FAIZAN CHOUDHARY

20BCS021

DBMS LAB

25th April 2022

1. Write a SQL function and stored procedure for average of three numbers.

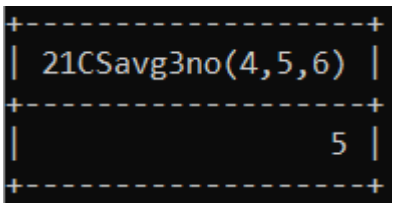
Function:

```
mysql> use 20bcs021_faizan;
mysql> DELIMITER ]        // to change the default delimiter to ]
mysql> CREATE FUNCTION 21CSavg3no (a INT, b INT, c INT)
    -> RETURNS INTEGER
    -> BEGIN
    ->     DECLARE sum, avg INT;
    ->     SET sum = a+b+c;
    ->     SET avg = sum/3;
    ->     RETURN avg;
    -> END]
mysql> DELIMITER ;        // to change back the default delimiter to ;
```

Function Query:

```
mysql> SELECT 21CSavg3no (4,5,6);
```

OUTPUT:



```
+-------------------+
| 21CSavg3no(4,5,6) |
+-------------------+
|                 5 |
+-------------------+
```

Stored Procedure:

```
mysql> DELIMITER ]
mysql> CREATE PROCEDURE 21CSavg3no (IN a INT, IN b INT, IN c INT, OUT t INT)
    -> BEGIN
    ->     DECLARE sum INT;
    ->     SET sum = a+b+c;
    ->     SET t = sum/3;
    -> END]
```

Stored Procedure Query:

```
mysql> CALL 21CSavg3no (4,5,5,@avg)]
mysql> SELECT @avg]                    // since it returns an INT
```

OUTPUT:

```
+------+
| @avg |
+------+
|    5 |
+------+
```

2. Write a SQL function and stored procedure to calculate factorial.

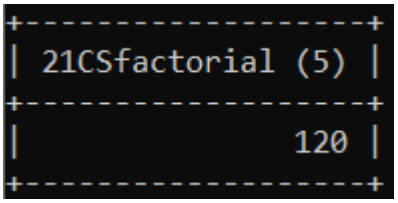
Function:

- ```
  mysql> CREATE FUNCTION 21CSfactorial (n INT)
      -> RETURNS INT
      -> BEGIN
      ->    DECLARE f,i INT DEFAULT 1;
      ->    myloop:LOOP
      ->    IF i > n THEN
      ->         LEAVE myloop;
      ->    ELSE
      ->         SET f = f * i;
      ->         SET i = i + 1;
      ->    ITERATE myloop;
      ->    END IF;
      ->    END LOOP;
      ->    RETURN f;
      -> END]
  ```

Function Query:

```
mysql> SELECT 21CSfactorial (5)]
```

OUTPUT:



```
+-------------------+
| 21CSfactorial (5) |
+-------------------+
|               120 |
+-------------------+
```

Stored Procedure:

- ```
  mysql> CREATE PROCEDURE 21CSfactorial (IN n INT, OUT fact INT)
      -> BEGIN
      ->    DECLARE f,i INT DEFAULT 1;
      ->     myloop:LOOP
      ->     IF i > n THEN
      ->         LEAVE myloop;
      ->    ELSE
  ```

```
->          SET f = f * i;
->          SET i = i + 1;
->    ITERATE myloop;
->    END IF;
->    END LOOP;
->    SET fact = f;
-> END]
```

Stored Procedure Query:

```
mysql> CALL 21CSfactorial (6,@fact)]
mysql> SELECT @fact]
```

OUTPUT:

```
+-------+
| @fact |
+-------+
|   720 |
+-------+
```

3. Write a SQL function and stored procedure to print fibonacci series upto n terms and its sum.

Function:

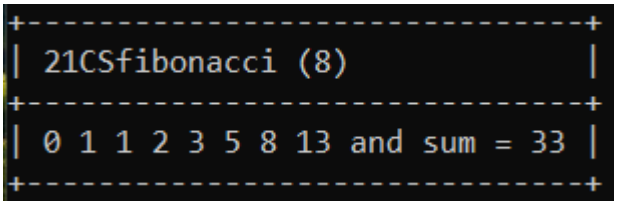
- ```
  mysql> CREATE FUNCTION 21CSfibonacci (n INT)
      -> RETURNS VARCHAR(1000)
      -> BEGIN
      ->    DECLARE i INT DEFAULT 3;
      ->    DECLARE a, temp INT DEFAULT 0;
      ->    DECLARE b, sum INT DEFAULT 1;
      ->    DECLARE str VARCHAR(1000);
      ->    SET str = CAST(a AS CHAR(2));
      ->    SET str = CONCAT(str, " ");
      ->    myloop:LOOP
      ->    IF i > n THEN
      ->         LEAVE myloop;
      ->    ELSE
      ->         SET temp = a + b;
      ->         SET a = b;
      ->         SET b = temp;
  ```
  ```

