***NAME : Himanshu Dixit***

***ENROLL NO. : 21103262***

***BATCH : B11***

***DSW Lab***

***Week 11-12***

*PL/SQL Fundamentals (Procedures and Functions)*

***1.*** *Write a program to take four numbers as input from user. Display the largest number among them.*

***Solution :***

*DELIMITER $$*

*CREATE PROCEDURE Largest(IN A INT, IN B INT, IN C INT, IN D INT, OUT E INT)*

*BEGIN*

*IF A>B THEN*

*IF A>C THEN*

*IF A>D THEN*

*SET E=A;*

*ELSE*

*SET E=D;*

*END IF;*

*ELSE*

*IF C>D THEN*

*SET E=C;*

*ELSE*

*SET E=D;*

*END IF;*

*END IF;*

*ELSE*

*IF B>C THEN*

*IF B>D THEN*

*SET E=B;*

*ELSE*

*SET E=D;*

*END IF;*

*ELSE*

*IF C>D THEN*

*SET E=C;*

*ELSE*

*SET E=D;*

*END IF;*

*END IF;*

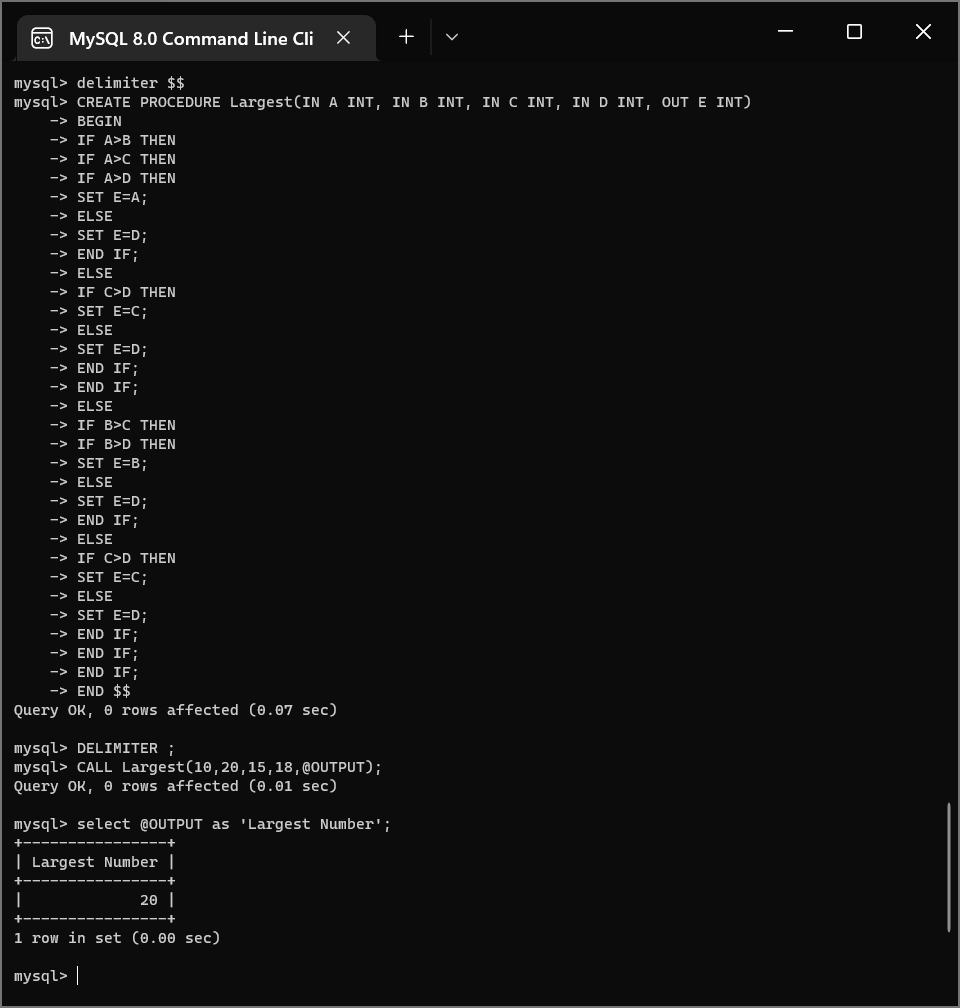
*END IF;*

*END$$*

*DELIMITER ;*

*CALL Largest(10,20,15,18,@OUTPUT);*

*select @OUTPUT as 'Largest Number';*

**

*2. Create a program that accepts a single number X. Display the message “Hello World” X times, where X is the number entered.*

