Lab Content:

Introduction to database
- What is database
- Differences between database and files
- Entity-Relationships (ER) model
- Case Study for Students

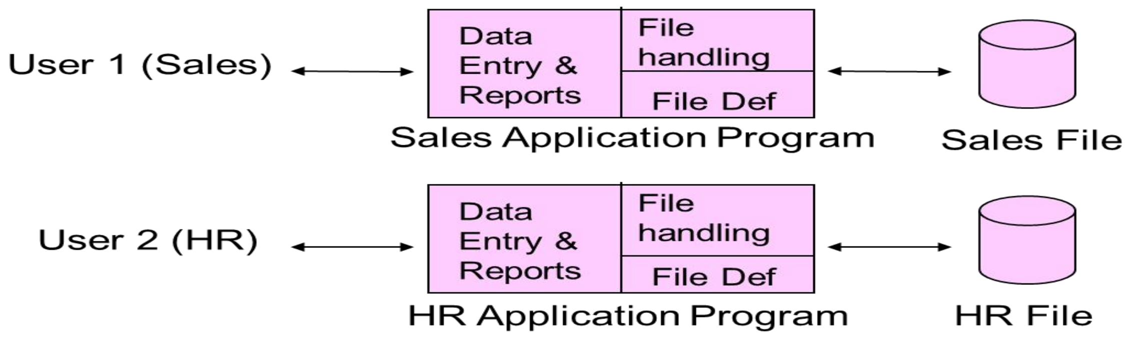
What is database?

A **database (DB)** is a collection of a logically related *persistent* data, designed to meet the information needs of an organization. Can be generated & maintained manually or automatically.

Data is what you store in *database.*
Information is what you retrieve from a database.

Differences between database and files

File-Based Systems (FBS)
FBS is a collection of application programs that perform tasks (e.g. reports) where each program defines and manages its own data.

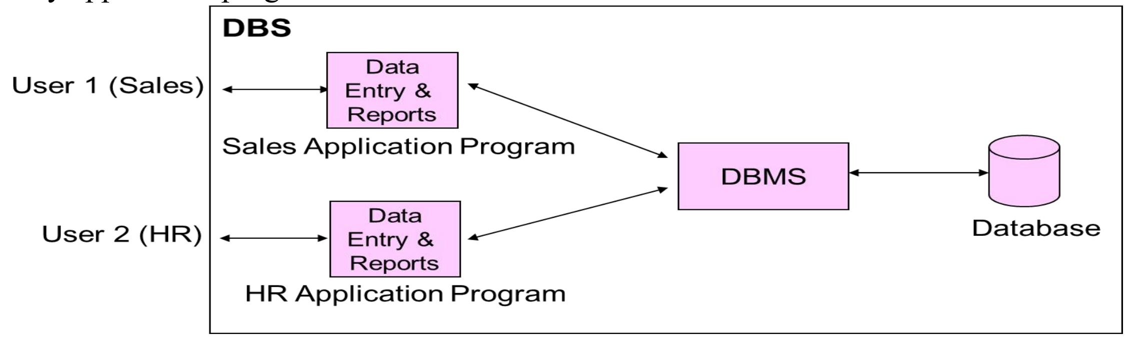


Major limitations of FBS:
- Data Redundancy and Inconsistency.
- Data isolation and separation.
- Data dependence.
- Incompatible file formats.
- Fixed Queries/Proliferation of application programs.

These limitations of the FBS approach attributed to two factors:
- The definition of data is embedded in the application programs, rather than being stored separately and independently.
- There is no control over the access & manipulation of data beyond the imposed by the application program.

Database Systems (DBS)

DBS is a single large repository of data, defined once and managed using DBMS while used by many application programs.



<u>Major characteristics of DBS:</u>

- Self-describing nature of a database system.
- Program-data independence.
- Sharing of data and multi-user transactions processing (guarantee *Concurrency Control*).
- Support of multiple views of the data.

<u>Disadvantages of DBS:</u>

- Complexity.
- Size.
- Cost (DBMS, Hardware, Staff, Training).
- Performance.
- Higher impact of a failure.

