

# CS4221: Database Schema Design

## A Brief Introduction on Hierarchical and Network Data Models

Ling Tok Wang  
National University of Singapore

# Database Models

- File system
- **Hierarchical** Model (IMS)
- **Network** Model (IDMS)
- **Relational** Model
- **Nested Relational** Model
- **Entity-Relationship** Approach
- **Object-Oriented** Data Model
- **Deductive and Object-Oriented** Database
- **Semi-structured** Data Model (XML data)

# 1. Hierarchical Model

## IMS (Information Management System) Data Model

(IBM product) Ref: CJ Date's book

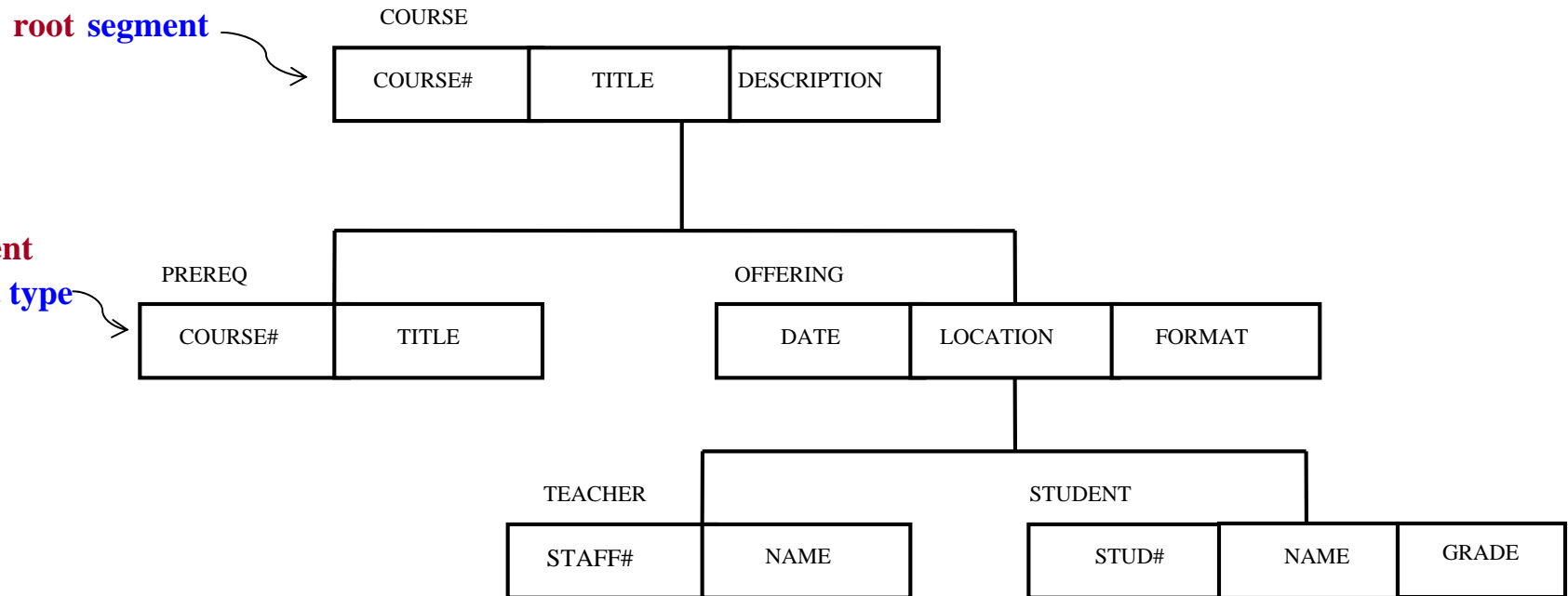


Fig. 1. PDBR (physical database record) type for the education database (schema)

