CSLR51 – Database Management Systems Laboratory

Session: 1

1. Write SQL queries in MySQL for the following:

a. Write an SQL Query to find the year from date.

```
SELECT YEAR("2024-12-10") AS year;
```

```
mysql> select year("2024-12-10") as year;
+------+
| year |
+------+
| 2024 |
+------+
1 row in set (0.00 sec)
```

b. Check whether the date passed to the query is the date of a given format or not.

```
SELECT STR_TO_DATE("2024-07-18", "%Y-%m-%d") as valid_date;
```

```
mysql> select str_to_date('2024-07-18','%Y-%m-%d') as valid_date;
+------------+
| valid_date |
+------------+
| 2024-07-18 |
+------------+
1 row in set (0.00 sec)
```

c. Find the size of the SCHEMA/USER.

```
SELECT SUM(ROUND(((DATA_LENGTH + INDEX_LENGTH) / 1024 / 1024), 2)) AS "SIZE IN MB" FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = "sys";
```

```
mysql> SELECT SUM(ROUND(((DATA_LENGTH + INDEX_LENGTH) / 1024 / 1024), 2)) AS "SI
ZE IN MB" FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA = "sys";
+------------+
| SIZE IN MB |
+------------+
|       0.02 |
+------------+
1 row in set (0.02 sec)
```

d. Display the current time.

```
SELECT CURRENT_TIMESTAMP;
```

```
mysql> SELECT CURRENT_TIMESTAMP;
+---------------------+
| CURRENT_TIMESTAMP   |
+---------------------+
| 2024-07-26 16:31:25 |
+---------------------+
1 row in set (0.00 sec)
```

e. Given a date, retrieve the next day's date.

```
SELECT DATE_ADD("2024-07-18", INTERVAL 1 DAY);
```

```
mysql> SELECT DATE_ADD("2024-07-18", INTERVAL 1 DAY);
+----------------------------------------+
| DATE_ADD("2024-07-18", INTERVAL 1 DAY) |
+----------------------------------------+
| 2024-07-19                             |
+----------------------------------------+
1 row in set (0.00 sec)
```

f. Get database's date.

```
SELECT CURDATE();
```

```
mysql> select curdate();
+------------+
| curdate()  |
+------------+
| 2024-07-26 |
+------------+
1 row in set (0.00 sec)
```

g. Returns the default(current) database name.

```
SELECT DATABASE();
```

```
mysql> select database();
+------------+
| database() |
+------------+
| NULL       |
+------------+
1 row in set (0.00 sec)
```

h. Retrieve the current MySQL user name and host name.

```
SELECT user, authentication_string FROM mysql.user;
```

```
mysql> SELECT user, authentication_string FROM mysql.user;
+------------------+------------------------------------------------------------
-----------+
| user             | authentication_string
          |
+------------------+------------------------------------------------------------
-----------+
| debian-sys-maint | $A$005$OqAQ19x] 1q5,":4VmuaHo8Ch5q7MsFve4qGnt7J3SbKclcjQGZX
231fZH/ |
| mysql.infoschema | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNE
VERBRBEUSED |
| mysql.session    | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNE
VERBRBEUSED |
| mysql.sys        | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNE
VERBRBEUSED |
| root             |
          |
+------------------+------------------------------------------------------------
-----------+
5 rows in set (0.00 sec)
```

i. Find the string that tells the MySQL server version.

```
SELECT VERSION();
```

```
mysql> SELECT VERSION();
+-------------------------+
| VERSION()               |
+-------------------------+
| 8.0.37-0ubuntu0.22.04.3 |
+-------------------------+
1 row in set (0.00 sec)
```

j. Perform Bitwise OR, Bitwise XOR and Bitwise AND.

```
SELECT 1|2 AS ans;
```

```
mysql> SELECT 1|2 AS ans;
+-----+
| ans |
+-----+
|   3 |
+-----+
1 row in set (0.00 sec)
```

```
SELECT 1^2 AS ans;
```

```
mysql> SELECT 1^2 AS ans;
+-----+
| ans |
+-----+
|   3 |
+-----+
1 row in set (0.00 sec)
```

```
SELECT 1&2 AS ans;
```

```
mysql> SELECT 1|2 AS ans;
+-----+
| ans |
+-----+
|   3 |
+-----+
1 row in set (0.00 sec)
```

