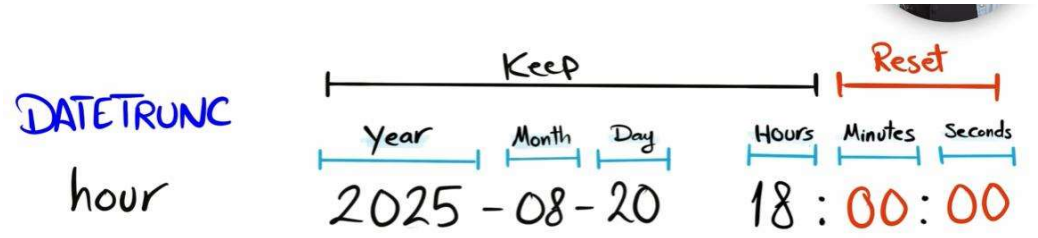
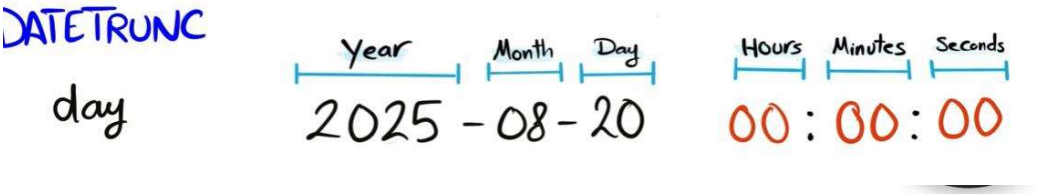
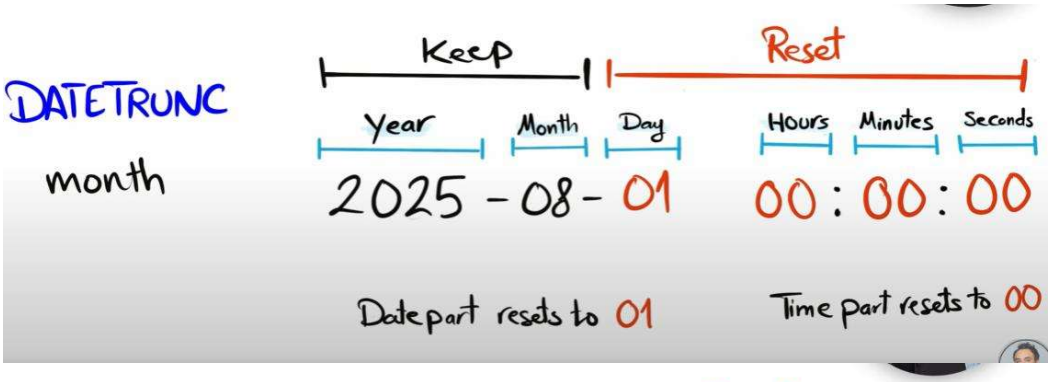
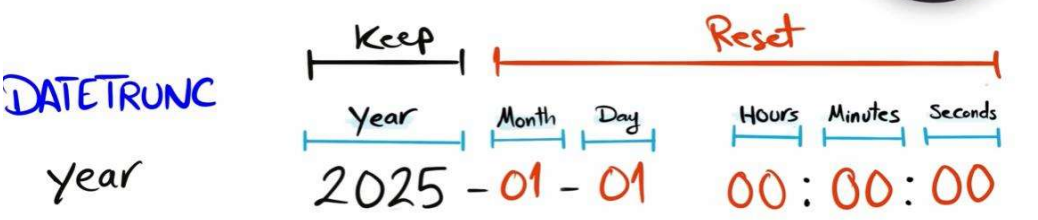
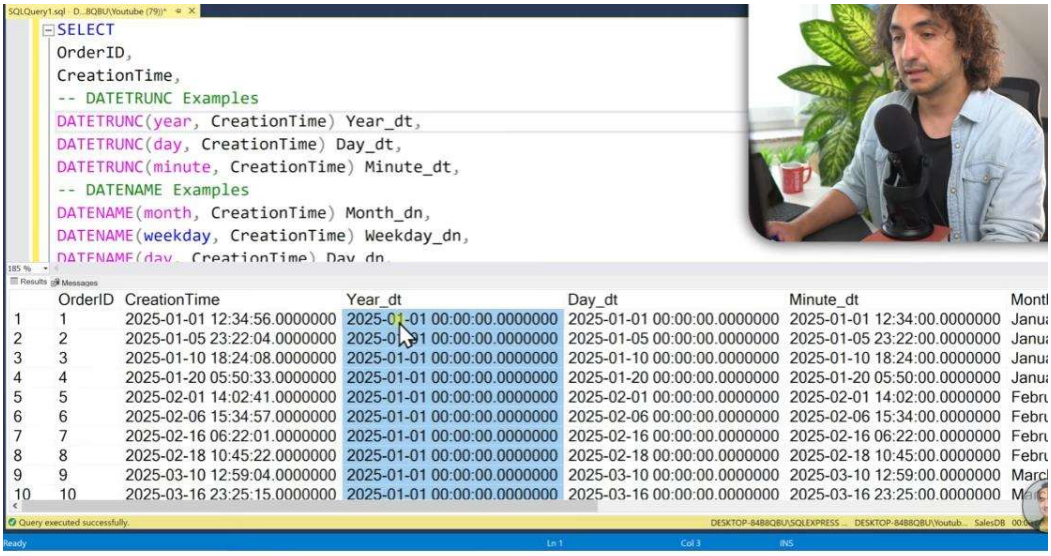

DATETRUNC
hour
Keep
Reset
Year
Month
Day
Hours
Minutes
Seconds
2025 - 08 - 20
18 : 00 : 00


DATETRUNC
day
Year
Month
Day
Hours
Minutes
Seconds
2025 - 08 - 20
00 : 00 : 00

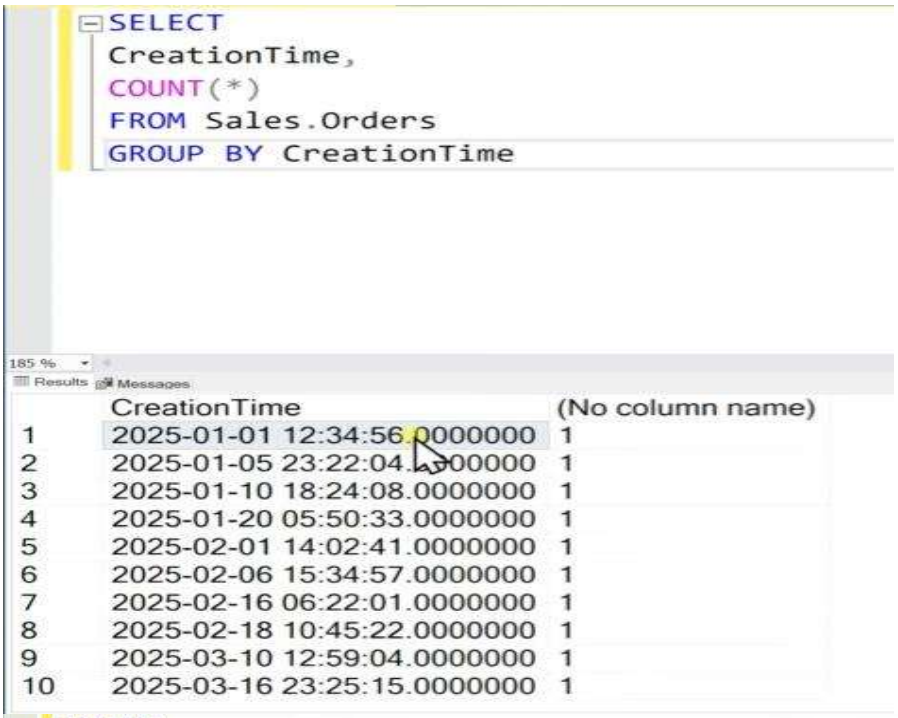

DATETRUNC
month
Keep
Reset
Year
Month
Day
Hours
Minutes
Seconds
2025 - 08 - 01
00 : 00 : 00
Datepart resets to 01
Time part resets to 00


DATETRUNC
year
Keep
Reset
Year
Month
Day
Hours
Minutes
Seconds
2025 - 01 - 01
00 : 00 : 00


SELECT
OrderID,
CreationTime,
-- DATETRUNC Examples
DATETRUNC(year, CreationTime) Year_dt,
DATETRUNC(day, CreationTime) Day_dt,
DATETRUNC(minute, CreationTime) Minute_dt,
-- DATENAME Examples
DATENAME(month, CreationTime) Month_dn,
DATENAME(weekday, CreationTime) Weekday_dn,
OrderID CreationTime Year_dt Day_dt Minute_dt
1 1 2025-01-01 12:34:56.0000000 2025-01-01 00:00:00.0000000 2025-01-01 00:00:00.0000000 2025-01-01 12:34:00.0000000
2 2 2025-01-05 23:22:04.0000000 2025-01-01 00:00:00.0000000 2025-01-05 00:00:00.0000000 2025-01-05 23:22:00.0000000
3 3 2025-01-10 18:24:08.0000000 2025-01-01 00:00:00.0000000 2025-01-10 00:00:00.0000000 2025-01-10 18:24:00.0000000
4 4 2025-01-20 05:50:33.0000000 2025-01-01 00:00:00.0000000 2025-01-20 00:00:00.0000000 2025-01-20 05:50:00.0000000
5 5 2025-02-01 14:02:41.0000000 2025-01-01 00:00:00.0000000 2025-02-01 00:00:00.0000000 2025-02-01 14:02:00.0000000
6 6 2025-02-06 15:34:57.0000000 2025-01-01 00:00:00.0000000 2025-02-06 00:00:00.0000000 2025-02-06 15:34:00.0000000
7 7 2025-02-16 06:22:01.0000000 2025-01-01 00:00:00.0000000 2025-02-16 00:00:00.0000000 2025-02-16 06:22:00.0000000
8 8 2025-02-18 10:45:22.0000000 2025-01-01 00:00:00.0000000 2025-02-18 00:00:00.0000000 2025-02-18 10:45:00.0000000
9 9 2025-03-10 12:59:04.0000000 2025-01-01 00:00:00.0000000 2025-03-10 00:00:00.0000000 2025-03-10 12:59:00.0000000
10 10 2025-03-16 23:25:15.0000000 2025-01-01 00:00:00.0000000 2025-03-16 00:00:00.0000000 2025-03-16 23:25:00.0000000
Query executed successfully.

It helps us to navigate to the heriarchy of date time and we can truncate at thw level we want

Why date trunc is amazing function for data analysis

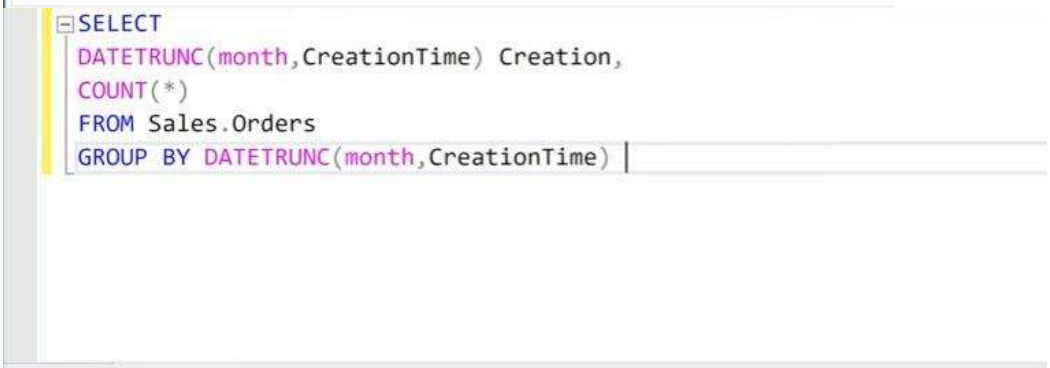


```
SELECT
CreationTime,
COUNT(*)
FROM Sales.Orders
GROUP BY CreationTime
```

185 %

Results Messages

	CreationTime	(No column name)
1	2025-01-01 12:34:56.0000000	1
2	2025-01-05 23:22:04.0000000	1
3	2025-01-10 18:24:08.0000000	1
4	2025-01-20 05:50:33.0000000	1
5	2025-02-01 14:02:41.0000000	1
6	2025-02-06 15:34:57.0000000	1
7	2025-02-16 06:22:01.0000000	1
8	2025-02-18 10:45:22.0000000	1
9	2025-03-10 12:59:04.0000000	1
10	2025-03-16 23:25:15.0000000	1



