DSW Week 7 and 8

Name: Maanya Gupta Enrollment no.: 23103258

Batch: B9

```
Q1
DELIMITER //
CREATE PROCEDURE largest_number(IN a INT, IN b INT, IN c INT, IN d INT)
BEGIN
DECLARE largest INT;
SET largest = a;
IF b > largest THEN
SET largest = b;
END IF;
IF c > largest THEN
SET largest = c;
END IF;
IF d > largest THEN
SET largest = d;
END IF;
SELECT CONCAT('The largest number is: ', largest) AS result;
END //
DELIMITER;
CALL largest_number(14, 8, 18, 3);
```

```
MySQL 8.0 Command Line Cli X
 mysql> use plsql;
 Database changed
 mysql> DELIMITER //
 mysql> CREATE PROCEDURE largest_number(IN a INT, IN b INT, IN c INT, IN d INT)
    -> BEGIN
    -> DECLARE largest INT;
    -> SET largest = a;
     -> IF b > largest THEN
     -> SET largest = b;
    -> END IF;
-> IF c > largest THEN
    -> SET largest = c;
    -> END IF;
    -> IF d > largest THEN
    -> SET largest = d;
    -> END IF;
    -> SELECT CONCAT('The largest number is: ', largest) AS result;
 Query OK, 0 rows affected (0.07 sec)
 mysql> DELIMITER ;
 mysql> CALL largest_number(14, 8, 18, 3);
 result
 | The largest number is: 18 |
 1 row in set (0.01 sec)
Q2
DELIMITER //
CREATE PROCEDURE print_hello_world(IN x INT)
BEGIN
DECLARE I INT DEFAULT 1;
IF x < 1 THEN
SELECT 'Please enter a number greater than or equal to 1' AS result;
WHILE i <= x DO
SELECT 'Hello World' AS result;
SETi = i + 1;
END WHILE;
END IF;
END //
DELIMITER;
```

CALL print_hello_world(5);

```
lacktriangleq MySQL 8.0 Command Line Cli 	imes 	imes 	imes 	imes
mysql> DELIMITER //
mysql> CREATE PROCEDURE print_hello_world(IN x INT)
   -> BEGIN
   -> DECLARE i INT DEFAULT 1;
   \rightarrow IF x < 1 THEN
    -> SELECT 'Please enter a number greater than or equal to 1' AS result;
   -> WHILE i <= x DO
   -> SELECT 'Hello World' AS result;
   -> SET i = i + 1;
   -> END WHILE;
   -> END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)
mysql> DELIMITER ;
mysql> CALL print_hello_world(5);
result
| Hello World |
1 row in set (0.00 sec)
 result
 Hello World |
```

```
MySQL 8.0 Command Line Cli X
result
| Hello World |
1 row in set (0.01 sec)
result
| Hello World |
1 row in set (0.01 sec)
result
| Hello World |
1 row in set (0.01 sec)
result
| Hello World |
1 row in set (0.01 sec)
Query OK, 0 rows affected (0.01 sec)
mysql>
```

Q3

```
Q4
CREATE TABLE AREA (
RADIUS INT,
AREA DECIMAL(14,2)
);
DELIMITER //
CREATE PROCEDURE insert_area_circle()
BEGIN
DECLARE radius INT;
DECLARE area_circle DECIMAL(14,2);
SET radius = 1;
WHILE radius <= 10 DO
SET area_circle = 3.14 * radius * radius;
INSERT INTO AREA (RADIUS, AREA) VALUES (radius, area_circle);
SET radius = radius + 1;
END WHILE;
SELECT 'Data inserted into AREA table' AS result;
END //
DELIMITER;
```

