**Experiment No: 02**

**create, alter, drop, rename, truncate, description(All Basic commands: show: databases, tables, use database, create tables, drop: database, tables, description of table)**

SHOW DATABASES;

CREATE DATABASE University;

SHOW DATABASES;

USE University;

CREATE TABLE Student(student\_id int(20), student\_name varchar(20), semester int(2), contact int(10), address varchar(40));

SHOW TABLES;

DESC Student;

CREATE TABLE Course(credit int(3), title varchar(10), teacher varchar(20));

SHOW TABLES;

DESC Course;

CREATE TABLE Stay(accommodation varchar(20), services(20), timings int(5));

SHOW TABLES;

DESC Stay;

DROP TABLE Stay;

SHOW TABLES;

DROP DATABASE University;

SHOW DATABASES;

**Experiment No: 03**

**Company Database and SQL- DDL Commands continuation (rename, alter, truncate)**

SHOW DATABASES;

CREATE DATABASE Company;

USE Company;

CREATE TABLE Employ(employ\_id int(20), name varchar(40), salary int(10), address varchar(40), contact\_number int(10), primary key(employ\_id));

CREATE TABLE Project(p\_id int(10), project\_name varchar(10), dept char(10), primary key(p\_id));

SHOW TABLES;

DESC Employ;

RENAME TABLE Project TO on\_going\_projects;

SHOW TABLES;

ALTER TABLE Employ add gender varchar(20);

DESC Employ;

TUNCATE TABLE Employ;

DESC Employ;

SHOW TABLES;

DESC on\_going\_projects;

DROP TABLE on\_going\_projects;

SHOW TABLES;

DROP DATABASE Company;

SHOW DATABASES;

**Experiment No: 04**

**(INSERT,UPDATE,DELETE and SELECT QUERY) on Company Database)**

SHOW DATABASES;

CREATE DATABAE Company;

USE Company;

CREATE TABLE Employ(employ\_id int(20), name varchar(40), salary int(10), address varchar(70), contact\_number int(10), primary key(employ\_id));

CREATE TABLE Projects(Project\_id int(10), Project\_name varchar(10), dept char(10), primary key(Project\_id));

SHOW TABLES;

DESC Employ;

ALTER TABLE Employ DROP contact\_number;

DESC Employ;

ALTER TABLE Employ ADD contact\_number double;

DESC Employ;

INSERT INTO Employ VALUES(52, “SHREERANG MHATRE”, 3000000, “302, Shilpa society , Kothrud”, 9547863218);

INSERT INTO Employ VALUES(56, “Vijay SHnakar”, 2000000, “103, Rambaug colony , Kothrud”, 4368975126);

…

..

..

SELECT \* FROM Employ;

DELETE FROM Employ WHERE name = “SHREERANG MHATRE”;

SELECT \* FROM Employ;

UPDATE Employ SET salary\*0.03 WHERE Salary > 3000000;

SELECT \* FROM Employ;

SELECT employ\_id, name, salary/10 FROM employ;

**Experiment No: 05**

**Select query: with all clauses (from, where order by, group by, logical operators, between and null values)**

Same code from exp 4 and the new code below:

SELECT \* FROM Employ;

SELECT \* FROM Employ WHERE name = “Shreeranh Mhatre”;

SELECT \* FROM Employ ORDER BY Employ\_id asc; ///////// Ascending order

SELECT \* FROM Employ ORDER BY Employ\_id desc; ///////// Desending order

**Experiment No: 06**

**SQL Queries on Functions: Single row, Aggregate functions, Data sorting, Subquery.**

SHOW DATABASES;

CREATE DATABASE Company;

SHOW DATABASES;

USE Company;

CREATE TABLE Employees(

-> emp\_id INT PRIMARY KEY,

-> first\_name VARCHAR(50),

-> last\_name VARCHAR(50),

-> Salary DECIMAL(10, 2),

-> department\_id INT

-> );

SHOW TABLES;

INSERT INTO Employees (emp\_id, first\_name, last\_name, salary, department\_id)

VALUES

(1, ‘John’, ‘DOE’, 500000, 1),

(2, ‘SAM’, ‘Andrew’, 500000, 2);

DESC Employees;

SELECT \* FROM Employees;

SELECT UPPER(first\_name || ‘ ‘ || last\_name) AS full\_name, Salary FROM Employees; /// Single ROW functions

SELECT AVG(salary) AS average\_salary FROM Employees; /// Aggregate Functions

SELECT SUM(salary) AS total\_salary\_expenditure FROM Employees WHERE department\_id = 1;

SELECT emp\_id, first\_name, last\_name, salary //// Data Sorting

FROM Employees

WHERE department\_id = 2

ORDER BY salary DESC;

SELECT emp\_id, first\_name, last\_name, salary ///Subquery

FROM Employees

WHERE department\_id = 1

AND salary > (SELECT AVG(salary) FROM employees);

**Experiment No: 07**

**SQL Join operations, set operations, View, TCL commands**

SHOW DATABASES;

CREATE DATABASE Company;

USE Company;