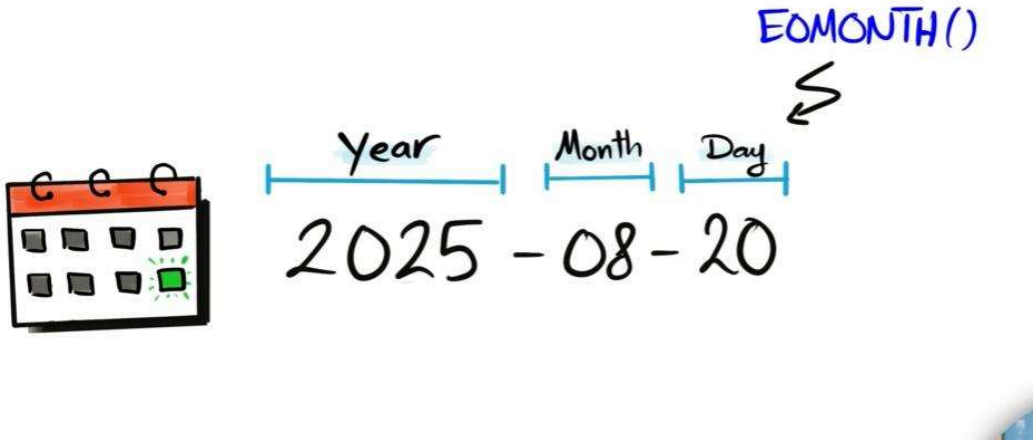
```
SELECT
DATETRUNC(month,CreationTime) Creation,
COUNT(*)
FROM Sales.Orders
GROUP BY DATETRUNC(month,CreationTime)
```

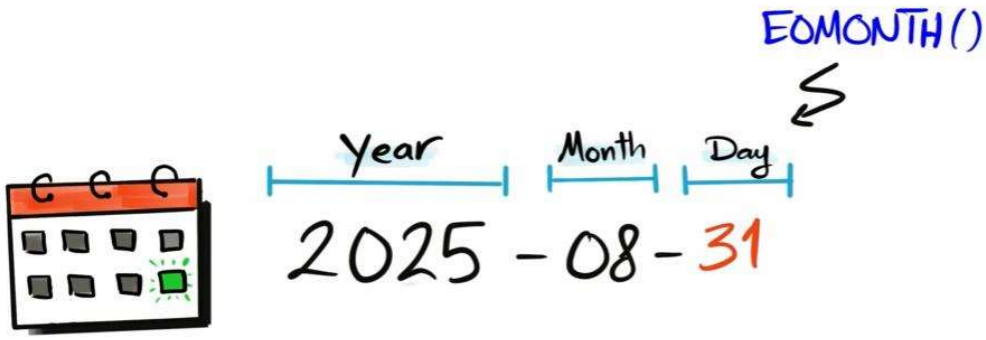
Results Messages

	Creation	(No column name)
1	2025-01-01 00:00:00.0000000	4
2	2025-02-01 00:00:00.0000000	4
3	2025-03-01 00:00:00.0000000	2

EOMONTH ( )

Returns the last day of a month.





EOMONTH()

Year Month Day

2025 - 08 - 31

2025 - 02 - 01

EOMONTH()

Year Month Day

2025 - 08 - 31

2025 - 02 - 28

EOMONTH()

Year Month Day

2025 - 08 - 31

2025 - 02 - 28

2025 - 03 - 31

Stay Same!

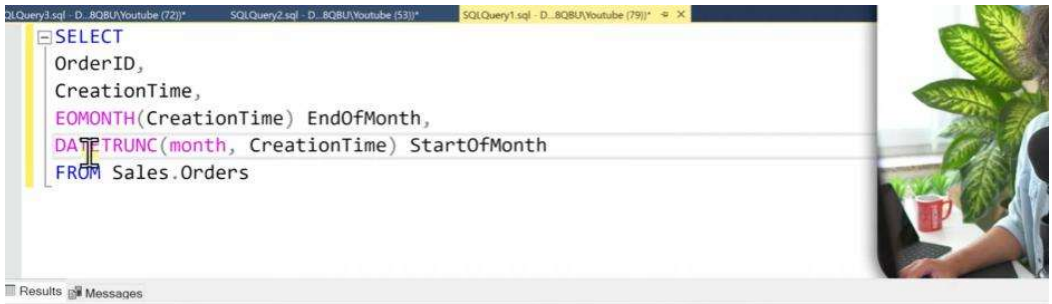
Syntax

```
DAY (date)
MONTH (date)
YEAR (date)
EOMONTH (date)
```

```sql
SELECT
OrderID,
CreationTime,
EOMONTH(CreationTime) EndOfMonth
FROM Sales.Orders
```

Results Messages

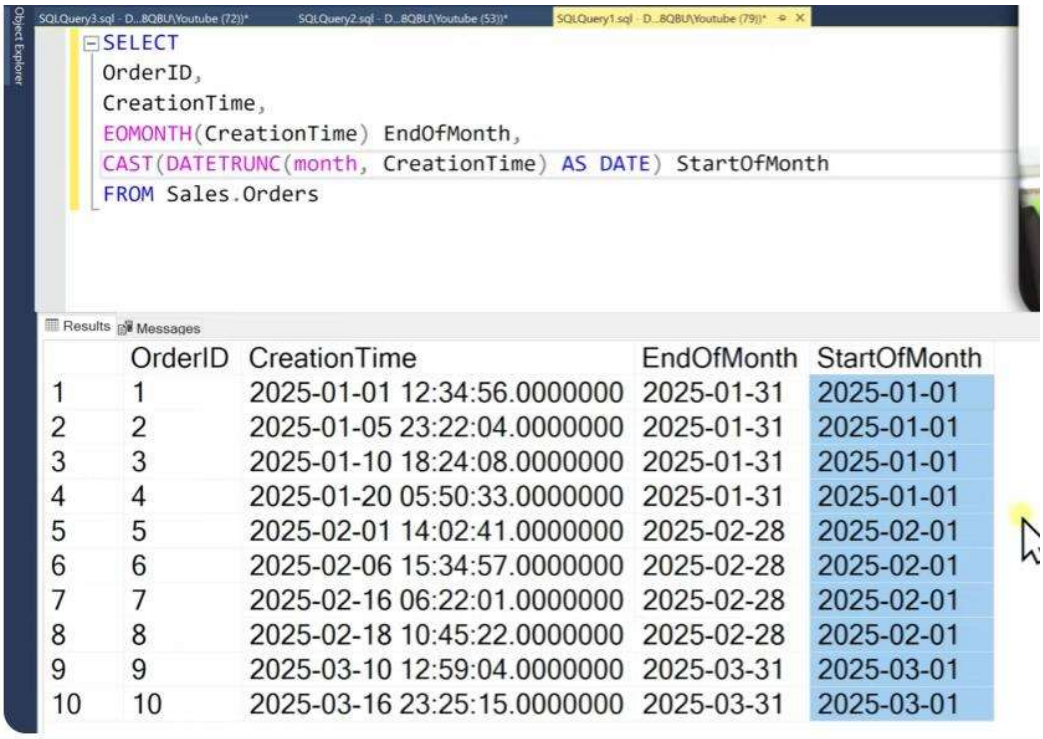
	OrderID	CreationTime	EndOfMonth
1	1	2025-01-01 12:34:56.0000000	2025-01-31
2	2	2025-01-05 23:22:04.0000000	2025-01-31
3	3	2025-01-10 18:24:08.0000000	2025-01-31
4	4	2025-01-20 05:50:33.0000000	2025-01-31
5	5	2025-02-01 14:02:41.0000000	2025-02-28
6	6	2025-02-06 15:34:57.0000000	2025-02-28
7	7	2025-02-16 06:22:01.0000000	2025-02-28
8	8	2025-02-18 10:45:22.0000000	2025-02-28
9	9	2025-03-10 12:59:04.0000000	2025-03-31
10	10	2025-03-16 23:25:15.0000000	2025-03-31



```
SELECT
OrderID,
CreationTime,
EOMONTH(CreationTime) EndOfMonth,
DATETRUNC(month, CreationTime) StartOfMonth
FROM Sales.Orders
```

Results Messages

	OrderID	CreationTime	EndOfMonth	StartOfMonth
1	1	2025-01-01 12:34:56.0000000	2025-01-31	2025-01-01 00:00:00.0000000
2	2	2025-01-05 23:22:04.0000000	2025-01-31	2025-01-01 00:00:00.0000000
3	3	2025-01-10 18:24:08.0000000	2025-01-31	2025-01-01 00:00:00.0000000
4	4	2025-01-20 05:50:33.0000000	2025-01-31	2025-01-01 00:00:00.0000000
5	5	2025-02-01 14:02:41.0000000	2025-02-28	2025-02-01 00:00:00.0000000
6	6	2025-02-06 15:34:57.0000000	2025-02-28	2025-02-01 00:00:00.0000000
7	7	2025-02-16 06:22:01.0000000	2025-02-28	2025-02-01 00:00:00.0000000
8	8	2025-02-18 10:45:22.0000000	2025-02-28	2025-02-01 00:00:00.0000000
9	9	2025-03-10 12:59:04.0000000	2025-03-31	2025-03-01 00:00:00.0000000
10	10	2025-03-16 23:25:15.0000000	2025-03-31	2025-03-01 00:00:00.0000000



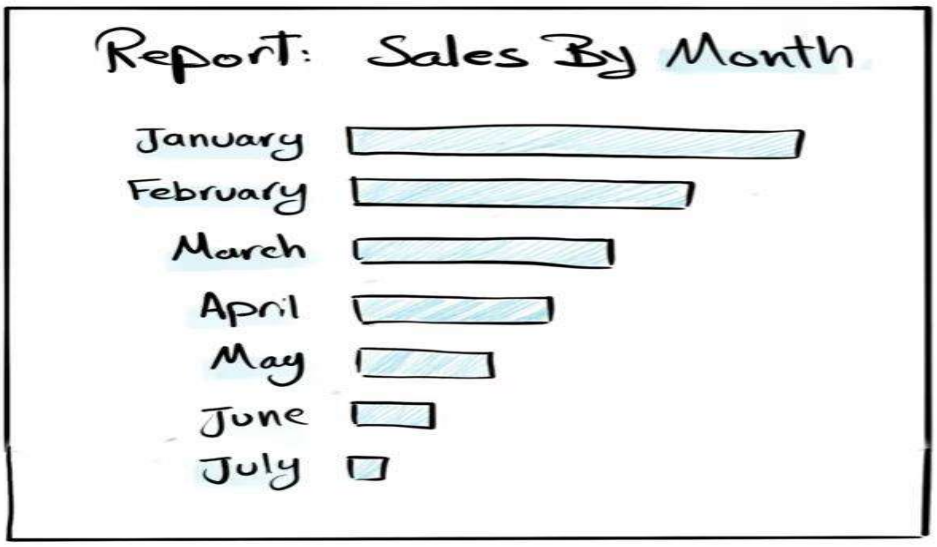
```
SELECT
OrderID,
CreationTime,
EOMONTH(CreationTime) EndOfMonth,
CAST(DATETRUNC(month, CreationTime) AS DATE) StartOfMonth
FROM Sales.Orders
```