# IMS (Information Management System) Data Model (cont.)

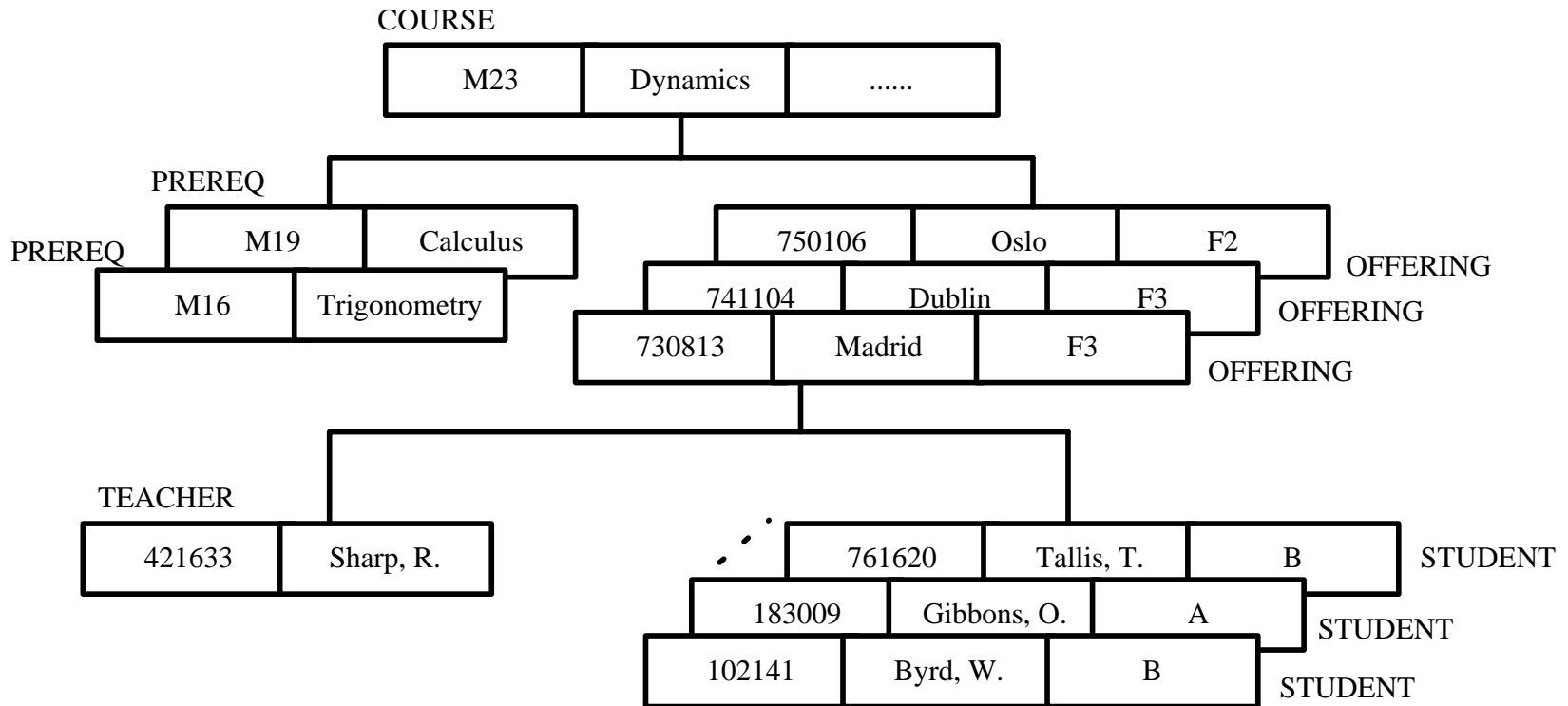


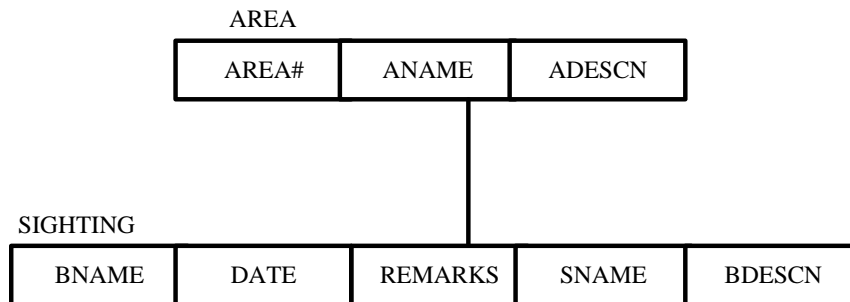
Fig. 2. Sample PDBR occurrence for the education database (**database instance**)

❖ **Note:** IMS is a **hierarchical database model**. It is similar to (but not exactly the same as) the **XML** data model.

# Many-to-many (m:m) relationships

- ❖ • Many-to-many relationships in hierarchical structure will contain **redundant data**
- IMS removes redundant data using **logical parent pointers**

# Many-to-many Relationships using logical parent pointers



**Note:** The same type of birds may appear in different areas, so the relationship between AREA and SIGHTING is a m:m relationship. The SNAME and BDESCN of a bird BNAME will be replicated under different areas.

Fig. 3. Required record structure for the survey database (**schema**).

**BNAME** – name of a bird. **SNAME** – scientific name of the bird.

To remove redundant data, we first create another database to store the information of birds as shown below:

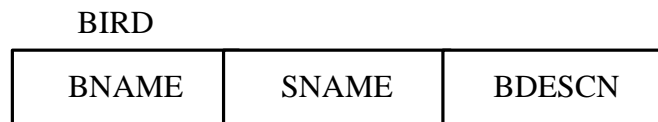


Fig. 4. Record structure of the bird database

# Many-to-many Relationships using logical parent pointers (cont.)

We then redesign the schema in Fig 3 to the below using **logical parent pointers**.

