ADVANCE DATABASE MANAGEMENT SYSTEM

Chapter 2

The Relational Data Models

Bikash Khadka Shah

MCSE, MSDA, OCP, RHCE, RHCVA, CCNA, CEH

9841766620 | 9801076620

Data Models

- Representation of reality is model.
- A collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints is data model.
- **Types of Data Models**
  - High Level- Conceptual Data Model
  - Low Level – Physical Data Model
  - Relational or Representational
  - Object-Oriented Data Model
  - Object-Relational Model

Data Models…

- **High level - Data Model**: User level data model is high level or conceptual model. It provides concepts which are close to way that many users perceive data.

- **Low level-Physical Data Model:** It provides concepts that describe details of how data is stored in computer model. Low-level data model is only for Computer specialists not for end-user.

- **Representation Data Model:** It is between High level & Low level data model which provides concepts that may be understood by end-user but that are not too far removed from way data is organized by within computer.

Schema and Instance

- **Schema:** The overall design of the database is called database schema.
- The database schema is the description of a database specified during database design. It is analogous to type information of variable of a program.
- Physical Schema
  - Database design at a physical level.
  - It describes physical storage of database.
- Logical Schema
  - Database design at logical level.
  - It hides the details of physical storage structure and concentrates on describing entities, data types, relationship etc.
- External View
  - It describes the part of database that a particular user group is interested in and hides the rest of the database from the user group.
- **Instance:** The actual content of database at a particular point in time is instance.

