Name- Mayuri Siddhu Bandgar

Batch No- 7669

Enroll Id- EBE0NO722114519

What is DBMS and its importance?

A database management system is **a software tool used to create and manage one or more databases, offering an easy way to create a database, update tables, retrieve information, and enhance data**. A DBMS is where data is accessed, modified and locked to prevent conflicts

SQL Commands

SQL- SQL means-structured Query Language. The SQL commands help in creating and managing the database.
There are 5 Types of SQL commands

1.DDL	2.DML	3.DQL	4.DCL	5.TCL
Create	Insert	Select	Grant	Commit
Drop	Update		Revoke	Roll Back
Alter	Delete			Save point
Truncate	Lock			Set transaction
Comment	Call			
Rename				

The most common SQL commands which are highly used are mentioned

1. CREATE command
2. UPDATE command
3. DELETE command
4. SELECT command
5. DROP command
6. INSERT command

CREATE Command

This command helps in creating the new database, new table, table view, and other objects of the database.

UPDATE Command

This command helps in updating or changing the stored data in the database.

DELETE Command

This command helps in removing or erasing the saved records from the database tables. It erases single or multiple tuples from the tables of the database.

SELECT Command

This command helps in accessing the single or multiple rows from one or multiple tables of the database. We can also use this command with the WHERE clause.

DROP Command

This command helps in deleting the entire table, table view, and other objects from the database.

INSERT Command

This command helps in inserting the data or records into the database tables. We can easily insert the records in single as well as multiple rows of the table.

1.DDL-Data Definition Language

Create-SQL CREATE TABLE statement is used to create table in a database.

Syntax- create table table_name (“column1”,data type, ”column2”,data type,.....”column” data type);

Example -SQL> create table monitor(id int not null,name varchar(20),age int,address varchar(20));

Table created.

SQL> desc monitor;

Name Null? Type

---------- -------- ----------

ID NOT NULL NUMBER(38)

NAME VARCHAR2(20)

AGE NUMBER(38)

ADDRESS VARCHAR2(20)

Drop Table- A SQL DROP TABLE statement is used to delete a table definition and all data from a table.

Syntax-Drop table table_name;

Example-SQL> drop table monitor;

Table dropped.

Alter Table-The Alter table statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

Alter table-ADD Columns

Syntax-Alter table table_name ADD column_name datatype;

Example-alter table monitor add email varchar(20);

Table altered.

SQL> desc monitor;

Name	Null?	Type
ID	NOT NULL	NUMBER(38)
NAME		VARCHAR2(20)
AGE		NUMBER(38)
ADDRESS		VARCHAR2(20)
EMAIL		VARCHAR2(20)

Alter table-DROP Columns

Syntax-alter table table_name DROP column column_name;

Example-

SQL> alter table monitor drop column age;

Table altered.

SQL> desc monitor;

Name	Null?	Type
ID	NOT NULL	NUMBER(38)
NAME		VARCHAR2(20)
ADDRESS		VARCHAR2(20)
EMAIL		VARCHAR2(20)

Truncate_Table-A truncate SQL statement is used to remove all rows (complete data) from a table. It is similar to the DELETE statement with no WHERE clause.

Example-SQL> truncate table s2;

Table truncated.

SQL> select *from s2;

no rows selected

Copy_Table-If you want to copy the data of one SQL table into another SQL table in the same SQL server, then it is possible by using the SELECT INTO statement in SQL.

2.DML- Data Manipulation Language

Insert -The INSERT INTO statement is used to insert new records in a table.

There are Two ways-

1. 1 Specify both the column names and the values to be inserted

Syntax-insert into table_name(column1,column2,column3...)values(value1,value2,value3...)

Example-

create table student1(id int,name varchar(20),age int);

Table created.

SQL> insert into student1(id,name,age) values ('101','rakhi','26');

1 row created.

SQL> select * from student1;

ID	NAME	AGE
101	rakhi	26

2. If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table.

   Syntax-insert into table_name values (value1,value2,value3....)

   Example-insert into student1 values(102,'sami',28);

1 row created.

