Funções no MySQL, PostgreSQL, Microsoft SQL Server e alternativas para o Databricks:

Funções de manipulação de Data e Hora:

* ILIKE
* DATE\_PART
* DATE\_TRUNC():
* EXTRACT():

PS:

YEAR: Extrai o ano.

MONTH: Extrai o mês.

DAY: Extrai o dia.

HOUR: Extrai a hora.

MINUTE: Extrai os minutos.

SECOND: Extrai os segundos.

QUARTER: Extrai o trimestre.

WEEK: Extrai a semana do ano.

WEEKDAY: Retorna o dia da semana (1 = domi, 2 = seg, etc.).

DAYOFYEAR: Retorna o dia do ano

DATE\_PART: Extrai partes de uma data em diferentes bancos de dados:

**PostgreSQL**

SELECT DATE\_PART('dow', NOW()); -- Retorna o dia da semana (0 = domingo, 1 = segunda)

Exemplo: SELECT DATE\_PART('year', '2024-10-09'::date) AS ano

**MySQL:**

O MySQL não possui a função DATE\_PART(), usar a função EXTRACT() para obter o mesmo resultado.

SELECT HOUR(NOW()); -- Retorna a hora

Exemplo: SELECT YEAR('2024-10-09') AS ano

**Microsoft SQL Server:**

SELECT DATEPART(MONTH, GETDATE()); -- Retorna o mês

**Alternativa Databricks:**

df\_data = df.select( year(df['data\_col']).alias('ano'), # Retorna o ano)

*No Databricks Runtime 13.0 não é necessário importar explicitamente as funções do módulo pyspark.sql.functions para operações comuns, pois as funções como year(), month(), etc., já estão disponíveis automaticamente.*

DATE\_TRUNC():

Permite truncar uma data para a parte especificada (por exemplo, ano, mês, dia, hora).

**PostgreSQL**

-- Trunca para o início do ano

SELECT DATE\_TRUNC('year', NOW()); -- Retorna: '2024-01-01 00:00:00'

-- Trunca para o início do mês

SELECT DATE\_TRUNC('month', NOW()); -- Retorna: '2024-10-01 00:00:00'

-- Trunca para o início do dia

SELECT DATE\_TRUNC('day', NOW()); -- Retorna: '2024-10-09 00:00:00'

**MySQL:**

Não existe uma função direta como DATE\_TRUNC(), mas podemos usar a função DATE\_FORMAT() para truncar ou formatar a data, ou usar funções como CAST() para truncar horas.

-- Trunca para o início do ano

SELECT DATE\_FORMAT(NOW(), '%Y-01-01 00:00:00') AS truncated\_year; -- Retorna: '2024-01-01 00:00:00'

-- Trunca para o início do mês

SELECT DATE\_FORMAT(NOW(), '%Y-%m-01 00:00:00') AS truncated\_month; -- Retorna: '2024-10-01 00:00:00'

-- Trunca para o início do dia

SELECT DATE(NOW()) AS truncated\_day; -- Retorna: '2024-10-09'

**Microsoft SQL Server:**

A função equivalente é o uso de FORMAT() para truncar a data ou usar CONVERT() para ajustar o formato, embora o truncamento direto seja menos intuitivo

-- Trunca para o início do ano

SELECT CAST(YEAR(GETDATE()) AS VARCHAR(4)) + '-01-01 00:00:00' AS truncated\_year; -- Retorna: '2024-01-01 00:00:00'

-- Trunca para o início do mês

SELECT CONVERT(DATETIME, CONVERT(VARCHAR(7), GETDATE(), 120) + '-01 00:00:00') AS truncated\_month; -- Retorna: '2024-10-01 00:00:00'

-- Trunca para o início do dia

SELECT CONVERT(DATE, GETDATE()) AS truncated\_day; -- Retorna: '2024-10-09'

**Alternativa Databricks:**

No **Databricks**, a função equivalente a DATE\_TRUNC() é a própria função date\_trunc() em SQL, e no **PySpark** você pode usar a função trunc() para truncar datas.

sql:

-- Trunca para o início do ano

SELECT DATE\_TRUNC('year', current\_date()) AS truncated\_year;

-- Trunca para o início do mês

SELECT DATE\_TRUNC('month', current\_date()) AS truncated\_month;

-- Trunca para o início do dia

SELECT DATE\_TRUNC('day', current\_date()) AS truncated\_day;

pyspark:

from pyspark.sql.functions import date\_trunc

# Trunca para o início do ano

display(df.select(date\_trunc('year', df['data\_col']).alias('truncated\_year')))

# Trunca para o início do mês

display(df.select(date\_trunc('month', df['data\_col']).alias('truncated\_month')))

# Trunca para o início do dia

display(df.select(date\_trunc('day', df['data\_col']).alias('truncated\_day')))

Sobre ‘truncar’ uma tabela:

**Truncar** uma data significa remover ou "cortar" as partes mais detalhadas da data e hora, mantendo apenas a parte desejada (como o ano, mês, ou dia) e ajustando as partes posteriores para o início do período.

Data original:

2024-10-09 15:37:45

1. **Truncar para o ano**:
   * Resultado: 2024-01-01 00:00:00
   * Apenas o ano é mantido, e o restante é ajustado para o início do ano.
2. **Truncar para o mês**:
   * Resultado: 2024-10-01 00:00:00
   * O mês é mantido, e o restante é ajustado para o início do mês.
3. **Truncar para o dia**:
   * Resultado: 2024-10-09 00:00:00
   * O dia é mantido, e o tempo (horas, minutos, e segundos) é ajustado para o início do dia.

EXTRACT(field FROM source):

Permite extrair partes específicas de uma data ou timestamp, como ano, mês, dia, hora, etc., de uma coluna ou valor de data.

**PostgreSQL:**

SELECT EXTRACT(YEAR FROM NOW()); -- Retorna: 2024

**MySQL:**

SELECT EXTRACT(YEAR FROM NOW()); -- Retorna: 2024

**Microsoft SQL Server:**

SELECT YEAR(GETDATE()) AS year; -- Retorna: 2024

**Alternativa Databricks:**

SQL: SELECT EXTRACT(YEAR FROM current\_date()) AS year; -- Retorna: 2024

PySpark: df.select(year(col("date\_column")).alias("year"))

TOCHAR(date, format): Utilizada para converter uma data em uma string com o formato especificado.

**PostgreSQL:**

SELECT TO\_CHAR(NOW(), 'YYYY-01-01 00:00:00') AS formatted\_year;

-- Retorna: '2024-01-01 00:00:00'

**MySQL:** Não existe uma função TO\_CHAR(). Para formatar datas, você pode utilizar a função DATE\_FORMAT() que desempenha um papel semelhante.

SELECT DATE\_FORMAT(NOW(), '%Y-01-01 00:00:00') AS formatted\_year;

-- Retorna: '2024-01-01 00:00:00'

**Microsoft SQL Server:**

TO\_CHAR() não está disponível. Para formatar datas, você pode usar a função FORMAT() ou CONVERT().

SELECT FORMAT(GETDATE(), 'yyyy-01-01 00:00:00') AS formatted\_year;

-- Retorna: '2024-01-01 00:00:00'

**Alternativa Databricks:**

SQL: SELECT DATE\_FORMAT(current\_date(), 'yyyy-01-01 00:00:00') AS formatted\_year; -- Retorna: '2024-01-01 00:00:00'

PySpark: df.select(date\_format(col("date\_column"), "yyyy-01-01 00:00:00").alias("formatted\_year"))