1. What is meant by a database?

* A Database is an organized, consistent, and logical collection of data that can easily be updated, accessed, and managed. Database mostly contains sets of tables or objects (anything created using create command is a database object) which consist of records and fields. A tuple or a row represents a single entry in a table. An attribute or a column represents the basic units of data storage, which contain information about a particular aspect of the table. DBMS extracts data from a database in the form of queries given by the user.

1. What is DBMS?

A Database Management System (DBMS) is a software application or system that enables users to efficiently store, organize, retrieve, and manipulate data. It serves as an intermediary between the user and the underlying database, providing a structured and controlled environment for managing data.

1. Types of DBMS.

* RDBMS: Organizes data in tables using SQL. Examples: Oracle, MySQL, SQL Server.
* NoSQL: Handles unstructured data, scales horizontally. Types: Document, Key-Value, Column-Family, Graph. Examples: MongoDB, Cassandra.
* OODBMS: Stores data in object-oriented models. Examples: ObjectDB, db4o.
* In-Memory DBMS: Stores data in main memory for speed. Examples: Redis, SAP HANA.
* Distributed DBMS: Spreads data across servers for scalability. Examples: Google Bigtable, Hadoop.

1. Benefits of Database.
2. Data Centralization: All data is stored in one place, making it easier to manage and maintain.
3. Data Consistency: DBMS enforces data integrity rules, ensuring that data remains accurate and consistent.
4. Data Security: Access control mechanisms protect sensitive information from unauthorized users.
5. Data Scalability: Many DBMS systems can scale to handle increasing data volumes and user loads.
6. Data Recovery: Built-in backup and recovery mechanisms safeguard against data loss.
7. Data Querying: SQL support allows users to retrieve and analyze data efficiently.
8. Key functions of DBMS.
9. Data Storage: Organizes data in structured formats like tables.
10. Data Retrieval: Allows efficient data access through queries.
11. Data Manipulation: Supports adding, updating, and deleting data.
12. Data Security: Controls data access for authorized users.
13. Data Integrity: Enforces data accuracy and reliability.
14. Concurrency Control: Manages simultaneous data access.
15. Backup and Recovery: Safeguards data and offers recovery tools.
16. What is database partitioning?

* It (Database partitioning) is a process where a logical database is divided into different independent parts. The database objects like tables, indexes are subdivided and managed, and accessed at the granular level. Data partitioning is done to divide the data into separate parts, such as training, validation, and testing datasets.

1. What is RDBMS?

* RDBMS is the Relational Database Management System which contains data in the form of the tables and data is accessed on the basis of the common fields among the tables.

1. Define Normalization?

* Normalization in simple terms is a process used in databases to organize data more efficiently. It involves breaking down large tables into smaller, related tables to reduce duplication (repeated data) and improve data consistency.

1. Enlist the advantages of a normalizing database?

* No duplicate entries
* Saves storage space
* Boosts the query performances.

1. Why do you need to use functional dependencies while normalizing databases?

* Functional dependencies are essential in the process of normalizing databases because they help ensure that a database design adheres to certain rules and best practices for data organization. Normalization is the process of structuring a database schema to eliminate redundancy and improve data integrity.

1. What are the different steps involved in normalizing a database?

* Identify entities and attributes.
* Define primary keys.
* Apply 1NF (atomic values, no repeating groups).
* Apply 2NF (full key dependency).
* Apply 3NF (remove transitive dependencies).
* Apply BCNF (no partial/overlapping dependencies).
* Apply 4NF (handle multi-valued dependencies).
* Apply 5NF (address join dependencies).
* Evaluate and optimize for performance.
* Document schema, implement tables, test, and maintain.

1. Is there any difference between “normalization” and “database normalization”? If yes, then what’s the difference?

* “Normalization” is a general concept of organizing data, while “database normalization” specifically refers to structuring data in relational databases to reduce redundancy and enhance integrity.

1. What are the different levels of abstraction in the DBMS?
2. Physical Level: This is the lowest level of the data abstraction which states how the data is stored in the database.
3. Logical Level: This is the next level of the data abstraction which states the type of the data and the relationship among the data that is stored in the database.
4. View Level: This is the highest level in the data abstraction which shows/states only a part of the database.
5. What is a checkpoint in DBMS and when does it occur?

* A checkpoint is a mechanism where all the previous logs are removed from the system and are permanently stored on the storage disk. So, basically, checkpoints are those points from where the transaction log record can be used to recover all the committed data up to the point of crash.

