a) Define the following terms as used in database systems:

(i**) Database:**

A database is a structured collection of data that is organized, managed, and stored in such a way that it allows for efficient retrieval, updating, and management of data. Databases are designed to store and manage large volumes of information, making it easy for users or applications to access, manipulate, and analyze the data as needed.

(ii) Normalization:

Normalization is the process of organizing a database in such a way that it reduces redundancy and dependency within the data. It involves breaking down a database into smaller, well-structured tables to minimize data duplication and anomalies. The main goal of normalization is to ensure data integrity, efficiency, and to prevent data update anomalies.

(iii) Input mask:

An input mask is a feature in a database system that defines the format and structure of data that can be entered into a specific field. It acts as a template or pattern that guides users when they input data, ensuring consistency and accuracy. Input masks are commonly used for fields with fixed formats, such as phone numbers, dates, or social security numbers.

(iv) Entity:

In the context of a database, an entity represents a distinct object, person, place, concept, or event about which data is stored. Entities are typically mapped to tables in a relational database, where each row in the table represents an instance of that entity, and each column represents an attribute of the entity.

(v) Attribute:

An attribute, also known as a field or column, is a characteristic or property that describes an entity. It represents a specific piece of information associated with an entity. For example, in a "Customer" entity, attributes may include "CustomerID," "Name," "Address," and "Phone Number."

(vi) Meta-data:

Meta-data, short for "metadata," refers to data that provides information about the structure, organization, and properties of the database and its contents. It describes the data elements, data types, relationships, constraints, and other details necessary for understanding and managing the data within the database.

b) Highlight three differences between data and information:

1. Meaning and Context:

- Data refers to raw facts, figures, or symbols without any context or interpretation.

- Information is the processed and organized data that has meaning and context, making it useful for decision-making and understanding.

2. **Structure and Representation:**

- Data can be unstructured and disorganized, often requiring further processing and organization.

- Information is structured, organized, and presented in a way that is easily understandable and usable.

3**. Purpose and Use:**

- Data is the foundation on which information is built, but in its raw form, it lacks actionable insights.

- Information is the output of data analysis, providing insights, conclusions, and knowledge that can be used for various purposes, such as decision-making or problem-solving.

c) Explain the ACID properties in a database:

The ACID properties are a set of properties that ensure database transactions are processed reliably and consistently. ACID stands for:

1. Atomicity: This property ensures that a transaction is treated as a single, indivisible unit of work. It means that either all the operations within a transaction are successfully completed, or none of them are. If any part of the transaction fails, the entire transaction is rolled back, and the database returns to its original state.

2**. Consistency:** This property ensures that a transaction brings the database from one valid state to another. The database must adhere to predefined rules and constraints, maintaining data integrity throughout the transaction.

3. **Isolation:** Isolation ensures that the intermediate states of a transaction are not visible to other transactions until the transaction is completed. It prevents interference between concurrent transactions, thus maintaining data integrity and preventing race conditions.

4. **Durability:** This property guarantees that once a transaction is committed, its changes are permanent and will survive any subsequent system failures or crashes. The changes made by a committed transaction are stored permanently in the database.

d) Outline the security mechanisms that an organization can use to secure its data in DDBMS:

1. **Authentication:** This mechanism ensures that users are who they claim to be. It involves verifying user identities through credentials like usernames, passwords, biometrics, or multi-factor authentication (MFA).

2. **Authorization**: Authorization controls determine the level of access each authenticated user has within the database. It involves assigning appropriate permissions and privileges to users or roles, limiting their access to specific data or operations.

3. **Encryption:** Encryption is the process of converting data into a secure, unreadable format using algorithms. It protects sensitive data from unauthorized access during storage and transmission.

4. **Access Control Lists (ACLs**): ACLs are lists associated with database objects that specify the permissions and access rights granted to specific users or user groups. They help regulate who can perform specific actions on the data.

5. **Role-Based Access Control (RBAC):** RBAC is a method of access control where access permissions are assigned to roles, and users are assigned to these roles. It simplifies the management of access control by grouping users with similar responsibilities.

6. **Auditing** and Logging: Implementing auditing and logging mechanisms allows organizations to monitor and track activities within the database. It helps in identifying security breaches or suspicious activities.

