COSC344
Database Theory and Applications

$\sigma_{a='c'}(P)$

$S \bowtie P$

$\pi_{A,c}(H)$

Lecture 3
The Relational Data Model

Previous Lecture

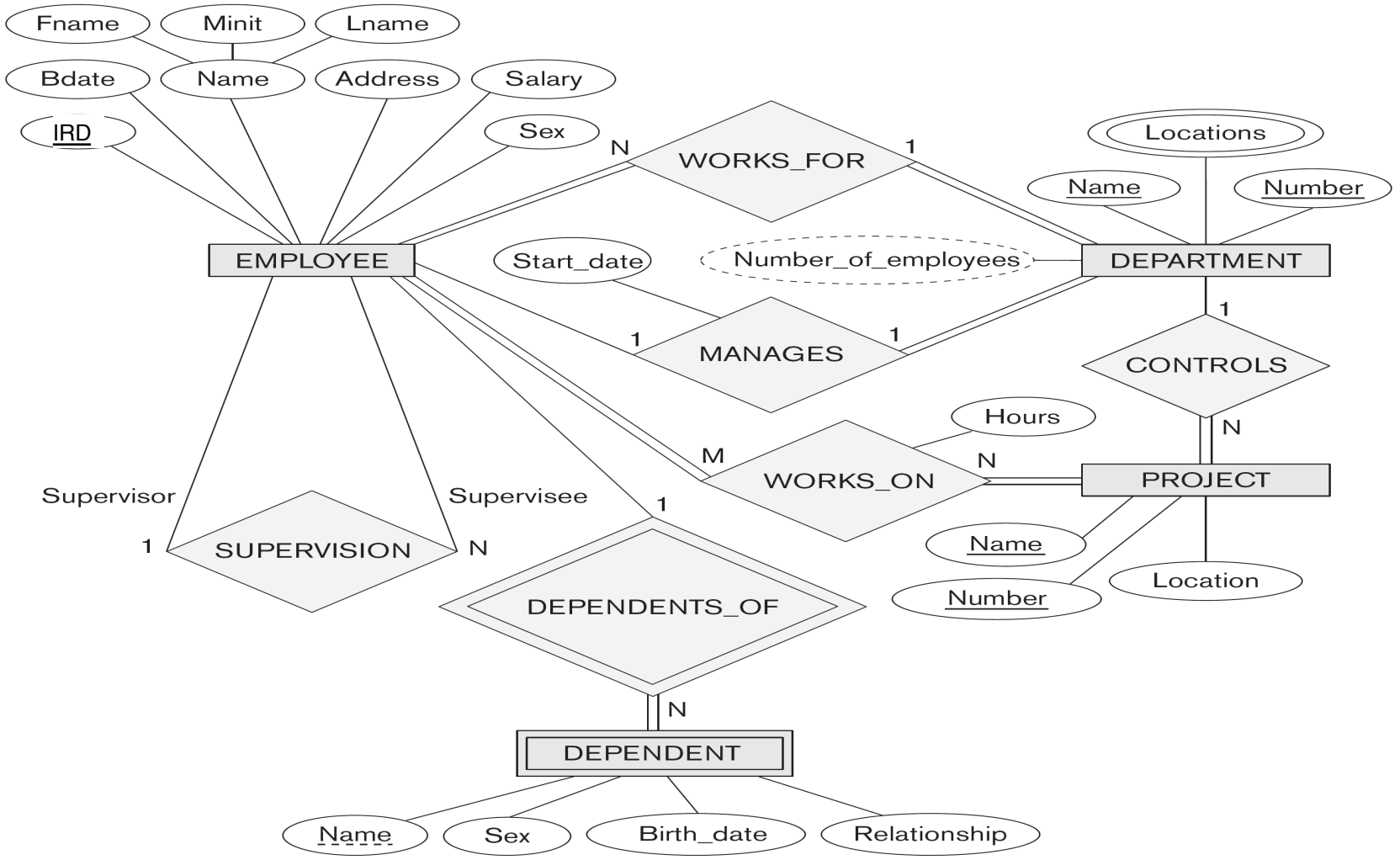
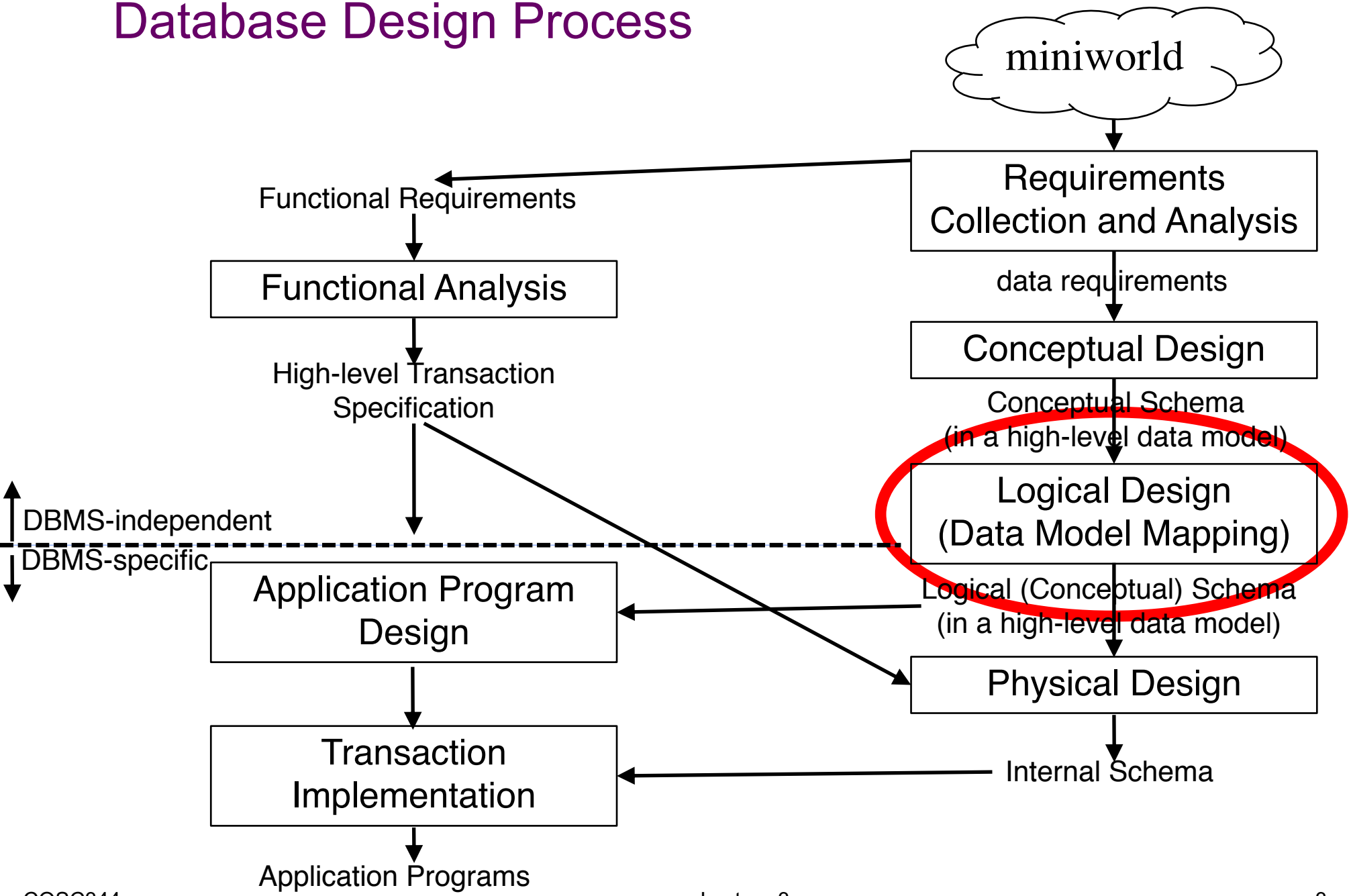


Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.

