Short Questions

Q.1. Define data.

A collection of raw facts and figures is called data. Data is collected from different sources and is not very meaningful for making decisions. Data may consist of numbers, characters, symbols or pictures etc. Data is collected for different purposes.

Q.2. Define information.

The processed data is called information. It is an organized and processed form of data.

Q.3. Define database.

Database can be defined as an organized collection of related data. The word "organized" means that data is stored in such a way that the user can store, manipulate and retrieve data easily. The word "related" means that a database is normally created to store the data about a particular topic.

Q.4. Define database management system.

A database management system (DBMS) is a collection of programs that are used to create and maintain a database.

Q.5. Define security.

The protection of the database from unauthorized users, which may involve passwords and access restrictions.

Q.6. Define data integrity.

The reliability and accuracy of data is called data integrity. The maintenance of the validity and consistency of the database by use of particular constraints that are applied to the data.

Q.7. List four examples of database systems.

- A system that maintains component part details for a car manufacturer
- An advertising company keeping details of all clients and adverts placed with them
- A training company keeping course information and participants' details
- An organization maintaining all sales order information

Q.8. State the purpose of a database.

The purpose of a database is to help people and organizations keep track of things.

Q.9. What is metadata? Give an example.

Data about the structure of a database is called metadata. Examples of metadata are the names of tables, names of columns and tables, properties of tables and columns etc.

Q.10. Briefly describes the history of database processing.

In file processing, data was maintained in sequential lists on magnetic tape. Database processing became possible with the availability of direct access disk storage in 1960s. Using this storage, both the hierarchical data model and network data model were developed. In 1970, E. E. Codd of IBM proposed relational model that is the standard model used today. Current DBMSs such as DB2, Oracle and SQL Server are based on relational model. More recent events include the appearance of microcomputer based DBMSs, introduction of Object Oriented DBMS and development of tools such as XML used with database systems over the Internet.

Q.11. What does DBMS stand for? List the functions of a DBMS.

DBM DBMS stands for Database Management System. The functions of the DBMS are:

- Create Database
- Create Tables
- Create Supporting Structures
- Read Database Data
- Update Database Data
- Maintain Database Structures
- Enforce Rules
- Control Concurrency
- Provide Security
- Perform Backup and Recovery

Q.12. Contrasts the following terms:

Structural Dependence and Data Dependence
Data dependence and data independence
Repository and database

Structural dependence and data dependence: Any chyange in the structure of a file such as addition or deletion of a field requires the modification of all programs using that file. This modification is necessary becase the access to a file is dependent on its structure. This type of dependence is known as *structural dependence*. Any change in file data characteristics such as changing a field from integer to decimal requires modification of all programs that access the file. This type of dependence is known as *data dependence*.

Data dependence and data independence: Data dependence is a tight relationship between data and specific programs. The application programs has to be changed if there is any change data. Data independence means that data and application programs are separate from each other. The data storage structures and operations can be changed without changing application programs.

Repository and Database: A repository is a centralized storehouse of all data definitions, relationships and other system components. A database is an organized collection of logically related data.

Q.13. Why have databases become preferred method to store data used by an information system?

Databases are a common point of access, management and control. They allow data to be managed as an enterprise-wide resource. They simultaneous access to many users and application programs. They solve many problems associated with separately maintained data stores such as redundancy, inconsistent security and inconsistent data access methods.

Q.14. Give an example of a large-enterprise database application.

A construction company uses a database to keep track of project costs, labor, materials and schedule. One database supports all of these different applications.

Q.15. What were some of the weaknesses of early organizational database applications?

The earily organizational database applications were slow and unreliable. DBMS developers did not know efficient ways to provide database access. The programmers did not know how to use database technology.

Q.16. What is the major drawback of personal databases?

The major drawback of personal databases is that their data cannot be shared with other users. For this reason, they should be limited to special situations where there is not a need to share the data.

Q.17. How is relationship between application programs and DBMS changing over time?

DBMS is gradually taking on more and more of the application programs functions and roles.

Q.18. What was the major factor that gave rise to workgroup database applications?

The major factor that gave rise to workgroup database is the development and acceptance of LAN technology and products.

Q.19. List different categories of databases with example of each type.

a. **Personal computer databases:** It is a set of data that describes patient visits, recorded by a home health-care professional.
b. **Workgroup database:** It is a database that supports the work of several scientists performing research on a new drug.
c. **Department database:** It is a database used by human resources department of large hospitals.
d. **Enterprise database:** It is a database that supports SAP enterprise information system.

Q.20. Which problems does data redundancy cause?

Redundant data uses additional storage space and makes it difficult to maintain the accuracy of database when changes are made.

Q.21. Why do people use databases for information-handling tasks?

(1) Databases make it easier to store large quantities of information.
(2) Databases make it easier to retrieve information quickly and flexibly.
(3) Databases make it easy to organize and reorganize information.
(4) Databases make it easy to print and distribute information in many ways.

Q.22. What is the function of application metadata? How does it differ from metadata?

Application metadata is a description of application components such as forms, reports, menus and queries. It differs from metadata that is a description of the database structure.

Q.23. Explain the purpose of forms, reports, queries and menus.

A form is used to enable users to create, read, modify and delete data. A report is a structured presentation of database data. Queries allow the users to answer questions from the data. Menus are structured presentations of allowed user actions.

Q.24. List and describe some of the problems of the traditional file environment.

Following are some problems with traditional file environment:

1. Data redundancy is the presence of duplicate data in multiple data files. In this situation, confusion results because the data can have different meanings in different files.

2. Program-data dependence is the tight relationship between data stored in files and specific programs required. This dependency is very inefficient, resulting in the need to make changes in many programs when a common piece of data changes.

3. Lack of flexibility means that it is very difficult to create new reports from data when needed.

4. Poor security results from the lack of control over the data because the data are so widespread.

5. Data sharing is virtually impossible because it is distributed in so many different files around the organization.

Q.25. List and describe each of the components in the data hierarchy.

The data hierarchy includes bits, bytes, fields, records, files and databases. Data are organized in a hierarchy that starts with bit. A bit is represented by either a 0 (off) or a 1 (on). Bits can be grouped to form a byte to represent one character, number or symbol. Bytes can be grouped to form a field such as a name or date. Related fields can be grouped to form a record. Related records can be collected to form files. Related files can be organized into a database.

Q.26. Name the key personnel involved with databases and briefly describe their roles

The database administrator has both technical and managerial responsibilities over the database resource. Database programmers are required to create efficient data processing computer code. The end users have a major impact on database design, use and efficiency.

Multiple Choice

Short Questions

Q.1. What is three-level architecture?

Three-level architecture is the basis of modern database architecture. Database can be viewed at three levels. The three levels are depicted by three models known as three-level schema. The purpopse of three-level architecture is to separate the way the database is physically repsesented from the way the user thinks about it.

Q.2. Name and explain the purpose of three schemas in ANSI/SPARC three schema model.

External Schema: The external schema or user view is a representation of how users view the database. An external schema portrays just a portion of the database.

Conceptual Schema: A conceptual schema is a complete logical view of database that contains a description of all data and relationships in database. The conceptual schema is logical. It is independent of any particular means of storing data.

Internal Schema: An internal schema is a representation of a conceptual schema as physically stored using a particular product and/or technique. The description of a set of tables, keys, foreign keys, indexes and other physical structures is an internal schema.

