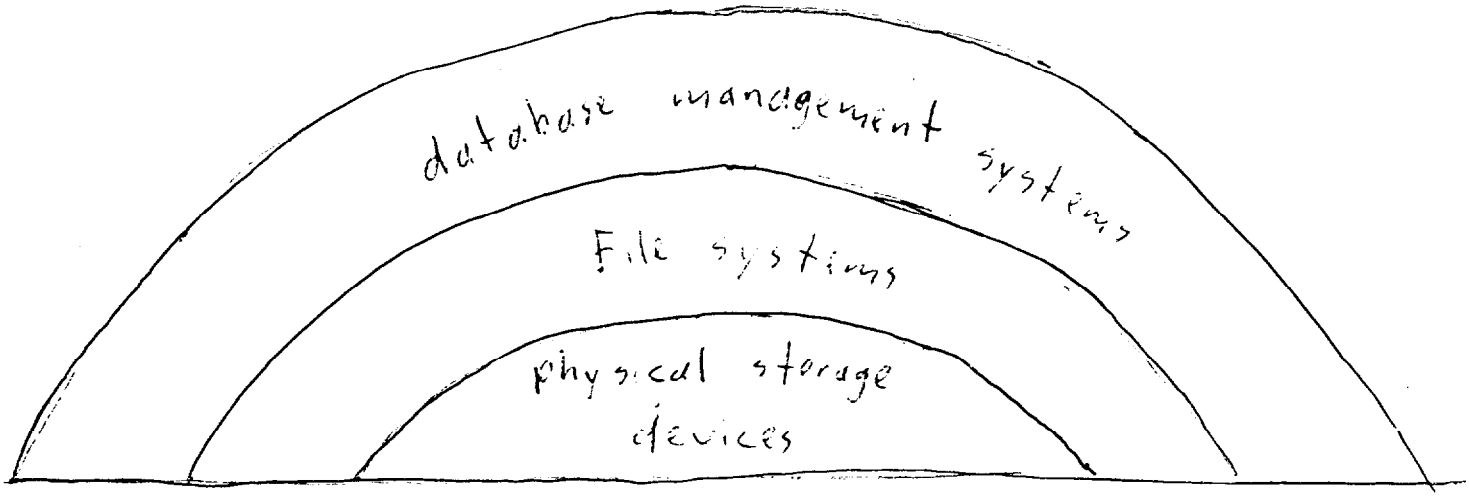
Lecture 22: Database Management Systems

Last few days: Multi-key Files
- Secondary indices
- Inverted Lists
- Threaded Files
- Query Processing

Today: Database Management Systems
- Database processing
- Data models
- Entity-Relationship diagrams

Smith & Barnes ch 10

Overview of a Database Management System (DBMS)



Each layer hides details of the lower level.

File Processing vs Database Processing

- Much of what we have seen so far is File processing.
- As the number of users and applications in an organization grows, file processing runs into new difficulties.
- To address these difficulties, file processing evolved into database processing
- The difficulties are
  - data independence
  - data sharing
  - data integrity

Data Independence

- In File processing, programs are written to manipulate data stored in files with a specific physical organization.

- If this organization changes programs might not work any longer.

- Typical changes include:
  - Add new fields to the records of a file.
  - Change a direct file to an indexed file.
  - Change the sort key of a file.
  - Add & remove secondary indices.

- Data Independence: Changes in file organization should not require changes to application programs

Data Sharing

- Typically, many users in an organization need access to the same data concurrently.

- They may read or modify the data and/or run application programs

- These concurrent operations must not interact in unexpected ways.

- Typical problems: What happens if one user tries to update a file while another user is reading it?

Data Integrity

- A File system may keep multiple copies of data (For speed & reliability).

