# Assignment No: 6

***Aim:*** Execute the aggregate functions like count, sum, avg etc. on the suitable database. Make use of built in functions according to the need of the database chosen. Retrieve the data from the database based on time and date functions like now (), date (), day (), time () etc. Use group by and having clauses.

***Objective:***

- To understand and implement various types of function inMYSQL.
- To learn the concept of group functions

**NUMBER FUNCTION*:***

Abs(n) :Select abs(-15) from dual;

Exp(n): Select exp(4) from dual;

Power(m,n): Select power(4,2) from dual;

Mod(m,n): Select mod(10,3) fromdual;

Round(m,n): Select round(100.256,2) from dual;

Trunc(m,n): ;Select trunc(100.256,2) from dual;

Sqrt(m,n);Select sqrt(16) from dual;

**Aggregate Functions:**

**1. Count:** COUNT following by a column name returns the count of tuple in that column. If DISTINCT keyword is used then it will return only the count of unique tuple in the column. Otherwise, it will return count of all the tuples (including duplicates) count (*) indicates all the tuples of thecolumn.

***Syntax:*** COUNT (Column name)

***Example:*** SELECT COUNT (Sal) FROM emp;

**2. SUM:** SUM followed by a column name returns the sum of all the values in that column. ***Syntax:*** SUM (Columnname)

***Example:*** SELECT SUM (Sal) From emp;

**3. AVG:** AVG followed by a column name returns the average value of thatcolumn values. ***Syntax:*** AVG (n1,n2...)

***Example:*** Select AVG (10, 15, 30) FROM DUAL;

**4. MAX:** MAX followed by a column name returns the maximum value ofthat column. ***Syntax:*** MAX (Columnname)

***Example:*** SELECT MAX (Sal) FROM emp;

mysql> select deptno, max(sal) from emp group by deptno;

DEPTNO MAX (SAL)

---- ------

10 5000
20 3000

30 2850

mysql> select deptno, max (sal) from emp group by deptno havingmax(sal)<3000;

DEPTNO MAX(SAL)

---- ------
30 2850

**5. MIN:** MIN followed by column name returns the minimum value of thatcolumn.
***Syntax:*** MIN (Column name)

***Example:*** SELECT MIN (Sal) FROM emp;

mysql> select deptno,min(sal) from emp group by deptno having min(sal)>1000;

DEPTNO MIN (SAL)
---- ------
10 1300

**DATE FUNCTIONS:**

CURDATE()

Returns the current date as a value in 'YYYY-MM-DD' or YYYYMMDD format, depending on whether the function is used in a string or numeric context.

```
mysql> SELECT CURDATE();
------------------
------------------
------------------
             ---+                                  +
| CURDATE()                  |
------------------
------------------
------------------                                  +
```

```
            ---+
| 1997-12-15 |
------------------
------------------
------------------
            ---+                                +
```

1 row in set (0.00 sec)

CURTIME()

Returns the current time as a value in 'HH:MM:SS' or HHMMSS format, depending on whether the function is used in a string or numeric context. The value is expressed in the current time zone.

```
mysql> SELECT CURTIME();

                +                                +
| CURTIME() |

                +                                +
| 23:50:26 |
                                                 +
```

1 row in set (0.00 sec)

DATE(expr)

Extracts the date part of the date or datetime expression expr.

```
mysql> SELECT DATE('2003-12-31 01:02:03');
-------------
-------------
-------------
-------------
 ---------+                                +
 | DATE('2003-12-31 01:02:03') |
-------------
-------------
-------------
-------------
 ---------+                                +
 | 2003-12-31 |
------------                                +
```

```
----------
----------
----------
 ..........+
1 row in set (0.00 sec)



DAY(date)
DAY() is a synonym for DAYOFMONTH().
```

DAYNAME(date)

Returns the name of the weekday for date.

```
mysql> SELECT DAYNAME('1998-02-05');
---------------
---------------
---------------
---------------
            -+                                          +
  |            DAYNAME('1998-02-05')                    |
---------------
---------------
---------------
---------------
            -+                                          +
  |                Thursday                        |
---------------
---------------
---------------
---------------
            -+                                          +
1 row in set (0.00 sec)
```

