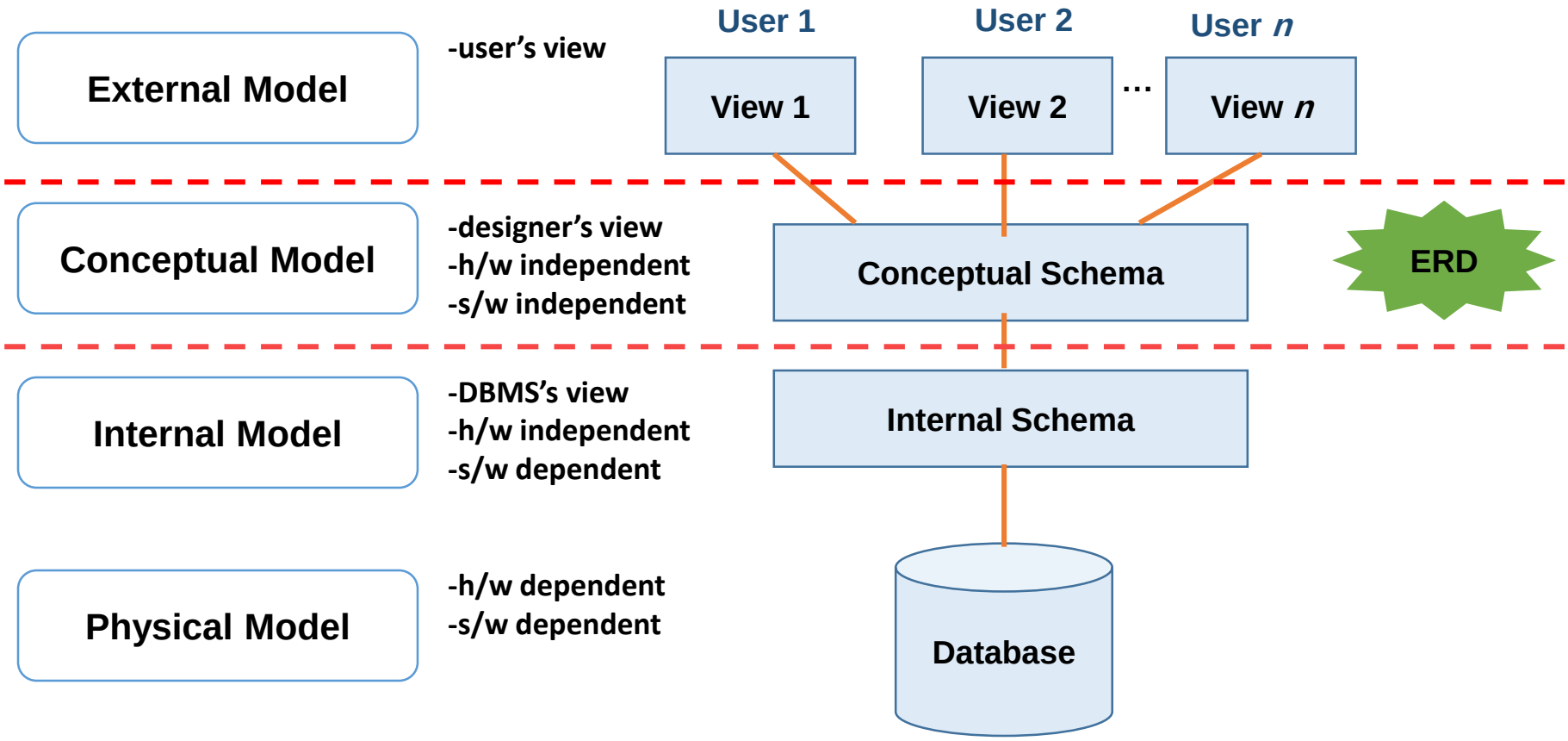
Database Design

Building ER diagram

Chapter 4: Entity Relationship (E-R) Modeling

Basic Modeling Concept



Example of a Database

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Main characteristics of the database approach

- Self-describing nature of a database system:
  - A DBMS catalog stores the description of a particular database (e.g. data structures, types, and constraints).
  - The description is called meta-data.
  - This allows the DBMS software to work with different database applications.
- Insulation between programs and data:
  - Called program-data independence.
  - Allows changing data structures and storage organization without having to change the DBMS access programs.
- Data abstraction:
  - A data model is used to hide storage details and present the users with a conceptual view of the database.
  - Programs refer to the data model constructs rather than data storage details.
- Support of multiple views of the data:
  - Each user may see a different view of the database, which describes only the data of interest to that user.

Database Design Process

- Two main activities:
  - Database design.
  - Applications design.
- Focus in this course on database design.
  - To design the conceptual, logical model for a database application.
- Applications design focuses on the programs and interfaces that access the database.
  - Generally considered part of software engineering.

Database Design Process

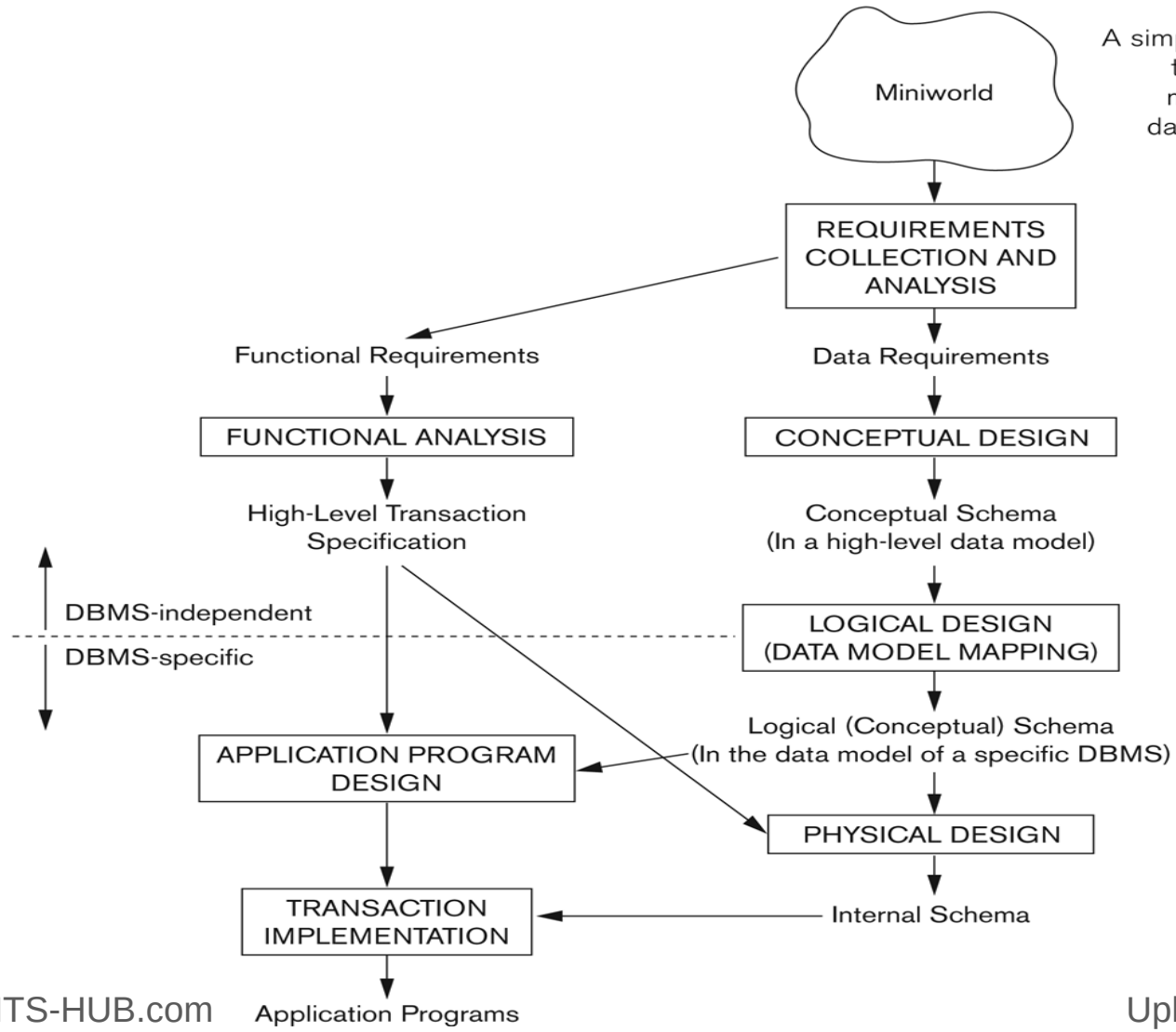
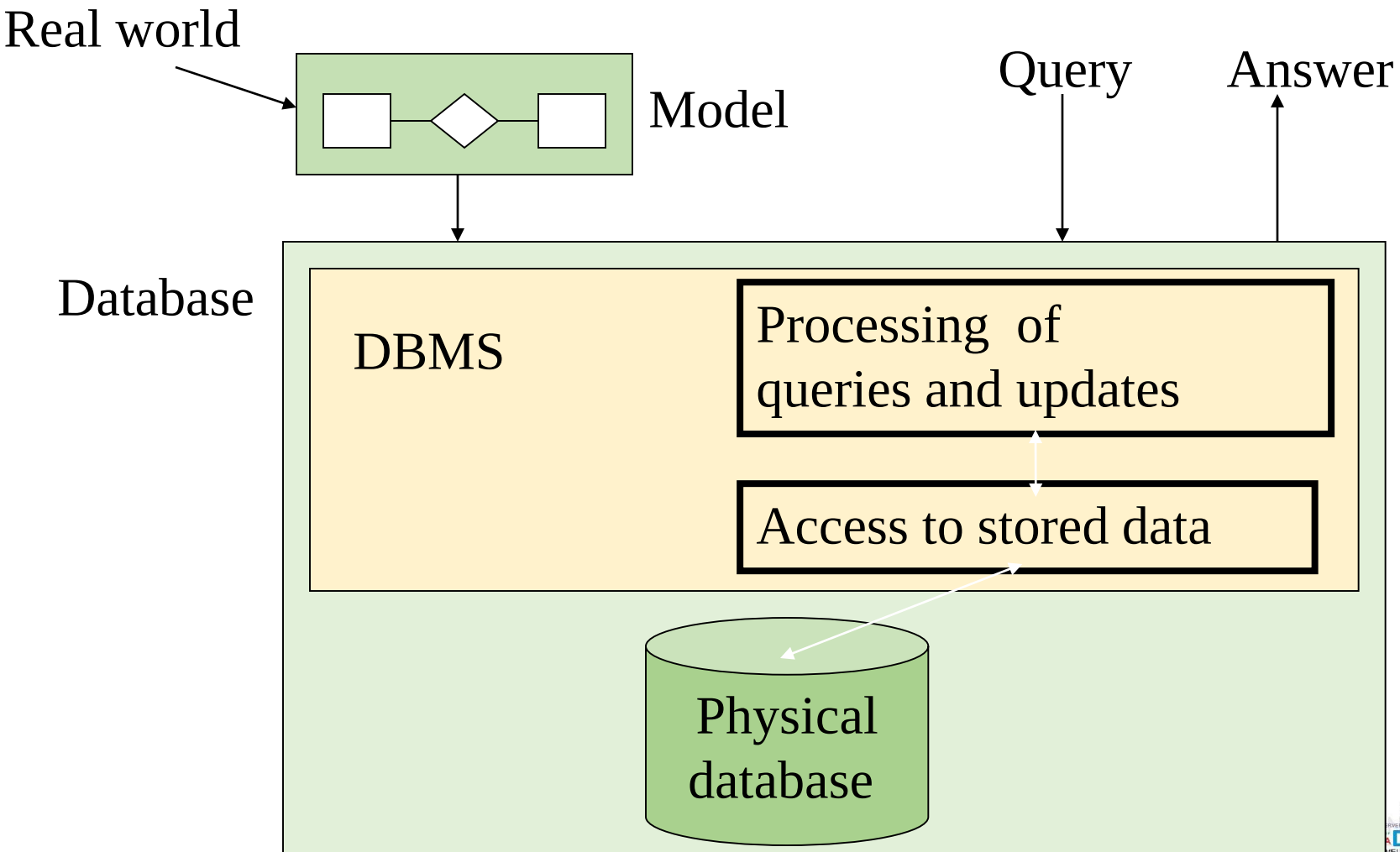


Figure 3.1
A simplified diagram to illustrate the main phases of database design.

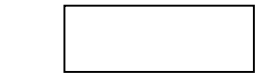
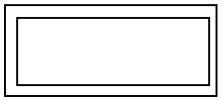
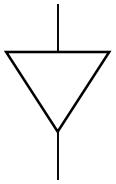
Overview

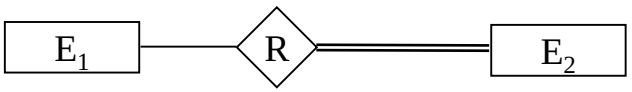


Entity-relationship (ER) model

- High-level conceptual data model.
  - An overview of the database.
  - Easy to discuss with non-database experts.
  - Easy to translate to data model of DBMS.
- ER diagram.
- Based on modelling objects in the real-world.

ER Notations & Symbols

Symbol	Meaning
	ENTITY TYPE
	WEAK ENTITY TYPE
	RELATIONSHIP TYPE
	IDENTIFYING RELATIONSHIP TYPE
	ATTRIBUTE
	KEY ATTRIBUTE
	MULTIVALUED ATTRIBUTE
	DERIVED ATTRIBUTE
	DISCRIMINATING ATTRIBUTE OF WEAK ENTITY SET
	ISA (SPECIALIZATION OR GENERALIZATION

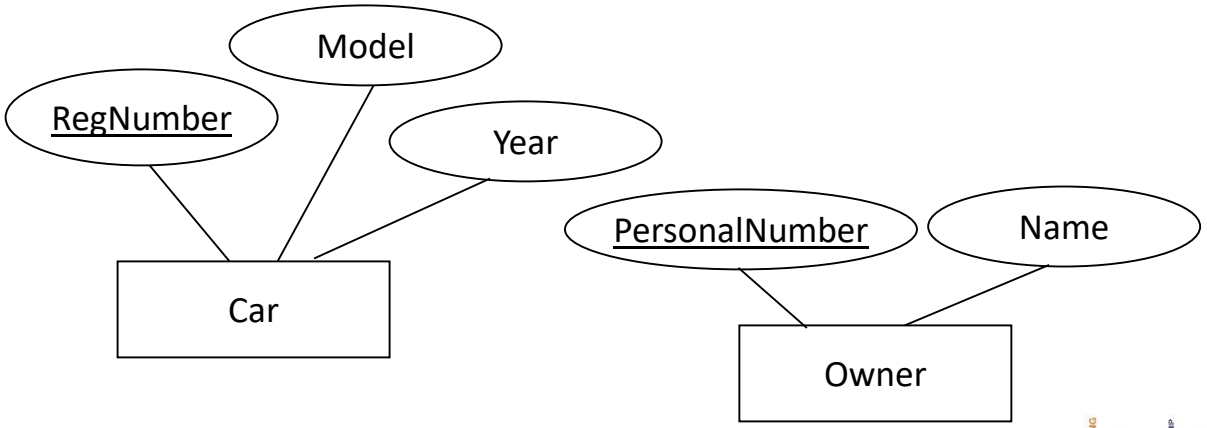


TOTAL PARTICIPATION OF E_2 IN R

CARDINALITY RATIO 1:N FOR E_1:E_2 IN R

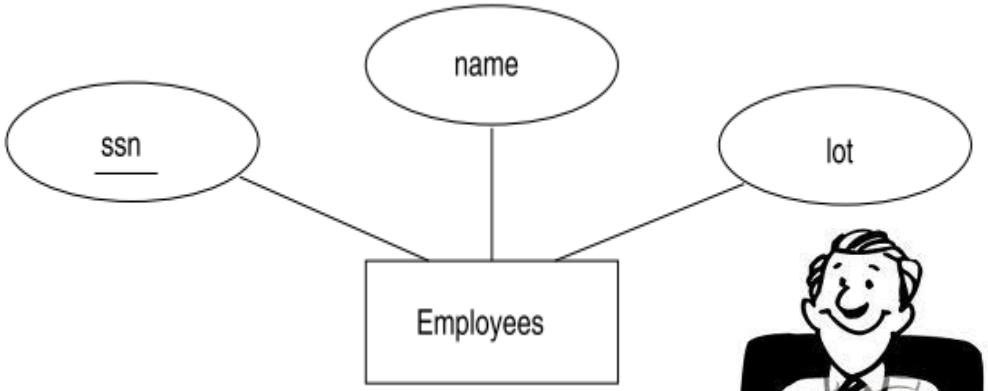
Entity and Entity type/set

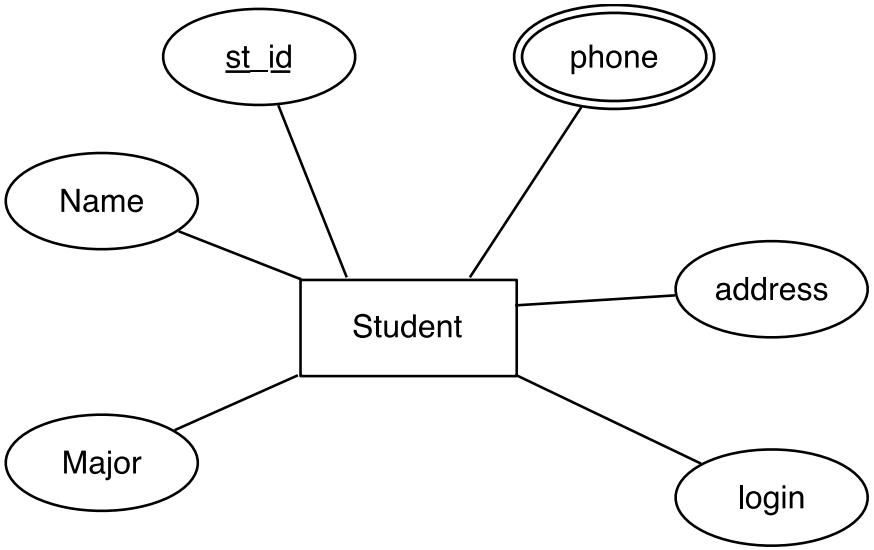
- Entity: A ”thing” in the real world with an independent existence.
- Attributes: Properties that describes an entity.
- Entity type: A collection of entities that have the same set of attributes.

