-one (N:1) RELATIONSHIP



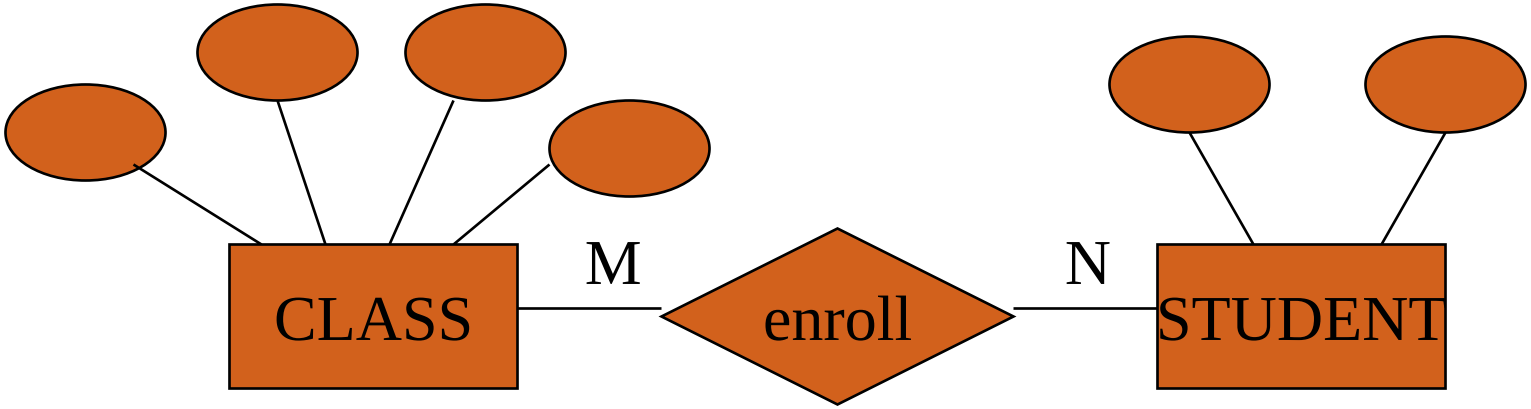
Many-To-Many Relationship



CLASS
C1
C2
C3
C4

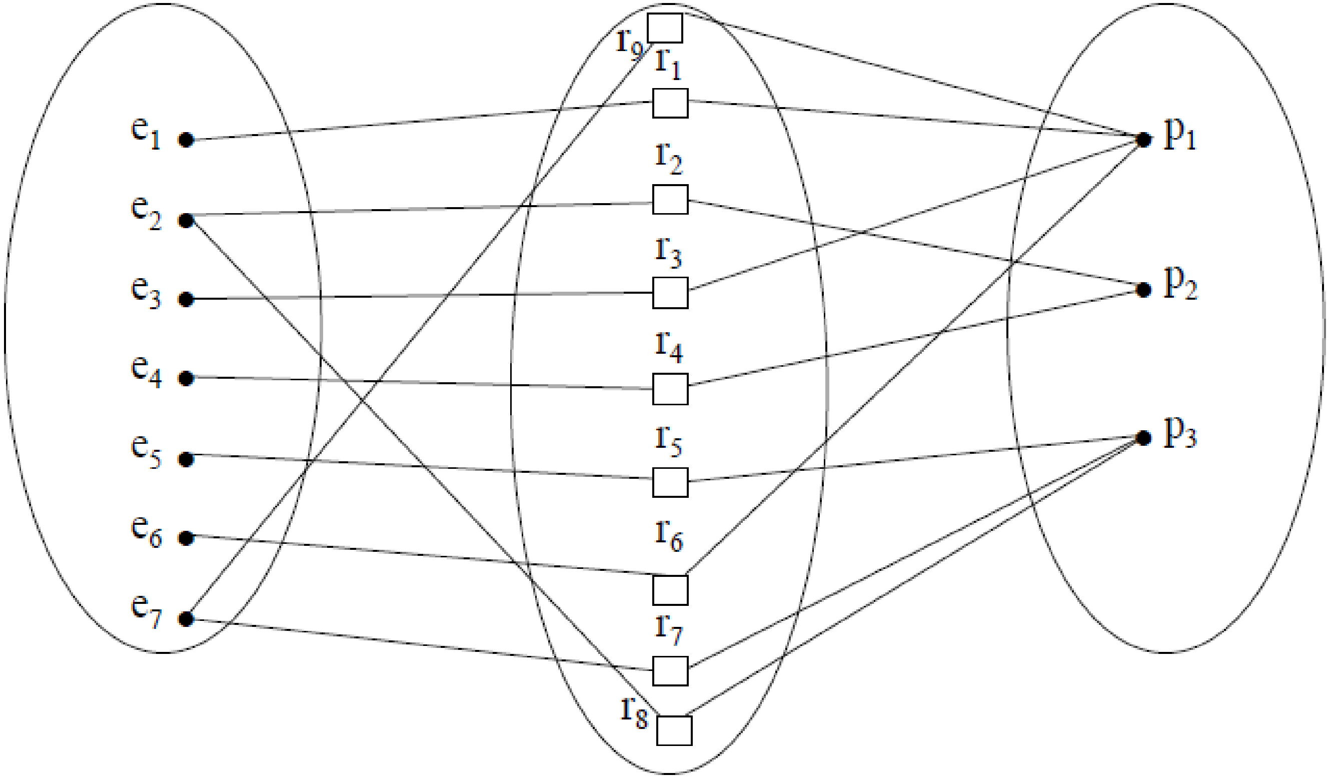
enroll
r1
r2
r3
r4

STUDENT
S1
S3
S4
S5
S2



A class enrolls many students; and each student is enrolled in many classes.

