SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, Ramapuram

FACULTY OF ENGINEERING AND TECHNOLOGY
SCHOOL OF COMPUTER SCIENCE ENGINEERING

SUBJECT NAME: DATABASE MANAGEMENT SYSTEM LAB MANUAL
SUBJECT CODE: 21CSC205P
YEAR/SEMESTER: II/IV

EX. NO :1 a
DATE:

DATA DEFINITION LANGUAGE COMMANDS

AIM:

To demonstrate the use of SQL Data Definition Language (DDL) queries for creating, modifying, and managing database objects.

PROCEDURE:

Data Definition Commands:

DDL consists of SQL commands used to define and manage the structure of a database. It deals with the descriptions of the database schema and is used to create, modify, and delete database objects such as tables, indexes, and views.

Examples of DDL commands:

1. **CREATE**: Used to create a new database or its objects (such as tables, indexes, functions, views, stored procedures, and triggers).
2. **DROP**: Used to delete database objects (such as tables, indexes, or views) from the database permanently.
3. **ALTER**: Used to modify the structure of an existing database object (such as adding, modifying, or dropping columns in a table).
4. **TRUNCATE**: Used to remove all records from a table, freeing the space allocated for the data. The table structure remains intact.
5. **COMMENT**: Used to add comments or descriptions to the data dictionary or schema objects.
6. **RENAME**: Used to rename an existing object in the database, such as a table or column.

ì CREATE TABLE

The CREATE TABLE command is used to create a new table in the database.

Rules:

1. **Reserved Words**: Oracle reserved keywords cannot be used as table names or column names.
2. **Naming Restrictions**: Table names can contain underscores, numerals, and letters, but cannot contain spaces.
3. **Maximum Length**: The maximum length for a table name is 30 characters.
4. **Unique Table Names**: Each table in a database must have a unique name.
5. **Column Names**: Each column in the table must have a unique name.

6. **Data Types**: Proper data types must be specified for each column, along with the appropriate size or precision (where applicable).

Syntax for CREATE TABLE:
SQL> CREATE TABLE table_name (
column_name1 data_type [constraints],
column_name2 data_type [constraints],
...);

Explanation: The CREATE TABLE command is used to create a new table in the database with specified column names, data types, and optional constraints.

Syntax for DESC:
SQL> DESC table_name;

Explanation: The DESC (short for "DESCRIBE") command displays the structure of a table, including column names, data types, and whether the column can accept NULL values.

Creating a New Table from an Existing Table:

You can create a new table based on the structure and data of an existing table using the following syntax:

SQL> CREATE TABLE new_table_name AS
SELECT column1, column2, ...
FROM existing_table_name WHERE
condition;

Explanation: The CREATE TABLE ... AS SELECT command creates a new table by selecting specific columns from an existing table based on a condition (optional). This can be used to copy data and structure from the existing table.

Syntax:
SQL>Create table tablename (column_name1 data_ type constraints, column_name2 data_ type constraints …);

2. DROP TABLE

The DROP TABLE command is used to delete a table and all of its data from the database permanently. Once a table is dropped, it cannot be recovered unless there is a backup.

Syntax:
SQL> DROP TABLE table_name;

3. ALTER COMMAND

The ALTER TABLE command is used to modify the structure of an existing table. It can perform the following actions:

1. Add a new column.
2. Modify the definition of an existing column.
3. Add or drop integrity constraints (such as PRIMARY KEY, FOREIGN KEY, etc.).

i) ADD COMMAND

To add a new column to an existing table, use the ADD command.

Syntax:
SQL> ALTER TABLE table_name ADD column_name data_type(size);

Explanation: The ADD command adds a new column to the table with the specified data type and size.

ii) MODIFY COMMAND

To modify an existing column's definition (such as its data type or size), use the MODIFY command.

Syntax:
SQL> ALTER TABLE table_name MODIFY column_name data_type(size);

Explanation: The MODIFY command changes the definition of an existing column, allowing you to update its data type, size, or other attributes.