Three Schema Architecture

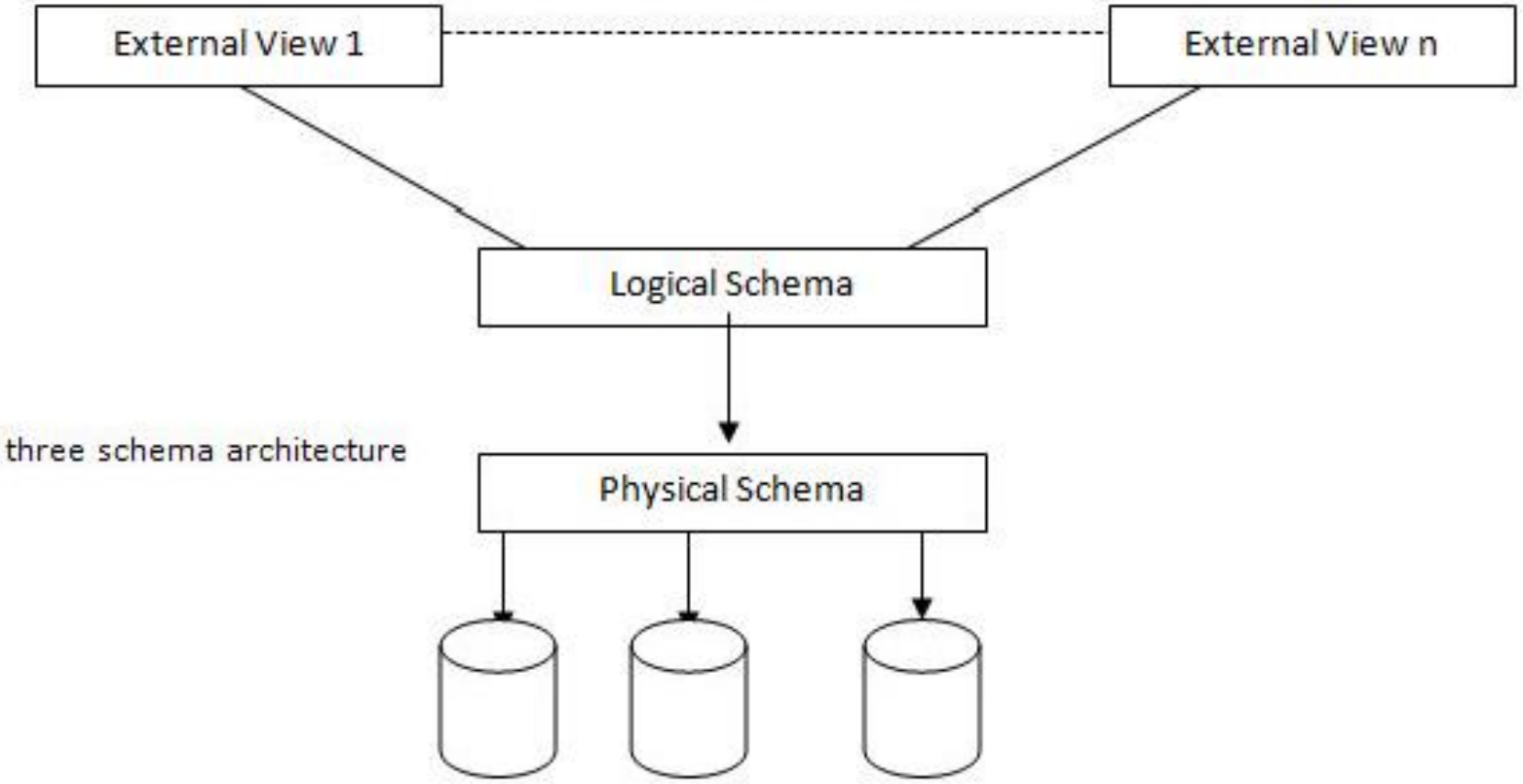


Fig: The three schema architecture

Data Independence

- The ability to modify a schema definition in one level without affecting a schema definition in a higher level is called **data independence**.
- There are two kinds:
  - **Physical data independence**
    - The ability to modify the physical schema without causing application programs to be rewritten
    - Modifications at this level are usually to improve performance
  - **Logical data independence**
    - The ability to modify the conceptual schema without causing application programs to be rewritten
    - Usually done when logical structure of database is altered
- Logical data independence is harder to achieve as the application programs are usually heavily dependent on the logical structure of the data.

Database Languages

- Database Languages are the set of statements, that are used to define and manipulate a database.
- Broadly, a Database language has:
  - Data Definition Language (DDL), which is used to construct a database
  - Data Manipulation Language (DML), which is used to access a database.

Database Languages: DML

- Data Manipulation Language is used to edit the data present in the database.
- It is used to change the data by retrieve, store, modify, delete, insert and update.
- These are the following commands used to manipulate the data in the database.
  - SELECT- This command is used to retrieve data according to a specified condition in the query
  - INSERT- It is used to enter the data into the records of the table.
  - UPDATE- It is used to update the records in the table
  - DELETE- It is used to delete the existing records in the table.

Database Languages: DDL

- Data Definition Language is to either create or modify the table or the database structure.
  - CREATE — It is to create a table and database.
  - ALTER — ALTER is used to alter the data that is present in the records of the table.
  - DROP — This command used to delete the whole table or database at once with data.
  - TRUNCATE — It is used to delete the records from the and to reset the identity value or auto-update value to the initial value.
  - RENAME — It is to used to rename a table.

Concept of Interfaces

- A database management system (DBMS) interface is a user interface which allows for the ability to input queries to a database without using the query language itself.
- The **interfaces** present the user with lists of options (called menus) that lead the user through the formation of a request.

Concept of Interfaces…

- Interfaces provide by DBMS may include the following:
  - Menu-Based Interfaces for Web Clients or Browsing
  - Forms-Based Interfaces
  - Graphical User Interfaces (may contain menu/forms)
  - Natural language Interfaces
  - Speech Input and Output
  - Interfaces for DBA (with privileged commands)

DBMS Users

- Database Administrator
  - A person who has a central control over the system (both the data and program) is called database administrator (DBA).
  - Functions of DBA:
    - Schema Definition
    - Storage Structure and access method definition
    - Schema and physical modification
    - Granting of authorization for data access
    - Routine maintenances

DBMS Users…

- Database Designers
  - The database designer is responsible for defining the detailed database design.
  - He/she is responsible for designing database- specific constructs needed to store, retrieve, and delete persistent objects.

DBMS Users…

- Application Programmers
  - These are the computer professional who write application program to interact database.
- End Users
  - These are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously.
  - E.g.: a bank teller who need to transfer Rs 5000 from account A to account B. Then he/she invokes a program called transfer.

DBMS Users…

- Database system designers and implementers
  - These are the persons who work for overall architecture of the applications
  - Solution Architect
- Tool Developers
  - Develop utility tools for the administrators
- Operators and maintenance personnel
  - Regular Maintenance / Backups
  - DBA responsibilities in the background

Entity Relationship Model (E-R Model)

- It is base on a perception of real world that consists of collection of basic object called entities and relationship among entities.
- An entity is a things or object in the real world i.e. distinguishable from other objects. E.g. each person is entity.
- Each entity is described by the set of attributes.
- Relationship is an association among several entities.
- The set of all entities of same type and set of all relationship of same type are termed as entity set and relationship set respectively.

Entity

- An entity is a real-world thing which can be distinctly identified like a person, place or a concept. It is an object which is distinguishable from others. If we cannot distinguish it from others then it is an object but not an entity. An entity can be of two types:
- **Tangible Entity:** Tangible Entities are those entities which exist in the real world physically. **Example:** Person, car, etc.
- **Intangible Entity:** Intangible Entities are those entities which exist only logically and have no physical existence. **Example:** Bank Account, etc.

Entity...

