Chapter 2: Data Models

Data Model

- It is a collection of conceptual tools for describing data, data relationship, data semantics and consistency constraints.
- It provides a way to describe the design of a database at physical , logical and view level.
- It is a Communications tool to facilitate interaction among the designer, the applications programmer, and the end user
- It is relatively simple representation, usually graphical, of complex real-world data structures.
- Good database design uses an appropriate data model as its foundation

Importance of Data Modeling

- End-users have different views and needs for data
- Data model organizes data for various users

Data Model Basic Building Blocks

- Entity is anything about which data are to be collected and stored
- Attribute is a characteristic of an entity
- Relationship describes an association among (two or more) entities
  - **One-to-many (1:M) relationship**
  - **Many-to-many (M:N or M:M) relationship**
  - **One-to-one (1:1) relationship**

Business Rules

- Brief, precise, and unambiguous description of a policy, procedure, or principle within a specific organization's environment
- Apply to *any organization that stores and* uses data to generate information
- Description of operations that help to create and enforce actions within that organization's environment
- Describe characteristics of the data *as viewed by the company*
- Must be rendered in writing
- Must be kept up to date
- Sometimes are external to the organization
- Must be easy to understand and widely disseminated

Data Models

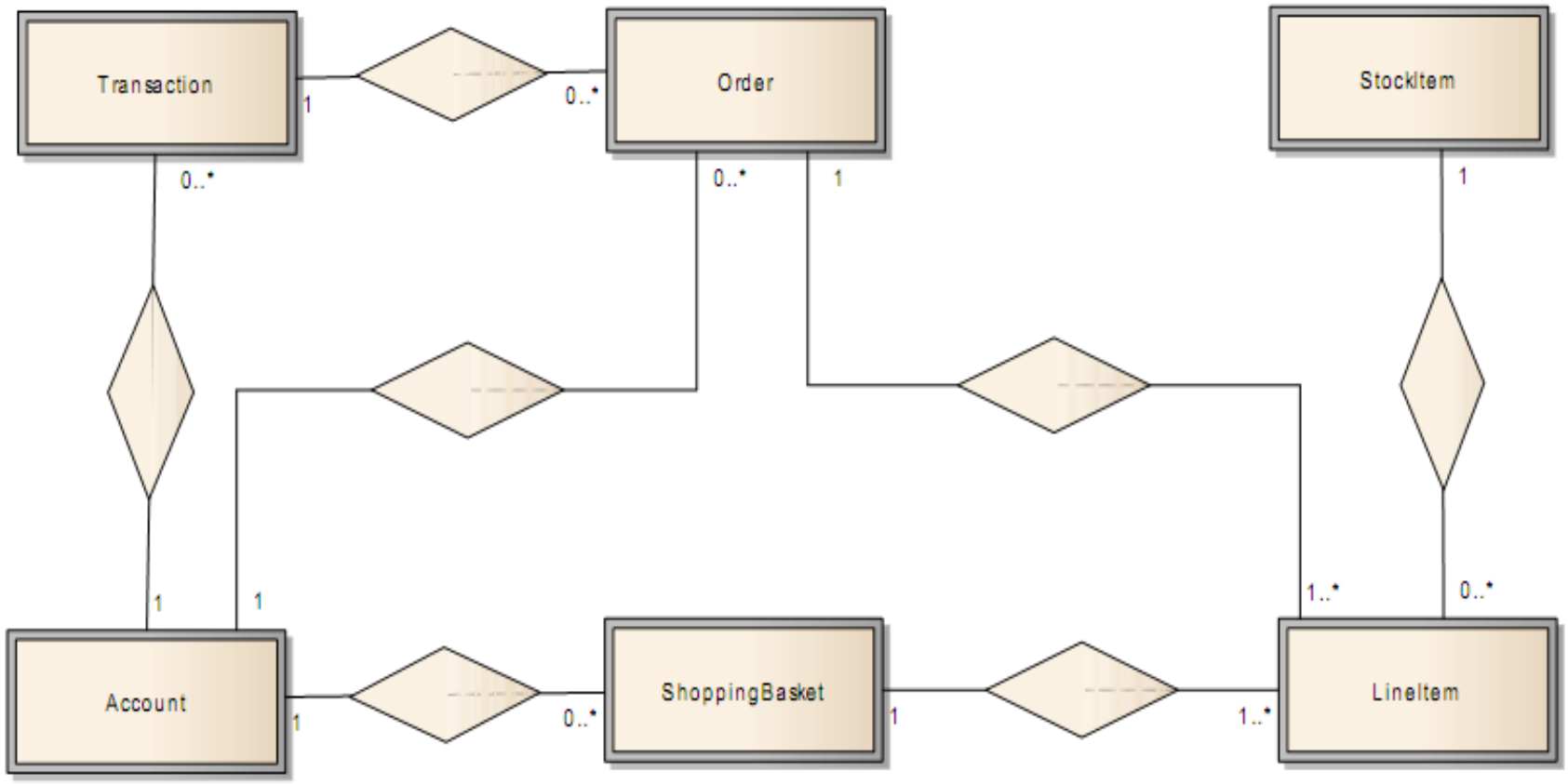
On the basis of DBMS architecture:

- Conceptual Model
- Logical Model
- Physical Model

Conceptual Model

- The purpose of a Conceptual model is to simply establish the Entities and their 'high-level' relationships.
- When modeling using UML, the Domain Model is used to define the initial structural layout (later to be used for Classes). Where the Class design is parallel to the data structure design, it is sensible to use the Domain model as a seed for the Conceptual model.
- At the conceptual level *there is little detail.* The diagrams consist basically of Entities and their simple relationships. If there are Attributes defined, these are loosely typed (for example - no length settings), and connectors between Entities do not define relationships to specific Attributes

Figure below is an example of a simple Conceptual diagram for an Online Bookstore.



Logical Model

❖The Logical model includes more detail, specifically Attributes, but the modeling is still generic as it is not bound to a specific DBMS. The process of creating a Logical model based on a Conceptual model involves:

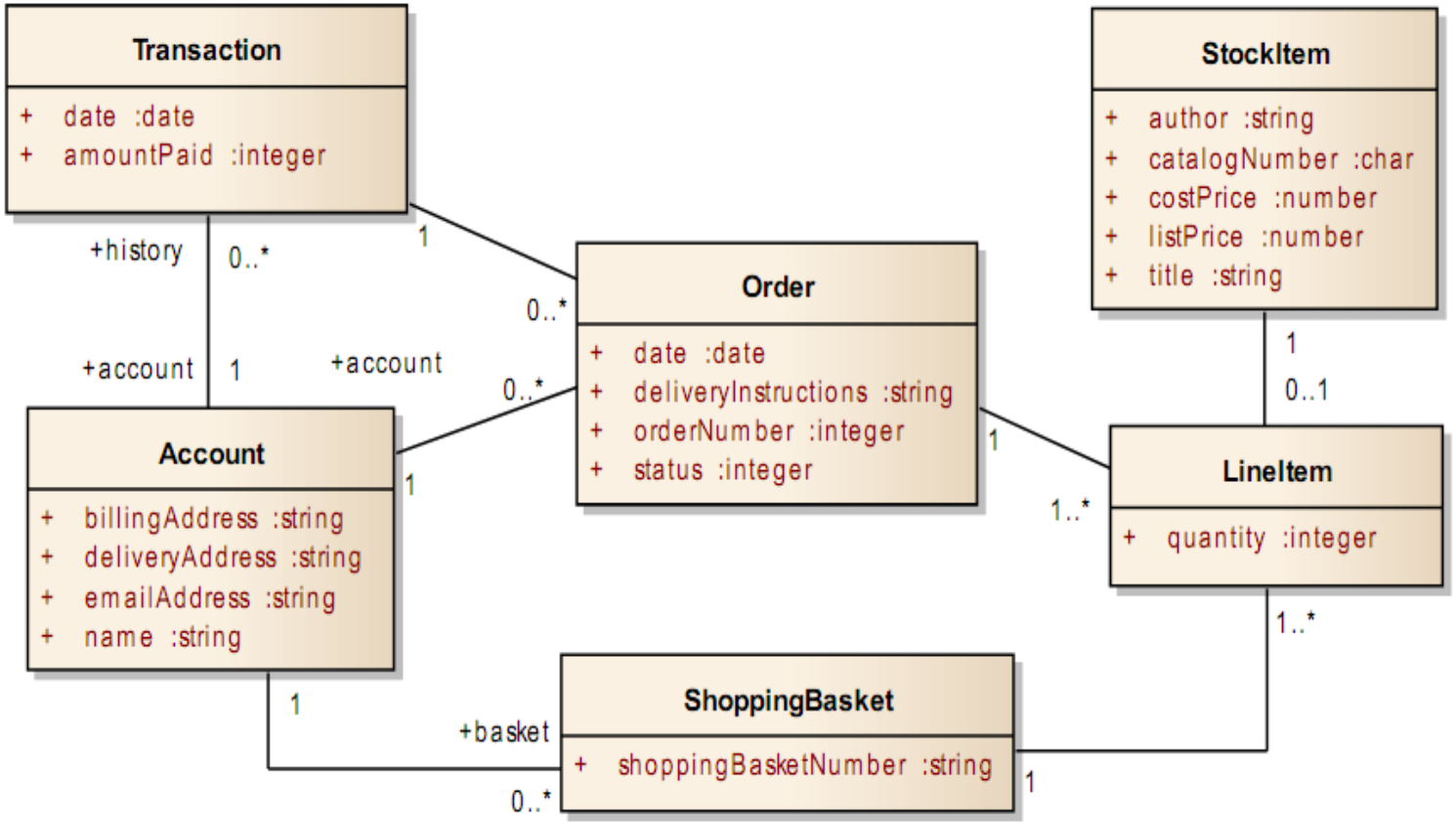
- Setting the Attributes

-At the Logical level, the attributes (which later become Table Columns), are modeled independently of any DBMS product. They are typed using primitive UML data types, such as integer, Boolean and string etc.

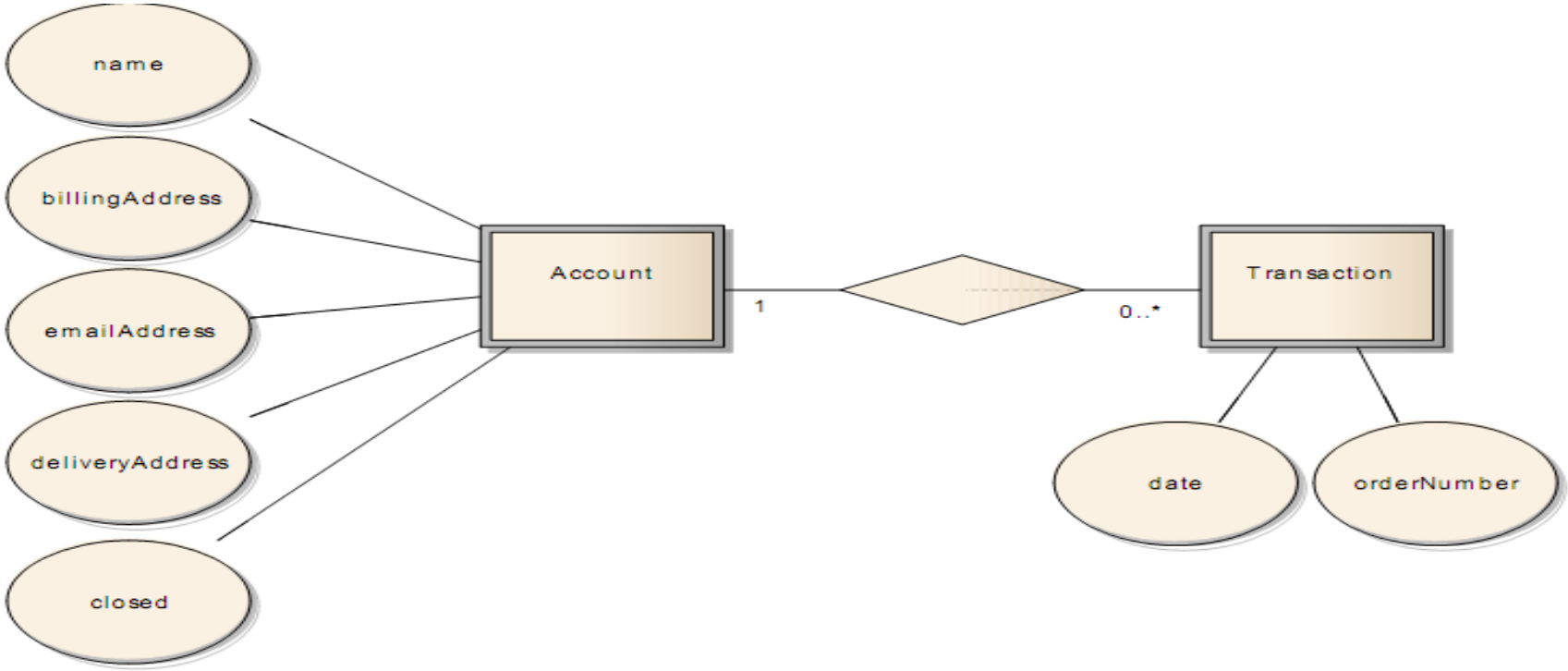
- Setting the Relationships

-At the Logical level we do not yet set the Primary Keys & Foreign Keys etc. At this level it is best to verify and adjust the Connector 'multiplicity' (also known as 'cardinality' in database terminology) details that were set earlier for relationships in the Conceptual model.

Figure below is a **diagram of the Logical model** derived from the Conceptual model



-an equivalent model of the Account and Transaction elements using an Entity Relationship Diagrams (ERD). The Entities do not contain attributes, rather the 'table-columns' are represented as ellipses connected to the Entity. The Entity relationships are represented as diamond-shape connectors.

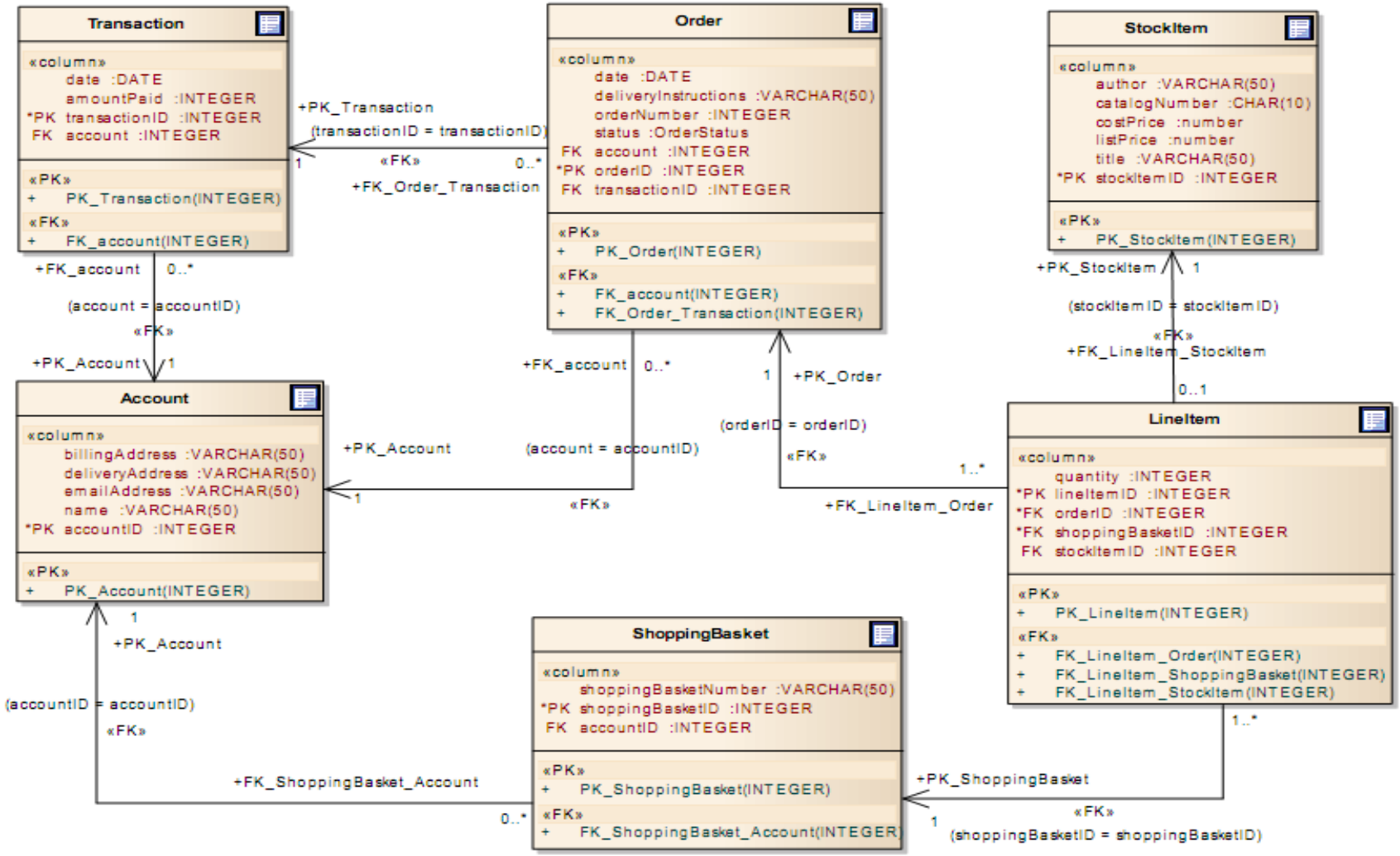


Physical Model

- Modeling on the Physical level involves adding “platform specific” detail to the model. That is, detail specific to the DBMS where the database is to be deployed.
- To set up the Physical model, create a copy of the Logical Model and start the process of adding the Physical definitions to this model.
- The key aspects of this are:

  - For each ‘Class’:
    - Construct a table
    - The Database setting must be set to a specific DBMS
    - Update the Attributes to reflect Columns ‘Typed’ to the specific DBMS Field types.
  - Add more detail to the Connectors (relationships), to define the Primary Key (&Foreign Key) linking.

