SOFTWARE LAB 4

SQL

Program: - 1

TABLE CREATION

a. Create student table.

Aim: - Create a table student with the following fields: Rollno, Name, Age, Mark1, Mark2, Mark3, City. Rollno is the primary key, All other fields should not contain NULL. Insert 5 rows of details and display all data.

```
SQL> create table student(Rollno int not null,Name varchar(25) not null,Age int not null, Mark1
        int not null,Mark2 int not null,Mark3 int not null,City varchar(25) not null,primary
        key(Rollno));

SQL> desc student;
```

b. Create staff table.

Aim: - Create a staff table with the following fields: ,Staff_ID, Staff_Name, Department, Age. Staff_id is the primary key. Age field should >25. All fields not be NULL.

```
SQL> create table staff(Staff_ID int not null,Staff_Name varchar(25) not null,Department
        varchar(25) not null, Age int not null check(Age>25), primary key(Staff_ID));

SQL> desc staff;
```

c. Create employee table.

Aim: - Create table employee with the following fields: Emp_ID, Name, DOB, Designation, Salary. Emp_ID is the primary key. All fields not to be NULL.

```
SQL> create table employee(Emp_ID int not null primary key,Name varchar(25) not null,DOB
        date not null,Designation varchar(25) not null,Salary int  not null);
SQL> desc employee;
```

Program: - 2

INSERT VALUES INTO TABLES

a. Insert values into student table.

```
SQL> insert into student values(1,'Anu',20,79,80,82,'Idukki');
SQL> insert into student values(2,'Manu',22,72,78,80,'Kottayam');
SQL> insert into student values(3,'Alexa',19,60,65,70,'Pala');
SQL> insert into student values(4,'Jeeva',20,67,78,89,'Kochi');
SQL> insert into student values(5,'Sam',22,78,75,81,'Kollam');
SQL> select * from student;
```

b. Insert values into staff table.

```
SQL> insert into staff values(101,'Thomas','Sales',30);
SQL> insert into staff values(102,'Mathew','Marketing',35);
SQL> insert into staff values(103,'Nandhu','Accounts',38);
SQL> insert into staff values(104,'Seema','Sales',36);
```

SQL> insert into staff values(105,'Bibin','Loan',29);

SQL> select * from staff;

c. Insert values into employee table.

SQL> insert into employee values(1001,'Miya','20-09-1991','Manager',37000);

SQL> insert into employee values(1002,'Anu','11-06-1992','Accountant',27000);

SQL> insert into employee values(1003,'Ajith','27-12-1990','Executive',16000);

SQL> insert into employee values(1004,'Sanu','07-09-1992','Executive',16000);

SQL> insert into employee values(1005,'Amal','15-03-1991','Accountant',28000);

SQL> select * from employee;

Program: - 3

RETRIEVING RECORDS FROM THE TABLE

a. List student names and marks from student table.

SQL> select Name,Mark1,Mark2,Mark3 from student;

b. List all student whose age is less than 20 from student table.

SQL> select * from student where Age<20;

c. List all Staff_Name and Department which works in Sales from staff table.

SQL> select Staff_Name,Department from staff where Department="Sales";

d. List employee details whose Salary is greater than 20000 from employee table.

SQL>select * from employee where Salary>20000;

e. List Emp_ID and Name from employee table whose Designation is Manager.

SQL> select Emp_ID,Name from employee where Designation="Manager";

Program: - 4

UPDATING RECORDS IN THE TABLE

a. Modify the value of City to Kochi of student whose Rollno is 5 in student table.

SQL>update student set City="Kochi" where rollno=5;

SQL> select Rollno,Name,City from student;

b. Change the age of staff who are in Sales department to '35' in staff table.

SQL> update staff set Age=35 where Department="Sales";

SQL> select * from staff;

c. Update the salary of Executives to 20000 in employee table.

SQL> update employee set Salary=20000 where Designation="Executive";

SQL> select * from employee;

Program: - 5

DELETING RECORDS FROM THE TABLE

a. Delete all students whose Mark1 is less than 70 from student table.

SQL> delete from student where Mark1<75;

SQL> select * from student;

b. Delete all staff who is working in Sales department from staff table.

SQL> delete from staff where Department="Sales";

SQL> select * from staff;

c. Delete all employee whose salary is equal to 20000.

SQL> delete from employee where Salary=20000;

SQL> select * from employee;

Program: - 6

ALTERING THE TABLE STRUCTURE

a. Add a column called 'Phone' of data type int to the student table.

SQL> alter table student add Phone int;

SQL> desc student;

b. Change the size of Department column in staff table to 35.

SQL> alter table staff modify Department varchar(35);

SQL> desc staff;

c. Add a new column 'City' of data type varchar and size 15 in employee table.

SQL> alter table employee add City varchar(15);

SQL> desc employee;

Program: - 7

DELETING THE TABLE ALONG WITH DATA

a. Destroy the student table along with the data.

SQL> drop table student;

b. Delete table employee along with data;

SQL> drop table employee;

Program: - 8

SORTING DATA IN A TABLE

a. Retrieve all rows in student table and display it on ascending order based on Names.

SQL> select * from student order by Name;

b. Retrieve Staff_Name and Age in staff table and display it on descending order based on Age.