- In E-R model we don't represent the data but we represent the structure or schema. When we convert E-R model to relational model then data can be stored in tuple or row and hence, represented as an entity.
- **Example:** If we have a table of a Student (Roll_no, Student_name, Age, Mobile_no) then each student in that table is an entity and can be uniquely identified by their Roll Number i.e Roll_no.

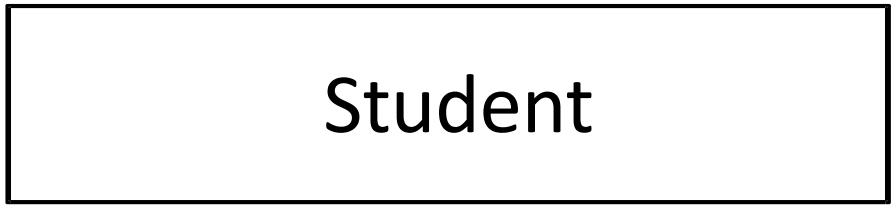
Entity Type

- The entity type is a collection of the entity having similar attributes.
- Entity type is also known as entity set.
- An entity set in an ER diagram is defined by a name, and a set of attributes.

  In the above Student table example, we have each row as an entity and they are having common attributes i.e each row has its own value for attributes Roll_no, Age, Student_name and Mobile_no.
- So, we can define the above STUDENT table as an entity type because it is a collection of entities having the same attributes.

Entity type notation

- Entity type is represented as a named rectangle in ER Diagram.
- We use a rectangle to represent an entity type in the E-R diagram, not entity.
- The E-R representation of the above Student Entity Type is:



Attributes

- Each entity has attributes - the particular properties that describe it.
- A particular entity will have a value for each of its attributes.
- The attribute values that describe each entity become a major part of the data stored in the database.
- Attribute sets are represented by named oval and connected to their entity set via a straight line.
- Several types of attributes occur in the ER model:
  - *simple* versus *composite*
  - *single-valued* versus *multi-valued*
  - *stored* versus *derived*

Key Attributes

- A key attribute is the unique characteristic of the entity.
- In ER diagram, key attribute is represented by an oval with underlying lines in the attribute name.
- Composite Keys are the key values generated from two or more attributes of an entity.
- Attributes that are not unique (attributes other than Key Attribute) are known as non key attributes.
- The non key attribute name is not underlined.

Types of Keys

- Super key
  - It is the set of one or more attributes whose combine value uniquely identifies the entities in the entity set. E.g. {empno, ename, address} can be considered as a super key. If we assume that there are no two employees of the same name and address then the set {ename, address} is another super key.
- Candidate key
  - A candidate key is a minimal super key i.e. a super key which does not have any proper subset which is also a super key. E.g. {empno} and {ename, address} are two candidate key.

Types of Keys...

- Primary key
  - It is a candidate key i.e. chosen by a database designer as a principle means of uniquely identifies entities within the entity set. E.g. The candidate key {empno} can be considered to be the primary key.
- Composite key
  - In some entity set a single attribute can't be use to uniquely identify entities. In that case we have to use two or more attributes to uniquely identify entities within the entity set. When primary key contains sets o two or more than two attributes, it is called a composite primary key. E.g. {ename, address} can be a composite key, if ename and address of two employees are not same.

Relationships, Relationship Sets, Roles

- A relationship is an association between several entities.
- A relationship set is a set of relationships of the same type.
- A relationship set is denoted by named diamond and is connected to associated entity sets via straight lines.
- A relationship can exist among the entities of same entity types; the role of an entity is the function it plays in a relationship.
- A relationship may also have descriptive attributes

Degrees of relationship: Unary

- Also called a recursive relationship. It is a relationship between instances of one entity type.

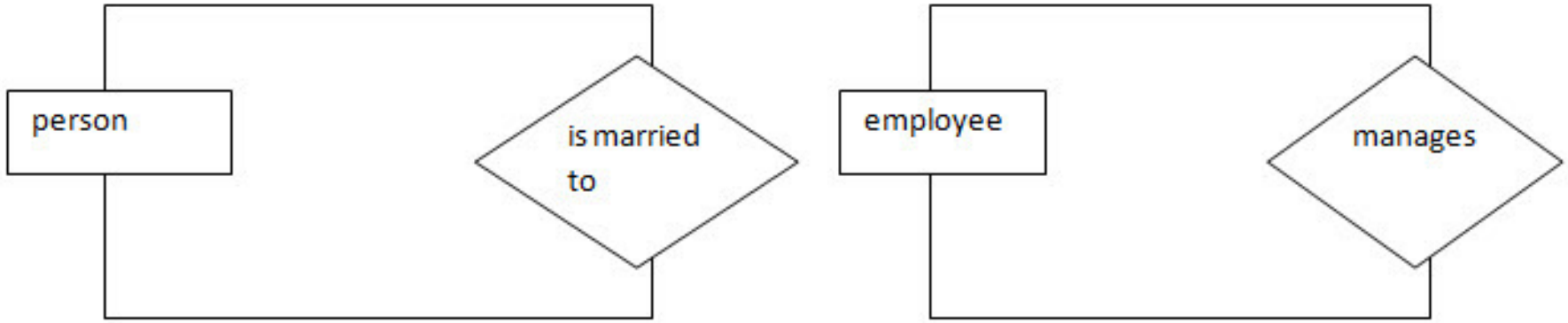


Fig : unary relationship

Degrees of relationship: Binary

- It is a relationship between instances of two entity types and is the most common encountered in data modeling.