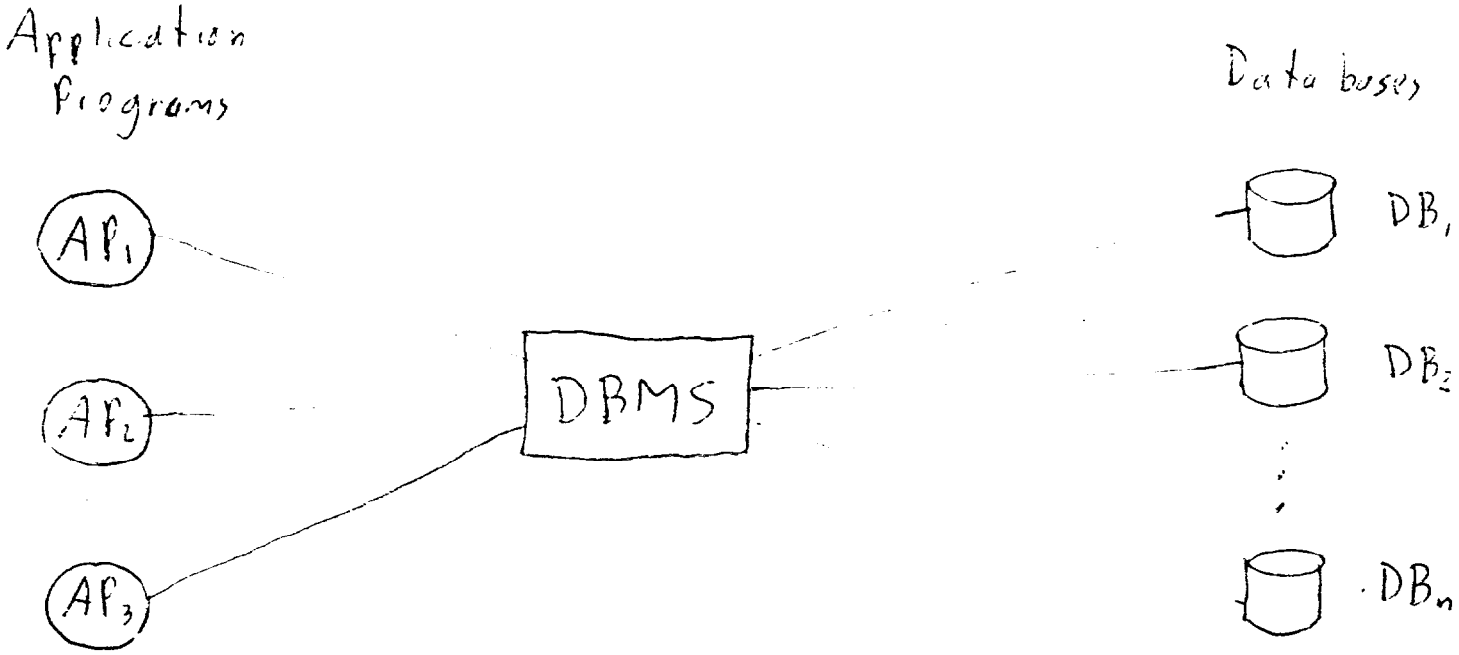
- Also, data in one file may be related to data in other files in subtle ways.

- How do we ensure consistency of the data?

- ie, whenever data is updated, related data (including copies) may also have to be updated.

- A Database Management System (DBMS) solves these problems.

Database Processing



DBMS: A (large) software module that maintains the data of an organization and mediates between the data and the application programs

- Each application program asks the DBMS for data
- DBMS figures out the best way to locate the data.
- The applications are coded without knowing the physical organization of the data.

Data Models

- An application program must describe to the DBMS what data it wants.

- This description is in terms of a high-level conceptual view of the data, called a *data model*.

- A data model provides "building blocks" a simple, logical description of the data

- Application programs view data in terms of a specific data model.