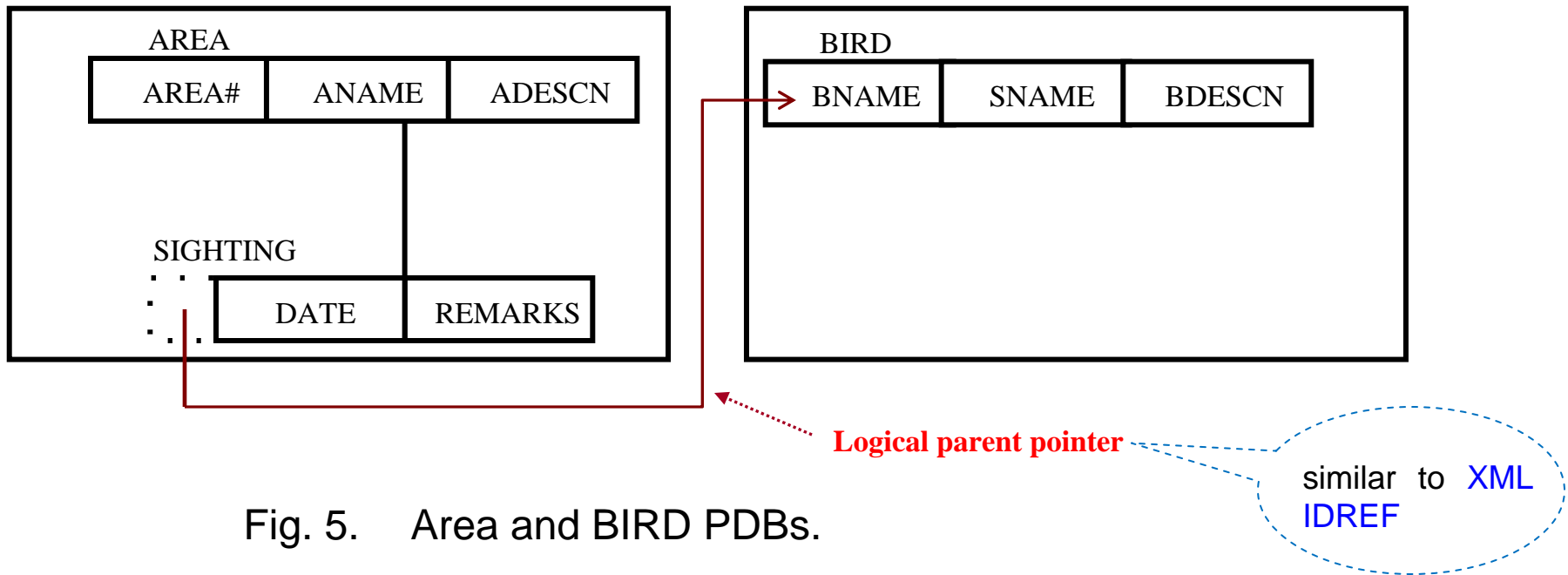


Fig. 5. Area and BIRD PDBs.

**Note:** DATE and REMARKS depend on AREA and SIGHTING

❖ **Note:** Logical parent pointer is similar to **XML IDREF**.

# Many-to-many Relationships using logical parent pointers (cont.)

We can then create a **logical database** (or **view**) of Fig 5, called SURVEY LDB which is the same as the original database.

