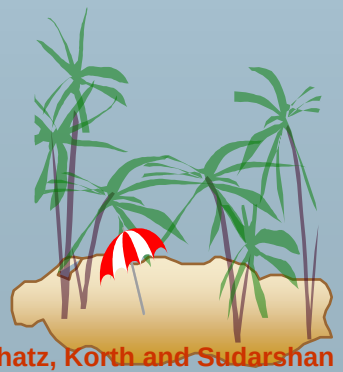
Data Base Management System

Dr. Anjali Gautam



Course Aim

- **This course is relevant to students because information / data form the foundation of any business/education enterprise.**
- **Thus a thorough understanding of how to manipulate, design and manage databases.**
- *Behind the development and design of this course is to know how to design, manipulate and manage databases.*
- The course participants are exposed to the various forms, types and models of database systems to enable them make viable choices.
- The **ultimate aim** is to encourage the usage of database management systems for effective business decision making.

About Course Syllabus

Course Plan:

Component	Unit	Topics for Coverage	Chapter No
Component 1	Unit 1	Evolution of Data Centric Systems, Need & Purpose of Database Systems. Database User categories and Database Architecture,	Korth sixth edition Chapter 1
	Unit 2	Data Modeling-ER Diagrams, Case Study discussions for ER Diagrams.	Chapter- 7
		Relational Database-concepts of Keys: Super Key, Primary, Candidate and Foreign Keys. Weak Entity Surrogate Keys.	Chapter-2
Component 2	Unit 3	Integrity constraints, Referential constraints and SQL Constructs. Database languages-Relational Algebra, SQL,PL-SQL.Triggers Embedded SQL and Application Programming Interfaces. Database connectivity-ODBC	Chapter -6
	Unit 4	Relational Database Design issues and Normalization.Functional Dependencies and Various Normal Form Tests.	Korth &.Fundamentals of Database Mangement System–Elmasri Navathe

Text Book:

1. Database System Concepts by AviSilberschatz,Henry F. Korth,SSudarshan.
2. Fundamental of Database Systems- Elmasari, Navathe

References:

1. Introduction to Database Systems- B.C. Desai
2. Database Management Systems by Jerry Post
3. Ramakrishnan, Gehrke, Database Management System

“ We are pouring in data but starving in Knowledge ”

Jiawei Han

Database Management System

- Data – Information – Knowledge
- Chronology: Data & Knowledge Engineering
- DBMS Definition & Application
- Why DBMS ?
  - Developing Information Systems
  - Transaction Management
  - DBMS Types & users
- DBMS languages
  - DDL Vs. DML
- DBMS Architecture
  - Client-server based
  - Web-based architecture
- DBMS Implementation
  - Data Modeling
    - E-R Diagram
    - From diagram to Tables
  - Data Base design
  - Data Querying: Concepts & Constructs
  - Query / Report format design, development.
- Recent trends in DBMS

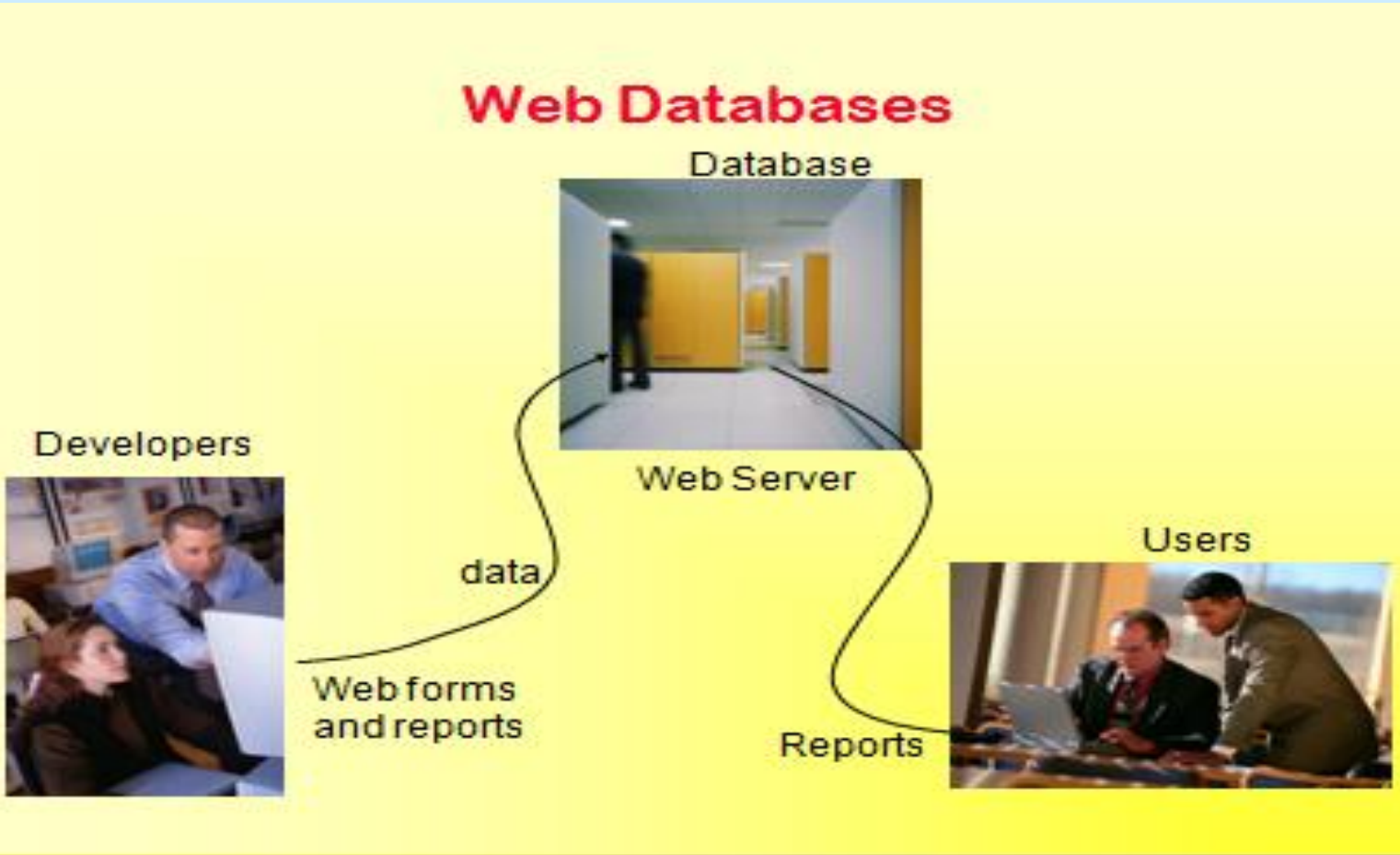
Database Management System (DBMS)

- **Data Base - Collection of interrelated data**
  - **A database is an organized collection of structured information, or data, typically stored electronically in a computer system.**
- **DBMS** is designed to manage large bodies of data, contains information about a particular enterprise.
- DBMS - ***Set of programs to access the data.***
- ***DBMS provides an environment that is both convenient and efficient to use.***
- **Database Applications:**
  - Banking: all transactions uses DBMS
  - Railways: reservations, train-schedules (first database in a geographical distributed)
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions
  - Accessing Data through Mobile phones via Internet…
- Databases touch all aspects of our lives.

Our Approach….

- **Conceptual understanding & then Implementation...**
  - **Case Study & Modeling discussions**
    - **Through Class Assignments, Group presentations**
  - **Practical on SQL, PL/SQL on Oracle**
  - **Database Designing: Concepts & Tests.**
  - **Recent Trends in DBMS**
- **Reference Material:**
  - Database system concepts by Abraham Silberschatz ; Henry F. Korth ; S. Sudarshan
  - Fundamentals of Database Systems by Elmasri, Ramez; Navathe, Shamkant B.
  - Book on SQL / Oracle : Complete Reference

Web Enabled Databases



Information systems

- Did we ever bother to see how this computerization of railways reservation system would have first conceived, designed, developed and implemented.
- Database designing is important step for developing such a system ,
- Same is true for developing a business information system for a big corporate house like Hindustan Lever , Godrej, Reliance group of companies etc.
- Database system has evolved a "**systematic mechanism for this aspect of developing information system** and the very initial step is called "Modeling "

Developing Information systems

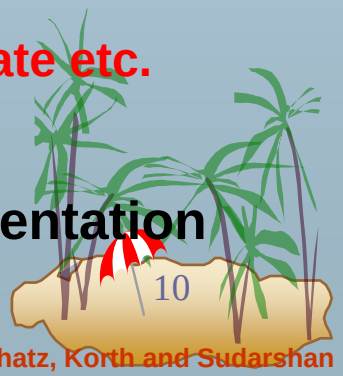
- The purpose of a implementing a database system in any organization is to develop an effective Information System in the organization.

What is information system?

- Information system is an "computerization of some existing manual system which after automation provides useful information to the targeted user".
- One of the most popular example of an information system is computerization of "Railway Reservation Systems"
- Developed by "CRIS" [Centre for Railways Information Systems] Is providing very valuable information like

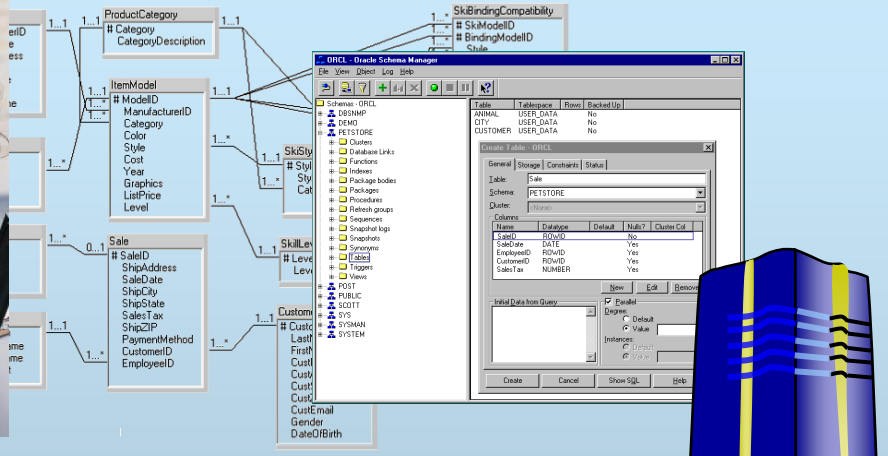
1. Availability of trains between two stations .
2. Availability of Tickets in a particular Train, on particular date etc.
3. Booking a Ticket online.

- Being used as Web based system it is sound implementation of DBMS !



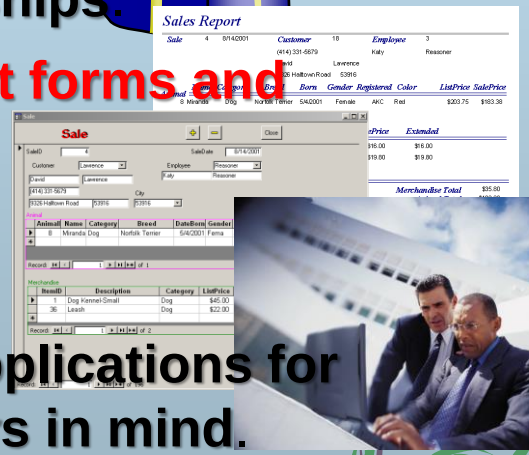
DBMS Application Design

1. **Identify business rules.**



2. **Define tables and relationships.**

3. **Create input forms and reports.**

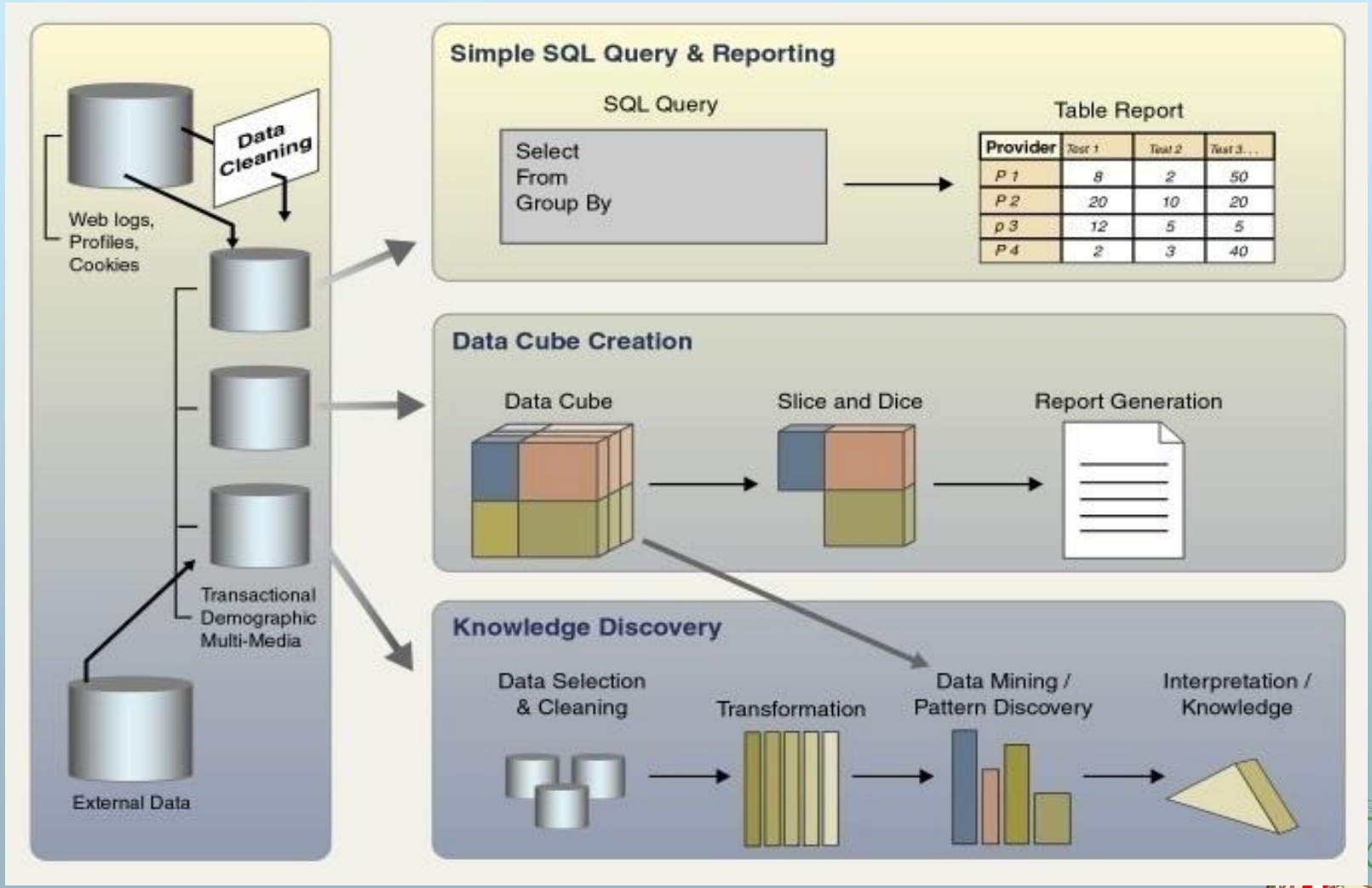


4. **Develop applications for different users in mind.**

Data based Information Systems: Chronology

Evolutionary steps	Business query	Enabling technologies	Product vendors	Characteristics
Data Collection (1960s)	**" What was my total revenue in the last five years"**	**Computers, tapes, disks**	IBM	**Retrospective, static data delivery**
Data base management system 1980s)	**" What were A.C unit sales in New England last March?**	**Relational Databases, SQL, ODBC (Open Database Connectivity)**	Oracle, Sybase, Informix, IBM,Microsoft	**Retrospective dynamic data delivery at record level.**
Data Warehousing & Decision support systems1990s)	**" What were A.C.unit sales in New England last March? Drill down to Boston."**	**OLAP (Online analytical processing,), Multi - dimensional databases, DW**	Pilot, Comshare, Arbor,Cognos, Microstrategy.	**Retrospective, dynamic data delivery at multiple levels.**
Data Mining (Emerging)	**"What's likely to happen to Boston's A.C.unit sales next month? Why ?**	**Advanced algorithms, multiprocessor computers, massive databases.**	Pilot, Lockheed, IBM, SGI, Mineset etc.	**Prospective, proactive information delivery.**

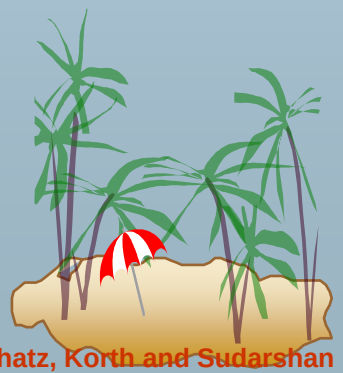
Information Systems Categories



DATA to Information

- I want to go Allahabad to Ahmedabad and I want to know which trains, how much fare & ticket availability, so that I can go to reservation clerk or railway reservation site for reservation, I will submit my input data & railway reservation system gives me information about:

- How many trains are there.
- What is availability of Tickets
- What is fare in different class of travel.

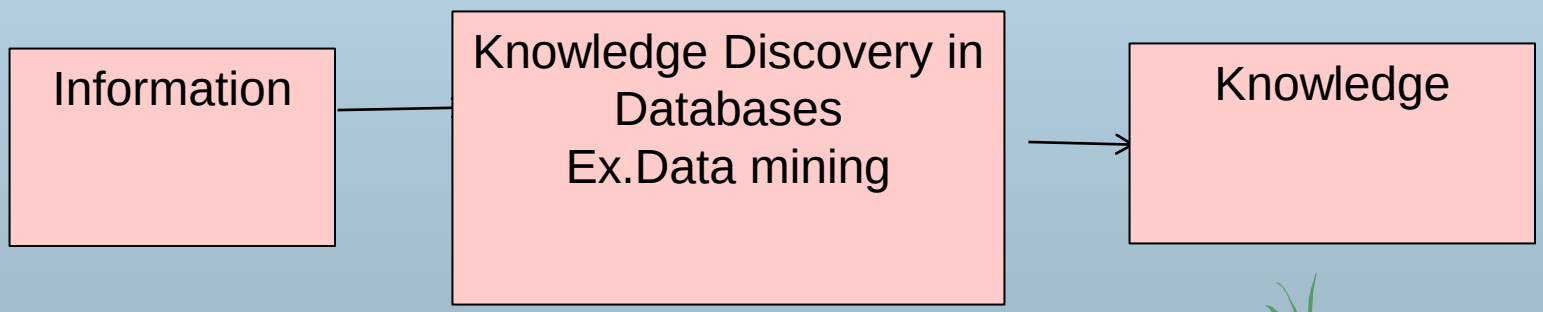


Information to…Knowledge

- I am Suffering from cold and cough I go to a Doctor, Doctor prescribes his prescription advices to take hot water, precautions and gives me prescription of medicine.
- Has doctor provided me information or given treatment based on his knowledge

Cont….

- This is information processed along with experience of doctor or produce “insightful”Knowledge”.