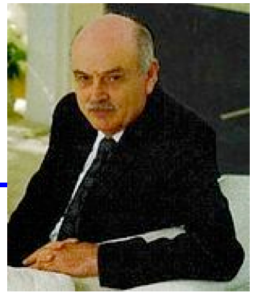
Database Design Process



Learning Objectives of Lecture 3

- You should
  - understand the fundamentals of relational model
    - relation
    - domain
    - superkey, key, candidate key, primary key
    - foreign key
  - understand the difference between ER model and relational model
- Source
  - Textbook: Chapter 5

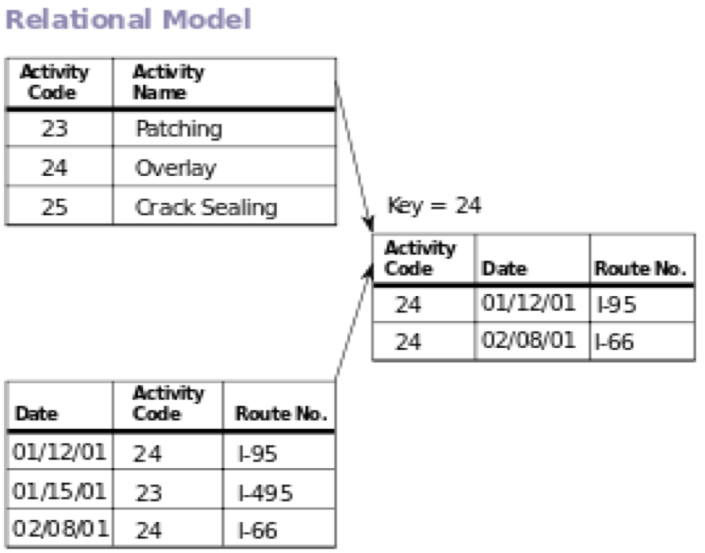
Relational Model



- Invented by E.F. Codd, IBM Research, in 1970

  E.F.Codd, "A Relational Model for Large Shared Data Banks", *Communications of the ACM*, 13:6, June 1970

- Subsequently maintained and developed by Chris Date and Hugh Darwen among others
- Has mathematical theoretical basis
  - Set theory
  - First order predicate logic
- The dominant model for database
  - Oracle, 1979
  - IBM DB2, 1983
  - Microsoft SQL server, 1989



The Relational Data Model

The relational model has 3 core components:

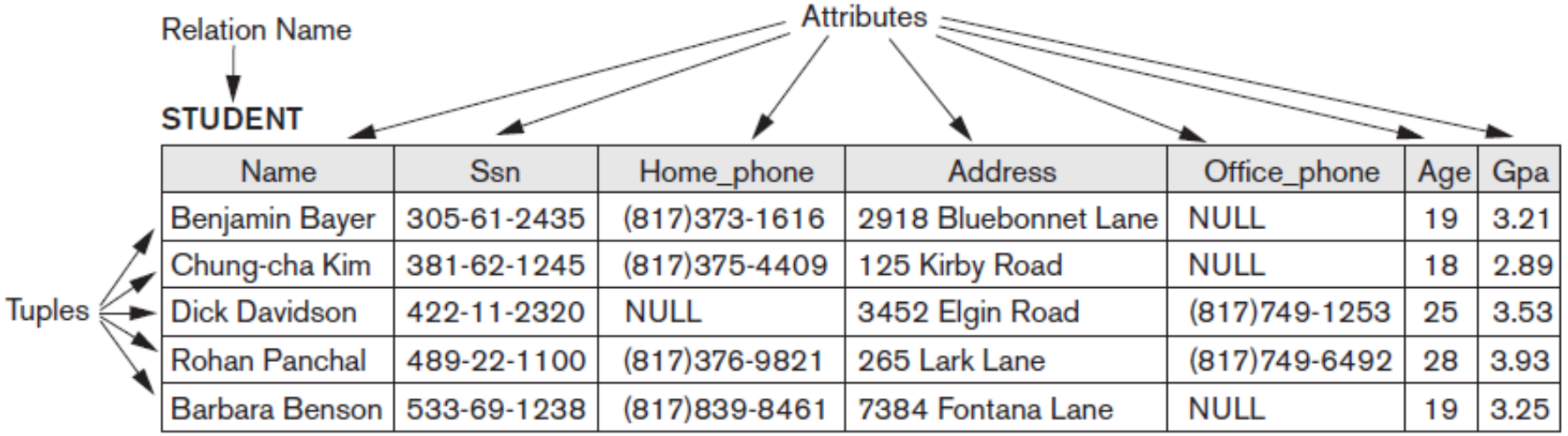
- **Objects (or relations)** – structure of the data organization
- **Integrity** – enforcing constraints and rules
- **Operators** – data manipulation