<u>Entity-Relationships (ER) model</u>

Entity-Relationship Model (E/R)

A picture is worth a thousand words

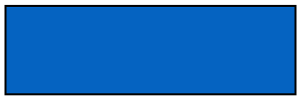
The **Entity-Relationship model** (ER) is a high-level description of the structure of the DB.

The **Entity-Relationship Diagram** (ERD) is a graphical model for representing the conceptual model for the data.

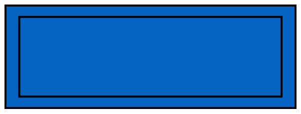
A E/R models the DB using three element types:

- Entities
- Attributes
- Relationships

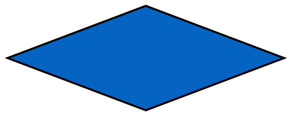
ERD notations



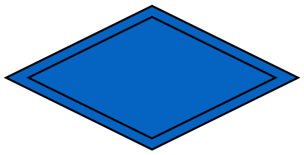
ENTITY



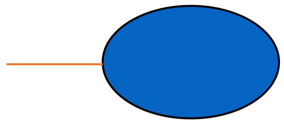
WEAK ENTITY



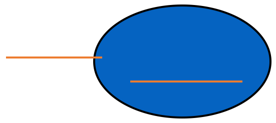
RELATIONSHIP



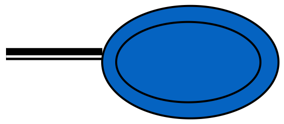
IDENTIFYING RELATIONSHIP



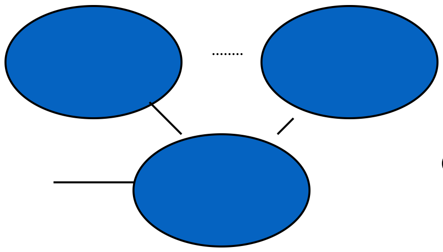
ATTRIBUTE



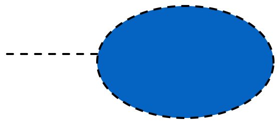
KEY ATTRIBUTE



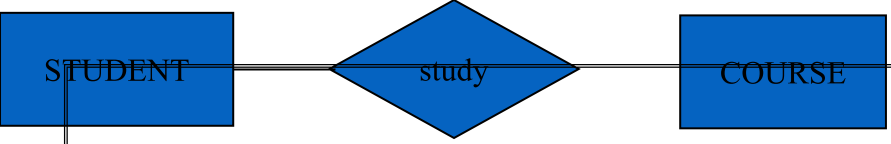
MULTI-VALUED

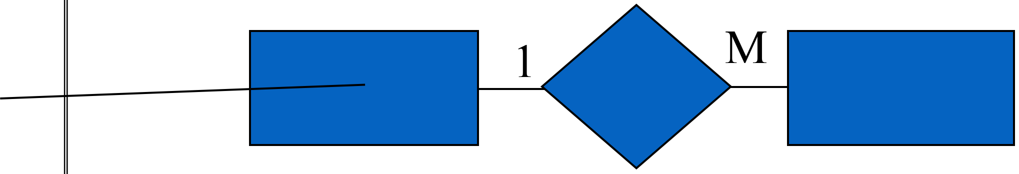


COMPOSITE

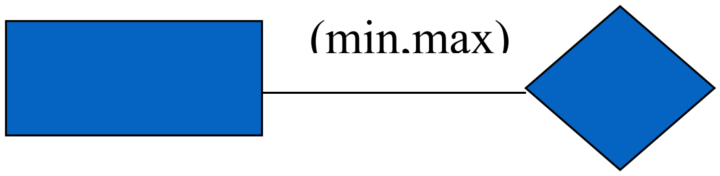


DERIVED





CARDINALITY RATIO



PARTICIPATIO CONSTRAINTS

Entities & Entity Type

Entity is an object that exists and is distinguishable from other objects.