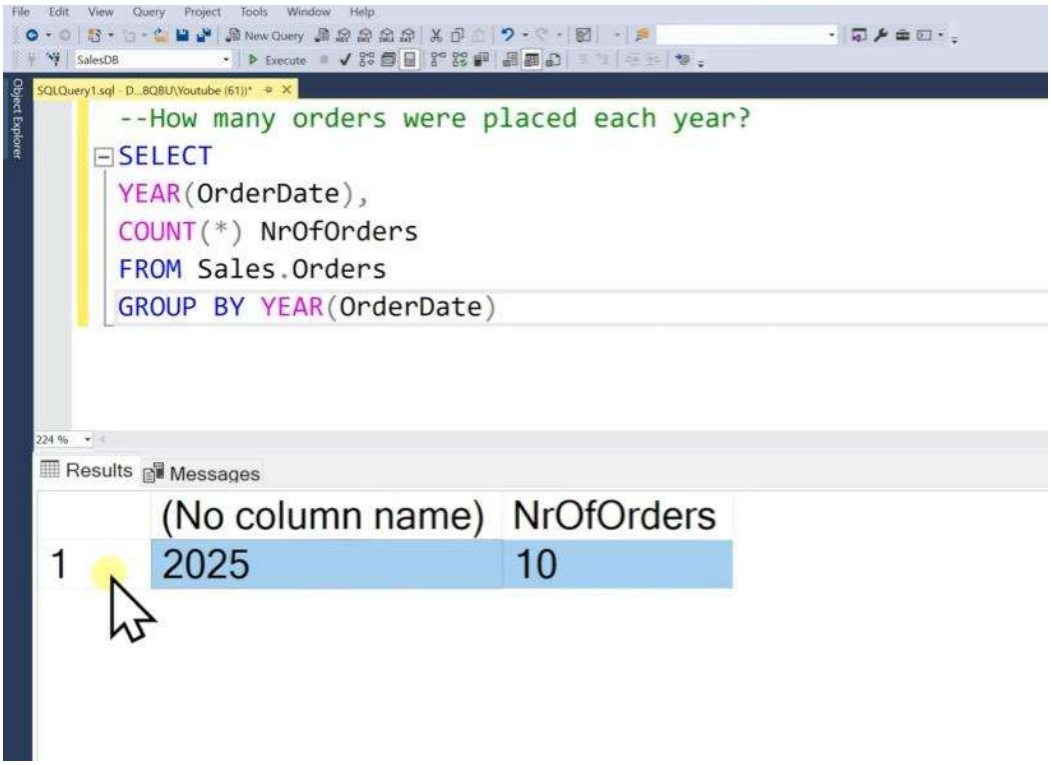
Results Messages

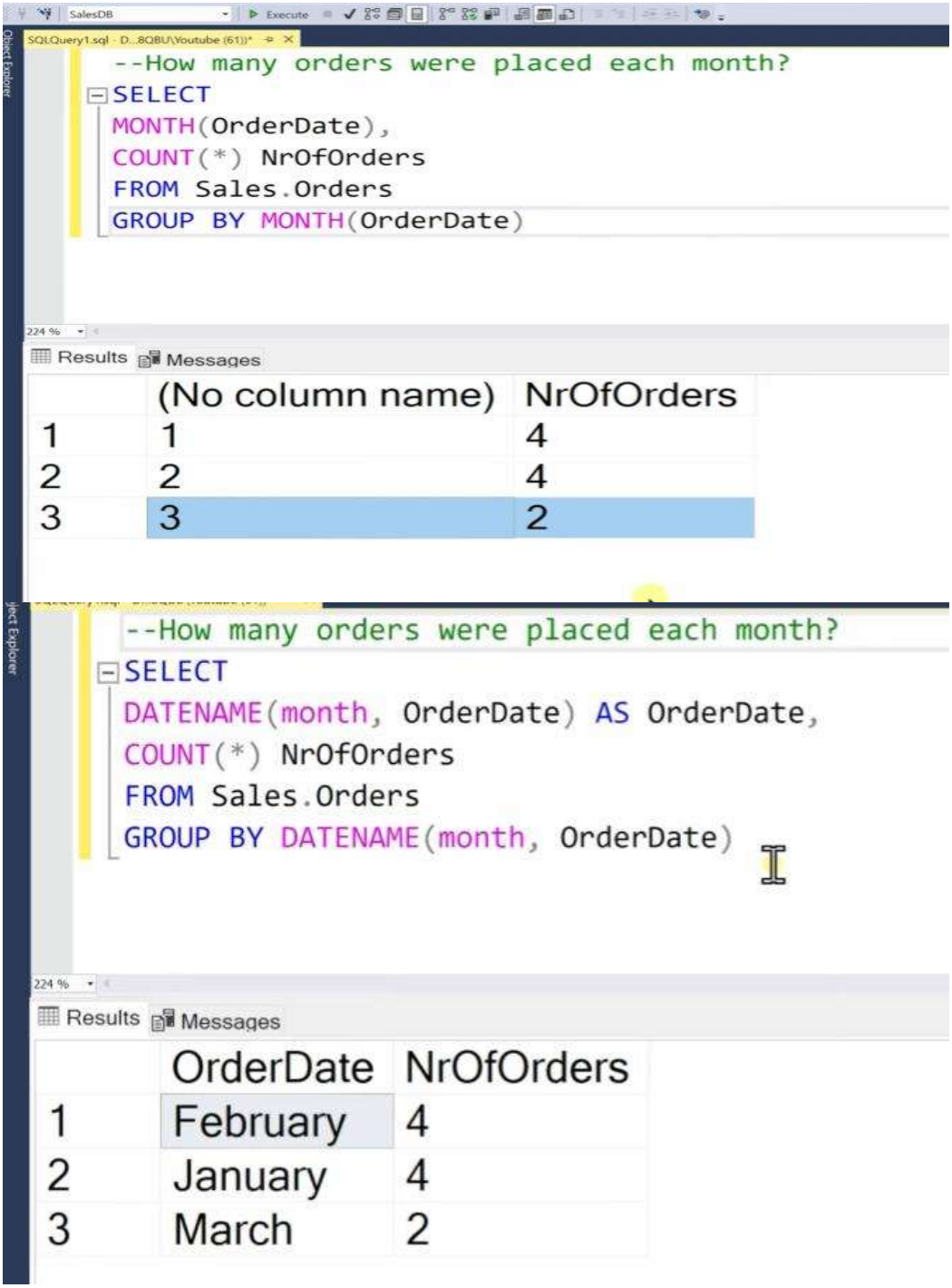
	OrderID	CreationTime	EndOfMonth	StartOfMonth
1	1	2025-01-01 12:34:56.0000000	2025-01-31	2025-01-01
2	2	2025-01-05 23:22:04.0000000	2025-01-31	2025-01-01
3	3	2025-01-10 18:24:08.0000000	2025-01-31	2025-01-01
4	4	2025-01-20 05:50:33.0000000	2025-01-31	2025-01-01
5	5	2025-02-01 14:02:41.0000000	2025-02-28	2025-02-01
6	6	2025-02-06 15:34:57.0000000	2025-02-28	2025-02-01
7	7	2025-02-16 06:22:01.0000000	2025-02-28	2025-02-01
8	8	2025-02-18 10:45:22.0000000	2025-02-28	2025-02-01
9	9	2025-03-10 12:59:04.0000000	2025-03-31	2025-03-01
10	10	2025-03-16 23:25:15.0000000	2025-03-31	2025-03-01

Part Extraction of Date Time Usecases Data Aggregation:
Why we need to extract parts from date

1. Data Aggregation and Reporting
Ex- we are building report Inorder to show sales by year

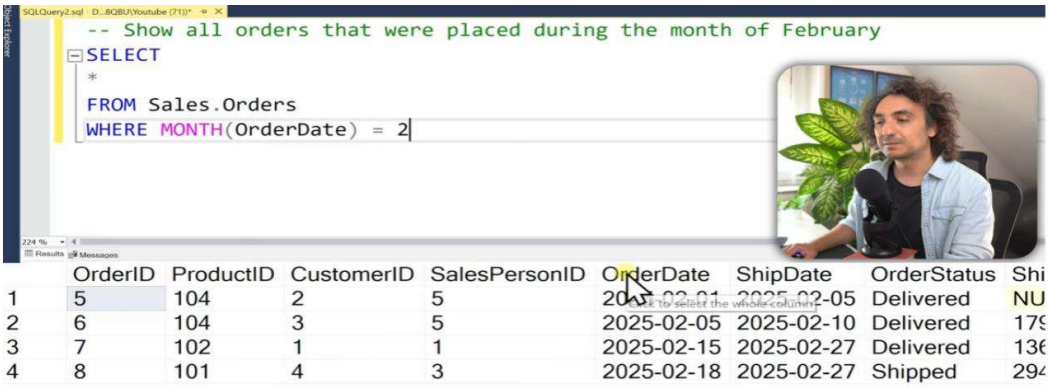

Report: Sales By Month
January
February
March
April
May
June
July

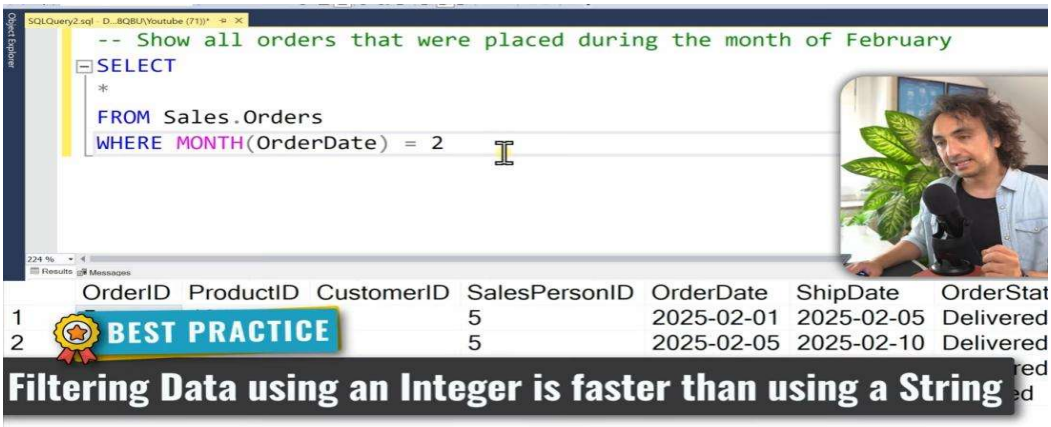

File Edit View Query Project Tools Window Help
New Query
SalesDB
Execute
SQLQuery1.sql - D...8QBU\Youtube (61))*
Object Explorer
--How many orders were placed each year?
SELECT
YEAR(OrderDate),
COUNT(*) NrOfOrders
FROM Sales.Orders
GROUP BY YEAR(OrderDate)
224 %
Results
Messages
(No column name)
NrOfOrders
1
2025
10