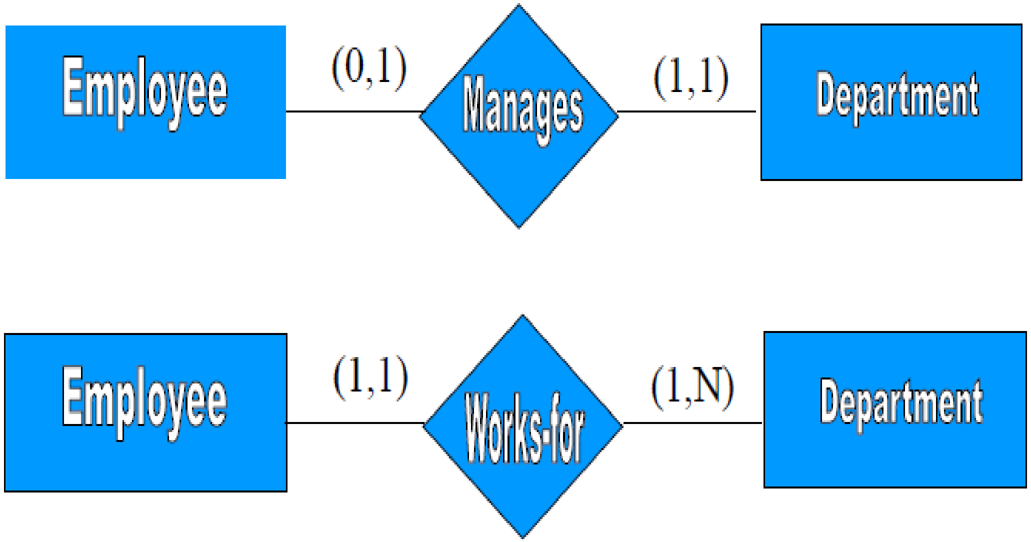
Many-to-many (M:N) RELATIONSHIP



ALTERNATIVE (MIN, MAX) NOTATION FOR RELATIONSHIP STRUCTURAL CONSTRAINTS:

- Specified on *each participation of an entity type E in a relationship type R*
- Specifies that each entity e in E participates in *at least min and at most max* relationship instances in R
- Default(no constraint): min=0, max=n
- Must have min<=max, min>=0, max >=1

The (min,max) notation relationship constraints



- **Examples:**
- **A department has *exactly one manager and an employee can manage at most one department.***
  - **Specify (0,1) for participation of EMPLOYEE in MANAGES**
  - **Specify (1,1) for participation of DEPARTMENT in MANAGES**
- **An employee can work for *exactly one department but a department can have any number of employees.***
  - **Specify (1,1) for participation of EMPLOYEE in WORKS_FOR**
  - **Specify (0,n) for participation of DEPARTMENT in WORKS_FOR**

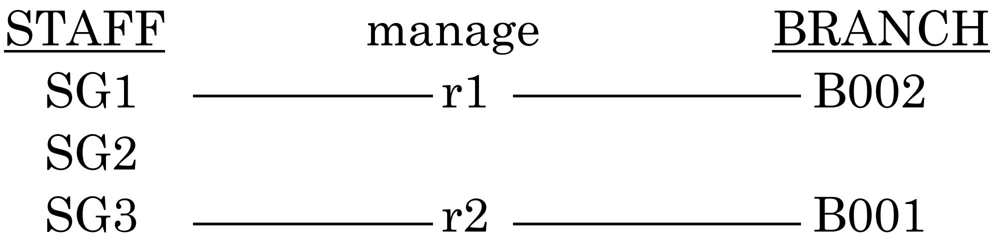
Multiplicity (min, max) notation

Multiplicity is the number (range) of possible entities that may relate to a single association through a particular relationship.

It is best determined using sample data.

Takes the form (min#,max#)