7. **Data Masking:** Data masking involves concealing original data with fictional but realistic data for testing, development, or sharing purposes. It ensures sensitive information is not exposed to unauthorized users.

8. **Firewalls and Network Security**: Securing the network infrastructure with firewalls, intrusion detection systems (IDS), and other security measures prevents unauthorized access to the database servers.

e) **Elucidate three: types of users in a DBMS**

1**. End Users:** End users are the individuals who interact directly with the database to perform various operations and retrieve information. They do not need to have knowledge of the underlying database structure or technical details. End users use front-end applications, such as web interfaces or query tools, to data.

2**. Database Administrators (DBAs):** DBAs are responsible for managing and maintaining the database system. They access the handle tasks such as database design, configuration, backup and recovery, performance tuning, and security management. DBAs ensure that the database operates efficiently, securely, and with minimal downtime.

3. **Application Developers:** Application developers design and develop software applications that interact with the database to perform specific functions or tasks. They write queries, stored procedures, and application code to access, manipulate, and present data from the database. These developers may work on web applications, desktop software, or mobile apps.

f) Microsoft Access is one of the most popular DBMS nowadays. Explain five data types supported by Microsoft Access:

1. **Text**: The Text data type is used to store alphanumeric characters, such as names, addresses, or descriptions. It can accommodate up to 255 characters.

2. **Number:** The Number data type is used for numeric values, including integers, decimals, and floating-point numbers. It can store both whole numbers and fractions.

3. **Date/Time:** The Date/Time data type is used to store dates and times. It allows for the storage of precise timestamps and supports various date and time formats.

4. **Yes/No:** The Yes/No data type is a boolean data type that can store only two values: Yes (True) or No (False). It is suitable for representing binary or true/false values.

5. **Memo:** The Memo data type is used to store large blocks of text, accommodating up to 65,536 characters. It is suitable for storing lengthy descriptions, notes, or comments.

a) **Apart from the relational database model, discuss any other database models (8 marks):**

1. Hierarchical Database Model:

The hierarchical database model organizes data in a tree-like structure, where each record has a single parent record and may have multiple child records. It is well-suited for representing one-to-many relationships. In this model, records are connected through parent-child relationships, forming a hierarchy.

2. **Network Database Model:**

The network database model is an extension of the hierarchical model and allows for more flexible relationships between records. It uses pointers to represent relationships, enabling a record to have multiple parent and child records. This model is suitable for many-to-many relationships.

3. **Object-Oriented Database Model:**

The object-oriented database model extends the concepts of object-oriented programming to databases. It stores data in the form of objects, which encapsulate data and the operations that can be

performed on that data. This model is useful for managing complex data with rich relationships and behaviors.

4. Entity-Relationship Model (ER Model):

The entity-relationship model is a conceptual data model used to represent the relationships between entities in a database. It uses entities to represent real-world objects, and relationships to represent associations between these entities. The ER model is widely used in database design to create a blueprint of the database structure.

5. **Document-Oriented Database Model:**

The document-oriented database model stores data in the form of documents, usually in formats like JSON or BSON. Each document can have its own unique structure, making it suitable for handling semi-structured and unstructured data. This model is commonly used for web applications and content management systems.

6. Columnar Database Model:

The columnar database model stores data in columns rather than rows. This format is optimized for analytical processing and data warehousing because it allows for faster data retrieval and compression. It is suitable for handling large volumes of data and complex queries.

b) **Learning institution maintains the details of its lecturers who are teaching various units as follows: Lecturer PF NO, lecturer - Name, Grade, department - code, Department - Name, Subject - code subject - name, Subject - level. Each lecturer may teach many subjects but cannot belong to more than one department. Do the following:**

(i) **Normalize this data up to the 2nd normal form (6 marks):**

To normalize the data up to the 2nd normal form (2NF), we need to identify the partial dependencies and remove them. 2NF requires that every non-prime attribute (an attribute not part of any candidate key) must be fully functionally dependent on the entire candidate key. Let's break the data into two tables:

Table 1: Lecturer (PF NO, Lecturer Name, Grade, Department Code)

- PF NO (Primary Key)

- Lecturer Name

- Grade

- Department Code (Foreign Key referencing Department table)

Table 2: Subject (Subject Code, Subject Name, Subject Level, PF NO)

- Subject Code (Primary Key)

