Revision
Classes and ERDs

COMP 1531, 17s2

Aarthi Natarajan

Week 12

Case Study

- UNSW has several departments. Each department is managed by a chair, and at least one professor.
- Professors must be assigned to one, but possibly more departments. At least one professor teaches each course, but a professor may be on sabbatical and not teach any course.
- Each course may be taught more than once by different professors.
- We know of the department name, the professor name, the professor employee id, the course names, the course schedule, the term/year that the course is taught, the departments the professor is assigned to, the department that offers the course.

Steps to drawing the class diagram (contd…)

1. Identify classes
   - Abstract or tangible “things” in our problem domain (nouns and noun phrases) determined from requirement analysis
   - e.g., departments, chair, professor
2. Find associations
   - Verbs that join the nouns e.g., professor (noun) teaches (verb) students (noun)
3. Draw CRC diagram

Defining the CRC cards

Professor	
Assigned to a Department *Teaches* Course *Knows* Name *Knows* Employee ID	Department Course

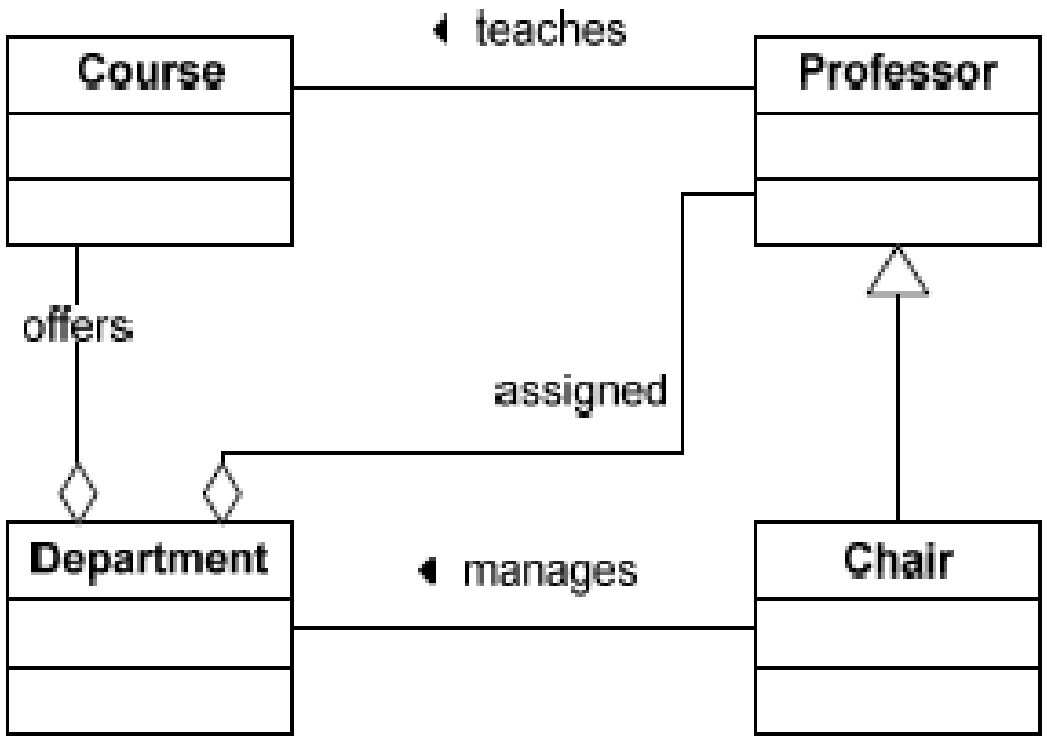
Department	
Managed by a Chair *Is Assigned* Professors *Offers* Courses *Knows* Department Name	Chair Professor Course

Course	
Offered by a Department *Taught by* Professor Knows schedule Knows term/year offered	Department Professor

Chair	
Manages a Department *Is a* Professor *Knows* Department Name	Department Professor

