RAMAIAH
Institute of Technology

Database Systems

Subject Code: CS52 (Credits: 3:1:0)

RAMAIAH Institute of Technology

Unit-I

Introduction:

- Characteristics of Database approach,
- Actors on the Scene,
- Advantages of using DBMS approach,

Database system Concepts and Architecture:

- Data models, schemas and instances,
- Three schema architecture and data independence,
- Database languages and interfaces,
- The database system environment,
- Centralized and client-server architectures,
- Classification of Database Management systems,

Entity-relationship(ER) Model:

- Conceptual Database using high level Conceptual data models for Database Design,
- A Sample Database Application,
- Entity types, Entity sets Attributes and Keys Relationship types,
- Relationship Sets, Roles and Structural Constraints
- Weak Entity Types.

CHAPTR-1

Elmasri and Navathe: Fundamentals of Database Systems, 5th Edition, Addison-Wesley, 2011.

RAMAIAH
Institute of Technology

Introduction:

DATA:

- It could be any **Fact/Information**, that can be recorded
- Example: Text, number, multimedia (audio /video /image), Geographic Information System (GIS),
- It may be Stored data or Real time data(Stock Market/Super Market)
- It can be any size and complexity(KB , GB, TB or few Peta Byte etc)

DATABASE:

- It is a collection of **related data (Logically)**
- A database is an **organized collection of data**, generally stored and accessed from a computer system.

DATABASE MANAGEMENT SYSTEM(DBMS):

- The database management system **(DBMS) is the software** that interacts with end users, applications, and the database itself to capture and analyze the data.

RAMAIAH Institute of Technology

DB+DBMS = Database Systems

A database is a collection of related data.

- By data, we mean known **facts that can be recorded** and **that have implicit meaning.**
- A database represents **some aspect of the real world**
- A database is a **logically collection of data** with **some inherent meaning**.
- A database is designed, built, and populated with **data for a specific purpose.**

A **D**ata **B**ase **M**anagement **S**ystem is a system software for easy, efficient and reliable data processing and management.

The DBMS is a **general-purpose software system** that facilitates the processes of

- **Defining**, (Data type/ data structures/ Constraints)
- **Constructing**,(How data is placed on the Disk/processing of stored data)
- **Manipulating**(Querying the DB to retrieve specific data), and
- **Sharing** databases among various users and applications.(Simultaneously)

RAMAIAH
Institute of Technology

Applications of DBMS

- **Banking System** (Account Information/ Loan information/ Payment information/ Transition Details/ Customer details etc.)
- **Online Shopping and Retail** (Amazon, Flipkart, Snapdeal, eBay etc.)
- **Sales and Marketing** (Product packing/Order Tracking/Product distributions)
- **Airline/ railway Reservation System** (Passenger details, Booking details)
- **Telecommunication** (Call details/ monthly bill details)
- **Human Resources**(Employee details/ Salary details/ tax details)
- **Tourism: Hotels & Restaurants** (Customer details/ Booking details/cancellation details)
- **Hospital management** (Doctor and Patient details/ patient record or reviews)

RAMAIAH Institute of Technology

Database System Environment

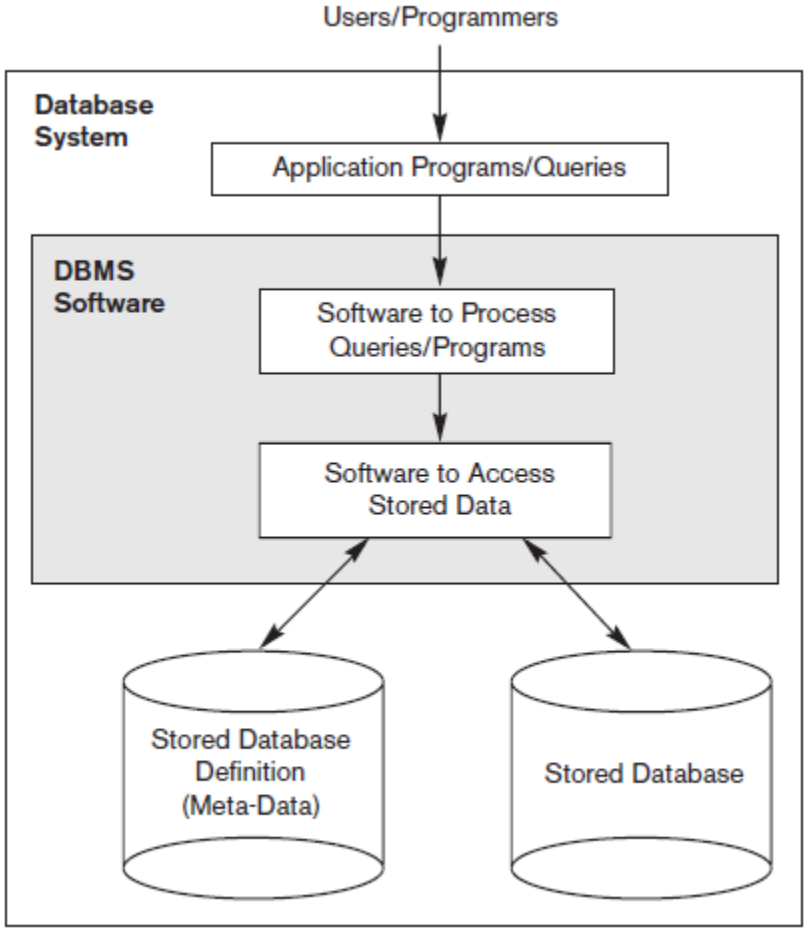


Figure 1.1
A simplified database system environment.

RAMAIAH
Institute of Technology

University Database

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

RAMAIAH Institute of Technology

Characteristics of the Database Approach:

The main characteristics of the database approach versus the file-processing approach are the following:

1. **Self-describing nature of a database system**
2. **Insulation between programs and data, and data abstraction**
3. **Support of multiple views of the data**
4. **Sharing of data and multiuser transaction processing**

RAMAIAH
Institute of Technology

Self-describing nature of a database system:

- DBMS contains complete **definition or description of database**
  - Structure and constrains
- The **DBMS catalog,** which contains information such as
  - the structure of each file,
  - the type and storage format of each data item, and
  - The various constraints on the data.
- The information stored in the catalog is called **meta-data**, and it describes the structure of the primary database (Figure).

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
....	....	.....
....	....	.....
....	....	.....
Prerequisite_number	XXXXNNNN	PREREQUISITE

Note: Major_type is defined as an enumerated type with all known majors.
XXXXNNNN is used to define a type with four alpha characters followed by four digits.

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

RAMAIAH
Institute of Technology

Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
Student_number	31	4
Class	35	1
Major	36	4

Figure 1.4
Internal storage format for a STUDENT record, based on the database catalog in Figure 1.3.

Insulation between programs and data, and data abstraction

- In **traditional file processing**: **The structure of data files is embedded in application programs**, so changes to the structure of a file may require changing all the programs that access that file.
    - The structure of data files is stored in the DBMS catalog separately from the access programs. We call this property **program-data independence**.
    - The characteristic that allows program-data independence and program-operation independence is called **data abstraction**.
- A DBMS provides users with a **conceptual representation of data** that does not include many of the details of
    - **how the data is stored or how the operations are implemented.**
    - The data model uses logical concepts, **the data model hides storage and implementation details** that are not of interest to most database users.

RAMAIAH
Institute of Technology

Support of multiple views of the data:

A database typically has many **users (DB Administrator, DB designer and End User),**

- Each of whom may require a **different perspective or view of the database.**
  - one user of the database of Figure 1.2 may be interested only in accessing and printing the transcript of each student; the view for this user is shown in Figure 1.5(a).
  - A second user, who is interested only in all the prerequisites of each course for which they register, may require the view shown in Figure 1.5(b).

(a)

TRANSCRIPT

Student_name	Student_transcript				
	Course_number	Grade	Semester	Year	Section_id
Smith	CS1310	C	Fall	08	119
	MATH2410	B	Fall	08	112
Brown	MATH2410	A	Fall	07	85
	CS1310	A	Fall	07	92
	CS3320	B	Spring	08	102
	CS3380	A	Fall	08	135

(b)

COURSE_PREREQUISITES

Course_name	Course_number	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data Structures	CS3320	CS1310

Figure 1.5
Two views derived from the database in Figure 1.2. (a) The TRANSCRIPT view. (b) The COURSE_PREREQUISITES view.