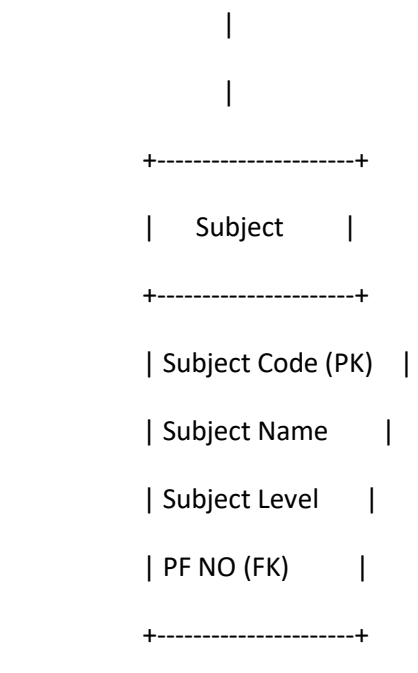
- Subject Name

- Subject Level

- PF NO (Foreign Key referencing Lecturer table)

(ii**) Draw an entity-relationship diagram for this case (6 marks):**

```
                +------------------+
                |     Lecturer     |
                +------------------+
                | PF NO (PK)       |
                | Lecturer Name    |
                | Grade            |
                | Department Code  |
                +------------------+
                       |
                       |
```




```

The ER diagram shows that each lecturer can teach multiple subjects (one-to-many relationship), and the Department Code in the Lecturer table is a foreign key referencing the Department Code in the Department table.

QUESTION THREE (20 MARKS)

a) **Differentiate between physical and logical database design (4 marks):**

**Logical Database Design:**

- Logical database design focuses on creating a high-level representation of the database's structure and relationships without considering the physical aspects or implementation details.

- It involves defining entities, attributes, and relationships based on the requirements gathered during the database analysis phase.

- The result of logical database design is an Entity-Relationship Diagram (ERD) that serves as a blueprint for the database's structure and constraints.

- Logical design is independent of any specific database management system (DBMS) and can be implemented in different systems.
```

Physical Database Design:

- Physical database design involves translating the logical model into an actual database implementation, considering the specific features and limitations of the chosen DBMS.

- It focuses on optimizing the database for performance, storage efficiency, and scalability.

- Physical design decisions include defining data types, indexing, partitioning, table structures, file organizations, and access methods.

- The physical design ensures that the database performs efficiently and effectively based on the hardware and software resources available.

b) Assume you have a database table with the following fields: Vehicle ID, Vehicle Name, Price, and Description. Perform the following:

(i**) Choose the appropriate primary key and justify your choice (2 marks):**

The primary key in a database table uniquely identifies each record in the table and ensures data integrity. In this case, "Vehicle ID" is the most suitable primary key choice because it is a unique identifier for each vehicle. It is common for vehicle data to have unique identification numbers, making "Vehicle ID" an appropriate candidate for the primary key.

(ii) Differentiate between a foreign key and a primary key (2 marks):

Primary Key:

- A primary key is a unique identifier for each record in a table.

- It ensures that each record in the table has a distinct identity.

- There can be only one primary key in a table.

- It is used to establish relationships with other tables through foreign keys.

- A primary key can't contain null values.

Foreign Key:

- A foreign key is a field in a table that refers to the primary key in another table.

- It establishes a relationship between two tables, representing a one-to-many or many-to-one association.

- There can be multiple foreign keys in a table, each referencing a different table.

- A foreign key helps maintain referential integrity by ensuring that the data in the referencing table matches the data in the referenced table's primary key.

(iii) Write SQL command to insert a record into this table (4 marks):

To insert a record into the table, we use the SQL INSERT INTO statement. Assuming we want to insert a new vehicle with the following details:

- Vehicle ID: 1234

- Vehicle Name: "Toyota Camry"

- Price: 25000

- Description: "Mid-size sedan, comfortable and reliable."

The SQL command will be:

```sql
INSERT INTO TableName (VehicleID, VehicleName, Price, Description)
VALUES (1234, 'Toyota Camry', 25000, 'Mid-size sedan, comfortable and reliable.');
```

Replace "

TableName" with the actual name of the table where you want to insert the record.