e.g. person, company, course, university

Entity Type is a set of entities of the same type that share the same properties.

e.g. set of all persons, companies, trees, courses

STUDENT
COURSE

Relationships & RelationshipTypes

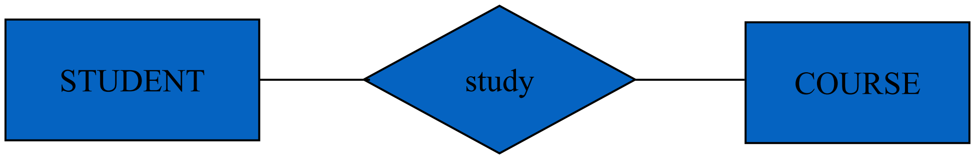
- A **relationship** associates 2 or more entities.
- A **relationship type** is a set of associations between entity types.

Degree of Relationship Type

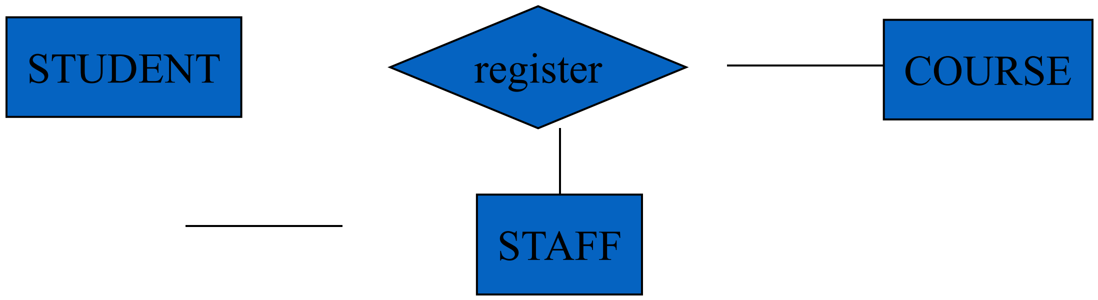
Degree of relationship refers to number of participating entity types in a relationship.

- A relationship of degree two (2 entity types) are **binary.**
- A relationship of degree three (3 entity types) are **ternary.**

A relationship of degree two (2 entity types) are **binary.**

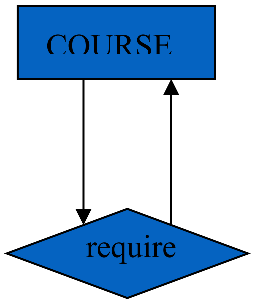


A relationship of degree three (3 entity types) are **ternary.**
e.g. registration of a student in a course by a staff.



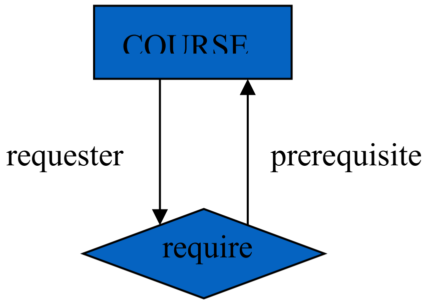
Recursive Relationship

Recursive relationship is a relationship type where the same entity type participates more than once in a different role. It is a **unary relationship.**

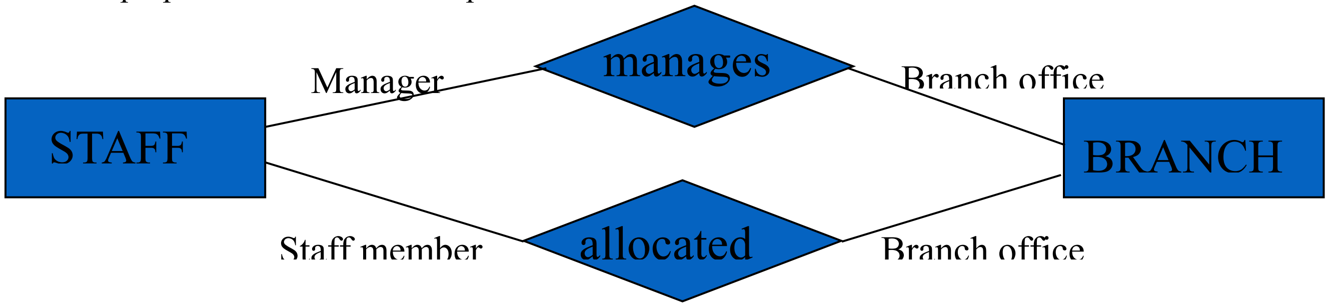


Roles

Role indicates the purpose that each participating entity type plays in a relationship. (e.g. prerequisite, requester)