***Solution :***

*DELIMITER $$*

*CREATE PROCEDURE Print(IN X INT)*

*BEGIN*

*DECLARE I INT;*

*SET I=0;*

*WHILE I<X/2 DO*

*select "Hello World";*

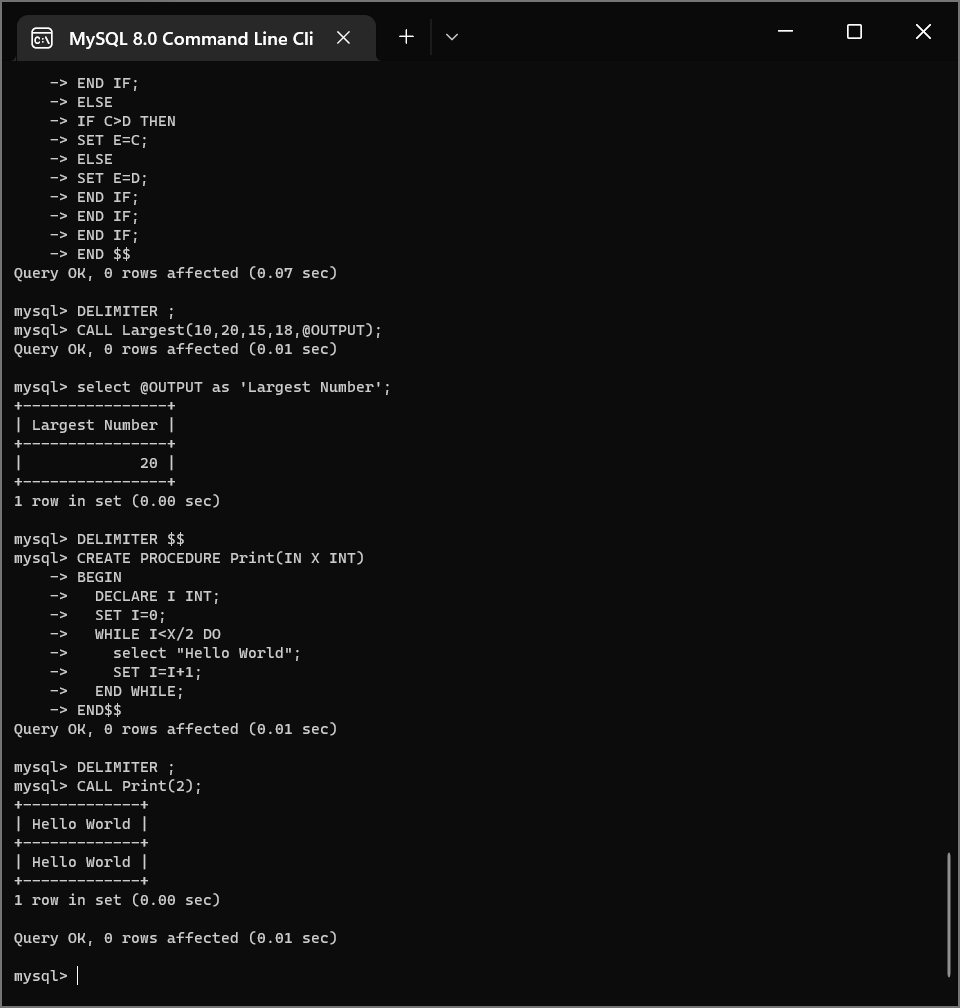
*SET I=I+1;*

*END WHILE;*

*END$$*

*DELIMITER ;*

*CALL Print(5);*

**

***3.*** *Try running the above program and entering a negative number, what happens? Change the code to print a message if a number less than 1 is entered that informs the user they must enter a number greater than or equal to 1.*