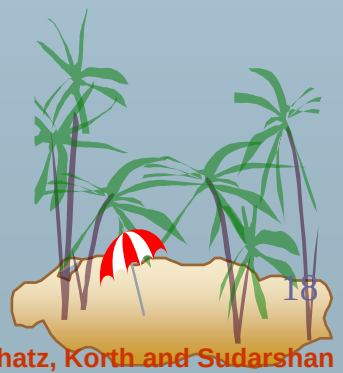


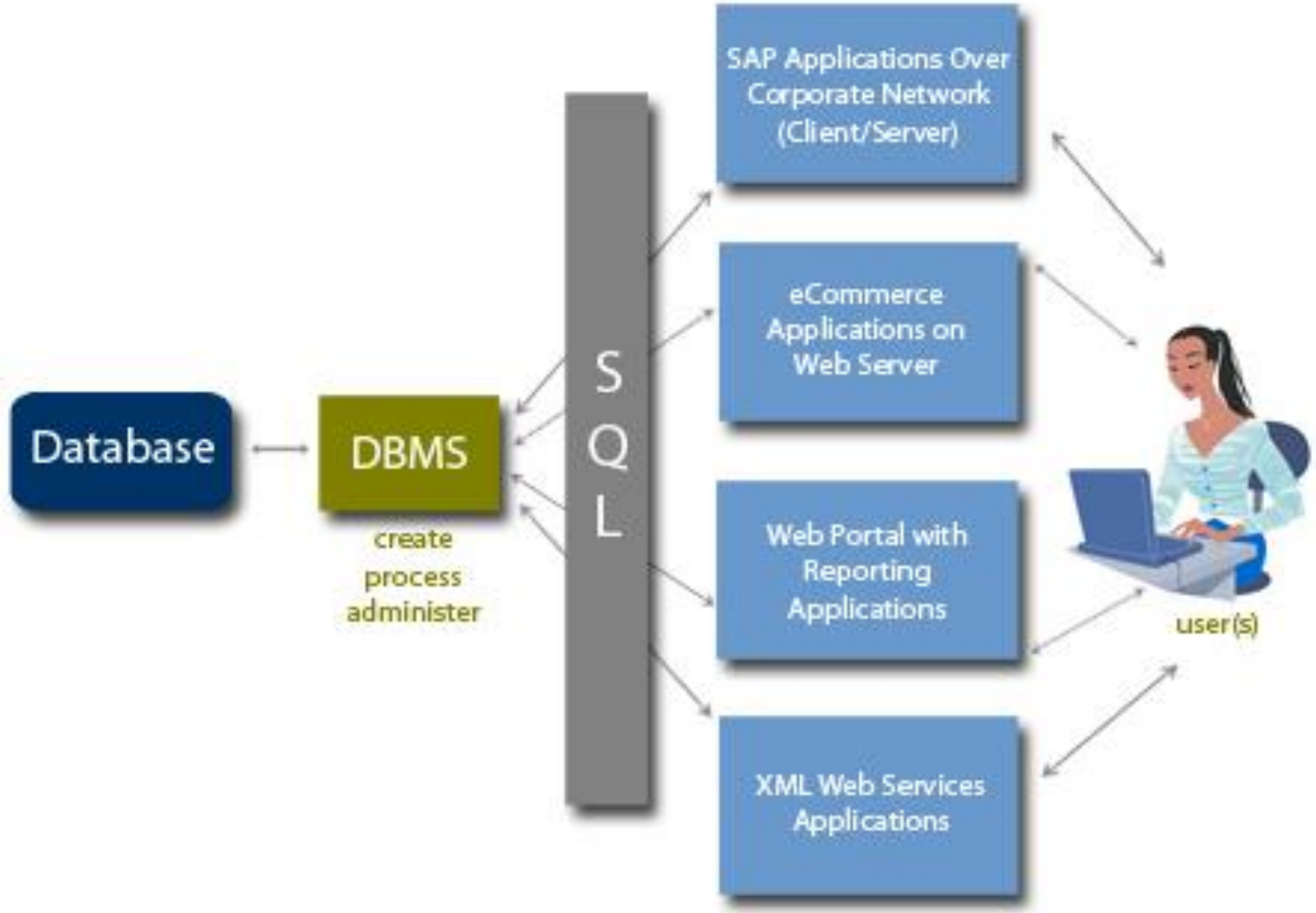
Data, Information & Knowledge

- Computer processing requires **data,** which is a collection of raw facts, figures and symbols, such as numbers, words, images, video and sound, given to the computer during the **input phase.**
- Computer processes the data to create **Information** which is data that is organized, meaningful, and useful.
- During the **output Phase**, the information that has been created is put into some form, such as a printed report.
- A **DBMS** processes the data & provide information.
- **Knowledge** is not simply the information presented, but is information further processed with intelligent mechanism incorporating experience, domain knowledge and specialized techniques.
- To generate (business) insight / strategy based on the **Knowledge** acquired from IT based systems. KDD (**Knowledge Discovery in Database**) is an significant concept related to data mining and business intelligence.

DBMS: Definition

- Database
  - A collection of data stored in a standardized format, designed to be shared by multiple users.
- Database Management System
  - Software that defines a database, stores the data, supports a query language, produces reports, and creates data entry screens.




Database
DBMS
create
process
administer
S
Q
L
SAP Applications Over Corporate Network (Client/Server)
eCommerce Applications on Web Server
Web Portal with Reporting Applications
XML Web Services Applications
user(s)

Database Management system

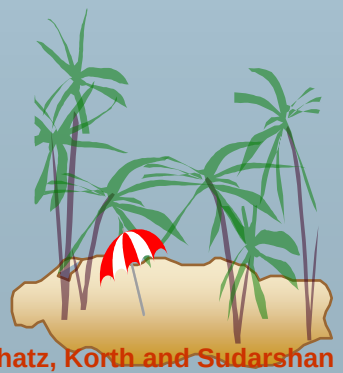
- **Purpose of Database Systems**: **Developing Information system**
- **Data Models**
  - DBMS Languages : Procedural Language Vs Non-procedural Language
  - SQL & PL SQL
- **Transaction Management & Storage Management**
- **Database Administrator**
- **Database Users**
- **Overall System Structure**

Database Management System

- Data – Information – Knowledge
- Chronology: Data & Knowledge Engineering
- DBMS Definition & Application
- Why DBMS ?
  - DBMS Implementation
  - DBMS Types & Users
- DBMS Architecture
  - Client-server based
  - Web-based architecture
  - Purpose of Data base Management System
- DBMS languages
  - DDL Vs. DML
- DBMS Implementation
  - Data Modeling
    - E-R Diagram
    - From diagram to Tables
  - Data Base design
  - Data Querying: Concepts & Constructs
  - Query / Report format design, development.
- Recent trends in DBMS

Data Model

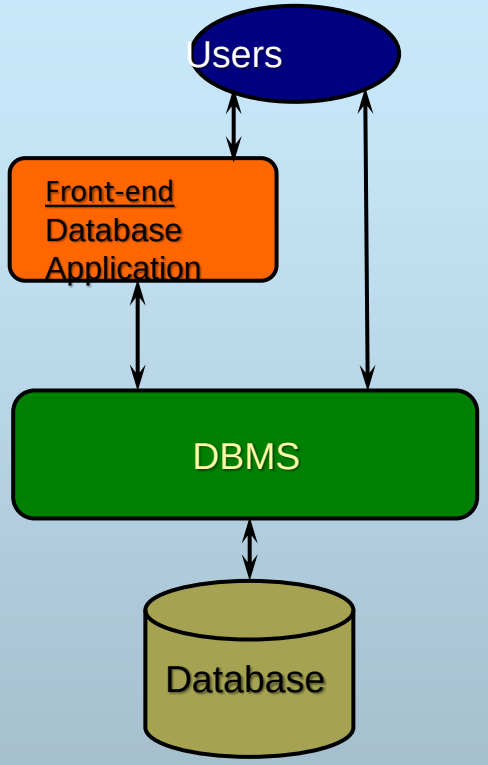
- Modeling of data description, data semantics, and consistency constraints of the data.
- It provides the conceptual tools for describing the design of a database at each level of data abstraction.
- Four Models
  - Relational Data Model (Row,Column)
  - Entity-Relationship Datal Model (objects)
  - Object-based Data Model (extension of ER, functions)
  - Semistructured Data Model (specification for individual data items of same type).



Database implementation

- Database
- E-R (Entity-Relationship) Modeling
- Conversion of ER model into Table
- Relational Database software
- Querying and Table creation.

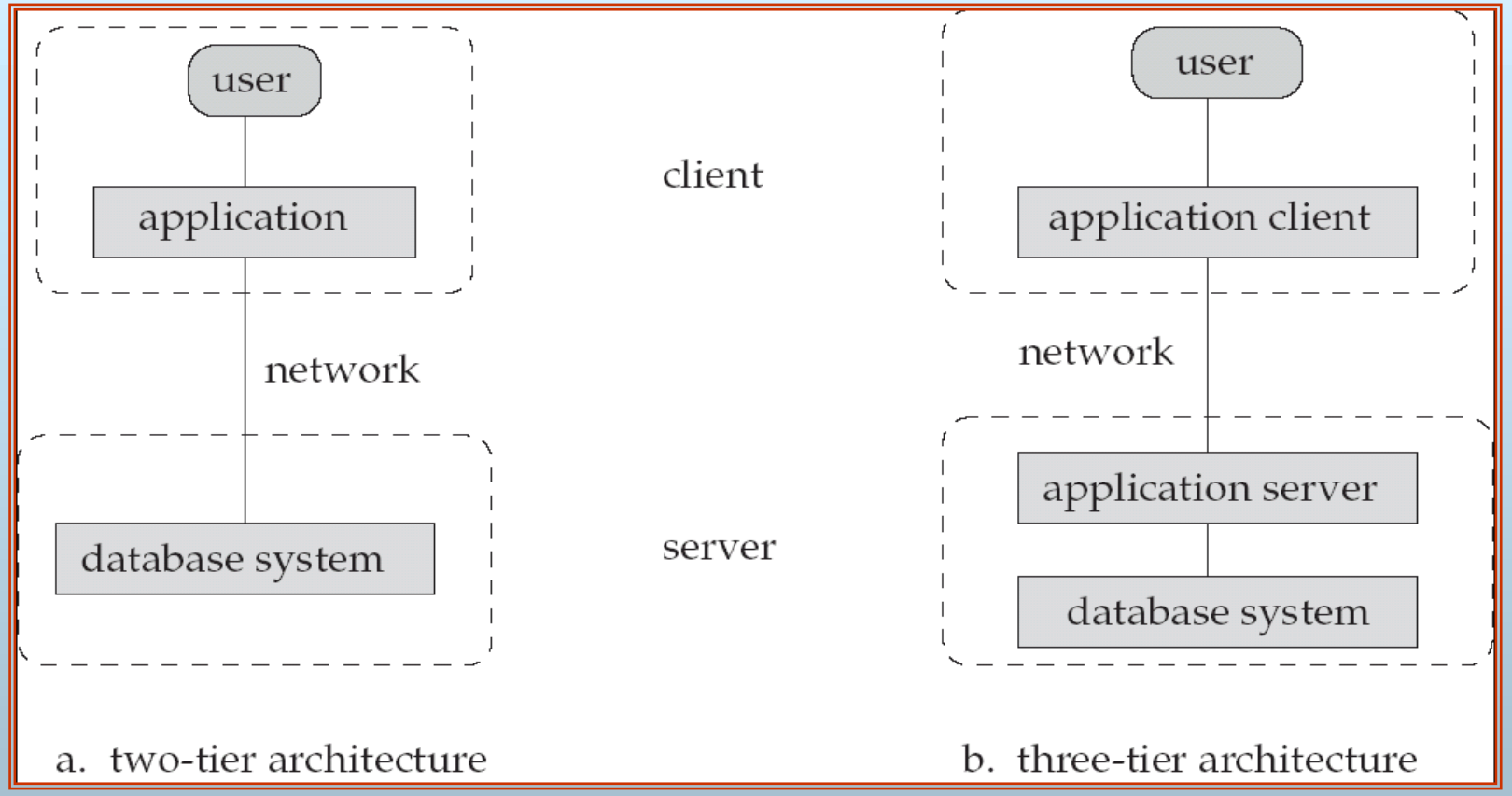
Data Base Management System



Ordinary Users interact with DBMS via Menu in Front-End Application like **Visual Basic**.

Whereas Sophisticated Users & Database administrator **interacts with DBMS via** SQL **interface or directly with Oracle / MS SQL Server / IBM DB-2**

Database Application Architectures

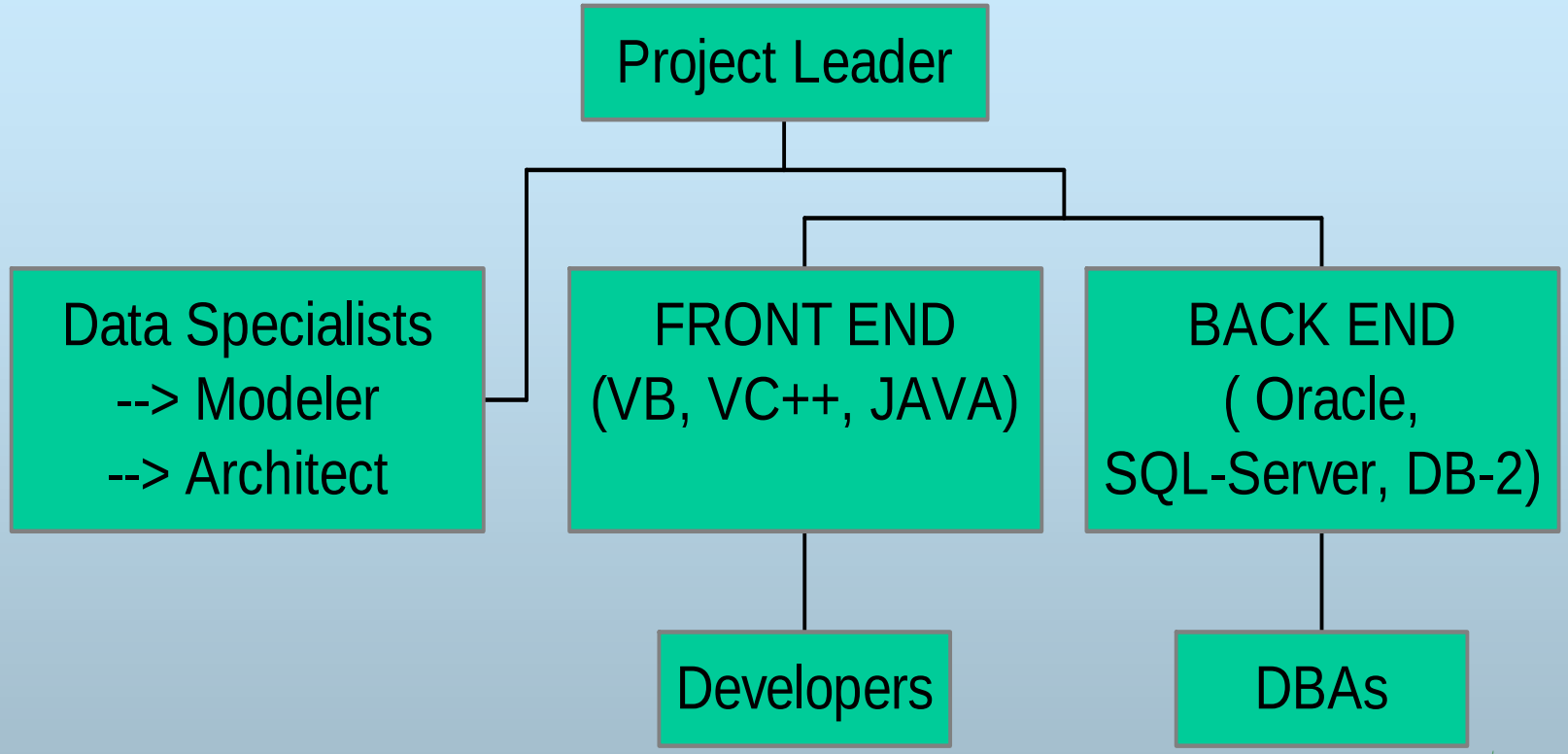


Old

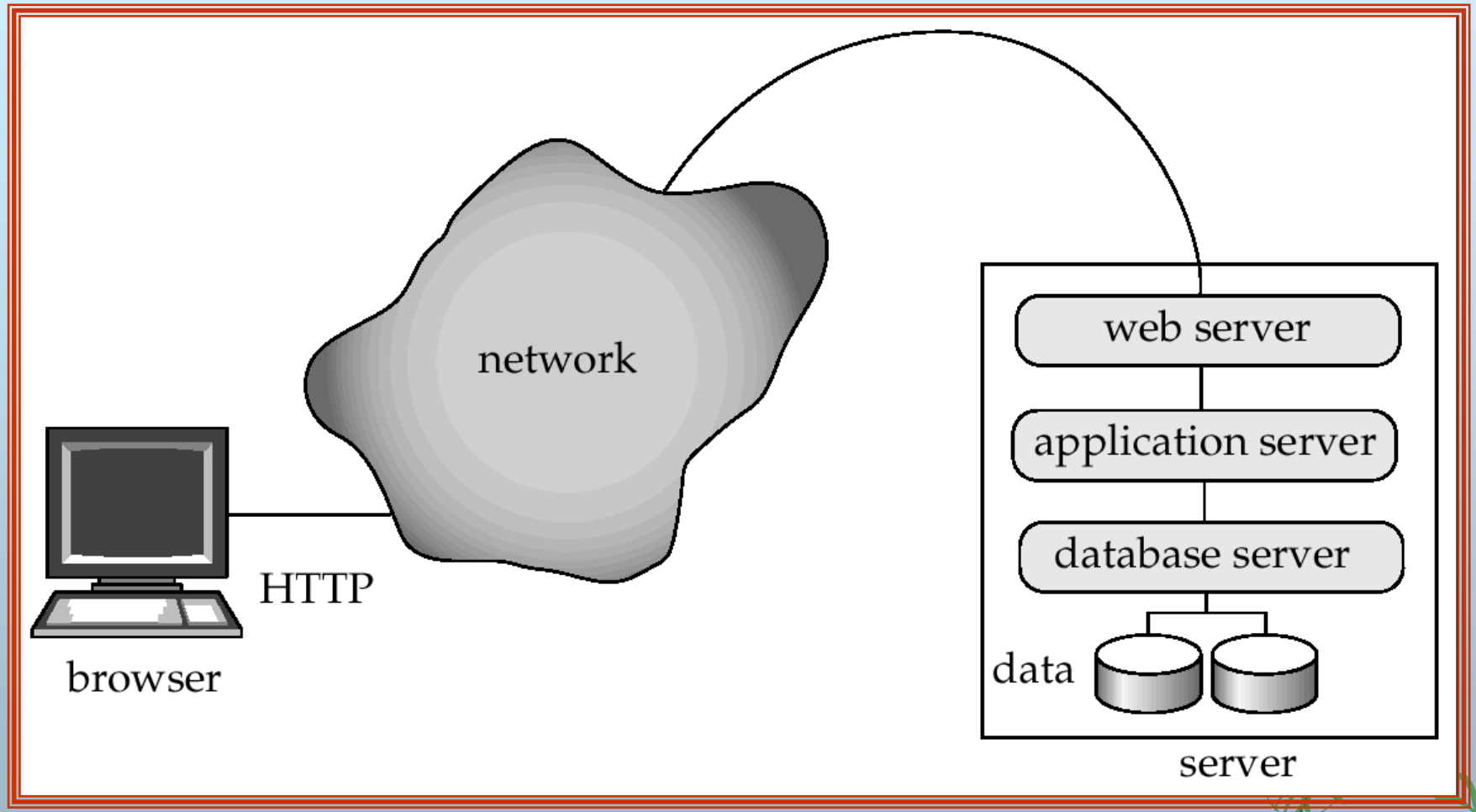
Modern

- **Two-tier architecture**: E.g. client programs using ODBC/JDBC to communicate with a database
- **Three-tier architecture**: E.g. web-based applications, and applications built using "middleware"

DBMS Project Team

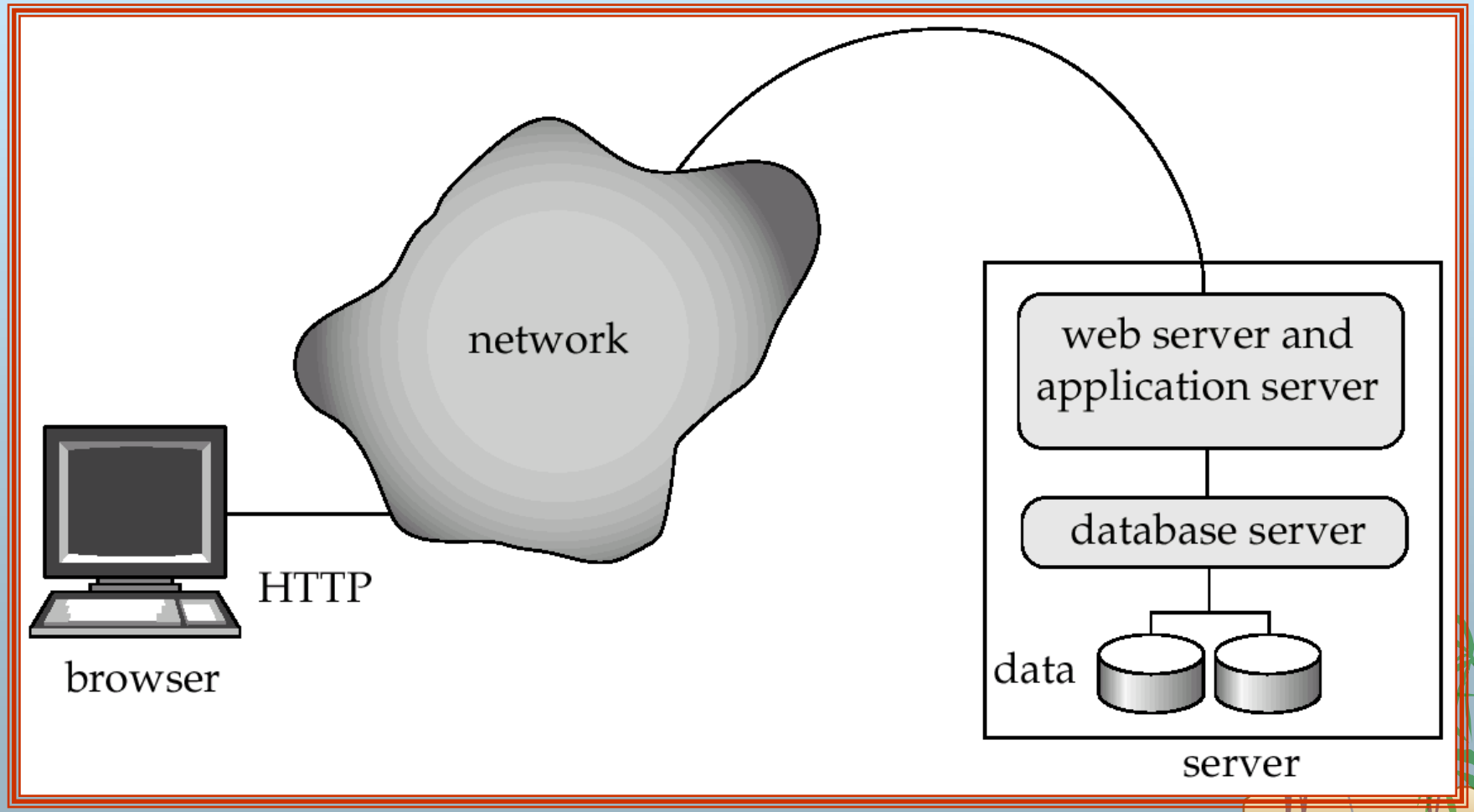


Three-Tier Web Architecture



Two-Tier Web Architecture

- Multiple levels of indirection have overheads
  - Alternative: two-tier architecture