SQL> select * from student1;

ID	NAME	AGE
101	rakhi	26
102	sami	28

Update-The UPDATE statement is used to modify the existing records in a table.

Syntax- Update table_name set column1=value1,column2=value2...

Where condition;

Example-

update student1 set name='komu',age='30' where id=102;

1 row updated.

SQL> select * from student1;

ID	NAME	AGE
101	rakhi	26
102	komu	30

Delete Table -The DELETE statement is used to delete rows from a table. If you want to remove a specific row from a table you should use WHERE condition. But they have it does not use Where condition then all rows are delete.

Syntax- Delete from table_name[where condition];

Example-select *from s2;

STUD_ID	NAME	ROLL_NO	M1_MARKS	M2_MARKS	M3_MARKS
110	samu	250	89	63	64
111	mamata	2	96	58	76
112	samata	5	69	89	68
113	mono	7	98	87	76
114	hari	10	96	68	87
115	nikita	25	63	65	45
116	manasi	27	65	96	98

7 rows selected.

SQL> delete from s2 where name='mamata';

1 row deleted.

SQL> select *from s2;

STUD_ID	NAME	ROLL_NO	M1_MARKS	M2_MARKS	M3_MARKS
110	samu	250	89	63	64
112	samata	5	69	89	68
113	mono	7	98	87	76
114	hari	10	96	68	87
115	nikita	25	63	65	45
116	manasi	27	65	96	98

6 rows selected.

3.DQL- Data Query Language

Select-The SELECT statement is used to select data from a database.

Syntax-select column1,column2,.....from table_name;

Here, column1, column2, ... are the field names of the table you want to select data from. If you want to select all the fields available in the table, use the following syntax:

Select * from table_name

Example-select name,age from student1;

NAME	AGE
rakhi	26
komu	30

SQL> select * from student1;

ID	NAME	AGE
101	rakhi	26
102	komu	30

4.DCL- Data Control Language

Grant-GRANT command is used to give access privileges to the users or other rights or opportunities for the database. This command also allows users to grant permissions to other users too.

Revoke-The REVOKE command does just opposite to the GRANT command. It withdraws user privileges on database objects.

Temp variable-temporary variable

1.Global temp value-Temporary tables generally contain all of the features that ordinary tables have like triggers, join cardinality, information about rows and block etc. the main difference is that the temporary tables can't have foreign keys related to other tables.

Syntax-

Createglobaltemporarytabletable_name(column1datatype[null/notnull],column2[null/notnull],....columnN datatype[null/notnull]);

Example -create global temporary table students2(student_id numeric(10),student_name varchar(20),student_address varchar(20));

Table created.

SQL> desc students2;

Name	Null?	Type
STUDENT_ID		NUMBER(10)

STUDENT_NAME	VARCHAR2(20)
STUDENT_ADDRESS	VARCHAR2(20)

2.Local temp -In Oracle, local temporary tables are distinct within modules. These tables are defined and scoped to the session in which you created it.

Syntax- Declare local temporary table table_name(column1 datatype[null/not null],column2 datatype[null/not null],.....columnN datatype[null,/not null]);

Example-

JOIN—— A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

There are 4 Types of Join

- 1.INNER JOIN- Returns records that have matching values in both

Table

Syntax -select table_name1.column_name,table_name2.column_name from table_name1 INNER JOIN table_name2 ON table_name1.column_name=table_name2.column_name;

Example- select * from s1;

ROLL_NO	AGE	NAME	CITY
1	25	mayu	dubai
2	23	komal	mumbai
3	24	rekha	pune
4	25	sneha	baramati
5	26	kamal	gujarat
6	27	surekha	thane
7	25	payal	nagapur

```
         8        26 nivya      nashik

         9        24 rutu       bangalor

        10        25 meena      belgaon

10 rows selected.

SQL> select * from s2;

 STUD_ID NAME         ROLL_NO   M1_MARKS   M2_MARKS   M3_MARKS

---------- ---------- ---------- ---------- ---------- ----------

      110 samu            250        89        63        64

SQL> insert into s2 values(111,'mamata',2,96,58,76);

1 row created

SQL> insert into s2 values(112,'samata',5,69,89,68);

1 row created

SQL> insert into s2 values(113,'mono',7,98,87,76);

1 row created

       SQL> insert into s2 values(114,'hari',10,96,68,87);

1 row created.

SQL> insert into s2 values(115,'nikita',25,63,65,45);

1 row created.

SQL> insert into s2 values(116,'manasi',27,65,96,98);

1 row created.

SQL> select * from s2;

STUD_ID NAME         ROLL_NO   M1_MARKS   M2_MARKS   M3_MARKS

---------- ---------- ---------- ---------- ---------- ----------

      110 samu            250        89        63        64

      111 mamata            2        96        58        76

      112 samata            5        69        89        68

      113 mono              7        98        87        76
```

