Deccan Education Society's (DES)
Pune University, Pune
School of Engineering and Technology
Department of Computer Engineering and Technology
Program: B. Tech in Computer Science and Engineering

Academic Year: 2024-25	**Year:** Second Year	**Semester:** II
PRN No.:	**Name:**	
Subject: Database Management System		
Assignment No.: 2		
Date:		

Lab Assignment: 02

Title: DDL Queries: Design and develop SQL DDL (Data Definition Language) statements to demonstrate the creation and management of various SQL objects like Creating tables with appropriate data types and constraints such as PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL, DEFAULT and CHECK.
Defining indexes to optimize query performance on large datasets.
Use auto_increment to generate unique numeric values for a primary key column.

Theory:

What is SQL?

SQL (Structured Query Language) is a language used to manage and manipulate relational databases. It allows users to create, retrieve, update, and delete data efficiently. SQL is used in databases like MySQL, PostgreSQL, SQL Server, and Oracle.

Types of SQL Commands:

- DDL (Data Definition Language) – Defines database structures (e.g., CREATE, ALTER, DROP).
- DML (Data Manipulation Language) – Modifies data (e.g., INSERT, UPDATE, DELETE).
- DCL (Data Control Language) – Manages permissions (e.g., GRANT, REVOKE).
- TCL (Transaction Control Language) – Controls transactions (e.g., COMMIT, ROLLBACK).

What is DDL?

DDL (Data Definition Language) is a category of SQL commands used to define, modify, and manage the structure of database objects such as tables, indexes, and schemas.

DDL Commands:

1. CREATE – Creates a new database, table, or index.

```
CREATE TABLE Students (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100),
    age INT
);
```

2. ALTER – Modifies an existing table (e.g., add/drop a column).

```
ALTER TABLE Students ADD COLUMN email VARCHAR(100);
```

3. DROP – Deletes an entire database or table.

```
DROP TABLE Students;
```

4. TRUNCATE – Removes all records from a table but keeps its structure.

```
TRUNCATE TABLE Students;
```

Database Keys and their explanation

Keys in a database help in uniquely identifying records in a table.

Types of Database Keys:

1. Primary Key – Uniquely identifies each record in a table.

```
CREATE TABLE Employees (
    emp_id SERIAL PRIMARY KEY,
    emp_name VARCHAR(100)
);
```

2. Foreign Key – Establishes a relationship between two tables.

```
CREATE TABLE Orders (
    order_id SERIAL PRIMARY KEY,
    emp_id INT,
    FOREIGN KEY (emp_id) REFERENCES Employees(emp_id)
);
```

3. Unique Key – Ensures unique values in a column but allows NULL values.

```
CREATE TABLE Users (
    user_id SERIAL PRIMARY KEY,
    email VARCHAR(100) UNIQUE
);
```

4. Candidate Key – A set of attributes that could be a primary key.
5. Super Key – A combination of attributes that uniquely identifies a record.
6. Composite Key – A primary key consisting of multiple columns.

```
CREATE TABLE Enrollments (
    student_id INT,
    course_id INT,
    PRIMARY KEY (student_id, course_id)
);
```

Constraints and Types

Constraints are rules applied to database columns to maintain data integrity.

Types of Constraints:

1. NOT NULL – Prevents a column from storing NULL values.

```
CREATE TABLE Employees (
   emp_id SERIAL PRIMARY KEY,
   emp_name VARCHAR(100) NOT NULL
);
```

2. UNIQUE – Ensures all values in a column are unique.

```
CREATE TABLE Users (
   user_id SERIAL PRIMARY KEY,
   email VARCHAR(100) UNIQUE
);
```

3. PRIMARY KEY – Ensures uniqueness and NOT NULL constraint.
4. FOREIGN KEY – Establishes a relationship between tables.
5. CHECK – Ensures a column meets specific conditions.

```
CREATE TABLE Products (
   product_id SERIAL PRIMARY KEY,
   price DECIMAL(10,2) CHECK (price > 0)
);
```