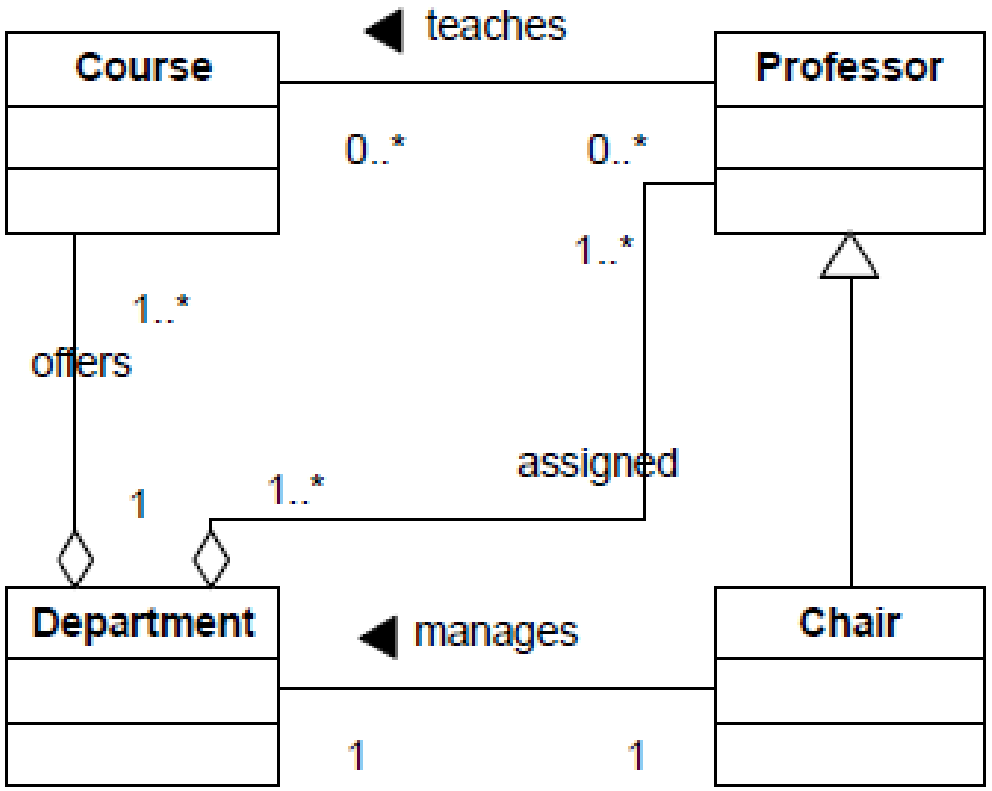
Steps to drawing the class diagram (contd…)

4. Draw the conceptual class diagram



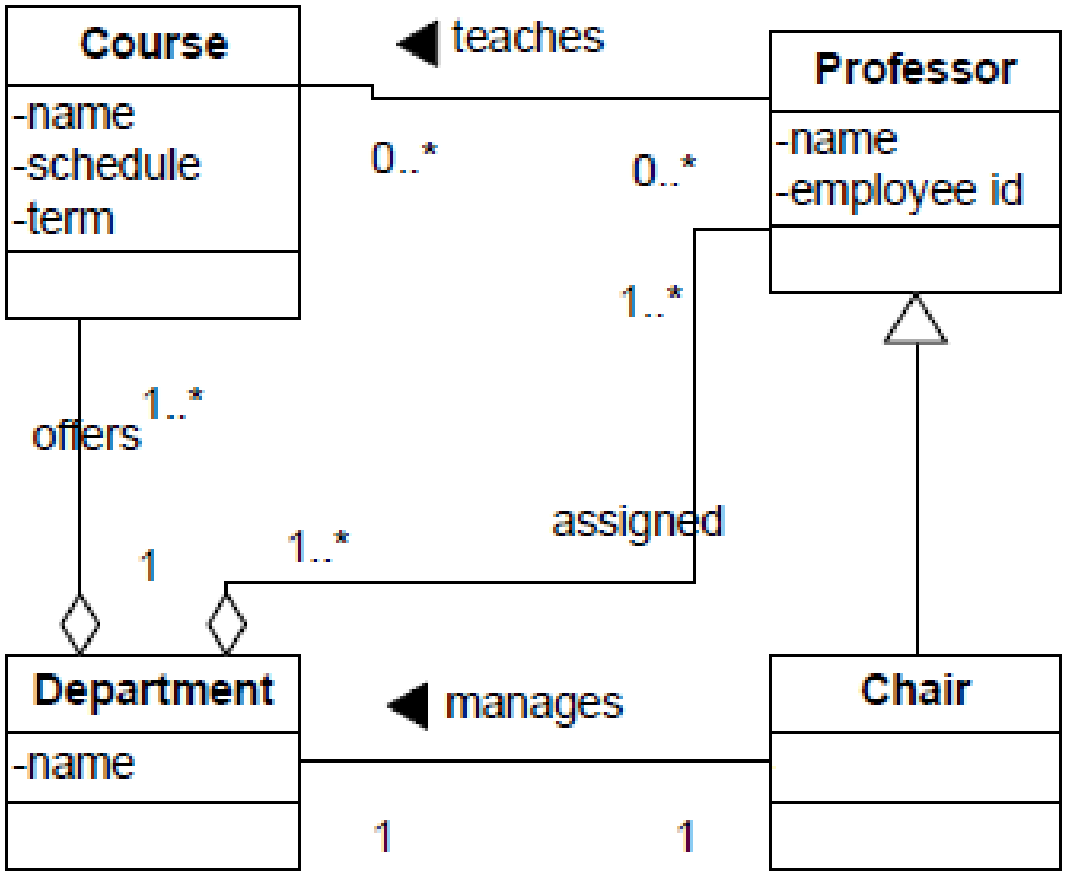
Steps to drawing the class diagram (contd…)