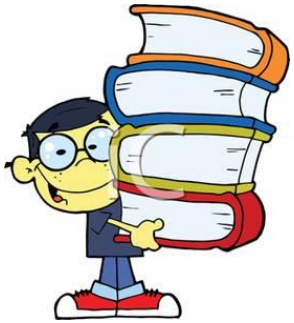


Conceptual Design

- Entities
- Attributes
- Attribute Domain
- Key
- Primary Key
- Candidate Keys



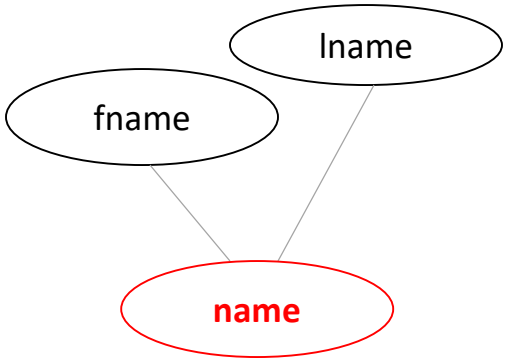




Attributes

Composite Attributes

- Can be divided into further parts.
- Ex: Name → First Name, Middle Name, Last Name



Simple Attributes

- Cannot be divided further.
- Ex: Weight/Age/Salary

Attributes

Single-valued attributes

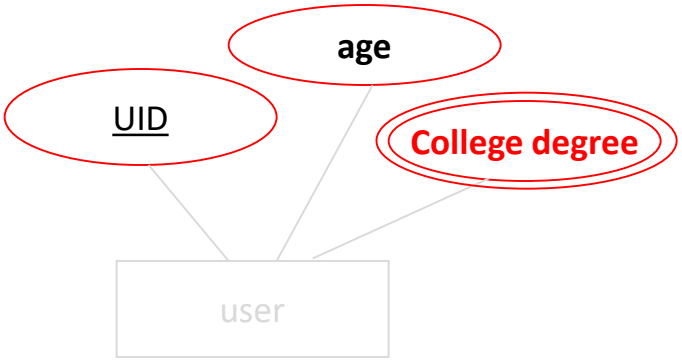
- Have a single value for a particular entity

- Ex: Age

Multivalued attributes

- Can have a set of values for a particular entity

- Ex: College Degree, Language known



Attributes

Derived Attributes

- Can be derived from other attributes

Stored Attributes

- From which the value of other attributes are derived.

- Ex: Age → can be derived from date of birth

- Ex: BirthDate of a person

Complex Attributes

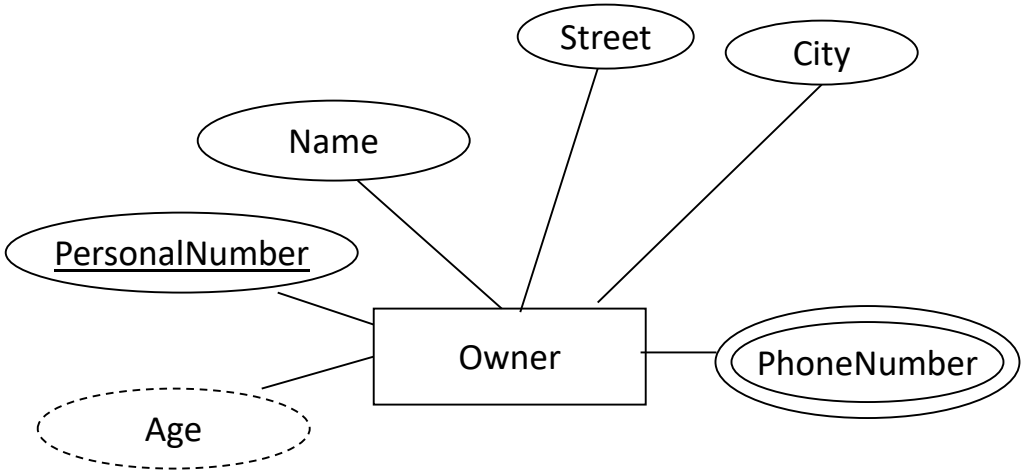
- Complex Attributes
  - Has multivalued & Composite components in it.
  - Multivalued attributes → represented within '{ }'
  - Composite attributes → represented within '( )'

  - Ex: {CollegeDegrees(College, Year, Degree, Field)}

- Null Value
  - Null is something which is not applicable or unknown

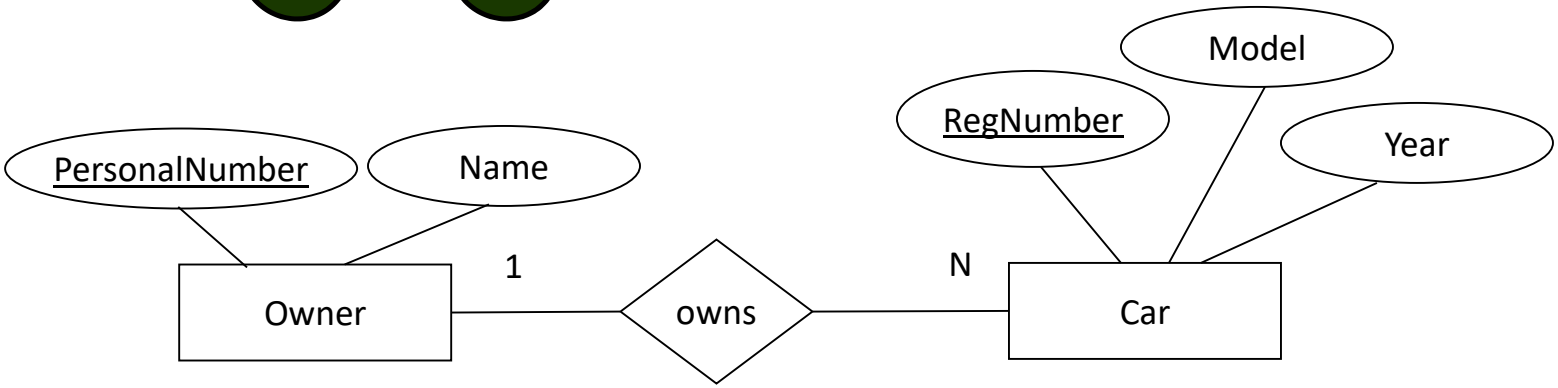
Constraints on Attributes

- Value sets (domains) of attributes.
- Key attributes.



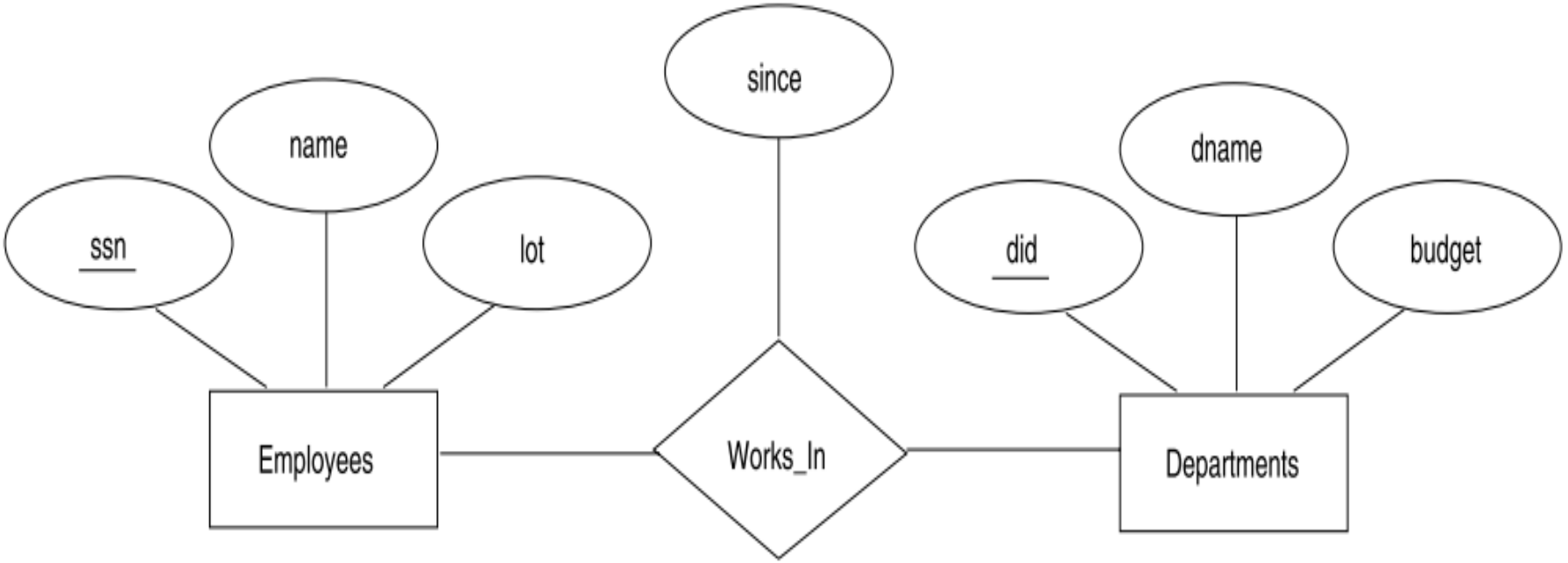
Relationship Type

- Relationship type: Association among entity types.



Conceptual Design 2

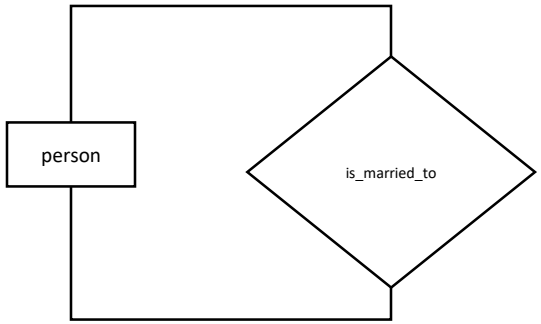
- Relationship: association among 2 or more entities
- Descriptive Attributes



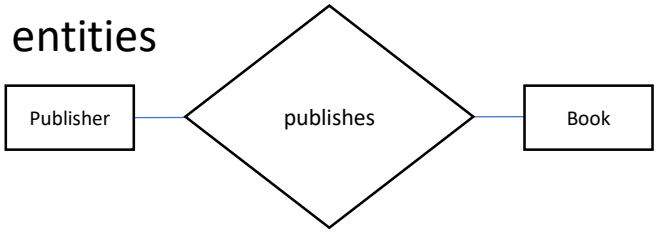
Degree of relationship

Degree of relationship: denote the number of entity types that participate in a relationship

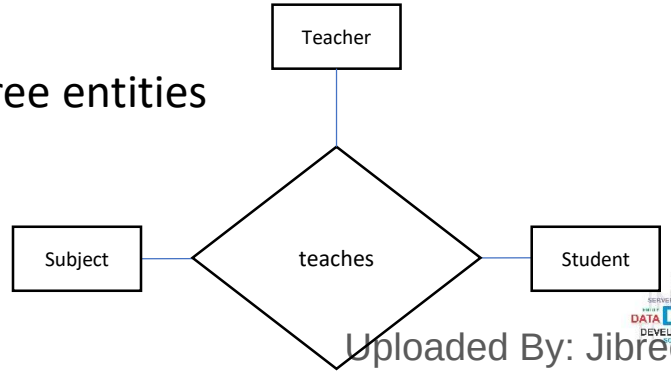
- Unary relationship
  - Exists when there is an association with only one entity



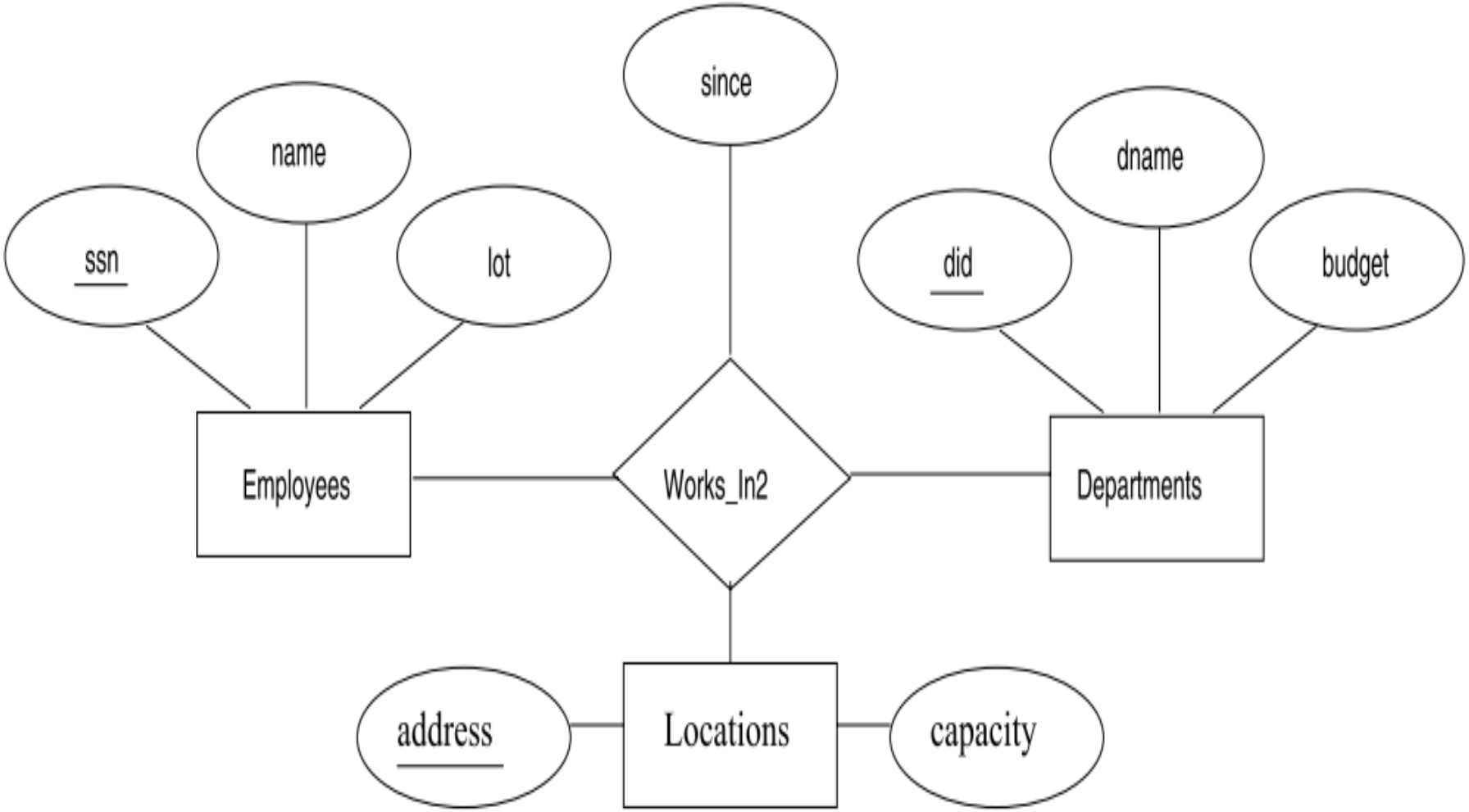
- Binary relationship
  - Exists when there is an association among two entities



- Ternary relationship
  - Exists when there is an association among three entities

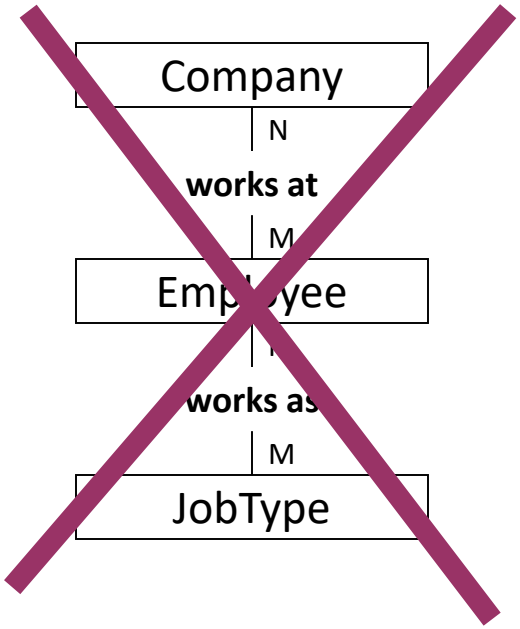


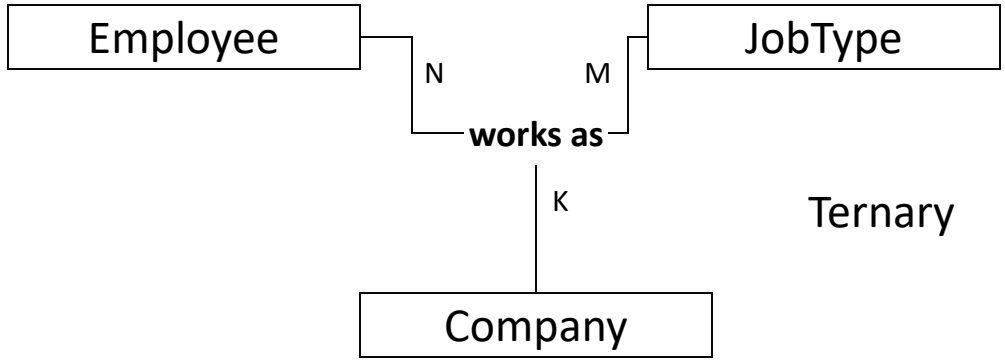
Ternary Relationship



N-ary relationships

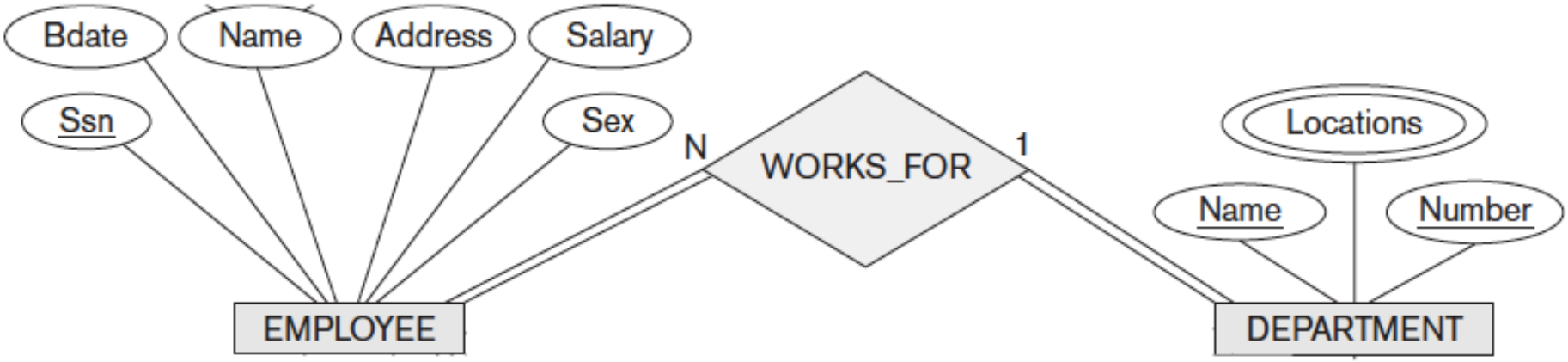
- Example. A person works as an engineer at one company and as a gym instructor at another company.