Data Models

- There are three principal, commercial data models:
  - hierarchical
  - network

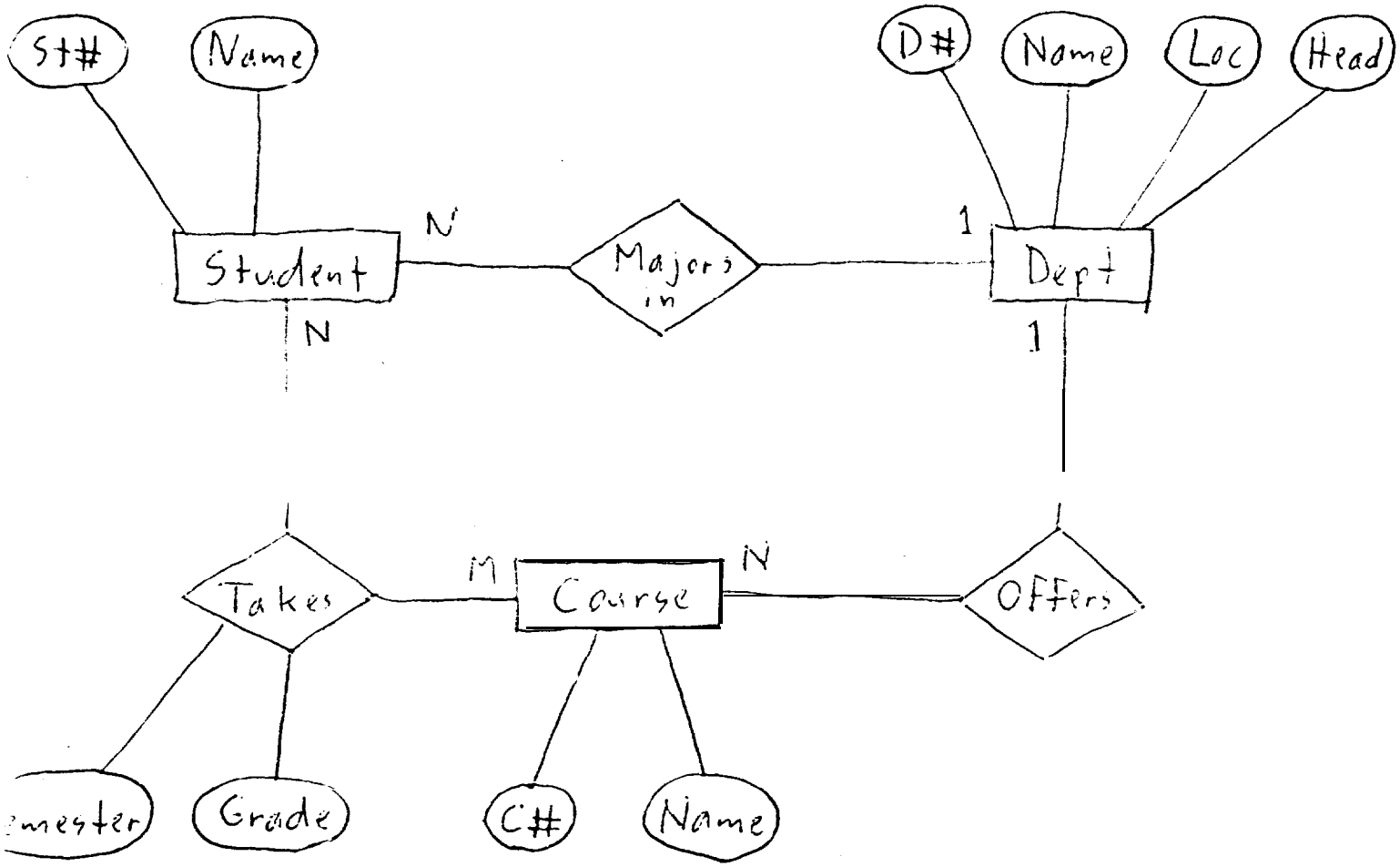
  } old fashioned, but still important.
  - relational

  } used by most new systems.

- Database Design: before describing a database with one of these models, an even more abstract description is usually prepared first, using Entity Relationship diagrams (ER diagrams)

- Simple idea: Any database consists of
  - Entities, which have attributes.
  - Relationships between entities.

Entity Relationship Diagram : Example



— Attributes

— Entities

— Relationships

The Relationships:

- A student majors in a dept.
- A student takes a course (in a given semester, & receiving a given grade).
- A dept offers courses.

