PROGRAM -1

AIM: To create a database containing table employee with employee details.Write PLSQL to update the experience level of employee as beginner,intermediate and advanced.

CODE:

```
create database company;
use company;
create table emp(emp_id int primary key,emp_name varchar(20),salary varchar(20));
create table dept(dept_id int primary key,emp_id int,designation varchar(20),experience int(10) ,foreign key(emp_id) references emp(emp_id));
insert into emp(emp_id,emp_name,salary)values(101,'Shibu',25000);
insert into emp(emp_id,emp_name,salary)values(102,'Raju',35000);
insert into emp(emp_id,emp_name,salary)values(103,'Shanku',50000);

select * from emp;

insert into dept(dept_id,emp_id,designation,experience)values(201,101,'Peon',2);
insert into dept(dept_id,emp_id,designation,experience)values(202,102,'Clerk',6);
insert into dept(dept_id,emp_id,designation,experience)values(203,103,'Manager',12);

select * from dept;

create table level(emp_id int,dept_id int,experience_level varchar(20),foreign key(emp_id) references emp(emp_id),foreign key(dept_id) references dept(dept_id));

call exp(2,101,201);
call exp(6,102,201);
call exp(12,103,203);

select * from level;
select emp.emp_name,emp.salary,new_salary(level.experience_level,emp.salary) from emp,level where emp.emp_id=level.emp_id;



////STORED PROCEDURE

CREATE DEFINER=`root`@`localhost` PROCEDURE `exp`(experience int,emp_id
```

```
int,dept_id int)
BEGIN
DECLARE
   levels varchar(45);

if (experience > 0 && experience<5)
   then set levels = 'beginner';
   insert into employe(emp_id,experience,salary,levels)
values(emp_id,experience,salary,levels);
   end if;
   if( exp>=6 && exp <10)
   then set levels = 'intermediate';

    insert into employe(emp_id,experience,salary,levels)
values(emp_id,experience,salary,levels);
   end if;
   if (exp >= 10)
   then set levels = 'Experienced';

   insert into employe(emp_id,experience,salary,levels)
values(emp_id,experience,salary,levels);
   end if;
END

////FUNCTION////

CREATE DEFINER=`root`@`localhost` FUNCTION `new_salary`(experience_level
varchar(20),sal varchar(10)) RETURNS int(11)
BEGIN
if(experience_level = 'Experienced')
then
return(sal+1000);
else
return(sal);
end if;

RETURN 1;
END
```

OUTPUT:

```
27 •  select * from level;
28 •  select emp.emp_name,emp.salary,new_salary(level.experience_level,em
29
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	dept_id	experience_level
101	201	Beginner
102	201	Intermediate
103	203	Experienced

```
26
27 •  select * from level;
28 •  select emp.emp_name,emp.salary,new_salary(level.experier
29
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_name	salary	new_salary(level.experience_level,emp.salary)
Shibu	25000	25000
Raju	35000	35000
Shanku	50000	51000

PROGRAM -2

AIM: Given an integer i, write a PL/SQL procedure to insert the tuple (i, 'xxx') into a given relation

CODE:

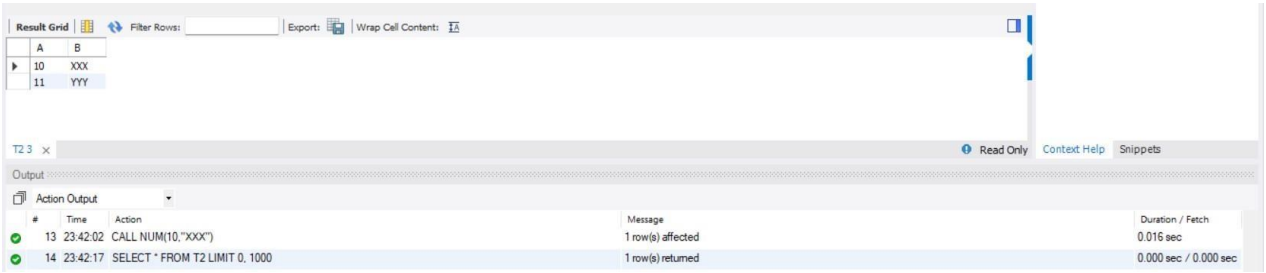
```
CREATE DATABASE NUMBER;
USE NUMBER;
CREATE TABLE T2(A INT ,
                B CHAR(10) );
DROP TABLE T2;
CALL NUM(10,"XXX");
CALL NUM(11,"YYY");

SELECT * FROM T2;
SHOW DATABASES;
```

STORED PROCEDURE :

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `NUM`(I INT,J CHAR(10))
BEGIN
IF(SELECT A FROM T2 WHERE A LIKE (I))
THEN
INSERT INTO T2 (A,B) VALUES (NULL,NULL);
ELSE
INSERT INTO T2 (A,B) VALUES (I,J);
END IF;
END
```

OUTPUT:



PROGRAM-3

AIM: To write a PL/SQL block to calculate the incentive of an employee whose ID is 110

CODE:

Table creation and insertion

