

{ **Important,* ****Very Important* }

CHAPTER 1:

Data: Raw facts. Processed data is called Information.

*** Database:** A database is collection of related data that represents a real-world entity (object).

*** DBMS:** A database management system (DBMS) is a computerized system that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.

*** Data Catalog / Meta Data:** The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called meta-data.

Phases for Designing DBMS:

- Requirement Specification and Analysis
- Conceptual Design
- Logical Design
- Physical design

*****Characteristics of DB Approach:**

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

- Self-describing nature of a database system.
 - A complete definition or description of the database structure and constraints.
 - This definition is stored in the DBMS catalog, which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data.
 - The information stored in the catalog is called meta-data.
- Insulation between programs and data, and data abstraction.
 - DBMS access programs do not require changes in most cases.
 - 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.
- Support of multiple views of the data.
 - A database typically has many types of users, each of whom may require a different perspective or view of the database.
 - A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.
 - Some users may not need to be aware of whether the data they refer to is stored or derived.

- 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.

Actors on the Scene:

- Database Administrators.
- Database Designers
- End users*
- System Analysts and Application Programmers(Software Engineers).

Workers behind the Scene:

- DBMS -system designers and implementers.
- Tool developers
- Operators and maintenance personnel

***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.
- The different Types are:-
 - **Casual end users:**
 - Occasionally access the database, but they may need different information each time.
 - They use a sophisticated database query interface to specify their requests and are typically middle- or high-level managers or other occasional browsers.
 - **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. The tasks that such users perform are varied. A few examples are:
 - ❖ Bank customers and tellers check account balances and post withdrawals and deposits.
 - ❖ Reservation agents or customers for airlines, hotels, and car rental companies check availability for a given request and make reservations.
 - ❖ Employees at receiving stations for shipping companies enter package identifications via bar codes and descriptive information through buttons to update a central database of received and in-transit packages.
 - **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.
- An example is the user of a financial software package that stores a variety of personal financial data.

***** Advantages of DBMS over Traditional (File Processing System) Approach:**

{some of these can be explained in own words - **write based on marks allotted for this question, easier points are on the top and marked ***}

- ***Potential for Enforcing Standards.** The database approach permits the DBA (DB Admin) to define and enforce standards among database users in a large organization.
- ***Reduced Application Development Time.** Designing and implementing a large multiuser database from scratch may take more time than writing a single specialized file application. However, once a database is up and running, substantially less time is generally required to create new applications. **Development time using a DBMS is estimated to be one-sixth to one-fourth of that for a file system.**
- ***Flexibility.** Modern DBMSs allow certain types of evolutionary changes to the structure of the database without affecting the stored data and the existing application programs
- ***Availability of Up-to-Date Information.** As soon as one user's update is applied to the database, all other users can immediately see this update. This availability of up-to-date information is essential for many transaction-processing applications, such as reservation systems or banking databases, and it is made possible by the concurrency control and recovery subsystems of a DBMS
- ***Economies of Scale.** The DBMS approach permits consolidation of data and applications, thus reducing the amount of wasteful overlap between activities of data-processing personnel in different projects or departments as well as redundancies among applications. This enables the whole organization to invest in more powerful processors, storage devices, or networking gear, rather than having each department purchase its own (lower performance) equipment. This reduces overall costs of operation and management.
- ***Restricting unauthorized Access:** For example, only the DBA's staff may be allowed to use certain privileged software, such as the software for creating new accounts. Similarly, parametric users may be allowed to access the database only through the predefined apps or canned transactions developed for their use.
- ***Providing Persistent storage for program objects:** 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 them were incompatible with the programming language's data structures. Object-oriented database systems typically offer data structure compatibility with one or more object-oriented programming languages.
- ***Providing Backup and Recovery.**
- ***Providing Multiple User Interfaces.** These include apps for mobile users, 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.
- ***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.

- **Controlling Redundancy**
- **Providing Storage Structures and Search Techniques for Efficient Query Processing.** 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. 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.
- **Enforcing Integrity Constraints.**
- **Permitting Inferencing and Actions Using Rules and Triggers.** Some database systems provide capabilities for defining deduction rules for inferencing new information from the stored database facts. These can be specified declaratively as rules, 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.