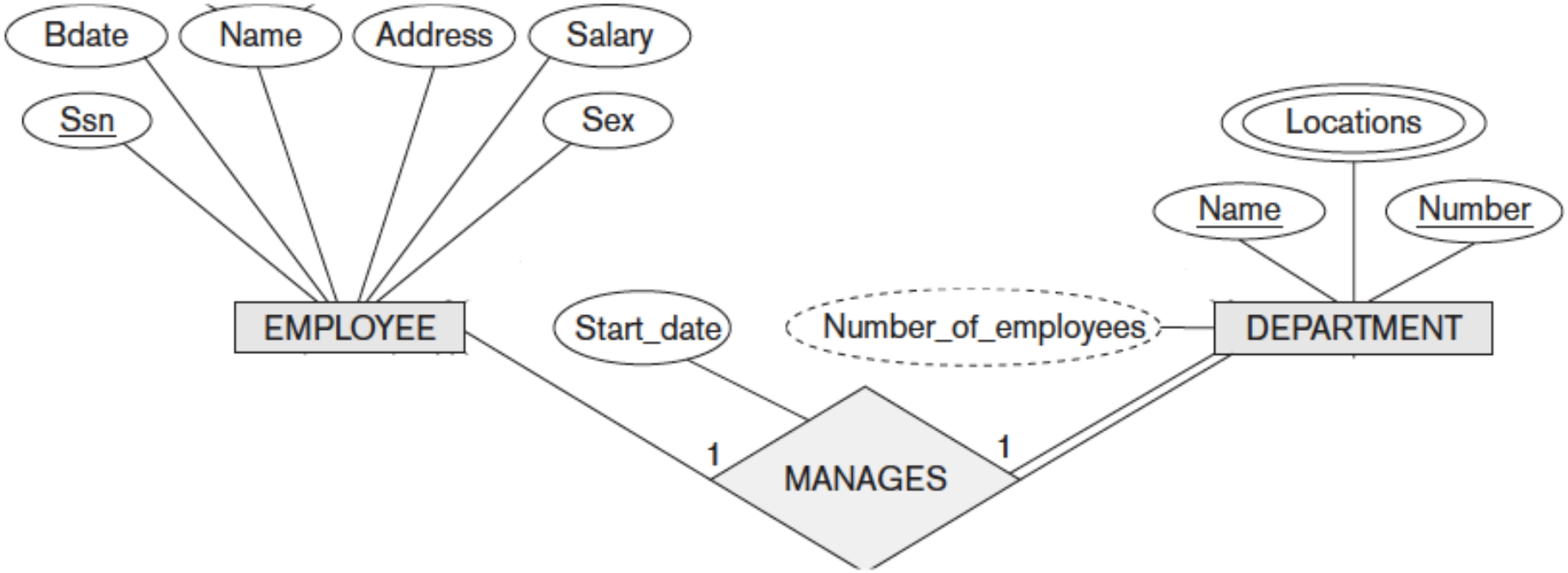


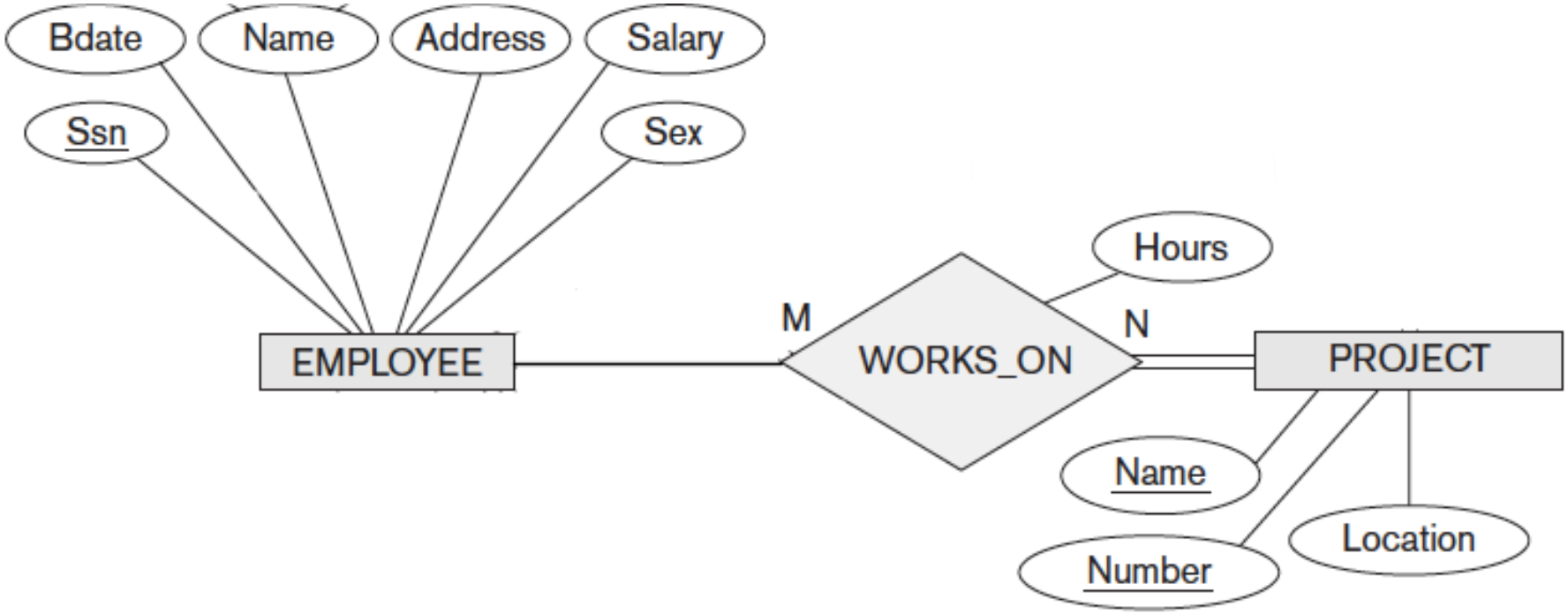
Ternary

Examples: Employee



Examples: Manage

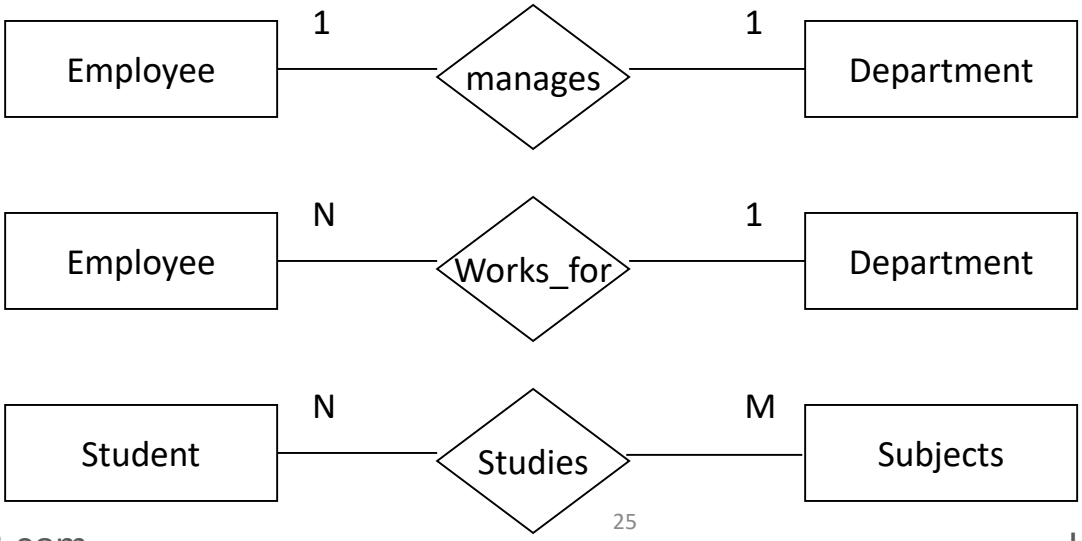




Constraints on relationship types

1- Cardinality ratio:

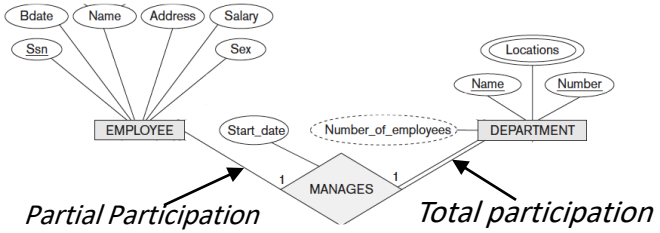
- **Maximum** number of relationship instances that an entity can participate in.
- Possible cardinality ratios for binary relationship: 1:1, 1: N, N:1, and N:M

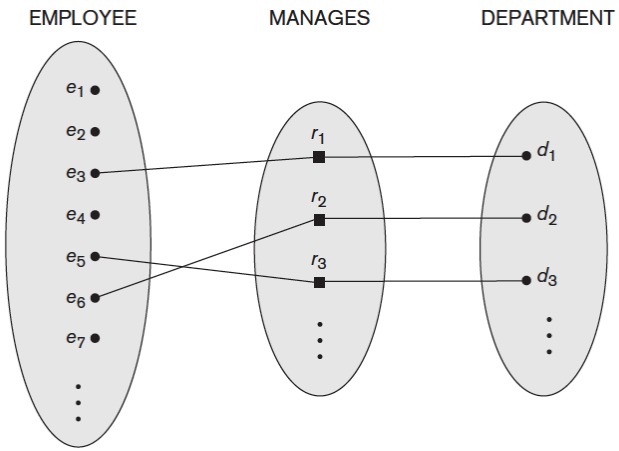


Constraints on relationship types

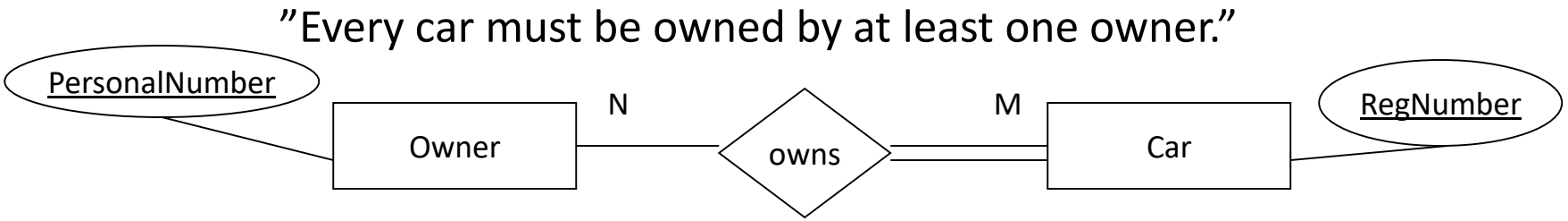
2- Paritıcpation Constraints:

- Specifies whether existence of an entity depends on its being related to another entitv
- 2 types:
  - TotalParticipation
  - Partial participation

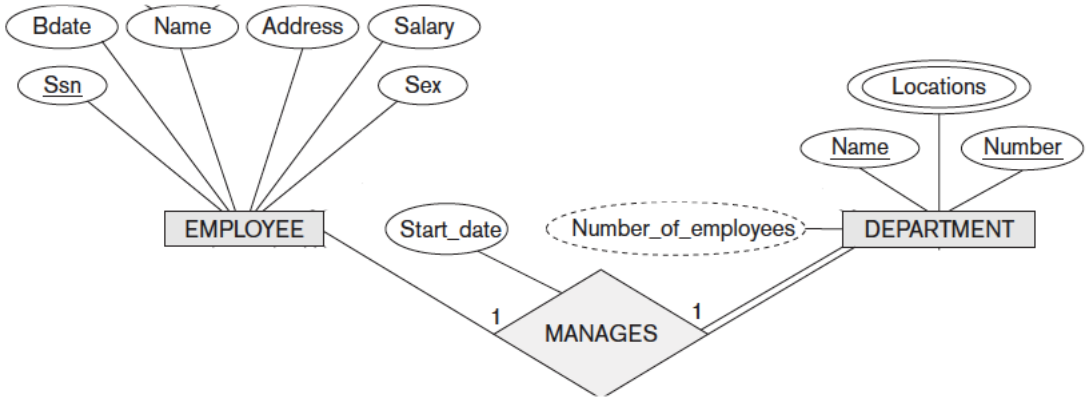


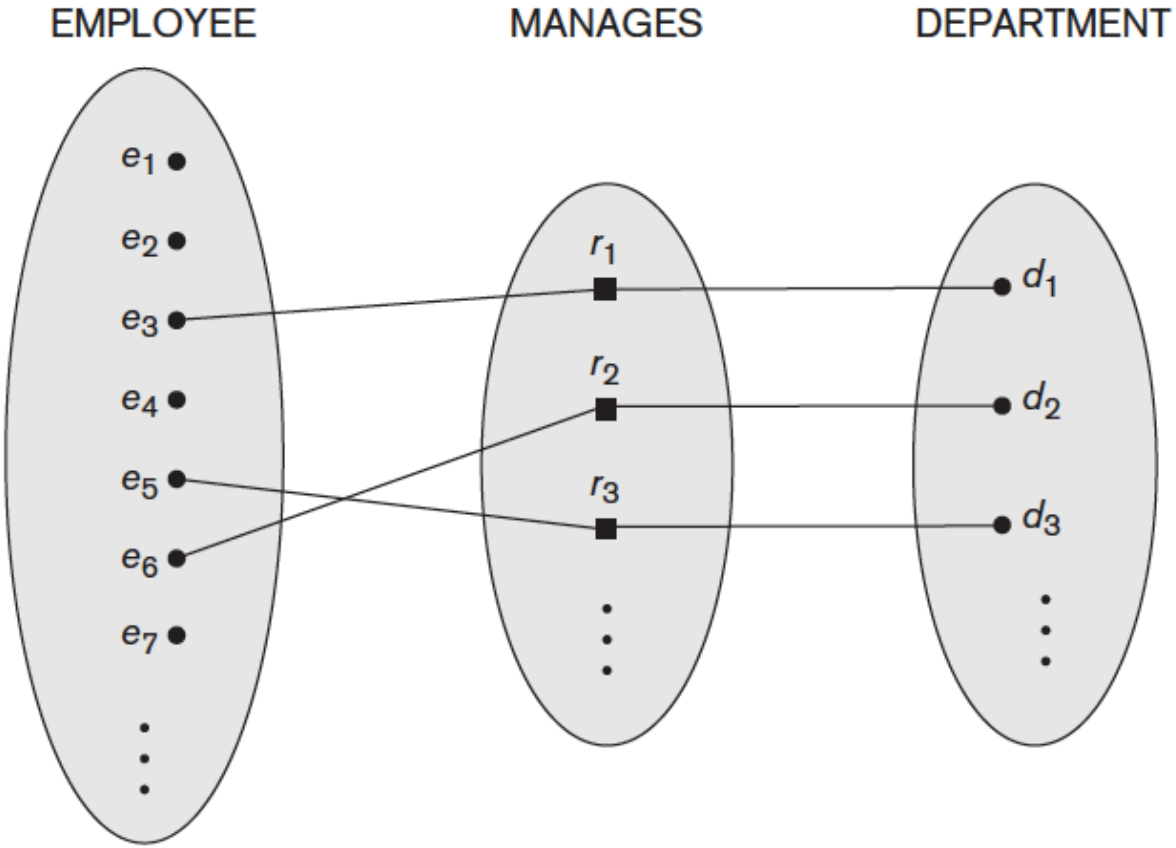


Example: Participant constraint.