MULTIPLICITY (MIN, MAX) NOTATION

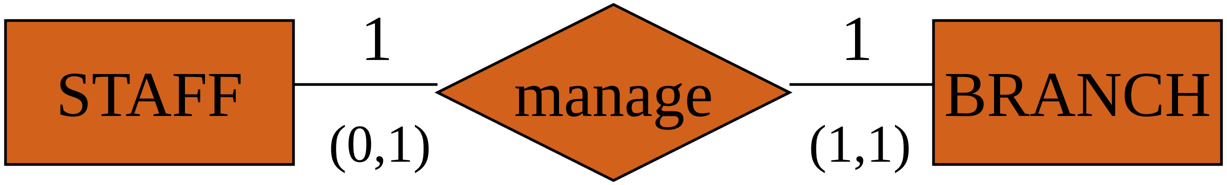


STAFF manage BRANCH

SG1 r1 B002

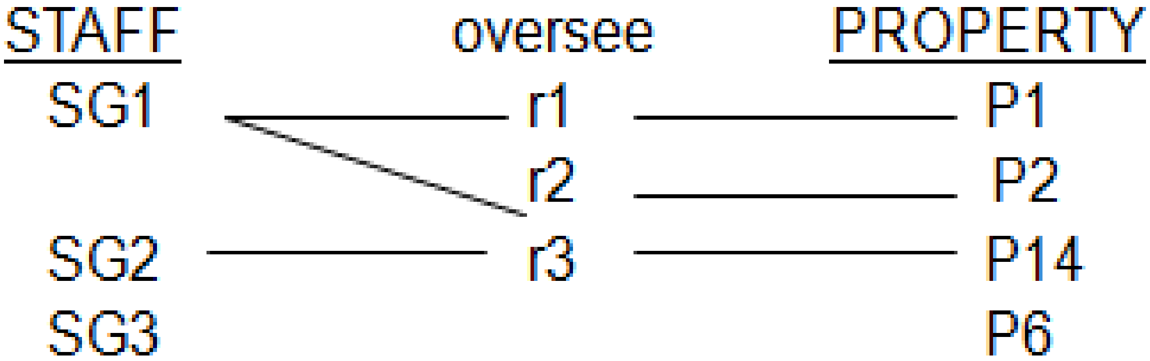
SG2

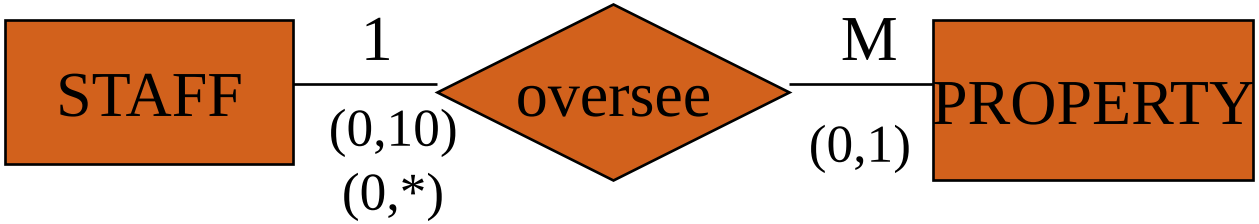
SG3 r2 B001



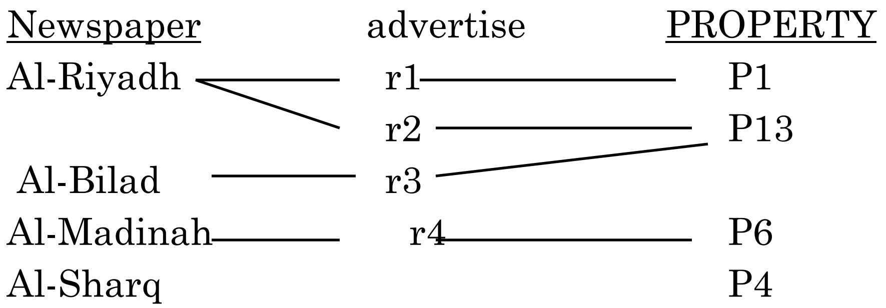
STAFF 1 (0,1) manage 1 (1,1) BRANCH

Multiplicity (min, max) notation





Multiplicity (min, max) notation



Newspaper advertise PROPERTY

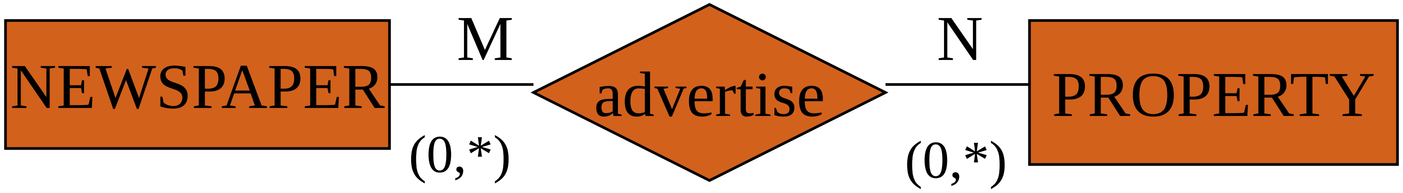
Al-Riyadh r1 P1

r2 P13

Al-Bilad r3

Al-Madinah r4 P6

Al-Sharq P4



NEWSPAPER M advertise N PROPERTY

(0,*) (0,*)

Constraints on Relationships

Constraints on Relationship Types (Also known as ratio constraints)

- – Maximum Cardinality
  - One-to-one (1:1)
  - One-to-many (1:N) or Many-to-one (N:1)
  - Many-to-many
- – Minimum Cardinality (also called participation constraint or existence dependency constraints)
  - zero (optional participation, not existence-dependent)
  - one or more (mandatory, existence-dependent)

Participation Constraints

Participation constraints determine whether all or only some entities participate in a relationship.

Two types of participation:

- **Mandatory** (total)
- **Optional** (partial)

Participation Constraints

- **Mandatory** (total) (1:*): if an entity's existence requires the existence of an associated entity in a particular relationship (existence-dependent).

    (e.g. CLASS taught-by PROFESSOR)

    CLASS is a *total participator* in the relation

A weak entity always has a mandatory participation constraints but the opposite not always true.

Participation Constraints

- **Optional** (partial) (0:*): if an entity's existence does not require a corresponding entity in a particular relationship. (Not existence-dependent)

(e.g. PROFESSOR teach CLASS)

PROFESSOR is a *partial participator* in the relation

Structural Constraints –one way to express semantics of relationships

- **Structural constraints on relationships:**
  - **Cardinality ratio (of a binary relationship): 1:1, 1:N N:1,** or M:N
  - **Shown By Placing Appropriate Number On The Link.**
- **Participation constraint (on each participating entity type):**
  - total *or partial.*
  - Total **Shown By Double Lining The Link** (Total Called *Existence Dependency)*
- NOTE: These are easy to specify for Binary Relationship Types.

Data Modeling Tools

- A number of popular tools that cover conceptual modeling and mapping into relational schema design. Examples: ERWin, S- Designer (Enterprise Application Suite), ER- Studio, etc. POSITIVES: serves as documentation of application requirements, easy user interface - mostly graphics editor support

Problems with Current Modeling Tools

- DIAGRAMMING

– Poor conceptual meaningful notation.

