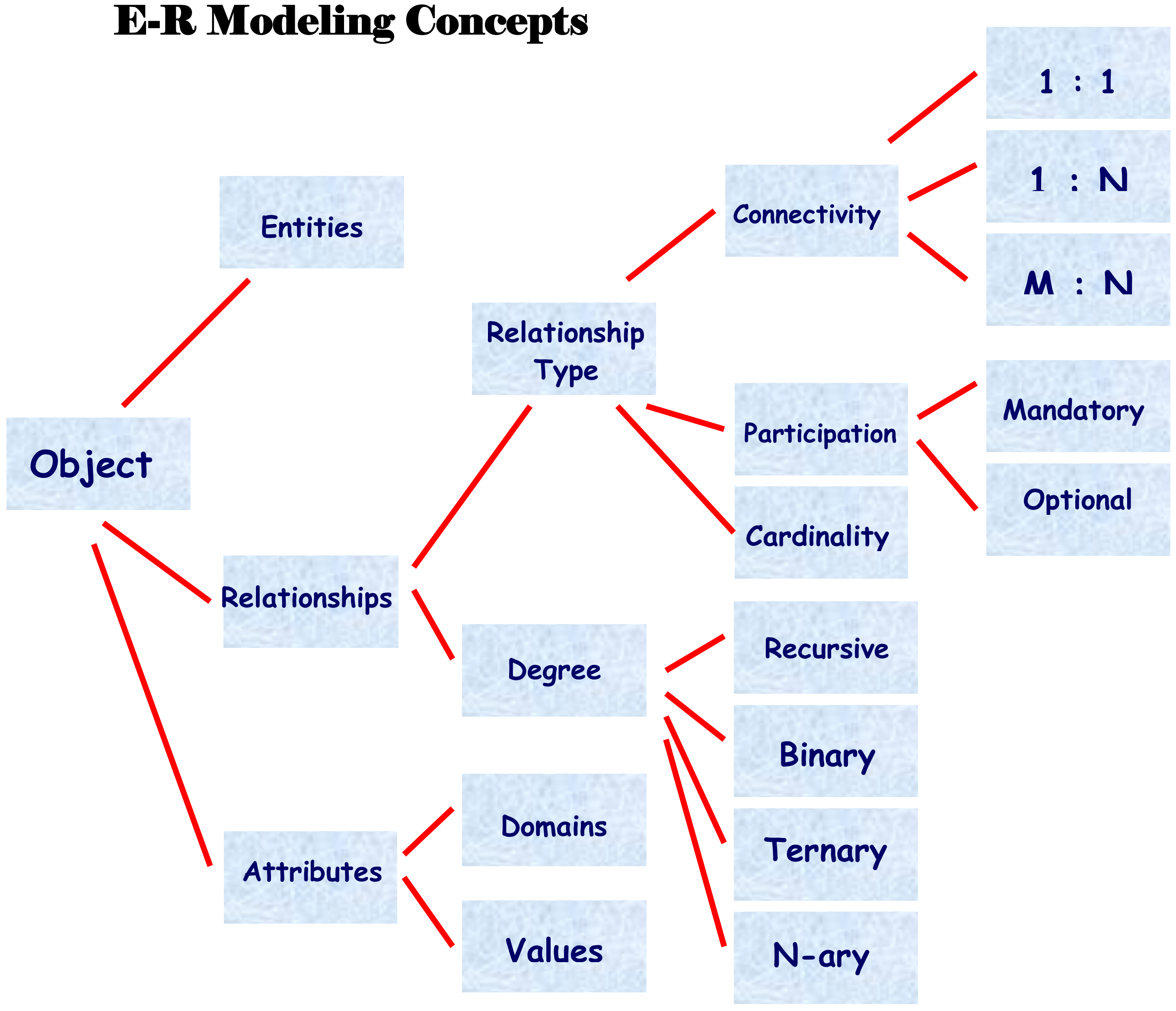
Entity Relationship Models (ERMs)

Contents

E-R Modeling Concepts



1- Introduction (ERMs)

Introduced by Peter Chen in 1975/6.An ERM provides a quick reference to an organisation's data structures - the relationships between the data components of a system Completed ERM can be used stand-alone to design an information system's data structures or in conjunction with a DFD to provide a more comprehensive IS logical design It was intended for designing databases but is also used more widely to document the data requirements of the system under investigation and to diagrammatically represent the relationships between data entities.The Entity-Relationship model is the name of a top down approach to system design