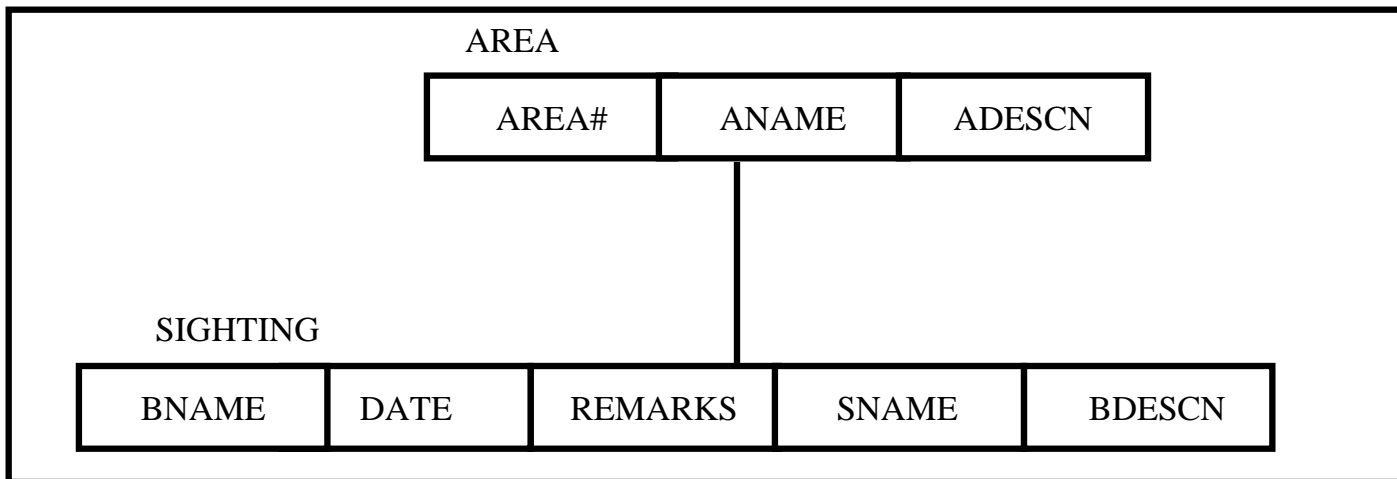


Fig. 6. The SURVEY LDB

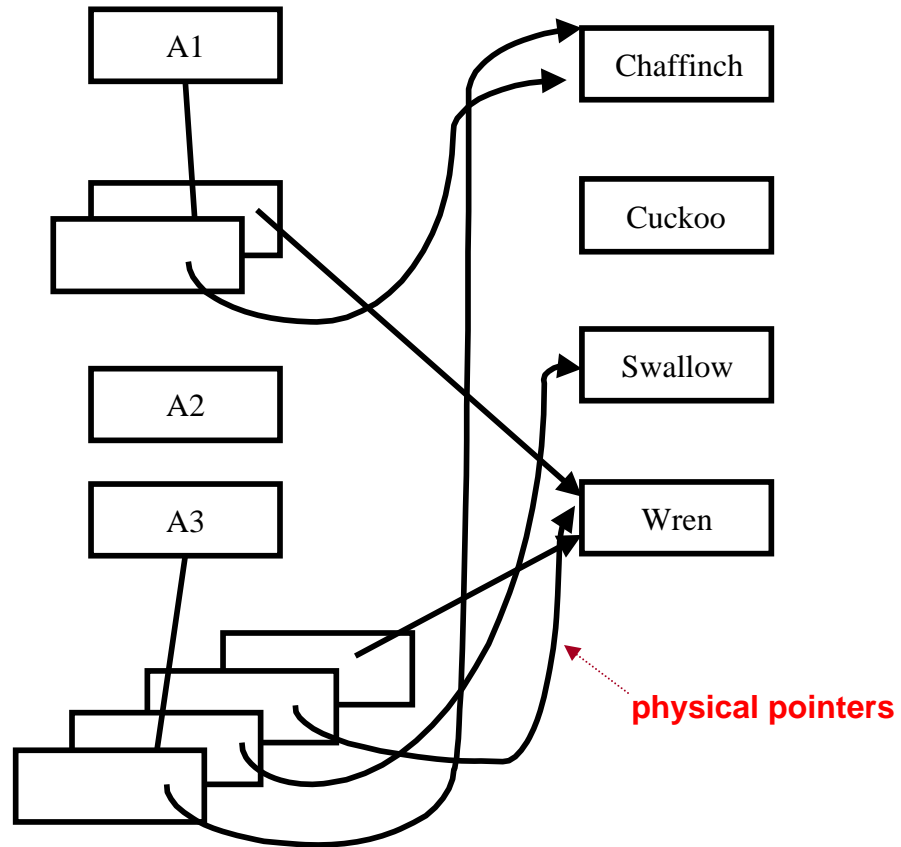


Fig. 7. Sample PDBs (AREA and BIRD)

The database can be **viewed** as:

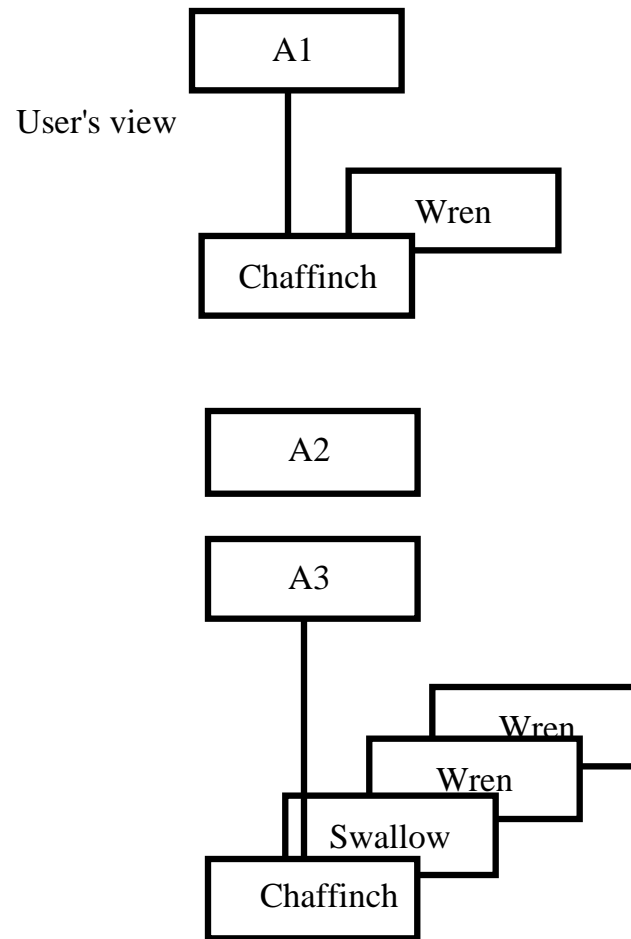


Fig. 8. Corresponding LDB (SURVEY)

# 2. Network Model

Network Model was proposed by **DBTG** (Database Task Group) in 1971.