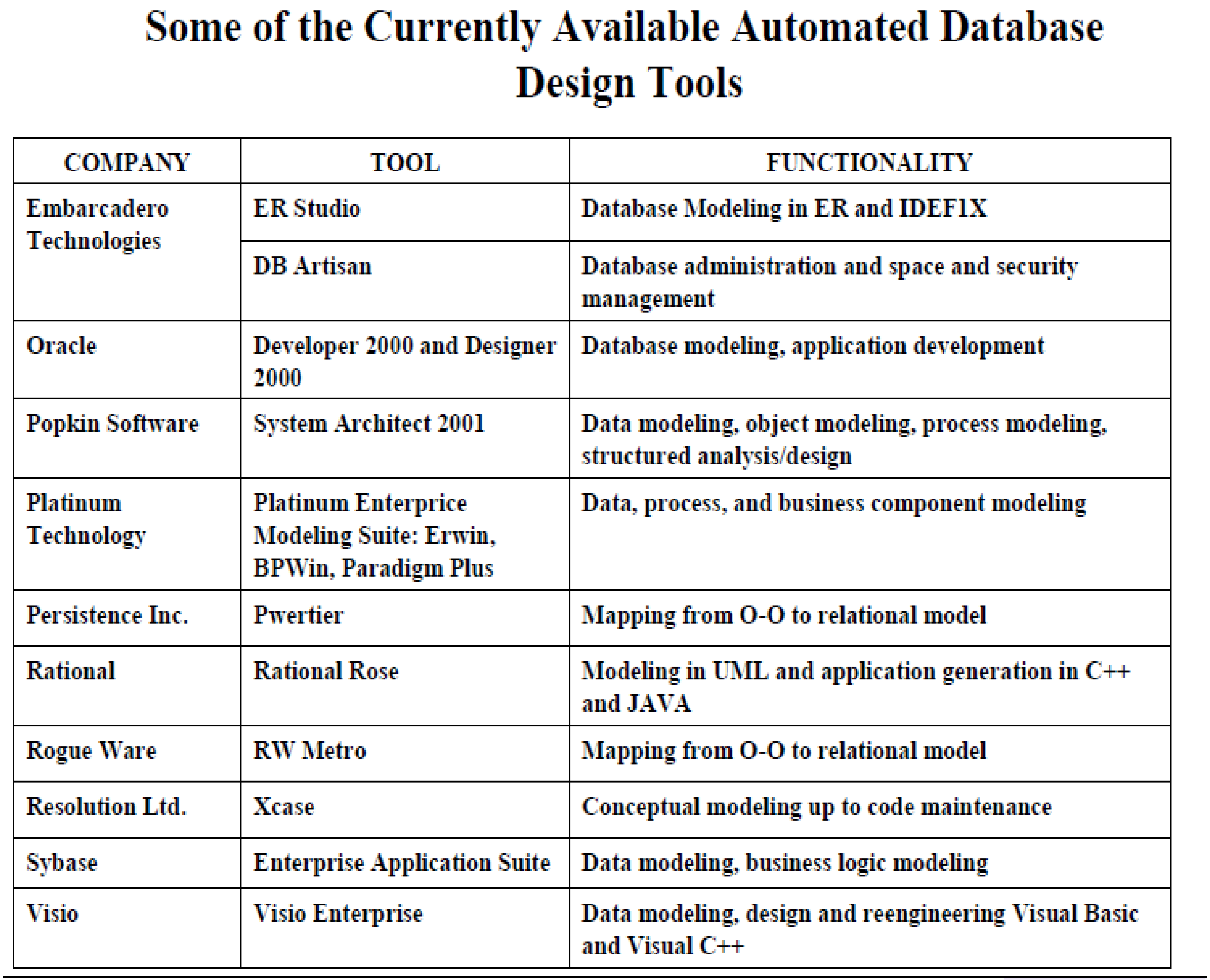
– To avoid the problem of layout algorithms and aesthetics of diagrams, they prefer boxes and lines and do nothing more than represent (primary-foreign key) relationships among resulting tables.(a few exceptions)

- METHODOLGY

– lack of built-in methodology support.

– poor tradeoff analysis or user-driven design preferences.

– poor design verification and suggestions for improvement.



Some of the Currently Available Automated Database Design Tools

COMPANY	TOOL	FUNCTIONALITY
Embarcadero Technologies	ER Studio	Database Modeling in ER and IDEF1X
	DB Artisan	Database administration and space and security management
Oracle	Developer 2000 and Designer 2000	Database modeling, application development
Popkin Software	System Architect 2001	Data modeling, object modeling, process modeling, structured analysis/design
Platinum Technology	Platinum Enterprice Modeling Suite: Erwin, BPWin, Paradigm Plus	Data, process, and business component modeling
Persistence Inc.	Pwertier	Mapping from O-O to relational model
Rational	Rational Rose	Modeling in UML and application generation in C++ and JAVA
Rogue Ware	RW Metro	Mapping from O-O to relational model
Resolution Ltd.	Xcase	Conceptual modeling up to code maintenance
Sybase	Enterprise Application Suite	Data modeling, business logic modeling
Visio	Visio Enterprise	Data modeling, design and reengineering Visual Basic and Visual C++

