8.4 Designs the conceptual schema of a database

Time: 12 periods

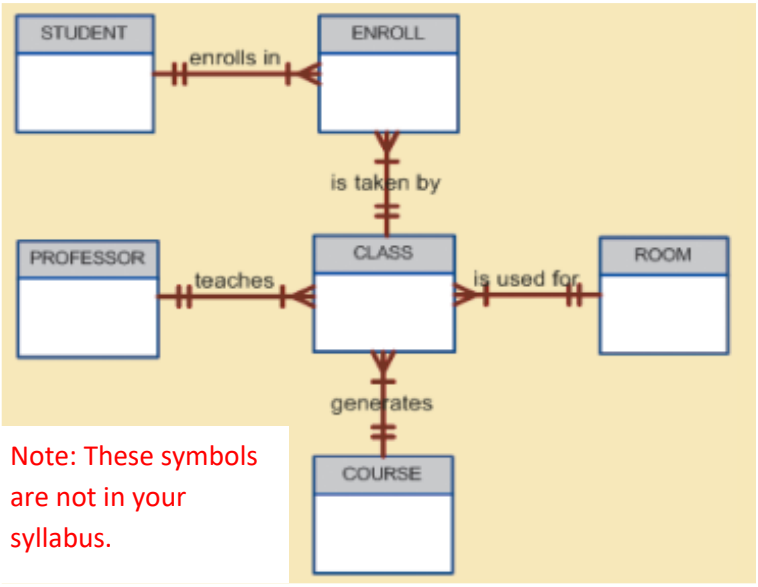
Learning Outcomes

- Describes ER diagram
- Describes the components of an ER diagram (entities, attributes)
- Describes entity identifiers
- Lists and describes relationships
- Describes cardinality
- Identifies the requirements of a given scenario
- Selects entities, attributes and according to the requirement
- Draws the ER diagram
- Explains the EER diagrams

What is a Data Model?

- Simple representation (usually graphical) of more complex real world data structures

- Tool for facilitating interactions among
  - Designers
  - Application programmers
  - End users



Note: These symbols are not in your syllabus.

Building a House	Building a DB
Blueprints	Data Models

- Different users have different **views and needs** of data
- A data model **organizes** data for different users
- Data modelling is:
  - The **first step** in designing a DB
  - The **foundation** of good database design

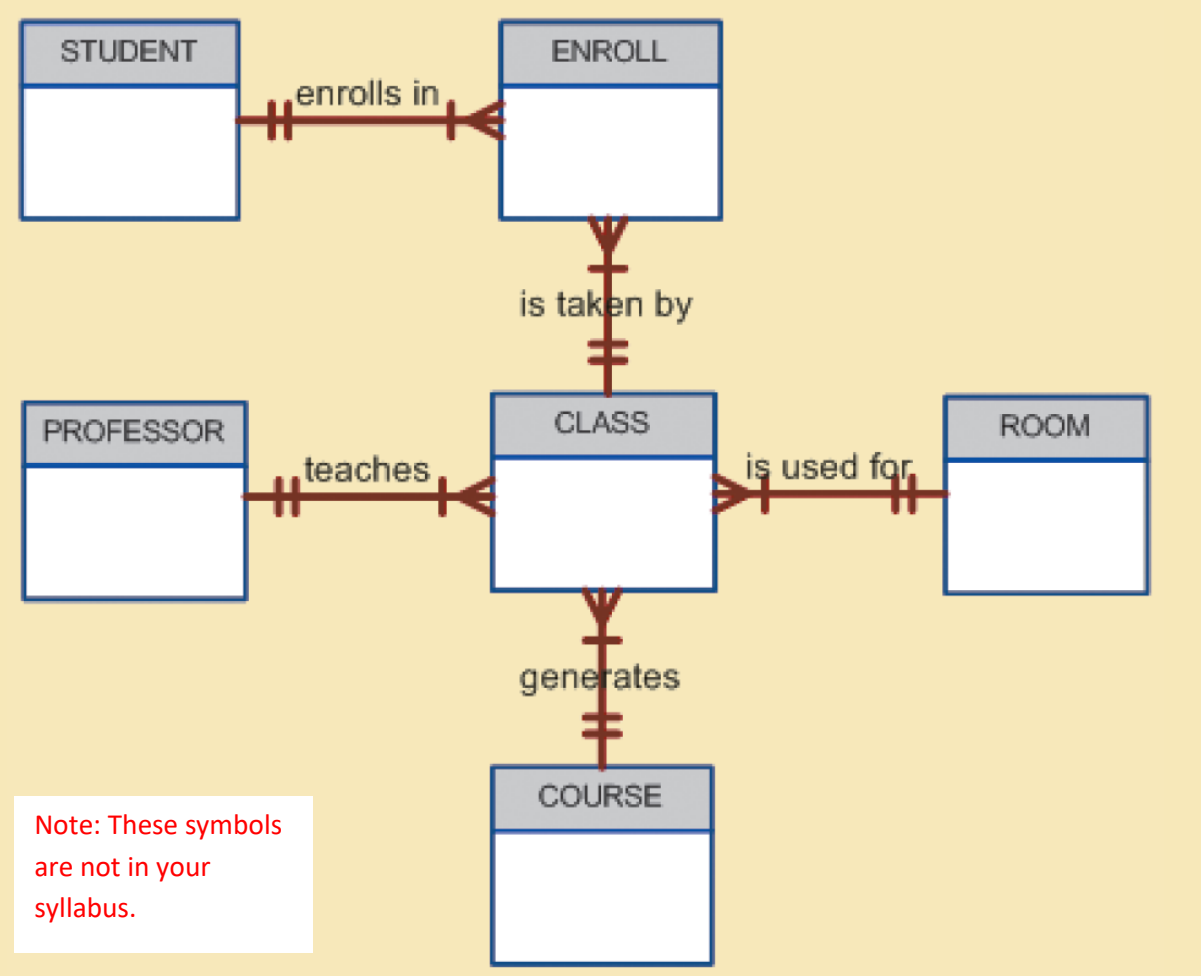
Building Blocks of Data Models

- **Entities**
  - Things about which data is to be collected
  - e.g., employee, students, courses
- **Attributes**
  - Characteristics of the entity
  - e.g., ID, name, DOB, manufacturer number
- **Relationships**
  - 1:1 (one-to-one)
  - 1:M (one-to-many)
  - M:N (many-to-many)

