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 strfunc;  delimiter @  create procedure strfunc(\_str varchar(100))  BEGIN  declare count1 int;  declare x int;  set x:=0;  set count1:=length(\_str);  l1:loop  set x:=x+1;  set count1:=count1-1;  if count1>=0 then  select substr(\_str,x,1);  else  leave l1;  end if;  end loop l1;  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 strfunc;  delimiter $  create procedure strfunc(\_str varchar(100))  BEGIN  declare count1 int;  declare x int;    declare temp varchar(200);  set x:=0;  set count1:=length(\_str);  set @ch= " ";  l1:loop  set x:=x+1;  set count1:=count1-1;  if count1>=0 then  set temp:= substr(\_str,x,1);  set @ch:=concat(@ch,temp,",");  else  leave l1;  end if;  end loop l1;  select @ch as name;  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 strfunc;  delimiter #  create procedure strfunc(\_str varchar(100))  BEGIN  declare cnt int;  set cnt := 0;  set @ans:=" ";  set @x=" ";  set @y=" ";  l1:loop  set cnt:=cnt+1;  set @ans := substr(\_str,cnt,1);  if (@ans between "A" and "Z") then  set @x:=concat(@x,@ans);  end if;  if (@ans between '0' and '9') then  set @y:=concat(@y,@ans);  end if;  if cnt>length(\_str)-1 then  leave l1;  end if;  end loop l1;  select \_str,@x,@y;  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 pro1 ;  delimiter $  CREATE PROCEDURE pro1(name varchar(200))  BEGIN  DECLARE a INT;  set a=0;    set @x=" ";  set @y=" ";  set @z=" ";    lb:LOOP  set a =a+1;  set @x = substring(name,1,1);  set @y = substring(name,2,4);  set @z = substring(name,5);  set @p := concat(@x,'(',@y,')',' is ',@z);  if a > LENGTH(name)-1 THEN  leave lb;  end if;    end LOOP lb;  SELECT @p;  END $  delimiter ; |
|  |
| 1. Write a procedure to print all upper and lower characters separately.   Input: - AbCdEfG  Output: - ACEG  bdf |
| DROP PROCEDURE if EXISTS pro1 ;  delimiter $  CREATE PROCEDURE pro1(name varchar(200))  BEGIN  DECLARE a INT;  set a=0;  set @x:=" ";  set @y:=" ";  set @z:=" ";  lb:LOOP  set a :=a+1;  set @x := substring(name,a,1);  if(ascii(@x) BETWEEN 65 AND 90) THEN  set @y:=concat(@y,@x);  end if;  if(ascii(@x) BETWEEN 97 AND 122) THEN  set @z:=concat(@z,@x);  end if;  if a > LENGTH(name)-1 THEN  leave lb;  end if;      end LOOP lb;  SELECT @y As Uppercase;  SELECT @z As Lowercase;  END $  delimiter ; |
|  |
| 1. Write a procedure to find the number of vowels, digits and white spaces |
| drop procedure if exists que6;  delimiter $  create procedure que6(str1 varchar(100))  BEGIN  DECLARE x int;  set x:=0;  set @y:=" ";  set @cout:=0;  set @dig=0;  set @white=0;    lbl:LOOP  set x:=x+1;  set @y:=substring(str1,x,1);    if(@y like 'a' or @y like 'e' or @y like 'i' or @y like 'o' or @y like 'u') THEN  set @cout=@cout+1;  end if;    if(@y BETWEEN '0' AND '9') THEN  set @dig=@dig+1;  end if;    if(@y like ' ' ) THEN  set @white=@white+1;  end if;  if x >= length(str1) THEN  leave lbl;  end if;      end LOOP lbl;  SELECT @cout as "Vowel Count",@dig as "Digit Count", @white as "Whitespace Count";  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 pro1 ;  delimiter $  CREATE PROCEDURE pro1(name varchar(200))  BEGIN  DECLARE a INT;  set a=0;  set @x=" ";  set @y=" ";  lb:LOOP  set a =a+1;  set @x = substring(name,a,1);  if(@x BETWEEN 'a' AND 'z') THEN  set @y=concat(@y,@x);  SELECT @y;    if a > LENGTH(name)-1 THEN  leave lb;  end if;  end if;  end LOOP lb;  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 que8;  delimiter $  create procedure que8()  BEGIN  SET @x :=0;  l1:LOOP  set @x:=@x+1;  if (@x%2=0) THEN  insert into R100 values(@x,'i is even');  else  insert into R100 values(@x,'i is odd');  end IF;  if (@x>=10) THEN  leave l1;  end if;  end loop l1;  select \* from R100;  end $  delimiter ; |
