DBMS Practical’s

```
-- Practical 1 (Create, Alter, Rename, Truncate, Drop Commands.) (DDL)

-- 1. Create
CREATE TABLE Empolyee(
    EmpId int,
    EmpName varchar(50),
    EmpDpt varchar(50),
    Salary int
);

-- DESC
DESC Empolyee1;

-- 2. Alter (add, modify, drop)
ALTER TABLE Empolyee add EmpAdd varchar(50);
ALTER TABLE Empolyee modify EmpName varchar(20);
ALTER TABLE Empolyee drop column EmpAdd;

-- 3. Rename
Rename Empolyee to Empolyee1;

-- 4. Truncate (Deletes all the rows)
TRUNCATE TABLE Empolyee1;

-- 5. Drop (Deletes the entire table)
DROP TABLE Empolyee1;
```

```
-- Practical 2 (Insert, Select, Update, Delete Commands.) (DML)

CREATE TABLE People(
    ID int,
    Name varchar(20),
    Address varchar(50),
    PhoneNumber number
);

DESC People;

-- 1. Insert
INSERT into People values(1, 'Kamal', 'Wadala', 9876543210)
INSERT into People values(2, 'Amaan', 'Sion', 1234567890)
INSERT into People values(3, 'Hari Om', 'Wadala', 0987612345)
INSERT into People values(4, 'Om Babu', 'Wadala', 1234509876)
INSERT into People values(5, 'Sahil', 'Wadala', 980748390)

-- 2. Select (*, Where, ColumnName)
SELECT *from People
SELECT *from People where ID = 1
SELECT id, Name from People

-- 3. Update (Set all, Where)
```

```
UPDATE People SET PhoneNumber = 1234567890
UPDATE People SET PhoneNumber = 0987654321 Where ID = 1

-- 4. Delete (All, Where)
DELETE from People
DELETE from People where ID = 3
```

```
/* Practical 3 Types of Constraints (Primary Key, Not Null, Unique,
Foreign key, Check, Default.) */

-- CREATE TABLE <table_name>(column1 data_type, column2 data_type,
-- REFERENCES[primary_key_tablename] (column_list_of_primary_key_table));

CREATE TABLE BROSDUH(
    Rollno int PRIMARY KEY,
    Name varchar(20) NOT NULL,
    PhoneNumber int UNIQUE,
    Age int CHECK (Age > 0),
    Major varchar(20) DEFAULT 'BSC IT'
);

DESC BROSDUH;

INSERT into BROSDUH values(1, 'Kamal', 1234567890, 17, '')
INSERT into BROSDUH values(2, 'Amaan', 0978654321, 18, '')
INSERT into BROSDUH values(3, 'Om Babu', 5477844746, 17, '')
INSERT into BROSDUH values(4, 'Sahil', 0796969869, 18, '')
INSERT into BROSDUH values(5, 'Hari Om', 9079565847, 19, '')

SELECT *from BROSDUH
```

```
-- Practical 4B (Functions)

-- Display minimum salary of employee.
Select min(sal) from Emp

-- Display maximum salary of employee.
Select max(sal) from Emp

-- Display average salary of employee who is working as Analyst.
Select avg(sal) from Emp where job='Analyst'

-- Calculates sum of salary for each department.
Select sum(sal), job from Emp group by job;

-- Lists the maximum salary drawn under each job category.
Select max(sal), job from Emp group by job;

---

-- Display total number of employees from Employee table.
Select count(empno) from Emp;

-- Display the total number of employees in each job category.
Select count(empno), job from Emp group by job;

-- Display maximum salary of employee having job President.
Select max(sal) from Emp group by job having job='President'
```

```
-- Display the details of employees in their ascending order of names.
Select * from Emp order by ename;

-- Display the details of employees in their descending order of their salary.
