ORACLE LAB

BCA-DS-522

Manav Rachna International Institute of Research and Studies

School of Computer Applications

Department of Computer Applications

Submitted By	
Student Name	**Samikshya Ray**
Roll No	**22/FCA/BCA(AI&ML)/038**
Programme	**Bachelor of Computer Applications**
Semester	**5th Semester**
Section/Group	**D/A**
Department	**School Of Computer Applications**
Session/Batch	**2022-25**
Submitted To	
Faculty Name	

Manav Rachna International Institute of Research and Studies
Deemed to be University under section 3 of the UGC Act, 1956

SCHOOL OF COMPUTER APPLICATIONS

INDEX

PL/SQL Programs:-			
16	Write a PL/SQL block to calculate total salary of employee having employee number 100	15	
17	Write a PL/SQL code to find the greatest of three numbers	16	
18	Write a PL/SQL code to print the numbers from 1 to n	17	
19	Write a PL/SQL code to reverse a string using for loop.	18	
20	Write a PL/SQL code to find the sum of n numbers	18	
21	Consider a PL/SQL code to display the empno, ename, job of employees of department number 10.	19	
22	Consider a PL/SQL code to display the employee number & name of top five highest paid employees	20	
23	Consider a PL/SQL procedure that accepts 2 numbers & return addition, subtraction, multiplication & division of two numbers using stored procedure AND local procedure.	21-22	
24	Consider a PL/SQL code that accepts 2 numbers & return addition, subtraction, multiplication & division of two numbers using stored functions and local function.	22-23	
25	Write a PL/SQL block to show the use of NO_DATA FOUND exception.	24	
26	Write a PL/SQL block to show the use of TOO_MANY ROWS exception.	25	
27	Write a PL/SQL block to show the use of ZERO_DIVIDE exception.	26	
28	To create a trigger on the emp table, which store the empno& operation in the table auditor for each operation i.e. Insert, Update & Delete.	27-28	
29	To create a trigger so that no operation can be performed on emp table.	29	

Oracle Lab

SQL Programs:-

1. Create the following tables

Customer

Column_name	Data type	Size	Constraint
SID	Varchar2	4	Primary Key
First_Name	Char	20	
Last_name	Char	20	

Source code

```
CREATE TABLE Customers (
    SID VARCHAR2(4) PRIMARY KEY,
    First_Name CHAR(20),
    Last_Name CHAR(20)
);
```

Output

Customers

SID	First_Name	Last_Name
empty		

Orders

Column_name	Data type	Size	Constraint
Order_ID	Varchar2	4	Primary Key
Order_date	Char	20	
Customer_SID	Varchar2	20	Foreign Key
Amount	Number		Check > 20000

Source Code

```
CREATE TABLE Orders (
    Order_ID VARCHAR2(4) PRIMARY KEY,
    Order_date CHAR(20),
    Customer_SID VARCHAR2(4),
    Amount NUMBER,
    CONSTRAINT fk_customer
        FOREIGN KEY (Customer_SID)
        REFERENCES Customers(SID),
    CONSTRAINT check_amount
        CHECK (Amount > 20000)
);
```

Output

Orders

Order_ID	Order_date	Customer_SID	Amount
empty			

2. Insert five records for each table

Source Code

```
insert into Customers(SID,First_Name,Last_Name)
Values('101','Shruti','Jha');

insert into Customers(SID,First_Name,Last_Name)
Values('102','Srishty','Sharma');

insert into Customers(SID,First_Name,Last_Name)
Values('103','Vritti','Sachdeva');

insert into Customers(SID,First_Name,Last_Name)
Values('104','Samiksha','Ray');

insert into Customers(SID,First_Name,Last_Name)
Values('105','Khushi','Jain');
```

```
insert into Customers(SID,First_Name,Last_Name)
Values('106','Shreya','Thomos');

insert into Customers(SID,First_Name,Last_Name)
Values('107','Meera','Fernandes');
```

Output

Customers

SID	First_Name	Last_Name
101	Shruti	Jha
102	Srishty	Sharma
103	Vritti	Sachdeva
104	Samiksha	Ray
105	Khushi	Jain
106	Shreya	Thomas
107	Meera	Fernandes