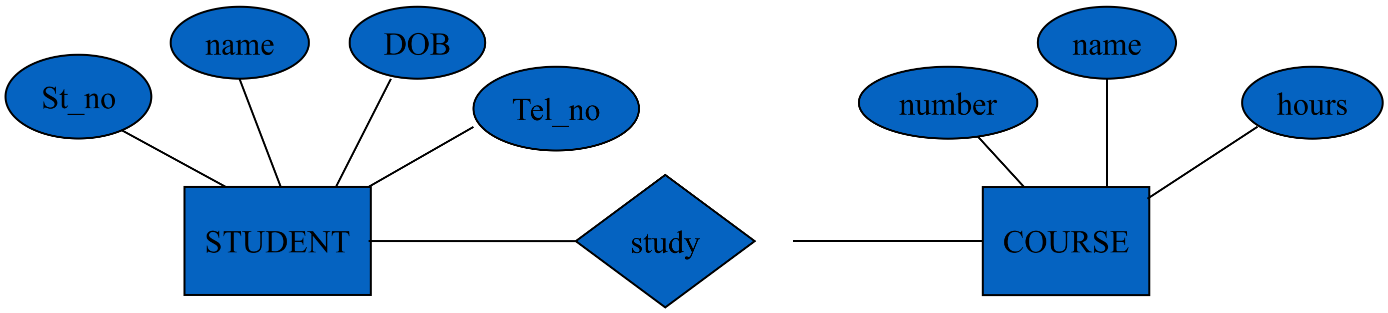
Role can be used when two entities are associated through more than one relationship to classify the purpose of each relationship.



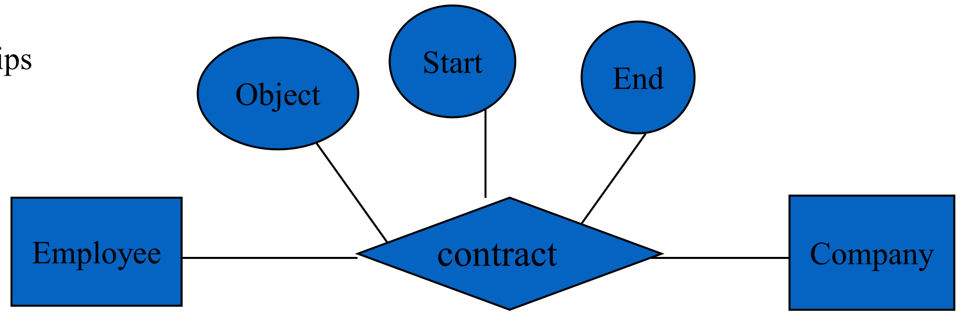
Attributes

Attributes are descriptive properties for an entity type **or** a relationship type.
All entities in one entity type have the same attributes.

Attributes of Entities



Attributes of Relationships

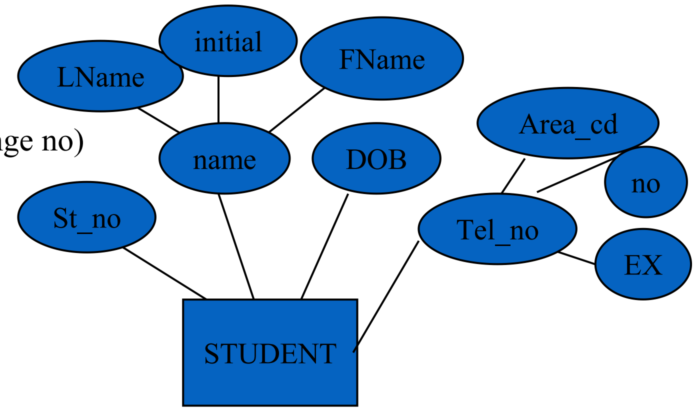


All relationships in one relationship type have the same attributes.
Relationships can be distinguished not only by their attributes but also by their participating entities.

