1. **Database**
   A Database is a collection of related data organised in a way that data can be easily accessed, managed and updated. Databases can be software based or hardware based, with one sole purpose, storing data.

2. **DBMS**
   A DBMS is a software that allows creation, definition and manipulation of databases, allowing users to store, process and analyse data easily. DBMS provides us with an interface or a tool, to perform various operations like creating a database, storing data in it, updating data, creating tables in the database.DBMS also provides protection and security to the databases. It also maintains data consistency in case of multiple users.

3. **Need for DBMS**
   - Processing queries and object management.
   - Controlling redundancy and inconsistency.
   - Efficient memory management and indexing.
   - Concurrency control and transaction management.
   - Access control and ease in accessing data.

4. **Keys**
   Super Key - The set of attributes which can uniquely identify a tuple is known as super key.
   Candidate Key - The minimal set of attributes which can uniquely identify a tuple is known as candidate key.
   Primary Key - There can be more than one candidate key in relation, out of which one can be chosen as primary key.
   Alternate Key - The candidate key other than primary key is known as alternate key.

5. **Transaction**
   A transaction can be defined as a group of tasks. It is a very small unit of a program and it may contain several low level tasks. A transaction must maintain ACID properties in order to ensure accuracy, completeness and data integrity.

   Atomicity(A) - This property states that a transaction must be treated as an atomic unit that is either all of its operations are executed or none. There must be no state in the database where a transaction is left partially completed. State should be defined either before the execution of the transaction or after the execution/abortion/failure of transaction.
   Consistency(C) - The database must remain in a consistent state after any transaction. No transaction should have any adverse effect on the data residing in the database.

Isolation(I) - In a database system where more than one transaction is being executed simultaneously and in parallel, the property of isolation states that all the transactions will be carried out and executed as if it is the only transaction in the system. No transaction will affect the existence of any other transaction.
Durability(D) - The database should be durable enough to hold all its latest updates even if the system fails or restarts. If a transaction commits but the system fails before the data could be written on disk, then the data will be updated once the system springs back into action.

6. **States of Transaction**
    - Active
    - Partially committed
    - Failed
    - Aborted
    - Committed

7. **Normalization**
   It is a technique of organizing the data in the database. Systematic approach of decomposing tables to eliminate data redundancy and undesirable characteristics like insertion,update and deletion anomalies.

   Problems without normalization
    - Extra memory space.
    - Difficult to handle and update the database without facing data loss.

8. **Normalization rule**
   Normalization rules are divided into the following normal forms
    - First Normal Form (1NF)
    - Second Normal Form (2NF)
    - Third Normal Form (3NF)
    - Boyce and Codd Normal Form (BCNF)
    - Fourth Normal Form (4NF)
    - Fifth Normal Form (5NF)

9. **First Normal Form (1NF)**

- Should have only single valued attributes/columns.
- Values stored in a column should be of the same domain.
- All the columns in a table should have unique names.
- Order in which data is stored does not matter.

Using the 1NF, data redundancy increases as there will be many columns with the same data in multiple rows but each row as a whole will be unique.

10. **Second Normal Form (2NF)**
    - Should be in 1NF.
    - No Partial dependency.

    Partial dependency means that a non-prime attribute is functionally dependent on part of a candidate key.

11. **Third Normal Form (3NF)**
    - Should be in 2NF.
    - Does not have transitive dependency.

    Transitive dependency is when a non-prime attribute depends on other non-prime attributes rather than depending upon the prime attribute.

    Advantage of removing transitive dependency

- Amount of data duplication is reduced.
- Data integrity is achieved.

12. **Boyce and Codd Normal Form (BCNF)**
    A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF.
    - It should be in 3NF.
    - For each functional dependency X->Y, X should be a super key.

13. **Fourth Normal Form (4NF)**
    - It should be in BCNF.
    - Should not have multi-valued dependency.

    Multi-valued dependency
    For a dependency A->B, if for a single value of A, multiple values of B exist, then the table may have multi-valued dependency. Table should have at least 3 columns and for a relation R(A,B,C) if there is a multi-valued dependency between A and B, then B and C should be independent of each other.

14. **Fifth Normal Form (5NF)**
    - Should be on 4NF.
    - Should not have join dependency.

15. **Different levels of abstraction in the DBMS**

    Physical Level: It is the lowest level of abstraction and describes how the data is stored.