1. What is the Relationship?

* The Relationship is defined as an association among two or more entities. There are three types of relationships in DBMS-

1. One-To-One: Here one record of any object can be related to one record of another object.
2. One-To-Many (many-to-one): Here one record of any object can be related to many records of other objects and vice versa.
3. Many-to-many: Here more than one record of an object can be related to n number of records of another object.
4. Enlist the disadvantages of the query?

* No indexes
* Stored procedures are excessively compiled.
* Triggers and procedures are without SET NOCOUNT ON.
* Complicated joins making up inadequately written queries.
* Cursors and temporary tables showcase a bad presentation.

1. What is an ER diagram in DBMS?

* An entity-relationship model or an entity-relationship diagram is a visual representation of data that is represented as entities, attributes, and relationships are set between entities.

1. What are the integrity rules in DBMS?

* Data integrity is one significant aspect while maintaining the database. So, data integrity is enforced in the database system by imposing a series of rules. Those set of integrity is known as the integrity rules.
* So basically, Integrity rules are the rules that should be followed while creating a data in a database.

1. Entity Integrity: It specifies that the “Primary key cannot have a NULL value.”
2. Referential Integrity: It specifies that the “Foreign Key can be either a NULL value or should be the Primary Key value of other relation”.
3. Define Atomicity and Aggregation.

* Atomicity: Atomicity is a concept from database management that refers to the idea that a series of operations should be treated as a single unit. It means that all parts of a transaction (a set of database operations) should be completed successfully together, or if any part fails, none of the changes should be saved.

1. All or Nothing: Imagine you are transferring money from one bank account to another. Atomicity ensures that either both the debit from one account and the credit to the other account happen together, or if there’s a problem, neither operation is done. This prevents situations where money might be deducted from one account but not added to the other.
2. Consistency: By ensuring that operations are atomic, databases maintain consistency. This means the database will not be left in a half-updated state.

* Aggregation: Aggregation refers to the process of combining multiple pieces of data to get a summary or a single value. It’s like summarizing detailed information into a more meaningful result.

1. Combining Data: For example, if you have a list of sales transactions, aggregation might involve calculating the total sales, the average sale amount, or the maximum sale value.
2. Common Uses: Aggregation is used in many scenarios, such as summarizing data in reports, creating dashboards, or analyzing trends. It helps in understanding and making decisions based on the combined data.
3. What is an entity-relationship model?

* It is a diagrammatic approach to database design, where you represent real-world objects as entities and mention relationships between them. This approach helps the team of DBA’s to understand the schema easily.

1. What is concurrency control?

* Concurrency control is a concept in database management that ensures that multiple users or processes can access and modify the database at the same time without causing inconsistencies or errors.
* When multiple users or processes try to read from or write to the database simultaneously, concurrency control ensures that their actions do not interfere with each other. For instance, two people shouldn’t be able to book the last seat on a flight at the same time.
* It ensures that even with simultaneous access, the database remains accurate and consistent. For example, if two users are trying to update the same record, concurrency control ensures that the database reflects both changes correctly without data loss or corruption.
* Working: Transactions - Operations are grouped into transactions, which are sequences of operations that are treated as a single unit. Concurrency control manages these transactions to ensure that they do not overlap in a way that causes data issues.

1. Disadvantage of DBMS?

* Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization.
* Most database management systems are often complex systems, so training for users to use the DBMS is required.
* In some organizations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media
* Use of the same program at a time by many users sometimes leads to the loss of some data.
* DBMS can’t perform sophisticated calculations.

1. What are the unary operations in Relational Algebra?

* Unary operations are operations that use single operands. Unary operations in relational algebra are PROJECTION and SELECTION.
* -=,>=,<=,+= are the relational operators used in SELECTION.

1. How many types of database languages are?
2. Data Definition Language (DDL): These commands are used for updating the data. CREATE, ALTER, DROP, TRUNCATE, RENAME are some examples of DDL commands.
3. Data Manipulation Language (DML): These commands are used for the manipulation of already updated data. SELECT, UPDATE, INSERT, DELETE are some examples of DML commands.
4. Data Control Language (DCL) : These commands are used for giving and removing user access to the database. GRANT and REVOKE are the examples of DCL commands.
5. Transaction Control Language (TCL) : These are the commands used for managing transactions in the database. TCL is used for managing the changes made by DML. COMMIT, ROLLBACK, and SAVEPOINT are the examples of TCL commands.
6. Define a Relation Schema?

* A relation schema is known as the blueprint with the help of which we can explain how the data is organized into tables.

1. What is a degree of Relation?

* A degree of relation is also known as Cardinality it is defined as the number of occurrence of one entity which is connected to the number of occurrence of other entity.