Insert records for Orders table

```
insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('1','2-8-2024','101','30000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('2','4-8-2024','102','50000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('3','30-7-2024','103','80000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('4','1-8-2024','105','90000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('5','31-7-2024','105','70000');
insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('6','10-8-2024','106','60000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('7','12-8-2024','107','40000');
```

Output

Orders

Order_ID	Order_date	Customer_SID	Amount
1	2-8-2024	101	30000
2	4-8-2024	102	50000
3	30-7-2024	103	80000
4	1-8-2024	104	90000
5	31-7-2024	105	70000
6	10-8-2024	106	60000
7	12-8-2024	107	40000

3. **Customer_SID column in the ORDERS table is a foreign key pointing to the SID column in the CUSTOMER table.**

Source Code

```
SELECT SID,First_Name,Last_Name,
(SELECT Amount FROM Orders
 WHERE Orders.Customer_SID = Customers.SID) AS Amount FROM Customers;
```

Output

SID	First_Name	Last_Name	Amount
101	Shruti	Jha	30000
102	Srishty	Sharma	50000
103	Vritti	Sachdeva	80000
104	Samiksha	Ray	90000
105	Khushi	Jain	70000
106	Shreya	Thomas	60000
107	Meera	Fernandes	40000

4. **Insert five records for both tables**

Customers Table

```
insert into Customers(SID,First_Name,Last_Name)
Values('101','Shruti','Jha');

insert into Customers(SID,First_Name,Last_Name)
Values('102','Srishty','Sharma');

insert into Customers(SID,First_Name,Last_Name)
Values('103','Vritti','Sachdeva');

insert into Customers(SID,First_Name,Last_Name)
Values('104','Samiksha','Ray');

insert into Customers(SID,First_Name,Last_Name)
Values('105','Khushi','Jain');

insert into Customers(SID,First_Name,Last_Name)
Values('106','Shreya','Thomos');

insert into Customers(SID,First_Name,Last_Name)
Values('107','Meera','Fernandes');
```

Orders Table

```
insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('1','2-8-2024','101','30000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('2','4-8-2024','102','50000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('3','30-7-2024','103','80000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('4','1-8-2024','105','90000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('5','31-7-2024','105','70000');
```

```
insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('6','10-8-2024','106','60000');

insert into Orders(Order_ID,Order_date,Customer_SID,Amount)
Values('7','12-8-2024','107','40000');
```

Output

Customers

SID	First_Name	Last_Name
101	Shruti	Jha
102	Srishty	Sharma
103	Vritti	Sachdeva
104	Samiksha	Ray
105	Khushi	Jain
106	Shreya	Thomas
107	Meera	Fernandes

Orders

Order_ID	Order_date	Customer_SID	Amount
1	2-8-2024	101	30000
2	4-8-2024	102	50000
3	30-7-2024	103	80000
4	1-8-2024	104	90000
5	31-7-2024	105	70000
6	10-8-2024	106	60000
7	12-8-2024	107	40000

5. List the details of the customers along with the amount.:

Source Code

```
SELECT SID,First_Name,Last_Name,
(SELECT Amount FROM Orders
WHERE Orders.Customer_SID = Customers.SID) AS Amount FROM Customers;
```

Output

SID	First_Name	Last_Name	Amount
101	Shruti	Jha	30000
102	Srishty	Sharma	50000
103	Vritti	Sachdeva	80000
104	Samiksha	Ray	90000
105	Khushi	Jain	70000
106	Shreya	Thomas	60000
107	Meera	Fernandes	40000

6. List the customers whose names end with "s".

Source Code

```
SELECT SID, First_Name, Last_Name FROM Customers WHERE Last_Name LIKE '%s';
```

Output

SID	First_Name	Last_Name
106	Shreya	Thomas
107	Meera	Fernandes

7. **List the orders where amount is between 21000 and 30000**

Source Code

```
SELECT Order_ID,Order_date,Customer_SID,Amount FROM Orders WHERE Amount BETWEEN 21000 AND 30000;
```

Output

Order_ID	Order_date	Customer_SID	Amount
1	2-8-2024	101	30000

8. **List the orders where amount is increased by 500 and replace with name "new amount"**

Source Code

```
SELECT Order_ID, Order_date,Customer_SID,Amount + 500 AS "new amount" FROM Orders;
```

Output

Order_ID	Order_date	Customer_SID	new amount
1	2-8-2024	101	30500
2	4-8-2024	102	50500
3	30-7-2024	103	80500
4	1-8-2024	104	90500
5	31-7-2024	105	70500
6	10-8-2024	106	60500
7	12-8-2024	107	40500