Q.3. How are external schemas used to devise one conceptual schema?

External schemas are descriptions of the world as end users see the world. During the database design process, designers collect information from end users and develop different external schemas for different end users. These external schemas are then combined into one conceptual schema.

Q.4. What is hierarchical database? Write down the advantages and disadvantages of Hierarchical database model.

Hierarchical database organize data in a series like a family tree or organizational chart. The hierarchical database has branches made up of parent and child records. Each parent record can have multiple child records but a child record can have only one parent.

Advantages

1. Hierarchical DBMS is good for one-to-many relationships.
2. It can store large numbers of segments and process information efficiently.
3. It is a simple structure and minimizes disk input and output.

Disadvantages

1. It is less user-friendly.
2. It creates inflexibility and programming complexity.
3. It is difficult ot reorganize the database as hierarchy has to be maintained.

Q.5. What is a data model? List main types of data models.

A collection of concepts to describe and manipulate data, relationships between data and constraints on data is called data model. Object-based data models such as the Entity–Relationship model. Record-based data models such as the relational data model, network data model, and hierarchical data model. Physical data models describe how data is stored in the computer .

Q.6. What is relational database? List its advantages and disadvantages.

Relation database organizes data in two-dimensional tables called relation. Each row in a table is called tuple and each column is called an attribute. Tuple is also called record or row. The ranges of values that an attribute can have is called domain.

Advantages

1. It has the capacity to link multiple files together.
2. It provides the facility of data independence.
3. It provides the facility of structural independence.

4. It provides the facility of data integrity.
5. It has the capability to add new fields.
6. It provides the facility to establish relationship at any time.

Disadvantages

1. It is a bit poor in processing efficiency.
2. Response time can be very slow if large numbers of accesses to data are required to select, join, and extract data from tables.

Q.7. Describe the role and responsibilities of a database administrator (DBA).

1. The development and management of the organization's databases.
2. He must work with a number of people including systems analysts, programmers, users, managers, and security personnel.
3. He assists in design and implementation of database.
4. He helps to establish policies and procedures
5. He assists to ensure security of the database.

Q.8. What is difference between Data Administrator and Database Administrator.

Data administrator is responsible for corporate data resource. It includes non-computerized data. He manages shared data of users or application areas of an organization. He determines long-term goals and enforces standards, policies and procedures. He also determines data requirements and develops conceptual and logical desing. His work is DBMS-independent.

Database adminstrator is more technically oriented. He requires the knowledge of a DBMS. He develops and maintains systems using DBMS. He executes plans to achieve goals. He enforces standards, policies and procedures developed by DA. He develops logical and physical desing. His work is DBMS-dependent.

Q.9. What is a data dictionary?

Data dictionary is a repository of information that describes the logical structure of the database. It contains record types, data item types and data aggregates etc. The data dictionaries in some systems store database schema and can be used to create and process the database. Data dictionary contains metadata. Metadata is the data about the data stored in the database.

Q.10. List some uses of data dictionary.

1. It is used to collect and store information about data in a central location.
2. It provides great help in communication as it stores exact meanings of data items.
3. It keeps track of changes to the database structure.
4. It records each item, its relationships and users. DBA can see the effects of a change.
5. It stores all information about different authorized users.
6. It can also keep record of each access to the database to be used for audit purposes.

Q.11. Describe different types of data dictionaries.

Different types data dictionaries are as follows:

1. Integrated Data Dictionary: A data dictionary that is part of DBMS is called integrated data dictionary. They perform many functions throughout the life of the database not only in design phase

2. Freestanding Data Dictionary: A data dictionary that is available without a particular DBMS, it is called freestanding data dictionary. It can be a commercial product or a simple file developed by the designer. Many CASE packages provide a data dictionary tool. It is preferable in initial design stages before choosing any particular DBMS.

Q.12. Differentiate between active and passive data dictionary.

The integrated data dictionary is called active if it is checked by DBMS every time a database is accessed. It is always consistent with actual database structure. It is automatically maintained by the system. The integrated data dictionary is called passive if it is not used in day-to-day database processing.

Q.13. What are the effectiveness measures of database design?

The major effectiveness measures of database design are as follows:

a. Ensuring that data can be shared among users for a variety of applications.

b. Maintaining data that are both accurate and consistent.

c. Ensuring all data required for current and future applications will be readily available.

d. Allowing the database to evolve and the needs of the user to grow.

e. Allowing users to construct personal view of data without concerning the way it is physically stored.

Q.14. What elements of a data flow diagram should be analyzed as part of data modeling?

Data stores, data flows and even processes provide information for data modeling. A data store often represents one or more data entities and their associated attributes. All data in data flows must either be stored in some entity, be computed from data in entities or pass through the system. The description of a process can indicate a business rules that must be represented in the data model

Short Questions

Q.1. Define entity and give an example.

An entity is a person, place, thing or event for which data is collected and maintained. For example, a library system may contain data about different entities like BOOK and MEMBER. A college system may include entities like STUDENT, TEACHER, and CLASS etc.

Q.2. Why is it important to define the meaning and use of entities?

It is very important to define the meanings and use of entities. It is possible that different users use different names for the same entity. Suppose a person refers to a purchase. It is not clear that the purchase is from the firm by a customer or purchase from a supplier by the firm. Defining the meaning and use of entities makes it easy to determine if they are the same or different.

Q.3. Explain the difference between an entity class and an entity instance.

A group of entities of the same type is called entity class. All entities in an entity type share common characteristics. For example, STUDENT entity class is a collection all students. A member of an entity class is also known as an entity instance. For example, a student Abdullah of STUDENT entity type is an entity instance.

Q.4. Explain the difference between attributes and identifiers.

The characteristics of an entity are called **attributes** or **properties**. For example, Name, Address, Class and Email of a student are his attributes. Identifier is an attribute that identifies an entity instance among other instances in entity class.

Q.5. What is ER Model? What are its main building modules?

Entity-Relationship model is a logical representation of data in organization. It views the entire system as a collection of entities related to one another. It is used to describe elements of a system and their relationships. The main building modules of the ER Model are Entity, Attribute and Relationship.

Q.6. What is a domain?

A domain is a named set of values that an attribute can have. A domain can consist of a specific list of values. It can also be defined more generally like a set of strings of maximum length 50.

Q.7. Explain two ways that domains reduce ambiguity.

Domains can reduce ambiguity when two attributes have the same name but represent different things. For example, different entities may have an attribute Tax. If it represents Sales Tax for one attribute and Income Tax for other attribute, domains can be defined to distinguish between the two. The named domains eliminate ambiguity from attributes whose values look similar but are not same.

Q.8. How domains are practically useful.

One way domains become useful is to assign like attributes to the same domain. For example, if a domain is created for Dates and it used a format of MM-DD-YY, all dates assigned to that domain are forced to use that format. If the formats for dates need four-digit year, it can be changed and all dates will now use the new format. Another practical use for domains is to assess whether two attributes that are named differently are referring to the same thing. They would be if they have the same domain.

Q.9. Explain what a composite identifier is and give an example.

An identifier that consists of composite attribute is called **composite identifier**. For example, OrderID identifier may consist of OrderNo and Date.

Q.10. What do you know about simple and Composite Attribute. Give example of each.

An attribute that cannot be subdivided into smaller component is known as **simple attribute**. For example, a person can have only one gender and one date of birth. An attribute that can be subdivided into smaller components is called **composite attribute**. For example, Address is an example composite attribute. It can be subdivided into Street, City, Country.