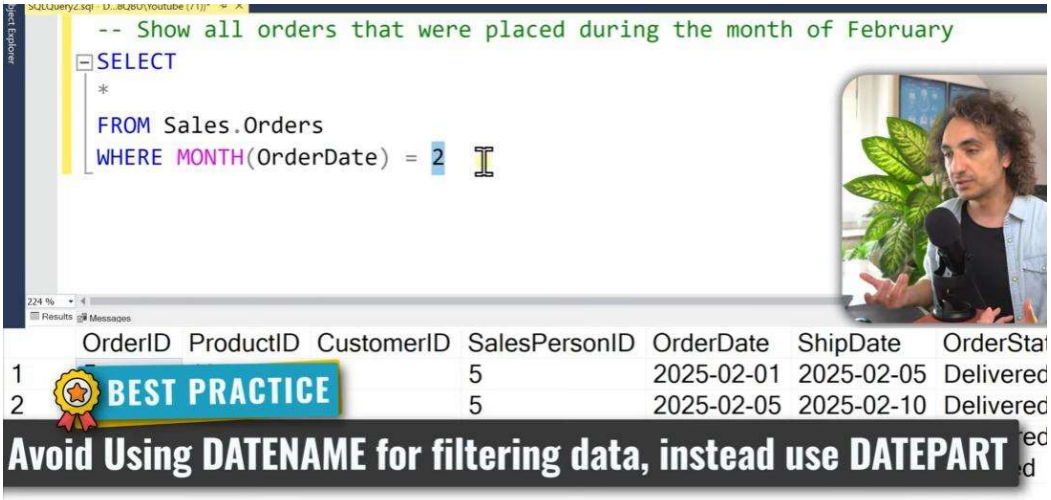


2. Data Filtering

Ex- show all order that were placed during the month of february







Functions Comparision

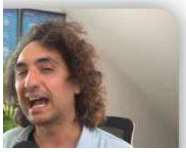
DATA TYPE

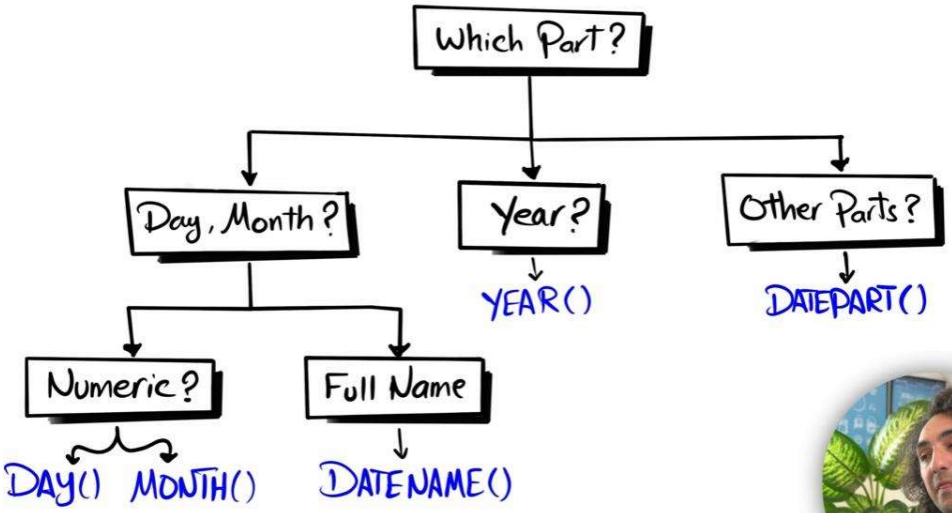
DAY MONTH YEAR DATEPART → INT

DATENAME → STRING

DATETRUNC → DATETIME

EOMONTH → DATE





2025-08-20
09:38:54.840

Date Parts

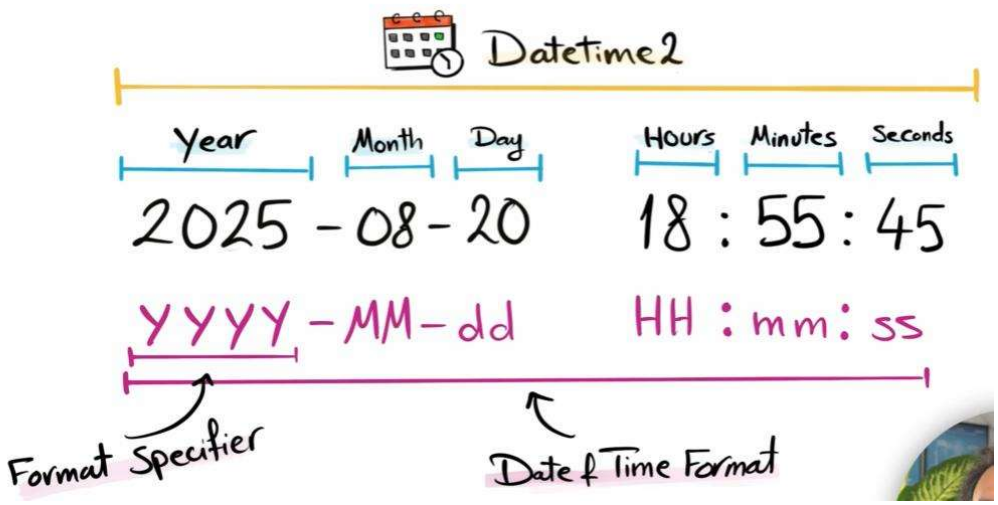
DATEPART

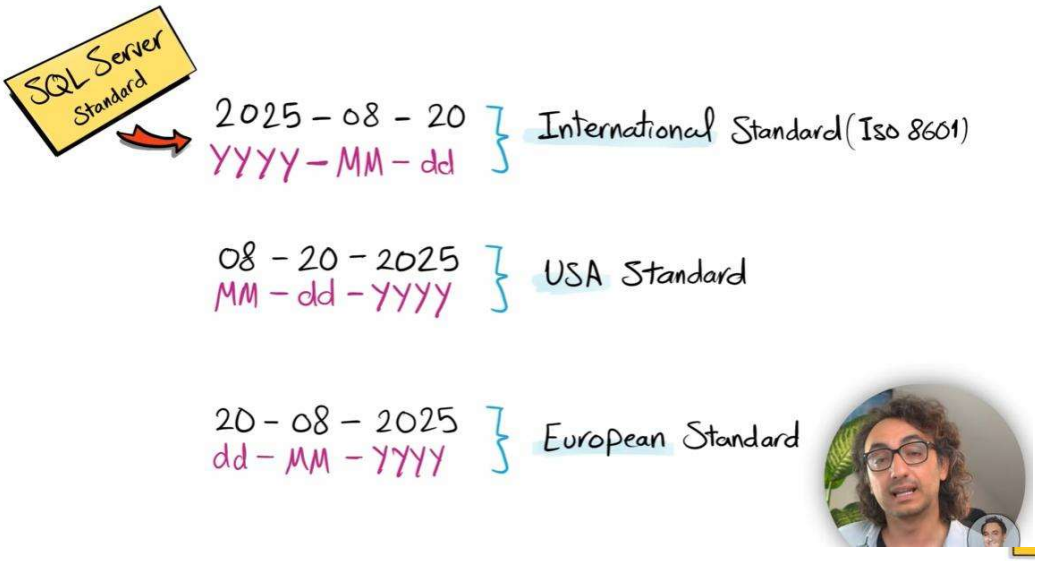
DATENAME

DATETUNC

Part	Abbre.	INT DATEPART	String DATENMAME	Datetime2 DATETRUNC
year	yy, yyyy	2025	2025	2025-01-01 00:00:00
quarter	qq,q	3	3	2025-07-01 00:00:00
month	mm,m	8	August	2025-08-01 00:00:00
dayofyear	dy,y	232	232	2025-08-20 00:00:00
day	dd, d	20	20	2025-08-20 00:00:00
weekday	dw	4	Wednesday	Not supported
week	wk,ww	34	34	2025-08-17 00:00:00
iso_week	ns	34	34	2025-08-18 00:00:00
hour	hh	9	9	2025-08-20 09:00:00
minute	mi,n	45	45	2025-08-20 09:45:00
second	ss,s	21	21	2025-08-20 09:45:21
millisecond	ms	0	0	2025-08-20 09:45:21
microsecond	msc	0	0	2025-08-20 09:45:21
nanosecond	ns	0	0	Not supported
iso_week	isowk, isoww	0	+00:00	Not supported

Date Format :



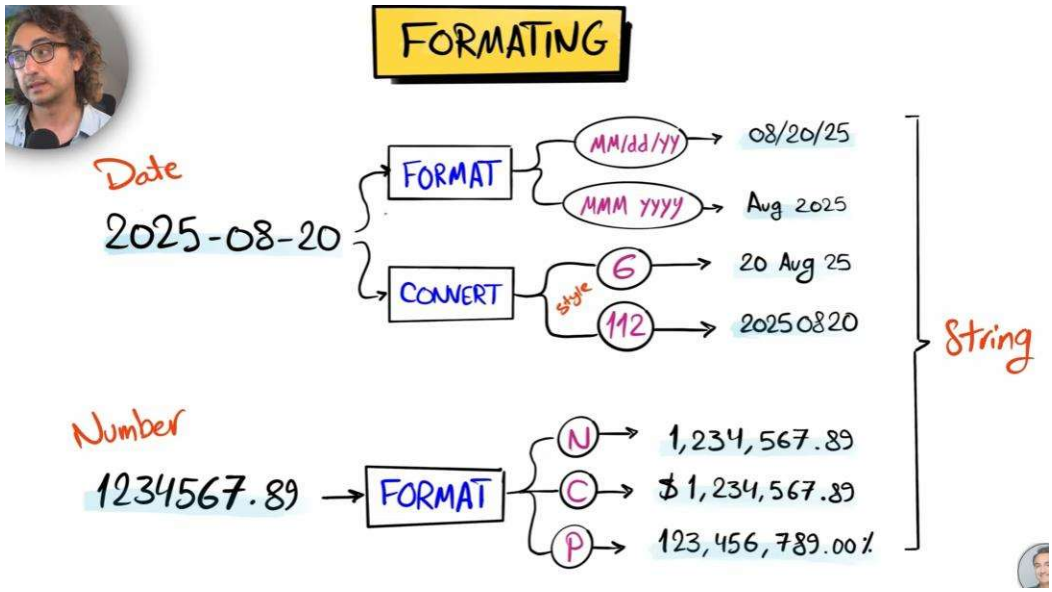


FORMATTING

Changing the format of a value from one to another.

Changing how the data looks.

We can chage date format and number format both things



CAST()
CONVERT()

CASTING "change Data Types"

String '123' → 123 Number

Date 2025-08-20 → '2025-08-20' String

String '2025-08-20' → 2025-08-20 Date

FORMAT ()

Formats a date or time value

Syntax

```
FORMAT(value, format [,culture])
```

Optional

Examples

```
FORMAT(OrderDate,'dd/MM/yyyy')
```

```
FORMAT(OrderDate,'dd/MM/yyyy','ja-JP')
```

```
FORMAT(1234.56, 'D' ,'fr-FR')
```

```sql
SELECT
OrderID,
CreationTime,
FORMAT(CreationTime, 'dd') dd
FROM Sales.Orders
```

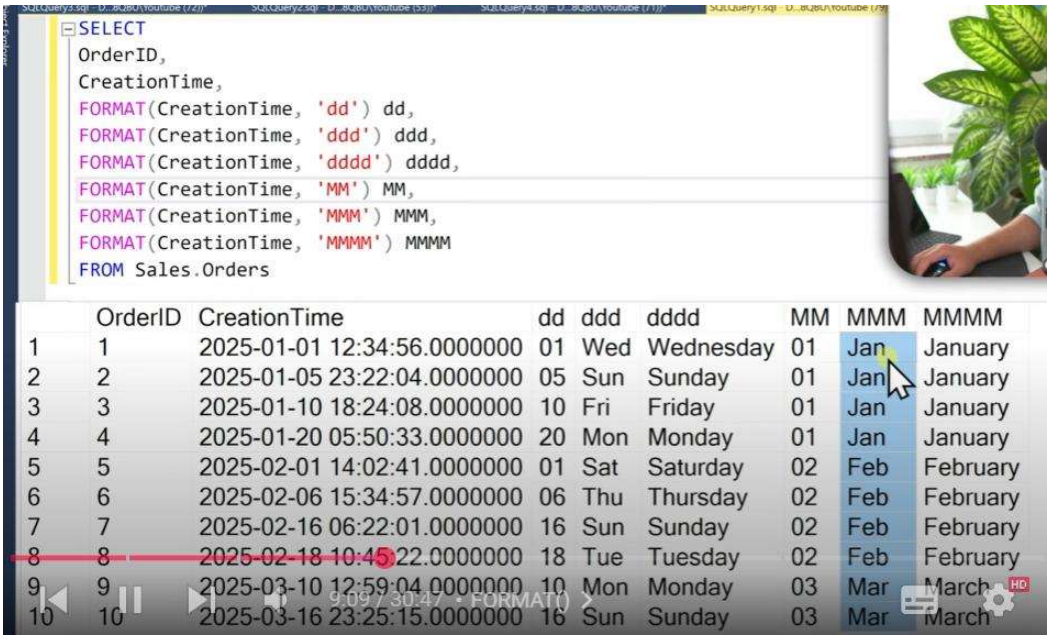
	OrderID	CreationTime	dd
1	1	2025-01-01 12:34:56.0000000	01
2	2	2025-01-05 23:22:04.0000000	05
3	3	2025-01-10 18:24:08.0000000	10
4	4	2025-01-20 05:50:33.0000000	20
5	5	2025-02-01 14:02:41.0000000	01
6	6	2025-02-06 15:34:57.0000000	06
7	7	2025-02-16 06:22:01.0000000	16
8	8	2025-02-18 10:45:22.0000000	18
9	9	2025-03-10 12:59:04.0000000	10
10	10	2025-03-16 23:25:15.0000000	16

```sql
SELECT
OrderID,
CreationTime,
FORMAT(CreationTime, 'dd') dd,
FORMAT(CreationTime, 'ddd') ddd
FROM Sales.Orders
```