114 hari 10 96 68 87

115 nikita 25 63 65 45

116 manasi 27 65 96 98

7 rows selected.

SQL> select s1.roll_no,s2.name,s1.age from s1 INNER JOIN s2 ON s1.roll_no=s2.roll_no;

ROLL_NO NAME AGE

---------- ---------- ----------

2 mamata 23

5 samata 2

7 mono 25

10 hari 25

LEFT JOIN- Returns all records from the left table, and the matched records from the right table.

Syntax-select table_name1.colum_name,table_name2.column_name from table_name1 LEFT JOIN table_name2 ON table_name1.column_name=table_name2.column_name order by table_name1.column_name

Example-

select s1.name,s2.stud_id from s1 LEFT JOIN s2 ON s1.roll_no=s2.roll_no order by s1.name;

NAME STUD_ID

---------- ----------

kamal 112

komal 111

mayu

meena 114

nivya

payal 113

rekha

rutu

sneha

surekha

10 rows selected.

RIGHT JOIN- Returns all records from the right table, and the matched records from the left table

Syntax- select table_name1.colum_name,table_name2.column_name from table_name1 RIGHT JOIN table_name2 ON table_name1.column_name=table_name2.column_name order by table_name1.column_name

Example- SQL> select s1.name,s1.age,s2.m1_marks from s1 RIGHT JOIN s2 on s1.roll_no=s2.roll_no order by s1.name;

NAME AGE M1_MARKS

---------- ---------- ----------

kamal 26 69

komal 23 96

meena 25 96

payal 25 98

65

63

89

7 rows selected.

4.FULL JOIN- Returns all records when there is a match in either left or right table

Syntax-

Example-

SQL> select * from s1 FULL outer join s2 on s1.name=s2.name;

ROLL_NO AGE NAME CITY STUD_ID NAME ROLL_NO

---------- ---------- ---------- ---------- ---------- ---------- ----------

M1_MARKS M2_MARKS M3_MARKS

---------- ---------- ----------

1 25 mayu dubai

2 23 komal mumbai

3 24 rekha pune

ROLL_NO AGE NAME CITY STUD_ID NAME ROLL_NO

---------- ---------- ---------- ---------- ---------- ---------- ----------

M1_MARKS M2_MARKS M3_MARK

---------- ---------- ----------

4 25 sneha baramati

5 26 kamal gujarat

6 27 surekha thane

ROLL_NO AGE NAME CITY STUD_ID NAME ROLL_NO

---------- ---------- ---------- ---------- ---------- ---------- ----------

M1_MARKS M2_MARKS M3_MARKS

---------- ---------- ----------

7 25 payal nagapur

8 26 nivya nashik

9 24 rutu bangalor

ROLL_NO AGE NAME CITY STUD_ID NAME ROLL_NO

---------- ---------- ---------- ---------- ---------- ---------- ----------

M1_MARKS M2_MARKS M3_MARKS

---------- ---------- ----------

10 25 meena belgaon

114 hari 10

96 68 87

116 manasi 27

65 96 98

17 rows selected.

5.TCL-Transaction Control Language

A single unit of work in a database is formed after the consecutive execution of commands is known as a transaction.There are certain commands present in SQL known as TCL commands that help the user manage the transactions that take place in a database.**COMMIT. ROLLBACK** and **SAVEPOINT** are the most commonly used TCL commands in SQL.