Q.11. What is difference between single-valued and multi-valued attribute. Give example.

An attribute that may contain single value is called **single-valued attribute**. For example, Age of a person is single-valued attribute. An attribute that may contain two or more values is called **multi-valued attribute**. For example, a person can have two or more college degrees.

Q.12. Give an example of an entity that has no obvious identifier.

A very simple example is a house or address. Initially you may think that the phone number of a house would be an identifier but some houses may not have phones and all instances of the entity must have an identifier. Next you may think house number and street name but what about rural areas. These areas typically have Route Numbers but the houses do not have house numbers. Then you could use Route Numbers and P.O. Box but houses in the city do not have P.O. Boxes.

Q.13. Define relationship and give an example.

A **relationship** is a logical connection between different entities. The entities that participate in a relationship are called **participants**. The relationship may be between different entities or between an entity and itself. A relationship is established on the basis of interaction among entities. For example, a relationship exists between a STUDENT and TEACHER because the teacher *teaches* the students.

Q.14. Define degree of relationship. Give example of a relationship greater than degree 2.

The number of entities in a relationship is called **degree** of relationship. Assume entity classes FACULTY, STUDENT and MAJOR. Assume FACULTY is assigned to advise STUDENT for a given MAJOR.

Q.15. Explain the difference between a relationship class and a relationship instance.

A relationship class is an association among entity classes. A relationship instance is an association among entity instances.

Q.16. Define the terms maximum cardinality and minimum cardinality.

The minimum number of instances of one entity that may be associated with each instance of another entity is known as **minimum cardinality**. The maximum number of instances of one entity that may be associated with each instance of another entity is known as **maximum cardinality**.

Q.17. Define the term weak entity and give an example.

An entity that can exist only if another entity exists is known as weak entity. It means that weak entities depend upon the existence of another entity. Suppose we want to store the data of a student after assigning a class to him. It means that the data cannot be stored if CLASS entity does not exist. In order to store the record of the student, we first need to create an entity that represents a class. Here, STUDENT is a weak entity because it depends upon CLASS entity.

Q.18. Differentiate between single-valued attributes and simple attributes with example.

A single-valued attribute is an attribute that can have only one value. For example, a person has only one first name and only one social security number. A simple attribute is an attribute that cannot be decomposed. For example, a person's gender can be either M or F. It cannot be decomposed. Single-valued attributes are not necessarily simple. For example, an inventory code HWPR145 may refer to a classification scheme. HW indicates HardWare, PR indicates Printer, and 145 indicates an inventory number. It can be decomposed into its component parts even if it is single-valued.

Q.19. What cardinalities indicate functional, optional and mandatory relationships?

A maximum cardinality of one indicates a functional relationship. A minimum cardinality of zero indicates an optional relationship. A minimum cardinality of one or more indicates a mandatory relationship.

Q.20. What is the difference between an existence-dependent and a weak entity type?

A weak entity is a specialized kind of existence-dependent entity. A weak entity has mandatory relationship like an existent-dependent entity. In addition, weak entities borrow part or their entire primary key.

Q.21. When should an ERD contain weak entities?

An ERD contains weak entity types when entities are closely associated with other entities. Identification dependency often occurs when entities are physically located inside another entity such as rooms inside a building.

Q.22. Describe subtype entities and give an example.

A subtype entity is an entity that represents a special case of another entity called its supertype. For example, the entity STUDENT can have subtypes of GRADUATE_STUDENT and UNDERGRADUATE_STUDENT. The entity BUILDING can have subtypes of CLASSROOM, OFFICE, or RECREATIONAL etc.

Q.23. Differentiate between a HAS-A and an IS-A relationship with example.

The relationship between a supertype and its subtypes is also called an IS-A relationship. Entities with an IS-A relationship should have the same identifier as they represent different aspects of the same thing. Entities with an HAS-A relationships represent aspects of different things. They have different identifiers. These relationships do not involve subtypes.

BUILDING with an identifier of Building Code and subtypes of CLASSROOM, OFFICE and RECREATIONAL have an IS-A relationship. All types of building are identified by building code. All subtypes IS-A BUILDING. A relationship between ADVISOR and STUDENT is a HAS-A relationship because a STUDENT HAS-A ADVISOR, ADVSIOR is not a type of STUDENT.

Q.24.Constract the following terms:

a. Entity type and relationship type b. Degree and cardinality

a. An entity type is a collection of entities that share common properties or characteristics. A relationship type is a meaningful association between entity types.

b. The degree (of a relationship) is the number of entity types that participate in that relationship. Cardinality is a constraint on the number of instances of one entity that can (or must) be associated with each instance of another entity.

Q.25. Name the symbols used in E-R model for (a) entity, (b) relationship, (c) weak entity and its relationship, (d) recursive relationship and (e) subtype entity.

(a) Rectangle (b) Diamond (c) Double-lined rectangle and double-lined diamond
(d) Line with diamond back to entity (e) Entity with existence symbol

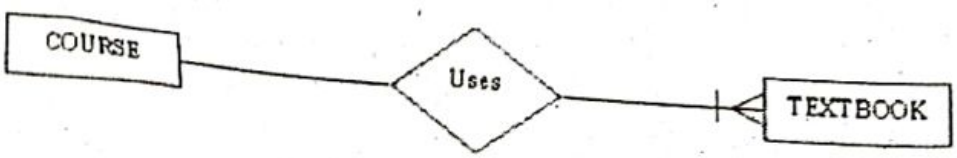
Q.26. List four types of Cardinality constraint and draw an example of each.

a. Optional one:

b. Mandatory one:

c. Optional many

d. Mandatory many:



Q.27. For each of the following, indicate whether there is a one-to-many or a many-to-many relationship. Also draw a diagram of the following relationship.

a. STUDENT and COURSE (student register for course)
b. BOOK and BOOK COPY (book have copies)
c. COURSE and SECTION(courses have section)
d. SECTION and ROOM(sections are scheduled in rooms)
e. INSTRUCTOR and COURSE

a. Many-to-many:

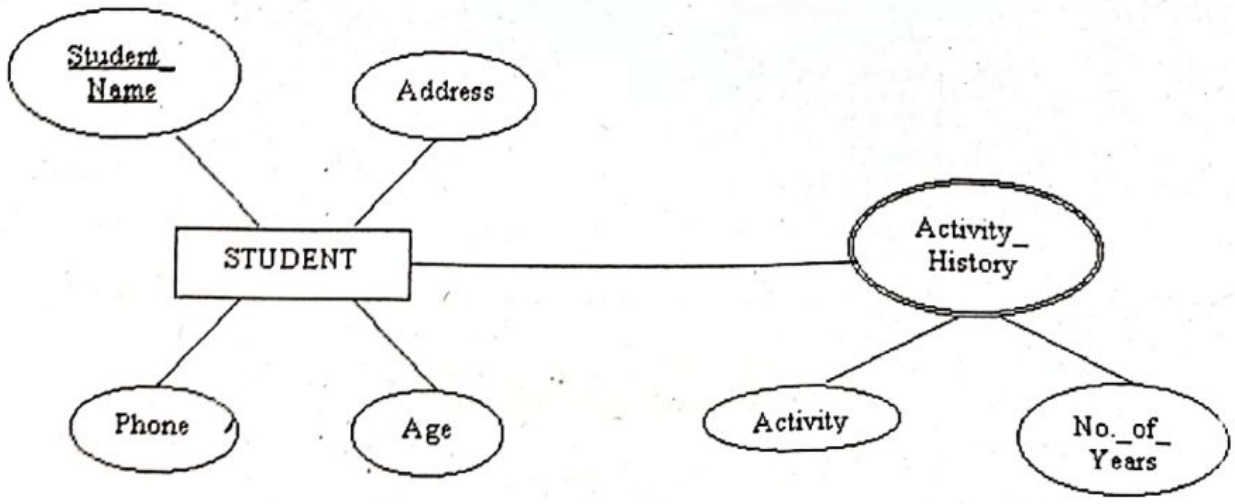
b. One-to-many:

c One-to-many:

d. One-to-many:

e. Many-to-many:

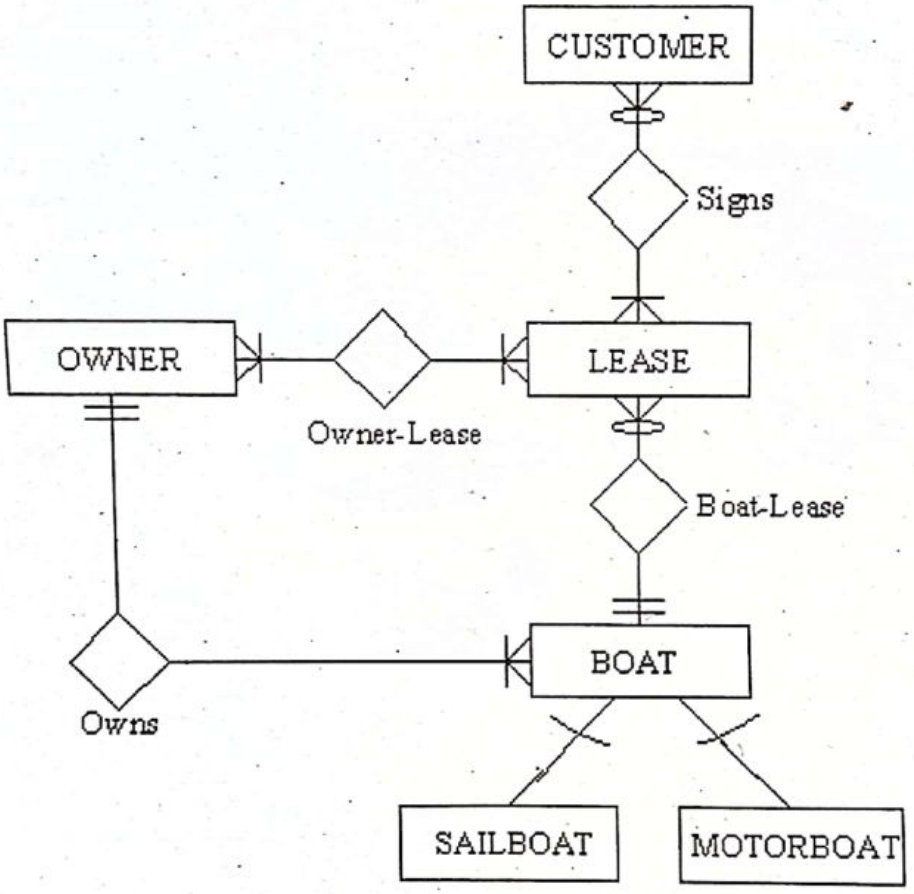
Q.28. The entity type STUDENT has attributes Student_name, address, phone, age, activity and no_of_years. Activity represents campus-based student activity while no_of_years represents the number of years student has engaged in this activity. A given student may engage in more than one activity. Draw an E-R Diagram for this situation.



Notes:

1. Assume that Student_Name is the identifier.
2. Age cannot be calculated without date-of-birth

Q.27. Examine the following entity-relationship diagram and answer the questions that follow. The arcs imply the disjoint sub type/super type relation.



a. List one example of a one-to-many relationship.

A one-to-many relationship exists between OWNER and BOAT.

b. What is the degree of relationship between CUSTOMER and LEASE?

Binary, many to many relationship

c. What is the minimum cardinality of the relationship between CUSTOMER and LEASE?

Zero (for CUSTOMER) and 1 (for LEASE).

d. According to the figure, is it required for all boats to be owned by an owner?

Yes. It is indicated by a minimum cardinality of 1 on the OWNER side in the OWNER-BOAT relationship. It means that each owner should own at least one boat.

e. What is the maximum number of boats that can be owned by an owner?

Many (N).

Short Questions

Q.1. Explain why the E-R model and the semantic object model are like lenses.

These models are tools for representing and expressing the views of users' data structures. They shape the image of the representation that is seen and documented.

Q.2. Define semantic object.

A semantic object is a thing that can be identified in the user's working environment. Semantic means "meaning". Semantic objects are used to represent or model the meaning of user's data. For example, different objects in a university are students, teachers and departments etc.

Q.3. Differentiate between object class name and object instance name with example.

An object class name refers to the whole class or type. An object entity name refers to a member of a particular class. For example, STUDENT is an object class name. Ali can be an object instance name.

Q.4. What is required for a set of attributes to be a sufficient description?

The properties represent all of the characteristics that the users need to perform their work.

Q.5. Explain the words distinct identity as they pertain to definition of a semantic object.

They mean something that users recognize as independent and separate. That thing stands on it own in the users' minds. Each instance of an object is unique and identifiable in its own right. Users have names for them such as Order-number or Employee-number.

Q.6. List the three types of attributes

Three types of attributes are Simple attributes, group attributes and semantic object attributes.

Q.7. Give an example of each of the following:

In the semantic object APARTMENT:

a. A *simple, single-value attribute:*	NumberOfBedrooms
b. A *group, single-value attribute:*	ApartmentName as (BuildingName, ApartmentNumber)
c. A *simple, multi-value attribute:*	Phone jack location (one value for each phone jack in the apartment)
d. A *group, multi-value attribute:*	Occupant (FirstName, LastName), where more than one person can live in an apartment.
e. A *simple object attribute:*	BUILDING
f. *A multi-value object attribute:*	REPAIR

Q.8. What is minimum cardinality? How is it used? Which types of attributes have minimum cardinality?

The minimum number of values of an attribute that are required is called minimum cardinality. No object is allowed to exist for which this number is not satisfied. All types have minimum cardinality.

Q.9. What is maximum cardinality? How is it used? Which types of attributes have maximum cardinality?

The maximum number of values an attribute can have in a semantic object. No object is allowed to exist that has more than this number. All types have maximum cardinality.

Q.10. What are paired attributes? Why are they needed?

Semantic object attributes are always paired. If OBJECT1 contains an attribute OBJECT2, then OBJECT2 will always contain an attribute OBJECT1. A one-way relationship is logically impossible.

Q.11. What is an object identifier? Give an example of a simple attribute object identifier and an example of a group attribute object identifier.

An object identifier is an attribute or collection of attributes that are used to identify an object instance. In AUTO, LicenseNumber is a simple attribute identifier. In APARTMENT, (BuildingName, ApartmentNumber) is a composite identifier

Q.12. Define attribute domain. What are the types of attribute domain? Why is a semantic description necessary?