Logical Level: This is the next level of abstraction after the Physical level. This layer determines what data is stored in the database, and what is the relationship between the data points.

View Level: The View Level is the highest level of abstraction and it describes only a part of the entire database.

16. Entity-relationship model

It is a diagrammatic approach to database design, where you represent real-world objects as entities and mention relationships between them.

17. Entity and Entity Set

Entity: An entity is a real-world object having attributes, which are nothing but characteristics of that particular object.
Entity Set: An entity set is the collection of all the entities of a particular entity type in a database.

18. Types of attributes

Simple attribute: The attributes with values that are atomic and cannot be broken down further are simple attributes.
Composite attribute: A composite attribute is made up of more than one simple attribute.
Derived attribute: These are the attributes which are not present in the whole database management system, but are derived using other attributes.
Single-valued attribute: As the name suggests, they have a single value.
Multi-valued attribute: And, they can have multiple values.

19. Difference between Unique key and Primary key

Unique key	Primary key
Unique Key can have a NULL value	The primary key cannot have a NULL value
Each table can have more than one unique key	Each table can have only one primary key

20. Relational Database Management System (RDBMS)

RDBMS is the most commonly used database. In RDBMS data is represented in terms of tuples(rows). It contains a number of tables and each table has its own primary key. Due to the collection of an organized set of tables, data can be accessed easily in RDBMS.

21. Difference between procedure and function

Procedure	Function
Used mainly to execute certain process	Used mainly to perform some calculation
RETURN will simply exit the control from sub program	RETURN will exit the control from sub program and also returns the value
Uses OUT parameter to return a value	Uses RETURN to return a value
Return data type will not be specified at the time of creation	Return data type is mandatory at the time of creation

22. Indexing

Indexing is a data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done.

Types of indexing

Primary Index − Primary index is defined on an ordered data file. The data file is ordered on a key field. The key field is generally the primary key of the relation.

Secondary Index − Secondary index may be generated from a field which is a candidate key and has a unique value in every record, or a non-key with duplicate values.

Clustering Index − Clustering index is defined on an ordered data file. The data file is ordered on a non-key field.

23. Hashing

Hashing is an effective technique to calculate the direct location of a data record on the disk without using index structure.Hashing uses hash functions with search keys as parameters to generate the address of a data record.

Types of hashing

Static Hashing - In static hashing, when a search-key value is provided, the hash function always computes the same address.

Dynamic Hashing - The problem with static hashing is that it does not expand or shrink dynamically as the size of the database grows or shrinks.

Dynamic hashing provides a mechanism in which data buckets are added and removed dynamically and on-demand. Dynamic hashing is also known as extended hashing.

24. Deadlock

Deadlock is a situation which occurs when two transactions wait on a resource which is locked or other transaction holds. Deadlocks can be prevented by making all the transactions acquire all the locks at the same instance of time. So, once a deadlock occurs, the only way to cure it is to abort one of the transactions and remove the partially completed work.

25. Differences between an exclusive lock and a shared lock

Exclusive lock	Shared lock
An exclusive lock is a lock on a data item when a transaction is about to perform the write operation.	A shared lock allows more than one transaction to read the data items.

SQL

1. SQL

Structured Query Language is the core of the relational database which is used for accessing and managing the databases. This language is used to manipulate and retrieve data from a structured data format in the form of tables and holds relationships between those tables. So, SQL is used to communicate with the database.

2. Sub-queries

A subquery is a query inside another query where a query is defined to retrieve data or information back from the database. In a subquery, the outer query is called the main query whereas the inner query is called subquery. Subqueries are always executed first and the result of the subquery is passed on to the main query. It can be nested inside a SELECT, UPDATE or any other query. A subquery can also use any comparison operators such as >,< or =.

3. **Data Definition Language(DDL)**

DDL changes the structure of the table like creating a table, deleting a table, altering a table.All the commands of DDL are auto-committed that means it permanently saves all the changes in the database.

DDL Commands

1. Create - It is used to create a new database or table.

```
CREATE TABLE TABLE_NAME (COLUMN1 DATATYPE, COLUMN2 DATATYPE,...);
```

2. Alter - It is used to alter the structure of the database like adding,renaming,dropping and changing the datatype of column.

```
ALTER TABLE TABLE_NAME ADD COLUMN_NAME DATATYPE;
```

3.Truncate - It is used to delete all the rows from the table and free the space containing the table.

```
TRUNCATE TABLE TABLE_NAME;
```

4. Drop - It is used to delete both the structure and record stored in the table.

```
DROP TABLE TABLE_NAME;
```

4. **Data Manipulation Language(DML)**

DML commands are used for manipulating the data stored in the table. It is responsible for all forms of changes in the database.The command of DML is not auto-committed, which means it can't permanently save all the changes in the database.They can be rolled back.

