Q:1 What is database ? white advantages and disadvantages of DBMS.

The database is a Collection of inter-related data which is used to retrieve insert and delete the data efficiency . It is also used to organize the data in the form of table scheme, views and reports etc . It is a Collection of related data.

Data base management System is a software which is used to mangge the data base Ex My SQL etc.

A database is a seperate application that stores a collection of data . Each database has one or more distinct APIs for Creating , accessing , managing , Searching, and replicating the data it holds.

Other kinds of data stores can also be used , such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.

ADVANTAGES OF DBMS

- Sharing of data
- Backup and recovery
- Support multiple users.
- Avoidance of inconsistency.
- Redundancy problem can be solved.
- Has a very high security level.
- presence of data integrity.
- Any unauthorized access is restricted.
- Easily maintenance

DISADVANTAGES OF DBMS

- Complexity
- Performance
- Higher impact of a failure
- Cost of DBMS
- Size.
- Techniqus Staff requirment

Q:2 Explain data Abstraction in detail.

Data abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

The database systems consists of complicated data structures and relations for users to access the data easily, these complications are kept hidden, and only the relevant part of the database is made accessible to the users through data abstraction.

LEVELS OF ABSTRACTION FOR DBMS

Database systems include complex data structures. In terms of retrieval of data, reduce complexity in terms of usability of users and in order to make the system efficient, developers use levels of abstraction that hide irrelevant details from the users. levels of abstraction simplify database design.

Mainly there are three levels of abstraction for DBMS, which are as follows :-

1. Physical or Internal level.
2. logical or Conceptual level.
3. View or external level.

1. PHYSICAL OR INTERNAL LEVEL

It is the lowest level of abstraction for DBMS which defines how the data is actually stored, it defines data - structured to stores data and access methods used by the database. Actually it is decided by developers or database application programmers how to store the data in the database.

2. LOGICAL OR CONCEPTUAL LEVEL

Logical level is the intermediate level or next higher level. It describes what data is stored in the database and what relationship exists among those data. It tries to describe the entire or whole data because it describe what tables to be created

and what are the links among those tables that are Created. It is less Complex than the physical level. Logical level is used by developers or database administrator (DBA).

3. VIEW OR EXTERNAL LEVEL

It is the highest level. In view level, there are different levels of views and every view only defines a part of the entire data. It also simplifies interaction with the user and it provides many views or multiple views of the same database.

View level can be used by all users (all levels' users). This level is the least Complex and easy to understand.

Q:3 Explain —

1) TUPLE

A tuple also known as a record or row, is a basic unit of data in a relational database management system (DBMS). A tuple represents a single instance of a relation, or table, in the database. Each tuple contains a set of values, or attributes, that correspond to the columns, or fields or attributes, that correspond to the columns, of the relation.

A tuple in a database management system is one record in the context of relational database. You can compare the data present in database with a spreadsheet, with rows (known as tuples) and columns (known as attributes) representing various data types.

In DBMS, a unique key is assigned to each table that is used to organize and identify the elements.

2) ATTRIBUTE

Attributes are an important Component of database management systems (DBMS). They refers to the specific characteristics or properties of an entity that define its identity. An entity can be any object or Conceptual ; it identifies as an independent unit, such as a person, place or thing.

TYPES OF ATTRIBUTES

1. SIMPLE ATTRIBUTE

Simple attributes are single-valued traits that cannot be further split. In a student database, for example, the student's name can be simple attribute.

2. COMPOSITE ATTRIBUTE

Composite qualities can be further subdivided into smaller sub-parts. A student's address, for example, can be further subdivided into street, city, state, and zip code.

3. KEY ATTRIBUTES

Key attributes are those that uniquely identify a database entity. In a student database for example, the roll number can be the key property that uniquely identifies each student.

4. MULTIVALUED ATTRIBUTES

Multivalued characteristics can have numerous values for the same entity. A student, for example, may have many phone numbers or email addresses.

5. DERIVED ATTRIBUTES

Derived attributes are qualities that originate from other attributes in the database. The data of birth features, for example can be used to calculate a student's age.

3. SCHEMA

A Schema in SQL is a Collection of database objects associated with a database. The username of a database is called a Schema owner (owner of logically grouped structures of data). Schema always belongs to a single database whereas a database can have single or multiple schemas. Also, it is also very similar to separate namespaces or Containers, which stores database objects. It includes various database objects including your tables, views, procedures, index, etc.

ADVANTAGES OF USING SCHEMA

- A single Schema can be used in multiple data bases.
- The objects created in the database can be moved among schemas.
- The schema also help in adding security.

4. DOMAIN

A domain is essentially a datatype with optional constraints (restriction on the allowed set of values). The user who defines a domain becomes its owner. If a schema name is given (for example, CREATE DOMAIN myschema.mydomain) then the domain is created in the specified schema.

A domain type is a property of domain that determines which assets can be included in the domain, based the asset's type. Some asset type can be only created in a specific Collibra application and in a specific domain type.