❖Figure below shows the **Physical model** derived from the above Logical model for a specific DBMS.



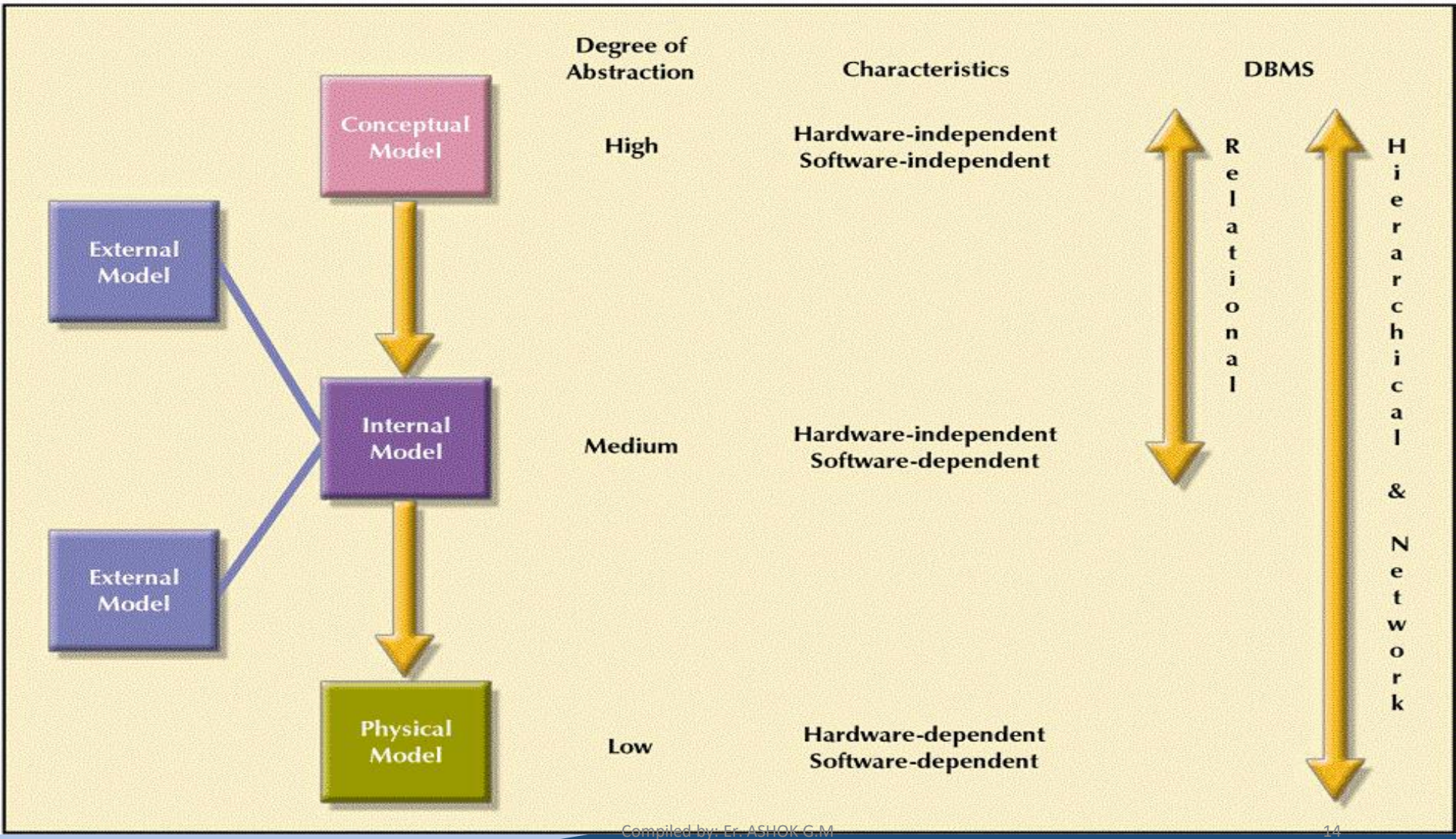
Data models categorization

- American National Standards Institute/Standards Planning and Requirements Committee (ANSI/SPARC)

–Classified data models according to their degree of abstraction (1970s):

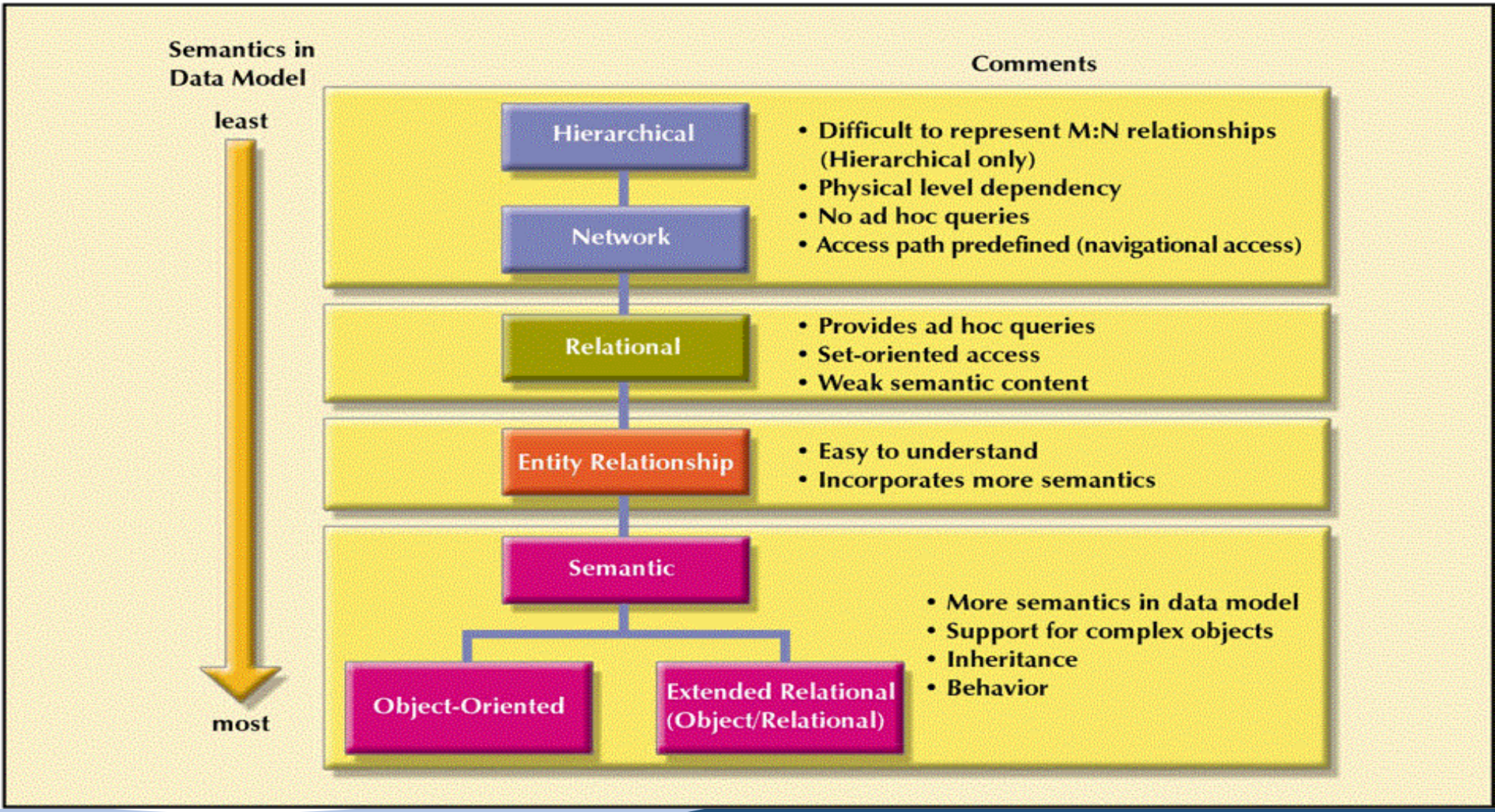
- Conceptual Model
- External Model
- Internal Model
- Physical Model

FIGURE 2.10 DATA ABSTRACTION LEVELS



Evolution of Data Models

FIGURE 2.9 THE DEVELOPMENT OF DATA MODELS



- Hierarchical Data Model
- Network Data Model
- Relational Data Model
- Entity relationship Data Model
- Semantic Data Model
  - Object oriented Data Model
  - Object/ relational Data Model

Hierarchical Data Model

- This is one of the oldest models in a data model which was developed by IBM, in the 1950s.
- In a hierarchical model, data are viewed as a collection of tables, or we can say segments that form a hierarchical relation.
- Data is organized into a tree-like structure where each record consists of one parent record and many children.
- In the hierarchical model, segments pointed to by the logical association are called the **child segment** and the other segment is called the **parent segment**.
- If there is a segment without a parent is then that will be called the **root** and the segment which has no children are called the **leaves**.
- Hierarchical models are also commonly used as physical models because of the inherent hierarchical structure of the disk storage system like tracks, cylinders, etc. There are various examples such as **Information Management System (IMS) by IBM, NOMAD by NCSS, etc.**

Hierarchical Data Model

- Characteristics
- Basic concepts form the basis for subsequent database development
- Limitations lead to a different way of looking at database design
- Each parent can have many children
- Each child has only one parent

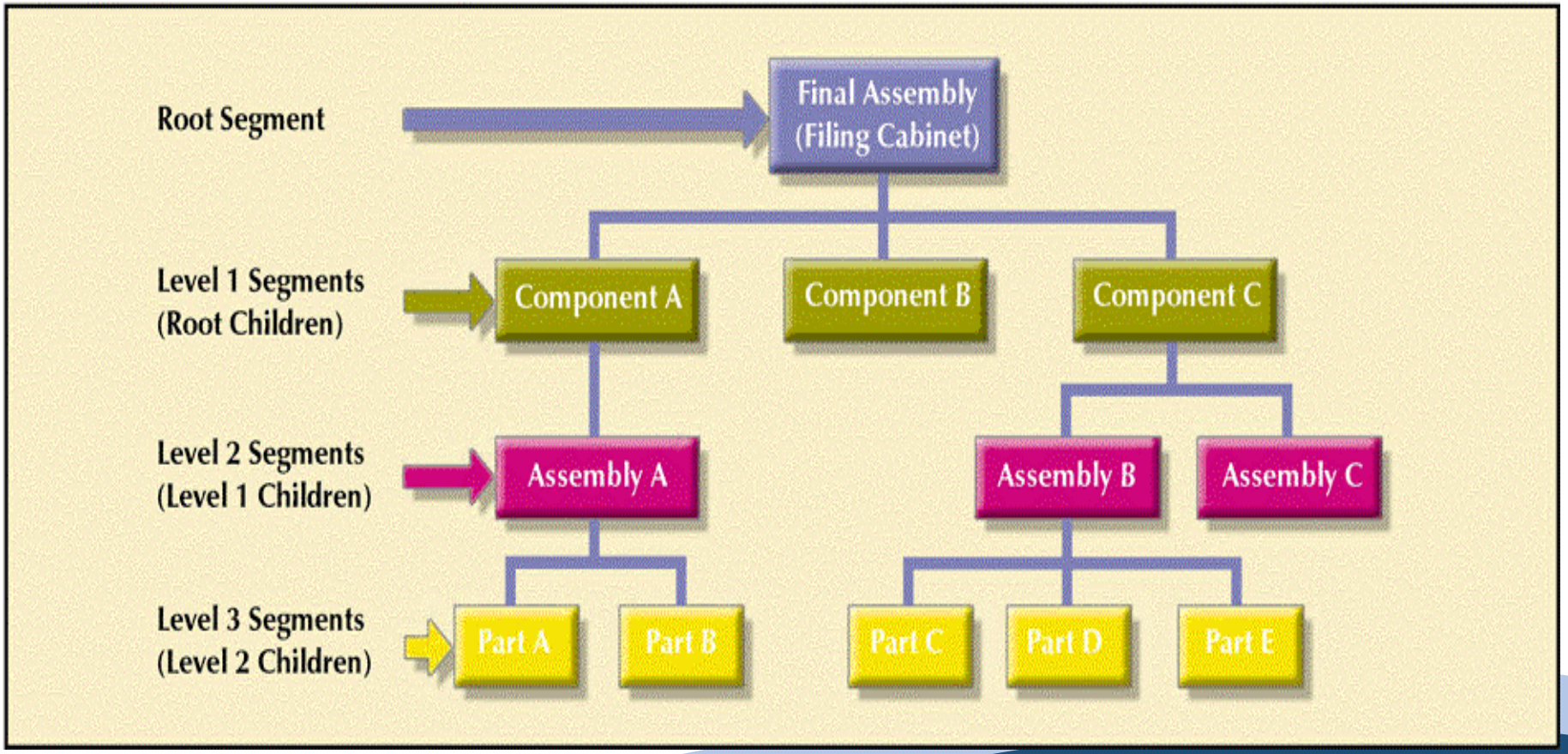
Advantages

-Conceptual simplicity
–Database security
–Data independence
–Database integrity
–Efficiency