9. Display the order_id and total amount of orders

Source Code

```
SELECT Order_ID,  SUM(Amount) AS Total_Amount FROM Orders GROUP BY Order_ID;
```

Output

Order_ID	Total_Amount
1	30000
2	50000
3	80000
4	90000
5	70000
6	60000
7	40000

10. Calculate the total amount of orders that has more than 15000

Source Code

```
SELECT SUM(Amount) AS Total_Amount FROM Orders
WHERE Amount > 15000;
```

Output

Total_Amount
420000

11. Display all the string functions used in SQL

1) CONCAT: Concatenates two or more strings.

```
SELECT CONCAT('Hello', ' ', 'World');
```

2) LENGTH(or LEN): Returns the length of a string.

```
SELECT LENGTH('Hello World');
```

3) SUBSTRING(or SUBSTR): Extracts a substring from a string.

```
SELECT SUBSTRING('Hello World', 1, 5);
```

4) TRIM: Removes leading and trailing spaces from a string.

```
SELECT TRIM( ' Hello World ');
```

5) UPPER: Converts a string to uppercase.

```
SELECT UPPER('Hello World');
```

6) LOWER: Converts a string to lowercase.

```
SELECT LOWER('Hello World');
```

7) REPLACE: Replaces occurrences of a substring within a string.

```
SELECT REPLACE('Hello World', 'World', 'SQL');
```

8) LEFT: Returns the left part of a string up to a specified number of characters.

```
SELECT LEFT('Hello World', 5);
```

9) RIGHT: Returns the right part of a string up to a specified number of characters.

```
SELECT RIGHT('Hello World', 5);
```

10) POSITION: finds the position of a substring within a string.

```
SELECT POSITION('World' IN 'Hello World');
```

11) FORMAT: Formats a string according to a specified format (specific to some DBMS like SQL Server).

```
SELECT FORMAT(GETDATE(), 'yyyy-MM-dd');
```

12. Create the following tables

Student

Column_name	Data type	Size	Constraint
RollNo	Varchar2	20	Primary Key
Name	Char	20	
Class	Varchar2	20	
Marks	Number	6,2	

Source Code

```
CREATE TABLE Student (
    RollNo Varchar2(20) PRIMARY KEY,
    Name Char(20),
    Class Varchar2(20),
    Marks Number(6,2)
);
```

Output

Student

RollNo	Name	Class	Marks
empty			

Student1

Column_name	Data type	Size	Constraint
R_No	Varchar2	20	Primary Key
Name	Char	20	
Class	Varchar2	20	
Marks	Number	6,2	

Source Code

```
CREATE TABLE Student1 (
    R_No Varchar2(20) PRIMARY KEY,
    Name Char(20),
    Class Varchar2(20),
    Marks Number(6,2)
);
```

Output

Student1

R_No	Name	Class	Marks
empty			

13. Display all the contents of student and student1 using union clause

Source Code

```
SELECT RollNo, Name, Class, Marks FROM Student
UNION
SELECT R_No AS RollNo, Name, Class, Marks FROM Student1;
```

Output

RollNo	Name	Class	Marks
1	Ananya	12th	88
1	Shivani	12th	75
2	Shruti	12th	93
2	Srishty	12th	80
3	Priyanka	11th	90
3	Shivpriya	11th	92
4	Kartik	11th	95
4	Samiksha	10th	88
5	Ritika	11th	86
5	Vritti	10th	92