Business Rules

- **Business rules** determine the building blocks
- **Any organization** that stores data **has business rules**
- Business rules are:
  - Policies, procedures, or principles
  - Specific to an organization
- For example, each full-time student must select at least 3 units each semester

- This diagram appears to be modelling a teaching institute.



A system description:

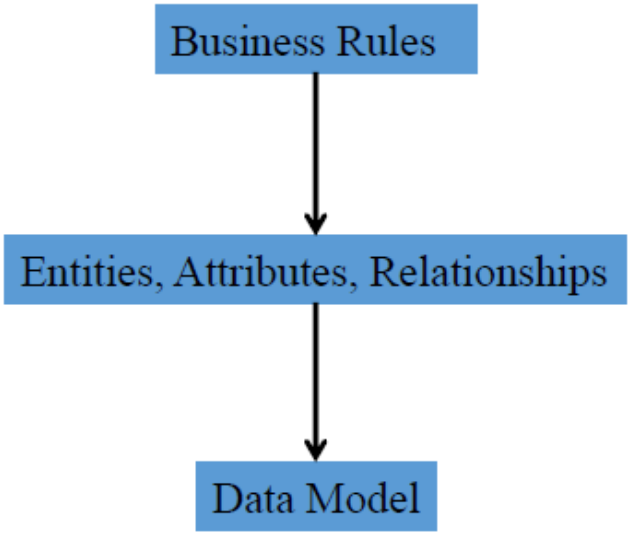
Business Rules: Example

The Jonesburgh County Basketball Conference (JCBC) is an amateur basketball association Each city in the county has one team that represents it Each team has a maximum of twelve players and a minimum of nine players Each team also has up to three coaches (defensive and PT coaches) Each team plays two games (home and visitor) against each of the other teams during the season.

• Determine entities and relationships

Importance of Business Rules

- Help **verify** data model
  - Standardize company's view of data
  - Defines relationships (and the constraints)
- Communication **tool**
  - Between users and designers (you)
  - Understand nature, role and scope of data
  - Understand business process



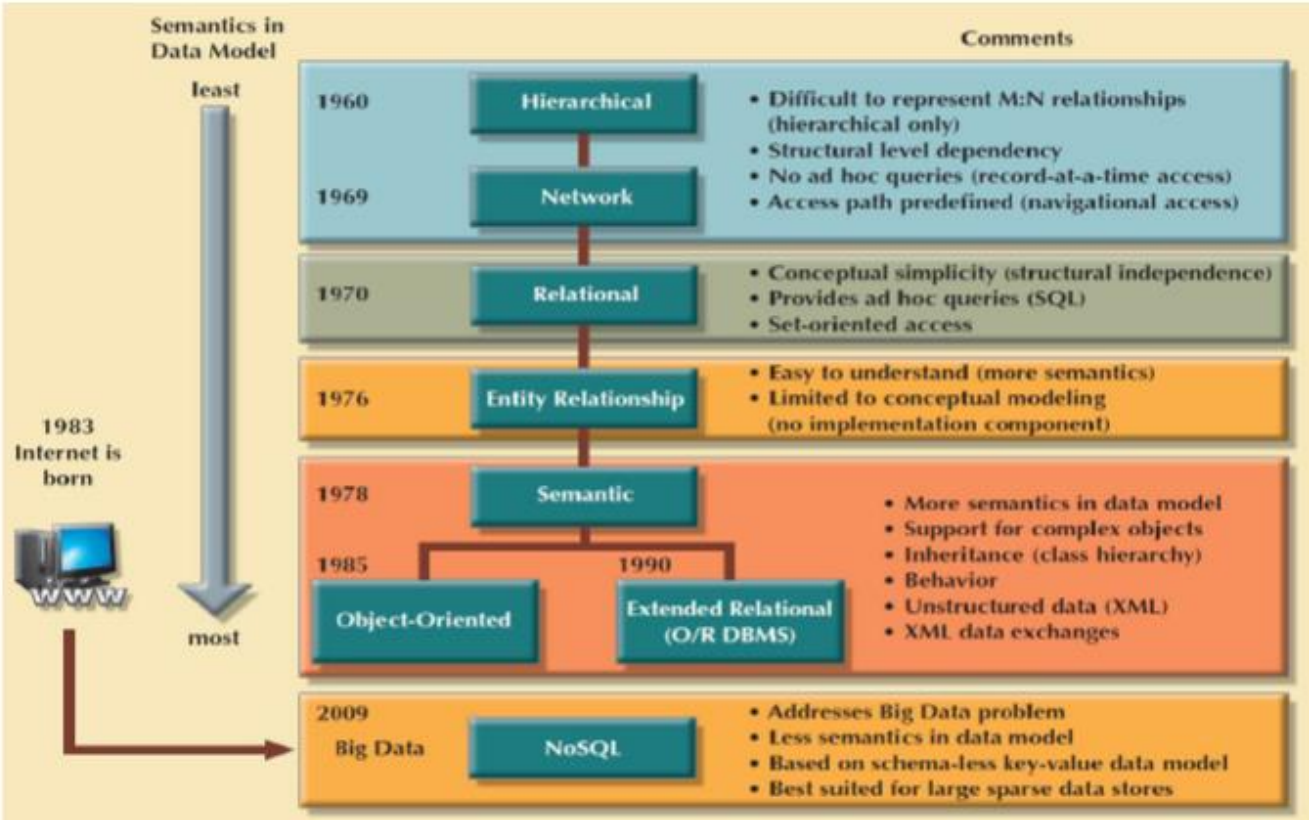
Evolution of Data Models

- **Early age** of data models
  - Hierarchical and Network models

- **Modern age** of data models (after 1970)
  - Relational model
  - Entity-Relationship (ER) model