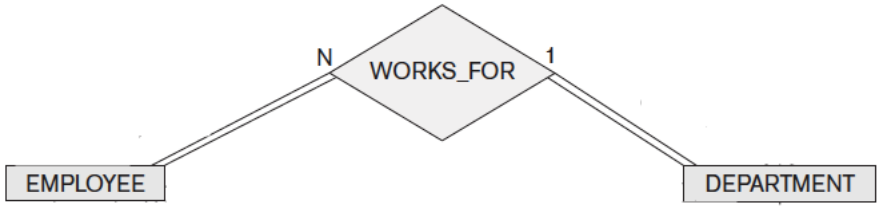


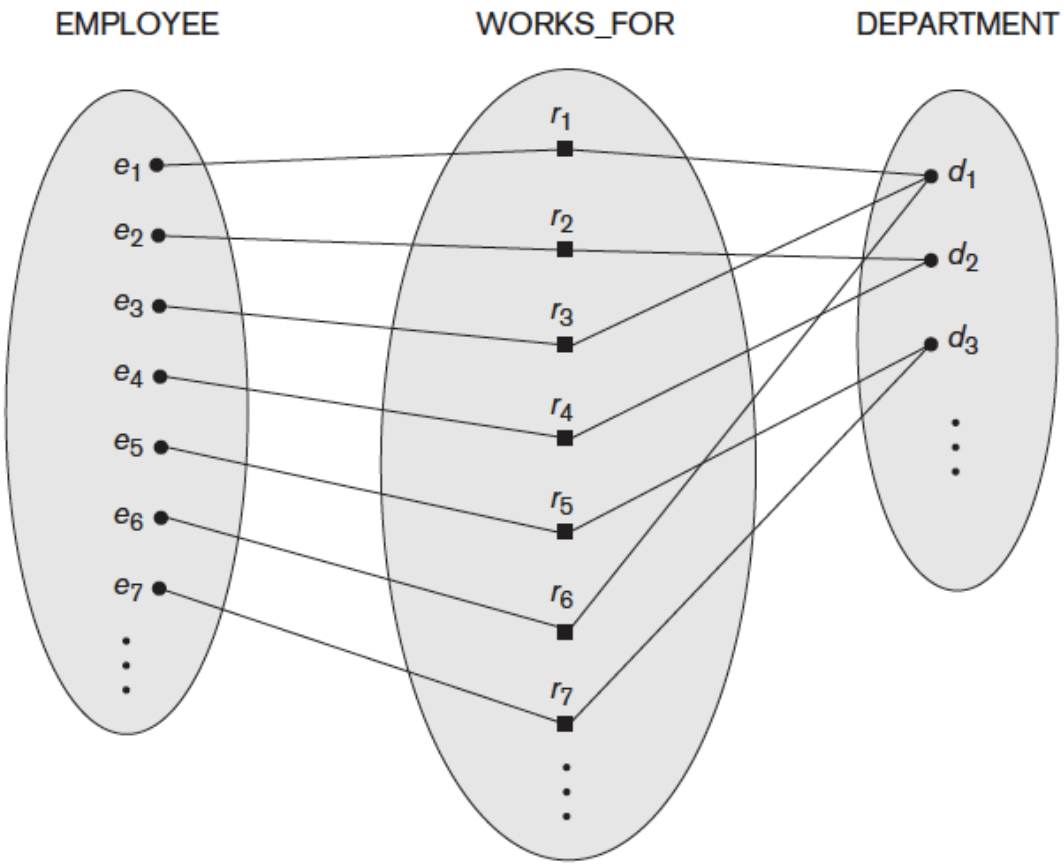
One-to-One



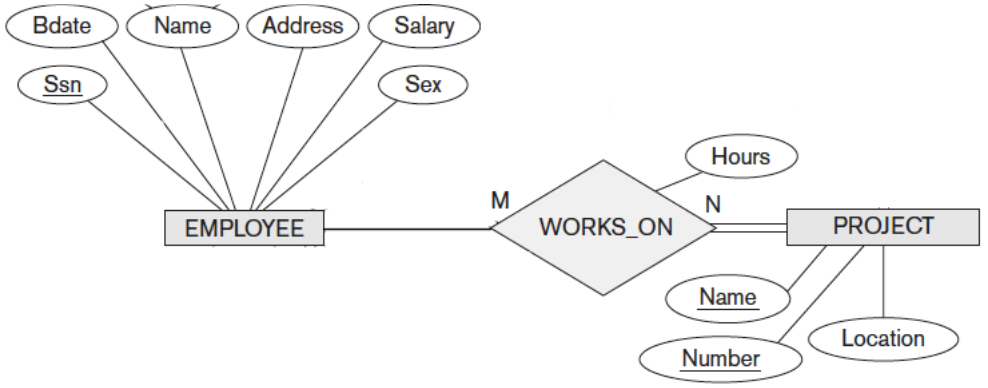


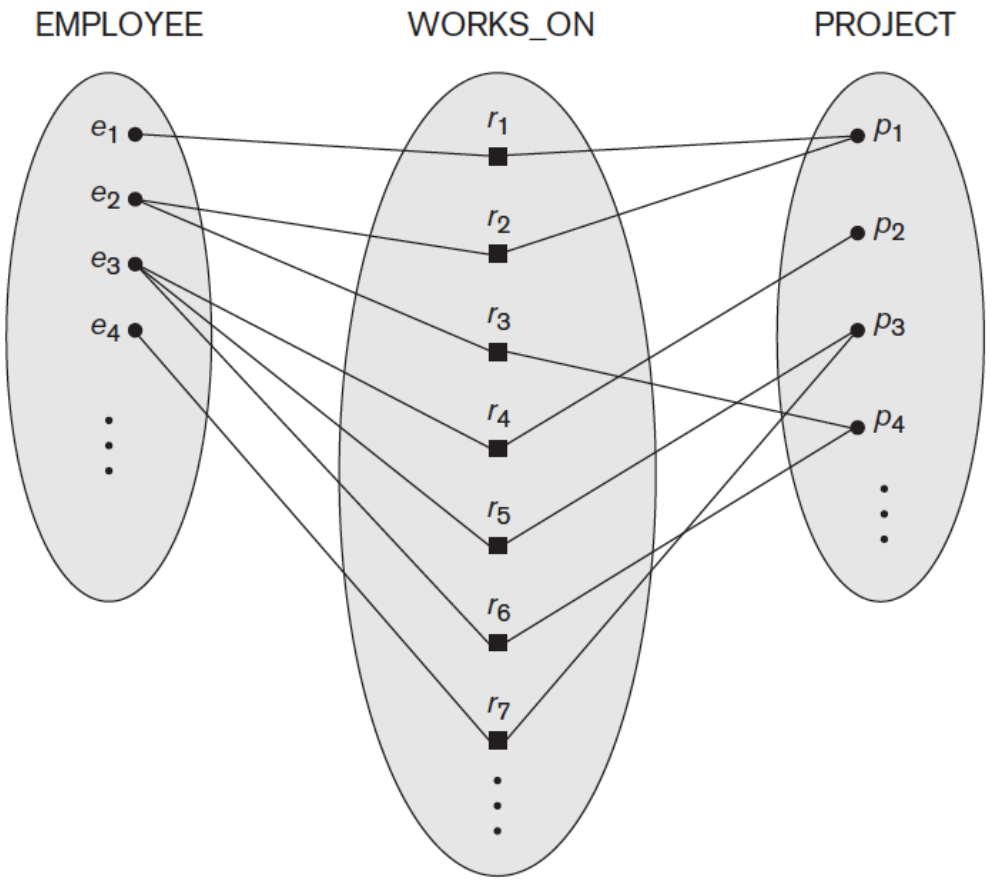
One-to-Many





Many-to-Many





Why we need keys ?

Employee ID	Name	SSN	Salary	Phone	Email
101	Ahmed	907683996	5000	599000000	a@gmail.com
102	Salem	945878065	6000	599000000	b@gmail.com
103	Alya	408876650	3500	599000000	c@gmail.com
104	Yusuf	416325448	6800	599000000	d@gmail.com
105	Hani	914875525	6200	599000000	e@gmail.com
106	Ruba	924665547	4500	599000000	f@gmail.com
107	Salem	443325898	2500	599000000	g@gmail.com
108	Ahmed	765454545	3500	599000000	h@gmail.com
109	khaled	224598711	2500	599000000	i@gmail.com

Let's give Ahmed a salary raise by 10%

BUT which Ahmed?

Super key
Candidate key
Primary key
Alternate key
Unique key
Composite key
Foreign key

Employee ID	Name	SSN	Salary	Phone	Email
101	Ahmed	907683996	5000	599000000	a@gmail.com
102	Salem	945878065	6000	599000000	b@gmail.com
103	Alya	408876650	3500	599000000	c@gmail.com
104	Yusuf	416325448	6800	599000000	d@gmail.com
105	Hani	914875525	6200	599000000	e@gmail.com
106	Ruba	924665547	4500	599000000	f@gmail.com
107	Salem	443325898	2500	599000000	g@gmail.com
108	Ahmed	765454545	3500	599000000	h@gmail.com
109	khaled	224598711	2500	599000000	i@gmail.com

Keys

- A **super key (superset)** of an entity set is a set of one or more attributes whose values uniquely determine each entity. (unique but can be null)
  - {Employee ID} {SSN} {Employee ID, Name} {Employee ID, SSN} {Employee ID, Salary} {Employee ID, Phone} {Employee ID, Email} {Name, SSN} {Name, Phone} {Name, Email} {SSN, Salary} {SSN, Phone} {SSN, Email} {Salary, Phone} {Salary, Email} {Phone, Email}

- A **candidate** key of an entity set is a minimal super key
  - {Employee ID} {SSN} {Name, Phone} {email}

  - Although several candidate keys may exist, one of the candidate keys is selected to be the **primary key.** (unique & cannot be NULL)
    - **Either {Employee ID}** or {SSN} → DBA has to decide which one is the primary key since this key never/very rare changed
  - The Alternate Key is the candidate key **other than** the primary key {SSN} {Name, Phone} {email}
  - Unique can is the alternate key that can be NULL {Name, Phone} {email}…
  - Composite key {Name, Phone},….
  - Foreign key (Between tables)

Keys for Relationship Sets

- The combination of primary keys of the participating entity sets forms a super key of a relationship set.
  - (s_id, i_id) is the super key of advisor
  - 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 multiple meeting dates between a student and her advisor, we cannot assume a relationship for each meeting. We can use a multivalued attribute though
- Must consider the mapping cardinality of the relationship set when deciding 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

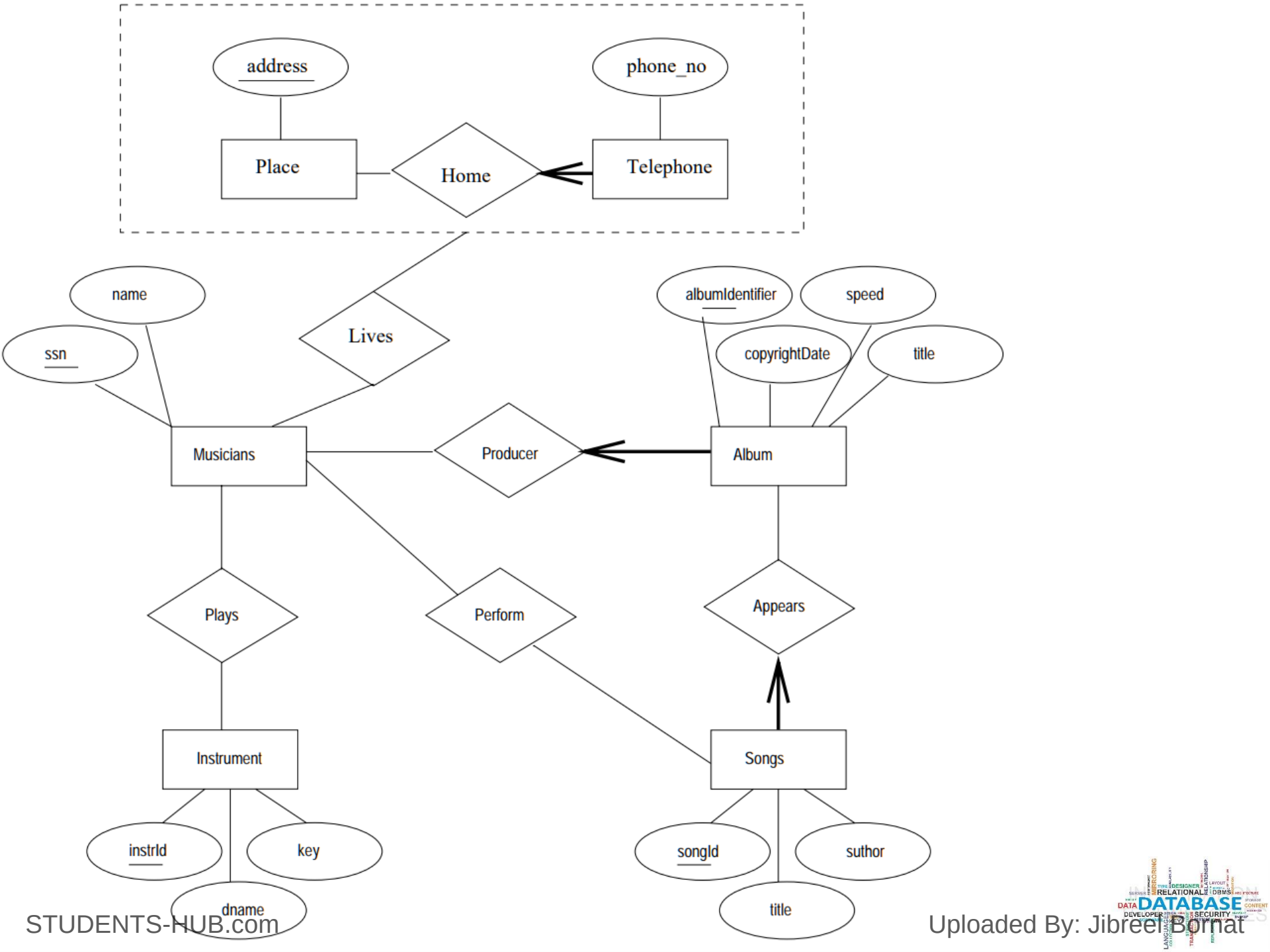
Musicians Example

Exercise 2.5 Notown Records has decided to store information about musicians who perform on its albums (as well as other company data) in a database. The company has wisely chosen to hire you as a database designer (at your usual consulting fee of $2500/day).

- Each musician that records at Notown has an SSN, a name, an address, and a phone number. Poorly paid musicians often share the same address, and no address has more than one phone.
- Each instrument used in songs recorded at Notown has a unique identification number, a name (e.g., guitar, synthesizer, flute) and a musical key (e.g., C, B-flat, E-flat).
- Each album recorded on the Notown label has a unique identification number, a title, a copyright date, a format (e.g., CD or MC), and an album identifier.
- Each song recorded at Notown has a title and an author.
- Each musician may play several instruments, and a given instrument may be played by several musicians.
- Each album has a number of songs on it, but no song may appear on more than one album.
- Each song is performed by one or more musicians, and a musician may perform a number of songs.
- Each album has exactly one musician who acts as its producer. A musician may produce several albums, of course.

- Musicians: Musicians have a unique SSN, a name, an address, and a phone number.
- Instruments: Instruments have a unique identifier, a name, and a musical key.
- Songs: Songs have a title and an author.
- Albums: Albums have a title, a copyright date, a format, and an album identifier.
- Plays: A musician plays an instrument on a song.
- Appears: A song appears on an album.
- Produces: A musician produces an album.

- ✓ Musician-Instrument: Many-to-many
- ✓ Musician-Song: Many-to-many
- ✓ Song-Album: One-to-many
- ✓ Musician-Album: One-to-many (via the Produces relationship)



University Example

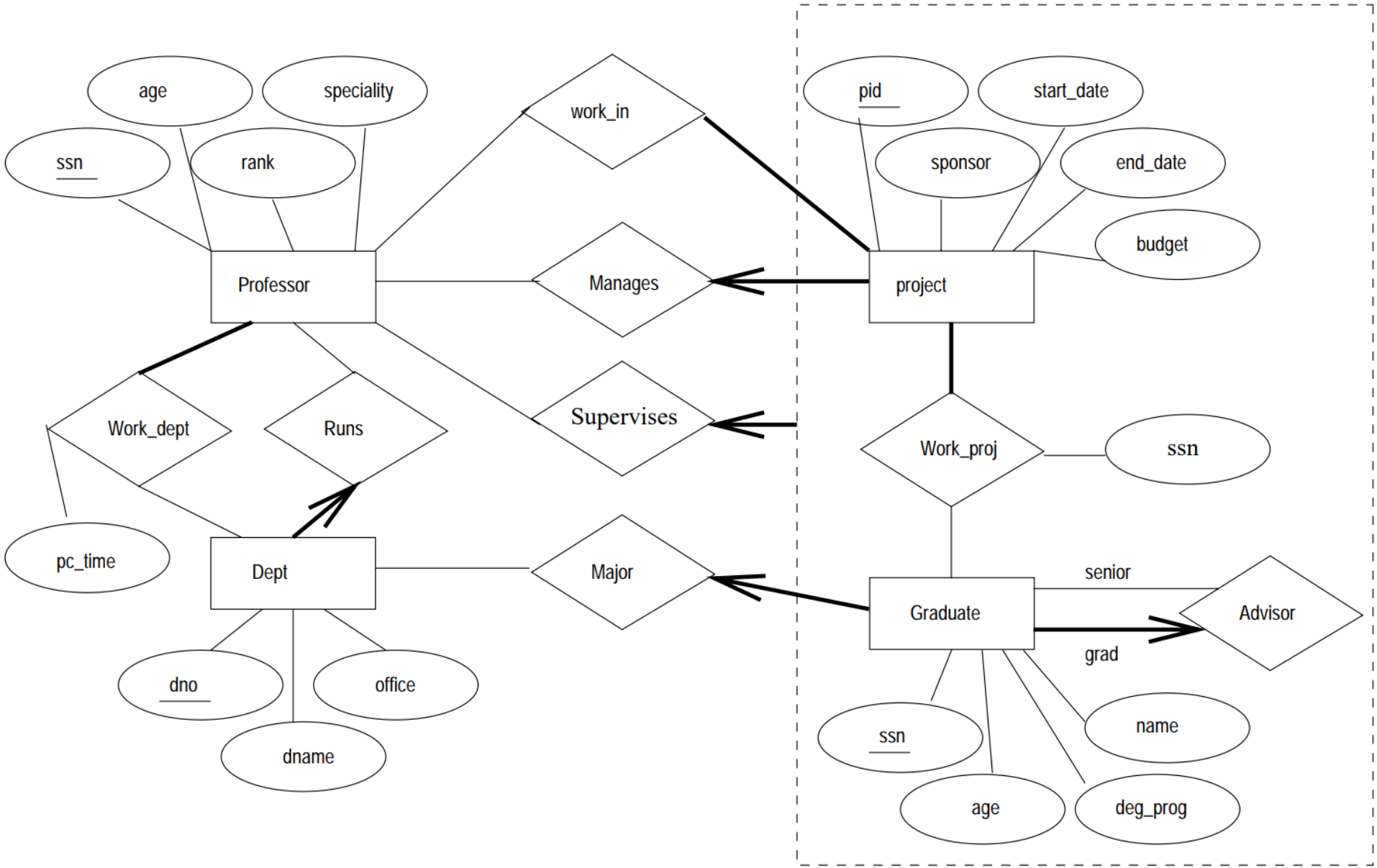
Exercise 2.3 Consider the following information about a university database:

- Professors have an SSN, a name, an age, a rank, and a research specialty.
- Projects have a project number, a sponsor name (e.g., NSF), a starting date, an ending date, and a budget.
- Graduate students have an SSN, a name, an age, and a degree program (e.g., M.S. or Ph.D.).
- Each project is managed by one professor (known as the project's principal investigator).
- Each project is worked on by one or more professors (known as the project's co-investigators).
- Professors can manage and/or work on multiple projects.
- Each project is worked on by one or more graduate students (known as the project's research assistants).
- When graduate students work on a project, a professor must supervise their work on the project. Graduate students can work on multiple projects, in which case they will have a (potentially different) supervisor for each one.
- Departments have a department number, a department name, and a main office.
- Departments have a professor (known as the chairman) who runs the department.
- Professors work in one or more departments, and for each department that they work in, a time percentage is associated with their job.
- Graduate students have one major department in which they are working on their degree.
- Each graduate student has another, more senior graduate student (known as a student advisor) who advises him or her on what courses to take.

- Professors: Professors have an SSN, a name, an age, a rank, and a research specialty.
- Projects: Projects have a project number, a sponsor name, a starting date, an ending date, and a budget.
- Graduate students: Graduate students have an SSN, a name, an age, and a degree program.