Simple & Composite Attributes

Simple attribute is an attribute that have a single value with an independent existence.
e.g. salary, age, gender,...

Composite attribute is an attribute composed of multiple distinct components, each with an independent existence.
e.g. address (street, area, city, post code)
name (First name, initial, Last name)
phone no. (area code, number, exchange no)



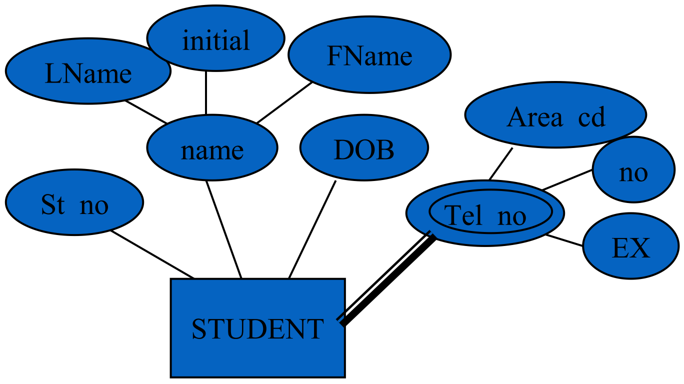
Single-valued & Multi-valued Attributes

Single-valued attribute is an attribute that holds a single value for a single entity. It is not necessarily a simple attribute.

e.g. student_no, age, gender,...

Multi-valued attribute is an attribute that may hold multiple values, of the same type, for a single entity.

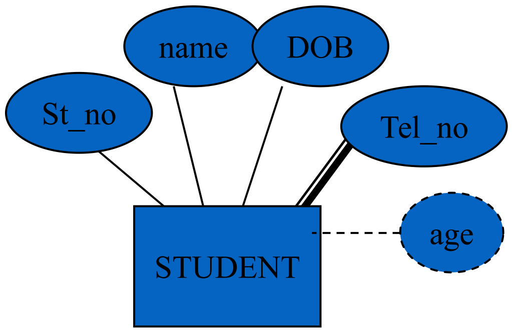
e.g. tel_no, degrees,…



Derived attribute is an attribute that represents a value that is derived from the value of a related attribute,not necessarily in the same entity type.

e.g. **age** is derived from **date_of_birth**

total_cost is derived from **quantity*unit_price**



Keys

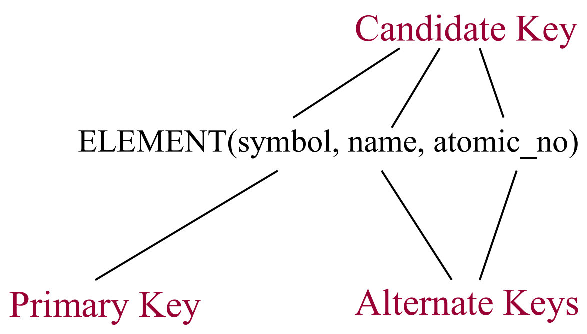
Candidate key(CK) is the minimal set of attributes that *uniquely* identifies an entity. It cannot contain null.

e.g. student_no, social_security_no, branch_no…

Primary Key(PK) is a candidate key that is selected to uniquely identify each entity.

Alternate Key(AK) is a candidate key that is NOT selected to be the primary key.