14. Find out the intersection of student and student1 tables

Source Code

```
CREATE TABLE Student
(
    RollNo Varchar2(20) PRIMARY KEY,
    Name Char(20),
    Class Varchar2(20),
    Marks Number(6,2)
);


CREATE TABLE Student1
(
    R_No Varchar2(20) PRIMARY KEY,
    Name Char(20),
    Class Varchar2(20),
    Marks Number(6,2)
);

Insert Into Student(RollNo,Name,Class,Marks) Values(1,'Ananya','12th',88);
Insert Into Student(RollNo,Name,Class,Marks) Values(2,'Shivani','12th',75);
Insert Into Student(RollNo,Name,Class,Marks) Values(3,'Shruti','12th',93);
Insert Into Student(RollNo,Name,Class,Marks) Values(4,'Srishty','12th',80);
Insert Into Student(RollNo,Name,Class,Marks) Values(5,'Priyanka','11th',90);
Insert Into Student(RollNo,Name,Class,Marks) Values(6,'Shivpriya','11th',92);
Insert Into Student(RollNo,Name,Class,Marks) Values(7,'Kartik','11th',95);
```

```
Insert Into Student1(R_No,Name,Class,Marks) Values(1,'Ananya','12th',88);
Insert Into Student1(R_No,Name,Class,Marks) Values(5,'Ritika','11th',86);
Insert Into Student1(R_No,Name,Class,Marks) Values(6,'Virti','10th',92);
Insert Into Student1(R_No,Name,Class,Marks) Values(7,'Kartik','11th',95);

Select RollNo,Name,Class,Marks From Student
INTERSECT
Select R_No As RollNo,Name,Class,Marks From Student1;
```

Output

Output

RollNo	Name	Class	Marks
1	Ananya	12th	88
7	Kartik	11th	95

15. **Display the names of student and student1 tables using left, right, inner and full join.**

Left Join

Source Code

```
SELECT Student.Name, Student1.Name
FROM Student
LEFT JOIN Student1
ON Student.RollNo = Student1.R_No;
```

Output

Name	Name
Ananya	Shivani
Srishty	Shruti
Shivpriya	Priyanka
Kartik	Samiksha
Vritti	Ritika

Inner Join

Source Code

```
SELECT Student.Name, Student1.Name
FROM Student
INNER JOIN Student1
ON Student.RollNo = Student1.R_No;
```

Output

Name	Name
Ananya	Shivani
Srishty	Shruti
Shivpriya	Priyanka
Kartik	Samiksha
Vritti	Ritika

PL/SQL Programs:-

16. Write a PL/SQL block to calculate total salary of employee having employee number 100.

Source Code

```
CREATE TABLE Employee
(
    EmployeeID Number PRIMARY KEY,
    FirstName varChar2(50),
    LastName varChar2(50),
    Basic Number,
    HRA Number,
    DA Number,
    PF Number
);
Insert Into Employee(EmployeeID,FirstName,LastName,Basic,HRA,DA,PF) Values(100,'Pankaj','Jain',40000,12000,5000,1250);
Insert Into Employee(EmployeeID,FirstName,LastName,Basic,HRA,DA,PF) Values(101,'Sumit','Pathak',50000,14000,7000,1250);
Insert Into Employee(EmployeeID,FirstName,LastName,Basic,HRA,DA,PF) Values(102,'Prateek','Sharma',35000,11000,4000,1250);
```

```
Declare
    EmpID Number :=100;
    Basic Number;
    HRA Number;
    DA Number;
    PF Number;
    Salary Number;
Begin

    SELECT Basic INTO Basic FROM Employee WHERE EmployeeID = EmpID;
    SELECT HRA INTO HRA FROM Employee WHERE EmployeeID = EmpID;
    SELECT DA INTO DA FROM Employee WHERE EmployeeID = EmpID;
    SELECT PF INTO PF FROM Employee WHERE EmployeeID = EmpID;
    Salary := (Basic + HRA + DA) - PF;
    dbms_output.Put_line('Total Salary = ' || Salary);
End;
```

Output

```
Output:

Total Salary = 55750
```

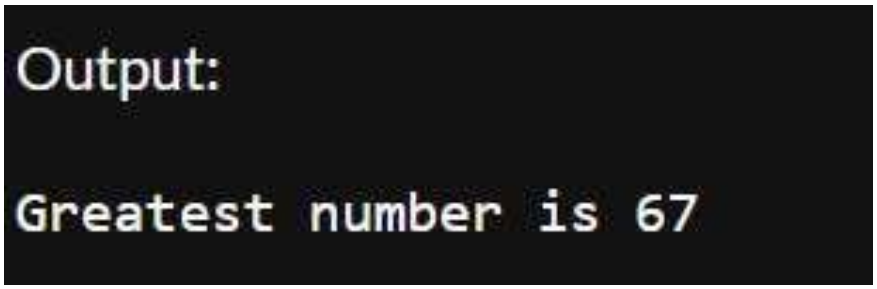
17. Write a PL/SQL code to find the greatest of three numbers.