4. TRUNCATE TABLE

The TRUNCATE TABLE command removes all rows from a table but retains the table structure for future use. It is faster than DELETE, and unlike DELETE, it cannot be rolled back.

Syntax:
SQL> TRUNCATE TABLE table_name;

Explanation: The TRUNCATE command deletes all rows from the table while keeping the table structure intact. It frees up the space used by the data.

5. COMMENT

Comments can be written in SQL in three formats:

1. **Single-Line Comments**: Comments that start and end on the same line.
2. **Multi-Line Comments**: Comments that span multiple lines.
3. **Inline Comments**: Comments placed within a SQL statement.

Single-Line Comment

A comment that starts with -- and extends to the end of the line.

Syntax:
-- This is a single-line comment
-- Another comment

Multi-Line Comment

A comment that spans multiple lines, starting with /*and ending with */.

Syntax:
/* This is a multi-line comment
spanning multiple lines */

Inline Comment

An extension of the multi-line comment that can be placed within a SQL statement.

Syntax:
SQL> SELECT * FROM /* comment here */ table_name;

6. RENAME

The RENAME command is used to change the name of an existing database object, such as a table. It allows database users to give more relevant names to tables or other objects.

Syntax:
SQL> RENAME old_table_name TO new_table_name;

Explanation: The RENAME command changes the name of a table or other database object.
Program:

SQL> connect
Enter user-name: system
Enter password: admin
Connected.

```
SQL> CREATE TABLE emp (id NUMBER(10), name VARCHAR(10));
Table created.

SQL> desc emp;
Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)

SQL> alter table emp add(dept varchar(10));
Table altered.

SQL> desc emp;
Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)
DEPT                                               VARCHAR2(10)

SQL> alter table emp modify dept varchar(20);
Table altered.

SQL> desc emp;
Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)
DEPT                                               VARCHAR2(20)

SQL> alter table emp drop column dept;
Table altered.

SQL> desc emp;
Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)

SQL> alter table emp rename to emp1;
Table altered.
```

```
SQL> desc emp1;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)

SQL> desc emp2;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)
DEPT                                               VARCHAR2(10)
SQL> drop table emp2;

Table dropped.

SQL> select * from emp2;
select * from emp2
* ERROR at line
1:
ORA-00942: table or view does not exist
SQL> select * from emp1;

ID NAME       DEPT
---------- ---------- ----------
         1 aaa        cse
         2 aaa        cse
         3 aaa        ece
         4 aaa        cse
         5 aaa        cse

SQL> truncate table emp1;
Table truncated.
SQL> select * from emp1;

no rows selected

SQL> desc emp1;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
ID                                                 NUMBER(10)
NAME                                               VARCHAR2(10)
DEPT                                               VARCHAR2(10)
```

```
SQL> drop table emp1;
Table dropped.

SQL> select * from emp1;
select * from emp1
* ERROR at line
1:
ORA-00942: table or view does not exist


SQL> desc emp1;
ERROR:
ORA-04043: object emp1 does not exist
```

Result:

Thus, the SQL Data Definition Language (DDL) queries were successfully executed to create, modify, and manage database objects.

EX.NO: 1 b
DATE:

DATA MANIPULATION COMMANDS FOR INSERTING, DELETING, UPDATING AND RETRIEVING TABLES

AIM:

To create a database and perform operations using Data Manipulation Commands (DML) for inserting, deleting, updating, and retrieving data, along with using Transaction Control statements.

DESCRIPTION:

DML statements access and manipulate data in existing tables. DML commands are the most frequently used SQL commands and is used to query and manipulate the existing database objects. Some of the commands are Insert, Select, Update, Delete.

Examples of DML:

1) **Insert Command:** This is used to add one or more rows to a table. The values are separated by commas and the data types char and date are enclosed in apostrophes. The values must be entered in the same order as they are defined.

SQL>INSERT INTO table_name (column1, column2, column3) VALUES (value1, value2, value3);

2) **Select Commands:** It is used to retrieve information from the table. It is generally referred to as querying the table. We can either display all columns in a table or only specify column from the table.