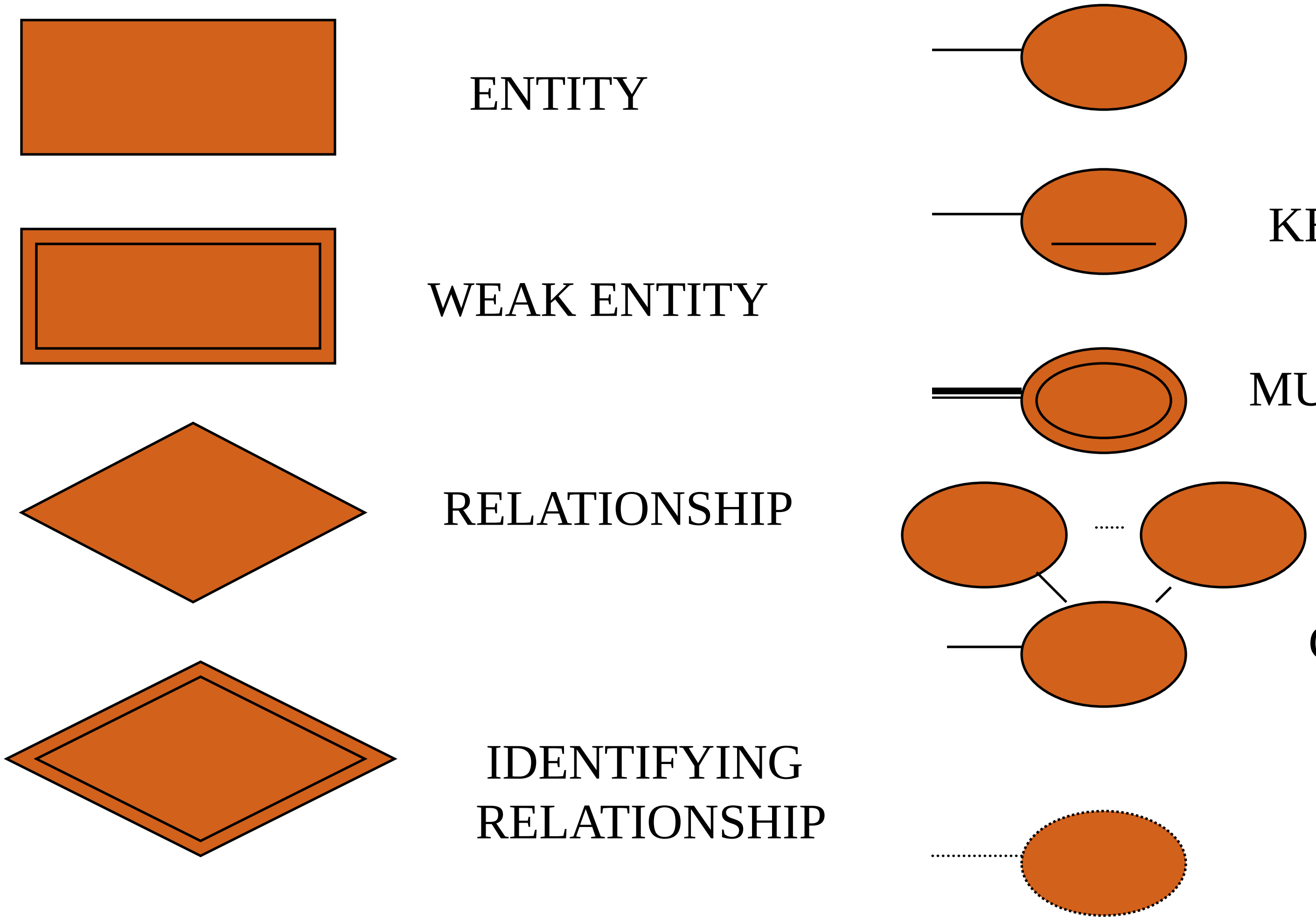
PROBLEM WITH ER NOTATION

- The entity relationship model in its original form did not support the specialization/ generalization abstractions

Extended Entity-Relationship (EER) Model

- Incorporates Set-subset relationships
- Incorporates Specialization/Generalization Hierarchies

SUMMARY OF ERD NOTATIONS (1)



ENTITY

WEAK ENTITY

RELATIONSHIP

IDENTIFYING RELATIONSHIP

ATTRIBUTE

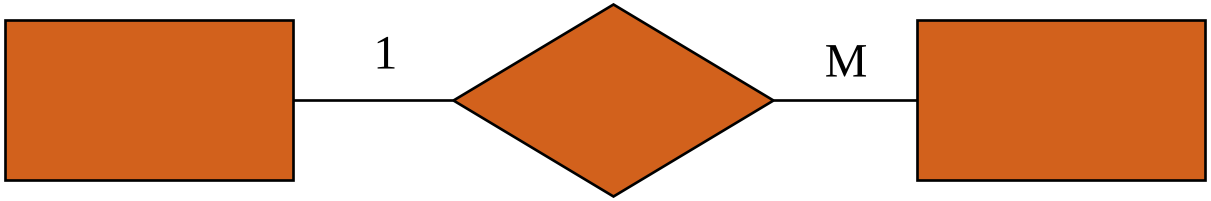
KEY ATTRIBUTE

MULTI-VALUED

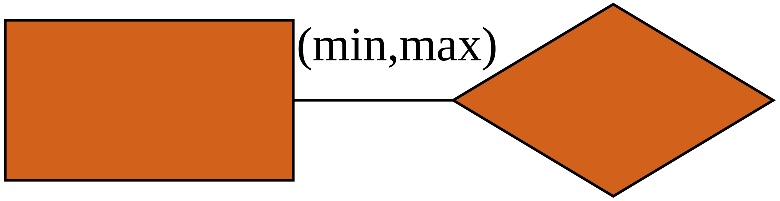
COMPOSITE

DERIVED

Summary of ERD notations (2)



CARDINALITY RATION



PARTICIPATION CONSTRAINTS

Example COMPANY Database

Requirements of the Company (oversimplified for illustrative purposes)
– The company is organized into DEPARTMENTs.
Each department has a name, number and an employee who *manages the department. We keep* track of the start date of the department manager.

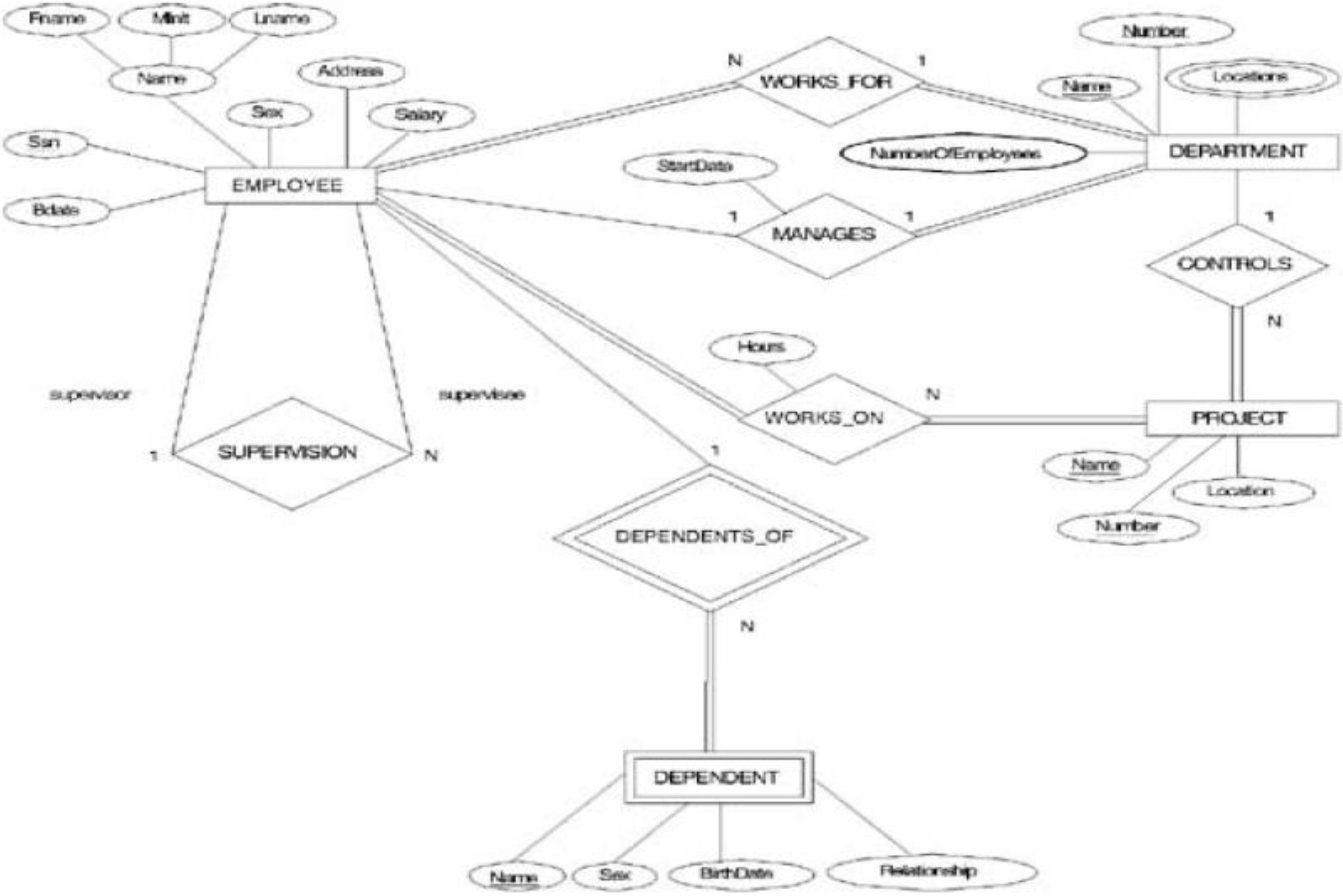
– Each department *controls a number of PROJECTs.* Each project has a name, number and is located at a single location.

