**DBMS**

**Lab Assignment number 3**

**Name:** Aamir Ansari **Batch** A **Roll no.** 01

**Aim: Experiment to study Data Definition Language Commands and Integrity constraints.**

**Theory:**

The Data Definition Language (DDL) is used to create and destroy databases and

database objects. These commands will primarily be used by database administrators during the

setup and removal phases of a database project.

Let’s take a look at the structure and usage of four basic DDL commands:

**CREATE:**

Create command is used to create different database objects like table,view etc.

Syntax: create table <table name>(<column name> <datatype>[size],….);

**ALTER:**

Once you’ve created a table within a database, you may wish to modify the definition of it.

The ALTER command allows you to make changes to the structure of a table without deleting and

recreating it. Take a look at the following command:

**DROP**

The final command of the Data Definition Language, DROP, allows us to remove entire database

objects from our DBMS. For example, if we want to permanently remove the personal\_info table

that we created, we’d use the following command:

Syntax: Drop table tablename;

Integrity constraints.

**Integrity constraints:**

**1. NOT NULL Constraint:** When column is defined as a NOT NULL

then that column becomes mendatory column i.e value must be

entered in that column for each row.

Syntax: columnname datatype(size) NOT NULL

**2. Primary key constraint:** This constraint ensures that

a)Data entered in the table column is unique across the entire table.

b)None of the cells belonging to the column are left emty.

Syntax: columnname datatype(size) Primary key

**3. Unique key constraint:** This constraint ensures that

a) Data entered in the table column is unique across the entire table.

b) Column can have NULL value.

Syntax: columnname datatype(size) Unique

**4. Foreign key constraint:** Foreign key constraint creates relationship between

Records. Often we wish to ensure that a value appearing in a relation for a

given set of attributes also appears for another set of attributes in another

relation. This is called referential integrity This constraint ensures that

a) Record cannot be inserted into a detail if corresponding records in

the master table do not exist.

b) Records of the master table cannot be deleted if corresponding

records in detail table exist.

Syntax: columnname datatype(size) references tablename(columnname)

On delete cascade

**5. Check Constraint:** Business rule validations can be applied column by using

check constraint. Check constraint must be specified as a logical

expression that evaluates either true or false.

Syntax: columnname datatype(size) CHECK(logical expression)

**Defining Integrity Constraints with the CREATE TABLE Command**

Example:

--Execution of different types of DDL Commands(create, Alter, Delete)

--create command

create table customer(cid int, cname varchar(20), address char(10));

--Alter command(add | alter column | drop column)

--adding new column in existing table

alter table customer add phno numeric(10)

--Modifying existing column

alter table customer alter column address varchar(20)

--dropping column from existing table

alter table customer drop column address

--Execution of different types of Integrity constraints

--Table Employee

create table employee(ssn int primary key check(ssn like &#39;\_\_&#39;),ename varchar(20)

not null, salary money, superssn int foreign key references employee(ssn),

dno int default 5)

--Table Department

create table dept(dno int primary key, dname varchar(20), mgrssn int references

employee, startdate datetime)

-- applying foreign key constarint on existing table

alter table employee add constraint fk\_dno foreign key(dno) references dept(dno)

--Table deptlocation

create table deptloc(dno int,dloc varchar(20),primary key(dno,dloc),foreign key(dno)

references dept(dno))

--Table Project

create table project(pno int primary key,pname varchar(20),dno int references, dept(dno))

--Table workson

create table workson(ssn int, pno int, noofhrs int, primary key(ssn,pno),

foreign key(ssn) references employee(ssn) on delete cascade on update cascade,

foreign key(pno) references project(pno) on delete cascade on update cascade)

--Table Dependent

create table dependent(ssn int,depname varchar(20),relation varchar(20),

primary key(ssn,depname),foreign key(ssn) references employee(ssn))

--droping primary key constraint from existing table

alter table dependent drop constraint PK\_\_dependent\_\_3B0BC30C

--Applying constraint on existing table

alter table dependent add constraint pk\_ssn primary key(ssn,depname)

**Code:**

CREATE TABLE Employee (

f\_name VARCHAR(30) NOT NULL,

m\_name VARCHAR(30),

l\_name VARCHAR(30),

ssn BIGINT NOT NULL,

dob DATE,

addr VARCHAR(50),

sex CHAR(1),

salary MONEY,

super\_ssn BIGINT NOT NULL,

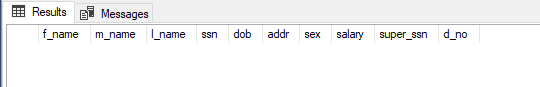
d\_no INT,

PRIMARY KEY (ssn),

FOREIGN KEY (super\_ssn) REFERENCES Employee(ssn)

);

SELECT \* FROM Employee;

.

\*\*\*

CREATE TABLE Department (

d\_no INT NOT NULL,

d\_name VARCHAR(20) NOT NULL,

mgr\_ssn BIGINT,

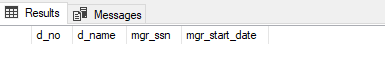
mgr\_start\_date DATE,

PRIMARY KEY (d\_no),

FOREIGN KEY (mgr\_ssn) REFERENCES Employee(ssn)

);

SELECT \* FROM Department;

.

\*\*\*

ALTER TABLE Employee

ADD FOREIGN KEY (d\_no) REFERENCES Department(d\_no);

CREATE TABLE Department\_location (

d\_no INT NOT NULL,

d\_location VARCHAR(20) NOT NULL,

PRIMARY KEY (d\_no, d\_location),

FOREIGN KEY (d\_no) REFERENCES Department (d\_no)

);

SELECT \* FROM Department\_location;



\*\*\*

CREATE TABLE Project (

p\_no INT NOT NULL,

p\_name VARCHAR(20) NOT NULL,

p\_location VARCHAR(20) NOT NULL,

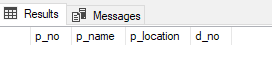
d\_no INT NOT NULL,

PRIMARY KEY (p\_no),

FOREIGN KEY (d\_no) REFERENCES Department (d\_no)

);

SELECT \* FROM Project;

.

\*\*\*

CREATE TABLE Works\_on (

e\_ssn BIGINT NOT NULL,

p\_no INT NOT NULL,

hours\_worked FLOAT,

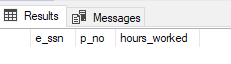
PRIMARY KEY (e\_ssn, p\_no),

FOREIGN KEY (e\_ssn) REFERENCES Employee (ssn),

FOREIGN KEY (p\_no) REFERENCES Project (p\_no)

);

SELECT \* FROM Works\_on;

.

\*\*\*

CREATE TABLE Dependant (

e\_ssn BIGINT NOT NULL,

dependent\_name VARCHAR(30) NOT NULL,

dependent\_sex CHAR(1),

dependent\_dob DATE,

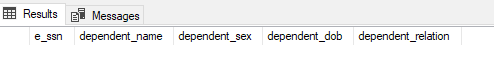
dependent\_relation VARCHAR(20),

PRIMARY KEY (e\_ssn, dependent\_name),

FOREIGN KEY (e\_ssn) REFERENCES Employee (ssn)

);

SELECT \* FROM Dependant;



\*\*\*

**Conclusion:** Thus we have implemented different DDL Commands and Integrity constraints successfully.