- **New age** of data models
  - Object Oriented, Object/Relational, XML, Big Data and NoSQL

Evolution of Data Models



Relational Model

Table name: AGENT (first six attributes)

AGENT_CODE	AGENT_LNAME	AGENT_FNAME	AGENT_INITIAL	AGENT_AREACODE	AGENT_PHONE
501	Alby	Alex	B	713	228-1249
502	Hahn	Leah	F	615	882-1244
503	Okon	John	T	615	123-5589

Link through AGENT_CODE

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_RENEW_DATE	AGENT_CODE
10010	Ramas	Alfred	A	615	844-2573	05-Apr-2004	502
10011	Dunne	Leona	K	713	894-1238	16-Jun-2004	501
10012	Smith	Kathy	W	615	894-2285	29-Jan-2005	502
10013	Olowski	Paul	F	615	894-2180	14-Oct-2004	502
10014	Orlando	Myron		615	222-1672	28-Dec-2004	501
10015	O'Brian	Amy	B	713	442-3381	22-Sep-2004	503
10016	Brown	James	G	615	297-1228	25-Mar-2004	502
10017	Williams	George		615	290-2556	17-Jul-2004	503
10018	Farriss	Anne	G	713	382-7185	03-Dec-2004	501
10019	Smith	Olette	K	615	297-3809	14-Mar-2004	503

Relational Model:

- Developed by E.F. Codd of IBM in 1970
- Conceptually **simple**
- Users/designers work in human **logical** environment
- **Powerful** modelling capabilities
- But considered **impractical in 1970**due to lack of computing power
- Modern computers can handle relational models without problems

Relational Model: Merits and Limits

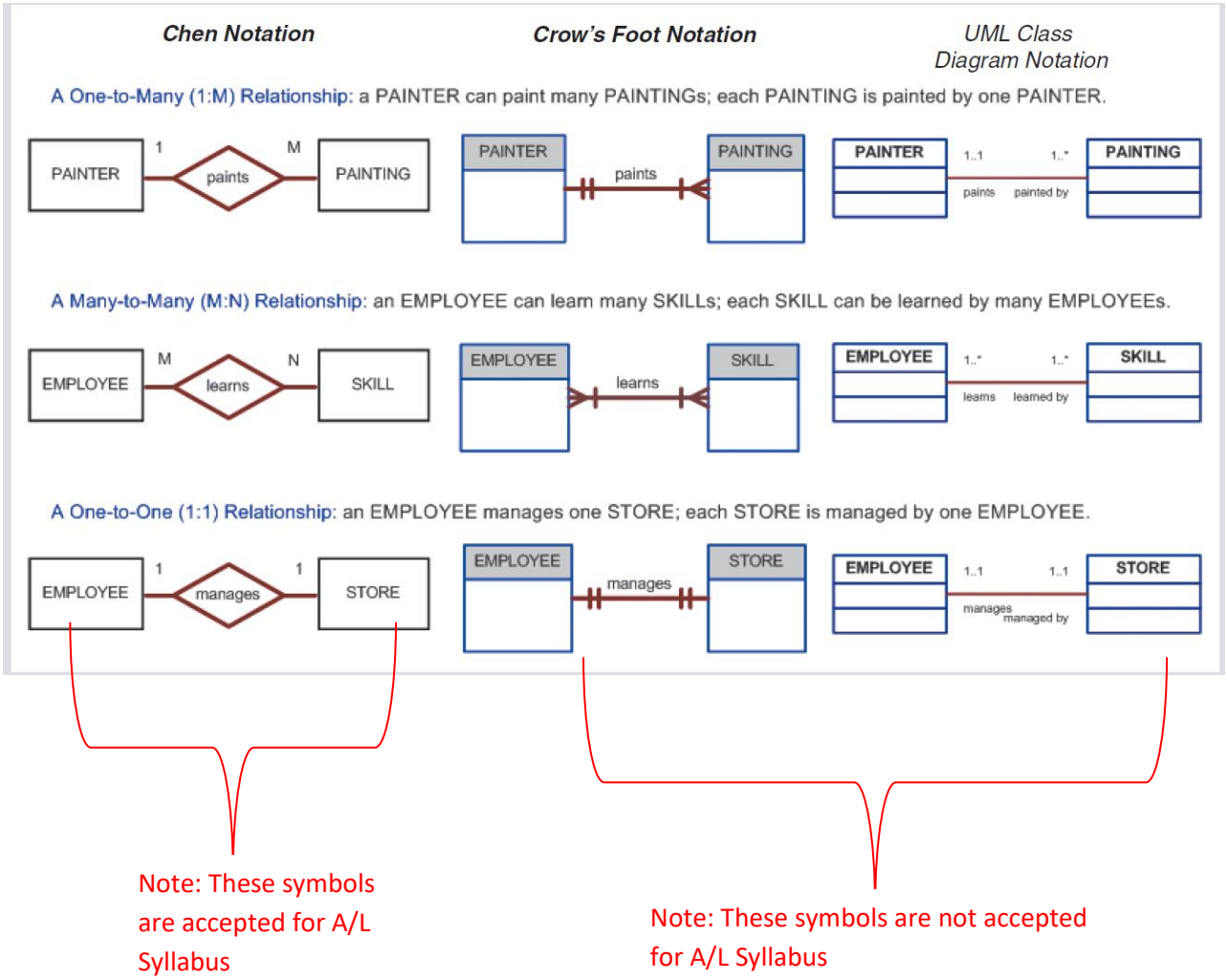
Merits

- Structural **independence** (DBMS handles the complexity)
- Conceptual **simplicity**
- Implementation **simplicity**

Limitations

- Substantial hardware/software **overhead**
- Can **facilitate poor design** and implementation!