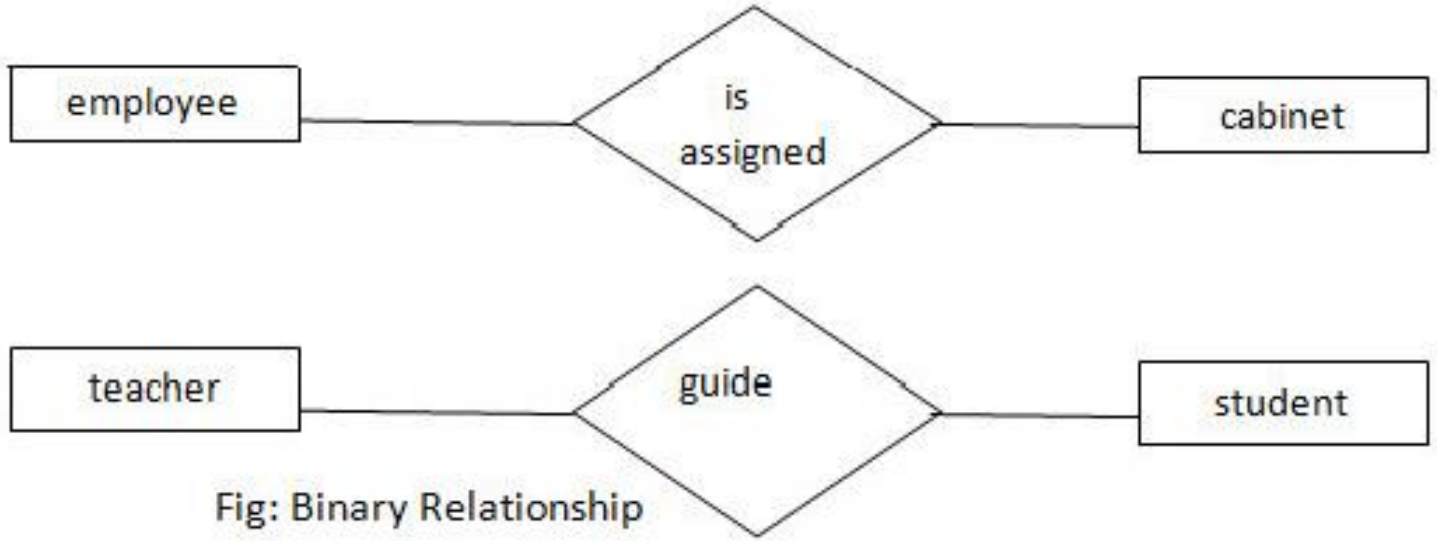
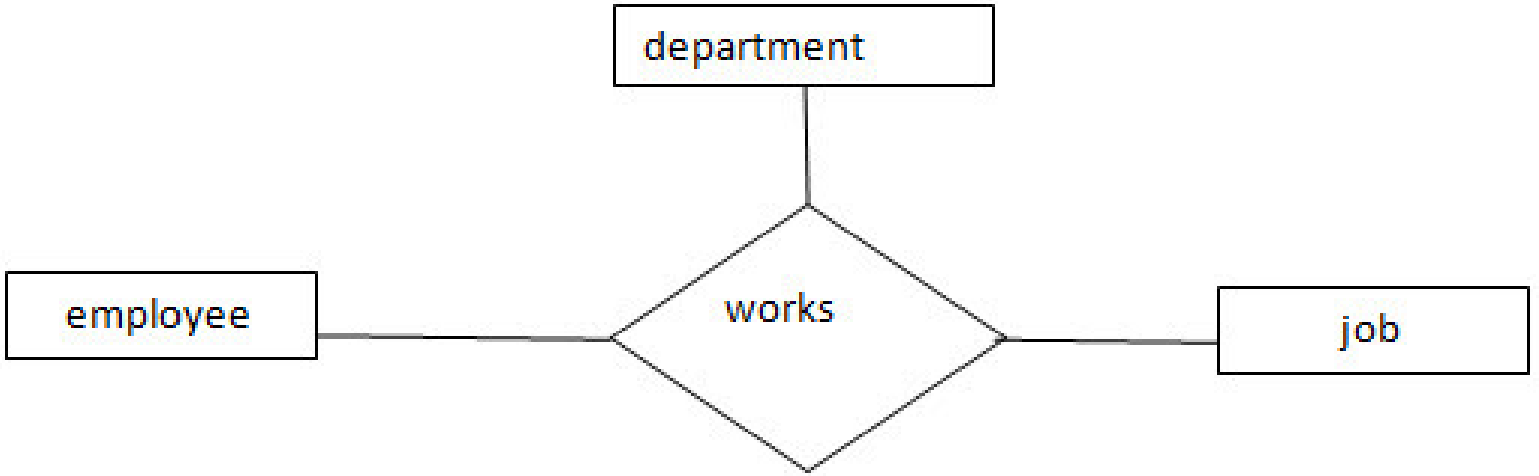


