RDBMS:

RDBMS stands for Relational Database Management System. It is a type of database management system (DBMS) that is based on the relational model of data. The relational model organizes data into tables, which consist of rows and columns. Each row represents a unique record, and each column represents a specific attribute or field of that record.

RDBMS provides a structured approach to storing, managing, and retrieving data. It uses a set of rules, known as ACID properties (Atomicity, Consistency, Isolation, Durability), to ensure data integrity, consistency, and reliability. Some popular examples of RDBMS are Oracle Database, MySQL, Microsoft SQL Server, PostgreSQL, and IBM DB2.

RDBMS allows users to perform various operations on the data, such as inserting, updating, deleting, and querying records using a standardized language called Structured Query Language (SQL). SQL provides a powerful and flexible way to interact with the data stored in the RDBMS.

Overall, RDBMS is widely used in businesses and organizations for managing structured data, supporting transactions, enforcing data integrity, and providing a high level of scalability and performance.

Difference between DBMS and RDBMS ?

DBMS (Database Management System) and RDBMS (Relational Database Management System) are both software systems designed to manage databases, but they have some key differences:

Type	**DBMS**	**RDBMS**
Data Model	A DBMS is a general term for any software system that facilitates the creation, maintenance, and manipulation of databases. It can support different data models, such as hierarchical, network, or object-oriented, in addition to the relational model.	RDBMS, on the other hand, specifically refers to a type of DBMS that is based on the relational model. It organizes data into tables with rows and columns, and it enforces relationships between tables using primary keys and foreign keys.
Data Structure		
Data Integrity		
Query Language		

DDL(Data Definition Language)

It is a subset of SQL (Structured Query Language) used to define and manage the structure of a database and its objects. DDL allows users to create, modify, and delete database objects such as tables, indexes, views, and more. Its primary purpose is to specify the database schema, which defines the logical and physical layout of the data.

1. CREATE : Used to create new database objects like tables, indexes, views, and schemas.

Example:
```
Create table student
(
 Id number,
 Name Varchar(20),
 Age  number
);
```

2.ALTER : Used to modify the structure of existing database objects, such as adding or dropping columns from a table.

Example : Altering a table to add a new column:
```
ALTER TABLE EMPLOYEE
ADD COLUMN hire_date  date;
```

Example: Altering a table to delete a column:
```
ALTER TABLE EMPLOYEE
DROP COLUMN hire_date;
```

3. DROP : Used to remove database objects like tables or views from the database.

Example: **DROP TABLE STUDENT;**

4. TRUNCATE: Used to remove all data from a table but keeps the table structure intact.

Example: **TRUNCATE TABLE STUDENT;**

Note: Its is faster than delete.

5. COMMENT : Used to add comments to the data dictionary or schema to provide information about the database objects.

Example: To add a comment to a table.

COMMENT ON TABLE table_name IS 'This is a comment about the table.';

Example: To add a comment to a specific column in a table:

COMMENT ON COLUMN table_name.column_name IS 'This is a comment about the column.';

6. RENAME: Used to rename database objects.

Example :

ALTER TABLE table_name RENAME COLUMN current_column_name TO new_column_name;

It is a subset of SQL (Structured Query Language) used to manipulate and work with the data stored in a database. Unlike DDL (Data Definition Language), which focuses on defining and managing the database structure, DML is concerned with performing operations on the data itself.
There are 3 DML statements in SQL.
1. Insert : it adds rows to a table.
Example: **Insert into Student values (21,'Rahul',25);**
2. Delete : Removes Rows From table.
Example: **Delete from Employee Where id='A001' and name="Rahul;**
3.Update : Changes Column values in Table.
Example: **Update Department**
Set department_name='cse'
Where department_id=10;

Modify:
Use to modify the size of data type and increase data size.
Example:
Alter Table Table_Name
Modify(Column_Name DataType(?));

Primary key ?
A primary key is a unique and non-null attribute or combination of attributes within a relational database table.Its main purpose is to uniquely identify each record(row) in the table,ensuring that there are no duplicate values for the primary key attribute and every row has a unique identifier.
Note: Any Key or Details which is unique,not null is called primary key.
Example:
Table Account

Foreign Key?
A foreign key is an attribute or combination of attributes in a table that references the primary key of another table.
Note: Any key or Details refers to another table or details that called Foreign key.
Example:
Table Loan.

