CS4221: Database Schema Design

A Brief Introduction on Hierarchical and Network Data Models

Database Models

- **File system**
  - field, record, fixed length record
  - direct access file
  - sequential access file
  - indexed sequential file
  - not a database model
- **Hierarchical** Model (IMS)
  - fixed length record (segment)
  - tree structure
  - storage structures: HSAM, HISAM, HDAM, HIDAM
    where H means hierarchical
- **Network** Model (IDMS)
  - field, fixed length record
  - owner, member, set (circular linked list)

- **Relational** Model
  - attribute (field), relation (table), fixed length
  - functional dependency, multivalued dependency
  - normal forms, normalization
- **Nested Relational** Model
  - not even in first normal form
  - an attribute can be a relation/table
- **Entity-Relationship Approach**
  - entity type, relationship type, attribute
  - ER Diagram
  - for conceptual database design
- **Object-Oriented (OO)** Data Model
  - influenced by object-oriented programming languages
  - object, object ID, class hierarchy, inheritance, method, encapsulation, polymorphism
- **Object Relational** Data Model

- **Deductive and Object-Oriented (DOOD)** Database
  - Datalog (Similar to Prolog)
  - Horn clauses as deductive rules
    - define derived relation

      <derived relation> if <conditions>

      e.g. ancestor(x,y) ← parent(x,y)

      ancestor(x,y) ← parent(x,z), ancestor(z,y)
    - similar to view in RDB, but more powerful
  - extended with negation in the body of a rule

    e.g. married(x) ← spouse(x,y)

    married(y) ← spouse(x,y)

    bachelor(x) ← male(x), not(married(x))
  - recursive rules
  - transitive closure & computing
- **Semi-structured** Data Model (XML data)
  - similar to hierarchical model, tree model, structure not rigid