HOUR(time)

Returns the hour for the time. The range of the return value is 0 to 23 for time-of-day values. However, the range of TIME values actually is much larger, so HOUR can return values greater than 23.

```
mysql>SELECTHOUR('10:05:03');
```

```
-----------
-----------
-----------
    ---------+                                    +

| HOUR('10:05:03')                  |
-----------
-----------
-----------
    ---------+                                    +

|                  10                    |

+-----------------------                             +
```

1 row in set (0.00 sec)

LAST_DAY(date)

Takes a date or datetime value and returns the corresponding value for the last day of the month. Returns NULL if the argument is invalid.

```
mysql> SELECT LAST_DAY('2003-02-05');

---------------
---------------
---------------
---------------
          -+                                       +

  |            LAST_DAY('2003-02-05')             |
---------------
---------------
---------------
---------------
          -+                                       +

  |               2003-02-28                   |
---------------
---------------
---------------
---------------
          -+                                       +
```

1 row in set (0.00 sec)

MINUTE(time)

Returns the minute for time, in the range 0 to 59.

```
mysql> SELECT MINUTE('98-02-03 10:05:03');
---------- ---------- ---------- ---------- ----------
        --+                                           +
  |           MINUTE('98-02-03 10:05:03')              |
---------- ---------- ---------- ---------- ----------
        --+                                           +
  |                    5                      |
---------- ---------- ---------- ---------- ----------
        --+                                           +
```

1 row in set (0.00 sec)

MONTH(date)

Returns the month for date, in the range 0 to 12.

```
mysql> SELECT MONTH('1998-02-03')
---------- ---------- ----------
 ----------+                                          +
|                MONTH('1998-02-03')                  |
---------- ---------- ----------
 ----------+                                          +
|                    2                      |
+                                                     +
```

1 row in set (0.00 sec)

MONTHNAME(date)

Returns the full name of the month for date.

```
mysql> SELECT MONTHNAME('1998-02-05');
---------
---------
---------
---------
     -----+                                        +
  |           MONTHNAME('1998-02-05')                |
---------
---------
---------
---------
     -----+                                        +
  |              February                       |
---------
---------
---------
---------
     -----+                                        +
```

1 row in set (0.00 sec)

NOW()

Returns the current date and time as a value in 'YYYY-MM-DD HH:MM:SS' or YYYYMMDDHHMMSS format, depending on whether the function is used in a string or numeric context. The value is expressed in the current time zone.

```
mysql> SELECT NOW();
---------------
---------------
---------------
---------------
          -+                                        +
 |                  NOW()                    |
---------------
---------------
---------------
---------------
          -+                                        +
```

```
|              1997-12-15 23:50:26              |
---------------
---------------
---------------
---------------
          -+                                   +
```

1 row in set (0.00 sec)

**GROUP BY:** This query is used to group to all the records in a relation together for each and every value of a specific key(s) and then display them for a selected set of fields the relation.

***Syntax:*** SELECT <set of fields> FROM <relation_name>

GROUP BY <field_name>;

***Example:***SELECT EMPNO, SUM (SALARY) FROM EMP GROUP BY EMPNO;

**GROUP BY-HAVING :**The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions. The HAVING clause must follow the GROUP BY clause in a query and must also precede the ORDER BY clause if used.

***Syntax:*** SELECT column_name, aggregate_function(column_name) FROM table_name
WHERE column_name operator value

GROUP BY column_name

HAVING aggregate_function(column_name) operator value;

***Example*** : SELECT empno,SUM(SALARY) FROM emp,dept

WHERE emp.deptno =20 GROUP BY empno;

**ORDER BY:**This query is used to display a selected set of fields from a relation in an ordered manner base on some field.