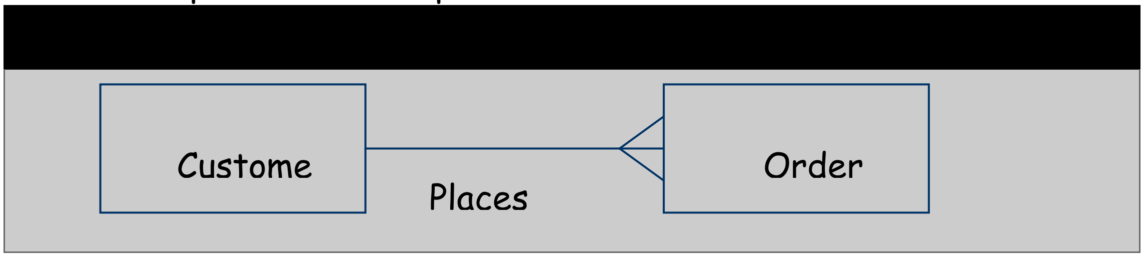
The data flow diagram is concerned with the question, What does the system do with the data?
The data model is concerned with the question, What data does the system need to store and what is the most efficient way of organizing the data

- ***Stored data requirements are not ignored by the data flow diagram; its data stores model the stored data.***
  However, when designing data stores, the system developer is not attempting to find an efficient way of organizing the data, merely identifying the stored items. Data modelling uses two separate techniques to achieve a satisfactory set of entities, entity-relationships (E-R) modelling and normalization.

- ***The identification of the data objects or entities in the system, their structure and relationships between entities.***
  The construction of a model of the stored data requirements of the system which is independent of specific processing requirements.

- ***The construction of a robust data model i.e. Minimal model of the data required to be stored in the system.***
  The construction of a logical model of data i.e. a model that is not concerned with how the data storage will be, or is currently physically implemented.
- ***Data modelling aims to identify the system entities, i.e. items which the system needs to store data (e.g. customers, orders products etc.)***

- ☒ ***The data model also shows the relationships between entities.***
  Customer places an order
- ☒ ***The data model concentrates on the properties of the data itself, independent of the processing requirements of the system.***
  The data model aims to collect all the data that needs to be stored for the system to operate and to organize that data into a sound structure
- ☒ ***Its is important that little redundant data is stored.***
- ☒ ***As far as possible one item of data should be stored in one place and only one place only.***
  If the same data is stored more than once this can lead to discrepancies between data that has been updated in place but not in another
- ☒ ***Data modelling is not concerned with how the data is physically stored in the current system or in the computer-based system being developed.***
  The data model aims to develop an efficient model of the data, independently of how it is implemented
- ☒ ***A relationship is a link between two entities.***
  For example a customer places an order

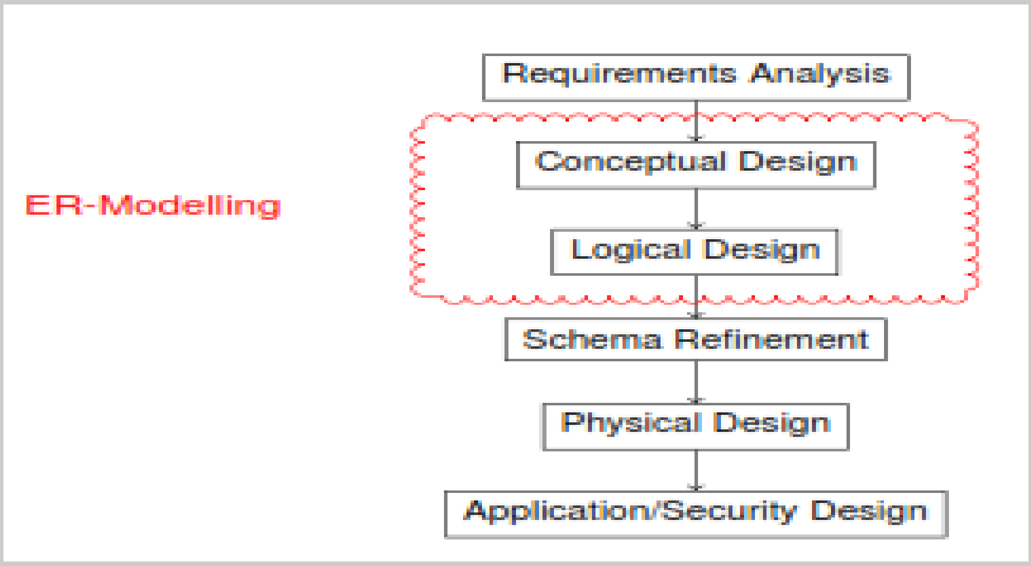


2- Database Design Process

The six main steps in designing a database

- Application & Security Design
- Physical Design
- Schema Refinement
- Logical Design
- Conceptual Design
- Requirements Analysis

designing a database



☒ ***Requirements Analysis***

Requirements Analysis is the process of determining what the database is to be used for. It involves interviews with user groups and other stakeholders to identify what functionality they require from the database, what kinds of data they wish to process and the most frequently performed operations. This discussion is at a non-technical level and enables the database designers to understand the business logic behind the desired database