***Solution :***

*DELIMITER $$*

*CREATE PROCEDURE Print(IN X INT)*

*BEGIN*

*DECLARE I INT DEFAULT 0;*

*IF X<1 THEN*

*select "Please enter a number greater than 0!" AS '';*

*ELSE*

*WHILE I<X DO*

*SET I=I+1;*

*select "Hello World" AS '';*

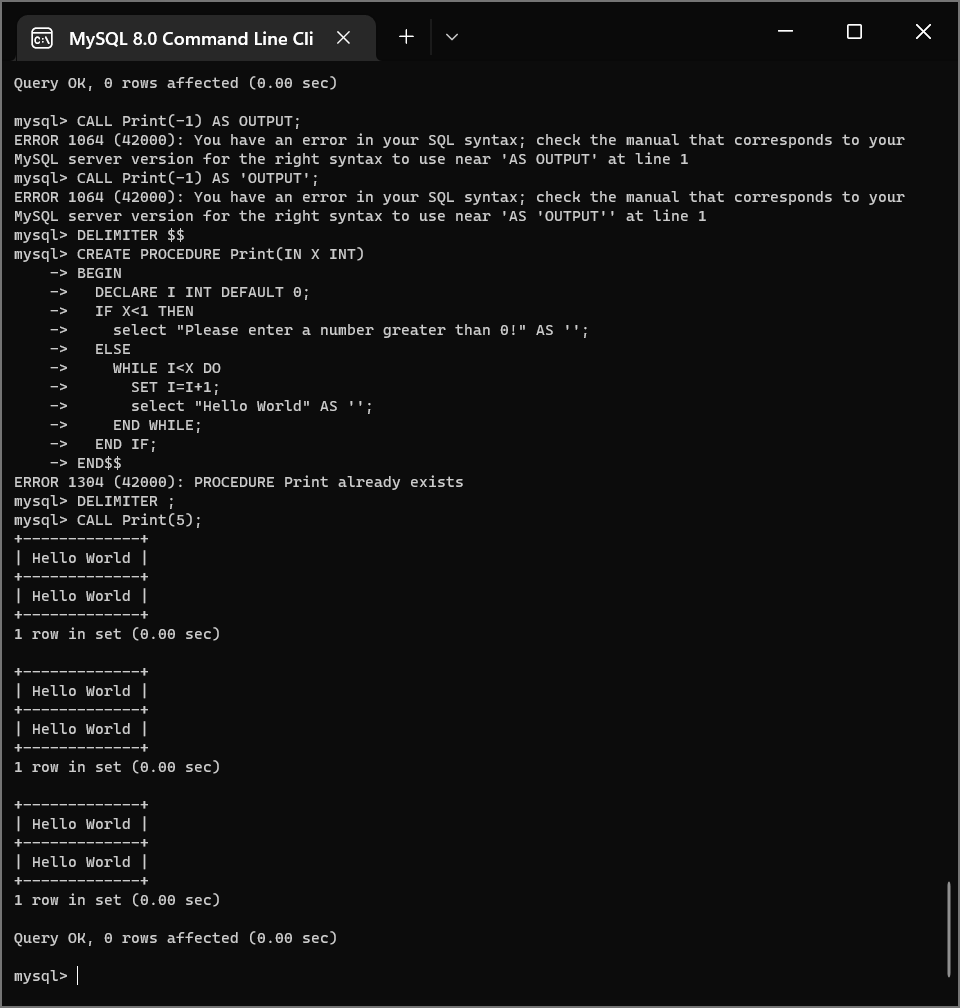
*END WHILE;*

*END IF;*

*END$$*

*DELIMITER ;*

*CALL Print(5);*

**

*SQL within PL/SQL*

*4. Create a table AREA(RADIUS NUMBER(5), AREA DECIMAL(14,2)) Write a program to calculate area of circle for radius [1, 10] and insert into the table AREA.*