Q:4 Explain DDL, DCL, DML and TCL with examples.

DDL {Data definition language}

DDL stands for data definition language. DDL changes the structure in the database like creating a table, deleting a table, altering a table etc. All the commands of DDL are automa committed that means it permanently save all the changes in the database.

DDL actually consists of SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database object in the database.

- CREATE TABLE :- used to create a new table.
- ALTER TABLE :- used to modify an existing table's structure.
- DROP TABLE :- Used to delete a table.
- CREATE INDEX :- Used to create an index on a table, improving query performance.

DML {Data manipulation language}

DML Commands are used to modify the database. DML stands for data manipulation language. DML Command is not auto-committed it means it can't permanently saves all the changes in the data base.

Data manipulation language are used to retrieve, insert, update, and delete data in the database. Common DML Commands include : DML Commands are essential for managing the data stored in a database.

- SELECT :- Used to retrieve data from one or more tables.
- INSERT :- Used to add new records to a table.
- UPDATE :- Used to modify existing records in a table.
- DELETE :- Used to remove records from a table.

DCL { Data Control language }

DCL Commands are used to manage database security and access control. The two primary DCL Commands are :

- Grant : Used to grant specific privileges to database users or roles.
- Revoke : Used to revoke previously granted privileges.

DCL Commands ensure that only authorized users can access and modify the database.

TCL { Transaction Control language }

TCL Commands are used to manage database transactions, ensuring data integrity. Key TCL Commands include :

- COMMIT – Commits a transaction, saving changes permanently.
- ROLLBACK – Undoes changes made during a transaction.
- SAVEPOINT – Sets a point within a transaction to which you can later roll back.

Q: 5 What are Keys? Explain PRIMARY KEY, FOREIGN KEY, ALTERNATE KEY, COMPOSITE KEY, CANDIDATE KEY.

A key is an attribute or a set of attributes that help to uniquely identify a tuple (or row) in a relation (or table). Key are also used to establish relationships between the different tables and columns of relational database. Individual values in a key are called key value.

1. PRIMARY KEY

A primary key is one of the Candidate key choosen by the database designer to uniquely identify the row in relation. The value of primary key can never be null. value of primary key can never be duplicate. it can identify one tuple (row) at a time.

An entity can Contain multiple keys, as we saw in the PERSON table. The key which is most suitable from those lists becomes a primary key.

2. CANDIDATE KEY

A Candidate key is an attribute or set of attributes which can uniquely identify a row. Candidate key is a minimal of super key. It most contain unique value. Every table must have atleast a single Candidate key.

Except for the primary key, the remaining attributes are considered a Candidate key. The Candidate keys are as strong as the primary key.

3. COMPOSITE KEY

A key that has more than one attribute is known as Composite key. It is also known as Compound key and Concatenated key.

The Combination of Columns guarantees uniqueness, though individual uniqueness is not guaranted.

4. FOREIGN KEY

Foreign key is used to link two tables together. A foreign key is the column whose value are the same as the primary key of another value, table.

Every employee works in a specific department in a company and employee and department are two different entities. So we can't store the department's information in the employee table. That's why we link these two tables through the primary key of one table.

5. ALTERNATE KEY

There may be one or attributes or a combination of attributes that uniquely identify each tuple in a relation. These attributes or combinations of the attributes are called the candidate keys. One key is choosen as the primary key from these candidate keys, and the remaining primary key, if it exists, is termed the alternate key.

Q:6 What is Joins? Explain its different types with suitable example.

Join means to Combine Something. In Case of SQL. Join means to Combine two or more tables. SQL Join clause takes records from two or more tables in a database and Combines it together. SQL Join statement is used to Combine data or row two or more tables based on the Common field btw the tables. It is denoted by ⋈.

TYPES OF JOIN

1. INNER JOIN :— Returns records that have matching values in both tables.

Example – Consider two tables, "Employees" and "Departments", —

Employees

Emp ID	Name	Dept ID
1	Alice	100
2	Bob	101
3	Charlie	102

Departments

Dept ID	Dept Name
100	sales
101	marketing

An inner join btw these tables will result in a table showing only the records where there's a match btw the "Dept ID" in both tables:

```
Sql
SELECT Employees.Name, Departments.DeptName
FROM Employees
INNER JOIN Departments ON Employees.Dept ID =
  Departments.Dept IR ;
```

2. LEFT JOIN (or outer join)

Returns all records from the left table and matched records from the right table.

Example: Continuing with the same tables, but using a left join.

```
Sql
SELECT Employees.Name, Departments.DeptName
FROM
LEFT JOIN Departments ON Employees.Dept ID = Departme-
  nts.Dept ID ;
```

Result

markdown

Name	DeptName
Alice	Sales
Bob	Marketing
Charlie	Sales

3. RIGHT JOIN (OR RIGHT OUTER JOIN):

Returns all records from the right table and the matched records from the left table.