RAMAIAH
Institute of Technology

Sharing of data and multiuser transaction processing:

A multiuser DBMS, as its name implies, must allow **multiple users to access the database at the same time.**

- This is essential if data for multiple applications is to be integrated and maintained in a single database.
- The DBMS must include **concurrency control software** to ensure **that several users trying to update the same data** do so in a controlled manner so that the result of the updates is correct.
- For example, when **several reservation agents try to assign a seat on an airline flight,**
  - the DBMS should ensure that each seat can be accessed by only one agent at a time for assignment to a passenger.
  - These types of applications are generally called **online transaction processing(OLTP) applications**.

RAMAIAH
Institute of Technology

Actors on the Scene:

Database Administrators:

- In a database environment, the primary resource is the database itself, and the secondary resource is the **DBMS and related software**.
- Administering these resources is the responsibility of the database administrator (DBA).
- The DBA is responsible for
    - authorizing access to the database,
    - coordinating and monitoring its use, and
    - acquiring software and hardware resources as needed.
- The DBA is accountable for problems such as **security** breaches and **poor system response time.**

RAMAIAH
Institute of Technology

Database Designers:

- Database designers are responsible for **identifying the data to be stored in the database and for choosing appropriate structures** to represent and store this data.

- It is the responsibility of database designers to communicate with all prospective database users in order to **understand their requirements and to create a design that meets these requirements**.

- Database designers typically **interact with each potential group of users and develop views of the database** that meet the data and processing requirements of these groups.

    - Each view is then analyzed and integrated with the views of other user groups.

RAMAIAH
Institute of Technology

End Users: End users are the people whose jobs require access to the database **for querying, updating, and generating reports;** the database primarily exists for their use. There are several categories of end users:

- **Casual end users** occasionally access the database, but **they may need different information each time.**
- **Naive or parametric end users** make up a sizable portion of database end users. Their main job function revolves around constantly querying and updating the database, using standard types of queries and updates—called **canned transactions**—that have been carefully programmed and tested
- **Sophisticated end users** include **engineers, scientists, business analysts**, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirements.
- **Standalone users** maintain personal databases by using ready-made program packages that provide **easy-to-use menu-based or graphics-based interfaces**.

RAMAIAH Institute of Technology

Advantages of Using the DBMS Approach

Controlling Redundancy: In DB approach, **views of different user group** are integrated during DB design and DB **ensures consistency and saves storage** space.

- For example, consider the UNIVERSITY database example; Here, two groups of users might be
  - **course registration File and**
  - **accounting office File**

- This redundancy in storing the same data multiple times leads to several problems.
  - First, **duplication of effort**.
  - Second, **storage space is wasted** when the same data is stored repeatedly, and this problem may be serious for large databases.
  - Third, files that represent the same **data may become inconsistent**. Because the updates are applied independently by each group.

RAMAIAH Institute of Technology

Advantages of Using the DBMS Approach

Restricting Unauthorized Access:

When multiple users share a large database, most users will no be authorized to access all the information in the database.

- For example, BANKING system : **financial data is often considered confidential**, and only authorized persons are allowed to access such data.
- In addition, **some users may only be permitted to retrieve data**, whereas **others are allowed to retrieve and update.**
- A DBMS should provide a **security and authorization subsystem**, which the DBA uses to create accounts and to specify account restrictions.

RAMAIAH
Institute of Technology

Providing Persistent Storage for Program Objects:

- Databases can be used to provide **persistent storage for program objects and data structures.**
- This is one of the main reasons for **object-oriented database systems**. Programming languages **typically have complex data structures**, such as record types in Pascal or class definitions in C++ or Java.
- The persistent storage of program objects and data structures is an important function of database systems.
- Traditional database systems often suffered from the so called **impedance mismatch problem**, since the data structures provided by the DBMS were incompatible with the programming language's data structures.

RAMAIAH
Institute of Technology

Providing Storage Structures and Search Techniques for Efficient Query Processing:

- Database systems must provide **capabilities for efficiently executing queries and updates.** Because the database is typically stored on disk,
- the DBMS must provide specialized data structures and search techniques to speed up disk search for the desired records. **indexes are used for this purpose**
- The **query processing and optimization module** of the DBMS is responsible for choosing an efficient query execution plan for each query based on the existing storage structures.

RAMAIAH
Institute of Technology

Providing Backup and Recovery:

- The **backup and recovery subsystem** of the DBMS is responsible for recovery.
  - For example, **if the computer system fails** in the middle of a complex update transaction, the recovery subsystem is responsible for **making sure that the database is restored to the state it was in before the transaction started executing.**

Providing Multiple User Interfaces: a DBMS should provide a **variety of user interfaces**.

- These include query languages for → **casual users,**
- programming language interfaces for → **application programmers**,
- forms and command codes for → **parametric users**, and
- menu-driven interfaces and natural language interfaces for → **standalone users.**
- Both **forms-style interfaces and menu-driven interfaces** are commonly known as **graphical user interfaces (GUIs).**

RAMAIAH
Institute of Technology

Representing Complex Relationships among Data: A DBMS must have the capability to represent a variety of complex relationships among the data, to define new relationships as they arise, and to **retrieve and update related data easily and efficiently.**

Enforcing Integrity Constraints: A DBMS should provide capabilities for defining and enforcing these constraints. The simplest type of **integrity constraint involves specifying a data type for each data item.**

Permitting Inferencing and Actions Using Rules:

- **Associate triggers with tables**: A **trigger is a form of a rule activated by updates to the table**, which results in performing some additional operations to some other tables, sending messages, and so on

- **Stored procedures**: they become a part of the overall database definition and are **invoked appropriately when certain conditions are met.**

- **Active database systems**: which provide active rules that can **automatically initiate actions when certain events and conditions occur.**

CHAPTR-2

Elmasri and Navathe: Fundamentals of Database Systems, 5th Edition, Addison-Wesley, 2011.

RAMAIAH Institute of Technology

Database System Concepts and Architecture:

Data Models, Schemas, and Instances:

A Data Model:

- a collection of concepts that can be used to **describe the structure of a database** provides the necessary means to achieve this abstraction.
    - By structure of a database we mean the **data types, relationships, and constraints** that apply to the data.

Categories of Data Models:

- High-level or **conceptual data models(ER Model)** provide concepts that are close to the way many users perceive data,
- **Representational** (or implementation) **data models(Relational Model)**, which provide concepts that may be easily understood by end users but that are not too far removed from the way data is organized in computer storage.
- low-level or **physical data models** provide concepts that describe the details of how data is stored on the computer storage media

RAMAIAH
Institute of Technology

Conceptual data models :

use concepts such as entities, attributes, and relationships. the **Entity-Relationship model**—a popular high-level conceptual data model.

- **An entity** represents a real-world object or concept, such as an employee or a project from the mini world that is described in the database.
- **An attribute** represents some property of interest that further describes an entity, such as the employee's name or salary.
- **A relationship** among two or more entities represents an association among the entities,

RAMAIAH Institute of Technology

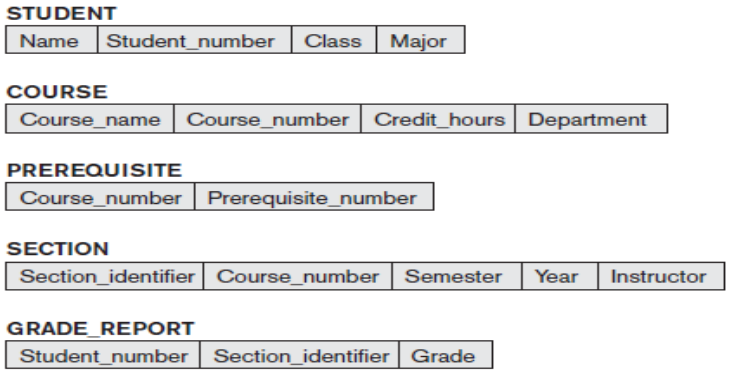
Schemas, Instances, and Database State:

- The description of a database is called the **database schema**, which is specified during database design and is not expected to change frequently. **The actual data in a database may change quite frequently.**
- Schema is of three types: **Physical schema, logical schema and view schema.**

- **an instance is a state** when data is loaded into the database or when any change is acquired by the corresponding database

- The data in the database at a particular moment in time is called a **database state or snapshot.** It is also called the **current set of occurrences or instances in the database**.