Entity Relationship (ER) Model



Note: These symbols are accepted for A/L Syllabus

Note: These symbols are not accepted for A/L Syllabus

ER Model

A **graphical** data model

- Introduced by Chen in 1976
- Defines database structures in terms of the **entities and their relationships**

ER ↔ Relational

- An ER model maps to relational tables
- Each **entity set** maps to a relational table
- Each **entity instance** maps to a table entry (row)

Merits and Limits of ER Model

Merits

- Exceptional conceptual **simplicity**
- **Effective** communication tool
- **Complements** the relational data model

Limits

- Representation of constraints and relationships
- No data manipulation language

E-R Modeling

- is a design tool
- is a graphical representation of the database system
- provides a high-level conceptual data model
- supports the user's perception of the data
- is DBMS and hardware independent
- is composed of entities, attributes, and relationships
- method of analyzing the logical structure of an organization
- Views the real world as entities and relationships.

Utility of an ER model

- It maps well to the relational model
- It is simple and easy to understand with a minimum of training
- Can be used by the database designer to communicate the design to the end user
- The model can be used as a design plan by the database developer to implement a data model in a specific database management software

Four Tests for an Entity Type

- It should be of importance to the system being studied
- There should be information associated with the entity type. This means that you should be able to think of a number of attributes that belong to the entity type
- There should be several occurrences of the entity type
- Each occurrence should be uniquely identifiable In other words, it must be possible to define a primary key for each entity type

Entity Type and Entity

Entity Type

- An object or concept that is identified by the enterprise as having an independent existence.
- For example EMPLOYEE, DEPARTMENT, PROJECT.
- Entities with the same basic attributes are grouped or typed into an entity type.

Entity

- A uniquely identifiable object or concept that is of significance to the system and about which information is to be held.
- For example the employee JOHN SMITH, the RESEARCH department, the PRODUCTX project

Entity Instance Vs. Entity type

- There is a distinction between an entity instance and an entity type
- Entity instance (entity occurrence)
    - Instance of an entity type,
    - Has values for attributes
- Entity type (entity set)
    - is a concept that represents a set of entity instances that share common characteristics
    - has a list of properties but no values for them
- Each entity type has a key.

Entity

- Something that an organization considers significant, and therefore stores information on
- Corresponds to an RDBMS Relation

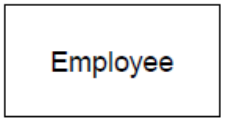
Organization	Entity Types
School	Student, Subject, Teacher
Hospital	Patient, Ward, Disease
Grocery	Customer, Order

Entity Type

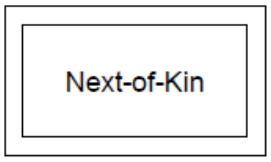
- Two classes of entity type
- **Weak entity**
  - an entity whose existence depends on other entities
  - Can be identified uniquely only by considering the primary key of another (*owner/strong*) entity.
  - represented by a doubled line rectangle labelled with the entity name
- **Regular / Strong / Owner entity**
  - an entity that is not weak
  - represented by a rectangle labelled with the entity name
  - Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities).
  - Weak entity set must have total participation in this *identifying* relationship set

ER Notation Entity

- Strong Entity



- Weak Entity



Entity Occurrence

- A particular instance of an entity type which can be uniquely identified
- Below are given **two** Occurrences of the **Student** Entity
- Corresponds to an RDBMS "Tuple"

Student –No	Student Name	Age
96001	Mr. Arun	18
96005	Ms. Shanthi	19

Attribute

- A property of an entity
- i.e holds all data elements associated with entities
- Corresponds to an RDBMS "Field"

Types

- Single / composite
- Key
- Single valued / multi valued
- Base / derived

Entity	Attribute
Student	Student-No, Name, Address

Types of Attributes

- **Attribute:** A property of an entity type or a relationship type.

For example the entity type STAFF may have attributes Name, Address, Tel_No, and Position.

- **Attribute Domain:** A set of allowable or permissible values that may be assigned to a single-valued attribute

Simple or Composite

- **Simple attribute**
  - No internal structure
  - Atomic or scalar
  - Shown as labelled ellipse attached to the entity by a link
- **Composite attribute**
  - Made of a set of simple or composite attributes
  - Has an internal structure
  - Represented as having further ellipse attached to it

Single or Multi valued

- **Single value attribute**
  - Can have a single value from the associated domain

- **Multi valued attribute**
  - Can have more than one value from the associated domain
  - Represented by doubled line ellipse

Based or Derived

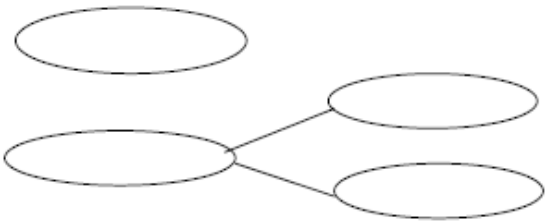
- **Derived attribute**
  - Can be deduced from a set of attribute
  - Represented by a dotted line ellipse
- **Base Attribute**
  - Cannot be deduced from other attributes

Primary key

  - Uniquely identify any instance of the entity type

E-R Notation –Attribute

- Simple Single valued
- Composite



- Key
- Multi valued

- Derived

Keys

Candidate Key

An attribute or set of attributes that uniquely identifies individual occurrences of an entity type.

