```
create or replace function get_distinctsalary
return number
as
  v_ret number := 0;
begin
  dbms_output.put_line('Name' || chr(9) ||
'Salary');
  for i in (
    select name, salary
    from (
      select name, salary, row_number() over
(partition by salary order by name) as rnk
      from Employee
    ) where rnk = 1
  ) loop
    dbms_output.put_line(i.name || chr(9) ||
i.salary);
  end loop;
  return v_ret;
end;
/
```

PART - B

1. Display Publisher Information

Ex. 1.

Create a library table with attributes book id, author_name, publisher, price and edition. Write PL/SQL code block to accept the publisher's name and count number of books under that publisher and display it. Also display the publisher with maximum publication.

```
create table Library
(
Book_Id number(6) not null,
Author_Name varchar2(100) not null,
Publisher varchar2(50) not null,
Price number(6,2),
edition varchar2(10)
);
```

```
insert into Library values (81,'Raman', 'ABC Publ', 350, 1);
insert into Library values(72, 'Raghav','XYZ Publ',900,1);
insert into Library values(62,'Raghav','PQR Publ',850,2);
insert into Library values(94,'Raman','LMN Publ',750,3);
insert into Library values(62,'Raghav','JKL Publ',850,2);
insert into Library values(94,'Hari','LMN Publ',1000,2);
```

```
DECLARE
  v_book_cnt   NUMBER(6);
  v_pub_name   VARCHAR2(20);
  Enter_Publisher VARCHAR2(50);
BEGIN

 Enter_Publisher :=:Enter_Publisher;
  --Query to display the Total Books for a given publisher
  SELECT COUNT(book_id) INTO v_book_cnt FROM Library WHERE Publisher = :Enter_Publisher;
  DBMS_OUTPUT.PUT_LINE('Number of books under ' || :Enter_Publisher || ' are: ' || v_book_cnt);
  --Query to display the Publisher with maximum publications
  WITH q1 AS (
    SELECT COUNT(book_id) cnt, Publisher FROM Library GROUP BY Publisher
  )
  SELECT cnt, Publisher INTO v_book_cnt, v_pub_name FROM q1 WHERE cnt = (SELECT MAX(cnt) FROM q1);
  DBMS_OUTPUT.PUT_LINE('Maximum publications are under ' || v_pub_name || '. Number of publications: ' || v_book_cnt);
EXCEPTION
  WHEN OTHERS THEN
    RAISE;
END;
/
```

PART - B

2. Display Distinct Salaries.

Ex. 2.

Write a function to display employee name with distinct salaries.

```
---------------------------------
create table Employee
(
name varchar(50) not null,
salary number(9,2);
);

insert into Employee values('Ram', 10000);
 insert into Employee values('Tom',20000);
 insert into Employee values('Ali', 10000);
 insert into Employee values('Giri',30000);
-------------------------------------------------------------

create or replace function get_distinctsalary
return number
as
  v_ret number := 0;
begin
  dbms_output.put_line('Name' || chr(9) || 'Salary');
  for i in (
    select name, salary
    from (
      select name, salary, row_number() over (partition by salary order by name) as rnk
      from Employee
    ) where rnk = 1
  ) loop
    dbms_output.put_line(i.name || chr(9) || i.salary);
  end loop;
  return v_ret;
end;
/

-- Main program to call function
declare
  a number(6);
begin
  a := get_distinctsalary();
end;
/
 ------------------------
```

Output:

PART - B

3. RANK function

Ex. 3.

Write a function to rank the employees based on their salary (use RANK function)

```
create table Employee
(
name varchar(50) not null,
salary number(9,2);
);

insert into Employee values('Ram', 10000);
 insert into Employee values('Tom',20000);
 insert into Employee values('Ali', 10000);
 insert into Employee values('Giri',30000);
-------------------------------------------------------------
CREATE OR REPLACE FUNCTION get_salarybyrank
RETURN NUMBER AS
 v_ret NUMBER := 0;
BEGIN
 DBMS_OUTPUT.PUT_LINE('Rank'||chr(9)||'Name'||chr(9)||'Salary');
 DBMS_OUTPUT.PUT_LINE('----------------------');
 FOR i IN (SELECT RANK() OVER (ORDER BY salary DESC) AS rnk,
                name,
                salary
          FROM Employee)
 LOOP
   DBMS_OUTPUT.PUT_LINE(i.rnk||chr(9)||i.name||chr(9)||i.salary);
 END LOOP;
 RETURN 1;
END;
/

-- Main program to call function
SET SERVEROUTPUT ON
DECLARE
   a number(6);
BEGIN
   a := get_salarybyRank();
END;
/
 ----------------------------
 Output:

Rank    Name   Salary
---------------------------------
1        Giri      30000
```