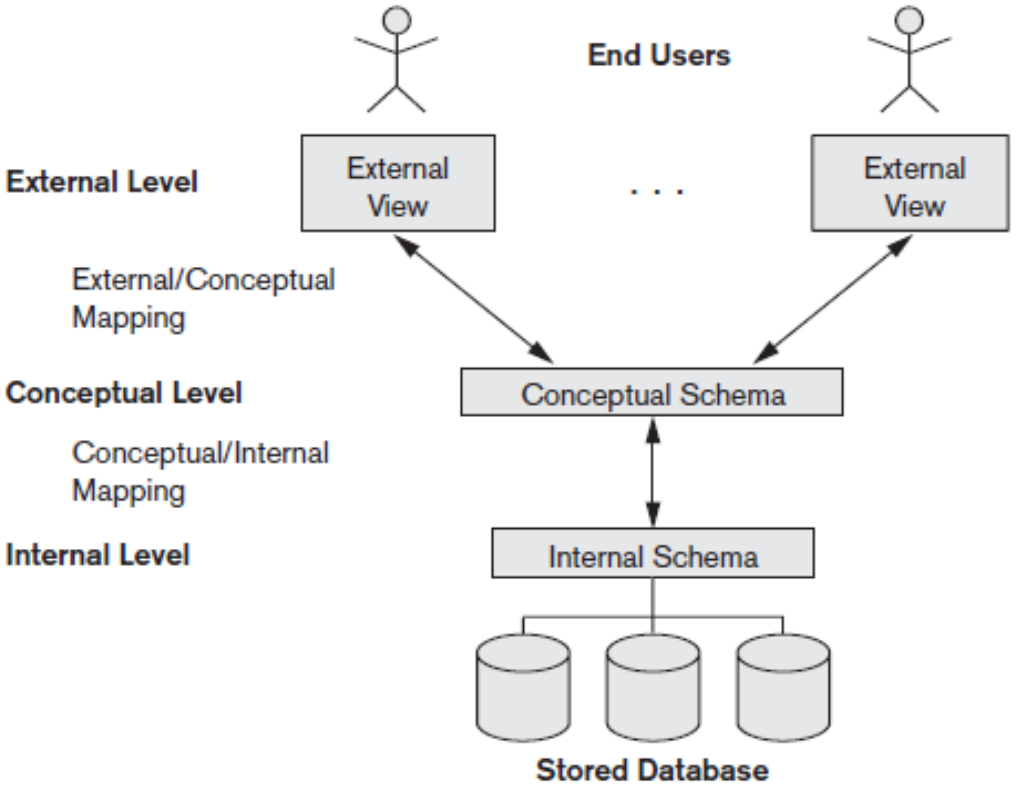
Figure 2.1
Schema diagram for the database in Figure 1.2.



RAMAIAH Institute of Technology

Three-Schema Architecture

Figure 2.2
The three-schema architecture.



Three of the four important characteristics of the database approach, listed in previous , are (1) use of a **catalog to store the database description (schema)** so as to make it self-describing, (2) **insulation of programs and data** (program-data and program-operation independence), and (3) **support of multiple user views**.

RAMAIAH
Institute of Technology

The goal of the three-schema architecture, illustrated in Figure 2.2, is **to separate the user applications from the physical database.**

- The **internal level has** **an internal schema**, which describes the physical storage structure of the database.
- The **conceptual level has** **a conceptual schema**, which describes the structure of the whole database for a community of users.
    - The **conceptual schema hides the details of physical storage structures** and **concentrates on describing entities, data types, relationships, user operations, and constraints.**
- The **external or view level includes** **a number of external schemas** or user views.
    - Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

RAMAIAH Institute of Technology

Data Independence

The **capacity to change the schema at one level of a database system without having to change the schema at the next higher level.**

- Data independence occurs because when the schema is changed at some level, the schema at the next higher level remains unchanged; **only the mapping between the two levels is changed.**

Logical data independence is the capacity to **change the conceptual schema without having to change external schemas** or application programs.

- Changes to constraints can be applied to the conceptual schema without affecting the external schemas or application programs.

Physical data independence is the **capacity to change the internal schema without having to change the conceptual schema**.

- Generally, physical data independence exists in most databases and file environments where physical **details such as the exact location of data on disk, and hardware details of storage encoding, placement, compression, splitting, merging of records, and so on are hidden from the user.**

RAMAIAH Institute of Technology

Database Languages

DBMS Languages:

- Data definition language (DDL),is used by the DBA and by database designers to define both schemas.
    - The DBMS provides a set of operations or a language called the data manipulation language(DML) and Data Control Languages(DCL) for Queries purposes.
- Storage definition language (SDL), is used to specify the internal schema. The mappings between the two schemas may be specified in either one of these languages.
- View definition language (VDL), to specify user views and their mappings to the conceptual schema, but in most DBMSs the DDL is used to define both conceptual and external schemas.

RAMAIAH Institute of Technology

DBMS Interfaces:

- **Menu-Based Interfaces for Web Clients or Browsing:**
  - **Pull-down menus are a very popular technique in Web-based user interfaces.** They are also often used in browsing interfaces, which allow a user to look through the contents of a database in an exploratory and unstructured manner.

- **Forms-Based Interfaces**
  - A forms-based interface displays a form to each user. Users can fill out all of the form entries to insert new data, or they can fill out only certain entries ,Many DBMSs have forms specification languages, which are special languages that help programmers specify such forms. SQL*Forms is a form-based language

- **Graphical User Interfaces.**
  - GUIs utilize both menus and forms. Most GUIs use a pointing device, such as a mouse, to select certain parts of the displayed schema diagram.

- **Natural Language Interfaces**
  - A natural language interface usually has its own schema, which is similar to the database conceptual schema, as well as a dictionary of important words.
- **Speech Input and Output**

RAMAIAH Institute of Technology

The Database System Environment

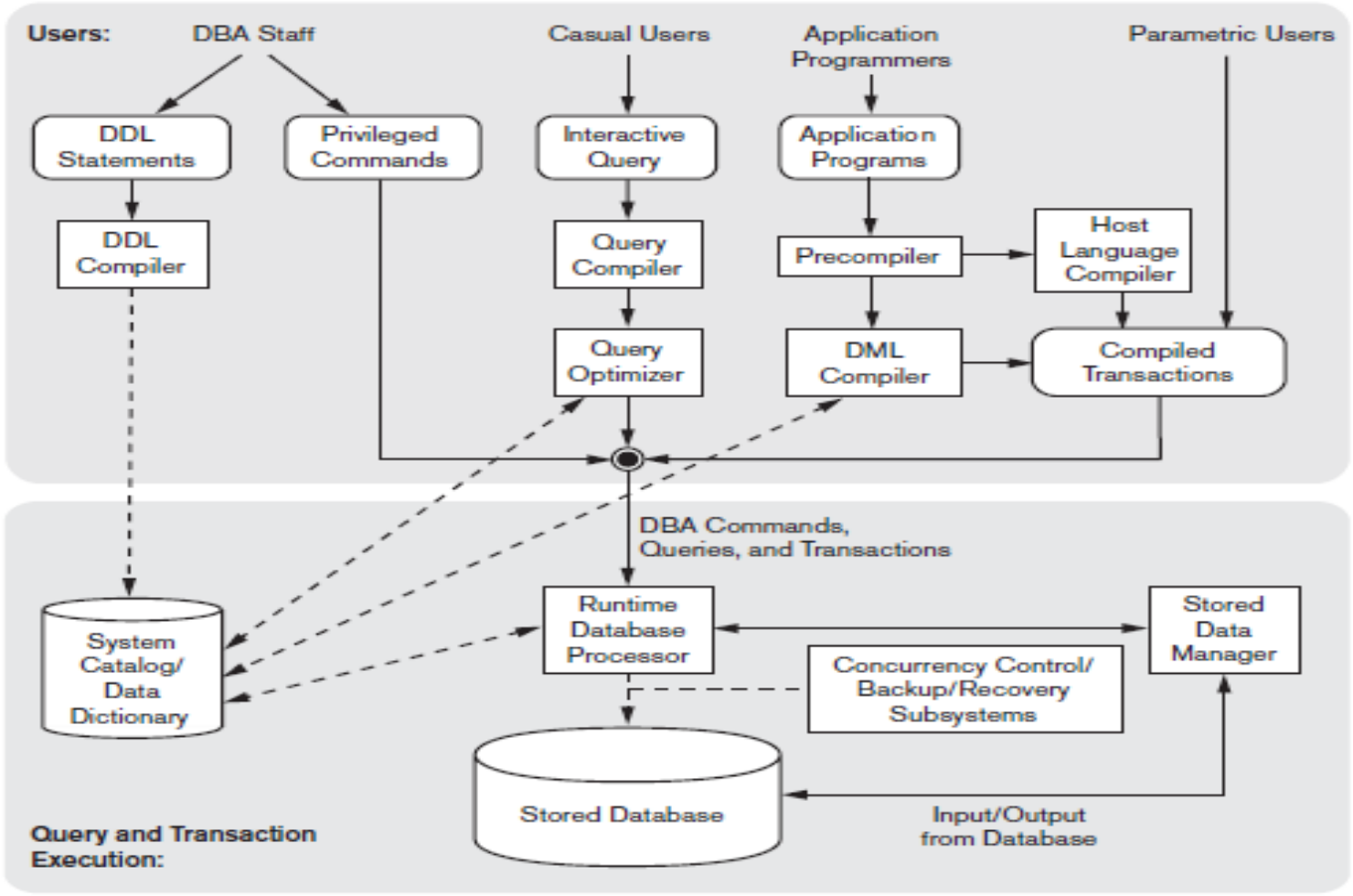


Figure 2.3
Component modules of a DBMS and their interactions.