1. Hierarchical Model

IMS (Information Management System) Data Model

(IBM product) **Ref**: CJ Date's book

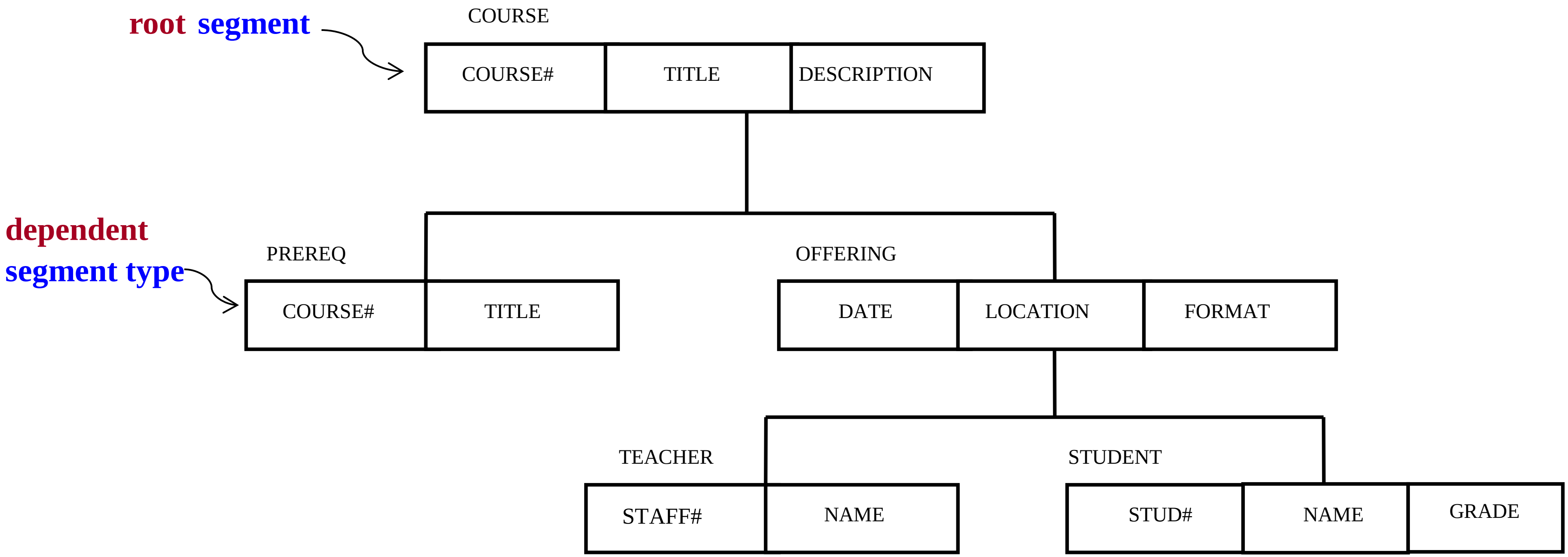
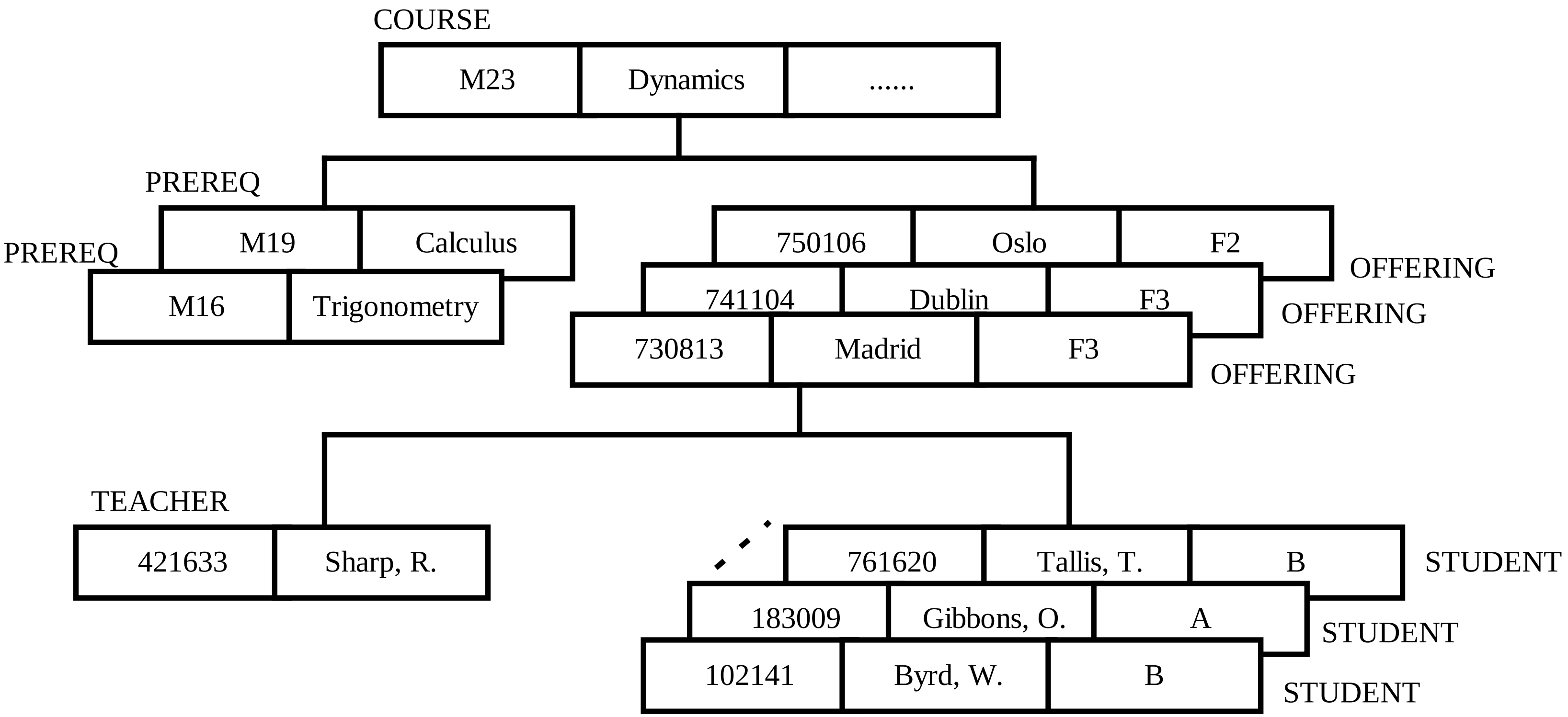


Fig. 1. PDBR (physical database record) type for the education database (schema)

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



COURSE

M23 | Dynamics | ......

PREREQ

M19 | Calculus

PREREQ

M16 | Trigonometry

OFFERING

750106 | Oslo | F2

OFFERING

741104 | Dublin | F3

OFFERING

730813 | Madrid | F3

TEACHER

421633 | Sharp, R.

STUDENT

761620 | Tallis, T. | B

STUDENT

183009 | Gibbons, O. | A

STUDENT

102141 | Byrd, W. | B

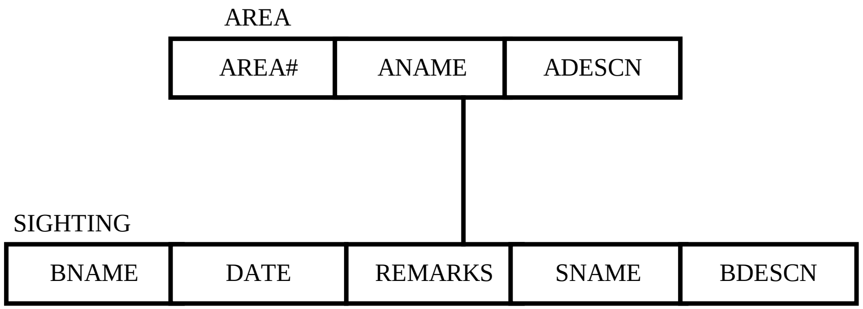
Fig. 2. Sample PDBR occurrence for the education database (database instance)