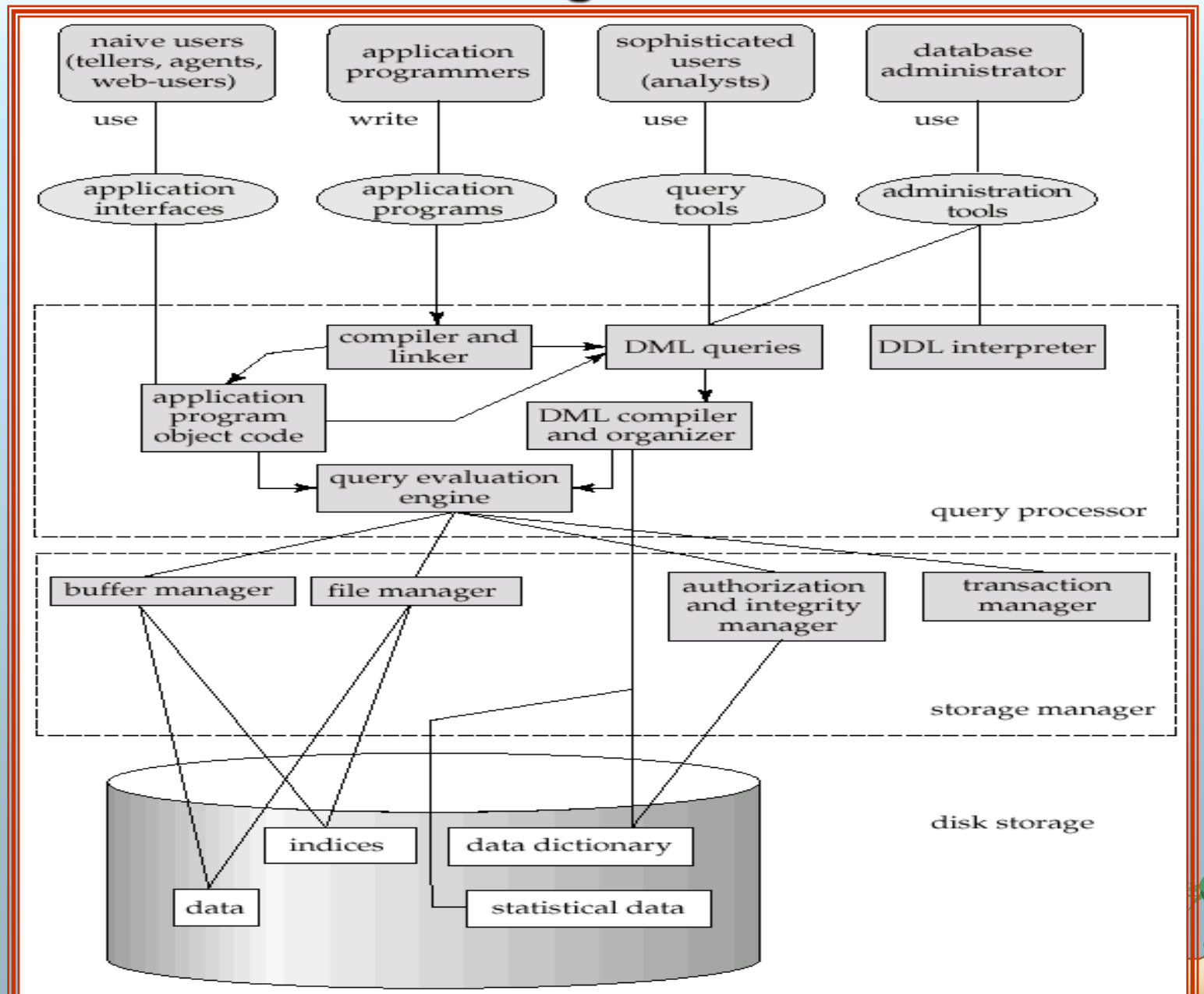
Database Users

- Users are differentiated by the way they expect to interact with the system
- Application programmers – interact with system through DML calls
- Sophisticated users –give requests in a database query language, who have their own way of accessing
- Specialized users – write specialized database applications that do not fit into the traditional data processing framework. Engineers, scientist, business analyst
- Naive users – invoke one of the permanent application programs that have been written previously
  - E.g. people accessing database over the web, bank tellers, clerical staff

Database Administrator

- **Coordinates all the activities of the database system;**
- **The database administrator has a good understanding of the enterprise's information resources and needs.**
- **Database administrator's duties include:**
  - Schema definition (how data is organized)
  - Storage structure and access method definition
  - Schema and physical organization modification
  - Granting user authority to access the database
  - Specifying integrity constraints
  - Monitoring performance and responding to changes in requirements

Data base User categories & Architecture

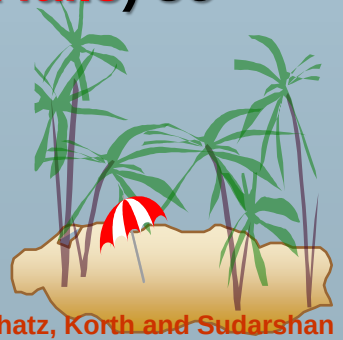


Database Management System

- Data – Information – Knowledge
- Chronology: Data & Knowledge Engineering
- DBMS Definition & Application
- Why DBMS ?
  - Developing Information Systems
  - Transaction Management
  - DBMS Types & users
- DBMS Architecture
  - Client-server based
  - Web-based architecture
  - Purpose of Data base Management System
- DBMS Implementation
  - Data Modeling
    - E-R Diagram
    - From diagram to Tables
  - Data Base design
  - Data Querying: Concepts & Constructs
- DBMS languages
  - DDL Vs. DML
  - Query / Report format design, development.
- Recent trends in DBMS

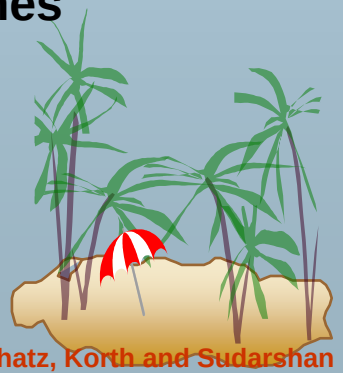
Purpose of Database Systems

- **In the early days, database applications were built directly on top of file systems** using any programming language…..
  - Using C / C++ and its files for storing data
- ***Transaction Management*** **is typical example which highlights the significance of a DBMS.**
  - **In case of *transferring funds from one bank A/C to other* if System failure occurs !**
  - **This failure may occur at any stage & the system failures (e.g., power failures, operating system crashes) will result not only as a transaction failure but may be in *serious inconsistencies in database.***
  - **Money withdrawn from one Account (and network fails) so not received in the Other A/C. !**
  - **Also in concurrent transactions how to ensure the consistency of the database.**
- **And many other problems….**



Purpose of Database Systems

- Drawbacks of using file systems to store data:
  - Data redundancy and inconsistency
    - Multiple file formats, duplication of information in different files
  - Difficulty in accessing data
    - Need to write a new program to carry out each new task
  - Data isolation — multiple files and formats
  - Integrity problems
    - Integrity constraints (e.g. account balance >0) become "buried" in program code rather than being stated explicitly
    - Hard to add new constraints or change existing ones

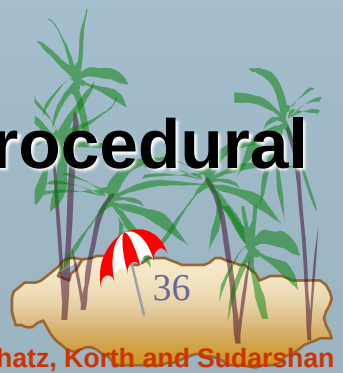


Purpose of Database Systems (Cont.)

- Drawbacks of using file systems (cont.)
  - Atomicity of updates
    - Failures may leave database in an inconsistent state with partial updates carried out
    - Example: Transfer of funds from one account to another should either complete or not happen at all
  - Concurrent access by multiple users
    - Concurrent accessed needed for performance
    - Uncontrolled concurrent accesses can lead to inconsistencies
      - Example: Two people reading a balance and updating it at the same time
  - Security problems
    - Hard to provide user access to some, but not all, data
- Database Management systems offer solutions to all the above problems

Database implementation

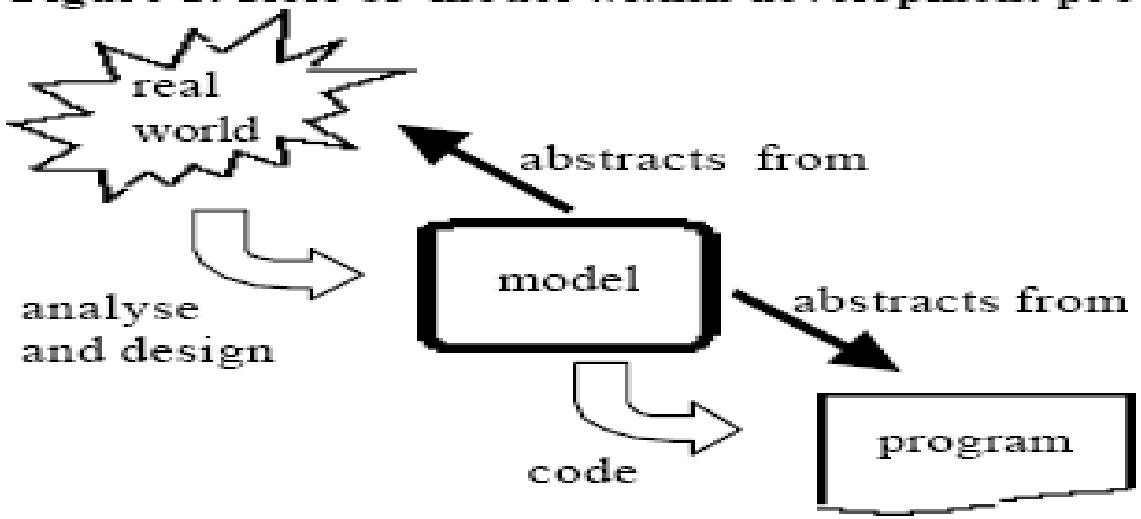
- Date Base Design: 03 Phases
- Modeling:
  - E-R Modeling : Purpose.
    - E-R model for Banking Enterprise, Car Insurance Company
- Conversion of ER model into Table
    - Table Creation
- Data Base Languages & Querying
  - Relation Algebra
  - Query Language : Procedural & Non-procedural
  - DDL, DML



Modeling in DBMS

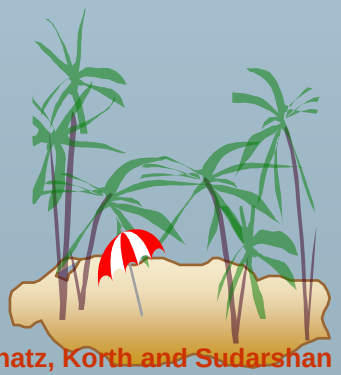
- With the variety of real life scenario to be computerized for Information System, modeling has become the **de-facto** first and significant step in the early phases of a software development process..
- **The most single important ability in modeling is;**
  - to skilfully identify major components of the given scenario and convert them in to model for onward conversion into software systems.

Figure 1: Role of model within development process



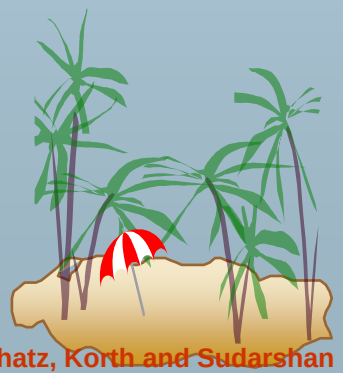
Data Models

- A collection of tools for describing
  - data
  - data relationships
  - data constraints
- Entity-Relationship model
- Relational model
- Other models:
  - Older models:
  - Network model and
  - Hierarchical model



Relational Database Management System

- RDBMS is one of the most successful model of Data Base Management.
- RDBMS uses the concepts introduced first by Researcher Boyce F. Codd.
- In RDBMS is a DBMS in which data is stored in tables and the relationships among the data are also stored in tables.
- The data can be **accessed** or reassembled in many different ways without having to change the table forms.
- In RDBMS large body of data can be modeled and described as group of tables which are connected to each other through Relations.

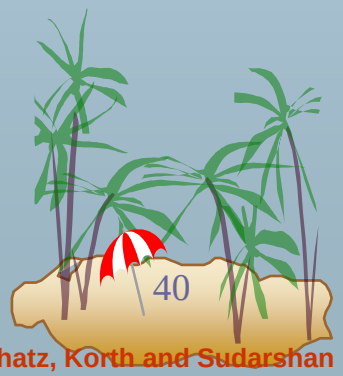


Relational Model

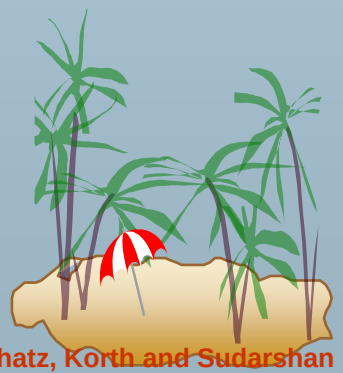
- Example of Tabular data in the Relational model

Attributes

Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
019-28-3746	Smith	North	Rye	A-201



- Design Process Modeling Constraints
- E-R Diagrams
- Design Issues Weak Entity Sets
- Extended E-R Features Reduction to Relation Schemas



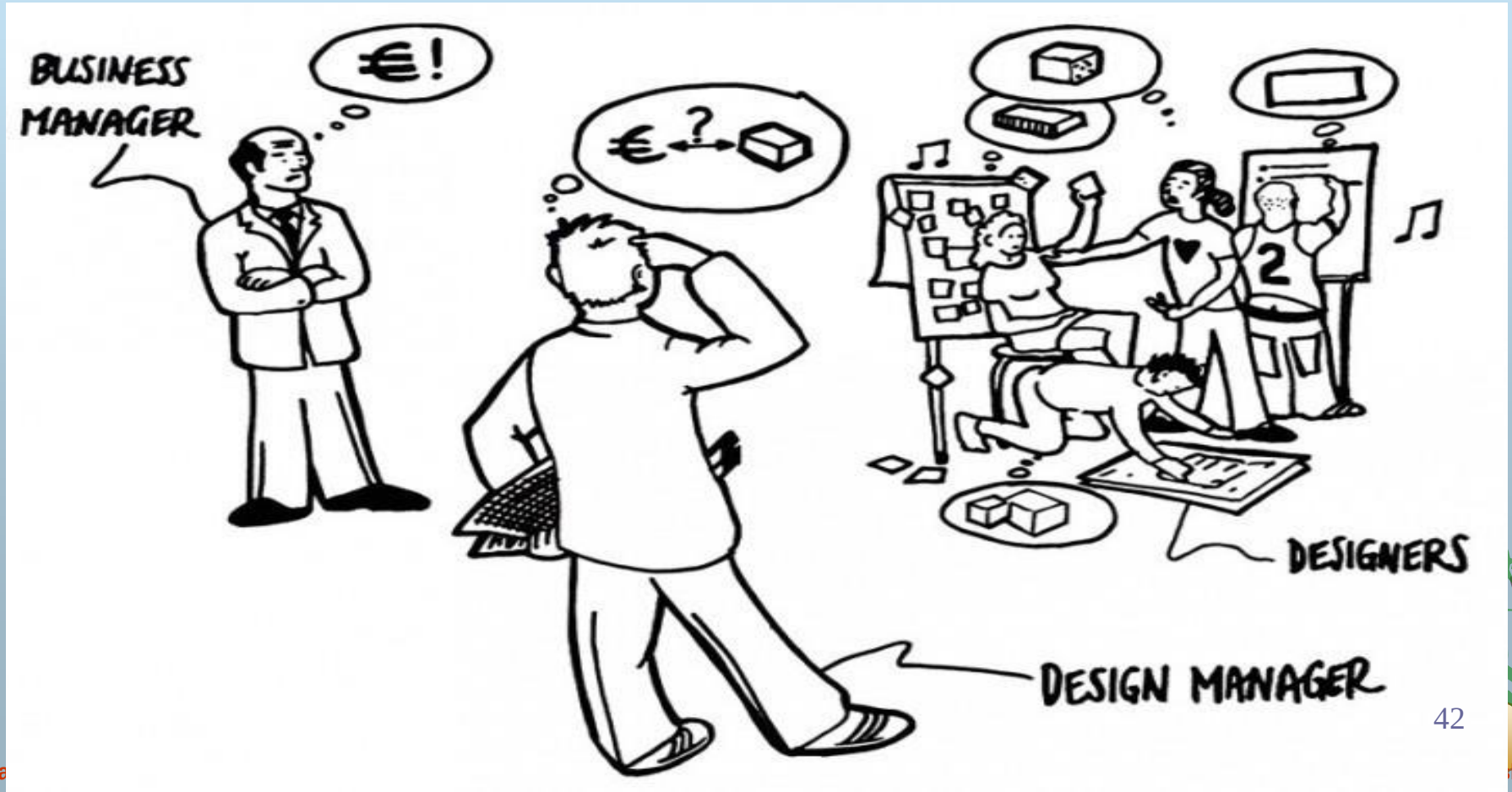
Today

Designing databases

-

i.e., ~~h~~ow to get from your customer's requirements…

...to ~~a~~ set of tables and attributes



Database Design

Three Phases of DB Design

① **Conceptual Database Design.** Produces the initial model of the mini world in a **conceptual data model** (*e.g.*, in the ER model).

② **Logical Database Design.** Transforms the conceptual schema into the data model supported by the DBMS (*e.g.*, the relational model).

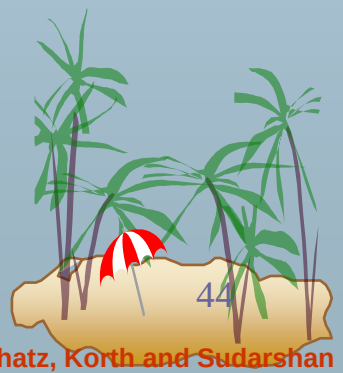
The Normalization Theory in Database also relates to Logical Database Design

③ **Physical Database Design.** Design indexes, table distribution, buffer size, *etc.*, to maximize performance of the final system (subject of "*Datenbanken II*").

(Deciding on the physical layout of the database)

Database Design

- Initial phase: requirements engineering
  - characterize fully the data needs of the prospective database users
  - which data needs to be stored?
    - ...and in which volumes?
    - which queries should be answered?
- **Conceptual schema**
  - which types of entities and relations exist?
  - what attributes do they have?



Database Design

Final phase: from a conceptual to physical data model

The process of designing the general structure of the database:

- Logical Design – Deciding on the database schema. Database design requires that we find a “good” collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical Design – Deciding on the physical layout of the database

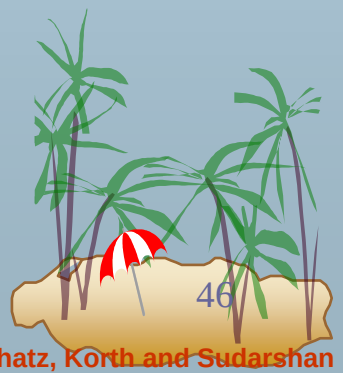
Database Design Approaches

Entity Relationship Model (today)

- Model an enterprise as a collection of *entities* and *relationships*
- Entity: a “thing” or “object” in the enterprise that is distinguishable from other objects
  - Described by a set of attributes
- Relationship: an association among several entities
  - Represented diagrammatically by an entity-relationship diagram

Normalization Theory (coming Classes)

- Formalize what designs are bad, and test for them



Entity-Relationship Model

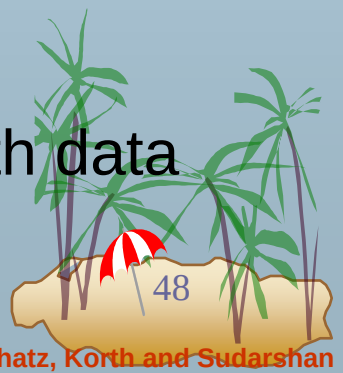
Diagrams

Purpose of E/R Model

- The E/R model allows us to sketch the design of a database informally.
- Designs are pictures called *entity-relationship diagrams*.
- Fairly mechanical ways to convert E/R diagrams to real implementations like relational databases exist.

