



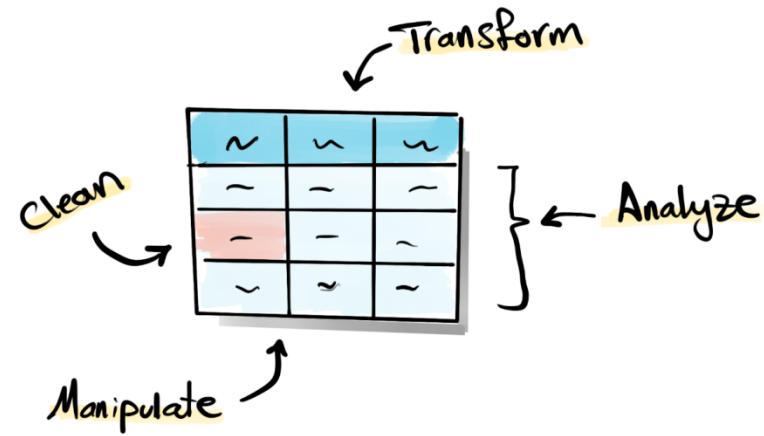
DATA WITH BARAA

SQL FUNCTIONS

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | SQL Functions



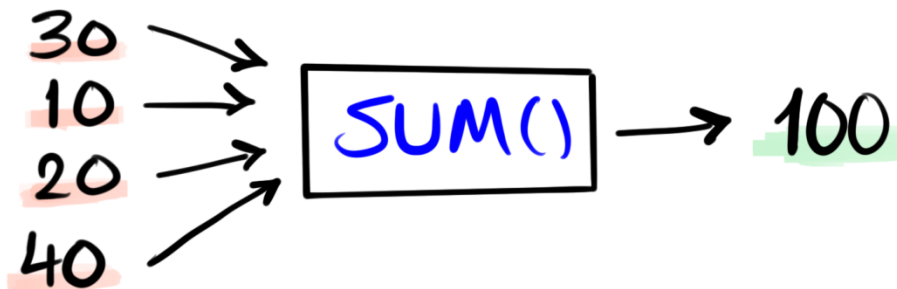
What are Functions?



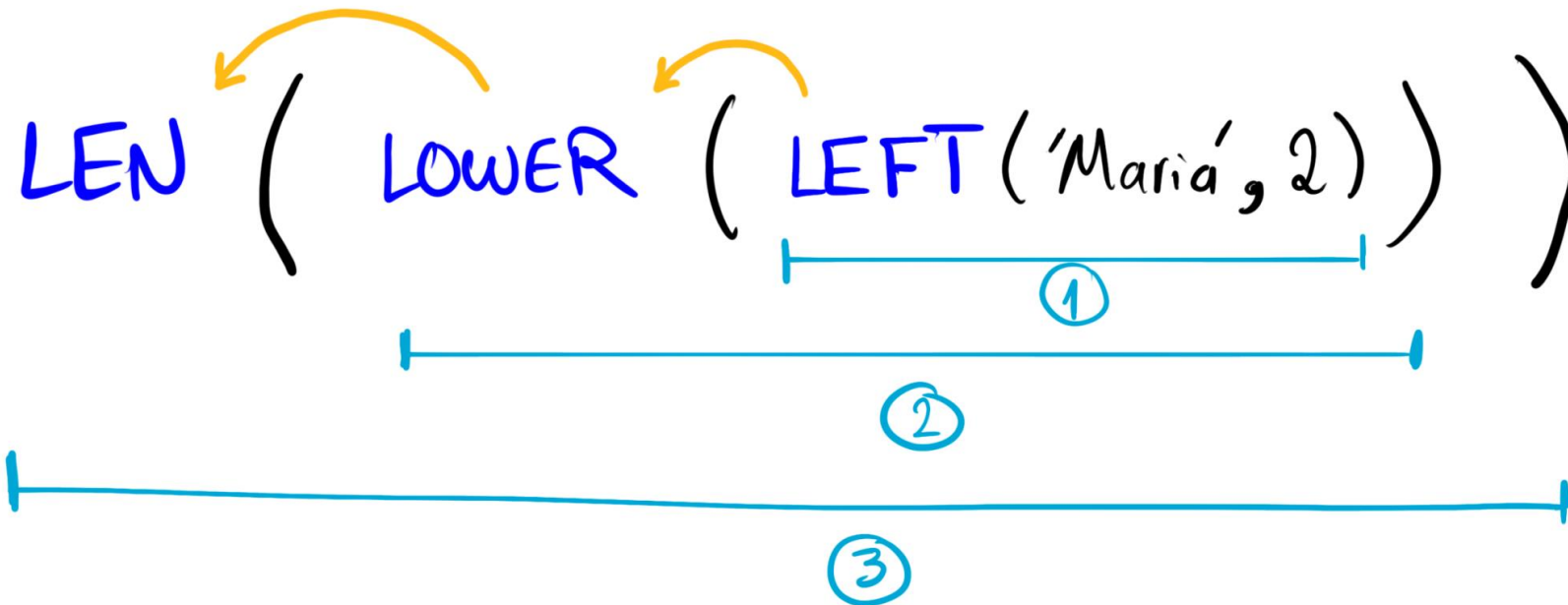
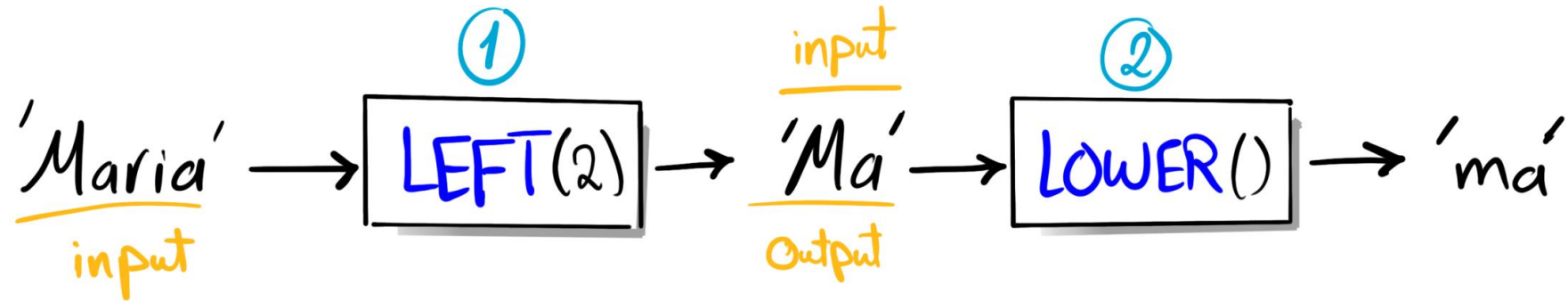
1 Single-Row Functions