***Solution :***

*CREATE TABLE AREA(RADIUS INTEGER(5), AREA DECIMAL(14,2));*

*DELIMITER $$*

*CREATE PROCEDURE Calc\_Area()*

*BEGIN*

*DECLARE I INT DEFAULT 1;*

*WHILE I<=10 DO*

*INSERT INTO AREA VALUES(I,(22\*I\*I)/7);*

*SET I=I+1;*

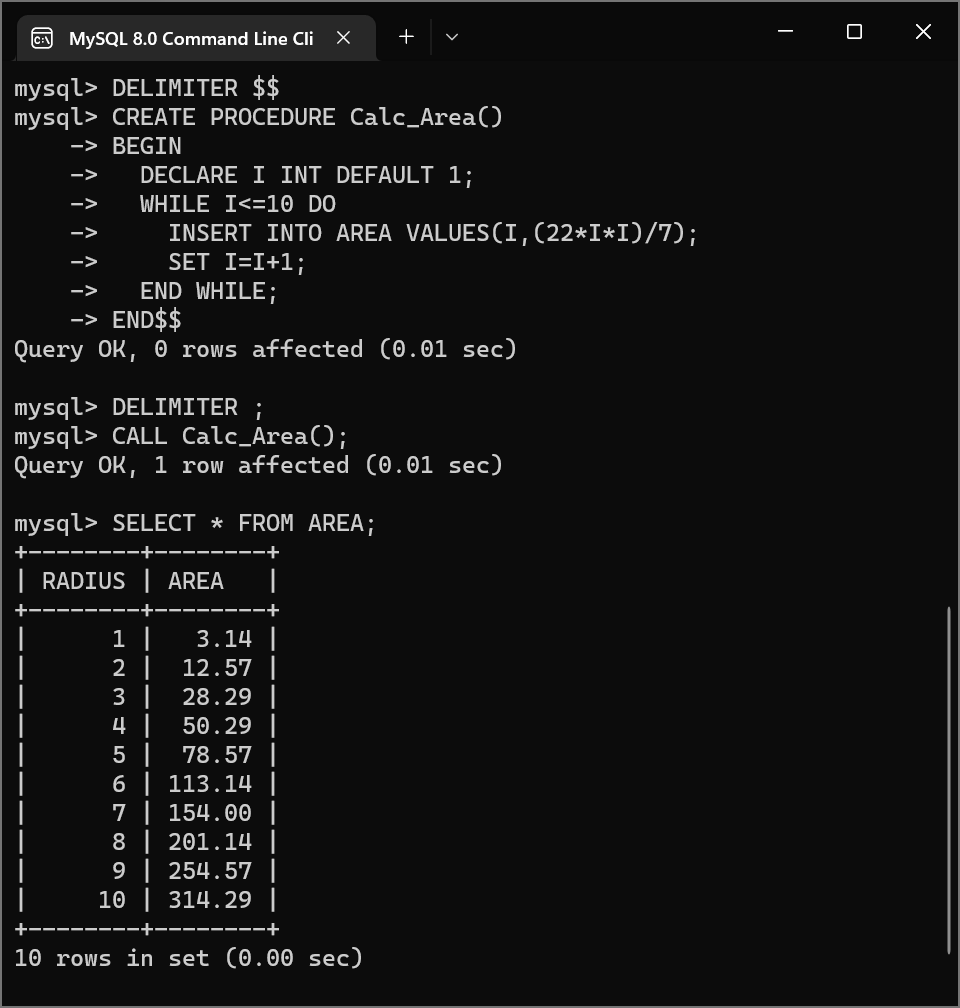
*END WHILE;*

*END$$*

*DELIMITER ;*

*CALL Calc\_Area();*

*SELECT \* FROM AREA;*

**

*5. Create a table named POWER. A record in POWER must contain a number, its square, and its cube. Insert record using PL/SQL for numbers [20-30].*