|  |
| 1. Write a procedure to print five highest paid employees from the emp table using cursor. |
| drop procedure if exists que9;  delimiter $  create procedure que9()  BEGIN  DECLARE \_ename varchar(1000);  DECLARE \_sal int;  DECLARE c1 cursor for  select ename, sal from emp order by sal desc limit 5;  open c1;  lbl: LOOP  FETCH c1 into \_ename,\_sal;  select \_ename,\_sal;  end LOOP lbl;  close c1;  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 que10;  delimiter $  create procedure que10()  BEGIN    DECLARE \_empno int;  DECLARE \_ename varchar(12);  DECLARE \_gender char(1);  DECLARE \_JOB varchar(20);  DECLARE \_MGR INT;  DECLARE \_HIREDATE date;  DECLARE \_sal,\_comm,\_deptno,\_BONUSID int;  DECLARE \_username varchar(20);  DECLARE \_PWD varchar(20);  DECLARE \_phone varchar(45);  DECLARE \_isActive tinyint(1);    DECLARE c1 cursor for select \* from emp;  DECLARE EXIT handler for 1329 select "End of File";  open c1;  lbl : LOOP    FETCH c1 into \_empno,\_ename,\_gender,\_JOB,\_MGR,\_HIREDATE,\_sal,\_comm,\_deptno,\_BONUSID,\_username,\_PWD,\_phone,\_isActive;  if \_deptno=10 THEN  INSERT into emp10 values(\_empno,\_ename,\_gender,\_JOB,\_MGR,\_HIREDATE,\_sal,\_comm,\_deptno,\_BONUSID,\_username,\_PWD,\_phone,\_isActive);  end if;  if \_deptno=20 THEN  INSERT into emp20 values(\_empno,\_ename,\_gender,\_JOB,\_MGR,\_HIREDATE,\_sal,\_comm,\_deptno,\_BONUSID,\_username,\_PWD,\_phone,\_isActive);  end if;    if \_deptno=30 THEN  INSERT into emp30 values(\_empno,\_ename,\_gender,\_JOB,\_MGR,\_HIREDATE,\_sal,\_comm,\_deptno,\_BONUSID,\_username,\_PWD,\_phone,\_isActive);  end if;    end LOOP lbl;  close c1;  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 que11;  delimiter $  create procedure que11()  BEGIN  DECLARE \_deptno int;  DECLARE \_ename varchar(1000);  DECLARE c1 cursor for select deptno,group\_concat(ename) as Ename from emp group by deptno;  open c1;  lbl : LOOP  FETCH c1 into \_deptno,\_ename;  select CONCAT(\_deptno," -> (",\_ename," )") as ANS;  end LOOP lbl;  close c1;    end $  delimiter ; |
|  |
| 1. Write a procedure to accept customer number and display all his order. (Use customers and orders table) |
| drop procedure if exists que13;  delimiter $  create procedure que13(\_custid int)  BEGIN  select \* from orders where cnum=\_custid;  end $  delimiter ; |
|  |
| 1. Write a procedure to convert numbers into word   Input: - 45234  Output: - Four Five Two Three Four |
| drop procedure if exists que13;  delimiter $  create procedure que13(num varchar(20))  BEGIN  DECLARE a INT;  set a:=0;  set @y:=" ";    lbl:LOOP  set a :=a+1;  set @x := substring(num,a,1);  if(@x=0) THEN  set @y:=concat(@y,'Zero ');  end if;    if(@x=1) THEN  set @y:=concat(@y,'One ');  end if;  if(@x=2) THEN  set @y:=concat(@y,'Two ');  end if;    if(@x=3) THEN  set @y:=concat(@y,'Three ');  end if;    if(@x=4) THEN  set @y:=concat(@y,'Four ');  end if;    if(@x=5) THEN  set @y:=concat(@y,'Five ');  end if;    if(@x=6) THEN  set @y:=concat(@y,'Six ');  end if;    if(@x=7) THEN  set @y:=concat(@y,'Seven ');  end if;  if(@x=8) THEN  set @y:=concat(@y,'Eight ');  end if;    if(@x=9) THEN  set @y:=concat(@y,'Nine ');  end if;    if a > LENGTH(num)-1 THEN  leave lbl;  end if;    end LOOP lbl;  select @y;  end $  delimiter ; |