(iv) Write SQL command to delete a record from this table (4 marks):

To delete a record from the table, we use the SQL DELETE statement. Assuming we want to delete the vehicle record with Vehicle ID 1234, the SQL command will be:

```sql
DELETE FROM TableName WHERE VehicleID = 1234;
```

Replace "TableName" with the actual name of the table from which you want to delete the record.

(v) Write SQL command to view all records in the table (4 marks):

To view all records in the table, we use the SQL SELECT statement without any conditions. The SQL command will be:

```sql
SELECT * FROM TableName;
```

Replace "TableName" with the actual name of the table you want to view.

QUESTION FOUR (20 MARKS)

a) Discuss the steps you would follow in creating a query in Microsoft Access (4 marks):

1. Launch Microsoft Access and open the database that contains the tables you want to query.

2. Go to the "Create" tab on the Access Ribbon and click on "Query Design."

3. In the "Show Table" dialog box, add the tables you want to query by selecting them and clicking the "Add" button.

4. Close the "Show Table" dialog box to see the Query Design View.

5. In the Query Design View, arrange the tables and link them using the appropriate join type (inner join, outer join, etc.) by dragging and dropping the field names that create the relationships between the tables.

6. Select the fields you want to include in the query by dragging them from the tables into the design grid.

7. Specify any criteria for filtering data by entering the conditions in the "Criteria" row of the corresponding field in the design grid.

8. If needed, apply sorting to the results by right-clicking on a field and choosing "Sort Ascending" or "Sort Descending."

9. Save the query with a descriptive name.

10. Run the query by clicking the "Run" button on the Query Design Ribbon. The query will display the results based on the specified criteria and joins.

b) Explain challenges that an organization is likely to encounter when implementing a database management system (6 marks):

1. **Data Migration:** Transferring data from existing systems to the new DBMS can be challenging, especially when dealing with large volumes of data or data in different formats. Ensuring data integrity and consistency during migration is critical.

2. **Integration with Existing Systems:** Organizations may have multiple systems in place, and integrating the new DBMS with existing applications can be difficult. Ensuring seamless data flow and compatibility between systems can be a challenge.

3. **Training and User Adoption**: Employees may need training to use the new DBMS effectively. Resistance to change and the learning curve for adopting the new system can hinder successful implementation.

4. **Security Concerns:** Implementing a DBMS requires robust security mechanisms to protect sensitive data from unauthorized access or breaches. Designing and implementing a comprehensive security strategy can be complex.

5. **Performance Optimization:** Ensuring that the DBMS performs efficiently, especially when dealing with large datasets and complex queries, is a challenge. Proper indexing, query optimization, and hardware considerations are crucial for optimal performance.

6. Scalability: As the organization grows and data volume increases, the DBMS should be able to scale to accommodate the expanding data needs. Planning for future scalability is essential to avoid performance bottlenecks.

c) Discuss four types of attributes in a database (8 marks):

1. **Simple Attribute:** A simple attribute is an attribute that cannot be divided further into subparts. It represents a single value for an entity. For example, "Age" or "Name" of a person is a simple attribute.

2. **Composite Attribute**: A composite attribute is an attribute that can be divided into multiple subparts, each representing a distinct characteristic. For example, the attribute "Address" can be divided into subparts like "Street," "City," "State," and "ZIP Code."

3. **Single-Valued Attribute:** A single-valued attribute is an attribute that holds only one value for each instance of an entity. For example, the "Date of Birth" attribute for a person is single-valued because a person has only one birth date.

4. **Multi-Valued Attribute:** A multi-valued attribute is an attribute that can hold multiple values for an entity. For example, the "Phone Number" attribute for a person can have multiple phone numbers associated with it.

5. **Derived Attribute:** A derived attribute is an attribute whose value is calculated or derived from other attributes. It does not store data directly but is derived through a formula or computation. For example, the "Age" attribute of a person can be derived from the "Date of Birth" attribute.

6. **Key Attribute:** A key attribute is an attribute used to uniquely identify each instance of an entity in a table. It helps in establishing relationships between different tables in a database. Primary keys and foreign keys are examples of key attributes.

7. **Null Attribute:** A null attribute is an attribute that can have missing or unknown values. Null attributes are used when data is not available or when a value is not applicable or relevant for a particular entity instance.

8. **Composite Key:** A composite key is a key that consists of two or more attributes combined to form a unique identifier for an entity. It is used when a single attribute cannot uniquely identify an entity, but the combination of multiple attributes can.