	OrderID	CreationTime	dd	ddd
1	1	2025-01-01 12:34:56.0000000	01	Wed
2	2	2025-01-05 23:22:04.0000000	05	Sun
3	3	2025-01-10 18:24:08.0000000	10	Fri
4	4	2025-01-20 05:50:33.0000000	20	Mon
5	5	2025-02-01 14:02:41.0000000	01	Sat
6	6	2025-02-06 15:34:57.0000000	06	Thu
7	7	2025-02-16 06:22:01.0000000	16	Sun
8	8	2025-02-18 10:45:22.0000000	18	Tue
9	9	2025-03-10 12:59:04.0000000	10	Mon
10	10	2025-03-16 23:25:15.0000000	16	Sun

```sql
SELECT
OrderID,
CreationTime,
FORMAT(CreationTime, 'dd') dd,
FORMAT(CreationTime, 'ddd') ddd,
FORMAT(CreationTime, 'dddd') dddd
FROM Sales.Orders
```

OrderID	CreationTime	dd	ddd	dddd
1	2025-01-01 12:34:56.0000000	01	Wed	Wednesday
2	2025-01-05 23:22:04.0000000	05	Sun	Sunday
3	2025-01-10 18:24:08.0000000	10	Fri	Friday
4	2025-01-20 05:50:33.0000000	20	Mon	Monday
5	2025-02-01 14:02:41.0000000	01	Sat	Saturday
6	2025-02-06 15:34:57.0000000	06	Thu	Thursday
7	2025-02-16 06:22:01.0000000	16	Sun	Sunday
8	2025-02-18 10:45:22.0000000	18	Tue	Tuesday
9	2025-03-10 12:59:04.0000000	10	Mon	Monday
10	2025-03-16 23:25:15.0000000	16	Sun	Sunday



Date is in USA Format

```sql
SELECT
OrderID,
CreationTime,
FORMAT(CreationTime, 'MM-dd-yyyy') USA_Format,
FORMAT(CreationTime, 'dd') dd,
FORMAT(CreationTime, 'ddd') ddd,
FORMAT(CreationTime, 'dddd') dddd,
FORMAT(CreationTime, 'MM') MM,
FORMAT(CreationTime, 'MMM') MMM,
FORMAT(CreationTime, 'MMMM') MMMM
FROM Sales.Orders
```

	OrderID	CreationTime	USA_Format	dd	ddd	dddd	MM	MMM	MMMM
1	1	2025-01-01 12:34:56.0000000	01-01-2025	01	Wed	Wednesday	01	Jan	January
2	2	2025-01-05 23:22:04.0000000	01-05-2025	05	Sun	Sunday	01	Jan	January
3	3	2025-01-10 18:24:08.0000000	01-10-2025	10	Fri	Friday	01	Jan	January
4	4	2025-01-20 05:50:33.0000000	01-20-2025	20	Mon	Monday	01	Jan	January
5	5	2025-02-01 14:02:41.0000000	02-01-2025	01	Sat	Saturday	02	Feb	February
6	6	2025-02-06 15:34:57.0000000	02-06-2025	06	Thu	Thursday	02	Feb	February
7	7	2025-02-16 06:22:01.0000000	02-16-2025	16	Sun	Sunday	02	Feb	February
8	8	2025-02-18 10:45:22.0000000	02-18-2025	18	Tue	Tuesday	02	Feb	February
9	9	2025-03-10 12:59:04.0000000	03-10-2025	10	Mon	Monday	03	Mar	March
10	10	2025-03-16 23:25:15.0000000	03-16-2025	16	Sun	Sunday	03	Mar	March

Europe format

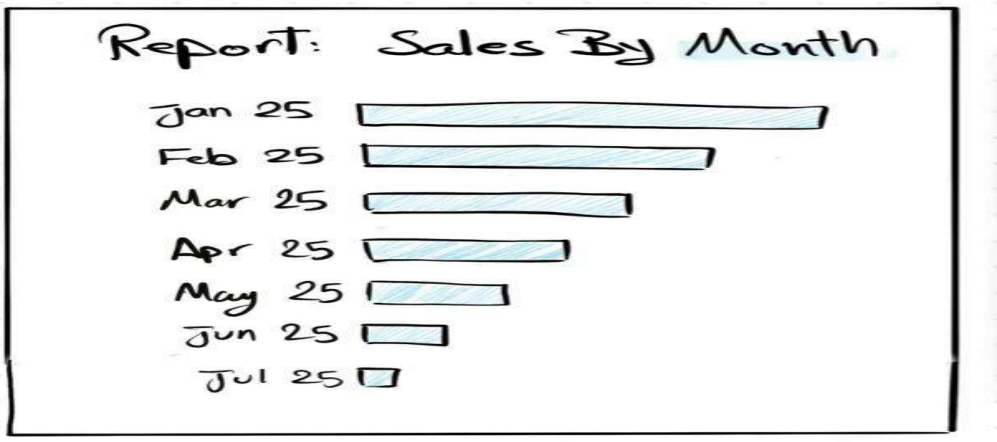
```sql
-- Show CreationTime using the following format:
-- Day Wed Jan Q1 2025 12:34:56 PM

SELECT
OrderID,
CreationTime,
'Day ' + FORMAT(CreationTime, 'ddd MMM') +
' Q'+ DATENAME(quarter, CreationTime) + ' ' +
FORMAT(CreationTime, 'yyyy hh:mm:ss tt') AS CustomeFormat
FROM Sales.Orders
```

	OrderID	CreationTime	CustomeFormat
1	1	2025-01-01 12:34:56.0000000	Day Wed Jan Q1 2025 12:34:56 PM
2	2	2025-01-05 23:22:04.0000000	Day Sun Jan Q1 2025 11:22:04 PM
3	3	2025-01-10 18:24:08.0000000	Day Fri Jan Q1 2025 06:24:08 PM
4	4	2025-01-20 05:50:33.0000000	Day Mon Jan Q1 2025 05:50:33 AM
5	5	2025-02-01 14:02:41.0000000	Day Sat Feb Q1 2025 02:02:41 PM
6	6	2025-02-06 15:34:57.0000000	Day Thu Feb Q1 2025 03:34:57 PM
7	7	2025-02-16 06:22:01.0000000	Day Sun Feb Q1 2025 06:22:01 AM
8	8	2025-02-18 10:45:22.0000000	Day Tue Feb Q1 2025 10:45:22 AM
9	9	2025-03-10 12:59:04.0000000	Day Mon Mar Q1 2025 12:59:04 PM
10	10	2025-03-16 23:25:15.0000000	Day Sun Mar Q1 2025 11:25:15 PM

Format date before aggregation -

Ex- how to extract or add date in report like extract sales by month

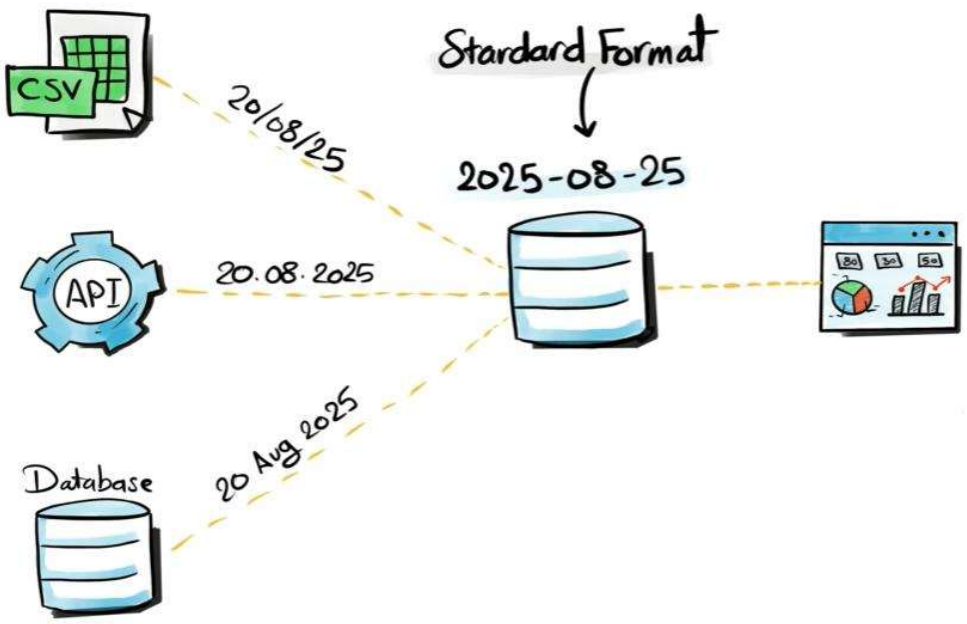



```
SELECT
FORMAT(OrderDate, 'MMM yy') OrderDate,
COUNT(*)
FROM Sales.Orders
GROUP BY FORMAT(OrderDate, 'MMM yy')
```

	OrderDate	(No column name)
1	Feb 25	4
2	Jan 25	4
3	Mar 25	2

Usecases of Date Formatting in Real Projects - In real life Projects

Format different formats of dates into one specific format



2025-08-20
18:55:45

FORMAT

Date & Time Format Speicifiers

Format	Description	Result
D	Full day name	
d	Day of the month	8/20/2025
dd	Day of the month (two-digit)	20
ddd	Abbreviated day name	Wed
dddd	Full day name	Wednesday
M	Month number	44044
MM	Month number (two-digit)	8
MMM	Abbreviated month name	Aug
MMMM	Full month name	August
yy	Year (two-digit)	25
yyyy	Year (four-digit)	2025
hh	Hour (12-hour format, two-digit)	06
HH	Hour (24-hour format, two-digit)	18
m	Minutes	August 20
mm	Minutes (two-digit)	55
s	Seconds	2025-08-20T18:55:45
ss	Seconds (two-digit)	45
f	Fractional seconds (one digit)	Wednesday, August 20, 2025 6:55 PM
ff	Fractional seconds (two digits)	00
fff	Fractional seconds (three digits)	000
tt	AM/PM designator	PM

2025-08-20
18:55:45

Number
Format Speicifiers

FORMAT

Format	Description	Query	Result
N	Numeric default	SELECT FORMAT(1234.56, 'N')	1,234.56
P	Percentage	SELECT FORMAT(1234.56, 'P')	123,456.00 %
C	Currency	SELECT FORMAT(1234.56, 'C')	$1,234.56
E	Scientific notation	SELECT FORMAT(1234.56, 'E')	1,23E+09
F	Fixed-point	SELECT FORMAT(1234.56, 'F')	1234.56
N0	Numeric no decimals	SELECT FORMAT(1234.56, 'N0')	1,235
N1	Numeric one decimal	SELECT FORMAT(1234.56, 'N1')	1,234.6
N2	Numeric two decimals	SELECT FORMAT(1234.56, 'N2')	1,234.56
N , de_DE	Numeric (German)	SELECT FORMAT(1234.56, 'N', 'de-DE')	1.234,56
N, en_US	Numeric (US)	SELECT FORMAT(1234.56, 'N', 'en-US')	1,234.56

CONVERT ( )

Converts a date or time value to a different data type & Formats the value.