Source Code

```
Declare
    A NUMBER := 46;
    B NUMBER := 67;
    C NUMBER := 21;

Begin
      If A > B And A > C Then
        dbms_output.Put_line('Greatest number is ' || A);
      ElsIf B > A And B > C Then
          dbms_output.Put_line('Greatest number is ' || B);
      ELsIf C > B And C > A Then
          dbms_output.Put_line('Greatest number is ' || C);
    End If;
End;
```

Output



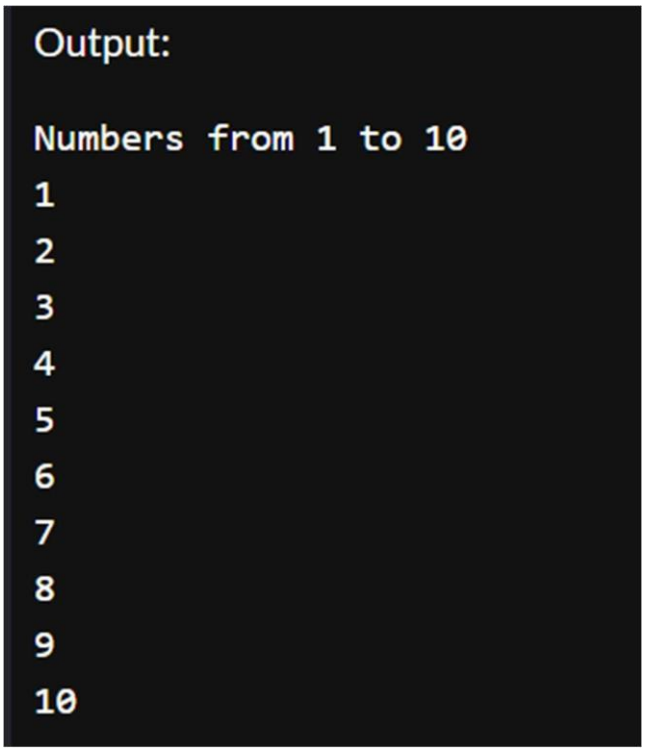
18. Write a PL/SQL code to print the numbers from 1 to n.

Source Code

```
Declare
    N Number := 10;
    CTR Number := 1;

Begin
      dbms_output.Put_line('Numbers from ' || Ctr || ' to ' || N);
      While CTR <= N
      Loop
        dbms_output.Put_line(CTR);
        CTR := CTR + 1;
      End Loop;
End;
```

Output



19. Write a PL/SQL code to reverse a string using for loop.

Source Code

```
Declare
    String Varchar(20) := 'Manav Rachna';
    L Number;
  ReverseString Varchar(20);

Begin
      L := Length(String);
      For I IN REVERSE 1.. L
      Loop
      ReverseString := ReverseString || Substr(String, I, 1);
    End Loop;
      dbms_output.Put_line('Reverse of string is ' || ReverseString);
End;
```

Output

```
Output:

Reverse of string is anhcaR vanaM
```

20. Write a PL/SQL code to find the sum of n numbers.

Source Code

```
Declare
    N Number := 10;
    CTR Number := 1;
    Sum Number :=0;
Begin

      While CTR <= N
    Loop
        Sum := Sum + CTR;
        CTR := CTR + 1;
    End Loop;
    dbms_output.Put_line('Sum of ' || N || ' Numbers = ' || Sum);
End;
```

Output

```
Output:

Sum of 10 Numbers = 55
```

21. Consider a PL/SQL code to display the empno, ename, job of employees of department number 10.

Source Code

```
CREATE TABLE Employee
(
    EmployeeNumber Number PRIMARY KEY,
    EmployeeName varChar2(50),
    Job varChar2(50),
    DepartmentNo Number

);

Insert Into Employee(EmployeeNumber,EmployeeName,Job,DepartmentNo) Values(100,'Pankj Jain','Data Analyst',5);
Insert Into Employee(EmployeeNumber,EmployeeName,Job,DepartmentNo) Values(101,'Sumit Pathak','Software Engineer',10);
Insert Into Employee(EmployeeNumber,EmployeeName,Job,DepartmentNo) Values(102,'Prateek Sharma','Librarian',15);

Declare
    DepartmentNo Number :=10;
    EmployeeNumber Number;
    EmployeeName varChar2(50);
    Job varChar2(50);
Begin

    SELECT EmployeeNumber INTO EmployeeNumber FROM Employee WHERE DepartmentNo = DepartmentNo;
    SELECT EmployeeName INTO EmployeeName FROM Employee WHERE DepartmentNo = DepartmentNo;
    SELECT Job INTO Job FROM Employee WHERE DepartmentNo = DepartmentNo;
    dbms_output.Put_line('Employee Number : ' || EmployeeNumber);
    dbms_output.Put_line('Employee Name : ' || EmployeeName);
    dbms_output.Put_line('Job : ' || Job);
End;
```