SQL>SELECT * FROM table_name; -- to retrieve all columns
SQL>SELECT column1, column2 FROM table_name; -- to retrieve specific columns

3) **Update Command:** It is used to alter the column values in a table. A single column may be updated or more than one column could be updated.

SQL> UPDATE table_name
SET column1 = value1, column2 = value2 WHERE
condition;

4) **Delete command:** After inserting row in a table we can also delete them if required. The delete command consists of a from clause followed by an optional where clause.

SQL>DELETE FROM table_name WHERE condition;

INSERTING VALUES INTO TABLE

Create table:

SQL> CREATE TABLE persons (
pid NUMBER(5),
firstname VARCHAR2(15),
lastname VARCHAR2(15),
address VARCHAR2(25), city
VARCHAR2(10)
);

```
SQL> Create table persons( pid number(5), firstname V
(15), address VarChar(25),city varchar(10));

Table created.

SQL> desc persons;
 Name                                      Null?    T
 ----------------------------------------- -------- --

 PID                                       N
```

INSERT COMMAND

Insert command is used to insert values into table.

Insert a single record into table.

Syntax: SQL> INSERT INTO <table name> VALUES (value list);

Example: SQL> INSERT INTO persons (pid, firstname, lastname, address, city)
VALUES (1, 'Nelson', 'Raj', 'No25, Annai Street', 'Chennai');
1 row created.

```
SQL> insert into persons values (001,'nelson','raj','r
');

1 row created.

SQL> /
```

Insert more than a record into persons table using a single insert command.

Example: SQL> INSERT INTO persons VALUES(&pid, '&firstname', '&lastname', '&address',
'&city');

```
SP2-0042: unknown command "poonamalle" - rest of line
SQL> insert into persons values(&pid,'&firstname','&l
);
Enter value for pid: 002
Enter value for firstname: ram
Enter value for lastname: kumar
Enter value for address: no26,raja nagar
Enter value for city: avadi
old   1: insert into persons values(&pid,'&firstname'
itw')
```

```
SQL> /
Enter value for pid: 003
Enter value for firstname: divya
Enter value for lastname: shivani
Enter value for address: no27,pallavan nagar
Enter value for city: adyar
old   1: insert into persons values(&pid,'&firstname'
ity')
```

Skipping the fields while inserting:

Example: SQL> INSERT INTO persons(pid, firstname) VALUES(500, 'prabhu');

```
SQL> insert into persons(pid,firstname) va

1 row created.
```

SELECT COMMAND

It is used to retrieve information from the table. It is generally referred to as querying the table. We can either display all columns in a table or only specify column from the table.
Syntax: SQL> Select * from tablename; // This query selects all rows from the table.

Example: SQL>Select * from persons;

```
SQL> select *from persons;

       PID FIRSTNAME       LASTNAME        ADDRESS
---------- --------------- --------------- ----------
         1 nelson          raj             no25,annai
         1 nelson          raj             no25,annai
```

THE RETRIEVAL OF SPECIFIC COLUMNS FROM A TABLE:

It retrieves the specified columns from the table

Syntax: SQL> SELECT column_name1, ..., column_nameN FROM table_name;

Example: SQL> SELECT pid, firstname FROM persons;

```
SQL> Select pid, firstname fi

       PID FIRSTNAME
---------- ---------------
         1 nelson
         1 nelson
```

Elimination of duplicates from the select clause:

It prevents retrieving the duplicated values. Distinct keyword is to be used.

Syntax: SQL> SELECT DISTINCT col1, col2 FROM table_name;

Example: SQL> SELECT DISTINCT lastname FROM persons;

```
SQL> Select DISTINCT  lastname f

LASTNAME
---------------

```

SELECT COMMAND WITH WHERE CLAUSE:

To select specific rows from a table we include 'where' clause in the select command. It can appear only after the 'from' clause.