```
CREATE TABLE employee(id int,basic double,hra double);
INSERT INTO employee VALUES(101,12000,3200),(102,15000,3200);
```

Function call

```
SELECT *,incentive(id) FROM employee
```

FUNCTION :

```
CREATE DEFINER=`root`@`localhost` FUNCTION `incentive`(id1 int) RETURNS
double
BEGIN
 DECLARE bp double;
 DECLARE h double;
 DECLARE inc double;
 SELECT basic INTO bp
        FROM employee
        WHERE id=id1;
 SELECT hra INTO h
        FROM employee
        WHERE id=id1;
 if(bp>10000) then
        set inc=bp+h+1200;

 else
 set inc=bp+h+4500;

 end if;
RETURN inc;
END
```

OUTPUT:

✔	63	22:44:50	create table employee(id int,basic double,hra double)	0 row(s) affected
✔	64	22:44:50	insert into employee values(101,12000,3200),(102,15000,3200)	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0

Function call :

✔	67	22:48:17	call insert1(1,"new")	1 row(s) affected

PROGRAM-4

AIM: To create the Book database and do the following: (Consider the attributes based on the question given)

book(book_name, author_name, price,quantity)

a. Write a query to update the quantity by double in the table book.
b. List all the book_name whose price is greater than those of book named "Database for Dummies"
c. Retrieve the list of author_name whose first letter is 'a' along with the book_name and price (Explore more about *Like* keyword)
d. Write a PL/SQL Procedure to find the total number of books of same author

CODE:

```
create database books;
use books;
create table book_info(book_name varchar (20),author varchar(20),price int,quantity
int);
insert into book_info values('randamoozham','MT',300,5);
insert into book_info values('ikigai','hector',500,7);
insert into book_info values('databse of dummies','xyz',250,7);
insert into book_info values('wings of flare','APJ',500,7);
insert into book_info values('oopol','MT',270,3);
select * from book_info;
```

a)
```
set sql_safe_updates=0;
update book_info  set quantity=quantity*2;
```

b)
```
select book_name from book_info where price>(select price from book_info
where   book_name='databse of dummies');
```

c)
```
select author,book_name,price from book_info where author like 'a%';
```

OUTPUT:

10	22:20:38	SET SQL_SAFE_UPDATES = 0	0 row(s) affected
11	22:20:40	update book set quantity = quantity * 2	5 row(s) affected Rows matched: 5 Changed: 5 Warnings: 0

book_name
hk
Pyari
Potte

author_name	book_name	price
Amal	harry	200
Arun	hk	430
AAA	Potte	900

totalbooks
7

PROGRAM-5

AIM: Create the Company database with the following tables and do the following:
Administration (employee_salary, development _cost, fund_ amount, turn_over,bonus)
Emp_details (emp_no, emp_name, DOB, address, doj, mobile_no, dept_no, salary).

a. Calculate the total and average salary amount of the employees of each department.

b. Display total salary spent for employees.

c. Develop a PL/SQL function to display total fund_amount spent by the administration department

CODE:

```
CREATE TABLE Administration (
employee_salary double,
development_cost double,
fund_amount double,
turn_over double,
bonus double);

CREATE TABLE Emp_details(
emp_no int,
emp_name varchar(20),
DOB date,
address varchar(20),
doj date,
mobile_no int(12),
dept_no int,
salary double);

INSERT INTO Administration VALUES
(12000,25000,560000,65000,5000),
(70000,55000,860000,15000,1000),
(18000,45000,160000,75000,7000),
(10000,27000,520000,60000,5000),
(18000,27000,360000,35000,3000);

INSERT INTO Emp_details VALUES
(1,"Ram","1999-10-10","Street - 2,vallakadavu","2020-10-10",9865986598,10,12000),
(2,"manoharan","1997-10-10","Street - 2,vallakadavu","2020-10-10",9865986598,10,12200),
(3,"mani","1996-10-10","Street - 2,vallakadavu","2020-10-10",9865986598,11,12500),
(4,"moran","1957-10-10","Street - 2,vallakadavu","2020-10-10",9865986598,11,17200),
```

(5,"sasi","1948-10-10","Street - 2,vallakadavu","2020-10-10",9865986598,12,12090)
(6,"kaka","1988-10-10","Street - 2,vallakadavu","2020-10-10-",9865986598,12,12050);

OUTPUT

a. SELECT

dept_no,
avg(salary) 'Average salary',sum(salary) 'Total Salary'
FROM Emp_details GROUP BY dept_no;

	dept_no	Average salary	Total Salary
▶	10	12100	24200
	11	14850	29700
	12	12070	24140

b) SELECT
sum(salary) 'SUM OF SALARY'
FROM Emp_details;

OUTPUT

Result Grid | Filter Rows:

	SUM OF SALARY
▶	78040

c)

//FUCTION//