A collection of possible values of an attribute is known as attribute domain. The domain of a simple attribute consists of two types of descriptions:

1. Physical Description

The physical description indicates the type of data. For example, Name of a student should be a string value and Roll No should not exceed 99 etc.

2. Semantic Description

The semantic description indicates the function or purpose of attribute. For example, Name of a student should be a valid name. It is a semantic description of Name attribute. This restriction is related to the meaning of the attribute. For example, "Math" is not a valid name even if it consists of string value because it does not satisfy the semantic description of Name attribute.

Q.13. What is a semantic object view? Give an example of an object and two views other than those in this text.

A semantic object view is a subset of a semantic object. Consider the object APARTMENT with attributes BuildingName, ApartmentNumber, RentAmount, LastOccupiedDate. One view might have the first three attributes and another might have all four attributes.

Q.14. Give an example of a simple object

SOFTWARE with properties Name, Type, Price, MemoryRequired.

Q.15. Give three examples of composite objects. One example should have one multi-value simple attribute; one should have two independent multi-value groups and third should have nested multi-value groups.

a. EMPLOYEE with properties Emp#, {ReviewDate, ReviewComments}$_{0,N}$.

b. EMPLOYEE with properties Emp#, {ReviewDate, ReviewComments}$_{0,N}$, and {SalaryRevisionDate, Salary}$_{0,N}$.

c. EMPLOYEE with properties Emp#, {ReviewDate, {ReviewerName, ReviewerComments}$_{0,N}$}$_{0,N}$. For last example, there are multiple reviews on a given date.

Q.16. Give an example of four sets of compound objects. One set should have a 1:1 relationship, one set should have a 1:N relationship, one set should have an M:1 relationship and one set should have an M:N relationship.

1:1	COMPUTER contains EMPLOYEE
	EMPLOYEE contains COMPUTER
1:N	PROJECT contains COMPUTERs
	COMPUTER contains only one PROJECT
M:1	(really same as 1:N, which is the point of this part of the question)
N:M	COMPUTER contains SOFTWARE-PACKAGEs
	SOFTWARE-PACKAGE contains COMPUTERs

Q.17. Give an example of a hybrid object.

COMPUTER contains the multi-valued group {SOFTWARE-PACKAGE, Price} 0.N, where SOFTWARE-PACKAGE is an object and Price is not.

Q.18. Give an example of one association and two compound objects.

JOB contains a single value of the object properties ARTIST and CUSTOMER.

CUSTOMER contains many values of JOB

ARTIST contains many values of JOB

Q.19. Give an example of a supertype object with three subtype objects.

SOFTWARE contains the properties Name, Price, Vendor and the subtype objects WORD-PROCESSING-SOFTWARE, SPREADSHEET-SOFTWARE and DBMS-SOFTWARE. WORD-PROCESSING-SOFTWARE contains the properties Name, Fonts0.N. SPREADSHEET-SOFTWARE contains the properties Name, GraphStyles0.N. DBMS-SOFTWARE contains the properties Name, MaxTables, MaxColumns, MaxRows.

Q.20. Explain the similarities between E-R model and the semantic object model.

Both are tools for understanding and documenting users' data models. Both are concerned with representing the things that are important to the users and the relationships among those things.

Q.21. Explain the major differences between the E-R model and the semantic object model.

E-R model takes entity as the basic element of interest to the user. Semantic object model takes semantic object as the basic element. Semantic objects show relationships in context of objects containing other objects. E-R model shows relationships as between entities. Usually, an entity is a different name for a table. Entities normally do not have composite attributes nor do they have multi-valued attributes. Entities do not contain other entities.

Short Questions

Q.1. Contrast the following:

a. Candidate key and primary key
b. Functional dependency and transitive dependency
c. Foreign key and primary key

a. **Candidate Key and Primary Key:** A primary key is an attribute or combination of attributes that uniquely identifies a row in a relation. When a relation has more than one such attribute or combination of attributes, each is called a candidate key.

b. **Functional Dependency and Transitive Dependency:** A functional dependency is a relationship between any two attributes or two sets of attributes. A transitive dependency is a functional dependency between two or more non-key attributes.

c. **Foreign Key and Primary Key:** A primary key uniquely identifies each row in a relation. a foreign key is an attribute or set of attributes in a relation that reference a primary key in another table.

Q.2. What is the difference between a candidate key and a superkey?

A superkey may not be minimal. A candidate key is a minimal superkey. A superkey is minimal if it does not remain unique by removing a column.

Q.3. What is the relationship between the referential integrity rule and foreign keys?

Referential integrity involves the values that can be used in foreign keys. A foreign key can store a value matching a candidate key value in some row of the table containing the associated candidate key or a null value.

Q.4. What is a primary key? Are duplicate primary keys allowed? Why or why not?

A primary key is a field or set of fields that can uniquely identify a row of a table. Duplicate key values are not allowed because primary keys must uniquely identify a row.

Q.5. What is a foreign key? Why are foreign keys used in a relational database? Are duplicate foreign key values allowed? Why or why not?

A foreign key is a field or set of fields stored in one table that also exists as a primary key value in another table. Foreign keys are used to represent relationships among entities that are represented as tables. Duplicate foreign keys are not allowed within the same table because they would redundantly represent the same relationship. Duplicate foreign keys may exist in different tables because they would represent different relationships.

Q.6. What are the difference between partial dependency and transitive dependency?

A partial dependency exists when an attribute is dependent on only a part of primary key. It is associated with 1NF. Transitive dependency is a condition in which an attribute is dependent on another attribute that is not part of primary key. It usually requires the decomposition of the table containing the transitive dependency.

Q.7. What restrictions must be placed on a table to consider it a relation?

- Rows contain data about an entity
- Columns contain data about attributes of the entity
- Cells of the table hold a single value
- All entries in a column are of the same kind
- Each column has a unique name
- The order of the columns in unimportant
- The order of the rows in unimportant
- No two rows may be identical

Q.8. Define relation, tuple, attribute, file, record, field, table, row and column.

- A relation is a two-dimensional table that has the characteristics.
- A tuple is one row of a table.
- An attribute is one column in a table.
- A file is a term use by many people to a table.
- A record is the same as row of a table
- A field is the same as a column of a table.
- A table is a relation. Generally these terms are interchangeable.
- A row contains all data about one instance of an entity represented in a table.
- A column contains all values assigned to an attribute in a table.

Q.9. What is a view? Discuss the difference between a view and a base relation.

View is the dynamic result of one or more relational operations operating on the base relations to produce another relation. Base relation exists as a set of data in database. A view does not contain any data. It is defined as a query on one or more base relations. The query on view is translated into a query on the associated base relations.

Q.10. Describe the purpose of normalizing data.

The process of producing a simpler and more reliable database structure is called **normalization**. It is used to create a suitable set of relations for storing data. The process of normalization identifies and corrects the problems and complexities of database design. It produces a new set of relations. The new design is as free of processing problems as possible.

Q.11. How is functional dependency associated with the process of normalization?

Normalization is a technique to analyze relations based on primary key and functional dependencies. Normalization is often performed as a series of tests on a relation to determine whether it satisfies or violates the requirements of a given normal form. Three normal forms were initially proposed which are called 1NF, 2NF and 3NF. All of these normal forms are based on the functional dependencies among the attributes of a relation.

Q.12. What is well-structured Relation? Why are well-structured relations important in logical database design?

A well-structured relation is a relation that contains a minimum amount of redundancy. It allows users to insert, modify and delete rows in a table without errors or inconsistency. Well-structured relations are important because they promote database integrity.