Syntax

```
CONVERT(data_type, value [,style])
```

Optional

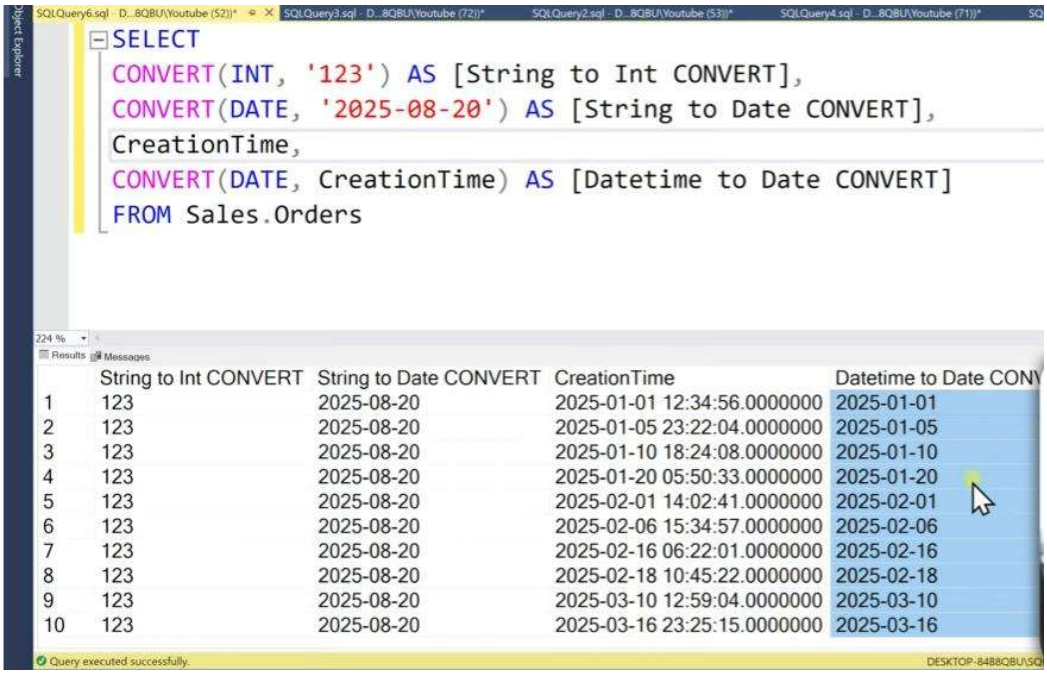
Examples

```
CONVERT(INT, '124')
```

```
CONVERT(VARCHAR, OrderDate, '34')
```

Default Style = 0

Casting - Changing data type from one to another



🔧 Technical Differences

Category	MySQL	SQL Server
Stored Procedures	Basic support	Advanced (with better error handling)
Triggers & Views	Supported	More robust & optimized
Transactions	Supported (InnoDB engine)	Fully supported
JSON Support	Native, flexible	Native, but more structured
Replication	Master-slave, Group Replication	Snapshot, transactional, merge
Partitioning	Manual	Built-in, advanced
Backup & Restore	Manual with `mysqldump`, etc.	GUI-based, differential backups, logs

✅ 1. MySQL Example

🔢 **Converting string to integer:**

```sql
SELECT CAST('12345' AS UNSIGNED) AS converted_value;
```

🏷️ **Column alias with special characters or spaces:**

```sql
SELECT
  CAST('12345' AS UNSIGNED) AS `Converted Value (MySQL)`
;
```

> 🔹 Use **backticks** (`` ` ``) for aliases with spaces/symbols in MySQL.

✅ 2. SQL Server Example

🔢 **Converting string to integer:**

```sql
SELECT CAST('12345' AS INT) AS converted_value;
```

🏷️ **Column alias with special characters or spaces:**

```sql
SELECT
  CAST('12345' AS INT) AS [Converted Value (SQL Server)]
;
```

> 🔹 Use **square brackets** ([ ]) for aliases with spaces/symbols in SQL Server.

🎯 Summary

Feature	MySQL	SQL Server
String to Integer	`CAST('12345' AS UNSIGNED)`	`CAST('12345' AS INT)`
Alias with space	`AS \`My Column\``	`AS [My Column]`
Alias without space	`AS my_column`	`AS my_column`

📌 Notes:

- `CAST('12345' AS UNSIGNED)` in MySQL converts the string to an **unsigned integer**.
- `CAST('12345' AS INT)` in SQL Server converts the string to a **standard integer**.
- **Backticks** (`` `\` ``) are MySQL's way of handling aliases with spaces.
- **Square brackets** (`[]`) are SQL Server's way.

🧠 Quick Comparison:

Region	Format	MySQL `DATE_FORMAT()`
USA	MM/DD/YY	`%m/%d/%y`
Europe	DD/MM/YY	`%d/%m/%y`
ISO	YYYY-MM-DD	`%Y-%m-%d`

```sql
SELECT
CreationTime,
CONVERT(DATE, CreationTime) AS [Datetime to Date CONVERT],
CONVERT(VARCHAR, CreationTime, 32) AS [USA Std. Style:32],
CONVERT(VARCHAR, CreationTime, 34) AS [EURO Std. Style:34]
FROM Sales.Orders
```

	CreationTime	Datetime to Date CONVERT	USA Std. Style:32	EURO Std. Style:34
	2025-01-01 12:34:56.0000000	2025-01-01	01-01-2025	01-01-2025
	2025-01-05 23:22:04.0000000	2025-01-05	01-05-2025	05-01-2025
	2025-01-10 18:24:08.0000000	2025-01-10	01-10-2025	10-01-2025
	2025-01-20 05:50:33.0000000	2025-01-20	01-20-2025	20-01-2025
	2025-02-01 14:02:41.0000000	2025-02-01	02-01-2025	01-02-2025
	2025-02-06 15:34:57.0000000	2025-02-06	02-06-2025	06-02-2025
	2025-02-16 06:22:01.0000000	2025-02-16	02-16-2025	16-02-2025
	2025-02-18 10:45:22.0000000	2025-02-18	02-18-2025	18-02-2025
	2025-03-10 12:59:04.0000000	2025-03-10	03-10-2025	10-03-2025
0	2025-03-16 23:25:15.0000000	2025-03-16	03-16-2025	16-03-2025

Date

#	Format	Example
1	mm/dd/yy	12/30/25
2	yy.mm.dd	25.12.30
3	dd/mm/yy	30/12/2025
4	dd.mm.yy	30.12.25
5	dd-mm-yy	30/12/2025
6	dd-Mon-yy	30-Dec-25
7	Mon dd, yy	Dec 30, 25
10	mm-dd-yy	12-30-25
11	yy/mm/dd	25/12/1930
12	yymmdd	251230
23	yyyy-mm-dd	30/12/2025
31	yyyy-dd-mm	2025-30-12
32	mm-dd-yyyy	12-30-2025
33	mm-yyyy-dd	12-2025-30
34	dd-mm-yyyy	30/12/2025
35	dd-yyyy-mm	30-2025-12
101	mm/dd/yyyy	12/30/2025
102	yyyy.mm.dd	2025.12.30
103	dd/mm/yyyy	30/12/2025
104	dd.mm.yyyy	30.12.2025
105	dd-mm-yyyy	30/12/2025
106	dd Mon yyyy	30-Dec-25
107	Mon dd, yyyy	Dec 30, 2025
110	mm-dd-yyyy	12-30-2025
111	yyyy/mm/dd	30/12/2025
112	yyyymmdd	20251230

Time

#	Format	Example
8	hh:mm:ss	00:38:54
14	hh:mm:ss:nnn	00:38:54:840
24	hh:mm:ss	00:38:54
108	hh:mm:ss	00:38:54
114	hh:mm:ss:nnn	00:38:54:840

Date & Time Styles

CONVERT

Datetime2

#	Format	Example
0	Mon dd yyyy hh:mm AM/PM	Dec 30 2025 12:38AM
9	Mon dd yyyy hh:mm:ss:nnn AM/PM	Dec 30 2025 12:38:54:840AM
13	dd Mon yyyy hh:mm:ss:nnn AM/PM	30 Dec 2025 00:38:54:840AM
20	yyyy-mm-dd hh:mm:ss	2025-12-30 00:38:54
21	yyyy-mm-dd hh:mm:ss:nnn	2025-12-30 00:38:54.840
22	mm/dd/yy hh:mm:ss AM/PM	12/30/25 12:38:54 AM
25	yyyy-mm-dd hh:mm:ss:nnn	2025-12-30 00:38:54.840
26	yyyy-dd-mm hh:mm:ss:nnn	2025-30-12 00:38:54.840
27	mm-dd-yyyy hh:mm:ss:nnn	12-30-2025 00:38:54.840
28	mm-yyyy-dd hh:mm:ss:nnn	12-2025-30 00:38:54.840
29	dd-mm-yyyy hh:mm:ss:nnn	30-12-2025 00:38:54.840
30	dd-yyyy-mm hh:mm:ss:nnn	30-2025-12 00:38:54.840
100	Mon dd yyyy hh:mm AM/PM	Dec 30 2025 12:38AM
109	Mon dd yyyy hh:mm:ss:nnn AM/PM	Dec 30 2025 12:38:54:840AM
113	dd Mon yyyy hh:mm:ss:nnn	30 Dec 2025 00:38:54:840
120	yyyy-mm-dd hh:mm:ss	2025-12-30 00:38:54
121	yyyy-mm-dd hh:mm:ss:nnn	2025-12-30 00:38:54.840
126	yyyy-mm-dd T hh:mm:ss:nnn	2025-12-30T00:38:54.840
127	yyyy-mm-dd T hh:mm:ss:nnn	2025-12-30T00:38:54.840

```
2025-08-20
18:55:45.840
```

CAST ()

Converts a value to a specified data type.

Syntax

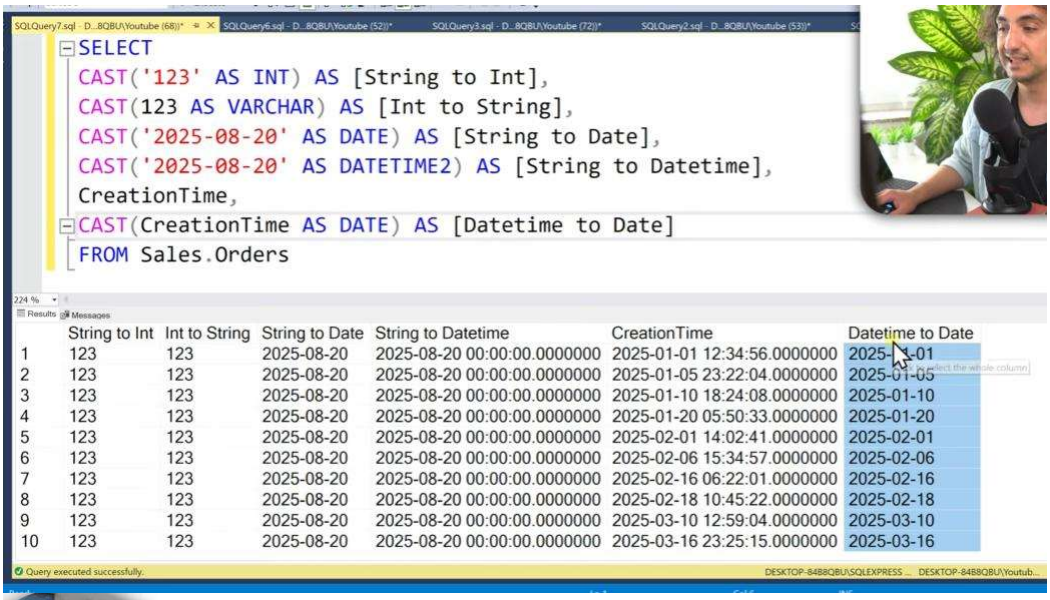
```
CAST(value AS data_type)
```

Examples

```
CAST('123' AS INT)
```

```
CAST('2025-08-20' AS DATE)
```

No format can be specified



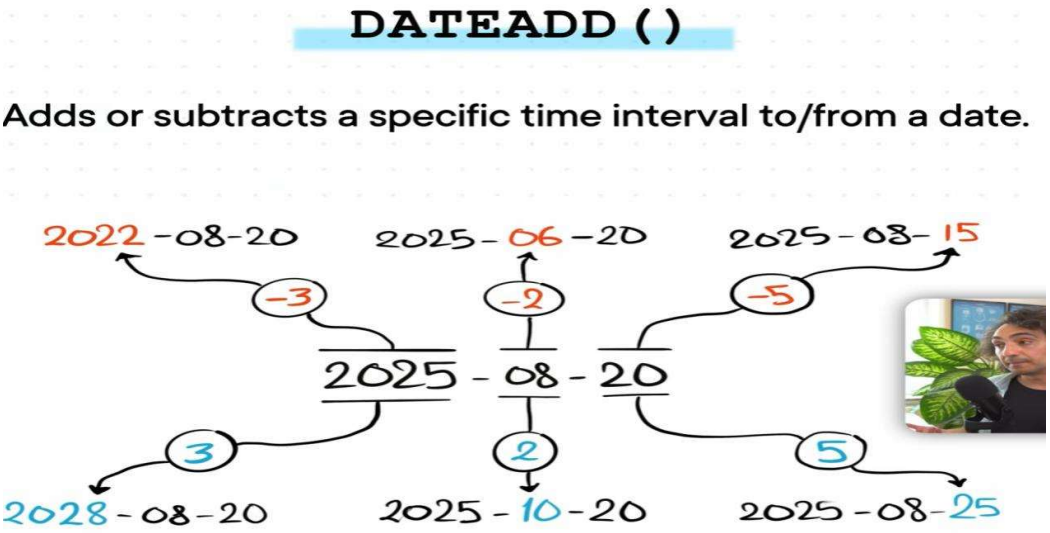
```
SELECT
CAST('123' AS INT) AS [String to Int],
CAST(123 AS VARCHAR) AS [Int to String],
CAST('2025-08-20' AS DATE) AS [String to Date],
CAST('2025-08-20' AS DATETIME2) AS [String to Datetime],
CreationTime,
CAST(CreationTime AS DATE) AS [Datetime to Date]
FROM Sales.Orders
```

Results Messages

	String to Int	Int to String	String to Date	String to Datetime	CreationTime	Datetime to Date
1	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-01-01 12:34:56.0000000	2025-01-01
2	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-01-05 23:22:04.0000000	2025-01-05
3	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-01-10 18:24:08.0000000	2025-01-10
4	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-01-20 05:50:33.0000000	2025-01-20
5	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-02-01 14:02:41.0000000	2025-02-01
6	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-02-06 15:34:57.0000000	2025-02-06
7	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-02-16 06:22:01.0000000	2025-02-16
8	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-02-18 10:45:22.0000000	2025-02-18
9	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-03-10 12:59:04.0000000	2025-03-10
10	123	123	2025-08-20	2025-08-20 00:00:00.0000000	2025-03-16 23:25:15.0000000	2025-03-16

Query executed successfully.