☒ ***Conceptual & Logical Design***

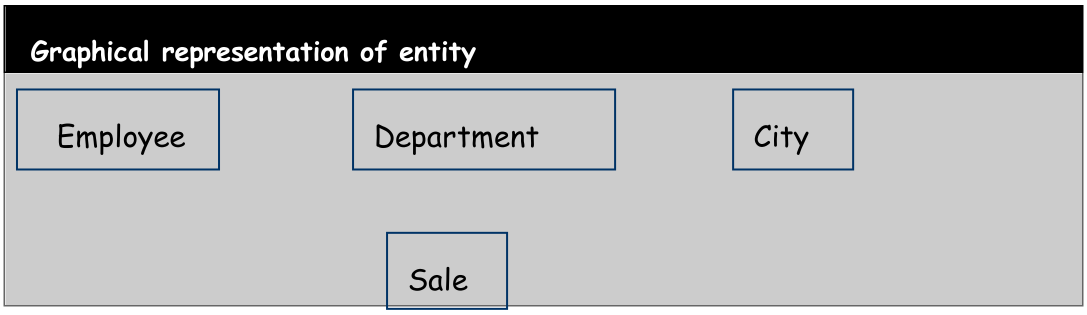
Using the ER data model, the conceptual design stage involves identifying the relevant entities and the relationships between them and producing an entity-relationship diagram. The logical design stage involves translating the ER diagram into actual relational database schema. Once a suitable ER model has been constructed, then this can be translated into a relational database fairly easily. However ER modelling is as much an art as a science, as there are usually many choices to be made and the consequences of each choice sometimes does not become apparent until problems arise later.

3- Basic ER Modelling

The E-R (entity-relationship) data model views the real world as a set of basic objects (entities) and relationships among these objects. It is intended primarily for the DB design process by allowing the specification of an enterprise scheme. This represents the overall logical structure of the DB.

- ✓ ***Entity***
  Real-world object distinguishable from other objects. An entity is described (in DB) using a set of attributes. An entity set represent classes of objects (facts, things, people,...) that have properties in common and an autonomous existence, e.g., City, Department, Employee, Purchase and Sale.

- ✓ ***Entity set***
  An entity is an instance/member of an entity set, e.g., Stockholm, Helsinki, are examples of instances of the entity City; Peterson and Johanson are examples of instances of the Employee entity set



- ***Attribute***

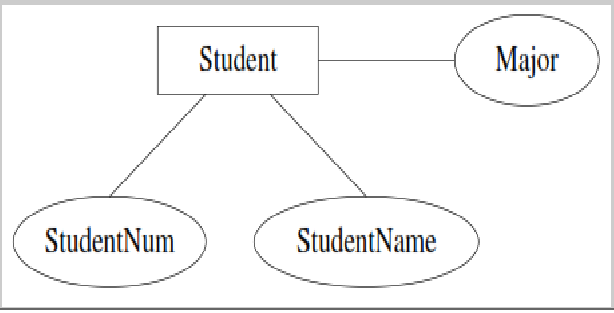
  properties that describe an Entity's characteristics

  - ***Simple*** - an attribute which cannot be broken down into smaller parts
    Example - Social Security Number
  - ***Composite*** - an attribute which can be broken down into smaller parts
    Example - student's full name
  - ***Multi-Valued*** - an attribute that may have more than one value for a given instance. Example - course IS480 has Section A and B...which quarter…which year
  - Derived - an attribute whose value can be calculated from other attributes .Example - gross wages = hours * rate…net wages = gross wages Minus Taxes

- ***Domain***:

  The domain of the attribute is the set of permitted values
  (Example the telephone number must be seven positive integers).

Graphical representation of attributes



- ***Identifier*** –

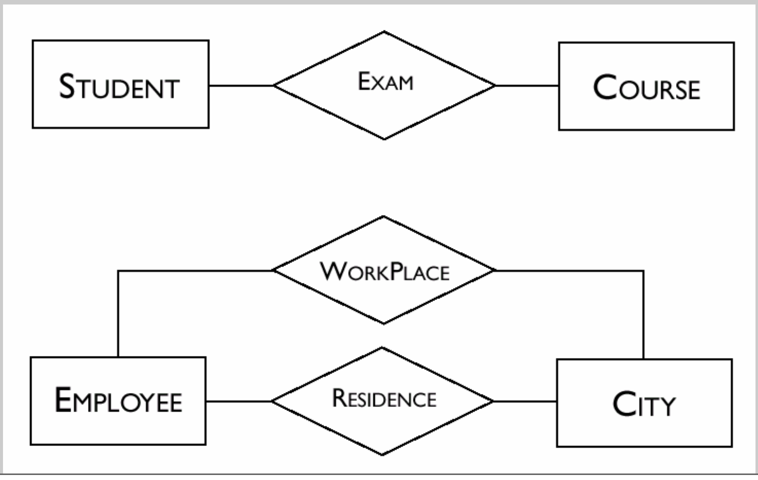
  at attribute or combination of attributes that uniquely identifies an entity instance
  Example. Student - social security number
  Composite Identifier - an identifier consisting of 2 or more attributes
  Example. Course Identifier - department (IS); course number (480); section letter (A)