```
->          SET i = i + 1;
->          SET sum = sum + temp;
->          SET str = CONCAT(str, CAST(a AS CHAR(2)));
->          SET str = CONCAT(str, " ");
->     END IF;
->     END LOOP;
->     SET str = CONCAT(str, CAST(b AS CHAR(2)));
->     SET str = CONCAT(str, " and sum = ");
->     SET str = CONCAT(str, CAST(sum AS CHAR(3)));
->     RETURN str;
-> END]
```

Function Query:

```
mysql> SELECT 21CSfibonacci (8)]
```

OUTPUT:



```
+-------------------------------+
| 21CSfibonacci (8)             |
+-------------------------------+
| 0 1 1 2 3 5 8 13 and sum = 33 |
+-------------------------------+
```

Stored Procedure:

-
```
mysql> CREATE PROCEDURE 21CSfibonacci (IN n INT, OUT retStr
VARCHAR(1000))
    -> BEGIN
    ->     DECLARE i INT DEFAULT 3;
    ->     DECLARE a, temp INT DEFAULT 0;
    ->     DECLARE b, sum INT DEFAULT 1;
    ->     DECLARE str VARCHAR(1000);
    ->     SET str = CAST(a AS CHAR(2));
    ->     SET str = CONCAT(str, " ");
    ->     myloop:LOOP
    ->     IF i > n THEN
    ->          LEAVE myloop;
    ->     ELSE
    ->          SET temp = a + b;
    ->          SET a = b;
    ->          SET b = temp;
```

```
    ->         SET i = i + 1;
    ->         SET sum = sum + temp;
    ->         SET str = CONCAT(str, CAST(a AS CHAR(2)));
    ->         SET str = CONCAT(str, " ");
    ->     END IF;
    ->     END LOOP;
    ->     SET str = CONCAT(str, CAST(b AS CHAR(2)));
    ->     SET str = CONCAT(str, " and sum = ");
    ->     SET str = CONCAT(str, CAST(sum AS CHAR(3)));
    ->     SET retStr = str;
    -> END]
```

Stored Procedure Query:

```
mysql> CALL 21CSfibonacci (10,@str)]
mysql> SELECT @str]
```

OUTPUT:

```
+--------------------------------------+
| @str                                 |
+--------------------------------------+
| 0 1 1 2 3 5 8 13 21 34 and sum = 88 |
+--------------------------------------+
```

4. Write a SQL function and stored procedure to calculate age.

Function:

- ```
  mysql> CREATE FUNCTION 21CScalcAge(dat DATE)
      -> RETURNS VARCHAR(25)
      -> BEGIN
      ->     DECLARE curDate DATE DEFAULT CURRENT_DATE();
      ->     DECLARE tempDate DATE;
      ->     DECLARE year, month, date INT DEFAULT 0;
      ->     DECLARE str VARCHAR(25) DEFAULT "";
      ->     SET year = TIMESTAMPDIFF(YEAR, dat, curDate);
      ->     SET month = TIMESTAMPDIFF(MONTH, dat, curDate);
      ->     SET month = month - (year * 12);
      ->     SET tempDate = DATE_ADD(dat, INTERVAL year YEAR);
      ->     SET tempDate = DATE_ADD(tempDate, INTERVAL month
  MONTH);
  ```
  ```

```
    ->     SET date = DATEDIFF(curDate, tempDate) + 1;
    ->     SET str = CONCAT(str, CAST(year AS char(2)));
    ->     SET str = CONCAT(str, "Y ");
    ->     SET str = CONCAT(str, CAST(month AS char(2)));
    ->     SET str = CONCAT(str, "M ");
    ->     SET str = CONCAT(str, CAST(date AS char(2)));
    ->     SET str = CONCAT(str, "D");
    ->     RETURN str;
    -> END]
```

Function Query:

```
mysql> SELECT 21CScalcAge ('2002-03-30')]
```

OUTPUT:

```
+----------------------------+
| 21CScalcAge ('2002-03-30') |
+----------------------------+
| 20Y 0M 28D                 |
+----------------------------+
```