Select * from Emp order by sal desc;
```

```
-- Practical 5 (Joins, Types of Joins and Subqueries)

-- Natural Join -
select EmpName,DName from Employee1 NATURAL JOIN Department

-- Cross Join -
select *from Employee1 CROSS JOIN Department

-- Left Join -
select EmpNo, EmpName, DName from Employee1 LEFT OUTER JOIN Department on(Employee1.DeptNo=Department.DeptNo)

-- Right Join -
select EmpNo, EmpName, DName from Employee1 RIGHT OUTER JOIN Department on(Employee1.DeptNo=Department.DeptNo)

-- Full Join -
select EmpNo, EmpName, DName from Employee1 FULL OUTER JOIN Department on(Employee1.DeptNo=Department.DeptNo)
```

```
-- Practical 6 (View, Operations on View & Its Types)
-- View = Virtual Table
-- Types = Read Only() & Updateable View
-- CREATE VIEW view_name as SELECT *FROM table_name WHERE condition (views with same name are not allowed)
-- create view emp_view as (select *from employee1 where age > 30) with read only <-- This is Read Only View
-- create view emp_view as select *from employee1 where age > 30 <-- This is Updatable View

create table employee1(
    empid int,
    emp_name char(20),
    city char(20),
    salary int,
    age int
);

insert into employee1 values(1,'Angelina','Chicago',200000,30);
insert into employee1 values(2,'Robert','Austin',300000,26);
insert into employee1 values(3,'Christian','Denver',100000,42);
insert into employee1 values(4,'Kristen','Washington',500000,29);
insert into employee1 values(5,'Russell','Los Angels',200000,36);
insert into employee1 values(6,'Marry','Canada',600000,48);
select *from employee1

create table project(
    project_no int,
    emp_id int,department char(20)
);

insert into project values(101,1,'Testing');
insert into project values(102,2,'Development');
insert into project values(103,3,'Designing');
insert into project values(104,4,'Development');
select *from project

create view emp_view as select *from employee1 where age > 30
select *from emp_view
```

```
create view emp_view1 as select emp_name,salary,age from employee1
select *from emp_view1

create view project_view as select *from project where emp_id=3
select *from project_view

create view project_view1 as select *from project where department='Development'
select *from project_view1

create view emp_view2 as select*from employee1 where salary!=200000
select*from emp_view2

create view emp_view3 as select *from employee1 where emp_name like '___s%' -- IMP
select *from emp_view3

create view emp_view96 as (select emp_name, city from employee1) with read only -- IMP
select *from emp_view96

create view emp_view5 as select *from employee1 where salary=200000
select*from emp_view5

create view emp_view6 as select emp_name, salary from employee1
select *from emp_view6

create view joinview as select *from employee1, project where employee1.empid=project.emp_id
select *from joinview

create view groupview as (select department from project group by department) -- IMP
select *from groupview

create view subsetview as (select emp_name,salary from employee1 where salary>300000) -- IMP
select *from subsetview
```

```
-- Practical 6B (Set Operations)
-- union (Combines two Statements ex.)
-- union all
-- intersect
-- minus

SELECT column_name FROM table_name UNION SELECT column_name FROM table_name
SELECT column_name FROM table_name UNION ALL SELECT column_name FROM table_name
SELECT column_name FROM table_name INTERSECT SELECT column_name FROM table_name
SELECT column_name FROM table_name MINUS SELECT column_name FROM table_name
```

```
-- Practical 7 (PL/SQL - Procedural Language SQL)

-- PL/SQL Struture -
-- Set Serveroutput on
-- DECLARE (optional)
-- BEGIN (main code body)
-- EXCEPTION (optional)
-- END;
-- /

-- Even or Odd
Set serveroutput On

DECLARE
n NUMBER := 1634;
r NUMBER;
```

```
BEGIN
r := MOD(n, 2);
IF r = 0 THEN
dbms_output.Put_line('Even');
ELSE
dbms_output.Put_line('Odd');
END IF;

END;
/

-- Factorial
Set serveroutput On

DECLARE
n NUMBER:= '&n';
f NUMBER:= 1;
t NUMBER;

BEGIN
t:=n;

WHILE(t>0)
LOOP
f:= f*t;
t:= t-1;
END LOOP;

DBMS_OUTPUT.PUT_LINE(f);

END;
/

-- Printing 1 - 10
Set serveroutput On

DECLARE
i NUMBER;

BEGIN
i:= 1;

WHILE(i<=10)
LOOP
DBMS_OUTPUT.PUT_LINE(i);
i:= i+1;
END LOOP;

END;
/

-- Positive, Neutral or Negative
DECLARE
num1 NUMBER := &get_num;

BEGIN
IF num1 < 0 THEN
DBMS_OUTPUT.PUT_LINE ('The number '||num1||' is a negative number');
ELSIF num1 = 0 THEN
DBMS_OUTPUT.PUT_LINE ('The number '||num1||' is equal to zero');
ELSE
DBMS_OUTPUT.PUT_LINE ('The number '||num1||' is a positive number');
END IF;
```

```
END;
/
```

```
-- Practical 8 (PL/SQL - Procedural Language SQL)

-- Square root SQRT()
Set Serveroutput on

DECLARE
n NUMBER := 25;

BEGIN
DBMS_OUTPUT.PUT_LINE(SQRT(n));

END;
/

-- Max (Using Function)
Set Serveroutput on

DECLARE
a NUMBER := 23;
b NUMBER := 45;
c NUMBER;

FUNCTION findMax(x IN number, y IN number)
RETURN number
IS
z number;
BEGIN IF x > y THEN
z:= x;
ELSE
Z:= y;
END IF;
RETURN z;
END;

BEGIN
c := findMax(a, b);

dbms_output.put_line(' Maximum of (23,45): ' || c);

END;
/

-- Add (using Function)
Set Serveroutput on

create or replace function adder(n1 in number, n2 in number)
return number
is
n3 number(8);
begin
 n3 :=n1+n2;
return n3;
end;
/

DECLARE
n3 number(2);
```

```
BEGIN
n3 := adder(11,22);
dbms_output.put_line('Addition is: ' || n3);

END;
/
```