Commit-COMMIT command in SQL is used to save all the transaction-related changes permanently to the disk. Whenever DDL commands such as INSERT, UPDATE and DELETE are used, the changes made by these commands are permanent only after closing the current session. So before closing the session, one can easily roll back the changes made by the DDL commands.

Syntax-commit;

Savepoint-We can divide the database operations into parts. For example, we can consider all the insert related queries that we will execute consecutively as one part of the transaction and the delete command as the other part of the transaction. Using the SAVEPOINT command in SQL, we can save these different parts of the same transaction using different names.

Syntax-savepoint savepoint_name;

RollBack-While carrying a transaction, we must create savepoints to save different parts of the transaction.according to changing thee requirement to user.

Syntax-rollback to savepoint_name;

Transaction- *START TRANSACTION* command is used to start the transaction.

Syntax-start transaction;

Example ues all TCL commonds.

SQL> create table teacher(ID int,name_teacher varchar(20),Email varchar(20),no_class int);

Table created.

SQL> insert into teacher values(101,'mayu','mayu@gmail123.com',1);

```
1 row created.

SQL> insert into teacher values(102,'geeta','geeta123@gmail.com',2);

1 row created.

SQL> insert into teacher values(103,'komal','komal123@gmail.com',3);

1 row created.

SQL> select * from teacher;

        ID NAME_TEACHER        EMAIL               NO_CLASS
---------- -------------------- -------------------- ----------
       101 mayu                 mayu@gmail123.com           1
       102 geeta                geeta123@gmail.com          2
       103 komal                komal123@gmail.com          3

SQL> start transaction;

SP2-0310: unable to open file "transaction.sql"

SQL> savepoint insertion;

Savepoint created.

SQL> select * from teacher;

        ID NAME_TEACHER        EMAIL               NO_CLASS
---------- -------------------- -------------------- ----------
       101 mayu                 mayu@gmail123.com           1
       102 geeta                geeta123@gmail.com          2
       103 komal                komal123@gmail.com          3

SQL> update teacher set name_teacher='madhu' where id=103;

1 row updated.

SQL> select * from teacher;

        ID NAME_TEACHER        EMAIL               NO_CLASS
---------- -------------------- -------------------- ----------
       101 mayu                 mayu@gmail123.com           1
```

```
102 geeta          geeta123@gmail.com        2

103 madhu          komal123@gmail.com        3

SQL> savepoint updation;

Savepoint created.

SQL> rollback to updation;

Rollback complete.

SQL> select * from teacher;

    ID NAME_TEACHER       EMAIL              NO_CLASS

---------- -------------------- -------------------- ----------

101 mayu           mayu@gmail123.com         1

102 geeta          geeta123@gmail.com        2

103 madhu          komal123@gmail.com        3
```

Like- The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards often used in conjunction with the LIKE operator:

- The percent sign (%) represents zero, one, or multiple characters
- The underscore sign (_) represents one, single character

Syntax-select column1,column2 .....from table_name where columnN like pattern;

Example-

SQL> select *from s1 where name like '%a%';

```
  ROLL_NO        AGE NAME       CITY

---------- ---------- ---------- ----------

         1         25 mayu       dubai

         2         23 komal      mumbai
```

3 24 rekha pune

4 25 sneha baramati

5 26 kamal gujarat

6 27 surekha thane

7 25 payal nagapur

8 26 nivya nashik

10 25 meena belgaon

9 rows selected.

select * from s1 where name like '%a';

ROLL_NO AGE NAME CITY

---------- ---------- ---------- ----------

3 24 rekha pune

4 25 sneha baramati

6 27 surekha thane

8 26 nivya nashik

10 25 meena belgaon

Min and Max –

The min function returns the smallest value of the selected column.

The max function returns the largest value of the selected column.