1. What is Relational Algebra?

* Relational Algebra is a Procedural Query Language that contains a set of operations that take one or two relations as input and produce a new relationship. Relational algebra is the basic set of operations for the relational model. The decisive point of relational algebra is that it is similar to the algebra which operates on the number.
* There are a few fundamental operations of relational algebra:

1. select
2. project
3. set difference
4. union
5. rename, etc.
6. Explain ACID properties?
7. ATOMICITY: Atomicity is more generally known as ? all or nothing rule.’ This implies all are considered as one unit, and they either run to completion or are not executed at all.
8. CONSISTENCY: This property refers to the uniformity of the data. Consistency implies that the database is consistent before and after the transaction.
9. ISOLATION: This property states that the number of transactions can be executed concurrently without leading to the inconsistency of the database state.
10. DURABILITY: This property ensures that once the transaction is committed it will be stored in the non-volatile memory and a system crash can also not affect it anymore.
11. What do you understand by Data Model?

* A data model is a conceptual framework that defines how data is organized, structured, and related within a database or information system. It includes entities (objects), attributes (properties), relationships (connections), and constraints (rules). Data models are essential for designing and managing data effectively in software and database development. accessible. In this architecture, the application on the client-end interacts with an application on the server which further communicates with the database system.

1. What is Data Warehousing?

* Data warehousing is the process of collecting, storing, and managing data from various sources in a central repository. It’s used to support business intelligence and analytics by providing a unified, historical, and structured view of an organization’s data for better decision-making.

1. Explain different types of Normalization forms in a DBMS.
2. 1NF: Ensure columns have atomic values, unique names, and order independence.
3. 2NF: Eliminate partial dependencies, ensuring non-key attributes depend on the entire primary key.
4. 3NF: Remove transitive dependencies, so non-key attributes rely solely on the primary key.
5. BCNF: Eliminate all non-trivial functional dependencies, ensuring tables are free from redundancy.
6. 4NF: Address multi-valued dependencies to prevent anomalies with multiple attribute values.
7. 5NF: Handle join dependencies, enabling further table decomposition without data loss.
8. What is the 3-Tier architecture?

* The 3-Tier Architecture in Database Management Systems (DBMS) is a way to organize and structure applications to separate different parts of the system. This architecture divides the application into three distinct layers, each with its own responsibilities.

1. Presentation Layer (Client Tier): This is the user interface or the front end of the application. It’s what users interact with directly. It handles user inputs and displays data to the user. For example, in a web application, this would be the web pages or mobile app screens that users see and use to interact with the system.
2. Application Layer (Business Logic Tier): This is the middle layer that processes the data received from the presentation layer and sends commands to the database. It handles the core functionality and business rules of the application. It processes user requests, performs calculations, and applies logic to the data. For example, if you submit a form to create a new account, this layer will process the request and handle the logic involved.
3. Database Layer (Data Tier): This is the backend layer where data is stored and managed. It manages and provides access to the database. This layer handles data storage, retrieval, and management. It ensures that data is stored securely and efficiently and is retrieved correctly when needed.
4. What is 2-Tier architecture?

* 2-Tier Architecture is a simpler way of organizing a database application, compared to the 3-Tier Architecture. It divides the application into two main layers:

1. Client Tier (Presentation Layer): This is the user interface or the front end of the application, where users interact directly with the system. It handles user inputs and displays the results. For example, this could be a desktop application, a web browser, or a mobile app where users enter data and view information.
2. Server Tier (Database Layer): This is the backend part of the application where the data is stored and managed. t processes requests from the client tier, manages the database, and provides the requested data back to the client. It handles data storage, retrieval, and management.
3. What is Weak Entity set?

* A Weak Entity Set is a concept in database design, particularly in Entity-Relationship (ER) models. It’s a way to handle entities (or objects) that don't have a unique identifier on their own and depend on another entity for identification.

1. What is an attribute?

* An attribute is a piece of information that helps describe or define an entity. For example, if you have an entity called "Person," attributes could include things like their name, age, and address.
* In a database, attributes are the columns in a table. Each attribute holds a specific type of information about the records (rows) in that table.

1. What are the Important differences between DBMS and RDBMS?

* Data Structure:

1. DBMS: Data is stored in a file or a collection of files, often in a hierarchical or network structure. It does not necessarily follow a specific structure for how data is related.
2. RDBMS: Data is stored in tables that are related to each other through keys. These tables are organized in a structured way following a relational model, where relationships between data are well-defined.

* Data Relationships:

1. DBMS: Relationships between data are not as strictly defined. It might not support complex relationships between different data elements.
2. RDBMS: Supports and enforces relationships between tables using keys (primary keys and foreign keys). This allows for complex queries and data integrity.

