SQL Standard

SQL DATE_TRUNC

7 months ago • by John Otieno

In SQL, the “DATE_TRUNC” is a powerful and very useful function that allows us to truncate or round off the date and time values to a specific level of precision. A common use case of this function is when you need to group or aggregate the data based on a specific part of a date. For example: a month, day, year, etc.

In this tutorial, we will dive deep in the functionality and usage of this function and learn how we can use it in SQL databases.

NOTE: You may not have access to this function depending on your database engine. However, it is supported by PostgreSQL and Oracle.

Databases such as MySQL do not support this function. You can check our tutorial on the extract() function as an alternative.

Function Syntax:

The following shows the syntax of the function in SQL:

```
DATE_TRUNC(unit, source_date)
```

The function accepts two main arguments:

1. Unit – This specifies the unit of time to which we wish to truncate. This can be one of the following values:
    1. microsecond
    2. millisecond
    3. second
    4. minute
    5. hour
    6. day
    7. week
    8. month
    9. quarter

10. year

2. source_date – This refers to the date or timestamp value that we wish to truncate.

Let us look at some example usage of this function.

Example 1: Basic Usage

To better demonstrate it, let us take a look at its practical usage.

Suppose we have a table called “sales” with a “transaction_date” column. What if we want to calculate the total sales for each month?

We can use the DATE_TRUNC to truncate the “transaction_date” at the beginning of the month and then group by the truncated date.

```
SELECT DATE_TRUNC('month', transaction_date) AS truncated_date,
SUM(sales_amount) AS total_sales
FROM sales
GROUP BY truncated_date
ORDER BY truncated_date;
```

In the given example, the DATE_TRUNC() function truncates the “transaction_date” column at the beginning of the month.

We then use the “sum” function to aggregate the sales for each truncated month.

Lastly, we group the data by the truncated date and order them with the same criteria.

Example 2: Truncate to Millisecond

In some cases, you might need a high precision. Consider the table called “sensor_data” with a “timestamp_ms” column that represents the sensor data with a millisecond precision.

We can truncate the timestamp to the millisecond level as shown in the following example:

```
SELECT DATE_TRUNC('millisecond', timestamp_ms) AS truncated_time,
sensor_value
FROM sensor_data;
```

Truncating to a smaller unit provides a higher level of precision for the data.

Conclusion

In this example, we demonstrate how to use the DATE_TRUNC() function in SQL to truncate the date values to specific precision levels.