Primary Key

An entity type may have one or more possible candidate keys, one of which is selected to be the primary key.

Composite Key

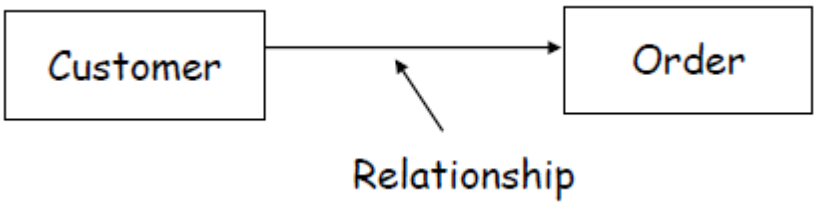
A candidate key that consists of two or more attributes.

Relationship Types

- **Relationship Type**
    - A meaningful association among entity types. **Relationships of the same type are grouped or typed** into a relationship type.
    - For example, the WORKS_ON relationship type in which EMPLOYEEs and PROJECTs participate, or the MANAGES relationship type in which EMPLOYEEs and DEPARTMENTs participate.
    - A relationship type can have attributes.

- **Relationship**
    - An association of entities where the association includes one entity from each participating entity type. Also known as relationship occurrence or relationship instance.
    - A relationship relates two or more distinct entities with a specific meaning.
    - For example, EMPLOYEE John Smith *works on* the ProductX PROJECT or EMPLOYEE Franklin Wong *manages* the Research DEPARTMENT.

- Entities are linked together by relationships
- They are associations between two entities
- Exists if one or more attributes are common to two entities
- Read in either direction



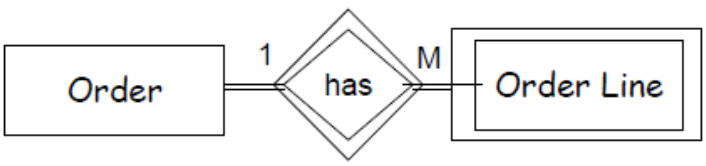
Degree

- The number of entities associated with the relationship
- **Unary relationships** : an entity associated with it self
- **Binary relationships** : The association between two entities
  - the most common type in the real world
- **Ternary relationship** : involves three entities
  - Used when a binary relationship is inadequate
  - Often ternary or n-ary relationships are decomposed into two or more binary relationships