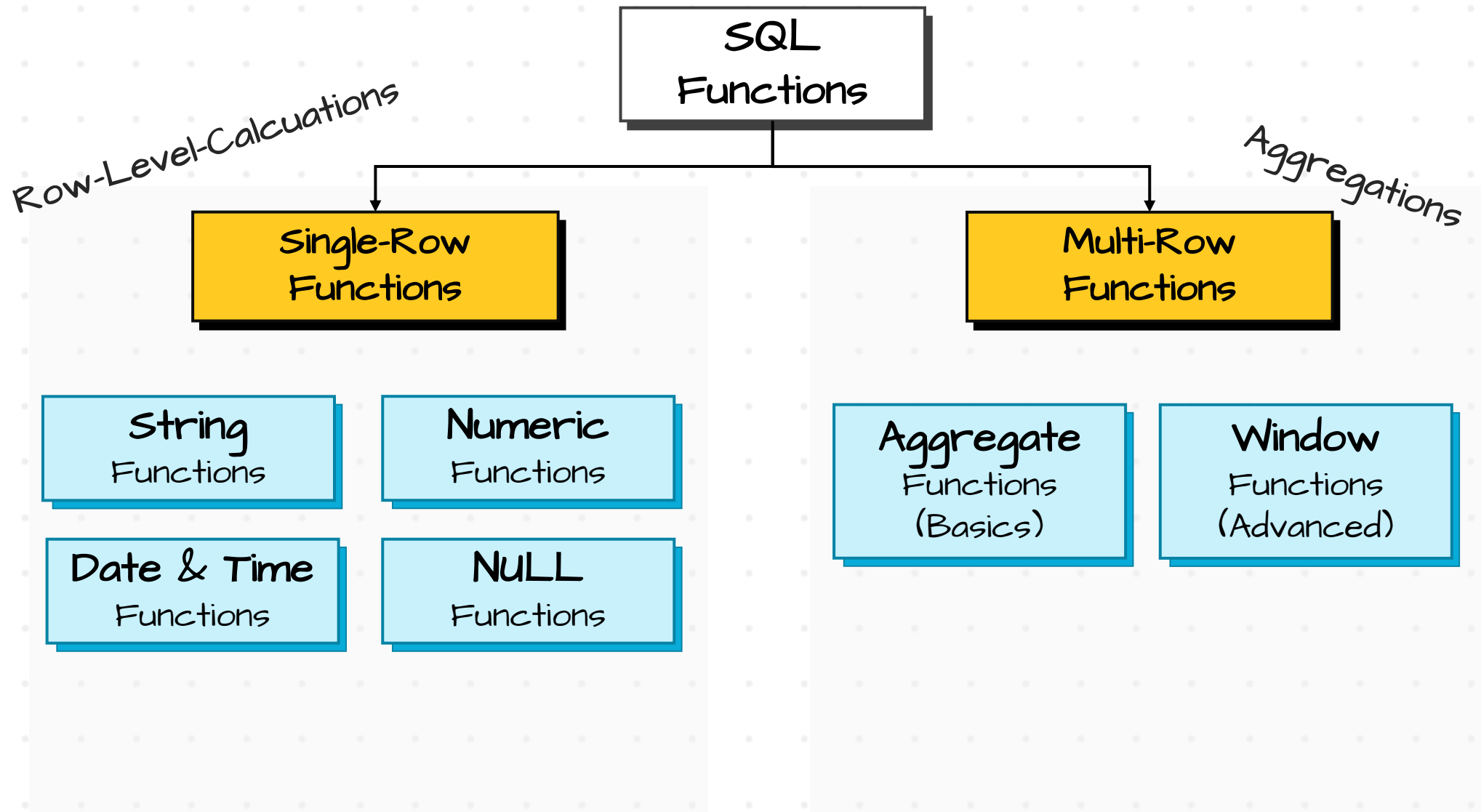


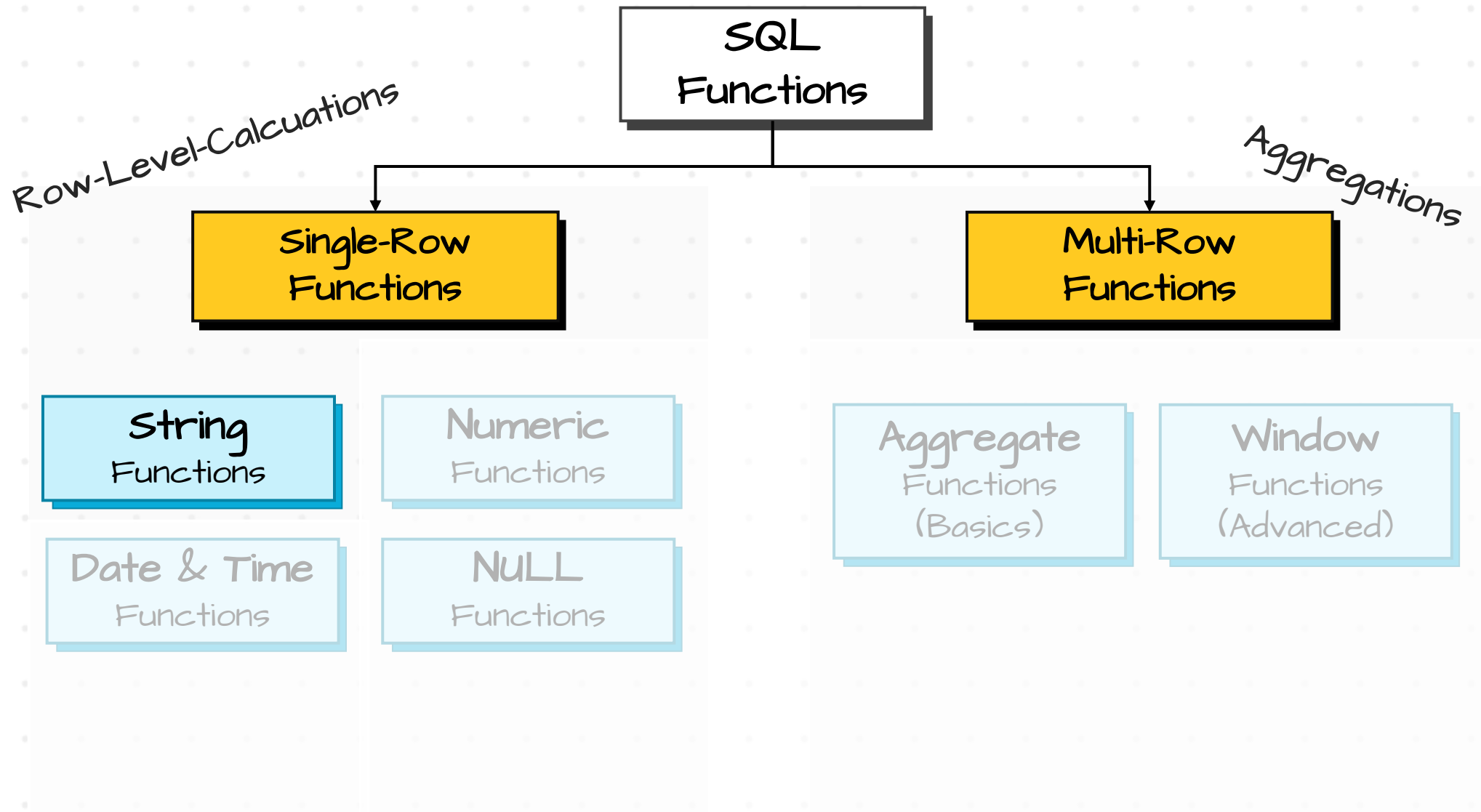
2 Multi-Row Functions



Nested Function







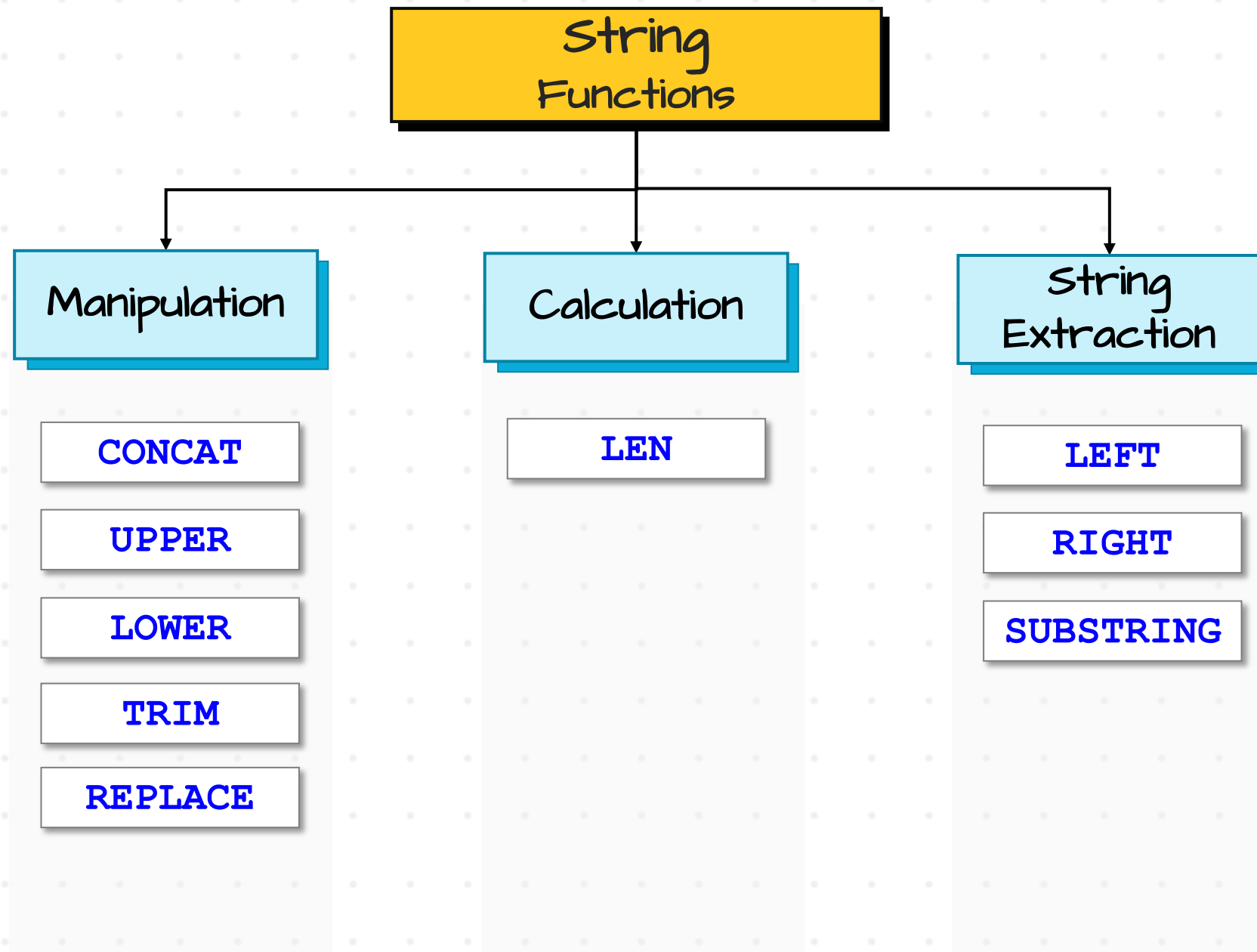


DATA WITH BARAA

STRING FUNCTIONS

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | String Functions





CONCAT

Combines multiple strings into one

UPPER

Converts all characters to uppercase

LOWER

Converts all characters to lowercase

TRIM