Ref: CJ Dates' book. Software Product: **IDMS**

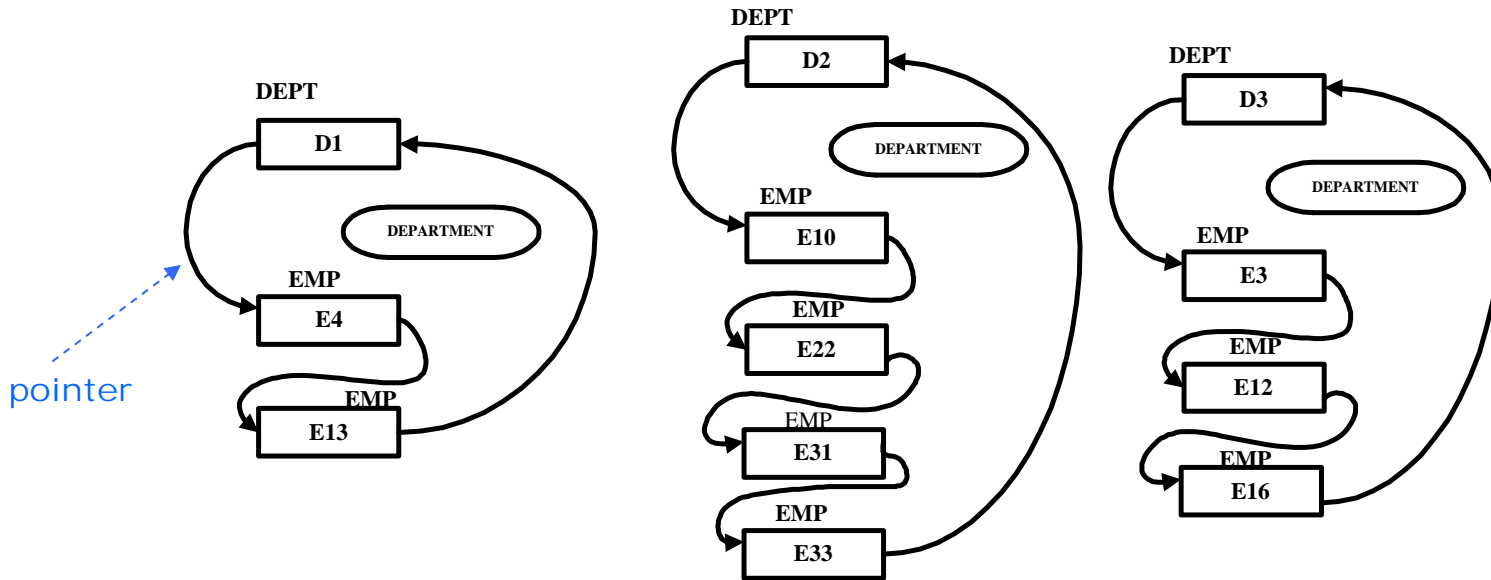


Fig. 9. A Department-employee database instance.

**Note:** Each employee only works for one department and a department may have many employees. The relationship between employee and department is a **many-to-one** relationships.

# Network Model Data Structure

(schema diagram)