Date Calculation:



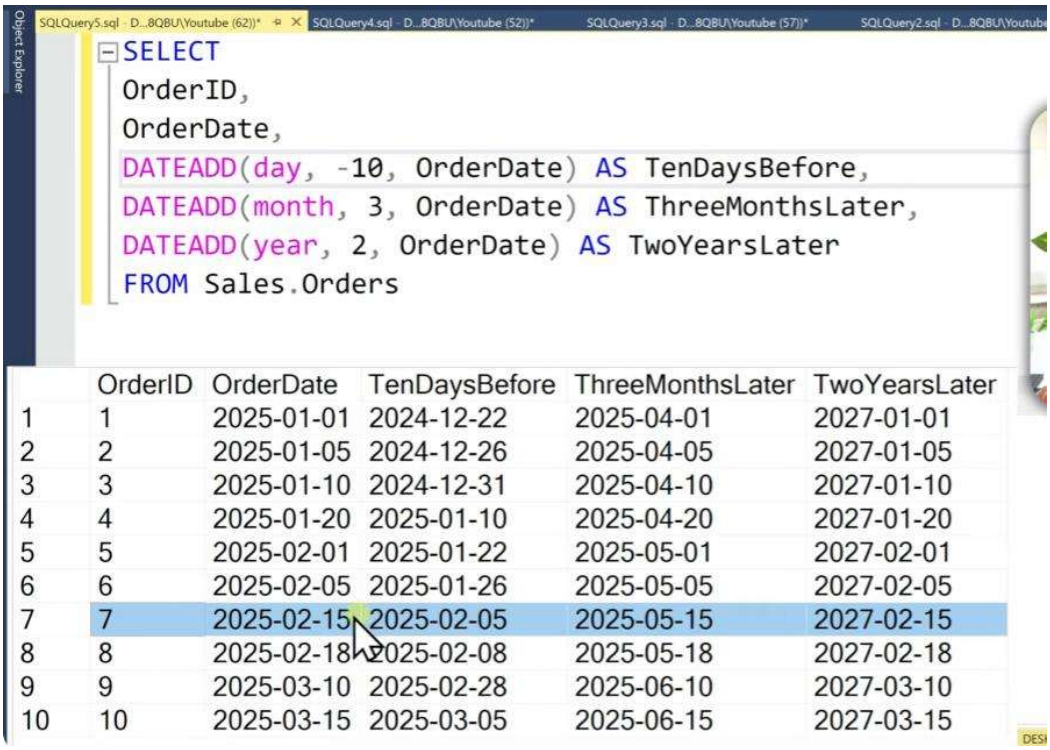
Syntax

```
DATEADD(part, interval, date)
```

Examples

```
DATEADD(year, 2, OrderDate)
```

```
DATEADD(month, -4, OrderDate)
```

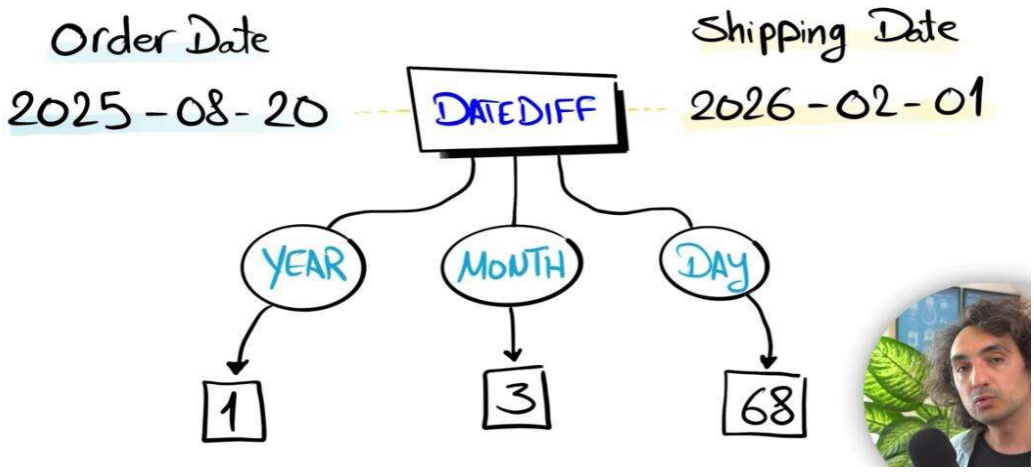


```
SELECT
OrderID,
OrderDate,
DATEADD(day, -10, OrderDate) AS TenDaysBefore,
DATEADD(month, 3, OrderDate) AS ThreeMonthsLater,
DATEADD(year, 2, OrderDate) AS TwoYearsLater
FROM Sales.Orders
```

	OrderID	OrderDate	TenDaysBefore	ThreeMonthsLater	TwoYearsLater
1	1	2025-01-01	2024-12-22	2025-04-01	2027-01-01
2	2	2025-01-05	2024-12-26	2025-04-05	2027-01-05
3	3	2025-01-10	2024-12-31	2025-04-10	2027-01-10
4	4	2025-01-20	2025-01-10	2025-04-20	2027-01-20
5	5	2025-02-01	2025-01-22	2025-05-01	2027-02-01
6	6	2025-02-05	2025-01-26	2025-05-05	2027-02-05
7	7	2025-02-15	2025-02-05	2025-05-15	2027-02-15
8	8	2025-02-18	2025-02-08	2025-05-18	2027-02-18
9	9	2025-03-10	2025-02-28	2025-06-10	2027-03-10
10	10	2025-03-15	2025-03-05	2025-06-15	2027-03-15

DATEDIFF()

Find the difference between two dates.



Syntax

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

Examples

```
DATEDIFF(year, OrderDate, ShipDate)
```

```
DATEDIFF(day, OrderDate, ShipDate)
```

```sql
-- Calculate the age of employees
SELECT
EmployeeID,
BirthDate,
DATEDIFF(year, BirthDate, GETDATE()) Age
FROM Sales.Employees
```

	EmployeeID	BirthDate	Age
1	1	1988-12-05	36
2	2	1972-11-25	52
3	3	1986-01-05	38
4	4	1977-02-10	47
5	5	1982-02-11	42

```sql
-- Find the average shipping duration in days for each month
SELECT
MONTH(OrderDate) AS OrderDate,
AVG(DATEDIFF(day, OrderDate, ShipDate)) AvgShip
FROM Sales.Orders
GROUP BY MONTH(OrderDate)
```

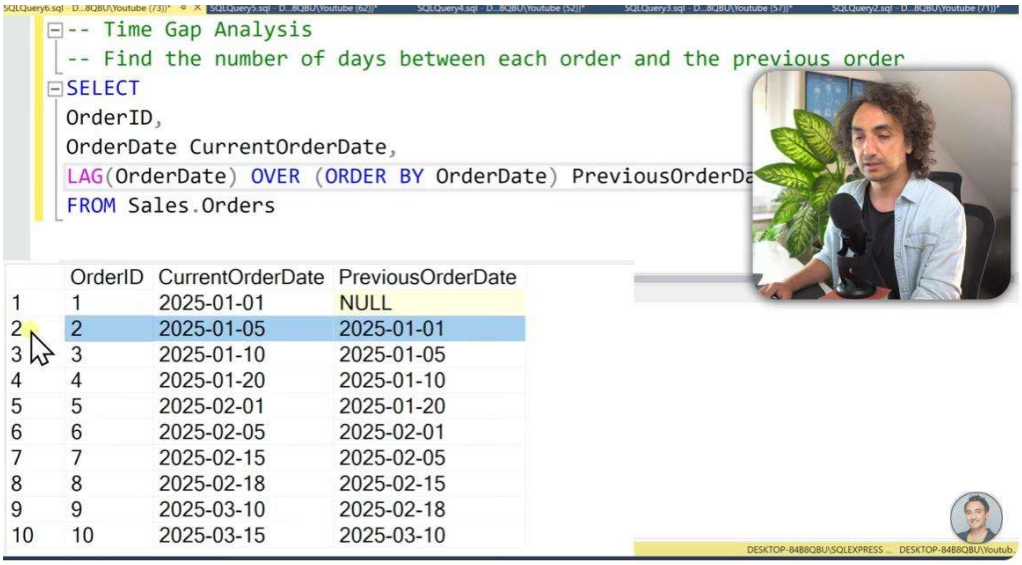
	OrderDate	AvgShip
1	1	7
2	2	7
3	3	5

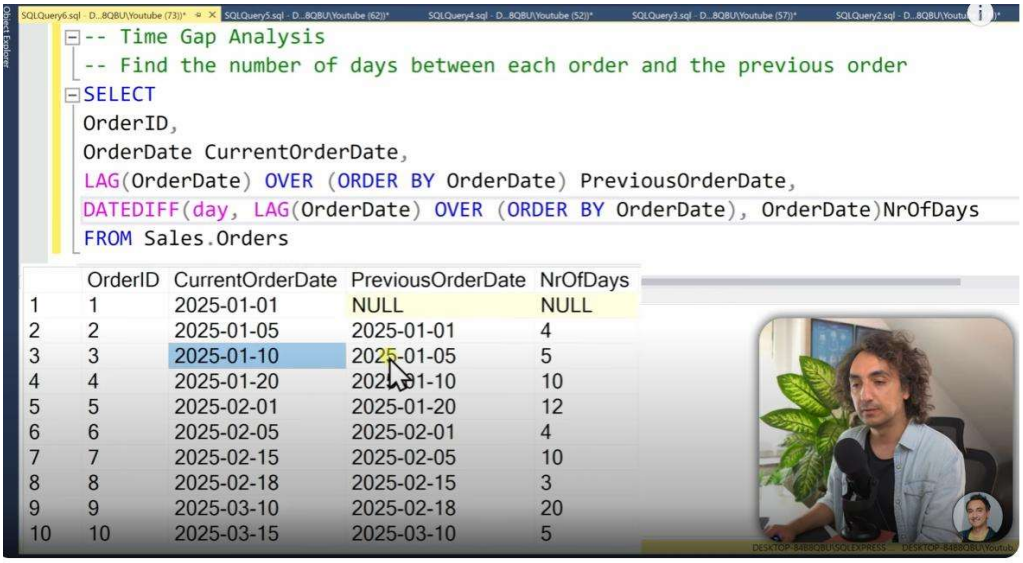
```sql
-- Time Gap Analysis
-- Find the number of days between each order and the previous order
SELECT
OrderID,
OrderDate
FROM Sales.Orders
```

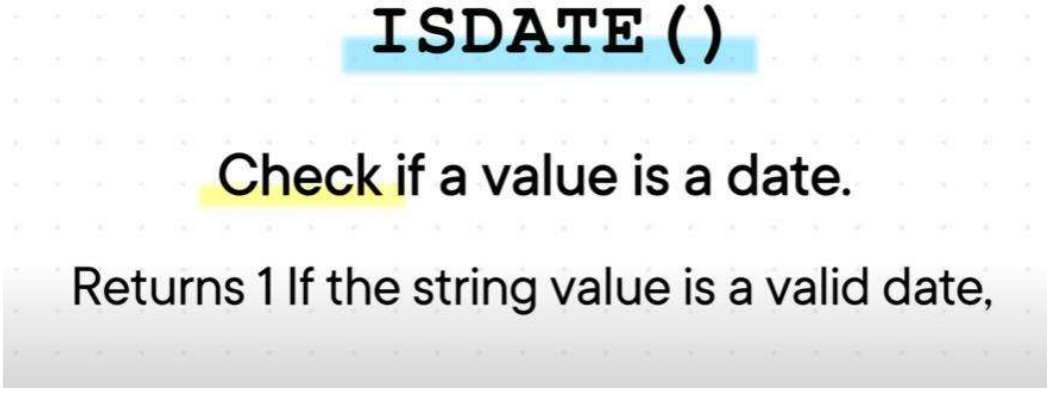
	OrderID	OrderDate
1	1	2025-01-01
		25-01-05
		25-01-10
7	7	2025-02-15
8	8	2025-02-18
9	9	2025-03-10
10	10	2025-03-15