***Solution :***

*CREATE TABLE POWER(NUM INT,NUM\_SQ INT,NUM\_CUBE INT);*

*DELIMITER $$*

*CREATE PROCEDURE Calc()*

*BEGIN*

*DECLARE I INT DEFAULT 20;*

*WHILE I<=30 DO*

*INSERT INTO POWER VALUES(I,I\*I,I\*I\*I);*

*SET I=I+1;*

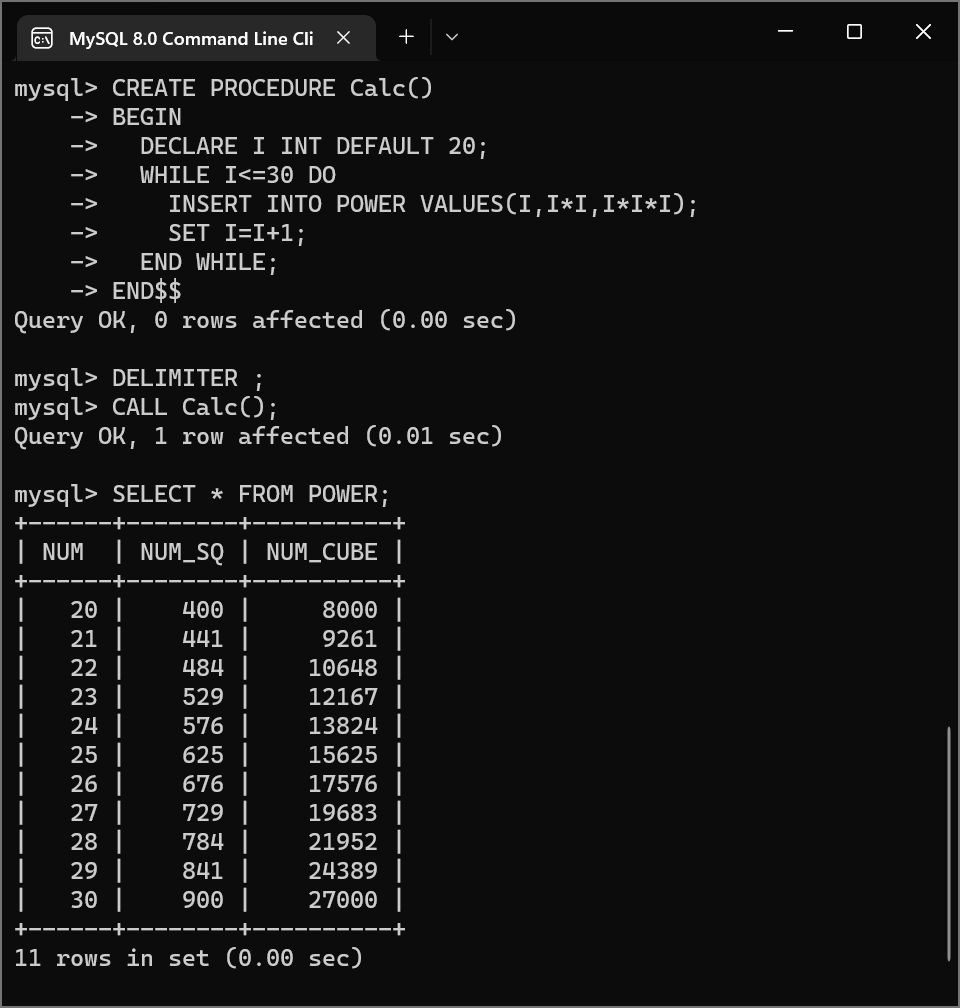
*END WHILE;*

*END$$*

*DELIMITER ;*

*CALL Calc();*

*SELECT \* FROM POWER;*

**

*6. Write a program to reverse a user input string using loop.*

***Solution :***

*DELIMITER $$*

*CREATE PROCEDURE REV(IN IP VARCHAR(30),OUT OP VARCHAR(30))*

*BEGIN*

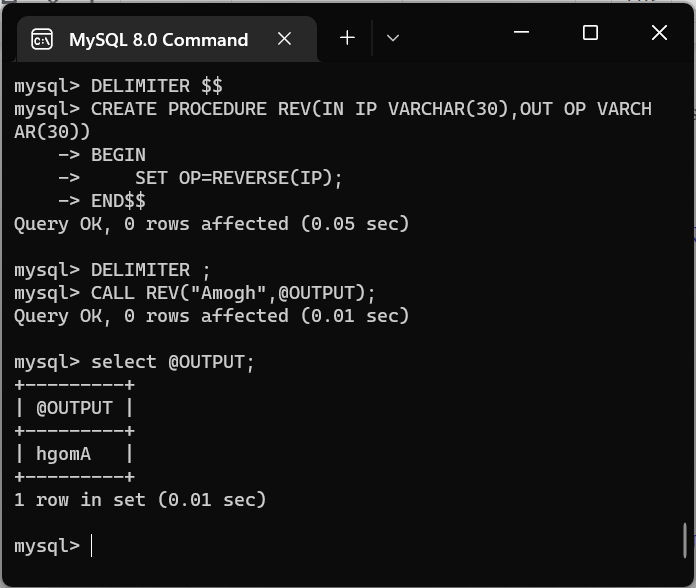
*SET OP=REVERSE(IP);*

*END$$*

*DELIMITER ;*

*CALL REV("Amogh",@OUTPUT);*

*select @OUTPUT;*

**

*7. Consider the EMPLOYEE table, having salary as one of the attributes. Write a program to increase the salary by 15% of an employee, whose SSN is entered by the user (Assume table attributes).*