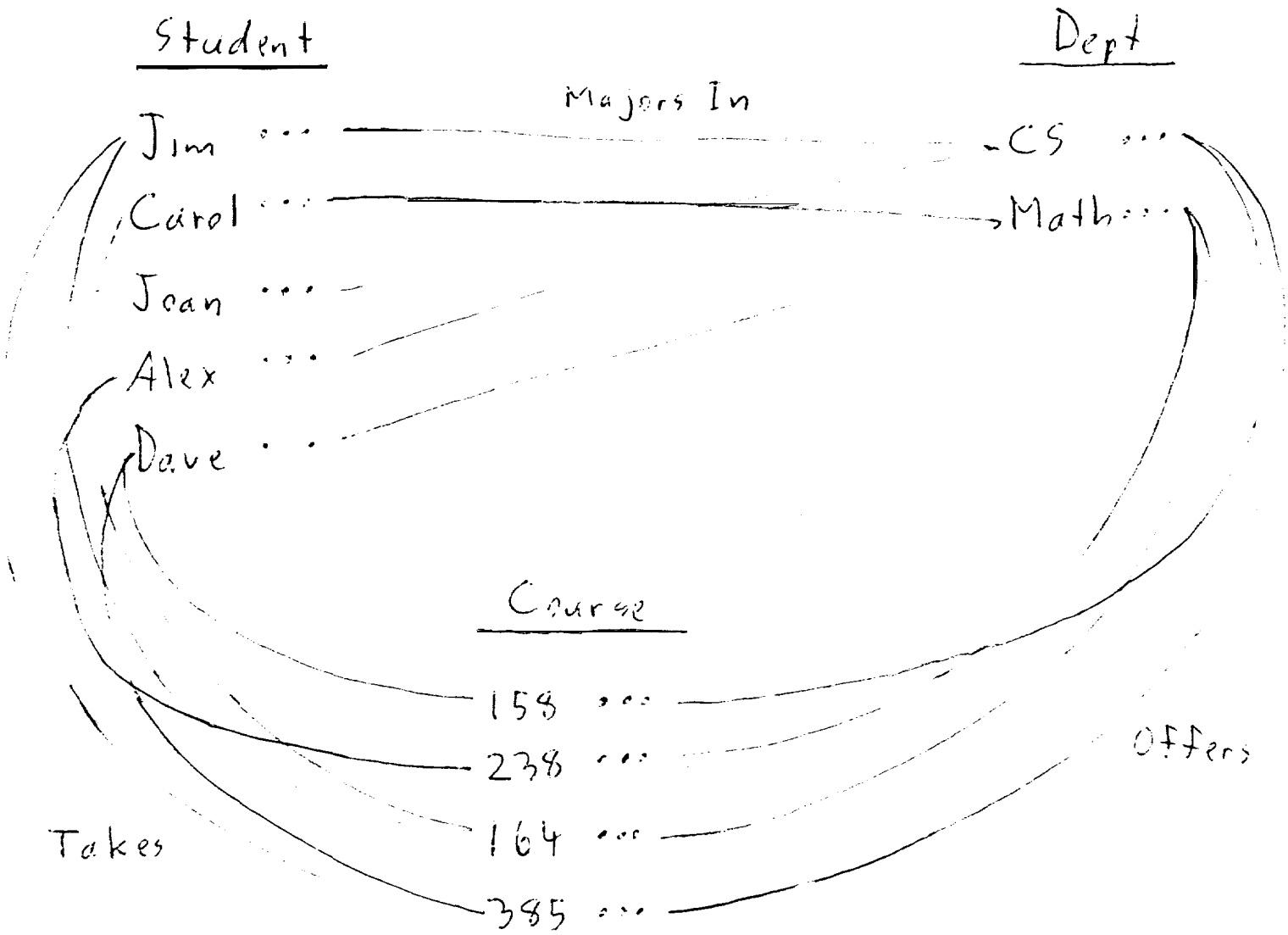
The Entities:

- A student has attributes St#, Name, Address
- A dept has attributes D#, Name, Loc, Head.
- A course has attributes C#, Name.