***Syntax:*** SELECT <set of fields> FROM<relation_name>

ORDER BY<field_name>;

***Example:*** SQL> SELECT empno, ename, job FROM emp ORDER BY job;

**LAB PRACTICE ASSIGNMENT:**

**Consider the following table structure for this assignment: CUSTOMER(Cust_id, C_name, City) BRANCH(Branch_id, bname, City)**

**DEPOSIT(Acc_no , Cust_id, Amount, Branch_id, Open_date)**
**BORROW(Loan_no, Cust_id, Branch_id, Amount)**

**Perform the following queries on the above table:**

**1)** List totalloan.
**2)** List totaldeposit.
**3)** List maximum deposit of customers living inMumbai.
**4)** Count total number of branchcities.
**5)** List branch_id and branch wisedeposit.
**6)** List the branches having sum of deposit more than4000.
**7)** List the names of customers having minimumdeposit.
**8)** Count the number of depositors living in'nagpur'.
**9)** Find the maximum deposit of the Akurdibranch.
**10)** Find out number of customers living inPune.
**11)** Find out the customers who are not living in Pune orMumbai.
**12)** List out Cust_id and C_name in descending order of theirC_name.
**13)** Display the number of depositors in branchwise.
**14)** Find out the branch which has notborrowers.
**15)** How many customers have opened deposit after'01-01-2016'

**Conclusion:-**

In this assignment, we have learned and executed Aggregate and date functions of MYSQL.

**Name – Aditya Somani Roll No. – T1851061 PRN No. : 71901204L**

## OUTPUT –

### 1) List total Loan

```
MySQL>SELECT SUM(AMOUNT) FROM BORROW;
+-------------+
| SUM(AMOUNT) |
+-------------+
| 265000 |
+-------------+
1 row in set (0.002 sec)
```

### 2) List total deposit

```
MySQL>SELECT SUM(AMOUNT) FROM DEPOSIT;
+-------------+
| SUM(AMOUNT) |
+-------------+
| 72300 |
+-------------+
1 row in set (0.001 sec)
```

### 3) List Maximum deposit of Customer living in Mumbai.

```
MySQL>SELECT MAX(ACCOUNT_BALANCE) FROM CUSTOMER WHERE ADDRESS = 'Mumbai';
+----------------------+
| MAX(ACCOUNT_BALANCE) |
+----------------------+
| 25000 |
+----------------------+
1 row in set (0.002 sec)
```

**4) Count total number of branch cities.**

```
MySQL> SELECT COUNT(BRANCH_NAME) FROM BRANCH;
+--------------------+
| COUNT(BRANCH_NAME) |
+--------------------+
| 8 |
+--------------------+
1 row in set (0.002 sec)
```

**5) List branch_id and branch wise deposit.**

```
MySQL> SELECT * FROM DEPOSIT GROUP BY BRANCH_ID;
+------------+---------+--------+-----------+------------+
| ACC_NUMBER | CUST_ID | AMOUNT | BRANCH_ID | OPEN_DATE |
+------------+---------+--------+-----------+------------+
| 11111 | Shreyas | 1000 | 1 | 2016-05-11 |
| 01111 | Piyush | 10000 | 2 | 2019-07-01 |
| 01115 | Ankit | 10000 | 3 | 2019-07-17 |
| 01315 | Aditya | 5000 | 4 | 2017-07-17 |
| 11315 | Patil | 500 | 5 | 2019-07-17 |
| 11415 | Yash | 6900 | 6 | 2019-07-20 |
| 24356 | Nilesh | 7900 | 7 | 2018-03-14 |
| 24355 | Shubham | 20000 | 8 | 2018-11-14 |
+------------+---------+--------+-----------+------------+
8 rows in set (0.002 sec)
```

**6) List branches having sum deposit more than 4000.**

MySQL> SELECT BRANCH_NAME FROM BRANCH B, DEPOSIT D WHERE D.AMOUNT> 4000 AND D.BRANCH_ID=B.ID;

+-------------+

| BRANCH_NAME |

+-------------+

| AKURDI STN |

Assignment 6 Roll No : T1851138

By Shreyas Sharad Patil 2

| NAGPUR CIT |

| NIGDI MAIN |

| NASHIK ROA |

| MUMBAI MAI |

| JALGAON MA |

| AKURDI STN |

+-------------+

7 rows in set (0.002 sec)