It is most Widely used for database design .

- Database design in E-R model is converted into design in the relational model.
- Entity-Relationship is a high-level data model.
- Relational model is a lower level model.
  - It uses a collection of tables to represent both data and the relationships among those data.



ER Model

- Proposed by Peter Chen (1976), today at Louisiana State University (`http://bit.csc.lsu.edu/~chen/chen.html`).
- The ER model comes with a **graphical notation** which helps to establish quick overviews of database application schemas.

  Such **ER diagrams** are also a great way to communicate schemas to non-expert/future DB users.
- There are *no* "ER Model DBMS".

  Instead, we will describe a translation of ER model concepts into the relational model.

Entity Sets

- *Entity* = “thing” or object.
- Eg .book
- *Entity set* = collection of similar entities.
  - Similar to a class in object-oriented languages.
- Attribute = property of an entity set.
- Eg. book has title, author,publisher,price,year publication.
  - Generally, all entities in a set have the same properties.
  - Attributes are simple values, e.g. integers or character strings.

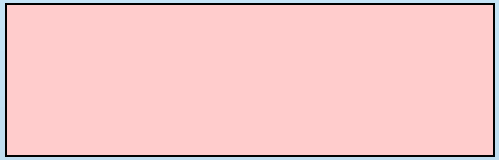
Cont.....

- There are three basic elements in ER models:
  - **Entities** are the "things" about which we seek information.
  - **Attributes** are the data we collect about the entities.
  - **Relationships** provide the structure needed to draw information from multiple entities

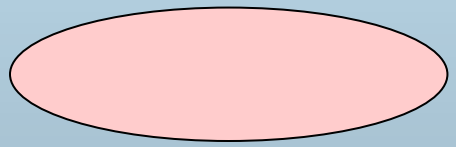
    or
  - Relationship is an Association among several entities.

E/R Diagrams

- In an entity-relationship diagram, each entity set is represented by a rectangle.

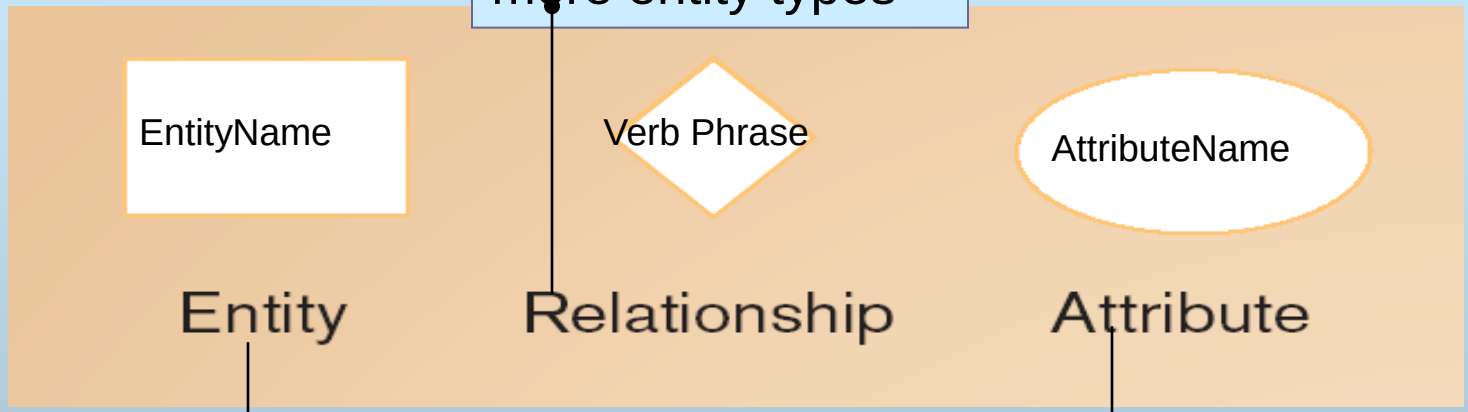


- Each attribute of an entity set is represented by an oval, with a line to the rectangle representing its entity set.



Chen Notation

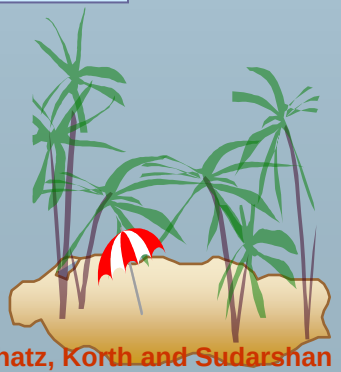
Association between the instances of one or more entity types



Person, place, object, event or concept about which data is to be maintained

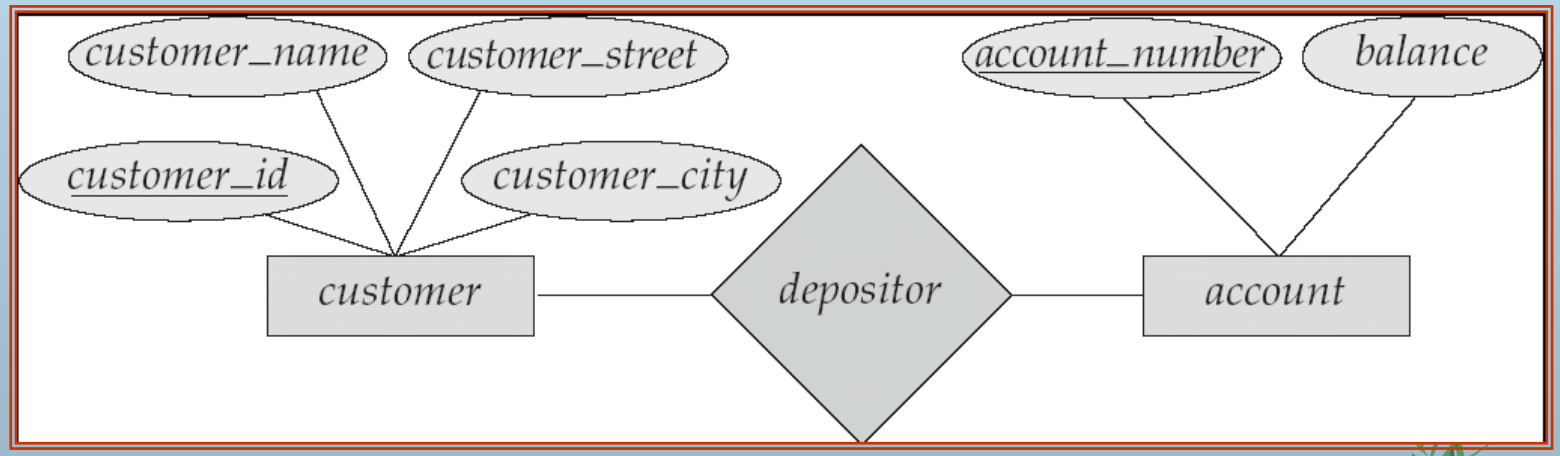
named property or characteristic of an entity

Represents a set or collection of objects in the real world that share the same properties



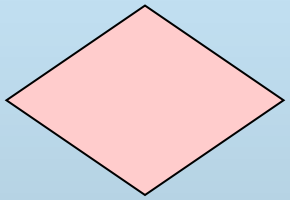
The Entity-Relationship Model

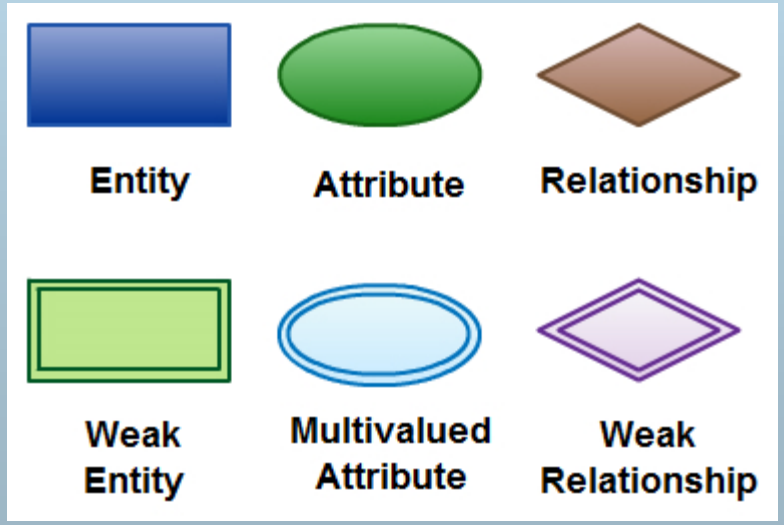
- Models an enterprise as a collection of *entities* and *relationships*
  - Entity: a “thing” or “object” in the enterprise that is distinguishable from other objects
    - Described by a set of *attributes*
  - Relationship: an association among several entities
- Represented diagrammatically by an *entity-relationship diagram:*



Relationships

- A relationship connects two or more entity sets.
- It is represented by a diamond, with lines to each of the entity sets involved.





Relationships

→ Associations between instances of one or more entity types that is of interest

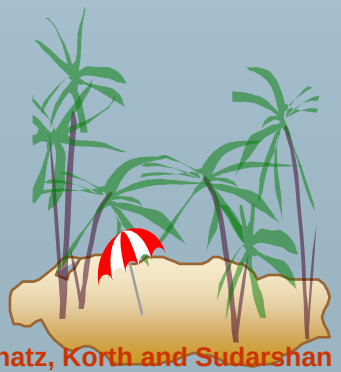
→ Given a name that describes its function.

Relationship name:
writes

Author

Book

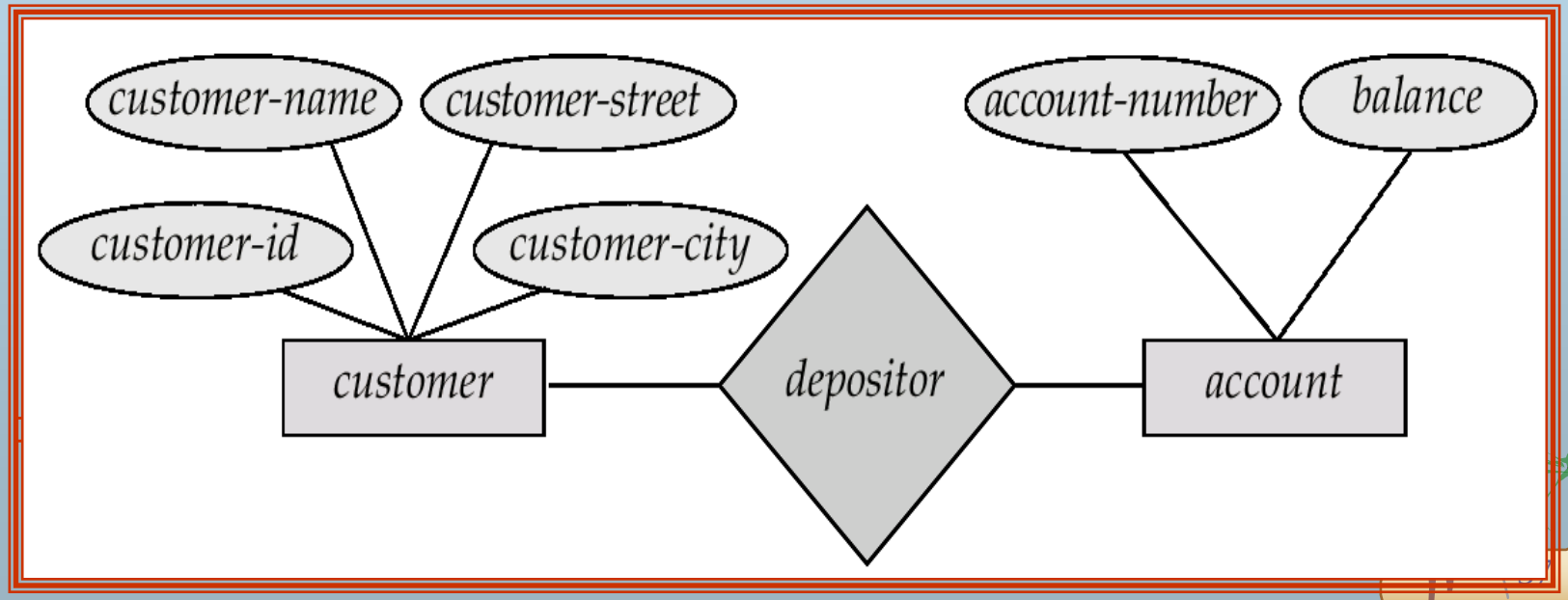
An author writes one or more books
A book can be written by one or more authors.



ER Models

- Example of a Bank: Customer having Depositor A/C

- In a bank each **customer** is “Entity”
  and **bank account** can be considered to be **Entity**, which are described by attributes.
  Here Attributes are ***account -number*** and ***balance,*** describing one particular Account in a bank.
  A **relationship** is an association among several entities. For Ex. a Depositor relationship associates a customer with each account that she has .



A Sample Relational Database

customer-id	*customer-name*	*customer-street*	*customer-city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
019-28-3746	Smith	4 North St.	Rye
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The *customer* table

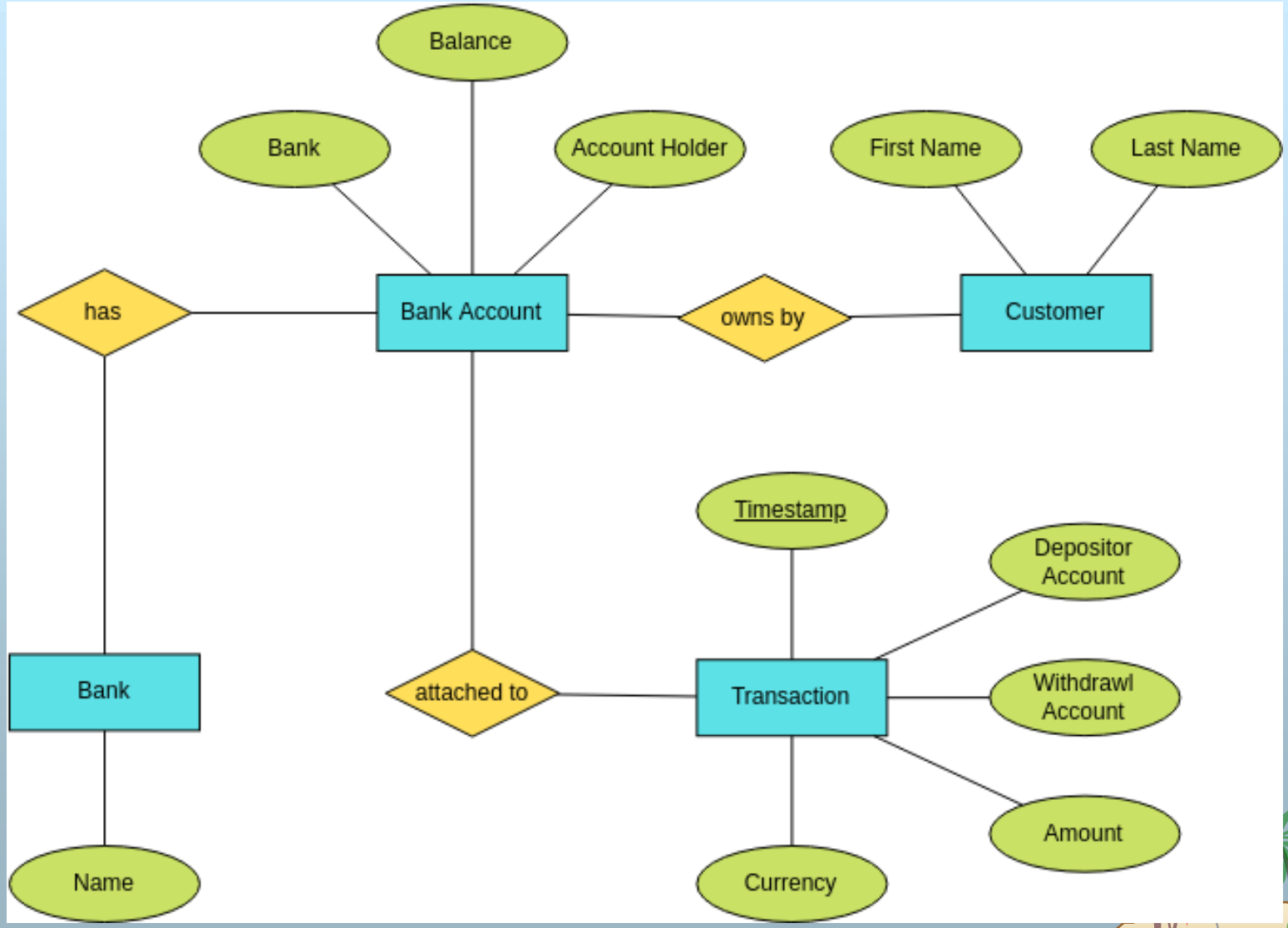
account-number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer-id	*account-number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

E-R Diagram of Banking System



Attributes

- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set.

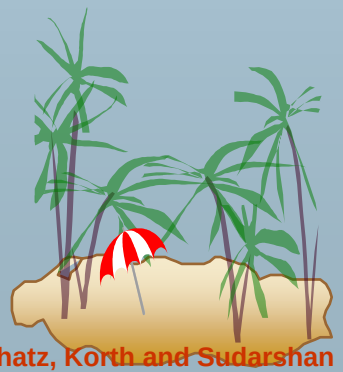
  Example:

  customer = (*customer_id*, *customer_name*, *customer_street*, *customer_city*)
  loan = (*loan_number*, *amount*)

- **Domain** – the set of permitted values for each attribute

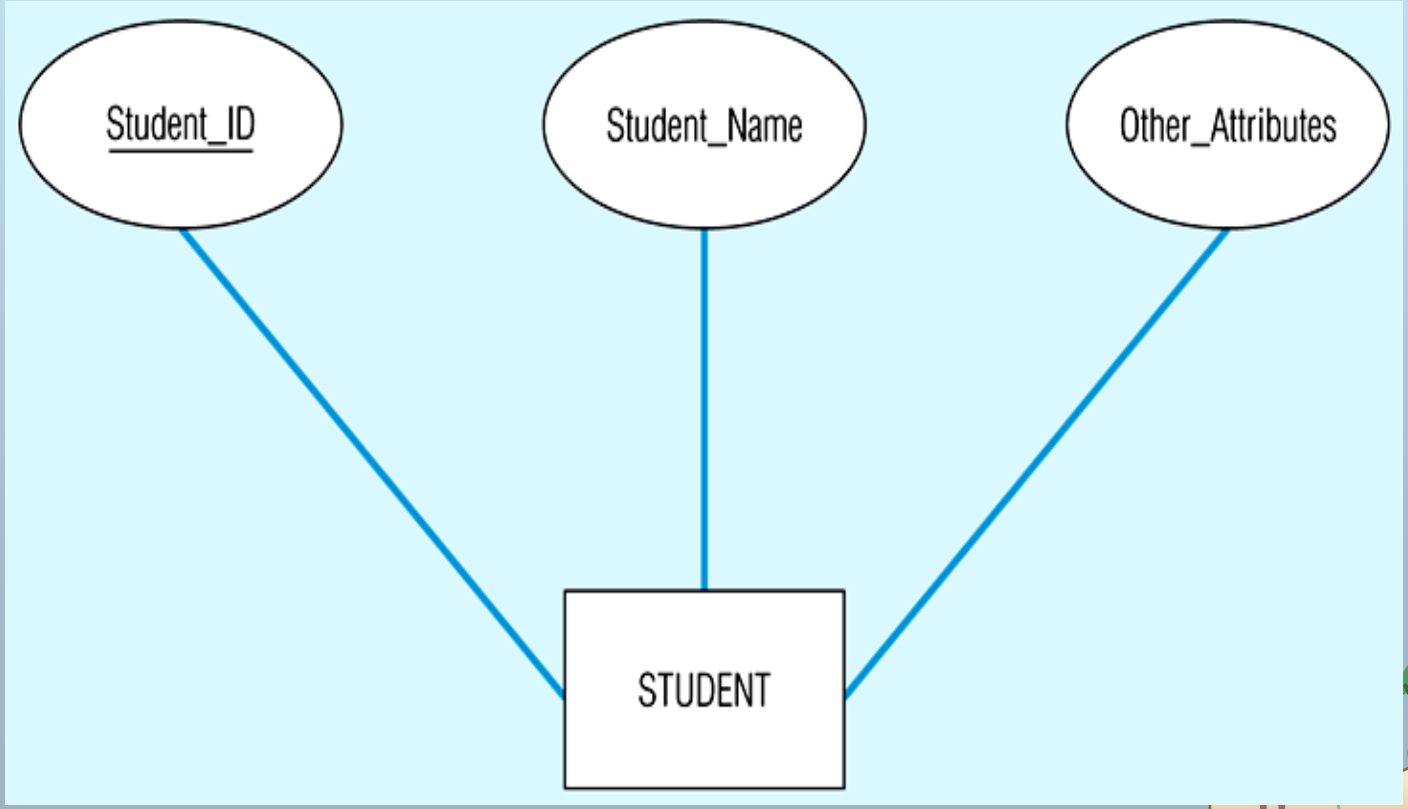
Attribute types:

- ***Simple* and *composite* attributes.**
- ***Single-valued* and *multi-valued* attributes**
- ***Derived* attributes & Null attribute**

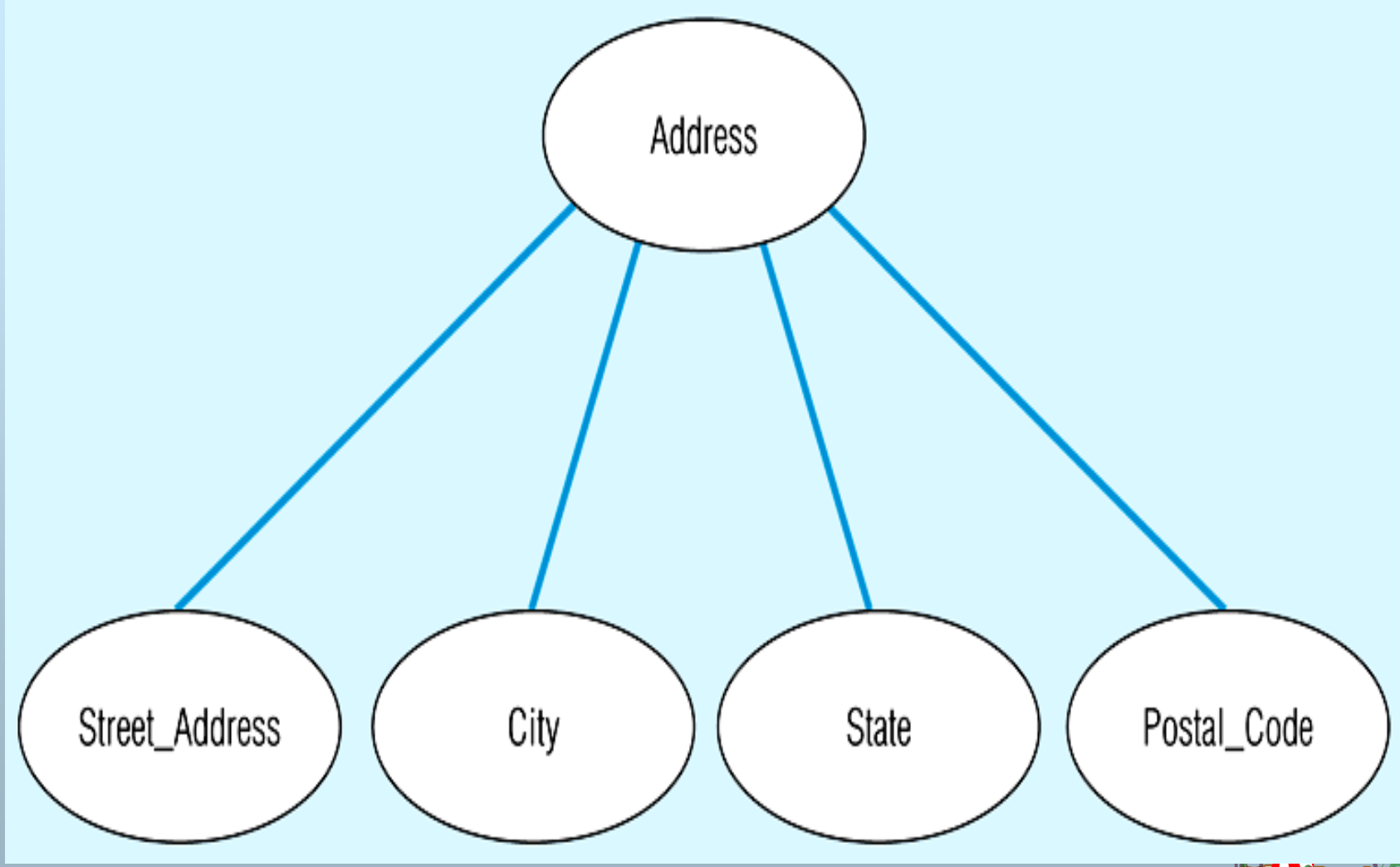


- The attributes that cannot be divided into sub-parts are called simple attributes.
- The attributes which can be divided into sub-parts are called composite attributes.

(a) Simple key attribute



A composite attribute

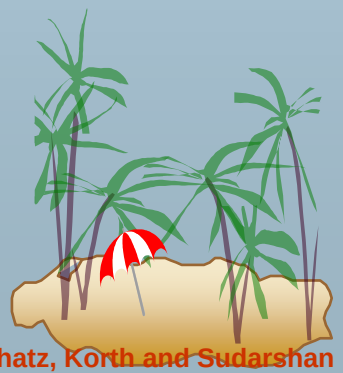


Single-Valued versus Multivalued Attribute

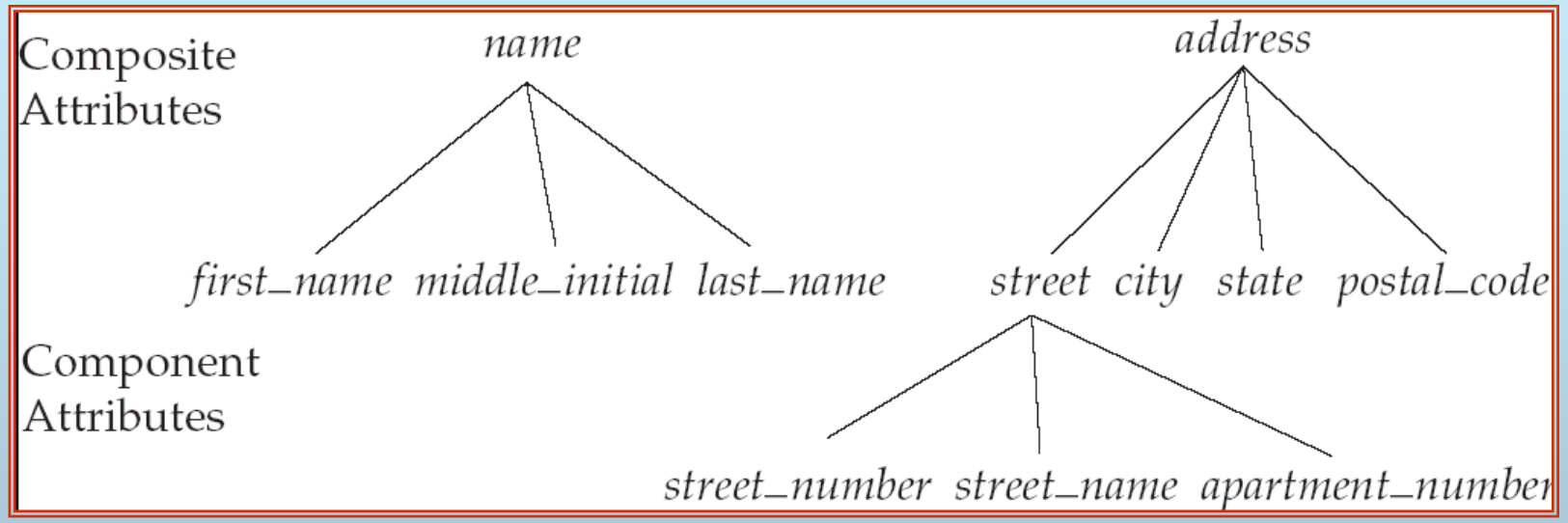
- The attribute have single value for a particular entity.
  - For example Loan Number specific loan entity refers to only loan_number
- It frequently happens that there is an attribute that may have more than one value for a given instance, e.g. EMPLOYEE may have more than one Skill.
- A multivalued attribute is one that may take on more than one value – it is represented by an ellipse with double lines
  - Example: multivalued attribute: *phone_numbers*

Derived Attributes

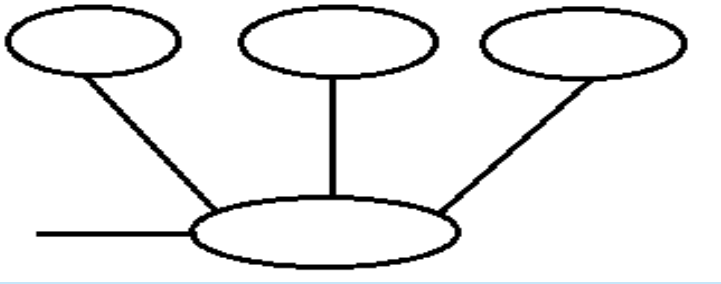
- Some attribute values can be calculated or derived from others
- e.g., if Years_Employed needs to be calculated for EMPLOYEE, it can be calculated using Date_Employed and Today's_Date
- A derived attribute is one whose value can be calculated from related attribute values (plus possibly other data not in the database)
- A derived attribute is signified by an ellipse with a dashed line.
- **Can be computed from other attributes**
  Example: age, given date_of_birth



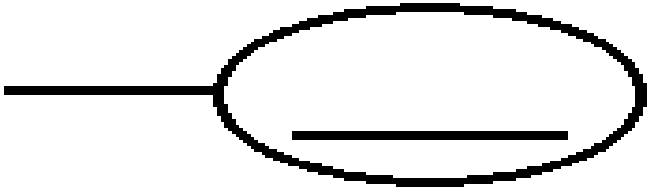
Composite Attributes



Compound/Composite Attribute



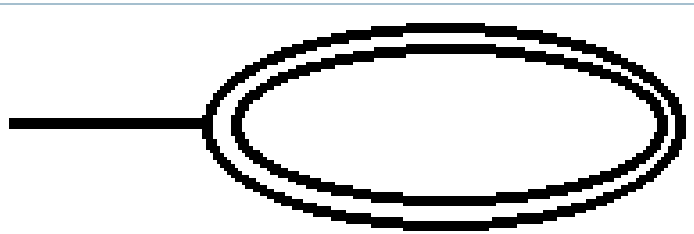
Key Attribute



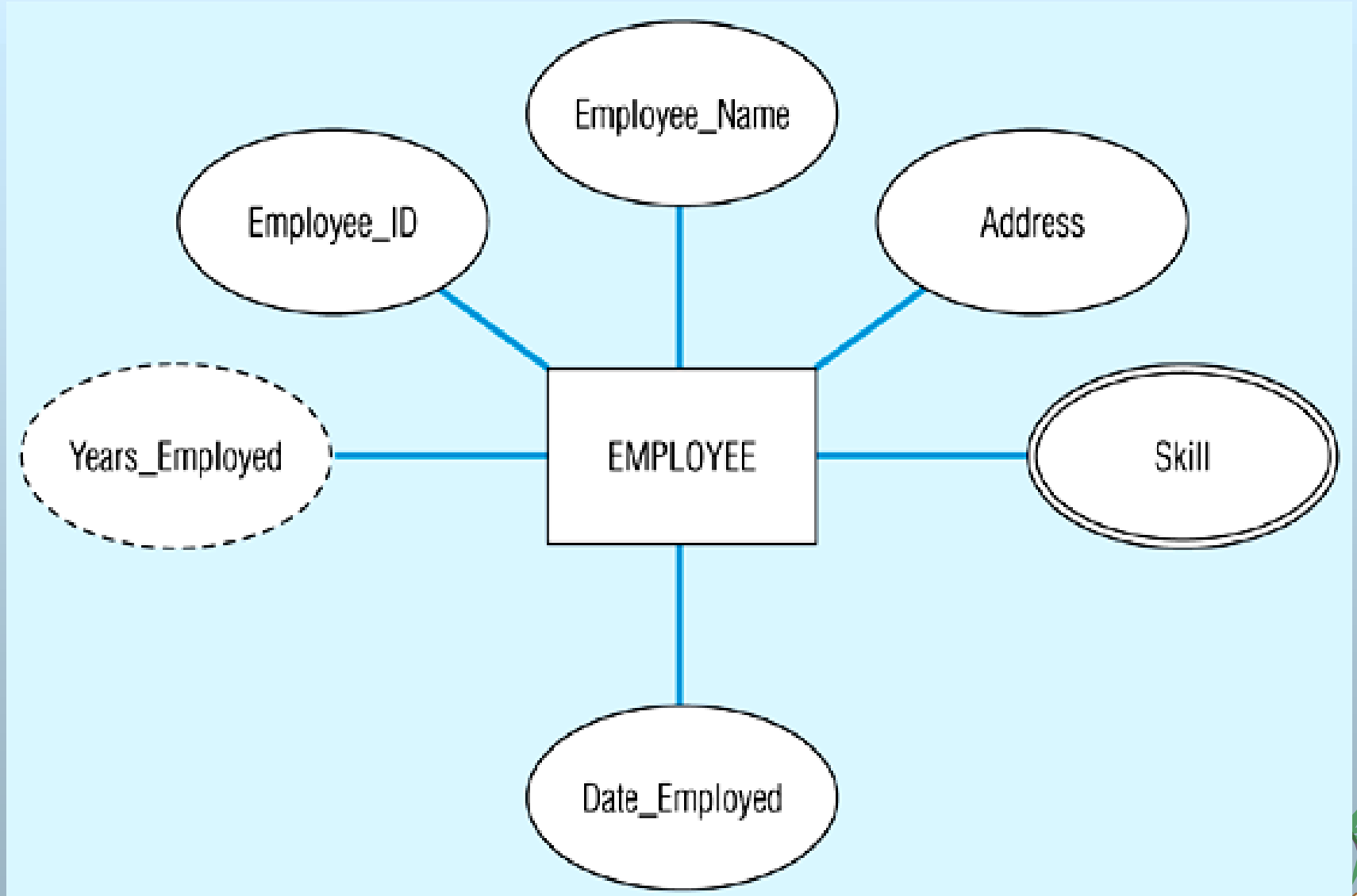
Attribute

Derived Attribute

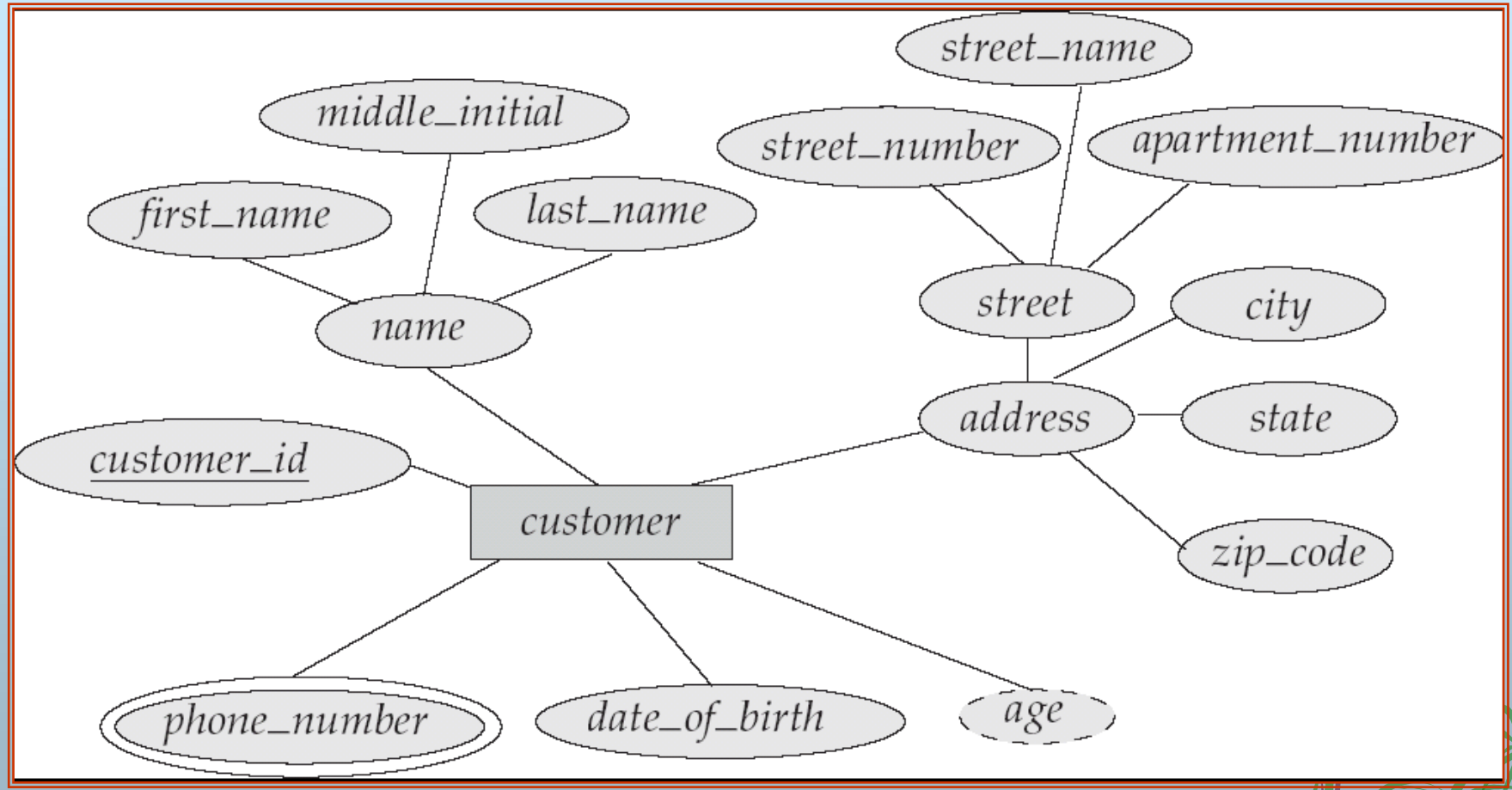
Multi-valued Attribute



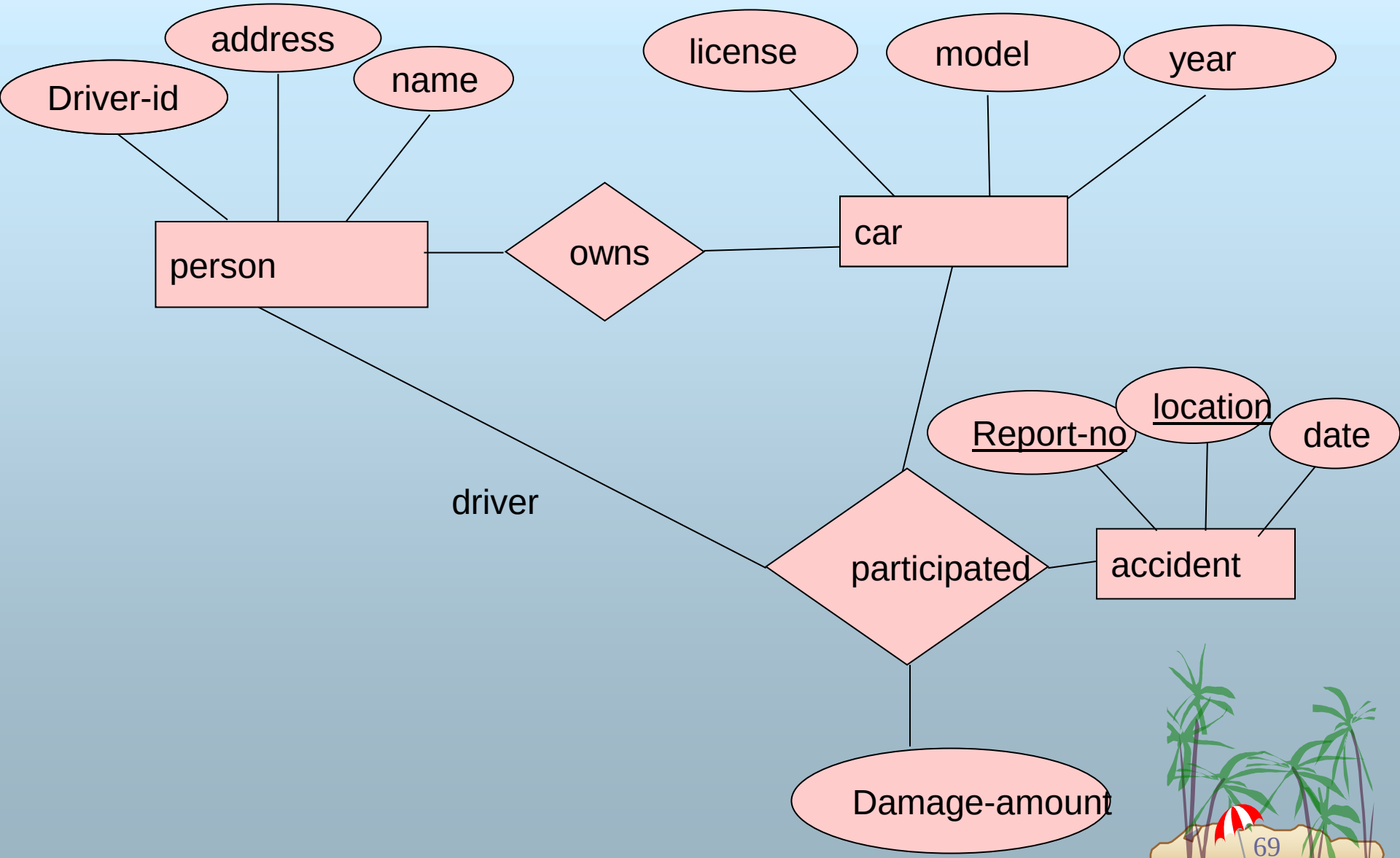
Entity with a multivalued attribute (Skill) and derived attribute (Years_Employed)