ER Notation Relationships

Relationship between two Strong Entities

Relationship between a Strong Entity and a Weak Entity

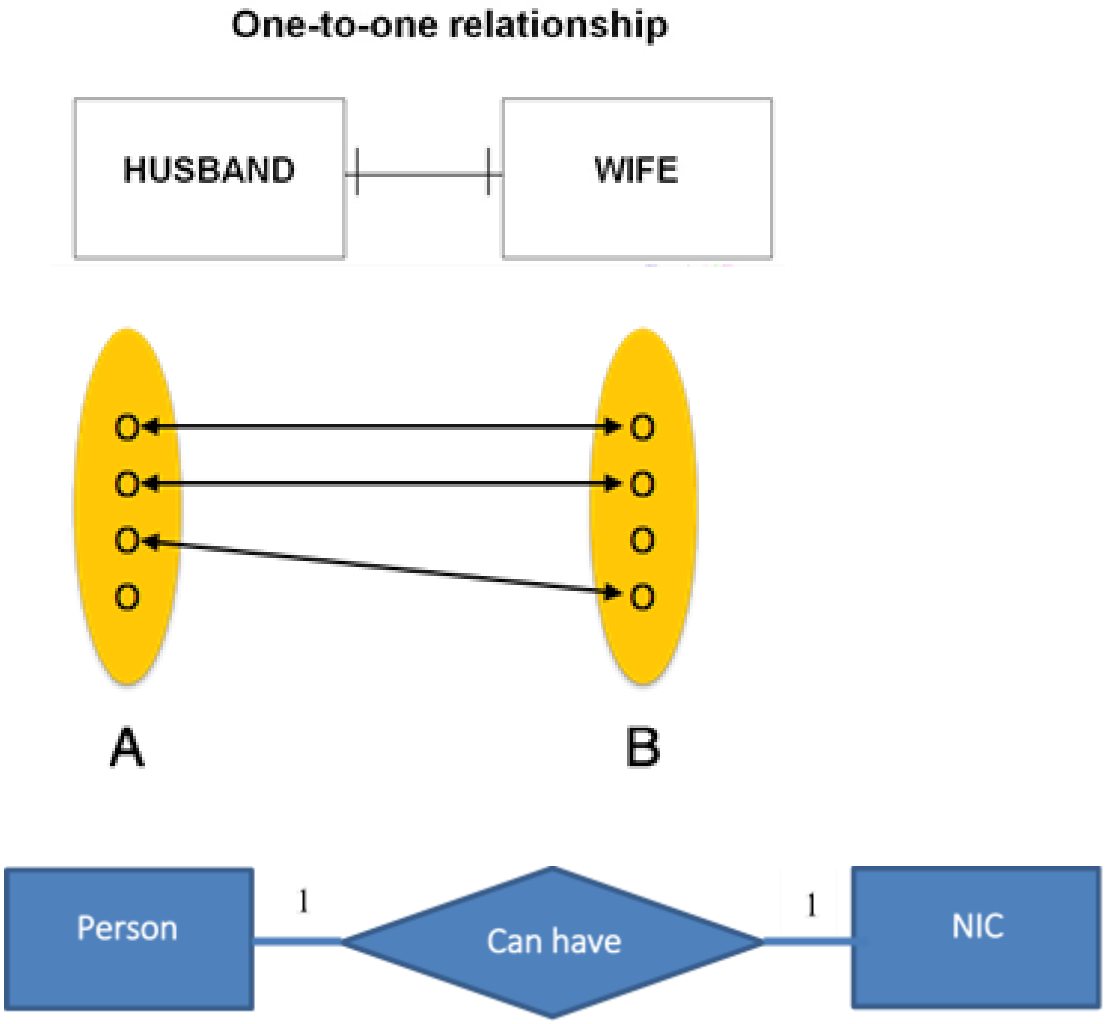


Structural Constraints - Cardinality

- A relationship may allow multiple occurrences of subject / object.
- Singular occurrence vs Multiple occurrence stands for —"one or more"
- Possible Cardinalities:
  - One to One 1:1
  - One to Many 1: m
  - Many to Many m: n

One to One (1:1)

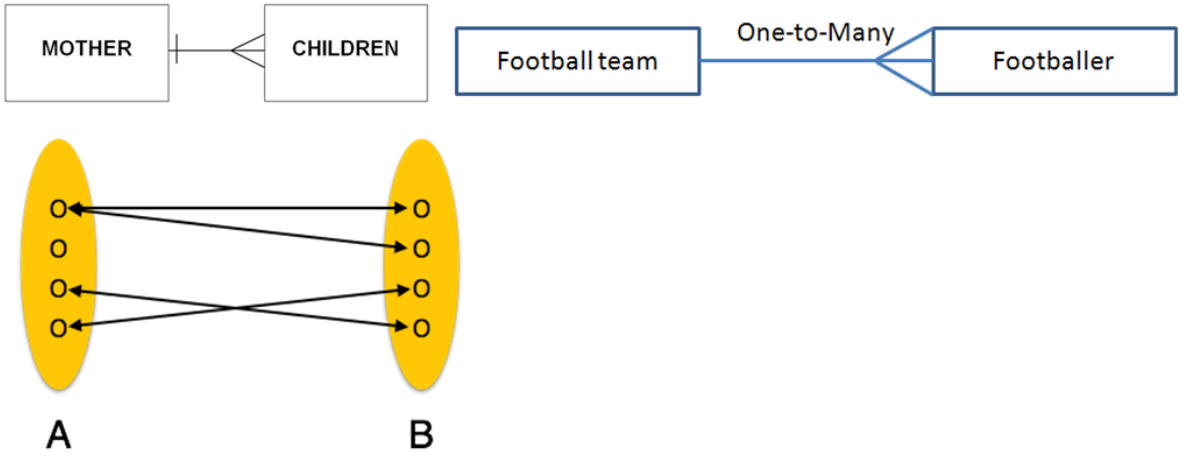
One entity from entity set X can be associated with at most one entity of entity set Y and vice versa.



A person has only one passport and a passport is given to one person.

One to Many (1: m)

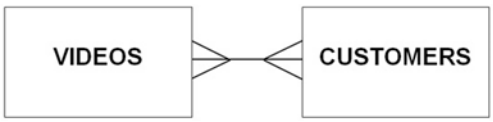
One entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity.

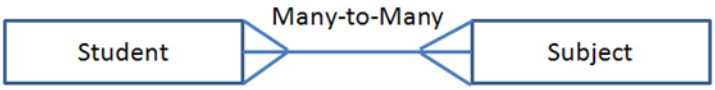


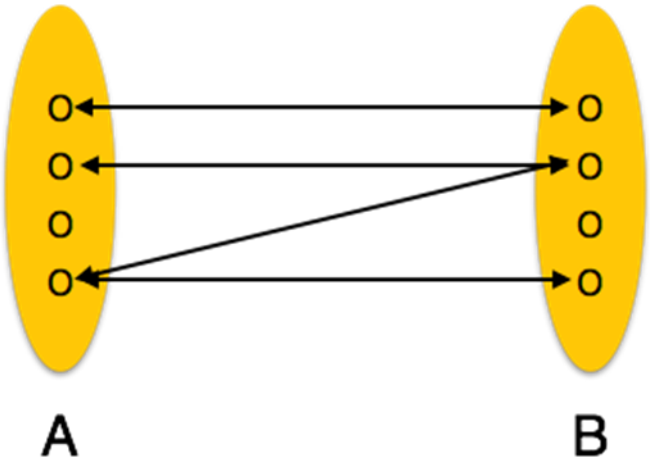
A customer can place many orders but an order cannot be placed by many customers.

Many to Many (m: n)

One entity from X can be associated with more than one entity from Y and vice versa.







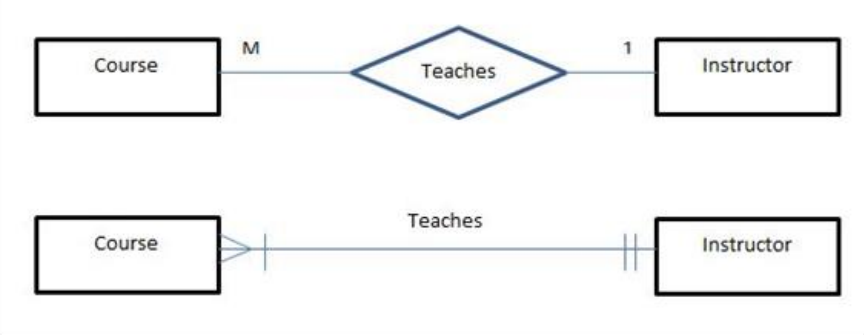
A Student can be assigned to many projects and a project can be assigned to many students.