LAG ()

Access a value from the previous row


-- Time Gap Analysis
-- Find the number of days between each order and the previous order
SELECT
OrderID,
OrderDate CurrentOrderDate,
LAG(OrderDate) OVER (ORDER BY OrderDate) PreviousOrderDa
FROM Sales.Orders
OrderID CurrentOrderDate PreviousOrderDate
1 1 2025-01-01 NULL
2 2 2025-01-05 2025-01-01
3 3 2025-01-10 2025-01-05
4 4 2025-01-20 2025-01-10
5 5 2025-02-01 2025-01-20
6 6 2025-02-05 2025-02-01
7 7 2025-02-15 2025-02-05
8 8 2025-02-18 2025-02-15
9 9 2025-03-10 2025-02-18
10 10 2025-03-15 2025-03-10


-- Time Gap Analysis
-- Find the number of days between each order and the previous order
SELECT
OrderID,
OrderDate CurrentOrderDate,
LAG(OrderDate) OVER (ORDER BY OrderDate) PreviousOrderDate,
DATEDIFF(day, LAG(OrderDate) OVER (ORDER BY OrderDate), OrderDate)NrOfDays
FROM Sales.Orders
OrderID CurrentOrderDate PreviousOrderDate NrOfDays
1 1 2025-01-01 NULL NULL
2 2 2025-01-05 2025-01-01 4
3 3 2025-01-10 2025-01-05 5
4 4 2025-01-20 2025-01-10 10
5 5 2025-02-01 2025-01-20 12
6 6 2025-02-05 2025-02-01 4
7 7 2025-02-15 2025-02-05 10
8 8 2025-02-18 2025-02-15 3
9 9 2025-03-10 2025-02-18 20
10 10 2025-03-15 2025-03-10 5


ISDATE()
Check if a value is a date.
Returns 1 If the string value is a valid date,

Syntax

```
ISDATE(value)
```

```
ISDATE('2025-08-20')
```

```
ISDATE(2025)
```

```
SELECT
ISDATE('123') DateCheck1,
ISDATE('2025-08-20') DateCheck2,
ISDATE('20-08-2025') DateCheck3,
ISDATE('2025') DateCheck4,
ISDATE('08') DateCheck5
```

221 %

Results Messages

	DateCheck1	DateCheck2	DateCheck3	DateCheck4	DateCheck5
1	0	1	0	1	0

\

```
SELECT
    CAST(OrderDate AS DATE) OrderDate
FROM
(
    SELECT '2025-08-20' AS OrderDate UNION
    SELECT '2025-08-21' UNION
    SELECT '2025-08-23' UNION
    SELECT '2025-08'
)t
```

221 %

Results Messages

```
Msg 241, Level 16, State 1, Line 2
Conversion failed when converting date and/or time from character string.
```

```sql
SELECT
    --CAST(OrderDate AS DATE) OrderDate,
    OrderDate,
    ISDATE(OrderDate)
FROM
(
    SELECT '2025-08-20' AS OrderDate UNION
    SELECT '2025-08-21' UNION
    SELECT '2025-08-23' UNION
    SELECT '2025-08'
)t
```

221 % Results Messages

	OrderDate	(No column name)
1	2025-08	0
2	2025-08-20	1
3	2025-08-21	1
4	2025-08-23	1

Even data is in bad quality we make rule and correct issue

```sql
SELECT
    --CAST(OrderDate AS DATE) OrderDate,
    OrderDate,
    ISDATE(OrderDate),
    CASE WHEN ISDATE(OrderDate) = 1 THEN CAST(OrderDate AS DATE)
    END NewOrderDate
FROM
(
    SELECT '2025-08-20' AS OrderDate UNION
    SELECT '2025-08-21' UNION
    SELECT '2025-08-23' UNION
    SELECT '2025-08'
)t
```

183 % Results Messages

	OrderDate	(No column name)	NewOrderDate
1	2025-08	0	NULL
2	2025-08-20	1	2025-08-20
3	2025-08-21	1	2025-08-21
4	2025-08-23	1	2025-08-23

```sql
SELECT
    --CAST(OrderDate AS DATE) OrderDate,
    OrderDate,
    ISDATE(OrderDate),
    CASE WHEN ISDATE(OrderDate) = 1 THEN CAST(OrderDate AS DATE)
    END NewOrderDate
FROM
(
    SELECT '2025-08-20' AS OrderDate UNION
    SELECT '2025-08-21' UNION
    SELECT '2025-08-23' UNION
    SELECT '2025-08'
)t
WHERE ISDATE(OrderDate) = 0
```

151 % Results Messages

	OrderDate	(No column name)	NewOrderDate
1	2025-08	0	NULL

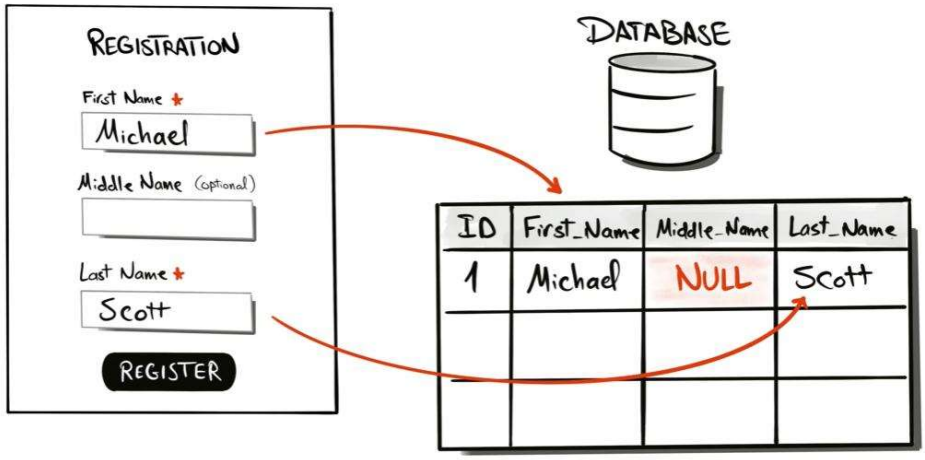
```sql
SELECT
    --CAST(OrderDate AS DATE) OrderDate,
    OrderDate,
    ISDATE(OrderDate),
    CASE WHEN ISDATE(OrderDate) = 1 THEN CAST(OrderDate AS DATE)
         ELSE '9999-01-01'
    END NewOrderDate
FROM
(
    SELECT '2025-08-20' AS OrderDate UNION
    SELECT '2025-08-21' UNION
    SELECT '2025-08-23' UNION
    SELECT '2025-08'
)t
--WHERE ISDATE(OrderDate) = 0
```

151 %

Results Messages

	OrderDate	(No column name)	NewOrderDate
1	2025-08	0	9999-01-01
2	2025-08-20	1	2025-08-20
3	2025-08-21	1	2025-08-21
4	2025-08-23	1	2025-08-23

This function helps to clean data



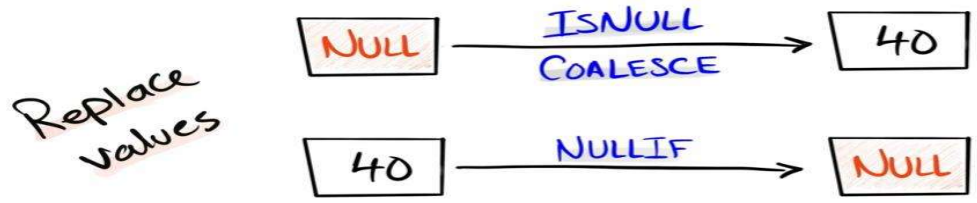
WHAT is NULL?

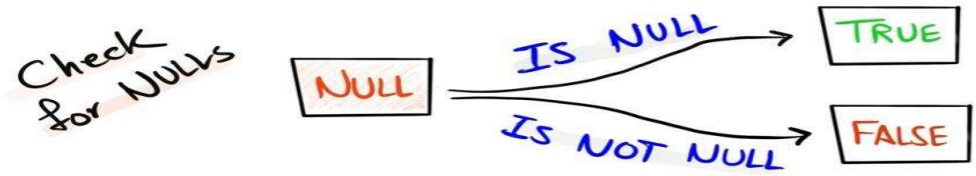
NULL means nothing, unknown!

NULL is not equal to anything!

- NULL is not zero
- NULL is not empty string
- NULL is not blank space

Null Functions:





Syntax

```
ISNULL(value, replacement_value)
```

Example

```
ISNULL(Shipping_Address, 'unknown')
```

Default Value

Example

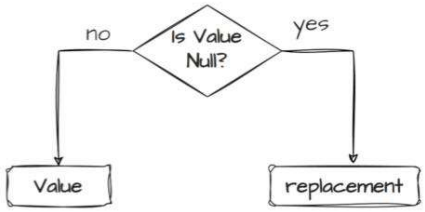
```
ISNULL(Shipping_Address, Billing_Address)
```

SYNTAX

```
ISNULL(value, replacement)
```

```
ISNULL(ShippingAddress, 'N/A')
```

OrderID	Shipment Address	ISNULL
1	A	A
2	NULL	N/A

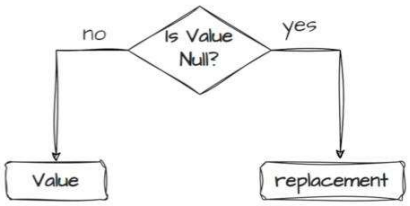


SYNTAX

```
ISNULL(value, replacement)
```

```
ISNULL(ShippingAddress, BillingAddress)
```

OrderID	Shipment Address	Billing Address	ISNULL
1	A	B	A
2	NULL	C	C
3	NULL	NULL	NULL



COALESCE ()

Returns the first non-null value from a list

Syntax

```
COALESCE(value1, value2, value3, ...)
```

Example

```
COALESCE(Shipping_Address, 'unknown')
```

Example

```
COALESCE(Shipping_Address, Billing_Address)
```

Example

```
COALESCE(Shipping_Address, Billing_Address, 'unknown')
```

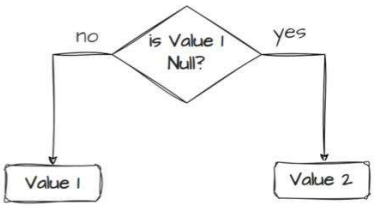
How it works

SYNTAX

```
COALESCE(value1, value2, value3)
```

```
COALESCE(ShippingAddress, BillingAddress)
```

OrderID	Shipment Address	Billing Address	COALESCE
1	A	B	A
2	NULL	C	C
3	NULL	NULL	NULL

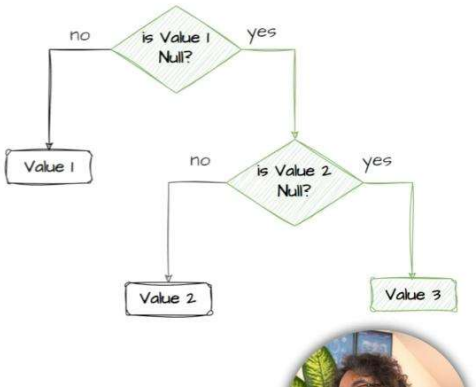


SYNTAX

```
COALESCE(value1, value2, value3)
```

```
COALESCE(ShippingAddress, BillingAddress, 'N/A')
```

OrderID	Shipment Address	Billing Address	COALESCE
1	A	B	A
2	NULL	C	C
3	NULL	NULL	N/A



ISNULL	COALESCE
Limited to two values	Unlimited
Fast	Slow
SQL Server → ISNULL Oracle → NVL MySQL → IFNULL	Available in All Databases

Usecase of data handling

ISNULL | COALESCE

- USE CASE -

Handle the NULL before doing data aggregations