CALL insert_area_circle();

```
MySQL 8.0 Command Line Cli X
mysql> CREATE TABLE AREA (
   -> RADIUS INT,
   -> AREA DECIMAL(14,2)
   -> );
Query OK, 0 rows affected (0.10 sec)
mysql> DELIMITER //
mysql> CREATE PROCEDURE insert_area_circle()
   -> BEGIN
   -> DECLARE radius INT;
   -> DECLARE area_circle DECIMAL(14,2);
   -> SET radius = 1;
   -> WHILE radius <= 10 DO
   -> SET area_circle = 3.14 * radius * radius;
   -> INSERT INTO AREA (RADIUS, AREA) VALUES (radius, area_circle);
   -> SET radius = radius + 1;
   -> END WHILE;
   -> SELECT 'Data inserted into AREA table' AS result;
   -> END //
Query OK, 0 rows affected (0.01 sec)
mysql> DELIMITER ;
mysql> CALL insert_area_circle();
result
Data inserted into AREA table
1 row in set (0.04 sec)
```

```
MySQL 8.0 Command Line Cli X
mysql> DELIMITER ;
mysql> CALL insert_area_circle();
result
Data inserted into AREA table
1 row in set (0.04 sec)
Query OK, 0 rows affected (0.05 sec)
mysql> select * from AREA;
| RADIUS | AREA
       1 I
             3.14
       2
            12.56
       3 I
            28.26
       4 |
            50.24
       5
            78.50
       6
         113.04
       7 | 153.86
           200.96
       8
       9
           254.34
      10 | 314.00 |
10 rows in set (0.00 sec)
```

```
Q5
CREATE TABLE POWER (
numb INT,
square int,
cube val int
);
DELIMITER //
CREATE PROCEDURE power_func()
BEGIN
DECLARE numb INT;
DECLARE square INT;
DECLARE cube_val INT;
SET numb = 20;
WHILE numb <= 30 DO
SET square = numb * numb;
SET cube_val = numb * numb * numb;
INSERT INTO POWER VALUES (numb,square,cube_val);
SET numb = numb + 1;
END WHILE;
SELECT * from POWER;
END //
```

DELIMITER;

CALL power_func();

```
mysql> CREATE TABLE POWER (
    -> numb INT,
    -> square int,
    -> cube_val int
    -> );
```

```
MySQL 8.0 Command Line Cli X + √
mysql> CREATE PROCEDURE power_func()
    -> BEGIN
    -> DECLARE numb INT;
    -> DECLARE square INT;
    -> DECLARE cube_val INT;
    -> SET numb = 20;
    -> WHILE numb <= 30 DO
    -> SET square = numb * numb;
    -> SET cube_val = numb * numb * numb;
    -> INSERT INTO POWER VALUES (numb, square, cube_val);
    -> SET numb = numb + 1;
    -> END WHILE;
    -> SELECT * from POWER;
    -> END //
Query OK, 0 rows affected (0.00 sec)
mysql> DELIMITER ;
mysql> CALL power_func();
```

```
MySQL 8.0 Command Line Cli X
mysql> DELIMITER ;
mysql> CALL power_func();
| numb | square | cube_val
            400 |
    20 I
                       8000
    20
            400
                       8000
    20
            400
                       8000
    21
            441 |
                       9261
    22 I
            484
                      10648
    23 I
            529
                      12167
    24 |
            576
                     13824
    25 I
            625 I
                      15625
                      17576
    26 I
            676 I
    27 I
            729
                      19683
    28 I
            784 |
                      21952
    29
            841 |
                      24389
    30 l
            900 |
                      27000
13 rows in set (0.03 sec)
Query OK, 0 rows affected (0.04 sec)
```

```
Q6
DELIMITER //
CREATE PROCEDURE reverse_string(IN input_string VARCHAR(100))
BEGIN
DECLARE reversed_string VARCHAR(100) DEFAULT ";
DECLARE i INT DEFAULT LENGTH(input_string);
WHILE i > 0 DO
SET reversed_string = CONCAT(reversed_string, SUBSTRING(input_string, i, 1));
SET i = i - 1;
END WHILE;
SELECT reversed_string AS reversed;
END //
DELIMITER;
CALL reverse_string('palindrome');
```

```
mysql> CREATE PROCEDURE reverse_string(IN input_string VARCHAR(100))
   -> DECLARE reversed_string VARCHAR(100) DEFAULT '';
   -> DECLARE i INT DEFAULT LENGTH(input_string);
   -> WHILE i > 0 DO
   -> SET reversed_string = CONCAT(reversed_string, SUBSTRING(input_string, i, 1));
   -> SET i = i - 1;
   -> END WHILE;
   -> SELECT reversed_string AS reversed;
    -> END //
Query OK, 0 rows affected (0.02 sec)
mysql> DELIMITER ;
mysql> CALL reverse_string('palindrome');
reversed
 emordnilap |
1 row in set (0.00 sec)
Query OK, 0 rows affected (0.01 sec)
```