- **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



AREA

AREA#	ANAME	ADESCN

SIGHTING

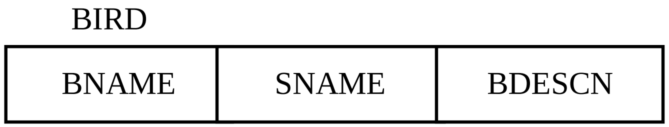
BNAME	DATE	REMARKS	SNAME	BDESCN

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:



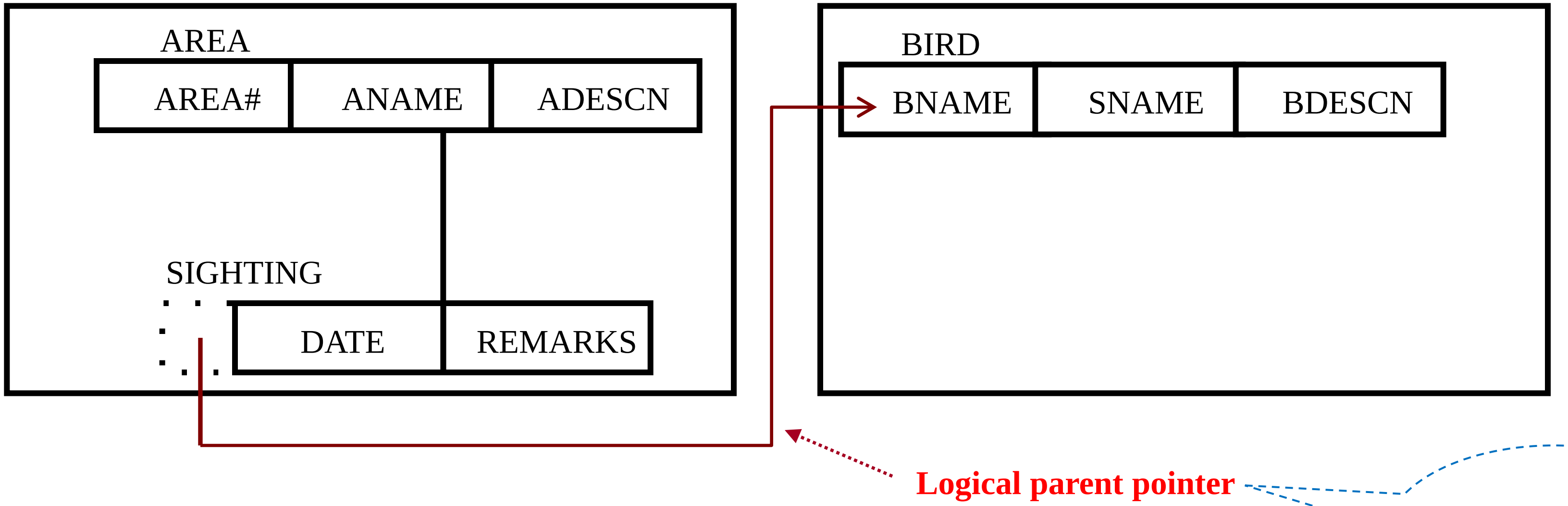
BIRD

BNAME	SNAME	BDESCN

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.



similar to XML IDREF

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.