Fig: Binary Relationship

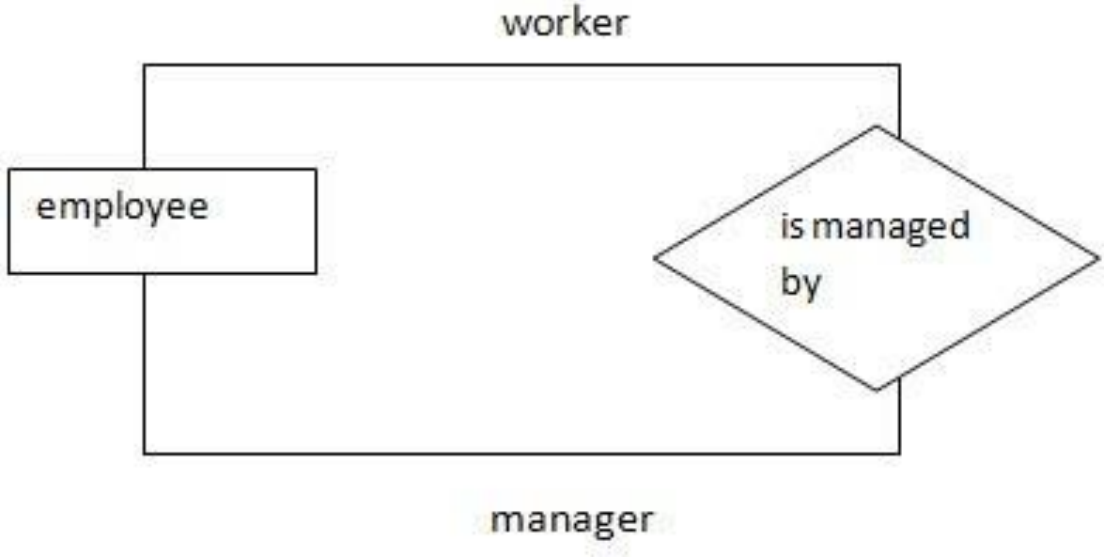
Degrees of relationship: Ternary

- The relationship that involves three entities is called ternary relationship.



Roles in relationship

- If the relationship involves single entities playing different role such are called roles in relationship.
- Role is indicated in E-R diagram by labeling the line.
- E.g.: worker and manager both are employee but are playing different roles.