What is capabilities of select statement?
Retrieving the data from the the hard disk of the computer can be done in two ways.

1. **(1) Projection**
2. **(2) Selection**

(1) Projection: Projection is a process of displaying the result using project queries.
Note: Project queries are all those queries which is used to display(Project) particular columns to the user based on the requirement from the user.

USN	AGE
1	21
2	21
3	21
4	21

(2) Selection: Selection is process of displaying the result using result queries.
Note: Select queries are those queries which displaying particular rows to the user Base on specific conditions.

USN	AGE
3	21

OPERATOR AS SYMBOLS

(1) ARITHMATIC OPERATOR

Operator	Precedence
*	1
/	2
+	3

-	4

Question1: WAQ to display the last_name and the salary by incrementing the salary by 1000 every employee in the company ?
Question2: WAQ to display the first_name and salary by reducing the salary by 2000 every employee in the company ?
Question3: WAQ to display the first_name and annual salary for all employees?

COLUMN ALIASES IN SQL

Aliases are the other names given to columns in SQL.
The alias names will never be reflected in the actual tables present on the hard disk of the computer.

```
                Column_name as aliasname
                            Or
                Column_name as “Aliasname”
Example: Select first_name, salary*12/2 as halfyearlySalary
                 From Employee;
```

CONCATENATION OPERATOR IN SQL

The operator || is the concatenation operator in Sql
Concatenation operator is used to combine multiple data or multiple columns.

```
Example: Select first_name || last_name from Employee;
                      Or
               Select first_name|| last_name as fullname from employee;
```

Example:
Question :Write a query in order to display the following output by ascending the data from Employee.
Salary details:
Sachin gets 24000 as Salary
Vinod gets 17000 as salary
::::::::::::::::::::::
Solution: Select first_name||’ ‘||’gets’||’ ‘||salary||’ ‘|| as salary as Salarydetails from Employee;
Question :Write a query to concatennate the data ‘Rahul’ and ‘Dravid’.

```
Solution: Select ‘Rahul’||’Dravid’
                 From Employee;
```

Getting Description of table in Oracle

In order to get description of a table(,view,synonym,package etc.) in sql we have to use the command desc

Syntax: desc table_name;

Example: desc Account;

Output:

Table	Column_name	Data Type	Length	Precision	Scale	Primary Key	Null able	Def ualt	Com ment
Account	Acc_no	Number							
	Name	Varchar							
	City	Varchar							
	Balance	Number							
	loan_taken	Varchar							

Relational Operators in SQL

Operator	Precedence
=	1
>	2
<	3
>=	4
<=	5
!=	6
< >	6
^=	6

Question: WAQ to display the rows from employees table for all the employees whose salary is greater than 18500.

Question: WAQ to display the last_name,hire_date,job_id of all employees whose salary is not equal to 24000.
Question: WAQ to display the last_name,hire_date,job_id of all employees whose salary is greater than equal to 24000.

Operators as keywords

DISTINCT KEYWORD

NOTE 1: Distinct keyword is used to avoid displaying of repeated values.
NOTE 2: Distinct keyword is not dependent on the datatype of the column.
Example:
```
Select Distinct column_name
       From Table_name;
```
Wrong Example:
```
Select Unique Column_name
       From Table_name;
```
Example:
```
Select distinct acc_no from Account;
```

BETWEEN AND OPERATOR

When the condition has a range of values to be compared we should be using BETWEEN AND operator.
Question: WAQ to display last_name,salary from employees whose salary is in between 24000 and 18000.
Solution:
```
Select last_name,salary
       from employee
       where salary BETWEEN 18000 AND 24000 ;
```
Question: WAQ to display last_name,salary from employees whose salary is not in range 24000 and 18000.
Solution:
```
Select last_name,salary
      From Employee
      Where salary NOT BETWEEN 18000 AND 24000 ;
```

IN OPERATOR

When ever comparison has to be done with respect to set of values we have to use the operator IN .
Question: WAQ to display the first_name,salary for all employees who are getting 18500,19500,24000 as the salary.
Solution:
```
     Select first_name,salary
     From employee
     Where salary IN( 18500,19500,24000);
```

Question: WAQ to display the first_name,salary for all employees who are not getting 18500,19500,24000 as the salary.
Solution:
```
Select first_name,salary
From employee
Where salary NOT IN (18500,19500,24000);
```

LIKE KEYWORD

The LIKE operator is used to search for a specified pattern in a column.
Note: Pattern matching in SQL.
There are two pattern matching symbols in sql.
(1) % (modulus) it matches 0 or more characters.
(2) _ (underscore) it matches exactly one character.