4- Basic Relationship

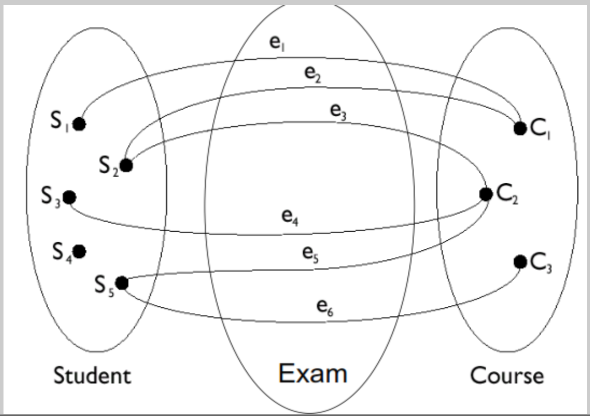
Association among two or more entities or logical links between two or more entities. Relationships are between *entities,* not attributes.

Residence is a relationship set that can exist between entity sets City and Employee; Exam is an example of a relationship that can exist between the entities Student and Course.

Graphical representation of attributes



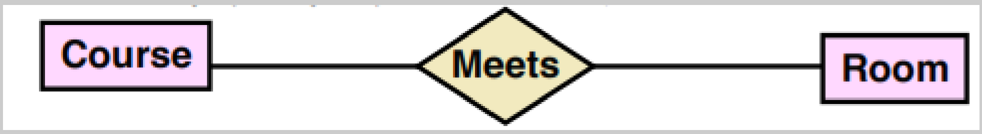
Example Instances for Exam



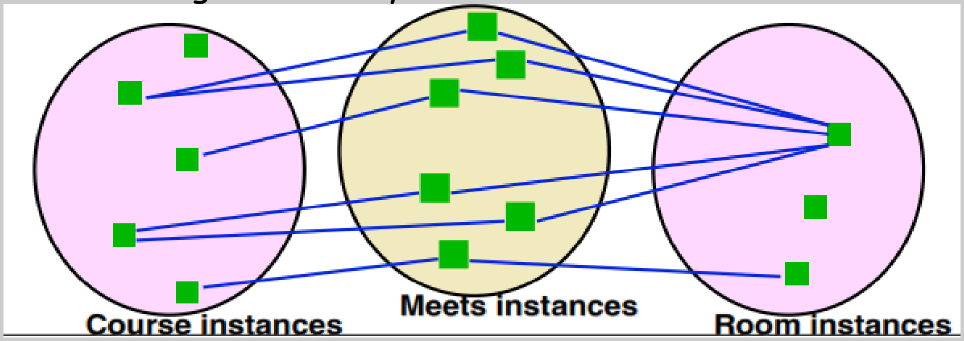
- ***Relationship Set***

  Collection of similar relationships. An n-ary relationship set R relates n entity sets E1 ... En; each relationship in R involves entities e1∈ E1, ..., en ∈ En Same entity set could participate in different relationship sets, or in different "roles" in same set.

What Does An ER Diagram Really Mean?

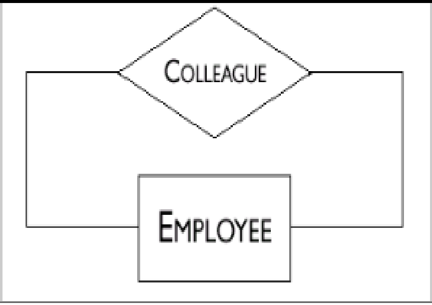


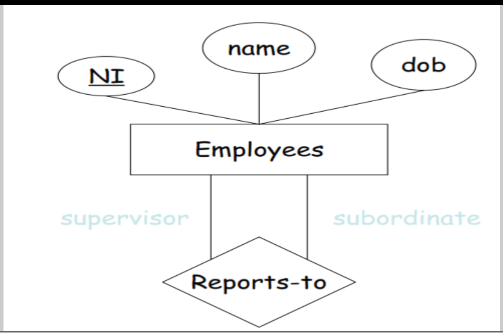
- ***Course and Room are entities.***
  Their instances are particular courses (eg CSC340F) and rooms (eg MS2172)
- ***Meets is a relationship.***
  -Its instances describe particular meetings.
  -Each meeting has exactly one associated course and room