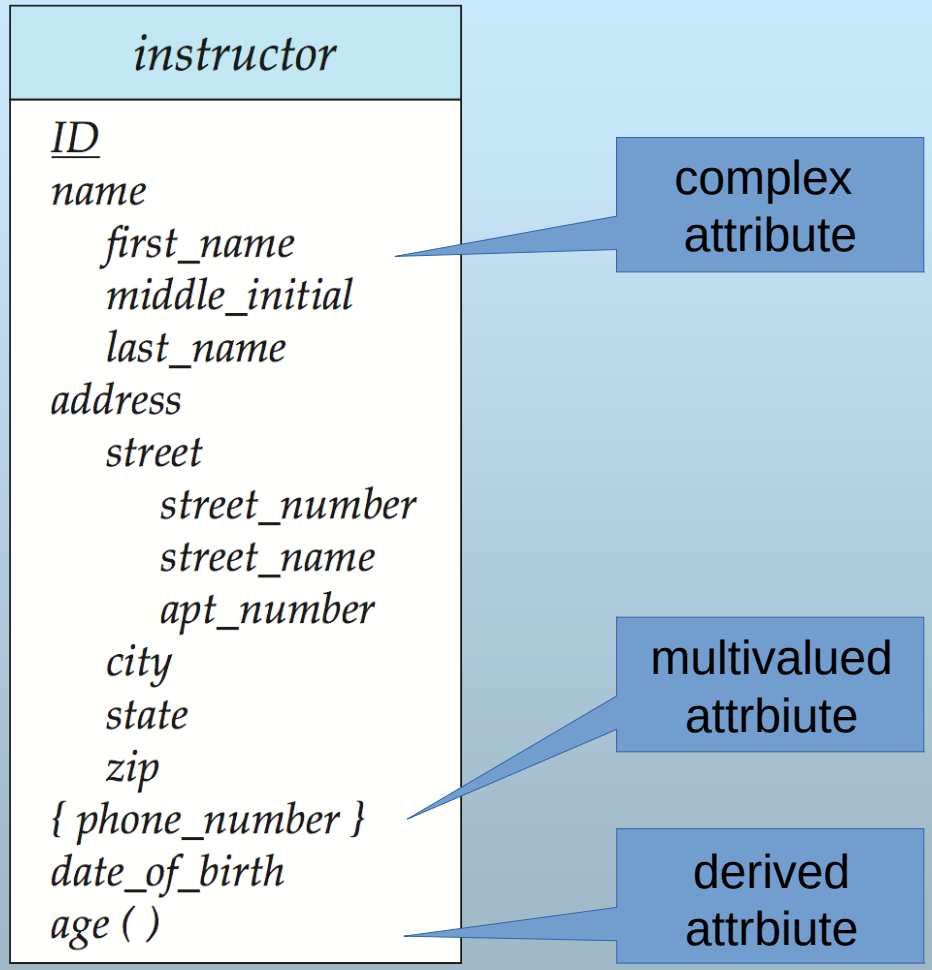
E-R Diagram With Composite, Multivalued, and Derived Attributes



E.R.Model for car Insurance company

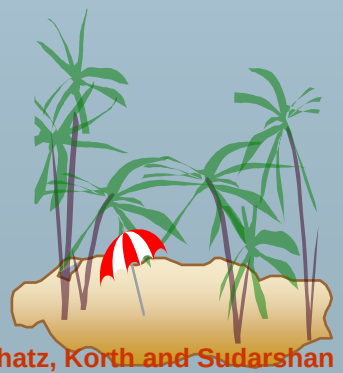


Notation of Attribute Types

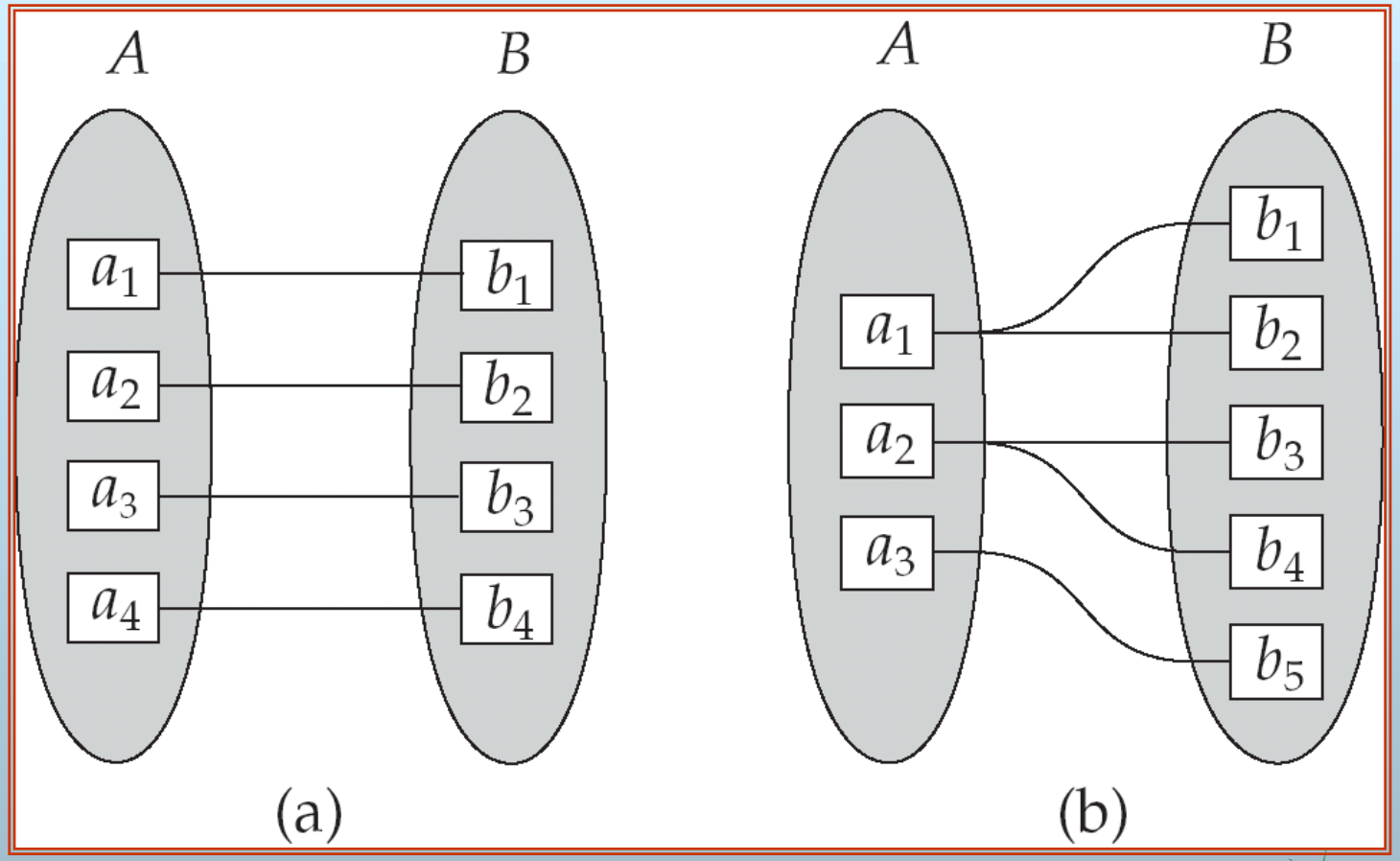


Mapping Cardinality Constraints

- Express the number of entities to which another entity can be associated via a relationship set.
- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to one
  - One to many
  - Many to one
  - Many to many



Mapping Cardinalities

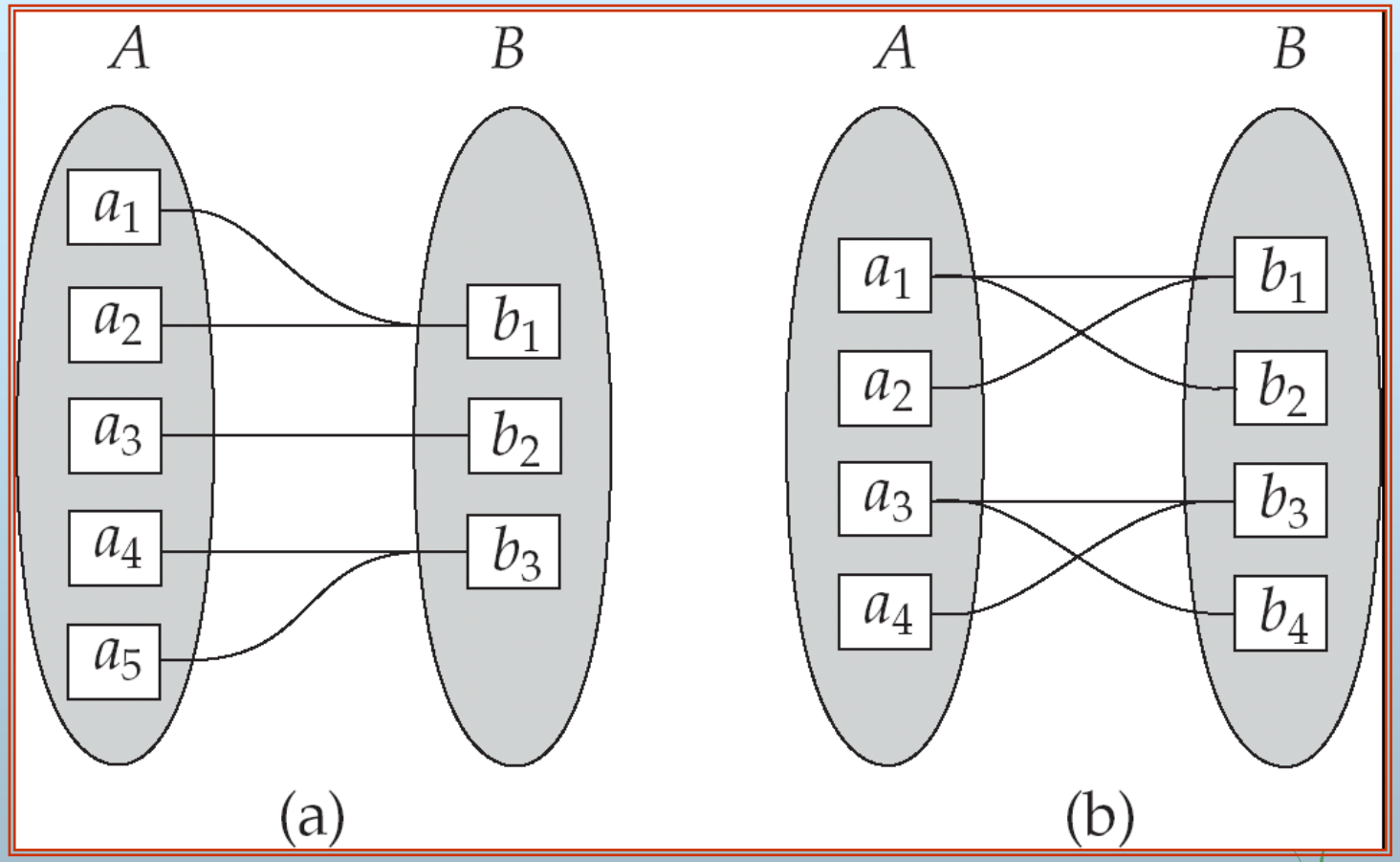


One to one

One to many

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

Mapping Cardinalities



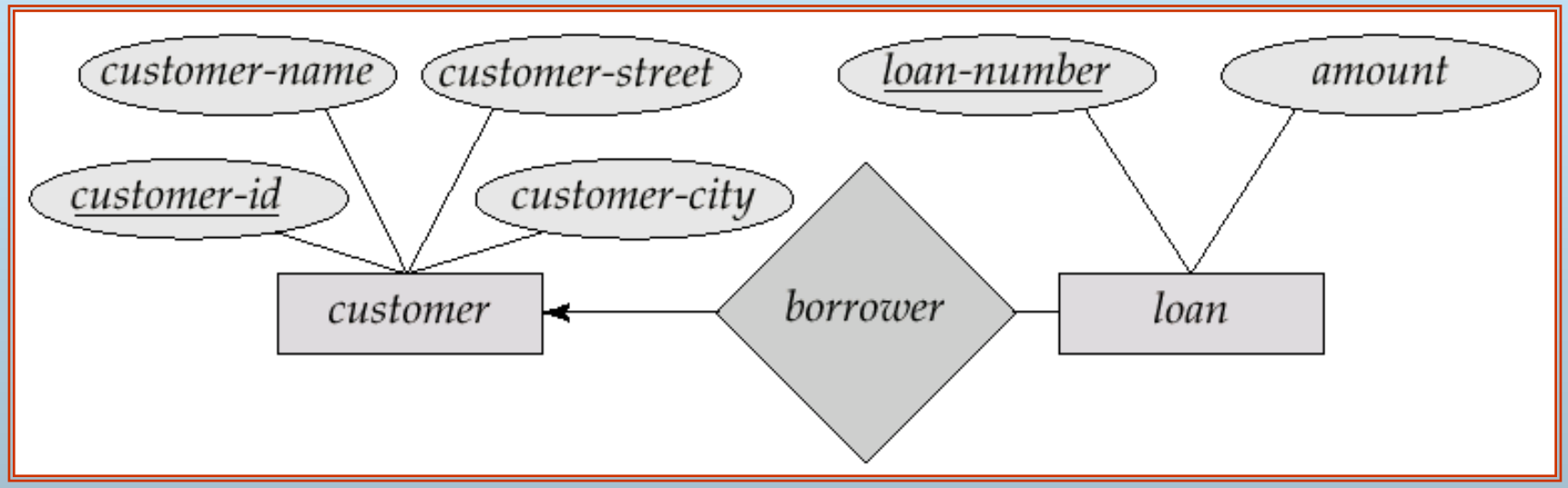
Many to one

Many to many

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

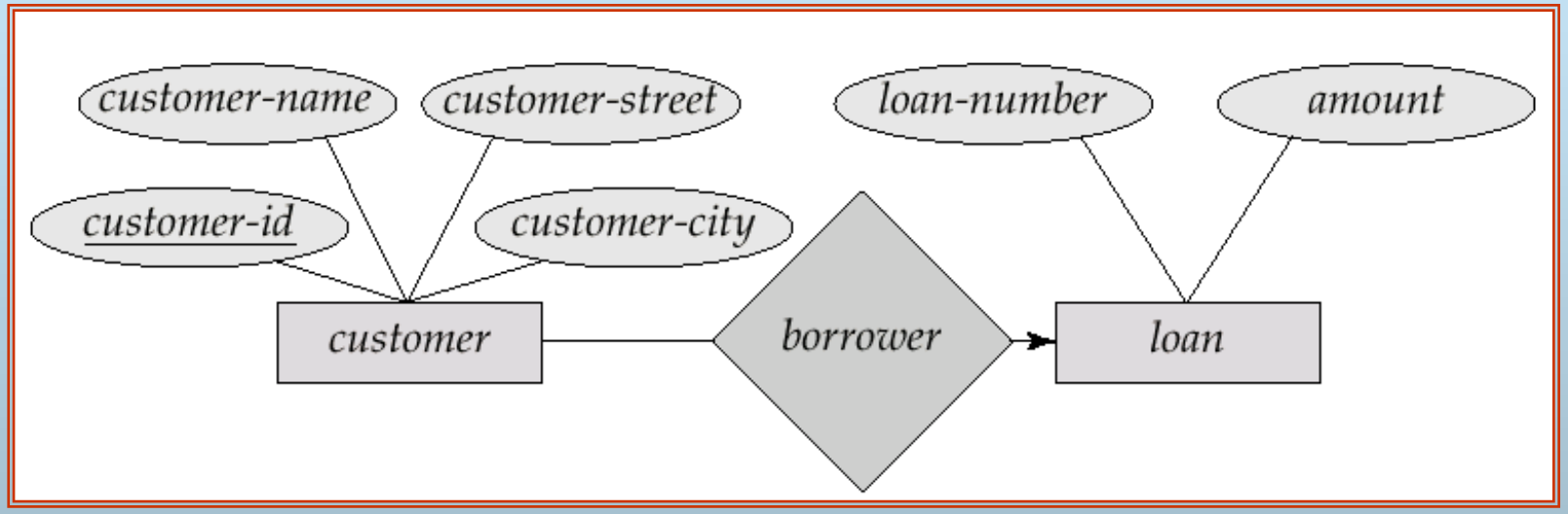
One-To-Many Relationship

- In the one-to-many relationship a loan is associated with at most one customer via *borrower*, a customer is associated with several (including 0) loans via *borrower*



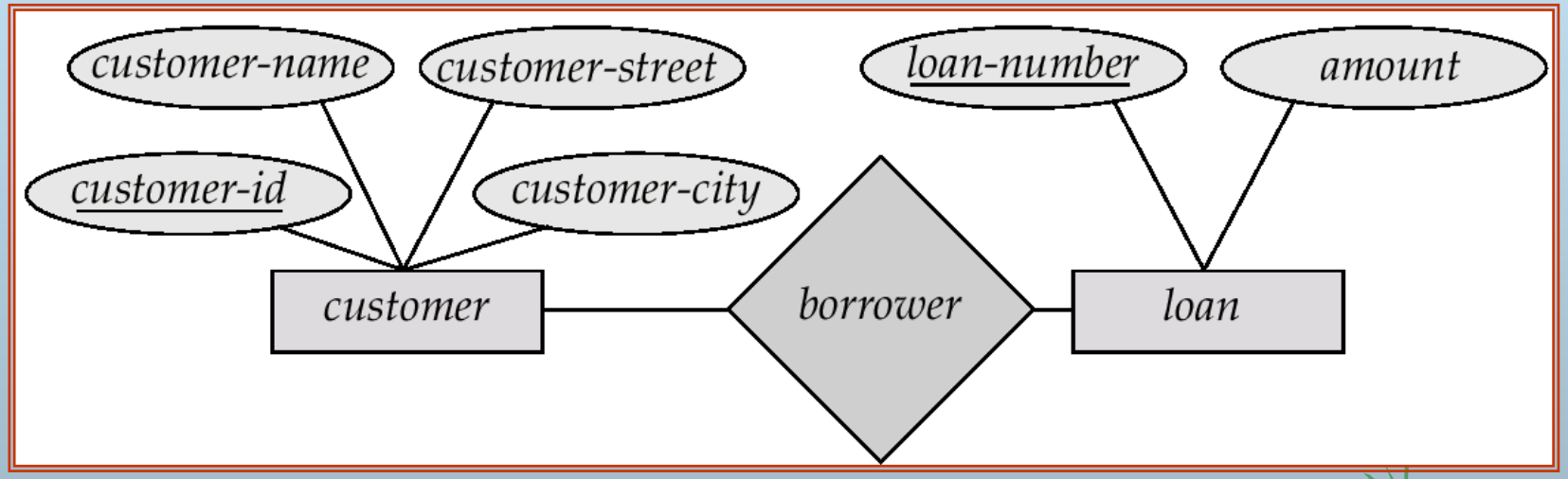
Many-To-One Relationships

- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*



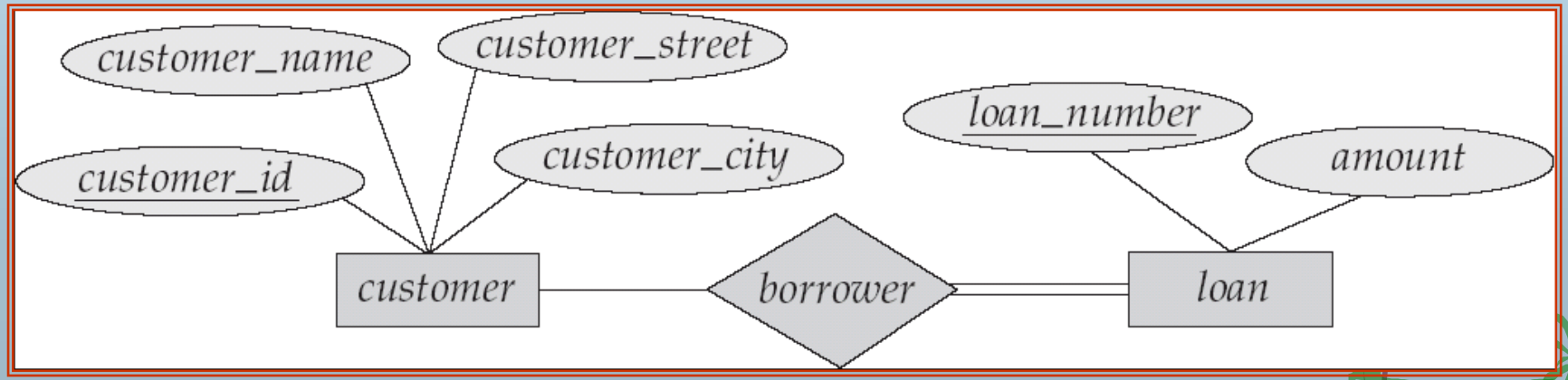
Many-To-Many Relationship

- A customer is associated with several (possibly 0) loans via borrower
- A loan is associated with several (possibly 0) customers via borrower