***Solution :***

*CREATE TABLE EMPLOYEE(SSN INT PRIMARY KEY NOT NULL,SALARY DOUBLE);*

*INSERT INTO EMPLOYEE VALUES(1,10000);*

*INSERT INTO EMPLOYEE VALUES(2,15000);*

*DELIMITER $$*

*CREATE PROCEDURE Upd(IN SSN1 INT)*

*BEGIN*

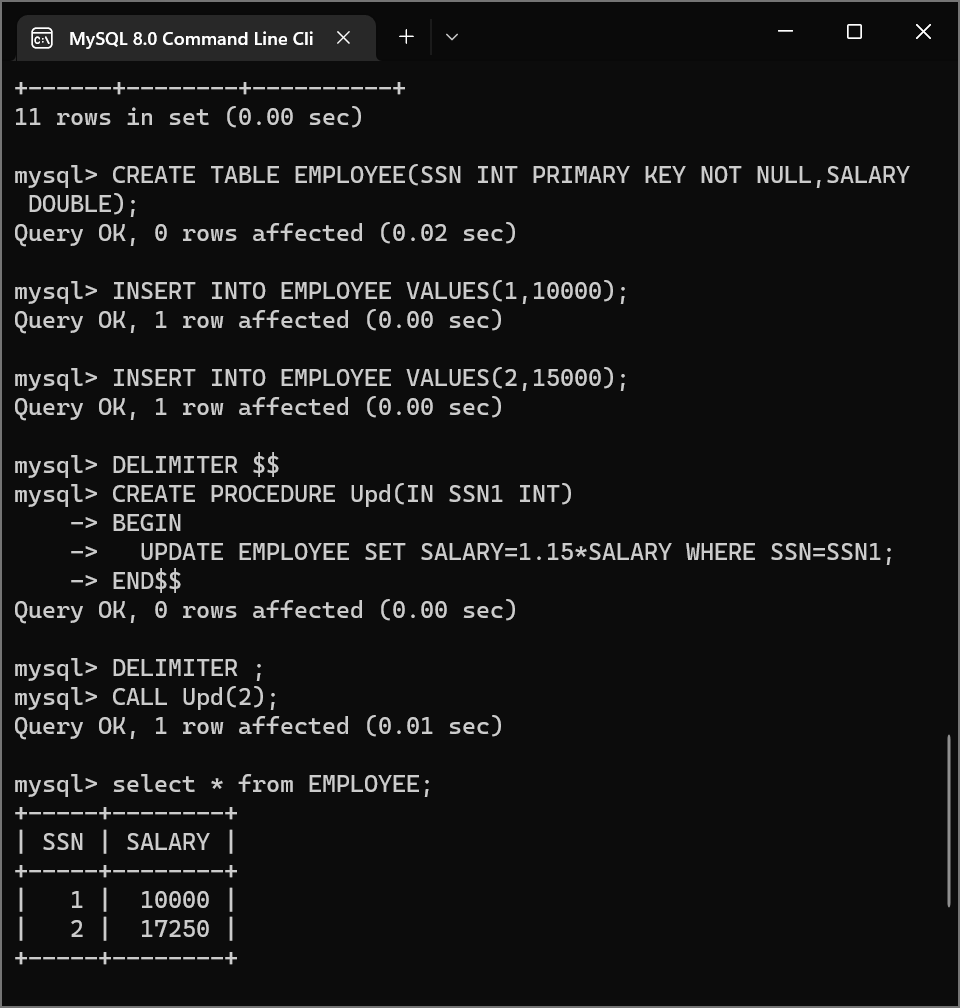
*UPDATE EMPLOYEE SET SALARY=1.15\*SALARY WHERE SSN=SSN1;*

*END$$*

*DELIMITER ;*

*CALL Upd(2);*

*select \* from EMPLOYEE;*

**

*8. Create a PL/SQL block that accepts a* ***new*** *job title and an* ***old*** *job title. Find all employees in the old job and give them the new job. Ensure a valid message is given back to the user even if no employees changed job.*