Question(1) Write a query to retrieve the first_name whose last_name starts with 'Ku'.
Solution:
```
Select first_name
From Employee
Where last_name like 'Ku%' ;
```
Question(2) Write a query to display the first_name,last_name whose last_name last two characters are 'ms' .
Solution:
```
Select first_name,last_name
From Employee
Where last_name like '%ms' ;
```
Question(3) WAQ to display all the details of the employee whose last_name contains as substring 'as' .
Solution:
```
Select *
From Employee
Where Last_name like ='%as%' ;
```
Question(4) WAQ to display the first_name and last_name where the last_names third character is 'n' .
Solution:
```
Select first_name, last_name
From Employee
Where last_name='_ _n%' ;
```
Question(5) WAQ to display first_name and Last_name where first_name second and the last character are 'a' .
Solution:
```
Select first_name,Last_name
From Employee
Where first_name like '_a%a' ;
```

DUAL TABLE

The DUAL table is a special one-row,one column table present by default in oracle and other database installations.
In Oracle, the table has a single VARCHAR2(1) column called DUMMY that has a value of 'X'
Note1: if the query "desc dual" is executed the following will be output.

Table	Column	Data type	Length	Precision	Scale	PK	Nullable	Defualt	comment
DAUL	DUMMY	Varchar2	1						

Note2: if the query "Select * from dua" is executed then following will be the output

DUMMY
X

IS NULL OPERATOR

It is used to select only the records with NULL values in the column.

```
Example: Select commission_pct
             From employees
             Where commission_pct is null;

Wrong:  select commission_pct
           From Employee
           Where commission_pct=null;
Wrong:  Where commission_pct not is null;
Correct:  Where commission_pct is not null;
```

FUNCTION IN SQL

In SQL we have two types functions.

1. **Single row functions**
2. **Multiple row functions**

(1) **Single row functions:** **Single row functions are such functions which will accept a single row as input(or) it accept multiple row as input but produces one result per row.**

(i) CHARACTER function
(ii) GENERAL function
(iii) NUMBER function
(iv) DATE function
(v)CONVERSION function

(2) **Multiple row functions:** **Multiple row functions/group functions/aggregate functions are such functions which will accept a single row or multiple row as input but produces one result per group.**

(i) AVG( )
(ii) COUNT ( )
(iii) MAX ( )
(iv) MIN ( )
(v) SUM ( )

Single row function

(i) CHARACTER function: Accepts character input and returns number or character value.

Case Manipulation function : Upper( ) , Lower( ) , InitCap( )

Character Manipulation function : Concat( ) , Substr ( ), instr( ) , length( ) , Trim( ) Replace( )

Note: UPPER function converts a string to upper case.
LOWER function converts a string to lower character.
INITCAP function converts only the initial alphabets of a string to upper case.

Note: CONCAT function concatenates two string values.
SUBSTR function returns a portion of a string from a given start point to an end point.
INSTR function returns numeric position of a character or a string in a given string.
LENGTH function returns the length of the input string.
TRIM function trims the string input from the start or end.
REPLACE function replaces characters from the input string with a given character.

Note:Clause in SQL is a built-in function that is used to retrieve the data from the records present in the database. Different clauses in SQL are used to fetch or retrieve the records from the database table. The ORDER BY clause in SQL is used to arrange the retrieved results in ascending order or descending order.