Removes Leading and Trailing spaces

REPLACE

Replaces specific character with a new character

LEN

Counts how many characters

LEFT

Extracts specific Number of Characters from the start

RIGHT

Extracts specific Number of Characters from the End

Substring

Extracts a part of string at a specified position

String Functions

CONCAT

First Name

Michael

Last Name

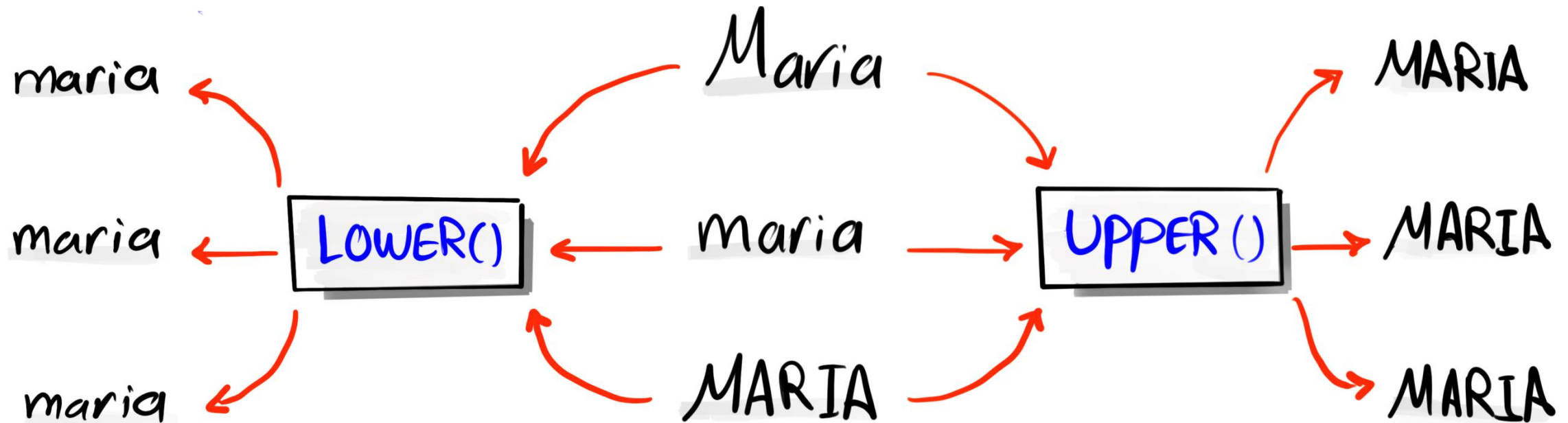
Scott

CONCAT()