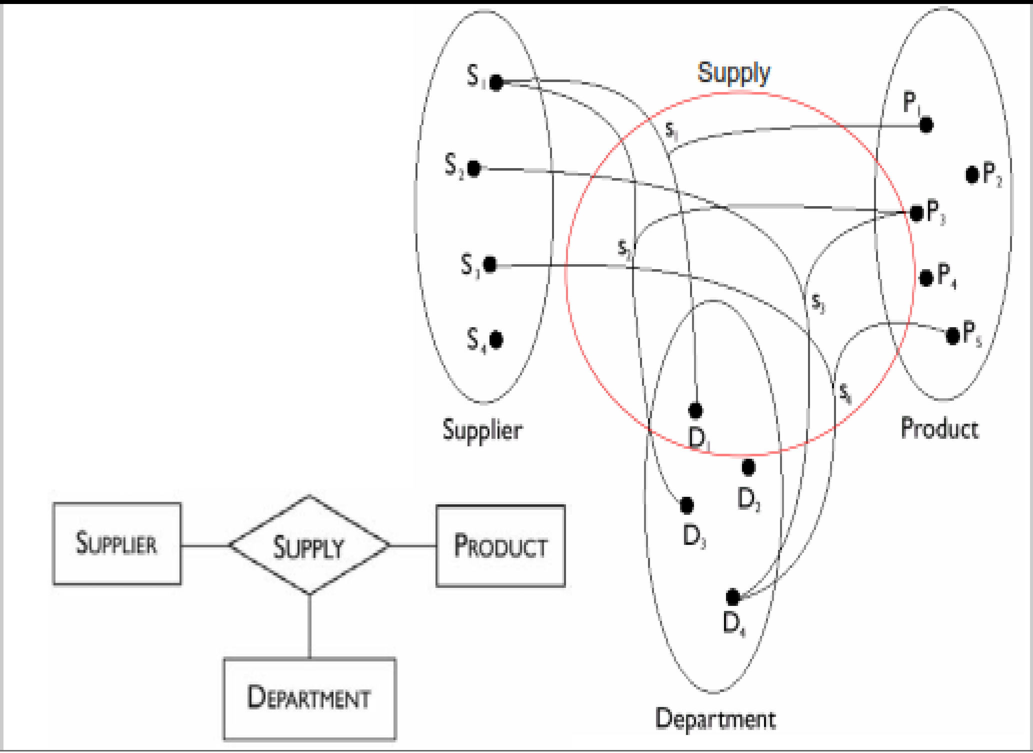
- ***Recursive Relationships***

  An entity can have relationships with itself.