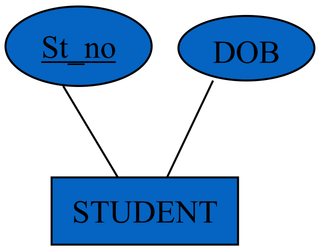
Keys Example



Choice of PK
Choice of Primary Key (PK) is based on:

- Attribute length
- Number of attributes required
- Certainty of uniqueness

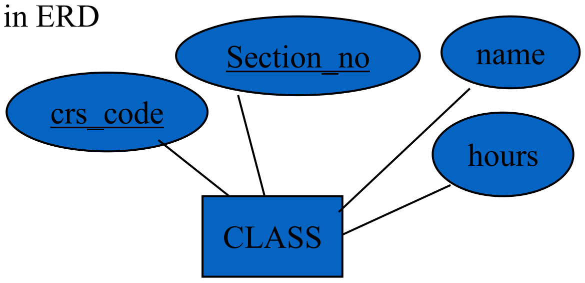
Primary Key in ERD



Keys
A key can be:
- **simple key** is a candidate key of one attribute.
  e.g. student_no, branch_no…
- **composite key** is a candidate key that consists of two or more attributes.

  e.g. STUDENT (Lname, Fname, Init)
  CLASS (crs_code, section_no)
  ADVERT (property_no, newspaperName, dateAdvert)

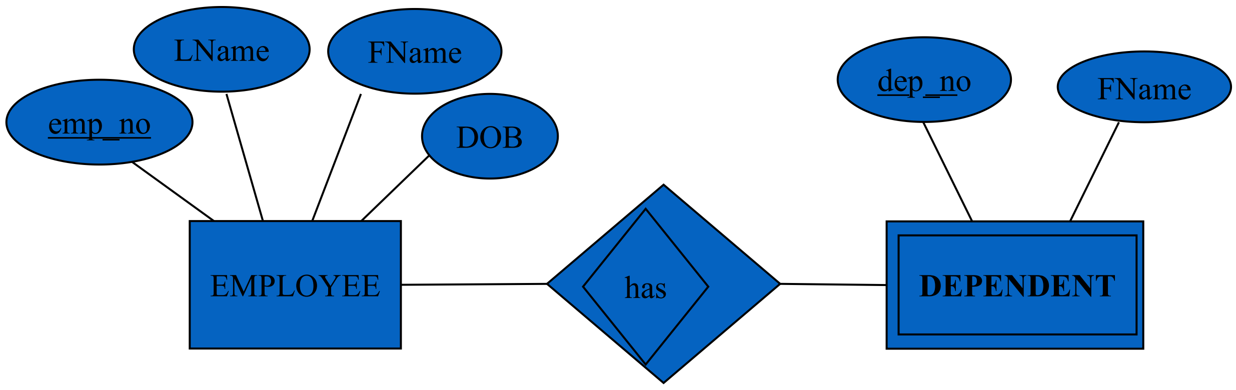
Composite Key in ERD



Strong & Weak Entity Types
A **strong entity type** is NOT existence-dependent on some other entity type. It has a PK.
A **weak entity type** is an entity type that is existence-dependent on some other entity type. It does not have a PK.

Weak Entity Type

- The existence of a weak entity type depends on the existence of a strong entity set; it must relate to the strong entity type via a relationship type called **identifying relationship**.
- The PK of a weak entity set is formed by the PK of its strong entity type, plus a weak entity type discriminator attribute.



Cardinalities

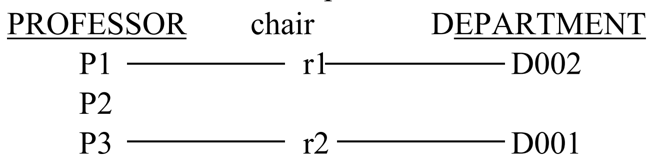
Cardinality ratio expresses the number of relationships an entity can participate in.

Most useful in describing binary relationship types.