**7) List names of customer having minimum deposit.**

MySQL> SELECT CUST_ID FROM DEPOSIT WHERE AMOUNT = (SELECT MIN(AMOUNT) FROM DEPOSIT);

+---------+

| CUST_ID |

+---------+

| Patil |

+---------+

1 row in set (0.001 sec)

**8) Count number of depositors living in 'Nagpur'**

```
MySQL> SELECT COUNT(CUST_ID) FROM DEPOSIT WHERE BRANCH_ID = 3;
+----------------+
| COUNT(CUST_ID) |
+----------------+
| 1 |
+----------------+
1 row in set (0.002 sec)
```

**9) Find number of depositors in Akurdi branch.**

```
MySQL> SELECT COUNT(CUST_ID) FROM DEPOSIT WHERE BRANCH_ID = 2;
+----------------+
| COUNT(CUST_ID) |
+----------------+
| 2 |
+----------------+
```

**10) Find out number of customer living in Pune.**

```
MySQL>SELECT COUNT(*) FROM CUSTOMER WHERE ADDRESS = 'PUNE';
+----------+
| COUNT(*) |
+----------+
| 3 |
+----------+
1 row in set (0.001 sec)
```

**11) Find out customer who are not living in Pune or Mumbai.**

```
MySQL>SELECT * FROM CUSTOMER WHERE ADDRESS != 'PUNE' AND ADDRESS != 'MUMBAI';
+---------+-----------+----------------+----------------+---------------------+
| CUST_ID | CUST_NAME | ADDRESS | ACCOUNT_BALANCE | ACCOUNT_OPENING_DATE |
+---------+-----------+----------------+----------------+---------------------+
| 2 | Anil | Akurdi | 500000 | 2019-07-30 |
| 3 | Piyush | Nagpur | 10000 | 2019-07-22 |
```

| 4 | Aditya | Nigdi | 200000 | 2019-07-19 |

| 7 | Yash | Pimpri Chinchwa | 90000 | 2019-05-29 |

| 8 | Shubham | Jalgaon | 30000 | 2019-01-29 |

| 10 | Nilesh | Nashik | 1000 | 2019-09-29 |

+---------+-----------+----------------+----------------+---------------------+

6 rows in set (0.001 sec)

**12) List cust_id and name in descending order of their name.**

MySQL> SELECT CUST_ID, CUST_NAME FROM CUSTOMER ORDER BY CUST_NAME DESC;

+---------+-----------+

| CUST_ID | CUST_NAME |

+---------+-----------+

Assignment 6 Roll No : T1851138

By Shreyas Sharad Patil 3

| 7 | Yash |

| 9 | Varun |

| 8 | Shubham |

| 1 | Shreyas |

| 3 | Piyush |

| 6 | Patil |

| 10 | Nilesh |

| 5 | Ankit |

| 2 | Anil |

| 4 | Aditya |

+---------+-----------+

**13) List the number of depositors branch wise.**

MySQL> SELECT COUNT(*) FROM DEPOSIT GROUP BY BRANCH_ID;

+----------+

| COUNT(*) |

+----------+

| 1 |

| 2 |

| 1 |

| 1 |

| 1 |

| 1 |

| 1 |

| 1 |

+----------+

8 rows in set (0.002 sec)

**14) Find out the branch which has not borrowers.**

MySQL> SELECT BRANCH_NAME FROM BRANCH WHERE ID NOT IN (SELECT BRANCH_ID FROM BORROW);

+-------------+

| BRANCH_NAME |

+-------------+

| NIGDI MAIN |

| PCMC MAIN |

| NASHIK ROA |

| JALGAON MA |

| MUMBAI MAI |

+-------------+

**15) How many customers opened deposit after ’01-01-2016’**

MySQL> SELECT COUNT(*) FROM DEPOSIT WHERE OPEN_DATE > '01-01-2016';

+----------+

| COUNT(*) |

+----------+

| 9 |

+----------+

1 row in set, 1 warning (0.002 sec)