5. Fill in the multiplicity



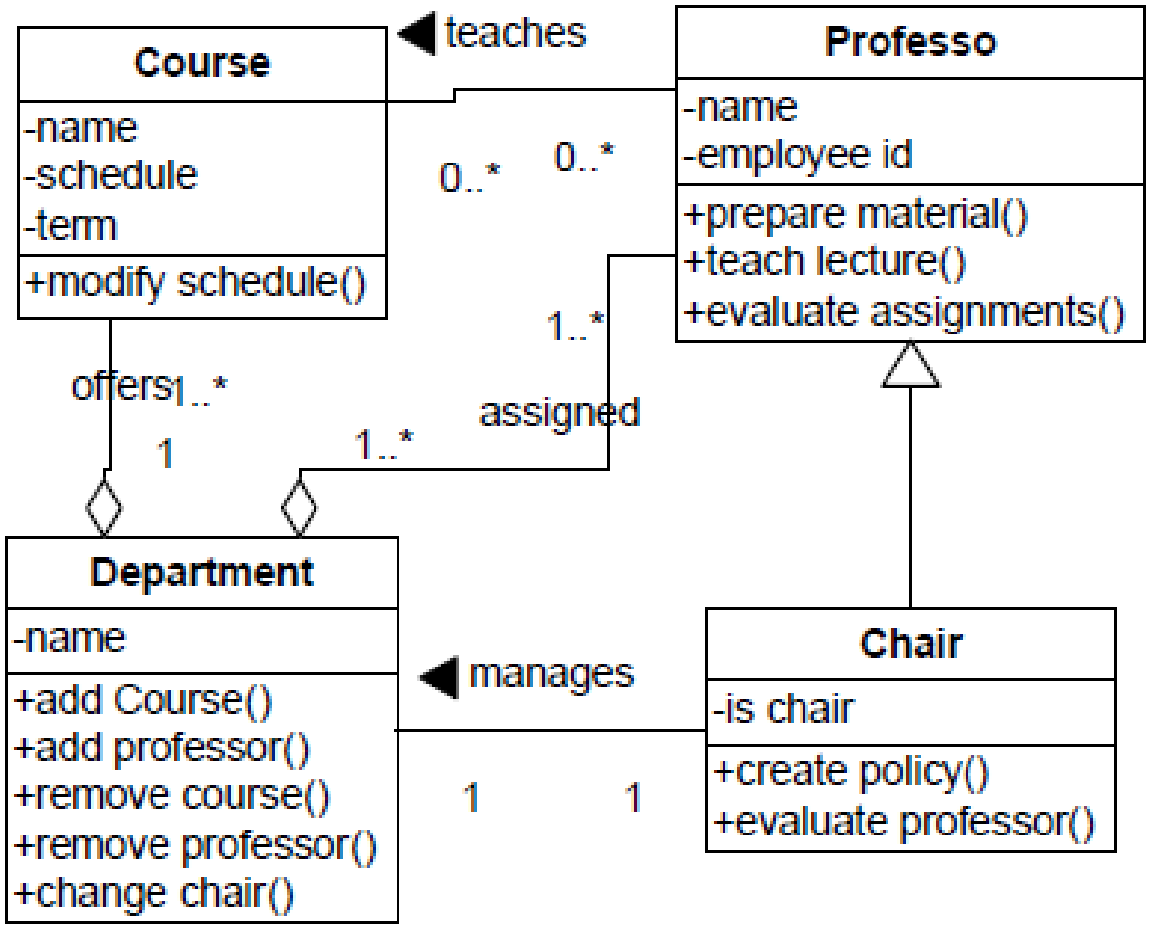
Steps to drawing the class diagram (contd…)

5. Identify attributes



Steps to drawing the class diagram (contd…)

5. Identify behaviours
6. Review class diagram and fine tune it



Database Revision

COMP 1531

Week 12

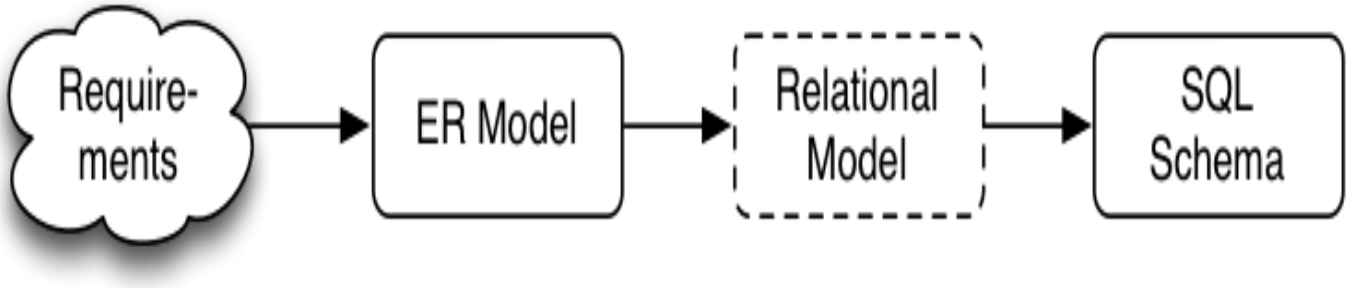
Designing a database

Two data models

- Logical: abstract model e.g., ER Model, OO Model
- Physical: record-based models e.g., relational model