Relational Model Concepts

- Represents the database as a collection of *relations*
- Informally, each relation resembles a **table** of values
  - Row
    - Represents a collection of related data values
    - A fact that corresponds to a real-world entity or relationship
  - Column
    - Column headers represent attributes, and are used to help interpret the meanings of the values in each row
    - Column values are derived from the domain of the corresponding attribute
- However, a table and a relation are not strictly equivalent

Attributes, Tuples, Relations

- Formally, in the relational model,
  - Relation -> table
  - Tuple -> row
  - Attribute -> column header



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 3.1
The attributes and tuples of a relation STUDENT.

Domains

- A domain
  - is a set of **atomic** values.
  - Has a **data type** or **format**
- Often useful to specify a name for a domain, for example
  - integer
  - real number
  - date
  - alphabetic character string
  - NZ passport number
  - IRD number
  - Address

Relation Schema

- A *relation schema* R, denoted by $R(A_1, A_2, \ldots, A_n)$, is made up of
  - a relation name R and
  - a set of attributes $(A_1, A_2, \ldots, A_n)$.
- Each *attribute* A_i is the name of a role played by some domain D in the relation schema R
  - D is called the *domain* of A_i
  - Is denoted by $dom(A_i)$.
- The ***degree*** of a relation is the number of attributes *n* of its relation schema.

Relation Schema Example

- STUDENT(Name, IRD, HomePhone, Address, OfficePhone, Age, GPA)
- What is the relation name?
- What is the degree of the relation?
- What are the attributes?
- List each dom(A_i).

STUDENT

Name	IRD	Home Phone	Address	Office Phone	Age	GPA

Relation Schema (continued)

- A *relation* (or *relation state*) r of the relation schema $R(A_1, A_2, . . ., A_n)$, also denoted by r(R), is a **set** of n-tuples $r = \{t_1, t_2, . . ., t_k\}$.
- Each *n-tuple* t is a mapping from R to D, and D is the union of the attribute domains.
- A tuple can be considered as a set of (<attribute>,<value>) pairs, where each pair gives the value of the mapping from an attribute A_i to a value v_i from $dom(A_i)$.
- Possible for several attributes to have the same domain. The attributes indicate different roles or interpretations for the domain.

Characteristics of Relations

- Ordering of tuples in a relation
  - Yes/No ?
- Ordering of values within a tuple
  - Yes/No ?
- Values in the tuples
  - First normal form - *i.e. atomic –can't be decomposed*
- How to represent multi-valued attributes?
- How to represent composite attributes?
- NULL
  - Unknown
  - May not apply

Relational Model Constraints

- Constraints
  - Restrictions on the actual values in the database state
  - Derived from the rules in the miniworld that the database represents
- Type of constraints in relational database
  - Domain constraints
  - Key constraints and constraints on NULL
  - Integrity constraints
  - Semantic integrity constraints
  - Functional dependency constraints