Stored Procedure:

- ```
  mysql> CREATE PROCEDURE 21CScalcAge(IN dat DATE, OUT retStr
  VARCHAR(25))
      -> BEGIN
      ->     DECLARE curDate DATE DEFAULT CURRENT_DATE();
      ->     DECLARE tempDate DATE;
      ->     DECLARE year, month, date INT DEFAULT 0;
      ->     DECLARE str VARCHAR(25) DEFAULT "";
      ->     SET year = TIMESTAMPDIFF(YEAR, dat, curDate);
      ->     SET month = TIMESTAMPDIFF(MONTH, dat, curDate);
      ->     SET month = month - (year * 12);
      ->     SET tempDate = DATE_ADD(dat, INTERVAL year YEAR);
      ->     SET tempDate = DATE_ADD(tempDate, INTERVAL month
  MONTH);
      ->     SET date = DATEDIFF(curDate, tempDate) + 1;
      ->     SET str = CONCAT(str, CAST(year AS char(2)));
      ->     SET str = CONCAT(str, "Y ");
      ->     SET str = CONCAT(str, CAST(month AS char(2)));
      ->     SET str = CONCAT(str, "M ");
  ```
  ```

```
    ->     SET str = CONCAT(str, CAST(date AS char(2)));
    ->     SET str = CONCAT(str, "D");
    ->     SET retStr = str;
    -> END]
```

Stored Procedure Query:

```
mysql> CALL 21CScalcAge ('2001-06-26', @age)]
mysql> SELECT @age]
```

OUTPUT:

```
+------------+
| @age       |
+------------+
| 20Y 10M 1D |
+------------+
```

5. Write a SQL function and stored procedure to count the total number of employees present in the employee table.

Employee table used:

```
mysql> SELECT * FROM Employee3]
```

```
+----------+------+----------+--------------+
| name     | sex  | salary   | deptName     |
+----------+------+----------+--------------+
| Amit     | M    | 30000.00 | Management   |
| Rita     | F    | 60000.00 | Headquarters |
| Jitendra | M    | 80000.00 | Headquarters |
| Riya     | F    | 40000.00 | Research     |
| Ravish   | M    | 20000.00 | Outreach     |
| Prachi   | F    | 25000.00 | Management   |
| Sita     | F    | 65000.00 | Research     |
| Utkarsh  | M    | 30000.00 | Research     |
+----------+------+----------+--------------+
```

Function:

- ```
  mysql> CREATE FUNCTION 21CStotalNoEmployees()
      -> RETURNS INT
      -> BEGIN
      ->     DECLARE s INT;
      ->     SELECT COUNT(*) FROM Employee3 INTO s;
      ->     RETURN S;
      -> END]
  ```

Function Query:

```
mysql> SELECT 21CStotalNoEmployees ()]
```

OUTPUT:
```

21CStotalNoEmployees()
8

Stored Procedure:

- ```
  mysql> CREATE PROCEDURE 21CStotalNoEmployees (OUT count INT)
      -> BEGIN
      ->     DECLARE s INT;
      ->     SELECT COUNT(*) FROM Employee3 INTO s;
      ->     SET count = s;
      -> END]
  ```

Stored Procedure Query:

```
mysql> CALL 21CStotalNoEmployees (@res)]
mysql> SELECT @res]
```

OUTPUT:

@res
8

6. Write a SQL function and stored procedure to calculate the budget of the department.

Employee table used: // from Assignment 5

```
mysql> SELECT * FROM Employee2]
```

emp_id	emp_name	salary	d_no
101	Amit	25000	D1001
102	Sunil	20000	D1002
103	Rakesh	18000	D1003
104	Ajay	16000	D1001
105	Suhail	20000	D1002
106	Arif	18000	D1004
107	Suresh	24000	D1002
108	Vijay	22000	D1003
```

Department table used: // from Assignment 5

```
mysql> SELECT * FROM Department3]
```

d_no	dept_name
D1001	IT
D1002	Sales
D1003	Marketing
D1004	HR

Function:

-
```
mysql> CREATE FUNCTION 21CScalcBudget (dept VARCHAR(30))
    -> RETURNS INT
    -> BEGIN
    ->     DECLARE deptnumber VARCHAR(5);
    ->     DECLARE budget INT DEFAULT 0;
    ->     SELECT d_no FROM Department3 WHERE dept_name = dept
INTO deptnumber;
    ->     SELECT SUM(salary) FROM Employee2 WHERE d_no =
deptnumber INTO budget;
    ->     RETURN budget;
    -> END]
```

Function Query:

```
mysql> SELECT 21CScalcBudget ('Marketing')]
```

OUTPUT:

21CScalcBudget ('Marketing')
40000

Stored Procedure:

-
```
mysql> CREATE PROCEDURE 21CScalcBudget (IN dept VARCHAR(30),
OUT budget INT)

    -> BEGIN

    ->     DECLARE deptnumber VARCHAR(5);

    ->     DECLARE sumSal INT DEFAULT 0;

    ->     SELECT d_no FROM Department3 WHERE dept_name = dept
INTO deptnumber;

    ->     SELECT SUM(salary) FROM Employee2 WHERE d_no =
deptnumber INTO sumSal;

    ->     SET budget = sumSal;

    -> END]
```

Stored Procedure Query:

```
mysql> CALL 21CScalcBudget ('IT', @budget)]
mysql> SELECT @budget]
```

OUTPUT:

```
+---------+
| @budget |
+---------+
|   41000 |
+---------+
```

7. Write a SQL function and stored procedure to print the following message:

   "Hello <name> How are you".

Function:

- ```
  mysql> CREATE FUNCTION 21CSprintMsg (name VARCHAR(50))
      -> RETURNS VARCHAR(100)
      -> BEGIN
      ->    DECLARE msg VARCHAR(100) DEFAULT "Hello ";
      ->    SET msg = CONCAT(msg, name);
      ->    SET msg = CONCAT(msg, " How are you?");
      ->    RETURN msg;
      -> END]
  ```

Function Query:

```
mysql> SELECT 21CSprintMsg ('Faizan')]
```

OUTPUT:

```
+---------------------------+
| 21CSprintMsg ('Faizan')   |
+---------------------------+
| Hello Faizan How are you? |
+---------------------------+
```

Stored Procedure:

- ```
  mysql> CREATE PROCEDURE 21CSprintMsg (IN name VARCHAR(50), OUT message VARCHAR(100))
      -> BEGIN
      ->    DECLARE msg VARCHAR(100) DEFAULT "Hello ";
      ->    SET msg = CONCAT(msg, name);
      ->    SET msg = CONCAT(msg, " How are you?");
      ->    SET message = msg;
      -> END]
  ```

Stored Procedure Query:

```
mysql> CALL 21CSprintMsg ('Faizan', @msg)]
mysql> SELECT @msg]
```

OUTPUT:

```
+--------------------------+
| @msg                     |
+--------------------------+
| Hello Faizan How are you? |
+--------------------------+
```

8. Triggers

Creating Tables:

a) Employee

```
mysql> CREATE TABLE Employees2 AS SELECT * FROM Employees;
          // similar table used in Assignment 10
mysql> SELECT * FROM Employees2;
```

```
+------+-------+--------+
| eid  | ename | salary |
+------+-------+--------+
|    1 | Ajay  |  30000 |
|    2 | Ajith |  85000 |
|    3 | Arnab |  50000 |
|    4 | Harry |  45000 |
|    5 | Ron   |  90000 |
|    6 | Josh  |  75000 |
|    7 | Ram   | 100000 |
|    8 | John  |  75000 |
|    9 | Uttam |  25000 |
+------+-------+--------+
```

b) LogTable