Disadvantages

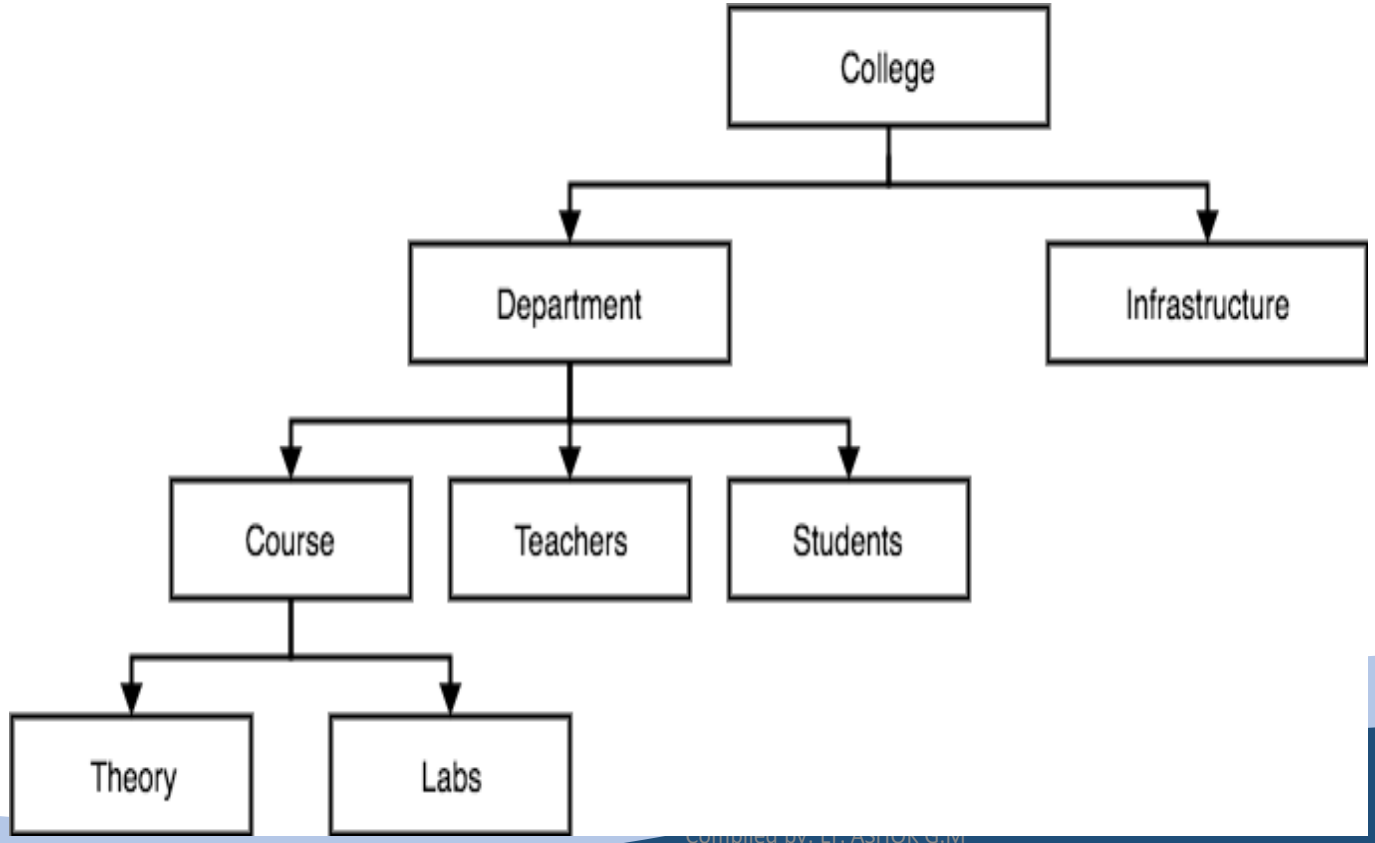
-Complex implementation
–Difficult to manage
–Lacks structural independence
–Complex applications programming and use
–Implementation limitations
–Lack of standards

FIGURE 2.1 A HIERARCHICAL STRUCTURE



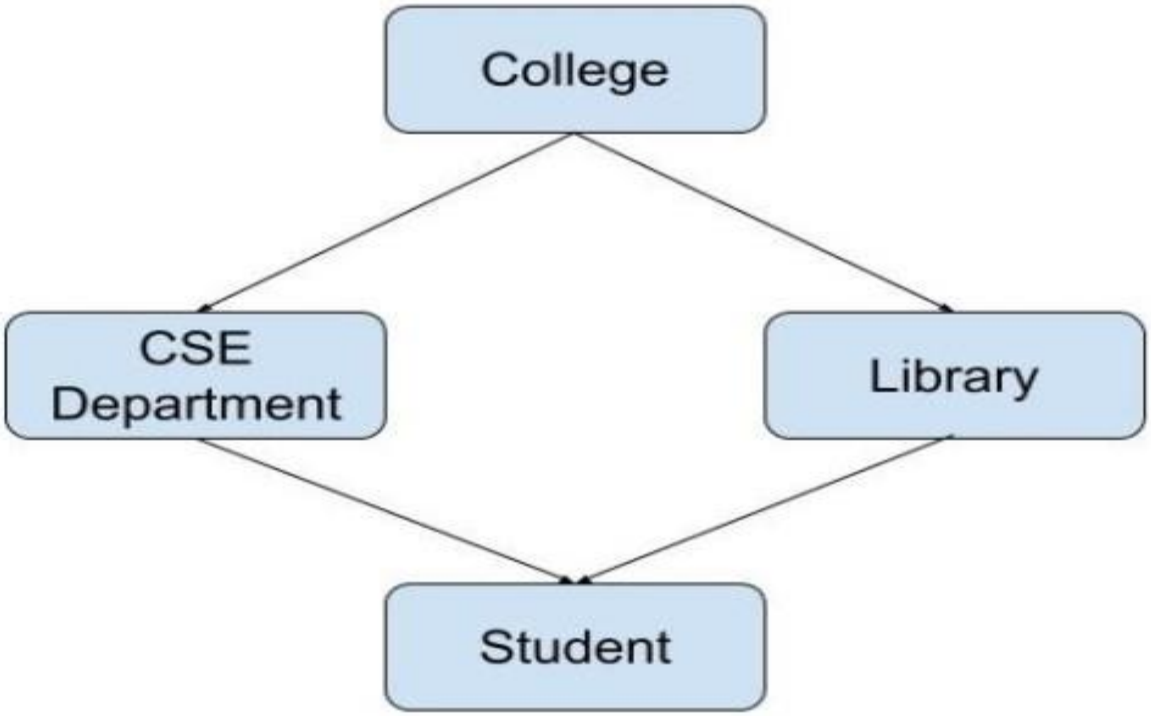
Example: **Hierarchical Database**

- each node represents a logical record type and is displayed by a list of its fields.
- The child node represents a set of records that are connected to each record of the parent type, which is due to a many-to-many relationship is from child to parent.



Network Model

- This model was formalized by the Database Task group(DBTG) in the 1960s. This model is the generalization of the hierarchical model.
- This model can consist of **multiple parent segments** and these segments are grouped as levels. Mostly, there exists a many-to-many logical association between any of the two segments.
- This model replaces the hierarchical tree with a graph-like structure, and with that, there can be more general connections among different nodes. It can have M: N relations i.e, many-to-many which allows a record to have more than one parent segment.
- Here, a relationship is called a set, and each set is made up of at least 2 types of record which are given below:
  - An **owner record** that is the same as of parent in the hierarchical model.
  - A **member record** that is the same as of child in the hierarchical model.



Network Model

Relational Model

- Developed by Edgar Frank Codd (IBM) in 1970
- Considered ingenious but impractical in 1970
- Conceptually simple
- Computers lacked power to implement the relational model
- Today, microcomputers can run sophisticated relational database software

Basic Structure

- Relational Database Management System (RDBMS)
- Performs same basic functions provided by hierarchical and network DBMS systems, plus other functions
- Most important advantage of the RDBMS is its ability to let the user/designer operate in a human logical environment

- Table (relations) –Matrix consisting of a series of row/column intersections
  - –Related to each other by sharing a common entity characteristic
- Relational schema
  - –Visual representation of relational database's entities, attributes within those entities, and relationships between those entities

Relational Table

- Stores a collection of related entities –Resembles a file
- Relational table is purely logical structure
  - –How data are physically stored in the database is of no concern to the user or the designer
  - –This property became the source of a real database revolution

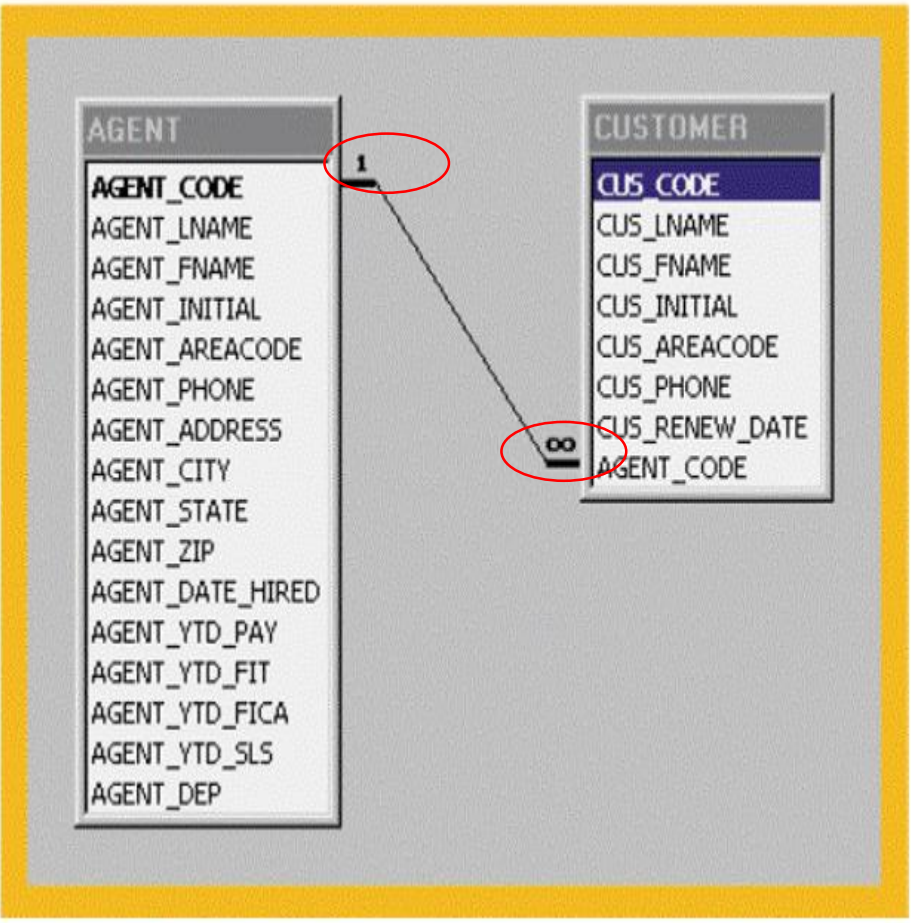


Fig: A relational Schema

Database name: Ch02_InsureCo 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

Fig: Linking relational Tables

Advantages

–Structural independence

–Improved conceptual simplicity

–Easier database design, management ,implementation, and use

–Ad hoc query capability

–Powerful database management system

Disadvantages

–Substantial hardware and system software overhead

–Can facilitate poor design and implementation

–May promote “islands of information” problems

Entity Relationship Model(ER Model)

- Widely accepted and adapted graphical tool for data modeling
- Introduced by Chen in 1976
- Graphical representation of entities and their relationships in a database structure

Basic Structure

- Entity relationship diagram (ERD)
  - –Uses graphic representations to model database components
  - –Entity is mapped to a relational table
- Entity instance (or occurrence) is row in table
- Entity set is collection of like entities
- Connectivity labels types of relationships
  - –Diamond connected to related entities through a relationship line

Figure: Basic Chen ERD notation

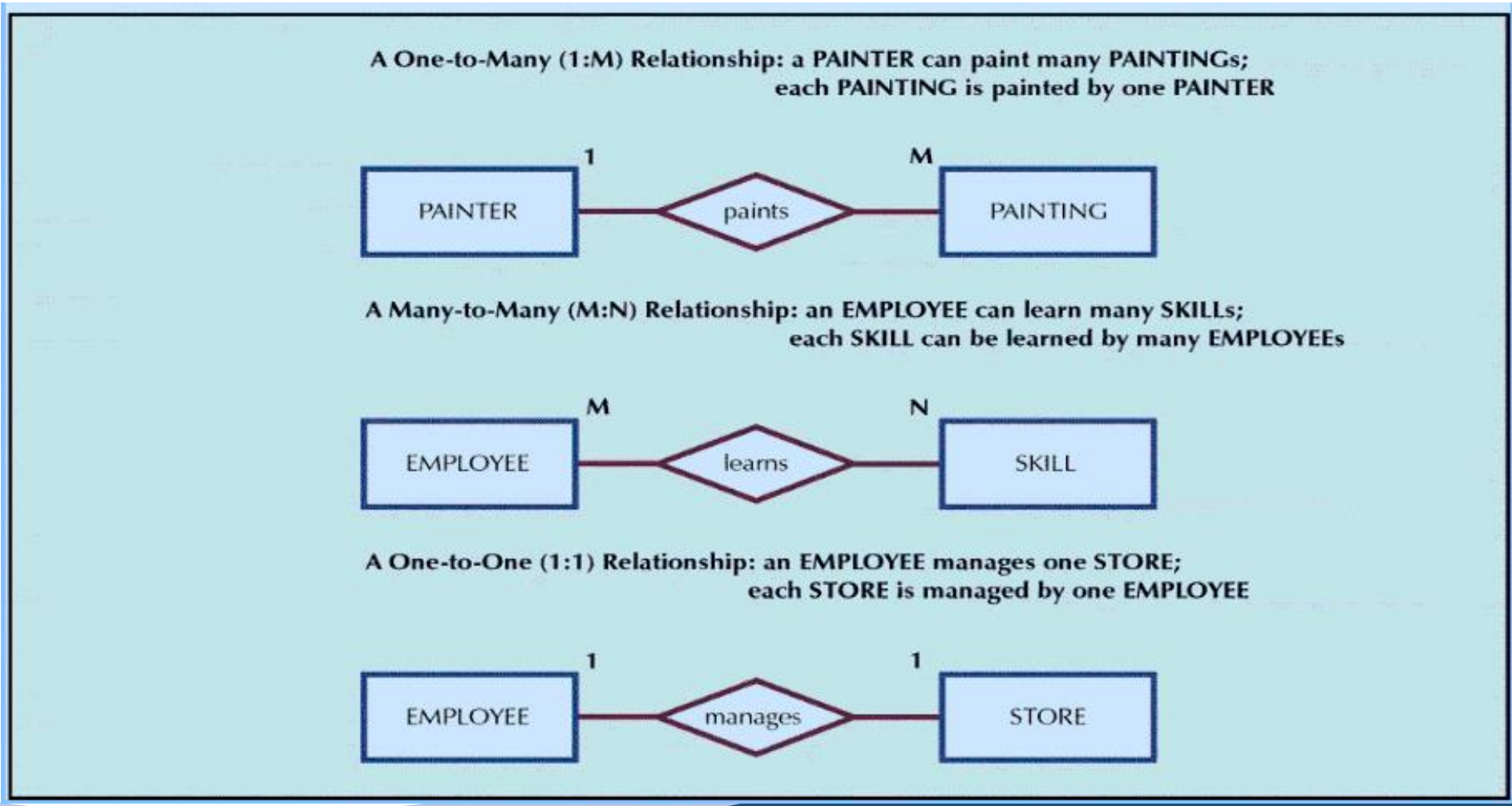
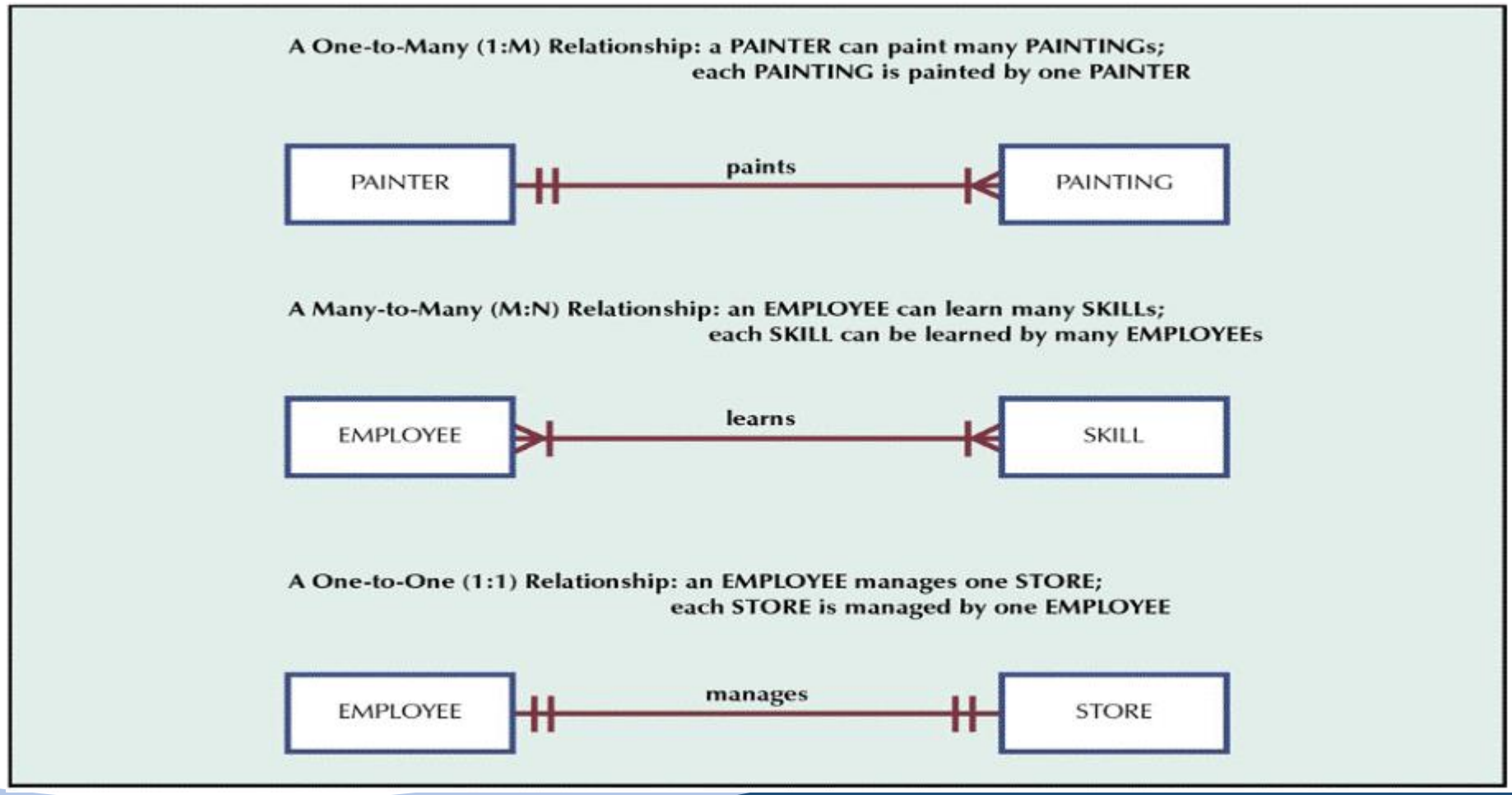
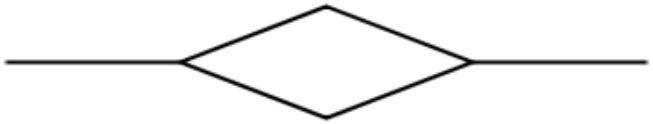
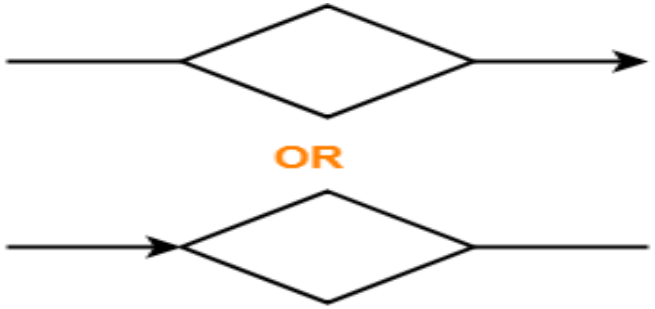