Question Bank:

1. List and discuss the advantages of Database Management System over File processing system.
2. Explain the characteristics of the Database approach.
3. Define the following terms with example. a) Database b) DBMS c) Meta-Data
4. Explain the types of end users with suitable exams.
5. List and explain the people who works on the scene and behind the scene for creating any database.
6. With a neat diagram, explain a simplified database system environment

=====

CHAPTER 2:

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.

Categories of Data Models:

- **High-level or conceptual data models** provide concepts that are close to the way many users perceive data
- **low-level or physical data models** provide concepts that describe the details of how data is stored on the computer storage media, typically magnetic disks.
- **representational (or implementation) data models**, 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.

DataBase Schema: The description of a database is called the database schema, which is specified during database design and is not expected to change frequently. A displayed schema is called a schema diagram.

***Entity:** 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.

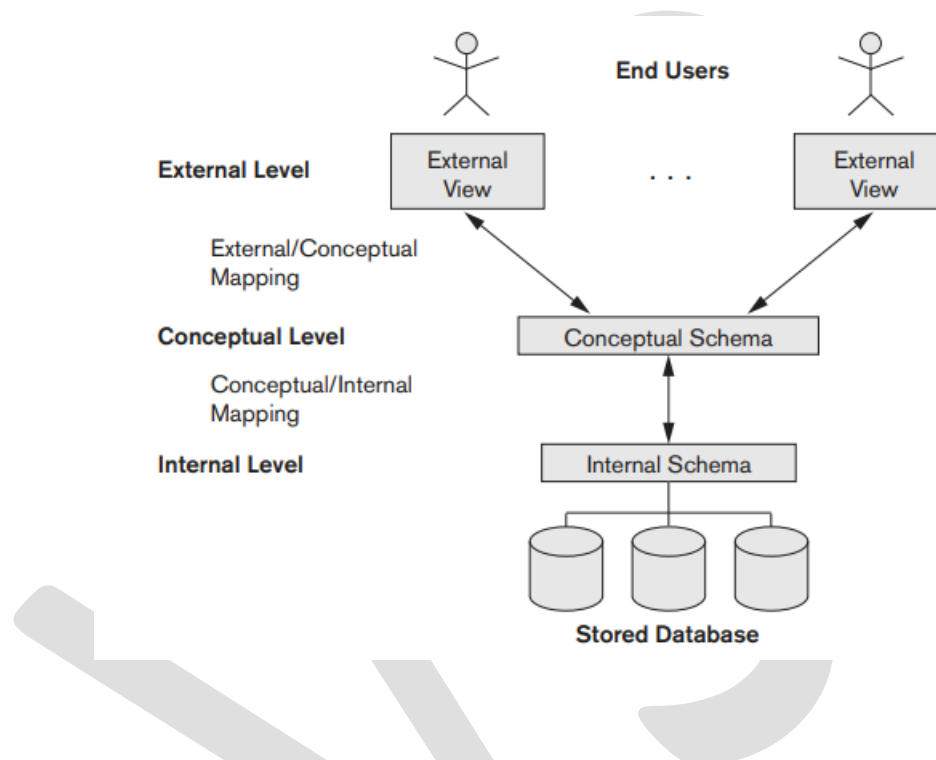
Attribute: An attribute represents some property of interest that further describes an entity, such as the employee's name or salary.

Relationship: A relationship among two or more entities represents an association among the entities, for example, a works-on relationship between an employee and a project.

***Snapshot/DB state:** 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. In a given database state, each schema construct has its own current set of instances.

*** Three-Schema Architecture:

The three-schema architecture is a convenient tool with which the user can visualize the schema levels in a database system.



- **The internal level**
 - has an internal schema, which describes the physical storage structure of the database.
 - The internal schema uses a physical data model and describes the complete details of data storage and access paths for 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.
 - Usually, a representational data model is used to describe the conceptual schema when a database system is implemented.
- **The external or view level**
 - includes several 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.

- each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level conceptual data model.

Data Independence:

- **Logical data independence** is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item).
- **Physical data independence** is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized.