```
mysql> CREATE TABLE LogTable
    -> (
    -> User   VARCHAR(50),
    -> Operation VARCHAR(20),
    -> Time   VARCHAR(20),
    -> Peid   VARCHAR(5),
    -> Pename VARCHAR(50),
    -> Pesal  VARCHAR(6),
    -> Neid   VARCHAR(5),
    -> Nename VARCHAR(50),
    -> Nesal  VARCHAR(6)
    -> );
```

1. INSERT TRIGGER

- mysql> CREATE TRIGGER insertTrig

  -> AFTER INSERT ON Employees2

  -> FOR EACH ROW

  -> BEGIN

  -> INSERT INTO LogTable VALUES

  -> (user(), 'Insert', now(), '-', '-', '-', new.eid, new.ename, new.salary);

  -> END]

- mysql> INSERT INTO Employees2 VALUES

  -> (10, 'Yousuf', 69000)]

- mysql> SELECT * FROM LogTable]

OUTPUT:

User	Operation	Time	Peid	Pename	Pesal	Neid	Nename	Nesal
root@localhost	Insert	2022-04-26 18:55:22	-	-	-	10	Yousuf	69000

2. UPDATE TRIGGER

- mysql> CREATE TRIGGER updateTrig

  -> AFTER UPDATE ON Employees2

  -> FOR EACH ROW

  -> BEGIN

  -> INSERT INTO LogTable VALUES

  -> (user(), 'Update', now(), old.eid, old.ename, old.salary, new.eid, new.ename, new.salary);

  -> END]

- mysql> UPDATE Employees2

  -> SET salary = 25000

  -> WHERE eid = 3]

- mysql> SELECT * FROM LogTable]

OUTPUT:

User	Operation	Time	Peid	Pename	Pesal	Neid	Nename	Nesal
root@localhost	Insert	2022-04-26 18:55:22	-	-	-	10	Yousuf	69000
root@localhost	Update	2022-04-26 23:56:08	3	Arnab	50000	3	Arnab	25000

3. DELETE TRIGGER

- mysql> CREATE TRIGGER deleteTrig

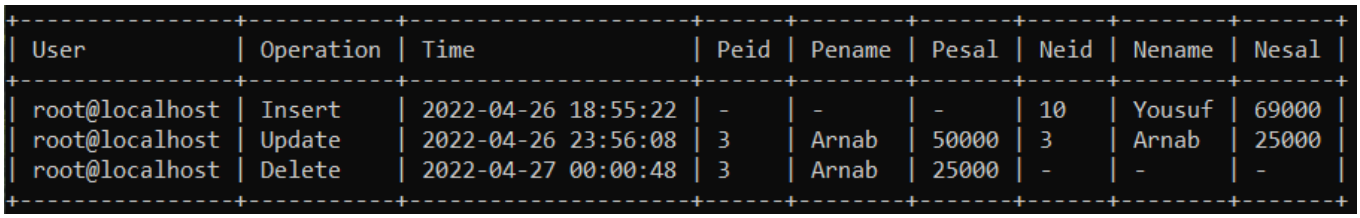
```
    -> AFTER DELETE ON Employees2
    -> FOR EACH ROW
    -> BEGIN
    ->    INSERT INTO LogTable VALUES
    ->    (user(), 'Delete', now(), old.eid, old.ename,
old.salary, '-', '-', '-');
    -> END]
```

- mysql> DELETE FROM Employees2

```
    -> WHERE eid = 3]
```

- mysql> SELECT * FROM LogTable]

OUTPUT:



User	Operation	Time	Peid	Pename	Pesal	Neid	Nename	Nesal
root@localhost	Insert	2022-04-26 18:55:22	-	-	-	10	Yousuf	69000
root@localhost	Update	2022-04-26 23:56:08	3	Arnab	50000	3	Arnab	25000
root@localhost	Delete	2022-04-27 00:00:48	3	Arnab	25000	-	-	-

4. CURSOR

Write a cursor to output salary of all employees in a string.

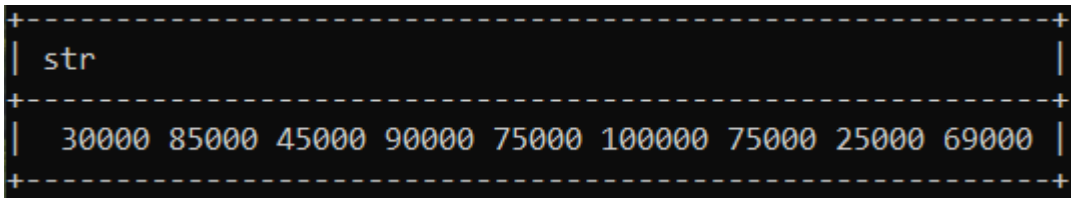
- mysql> CREATE PROCEDURE outSal (OUT s VARCHAR(15))

```
    -> BEGIN
    ->    DECLARE f INT DEFAULT 1;
    ->    DECLARE str LONGTEXT DEFAULT "";
    ->    DECLARE cur CURSOR FOR SELECT salary FROM
Employees2;
    ->    DECLARE continue HANDLER FOR NOT FOUND SET f=0;
    ->    OPEN cur;
    ->    myloop:LOOP
    ->    FETCH cur INTO s;
    ->    IF f=0 THEN
    ->         LEAVE myloop;
    ->    ELSE
    ->         SET str = CONCAT(str, " ", s);
    ->    END IF;
```

```
    ->     END LOOP;
    ->     CLOSE cur;
    ->     SELECT str;
    -> END]
```

- `mysql> CALL outSal (@s)]`

OUTPUT:



```
+-------------------------------------------------------------+
| str                                                         |
+-------------------------------------------------------------+
|  30000 85000 45000 90000 75000 100000 75000 25000 69000 |
+-------------------------------------------------------------+
```