Figure: Basic Crow Foot ERD Notation

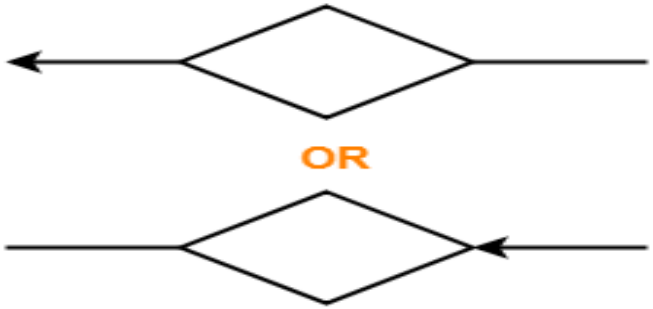




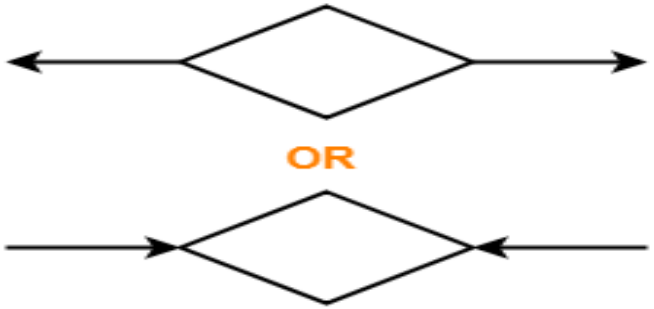
Many-to-Many relationship
(m:n)



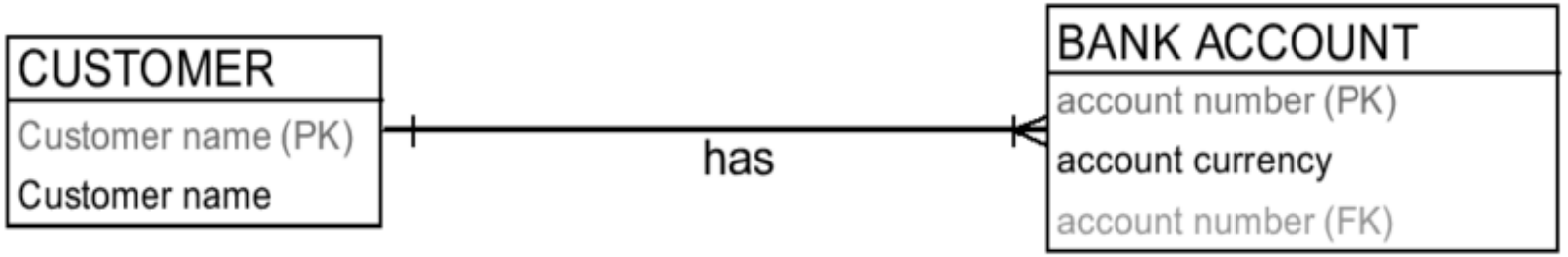
Many-to-One relationship
(m:1)

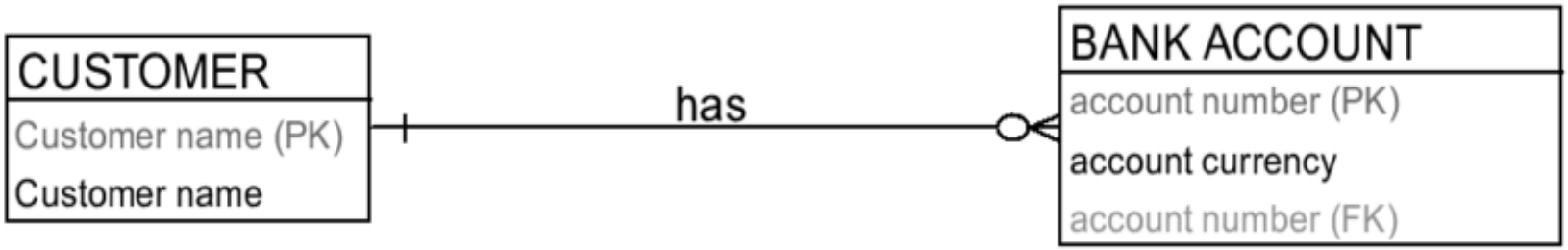


One-to-Many relationship
(1:n)



One-to-One relationship
(1:1)


CUSTOMER
Customer name (PK)
Customer name
has
BANK ACCOUNT
account number (PK)
account currency
account number (FK)


CUSTOMER
Customer name (PK)
Customer name
has
BANK ACCOUNT
account number (PK)
account currency
account number (FK)

The Three Types of Minimum Cardinality

(a) Mandatory-to-Mandatory (M-M) Relationship

(b) Optional-to-Optional (O-O) Relationship

(c) Optional-to-Mandatory Relationship

ER Model contd..

Advantages

–Exceptional conceptual simplicity

–Visual representation

–Effective communication tool

–Integrated with the relational data model

Disadvantages

–Limited constraint representation

–Limited relationship representation

–No data manipulation language

–Loss of information content

Object Oriented Model

- Semantic data model (SDM) developed by Hammer and McLeod in 1981.
- Modeled both data and their relationships in a single structure known as an object.
- Basis of object oriented data model (OODM)
- OODM becomes the basis for the object oriented database management system(OODBMS)
- Object is described by its factual content
  - –Like relational model's entity
- Includes information about relationships between facts within object and relationships with other objects
  - –Unlike relational model's entity

Contd..

- Subsequent OODM development allowed an object to also contain operations
- Object becomes basic building block for autonomous structures

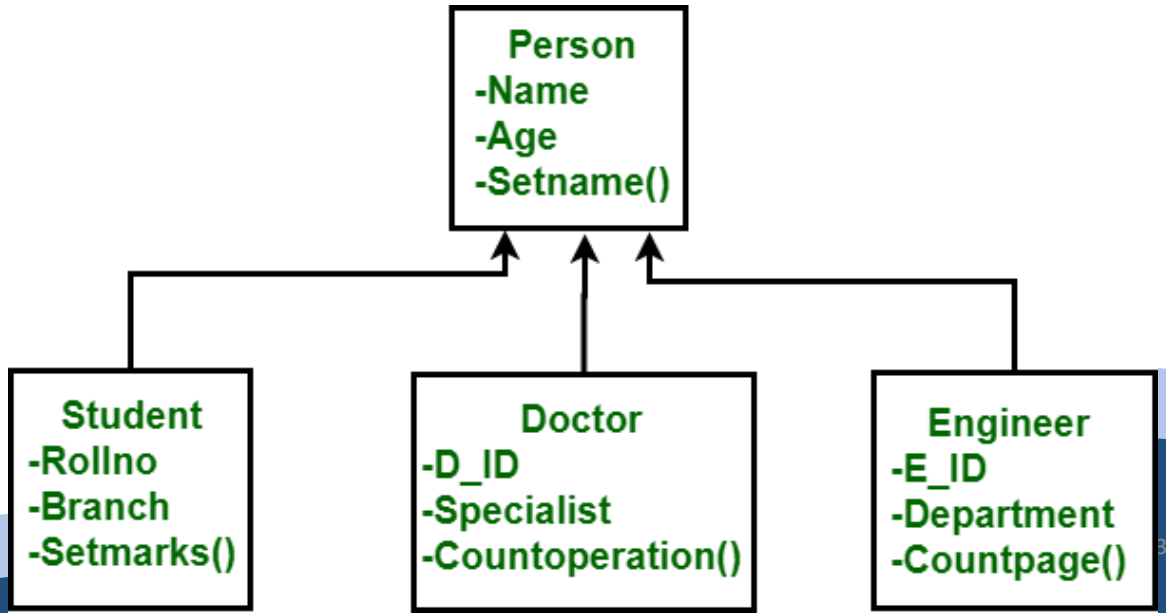
Developments that Boosted OODM's Popularity

- Growing costs put a premium on code reusability
- Complex data types and system requirements became difficult to manage with a traditional RDBMS
- Became possible to support increasingly sophisticated transaction & information requirements
- Ever-increasing computing power made it possible to support the large computing overhead required

Object Oriented Data Model—Basic Structure

- Object: abstraction of a real-world entity
- Attributes describe the properties of an object
- Objects that share similar characteristics are grouped in classes
- Classes are organized in a class hierarchy
- Inheritance is the ability of an object within the class hierarchy to inherit the attributes and methods of classes above it

Example: OODM



- **Objects –**
  An object is an abstraction of a real world entity or we can say it is an instance of class. Objects encapsulates data and code into a single unit which provide data abstraction by hiding the implementation details from the user. For example: Instances of student, doctor, engineer in above figure.
- **Attribute –**
  An attribute describes the properties of object. For example: Object is STUDENT and its attribute are Roll no, Branch, Setmarks() in the Student class.
- **Methods –**
  Method represents the behavior of an object. Basically, it represents the real-world action. For example: Finding a STUDENT marks in above figure as Setmarks().
- **Class –**
  A class is a collection of similar objects with shared structure i.e. attributes and behavior i.e. methods. An object is an instance of class. For example: Person, Student, Doctor, Engineer in above figure.
- **Inheritance –**
  By using inheritance, new class can inherit the attributes and methods of the old class i.e. base class. For example: as classes Student, Doctor and Engineer are inherited from the base class Person.

OODM contd..

Advantages

–Adds semantic content

–Visual presentation includes semantic content

–Database integrity

–Both structural and data independence

Disadvantages

–Slow pace of OODM standards development

–Complex navigational data access

–Steep learning curve

–High system overhead slows transactions

–Lack of market penetration

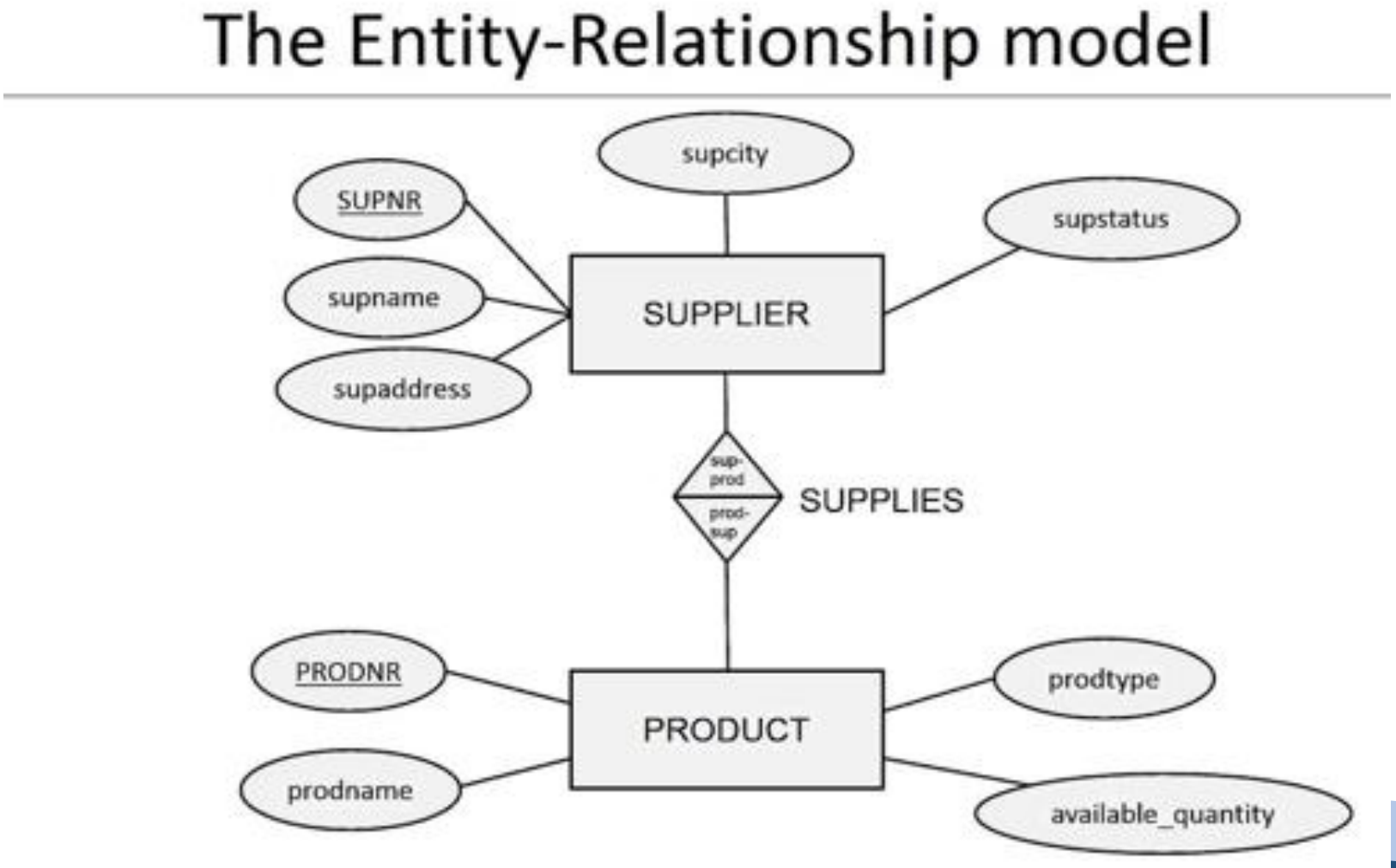
Entity Relationship Model(ER Model)

- An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram).
- An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.
- The ER model employs three basic concepts : Entity Set, Relationship set and attributes.

ER Diagram:

- An ER diagram shows the relationship among entity sets.
- An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database.