* Query Language:

1. DBMS: May not have a standardized query language, and if it does, it may be less powerful or less standardized.
2. RDBMS: Uses SQL (Structured Query Language), a powerful and standardized language for querying and managing relational databases.

* Scalability:

1. DBMS: Might face limitations in handling large amounts of data or complex queries due to less structured data management.
2. RDBMS: Generally designed to handle large volumes of data and complex queries efficiently due to its structured approach.
3. What are the Advantages of Using a DBMS system?

* DBMS offers various techniques and powerful functions to efficiently store & retrieve data.
* It is an efficient handler to balance the needs of multiple applications using the same data.
* Provides Uniform administration procedures for data management.
* Application programs that are never exposed to details of data representation and storage.
* It implies integrity constraints to get a high level of protection against prohibited access to data.
* It helps you to reduce Application Development Time.

1. What is Functional Dependency?

* Functional Dependencies (FD) in Database Management systems define the relationship between an attribute and another. It allows you to maintain the quality of data in the Database.

1. What is Primary Key?

* Primary keys are columns or sets of columns that uniquely identify every row in the table of an RDMS system. This key can’t be duplicated means the same value should not appear more than once in the table.

1. What is Foreign Key?

* A foreign key is a column that creates a relationship between two tables. They are used to maintain data integrity and facilitate navigation between two instances of the same entity. Moreover, it serves as a cross-reference between two tables since it refers to another table’s primary key.

1. What is a Distributed Database System?

* A Distributed Database System is a type of database system where data is stored across multiple locations or servers rather than in a single central place.
* Instead of having all the data in one central server or database, a distributed database system spreads the data across different servers or computers.
* Even though the data is distributed, the system makes sure that everything works together seamlessly. Users can access and manage the data as if it were all in one place, without needing to know about the different locations where the data is stored.

1. What is Database Architecture?

* Database Architecture is a DBMS design representation that helps you Design, develop, implement, and maintain the DBMS system. It divides the DBMS into separate components that can be independently changed, modified, replaced, and altered.

1. How Can you communicate with an RDBMS?

* You have to use SQL to communicate with the RDBMS using queries of SQL to provide the input to the Database. After processing the queries, the Database will provide us with the required output.

1. What is Data Mining?

* Data Mining is the process of discovering patterns, trends, and valuable information from large sets of data. Data mining involves analyzing data from various angles to uncover hidden patterns or relationships.

1. What is a Database Schema and Why is It Important?

* A database schema is a blueprint or architecture of how data is organized in a database. It defines the tables, the fields in each table, and the relationships between fields and tables.
* A schema is important because it provides a clear structure for the data, ensuring consistency, clarity, and integrity. It helps developers and database administrators understand how data is connected and how to retrieve and manipulate it efficiently.

1. Explain the Difference Between OLTP and OLAP Databases.

* OLTP (Online Transaction Processing): Databases designed for managing transaction-oriented applications. They are optimized for a large number of short online transactions (insert, update, delete). Example: Retail sales systems.
* OLAP (Online Analytical Processing): Databases designed for querying and reporting, often used for data analysis and business intelligence. They are optimized for read-heavy operations on large volumes of data. Example: Data warehousing.

1. Describe the Process of ETL (Extract, Transform, Load).

* Extract: Collecting data from different source systems.
* Transform: Converting the extracted data into a suitable format or structure for querying and analysis. This might involve cleaning the data, removing duplicates, and ensuring data consistency.
* Load: Inserting the transformed data into the target data warehouse or database.

1. What is a Relational Database and How does It Differ from a NoSQL Database?

* A relational database uses structured tables to store data, with predefined schemas and relationships (usually using SQL). It ensures data integrity through ACID properties and is suitable for complex queries and transactions.
* A NoSQL database, on the other hand, is designed for unstructured or semi-structured data and can store data in various formats like key-value pairs, documents, or graphs. NoSQL databases are often more flexible and scalable, suitable for big data and real-time web applications, but they might not provide the same level of ACID compliance as relational databases.

1. What are the Different types of Database Partitioning and When would we Use Each Type?
2. Horizontal Partitioning: Divides a table into multiple tables with the same structure, distributing rows based on a range or list of values. Used to improve performance and manageability by spreading the data across multiple storage locations.
3. Vertical Partitioning: Divides a table into multiple tables based on columns. Commonly used to separate frequently accessed columns from less frequently accessed ones, improving query performance for the former.
4. Range Partitioning: Divides data based on a range of values in a specific column, useful for date-based partitions (e.g., monthly partitions).
5. Hash Partitioning: Distributes data across partitions using a hash function, ensuring an even distribution of data. Used when data distribution needs to be uniform.
6. List Partitioning: Divides data based on a predefined list of values, useful for categorizing data into distinct groups.