

**Title:** Date and Time data types in SQL**Objective:**

- To be familiar with date and time data types and its implementation

**Theory:**

MySQL comes with the following data types for storing a date or a date/time value in the database:

**DATE** - format YYYY-MM-DD

**DATETIME** - format: YYYY-MM-DD HH:MI:SS

**TIMESTAMP** - format: YYYY-MM-DD HH:MI:SS

**YEAR** - format YYYY or YY

**TIME** A time. Format: hh:mm:ss.

**Commonly Used Date/Time Functions****➤ CURDATE()**

This function is used to get the current date and time.

**For example,**

```
SELECT CURDATE();
```

Here, the function returns the current date and time.

**➤ DATEDIFF(date\_part, start\_date, end\_date)**

This function is used to determine the difference between two dates.

For example,

```
SELECT DATEDIFF(month, '2020-12-31', '2022-01-01 ');  
-- output: 13
```

Here, the function returns the difference between the two dates in months.

The output 13 indicates that there's a difference of 13 months between 2020-12-31 and 2022-01-01 . we can similarly find the difference in terms of years, days etc.

**Note:** In MySQL DATEDIFF() function has only two parameters.

```
DATEDIFF( end_date, start_date)
```

This returns the date difference in terms of number of days.

**➤ TIMEDIFF()**

The **TIMEDIFF()** function returns the difference between two time/datetime expressions.

Note: time1 and time2 should be in the same format, and the calculation is time1 - time2.

**Syntax:**

**TIMEDIFF(time1, time2)**

### Problem 1:

1. Create table Orders with following columns and appropriate data types

**Orders(order\_id,product\_name,price,quantity,order\_date,delivery\_date)**

```
CREATE TABLE Orders (
    order_id INT PRIMARY KEY,
    product_name VARCHAR(50),
    price DECIMAL(10, 2),
    quantity INT,
    order_date
    DATE,
    delivery_date DATE
);
```

2. Insert 8 rows in orders table

```
INSERT INTO Orders
VALUES (1, 'T-shirt', 25.99, 2, '2023-07-15', '2023-07-25');
```

```
INSERT INTO Orders
VALUES (2, 'Jeans', 49.95, 1, '2023-07-17', '2023-07-20');
```

```
INSERT INTO Orders
VALUES (3, 'Shoes', 69.50, 1, '2023-07-20', '2023-07-30');
```

```
INSERT INTO Orders
VALUES (4, 'Sunglasses', 12.75, 3, '2023-07-22', '2023-07-28');
```

```
INSERT INTO Orders
VALUES (5, 'Backpack', 34.99, 2, '2023-07-25', '2023-07-29');
```

```
INSERT INTO Orders
VALUES (6, 'Headphones', 59.99, 1, '2023-07-29', '2023-08-05');
```

```
INSERT INTO Orders
```

```
VALUES (7, 'Smartphone', 299.99, 2, '2023-07-29', '2023-11-01');
```

```
INSERT INTO Orders
```

```
VALUES (8, 'Laptop', 799.95, 1, '2023-07-29', '2025-08-01');
```

**3) Retrieve all orders placed on a 2023-07-15**

```
SELECT *  
FROM Orders  
WHERE order_date = '2023-07-15';
```

**4) Find the number of days that required to delivered shoes**

```
SELECT DATEDIFF(delivery_date, order_date) AS delivery_time  
FROM Orders  
where product_name='shoes';
```

**5) Find all the orders that is received from '2023-07-15' to '2023-07-25'**

```
SELECT *  
FROM Orders  
WHERE order_date BETWEEN '2023-07-15' AND '2023-07-25';
```

**6) find all the orders that is received today**

```
SELECT *  
FROM Orders  
WHERE order_date = CURDATE();
```

**7) Calculate the average number of days it takes to deliver a orders**

```
SELECT AVG(DATEDIFF(delivery_date, order_date)) AS avg_delivery_time FROM  
Orders;
```

**Note:**

Here, in DATEDIFF() function we have passed two parameters that is

**DATEDIFF( date1, date2)**

This will returns date difference in terms of number of days (date1-date2)

This is for if we run query on **MySQL DBMS**.

But sometimes it is necessary to find out date difference in terms of number of month, week, year, quarter etc. In such case three parameters need to passed three parameters.