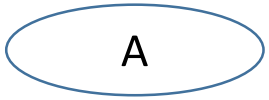
A simple ER diagram



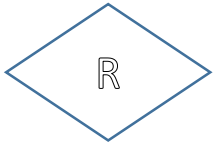
Symbols used for ER diagram

Entity

Attributes

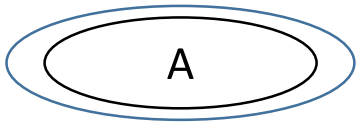


Relationship set



Links

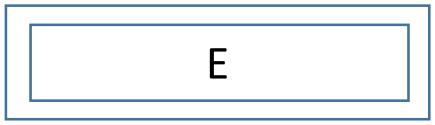
Multivalued Attribute



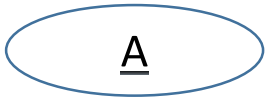
Derived Attribute

Total Participation

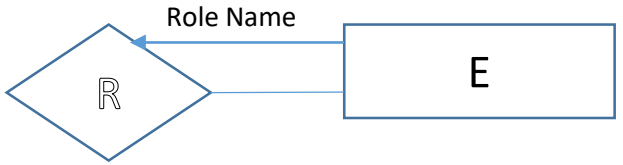
Weak Entity set



Primary Key



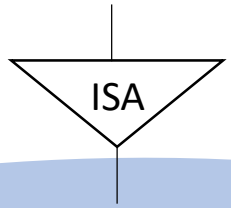
Role Name



Discriminating Attribute

Generalization/

Specialization



Entity:

- An entity is an object or component of data. An entity is represented as rectangle in an ER diagram.

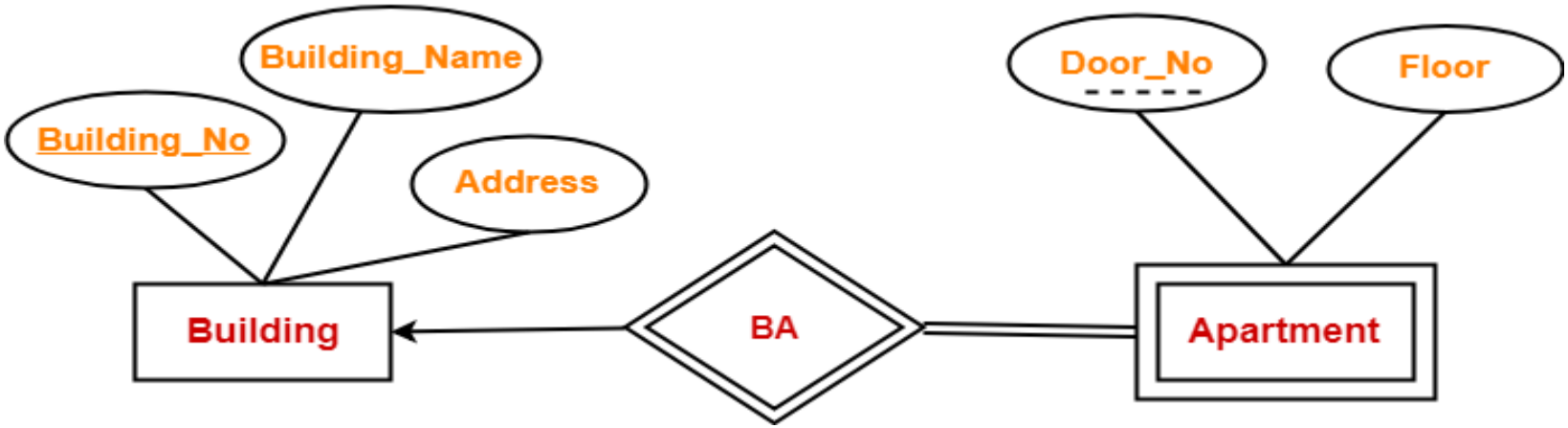
For example: In the following ER diagram we have two entities Student and College and these two entities have many to one relationship as many students study in a single college.

Strong Entity: an entity set which may have sufficient attributes to form a primary key is called strong entity

Weak Entity:

- An entity that cannot be uniquely identified by its own attributes and relies on the relationship with other entity is called weak entity. The weak entity is represented by a double rectangle.

Example: Strong and Weak Entity set



Strong entity set: Building
Primary key: Building_No
Weak Entity Set: Apartment
Discriminator: Door_No
Relationship set: BA
Note: relationship set associating the weak entity set is denoted by double diamond symbol

Attribute

An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. There are four types of attributes:

1. Key attribute
2. Simple attribute
3. Composite attribute
4. Multivalued attribute
5. Derived attribute

Key Attribute:

A key attribute can uniquely identify an entity from an entity set.

For example, student roll number can uniquely identify a student from a set of students. Key attribute is represented by oval same as other attributes however the **text of key attribute is underlined**.

Simple Attribute: An attribute that cannot be divided in sub class. Eg. College_name.

Composite Attribute:

An attribute that is a combination of other attributes is known as composite attribute.

For example, In student entity, the student address is a composite attribute as an address is composed of other attributes such as pin code, state, country.

Multivalued attribute:

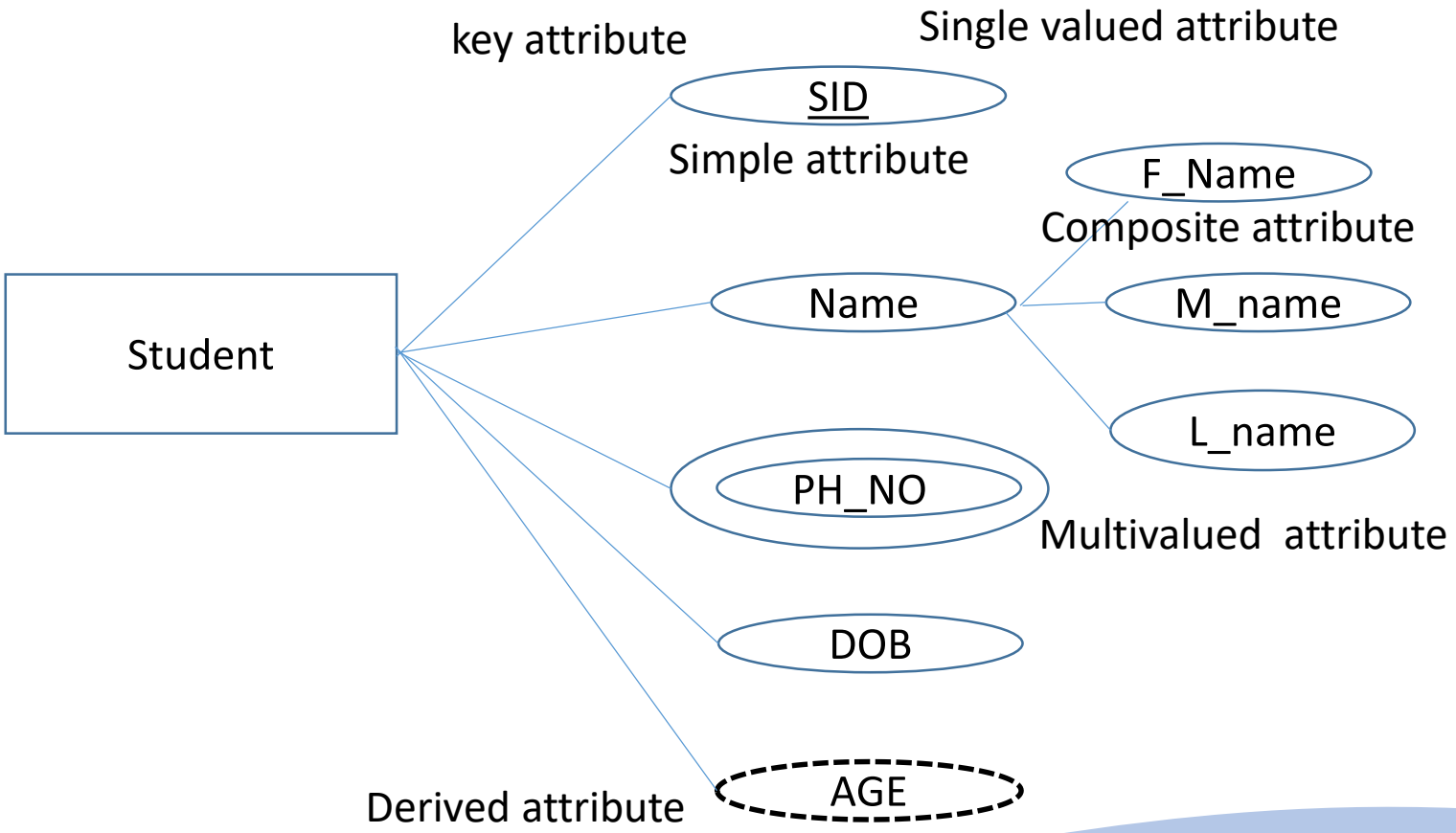
- An attribute that can hold multiple values is known as multivalued attribute. It is represented with **double ovals** in an ER Diagram.

For example – A person can have more than one phone numbers so the phone number attribute is multivalued.

Derived attribute:

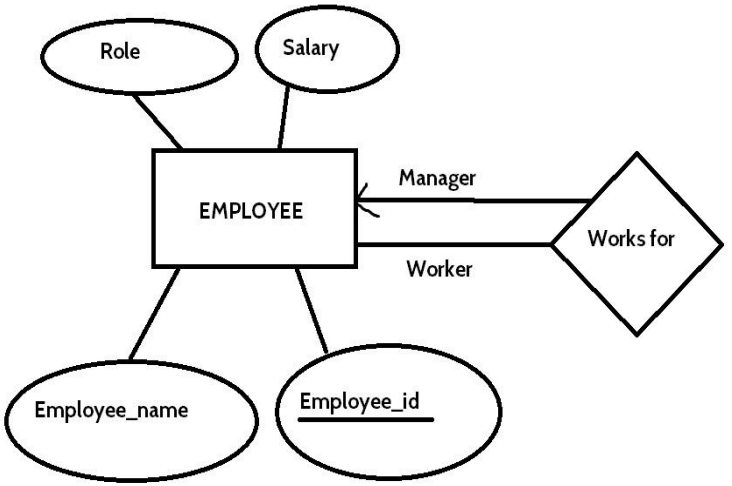
- A derived attribute is one whose value is dynamic and derived from another attribute. It is represented by **dashed oval** in an ER Diagram.
- For example – Person age is a derived attribute as it changes over time and can be derived from another attribute (Date of birth).

Example: Types of Attributes



Roles:

- A database role is a collection of any number of permissions/privileges that can be assigned to one or more users. A database role also is also given a name for that collection of privileges.

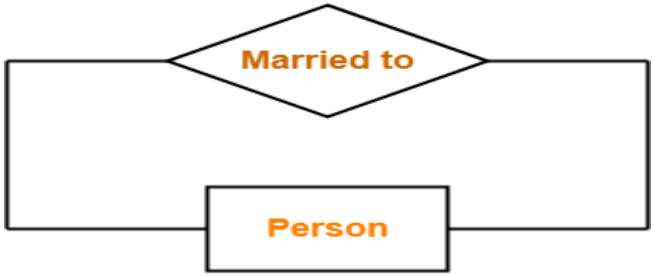


Degree of relationship:

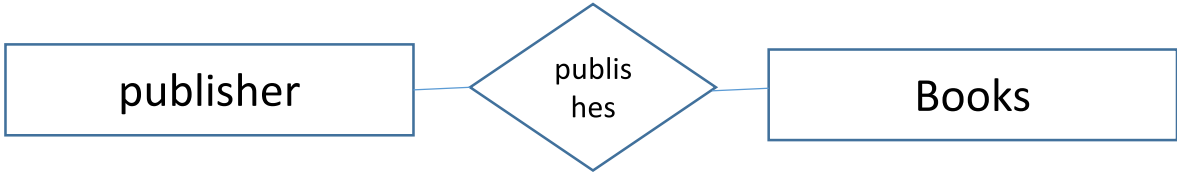
It is the number of entities associated with the relationship set. The n-ary relationship is the general form of degree n.

Types:
Unary relationship
Binary relationship
Ternary relationship
Quarternary relationship

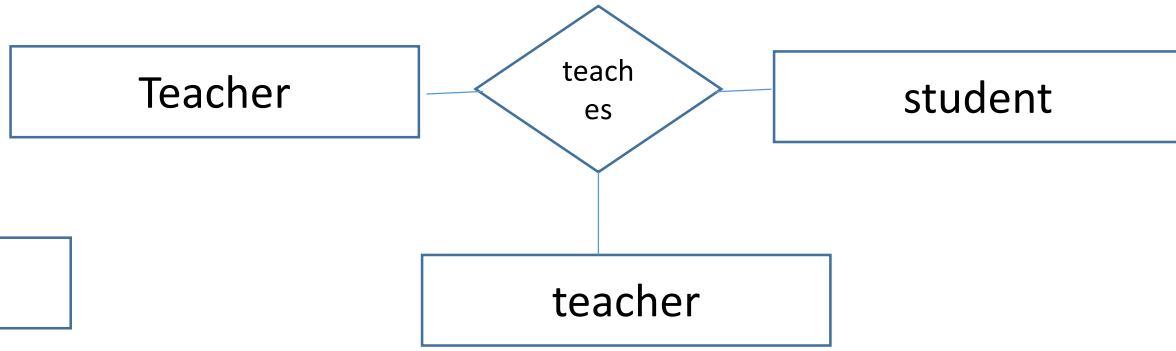
Example



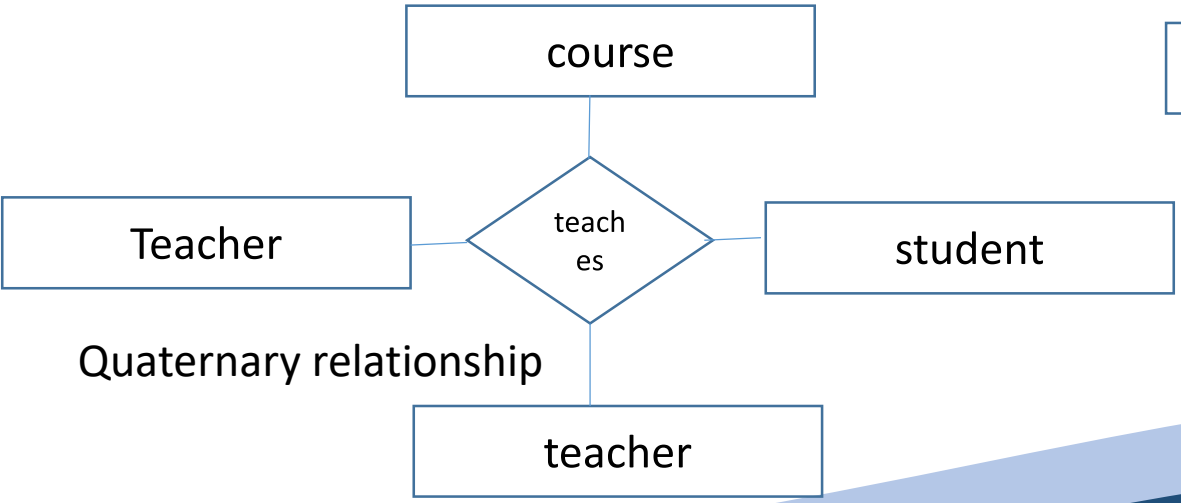
Unary Relationship Set



Binary relationship



Ternary relationship



Quaternary relationship

ER modeling - Constraints

a) Mapping Cardinalities

b) Participation Constraints

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:

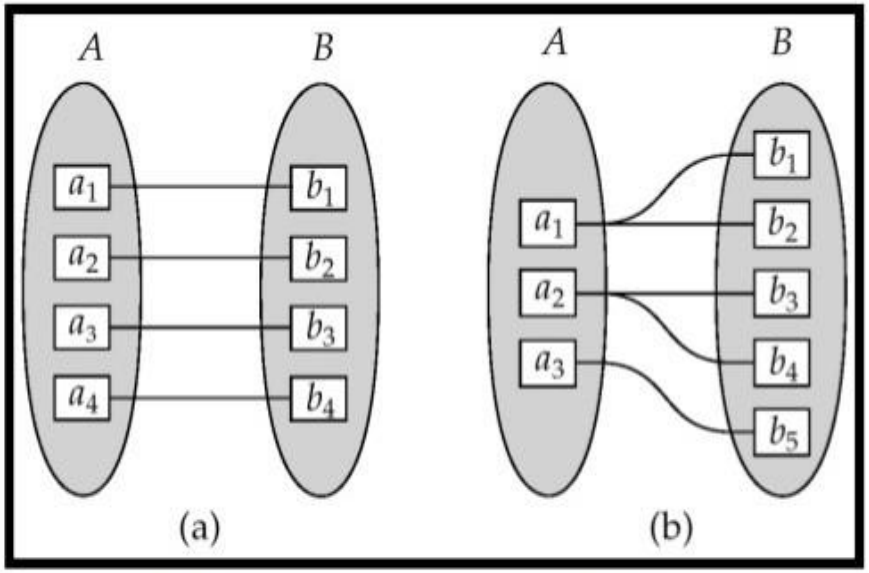
One-to-one: An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.

