SQL & PL/SQL LAB PROGRAMS

1. DDL PROGRAM

```
---------------------------------
CREATE TABLE student (
  id NUMBER,
  name VARCHAR2(20),
  marks NUMBER
);

INSERT INTO student VALUES (1,'Arun',80);
INSERT INTO student VALUES (2,'Priya',75);
INSERT INTO student VALUES (3,'Kiran',90);

SELECT * FROM student;

ALTER TABLE student ADD grade VARCHAR2(5);

UPDATE student SET grade='A' WHERE marks>=85;
UPDATE student SET grade='B' WHERE marks<85;

SELECT * FROM student;
```

2. CONSTRAINTS PROGRAM

```
---------------------------------
CREATE TABLE department (
  dept_id NUMBER PRIMARY KEY,
  dept_name VARCHAR2(20) NOT NULL
);

INSERT INTO department VALUES (1,'CS');
INSERT INTO department VALUES (2,'IT');

CREATE TABLE employee (
  emp_id NUMBER PRIMARY KEY,
  emp_name VARCHAR2(20),
  salary NUMBER CHECK (salary>0),
  dept_id NUMBER,
  FOREIGN KEY (dept_id) REFERENCES department(dept_id)
);

INSERT INTO employee VALUES (101,'Ravi',20000,1);
INSERT INTO employee VALUES (102,'Meena',25000,2);

SELECT * FROM department;
SELECT * FROM employee;
```

3. BUILT-IN FUNCTIONS

```
---------------------------------
CREATE TABLE product (
  p_id NUMBER,
  p_name VARCHAR2(20),
  price NUMBER
);
INSERT INTO product VALUES (1,'Pen',10);
```

```
INSERT INTO product VALUES (2,'Book',50);
INSERT INTO product VALUES (3,'Bag',500);

SELECT UPPER(p_name) FROM product;
SELECT MAX(price) FROM product;
SELECT AVG(price) FROM product;
SELECT COUNT(*) FROM product;
```

4. JOIN PROGRAM

```
---------------------------------
CREATE TABLE dept (
  dept_id NUMBER PRIMARY KEY,
  dept_name VARCHAR2(20)
);

CREATE TABLE staff (
  staff_id NUMBER PRIMARY KEY,
  staff_name VARCHAR2(20),
  dept_id NUMBER,
  FOREIGN KEY (dept_id) REFERENCES dept(dept_id)
);

INSERT INTO dept VALUES (1,'CS');
INSERT INTO dept VALUES (2,'IT');

INSERT INTO staff VALUES (1,'Arun',1);
INSERT INTO staff VALUES (2,'Priya',2);

SELECT dept.dept_name, staff.staff_name
FROM dept
JOIN staff
ON dept.dept_id = staff.dept_id;
```

5. SUBQUERY PROGRAM

```
---------------------------------
CREATE TABLE emp (
  emp_id NUMBER,
  emp_name VARCHAR2(20),
  salary NUMBER
);

INSERT INTO emp VALUES (1,'Ravi',20000);
INSERT INTO emp VALUES (2,'Meena',30000);
INSERT INTO emp VALUES (3,'Arun',25000);

SELECT emp_name
FROM emp
WHERE salary = (SELECT MAX(salary) FROM emp);
```

6. DECISION MAKING PROGRAM

```
---------------------------------
SET SERVEROUTPUT ON;

CREATE TABLE marks_table (
```

```
  student_name VARCHAR2(20),
  marks NUMBER
);

INSERT INTO marks_table VALUES ('Arun',80);

DECLARE
  m NUMBER;
BEGIN
  SELECT marks INTO m FROM marks_table WHERE student_name='Arun';

  IF m>=50 THEN
    DBMS_OUTPUT.PUT_LINE('Pass');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Fail');
  END IF;
END;
/
```

7. LOOP PROGRAM

```
---------------------------------
SET SERVEROUTPUT ON;

CREATE TABLE numbers (
  num NUMBER
);

INSERT INTO numbers VALUES (1);
INSERT INTO numbers VALUES (2);
INSERT INTO numbers VALUES (3);

DECLARE
BEGIN
  FOR i IN (SELECT num FROM numbers) LOOP
    DBMS_OUTPUT.PUT_LINE(i.num);
  END LOOP;
END;
/
```

8. COMPOSITE DATATYPE PROGRAM

```
---------------------------------
SET SERVEROUTPUT ON;

CREATE TABLE student_marks (
  id NUMBER,
  name VARCHAR2(20),
  marks NUMBER
);

INSERT INTO student_marks VALUES (1,'A',90);
INSERT INTO student_marks VALUES (2,'B',80);
INSERT INTO student_marks VALUES (3,'C',70);

DECLARE
  TYPE mark_array IS VARRAY(3) OF NUMBER;
```

```
  m mark_array := mark_array(90,80,70);
BEGIN
  FOR i IN 1..m.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE(m(i));
  END LOOP;
END;
/
```

9. TRIGGER PROGRAM

```
---------------------------------
CREATE TABLE employee_trigger (
  id NUMBER,
  name VARCHAR2(20),
  salary NUMBER
);

CREATE OR REPLACE TRIGGER check_sal
BEFORE INSERT ON employee_trigger
FOR EACH ROW
BEGIN
  IF :NEW.salary <= 0 THEN
    RAISE_APPLICATION_ERROR(-20001,'Invalid Salary');
  END IF;
END;
/

INSERT INTO employee_trigger VALUES (1,'Ravi',20000);
```

10. PROCEDURE PROGRAM

```
---------------------------------
SET SERVEROUTPUT ON;

CREATE TABLE greet_table (
  name VARCHAR2(20)
);

INSERT INTO greet_table VALUES ('Surya');

CREATE OR REPLACE PROCEDURE greet
IS
  n VARCHAR2(20);
BEGIN
  SELECT name INTO n FROM greet_table WHERE ROWNUM=1;
  DBMS_OUTPUT.PUT_LINE('Hello '||n);
END;
/

BEGIN
  greet;
END;
/
```