Domain Constraints

- Domain constraints
  - specify that the value of each attribute must be an **atomic** value **from the domain dom(A)**.
- Data types are usually associated with domains
  - Standard numeric data types
    - Integer
    - Real number
    - Strings,
    - Boolean
    - Date
    - ….
  - Subrange of values
    - Age is between 0 to 120
  - Enumerated data type
    - Colour

Key Constraints

- No two tuples can have the same combination of values for **all** their attributes.
- **Superkey** - a set of attributes such that for any two distinct tuples t_1 and t_2, $t_1[SK]$!= $t_2[SK]$
  - Any set of attributes satisfying the above is a superkey
  - One default superkey: the set of all its attributes
- **Key**
  - Is a superkey
  - removing any of the attributes from a Key leaves a set of attributes that is not a superkey (**minimal superkey**)
- A relation schema may have more than one key. Each of the keys is called a **candidate key**.
- Typically one candidate key is designated as **primary key**.

Example Schema

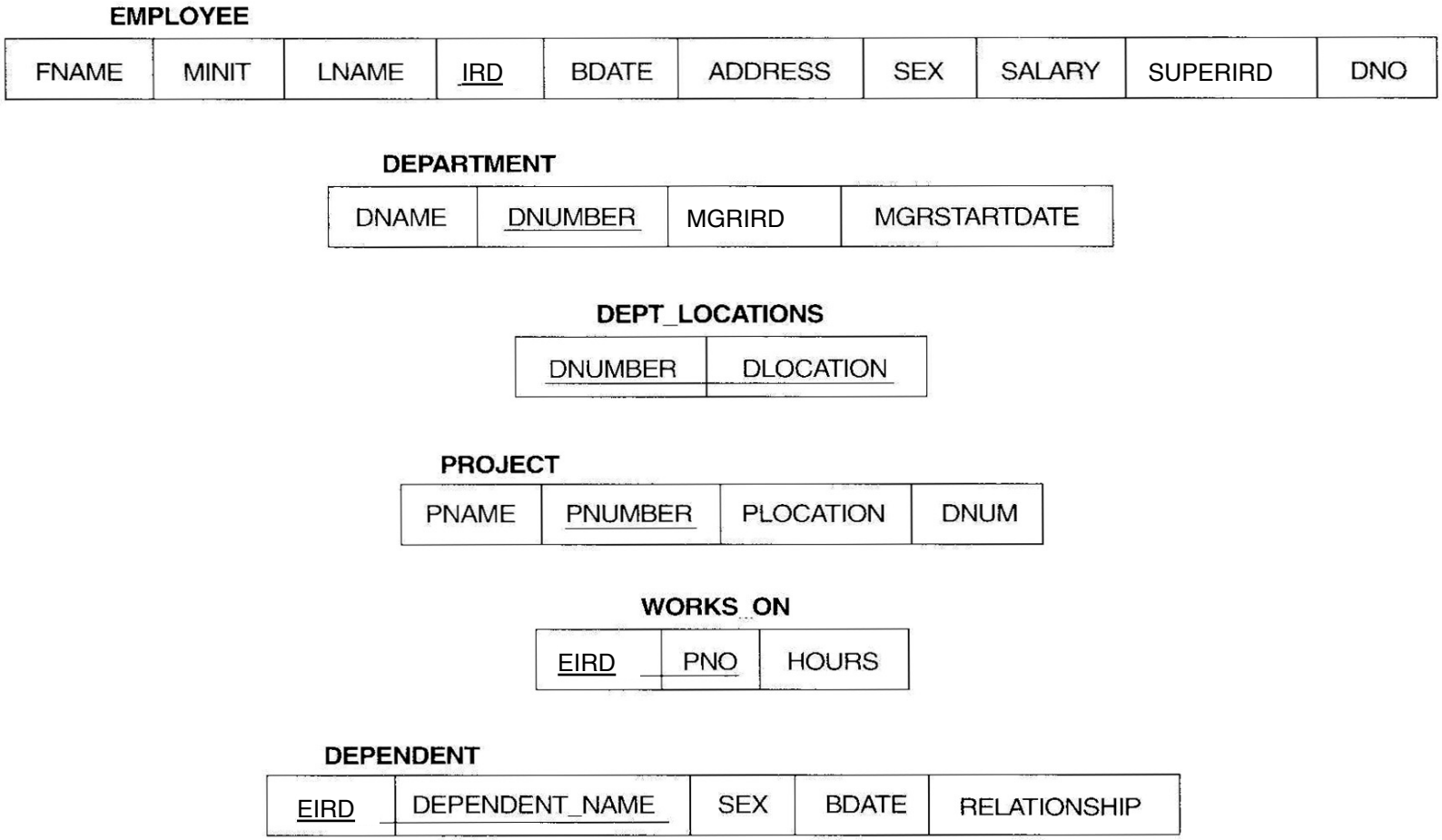


Figure 7.5 Schema diagram for the COMPANY relational database schema; the primary keys are underlined.

Integrity Constraints

- Integrity constraints:
  - Specified on a database schema
  - Are expected to hold on every valid database state of the schema
- Types
  - Domain, Key, NOT NULL
  - Entity Integrity constraint
    - No primary key values can be NULL
  - Referential Integrity constraint