One-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.

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-many: Entities in A and B are associated with any number from each other. (Figure 2.6)

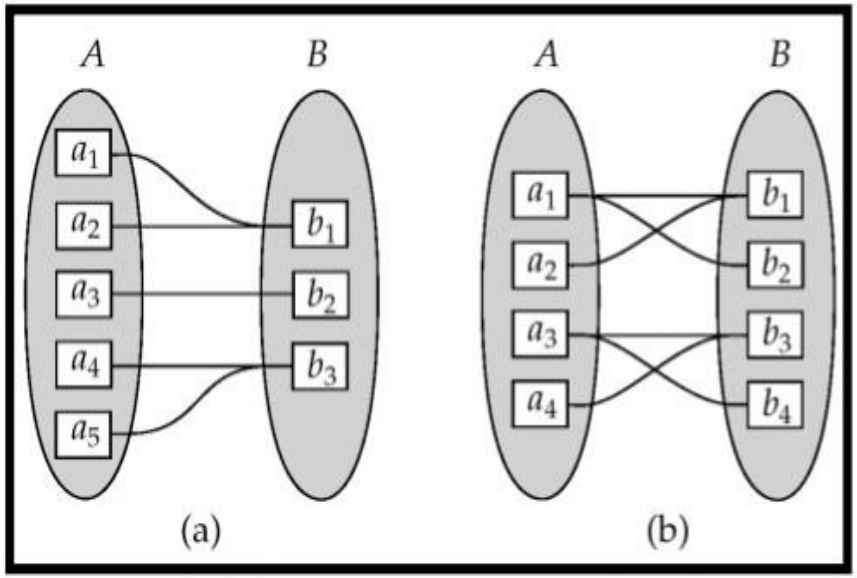
Example



One to

One to many

Note: Some elements in A and B may not be mapped to any elements in the other set



Many to one

Many to many

Note: Some elements in A and B may not be mapped to any elements in the other set

16

- one-to-one

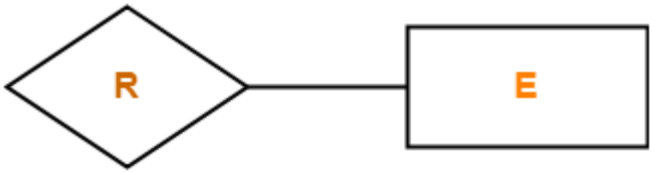
- one to many

- many-to-many

Participation Constraint:

- Total Participation: It specifies that each entity in the entity set must compulsorily participate in at least one relationship instance in that relationship set.
- That is why, it is also called as **mandatory participation.**
- Total participation is represented using a double line between the entity set and relationship set.

- Partial Participation: It specifies that each entity in the entity set may or may not participate in the relationship instance in that relationship set.
- That is why, it is also called as **optional participation.**
- Partial participation is represented using a single line between the entity set and relationship set.



ER Diagram: A library management System

Task 1: Finding out the *entity sets* and their *attribute list*

Entity Sets	Attribute list				
Employee	Eid	Name	Phno	Address	-
Student	Sid	Name	Batch	Faculty	phno
Teacher	Tid	Name	Faculty	Specialization	-
Book	ISBN	Name	Author	Publication	-
Distributer	PAN no	Dname	Address	Phno	-

- Task 2: Find out the relationship sets among the entity sets

	Employee	Student	Teacher	Book	Distributer
Employee	Works for	Book issue	Book issue	Manage book	Book order
Student	-	-	Refer book	Book issue	-
Teacher	-	-	-	Book issue	-
Book	-	-	-	-	Manage book
Distributer	-	-	-	-	-

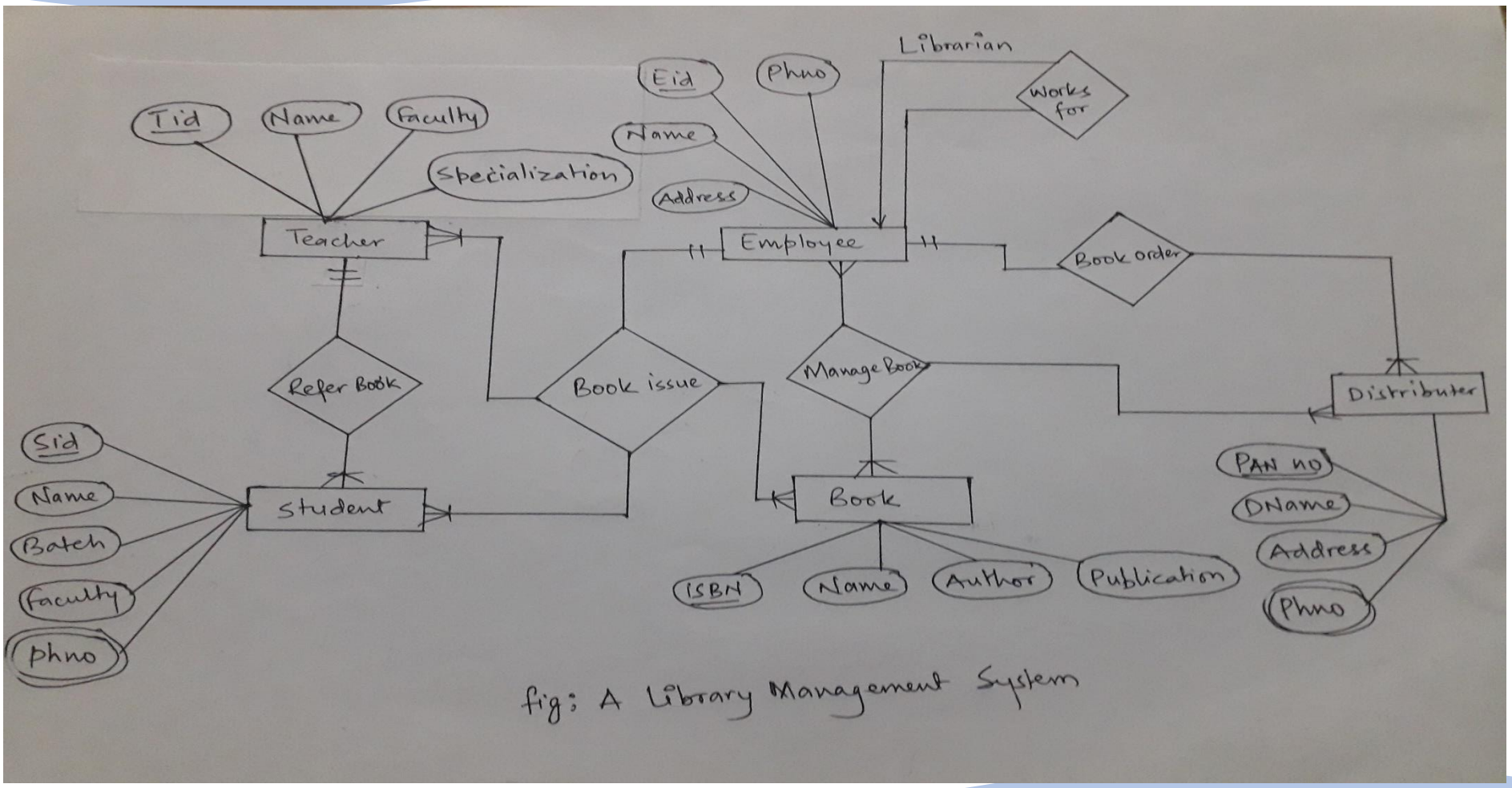


fig: A Library Management System

ER Diagram: A Banking system

Task 1: Finding out the *entity sets* and their *attribute list*

Entity Sets	Attribute List				
Customer	Phno	C_Id	C_name	C_address	-
Employee	E_id	Ename	Phno	DOB	Age
Loan	LoanNo	Amount	-	-	-
Account	AccNo	Balance	AccType	-	-
Branch	Bname	Bcity	Assets	-	-
Payment	PaymentNo	Amount	-	-	-

- Task 2: Find out the relationship sets among the entity sets

	Customer	Employee	Loan	Account	Branch	Payment
Customer	-	Service_for	Borrower	Depositor	-	-
Employee	-	Works_for	-	-	-	-
Loan	-	-	-	-	Managed_at	Loan Payment
Account	-	-	-	-	-	-
Branch	-	-	-	-	-	-
Payment	-	-	-	-	-	-

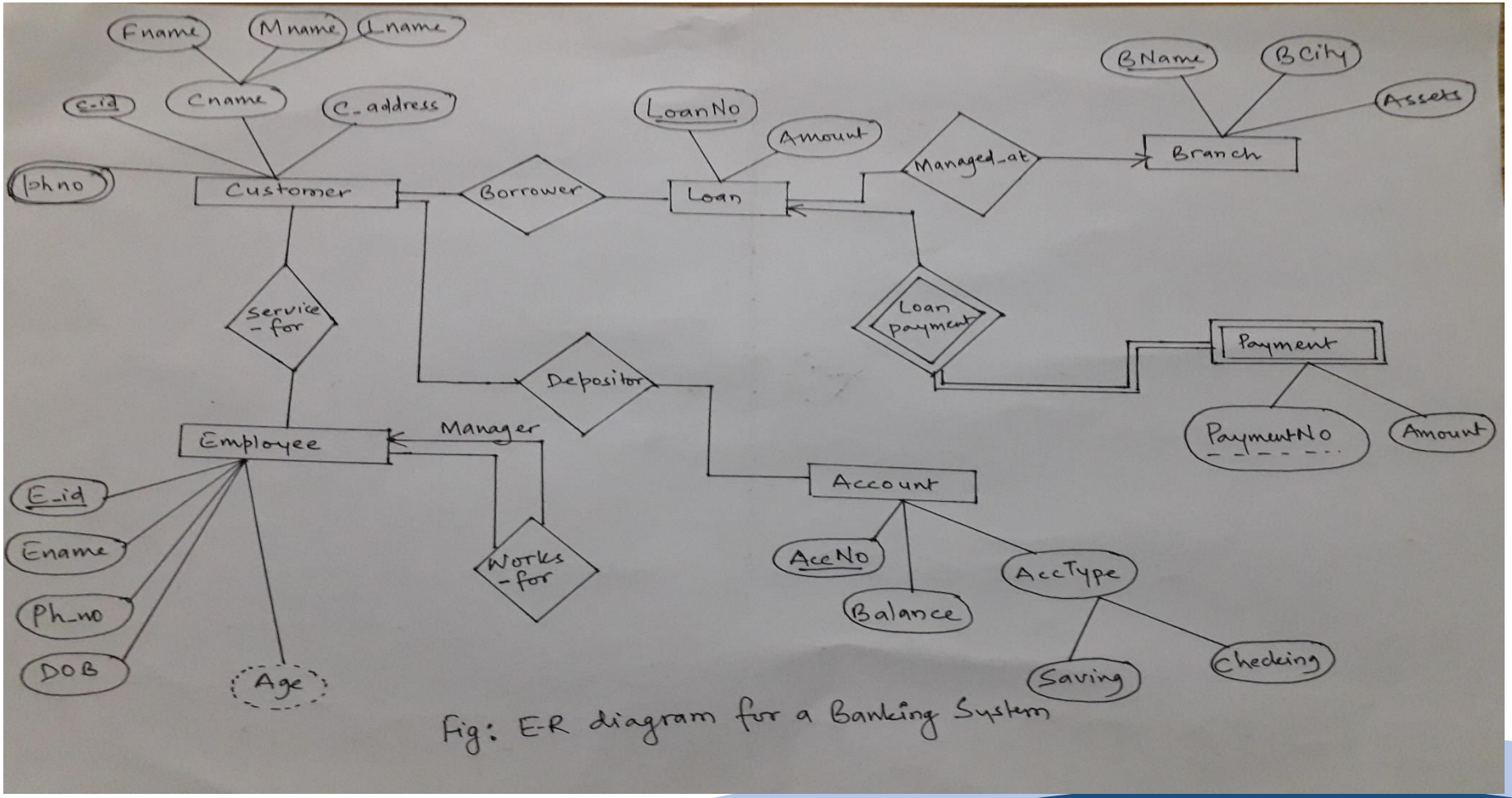


Fig: E-R diagram for a Banking System

ER Diagram: A Hotel management system

ER Diagram: An Online BOOK Store

Case Study 1

You are given the requirement for a simple database for National Football League(NFL). The NFL has many teams and each team has a name, a city, a coach, a captain and a set of players. Each player belongs to only one team and each player has a name, a position(left wing, mid fielder or goalkeeper) a skill level, a set of injury records. A team captain is also a player and a game is played between two teams(referred as host team and guest team) and has a match date (such as June 11,2018) and score such as (2 or 5). Construct an ER diagram for the following NFL database.

Task 1: Finding out the *entity sets* and their *attribute list*

Entity Sets	Attribute list			
NFL	-	-	-	-
Team	Name	City	Coach name	-
Player	name	Position	Skill level	Injury record

- Task 2: Find out the relationship sets among the entity sets

	NFL	Team	Player
NFL	-	NFL Team	-
Team	-	Match	TP
Player	-	-	Set Player

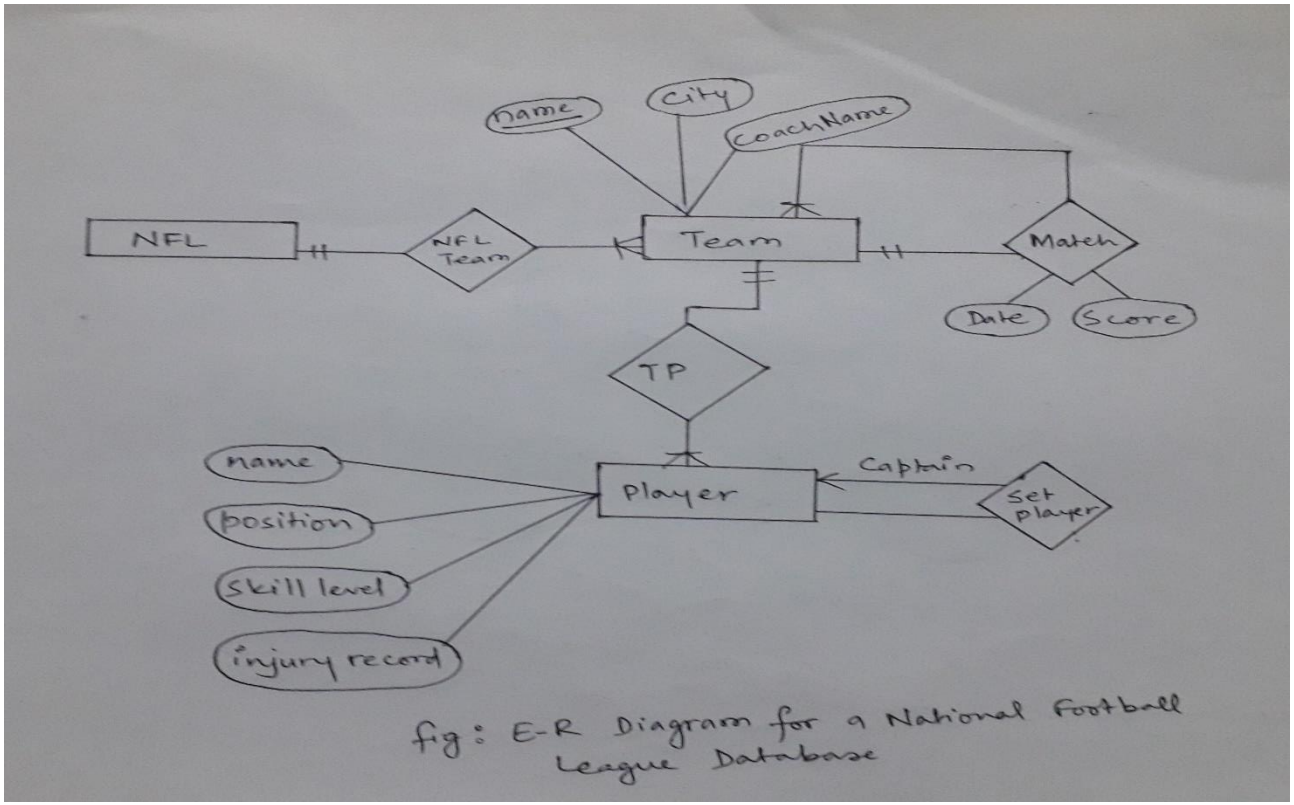


fig: E-R Diagram for a National Football League Database

ER Diagram: An Airline Reservation system

Case 2:

Design an E-R diagram for a database for an airlines system. The database must keep track of customers and their reservations, flights and their status' seat assignments on individual flights and the schedule and routing of future flights. Apply all the database design constraints as much as possible .