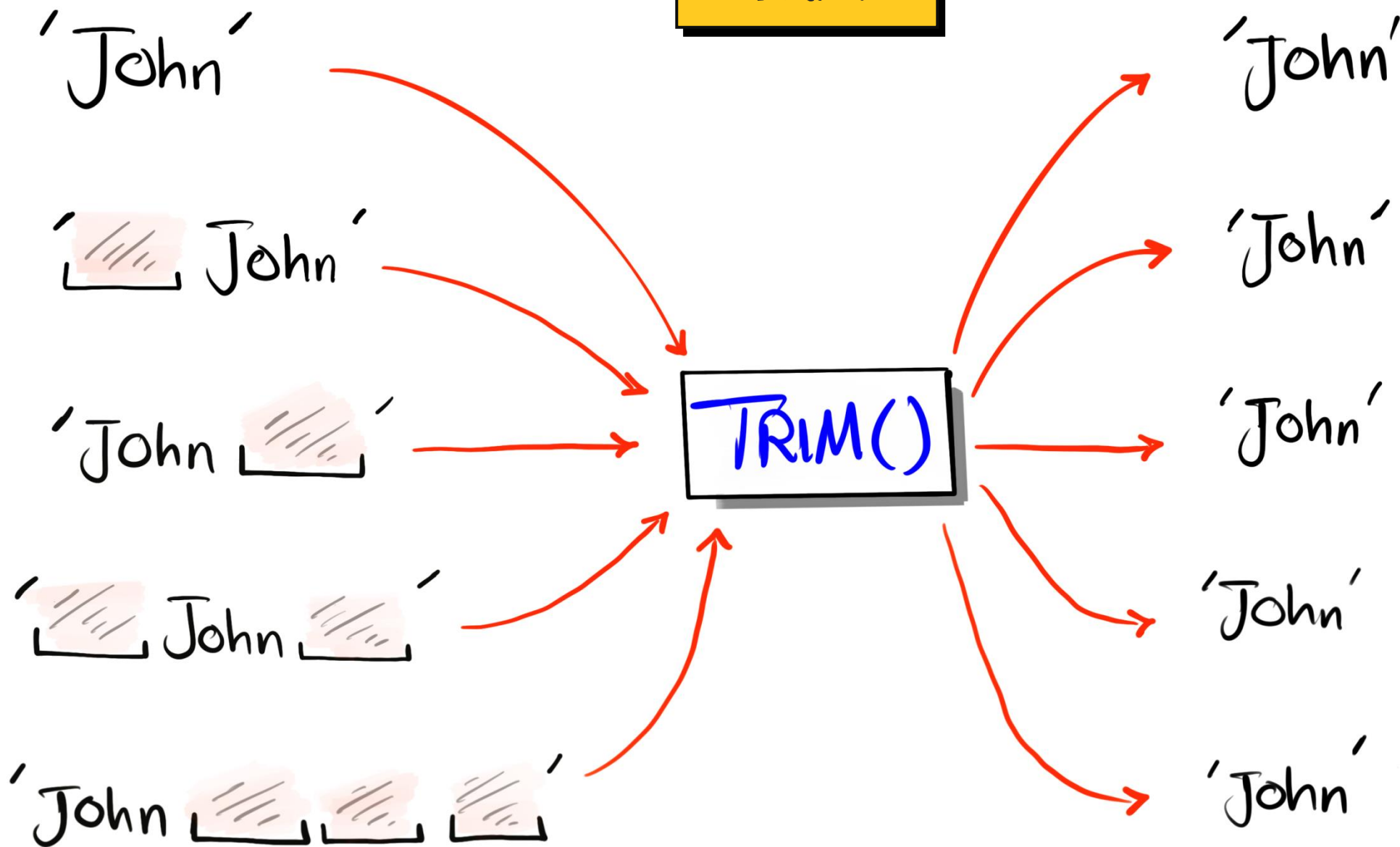
Michael Scott

Full Name

LOWER & UPPER



TRIM



REPLACE

Not only Replace but also Remove!

123 - 456 - 7890 →

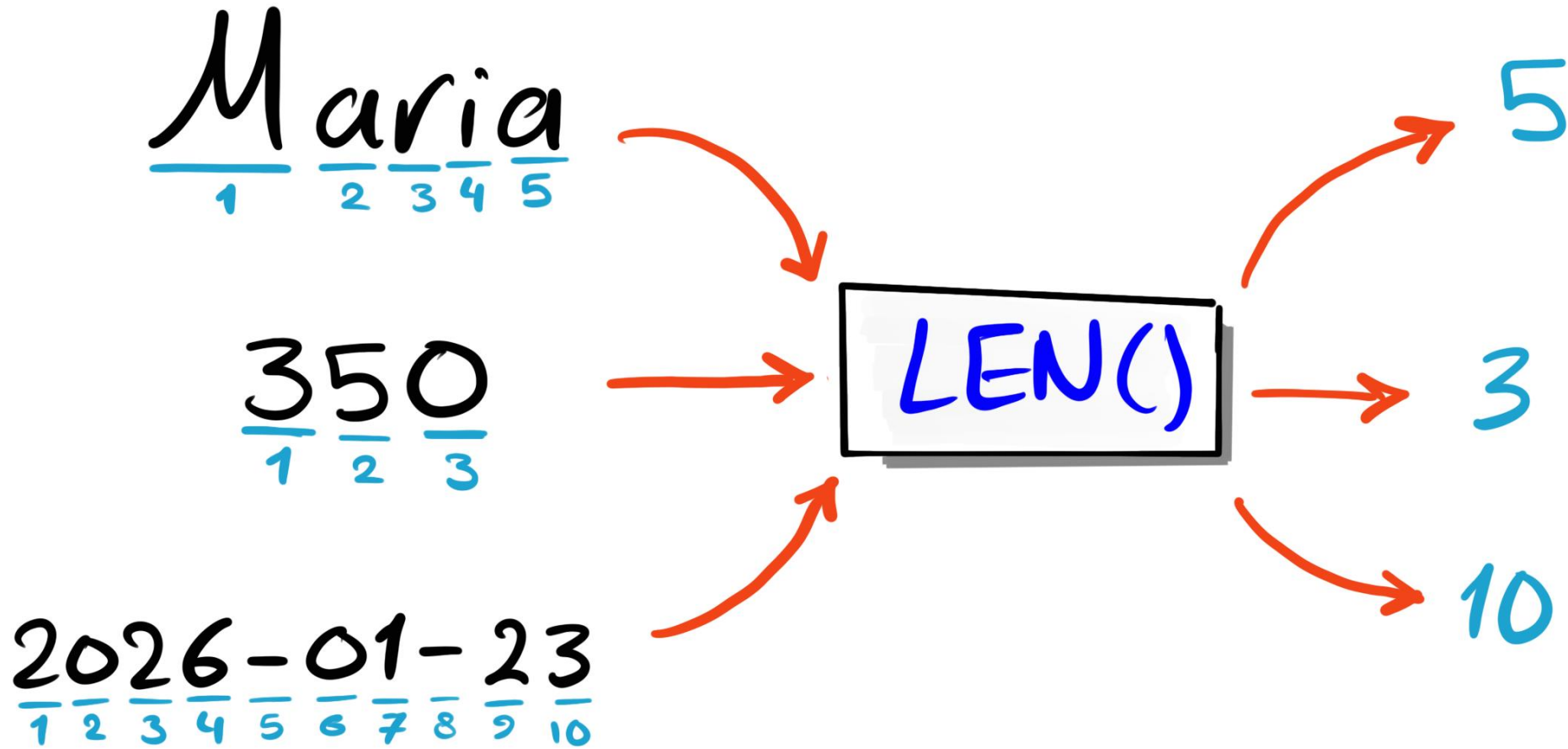
REPLACE

Old value = '-'

New Value = ''

→ 123 456 7890

Nothing ~ "Blank"



LEFT & RIGHT

LEFT (Value, Nr of Characters)

Extract
First 2 Characters

LEFT = 2

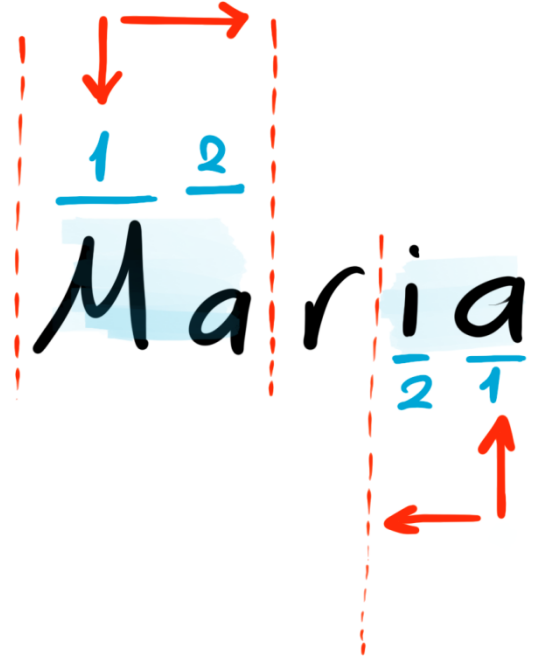
Ma

RIGHT (Value, Nr of Characters)

Extract
Last 2 Characters

RIGHT = 2

ia



SUBSTRING (Value, Start, Length)