Output

```
Output:

Employee Number : 101
Employee Name : Sumit Pathak
Job : Software Engineer
```

22. Consider a PL/SQL code to display the employee number & name of top five highest paid employees.

Source Code

```
CREATE TABLE Employee
(
    EmployeeNumber Number PRIMARY KEY,
    EmployeeName varChar2(50),
    Salary Number
);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(100,'Pankj Jain',20000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(101,'Sumit Pathak',30000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(102,'Prateek Sharma',15000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(103,'Arun Sharma',45000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(104,'Raushan Kumar',30000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(105,'Prateek Sharma',48000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(106,'Prince Joseph',65000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(107,'Manish Pal',72000);
Insert Into Employee(EmployeeNumber,EmployeeName,Salary) Values(108,'Nitin Dutt',62000);
```

```
Declare
    EmployeeNumber Number;
    EmployeeName varChar2(50);
    Salary Number;
    CTR Number :=0;
    CURSOR C1 is SELECT EmployeeNumber,EmployeeName,Salary From Employee Order By Salary Desc;
Begin
     Open C1;
       Loop
          FETCH C1 into EmployeeNumber,EmployeeName,Salary;
          dbms_output.Put_line('Employee Number : ' || EmployeeNumber || ',Employee Name : ' || EmployeeName || ',Salary : ' || Salary);
          CTR := CTR + 1;
          EXIT WHEN CTR = 5;
       End Loop;
End;
```

Output

```
Employee Number : 107,Employee Name : Manish Pal,Salary : 72000
Employee Number : 106,Employee Name : Prince Joseph,Salary : 65000
Employee Number : 108,Employee Name : Nitin Dutt,Salary : 62000
Employee Number : 105,Employee Name : Prateek Sharma,Salary : 48000
Employee Number : 103,Employee Name : Arun Sharma,Salary : 45000
```

23. Consider a PL/SQL procedure that accepts 2 numbers & return addition, subtraction, multiplication & division of two numbers using stored procedure AND local procedure.

Source Code

```
Create Or Replace Procedure Addition(N1 In number, N2 In number,Result Out number)
Is
Begin
  Result :=N1+N2;

End;
/

Create Or Replace Procedure Subtraction(N1 in number, N2 in number,Result Out number)
Is
Begin
  Result :=N1-N2;

End;
/

Create Or Replace Procedure Multiplication(N1 in number, N2 in number,Result Out number)
Is
Begin
  Result :=N1 * N2;

End;
/
```

```
Create Or Replace Procedure Divison(N1 in number, N2 in number,Result Out number)
Is
Begin
  Result :=N1 / N2;

End;
/
DECLARE
   A number := 20;
   B number := 5;
   C number(2);
   D number;
   E number;
   F number;
BEGIN

   Addition(A,B,C);
   Subtraction(A,B,D);
   Multiplication(A,B,E);
   Divison(A,B,F);
   dbms_output.put_line('Addition of ' || A || ' And ' || B || ' is : ' || C);
   dbms_output.put_line('Subtraction of ' || A || ' And ' || B || ' is : ' || D);
   dbms_output.put_line('Multiplication of ' || A || ' And ' || B || ' is : ' || E);
   dbms_output.put_line('Divison of ' || A || ' And ' || B || ' is : ' || F );
END;
/
```

Output

```
Output:

Addition of 20 And 5 is : 25
Subtraction of 20 And 5 is : 15
Multiplication of 20 And 5 is : 100
Divison of 20 And 5 is : 4
```

24. Consider a PL/SQL code that accepts 2 numbers & return addition, subtraction, multiplication & division of two numbers using stored functions and local function.