EXAMPLE COMPANY DATABASE (CONT.)

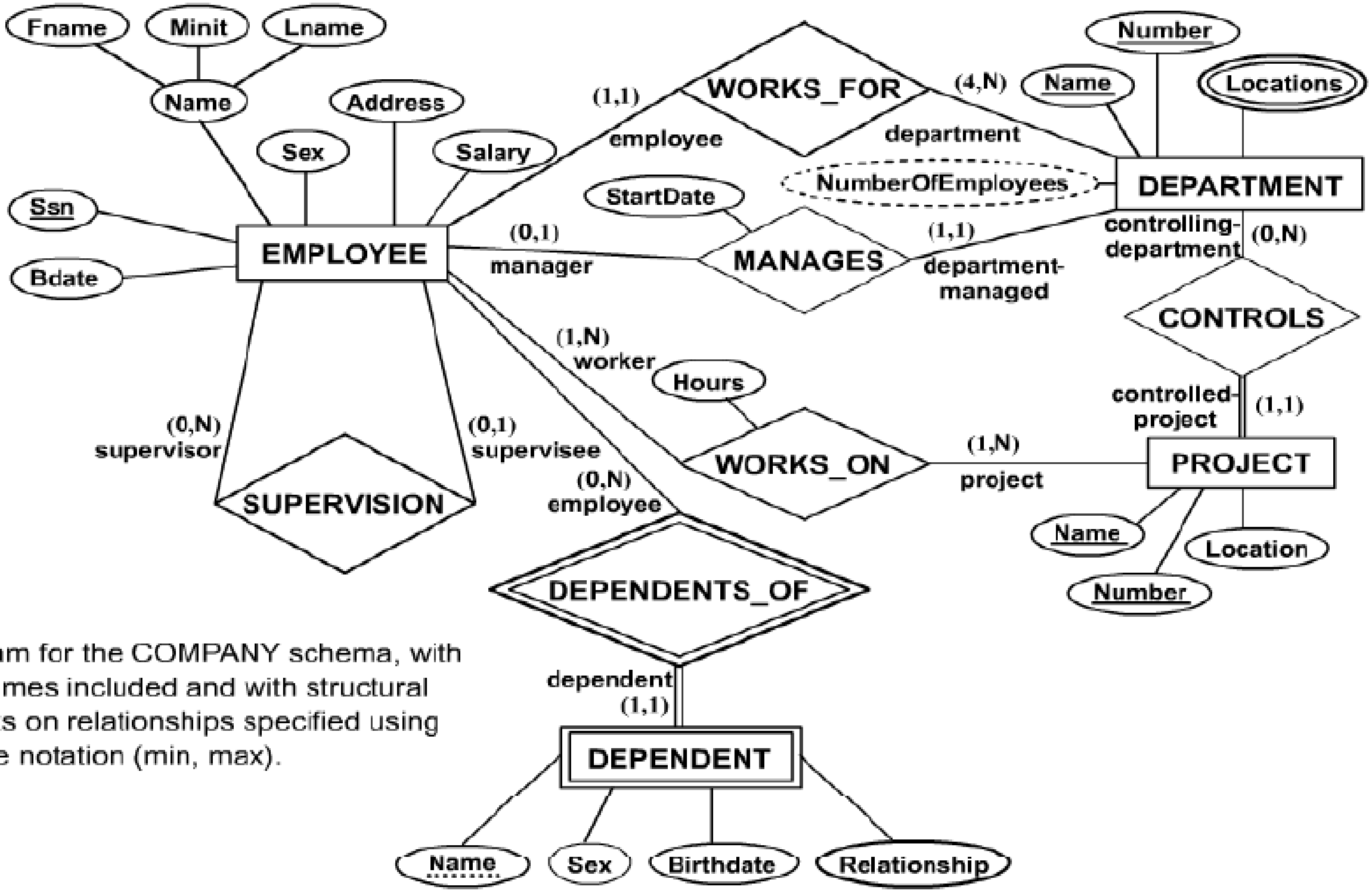
–We store each EMPLOYEE’s social security number, address, salary, sex, and birthdate. Each employee *works for one department but may work on several* projects. We keep track of the number of hours per week that an employee currently works on each project. We also keep track of the *direct supervisor* of each employee.