Q.13. Which three data anomalies are likely to be the result of data redundancy? How can such anomalies be eliminated?

The most common anomalies considered when data redundancy exists are update anomalies, addition anomalies and deletion anomalies. All these can easily be avoided through normalization. Data redundancy produces data integrity problems.

Q.14. What is a deletion anomaly? Give an example.

A deletion anomaly is a case where deletion of facts about one entity instance deletes facts about another entity instance. The user can lose facts about two entities with one deletion.

Q.15. What is an insertion anomaly? Give an example.

An insertion anomaly is a case where the user cannot insert a fact about one entity until he has an additional fact about another entity.

Q.16. What is the cause of modification anomalies?

Poor database design causes the modification anomaly. A good database design avoids modification anomalies by eliminating excessive redundancies.

Q.17. Describe the characteristics of an unnormalized table. How such table is converted to first normal form (1NF)?

A table in unnormalized form contains one or more repeating groups. It is converted to first normal form by removing the repeating group to a new relation along with a copy of original key attribute(s). The repeating group can be removed by entering appropriate data in the empty columns of rows containing the repeating data.

Q.18. What is a normal form?

A normal form is a rule about allowable dependencies. Each normal form removes certain kinds of redundancies.

Q.19. What does 1NF prohibit?

1NF prohibits repeating groups in tables. A table not in 1NF is unnormalized or non-normalized.

Q.20. What is a key column?

A column is a key column if it is a candidate key or part of a candidate key.

Q.21. What is a non-key column?

A column is a non-key column if it is not a key column.

Q.22. Define functional dependency. Give an example of two attributes with functional dependency and give an example of two attributes with no functional dependency.

A functional dependency is a relationship between attributes such that given the value of one attribute it is possible to determine the value of the other attribute. Example of functional dependency: Name--->Phone#. Example of attributes that are not functionally dependent: Age and Address.

Q.23. Which normal forms rely on the definition of functional dependency?

Second and third normal forms are based on a concept called functional dependency—a one-to-one correspondence between two field values. Second normal form ensures that every field in a table is functionally dependent on the primary key. Third normal form ensures that no non-key field is functionally dependent on any other non-key field.

Q.24. What is a view? How it is related to data independence?

A view is a virtual table that does not really exist in its own. It is derived from one or more base table. There is no stored file that represents the view. A definition of view is stored in data dictionary. The view can insulate users from the effects of restructuring and growth in database. It provides logical data independence. .

Q.25. Define determinant.

A determinant is an attribute whose value enables us to obtain the value(s) of other related attributes. It appears on the left side of a functional dependency. Thus, in A--->B, the determinant is A.

Q.26. What is removed when a relation is converted to the first normal form?

All repeating groups are removed and the primary keys are identified when a relation is converted to the first normal form.

Q.27. What is removed when a relation is converted from 1NF to 2NF?

All the partially dependent attributes are removed and placed in another relation when a relation is converted from 1NF to 2NF.

Q.28. What is difference between normal form and normalization?

Normal form is a state of a particular relation regarding functional dependencies while the normalization is a process of producing a simpler and more reliable database structure.

Q.29. What kinds of functional dependencies are not allowed in 2NF?

Functional dependencies in which part of a key determines a nonkey column are not allowed in 2NF.

Q.30. List three conditions that can be applied to determine that a relation in first normal form is also in Second Normal Form.

Three conditions that imply a relation is in second normal form:

1. The primary key consists of a simple attribute.
2. No non-key attributes exist in the relation.
3. Every non-key attribute is functionally dependent on the full set of primary key attributes.

Q.31. What is removed when a relation is converted from 2NF to 3NF?

Any transitive dependencies, nonkey attributes dependent on other nonkey attributes, are removed when a relation is converted from 2NF to 3NF.

Q.32. What kinds of functional dependencies are not allowed in 3NF?

Functional dependencies in which a nonkey column determine another nonkey column are not allowed in 3NF.

Q.33. What kinds of functional dependencies are not allowed in BCNF?

Functional dependencies in which the determinant is not a candidate key are not allowed in BCNF..

Q.34. Define second normal form. Give an example of a relation in 1NF but not in 2NF. Transform the relation into relations in 2NF.

A relation is in second normal form if all its non-key attributes are dependent on the entire key. A table that is in second normal form cannot have a partial dependency.

Assume: COURSE (Dept_Code, Course_Number, Course_Title, Credits, Department_Name)

The table is in first normal form because it has no repeating attributes. Since the key is composite, it is not in second normal form if Dept_Code → Department_Name. Only part of primary key determines an attribute. To put the table in second normal form use the following tables.

COURSE (Dept_Code, Course_Number, Course_Title, Credits)

DEPARTMENT (Dept_Code, Department_Name)

Q.35. Define third normal form. Give an example of a relation in 2NF but not in 3NF. Transform the relation into relations in 3NF.

A relation is in third normal form if it is in second normal form and has no transitive dependencies. All determinants must be keys.

Assume: STUDENT (SIDU, Stu_Name, P_O_Box, Major, Hours_Required)

The table is in first normal form because it has no repeating attributes. The table is in second normal form because it has a single attribute key. If you assume Major → Hours_Required them Major is a determinate but not the key therefore you have a transitive dependency. To put the table in second normal form use the following tables.

STUDENT (SID, Stu_Name, P_O_Box, Major)

DEPARTMENT (Major, Hours_Required)

Q.36. What are the special cases covered by BCNF but not by 3NF?

BCNF covers two special cases not covered by 3NF: (1) part of a key determines part of a key and (2) a nonkey column determines part of a key.

Q.37. The special cases covered by BCNF but not by 3NF are significant?

The special cases are not significant because they rarely occur.

Q.38. Is a relation with no non-key attribute is always in third normal form?

This is a special case of 3NF. A 3NF relation typically involves a functional dependency between two non-key attributes, and if no such non-key attributes are present then the relation must be 3NF.

Q.39. Can foreign keys contain null values? Explain why via an example.

Yes foreign keys should be allowed to have null values because there are situations in which such kind of information is not available as follows:

- Employees that do not belong to a specific department, yet;
- Products that have not been put on the shelves, yet;
- Students that have not chosen some options, yet;

Q.40. Define BCNF. Give an example of a relation in 3NF but not in BCNF. Transform the relation into relations in BCNF.

A relation is in BCNF if every determinant is a candidate key. Consider this relation:

FAC-OFFICE (FID, Department, Building, Office).

Assume faculty members in the same department have offices in same building. The key is FID that determines Department, Building and Office. Department is not a candidate key and it determines Building. These relations are in BCNF:

FACULTY (FID, Department, Office)

DEPARTMENT-LOCATIONS (Department, Building)

Q.41. Define fourth normal form. Give an example of a relation in BCNF but not in 4NF. Transform the relation into relations in 4NF.

A relation is in 4NF if every determinant is a candidate key and it has no multi-valued dependencies. The following relation is in BCNF but not 4NF:

EMPLOYEE-HISTORY (Name, Project, PublicServiceActivity).

Project can be multi-valued because an employee could have worked on many projects. PublicServiceActivity can also be multi-valued. But Project and PublicServiceActivity are unrelated. These relations are in 4NF:

EMPLOYEE-WORK-HIST (Name, Project).