DataBase Languages:

- **Data definition language (DDL)** • Used by DBA & Designers: Defines both schemas (Conceptual & Internal)
- **Storage definition language (SDL)** • Specifies the internal schema.
- **View definition language (VDL)** • Specifies user views/mappings to conceptual schema.
- **Data manipulation language (DML)** • Allows retrieval, insertion, deletion, modification.

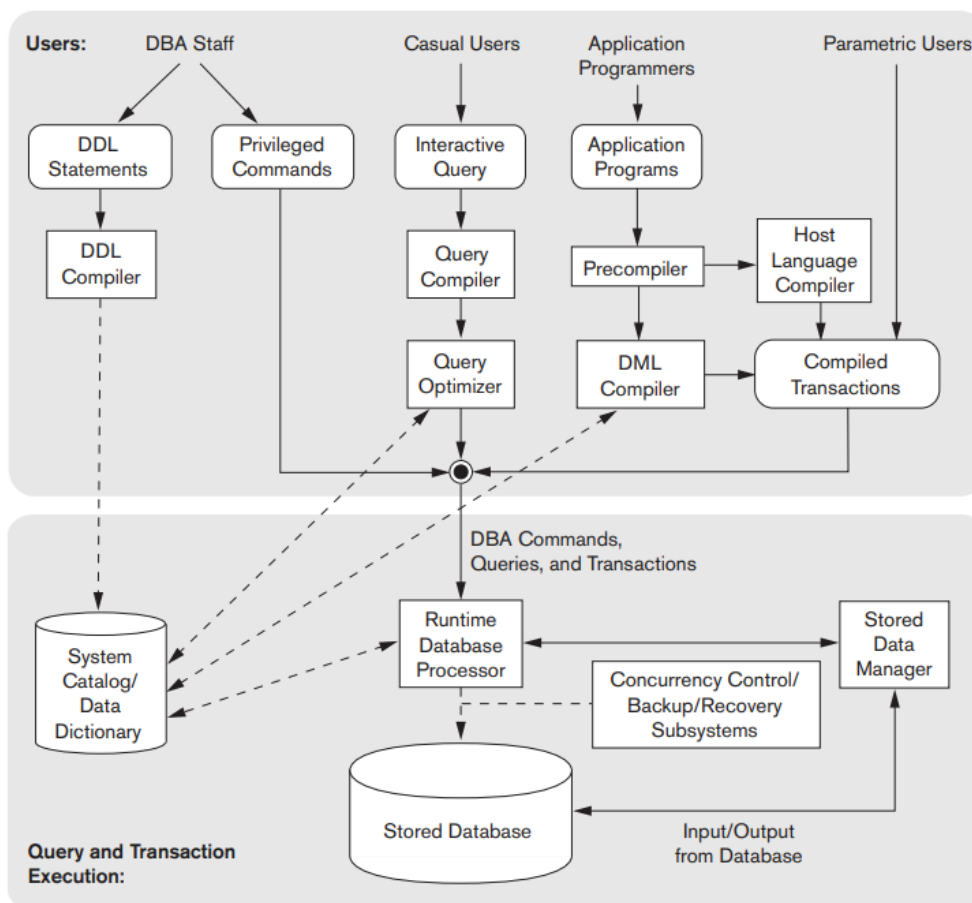
* DataBase Interfaces:

- **Menu-based interfaces for Web clients or browsing.** These interfaces present the user with lists of options (called menus) that lead the user through the formulation of a request.
- **Apps for Mobile Devices.** These interfaces present mobile users with access to their data. For example, banking, reservations, and insurance companies, among many others, provide apps that allow users to access their data through a mobile phone or mobile device.
- **Forms-based interfaces.** A forms-based interface displays a form to each user. Users can fill out all the form entries to insert new data, or they can fill out only certain entries, in which case the DBMS will retrieve matching data for the remaining entries.
- **Graphical user interfaces.** A GUI typically displays a schema to the user in diagrammatic form. The user then can specify a query by manipulating the diagram. In many cases, GUIs utilize both menus and forms.
- **Natural language interfaces.** These interfaces accept requests written in English or some other language and attempt to understand them. A natural language interface usually has its own schema, which is like the database conceptual schema, as well as a dictionary of important words
- **Keyboard based Database Search.** These are somewhat like Web search engines, which accept strings of natural language (like English or Spanish) words and match them with documents at specific sites (for local search engines) or Web pages on the Web at large (for engines like Google or Ask)
- **Speech Input and Output.** Limited use of speech as an input query and speech as an answer to a question or result of a request is becoming commonplace. Applications with limited vocabularies, such as inquiries for telephone directory, flight arrival/departure, and credit

card account information, are allowing speech for input and output to enable customers to access this information