SQL> select Staff_Name,Age from staff order by age desc;

Program: - 9

COMPUTATIONS ON TABLE DATA

a. Display details of all employees in employee table and add '500' rupees to each employee and retrieve new salary as 'Increment Salary'.

SQL> select Name,Salary+500 'Increment Salary' from employee;

b. Display content of all students in student table and add '20' marks to Mark3 of each student and retrieve new mark as 'Increment Mark3'.

SQL> select Name,Mark3+20 'New Mark3' from student;

Program: - 10

LOGICAL OPERATIONS

- **AND Operator**

a. List the employee details whose designation is executive and salary greater than 15000.

SQL> select * from employee where Designation='Executive' and Salary>15000;

- **OR Operator**

a. List all students whose city Idukki or Kochi.

SQL> select * from student where City='Idukki' or City='Kochi';

- **NOT Operator**

a. List all students whose cities are not in Idukki or Kottayam

SOL> select * from student where not(City='Idukki' or City='Kottayam');

❖ **BETWEEN Operator**

a. List all the employees whose salary is between 15000 and 20000.

SQL> select * from employee where Salary between 15000 and 20000;

b. List Rollno, Name, Age and Mark3 of all students whose Mark3 is between 75 and 85.

SQL> select Rollno,Name,Age,Mark3 from student where Mark3 between 75 and 85;

Program: - 11

AGGREGATE FUNCTIONS

a. Find sum of all Marks in student table.

SQL> select sum(Mark1),sum(Mark2),sum(Mark3) from student;

b. Find sum of salary of employees and display with new column heading as Total.

SQL> select sum(Salary) as Total from employee;

c. Find average of salary of employees.

SQL> select avg(Salary) from employee;

d. Find count of ages in student table.

SQL> select count(Age) from student;

e. Find count of ages in student table and avoid duplicate values.

SQL> select count(distinct Age)from student;

f. Find minimum salary of employee.

SQL> select min(Salary) from employee;

g.Find maximum salary of employee.

SQL> select max(Salary) as Maximum_Salary from employee;

Program: - 12

NESTED QUERIES

a. Retrieve employee details whose salary is above average salary.

SQL> select * from employee where Salary>(select avg(Salary) from employee);

b. List Name, Age and Mark1 of all students whose Mark1 is greater than or equal to average of Mark1.

SQL> select Name,Age,Mark1 from student where Mark1>=(Select avg(Mark1) from student);

c. List all staff details whose age is less than average age of all staffs.

SQL> select * from staff where Age<(select avg(Age) from staff);

Program: - 13

RENAME TABLE

a. Rename the table student to stud.

SQL> rename table student to stud;

SQL>select * from stud;

b. Rename the table staff to staffs.

SQL> rename table staff to staffs;

SQL>select * from staffs;

Program: - 14

DATE AND TIME FUNCTIONS

a. Get current date and time.

```
SQL> select now();
```

b. Get current date only.

```
SQL>select curdate();
```

c. Get current time only.

```
SQL>select curtime();
```

PL/SQL

1. PL/SQL Program to Check Number is Odd or Even

```
declare
   n number := &n;
begin
   if mod(n, 2) = 0 then
      dbms_output.put_line('number is even');
   else
      dbms_output.put_line('number is odd');
   end if;
end;
/
```

2. PL/SQL Program to find greatest among three numbers.

```
declare
  a number := &a;
  b number := &b;
  c number := &c;
  large number;
begin
  large := a;
  if b > large then
    large := b;
  end if;
  if c > large then
    large := c;
  end if;
  dbms_output.put_line('greatest: ' || large);
end;
/
```

3. PL/SQL Program to print n even numbers.

```
declare
  n   number := &n;
begin
  if n < 2 then
```

```
    dbms_output.put_line('no even numbers in that range.');
  else
    for i in 2 .. n loop
      if mod(i, 2) = 0 then
        dbms_output.put_line(i);
      end if;
    end loop;
  end if;
end;
/
```

4. PL/SQL program to calculate factorial of a number using function.

```
create or replace function factorial(x number)
 return number
is
 f number;
begin
 if x = 0 then
  f := 1;
 else
  f := x * factorial(x - 1);
 end if;
 return f;
```

```
end;
/

declare
  num number;
  result number;
begin
  num := 5;
  result := factorial(num);
  dbms_output.put_line('factorial of ' || num || ' is ' || result);
end;
/
```

5. PL/SQL program to find sum of two numbers using function.

```
function add(num1 number, num2 number)
  return number
is
  sum number;
begin
  sum := num1 + num2;
  return;
end;
/
```

```
declare
  num1 number := 10;
  num2 number := 20;
  result number;
begin
  result := add(num1, num2);
  dbms_output.put_line('the sum of ' || num1 || ' and ' || num2 || ' is: ' || result);
end;
/
```

6. PL/SQL program to create a simple procedure that displays a message.

```
CREATE OR REPLACE PROCEDURE greetings
AS
BEGIN
   dbms_output.put_line('Hello World!');
END;
/
SQL> EXECUTE greetings;
```

7. PL/SQL program to find the minimum of two values using procedure.

```
declare
   a number;
   b number;
   c number;
```

```
procedure findmin(x in number, y in number, z out number) is
begin
   if x < y then
      z:= x;
   else
      z:= y;
   end if;
end;
begin
   a:= 23;
   b:= 45;
   findmin(a, b, c);
   dbms_output.put_line(' minimum of (23, 45) : ' || c);
end;
/
```