Source Code

```
Create Or Replace Function Addition(N1 in number, N2 in number) Return Number
Is
Result number(8);
Begin
  Result :=N1+N2;
  Return Result;
End;
/

Create Or Replace Function Subtraction(N1 in number, N2 in number) Return Number
Is
Result number(8);
Begin
  Result :=N1-N2;
  Return Result;
End;
/

Create Or Replace Function Multiplication(N1 in number, N2 in number) Return Number
Is
Result number(8);
Begin
  Result :=N1 * N2;
  Return Result;
End;
/
```

```
Create Or Replace Function Divison(N1 in number, N2 in number) Return Number
Is
Result number(8);
Begin
  Result :=N1 / N2;
  Return Result;
End;
/
DECLARE
   A number := 20;
   B number := 5;
   C number(2);
   D number;
   E number;
   F number;
BEGIN
   C := Addition(A,B);
   D := Subtraction(A,B);
   E := Multiplication(A,B);
   F := Divison(A,B);
   dbms_output.put_line('Addition of ' || A || ' And ' || B || ' is : ' || C);
   dbms_output.put_line('Subtraction of ' || A || ' And ' || B || ' is : ' || D);
   dbms_output.put_line('Multiplication of ' || A || ' And ' || B || ' is : ' || E);
   dbms_output.put_line('Divison of ' || A || ' And ' || B || ' is : ' || F);
END;
/
```

Output

```
Output:

Addition of 20 And 5 is : 25
Subtraction of 20 And 5 is : 15
Multiplication of 20 And 5 is : 100
Divison of 20 And 5 is : 4
```

25. Write a PL/SQL block to show the use of NO_DATA FOUND exception.

Source Code

```
Create Table Employees
(
  ID Number,
  EmployeeNumber Number,
  FirstName Varchar2(50),
  LastName Varchar2(50),
  Basic Number
);

Insert Into Employees(ID,EmployeeNumber,FirstName,LastName,Basic) Values(1,1234,'Sumit','Pathak',40000);
Insert Into Employees(ID,EmployeeNumber,FirstName,LastName,Basic) Values(2,1235,'Pankaj','Sinha',30000);
Insert Into Employees(ID,EmployeeNumber,FirstName,LastName,Basic) Values(3,1236,'Rashmi','Jain',60000);
Insert Into Employees(ID,EmployeeNumber,FirstName,LastName,Basic) Values(4,1237,'Prateek','Sharma',35000);
Insert Into Employees(ID,EmployeeNumber,FirstName,LastName,Basic) Values(5,1238,'Ranjan','Tiwari',49000);

DECLARE
    EmpName   Varchar2(50);
    EmpID     NUMBER := 225;
BEGIN
   BEGIN
      SELECT FirstName INTO EmpName FROM employees WHERE EmployeeNumber = EmpID;
      DBMS_OUTPUT.PUT_LINE('Employee Name: ' || EmpName);
   EXCEPTION
      WHEN NO_DATA_FOUND THEN
         DBMS_OUTPUT.PUT_LINE('Error: No matching record found');
   END;
END;
```

Output

```
Output:

Error: No matching record found
```

26. Write a PL/SQL block to show the use of TOO_MANY ROWS exception.

Source Code

```
 Table Employees
(
  ID Number,
  DepartmentID Number,
  EmployeeNumber Number,
  FirstName Varchar2(50),
  LastName Varchar2(50),
  Basic Number
);

Insert Into Employees(ID,DepartmentID,EmployeeNumber,FirstName,LastName,Basic) Values(1,10,1234,'Sumit','Pathak',40000);
Insert Into Employees(ID,DepartmentID,EmployeeNumber,FirstName,LastName,Basic) Values(2,11,1235,'Pankaj','Sinha',30000);
Insert Into Employees(ID,DepartmentID,EmployeeNumber,FirstName,LastName,Basic) Values(3,12,1236,'Rashmi','Jain',60000);
Insert Into Employees(ID,DepartmentID,EmployeeNumber,FirstName,LastName,Basic) Values(4,13,1237,'Prateek','Sharma',35000);
Insert Into Employees(ID,DepartmentID,EmployeeNumber,FirstName,LastName,Basic) Values(5,10,1238,'Ranjan','Tiwari',49000);
```

```
DECLARE
    EmpName   Varchar2(50);
    DepID     NUMBER := 10;
BEGIN
   BEGIN
      SELECT FirstName INTO EmpName FROM Employees WHERE DepartmentID = DepID;
      DBMS_OUTPUT.PUT_LINE('Employee Name: ' || EmpName);
   EXCEPTION
      WHEN TOO_MANY_ROWS THEN
         DBMS_OUTPUT.PUT_LINE('Multiple rows found for the given department ID: ' || DepID);
   END;
END;
```