After the 2nd Character extract All Characters

SUBSTRING

1 2 3 →
M a r i a
1 2 3
1 2 3 4 →
M a r t i n
3

Start
3

Length
LEN()
5
6
10

Dynamic

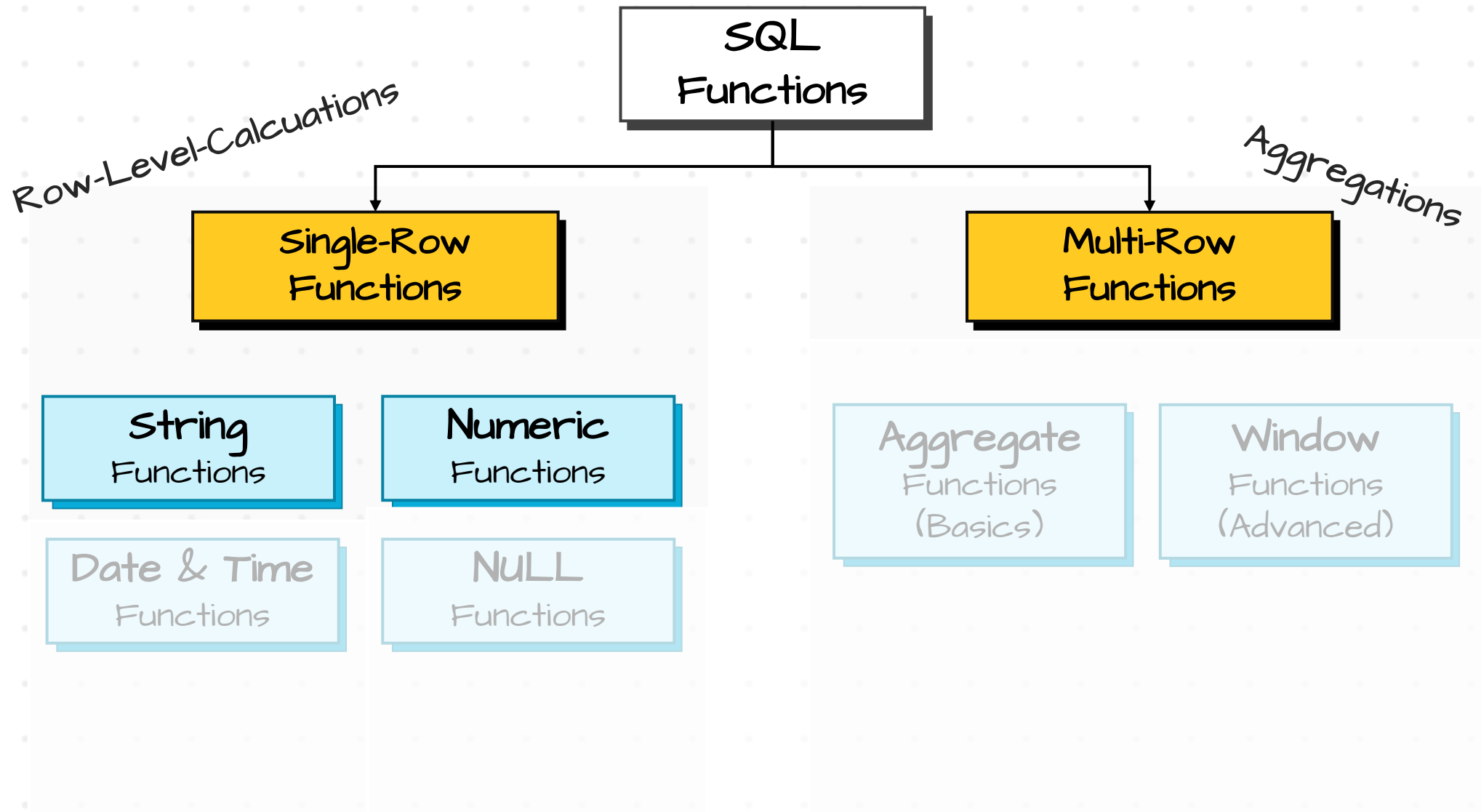


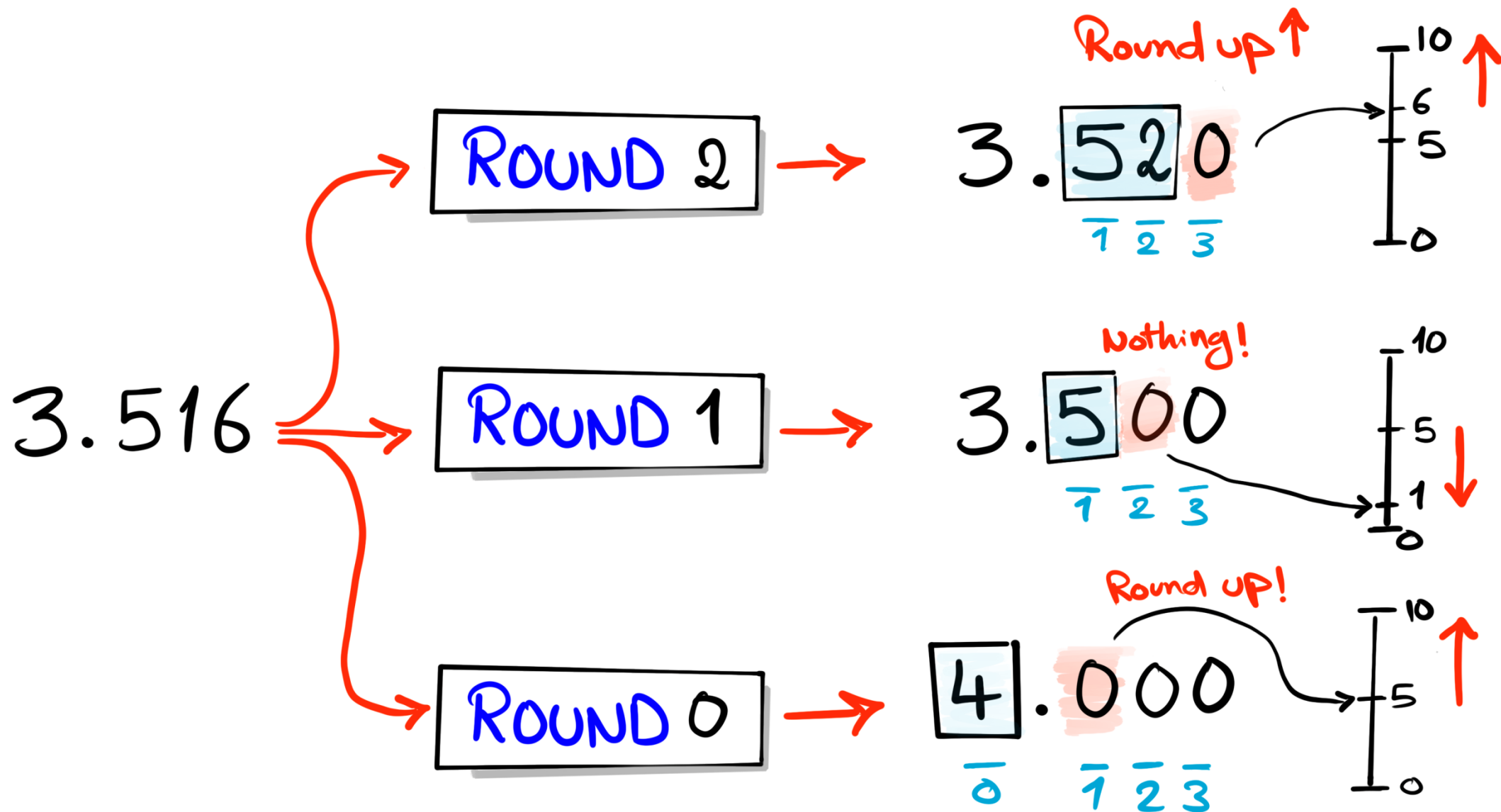
DATA WITH BARAA

NUMERIC FUNCTIONS