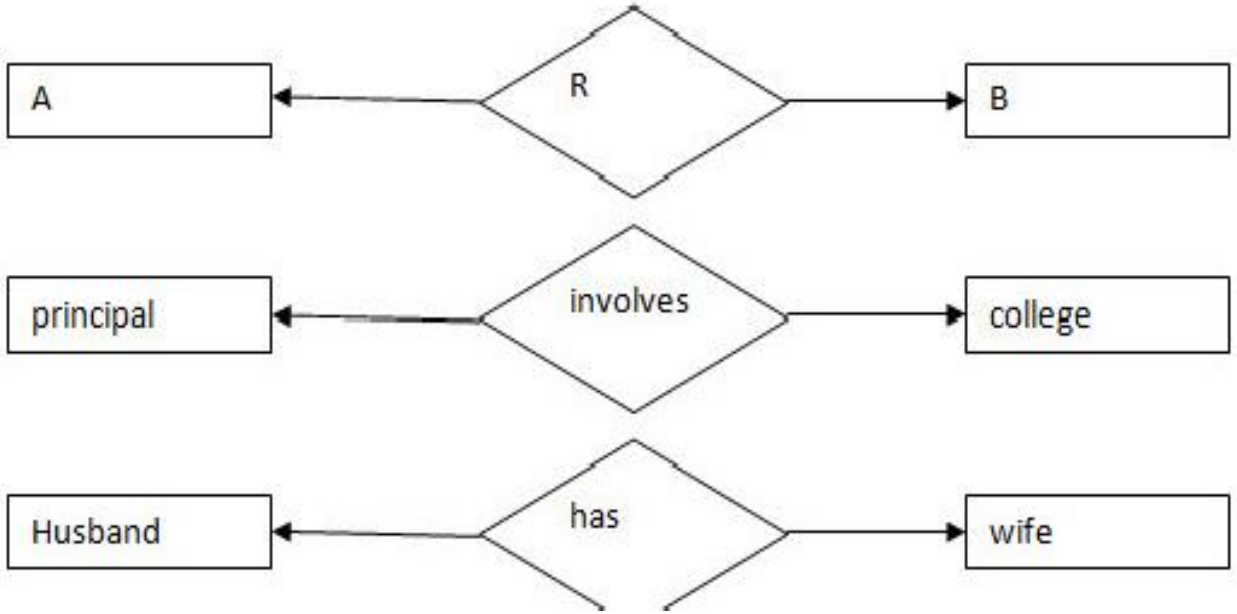


Mapping cardinalities

- It expresses the number of entity to which another entity can be associated through a relationship set.
- Types of mapping cardinalities:
  - One to one relationship
  - One to many relationship
  - Many to one relationship
  - Many to many relationship

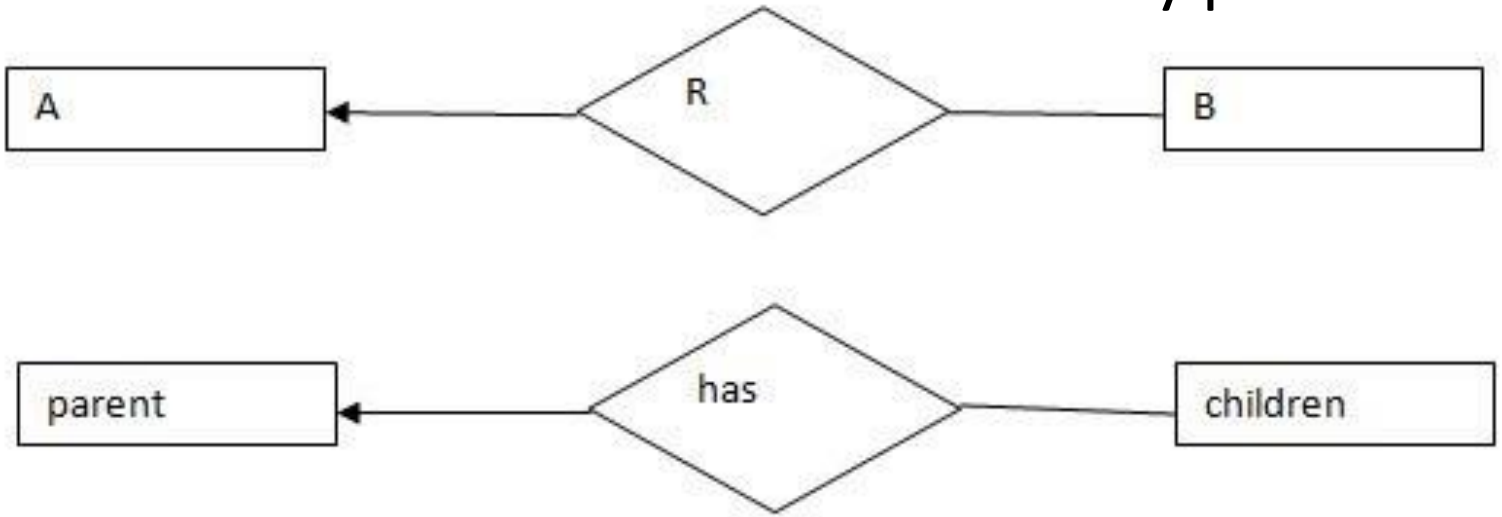
One-to-one relationship

- If an entity set A can relate at most one entity in entity set B and entity in entity set B can relate with at most one entity in entity set A then such relationship is one-to-one relationship.