    - Specified between two relations
    - Maintains consistency among tuples in two relations
    - Informally, it states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation.

Example

Student

Name	**Home Phone**	**Address**
Dan	479001	ertyu
Emma	479002	qwer
Frank	479002	qwer
Gina	479002	qwer
Hamish	479005	tyufgd
Ian	479006	ghjgjh
Jo	479006	ghjghj
NULL	479008	iuwer

Tutors

Paper	**Stream**	**Senior Student**	**Student Rep**
COSC344	A	Dan	NULL
COSC344	B	Emma	NULL
COMP112	A	Frank	Hamish
COMP112	B	Emma	Ian
COMP112	C	Gina	Jo
COMP112	D	Dan	Pete
COMP160	A	Frank	NULL

Foreign Keys

- A set of attributes FK in relation schema R_1 is a **foreign key** of R_1 that **references** relation R_2 if it satisfies these conditions:
  - The attributes in FK have the same domain(s) as the primary key attributes PK of R_2; the attributes FK are said to refer to the relation R_2.
  - A value of FK in a tuple t_1 of the current state $r_1(R_1)$ either occurs as a value of PK for some tuple t_2 in the current state $r_2(R_2)$ or is NULL. In the former case, $t_1[FK] = t_2[PK]$, and we say that the tuple t_1 refers to the tuple t_2.
- R_1 is called the referencing relation, and R_2 is called the referenced relation.

Referential Integrity Revisited

- What are the referential integrity constraints of the example database?
- Can be shown in the schema diagram as directed arrows drawn from each foreign key to the primary key of the referenced relation.
- A foreign key can refer to its own relation.