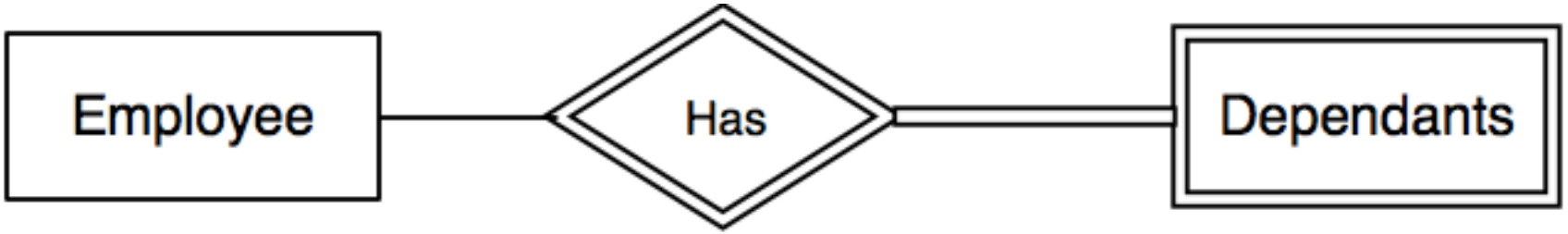
cardinalities for the relationships:

- Professor-Project: Many-to-many
- Professor-Graduate student: Many-to-many
- Graduate student-Project: Many-to-many



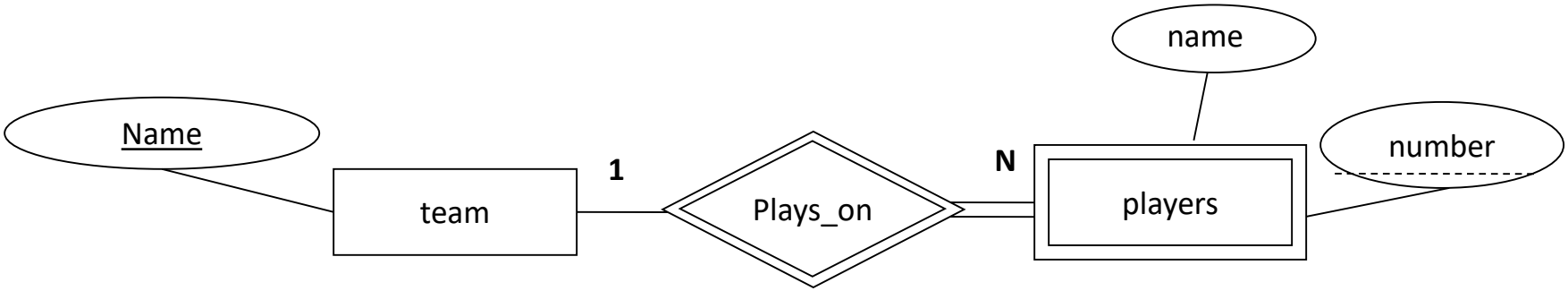
Weak Entity Types

- Do not have key attributes of their own
  - Identified by being related to specific entities from another entity type
- **Identifying relationship**
  - Relates a weak entity type to its owner
- Always has a total participation constraint

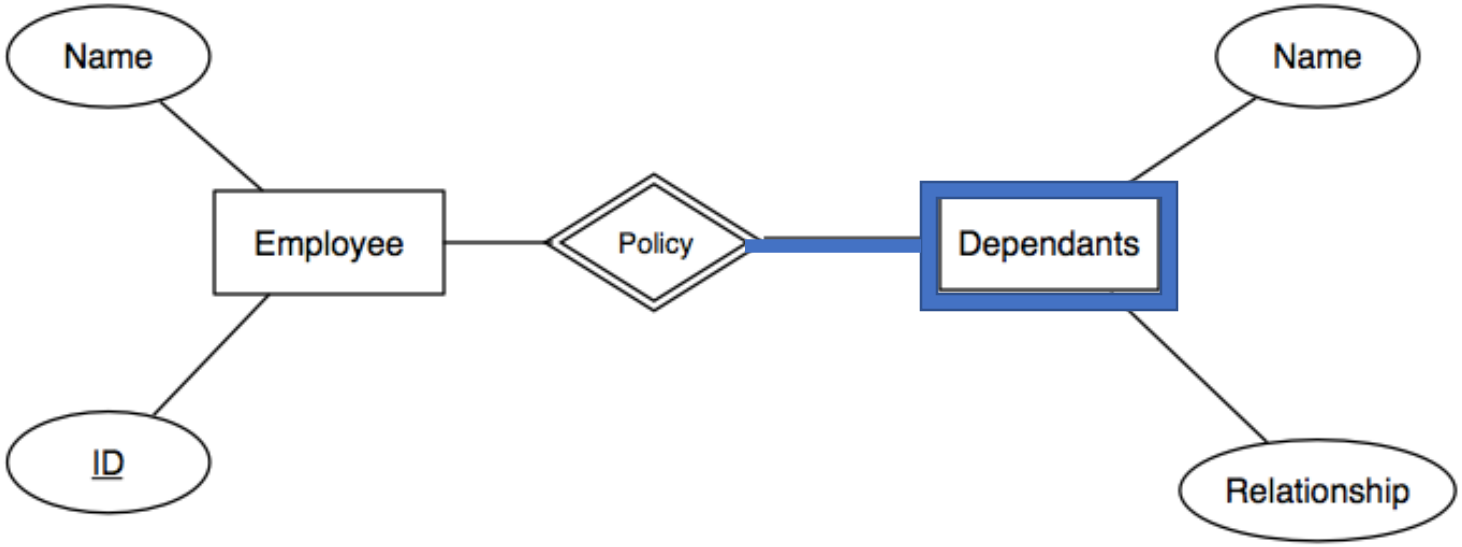


Constraints on Relationship Types

- Weak entity types: They do not have key attibutes of their own.
- A weak entity can be identified uniquely by being related to another entity (together with its own attributes).



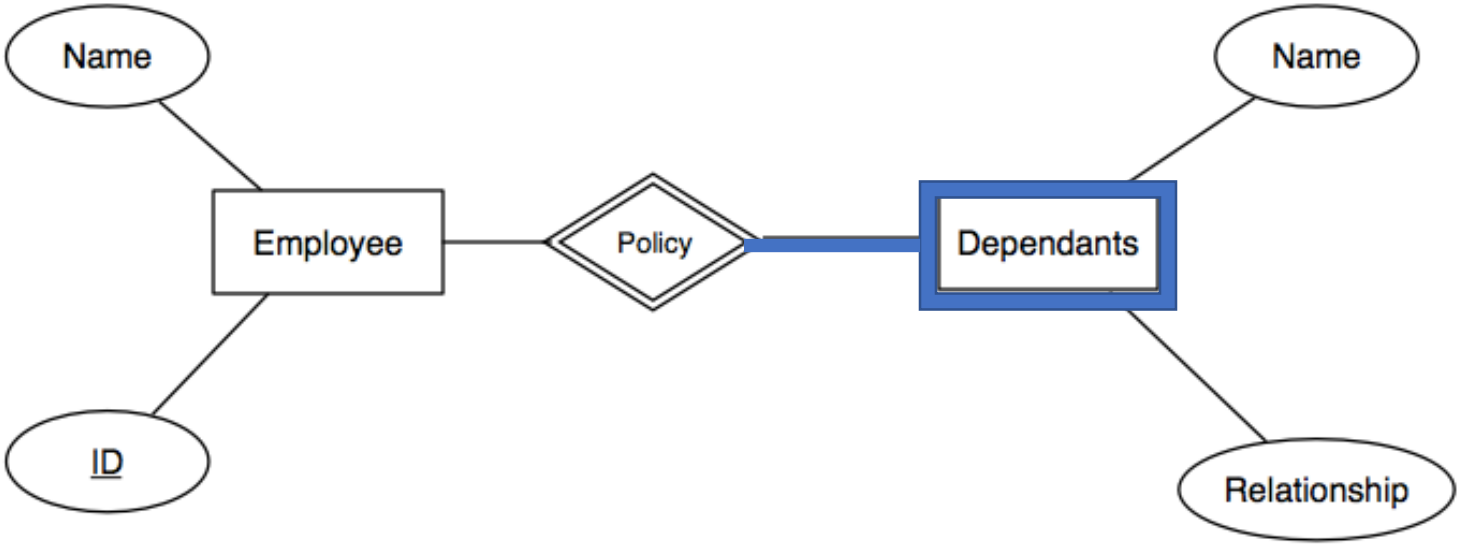
Weak Entity Sets



- If an entity set is weak, it will be shown as a rectangle with a double/thick border.
- Its supporting many-one relationships will be shown as diamonds with a double border.
- If an entity set supplies any attributes for its own key, then those attributes will be underlined.

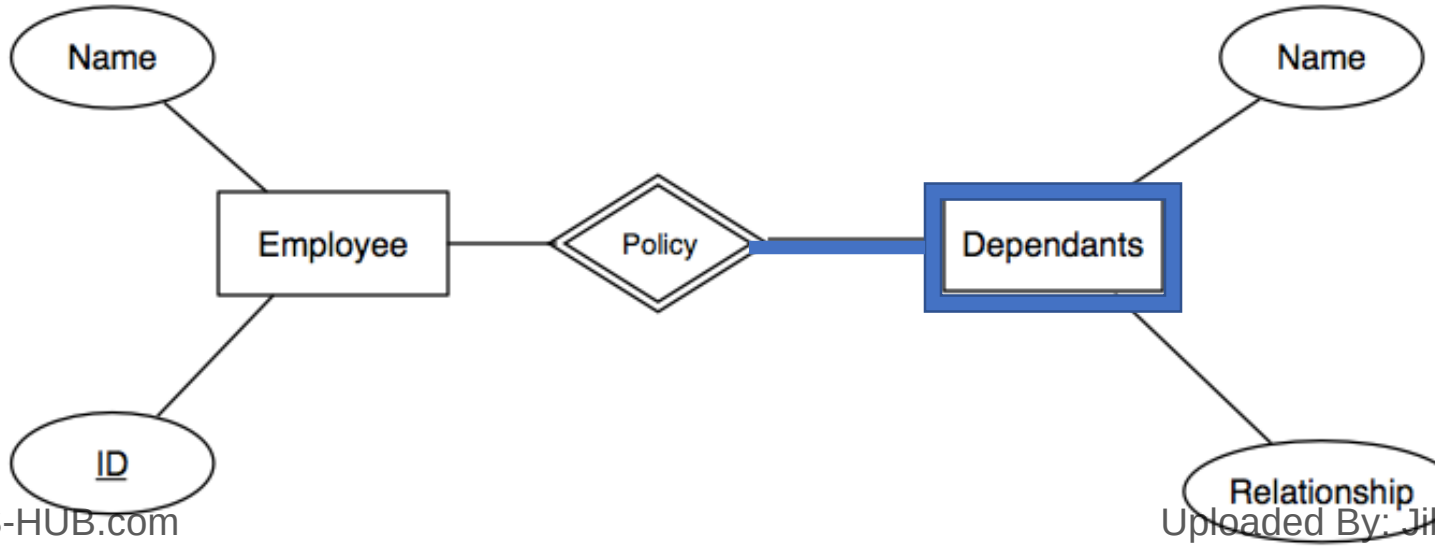
Requirements for Weak Entity Sets

- if E is a weak entity set, then its key consists of:
  - Zero or more of its own attributes, and
  - Key attributes from entity sets that are reached by certain many-one relationships from *E* to other entity sets. These many-one relationships are called supporting relationships for *E*.



Requirements for Weak Entity Sets

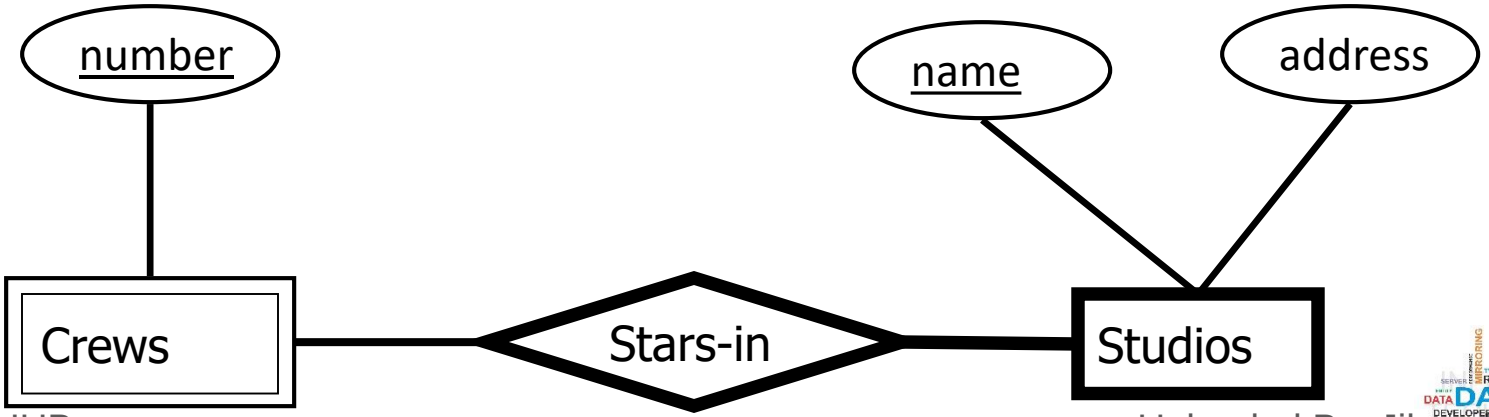
- In order for *R*, a many-one relationship from *E* to some entity set *F*, to be a supporting relationship for *E*, the following conditions must be obeyed:
  - *R* must be a binary, many-one relationship from *E* to *F*.
  - *R* must have referential integrity from *E* to *F*.
  - The attributes that *F* supplies for the key of *E* must be key attributes of *F*.
- Multiple supporting relationships are possible



Weak Entity Sets Notation

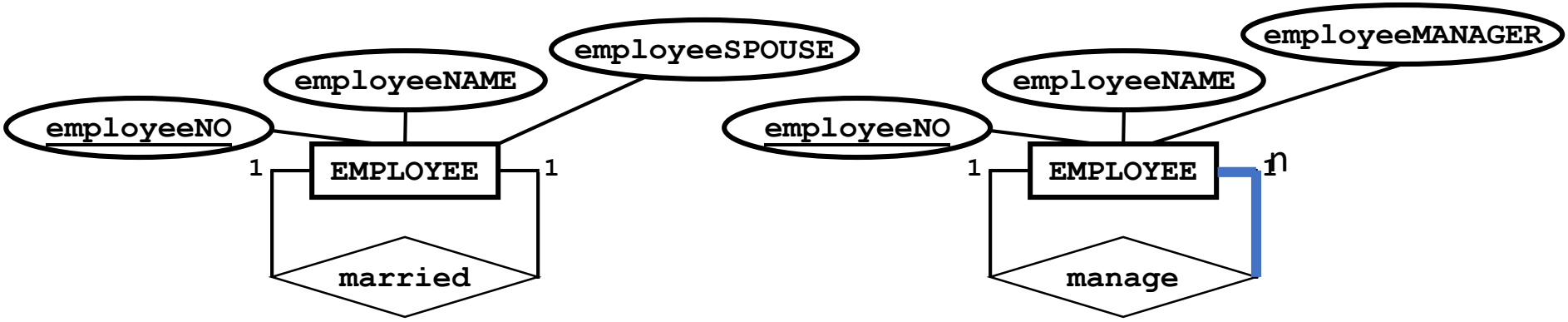
1. If an entity set is weak, it will be shown as a rectangle with a double border
2. Its supporting many-one relationship will be shown as diamonds with a double border
3. If an entity set supplies any attributes for its own key, then those attributes will be underlined

- Whenever we use an entity set *E with a double border,* it is weak. The key for *E is whatever attributes of E* are underlined plus the key attributes of those entity sets to which *E is connected by many-one* relationships with a double border.



Recursive Entity/Relationship

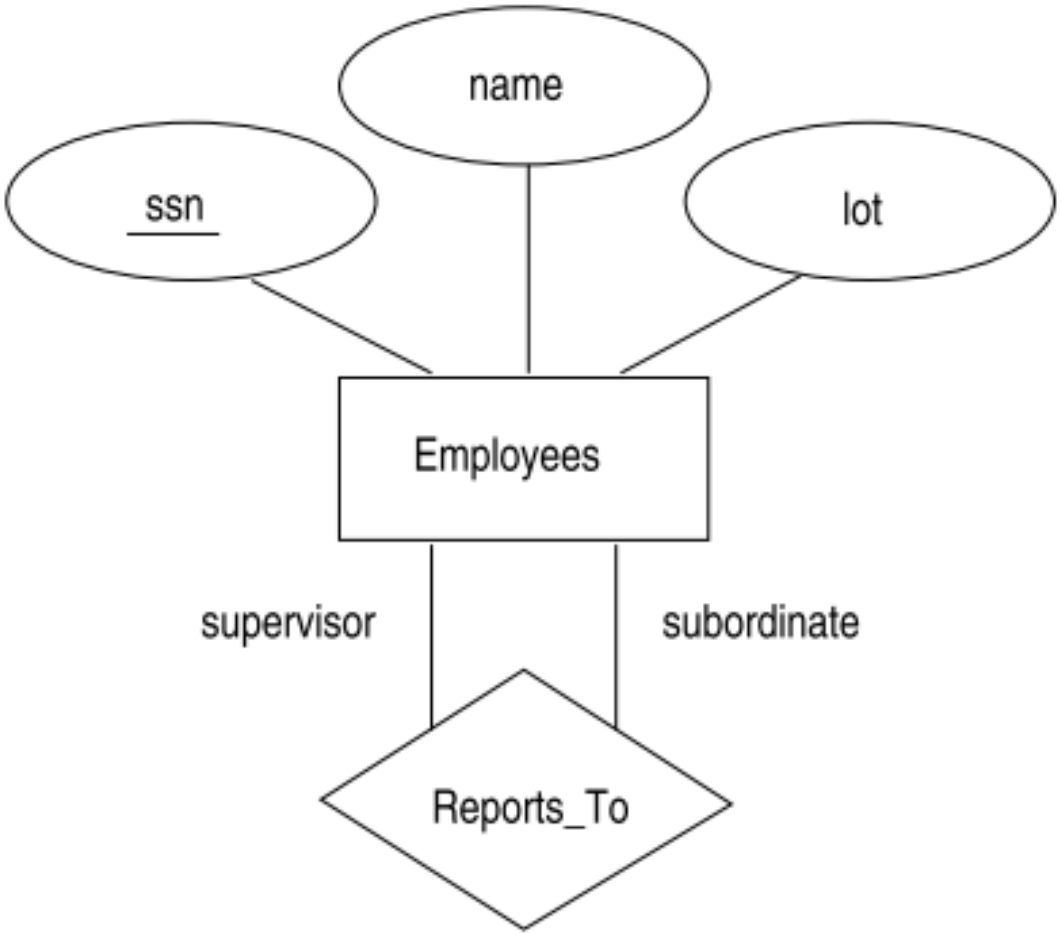
- Entity set that have relationship with the same entity set
- Example: **EMPLOYEE** entity



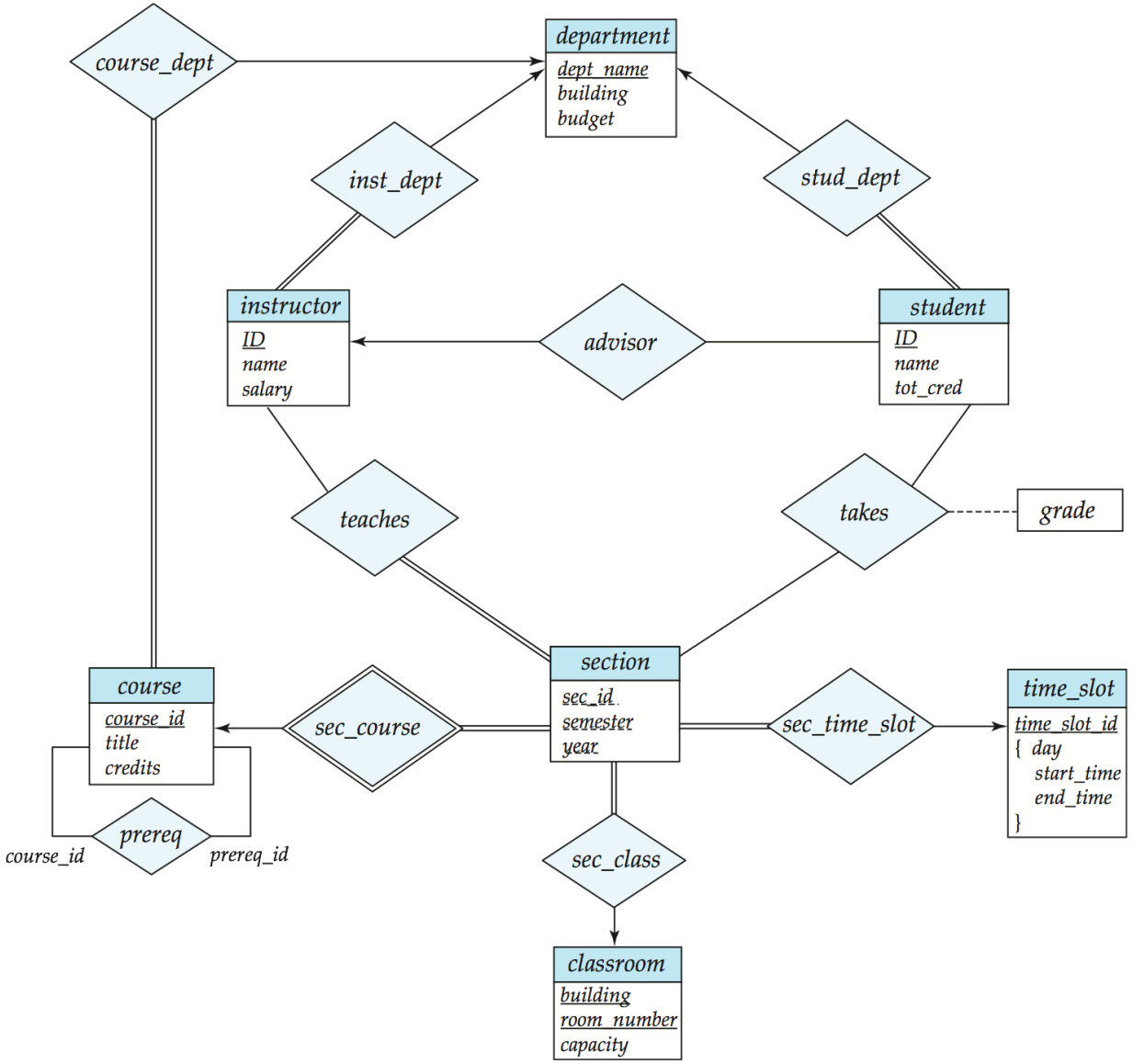
employeeNO	employeeNAME	employeeSPOUSE
111	Ali	444
222	Ah Chong	
333	Bazil	
444	Sheriz	111

employeeNO	employeeNAME	employeeMANAGER
111	Ali	333
222	Bassem	333
333	Sobhi	444
444	Allan	

Recursive Entity/Relationship



E-R Diagram for a University Enterprise



Enhanced ER (EER) model

- Why more? To comply with more complex data requirements.
  - Example. Only some employees can use a company car, only managers have to write a monthly report, but all employees have assigned personal number, salary account and a place in the office.
- Class Hierarchies
  - Subclass/superclass,
  - specialization/generalization,
  - union/category, and
  - attribute and relationship inheritance.