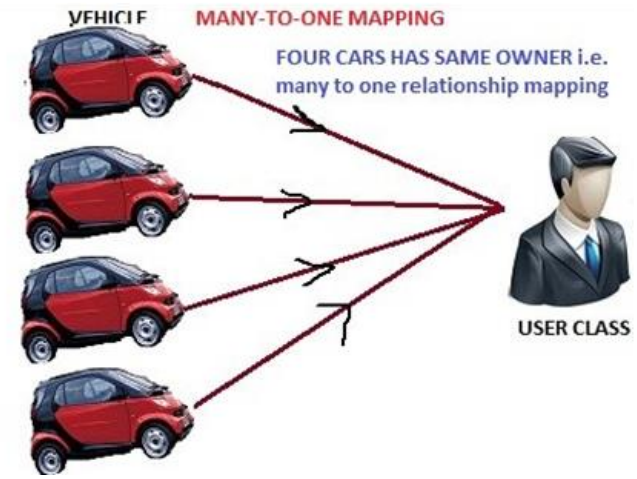
Note - relational databases are unable to handle m: n relationships.

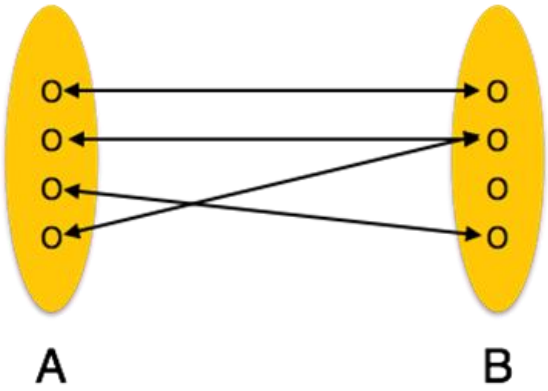
Therefore if they occur they must be resolved by two 1: n relationships

Many to One (M: 1) ***** not mentioned in your syllabus

More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set X may or may not be associated with more than one entity from entity set X.







Many students can study in a single college but a student cannot study in many colleges at the same time.

Structural Constraints - Participation/Optionality Constraints

- Participation constraints determine whether the existence of an entity depends upon its being related to another entity through the relationship.

- There are 2 types of participation constraints, namely,
  - Total (or mandatory) and
  - Partial (or optional) participation.

- The participants in each relationship are connected by lines, which are single if the participation is partial and double if the participation is total.

- **Total/Mandatory Participation**

  - A mandatory relationship is where one occurrence of an entity type must be associated with an occurrence of the other entity type.

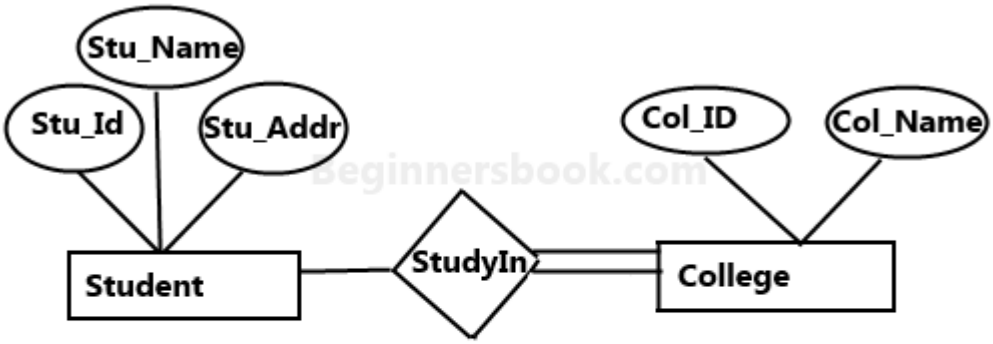
  - The participating entity type is connected by a double line to the relationship type.

- **Partial/Optional Participation**

  - A relationship is said to be optional between 2 entities if an occurrence of one may exist without associated occurrences of the other.

  - The participating entity type is connected by a single line to the relationship type.

  - Participation Constraints of Branch is Allocated Staff Relationship



E-R Digram with total participation of College entity set in StudyIn relationship Set - This indicates that each college must have atleast one associated Student.

Basic steps involved in building an Entity Relationship (E R) Model (Conceptual Design/Data Model)

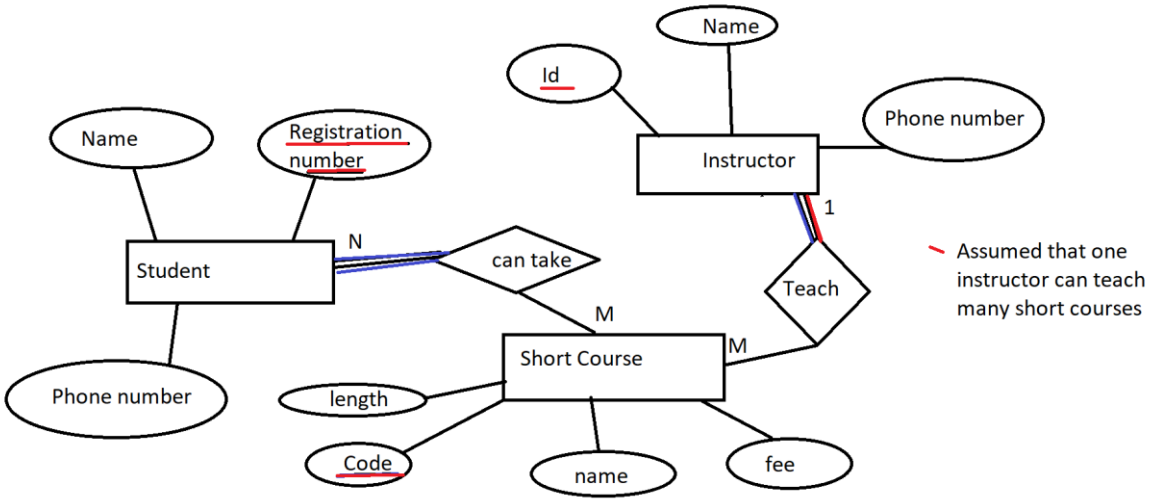
- Identify Entity Types
- Identify Relationship Types
- Determine Cardinality and Participation Constraints of Relationship Type
- Identify and Associate Attributes with Entity or Relationship Types
- Determine Attribute Domains
- Determine Candidate and Primary Key Attributes

Case Study

1.