Note:

- An ER diagram contains no data
- It describes the structure of the data,
- An instance of an ER diagram contains data.

Database Instance: Example



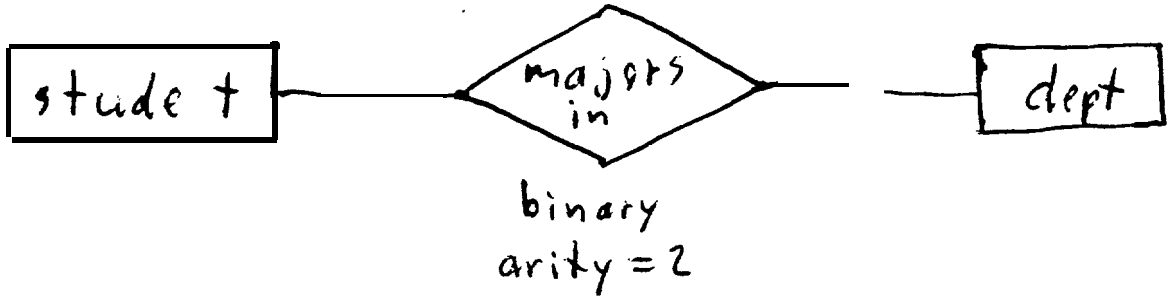
- Each relationship in an ER diagram becomes a set of links between specific entities
- Note: This is a conceptual so far.

Properties of Relationships

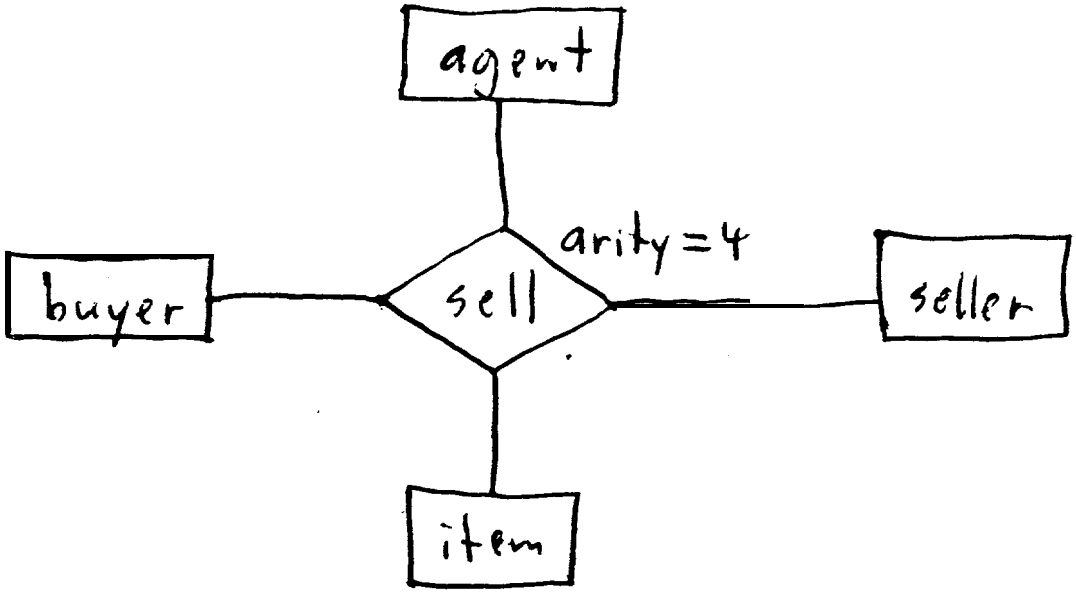
The implementation of a relationship depends on its properties.

Property 1: Arity, the number of entities involved.

eg.