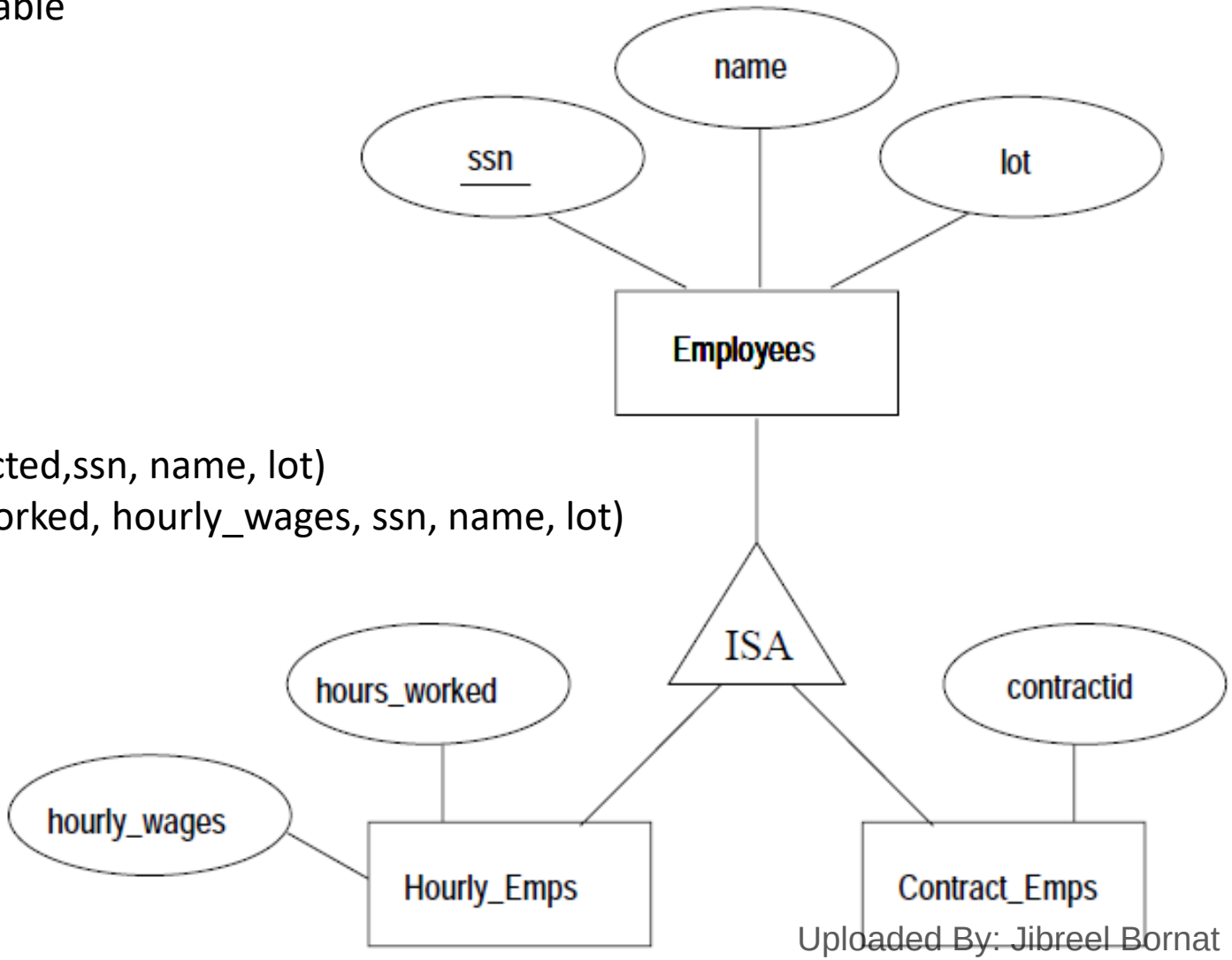
Class Hierarchies

Lets create a scheme/table
→ Method 1

look at leaf entities

What are leaf entities?
Contract_Emps
Hourly_Emps

Contract_emps(contracted,ssn, name, lot)
Hourly_Emps(hours_worked, hourly_wages, ssn, name, lot)

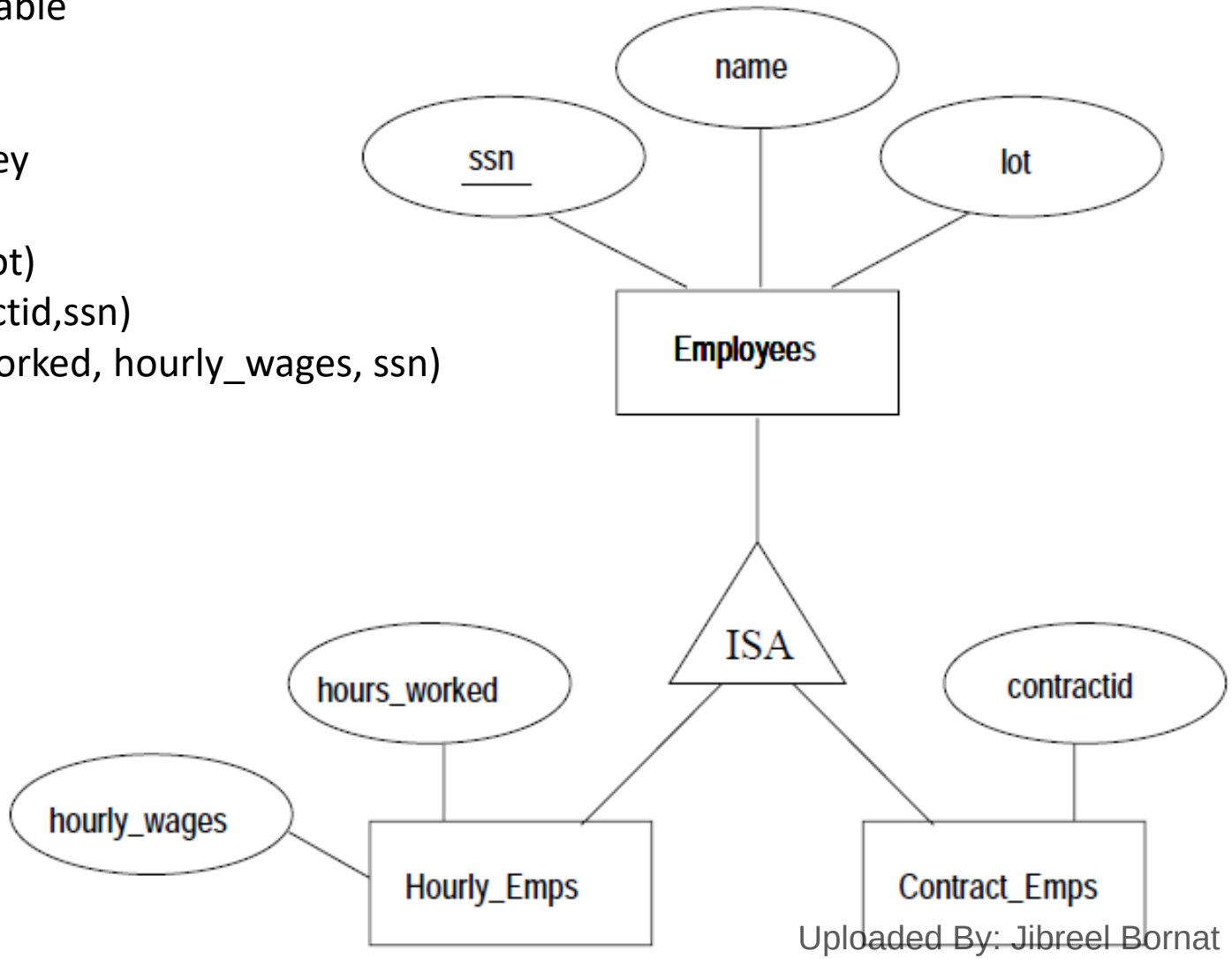


Class Hierarchies

Lets create a scheme/table
→ Method 2

Share the Superclass Key

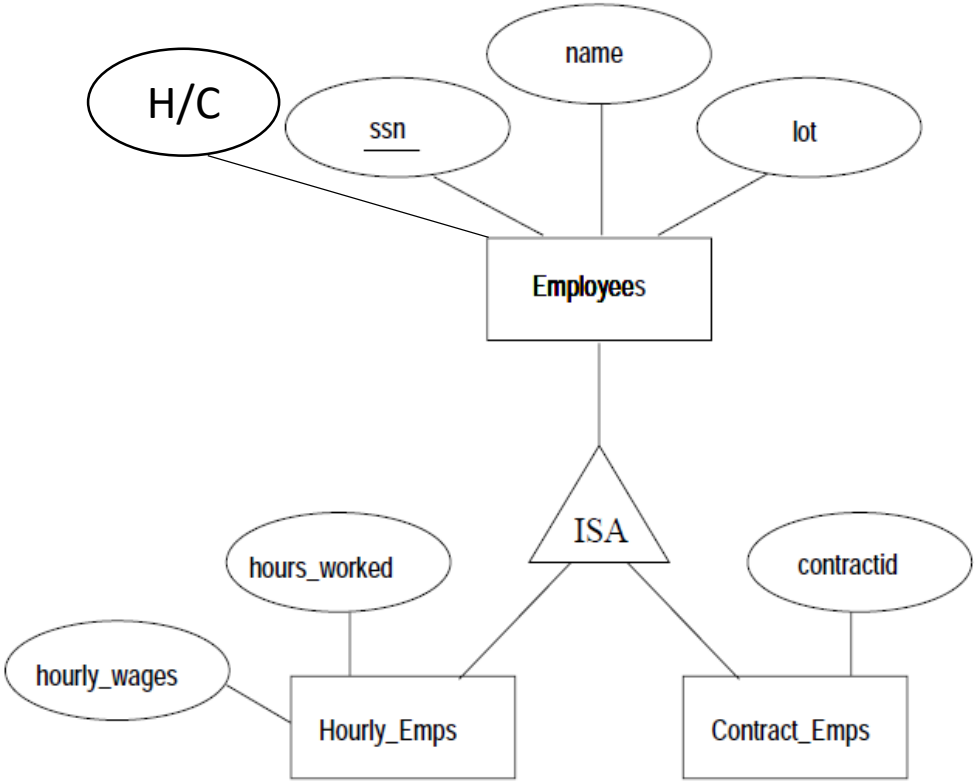
Employees(ssn,name,lot)
Contract_emps(contractid,ssn)
Hourly_Emps(hours_worked, hourly_wages, ssn)



Class Hierarchies

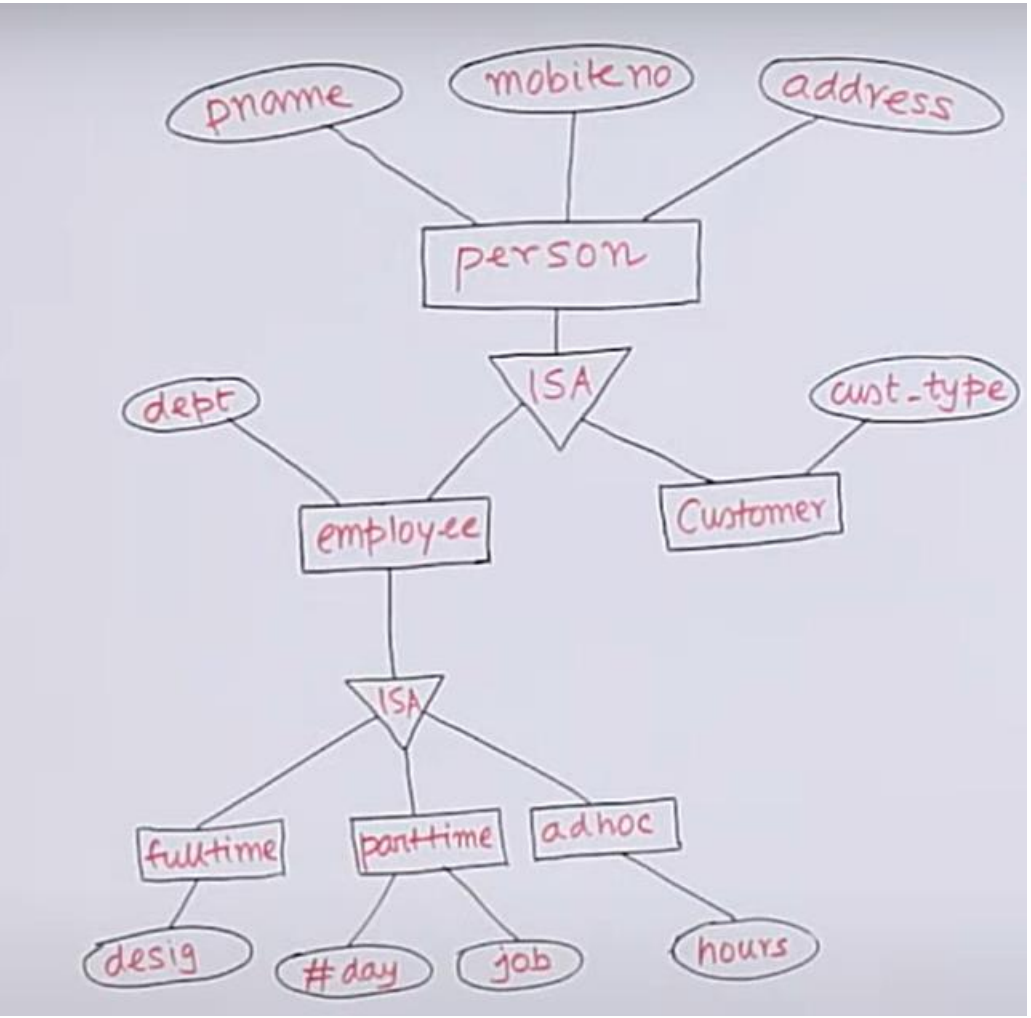
Lets create a scheme/table
→ Method 3

Share the Superclass Key



ssn	name	lot	Hourly_ wages	Hours_ worked	contract id	H/C

Make a schema (table)



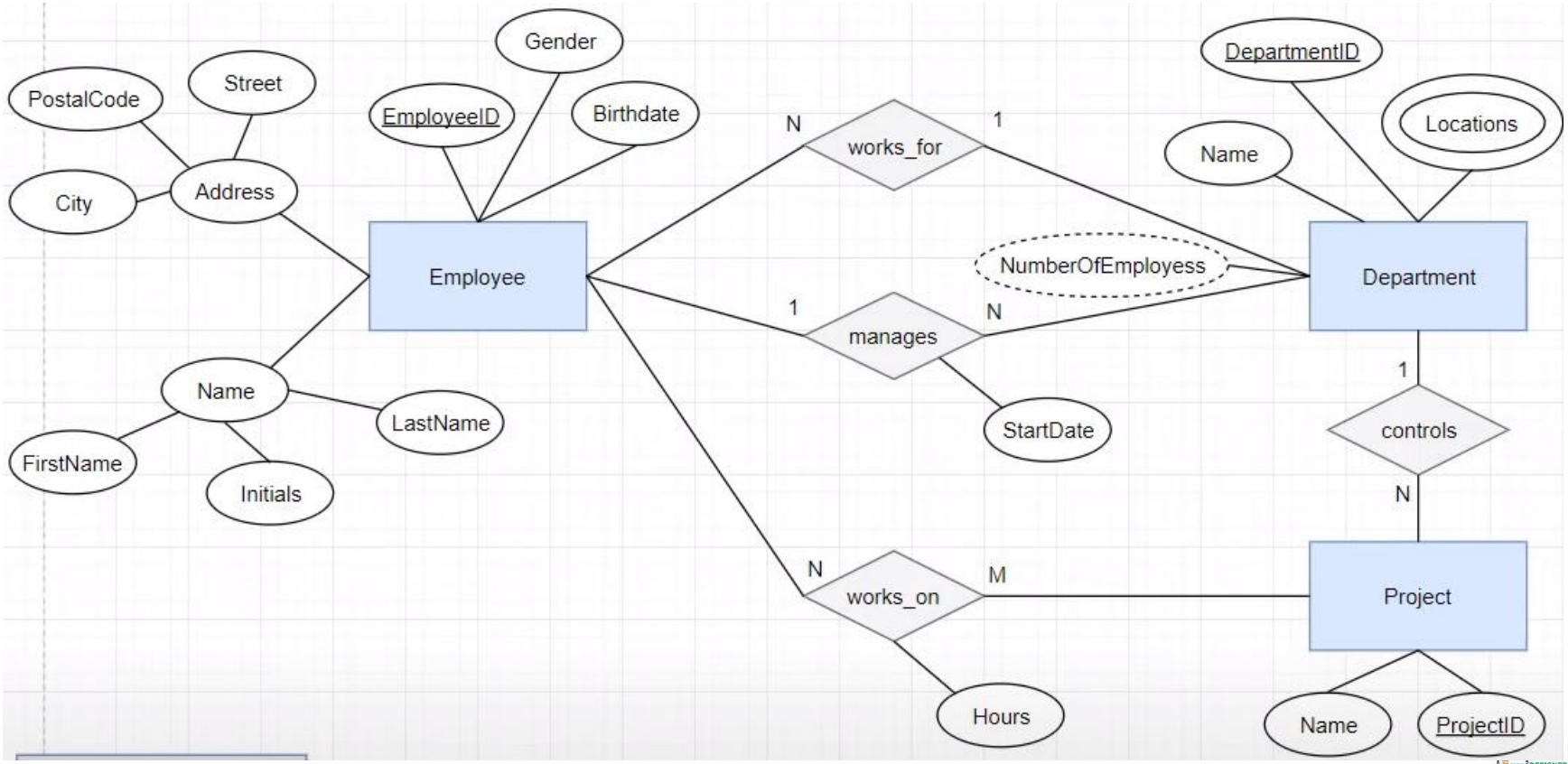
Customer(pname, mobile no, address, cust_type)