6. DEFAULT – Assigns a default value if no value is provided.

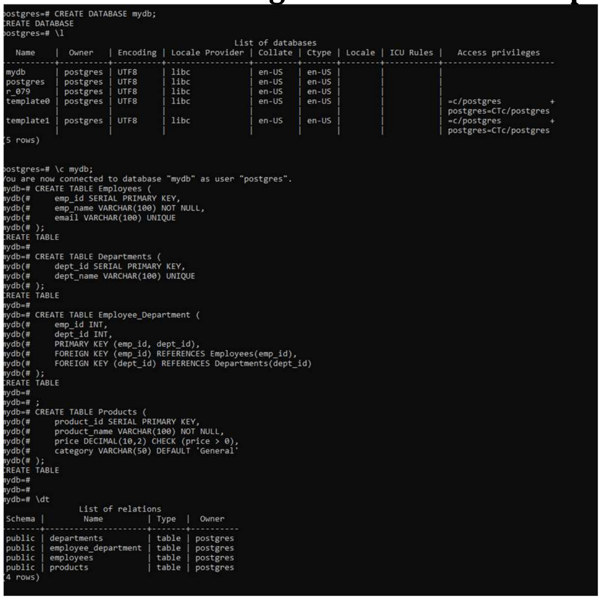
```
CREATE TABLE Employees (

    emp_id SERIAL PRIMARY KEY,

    department VARCHAR(50) DEFAULT 'HR'

);
```

Auto Increment

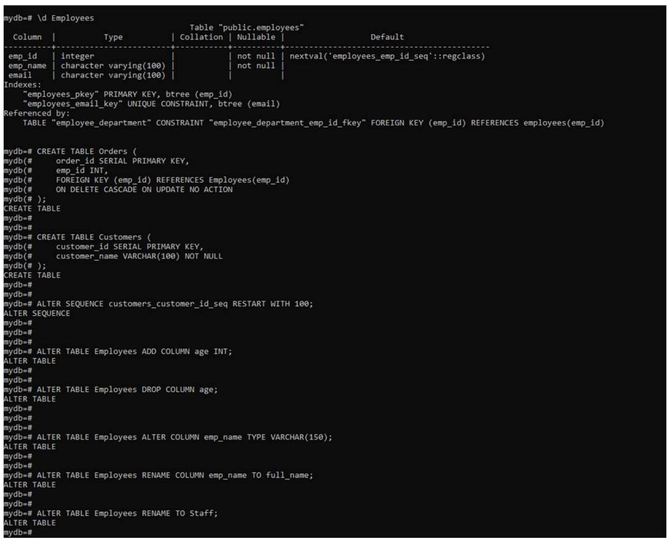
Auto-increment is used to automatically generate unique values for a column.

Show Query Execution Screenshots for:

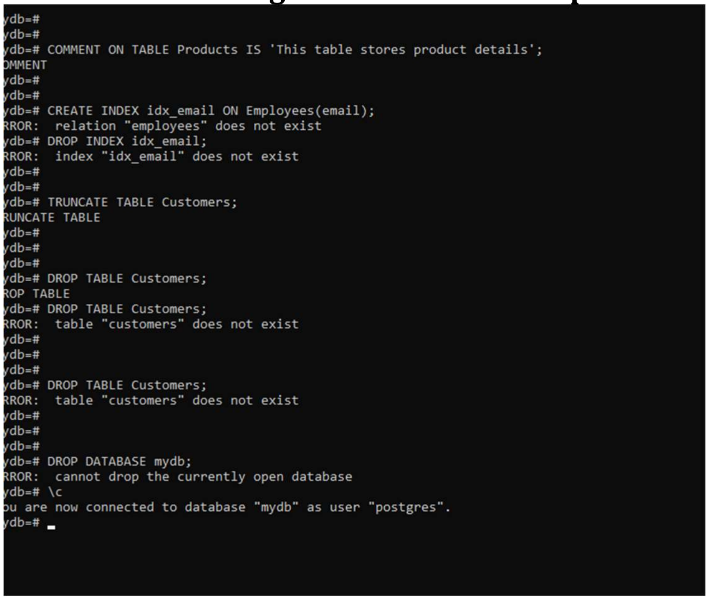
- Database Creation
- Show database list
- Table Creation with Primary, Foreign and Unique Key
- Table Creation with use of Constraints – Not Null, Check, Default
- Show Table List
- Display Table Structure
- Foreign Key with On Delete cascade and On update no action
- Auto Increment
- Auto Increment with Alter to set new Initial Value
- **Alter Table:**
  1. Add New Attribute at second position
  2. Drop existing attribute
  3. Change data type of any attribute
  4. Change attribute name
  5. Change Table Name
- Show Comments
- Create and drop Index
- Truncate Table
- Drop table
- Drop Database



```
ostgres=# CREATE DATABASE mydb;
REATE DATABASE
ostgres=# \l
                                                  List of databases
   Name    |  Owner   | Encoding | Locale Provider | Collate | Ctype | Locale | ICU Rules |   Access privileges
-----------+----------+----------+-----------------+---------+-------+--------+-----------+-----------------------
 mydb      | postgres | UTF8     | libc            | en-US   | en-US |        |           |
 postgres  | postgres | UTF8     | libc            | en-US   | en-US |        |           |
 r_079     | postgres | UTF8     | libc            | en-US   | en-US |        |           |
 template0 | postgres | UTF8     | libc            | en-US   | en-US |        |           | =c/postgres          +
           |          |          |                 |         |       |        |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | en-US   | en-US |        |           | =c/postgres          +
           |          |          |                 |         |       |        |           | postgres=CTc/postgres
5 rows)


ostgres=# \c mydb;
ou are now connected to database "mydb" as user "postgres".
ydb=# CREATE TABLE Employees (
ydb(#     emp_id SERIAL PRIMARY KEY,
ydb(#     emp_name VARCHAR(100) NOT NULL,
ydb(#     email VARCHAR(100) UNIQUE
ydb(# );
REATE TABLE
ydb=#
ydb=# CREATE TABLE Departments (
ydb(#     dept_id SERIAL PRIMARY KEY,
ydb(#     dept_name VARCHAR(100) UNIQUE
ydb(# );
REATE TABLE
ydb=#
ydb=# CREATE TABLE Employee_Department (
ydb(#     emp_id INT,
ydb(#     dept_id INT,
ydb(#     PRIMARY KEY (emp_id, dept_id),
ydb(#     FOREIGN KEY (emp_id) REFERENCES Employees(emp_id),
ydb(#     FOREIGN KEY (dept_id) REFERENCES Departments(dept_id)
ydb(# );
REATE TABLE
ydb=#
ydb=# ;
ydb=# CREATE TABLE Products (
ydb(#     product_id SERIAL PRIMARY KEY,
ydb(#     product_name VARCHAR(100) NOT NULL,
ydb(#     price DECIMAL(10,2) CHECK (price > 0),
ydb(#     category VARCHAR(50) DEFAULT 'General'
ydb(# );
REATE TABLE
ydb=#
ydb=#
ydb=# \dt
              List of relations
 Schema |        Name         | Type  |  Owner
--------+---------------------+-------+----------
 public | departments         | table | postgres
 public | employee_department | table | postgres
 public | employees           | table | postgres
 public | products            | table | postgres
4 rows)
```



```
mydb=# \d Employees
                                        Table "public.employees"
  Column  |          Type          | Collation | Nullable |                  Default
----------+------------------------+-----------+----------+-------------------------------------------
 emp_id   | integer                |           | not null | nextval('employees_emp_id_seq'::regclass)
 emp_name | character varying(100) |           | not null |
 email    | character varying(100) |           |          |
Indexes:
    "employees_pkey" PRIMARY KEY, btree (emp_id)
    "employees_email_key" UNIQUE CONSTRAINT, btree (email)
Referenced by:
    TABLE "employee_department" CONSTRAINT "employee_department_emp_id_fkey" FOREIGN KEY (emp_id) REFERENCES employees(emp_id)


mydb=# CREATE TABLE Orders (
mydb(#     order_id SERIAL PRIMARY KEY,
mydb(#     emp_id INT,
mydb(#     FOREIGN KEY (emp_id) REFERENCES Employees(emp_id)
mydb(#     ON DELETE CASCADE ON UPDATE NO ACTION
mydb(# );
CREATE TABLE
mydb=#
mydb=#
mydb=# CREATE TABLE Customers (
mydb(#     customer_id SERIAL PRIMARY KEY,
mydb(#     customer_name VARCHAR(100) NOT NULL
mydb(# );
CREATE TABLE
mydb=#
mydb=#
mydb=# ALTER SEQUENCE customers_customer_id_seq RESTART WITH 100;
ALTER SEQUENCE
mydb=#
mydb=#
mydb=#
mydb=# ALTER TABLE Employees ADD COLUMN age INT;
ALTER TABLE
mydb=#
mydb=#
mydb=# ALTER TABLE Employees DROP COLUMN age;
ALTER TABLE
mydb=#
mydb=#
mydb=#
mydb=# ALTER TABLE Employees ALTER COLUMN emp_name TYPE VARCHAR(150);
ALTER TABLE
mydb=#
mydb=#
mydb=# ALTER TABLE Employees RENAME COLUMN emp_name TO full_name;
ALTER TABLE
mydb=#
mydb=#
mydb=# ALTER TABLE Employees RENAME TO Staff;
ALTER TABLE
mydb=#
```



```
ydb=#
ydb=#
ydb=# COMMENT ON TABLE Products IS 'This table stores product details';
OMMENT
ydb=#
ydb=#
ydb=# CREATE INDEX idx_email ON Employees(email);
RROR:  relation "employees" does not exist
ydb=# DROP INDEX idx_email;
RROR:  index "idx_email" does not exist
ydb=#
ydb=#
ydb=# TRUNCATE TABLE Customers;
RUNCATE TABLE
ydb=#
ydb=#
ydb=#
ydb=# DROP TABLE Customers;
ROP TABLE
ydb=# DROP TABLE Customers;
RROR:  table "customers" does not exist
ydb=#
ydb=#
ydb=#
ydb=# DROP TABLE Customers;
RROR:  table "customers" does not exist
ydb=#
ydb=#
ydb=#
ydb=# DROP DATABASE mydb;
RROR:  cannot drop the currently open database
ydb=# \c
ou are now connected to database "mydb" as user "postgres".
ydb=# _
```