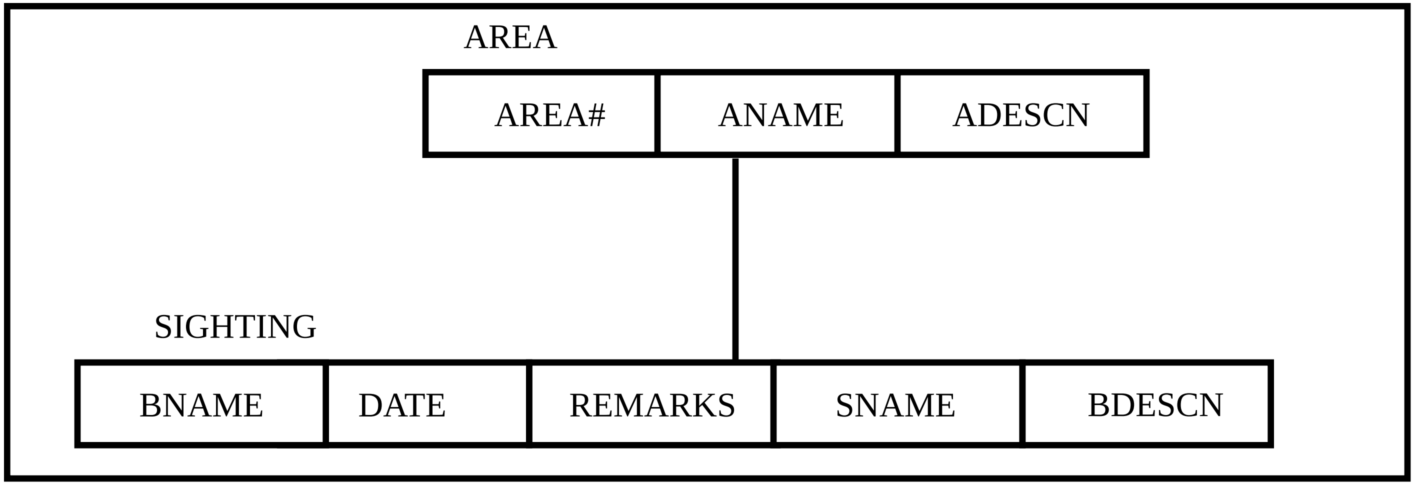


Fig. 6. The SURVEY LDB

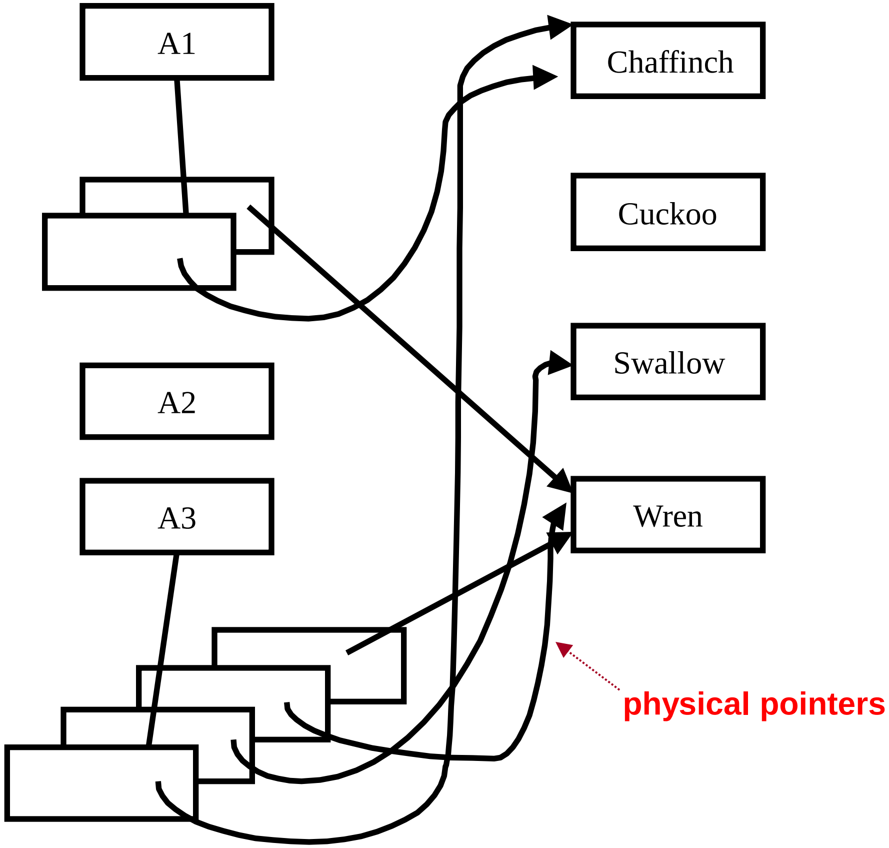


Fig. 7. Sample PDBs (AREA and BIRD)

The database can be viewed as:

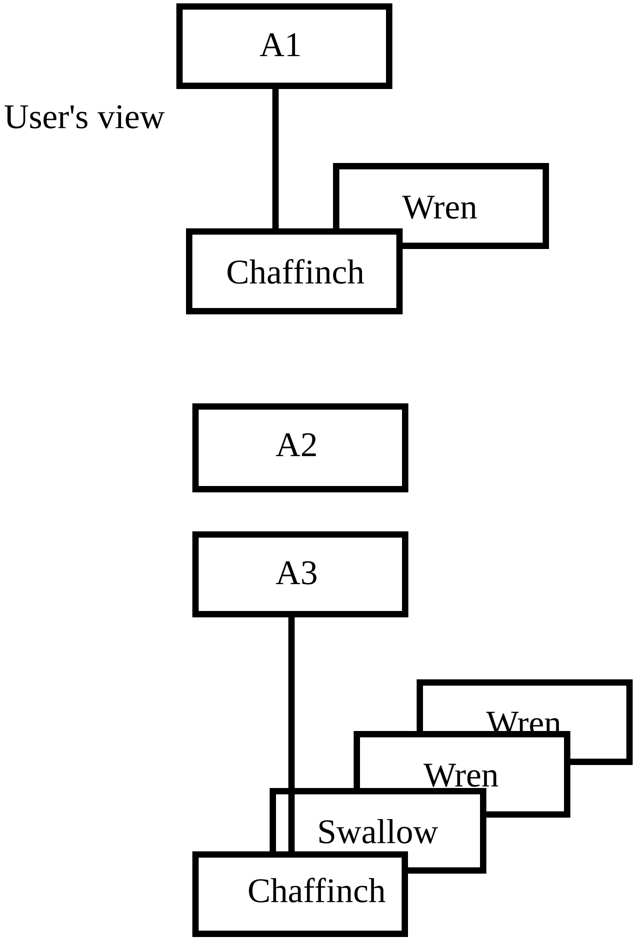


Fig. 8. Corresponding LDB (SURVEY)

2. Network Model

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

Ref: CJ Dates' book. Software Product: IDMS

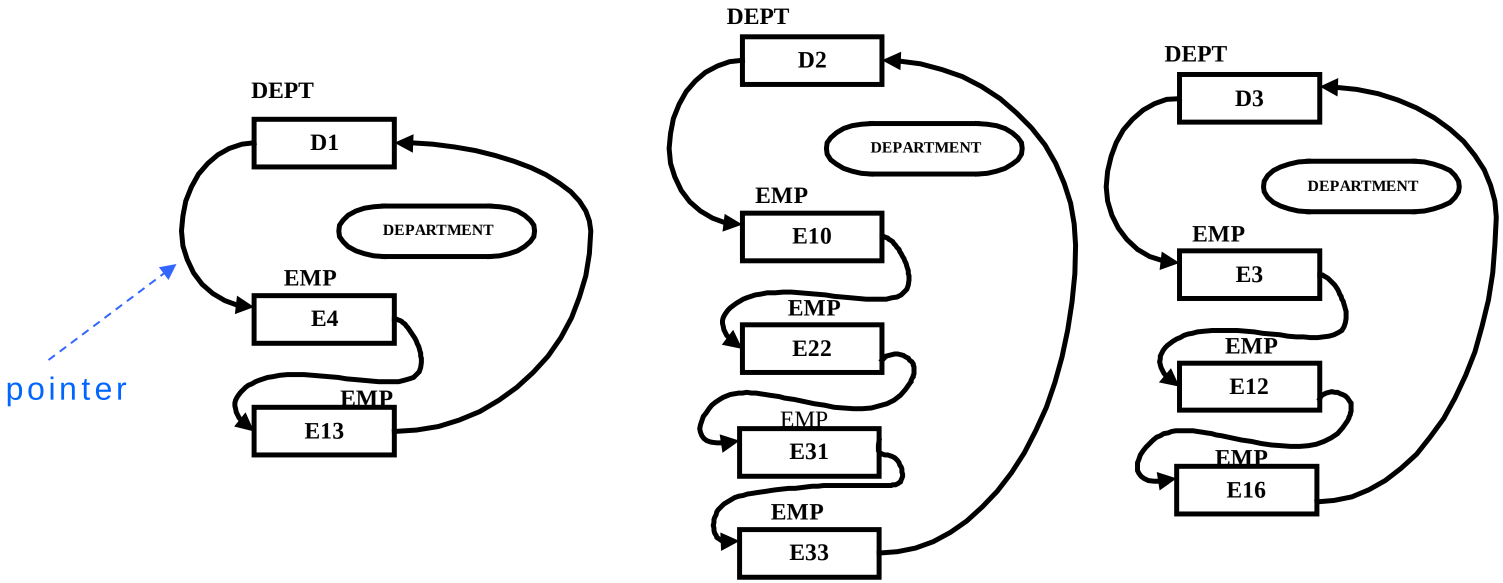
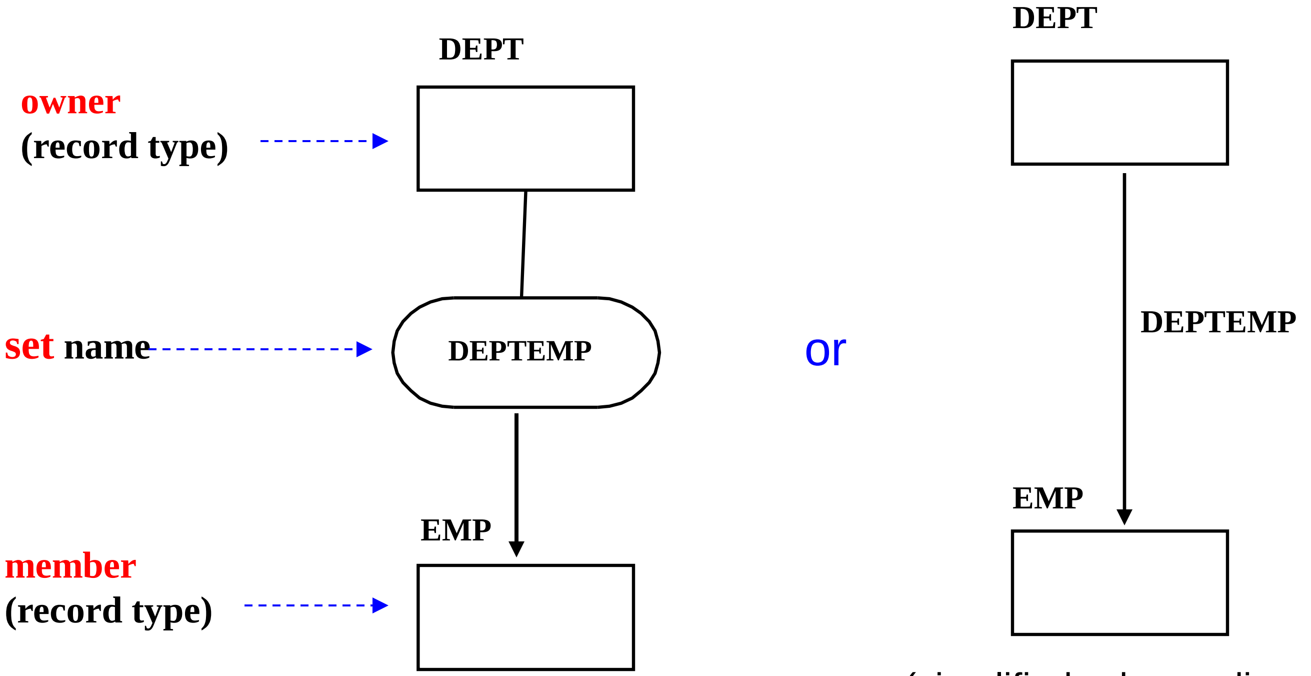