- **Interfaces for parametric users.** Parametric users, such as bank tellers, often have a small set of operations that they must perform repeatedly. Usually, a small set of abbreviated commands is included, with the goal of minimizing the number of keystrokes required for each request
- **Interfaces for the DBA.** Most database systems contain privileged commands that can be used only by the DBA staff. These include commands for creating accounts, setting system parameters, granting account authorization, changing a schema, and reorganizing the storage structures of a database.

Database System Environment:



DBMS component modules:

- Runtime database processor.
- System catalog.
- Concurrency control system.
- Backup and recovery system

Database System Utilities:

- **Loading:** Load existing data files. Conversion tools.
- **Backup:** Creates a backup copy of the database. Incremental backups.
- **Database storage reorganization:** Reorganize a set of database files into different file organizations.

- **Performance monitoring:** Monitors database usage and provides statistics to the DBA.

Question Bank:

1. Explain three schema architecture and reason for need of mapping among schema level.
2. Discuss the different types of user-friendly interfaces for Database design.
3. Define the terms i) Data Model ii) Database Schema iii) Database State or Snapshot iv) Instances v) Entity
4. Explain the different categories of Data Model

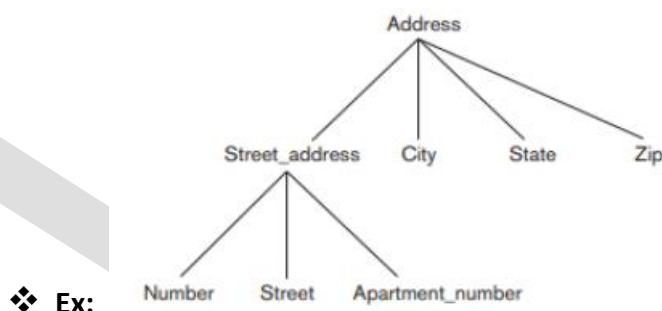
CHAPTER 3:

***ENTITY ATTRIBUTES types (explain with example for each):

The Attribute Types

- **Composite versus simple (atomic) attributes:**

- ❖ Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings.
- ❖ Attributes that are not divisible are called simple or atomic attributes.



- **Single-Valued versus Multivalued Attributes:**

- ❖ Most attributes have a single value for a particular entity; such attributes are called single-valued. **Ex:** Age is a single-valued attribute of a person.
- ❖ In some cases, an attribute can have a set of values for the same entity, Such attributes are called multivalued **Ex:** Colors attribute for a car, or a College_degrees attribute for a person. Multivalued attributes are represented with double oval symbol.

- **Stored versus Derived Attributes:**

- ❖ In some cases, two (or more) attribute values are related to each other.
- ❖ Stored attribute is an attribute which are physically stored in the database.
- ❖ Assume a table called as student. **Ex:** There are attributes such as student_id, name, roll_no, course_id. We cannot derive value of these attribute using other attributes. So, these attributes are called as stored attribute.

- ❖ A derived attribute is an attribute whose values are calculated from other attributes.
Ex: In a student table if we have an attribute called as date_of_birth and age. We can derive value of age with the help of date_of_birth attribute.
- ❖ Stored attribute is represented by an oval.
- ❖ Derived attribute is represented by a dotted oval.
- **NULL Values:**
 - ❖ In some cases, a particular entity may not have an applicable value for an attribute. For such situations, a special value called NULL is created.
 - ❖ **Ex:** Apartment_number, College_Degree
- **Complex Attributes:**
 - ❖ composite and multivalued attributes can be nested arbitrarily. Such attributes are called complex attributes.
 - ❖ **Ex:** {Address_phone({Phone(Area_code,Phone_number)}), Address(Street_address(Number,Street,Apartment_number), City,State,Zip) }}

Key Attribute: An entity type usually has one or more attributes whose values are distinct for each individual entity in the entity set. Such an attribute is called a key attribute, and its values can be used to identify each entity uniquely.