Referential Integrity Example

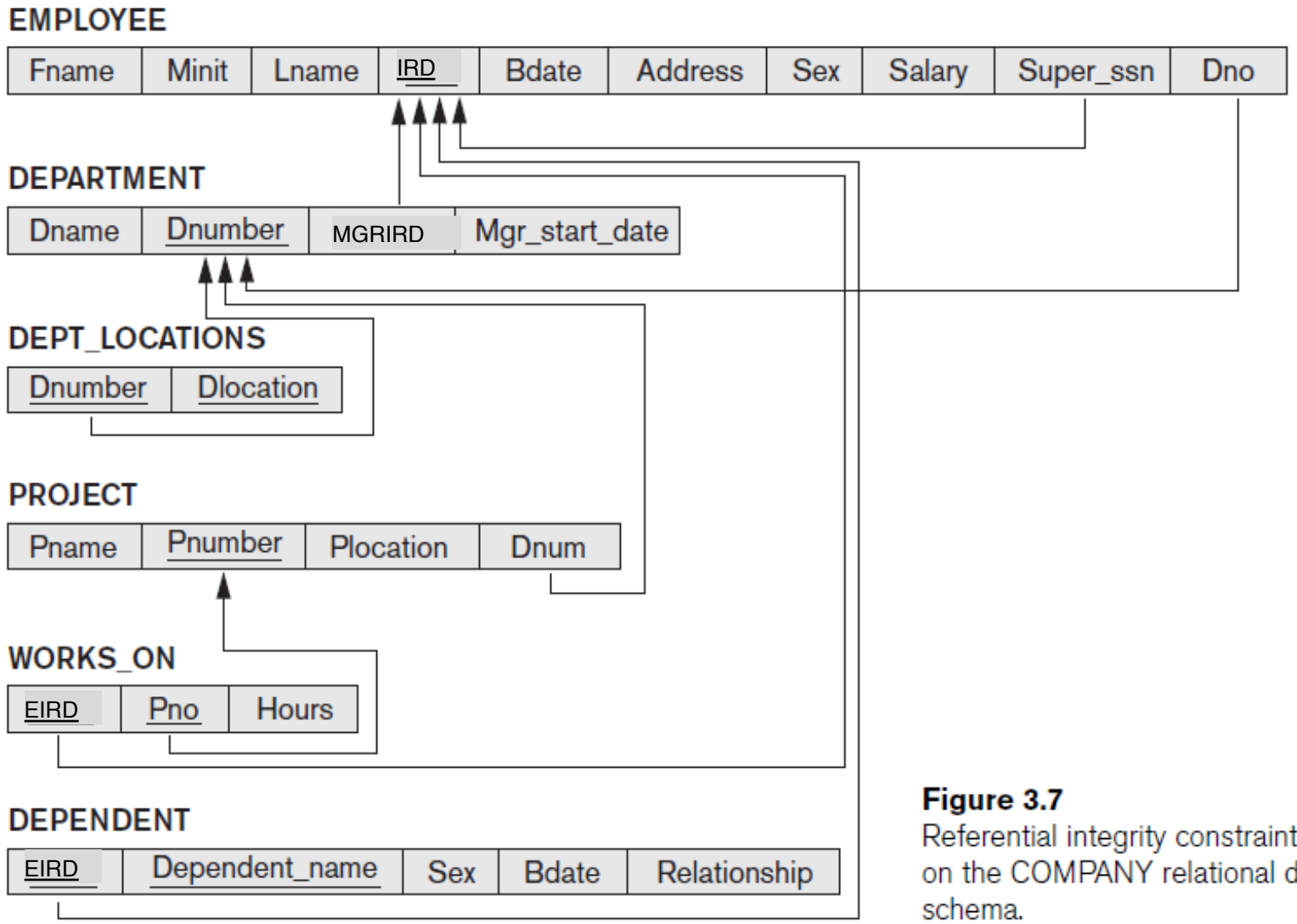


Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

Other Types of Constraints

- Semantic integrity constraints
  - Example: the salary of an employee should not exceed the salary of the employee's supervisor.
  - Enforced using trigger and assertion mechanism (will be discussed later)
- Functional dependency constraints
  - Establish a functional relationship among two sets of attributes X and Y
  - Will be discussed in later lectures

Question to Ponder

- How do you turn an ER diagram into a relational model?