For a binary relationship type the mapping cardinality must be one of the following types:

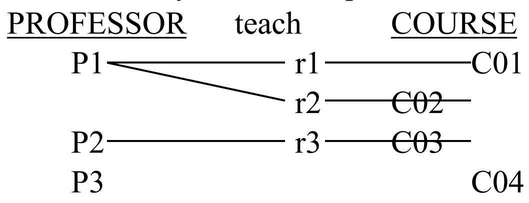
- One to one (1:1)
- One to many(1:M)
- Many to one (M:1)
- Many to many (M:N)

One-To-One Relationship



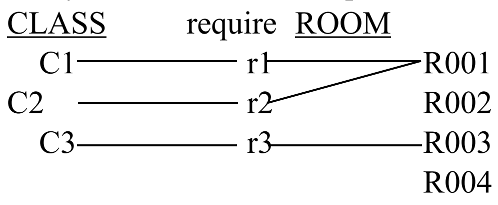
A professor chairs at most one department; and a department is chaired by only one professor.

One-To-Many Relationship



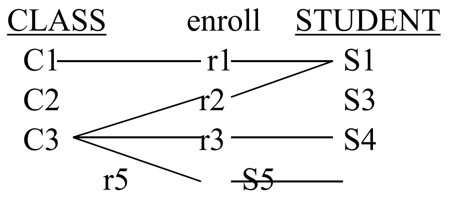
A course is taught by at most one professor; a professor teaches many courses.

Many-To-One Relationship



A class requires one room; while a room can be scheduled for many classes.

Many-To-Many Relationship



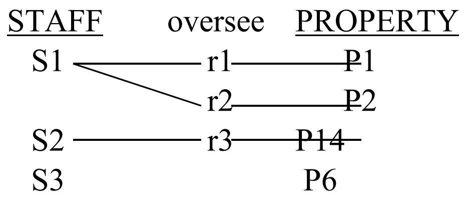
A class enrolls many students; and each student is enrolled in many classes.

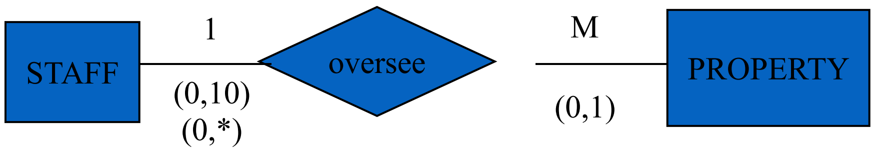
Multiplicity

Multiplicity is the number (range) of possible entities that may relate to a single association through a particular relationship.

It is best determined using sample data.

Takes the form (min#, max#)





Participation Constraints

Participation constraints determine whether all or only some entities participate in a relationship.

Two types of participation:

- **Mandatory** (total)
- **Optional** (partial)

- **Mandatory** (total) (1:*): if an entity's existence requires the existence of an associated entity in a particular relationship (existence-dependent).

  e.g. CLASS taught-by PROFESSOR

  i.e. CLASS is a *total participator* in the relation.

A weak entity always has a mandatory participation constraints but the opposite not always true.

- **Optional** (partial) (0:*): if an entity's existence does not require a corresponding entity in a particular relationship. (Not existence-dependent).