Example –

Sql

```
SELECT Employees.Name, Departments.DeptName
FROM Employees
RIGHT JOIN Departments ON Employees.DeptID =
    Departments.DeptID;
```

Result

Name	DeptName
Alice	Sales
Bob	Marketing
NULL	NULL // Department record without a matching employee.

Q:7 Write an SQL query to be Create the table 'Employee' with the following structure :

Name	type
Empno	Int
Ename	Varchar
Job	Varchar
Sal	Int

→ Create the 'employee' table :

```
CREATE TABLE Employee (
    Empno   INT,
    Ename   VARCHAR (10),
    Job     VARCHAR (10),
    Sal     INT
);
```

a) Add a Column 'Commmission' with domain to the 'Employee' table :

```
ALTER TABLE Employee
ADD Commission DECIMAL (10,2);
```

b) Insert any five records into the tables :

```
INSERT INTO EMPLOYEE (Empno, Ename, Job,
```

```
Sal , Comission)
VALUES (101, 'John' , 'Manager' , 60000 , 0.05),
       102, 'Jane' , Developer, 50000, 0.03),
       103, 'Alice' , 'Analyst', 45000, 0.02),
       104, 'Bob' , 'Designer' , 48000 , 0.04),
       105, 'Eva' , 'Marketing' , 55000 , 0.06);
```

c) Update the Column details of the 'Job'.

Column :

```
ALTER TABLE Employee
CHANGE COLUMN Job position VARCHAR (15);
```

d) Rename the Column of the Employee table using the ALTER Command :

```
ALTER TABLE Employee
CHANGE COLUMN Ename EmployeeName VARCHAR (10);
```

e) Delete the employee whose Empno is 105 :

```
DELETE FROM Employee
WHERE Empno = 105;
```

Q:8 Write and explain the structure of SQL SELECT statement with where clause, GROUP BY, and HAVING by suitable example.

The SQL 'SELECT' statement is used to retrieve data from a database. When combined with the 'WHERE' clause, 'GROUP BY', and 'HAVING', it becomes more powerful for filtering, grouping, and applying conditions to the data being retrieved.

SELECT Statement with WHERE Clause:

The 'WHERE' clause filters rows based on a specified condition.

Example:

```
SELECT * FROM Employees
WHERE Department = 'Sales';
```

SELECT Statement with GROUP BY:

The 'GROUP BY' clause is used to group rows that have the same values into summary rows, often to perform aggregate functions on them.

A4 NOTE BOOK

```
SELECT Department, AVG(salary) AS Avgsalary
FROM Employees
GROUP BY Department;
```

This query groups the data by 'Department' and calculates the average salary ('AVG(salary)') for each department.

SELECT Statement with HAVING:

The 'HAVING' clause filters data grouped by the 'GROUP BY' clause based on specified conditions.

Example:

```
SELECT Department, AVG(salary) AS Avgsalary
FROM Employees
GROUP BY Department
HAVING AVG(salary) > 50000;
```

This query first groups the data by 'Department' Calculates the average salary ('AVG(salary)') and then filters out only those groups ('Department') where the average salary is greater than 50000.

Q:9 What do you mean by aggregate function in SQL ? Explain AVG(), MIN(), MAX(), COUNT(), SUM() with suitable example.

My SQL's aggregate function is used to perform calculations on multiple values and return the result in a single value like the average of all values, the sum of all values, and the maximum and minimum of values among certain groups of values.

We mostly use the aggregate function with select statements in the data query language (DQL).

AVG ()

It returns the average value of an expression.

Syntax – Select AVG (aggregate . expression) from table . name WHERE Condition ;

MIN ()

It returns the minimum value in a set.

Syntax – select MIN (DISTINCT aggregate . expression) from table . name WHERE Condition ;

MAX ()

It returns the maximum value in a set.

Syntax – Select Max (Distinct aggregate. expression) From table. name WHERE Condition ;

COUNT ()

It returns the no. of rows, including rows with null value in a group.

Syntax – select Count table. name WHERE Condition.

SUM ()

It returns the total Summed values (Non - null) in a set.

Syntax – Select Sum (aggregate . expression) from table. name WHERE Conditions;

Q:10 Write SQL query for following Consider table.
Emp (empno, deptno, ename, salary, Designation, joiningdate, DoB, city)

i) Display names of employees whose experience is more than 10 years:

```
SELECT ename
FROM EMP
WHERE DATE_DIFF (CURRENT_DATE(), joiningdate,
    YEAR) > 10;
```

ii) Display age of employees:

```
SELECT ename, TIMESTAMPDIFF (YEAR, DOB,
    CURRENT_DATE()) AS age
FROM EMP;
```

iii) Display average salary of all employee.

```
SELECT AVG (salary) AS average_salary
FROM EMP;
```

iv) Display name of employee who earned the highest salary:

```
SELECT ename
FROM EMP
WHERE salary = (SELECT MAX (salary) FROM EMP;
```