PART - B

4. Validate the email id.

Ex. 4.

Write a function to validate the Employee email id.

```
----------------------------------------
create or replace function validateemail(p_email_id IN VARCHAR2)
 return BOOLEAN is
      b_isvalid   BOOLEAN;
BEGIN
b_isvalid:=REGEXP_LIKE (p_email_id, '^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}$');
return b_isvalid;
 END;
 /
/* Main Program */
 Declare
 a boolean;
eid varchar2(100) := '&Enter_Email_ID';
begin
a:=validateemail(eid);
                        if(a=true) then
dbms_output.put_line('Valid Email ID');
else
dbms_output.put_line('Invalid Email ID');
                        end if;
end;
/
 ----------
```

Output:

example@email.com

PL/SQL procedure successfully completed.

Valid Email ID

« PREVIOUS ACTIVITY
PART - A

NEXT ACTIVITY
Introduction »

Jump to...

PART - B

5. Capture the error log

Ex.5.

Write a procedure to capture the error log in a table in case of an exception using autonomous transaction.

```
-- Create table for logging errors
CREATE TABLE logging_table (
 name VARCHAR2(20),
 error_message VARCHAR2(100)
);
-----------------------------------------------------

-- Create procedure to log errors
CREATE OR REPLACE PROCEDURE log_error(p_name VARCHAR2, p_error_message VARCHAR2) AS
 PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
 INSERT INTO logging_table (name, error_message) VALUES (p_name, p_error_message);
 COMMIT;
END;
/
-- Main program
DECLARE
 lv_val NUMBER;
BEGIN
 lv_val := 5/0;
EXCEPTION
 WHEN ZERO_DIVIDE THEN
  log_error('ZERO DIVIDE', SQLERRM);
END;
/

-----------------------------------------------

select * from logging_table

-----------------------------------------------
```

Output:

	NAME	ERROR_MESSAGE
1	ZERO DIVIDE	ORA-01476: divisor is equal to zero

« PREVIOUS ACTIVITY
PART - A

NEXT ACTIVITY
Introduction »

PART - B

6. Raise user defined exception

Ex.6.

Write an Anonymous block which raise a user defined exception on Thursday.

```
DECLARE
   expl EXCEPTION;
   weekday number;
BEGIN
            select to_char(sysdate, 'd') into weekday from dual;
           IF weekday=5 then
              raise expl ;
           END IF;
EXCEPTION
WHEN expl THEN
dbms_output.put_line('Exception Note: Today is Thursday');
END;
```

Output:

```
PL/SQL procedure successfully completed.

Exception Note: Today is Thursday
```

PART - B

7. A Cursor program

Ex. 7.

Write a PL/SQL cursor program which is used to calculate total salary from emp table without using sum() function

```
create table emp
(
eno number(4) primary key,
 ename varchar2(10),
 sal number(8,2)
);

insert into emp values(103,'suman',45000);
insert into emp values(105,'Neeta',56000);


Declare
v_ename varchar2(10);
v_eno varchar2(10);
v_sal number(8);
 x number(8,2):=0;

cursor cl is
select ename,eno,sal from emp;        --Declaring Cursor

BEGIN
open cl;  --Opening Cursor.
    loop
fetch cl into v_ename,v_eno, v_sal;--Fetching
exit when cl %notfound;
        dbms_output.put_line(v_ename||' '|| v_eno||' '||v_sal);
          x:=x+v_sal;
      end loop;
close cl; --Closing Cursor.
dbms_output.put_line('Total salary is '||x);
end;
/
```

Output:

```
suman  103   45000
Neeta   105   56000
Total salary is 101000
```