DML Commands

1. Insert - It is used to insert data into the row of a table.

```
INSERT INTO TABLE_NAME VALUES(VALUE1,VALUE2,VALUE3,...);
```

2. Update - This command is used to update or modify the value of a column in the table.

```
UPDATE TABLE_NAME SET COLUMN1=VALUE1,COLUMN2=VALUE2 WHERE
CONDITION;
```

3. Delete - It is used to remove one or more rows from a table.

```
DELETE FROM TABLE_NAME WHERE CONDITION;
```

5. **Transaction Control Language(TCL)**
   These commands are used to keep a check on other commands and their effect on the database.These commands can annul changes made by other commands by rolling the data back to its original state.It can also make any temporary changes permanent.

   TCL Commands
   1. Commit - Commit command is used to save all the transactions to the database.

```
COMMIT;
```

   2. Rollback - Rollback command is used to undo transactions that have not already been saved to the database.

```
ROLLBACK;
```

   3. Savepoint - It is used to roll the transaction back to a certain point without rolling back the entire transaction.

```
SAVEPOINT SAVEPOINT_NAME;
```

6. **Data Control Language(DCL)**
   DCL commands are used to grant and take back authority from any database user.

   DCL Commands
   1. Grant - It is used to give user access privileges to a database.
   2. Revoke - It is used to take back permissions from the user.

7. **Data Query Language(DQL)**
   DQL is used to fetch the data from the database.

   DQL Command
   1. Select - It is used to select the attribute based on the condition described by WHERE clause.

```
SELECT COLUMN1,COLUMN2 FROM TABLE_NAME;
```

8. **Difference between Drop, Truncate, Delete**

Drop	Truncate	Delete
Used to delete a database, table or a view	Used to delete all rows from a table	Used to delete a row in the table
Deletes the full structure of the table	Preserves the structure of the table	Deletes the structure of the row from a table

9. **Difference between WHERE and HAVING clause**

WHERE	HAVING
WHERE clause works on row's data and cannot be used with aggregates	HAVING clause works on aggregated data
Used only with SELECT statement	Used in a GROUP BY clause
It is called pre filter	It is called post filter

10. **LIKE clause**

LIKE clause is used in the condition in SQL query with the WHERE clause. LIKE clause compares data with an expression using wildcard operators to match patterns given in the condition.

There are two wildcard operators that are used in the LIKE clause.

- **Percent sign %:** represents zero, one or more than one character.
- **Underscore sign _:** represents only a single character.

11. **Order by clause**

Order by clause is used with a SELECT statement for arranging retrieved data in sorted order. The Order by clause by default sorts the retrieved data in ascending order. To sort the data in descending order DESC keyword is used with Order by clause.

```
SELECT column-list|* FROM table-name ORDER BY ASC | DESC;
```

12. **Group by clause**

Group by clause is used to group the results of a SELECT query based on one or more columns. It is also used with SQL functions to group the result from one or more tables.

13. **Having clause**

Having clause is used with SQL Queries to give more precise condition for a statement. It is used to mention condition in Group by based SQL queries, just like WHERE clause is used with SELECT query.

14. **Distinct**

The distinct keyword is used with the SELECT statement to retrieve unique values from the table. Distinct removes all the duplicate records while retrieving records from any table in the database.

```
SELECT DISTINCT column-name FROM table-name;
```

15. **SQL Joins**

A JOIN clause is used to combine rows from two or more tables, based on a related column between them. It is used to merge two tables or retrieve data from there.