Fig. 9. A Department-employee database instance.

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)



DEPT

owner
(record type)

set name

DEPTEMP

EMP

member
(record type)

or

DEPT

DEPTEMP

EMP

(simplified 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**

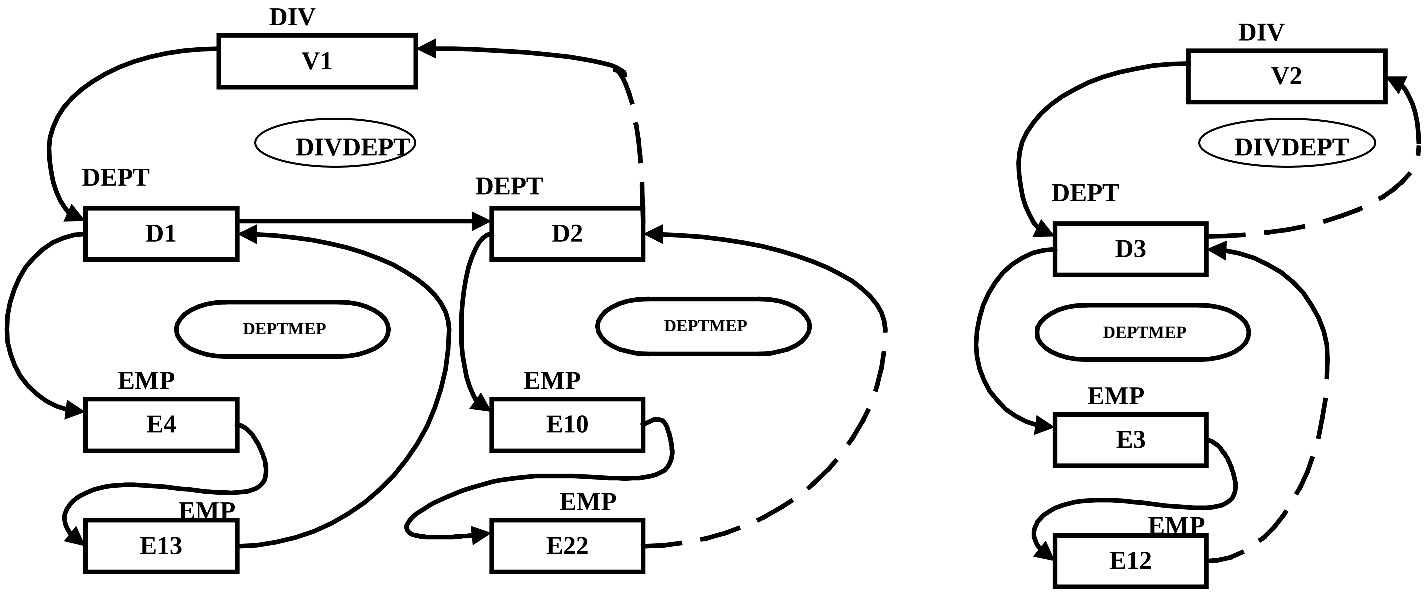


Fig. 11. A division-department-employee database instance.