Syntax: SQL> SELECT column_name1, column_name2, ..., column_nameN FROM table_name WHERE condition;

Example: SQL> SELECT firstname, lastname FROM persons WHERE pid > 2;

```
SQL> Select firstname, lastname from per

FIRSTNAME       LASTNAME
--------------- ---------------
```

Select command with order by clause:

Syntax: SQL> SELECT column_name1, column_name2, ..., column_namen FROM table_name WHERE condition ORDER BY column_name;

Example: SQL>Select firstname, lastname from persons order by pid;

```
SQL> Select firstname, lastname from perso

FIRSTNAME       LASTNAME
--------------- ---------------
nelson          raj
nelson          raj
```

Select command to create a table:

Syntax: SQL> CREATE TABLE tablename AS SELECT * FROM existing_tablename;

Example: SQL> CREATE TABLE persons1 AS SELECT * FROM persons;
Table created

SELECT COMMAND TO INSERT RECORDS:

Syntax: SQL> INSERT INTO persons1 (SELECT * FROM persons);

Example: SQL> INSERT INTO persons1 (SELECT pid, firstname, lastname FROM persons WHERE city = 'Chennai');

PID	FIRSTNAME	LASTNAME	ADDRESS	CITY	PHONENO
001	nelson	raj	no25,annai street	Chennai	
100	niranjan	kumar	10/25 krishna street	Mumbai	999999999
102	arjun	kumar	30 sundaram street	coimbatore	
300	gugan	chand	5/10 mettu street	Coimbatore	
500	prabhu				

SELECT COMMAND USING IN KEYWORD:

Syntax: SQL> SELECT column_name1, column_name2, ..., column_n FROM table_name WHERE column_name IN (value1, value2, ...);

Example: SQL> SELECT * FROM persons WHERE pid IN (100, 500);
(OR)
SQL> SELECT * FROM persons WHERE (pid = 100 OR pid = 500);

PID	FIRSTNAME	LASTNAME	ADDRESS	CITY	PHONENO
100	niranjan	kumar	10/25 krishna street	Mumbai	999999999
500	prabhu				

SELECT COMMAND USING BETWEEN KEYWORD:

Syntax: SQL> SELECT column_name1, column_name2, ..., column_n FROM table_name WHERE column_name BETWEEN value1 AND value2;

Example: SQL>Select * from persons where pid between 100 and 500;

PID	FIRSTNAME	LASTNAME	ADDRESS	CITY	PHONENO
100	niranjan	kumar	10/25 krishna street	Mumbai	999999999
500	prabhu				

SELECT COMMAND USING PATTERN:

Syntax: SQL> SELECT column_name1, column_name2, ..., column_n FROM table_name WHERE column_name LIKE 'pattern';

Example: SQL>Select * from persons where firstname like 'nir_n%';

PID	FIRSTNAME	LASTNAME	ADDRESS	CITY	PHONENO
100	niranjan	kumar	10/25 krishna street	Mumbai	999999999

RENAMING THE FIELDNAME AT THE TIME OF DISPLAY USING SELECT STATEMENT:

Syntax: SQL> SELECT old_column_name AS new_column_name FROM table_name WHERE condition;

Example: SQL> SELECT pid AS personid FROM persons;

SELECT COMMAND TO RETRIEVE NULL VALUES:

Syntax: SQL> SELECT column_name FROM table_name WHERE column_name IS NULL;

Example: SQL> SELECT * FROM persons WHERE lastname IS NULL;

PID	FIRSTNAME	LASTNAME	ADDRESS	CITY	PHONENO
500	prabhu				

UPDATE COMMAND:

Syntax: SQL> UPDATE table_name SET column_name = value [, column_name = value]... [WHERE condition];

Example: SQL> UPDATE persons SET pid = 5 WHERE firstname = 'prabhu';

Table updated.

DELETE COMMAND

Syntax: SQL> DELETE FROM table_name WHERE condition;

Example: SQL> DELETE FROM persons WHERE pid = 500;

1 row deleted.

RESULT:

Thus, the database was successfully created, and data was inserted, deleted, modified, updated, and altered. Records were also retrieved based on specific conditions.