–Each employee may *have a number of* DEPENDENTs. For each dependent, we keep track of their name, sex, birthdate, and relationship to employee.

ER DIAGRAM – ENTITY TYPES ARE: EMPLOYEE, DEPARTMENT, PROJECT, DEPENDENT



COMPANY ER SCHEMA DIAGRAM USING (MIN, MAX) NOTATION



ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using alternative notation (min, max).

ER DIAGRAM FOR A BANK DATABASE

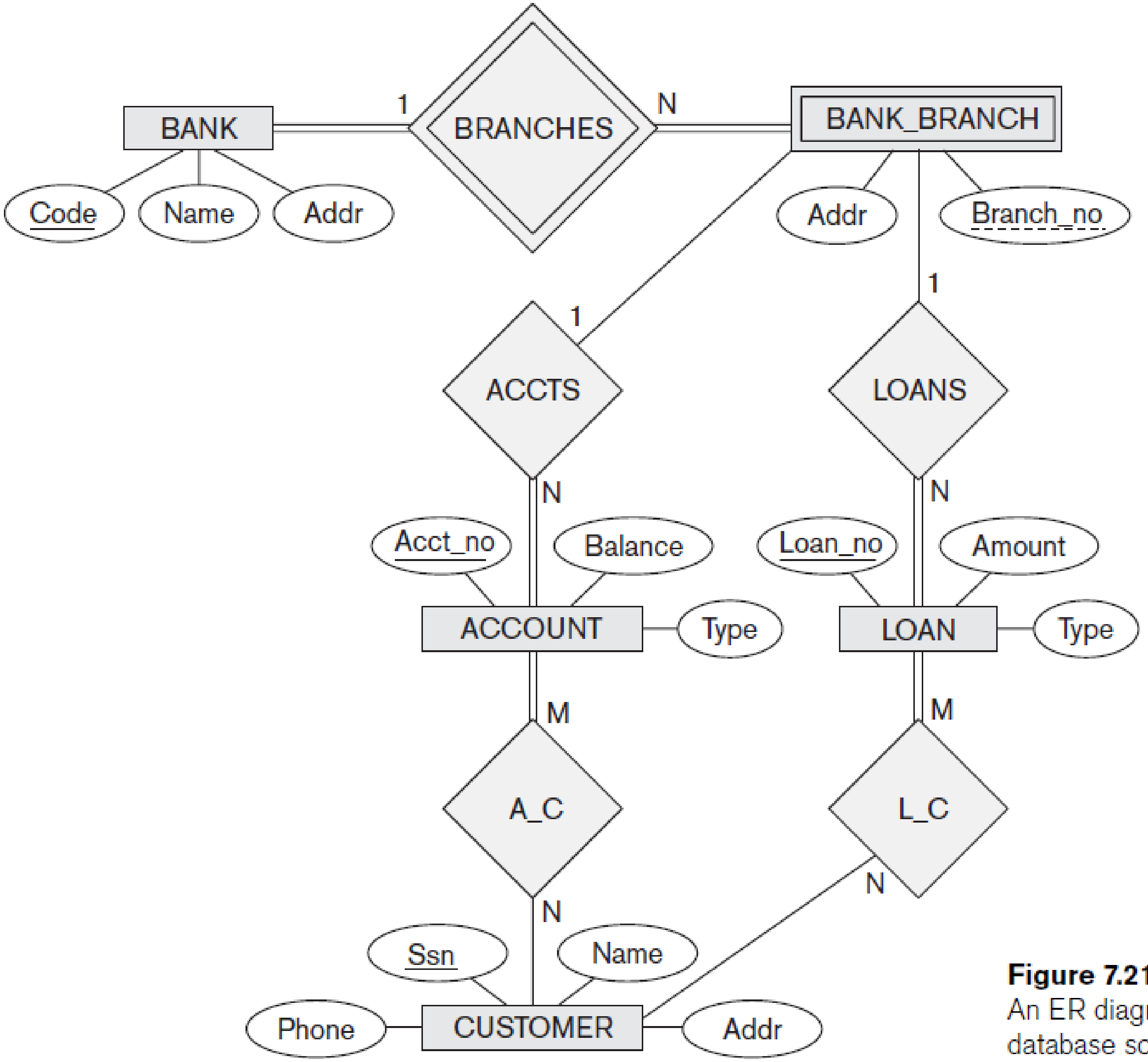
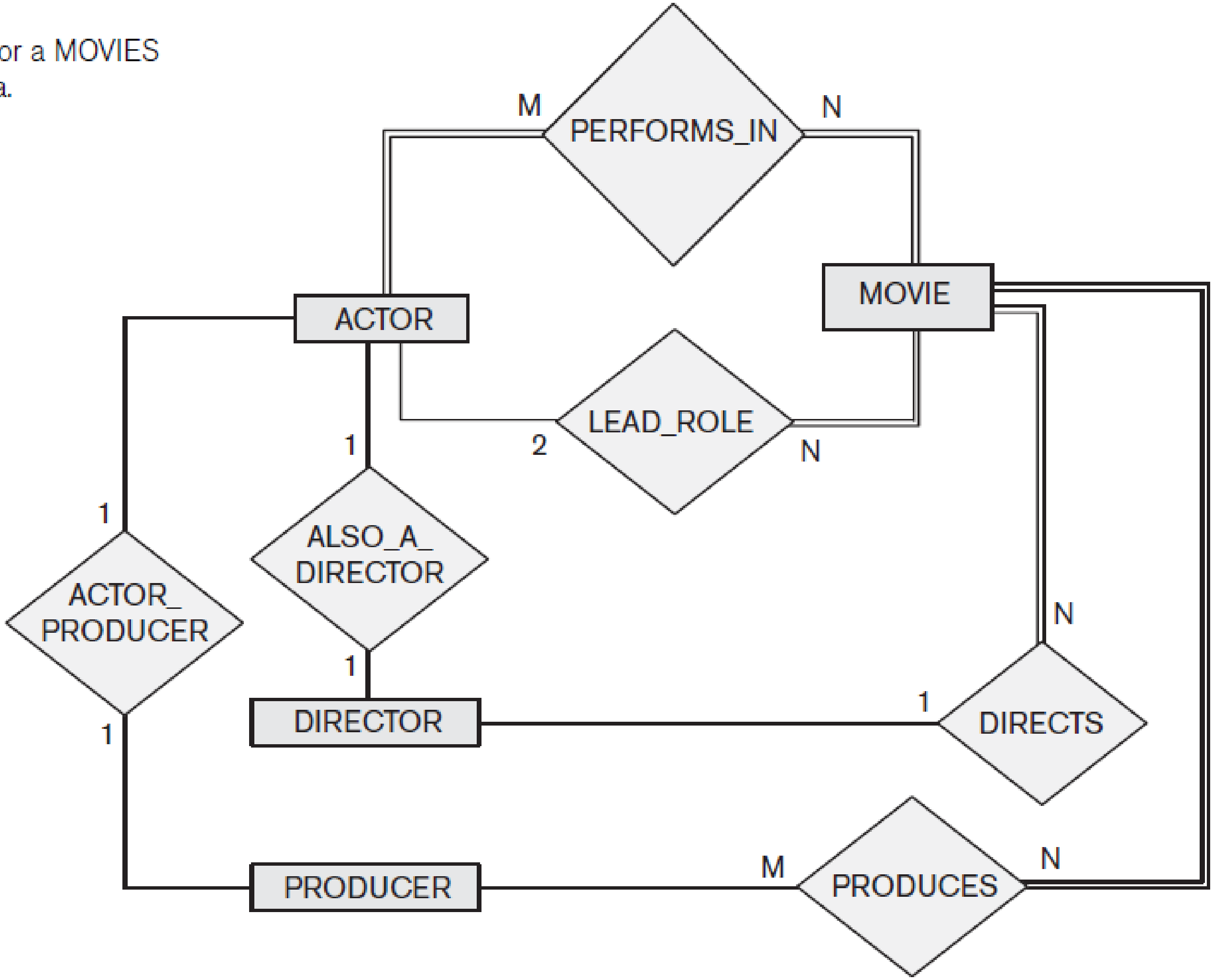