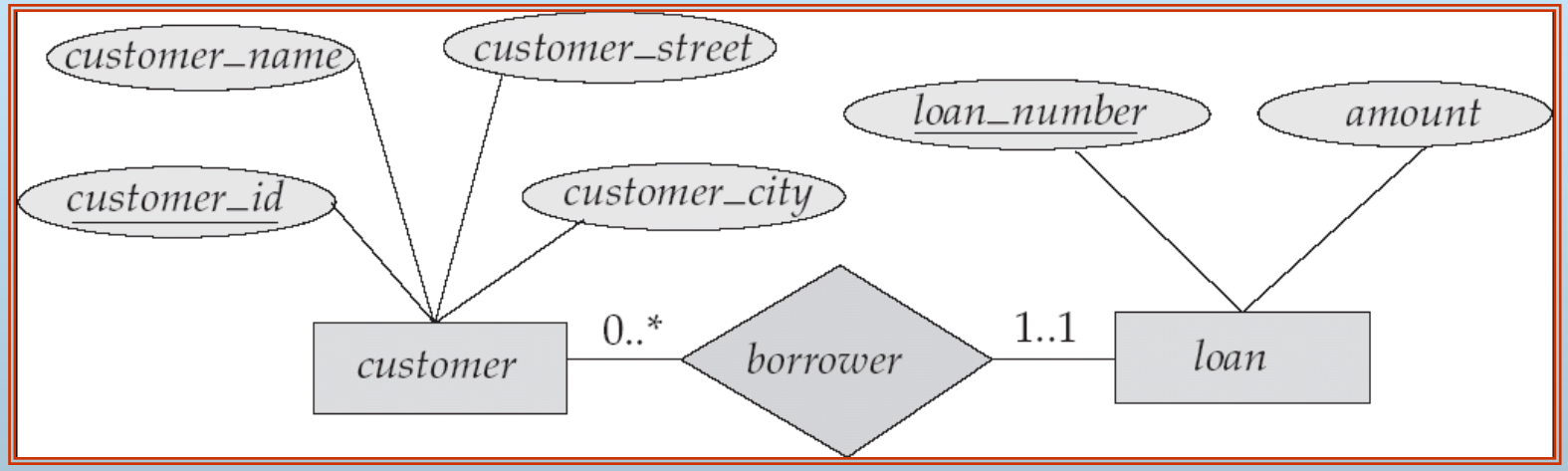
Participation of an Entity Set in a Relationship Set

- Total participation (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
  - E.g. participation of loan in borrower is total
    - every loan must have a customer associated to it via borrower
- Partial participation: some entities may not participate in any relationship in the relationship set
  - Example: participation of customer in borrower is partial



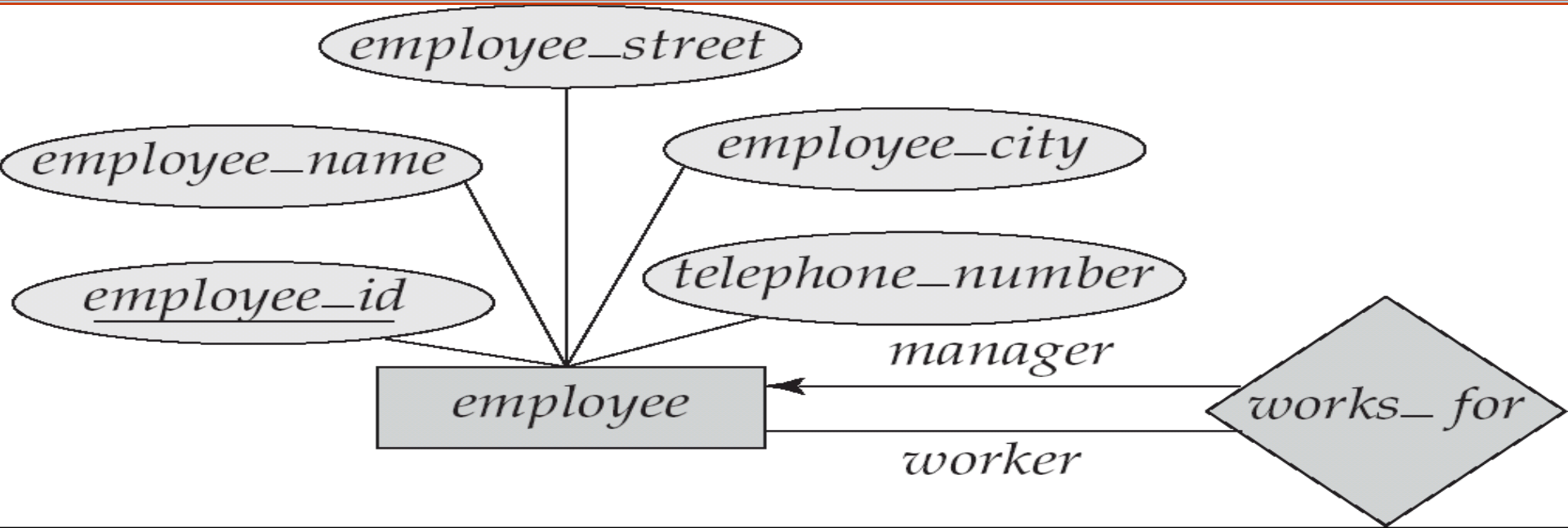
Alternative Notation for Cardinality Limits

- Cardinality limits can also express participation constraints



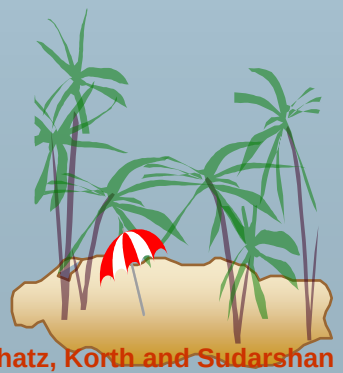
Roles

- Entity sets of a relationship need not be distinct
- The labels “manager” and “worker” are called **roles**; they specify how employee entities interact via the works_for relationship set.
- Roles are indicated in E-R diagrams by labeling the lines that connect diamonds to rectangles.
- Role labels are optional, and are used to clarify semantics of the relationship

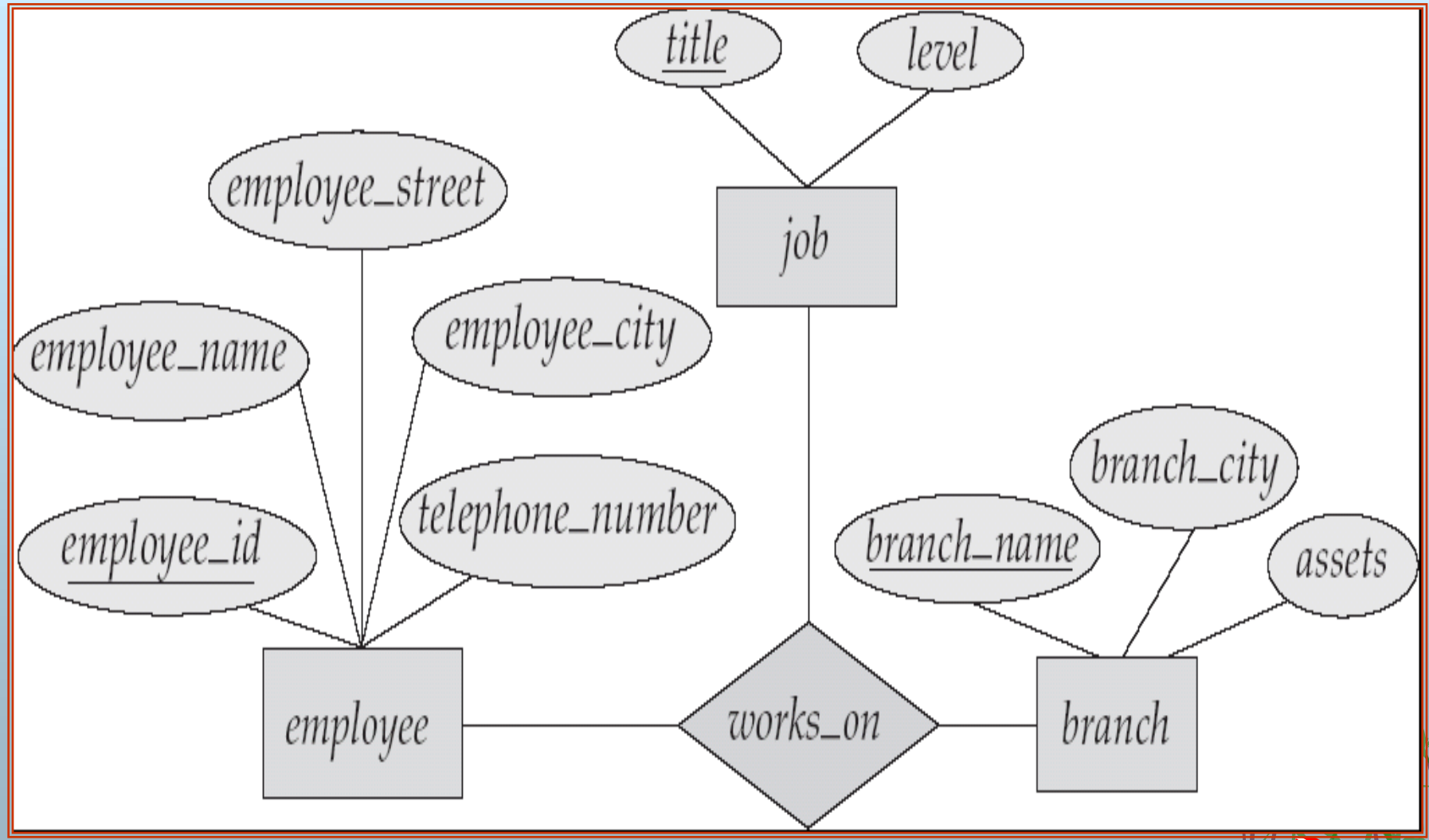


Cardinality Constraints

- We express cardinality constraints by drawing either a directed line (→), signifying "one," or an undirected line (—), signifying "many," between the relationship set and the entity set.
- One-to-one relationship:
  - A customer is associated with at most one loan via the relationship *borrower*
  - A loan is associated with at most one customer via *borrower*



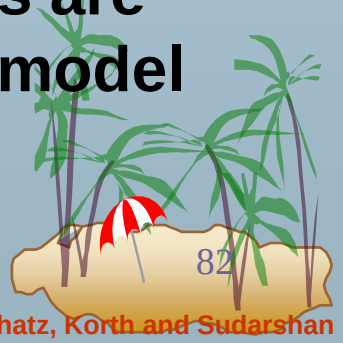
E-R Diagram with a Ternary Relationship



Redundant Attributes

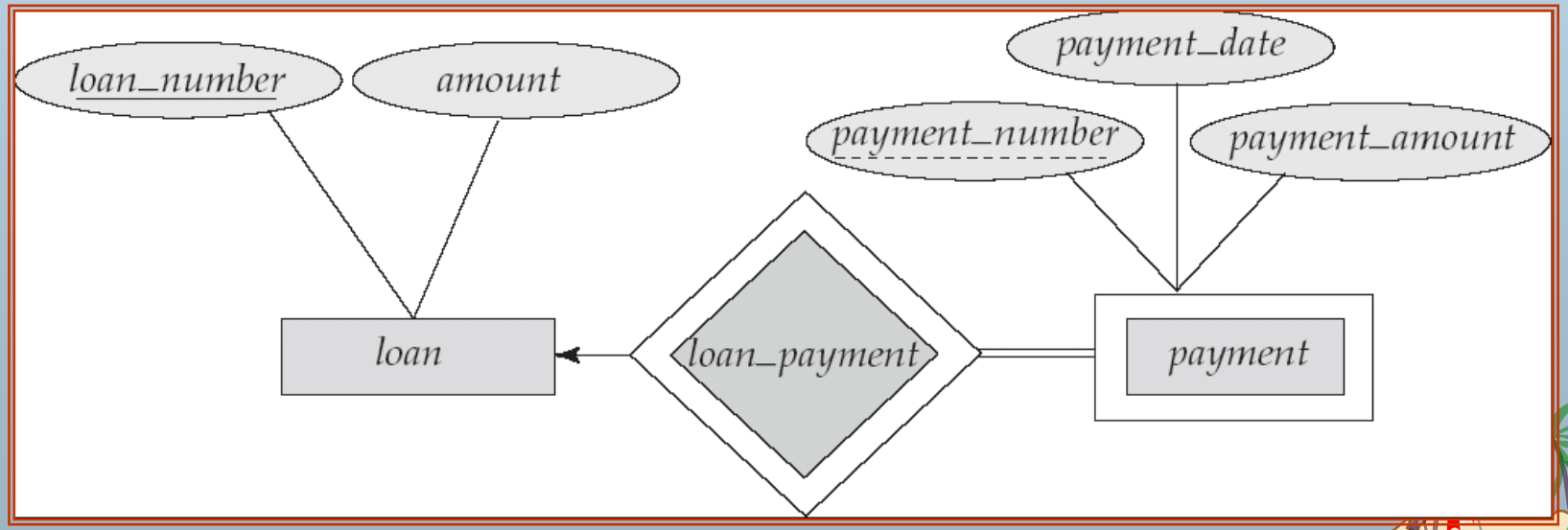
- **Suppose we have entity sets:**
  - **instructor, with attributes: ID, name, dept_name, salary**
  - **department, with attributes: dept_name, building, budget**
- Example, instructors and departments are connected by a relation set
  - e.g., instructor_belong_to_department (ID,dept_name)
- Now, dept_name is no longer needed in the instructor entity set
  - It is redundant there Hence,
  - we will remove it

Note: **sometimes, removed redundant attributes are reintroduced when converting the conceptual model into a logical model**



Weak Entity Sets (Cont.)

- We depict a weak entity set by double rectangles.
- We underline the discriminator of a weak entity set with a dashed line.
- payment_number – discriminator of the *payment* entity set
- Primary key for *payment* – (*loan_number*, *payment_number*)



Weak Entity Sets

- An **entity set** that does not have a primary key is referred to as a ***weak entity set***.
- The existence of a weak entity set depends on the existence of a ***identifying entity set***
- The primary key of a weak entity set is formed by the ***primary key of the strong entity set*** on which the weak entity set is existence dependent, plus the weak entity set's **discriminator**.

Weak Entity

- In real world there are many such cases where Weak Entity exists and we are required to uniquely records as well !!
- If an entity does not have a key by itself, but it is only unique in the context of some other (master) entity.

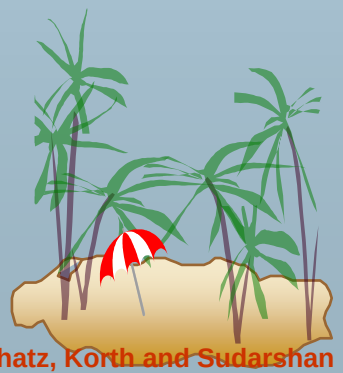
Note: In such cases, keys are always composite.

Examples:

- A classroom is identified by a building and a room number.
- A section in a book is identified by a chapter and a section title.

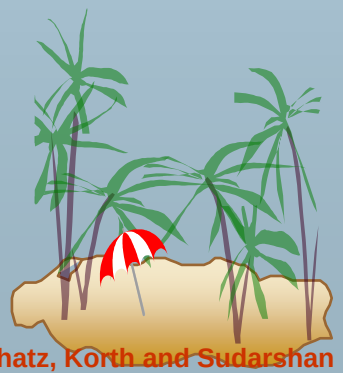
There is also an **existence dependency**.

- If the building is pulled down, the classrooms automatically disappear.



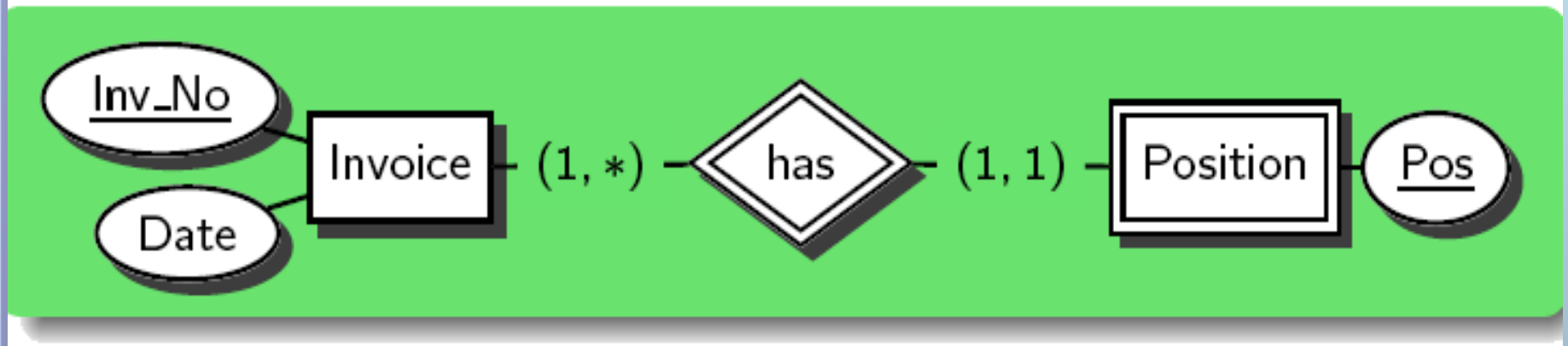
Weak Entity Sets (Cont.)

- Note: the primary key of the strong entity set is not explicitly stored with the weak entity set, since it is implicit in the identifying relationship.
- If *loan_number* were explicitly stored, *payment* could be made a strong entity, but then the relationship between *payment* and *loan* would be duplicated by an implicit relationship defined by the attribute *loan_number* common to *payment* and *loan*



Weak Entities....

- In the ER model, such scenarios are modelled via **weak entities**[3].
- In ER diagrams, weak entities and their identifying relationships are indicated by **double lines:**

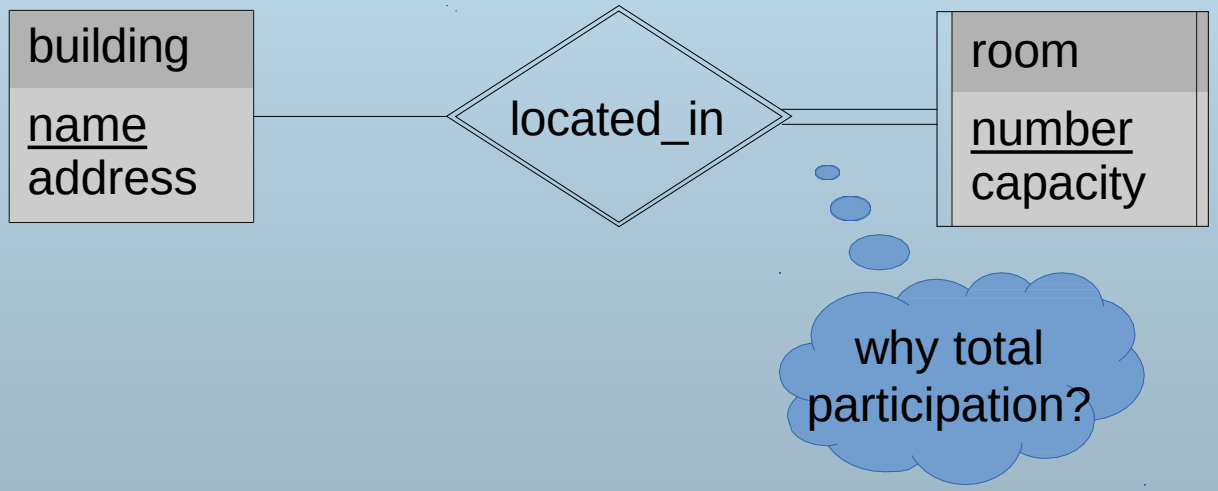


- For the weak entity, the **inherited part of the key is not shown.**

[3] Non-weak entities are also called strong entities.