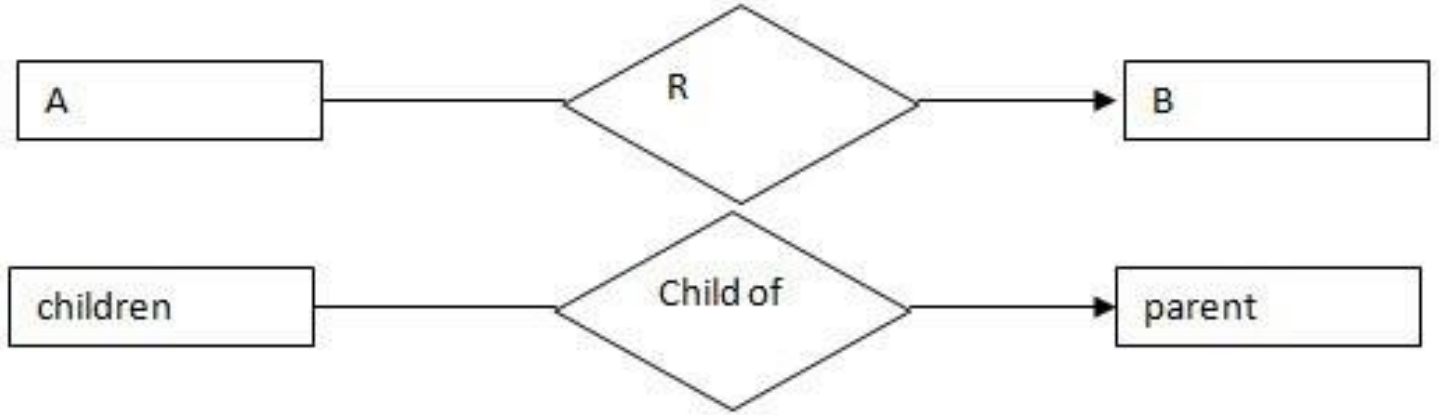
One-to-many relationship

- If an entity set A can relate with more than one entity in entity set B and entity in entity set B can relate with at most one entity in entity set A then such relationship is one-to-many relationship. E.g.: a parent can have many children but the children can't have many parents.



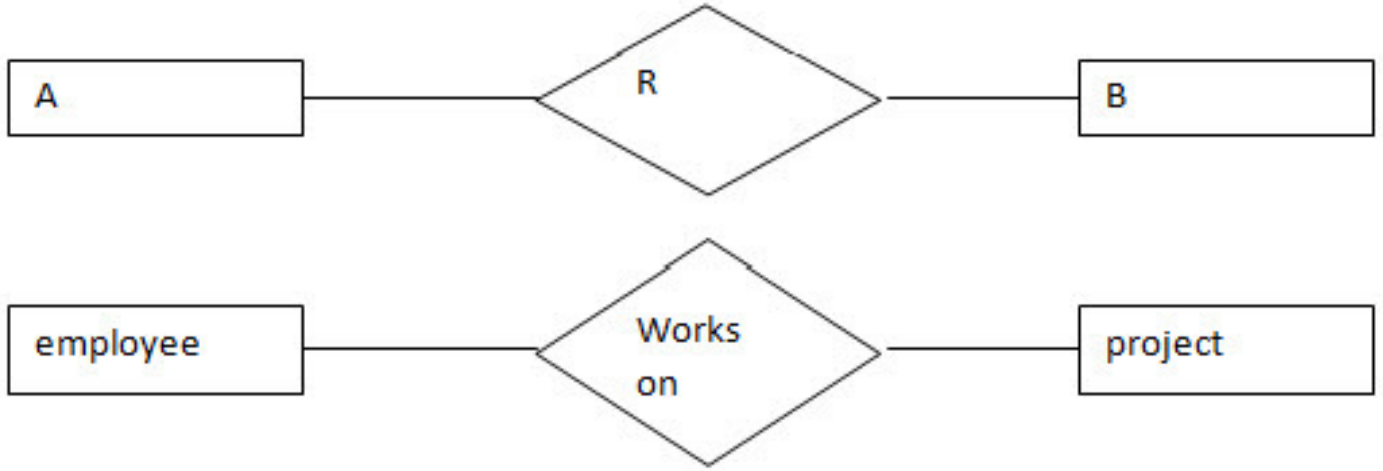
Many-to-one relationship

- If an entity set A can relate with at most one entity in entity set B and entity in entity set B can relate with more than one entity in entity set A then such relationship is many-to-one relationship.



Many-to-many relationship

- If an entity set A can relate with more than one entity in entity set B and entity in entity set B can relate with more than one entity in entity set A then such relationship is many-to-many relationship.



Creating an ERD

- Input to the ER Modeling is requirement story
- The output is ER Diagram
- Steps included in ER Modeling:
  - Entity Identification
  - Relationship Identification
  - Cardinality Identification
  - Attribute Identification
  - Create ER Diagram

Creating an ERD…

- Requirement Story (for an University system):
  - In a university, a Student enrolls in Courses. A student must be assigned to at least one or more Courses. Each course is taught by a single Professor. To maintain instruction quality, a Professor can deliver only one course

Creating an ERD…

- Step 1: Entity Identification
- Entities are generally nouns which represent physical or conceptual real world objects
- We have three entities
  - Student
  - Course
  - Professor

Creating an ERD…

- Step 2: Relationship Identification
- Relationships are generally verbs which represent association between entities
- We have the following two relationships
  - The student is **assigned** a course
  - Professor **delivers** a course