A strategy for designing a database

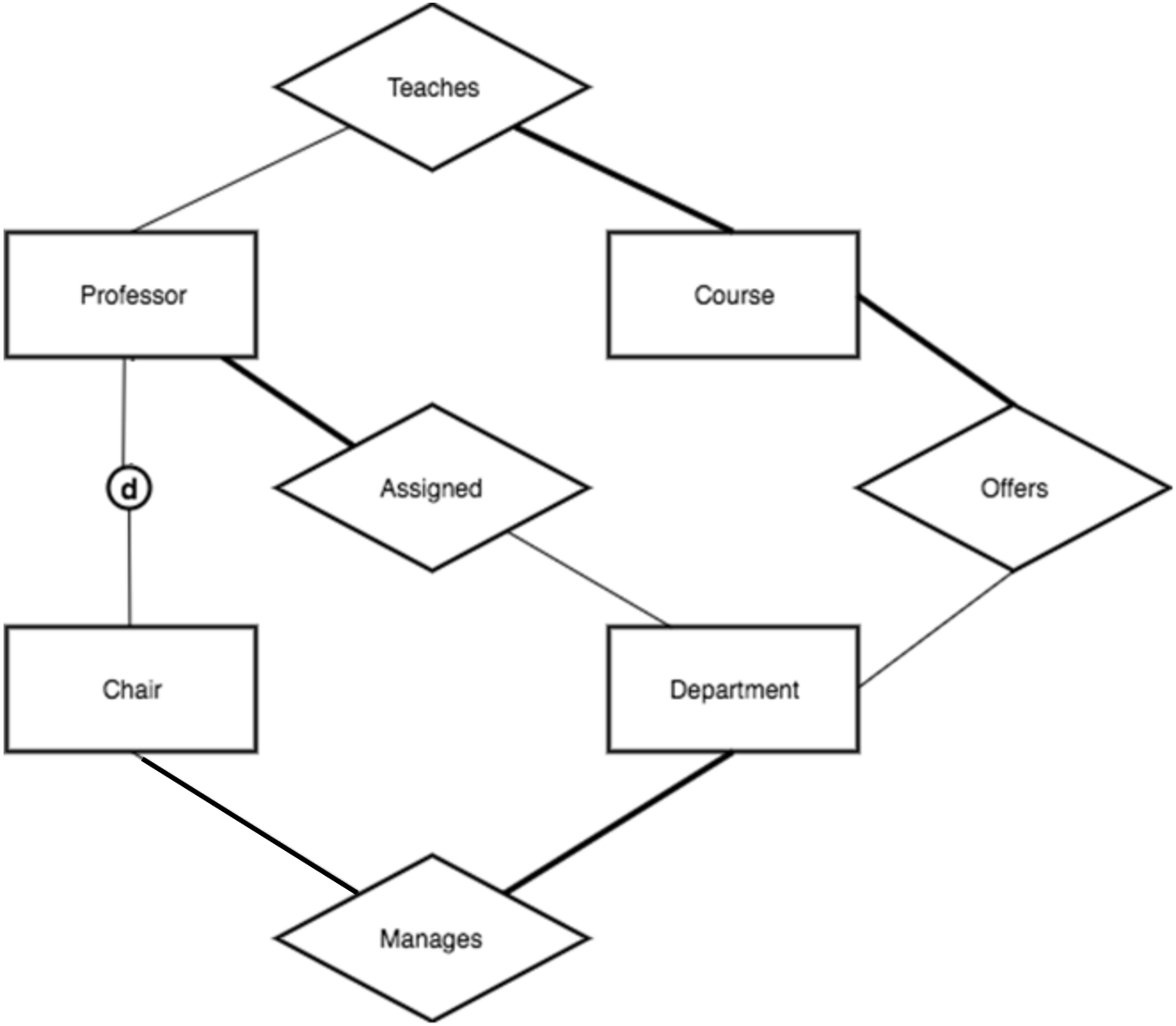
- Design using abstract model (conceptual-level modelling)
- Map to physical model (implementation-level modelling)