Output

```
Output:

Multiple rows found for the given department ID: 10
```

27. Write a PL/SQL block to show the use of ZERO_DIVIDE exception.

Source Code

```
Declare
  Numerator Number := 10;
  Denominator Number := 0;
  Result Number;

Begin
  Result:= numerator / denominator;
  DBMS_OUTPUT. PUT_LINE('Result: ' || result);
  EXCEPTION WHEN ZERO_DIVIDE THEN
  DBMS_OUTPUT.PUT_LINE('Error: Division by zero');
End;
```

Output

```
Output:

Error: Division by zero
```

28. To create a trigger on the emp table, which store the empno & operation in the table auditor for each operation i.e. Insert, Update & Delete.

Source Code

```
Create Table Emp
(
   ID Number,
   Empno Number,
   FirstName Varchar2(50),
   LastName Varchar2(50),
   Basic Number
);
```

```
Create Table Auditor
(
    Empno Number,
    Operation Varchar2(50)
);
CREATE TRIGGER Emp_After_Insert
After Insert ON Emp
FOR EACH ROW
BEGIN
   INSERT INTO Auditor(Empno,Operation)Values(new.Empno,'Insert');
END;
```

```
CREATE TRIGGER Emp_Before_Update
Before Update ON Emp
FOR EACH ROW
BEGIN
   INSERT INTO Auditor(Empno,Operation)Values(new.Empno,'Update');
END;
```

```
CREATE TRIGGER Emp_Before_Delete
Before Update ON Emp
FOR EACH ROW
BEGIN
   INSERT INTO Auditor(Empno,Operation)Values(old.Empno,'Delete');
END;
```

```
Insert Into Emp(ID,Empno,FirstName,LastName,Basic) Values(1,10,'Sumit','Pathak',40000);
Insert Into Emp(ID,Empno,FirstName,LastName,Basic) Values(2,11,'Pankaj','Sinha',30000);
Select ID,Empno,FirstName,LastName,Basic From Emp;
Select Empno,Operation From Auditor;
Update Emp Set FirstName='Ankit' Where ID=1;
Select ID,Empno,FirstName,LastName,Basic From Emp;
Select Empno,Operation From Auditor;
Delete From Emp Where ID=2;
Select ID,Empno,FirstName,LastName,Basic From Emp;
Select Empno,Operation From Auditor;
```

Output

Output

ID	Empno	FirstName	LastName	Basic
1	10	Sumit	Pathak	40000
2	11	Pankaj	Sinha	30000

Empno	Operation
10	Insert
11	Insert

ID	Empno	FirstName	LastName	Basic
1	10	Ankit	Pathak	40000
2	11	Pankaj	Sinha	30000

Empno	Operation
10	Insert
11	Insert
10	Delete
10	Update

ID	Empno	FirstName	LastName	Basic
1	10	Ankit	Pathak	40000

Empno	Operation
10	Insert
11	Insert
10	Delete
10	Update

29. To create a trigger so that no operation can be performed on emp table.

Source Code

```
Create Table Emp
(
   ID Number,
   Empno Number,
   FirstName Varchar2(50),
   LastName Varchar2(50),
   Basic Number
);
```

```
CREATE TRIGGER Prevent_Update_Trigger
Before UPDATE ON Emp
FOR EACH ROW
BEGIN
    IF  EXISTS(SELECT NULL FROM Emp E JOIN Deleted D ON E.Id=D.Id)
    BEGIN
        RAISERROR('You can not update Emp Table', 16, 1)
        ROLLBACK TRAN
        RETURN
    END
END;
```

```
Insert Into Emp(ID,Empno,FirstName,LastName,Basic) Values(1,10,'Sumit','Pathak',40000);
Insert Into Emp(ID,Empno,FirstName,LastName,Basic) Values(2,11,'Pankaj','Sinha',30000);
Update Emp Set Basic=5000
```

Output

```
Msg 50000, Level 16, State 1, Procedure Prevent_Update_Trigger, Line 7
You can not update Emp Table
Msg 3609, Level 16, State 1, Line 1
The transaction ended in the trigger. The batch has been aborted.
```