Assignment

Sept23/ DBT/126.1

Database Technologies

Diploma in Advance Computing

September 2023

**Procedure and Function**

|  |
| --- |
| 1. Write a procedure to accept a string and print all characters in separate lines.   Input: - Ram  Output: - R  a  m |
| drop procedure if exists String1;  drop table if exists charString;  delimiter $  create procedure String1(in strinput varchar(50))  begin  declare ch char(1);  declare i,strlength int;  set i := 1;  set ch := "";  set strlength := length(strinput);  create table charString( YourOutput char(1));  l1:loop  set ch := substring(strinput,i,1);  if i > strlength then  leave l1;  else  insert into charString values(ch);  set i := i + 1;  end if;  end loop l1;  select \* from charString;  drop table charString;  end $  delimiter ; |
|  |
| 1. Write a procedure to accept a string and print every character separated by a comm sign.   Input: - SALEEL  Output: - S, A, L, E, E, L |
| drop procedure if exists String1;  delimiter $  create procedure String1(in strinput varchar(50))  begin  declare i,strlength int;  set @string := '' ;  set i := 0;  set strlength := length(strinput);  l1:loop  set i := i + 1;  if ( i<strlength+1) then  if i=1 then  set @string := substring(strinput,i,1);  else  set @string := concat(@string,',',substring(strinput,i,1));  end if;  else  leave l1;  end if;  end loop l1;  select @string;  end $  delimiter ; |
|  |
| 1. Write a procedure to accept an alpha numeric string and separate number and characters of the string.   Input: - SAL1234EEL  Output: - SALEEL  1234 |
| drop procedure if exists String1;  delimiter $  create procedure String1(in strinput varchar(30))  begin  declare i,strlength int;  declare ch char(1);  set @string := '' ;  set @num := '' ;  set i := 0;  set strlength := length(strinput);  l1:loop  set ch := substring(strinput,i,1);  if i<=strlength then  if ch/1 then  set @num := concat(@num,'', ch);  else  set @string := concat(@string,'', ch);  end if;  else  leave l1;  end if;  set i := i + 1;  end loop l1;  select @string;  select @num;  end $  delimiter ; |
|  |
| 1. Write a procedure to print all employee name and his job in following format.   Input: - KING PRESIDENT  SCOTT ANALYST  Output: - K(ING) is PRESIDENT  S(COTT) is ANALYST |
| drop procedure if exists empJob;  delimiter $  create procedure empJob()  begin  select ename, job from employee ;  select concat(substr(ename, 1, 1), "(", substr(ename,2), ") is ", job) YourOutput from employee ;  end $  delimiter ; |
|  |
| 1. Write a procedure to print all upper and lower characters separately.   Input: - AbCdEfG  Output: - ACEG  bdf |
| drop procedure if exists upperLower;  delimiter $  create procedure upperLower(string1 varchar(20))  begin  declare i,length1 int;  declare ch char(1);  set i := 0;  set @Uppercase := '';  set @Lowercase := '';  set length1 := length(string1);  l1: loop  set ch := substring(string1,i,1);  if ascii(ch) >= 65 and ascii(ch) <= 97 then  set @Uppercase := concat(@Uppercase,ch);  else  set @Lowercase := concat(@Lowercase,ch);  end if;  if i>length1 then  leave l1;  end if;  set i := i+1;  end loop l1;  select @Uppercase;  select @Lowercase;  end $  delimiter ; |
|  |
| 1. Write a procedure to find the number of vowels, digits and white spaces |
| drop procedure if exists findNumber;  delimiter $  create procedure findNumber(string1 varchar(100))  begin  declare i,length1 int;  declare ch char(1);  set i := 0;  set @vowels := '';  set @digits := '';  set @whiteSpaces := '';  set length1 := length(string1);  l1: loop  set ch := substring(string1,i,1);  if ch='a' or ch='e' or ch='i' or ch='o' or ch='u' or ch='A' or ch='E' or ch='I' or ch='O' or ch='U' then  set @vowels := @vowels + 1 ;  end if;  if ch/1 then  set @digits := @digits + 1 ;  end if;  if ch = "\0" then  set @whiteSpaces := @whiteSpaces + 1 ;  end if;  set i := i+1;  if i>length1 then  leave l1;  end if;  end loop l1;  select @vowels;  select @digits;  select @whiteSpaces;  end $  delimiter ; |