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | Number Functions







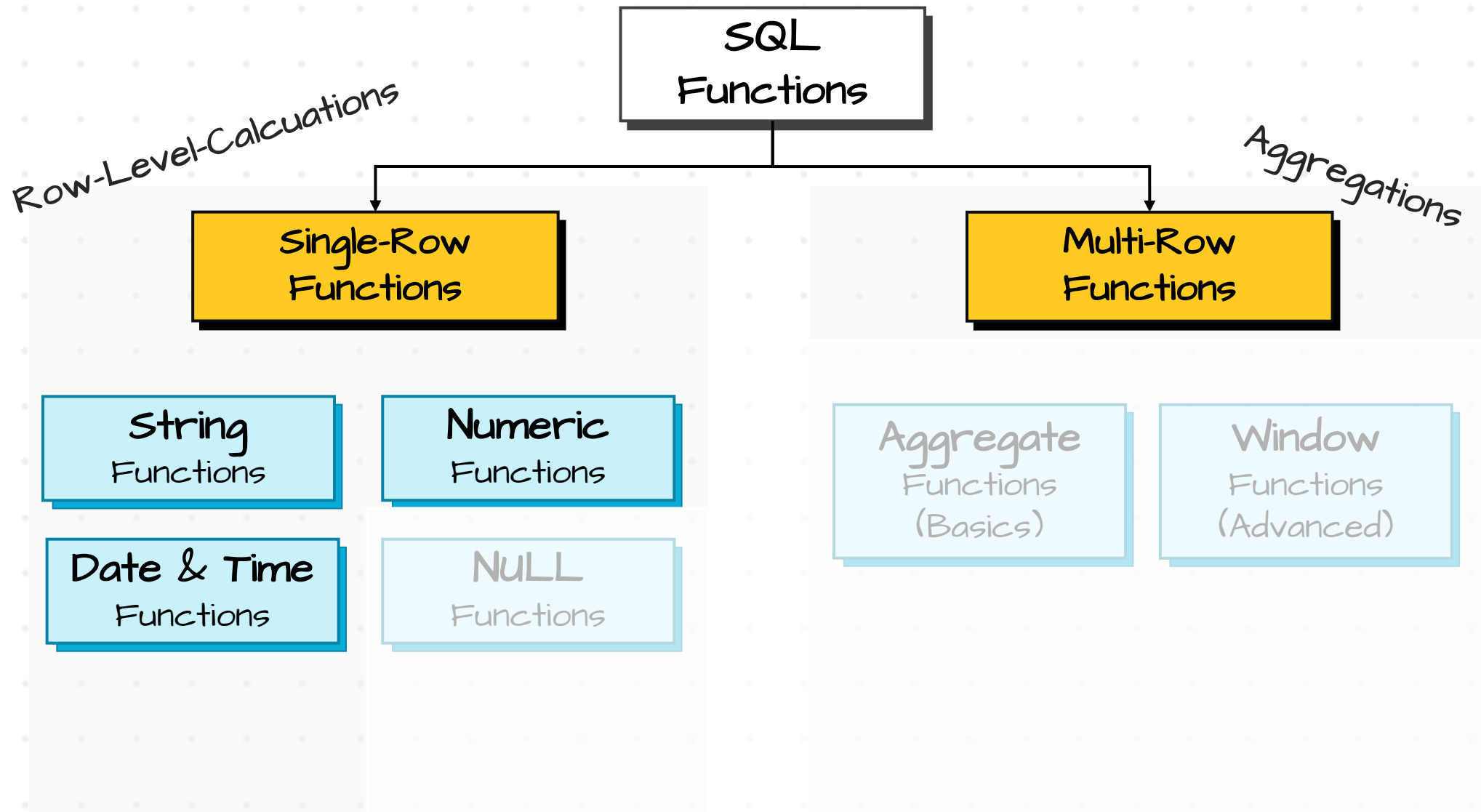


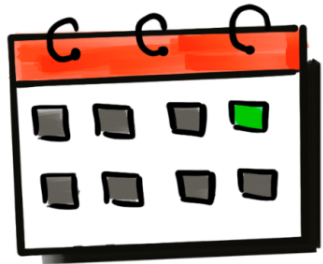
DATA WITH BARAA

DATE & TIME FUNCTIONS

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | Date & Time Functions