RAMAIAH Institute of Technology

The Database System Environment

- **The top part of the figure refers** to the various users of the database environment and their interfaces.
- The **lower part shows** the internals of the DBMS responsible for storage of data and processing of transactions.

  - The **DBMS catalog** are usually stored on disk. Access to the disk is controlled primarily by the operating system (OS), which schedules disk read/write.

  - **Top part** It shows interfaces for the DBA staff, casual users who work with interactive interfaces to formulate queries, application programmers who create programs using some host programming languages, and parametric users who do data entry work by supplying parameters to predefined transactions.

  - The **DDL compiler** processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog.

  - The catalog includes information such as the **names and sizes of files, names and data types of data items, storage details of each file, mapping information among schemas, and constraints**

RAMAIAH
Institute of Technology

- These queries are parsed and validated for correctness of the query syntax, the names of files and data elements, and so on by a **query compiler** that compiles them into an internal form.

- The **query optimizer** is concerned **with the rearrangement and possible reordering of operations, elimination of redundancies, and use of correct algorithms and indexes during execution.**

- The **pre-compiler extracts DML commands** from an application program written in a host programming language. These commands are sent to the DML compiler for compilation into object code for database access.

- **In the lower part of Figure 2.3**,the **runtime database processor** executes
  - privileged commands,
  - executable query plans, and
  - canned transactions with runtime parameters.
  - It works with the system catalog and may update it with statistics. It also works with the stored data manager, which in turn uses basic operating system services for carrying out low-level input/output (read/write) operations between the disk and main memory. The

RAMAIAH
Institute of Technology

Centralized and Client/Server Architectures for DBMSs

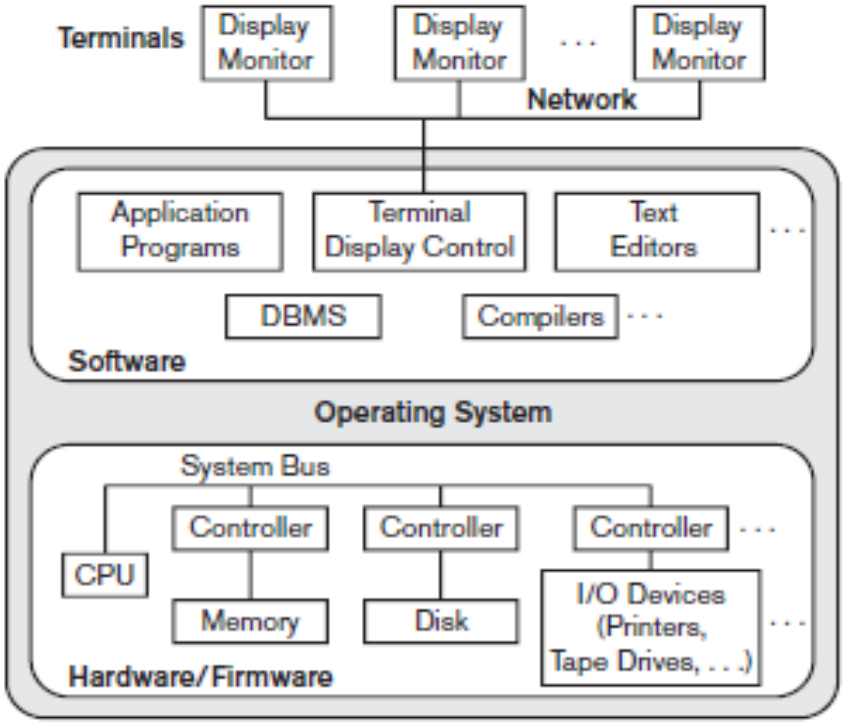


Figure 2.4
A physical centralized architecture.

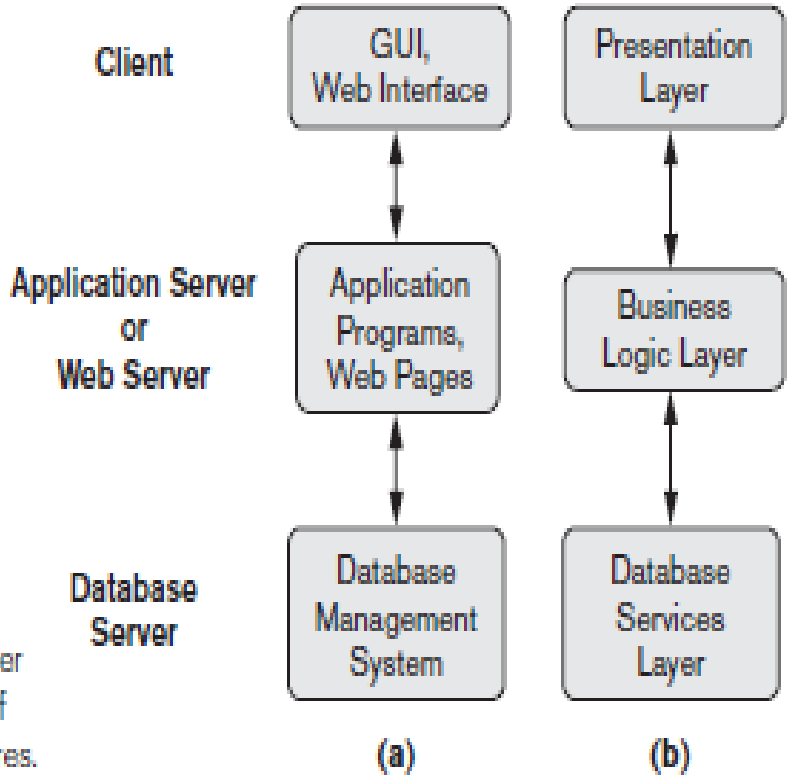


Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

RAMAIAH Institute of Technology

Classification of Database Management Systems

Several criteria are normally used to classify DBMSs.

- The first is the **data model** on which the DBMS is based.
    - The main data model used in many current commercial DBMSs is the **relational data model**.
    - The **object data model** has been implemented in some commercial systems but has not had widespread use.
    - Many legacy applications still run on database systems based on the **hierarchical and network data models.**

The second criterion used to classify DBMSs is **the number of users supported by the system.**

- **Single-user systems** support only one user at a time and are mostly used with PCs.
- **Multiuser systems**, which include the majority of DBMSs, support concurrent multiple users.

The third criterion is **the number of sites over which the database is distributed.**

- A **DBMS is centralized** if the data is stored at a single computer site. A centralized DBMS can support multiple users, but the DBMS and the database reside totally at a single computer site.

- A **distributed DBMS (DDBMS)** can have the actual database and DBMS software distributed over many sites, connected by a computer network.

CHAPTR-3

Elmasri and Navathe: Fundamentals of Database Systems, 5th Edition, Addison-Wesley, 2011.

RAMAIAH Institute of Technology

Entity-Relationship (ER) Model

•**Conceptual modeling** is very important phase in designing a successful database application.

- •For example, a BANK database application that keeps track of customer accounts would include programs that implement database updates corresponding to customer deposits and withdrawals.

•Traditionally, the design and testing of application programs has been considered to be part of **software engineering** rather than **database design**.