## Syntax

```
DATEDIFF(interval, date1, date2)
```

## Parameter Values

Parameter	Description
<i>interval</i>	Required. The part to return. Can be one of the following values:
	<ul style="list-style-type: none"> <li>• year, yyyy, yy = Year</li> <li>• quarter, qq, q = Quarter</li> <li>• month, mm, m = month</li> <li>• dayofyear = Day of the year</li> <li>• day, dy, y = Day</li> <li>• week, ww, wk = Week</li> <li>• weekday, dw, w = Weekday</li> <li>• hour, hh = hour</li> <li>• minute, mi, n = Minute</li> <li>• second, ss, s = Second</li> <li>• millisecond, ms = Millisecond</li> </ul>

*date1, date2* Required. The two dates to calculate the difference between

### Note:

- DATEDIFF() function with two parameters are supported in MYSQL DBMS.
- DATEDIFF() function with three parameters are supported in MS SQL Server DBMS

If you want test query in different DBMS ,you can follow this link

<http://www.sqlfiddle.com>

### 8) Find the number of months required to deliver smartphone

```
SELECT DATEDIFF(delivery_date, order_date) AS delivery_time FROM
Orders
where product_name='smartphone';
```

If we run this query in MySQL DBMS.

DATEDIFF() function will returns 95 for this query by considering above relations.

delivery_time
95

**DATEDIFF() with three parameters are not supported in MySQL DBMS.**

**This query can be re-written as follows by passing three parameters in MS SQL server DBMS.**

```
SELECT DATEDIFF(month, order_date,delivery_date) AS delivery_time
FROM Orders
where product_name='smartphone';
```

**Note: you can write DATEDIFF() function with three parameters in exam.**

**9) Find the number of weeks required to deliver smartphone**

```
SELECT DATEDIFF(week, order_date,delivery_date) AS delivery_time
FROM Orders
where product_name='smartphone';
```

**10) Find the products that required more than 2 month to delivered**

```
SELECT product_name
FROM orders
WHERE DATEDIFF(month,order_date,delivery_date)>2;
```

**11) Find the products that required more than 3 weeks to delivered**

```
SELECT product_name
FROM orders
WHERE DATEDIFF(week,order_date,delivery_date)>3;
```

**12. Find the products that required more than 1 years to delivered.**

```
SELECT product_name
FROM orders
WHERE DATEDIFF(year,order_date,delivery_date)>1;
```

**Problem 2:**

**Write SQL statements for the following queries in reference to relation Emp\_time provided.**

Eid	Name	Start_time	End_time
E101	Hari	10:15	18:00
E102	Malati	8:00	15:30
E103	Kalyan	9:30	17:00

**i) create the table Eid as primary key and insert the values provided**

```
CREATE TABLE Emp_time (
Eid VARCHAR(10) PRIMARY KEY,
Name VARCHAR(30),
Start_time TIME,
End_time TIME
);
```

```
INSERT INTO Emp_time VALUES ('E101','Hari', '10:15:00', '18:00:00');
INSERT INTO Emp_time VALUES ('E102','Malati', '8:00:00', '15:30:00');
INSERT INTO Emp_time VALUES ('E103', 'Kalyan', '9:30:00', '17:00:00');
```

**ii) Select all employees and their total working hours**

```
SELECT Eid, Name, TIMEDIFF(End_time, Start_time) AS Total_Working_Hours
FROM Emp_time;
```

**iii) Find the Employee information as per their least working hours.**

```
SELECT Eid, Name, TIMEDIFF(End_time, Start_time) AS Work_Duration
FROM Emp_time
ORDER BY Work_Duration ASC;
```

**iv) Find the Employee information as per their long working hours.**

```
SELECT Eid, Name, TIMEDIFF(End_time, Start_time) AS Work_Duration
```

```
FROM Emp_time  
ORDER BY Work_Duration DESC;
```

- v) **Select the employee name who works long hours among all the employees**

```
SELECT Eid, Name, TIMEDIFF(End_time, Start_time) AS Work_Duration  
FROM Emp_time  
ORDER BY Work_Duration DESC LIMIT  
1;
```

- vi) **find employees who worked more than 7 hours**

```
SELECT Name  
FROM Emp_time  
WHERE TIMEDIFF(End_time, Start_time) > '07:00:00';
```

- vii) **Display the name of the employee whose name start from letter 'M' and who work more than seven hours**

```
SELECT Name  
FROM Emp_time  
WHERE Name LIKE 'M%' AND TIMEDIFF(End_time, Start_time) > '07:00:00';
```

- viii) **Delete the entire content of the table so that new records can be inserted.** TRUNCATE TABLE Emp\_time;

**Discussion:** (*This portion is left for student*)

**Conclusion:** (*This portion is left for student*)

\*\*\*\*\*THE END\*\*\*\*\*