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**

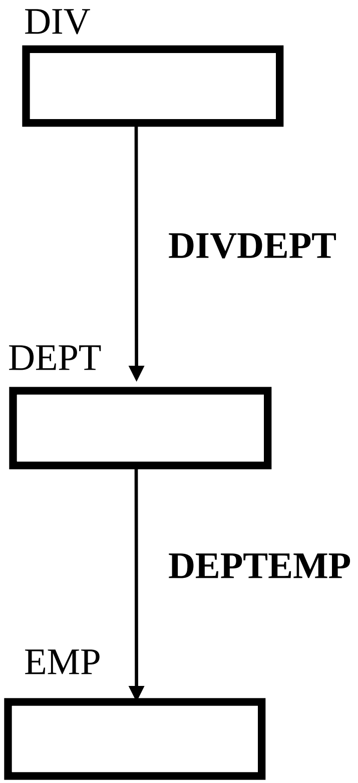


Fig. 12. Structure of the sets DIVDEPT and DEPTEMP

One owner with two members

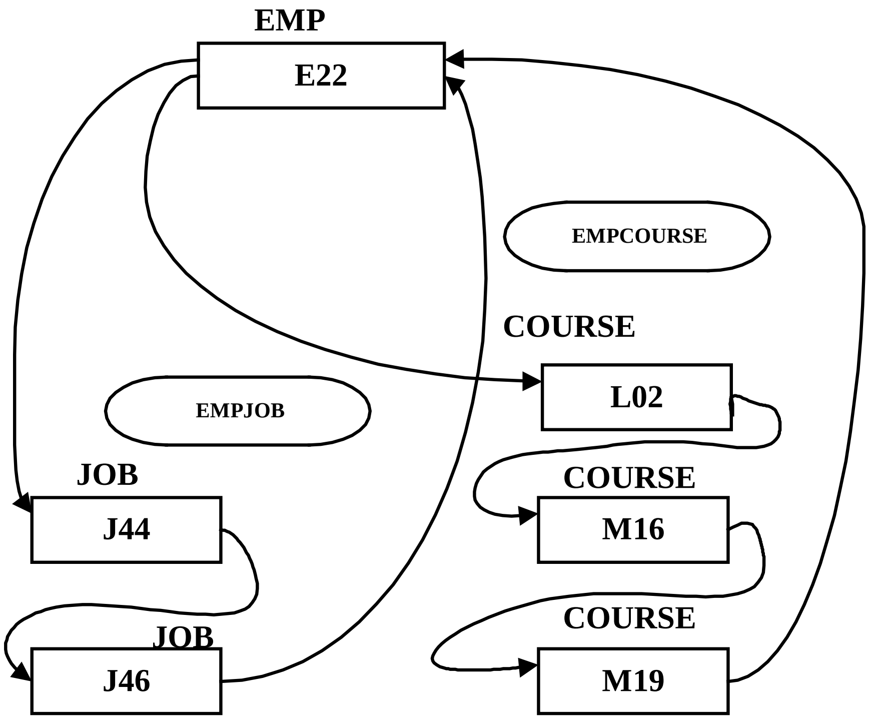


Fig. 13. An employee-history database instance

One owner with two members (cont.)