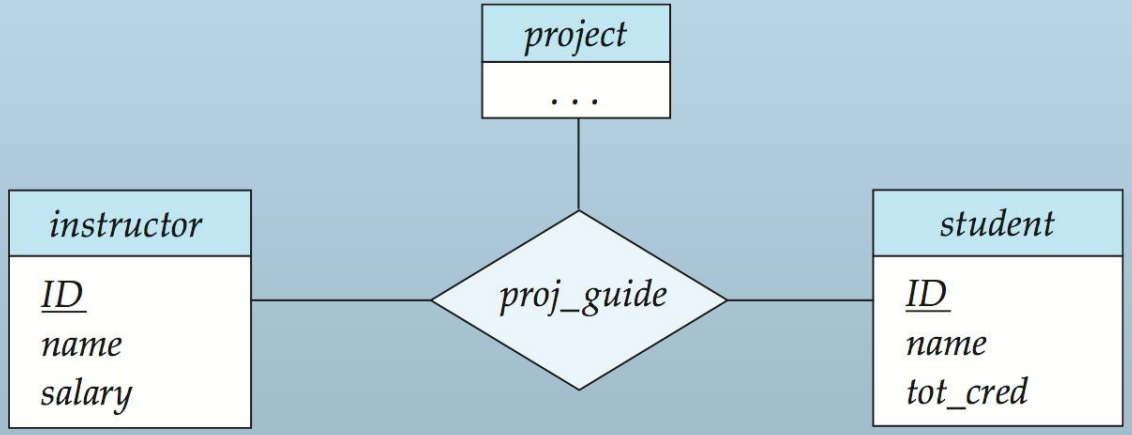
Expressing Weak Entity Sets

- A weak entity set is depicted via a double rectangle
  - The identifying relationship set is depicted by a double diamond
- The *discriminator* is underlined with a dashed line
  - Primary key for section – (course_id, sec_id, semester, year)



Higher Arity Relationship Sets

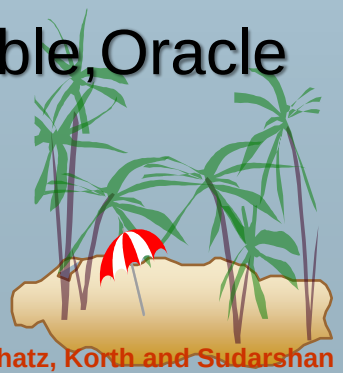
- Most relationship sets are binary
- Sometimes, ternary (or higher arity) relations occur
  - ER models support that
- Example:
  - Students work on projects under supervision of an instructor



VIEWS

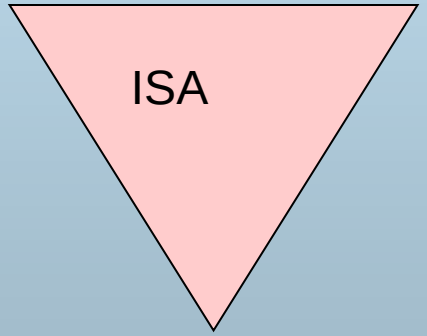
- After a table is created and populated with data,it may become necessary to prevent all users from accessing all columns of a table,for data security reasons.
- Creating several tables having the appropriate number of columns and assigning specific users to each table, as required.
- This will answer data security requirements very well but will give

  rise to a great deal of redundant data being resident in tables,in the database.
- To reduce redundant data to the minimum possible,Oracle allows the creation of

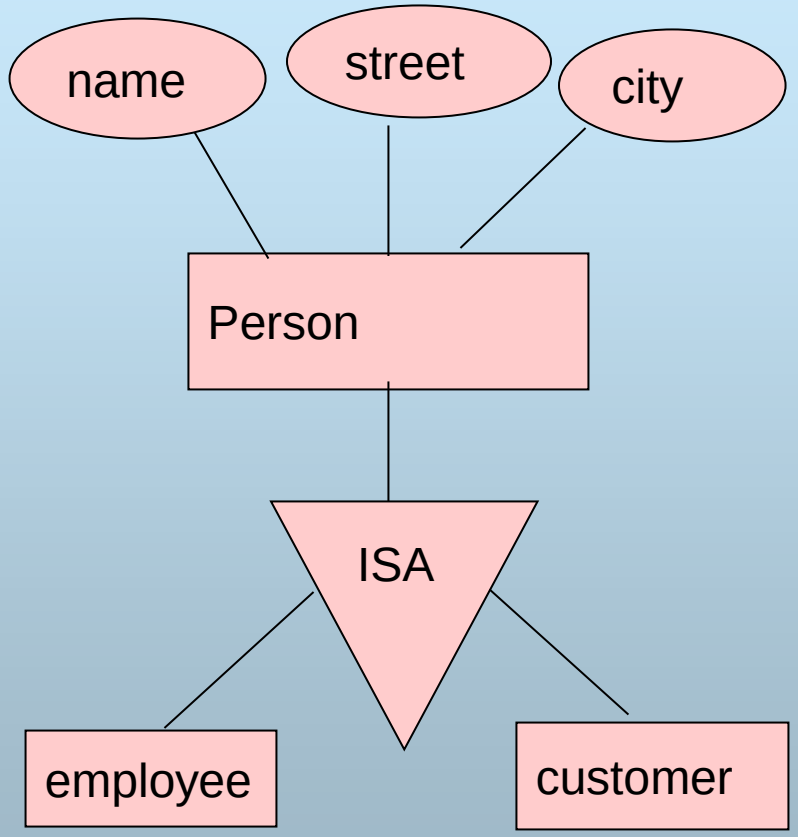


Specialization

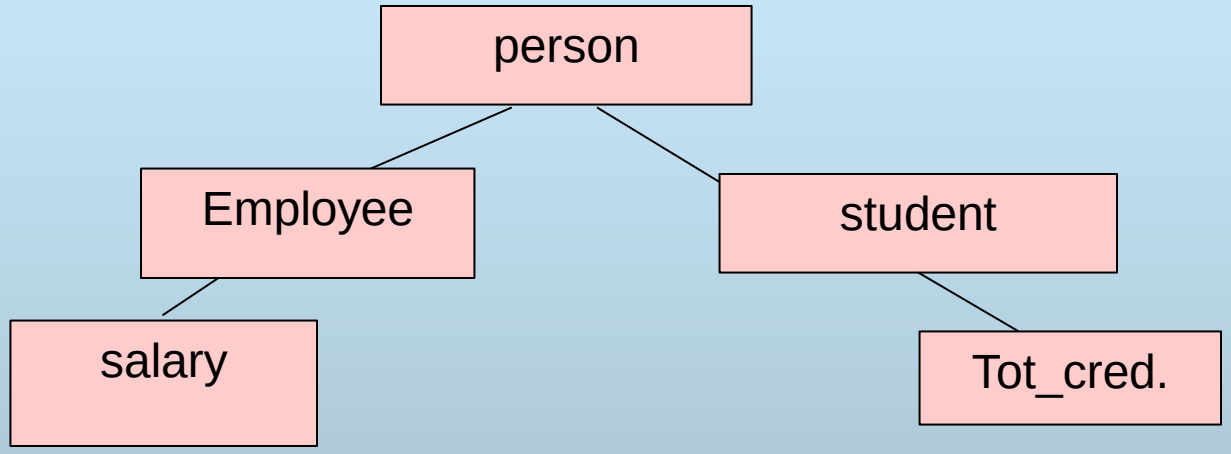
- An entity set may include subgroupings of entities that are
- distinct in some way from other entities in the set.
- For e.g A subset of entities within an entity set may have
- attributes that are not shared by all entites in the entity set.
- It is represented by



Example with specialization



Each of these person types is described by a set of attributes that includes all the attributes of entity set person plus possibly additional attributes.



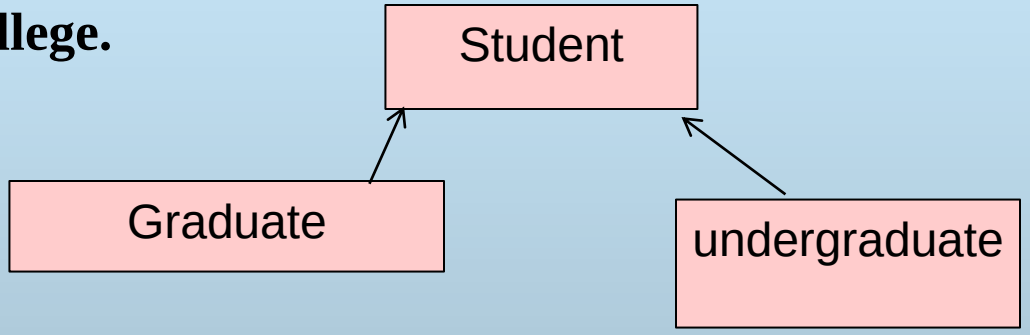
overlapping specialization (as is the case for student and employee specializations of person), two separate arrows are used.

The process of designating **sub-groupings** within an entity set is called specialization.

The specialization of person allows us to distinguish among person entities according to whether they correspond to employees or students: in general, a person could be an employee , a student, both, or neither.

Example-2

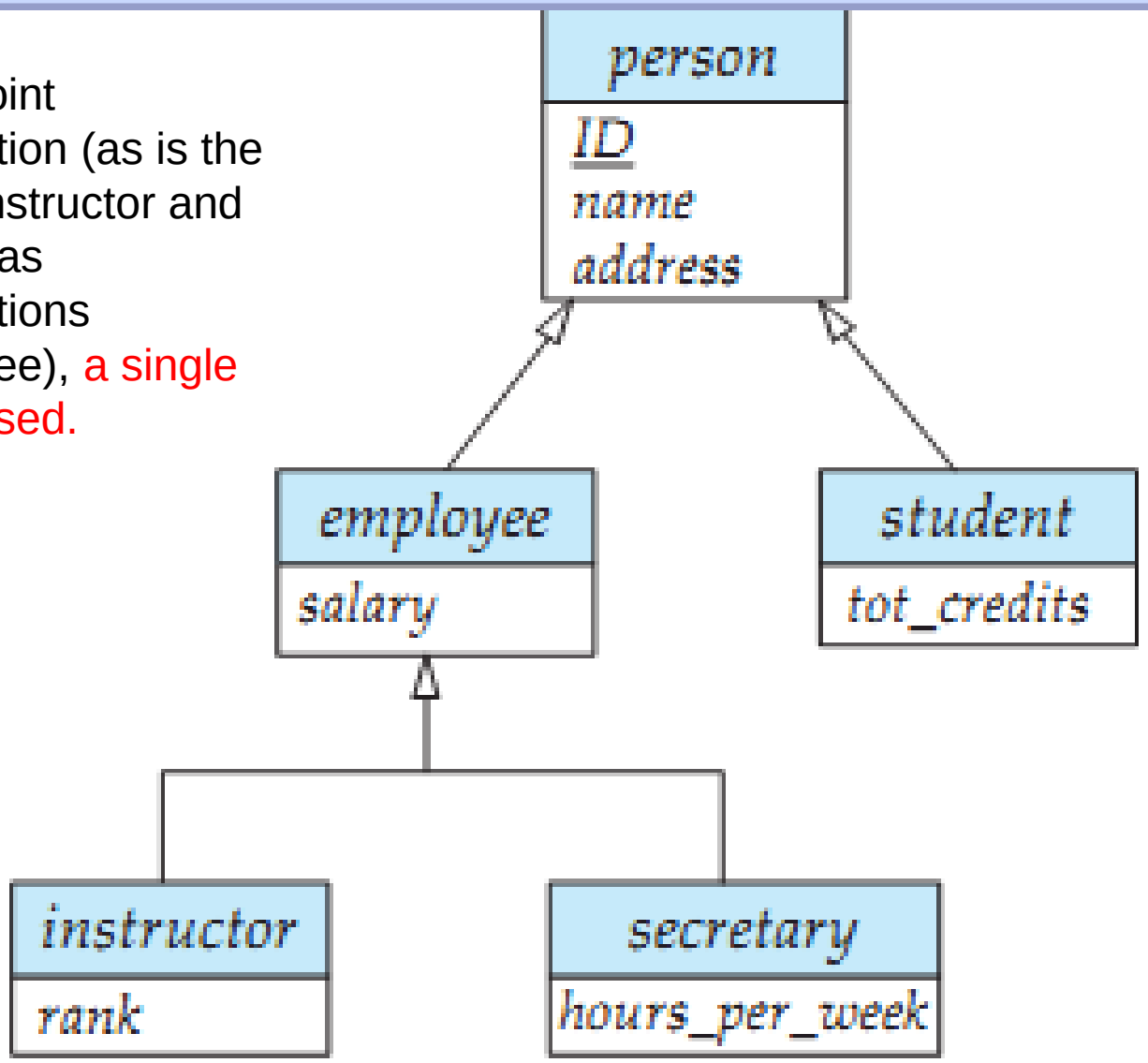
- The university divides students into two categories:
  - graduate and undergraduate.
  - Graduate students have an office assigned to undergraduate students are assigned to a residential college.



  - **ID, name, address ,and tot cred. The entity set graduate would have all the attributes of** student Additional attribute office **number** .
  - The entity set **undergraduate** would have all the attributes of student, and an additional attribute **residential college** .

We can apply specialization repeatedly to refine a design. For instance , university employees may be further classified as one of the following:

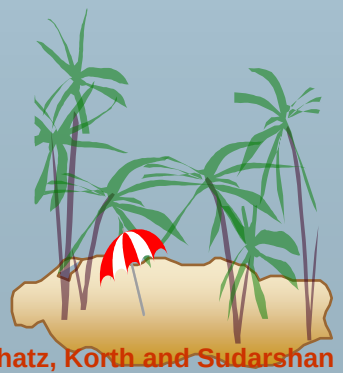
For a disjoint specialization (as is the case for instructor and secretary as specializations of employee), a single arrow is used.



Generalization

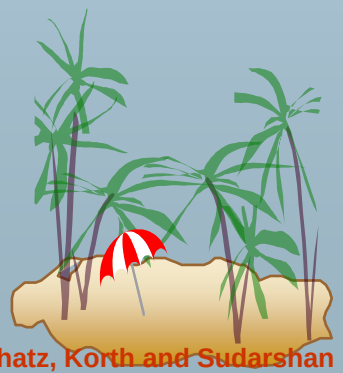
- The refine ment from an initial entity set into successive levels of entity subgroup-ings represents a top-down design process in which distinctions are made explicit.
- The design process may also proceed in a bottom-up manner, in which multiple entity sets are synthesized into a higher-level entity set on the basis of common features.
- The database designer may have first identified:
  - instructor entity set with attributes
  - salary,and rank .
  - secretary entity set with attributes
    - secretary id, secretary name, secre tary salary,

    -- and hours per week .



Assignment-01

- A university registrar's office maintains data about the following entities:
  - Courses including number ,title, credits, syllabus and pre requisties ;
  - Courses ,including number ,year ,semester ,section number ,instructor(s) timings and classroom;
  - students, including student- Id, name, and program:
  - Instructors, including identification number, name, department, and title. Further, the enrollment of students in courses and grades awarded to students in each course they are enrolled for must be appropriately modeled.
  - Construct ER diagram for registrar's office .
  -

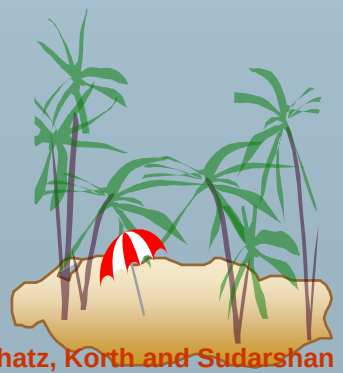


Hospital Information System

- Patients are treated in a single ward by the doctors assigned to them. Usually each patient will be assigned a single doctor, but in rare cases they will have two.
- Heathcare assistants also attend to the patients, a number of these are associated with each ward.
  Initially the system will be concerned solely with drug treatment. 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 staff are paid part time and doctors and care assistants work varying amounts of overtime at varying rates (subject to grade).
- The system will also need to track what treatments are required for which patients and when and it should be capable of calculating the cost of treatment per week for each patient (though it is currently unclear to what use this information will be put).

How do we start an ERD?

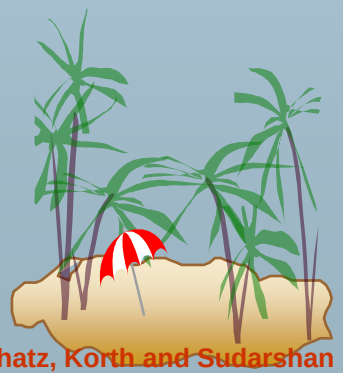
- 1. Define Entities: these are usually nouns used in descriptions of the system, in the discussion of business rules, or in documentation; identified in the narrative .
- 2. Define Relationships: these are usually verbs used in descriptions of the system or in discussion of the business rules .
- 3. Add attributes to the relations; these are determined by the queries, and may also suggest new entities, e.g. grade; or they may suggest the need for keys or identifiers.



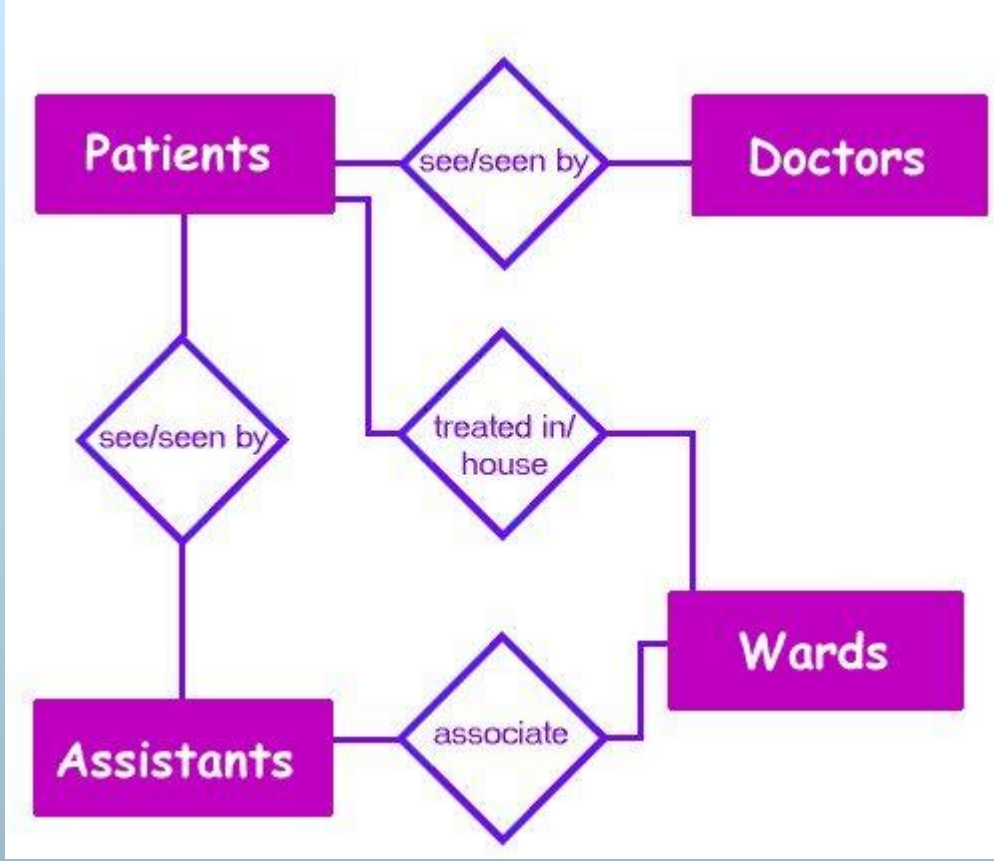
ER Diagram…..

- **What questions can we ask?**
  - a. Which doctors work in which wards?
  - b. How much will be spent in a ward in a given week?
  - c. How much will a patient cost to treat?
  - d. How much does a doctor cost per week?
  - e. Which assistants can a patient expect to see?
  - f. Which drugs are being used?
- **This allows us to consider a variety of questions such as:**

a. Which beds are free? b. Which assistants work for Dr. X? c. What is the least expensive prescription? d. How many doctors are there in the hospital? e. Which patients are family related?



E-R Diagram



Assignment -02

E-R Model Case Studies

1 : Suppose you are given the following requirements for a simple database for the National Hockey League (NHL):

- the NHL has many teams,
- each team has a name, a city, a coach, a captain, and a set of players,
- each player belongs to only one team,
- each player has a name, a position (such as left wing or goalie), a skill level, and a set of injury records,
- a team captain is also a player,
- a game is played between two teams (referred to as host_team and guest_team) and has a date (such as May 11th, 1999) and a score (such as 4 to 2).

Construct a clean and concise ER diagram for the NHL database.