```
CREATE DEFINER=`root`@`localhost` FUNCTION `fund_total`() RETURNS
double
BEGIN
DECLARE f DOUBLE;
 DECLARE i DOUBLE;
 SELECT SUM(fund_amount)
FROM Admins;
RETURN f;
END
```

//FUNCTION CALL//

```
SELECT fund_total() from Admins LIMIT 1;
```

OUTPUT

Result Grid Filter Rows:

	fund_total()
▶	2460000

PROGRAM-6

AIM: To write a program to implement trigger

CODE:

```
create database employees;
use employees;

create table employee(emp_id int,emp_name varchar(10),department_name
varchar(15));
insert into employee values(101,"Coen","mca");
insert into employee values(102,"aloe","mca");
insert into employee values(103,"Raimi","btech");
insert into employee values(104,"Ras","mca");
create table dpt_mca(dept_id int,dept_name varchar(20), dept_emp
varchar(15));
create table dpt_cs(dept_id int,dept_name varchar(20), dept_emp varchar(15));

select * from employee;
insert into employee values(105,"Anu","mca");
select * from dpt_mca;
insert into employee values(106,"R0se","CS");
select * from dpt_cs;


///TRIGGER///

CREATE DEFINER=`root`@`localhost` TRIGGER
`employees`.`employee_BEFORE_INSERT` BEFORE INSERT ON
`employee` FOR EACH ROW
BEGIN
if new.department_name="mca" then
 INSERT INTO dpt_mca set
dept_id=new.emp_id,dept_name=new.emp_name,dept_emp="Asst.proff fill";
 end if;
 if new.department_name="cs" then
  INSERT INTO dpt_cs set
dept_id=new.emp_id,dept_name=new.emp_name,dept_emp="Asst.proff fill";
 end if;
END
```

OUTPUT:

```
14 •  select * from dpt_mca;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

dept_id	dept_name	dept_emp
104	Ras	Asst proff fill
105	Anu	Asst.proff fill

```
16 •  select * from dpt_cs;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

dept_id	dept_name	dept_emp
106	R0se	Asst proff fill

PROGRAM-7

AIM: . To create a student record database in which student marks assessment is recorded.In such schema,create a trigger so that the total and average of specified marks is automatically inserted whenever a record is inserted.

CODE:

```
create database students;
use students;
create table student(rollno varchar(10) primary key,studname varchar(10),sub1
varchar(10),sub2 varchar(10),sub3 varchar(10));
create table result(rollno varchar(10),studname varchar(10),total_marks
varchar(10),percentage varchar(10));
insert into student values("1","abhi","35","55","85"),
("2","adarsh","15","60","10"),
("3","anu","96","99","94");

insert into student values(8,"jain",67,90,76);
insert into student values(10,"bini",60,96,50);
select * from student;

select * from result;

///TRIGGER///

CREATE DEFINER=`root`@`localhost` TRIGGER
`students`.`student_AFTER_INSERT` AFTER INSERT ON `student` FOR
EACH ROW
BEGIN
declare total varchar(10);
declare perc varchar(10);
set total=new.sub1+new.sub2+new.sub3;
set perc=((total/300)*100);
insert into result values(new.rollno,new.studname,total,perc);
END
```

OUTPUT:

```
select * from result;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

rollno	studname	total_marks	percentage
8	jain	233	77.6666667
10	bini	206	68.6666667

PROGRAM-8

AIM: To write a program to implement cursors

CODE:

```
create database college1;
use college1;
create table library (shelf_no int(10),category varchar(10),book_name
varchar(20));
insert into library values(11,'science','algebra');
insert into library values(12,'science','Data Mining');
insert into library values(21,'comic','New Avengers');
insert into library values(22,'comic','Spiderman');
insert into library values(31,'drama','romeo and juliet');
insert into library values(32,'drama','hamlet');
create table book_by_order(book_shelf int (10),book_category
varchar(20),bookname varchar(20));

select * from library;


call book_details();
select * from book_by_order;


///CURSOR///

CREATE DEFINER=`root`@`localhost` PROCEDURE `book_details`()
BEGIN
declare book_shelf int;
declare bookname varchar(20);
declare book_category varchar(10);
declare C_finished integer default 0;
declare C1 cursor for select shelf_no,category,book_name from library;
declare continue handler for not found set C_finished = 1;
open C1;
book_details:loop
if C_finished=1 then
leave book_details;
end if;
```

```
if C_finished = 0 then
Fetch from C1 into book_shelf,book_category,bookname;
if book_category = 'comic' then
insert into book_by_order values(book_shelf,bookname,book_category);
end if;
end if;
end loop;
close C1;
END
```

OUTPUT :

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

book_shelf	book_category	bookname
21	New Avengers	comic
22	Spiderman	comic