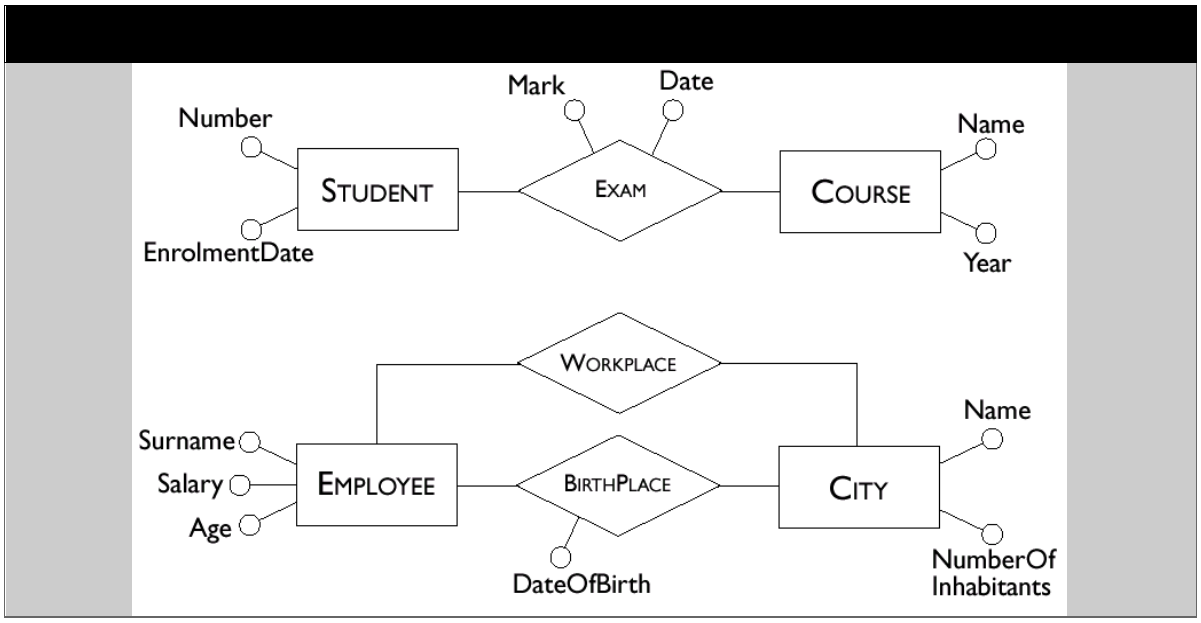


✓ ***Ternary Relationships***

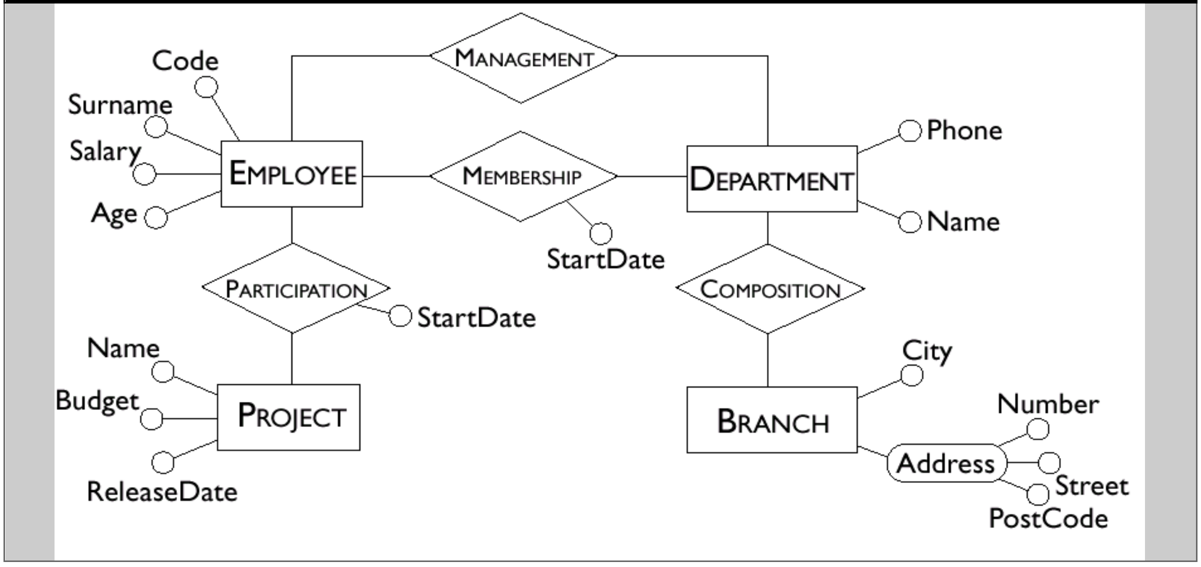


✓ ***Attributes***

Associates with each instance of an entity (or relationship) a value belonging to a set (the domain of the attribute).



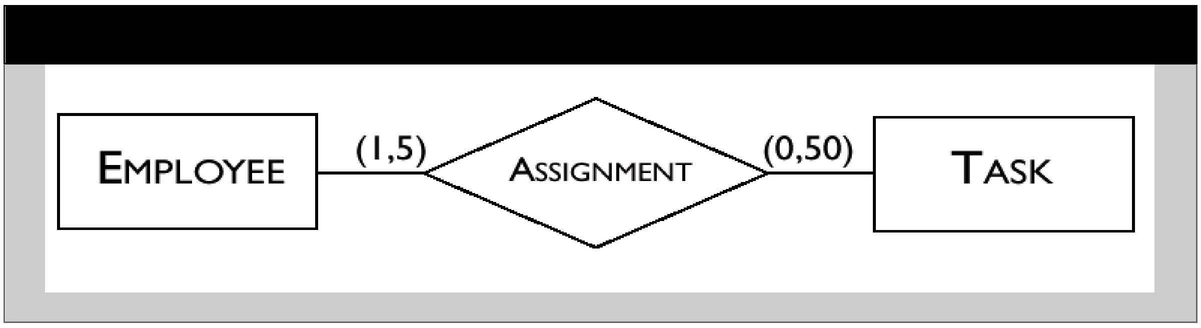
Schema with Attributes



- ✓ ***Cardinalities***

Cardinalities constrain participation in relationships

- ***maximum*** and ***minimum*** number of relationship instances in which an entity
- Instance can participate.