Conclusion:

FAQs:

1. Which version of MYSQL have you installed?

   MySQL 8.0

2. Explain MySQL Client - server architecture.
   MySQL follows a **Client-Server Architecture**, where:
   - The **MySQL Server** manages the database, processes queries, and handles transactions.
   - **Clients** (like MySQL Workbench, Command Line, or Applications) send SQL queries to the server.
   - The **server processes** the queries and returns results.
   - Communication happens using **TCP/IP** or **Unix sockets**.

3. Find out databases used for following applications.

A. Twitter:	MySQL, Manhattan	B. Facebook:	MySQL, TAO
C. Amazon / Flipkart:	DynamoDB, MySQL	D. Finacle:	Oracle Database
E. LinkedIn:	Espresso, MySQL	F. PokemonGo:	PostgreSQL

G. Panama Papers: Neo4j H. AADHAR Card: MongoDB

4. Categorize following commands under DDL, DML and DCL. Create, Update, Commit, Delete, Drop, Truncate, Rollback, Alter

   DDL (Data Definition Language) CREATE, DROP, ALTER, TRUNCATE

   DML (Data Manipulation Language) UPDATE, DELETE

   DCL (Data Control Language) COMMIT, ROLLBACK

5. Write a DDL statement to remove sales and suppliers database?

   DROP DATABASE sales;

   DROP DATABASE suppliers;

6. Write the statement that will add a column CGPA to a table Student which is already created

   ALTER TABLE Student ADD COLUMN CGPA DECIMAL(3,2);