Steps to drawing the entity relationship diagram

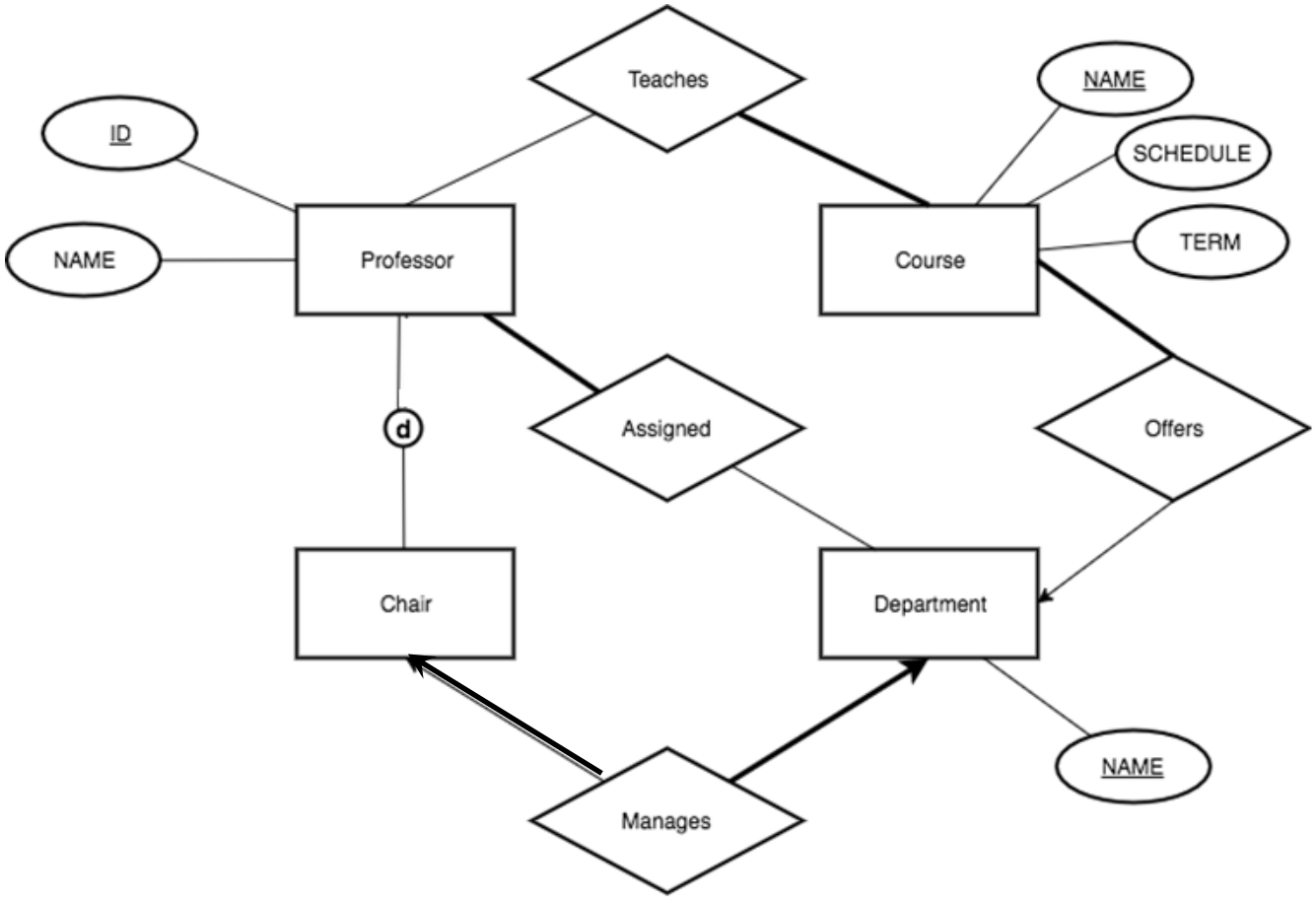
1. Identify entities
   - Typically the nouns and noun phrases determined from requirement analysis
   - Include only entities relevant to problem domain
   - e.g., departments, chair, professor
2. Find relationships
   - Verbs that join the nouns e.g., professor (noun) teaches (verb) students (noun)
3. Draw conceptual ER diagram

Steps to drawing the class diagram (contd…)



Steps to drawing the entity relationship diagram

5. Add cardinality
6. Identify the entity attributes
7. Identify the primary key



Relational Data Model

The relational data model describes the world as a collection of inter-connected relations (or tables)

Goal of relational model:

- a simple, general data modelling formalism
- maps easily to file structures (i.e. implementable)

Relational model has two styles of terminology:

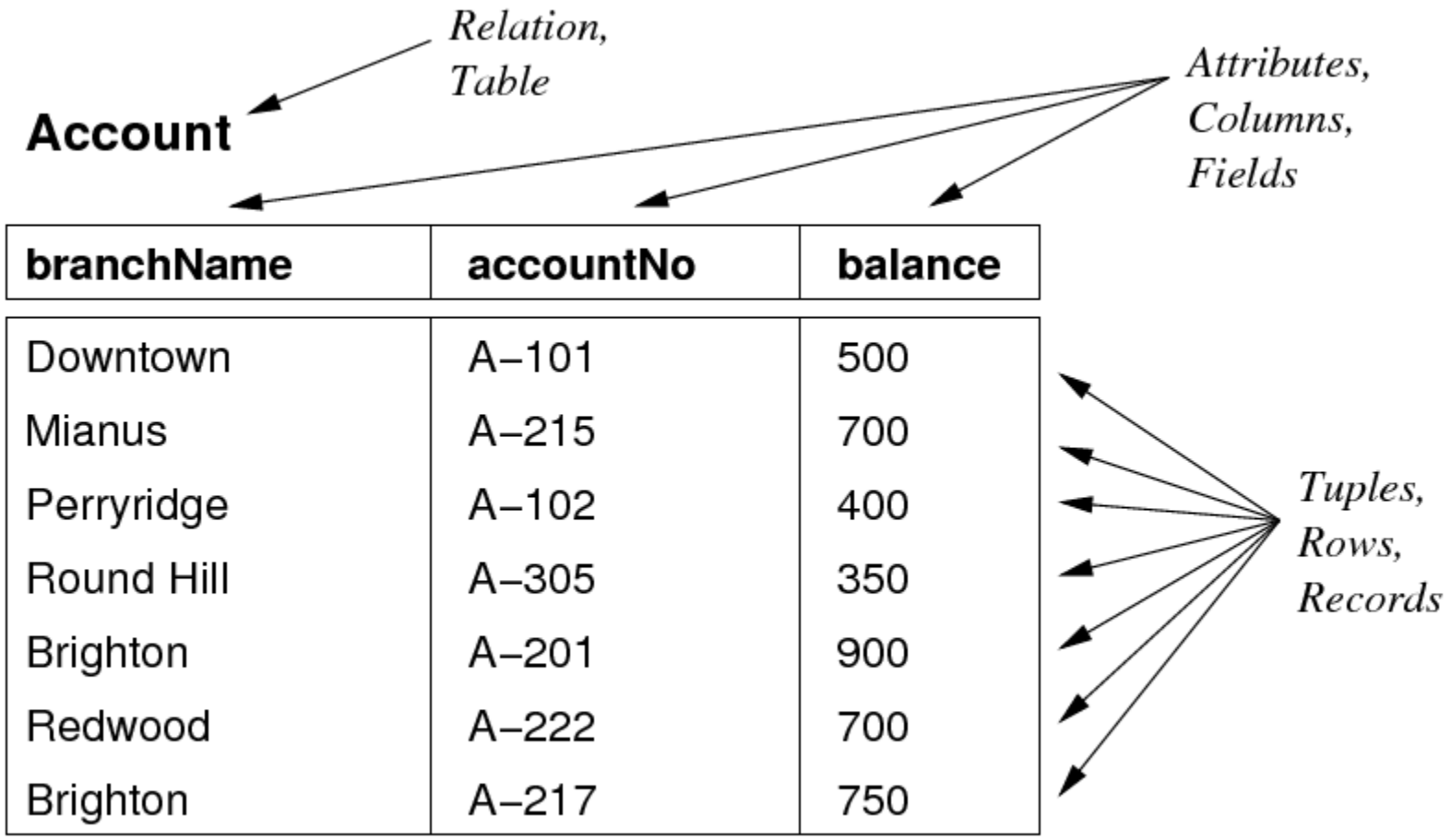
mathematical	relation	tuple	attribute
data-oriented	table	record (row)	field (column)

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

Relational Data Model

Example of a relation (table): Bank Account



Constraints

Relations are used to represent real-world *entities* and *relationships* between these *entities*

To represent real-world problems, need to describe

- what values are/are not allowed
- what combinations of values are/are not allowed

Constraints are logical statements that do this:

- Domain constraint
- Key constraint
- Entity constraint
- Integrity constraint
- Referential integrity

Referential Integrity Example

Table DEPARTMENT (Parent Table)

Primary Key in Parent Table

DEPT_NO	DEPT_NAME	CITY
10	MARKETING	SYDNEY
11	SALES	SYDNEY
12	TECH	MELBOURNE

Foreign Key in child table must match a primary key in the parent table

Table EMPLOYEE (Child Table)

EMP_NO	EMP_NAME	ROLE	DEPT
5012	John	CEO	10
5016	Alison	SALESPERSON	11
5018	Cathy	MANAGER	12

Insert Fails due to violates the referential integrity constraint

5015 Mitchell SALESPERSON 30

Relational Model vs Entity Model

Correspondences between relational (R) and ER data models:

- ER attribute → relational attribute
- ER entity → relational tuple
- ER entity-set → relational table (relation)
- ER relationship → relational table (relation)
- ER key → relational primary key