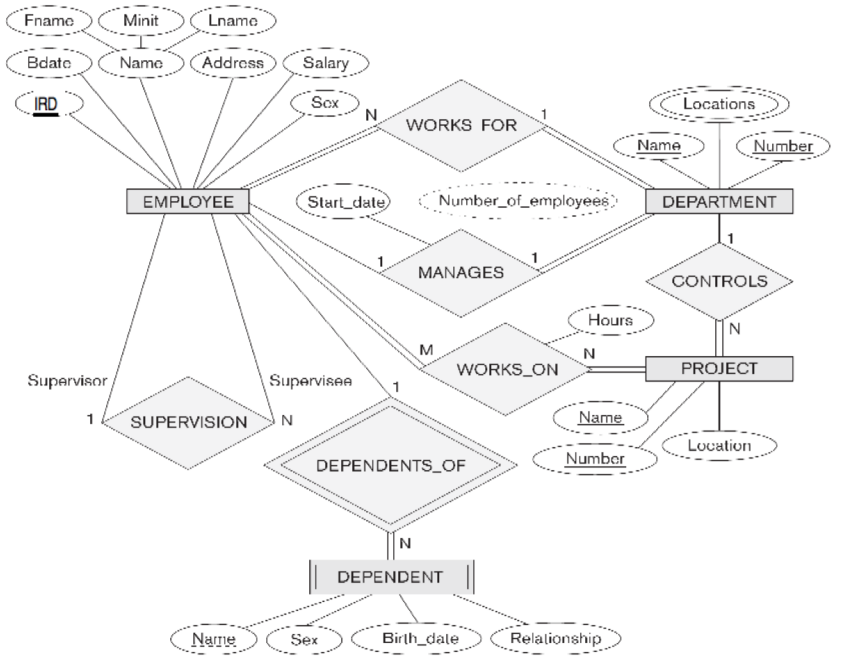


Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.

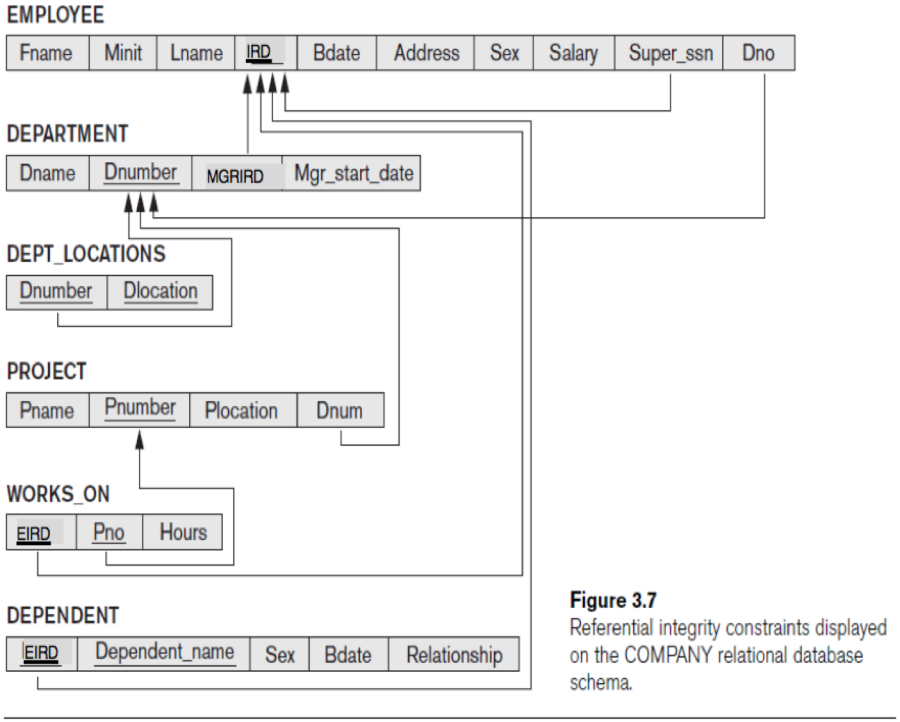


Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

Assignment 1

- Grouping
  - Each team has 4 members, randomly assigned.
  - Check the group you are in from Blackboard
- Assignment

  Pick an application domain or mini-world that your group is interested, and model the chosen mini-world
  - backpacker hotel, taxi service, Pokémon game …
- Tools for drawing the ER diagram
  - draw.io (https://app.diagrams.net)
  - Dia (http://dia-installer.de)
- Three videos illustrate how to use Dia for ER modelling
  - In 344 pickup directory /home/cshome/coursework/344/pickup/00-ERD_videos