- •In many software design tools, the database design methodologies and software engineering methodologies are connect since these activities are strongly related

- The design of application programs is typically covered in software engineering. The modeling concepts of the **Entity-Relationship (ER) model**, **which is a popular high-level conceptual data model.**

- Object modeling methodologies such as the **Unified Modeling Language (UML)** are becoming increasingly popular in both database and software design. These methodologies go beyond database design to specify detailed design of software modules and their interactions using various types of diagrams

RAMAIAH Institute of Technology

Entity-Relationship (ER) Model

Using High-Level Conceptual Data Models for Database Design:

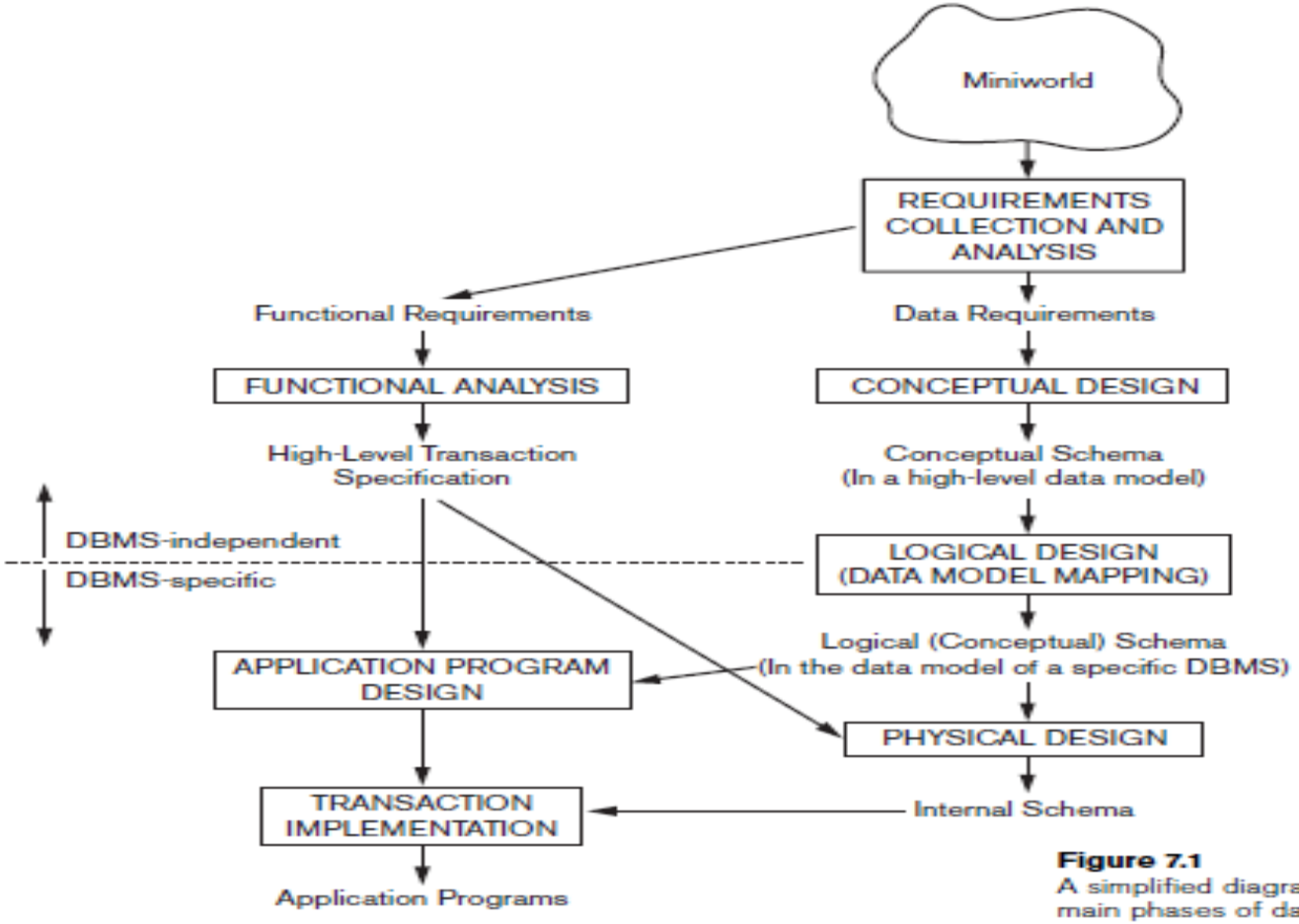


Figure 7.1
A simplified diagram to illustrate the main phases of database design.

RAMAIAH Institute of Technology

A Sample Database Application

COMPANY DATA BASE,

- **which serves to illustrate the basic ER model concepts** and their use in schema design.
- The COMPANY database keeps track of a company's employees, departments, and projects.

•The database designers provide the following description of the miniworld—the part of the company that will be represented in the database.

RAMAIAH Institute of Technology

COMPANY DATA BASE: An Example of DB Application

1. The company is organized into **departments.** Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.

2. A department controls a number of **projects**, each of which has a unique name, a unique number, and a single location.

3. We store each employee's name, Social Security number,2 address, salary, sex (gender), and birth date. An **employee** is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee).

4. We want to keep track of the **dependents of each employee** for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee.

RAMAIAH
Institute of Technology

Entity Types, Entity Sets, Attributes, and Keys

An Entity, which is a thing in the real world with an independent existence.

- An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or
- It may be an object with a conceptual existence (for instance, a company, a job, or a university course).

- **Each entity has attributes**—the particular properties that describe it.
  - For example, an EMPLOYEE entity may be described by the employee's name, age, address, salary, and job

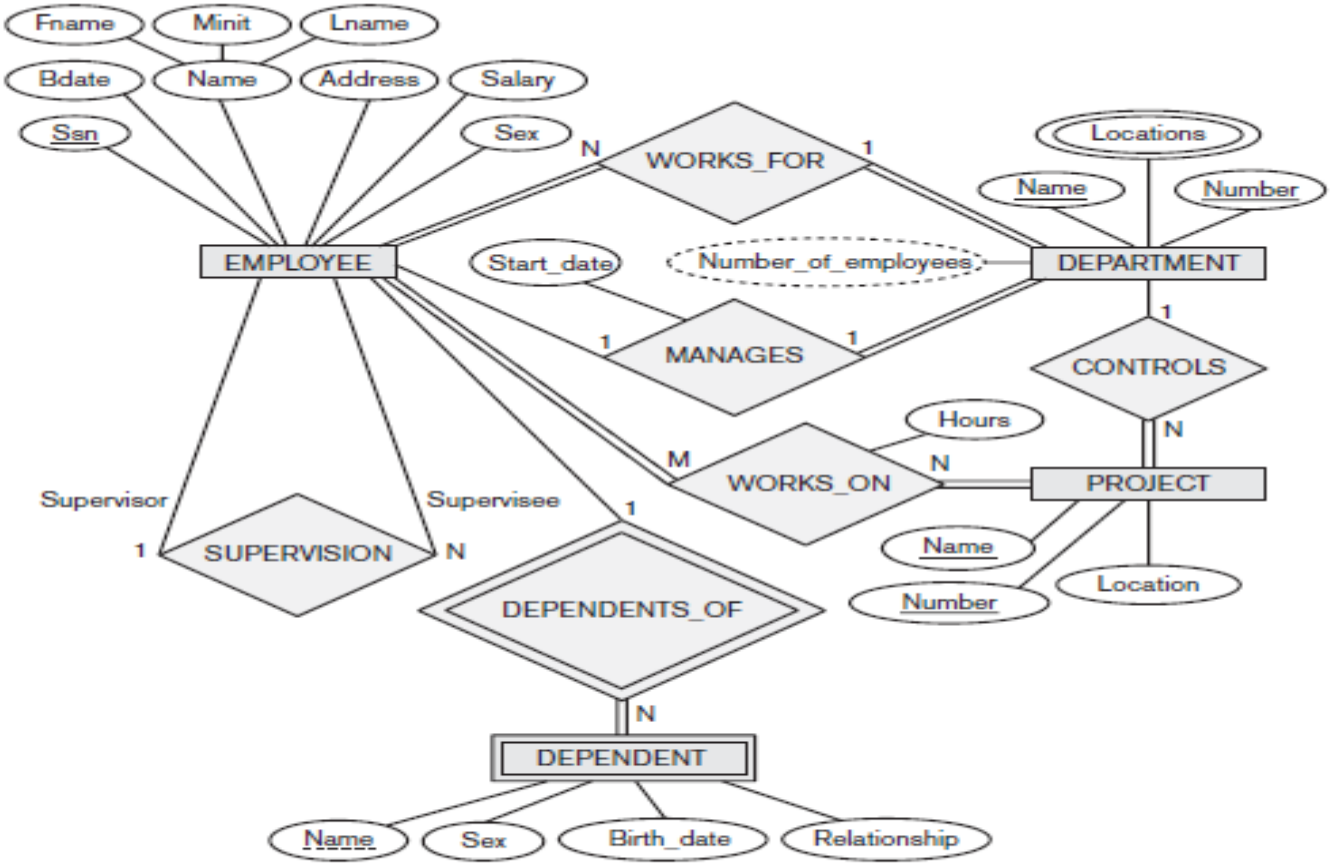


Figure 7.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 7.14.

