

Please do not copy without permission. © ALX 2024.



Introduction

Datetime functions are **built-in functions** that operate on **date and time values**. They allow for various manipulations on **dates**, **times**, or **timestamps** stored in the database.

Benefits of SQL datetime functions

01. Data manipulation

Transform and format date and time values.

04. Data consistency and integrity

Validate and format date and time values.

02. Querying and filtering

Filter data based on specific time conditions.

05. Application development

Efficiently handle time-related operations.

O3. Reporting and analysis

Generate reports and perform time-based analysis.

06. Collaboration and integration

Work effectively with others and integrate SQL with other tools.



Functions specific to MySQL—CURRENT_DATE()

The **CURRENT_DATE()** function is used to **retrieve the current date without the time component**.

Query

Output

Current_date

2023-06-26

SELECT

CURRENT_DATE() AS Current_date;

There are functions that are specific to a particular database management system (DBMS), such as MySQL. These functions are provided by the DBMS to offer additional functionality or cater to specific features of that particular database system.





Functions specific to MySQL - NOW()

The NOW() function is used to **retrieve the current date and time from the system.** It returns a datetime value representing the current timestamp.

Query

Output

```
SELECT
NOW() AS Current_timestamp;
```

```
Current_timestamp
```

2023-06-26 14:30:45



Functions specific to MySQL—CURRENT_TIMESTAMP()

The CURRENT_TIMESTAMP() function is used to retrieve the current date and time.

Query

Output

Current_timestamp

2023-06-26 14:30:45

SELECT

CURRENT_TIMESTAMP() AS Current_timestamp;

There is no functional difference between NOW() and CURRENT_TIMESTAMP(). Both functions are used to retrieve the current date and time and can therefore be used interchangeably.





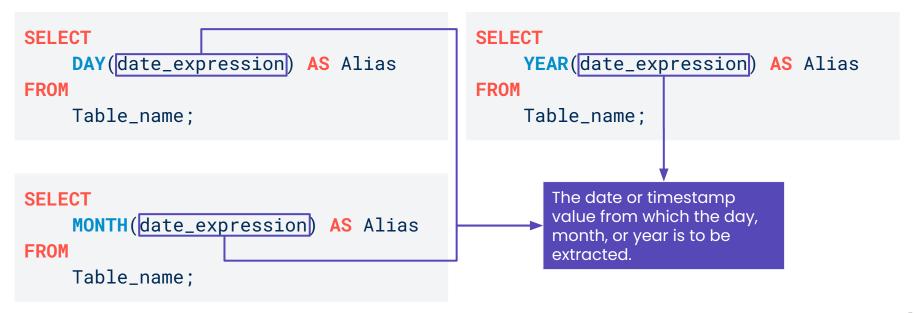
Data overview

To explain datetime functions that are universal in SQL, we will use the Water_consumption_data_sa table that represents the amount of water (liters) consumed within South African cities, as well as the date, time, and timestamp of their recording.

City	Water_consumption	Date_recorded	Time_recorded	Timestamp_recorded
Johannesburg	150000	2022-08-10	08:45:00	2023-06-26 08:45:00
Cape Town	100000	2022-07-25	09:15:00	2023-06-26 09:15:00
Durban	120000	2022-09-02	10:30:00	2023-06-26 10:30:00
Pretoria	140000	2022-08-15	11:45:00	2023-06-26 11:45:00
Port Elizabeth	80000	2022-09-01	12:15:00	2023-06-26 12:15:00
Bloemfontein	60000	2022-07-30	13:30:00	2023-06-26 13:30:00

Universal SQL functions - DAY(), MONTH(), and YEAR()

When working with a date or timestamp value, the DAY() function is used to extract the day portion, the MONTH() function is used to extract the month portion, and the YEAR() function is used to extract the year portion.





Universal SQL functions - DAY()

For instance, if we want to retrieve the day component from all entries in the **Date_recorded** column, we can utilize the **DAY()** function.

Query

```
SELECT
    City,
    Date_recorded,
    DAY(Date_recorded) AS Day
FROM
    Water_sources_sa_2022;
```

City	Date_recorded	Day
Johannesburg	2022-08-10	10
Cape Town	2022-07-25	25
Durban	2022-09-02	2
Pretoria	2022-08-15	15
Port Elizabeth	2022-09-01	1
Bloemfontein	2022-07-30	30



Universal SQL functions - MONTH()

If we aim to retrieve the month component from each entry in the **Date_recorded** column, we can utilize the **MONTH()** function.