Figure 7.21
An ER diagram for a BANK database schema.

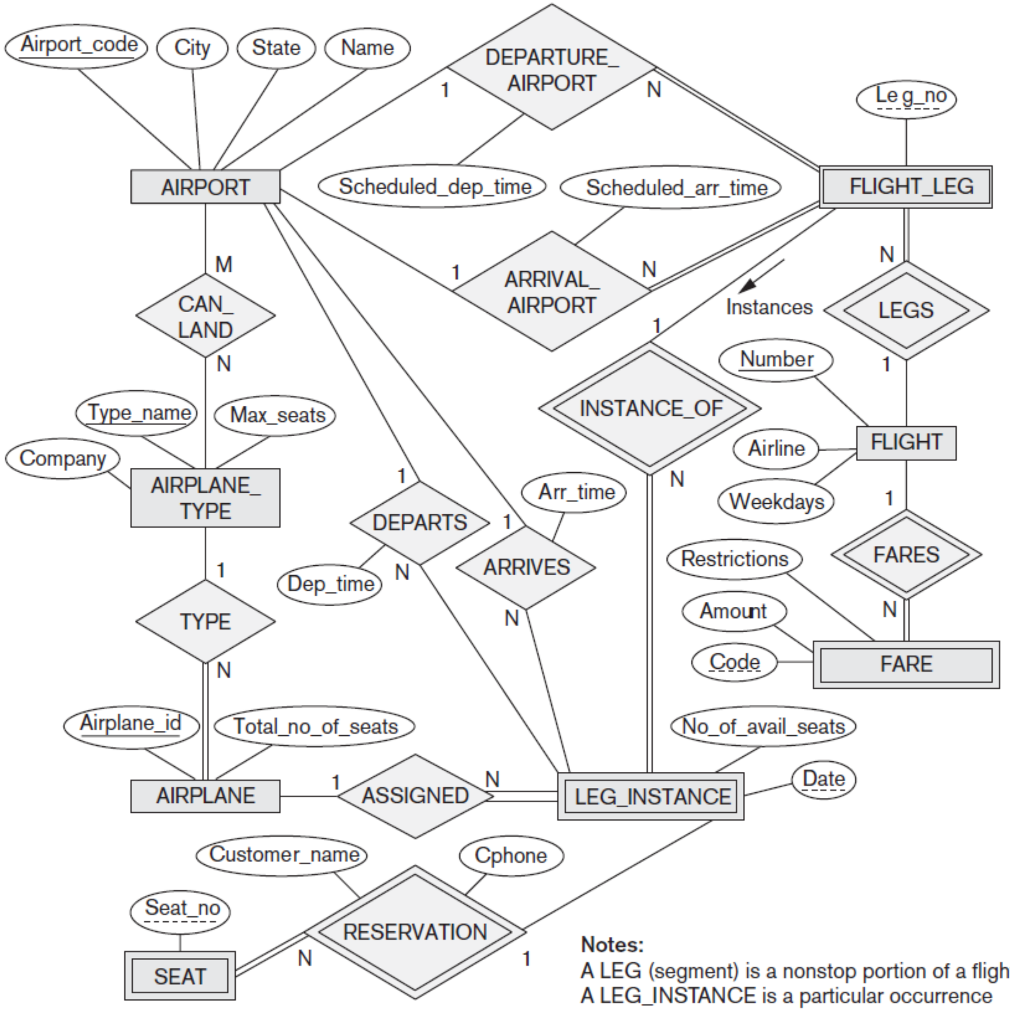
ER DIAGRAM FOR A MOVIES DATABASE

Figure 7.24
An ER diagram for a MOVIES database schema.



ER DIAGRAM FOR AIRLINE DATABASE

Figure 7.20
An ER diagram for an AIRLINE database schema.



Notes:
A LEG (segment) is a nonstop portion of a flight
A LEG_INSTANCE is a particular occurrence
of a LEG on a particular date.

THANK YOU!!!