***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.

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

***Cardinality Ratios:** The cardinality ratio for a binary relationship specifies the maximum number of relationship instances that an entity can participate in. For example, in the WORKS_FOR binary relationship type, (DEPARTMENT: EMPLOYEE) is of cardinality ratio (1:N), meaning that each department can be related to any number of employees (N), but an employee can be related to (work for) at most one department (1).

***Total Participation:** There are two types of participation constraints—total and partial—that we illustrate by example. 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. 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. Total participation is also called **existence dependency**.

Partial Participation: (considering above example) We do not expect every employee to manage a department, so 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.

***Weak Entity:** 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 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. We call this other entity type **the identifying or owner entity type**, and we call the relationship type that relates a weak entity type to its owner the **identifying**

relationship of the weak entity type. 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.

For Reference:








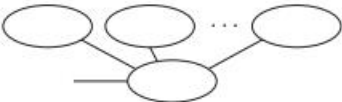



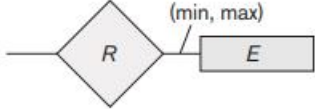
Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E_2 in R
	Cardinality Ratio 1: N for $E_1 : E_2$ in R
	Structural Constraint (min, max) on Participation of E in R

Figure 3.14
Summary of the notation for ER diagrams.

*(FEB 2022 Question paper) ER DIAGRAM FOR AIRLINES DB SCHEMA: (FROM TEXTBOOK)

