­­Assignment

Sept23/ DBT/126

Database Technologies

Diploma in Advance Computing

September 2023

**Procedure**

|  |
| --- |
| 1. Create a LOGIN table (username, password, and email). Write a procedure (named ***addUser***) to pass the username, password, and email-ID through the procedure and store the data in the LOGIN table. |
| DROP PROCEDURE IF EXISTS addUser;  delimiter $  CREATE PROCEDURE addUser(user\_nm varchar(20),em VARCHAR(20), pass VARCHAR(15) )  BEGIN  INSERT INTO login VALUES(user\_nm,em,pass);  end $  delimiter ;  mysql> call addUser('ritesh','ritz@gmail.com','rrr'); |
|  |
| 1. Create a LOG table having following columns (id (auto\_increment), curr\_date, curr\_time, and message). Write a procedure (named ***checkUser***) to pass the email-ID as an input, check whether passed email-ID is available in LOGIN table or not available. If the email-ID is available then display the username and his password. If the email-ID is not available then, insert (curr\_date, curr\_time, and message) in LOG table. |
| drop PROCEDURE if EXISTS checkUser ;  delimiter $  create PROCEDURE checkUser (em varchar(30))  BEGIN  set @e := em;    SELECT email into @x from login where email=em;  IF (@x =em ) THEN  select username,pwd from login where email=em;  ELSE  select "New user found...";  set @i := "INSERT INTO log(cur\_date,cur\_time,message) VALUES(CURRENT\_DATE(),CURRENT\_TIME(),?)";  prepare z from @i;  EXECUTE z using @e;  end IF;  end $  delimiter ; |
|  |
| 1. Write a procedure(named getQualification) that takes studentID as a parameter. If studentID is present in the student table, then print his student details along with STUDENT\_QUALIFICATION details and if the studentID is not present display message “Student not found…” (Use: STUDENT, and STUDENT\_QUALIFICATION tables) |
| drop procedure if EXISTS getQualification;  delimiter $  create procedure getQualification(\_studentid int )  begin  SELECT distinct s.id into @x from student s natural join student\_qualifications sq WHERE s.id=\_studentid;  IF @x = \_studentid THEN  select \* from student s join student\_qualifications sq on s.id=sq.studentid WHERE s.id=\_studentid;  ELSE  select "Student not found…";  end if;  end $  delimiter ; |
|  |
| 1. Write a procedure (named addStudent) that inserts a new student with his phone number and his address into the STUDENT, PHONE, and ADDRESS table. |
| drop procedure if EXISTS addStudent;  delimiter $  create procedure addStudent(\_id int ,\_namefirst varchar(20) ,\_namelast varchar(30),\_dob date,  \_emailid varchar(30),\_num varchar(30),\_address varchar(30))  BEGIN  insert INTO student values(\_id,\_namefirst,\_namelast,\_dob,\_emailid);  select max(id) into @nxt from student\_phone ;    insert INTO student\_phone values(@nxt+1,\_id,\_num,1);    select max(id) into @nxt from student\_address ;  insert INTO student\_address values(@nxt+1,\_id,\_address);  end $  delimiter ;   * call addStudent(29,'omkar','Divate','2001-05-29','omkar@gamil.com','Yeola'); |
|  |
| 1. Write a procedure (named addQualification) that takes studentID, and qualification details as a parameter. If studentID is present in the STUDENT table, then insert the qualification in STUDENT\_QUALIFICATION table and return a message “Record inserted” or else print ‘Student not found’. (hint: using OUT parameter) (Use: STUDENT, and STUDENT\_QUALIFICATION tables) |
| drop procedure if EXISTS addQualification;  delimiter $  create procedure addQualification(\_studentid int, \_degree varchar(10), \_college varchar(10),  \_university varchar (10), \_marks varchar (10), \_years int)  BEGIN  SELECT true into @ch from student where id=\_studentid;  IF @ch THEN  SELECT count(\*) INTO @i from student\_qualifications;  INSERT INTO student\_qualifications VALUES(@i+1,\_studentid,\_degree,\_college,\_university,\_marks,\_years);  select 'insert successfully' status;  ELSE  select 'student not fount...' status ;  end if;  end $  delimiter ;   * call addQualification(30,'BE','RHSapat','SPPU',95,2023); * call addQualification(29,'ME','RHSapat','SPPU',95,2025); |
|  |