Q7 create table employee(ssn int primary key, salary DECIMAL(10,2)); DELIMITER // CREATE PROCEDURE increase salary(IN employee ssn INT) **BEGIN** DECLARE current_salary DECIMAL(10,2); SELECT salary INTO current salary FROM EMPLOYEE WHERE SSN = employee ssn; **UPDATE EMPLOYEE** SET salary = current_salary * 1.15 WHERE SSN = employee ssn; SELECT CONCAT('Salary updated successfully for SSN: ', employee_ssn) AS result; END // **DELIMITER**; CALL increase salary(123456789);

```
MySQL 8.0 Command Line Cli X
 Query OK, 0 rows affected (0.03 sec)
 mysql>
 mysql>
 mysql> DELIMITER //
 mysql> CREATE PROCEDURE increase_salary(IN employee_ssn INT)
     -> BEGIN
    -> DECLARE current_salary DECIMAL(10,2);
     -> SELECT salary INTO current_salary
     -> FROM EMPLOYEE
     -> WHERE SSN = employee_ssn;
     -> UPDATE EMPLOYEE
     -> SET salary = current_salary * 1.15
     -> WHERE SSN = employee_ssn;
     -> SELECT CONCAT('Salary updated successfully for SSN: ', employee_ssn) AS result;
     -> END //
 Query OK, 0 rows affected (0.00 sec)
 mysql> DELIMITER ;
 mysql> CALL increase_salary(123456789);
  result
  Salary updated successfully for SSN: 123456789
 1 row in set (0.01 sec)
 Query OK, 0 rows affected (0.01 sec)
Q8
create table EMPLOYEE(
job title varchar(50),
salary int);
DELIMITER $$
CREATE PROCEDURE update_job_title(IN old_job_title VARCHAR(100), IN new_job_title
VARCHAR(100))
BEGIN
DECLARE rows_updated INT;
UPDATE EMPLOYEE
SET job title = new job title
WHERE job_title = old_job_title;
SET rows_updated = ROW_COUNT();
IF rows updated > 0 THEN
SELECT CONCAT('Job titles updated for ', rows updated, ' employees.') AS result;
ELSE
SELECT 'No employees found with the old job title.' AS result;
END IF;
END $$
DELIMITER;
```