THE GROUP BY CLAUSE

The SQL GROUP BY clause is used by collaboration with SELECT statement to arrange the identical data into group.
The GROUP BY clause follow the WHERE clause in a SELECT statement and precedes the ORDER BY clause.
Example: Write a query to display the department_id and the sum of salary from the all employee in each department_id.

```
Wrong: select department_id, sum(salary)
            From Employee;
  Output: Wrong
Correct: select department_id,sum(salary)
              From Employee
              Group By department_id;
```

Example:
Write a query to display the department_id and the least of salary from the all employee in each department_id.
solution:

```
Select department_id,min(salary)
From employee
Group by depatment_id;
```

Example:
Display the Department_id and heigest salary of each department for all departments whose department_id is greater than 50.

```
SELECT Department_id,max(salary)
From employee
Where department_id>50
Group by department_id;
```

THE HAVING CLAUSE

The HAVING clause was added to SQL because the WHERE keyword could not be used where In aggregate functions.

Example:

Write a query to display the department_id and the maximum salary for all the employee where maximum salary is greater than 20000. ?

```
SELECT DEPARTMENT_ID,MAX(SALARY)
FROM EMPLOYEE
HAVING MAX(SALARY)>20000
GROUP BY DEPARTMENT_ID;
```

THE ORDER BY CLAUSE

The output of the sql query can be predicted but the order in which the rows gets displayed cannot be predicted.

If we have to print the output in a specific order we should be using order by clause.

Syntax: Order by column_name Asc/Desc;

Example: Write a query to to display all the last_name of employees in ascending order.

```
Solution:  Select last_name
           From Employee
           Order by Last_name asc;
```

```
Note:  Select     // Column_Name
       From     // Table_name
       Where   // Condition on columns
       Having  // Condition on group functions
       Group By // Column_name
       Order By // column_name [asc/desc]
```

Question: Write a query to display department_id and maximum salary of all employees whose department_id is greater than 10 and having maximum salary greater than 20000 for each department . while displaying display the data in descending order with respect to department_id.

```
Solution: Select department_id , max(salary)
          From Employee
          Where department_id=10
           Having max(salary)>20000
           Group By department_id
            Order by department_id desc;
```

SUB QUERIES IN SQL:

A Subquery or Inner query or Nested query is a query within another SQL query and embedded within the WHERE clause.

A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved.

Properties of subqueries:

- **Subqueries must be enclosed within parentheses.**
- **A subquery can have only one column in the SELECT clause,unless multiple columns are in the main query for the subquery to compare its selected columns.**
- **An ORDER BY cannot be used in a subquery, although the main query can use an ORDER BY. The GROUP BY can be used to perform the same function as the ORDER BY in a subquery.**
- **Subqueries that return more than one row can only be used with multiple value operators,such as the IN operator.**

Question 1: WAQ to display the last_name, first_name and salary of all employees who earn more than 'tendulkar' ?

Solution: SELECT LAST_NAME,FIRST_NAME
FROM EMPLOYEES
WHERE SALARY > (SELECT SALARY FROM EMPLOYEE WHERE LAST_NAME='Tendulkar');

Question 2: WAQ a query to display the department_id and first_name for all employees who work in the same department in which 'Vinod' works ?

Solution:

```
 SELECT DEPARMENT_ID, FIRST_NAME
 FROM EMPLOYEES
WHERE DEPARTMENT_ID = ( SELECT DEPARTMENT_ID FROM EMPLOYEES
WHERE FIRST_NAME='Vinod');
```

Question 3: WAQ to display the last_name and job_id whose job_id is similar to 'King' and whose salary is greater than singh salary ?

Solution:

SELECT LAST_NAME, JOB_ID
FROM EMPLOYEES
WHERE JOB_ID = (SELECT JOB_ID FROM EMPLOYEES WHERE LAST_NAME='King') AND SALARY>(SELECT SALARY FROM EMPLOYEES WHERE LAST_NAME='Singh');

Question 4: WAQ to display the employee_id of all the employees whose department_id in employees table is equal to department_id department table ?

Solution:

SELECT EMPLOYEE_ID
FROM EMPLOYEE
WHERE DEPARTMENT_ID = (SELECT DEPARTMENT_ID FROM DEPRTMENT) // cannot handle more than 1 value.

(OR)

```
SELECT EMPLOYEE_ID
FROM EMPLOYEES
WHERE DEPARTMENT_ID IN ( SELECT DEPARTMENT_ID FROM DEPARTMENT)
```

JOIN