|  |
| 1. Write a procedure to find the sum of digits.   Input: - 5675  Output: - Twenty Three |
| drop function if exists firstNumber;  delimiter $  create function firstNumber(x bigint) returns varchar(20)  deterministic  begin  if x = 1 then  return "one";  elseif x = 2 then  return "two";  elseif x = 3 then  return "three";  elseif x = 4 then  return "four";  elseif x = 5 then  return "five";  elseif x = 6 then  return "six";  elseif x = 7 then  return "seven";  elseif x = 8 then  return "eight";  elseif x = 9 then  return "nine";  elseif x = 0 then  return " ";  end if;  end $  delimiter ;  drop procedure if exists pro14;  delimiter $  create procedure pro14(number1 int)  begin  declare inWords varchar(50);  declare num int;  declare temp int ;  declare tensPlace int;  set inWords := " ";  set num := 0;  loop1:loop  if number1 > 0 then  set temp := number1 % 10;  set number1 := number1 div 10;  set num := num + temp;  else  leave loop1;  end if;  end loop;  set tensPlace := num div 10;  if tensPlace = 2 then  set inWords := concat(inWords,"twenty");  elseif tensPlace = 3 then  set inWords := concat(inWords,"thirty");  elseif tensPlace = 4 then  set inWords := concat(inWords,"forty");  elseif tensPlace = 5 then  set inWords := concat(inWords,"fifty");  elseif tensPlace = 6 then  set inWords := concat(inWords,"sixty");  elseif tensPlace = 7 then  set inWords := concat(inWords,"seventy");  elseif tensPlace = 8 then  set inWords := concat(inWords,"eighty");  elseif tensPlace = 9 then  set inWords := concat(inWords,"ninty");  else  set inWords := concat(inWords," ");  end if ;  if tensPlace = 1 then  if num = 10 then  set inWords := concat(inWords,"ten");  elseif num = 11 then  set inWords := concat(inWords,"eleven");  elseif num = 12 then  set inWords := concat(inWords,"twelve");  elseif num = 13 then  set inWords := concat(inWords,"thirteen");  elseif num = 14 then  set inWords := concat(inWords,"forteen");  elseif num = 15 then  set inWords := concat(inWords,"fifteen");  elseif num = 16 then  set inWords := concat(inWords,"sixteen");  elseif num = 17 then  set inWords := concat(inWords,"seventeen");  elseif num = 18 then  set inWords := concat(inWords,"eighteen");  elseif num = 19 then  set inWords := concat(inWords,"nineteen");  end if;  end if;  if tensplace != 1 then  set inWords := concat(inWords," ",firstNumber(num % 10));  end if;  select inWords;  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 pro15;  delimiter $  create procedure pro15(date1 date,date2 date)  begin  declare count int;  set count := 0;  lbl1:loop  if date1<date2 then  if date\_format(date1,'%W') = "Sunday" then  set count := count + 1;  set date1 := date1 + interval 1 day;  else  set date1 := date1 + interval 1 day;  end if;  else  leave lbl1;  end if;  end loop lbl1;  select count;  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 pro16;  delimiter $  create procedure pro16(date1 date, weekday1 varchar(10))  begin  lbl1:loop  if date\_format(date1,'%W')=weekday1 then  select date1;  leave lbl1;  else  set date1 := date1+interval 1 day;  end if;  end loop lbl1;  end $  delimiter ; |
|  |
|  |