k. Find the difference between two dates and print in terms of the number of days.

```
SELECT DATEDIFF("2017-06-25", "2017-06-15");
```

```
mysql> SELECT DATEDIFF("2017-06-25", "2017-06-15");
+--------------------------------------+
| DATEDIFF("2017-06-25", "2017-06-15") |
+--------------------------------------+
|                                   10 |
+--------------------------------------+
1 row in set (0.00 sec)
```

i. Add one day to the current date.

```
SELECT DATE_ADD('2017/08/25', INTERVAL 1 DAY);
```

```
mysql> SELECT DATE_ADD('2017/08/25', INTERVAL 1 DAY);
+---------------------------------------+
| DATE_ADD('2017/08/25', INTERVAL 1 DAY) |
+---------------------------------------+
| 2017-08-26                            |
+---------------------------------------+
1 row in set, 1 warning (0.00 sec)
```

m. Add two hours and 5000 minutes to the current date and print the new date.

```
SELECT ADDTIME("12:00:00", "02:50:00") as updated;
```

```
mysql> SELECT ADDTIME("12:00:00", "02:50:00") as updated;
+----------+
| updated  |
+----------+
| 14:50:00 |
+----------+
1 row in set (0.00 sec)
```

n. Find the floor and ceil values of a floating point number. Also operate on the power, log, modulus, round off and truncate functions.

```
SELECT
FLOOR(5.76) AS floor_value,
CEIL(5.76) AS ceil_value,
```

```
    POWER(5.76, 2) AS power_value,
    LOG(5.76) AS natural_log,
    LOG10(5.76) AS base10_log,
    MOD(5.76, 3) AS modulus_value,
    ROUND(5.7654, 2) AS round_off,
    TRUNCATE(5.7654, 2) AS truncate_value;
```

```
mysql> SELECT
    ->     FLOOR(5.76) AS floor_value,
    ->     CEIL(5.76) AS ceil_value,
    ->     POWER(5.76, 2) AS power_value,
    ->     LOG(5.76) AS natural_log,
    ->     LOG10(5.76) AS base10_log,
    ->     MOD(5.76, 3) AS modulus_value,
    ->     ROUND(5.7654, 2) AS round_off,
    ->     TRUNCATE(5.7654, 2) AS truncate_value;
+-------------+------------+-------------+-----------------+--------------------
+---------------+-----------+----------------+
| floor_value | ceil_value | power_value | natural_log     | base10_log
| modulus_value | round_off | truncate_value |
+-------------+------------+-------------+-----------------+--------------------
+---------------+-----------+----------------+
|           5 |          6 |     33.1776 | 1.7509374747078 | 0.7604224834232121
|          2.76 |      5.77 |           5.76 |
+-------------+------------+-------------+-----------------+--------------------
+---------------+-----------+----------------+
1 row in set (0.00 sec)
```

o. Compare two strings and print the value ‘yes’ if they are equal, else print ‘no’.

```
    SELECT STRCMP('Mahesh', 'Mahesh') AS 'Cmp_Value';
```

```
mysql> SELECT STRCMP('Mahesh', 'Mahesh') AS 'Cmp_Value';
+-----------+
| Cmp_Value |
+-----------+
|         0 |
+-----------+
1 row in set (0.00 sec)
```

p. Simulate the “IF... ELSE” construct in MySQL for a mark and grade setup.

```
    SELECT IF(500>100, "YES", "NO");
```

```
mysql> SELECT IF(500>100, "YES", "NO");
+--------------------------+
| IF(500>100, "YES", "NO") |
+--------------------------+
| YES                      |
+--------------------------+
1 row in set (0.00 sec)
```

q. Use IFNULL to check whether a mathematical expression gives a NULL value or not.

```
SELECT IFNULL(1 / NULL, 'Expression is NULL') AS result;
```