- Case 3:
- Draw the Entity Relationship Diagram (ERD) with appropriate mapping cardinalities for the following scenario. Patients are treated in a single ward by the doctors assigned to them. Healthcare assistants also attend to the patients; a number of these are associated with each ward. Each patient is required to take a variety of drugs a certain number of times per day and for varying lengths of time. The system must record details concerning patient treatment and staff payment. Some staffs are paid part time and doctors and healthcare assistants work varying amounts of overtime at varying rates, the system will also need to track what treatments are required for which patients.

Extended E-R Diagram (EER diagram)

- With time the complexity of the data is increasing so it becomes more and more difficult to use the traditional ER model for database modeling.
- To reduce this complexity of modeling, improvements or enhancements were made to the existing ER model to make it able to handle the complex application in a better way.
- Extended entity-relationship diagrams are advanced database diagrams very similar to regular ER diagrams which represents requirements and complexities of complex databases.
- It is a diagrammatic technique for displaying the Sub Class and Super Class; Specialization and Generalization; Union or Category; Aggregation etc.

Specialization and Generalization

Concept of subclass and super class

Subclass: is an entity type that is the part of super class that has attributes distinct from other sub groups

Super class: includes distinct subclasses.

- superclass attributes are shared among subclass.

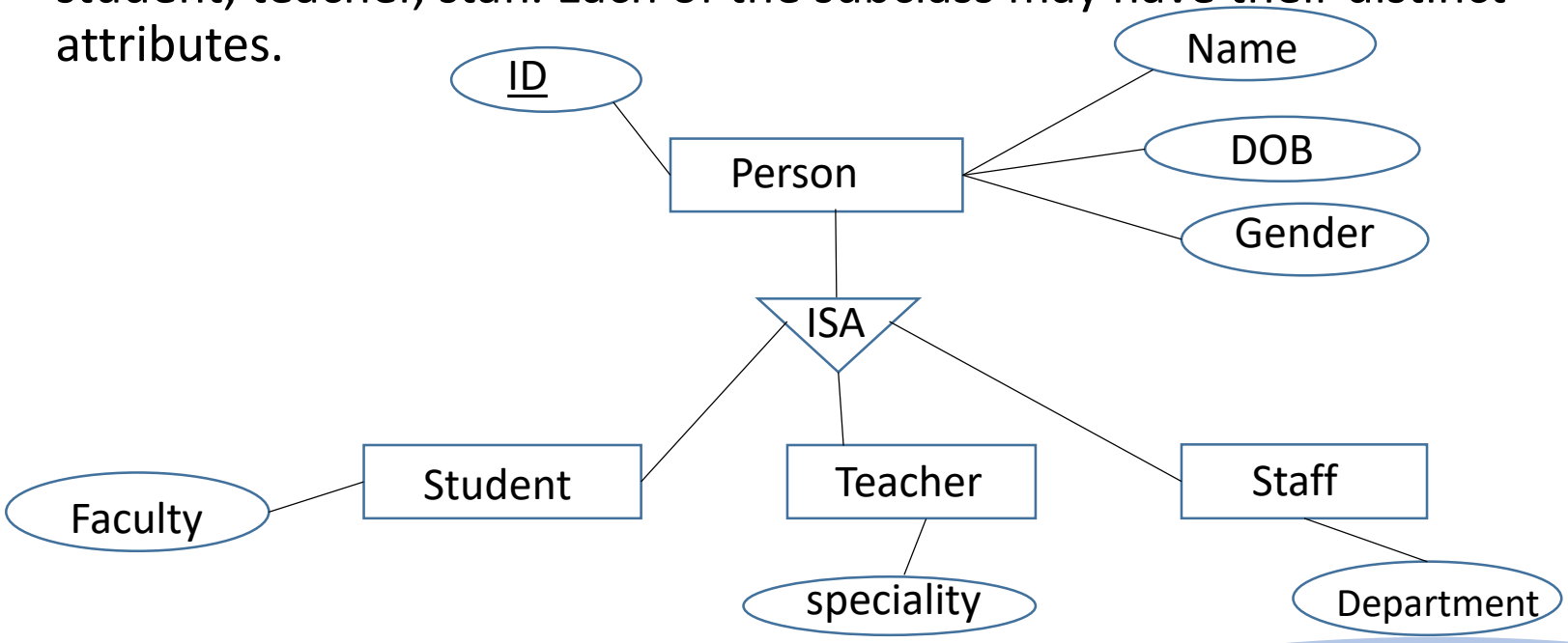
Eg: vehicle (superclass)

motorcycle, car, truck (subclass)

Specialization: process of forming a subclass

- top down process
- identifying sub groups within an entity set which have attributes not shared by all entities(unique properties)

- Example: Take a group ‘Person’. A person has name, date of birth, gender, etc. These properties are common in all persons, human beings. But in a College database, Person can be further classified as student, teacher, staff. Each of the subclass may have their distinct attributes.



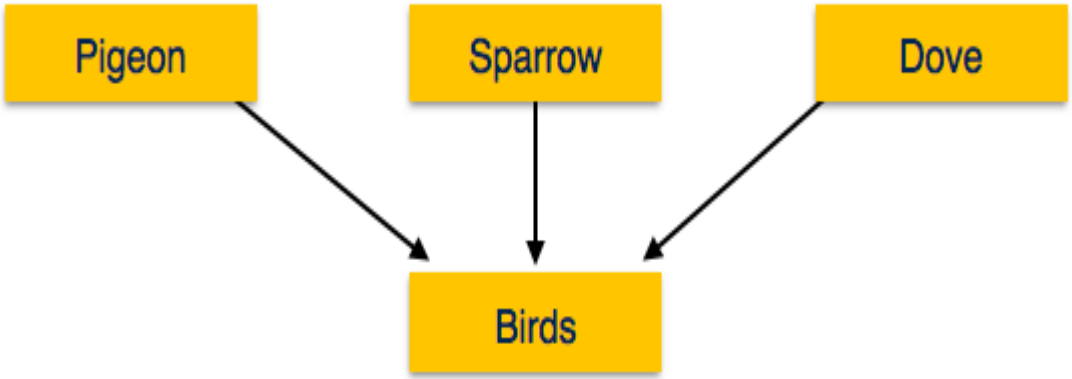
Contd..

Generalization:

-process of forming a super class

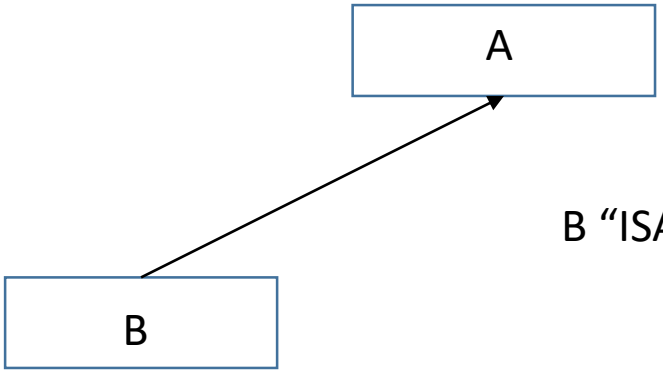
-proceeds from the recognition of a number of entities that share common attributes

-In generalization, a number of entities are brought together into one generalized entity based on their similar characteristics. For example, pigeon, house sparrow, crow and dove can all be generalized as Birds.



Attribute Inheritance

- A crucial property of higher and lower level entities created by specialization and generalization
- The attributes of higher level entity sets are said to be inherited by lower level entity sets.
- For example, Student, teacher and staff inherit the attributes of Person (as in previous figure)



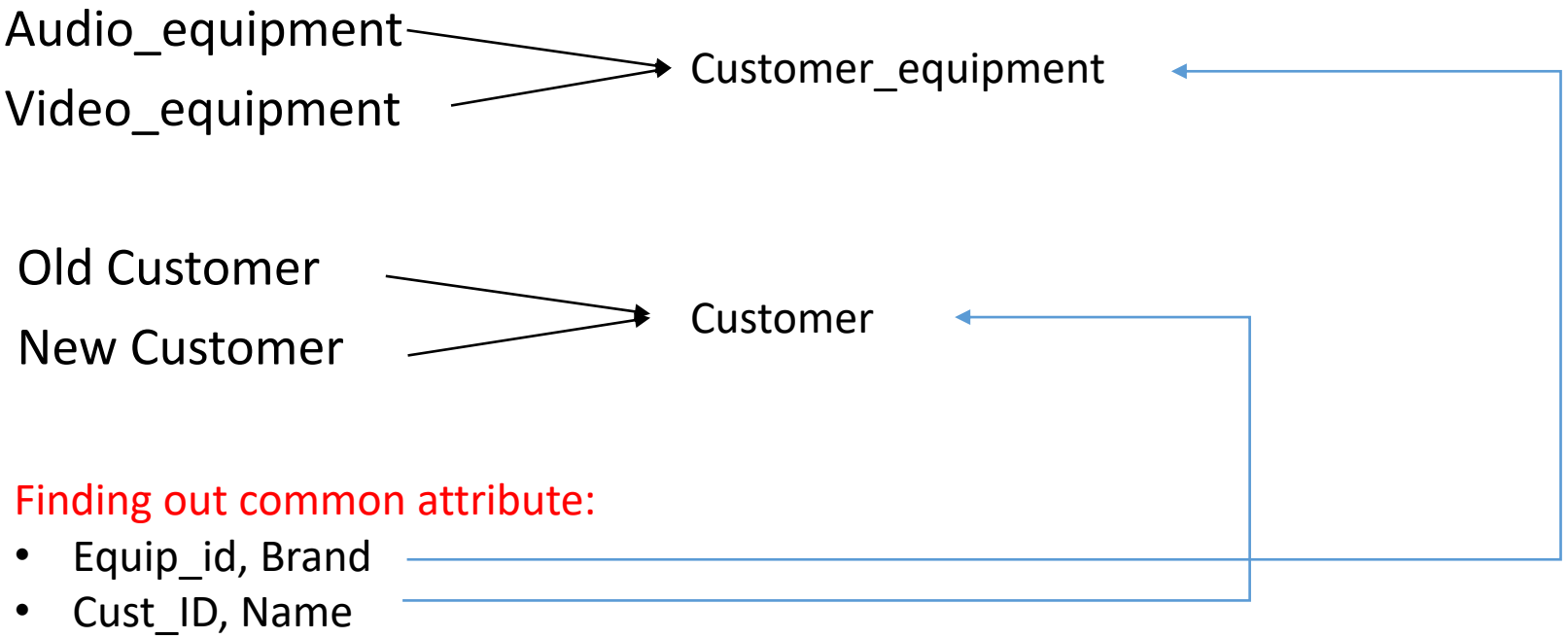
B “ISA” A (B inherits the feature of A)

- Case 4:
- Assume you are to compose database requirements of a wholesale dealer for audio, video consumer equipment from different manufacturers(brands). Customers are the various retailer outlets (retailers). Whole seller extends credits to old customer and special discounts are offered to new customers (retailers). You have to generate an ER model for the above DBMS application with the scope restricted to details of customers, products stocked and their price discounts and credits offered.

- Finding out Entity sets and their Attribute list

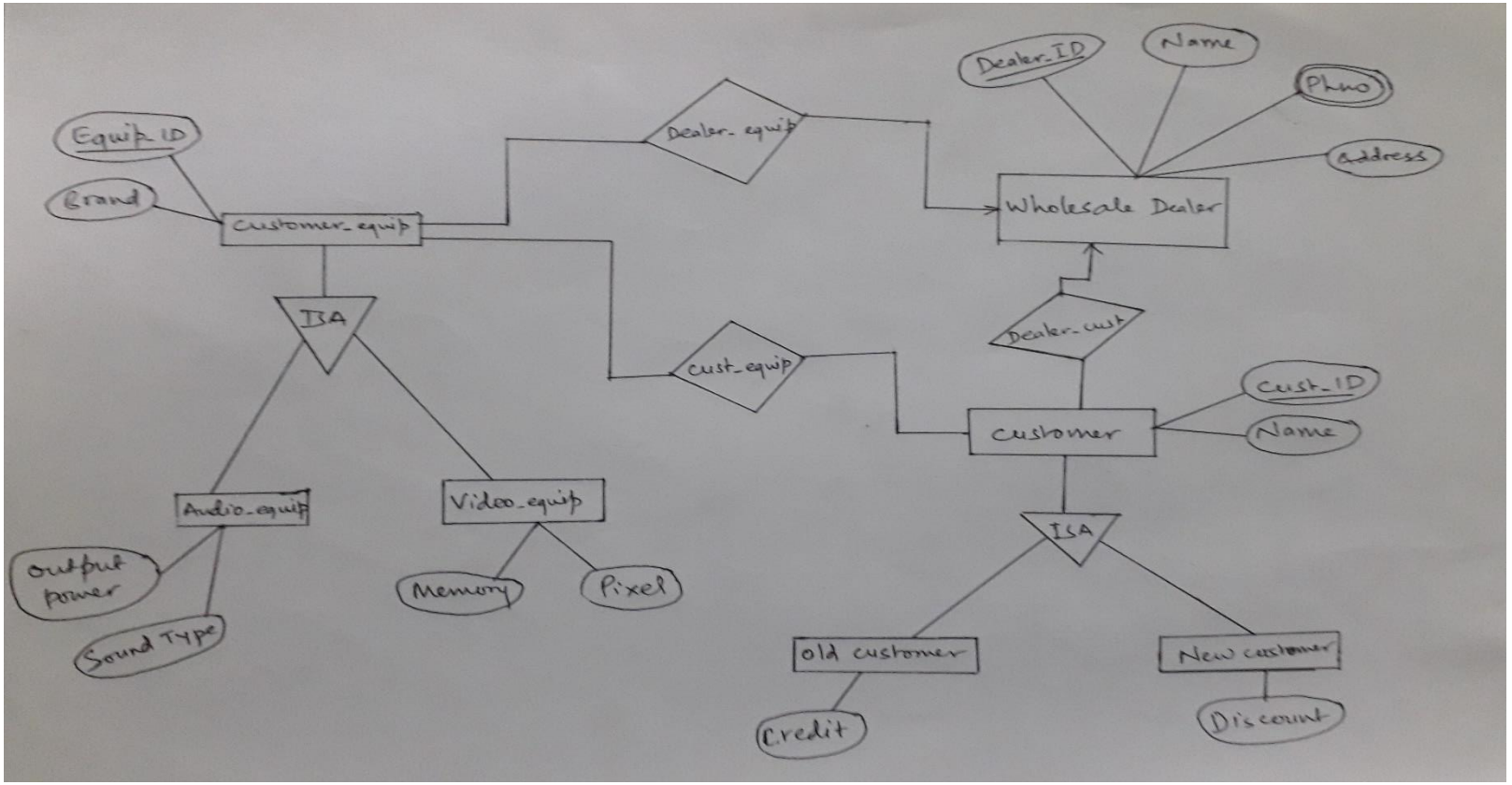
Entities	Attributes			
Audio_equipment	Equip_ID	Brand	Output power	Sound type
Video_equipment	Eqiup_ID	Brand	Memory	Pixel
Old customer	Cust_ID	Name	Credit	
New customer	Cust_ID	Name	Discount	
Wholesale dealer	Dealer_ID	Name	Ph_NO	Address

- Forming Super Class



Finding out common attribute:

- Equip_id, Brand
- Cust_ID, Name


Dealer_ID
Name
Phno
Address
Equip_ID
Dealer_equip
Brand
Customer_equip
Wholesale Dealer
ISA
Dealer_cust
cust_equip
Cust_ID
Name
Customer
Audio_equip
Video_equip
ISA
output power
Sound Type
Memory
Pixel
old customer
New customer
Credit
Discount