***Solution :***

*CREATE TABLE EMPLOYEE(SSN INT,JOB\_TITLE VARCHAR(30));*

*INSERT INTO EMPLOYEE VALUES(1,'Manager');*

*INSERT INTO EMPLOYEE VALUES(2,'Clerk');*

*INSERT INTO EMPLOYEE VALUES(3,'Supervior');*

*INSERT INTO EMPLOYEE VALUES(4,'Clerk');*

*INSERT INTO EMPLOYEE VALUES(5,'Mentor');*

*INSERT INTO EMPLOYEE VALUES(6,'Clerk');*

*DELIMITER $$*

*CREATE PROCEDURE CHNG(IN OLD\_JOB VARCHAR(30),IN NEW\_JOB VARCHAR(30))*

*BEGIN*

*UPDATE EMPLOYEE SET JOB\_TITLE=NEW\_JOB WHERE JOB\_TITLE=OLD\_JOB;*

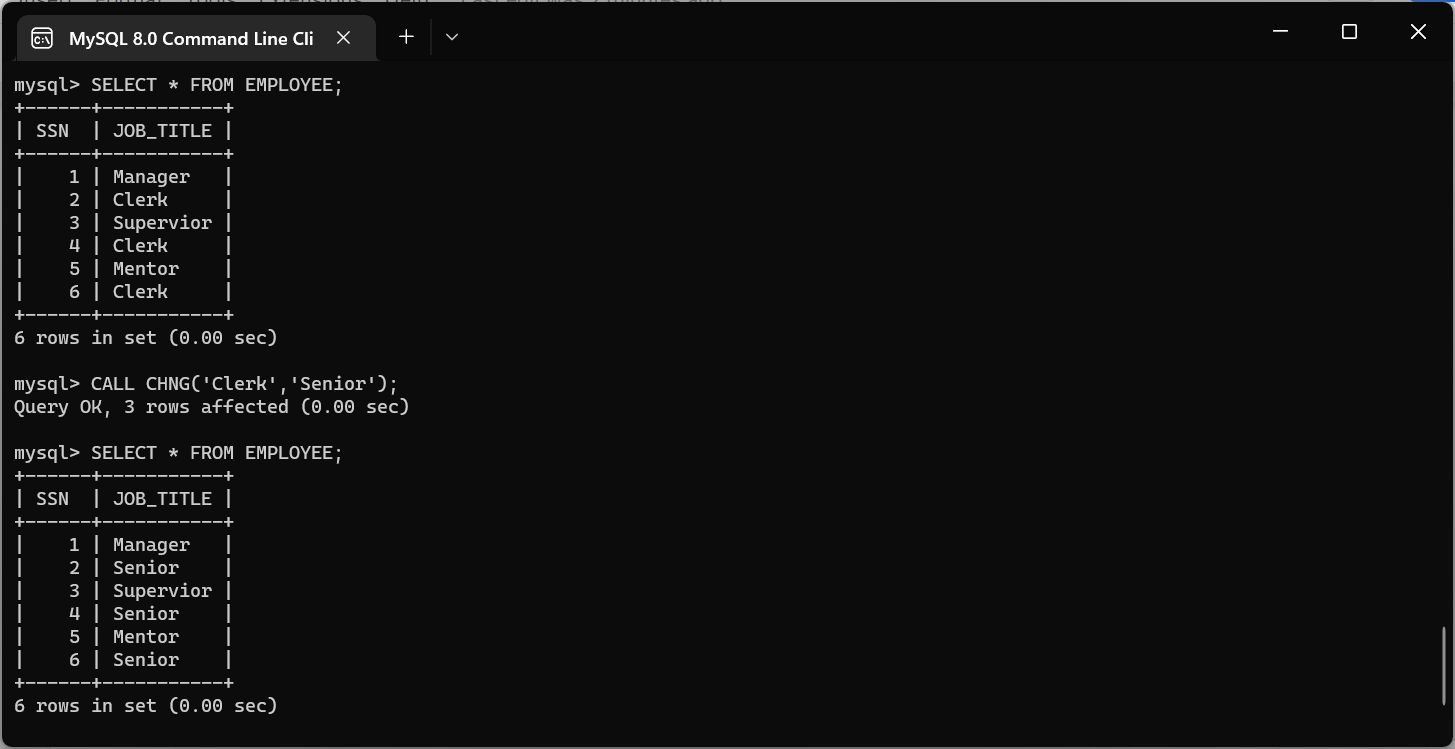
*END$$*

*DELIMITER ;*

*SELECT \* FROM EMPLOYEE;*

*CALL CHNG('Clerk','Senior');*

*SELECT \* FROM EMPLOYEE;*

**

*Cursors*

*9. Create a cursor that displays the name and salary of each employee in the EMPLOYEES table whose salary is less than that specified by a passed-in parameter value (Assume table attributes).*

***Solution :***

*CREATE TABLE EMPLOYEE(NAME VARCHAR(20),SALARY INT);*

*INSERT INTO EMPLOYEE VALUES('Amogh',100000);*

*INSERT INTO EMPLOYEE VALUES('Aditya',50000);*

*INSERT INTO EMPLOYEE VALUES('Shreya',60000);*

*INSERT INTO EMPLOYEE VALUES('Atharva',50000);*

*INSERT INTO EMPLOYEE VALUES('Archie',75000);*

*INSERT INTO EMPLOYEE VALUES('Ali',50000);*

*create table temp(name varchar(20),salary int);*

*DELIMITER $$*

*create procedure disp (in sal int)*

*begin*

*declare finished integer default 0;*

*declare n varchar(20);*

*declare s int;*

*declare curdetail cursor*

*for*

*select NAME, SALARY from EMPLOYEE where SALARY<sal;*

*declare continue handler for not found set finished =1;*

*open curdetail;*

*getdetail : loop*

*fetch curdetail into n, s;*

*if finished =1 then*

*leave getdetail;*

*end if;*

*insert into temp values(n,s);*

*end loop getdetail;*

*close curdetail;*

*end$$*

*DELIMITER ;*

*call disp(65000);*