Differences between relational and ER models:

- Relational uses *relations* to model *entities* and *relationships*
- Relational has **no** *composite* or *multi-valued* attributes (only atomic)
- Relational has **no** *object-oriented notions* (e.g. subclasses, inheritance)

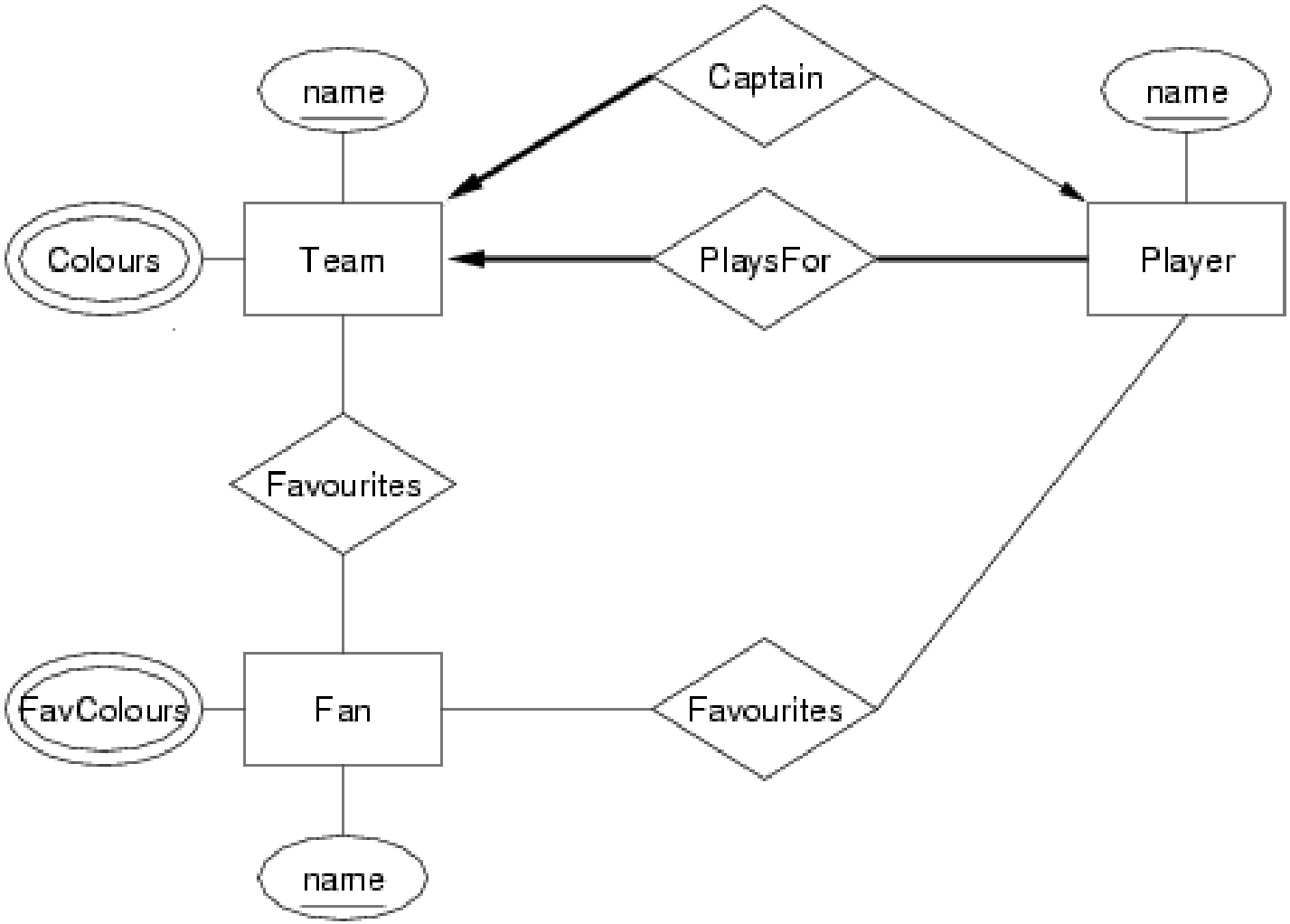
Case Study:

Develop an ER design for the following scenario:

A database records information about teams, players, and their fans, including:

- For each team, its name, its players, its captain (one of its players) and the colours of its uniform.
- For each player, their name and team.
- For each fan, their name, favourite teams, favourite players, and favourite colours.