|  |
| 1. Write a procedure to remove all characters in a string except alphabets   Input: - saleel.bagde123@gmail.com  Output: - saleelbagdegmailcom |
| drop procedure if exists findNumber;  delimiter $  create procedure findNumber(string1 varchar(100))  begin  declare i,length1 int;  declare ch char(1);  set i := 0;  set @string := '';  set length1 := length(string1);  l1: loop  set ch := substring(string1,i,1);  if (ascii(ch) >= 65 and ascii(ch) <= 90) or (ascii(ch) >= 97 and ascii(ch) <= 122) then  set @string := concat(@string,ch);  end if;  set i := i+1;  if i>length1 then  leave l1;  end if;  end loop l1;  select @string;  end $  delimiter ; |
|  |
| 1. Write a procedure to insert 10 rows in a table having following columns (using loop).   R (id int, message varchar(20)).  Output: -  id message  ---- -----------  1 i is odd  2 i is even  3 i is odd  4 i is even  5 i is odd  6 i is even  7 i is odd  8 i is even  9 i is odd  10 i is even |
| drop procedure if exists insertion;  drop table if exists insertionOddEven;  delimiter $  create table insertionOddEven(id int, message varchar(20));  create procedure insertion(in num int)  begin  if (num%2)=0 then  insert into insertionoddeven values(num, 'i is even');  else  insert into insertionoddeven values(num, 'i is odd');  end if;  end $  delimiter ; |
|  |
| 1. Write a procedure to print five highest paid employees from the emp table using cursor. |
| drop procedure if exists highestpaid;  delimiter $  create procedure highestpaid(x int)  begin  declare \_ename varchar(20);  declare \_sal int;  declare y int default 1;  declare c1 cursor for select ename,sal from emp order by sal desc limit x;  declare exit handler for 1329 select "EOF";  drop table if exists hp;  create table hp(name varchar(20), sal int);  open c1;  l1:loop  fetch from c1 into \_ename, \_sal;  insert into hp values(\_ename, \_sal);  set y := y + 1;  if y>x then  leave l1;  end if;  end loop l1;  close c1;  select name, sal from hp;  drop table if exists hp;  end $  delimiter ; |
|  |
| 1. Create the following table named (emp10, emp20, and emp30) which have the same structure of emp table.   Write a procedure to split employee records from emp table according to their department numbers and insert those records in the appropriate table using cursor. |
| drop procedure if exists empdept;  delimiter $  create procedure empdept()  begin  declare \_empno, \_deptno, \_sal , minemp, maxemp int default 0;  declare \_name, \_job varchar(20);  declare c1 cursor for select empno, deptno, Ename, job, sal from emp;  declare exit handler for 1329 select "EOF";  drop table if exists emp10;  drop table if exists emp20;  drop table if exists emp30;  create table emp10(Empno int, Deptno int, EName varchar(20), Job varchar(20), Sal int);  create table emp20(Empno int, Deptno int, EName varchar(20), Job varchar(20), Sal int);  create table emp30(Empno int, Deptno int, EName varchar(20), Job varchar(20), Sal int);  select min(empno) into minemp from emp ;  select max(empno) into maxemp from emp ;  open c1;  l1: loop  fetch c1 into \_empno, \_deptno, \_name, \_job, \_sal;  if \_deptno = 10 then  insert into emp10 values(\_empno, \_deptno, \_name, \_job, \_sal);  end if;  if \_deptno = 20 then  insert into emp20 values(\_empno, \_deptno, \_name, \_job, \_sal);  end if;  if \_deptno = 30 then  insert into emp30 values(\_empno, \_deptno, \_name, \_job, \_sal);  end if;  if minemp = maxemp then  leave l1;  end if;  set minemp := minemp + 1;  end loop l1;  close c1;  select empno, deptno, EName, job, sal from emp10;  select empno, deptno, EName, job, sal from emp20;  select empno, deptno, EName, job, sal from emp30;  end $  delimiter ; |