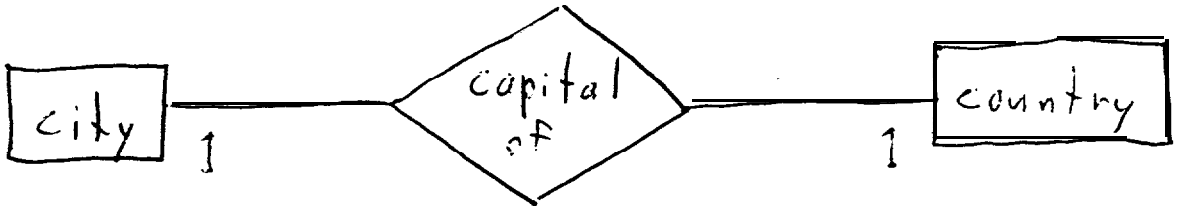
eg.



Property 2: Functionality

(2a) one-to-one (1:1)

eg.



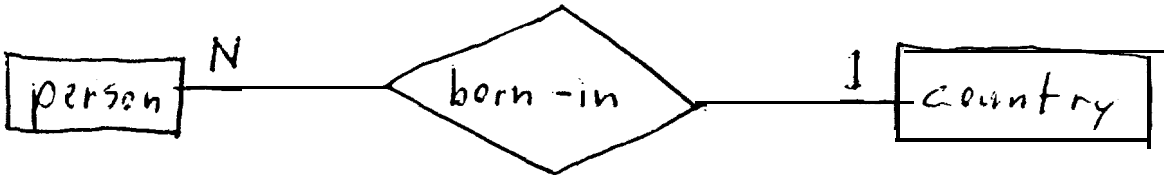
"capital of" is one-to-one since it associates each country with only one city and vice versa.

eg.

Athens — Greece
Ottawa — Canada
Paris — France
London — Britain

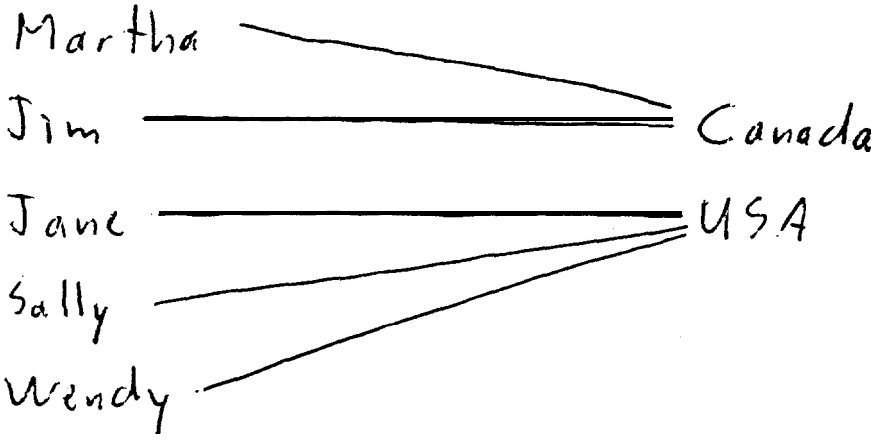
(2b) many-to-one (N:1)

eg.



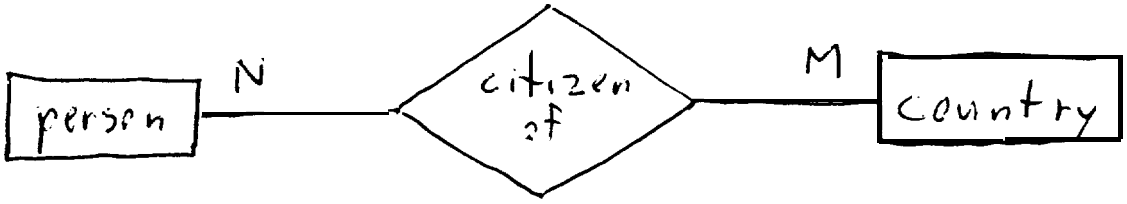
Each person is born in only one country but many people may be born in the same country.

eg.



(2c) many-to-many (N:M)

eg.



A person may be a citizen of many countries and a country may have many citizens.

eg.