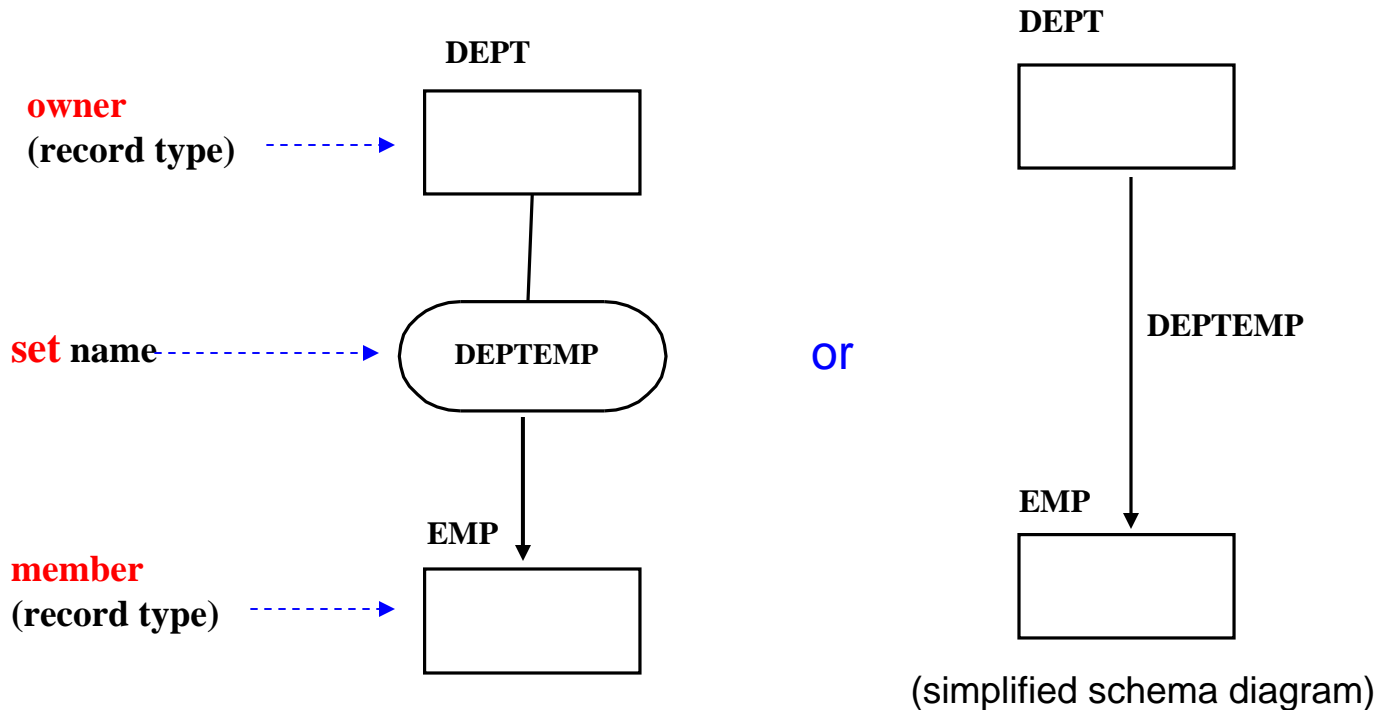


Fig. 10. Structure of the **set** DEPTEMP.

Each employee works for only **one** department

**Note:** A set is a **1:m relationship** from owner to member. E.g. a dept has many employees and each employee only works for one dept.

# A three level network example

a record type may be both a **owner** and **member** of two **set** types

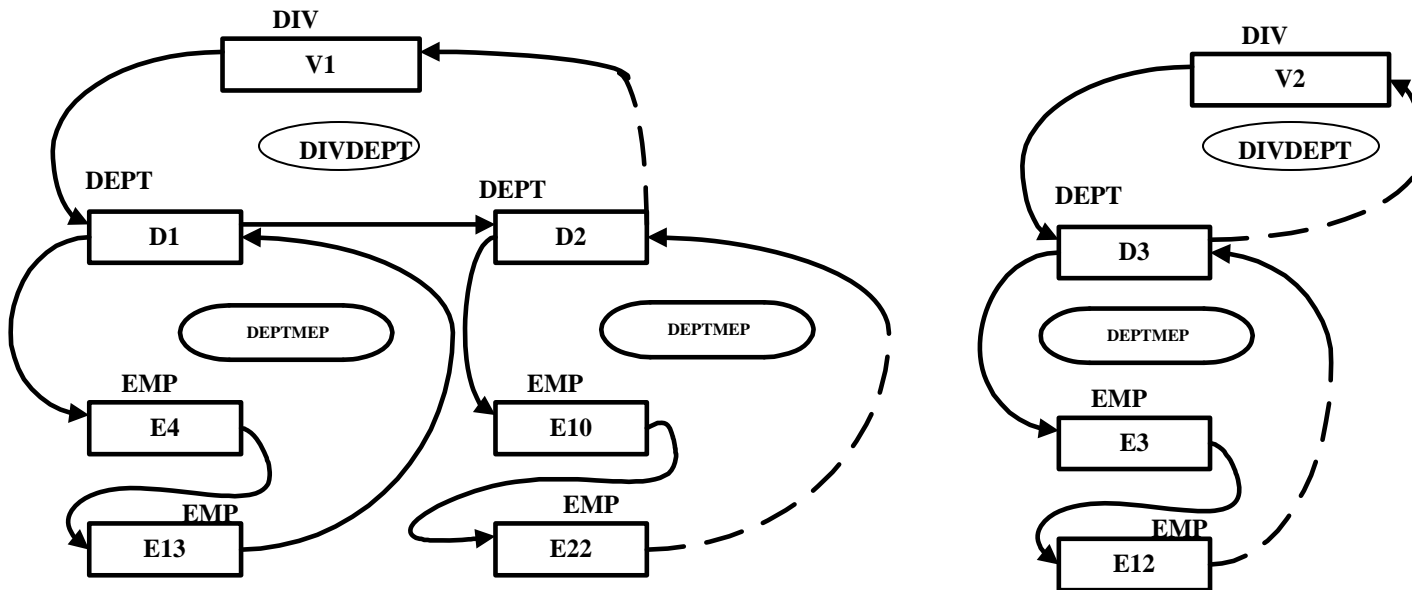


Fig. 11. A division-department-employee database instance.  
Each department belongs to one Division. There are 2 divisions.