CALL update_job_title('Developer', 'Senior Developer');

```
MySQL 8.0 Command Line Cli × + v
 mysql> create table EMPLOYEE(
    -> job_title varchar(50),
 -> salary int);
Query OK, 0 rows affected (0.02 sec)
 mysql> DELIMITER $$
 mysql> CREATE PROCEDURE update_job_title(IN old_job_title VARCHAR(100), IN new_job_title
    -> VARCHAR(100))
    -> BEGIN
    -> DECLARE rows_updated INT;
    -> UPDATE EMPLOYEE
    -> SET job_title = new_job_title
    -> WHERE job_title = old_job_title;
    -> SET rows_updated = ROW_COUNT();
    -> IF rows_updated > 0 THEN
    -> SELECT CONCAT('Job titles updated for ', rows_updated, ' employees.') AS result;
    -> SELECT 'No employees found with the old job title.' AS result;
    -> END IF;
    -> END $$
 Query OK, 0 rows affected (0.00 sec)
 mysql> DELIMITER ;
 mysql> CALL update_job_title('Developer', 'Senior Developer');
 result
 No employees found with the old job title.
 1 row in set (0.00 sec)
 Query OK, 0 rows affected (0.01 sec)
Q9
DELIMITER //
CREATE PROCEDURE get low salary employees(IN salary limit DECIMAL(10,2))
BEGIN
DECLARE done INT DEFAULT 0;
DECLARE emp name VARCHAR(100);
DECLARE emp_salary DECIMAL(10,2);
DECLARE emp_cursor CURSOR FOR
SELECT name, salary FROM EMPLOYEES WHERE salary < salary limit;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
OPEN emp cursor;
read loop: LOOP
FETCH emp_cursor INTO emp_name, emp_salary;
IF done THEN
LEAVE read loop;
END IF;
SELECT CONCAT('Name: ', emp_name, ' | Salary: ', emp_salary) AS result;
END LOOP;
CLOSE emp_cursor;
END //
DELIMITER;
CALL get_low_salary_employees(50000);
```

```
MySQL 8.0 Command Line Cli X
1 row in set (0.00 sec)
Query OK, 0 rows affected (0.01 sec)
mysql> DELIMITER //
mysql> CREATE PROCEDURE get_low_salary_employees(IN salary_limit DECIMAL(10,2))
    -> BEGIN
   -> DECLARE done INT DEFAULT 0;
   -> DECLARE emp_name VARCHAR(100);
   -> DECLARE emp_salary DECIMAL(10,2);
   -> DECLARE emp_cursor CURSOR FOR
   -> SELECT name, salary FROM EMPLOYEES WHERE salary < salary_limit;
   -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
   -> OPEN emp_cursor;
   -> read_loop: LOOP
   -> FETCH emp_cursor INTO emp_name, emp_salary;
   -> IF done THEN
   -> LEAVE read_loop;
   -> END IF;
   -> SELECT CONCAT('Name: ', emp_name, ' | Salary: ', emp_salary) AS result;
   -> END LOOP;
   -> CLOSE emp_cursor;
   -> END //
Query OK, 0 rows affected (0.01 sec)
```

```
Q10
CREATE TABLE MEMBERS (
MID VARCHAR(10),
NAME VARCHAR(100),
EMAIL VARCHAR(100),
DOB DATE
);
CREATE TABLE REMINDERS (
ID INT AUTO INCREMENT PRIMARY KEY,
MID VARCHAR(10),
MESSAGE VARCHAR(255)
);
DELIMITER $$
CREATE TRIGGER insert_reminder_if_dob_null
AFTER INSERT ON MEMBERS
FOR EACH ROW
BEGIN
IF NEW.DOB IS NULL THEN
INSERT INTO REMINDERS (MID, MESSAGE)
VALUES (NEW.MID, 'Your DOB is NULL. Please update.');
END IF:
END $$
DELIMITER:
INSERT INTO MEMBERS (MID, NAME, EMAIL, DOB)
('MID2', 'John Doe', 'john.doe@example.com', NULL),
```

('MID3', 'Kim Kole', 'kim.kole@example.com', '2000-01-01'); SELECT * FROM REMINDERS;

```
MySQL 8.0 Command Line Cli X + √
ERROR 1146 (42S02): Table 'plsql.employees' doesn't exist
mysql> CREATE TABLE MEMBERS (
    -> MID VARCHAR(10),
    -> NAME VARCHAR(100)
    -> EMAIL VARCHAR(100),
    -> DOB DATE
    -> );
Query OK, 0 rows affected (0.02 sec)
mysql> CREATE TABLE REMINDERS (
    -> ID INT AUTO_INCREMENT PRIMARY KEY,
    -> MID VARCHAR(10),
    -> MESSAGE VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.01 sec)
mysql> DELIMITER $$
mysql> CREATE TRIGGER insert_reminder_if_dob_null
    -> AFTER INSERT ON MEMBERS
    -> FOR EACH ROW
    -> BEGIN
    -> IF NEW.DOB IS NULL THEN
    -> INSERT INTO REMINDERS (MID, MESSAGE)
    -> VALUES (NEW.MID, 'Your DOB is NULL. Please update.');
    -> END IF;
    -> END $$
Query OK, 0 rows affected (0.01 sec)
mysql> DELIMITER ;
mysql> INSERT INTO MEMBERS (MID, NAME, EMAIL, DOB)
    -> VALUES
-> ('MID2', 'John Doe', 'john.doe@example.com', NULL),
-> ('MID3', 'Kim Kole', 'kim.kole@example.com', '2000-01-01');
Query OK, 2 rows affected (0.01 sec)
Records: 2 Duplicates: 0 Warnings: 0
mysql> SELECT * FROM REMINDERS;
 ID | MID | MESSAGE
  1 | MID2 | Your DOB is NULL. Please update. |
```