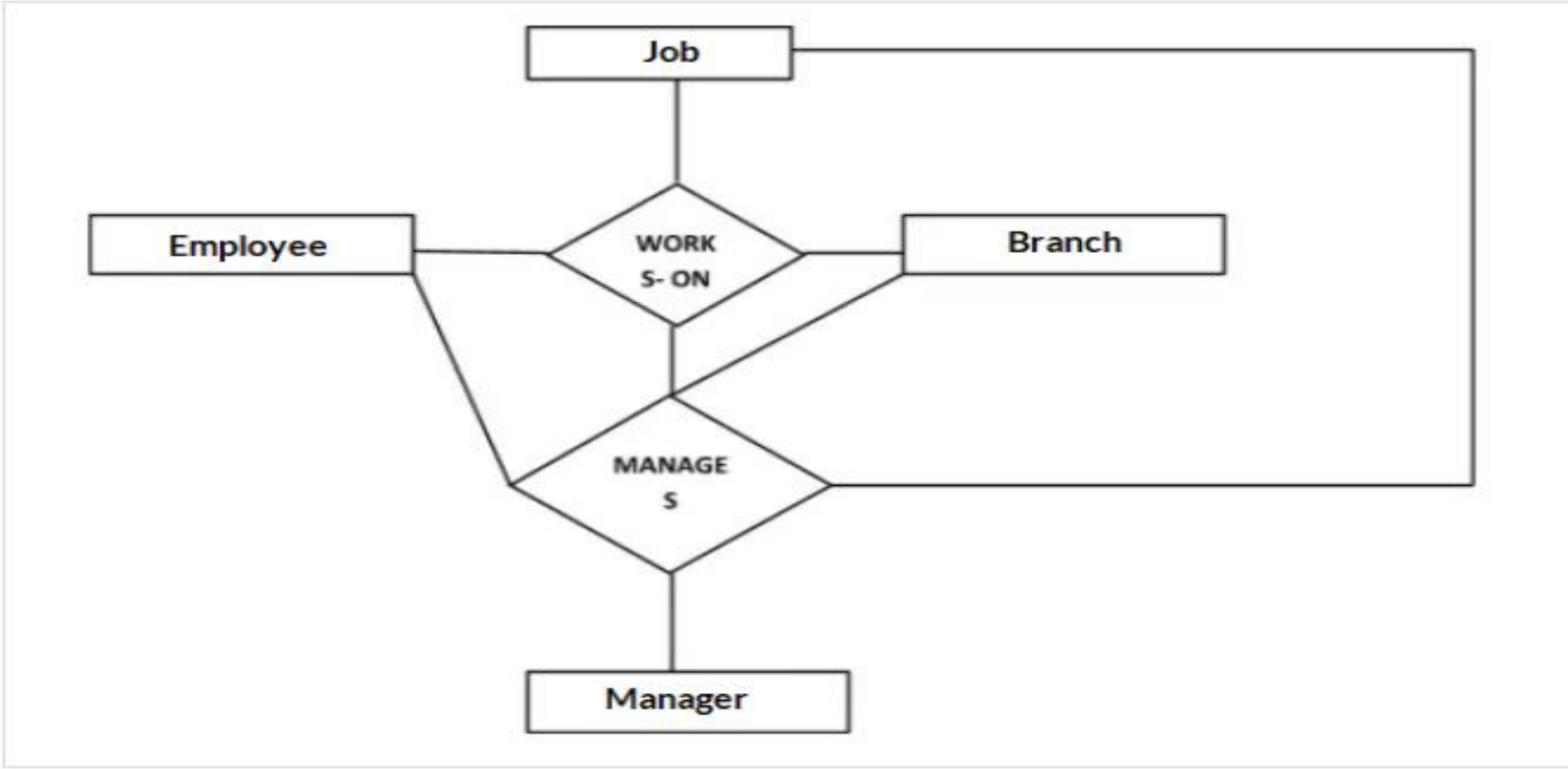
- Case 4:

An information system is to be designed for keeping the records of Universe cup cricket tournament. In all, there are teams from 10 countries participating in the tournament. Each country sends 15 players and 4 other members. For players, the runs he scores and the number of wickets taken (so far) are to be recorded. For the non-players, the role(manager, coach etc.) and the number of years of experience in that capacity are to be recorded. There are matches scheduled amongst the teams on several grounds on fixed dates. Each ground has a fixed seating capacity and a size. For 38 matches, 11 referees have been assigned duties. Every match will have 3 referees working for it. The performance of every player in every match is to be recorded in terms of the runs he scored and the wicket he took. Draw an E-R model of the system.

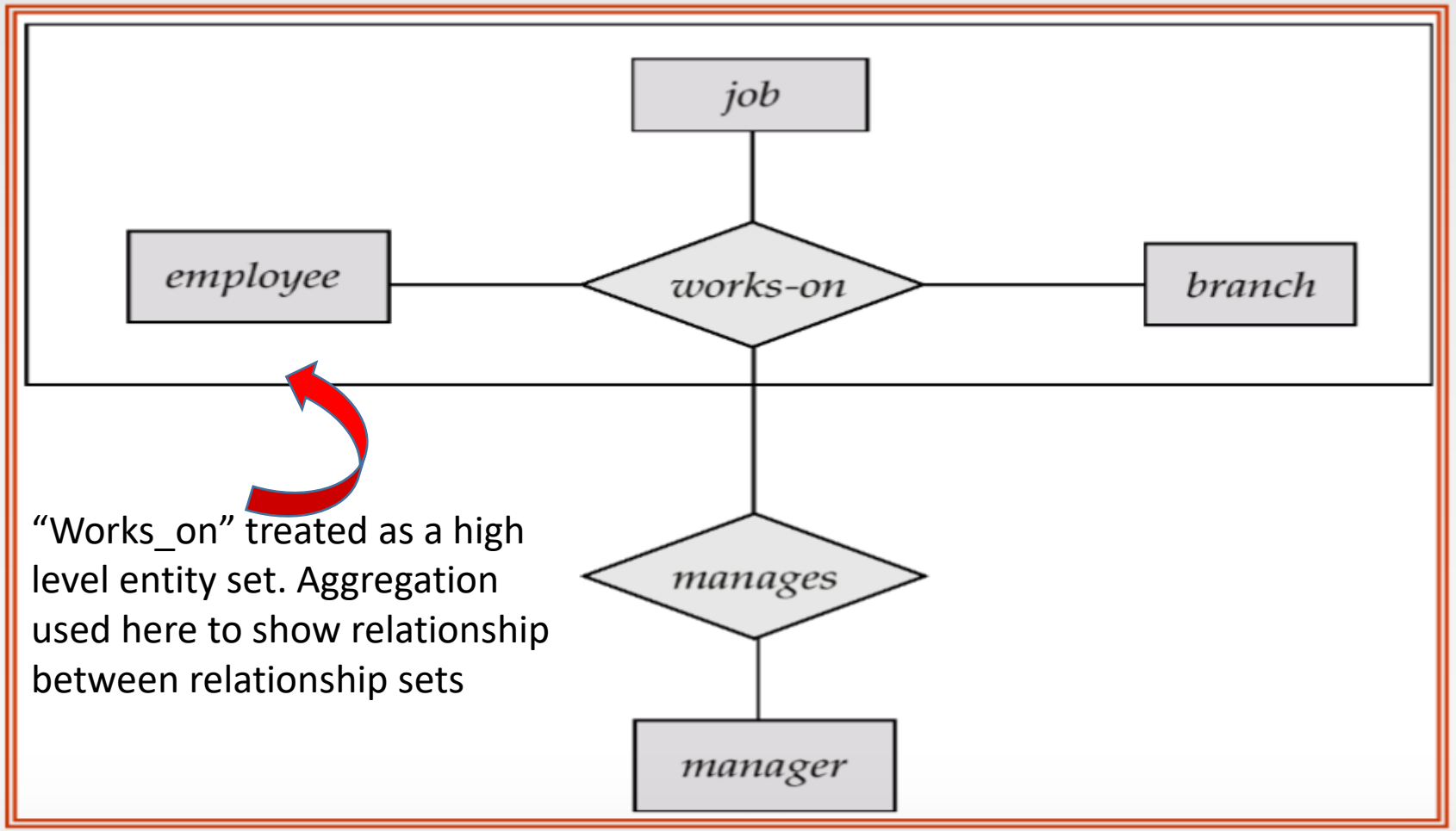
Aggregation

- One limitation of ER model is that it can't express relationship among relationships.
- Aggregation is the process of compiling information of an object
- It is an abstraction through which relationship are treated as higher level entities.

Ex: relationship between employee, branch, job, manager



Consider a ternary relationship Works_On between Employee, Branch and Manager. Now the best way to model this situation is to use aggregation, So, the relationship-set, Works_On is a higher level entity-set. Such an entity-set is treated in the same manner as any other entity-set. We can create a binary relationship, Manager, between Works_On and Manager to represent who manages what tasks.


job
employee
works-on
branch
manages
manager
"Works_on" treated as a high level entity set. Aggregation used here to show relationship between relationship sets

Schema Diagram

- A database which confirm to an ER diagram can be represented by a collection of schema.
- For each entity set and relationship set, there is a unique schema that is assigned the name of corresponding entity set of relationship set.
- Each schema has a number of column which have unique names.

Rules for schema Diagram:

- Each relation appears in rectangular box
- Attributes are listed inside box and relation name is shown above it.
- If there is PK , we draw a horizontal line below the attribute.
- FK dependencies are shown by the arrow from foreign key attribute to the PK in reference relation.

Consider an example:

branch(b_name, assets)

Depositor(cname,acc_no)

Loan(loan_no, b_name,amount)

Borrower(cname,loan_no)

Account (acc_no, b_name,balance)

Customer (cname,c_street,c_city)

Example

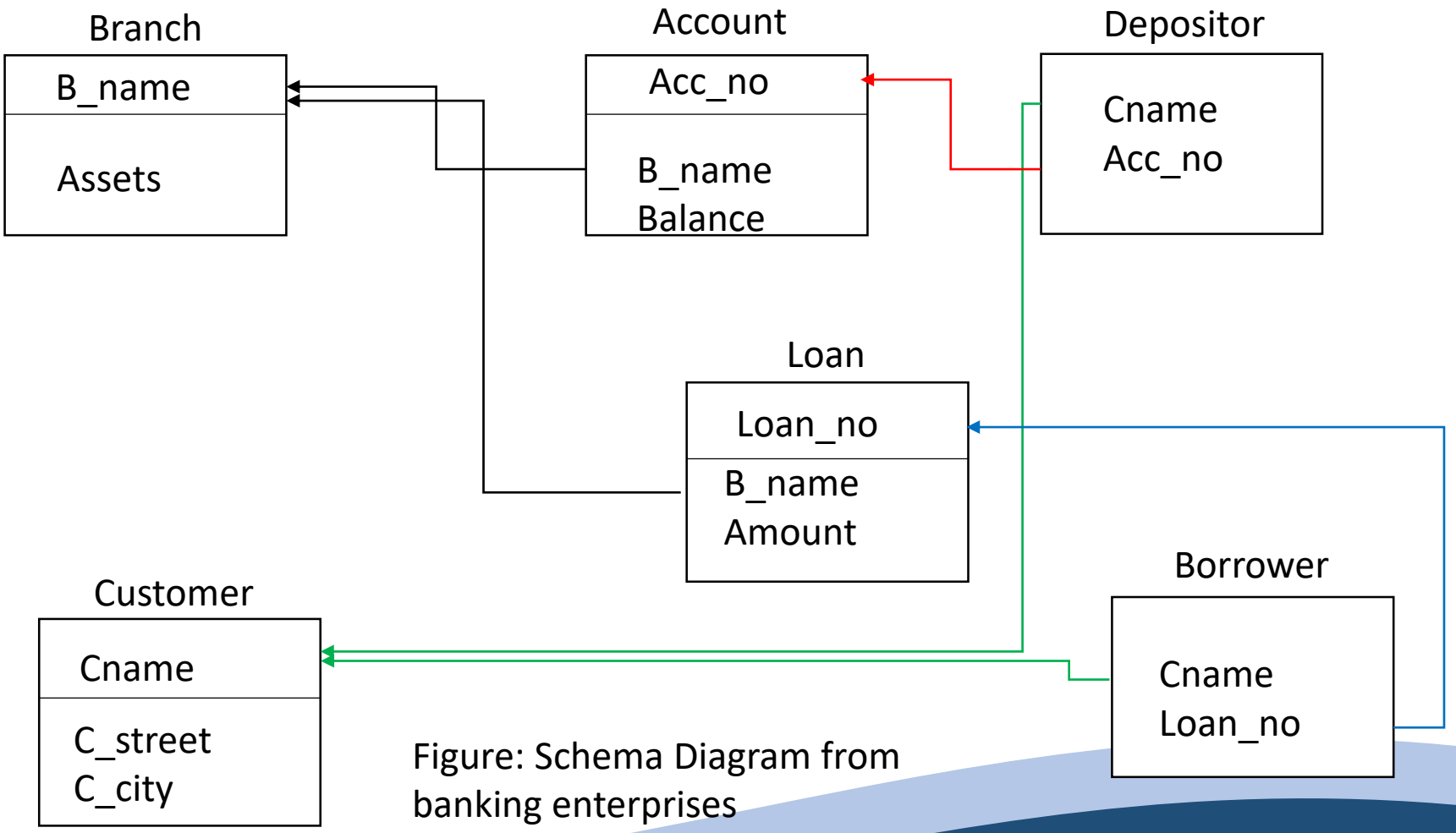


Figure: Schema Diagram from banking enterprises

Rules for converting ER diagram to Tables

- Strong entity sets reduces to a table with same attribute

Example: Student(sid, Name)

Student

Sid	Name

- A weak entity set becomes a table that includes a column for PK of reference relation plus column of its own entity set. Example: College(college_name, college_address)

College

SID	College_name	College_address

- Tabular representation of relationship set consist of Primary Keys representing the relations.

Example: Student(Sid, Name) Teacher(Tid, Name, Speciality)

Let the relationship set be “Teaches” then, it is represented in the relation(table) as:

Teaches

Tid	Sid

- Tabular representation of generalization:

-create a table for the superclass entity set. For sub class entity set, create a table which includes the column of the subclass plus PK of superclass.

Example: let a super class be- **Account**(Accno, balance), subclass be- **Saving Account**(interest_rate) and **Checking Account**(overdraft_amount). It’s representation is as follows:

Account

AccNo	Balance

Saving Account

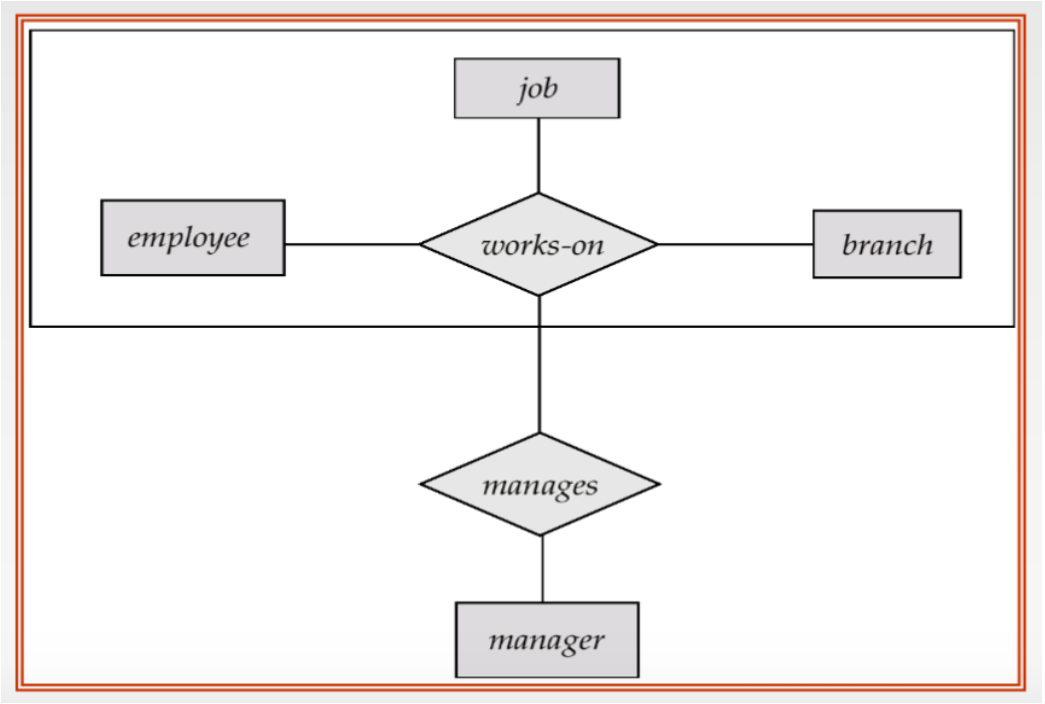
AccN0	Interest rate

Checking Account

AccNO	Overdraft amount

- To represent an aggregation, we count the primary keys of the aggregate relationship and primary key of the associate entity set.

A Table is formed with all the primary keys of entities like *employee, job, branch, manager*



Any Queries?