|  |
| 1. Write a procedure to display the department number and employee name in the following format.   Output: -  10 -> (AARAV, THOMAS, CLARK, KING, MILLER)  20 -> (SHARMIN, BANDISH, SMITH, JONES, SCOTT, FRED, ADAMS, FORD)  30 -> (GITA, ALLEN, WARD, MARTIN, BLAKE, TURNER, JAMES, HOFFMAN, GRASS)  40 –> (No employee work in department 40…)  50 -> (VRUSHALI, SANGITA, SUPRIYA) |
| drop procedure if exists details;  delimiter $  create procedure details()  begin  declare d int default 0;  declare gname varchar(1000);  drop table if exists ename;  create table ename(names varchar(1000));  set @x :='';  l1:loop  set d := d + 10;  if d then  if d=40 then  set gname := concat(d,'->','(No employee work in department 40…)');  insert into ename(names) values(gname);  set gname :='';  set @x :='';  else  select group\_concat(ename) into @x from emp group by deptno having deptno=d;  set gname := concat(d,'->',@x);  insert into ename(names) values(gname);  set gname :='';  set @x :='';  end if;  end if;  if d = 50 then  leave l1;  end if;  end loop l1;  select names from ename;  end $  delimiter ; |
|  |
| 1. Write a procedure to accept customer number and display all his order. (Use customers and orders table) |
| drop procedure if exists displayOrders;  delimiter $  create procedure displayOrders(in \_cnum int)  begin  declare check1 bool;  select true into check1 from customers where cnum = \_cnum ;  if check1 then  select cname CustomerName, ODATE, type, ONUM, AMT Amount from customers c join orders o where c.cnum = \_cnum and c.cnum = o.cnum order by odate;  end if;  end $  delimiter ; |
|  |
| 1. Write a procedure to convert numbers into word   Input: - 45234  Output: - Four Five Two Three Four |
| drop procedure if exists convertNum;  delimiter $  create procedure convertNum(in num int)  begin  declare n,i int;  declare ch char(1);  set i := 0;  set n := length(num);  set @yourOutput := '';  l1: loop  set ch := substring(num, i, 1);  if ch = '0' then  set @yourOutput := concat(@yourOutput, 'Zero ');  end if;  if ch = '1' then  set @yourOutput := concat(@yourOutput, 'One ');  end if;  if ch = '2' then  set @yourOutput := concat(@yourOutput, 'Two ');  end if;  if ch = '3' then  set @yourOutput := concat(@yourOutput, 'Three ');  end if;  if ch = '4' then  set @yourOutput := concat(@yourOutput, 'Four ');  end if;  if ch = '5' then  set @yourOutput := concat(@yourOutput, 'Five ');  end if;  if ch = '6' then  set @yourOutput := concat(@yourOutput, 'Six ');  end if;  if ch = '7' then  set @yourOutput := concat(@yourOutput, 'Seven ');  end if;  if ch = '8' then  set @yourOutput := concat(@yourOutput, 'Eight ');  end if;  if ch = '9' then  set @yourOutput := concat(@yourOutput, 'Nine ');  end if;  set i := i + 1;  if i> n then  leave l1;  end if;  end loop l1;  select @yourOutput;  end $  delimiter ; |
|  |
| 1. Write a procedure to find the sum of digits.   Input: - 5675  Output: - Twenty Three |