RAMAIAH
Institute of Technology

Several types of attributes occur in the ER model:

- **simple versus composite,**
- **Single-valued versus multi-valued, and**
- **stored versus derived.**

Composite versus Simple (Atomic) Attributes.:

- **Composite attributes can be divided into smaller subparts**, which represent more basic attributes with independent meanings. For example, the Address attribute of the EMPLOYEE entity
- Attributes that are **not divisible are called simple or atomic attributes**

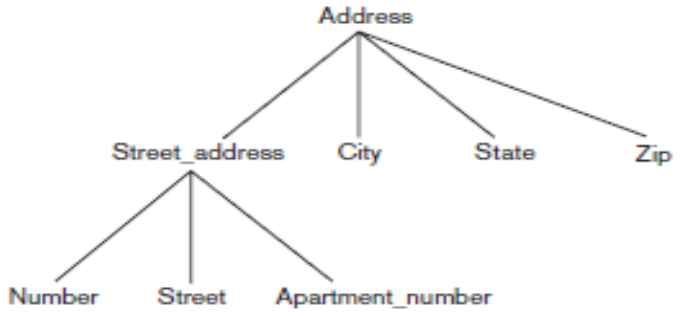


Figure 7.4
A hierarchy of composite attributes.

RAMAIAH
Institute of Technology

Single-Valued versus Multi-valued Attributes:

• Most attributes have a single value for a particular entity; such attributes are called **single-valued.**

•For example, **Age is a single-valued attribute of a person**.

•In some cases an attribute can have a set of values for the same entity—for instance,

- **a Colors attribute for a car, or**
- **a College_degrees attribute for a person**.

•One person may not have a college degree, another person may have one, and a third person may have two or more degrees;

•different people can have different numbers of values for the **College_degrees attribute. Such attributes are called multi-valued**

RAMAIAH
Institute of Technology

Stored versus Derived Attributes.:

- In some cases, two (or more) attribute values are related—for example, the **Age and Birth_date attributes of a person**.

  - For a particular person entity, **the value of Age can be determined from the current(today's) date** and the value of that person's Birth_date.

  - The **Age attribute is hence called a derived attribute** and is said to be derivable from the **Birth_date attribute, which is called a stored attribute**.

NULL Values:

- The meaning of the former type of **NULL is not applicable**, whereas the meaning of the latter **is unknown**.
- The unknown category of NULL can be further classified into two cases.
  - The first case arises when it is known that the **attribute value exists but is missing—for instance,** if the Height attribute of a person is listed as NULL.
  - The second case arises **when it is not known whether the attribute value exists**—for example, if the Home_phone attribute of a person is NULL.

RAMAIAH
Institute of Technology

Complex Attributes: Notice that, in general, **composite and multi-valued attributes can be nested arbitrarily.**

•arbitrary nesting by grouping components of a composite attribute between parentheses () and separating the components with commas, and by displaying multi-valued attributes between braces { }. Such attributes are **called complex attributes**

{Address_phone({Phone(Area_code,Phone_number)},Address(Street_address (Number,Street,Apartment_number),City,State,Zip))}

Figure 7.5
A complex attribute: Address_phone.

RAMAIAH Institute of Technology

Entity Types, Entity Sets, Keys, and Value Sets

- **Entity Type :** A database usually contains groups of entities that are similar. **An entity type defines a collection (or set) of entities that have the same attributes.** Each entity type in the database is described by its name and attributes.

- **Entity Set :** The **collection of all entities of a particular entity type in the database at any point in time is called an entity set**; the entity set is usually referred to using the same name as the entity type

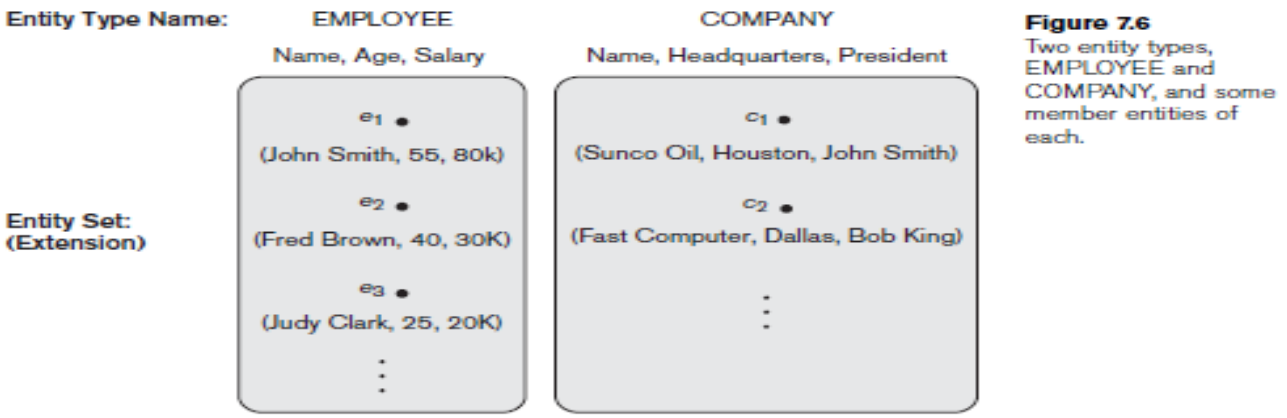


Figure 7.6
Two entity types, EMPLOYEE and COMPANY, and some member entities of each.

RAMAIAH
Institute of Technology

Value Sets (Domains) of Attributes: Each simple attribute of an entity type is associated with a value set (or domain of values), which specifies the set of values that may be assigned to that attribute for each individual entity.

- if the **range of ages allowed for employees is between 16 and 70**, we can specify the value set of the Age attribute of EMPLOYEE to be the set of integer numbers between 16 and 70.

- Similarly, we can specify the **value set for the Name attribute** to be the set of strings of alphabetic characters separated by blank characters, and so on.

- Value sets are not displayed in ER diagrams, and are typically specified using the **basic data types** available in most programming languages, such as integer, string, Boolean, float, enumerated type, subrange, and so on.

RAMAIAH Institute of Technology

Initial Conceptual Design of the COMPANY Database

1. An entity type **DEPARTMENT** with attributes Name, Number, Locations, Manager, and Manager_start_date. Locations is the only multi-valued attribute. We can specify that both Name and Number are (separate) key attributes because each was specified to be unique.

2. An entity type **PROJECT** with attributes Name, Number, Location, and Controlling_department. Both Name and Number are (separate) key attributes.

3. An entity type **EMPLOYEE** with attributes Name, Ssn, Sex, Address, Salary, Birth_date, Department, and Supervisor. Both Name and Address may be composite attributes; however, this was not specified in the requirements. We must go back to the users to see if any of them will refer to the individual components of Name—First_name, Middle_initial, Last_name—or of Address.

4. An entity type **DEPENDENT** with attributes Employee, Dependent_name, Sex, Birth_date, and Relationship (to the employee).

RAMAIAH Institute of Technology

Relationship Types, Relationship Sets, Roles,

A **relationship type R** among n entity types E1, E2, ..., En defines a set of associations— or a **relationship set**— among entities from these entity types.

Degree of a Relationship Type: The degree of a relationship type is the number of participating entity types. Hence, the WORKS_FOR relationship is of degree two. A **relationship type of degree two is called binary**, and one of degree three is called ternary.

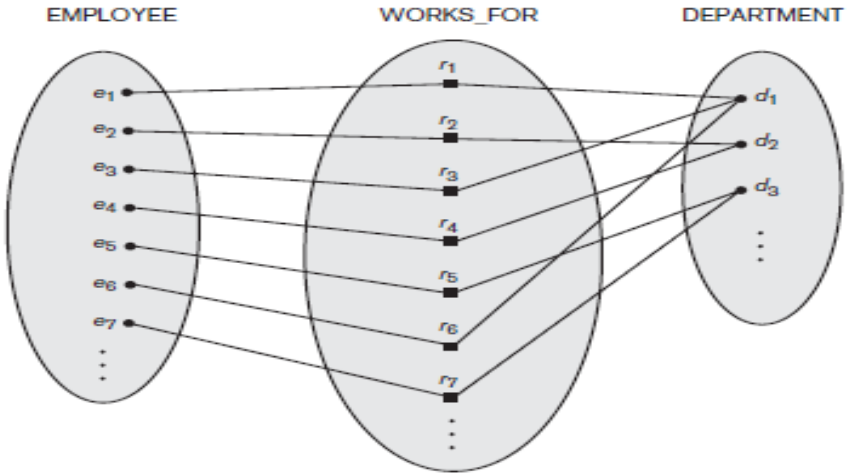


Figure 7.9
Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.

RAMAIAH
Institute of Technology

An example of **a ternary relationship is SUPPLY**, shown in Figure 7.10, where each relationship instance ri associates three entities—a supplier s, a part p, and a project j—whenever s supplies part p to project j.