Query

```
SELECT
    City,
    Date_recorded,
    MONTH(Date_recorded) AS Month
FROM
    Water_sources_sa_2022;
```

City	Date_recorded	Month
Johannesburg	2022-08-10	8
Cape Town	2022-07-25	7
Durban	2022-09-02	9
Pretoria	2022-08-15	8
Port Elizabeth	2022-09-01	9
Bloemfontein	2022-07-30	7



Universal SQL functions - YEAR()

To extract the year portion from all entries of the **Date_recorded** column, we use the **YEAR()** column.

Query

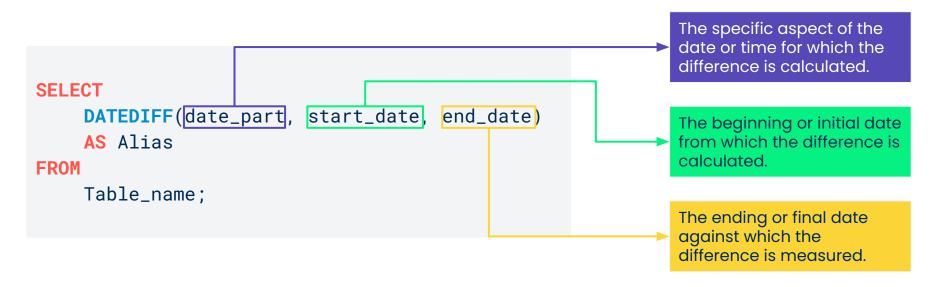
```
SELECT
    City,
    Date_recorded,
    YEAR(Date_recorded) AS Year
FROM
    Water_sources_sa_2022;
```

City	Date_recorded	Year
Johannesburg	2022-08-10	2022
Cape Town	2022-07-25	2022
Durban	2022-09-02	2022
Pretoria	2022-08-15	2022
Port Elizabeth	2022-09-01	2022
Bloemfontein	2022-07-30	2022



Universal SQL functions - DATEDIFF()

The **DATEDIFF()** function is used to **calculate the difference between two dates.** It takes three parameters: the **part of the date** for which to calculate the difference (day, month, or year), the **start date**, and the **end date**.





Universal SQL functions - DATEDIFF()

By utilizing the values in the **Date_recorded** column as the starting dates, we can determine the duration that has passed from the most recent recording until the present date.

Query

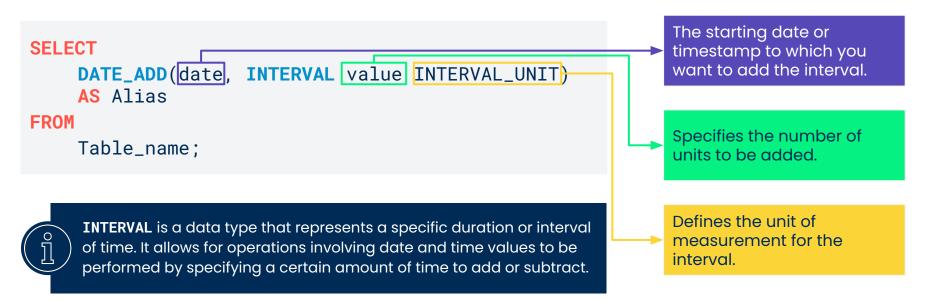
```
SELECT
    City,
    Date_recorded,
    DATEDIFF(day, Date_recorded, NOW()) AS
    Days_elapsed
FROM
    Water_sources_sa_2022;
```

City	Date_recorded	Days_elapsed
Johannesburg	2022-08-10	320
Cape Town	2022-07-25	336
Durban	2022-09-02	297
Pretoria	2022-08-15	315
Port Elizabeth	2022-09-01	298
Bloemfontein	2022-07-30	331



Universal SQL functions - DATE_ADD()

The **DATE_ADD()** function is used to **add a specified interval to a date or datetime value**. It takes three parameters: a **date or datetime** to which the interval will be added, a **value** representing the interval you want to add, and an **interval unit**, which can be DAY, MONTH, or YEAR.





Universal SQL functions - DATE_ADD()

Suppose we need to schedule a review exactly seven days after a recording has been made. In this case, we can utilize the **DATE_ADD()** function to calculate and set the review date for each entry accordingly.

Query

```
SELECT
    City,
    Date_recorded,
    DATE_ADD(Date_recorded, INTERVAL 7 DAY)
    AS Next_review
FROM
    Water_sources_sa_2022;
```

City	Date_recorded	Next_review
Johannesburg	2022-08-10	2022-08-17
Cape Town	2022-07-25	2022-08-01
Durban	2022-09-02	2022-09-09
Pretoria	2022-08-15	2022-08-22
Port Elizabeth	2022-09-01	2022-09-08
Bloemfontein	2022-07-30	2022-08-06