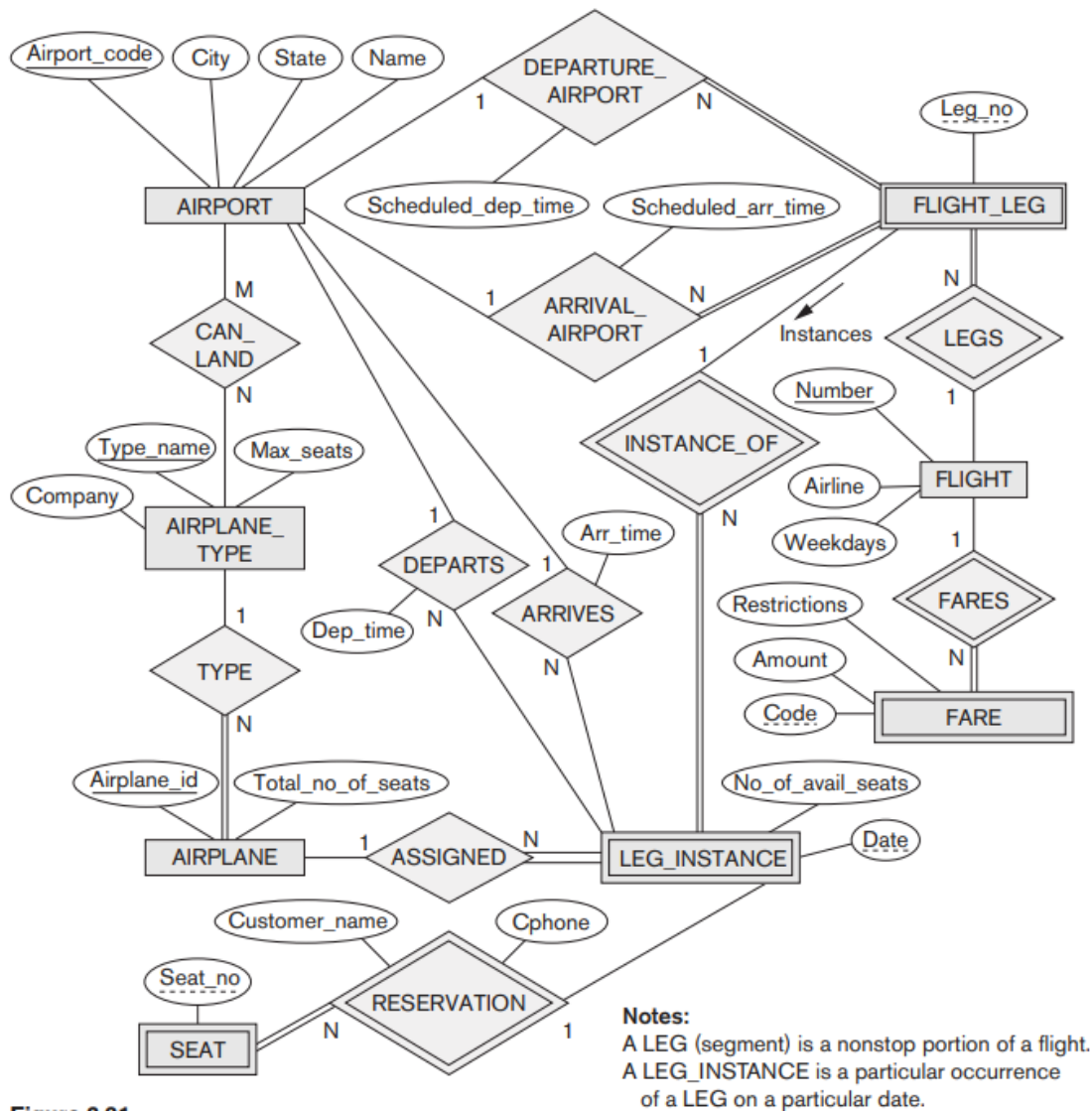
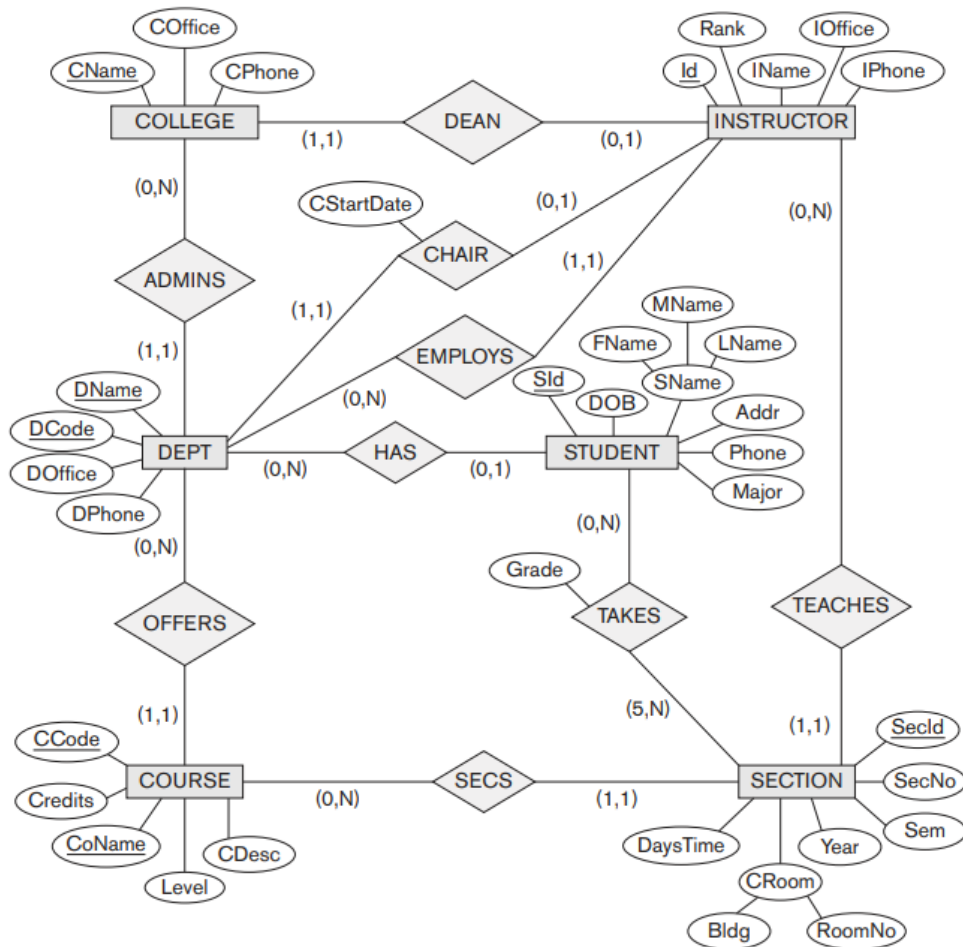