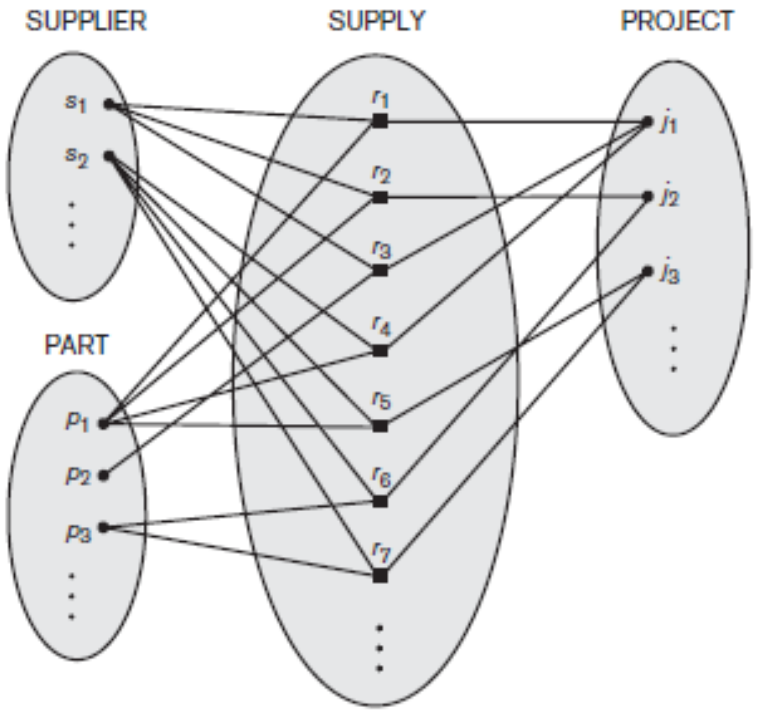


Figure 7.10
Some relationship instances in the SUPPLY ternary relationship set.

The **role name** signifies the role that a participating entity from the entity type plays in each relationship instance, and helps to explain what the relationship means.

Recursive Relationships: In some cases the same **entity type participates more than once in a relationship type in different roles.** In such cases the role name becomes essential for distinguishing the meaning of the role that each participating entity plays. Such relationship types are called **recursive relationships.**

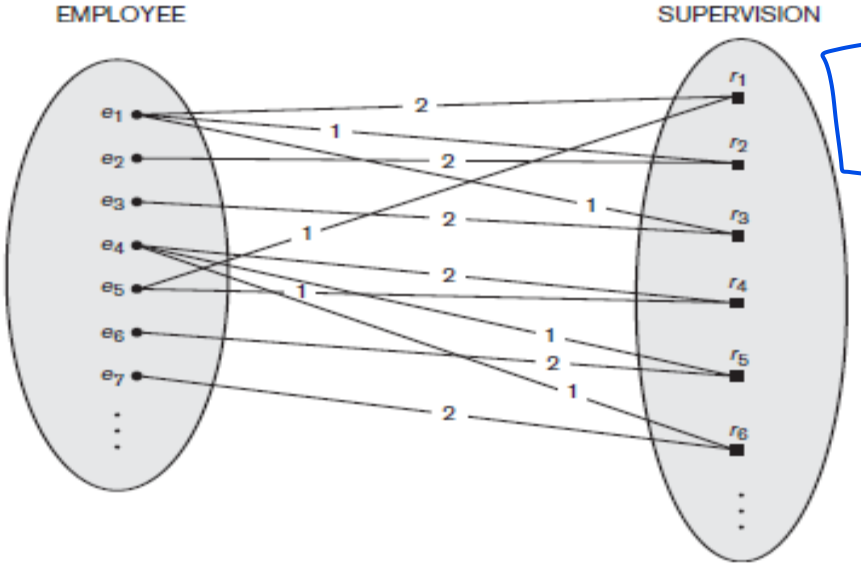


Figure 7.11
A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).

RAMAIAH
Institute of Technology

•**Constraints on Binary Relationship Types:** We can distinguish two main types of binary relationship constraints: **cardinality ratio and participation.**

•**Cardinality Ratios for Binary Relationships:** The cardinality ratio for a binary relationship specifies the **maximum number of relationship instances that an entity can participate in.**

•The possible cardinality ratios for binary relationship types are **1:1, 1:N, N:1, and M:N.**

An example of a 1:1 binary relationship is MANAGES (Figure 7.12), which relates a department entity to the employee who manages that department. This represents the mini-world constraints **that—at any point in time—an employee can manage one department only and a department can have one manager only.**

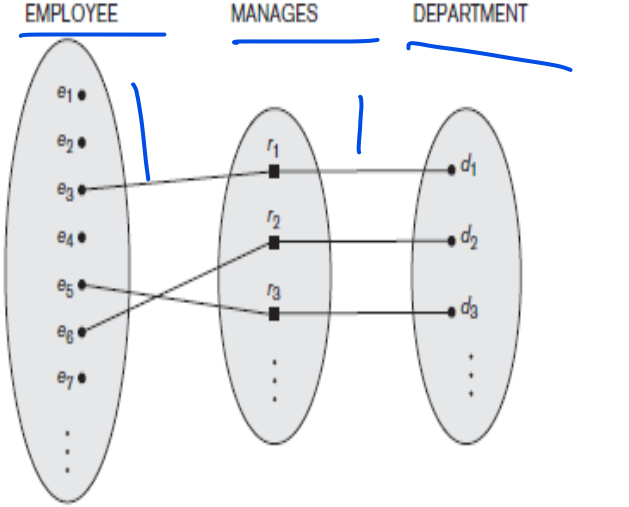


Figure 7.12
A 1:1 relationship, MANAGES.

RAMAIAH
Institute of Technology

•For example, in the **WORKS_FOR binary relationship type, v DEPARTMENT:EMPLOYEE is of cardinality ratio 1:N**, meaning that each department can be related to (that is, employs) any number of employees,9 but an employee can be related to (work for) only one department.

•The relationship type **WORKS_ON (Figure 7.13) is of cardinality ratio M:N**, because the mini-world rule is that an employee can work on several projects and a project can have several employees.

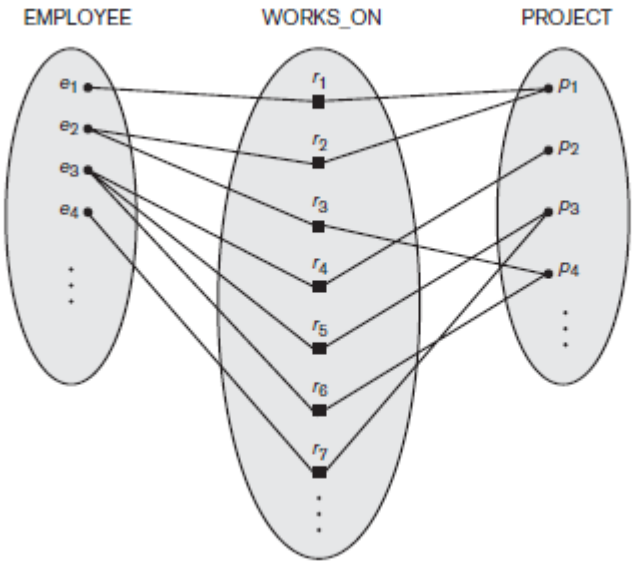


Figure 7.13
An M:N relationship, WORKS_ON.

RAMAIAH
Institute of Technology

The **participation constraint** specifies whether the existence of an entity depends on its being related to another entity via the relationship type.

This constraint specifies the minimum number of relationship instances that each entity can participate in, and is sometimes called the **minimum cardinality constraint.**

- There are two **types of participation constraints—total and partial**

If a company policy states that every employee must work for a department, then an employee entity can exist only if it participates in at least one WORKS_FOR relationship instance (Figure 7.9).

- Thus, the participation of **EMPLOYEE in WORKS_FOR is called total participation,** meaning that every entity in the total set of employee entities must be related to a department entity via WORKS_FOR.

- The participation of **EMPLOYEE in the MANAGES relationship type is partial,** meaning that some or part of the set of employee entities are related to some department entity via MANAGES, but not necessarily all.

RAMAIAH
Institute of Technology

Weak Entity Types:

- **Entity types** that **do not have key attributes of their own** are called **weak entity types**.

- In contrast, regular entity types that do have a key attribute—which include all the examples discussed so far—are called **strong entity types.**

- Entities belonging to a **weak entity type** are identified by being related to specific entities from another entity type in combination with one of their attribute values.

- Consider the entity type **DEPENDENT, related to EMPLOYEE**, which is used to keep track of the dependents of each employee via a 1:N relationship (Figure 7.2).

- **A weak entity type normally has a partial key**, which is the attribute that can uniquely identify weak entities that are related to the same owner entity.

RAMAIAH Institute of Technology

Refining the ER Design for the COMPANY Database: we specify the following relationship types:

1. **MANAGES, a 1:1 relationship type** between EMPLOYEE and DEPARTMENT. EMPLOYEE **participation is partial**..