CREATE TABLE Employees(

-> emp\_id INT PRIMARY KEY,

-> first\_name VARCHAR(50),

-> last\_name VARCHAR(50),

-> Salary DECIMAL(10, 2),

-> department\_id INT

-> );

SHOW TABLES;

CREATE TABLE Departments(

department\_id INT PRIMARY KEY,

department\_name VARCHAR(50)

);

SHOW TABLES;

--sample data for employees table (comments start with --)

INSERT INTO Employees (emp\_id, first\_name, last\_name, salary, department\_id)

VALUES

(1, ‘John’, ‘DOE’, 500000, 1),

(2, ‘SAM’, ‘Andrew’, 500000, 2);

DESC Employees;

--sample data for departments table (comments start with --)

INSERT INTO Department (department\_id, department\_name)

VALUES

(1, ‘HR’),

(2, ‘IT’);

DESC Department;

--- JOIN OPERATION

SELECT e.first\_name, e.last\_name, d.department\_name

FROM employees e

INNER JOIN department d ON e.department\_id = d.department\_id;

SELECT e.first\_name, e.last\_name, d.department\_name

FROM employees e

LEFT JOIN department d ON e.department\_id = d.department\_id;

--Join OPERATIONS

SELECT first\_name, last\_name

FROM employees

WHERE departments\_id = 1

UNION

SELECT first\_name, last\_name

FROM employees

WHERE departments\_id = 2;

SELECT e1.first\_name, e1.last\_name

FROM employees e1

WHERE e1.departments\_id = 1

INTERSECT

SELECT e2.first\_name, e2.last\_name

FROM employees e2

WHERE e2.departments\_id = 2;

CREATE VIEW hr\_employees AS

SELECT first\_name, last\_name, salary

FROM employees

WHERE department\_id = 1;

SELECT \* FROM hr\_employees;

COMMIT;

ROLLBACK;

SAVEPOINT my\_savepoint;

ROLLBACK TO my\_savepoint;

**Experiment No: 08**

**SQL Procedures and Functions**

SHOW DATABASES;

CREATE DATABASE IF NOT EXISTS Library;

USE Library;

CREATE TABLE IF NOT EXISTS BOOKS(

Book\_id INT PRIMARY KEY,

Tilte VARCHAR ( 255),

Author VARCHAR(255),

PublicationYear INT

);

INSERT INTO Books(BookID, Title, Author, PublicationYear)

VALUES

(1, ‘ The Catcher’, ‘ J.D. Salinger’, 1951),

(2, ‘ to kill ‘, ‘harper’,1950);

SHOW TABLES;

DESC books;

SELECT\*FROM books;

DELIMITER //

CREATE PROCEDURE GetBookDetails(IN book\_id INT)

BEGIN

SELECT \* FROM Books WHERE BookID = book\_id;

END //

DELIMITER;

CALL GetBookDetails(2);

CREATE FUNCTION GetAuthorByBOOKID(book\_id) RETURNS VARCHAR(255) DETERMINISTIC BEGIN

DECLARE author\_name VARCHAR(255);

SELECT Author INTO author\_name

FROM Books

WHERE BookID = book\_id;

RETURN author\_name;

END

DELIMITER

SELECT BOOKID, Title, GETAuthoerByBookID(BookID) AS Author

FROM Books;

**Experiment No: 09**

**SQL Triggers and Cursors**

CREATE DATABASE Library;

USE Library;

CREATE TABLE IF NOT EXISTS BOOKS(

Book\_id INT PRIMARY KEY,

Tilte VARCHAR ( 255),

Author VARCHAR(255),

PublicationYear INT

);

INSERT INTO Books(BookID, Title, Author, PublicationYear)

VALUES

(1, ‘ The Catcher’, ‘ J.D. Salinger’, 1951),

(2, ‘ to kill ‘, ‘harper’,1950);

SHOW TABLES;

DELIMITTER //

CREATE TRIGGER update\_total\_books

BEFORE INSERT ON books

FOR EACH ROW

BEGIN

SET NEW.total\_books = NEW.totaal\_books + NEW.available\_books;

END

DELIMITTER;

INSERT INTO books VALUES(3, ‘1994’, ‘george orwell’, 10, 10);

DELIMITTER//

CREATE PROCEDURE display\_books()

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE bookTitle VARCHAR(255);

DECLARE bookAuthor VARCHAR(255);

DECLARE totalBooks INT;

DECLARE available INT;

DECLARE cur CURSOR FOR SELECT title, author, total\_books, available\_books FROM v=books;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN cur;

Read\_loop: LOOP

FETCH cur INTO bookTITLW, BookAUTHOR, totalBooks, availableBooks;

IF done THEN

LEAVE read\_loop;

END IF;

SELECT CONCAT(‘TiTle: ‘, bookTitle, ‘AUTHOR: ‘, bookAuthor, ‘Total Books: ‘, totalBooks’ ‘Available Books:’, availableBooks) AS BookINFO;

END LOOP;

CLOSE cur;

END;

DELIMITER;

CALL display\_books();