Min Syntax-select min(column_name) from table_name where condition;

Max syntax- select max(column_name)from table_name where condition;

Example-

select * from s1;

ROLL_NO AGE NAME CITY

---------- ---------- ---------- ----------

```
         1         25 mayu        dubai
         2         23 komal       mumbai
         3         24 rekha       pune
         4         25 sneha       baramati
         5         26 kamal       gujarat
         6         27 surekha     thane
         7         25 payal       nagapur
         8         26 nivya       nashik
         9         24 rutu        bangalor
        10         25 meena       belgaon
10 rows selected.
SQL> select min(age) as smallestage from s1;
SMALLESTAGE
-----------
        23
SQL> select max(age) as largestage from s1;
LARGESTAGE
----------
        27
```

Fetch-The FETCH statement **retrieves rows of data from the result set of a multi-row query**. You can fetch rows one at a time, several at a time, or all at once. The data is stored in variables or fields that correspond to the columns selected by the query.

Syntax-select * from(select column_name(s) from table_name order by column_name(s))where rownum<=number;

Select column_name(s) from table_name where rownum<=number;

Select column_name(s) from table_name order by column_name(s) fetch First number rows only;

Example-

SQL> select roll_no from s1 where rownum<= 3;

ROLL_NO

1

2

3

SQL> select roll_no from s1 order by name fetch first number rows only;

select roll_no from s1 order by name fetch first number rows only

ERROR at line 1:

ORA-00933: SQL command not properly ended

SQL> select * from (select roll_no from s1 order by name)where rownum <= 3;

ROLL_NO

5

2

1

Count-The COUNT function returns the number of rows that matches a specified criterion.

Syntax-select count(column_name) from table_name where condition;

Example-select count(age) from s1;

COUNT(AGE)

10

AVG-The AVG function returns the average value of a numeric column.

Syntax-select AVG(column_name) from table_name where condition;

Example-

select AVG(age) from s1;

AVG(AGE)

25

SUM-The SUM function returns the total sum of a numeric column.

Syntax-select sum(column_name)from table_name where condition;

Example-

select sum(age)from s1;

SUM(AGE)

250

IN-The IN operator allows you to specify multiple values in a WHERE clause.

Syntax-select column_name (s) from table_name where column_name IN (value1,value2,...);

Example-select * from s1 where city in('nagapur','nashik','pune');

ROLL_NO	AGE	NAME	CITY
3	24	rekha	pune
7	25	payal	nagapur
8	26	nivya	nashik

BETWEEN-The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.

Syntax-select column_name(s) from table_name where column_name BETWEEN value1 AND value2;

Example-SQL> select * from s1 where age between 24 and 25;

ROLL_NO	AGE	NAME	CITY
1	25	mayu	dubai
3	24	rekha	pune
4	25	sneha	baramati
7	25	payal	nagapur
9	24	rutu	bangalor
10	25	meena	belgaon

6 rows selected.

Aliases-SQL aliases are used to give a table, or a column in a table, a temporary name.

Aliases are often used to make column names more readable.

An alias only exists for the duration of that query.

An alias is created with the AS keyword.

Syntax-select column_name AS alise_name from table_name;

Example-

SQL> select roll_no AS id,name as S1_name from s1;

ID	S1_NAME
1	mayu
2	komal
3	rekha
4	sneha

5 kamal

6 surekha

7 payal

8 nivya

9 rutu

10 meena

10 rows selected.

UNION Opertor-The UNION operator is used to combine the result-set of two or more SELECT statements.

Syntax-select column_name(s) from table1 UNION select column_name(s) from table_name;

Example-

SQL> select city from s1 union select loaction from per order by city;

CITY

---

bangalor

baramati

belgaon

dubai

goa

gujarat

mumbai

nagapur

nashik

pune

thane

11 rows selected.

GROUP BY-The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".

The GROUP BY statement is often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to group the result-set by one or more columns.

Syntax-select column_name(s) from table_name where condition group by column_name(s) order by column_name(s);

Example-

SQL> select count(roll_no),city from s1 group by city order by count(roll_no)desc;

COUNT(ROLL_NO) CITY

-------------- ----------

1 nashik

1 gujarat

1 mumbai

1 baramati

1 dubai

1 nagapur

1 pune

1 bangalor

1 belgaon

1 thane

10 rows selected.

Having clouse- The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

Syntax-select column_name(s) from table_name where condition group by column_name having condition order by column_name(s);

Example- SQL> select sum(salary),loaction from per GROUP BY location hav;

SUM(SALARY) LOACTION

---------- --------------------

20000 mumbai

35000 goa

40000 pune