Figure 3.21

Question Bank:

- 1) Explain different types of attributes that occur in an ER diagram with example.
2. Draw an ER diagram for an airline database schema with atleast five entities. Also mention the primary key and structural constraints.
3. Draw an ER diagram for a company database with atleast four entities.
4. Explain recursive relationship types with examples.
5. Write a note on Participation constraints on binary relationship types.
6. Define the following terms i) Cardinality ii) Weak entity iii) Program data Independence iv) Total participation v) Value Sets

ER Diagram for UNIVERSITY DB Schema: (FROM TEXTBOOK)



ER Diagram for COMPANY DB schema: (FROM TEXTBOOK)

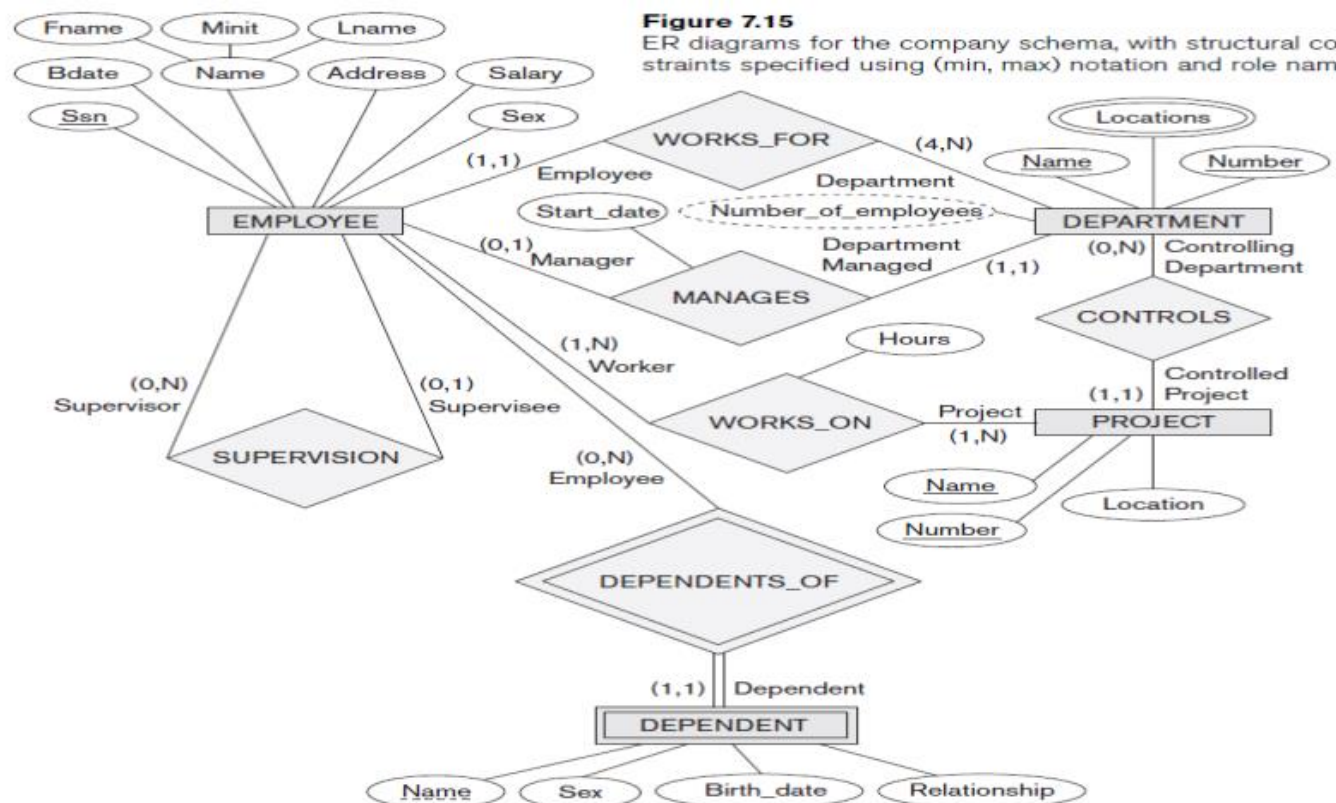


Figure 7.15

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