**Batch: A1 Roll No.: 1911004**

**Experiment / assignment / tutorial No. 5**

**Grade: AA / AB / BB / BC / CC / CD /DD**

|  |
| --- |
| **Title:** Implementation of Database in SQL -DDL |

**Objective:** Define/modify database definitions with proper constraints

**Expected Outcome of Experiment:**

CO 2: Convert entity-relationship diagrams into relational tables, populate a relational database and formulate SQL queries on the data Use SQL for creation and query the database.

CO 3: Define and apply integrity constraints and improve database design using normalization techniques.

**Books/ Journals/ Websites referred:**

1. Sharaman Shah,”*Oracle for Professional*”, SPD.
2. Dr. P.S. Deshpande, SQL and PL/SQL for Oracle 10g.Black book, Dreamtech Press
3. Korth, Slberchatz, Sudarshan: “Database Systems Concept”, 5th Edition , McGraw Hill
4. Peter Rob and Carlos Coronel,”Database Systems Design, Implementation and Management”, Thompson Learning, 5th Edition

**Pre Lab/ Prior Concepts**:

Resources used: Postgresql

**Theory:** The set of relations in a database must be specifies to the system by means of a data definition language (DDL). The SQL DDL allows specification of not only a set of relations but also specific information about the relation including,

1. The schema for each relation
2. The domain of values associated with each attribute
3. The integrity constraints
4. The set of indices to be maintained for each relation
5. The security and authorization information for each relation
6. The physical storage structure of each relation on disk

# Syntax Create Table:

# create table employee(ssn,fname varchar(10), mname varchar(10), lname varchar(10), desg varchar(20), gender varchar(5), addr varchar(20), bdate datetime, sal float,primary key(ssn));

create table manages(ssn int, dept\_code int, start\_dt datetime, foreign key(ssn)

create table manages(ssn int, dept\_code int, start\_dt datetime, foreign key(ssn)

references employee, foreign key(dept\_code) refrences department, key(ssn,dept\_code) ) on delete set null;primary

# Data Constraints

Busines managers of the organization determine the a set of rules that must be applied before the data is stored in the database. The application of such rules on raw data ensures **data integrity**.

**Eg:-** An employee belonging to Sales department cannot have salary higher than Rs. 1000.

An employee has an unique identification number.

# Applying Data Constraints

Oracle permits data constraints to be attached to table columns using SQL syntax. Constraints can be attached to table columns using

Alter table

# Unique Constraint

**Unique Constraint- At column level Syntax**

**<ColumnName><Datatype>(<size>)**

**UNIQUE Unique Constraint- At table level**

**CREATE TABLE<TableName>(**

**<ColumnName><Datatype>(<size>)**

**<ColumnName><Datatype>(<size>)**

**<Columnname><Datatype>(<size>) UNIQUE(<ColumnName1>,<ColumnName2>);**

**Implementation Details (Problem Statement, Query and Screenshots of Results):**

**Blogging Management System**

A usercan post his blogs. User can see the blogs of other and like/share them if he likes their content. He can follow them if he wants to prioritize their content. Also the user will be able to see who has liked or shared his/her blog. The database/system will have following data

- User's personal data

- User's All Posts like, Comments and who is sharing post

- Also who has Liked Comments of any posts

- User's Followers and Following

User can see who follows him. He can view his following too. He can also see following followers of his friends. User can also comment on any post. User can see who follows him. Also the user will have access to information of who likes his blog and comments. The system will automatically generate the blog id for a blog. User first has to provide his/her details and a password and username while logging in for the first time. After that the login can be done using the username and password. User can enter either a phone number or email id or both during login but one of them is necessary. A blog will contain various features like title, description, type. Content, etc that can be viewed by other users. It will be saved in the database with information such as blog id, owner’s id, date on which it has been posted. The likes and comments on a blog will keep on changing in the meantime. User’s can also share the link of a particular blog among his/her followers. The owner of the blog will come to know about who has shared his blog.

create table if not exists User(

user\_name varchar(20) primary key ,

name varchar(20) not null,

phone\_number int not null,

Email\_id varchar(30) not null,

Dob date not null,

password varchar(20) not null,

age int not null,

gender char

);

create table if not exists Blog(

blog\_id int primary key,

blog\_title varchar(100) not null,

blog\_description varchar(100) not null,

blog\_content varchar(1000) not null,

blog\_type varchar(20) not null,

blog\_date date not null,

user\_name varchar(20),

FOREIGN KEY(user\_name) references User(user\_name)

);

create table if not exists Likes (

user\_id int not null,

blog\_id int ,

FOREIGN KEY (blog\_id) references Blog (blog\_id)

);

create table if not exists shares(

user\_id int not null,

blog\_id int ,

FOREIGN KEY (blog\_id) references Blog (blog\_id)

);

create table if not exists comments(

user\_id int not null,

blog\_id int ,

FOREIGN KEY (blog\_id) references Blog (blog\_id)

);

insert into User( user\_name,name,phone\_number,Email\_id,Dob,password,age,gender)

values(" abc123 "," ABC ","5241452587"," abc@g.com ","2002-20-05"," abc@123 ","18","M");

insert into User( user\_name,name,phone\_number,Email\_id,Dob,password,age,gender )

values(" XYZ "," ABC ","5241452578"," xyz@g.com ","2002-30-03"," xyz@123 ","18","F");

insert into User( user\_name,name,phone\_number,Email\_id,Dob,password,age,gender )

values( " PQR "," PQR ","5241452586"," pqr@g.com ","1970-10-11"," pqr@123 ","50","M");

select \* from User;



select \* from User where gender="M";



select \* from User where age="18";



insert into Blog( blog\_id,blog\_title,blog\_description,blog\_content,blog\_type,blog\_date,user\_name)

values( "0001"," title"," description"," content"," Test blog","2021-02-13"," abc123" );

insert into Likes(user\_id,blog\_id) values("abc123","0001");



# Conclusion: We successfully learned how to implement & create a table in MySQL also insert values & select the values from table thus display it.

# Post Lab Questions:

# Which command is used for removing a table and all its data from the database:

# DROP Command

# TRUNCATE Command

# Both Commands

# ANS . A DROP Command

# For the given ER model, using DDL command: Write syntax to create CREATE Tables with all possible integrity constraints

# Problem Statement: A small accounting firm wants a simple HR application that will help it to keep track of its employees, their positions, allowances, salary scales, and which company vehicles their employees drive. The application must keep track of all the positions at the firm, the employees filling these positions, the allowances for these positions, the salary scales for these positions, and the company vehicles assigned to these positions.

# Case Study 1 detailed E-R diagram

create database HR\_firm;

use HR\_firm;

create table positions (

position\_id int PRIMARY KEY,

position\_name char (50) NOT NULL,

positions\_description varchar (200),

details char (100)

);

create table employees (

employee\_id int PRIMARY KEY,

ss\_number int NOT NULL,

last\_name varchar (100),

first\_name varchar (100),

gender char (20),

DOB DATE

);

create table allowances (

allowance\_id int PRIMARY KEY,

allowance\_name varchar (100) NOT NULL,

allowance\_description varchar (200),

amount float NOT NULL

);

create table salary\_scales (

salary\_scale\_code int PRIMARY KEY,

salary\_scale\_name varchar (50) NOT NULL,

salary\_scale\_description char (100),

min\_salary float,

max\_salary float

);

create table vehicles (

vehicle\_id int PRIMARY KEY,

VIN int NOT NULL UNIQUE,

reg\_no int NOT NULL UNIQUE,

make\_year year,

make char (100),

model char (100),

color varchar (100)

);