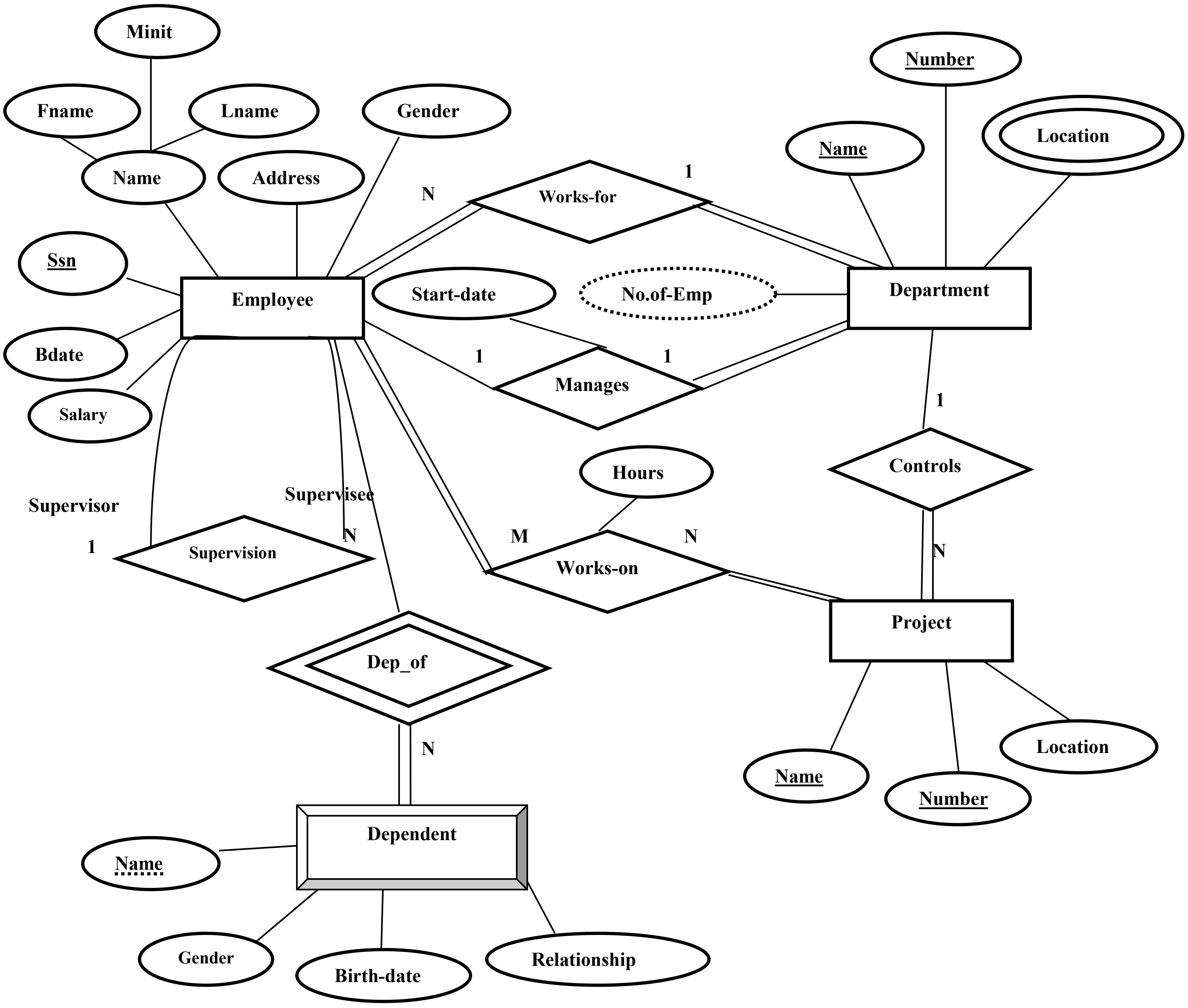
  e.g. PROFESSOR teach CLASS

  i.e. PROFESSOR is a *partial participator* in the relation.

Case Study for Students → Company:

- **The company keeps track of a company's → employees, departments, & projects.**
- **The company is organized into departments. Each department has a unique name, a unique number, & a particular employee who manages the department. Each department has a manager, we keep track of the start date when that manager began managing the department.**
- **A department may have several locations.**
- **A department controls a number of projects, each of which has a unique name, a unique number, & a single location.**
- **We store each employee's name, social security number, address, salary, gender, & birth date. An employee is assigned to one department but may work on several projects, which are not necessarily controlled by the same department. We keep track of the number of hours per week that an employee works on each project; we also keep track of the direct supervisor of each employee.**
- **We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, &relationship to the employee.**

The ERD for the Company DB Application


Minit
Fname
Lname
Gender
Name
Address
Ssn
Employee
Bdate
Salary
N
Works-for
1
Number
Name
Location
Department
Start-date
No.of-Emp
1
Manages
1
1
Controls
N
Supervisor
Supervisee
1
Supervision
N
Hours
M
Works-on
N
Project
Dep_of
N
Name
Number
Location
Dependent
Name
Gender
Birth-date
Relationship