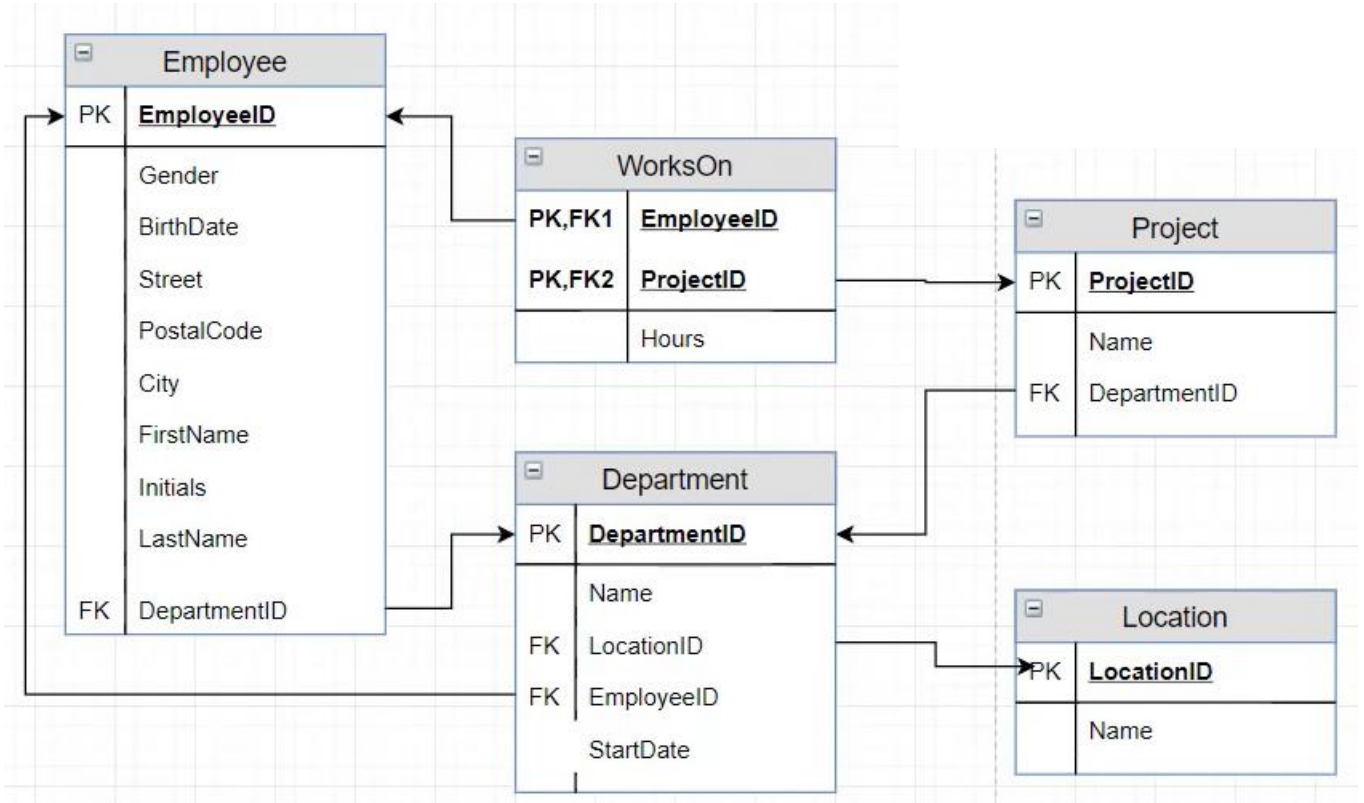
Fulltime(pname, mobile no, address, dept, desig)

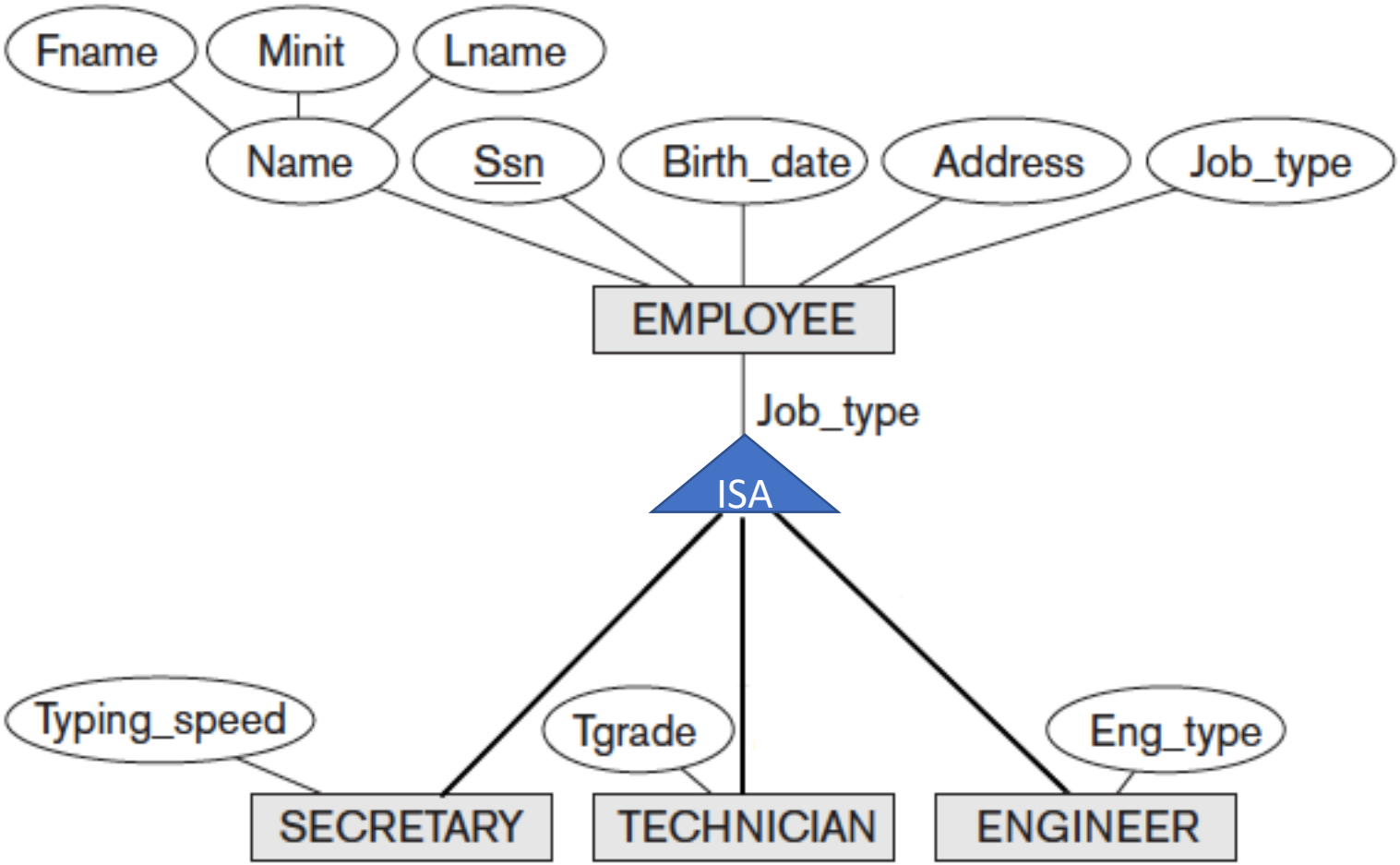
Partime(pname, mobile no, address, dept, #day, job)

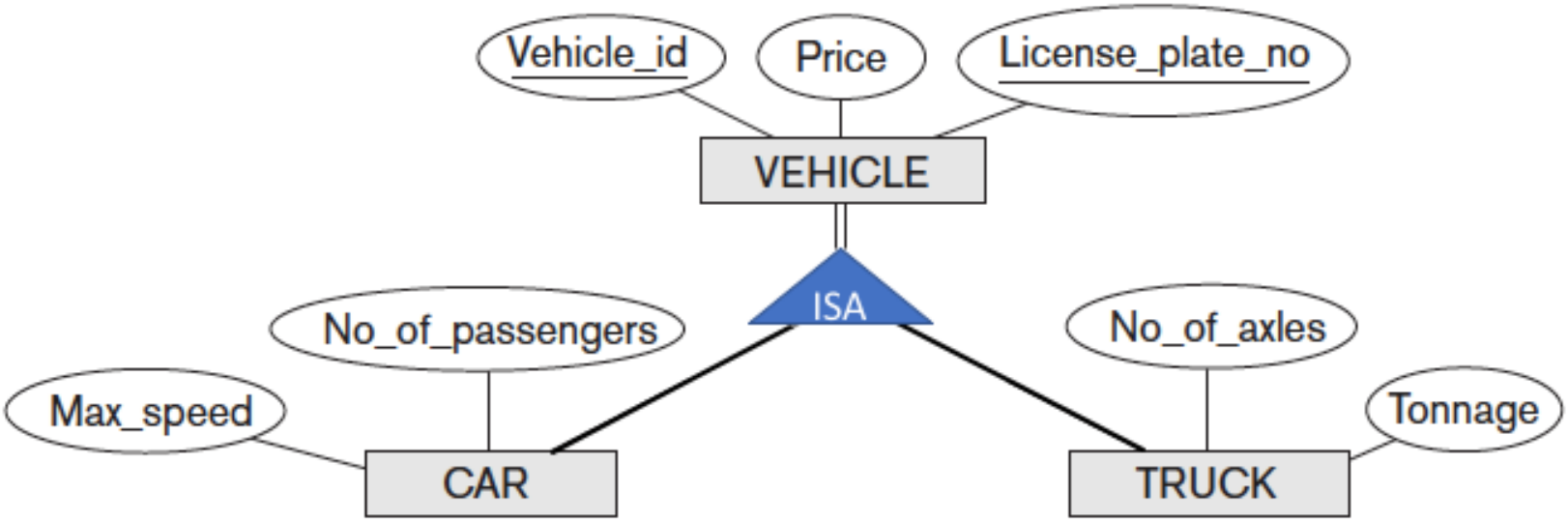
Adhoc(pname, mobile no, address, dept, hours)

Later→ from ER-diagram to Relational Model

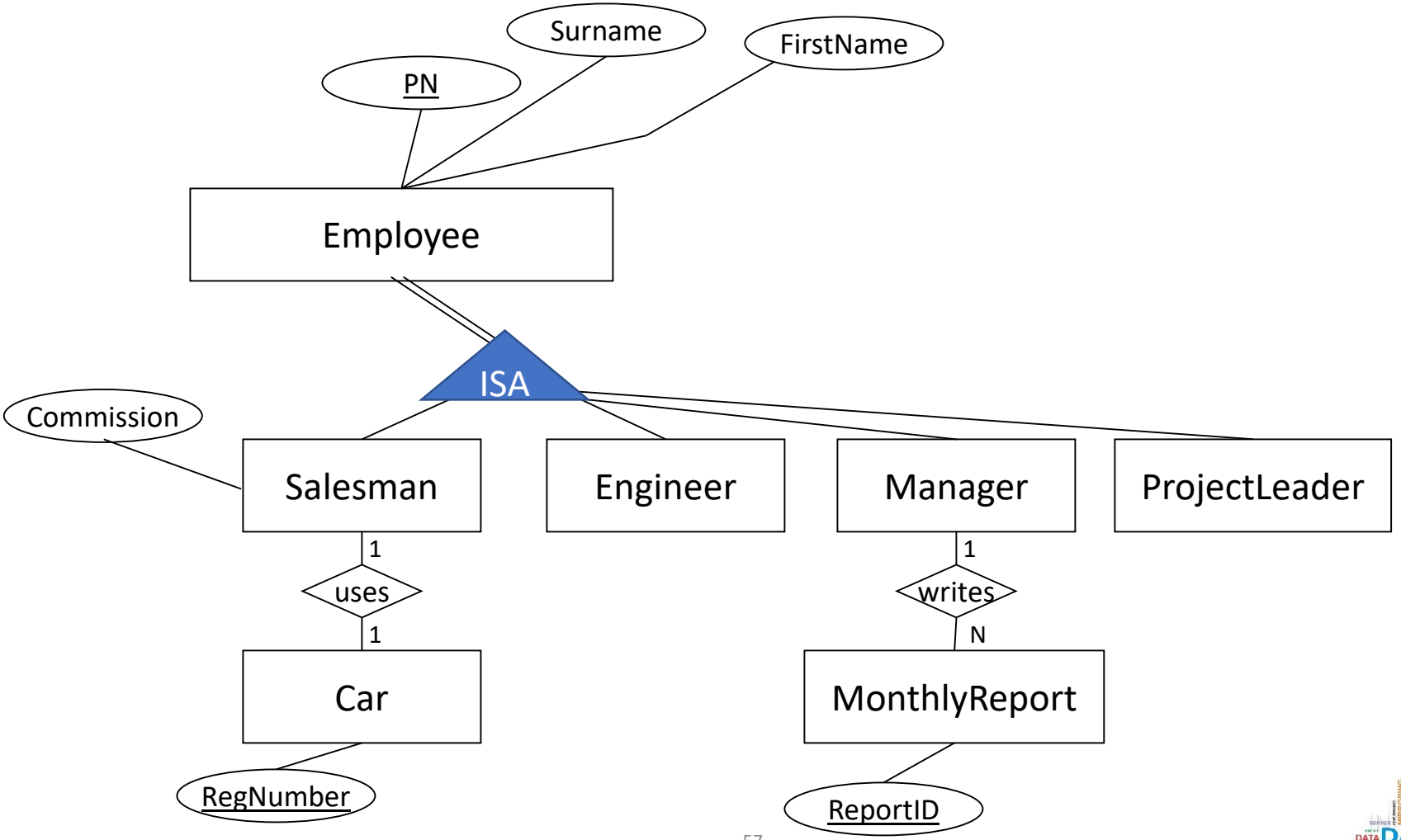






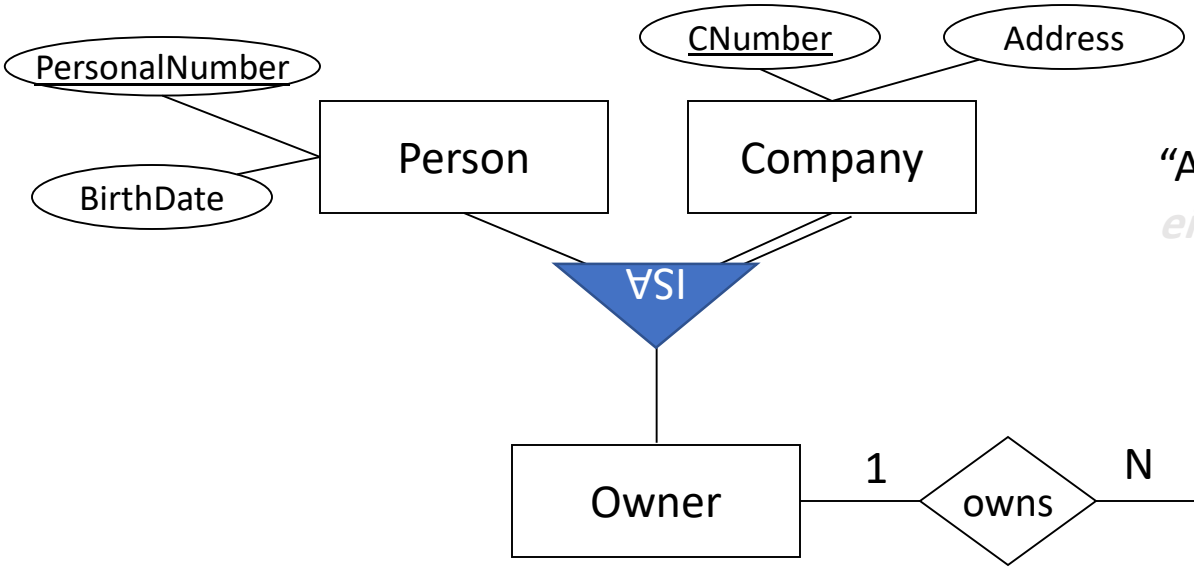


Subclass/superclass



Class Hierarchies

- A UNION subclass represents a collection of entities that is a subset of the UNION of the entities of the super classes.



"An owner of a car is *either* a person *or* a company."

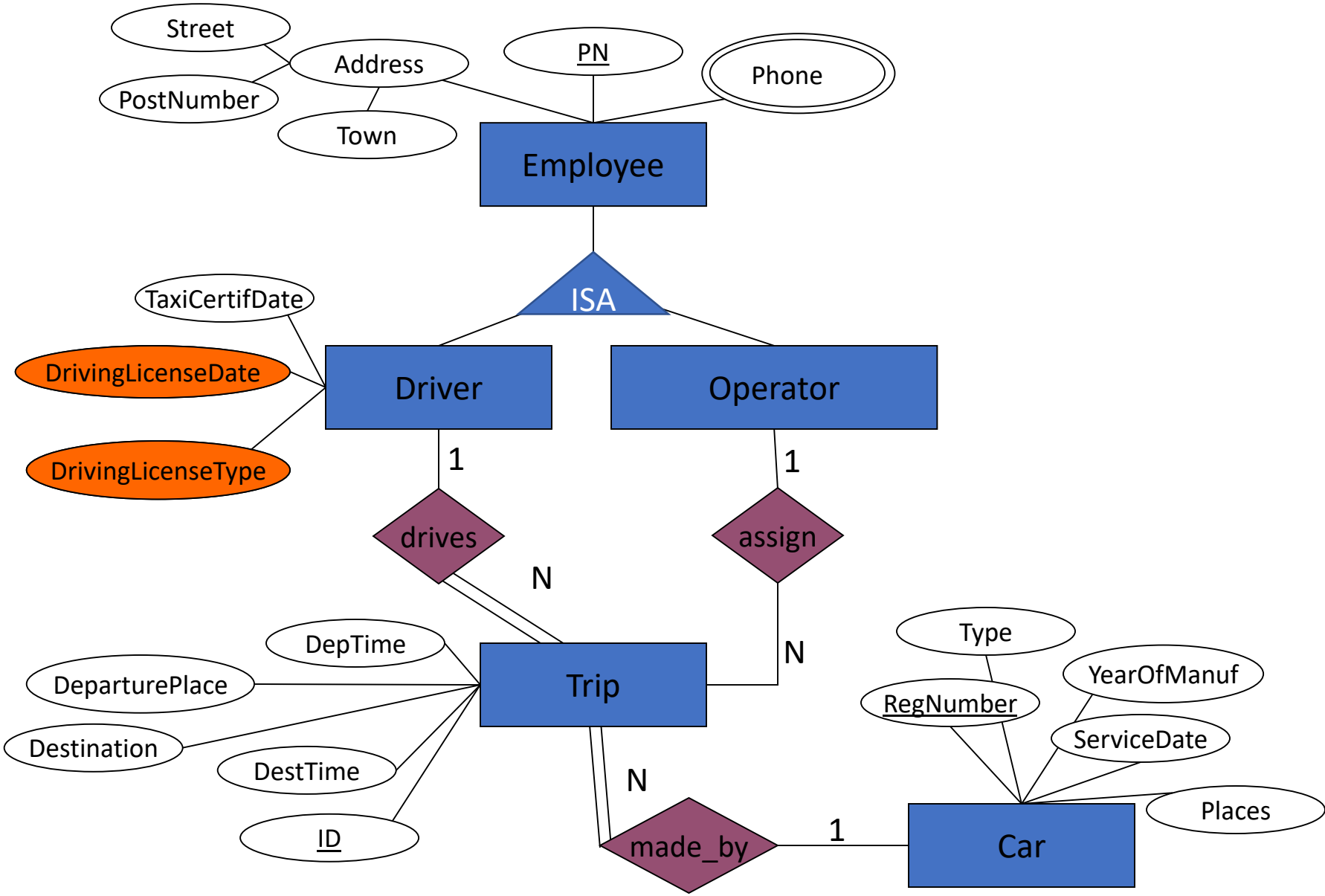
Examples

A taxi company needs to model their activities.