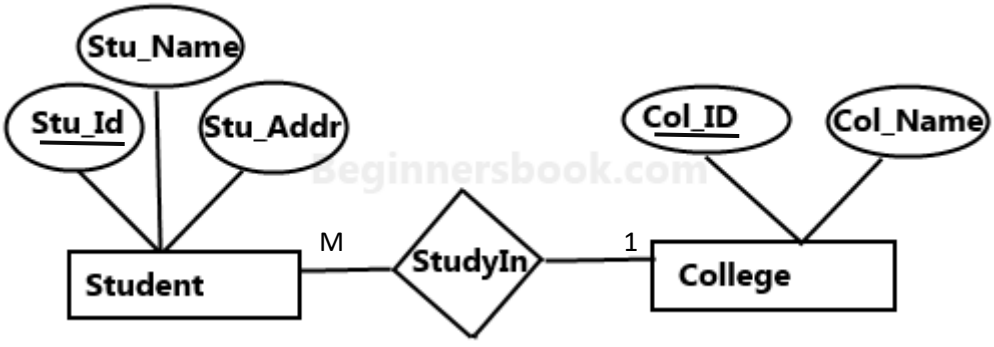
“I am the director of the university’s Short Course Programme Each of the short courses we teach has a code, a name, and a fee Beginners Systems Analysis and Web Design are two of our more popular courses.

Courses vary in length from three to ten days Reza and Jas are two of our best instructors We need each instructor's id, name and phone number The students can take several courses over time, and many do this We like to have each student’s registration no, name and phone number”



2.

In the following diagram we have two entities Student and College and their relationship. The relationship between Student and College is many to one as a college can have many students however a student cannot study in multiple colleges at the same time. Student entity has attributes such as Stu_Id, Stu_Name & Stu Addr and College entitv has attributes such as Col ID & Col_Name.



Extended ER Diagram -EERD)

EER is a high-level data model that incorporates the extensions to the original ER model. Enhanced ERD are high level models that represent the requirements and complexities of complex database.

In addition to ER model concepts EE-R includes:

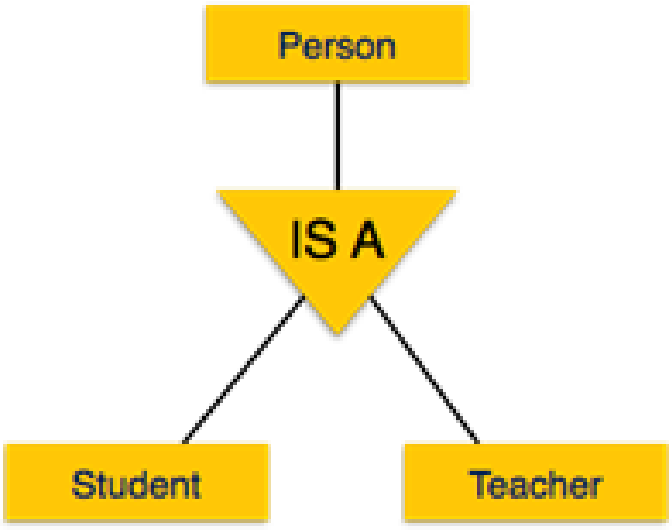
1. Subclasses and Super classes.
2. Specialization and Generalization.
3. Category or union type.
4. Aggregation.

These concepts are used to create EE-R diagrams.

Specialization –

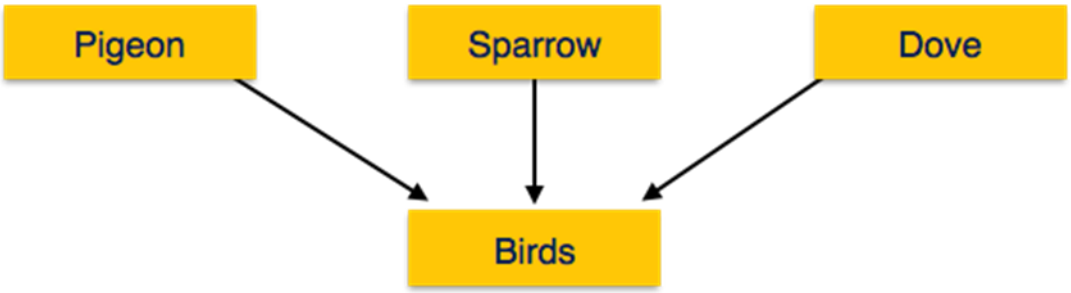
Specialization is a process of identifying subsets of an entity that share some different characteristic. It is a top down approach in which one entity is broken down into low level entity.

In above example Vehicle entity can be a Car, Truck or Motorcycle.



Generalization

It is a Bottom up process i.e. consider we have 3 sub entities Car, Truck and Motorcycle. Now these three entities can be generalized into one super class named as Vehicle.



Reference

Lecture notes - Dr. Dilani Wickramaarachchi (University of Kelaniya)

Teachers Guide (2107)