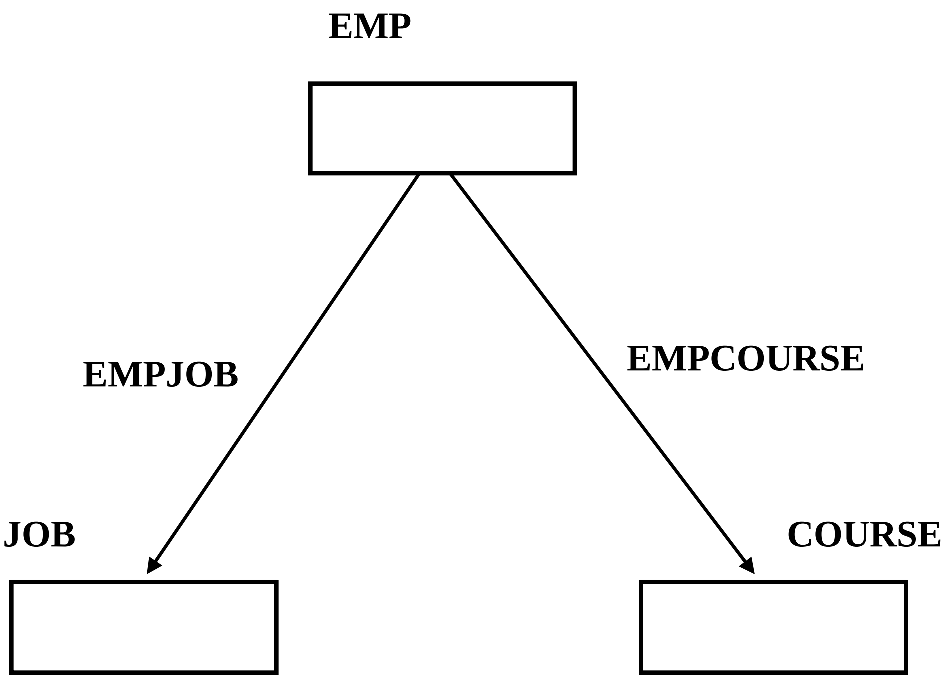


Fig. 14. Structure of the sets EMPJOB and EMPCOURSE

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!

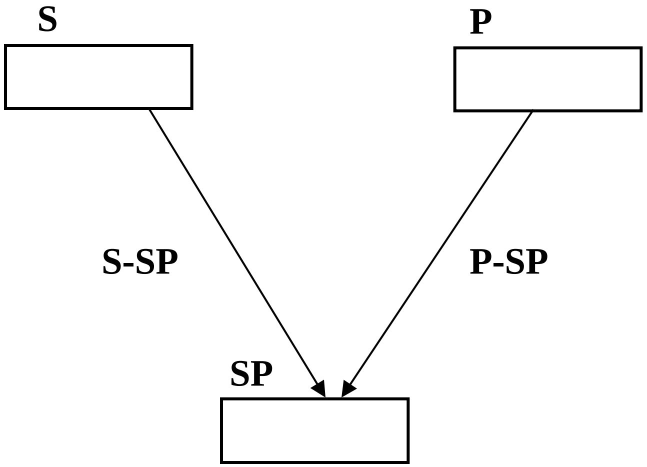
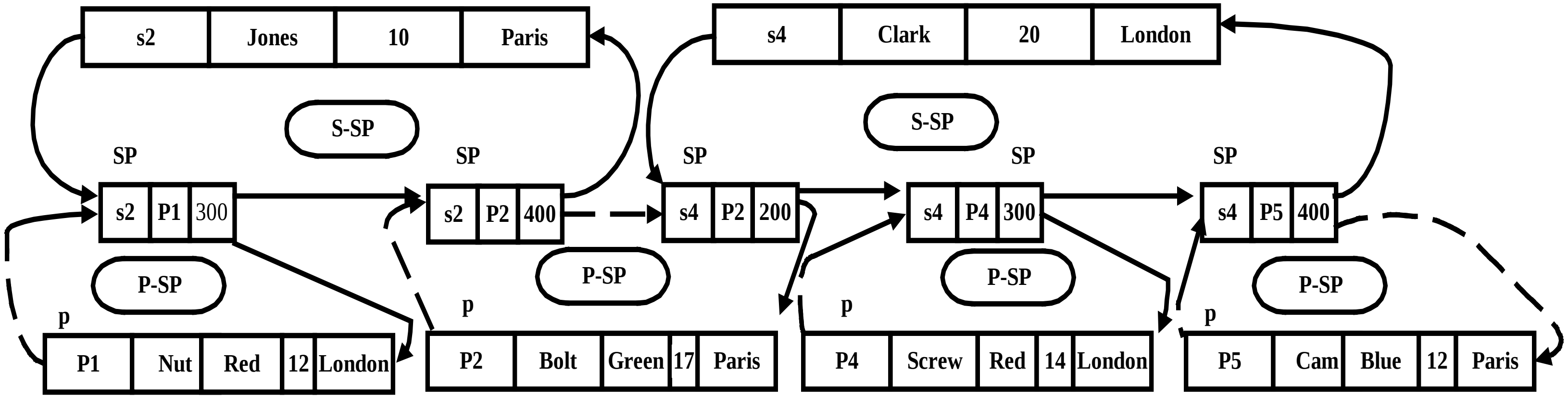


Fig. 15. Structure of the sets S-SP and P-SP.

Many-to-many relationship (cont.)



s2 | Jones | 10 | Paris

s4 | Clark | 20 | London

S-SP

S-SP

SP

SP

SP

SP

SP

s2 | P1 | 300

s2 | P2 | 400

s4 | P2 | 200

s4 | P4 | 300

s4 | P5 | 400

P-SP

P-SP

P-SP

P-SP

p

p

p

p

P1 | Nut | Red | 12 | London

P2 | Bolt | Green | 17 | Paris

P4 | Screw | Red | 14 | London

P5 | Cam | Blue | 12 | Paris

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

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