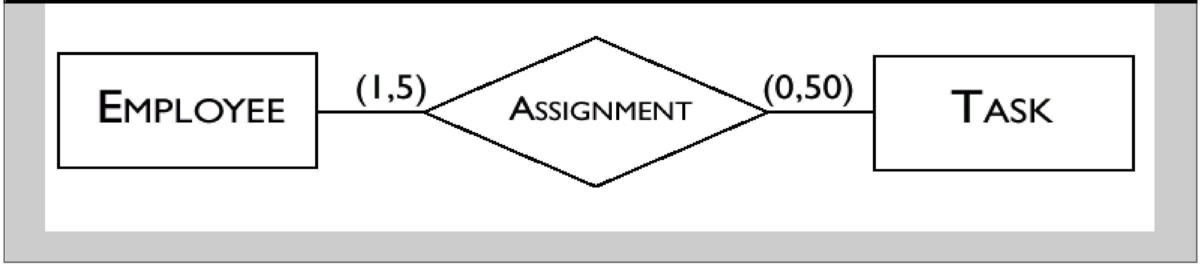


Cardinality is any pair of non-negative integers (a,b)

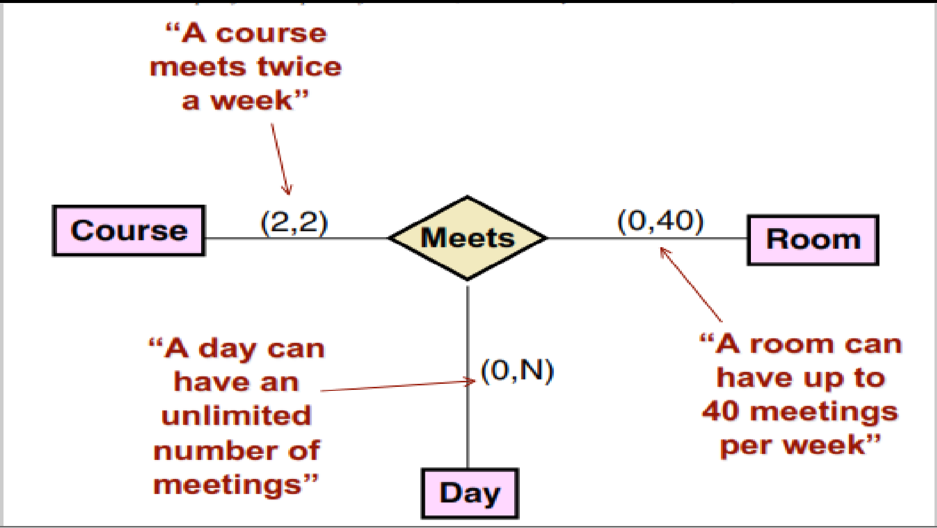
such that a≤b.

- If a=0 then entity participation in a relationship is optional
- If a=1 then entity participation in a relationship is mandatory.
- If b=1 each instance of the entity is associated at most with a single instance of the relationship
- If b="N" then each instance of the entity is associated with an arbitrary number of instances of the relationship.

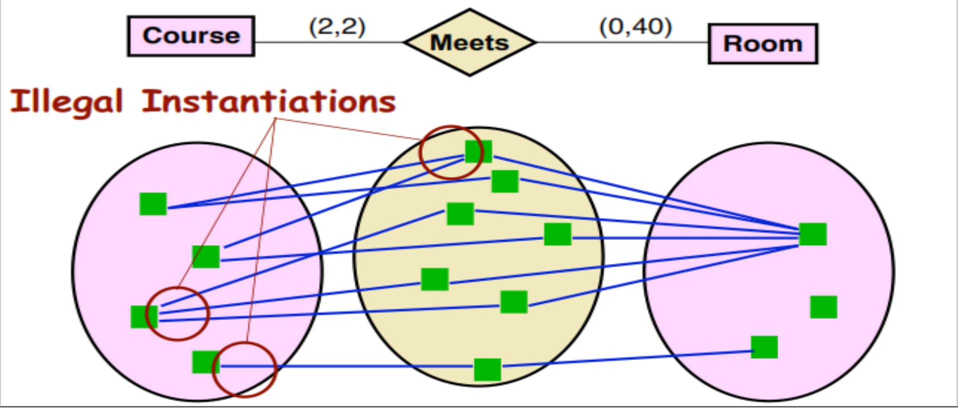
Cardinality Example



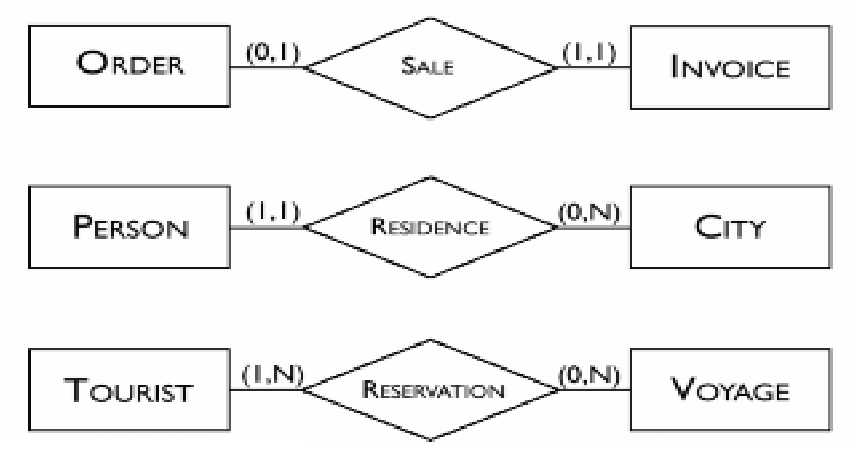
Cardinality Example


"A course meets twice a week"
Course
(2,2)
Meets
(0,40)
Room
"A day can have an unlimited number of meetings"
(0,N)
"A room can have up to 40 meetings per week"
Day