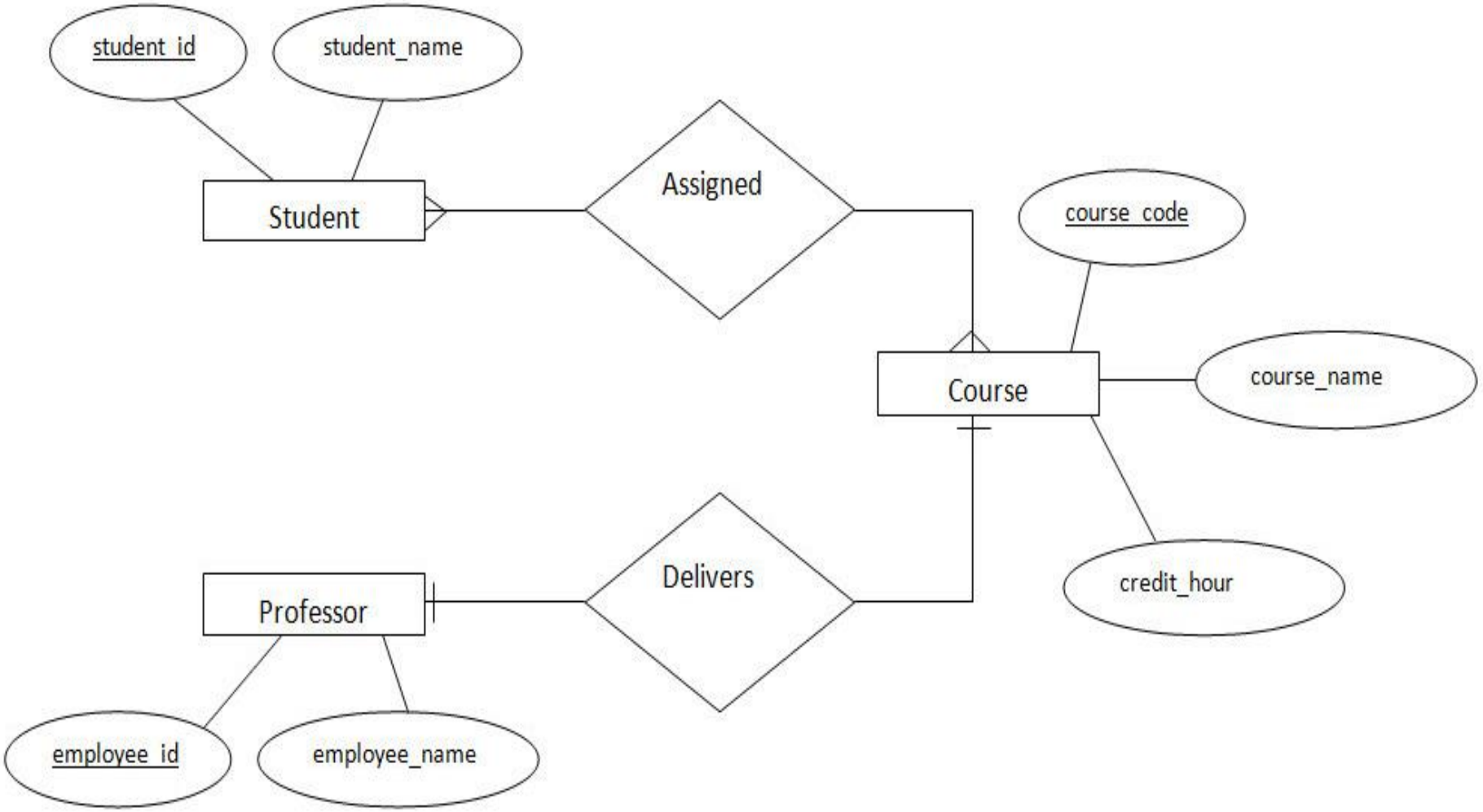
Creating an ERD…

- Step 3: Cardinality Identification
- For them problem statement (requirement story) we know that,
  - A student can be assigned **multiple** courses
    - Student and Course have many to many relationship
  - A Professor can deliver only **one** course
    - Professor and Course have one to one relationship

Creating an ERD…

- Step 4: Attribute Identification
- In this step we should list all the key as well as non key attributes.
- Key attributes:
  - Student: student_id
  - Course: course_code (like CSC260)
  - Professor: employee_id
- Non key attributes can be defined as per the business requirements.

Creating an ERD…



Creating an ERD…

- Best Practices for Developing Effective ER Diagrams
  - Eliminate any redundant entities or relationships
  - Make sure that all your entities and relationships are properly labeled
  - Make sure that the ER diagram supports all the data you need to store
  - Assure that each entity only appears a single time in the ER diagram
  - Name every relationship, entity, and attribute
  - Never connect relationships to each other
  - May use colors to highlight important portions of the ER diagram