16. **Cross Join or Cartesian Product**

This type of JOIN returns the cartesian product of rows from the tables in Join. It will return a table which consists of records which combine each row from the first table with each row of the second table.If there is no condition specified then it returns all possible pairing of rows from both the tables whether they are matched or unmatched.

```
SELECT column-name-list FROM table-name1 CROSS JOIN table-name2;
```

17. **Inner Join**

This is a simple JOIN in which the result is based on matched data as per the equality condition specified in the SQL query.

```
SELECT column-name-list FROM
table-name1 INNER JOIN table-name2
WHERE table-name1.column-name = table-name2.column-name;
```

18. **Natural Join**

Natural Join is a type of Inner join which is based on columns having the same name and same datatype present in both the tables to be joined.

```
SELECT * FROM table-name1 NATURAL JOIN table-name2;
```

19. **Outer Join**

Outer Join is based on both matched and unmatched data. Outer Joins subdivide further into,

Left Outer Join

The left outer join returns a resultset table with the matched data from the two tables and then the remaining rows of the left table and null from the right table's columns.

```
SELECT column-name-list FROM
table-name1 LEFT OUTER JOIN table-name2
ON table-name1.column-name = table-name2.column-name;
```

Right Outer Join

The right outer join returns a resultset table with the matched data from the two tables being joined, then the remaining rows of the right table and null for the remaining left table's columns.

```
SELECT column-name-list FROM
table-name1 RIGHT OUTER JOIN table-name2
ON table-name1.column-name = table-name2.column-name;
```

Full Outer Join

The full outer join returns a resultset table with the matched data of two tables then remaining rows of both left table and then the right table.

```
SELECT column-name-list FROM
table-name1 FULL OUTER JOIN table-name2
ON table-name1.column-name = table-name2.column-name;
```

20. Difference between Natural join and Inner Join

Natural Join	Inner Join
Natural Join joins two tables based on the same attribute name and datatypes.	Inner join joins two tables on the basis of the column which is explicitly specified in the ON clause.
The resulting table will contain all the attributes of both the tables but keep only one copy of each common column.	The resulting table will contain all the attributes of both the tables including duplicate columns also.

21. Difference between Inner Join and Outer Join

Inner Join	Outer Join
Returns combined tuple between two or more tables.	Returns the combined tuple from a specified table even when the join condition fails.
When attributes are not common then it will return nothing.	It does not depend upon the common attributes. If the attribute is blank then NULL is placed there.

22. Difference between Join and Union

Join	Union
Combines data from many tables based on a matched condition between them.	Combines the result set of two or more select statements.
It combines data into new columns.	It combines data into new rows.
Number of columns selected from each table may not be same.	Number of columns selected from each table should be same.
It may not return distinct columns.	It returns distinct rows.

23. Set Operations in SQL

SQL supports few Set operations which can be performed on the table data. These are used to get meaningful results from data stored in the table, under different special conditions.

The different Set Operations are

- Union
- Union All
- Minus
- Intersect

24. Sequence

Sequence is a feature supported by some database systems to produce unique values on demand. Some DBMS like MySQL support AUTO_INCREMENT in place of Sequence.

```
CREATE SEQUENCE sequence-name
START WITH initial-value
INCREMENT BY increment-value
MAXVALUE maximum-value
CYCLE | NOCYCLE;
```

25. View

A VIEW in SQL is a logical subset of data from one or more tables. View is used to restrict data access.

```
CREATE or REPLACE VIEW view_name
AS
SELECT column_name(s)
FROM table_name
WHERE condition
```

Types of views

- Simple
- Complex

26. SQL Alias

Alias is used to give an alias name to a table or a column, which can be a resultset table too. This is quite useful in case of large or complex queries. Alias is mainly used for giving a short alias name for a column or a table with complex names.

```
SELECT column-name FROM table-name AS alias-name
```

27. SQL Functions

SQL provides many built-in functions to perform operations on data. These functions are useful while performing mathematical calculations, string concatenations, sub-strings etc.

SQL functions are divided into two categories

- Aggregate function
- Scalar function

28. Aggregate Functions

These functions return a single value after performing calculations on a group of values.

Following are some Aggregate functions

- AVG()
- COUNT()
- FIRST()
- LAST()
- MAX()
- MIN()
- SUM()

29. Scalar Functions

Scalar functions return a single value from an input.

Following are some Aggregate functions

- UCASE()
- LCASE()
- MID()
- ROUND()

30. SQL Constraints

SQL Constraints are rules used to limit the type of data that can go into a table, to maintain the accuracy and integrity of the data inside the table.

Constraints can be divided into the following two types:

1. **Column level constraints:** Limits only column data.
2. **Table level constraints:** Limits whole table data.

Constraints are used to make sure that the integrity of data is maintained in the database. Following are the most used constraints that can be applied to a table.

1. NOT NULL
2. UNIQUE
3. PRIMARY KEY
4. FOREIGN KEY
5. CHECK
6. DEFAULT