Cardinality Example



Cardinality Example



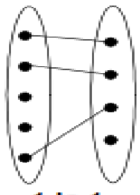
5- Mapping Constraints

An E-R scheme may define certain constraints to which the contents of a database must conform.

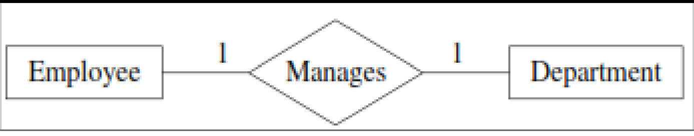
- ✓ ***Mapping Cardinalities*:**
  Express the number of entities to which another entity can be associated via a relationship. For binary relationship sets between entity sets A and B, the mapping cardinality must be one of:

1. One-to-one:

   An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.


1-to-1

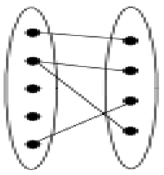
Example



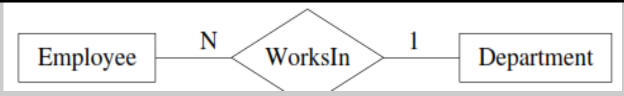
2. (

-to-many:

An entity in A is associated with any number in B. An entity in B is associated with at most one entity in A.

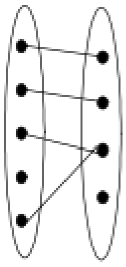

1-to Many

Example

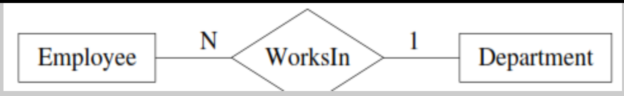


3. Many-to-one:

   An entity in A is associated with at most one entity in B. An entity in B is associated with any number in A.

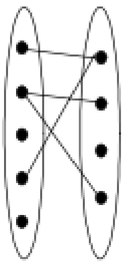

Many-to-1

Example

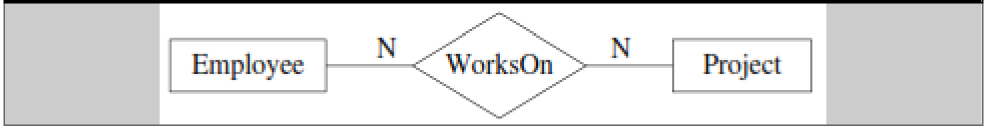


4. Many-to-many:

Entities in A and B are associated with any number from each other.



Example



The appropriate mapping cardinality for a particular relationship set depends on the real world being modeled.

- ✓ ***Existence Dependencies***:
  If the existence of entity X depends on the existence of entity Y, then X is said to be existence dependent on Y. (Or we say that Y is the dominant entity and X is the subordinate entity.)

For example,

- Consider account and transaction entity sets, and a relationship log between them.
- This is one-to-many from account to transaction.
- If an account entity is deleted, its associated transaction entities must also be deleted.
- Thus, account is dominant and transaction is subordinate.

6- Keys

Differences between entities must be expressed in terms of attributes.

- A superkey is a set of one or more attributes which, taken collectively, allow us to identify uniquely an entity in the entity set.
- For example, in the entity set customer, customer-name and S.I.N. is a superkey.
- Note that customer-name alone is not, as two customers could have the same name.

- A superkey may contain extraneous attributes, and we are often interested in the smallest superkey. A superkey for which no subset is a superkey is called a ***candidate key***.
- In the example above, S.I.N. is a candidate key, as it is minimal, and uniquely identifies a customer entity.
- A primary key is a candidate key (there may be more than one) chosen by the DB designer to identify entities in an entity set.

✓ ***Superkeys.***
A superkey is any collection of attributes, which uniquely identifies each entity in an entity set.

✓ ***Candidate Keys***.
A candidate key is the smallest combination of attributes that uniquely identifies an entity in an entity set. Multiple candidate keys can exist in an entity set.

✓ ***Primary Keys***:
A primary key is one of the candidate keys of an entity set, chosen to be the chief means of identifying entities in the database.

Note: Only primary keys of entity sets have graphical representation in E-R diagrams (underlined attributes). However, if other important candidate keys exist in an entity set, they must be identified in the conceptual model.