There are two types of employees in the company: drivers and operators. For drivers it is interesting to know the date of issue and type of the driving license, and the date of issue of the taxi driver's certificate. For all employees it is interesting to know their personal number, address and the available phone numbers.

The company owns a number of cars. For each car there is a need to know its type, year of manufacturing, number of places in the car and date of the last service.

The company wants to have a record of car trips (körningar). A taxi may be picked on a street or ordered through an operator who assigns the order to a certain driver and a car. Departure and destination addresses together with times should also be recorded.



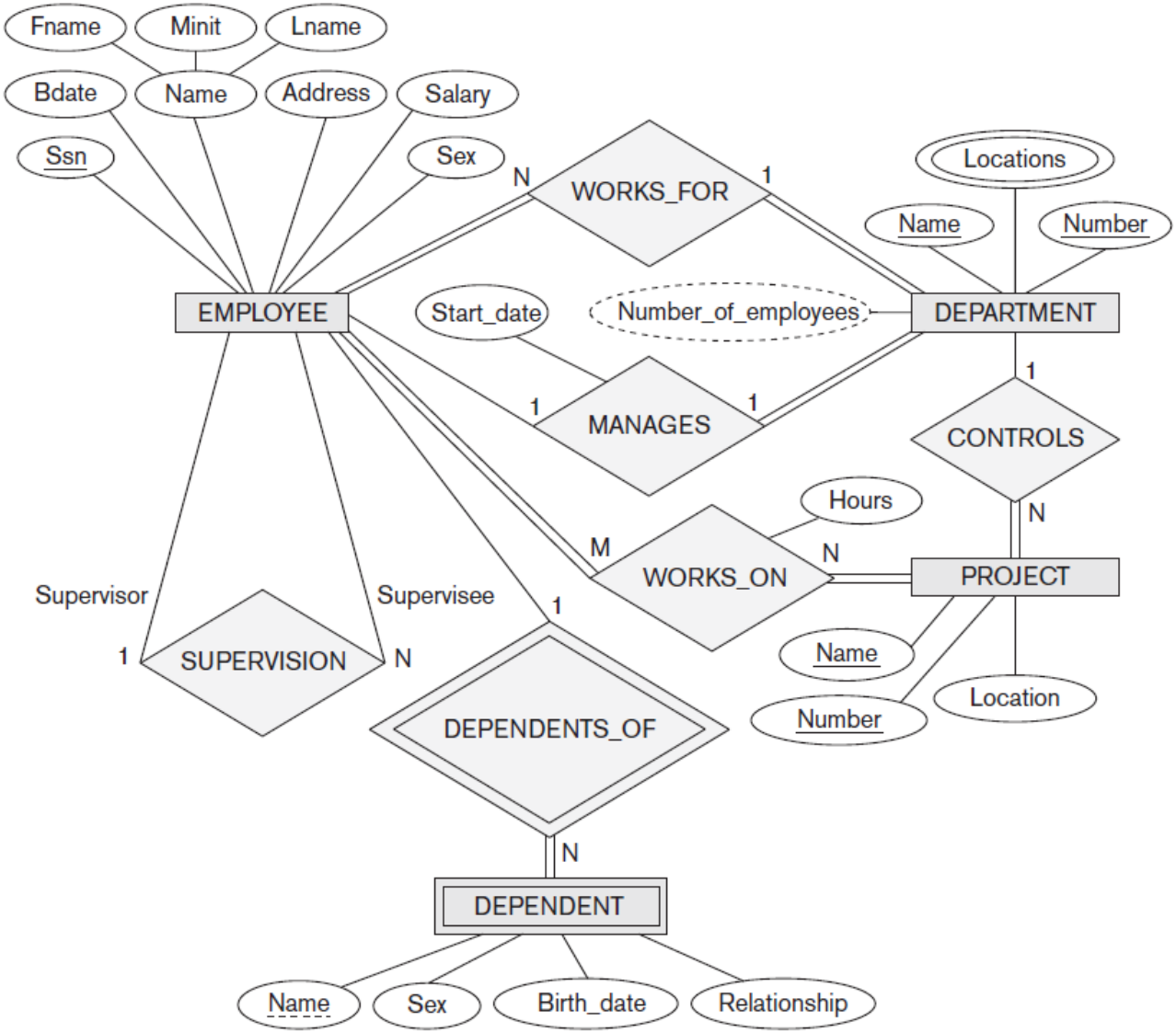
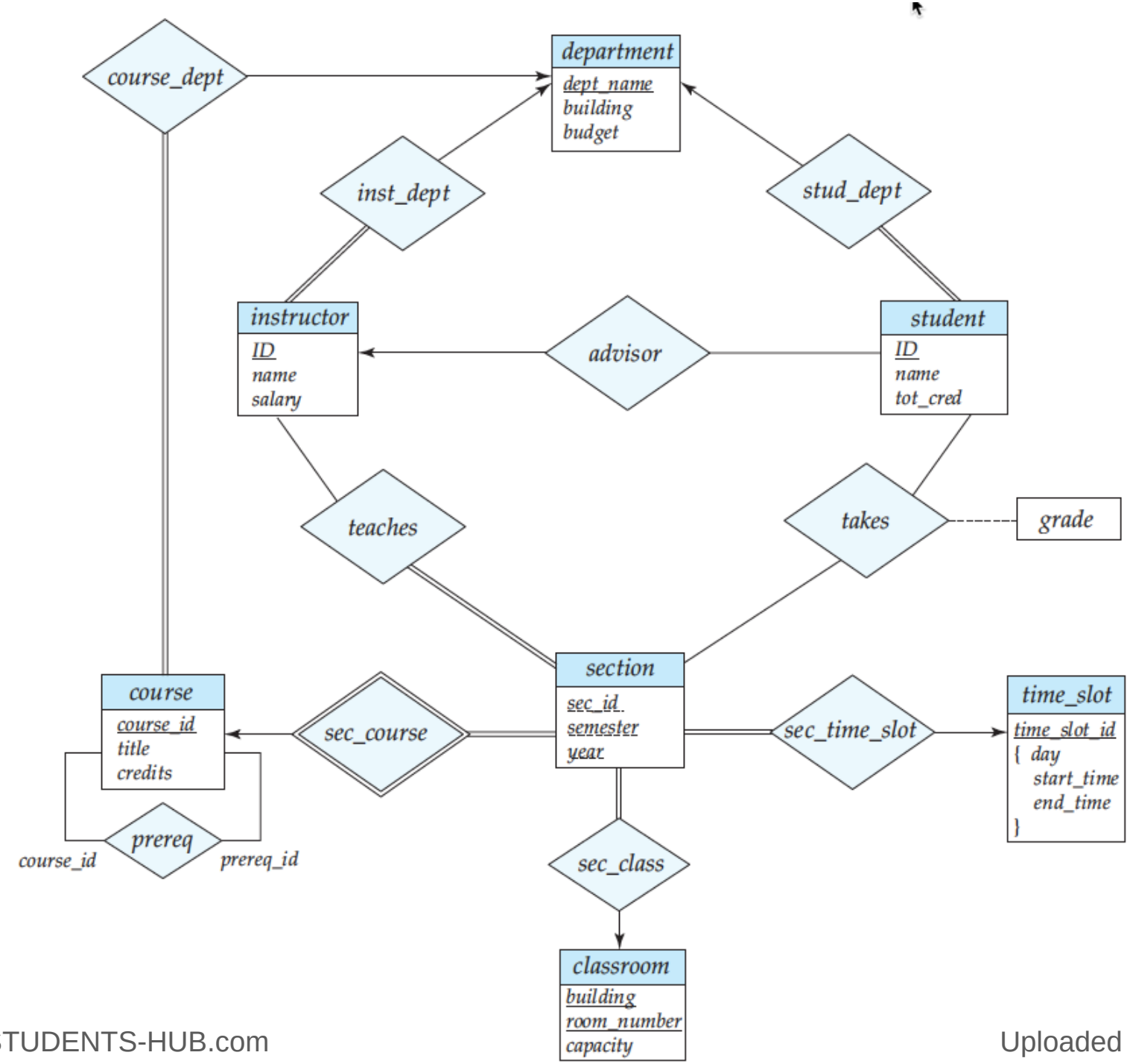
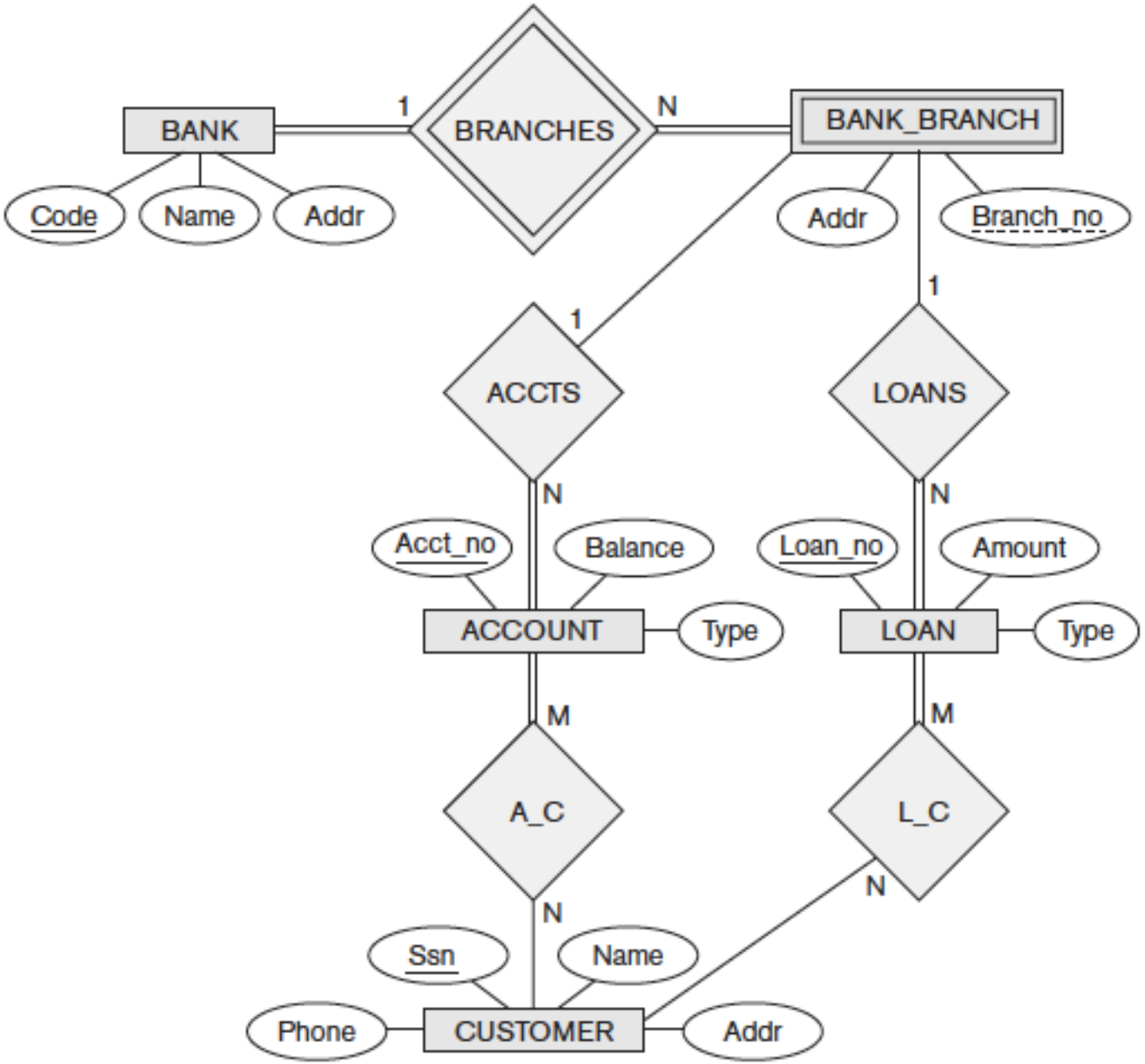
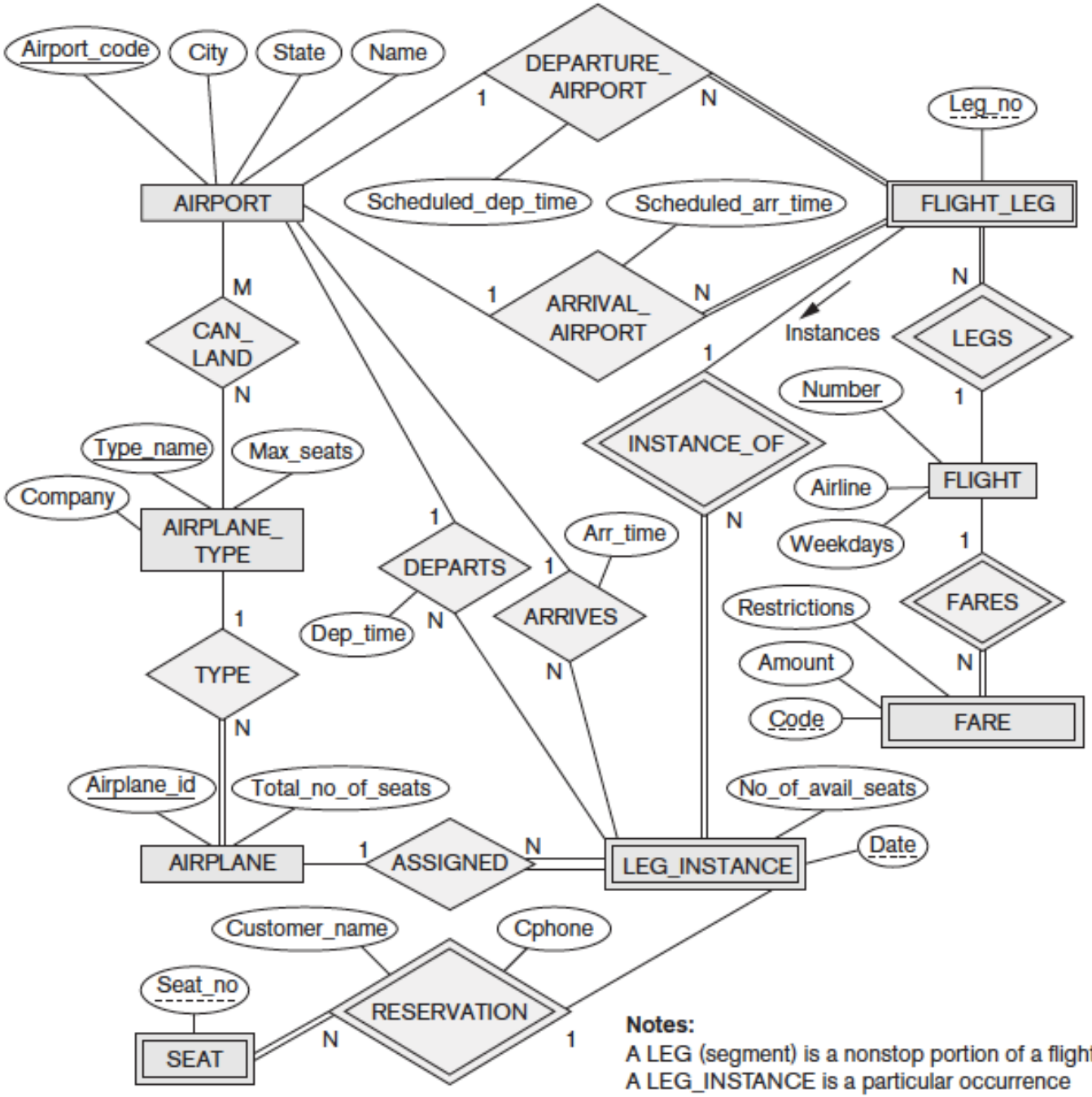


Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.







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.

- Every airplane has a registration number, and each airplane is of a specific model.
- The airport accommodates a number of airplane models, and each model is identified by a model number (e.g., DC-10) and has a capacity and a weight.
- A number of technicians work at the airport. You need to store the name, SSN, address, phone number, and salary of each technician.
- Each technician is an expert on one or more plane model(s), and his or her expertise may overlap with that of other technicians. This information about technicians must also be recorded.
- Traffic controllers must have an annual medical examination. For each traffic controller, you must store the date of the most recent exam.
- All airport employees (including technicians) belong to a union. You must store the union membership number of each employee. You can assume that each employee is uniquely identified by a social security number.
- The airport has a number of tests that are used periodically to ensure that airplanes are still airworthy. Each test has a Federal Aviation Administration (FAA) test number, a name, and a maximum possible score.
- The FAA requires the airport to keep track of each time a given airplane is tested by a given technician using a given test. For each testing event, the information needed is the date, the number of hours the technician spent doing the test, and the score the airplane received on the test.

- Patients are identified by an SSN, and their names, addresses, and ages must be recorded.
- Doctors are identified by an SSN. For each doctor, the name, specialty, and years of experience must be recorded.
- Each pharmaceutical company is identified by name and has a phone number.
- For each drug, the trade name and formula must be recorded. Each drug is sold by a given pharmaceutical company, and the trade name identifies a drug uniquely from among the products of that company. If a pharmaceutical company is deleted, you need not keep track of its products any longer.
- Each pharmacy has a name, address, and phone number.
- Every patient has a primary physician. Every doctor has at least one patient.
- Each pharmacy sells several drugs and has a price for each. A drug could be sold at several pharmacies, and the price could vary from one pharmacy to another.
- Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescriptions from several doctors. Each prescription has a date and a quantity associated with it. You can assume that, if a doctor prescribes the same drug for the same patient more than once, only the last such prescription needs to be stored.
- Pharmaceutical companies have long-term contracts with pharmacies. A pharmaceutical company can contract with several pharmacies, and a pharmacy can contract with several pharmaceutical companies. For each contract, you have to store a start date, an end date, and the text of the contract.
- Pharmacies appoint a supervisor for each contract. There must always be a supervisor for each contract, but the contract supervisor can change over the lifetime of the contract.