Date

2025 - 08 - 20

Year

Month

Day

Time

18 : 55 : 45

Hours

Minutes

Seconds

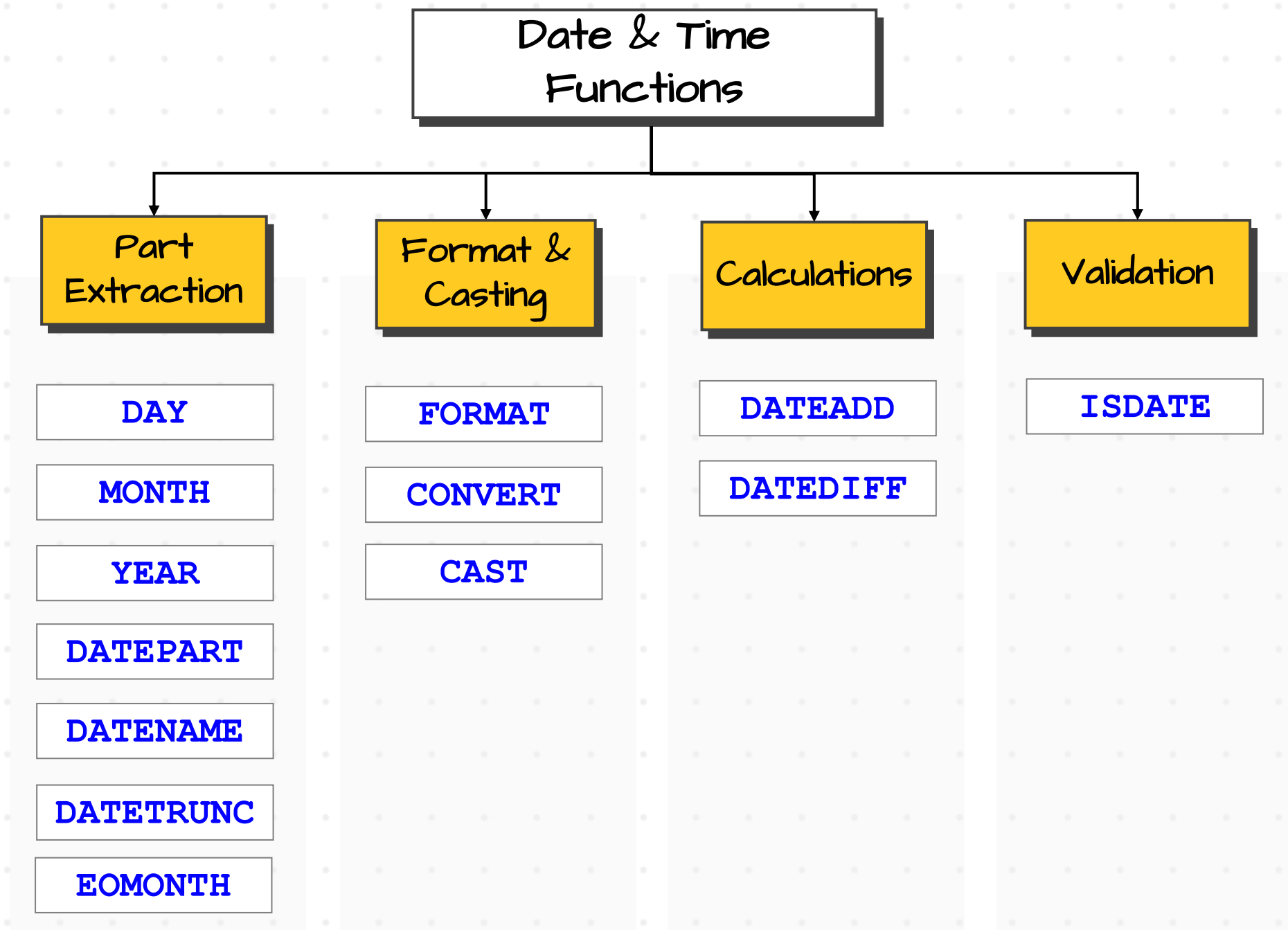


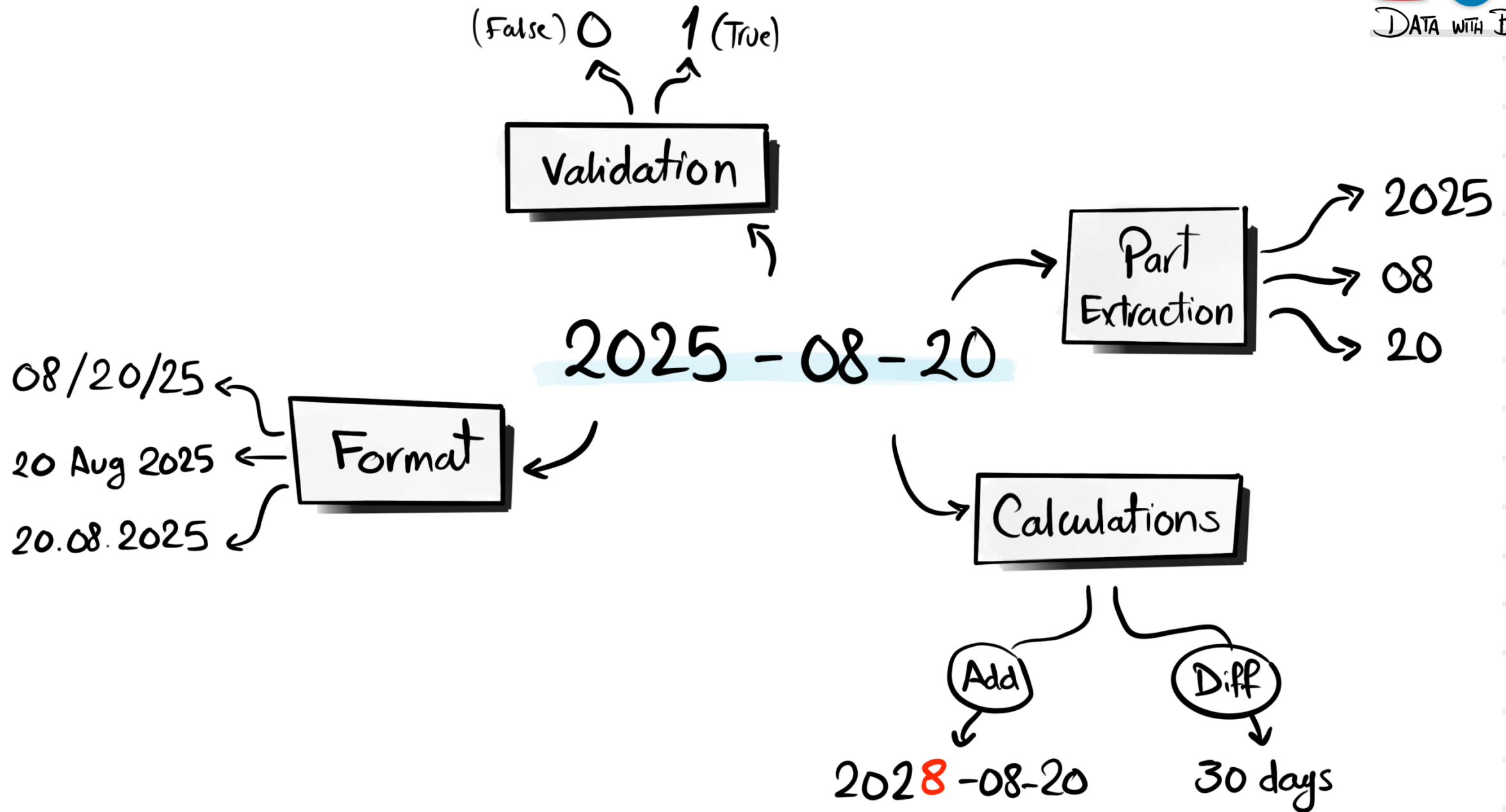
Timestamp

(Oracle, Postgres, MySQL)

Datetime2

(SQL Server)