EMPLOYEE-SERVICE-HIST (Name, PublicServiceActivity)

Q.42. What is the relationship between BCNF and 3NF? Is BCNF a stricter normal form than 3NF? Briefly explain your answer.

Ans: BCNF is the revised 3NF definition. Yes BCNF is stricter than 3NF in that every table in BCNF is in 3NF but not every table in 3NF is in BCNF.

Q.43. Why is the BCNF definition preferred to the 3NF definition?

Ans: BCNF is the preferred definition because it is a simpler definition and provides the basis for the simple synthesis algorithm.

Q.44. How many columns does an MVD involve?

An MVD involves three columns.

Q.45. What is a multivalued dependency (MVD)?

A multivalued dependency is a relationship that can be derived from other relationships.

Q.46. What is the relationship between MVDs and FDs?

MVDs are generalizations of FDs. Every FD is an MVD, but not every MVD is an FD.

Q.47. What is minimal normal form a relation must satisfy? Define this normal form.

The minimal normal form is 1NF. A relation in which the intersection of each row and column contains one and only one value.

Q.48. What is meant by 'union compatible'? Which operations require tables to be union compatible?

The tables are 'union compatible' if they have the same degree (number of attributes) and the domain of the corresponding attributes is same. Union, intersection and difference operations require tables to be union compatible.

Q.49. Why is it important to study the operators of relational algebra?

Knowledge of operators can improve query formulation skills. The operators are fundamental ways to retrieve data from a relational database. The study of operators of relational algebra is important to become proficient in query formulation.

Q.50. Why are the restrict and project operators widely used?

These operators are widely used because users often want to see a subset rather than an entire table. These operators are also popular because they are easy to understand.

Q.51. Why is the join operator so important for retrieving useful information?

It is important for retrieving useful information because information resides in different tables and combining tables is important. The join operator is used to combine tables to retrieve information.

Q.52. What happens to unmatched rows with the join operator?

Unmatched rows are removed from the result of the join operator. Unmatched rows are included in the result of the full outer join operator.

Q.53. State the difference between the following:

a. Union & Intersection
b. Product & join
c. Selection and projection

a. The **Union** operation on two tables returns all rows from both the tables, but it does not repeat the duplicate rows. The Intersection operation on two table returns rows that exist in both tables involved.

b. A **product** of two table matches every row from the first table to each and every row in the second table. The **join** operation on two tables joins rows from two tables based on a common attribute, and the rows with same common attribute value are joined to each other.

c. The **Selection** operation on a table returns rows that satisfy the supplied criteria. The projection operation returns the attributes specified from a table.

Q.54. What is the closure property of the relational algebra?

The operators of the relational algebra are closed over relations, in the same sense as that in which those of arithmetic are closed over numbers. In arithmetic, plus, minus, times and divide each operates on a pair of numbers and when invoked yields a number. It is this so-called closure property that allows us to use an invocation of an operator as an operand of another invocation. By repeatedly doing that we can build numerical expressions of arbitrary complexity. In relational algebra, each operator operates on one or more relations and when invoked yields a relation, thus allowing relational expressions of arbitrary complexity to be written.

Q.55. An organization works on a number of Projects, each identified by ProjectNo. There are number of employees identified by SIN who work on these projects. An employee can be working on more than one project at one time. The number of hours worked are also recorded. A preliminary table has been constructed as follows

SIN	ProjectNo	EmployeeName	ProjectName	EmpHours

Assuming the relation to be in 1NF,

a. What should be the Primary Key? Is it Composite?
b. Write down the Partial dependencies.

c. Split the relation such that the resulting relations are in 3NF.

a) **(SIN, ProjectNo)**

b) **SIN → EmpoyeeName, ProjectNo → ProjectName**

c)

SIN	ProjectNo	EmpHours

SIN	EmployeeName

ProjectNo	ProjectName

Q.56. A table contains sample data for parts and for vendors who supply parts. The part numbers uniquely identify parts and that vendor names uniquely identify vendors.

Part _No.	Description	Vendor_Name	Address	Unit_ Cost
1234	Logic chip	Ejaz	Peoples colony	10.00
		Naeem	Medina town	8.00
5678	Memory chip	Ali Raza	Peoples colony	3.00
		Anjum	Raza Abad	2.00
		Nasir	Saeed colony	5.00

a. Convert this table to a relation (named PART SUPPLIER) in first normal form. Illustrate the relation with the sample data in the table.

b. List the functional dependencies in PART SUPPLIER and identify a candidate key.

c. For the relation PART SUPPLIER, identify each of the following: an insert anomaly a delete anomaly and a modification anomaly.

d. Draw a relational schema for PART SUPPLIER and show the functional dependencies.

e. In what normal form is this relation?

a. PART SUPPLIER

Part_No	Description	Vendor_Name	Address	Unit_Cost
1234	Logic Chip	Ejaz	Peoples colony	10.00
1234	Logic Chip	Naeem	Medina town	8.00
5678	Memory Chip	Ali Raza	Peoples colony	3.00
5678	Memory Chip	Anjum	Raza Abad	2.00
5678	Memory Chip	Nasir	Medina town	5.00

b.

Part No → Description
Vendor Name → Address
Part No, Vendor Name → Unit_Cost

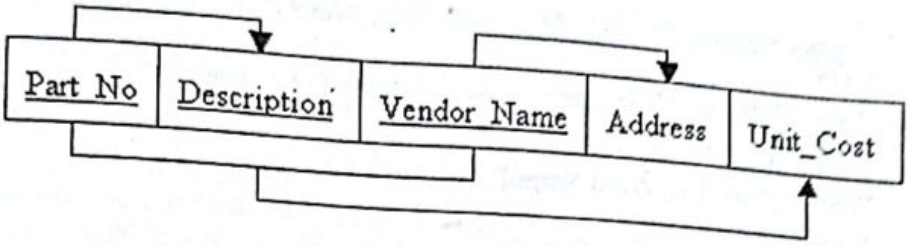
c.

Insert anomaly: It is not possible to insert a new vendor before including a part number.

Delete anomaly: If part information is deleted, information about a vendor who supplies that part is also lost.

Modification anomaly: If a vendor address changes, all records for that vendor has to be

d.



e. 1NF

Q.57. Examine the Patient Medication Form for Civil Hospital as follows:

Civil Hospital
Patient Medication Form
Patient Number: 9876
Full Name: Ali Ahmad **WardNumber: W11**
BedNumber: 87 **Ward Name: Fatima**

Drug Number	Name	Description	Dosage	Method of Admin	Units per Day	Start Date	Finish Date
10223	Morphine	Pain killer	10mg/ml	Oral	50	24/03/01	25/04/02
10334	Tetracycline	Antibiotic	0.5mg/ml	IV	10	24/03/01	17/04/01
10223	Morphine	Pain killer	10mg/ml	Oral	10	25/04/02	02/05/03

a. Identify the functional dependencies represented by the data shown in the form in
b. Describe and illustrate the process of normalizing the data shown in Figure to first (1NF), second (2NF), third (3NF), and BCNF
c. Identify the primary, alternate, and foreign keys in your BCNF relations

a)

Patient No → Full Name

Ward No → Ward Name

Drug No → Name, Description, Dosage, Method of Admin

Patient No, Drug No, Start Date → Units per Day, Finish date

Functional dependencies for Bed No are unclear. If Bed No is unique number for entire hospital then Bed No → Ward No. Bed No is concerned with allocation of patients on waiting list to beds.

b)

First Normal Form

Patient No, Drug No, Start Date, Full Name, Ward No, Ward Name, Bed No, Name, Description, Dosage, Method of Admin, Units per Day, Finish Date

Second Normal Form

Patient No, Drug No, Start Date, Ward No, Ward Name, Bed No, Units per Day, Finish Date

Drug No, Name, Description, Dosage, Method of Admin

Patient No, Full Name

Third Normal Form/BCNF

Patient No, Drug No, Start Date, Ward No, Bed No, Units per Day, Finish Date

Drug No, Name, Description, Dosage, Method of Admin

Patient No, Full Name

Ward No, Ward Name

c)

Patient No(FK), Drug No(FK), Start Date, Ward No(FK), Bed No, Units per Day, Finish Date

Drug No, Name, Description, Dosage, Method of Admin

Patient No, Full Name

WARD (Ward No, Ward Name)

The primary keys underlined.