2. **WORKS_FOR, a 1:N relationship type** between DEPARTMENT and EMPLOYEE**. Both participations are total.**

3. **CONTROLS, a 1:N relationship type** between DEPARTMENT and PROJECT. The participation of **PROJECT is total,** whereas that of **DEPARTMENT is determined to be partial,**

4. **SUPERVISION, a 1:N relationship type** between EMPLOYEE (in the supervisor role) and EMPLOYEE (in the supervisee role). Both participations are determined to be partial,

5. **WORKS_ON, determined to be an M:N relationship type** with attribute Hours, after the users indicate that a project can have several employees working on it. Both participations are determined to be total.

6. **DEPENDENTS_OF, a 1:N relationship type** between EMPLOYEE and DEPENDENT, which is also the identifying relationship for the weak entity

RAMAIAH
Institute of Technology

Design Choices for ER Conceptual Design

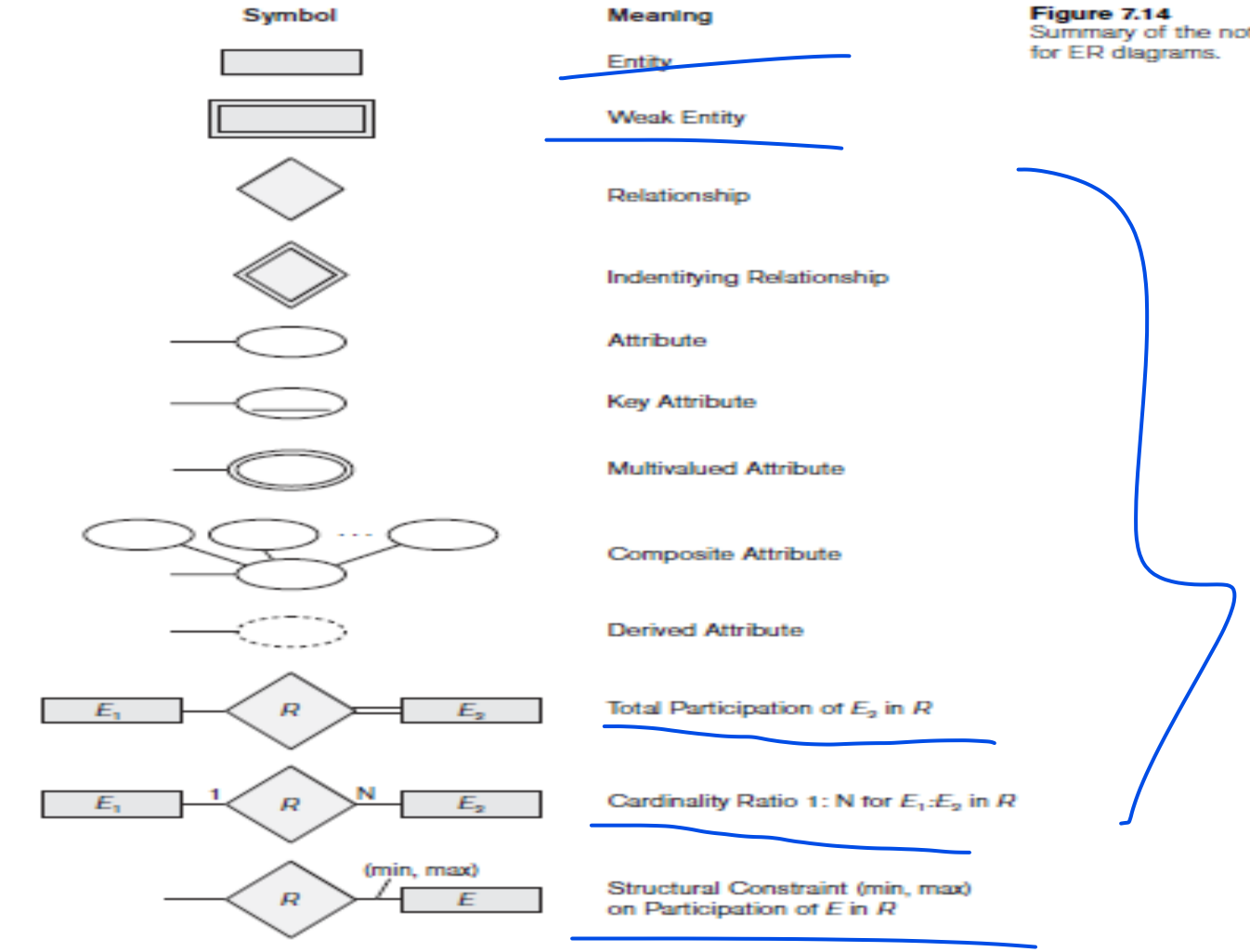


Figure 7.14
Summary of the notation for ER diagrams.

RAMAIAH
Institute of Technology

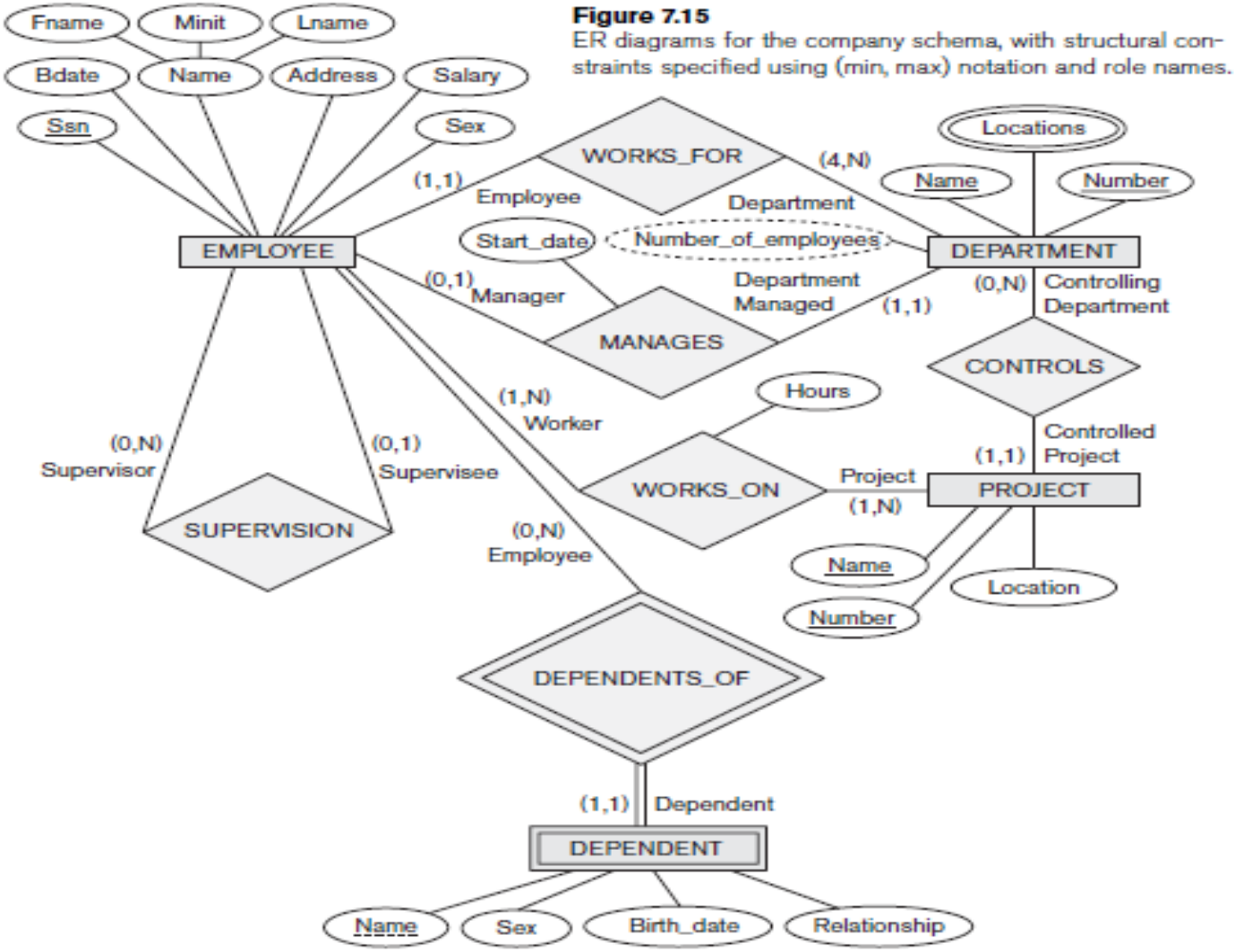


Figure 7.15
ER diagrams for the company schema, with structural constraints specified using (min, max) notation and role names.