| drop function if exists oneTen;  delimiter $  create function oneTen(numsum int) returns varchar(100)  deterministic  begin  declare yourOutput1 varchar(200) default '';    l1: loop  if (numsum mod 10 = 1) then  set yourOutput1 := concat(yourOutput1, 'One ');  end if;  if (numsum mod 10 = 2) then  set yourOutput1 := concat(yourOutput1, 'Two ');  end if;  if (numsum mod 10 = 3) then  set yourOutput1 := concat(yourOutput1, 'Three ');  end if;  if (numsum mod 10 = 4) then  set yourOutput1 := concat(yourOutput1, 'Four ');  end if;  if (numsum mod 10 = 5) then  set yourOutput1 := concat(yourOutput1, 'Five ');  end if;  if (numsum mod 10 = 6) then  set yourOutput1 := concat(yourOutput1, 'Six ');  end if;  if (numsum mod 10 = 7) then  set yourOutput1 := concat(yourOutput1, 'Seven ');  end if;  if (numsum mod 10 = 8) then  set yourOutput1 := concat(yourOutput1, 'Eight ');  end if;  if (numsum mod 10 = 9) then  set yourOutput1 := concat(yourOutput1, 'Nine ');  end if;  leave l1;  end loop l1;  return yourOutput1;  end $  delimiter ;  drop procedure if exists convertNum;  delimiter $  create procedure convertNum(in num int)  begin  declare numlength, numsum, digit, i int;  declare ch varchar(10) default '';  set i:= 0;  set numsum := 0;  set @yourOutput := '';  set numlength := length(num);  l1: loop  set i:= i + 1;  if i>numlength then  leave l1;  else  set digit := num mod 10;  set numsum := numsum + digit;  set num := num div 10;    end if;  end loop l1;  select numsum;    if numsum = 0 then  set @yourOutput := concat(@yourOutput, 'Zero ');  end if;  if (numsum > 0) and (numsum <10) then  set ch := oneTen(numsum);  set @yourOutput := concat(@yourOutput,ch );  set ch := '';  end if;  if numsum = 10 then  set @yourOutput := concat(@yourOutput, 'Ten ');  end if;  if numsum = 11 then  set @yourOutput := concat(@yourOutput, 'Eleven ');  end if;  if numsum = 12 then  set @yourOutput := concat(@yourOutput, 'Twelve ');  end if;  if numsum = 13 then  set @yourOutput := concat(@yourOutput, 'Thirteen ');  end if;  if numsum = 14 then  set @yourOutput := concat(@yourOutput, 'Fourteen ');  end if;  if numsum = 15 then  set @yourOutput := concat(@yourOutput, 'Fifteen ');  end if;  if numsum = 16 then  set @yourOutput := concat(@yourOutput, 'Sixteen ');  end if;  if numsum = 17 then  set @yourOutput := concat(@yourOutput, 'Seventeen ');  end if;  if numsum = 18 then  set @yourOutput := concat(@yourOutput, 'Eighteen ');  end if;  if numsum = 19 then  set @yourOutput := concat(@yourOutput, 'Nineteen ');  end if;  if (numsum div 10 = 2) then  set @yourOutput := concat(@yourOutput, 'Twenty ');  if numsum mod 10 then  set ch := oneTen(numsum);  set @yourOutput := concat(@yourOutput,ch );  set ch := '';  end if;  end if;  if (numsum div 10 = 3) then  set @yourOutput := concat(@yourOutput, 'Thirty ');  if numsum mod 10 then  set ch := oneTen(numsum);  set @yourOutput := concat(@yourOutput,ch );  set ch := '';  end if;  end if;  select @yourOutput;  end $  delimiter ; |
|  |
| 1. Write a procedure to find how many “Sundays” are present between two given dates.   Input: - Date1 and Date2  Output: - 3 Sunday’s |
| drop procedure if exists Sunday;  delimiter $  create procedure Sunday(in date1 date, in date2 date)  begin  declare scount int default 0 ;  l1: loop  if dayname(date1) = 'Sunday' then  set scount := scount + 1;  end if;  if date1 = date2 then  leave l1;  end if;  set date1 := date\_add(date1, interval + 1 day);  end loop l1;  select concat(scount,' ',"Sunday's") ;  end $  delimiter ; |
|  |
| 1. Writer a procedure which will accept date and weekday name from the user and print upcoming date on than weekday   Input: - (‘2023-04-26’, ‘Saturday’)  Output: - ‘2023-04-29’ |
| drop procedure if exists upcomingDate;  delimiter $  create procedure upcomingDate(in date1 date, in day1 char(10))  begin  declare date2 date ;  l1: loop  set date1 := date\_add(date1, interval + 1 day);  if dayname(date1) = day1 then  set date2 := date1 ;  leave l1;  end if;  end loop l1;  select date1 ;  end $  delimiter ; |
|  |