Q.58. Consider the following relation definition and sample data:

ProjectID	EmployeeName	EmployeeSalary
100a	Adnan	65000
100a	Salman	5100
100b	Salman	51000
200a	Adnan	64000
200b	adnan	64000
200c	Amir	28000
200c	Salman	51000
200d	Amir	28000

PROJECT (ProjectID, EmployeeName, EmployeeSalary)

where ProjectID is the name of a work project

EmployeeName is the name of an employee who works on that project

EmployeeSalary is the salary of the employee whose name is EmployeeName

Assuming that all of the functional dependencies and constraints are apparent in data, which of the following statements is true?

A. **ProjectID → EmployeeName**
FALSE: There are multiple Employee Names for each project. (see project 100A)

B. **ProjectID → EmployeeSalary**
FALSE: There are multiple Employee Salaries for each project. (see project 100a)

C. **(ProjectID, EmployeeName) → EmployeeSalary**
FALSE: Each employee's salary is always the same, regardless of the project. (see salman)

D. **EmployeeName → EmployeeSalary**
TRUE: An Employee Name always has the same Salary. (see salman)

E. **EmployeeSalary → ProjectID**
FALSE: There are multiple ProjectIDs for a given Salary. (see Salary 51000)

F. **EmployeeSalary → (ProjectID, EmployeeName)**
FALSE: There are multiple ProjectID and EmployeeName combinations for a given Salary (see Salary 51000)

Answer these questions:

G. **What is the key of PROJECT?**
ProjectID and EmployeeName (Composite Key)

H. **Are all non-key attributes (if any) dependent on the whole key?**
No: EmployeeSalary is non-key attribute and it is dependent on EmployeeName only.

I. **In what normal form is PROJECT?**
Project is in first normal form because it has no multi-valued attributes. It is not in second normal form because it has a partial dependency. Key: ProjectID+EmployeeName but EmployeeName → EmployeeSalary.

J. **Describe two modification anomalies from which PROJECT suffers.**
Insertion Anomaly: It is not possible to add an Employee until Employee is assigned to a Project. Likewise, you cannot add a Project until and Employee is assigned to the Project.

Update Anomaly: If you want to change Salman's Salary, you need to change three rows of data in order to change one Employee's salary.
Deletion Anomaly: If Amir did not work on Project 200C and worked in Project 200D only, deletion of ProjectC would delete the fact that Amir's salary was 28000.

K. **Is ProjectID a determinant?**
No

L. **Is EmployeeName a determinant?**
YES: EmployeeName → EmployeeSalary

M. **Is (ProjectID, EmployeeName) a determinant?**
No

N. **Is EmployeeSalary a determinant?**

No: In this case, it appears that it can be a determinate because no two people have same salary. Using logic however, one can assume that there is no business rule in a firm that says two people cannot have the same salary.

O. **Does this relation contain a partial dependency? If so, what is it?**

YES, the relation does contain a partial dependency. The key is ProjectID+EmployeeName but EmployeeName → EmployeeSalary.

P. **Redesign this relation to eliminate the modification anomalies.**
PROJECT (ProjectID, EmployeeName)
EMPLOYEE (EmployeeName, EmployeeSalary)

Q.59. Consider the following relation definition and sample data:

PROJECT-HOURS Relation

EmployeeName	ProjectId	TaskID	Phone	Total Hours
SALMAN	100A	B-1	788792	12
SALMAN	100A	P-1	788792	12
SALMAN	200B	B-1	788792	12
SALMAN	200B	P-1	788792	12
ADNAN	100A	C-1	788988	26
ADNAN	200A	C-1	788988	26
ADNAN	200D	C-1	788988	26

PROJECT-HOURS (EmployeeName, ProjectID, TaskID, Phone, TotalHours)
where EmployeeName is the name of an employee
ProjectID is the name of a project
TaskID is the name standard work task
Phone is the employee's telephone number
TotalHours is the hours worked by the employee on this project
Assuming that all of the functional dependencies and constraints are apparent in this data, which of the following statements is true?

A. **EmployeeName → ProjectID**
NO: There are multiple ProjectIDs for each EmployeeName

B. **EmployeeName →→ ProjectID**
YES: Each EmployeeName has three or more ProjectIDs

C. **EmployeeName → TaskID**
NO: There are multiple TaskIDs for each EmployeeName

D. **EmployeeName →→ TaskID**
YES: Salman has multiple (2) TaskIDs

E. **EmployeeName →: Phone**
YES: Each EmployeeName has exactly one Phone value

F. **EmployeeName →: TotalHours**
YES: Each EmployeeName has exactly one TotalHours value

G. **(EmployeeName, ProjectID) → TotalHours**

YES: It is true based on fourth assumption. Looking at data only, it is more probable that EmployeeName → TotalHours. It is because TotalHours is same for EmployeeName regardless of ProjectID.

H. **(EmployeeName, Phone) → TaskID**

NO: There are multiple TaskIDs for a given EmployeeName, Phone combination

I. **ProjectID → TaskID**

NO: There are multiple TaskIDs for a given ProjectID

J. **TaskID → ProjectID**

NO: There are multiple ProjectIDs for a given TaskID

Answer these questions:

K. **What are all of the determinants?**

EmployeeName → Phone
EmployeeName → → TaskID
EmployeeName, ProjectID → TotalHours

L. **Does this relation contain a partial dependency? If so, what is it?**

YES, it contains a partial dependency. The key is EmployeeName + ProjectID + TaskID but EmployeeName → Phone. So there is a partial dependency.

M. **Does this relation contain a multi-value dependency? If so, what are the unrelated attributes?**

YES: The related attributes are ProjectID and TaskID. EmployeeName → → ProjectID and EmployeeName → → TaskID

N. **What is the deletion anomaly that this relation contains?**

If Employee Salman no longer has TaskID B-1 two rows must be deleted, Row 1 and Row 3. The deletion of one fact requires deletion or two rows.

O. **How many themes does this relation have?**

It would appear that there are at least three themes. 1) Employees and their Tasks 2) Employees and their Phone Numbers and 3) Employees and hours worked on a project.

P. **Redesign this relation to eliminate the modification anomalies. How many relations did you use? How many themes does each of your new relations contain?**

EMPLOYEE (EmployeeName, Phone)
EMPLOYEE_TASKS (EmployeeName, TaskID)
PROJECT-HOURS (EmployeeName, ProjectID, TotalHours)

Three relations are required, one for each theme. Each relation now carries one theme