# A three level network example (cont.)

A record type may be both a **owner** and **member** of two **set** types

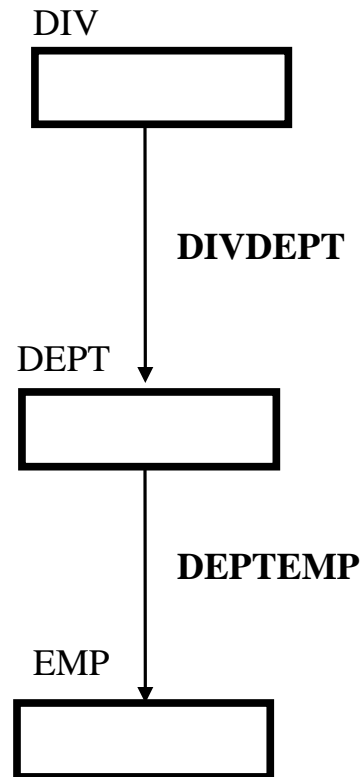


Fig. 12. Structure of the sets DIVDEPT and DEPTEMP

# One owner with two members

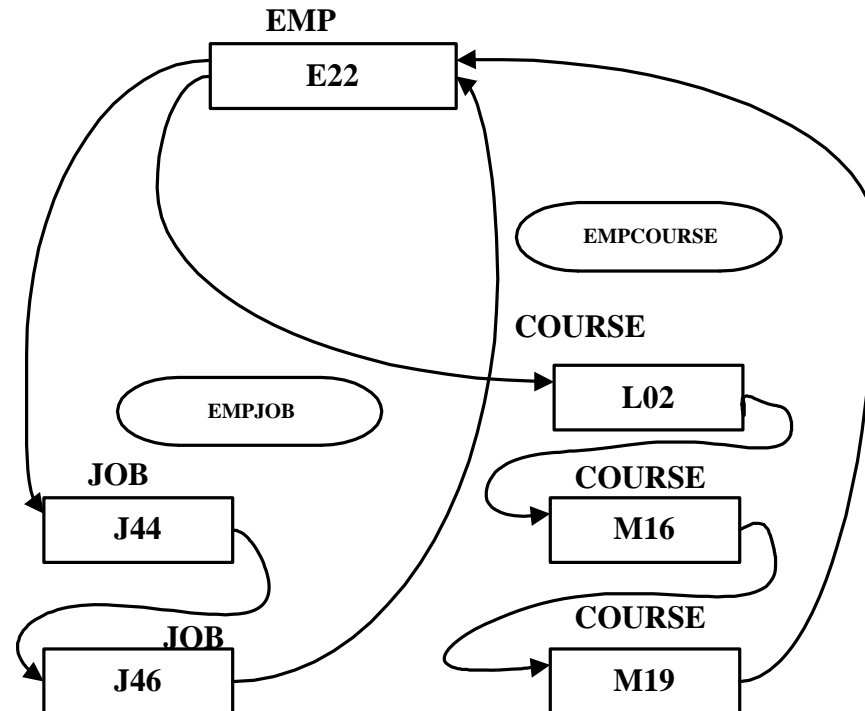


Fig. 13. An employee-history database instance

# One owner with two members (cont.)

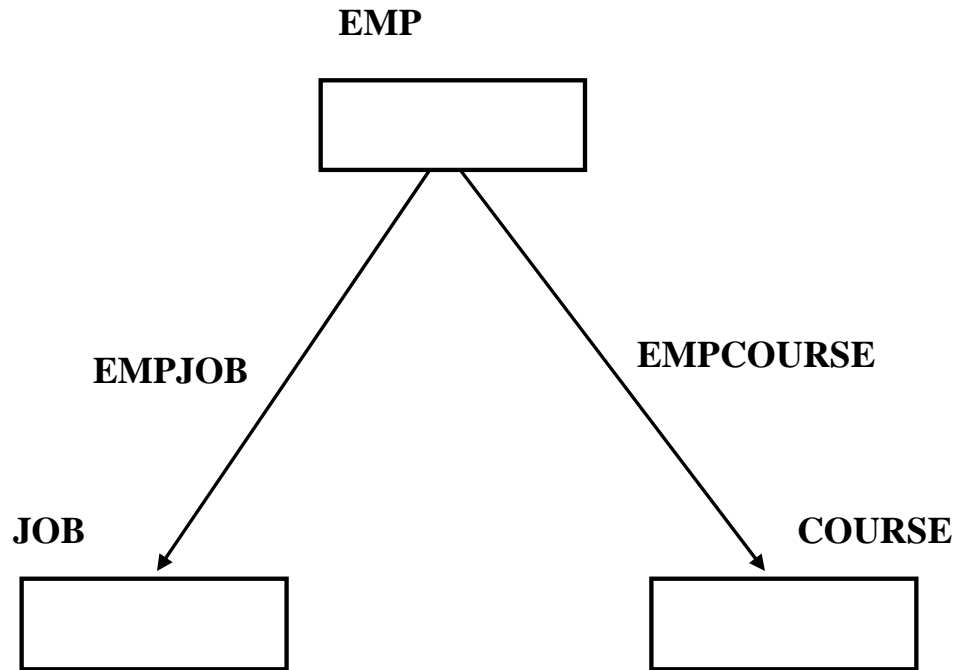


Fig. 14. Structure of the sets EMPJOB and EMPCOURSE

# Many-to-many relationship

E.g. part and supplier relationship.