*SELECT \* from temp;*

**

*Triggers*

*10. Consider a table MEMBERS (MID, NAME, EMAIL, DOB).*

*Consider another table REMINDERS (ID, MID, MESSAGE) that stores reminder messages to update DOB.*

*Create an AFTER INSERT trigger that inserts a reminder message into the REMINDERS table if the birth date of the member is inserted as NULL.*

***Explanation:***

*If we insert two rows into the MEMBERS table as:*

*INSERT INTO members (mid, name, email, DOB)*

*VALUES*

*(‘MID2’, 'John Doe', 'john.doe@example.com', NULL),*

*(‘MID3’'KimKole', 'kim.kole@example.com','2000-01-01');*

*And then, query data from the REMINDERStable:*

*SELECT \* FROM reminders;*

*It should display a record as*

***ID MID MESSAGE***

*1 MID2 Your DOB is NULL. Please update.*

*We inserted two rows into the members table. However, only the first row has a birth date value NULL, therefore, the trigger inserted only one row into the reminders table.*

***Solution :***

*DROP TABLE IF EXISTS members;*

*CREATE TABLE members (*

*id VARCHAR(30),*

*name VARCHAR(100) NOT NULL,*

*email VARCHAR(255),*

*birthDate DATE,*

*PRIMARY KEY (id)*

*);*

*DROP TABLE IF EXISTS reminders;*

*CREATE TABLE reminders (*

*id INT AUTO\_INCREMENT,*

*memberId VARCHAR(30),*

*message VARCHAR(255) NOT NULL,*

*PRIMARY KEY (id , memberId)*

*);*

*DELIMITER $$*

*CREATE TRIGGER after\_members\_insert*

*AFTER INSERT*

*ON members FOR EACH ROW*

*BEGIN*

*DECLARE A VARCHAR(30);*

*SELECT ID INTO A FROM members limit 1;*

*IF NEW.birthDate IS NULL THEN*

*INSERT INTO reminders(memberId, message)*

*VALUES(new.id,CONCAT("Your DOB is NULL. Please update."));*

*END IF;*

*END$$*

*DELIMITER ;*

*INSERT INTO members VALUES('MID2', 'John Doe', 'john.doe@example.com', NULL),*

*('MID3','KimKole', 'kim.kole@example.com','2000-01-01');*

*SELECT \* FROM reminders;*