Solution: ER Design

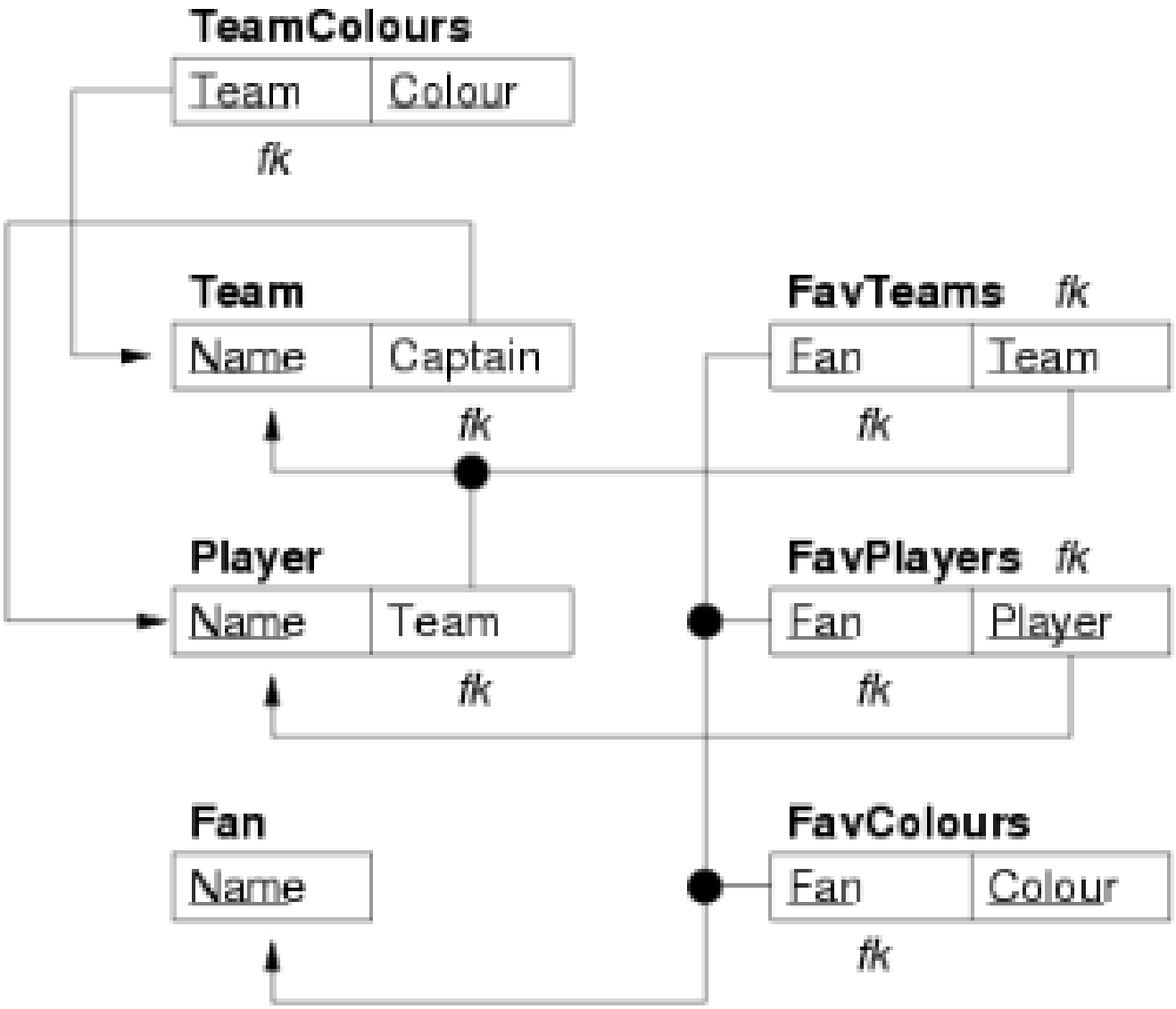


Convert the ER design into a relational data model

(1) first as a box-and-arrow diagram

(2) as a sequence of statements in the SQL data definition language:

Solution: Relational Model



SQL Schema

```
CREATE TABLE Team
(
        name            varchar(50) PRIMARY KEY,
        captain         varchar(40) NOT NULL REFERENCES Player(name)
);
CREATE TABLE Player
(
        name            varchar(40) PRIMARY KEY,
        team            varchar(50) NOT NULL REFERENCES Team(name)
);
CREATE TABLE Fan
(
        name            varchar(40) PRIMARY KEY,
);
CREATE TABLE TeamColours
(
        team            varchar(50) REFERENCES Team(name),
        colour          varchar(30),
        PRIMARY KEY     (team,colour)
);
CREATE TABLE FavTeams
(
        fan             varchar(50) REFERENCES Fan(name),
        team            varchar(50) REFERENCES Team(name),
        PRIMARY KEY     (fan,team)
);
CREATE TABLE FavPlayers
(
        fan             varchar(50) REFERENCES Fan(name),
        player          varchar(50) REFERENCES Player(name),
        PRIMARY KEY     (fan,player)
);
CREATE TABLE FavColours
(
        fan             varchar(50) REFERENCES Fan(name),
        colour          varchar(30),
        PRIMARY KEY     (fan,colour)
);
```