- ❖ **Note:** Network model **cannot** represent **m:m relationships** directly. A m:m relationship can be simulated by two 1:m relationships and a **dummy record type** (e.g. SP in Fig 15). Not a very nice solution!

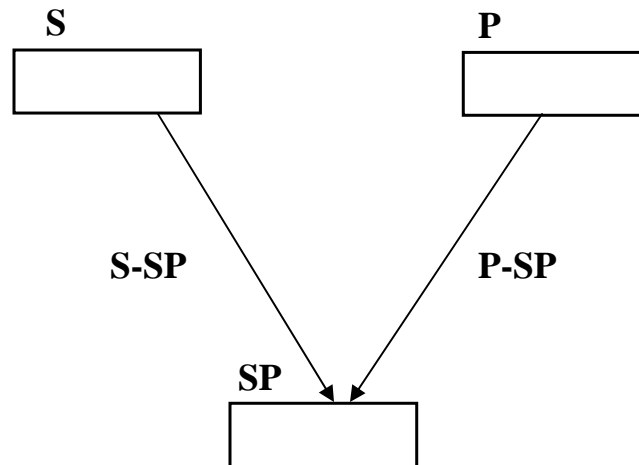


Fig. 15. Structure of the sets S-SP and P-SP.

# Many-to-many relationship (cont.)

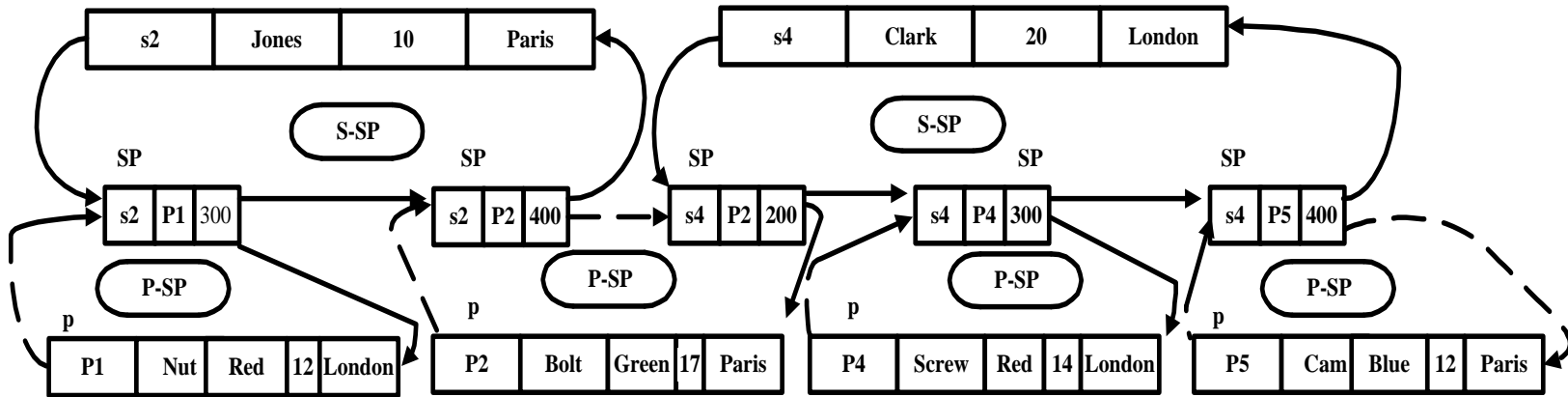


Fig. 16. A suppliers-and-parts database instance

❖ **Q:** How to represent 1:1 relationships? N-ary relationships?