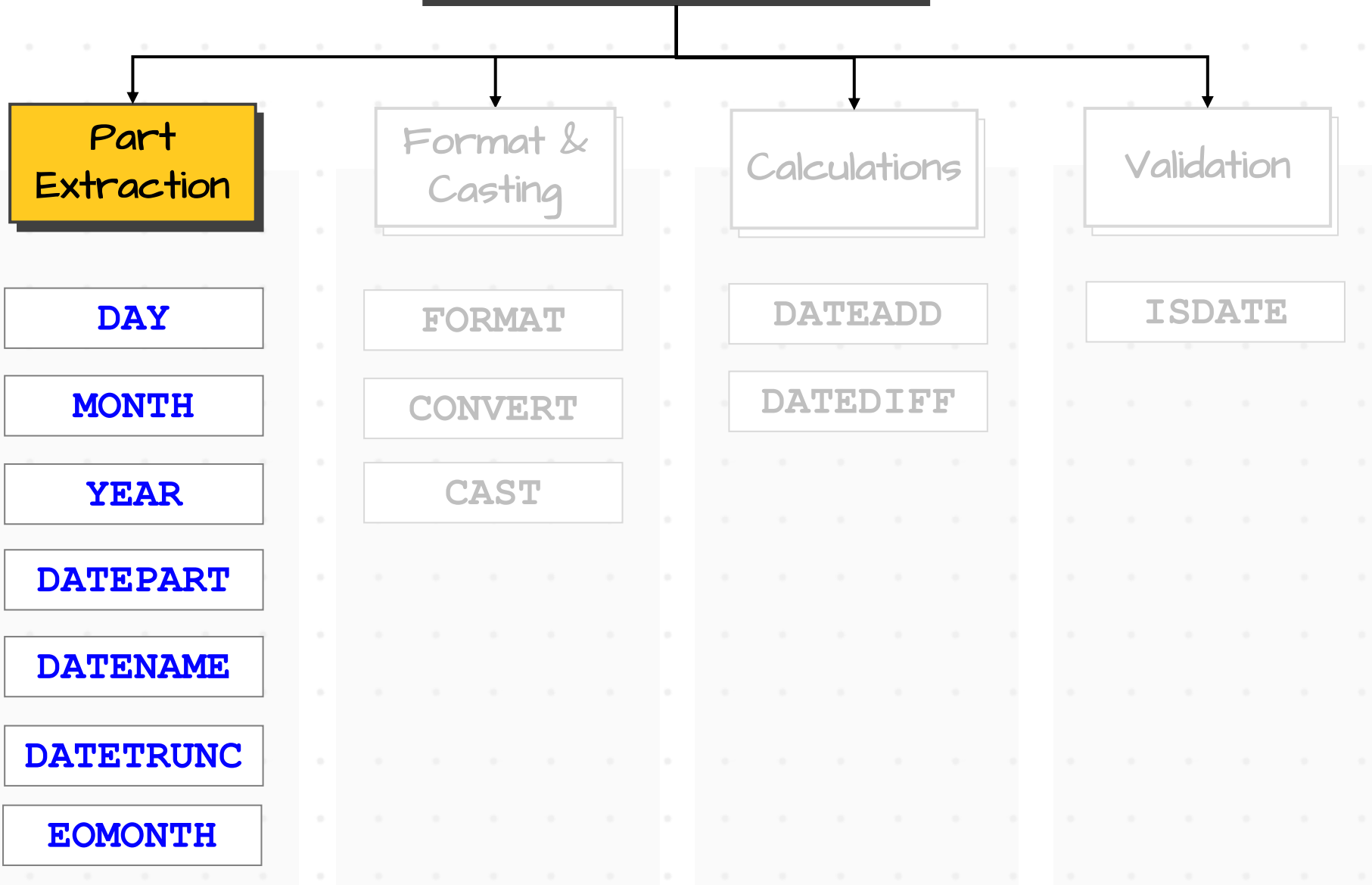
DATA WITH BARAA

PARTS EXTRACTION

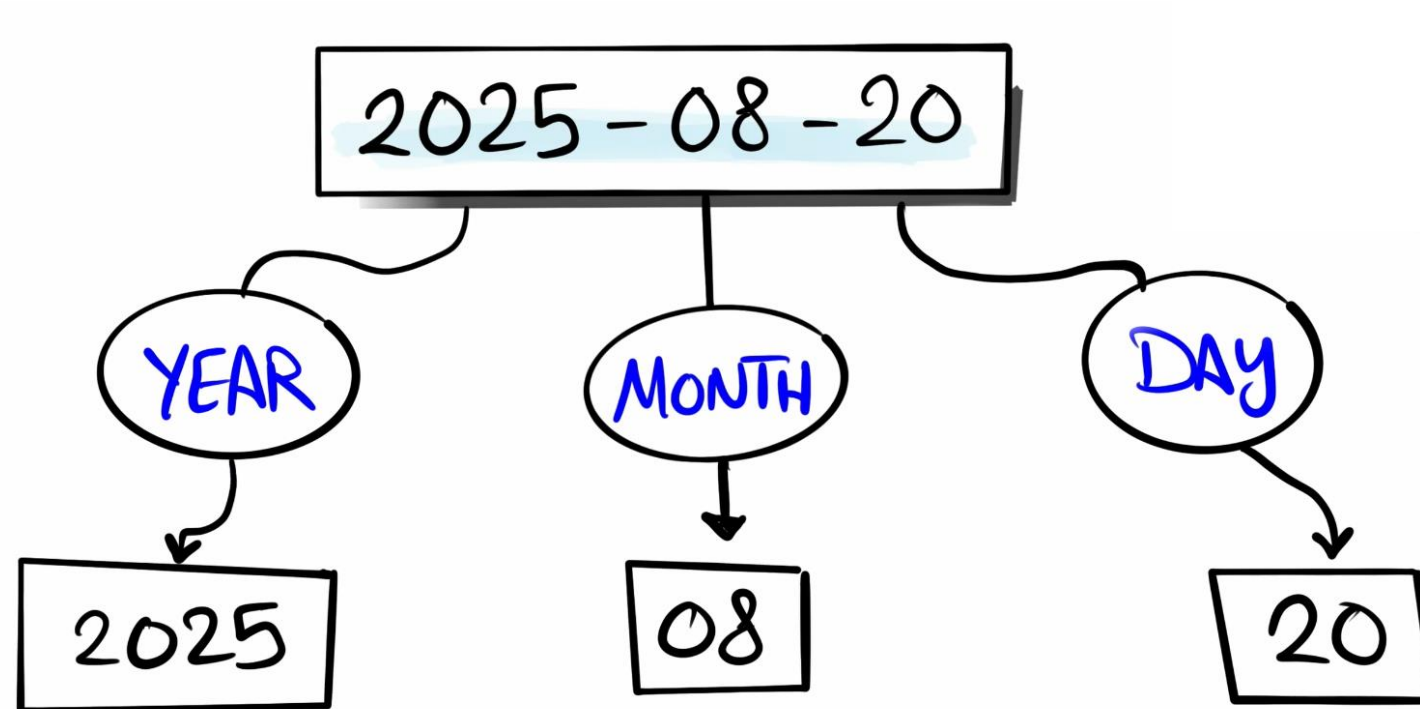
Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | Date & Time Functions



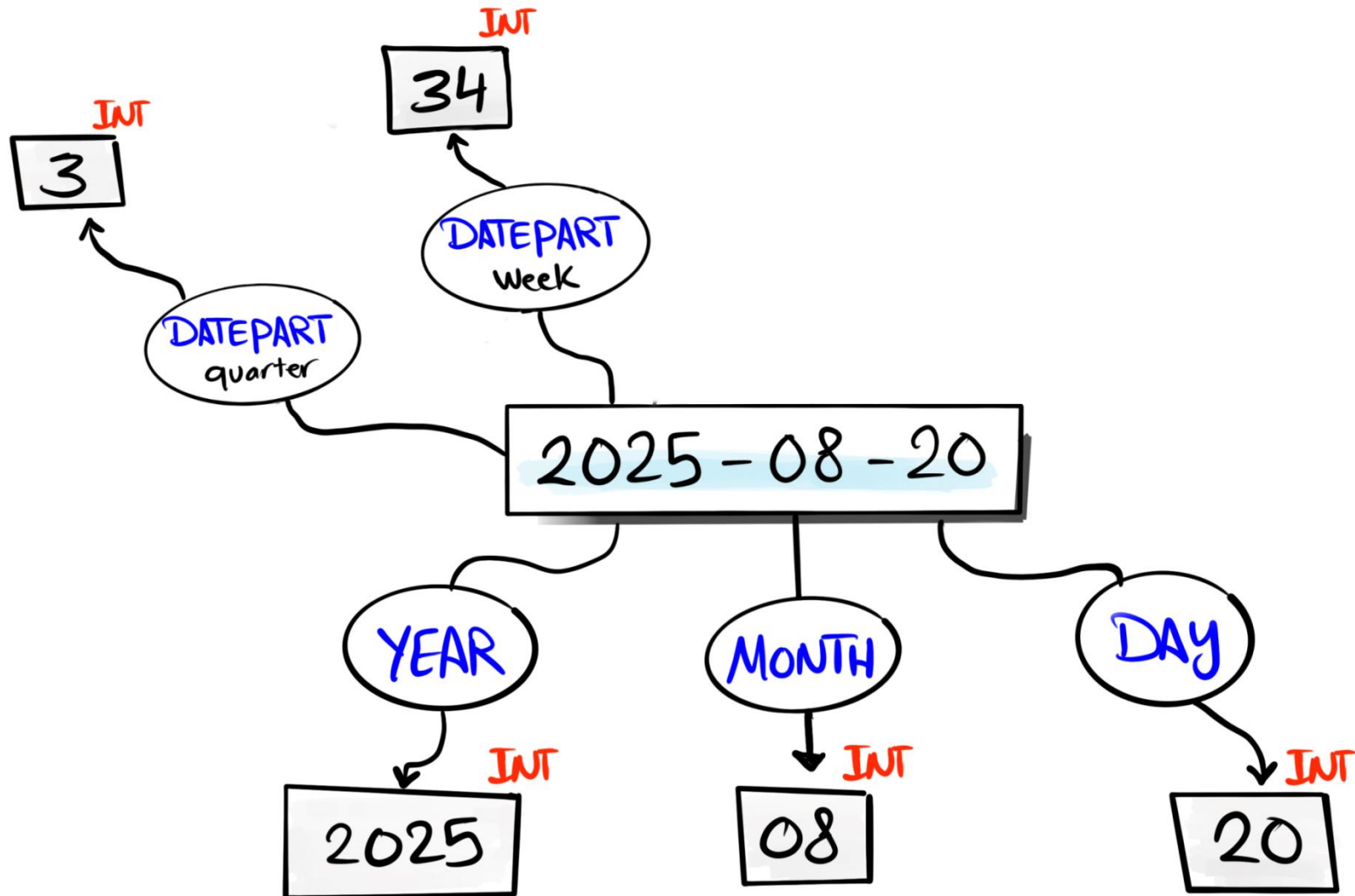
Date & Time Functions



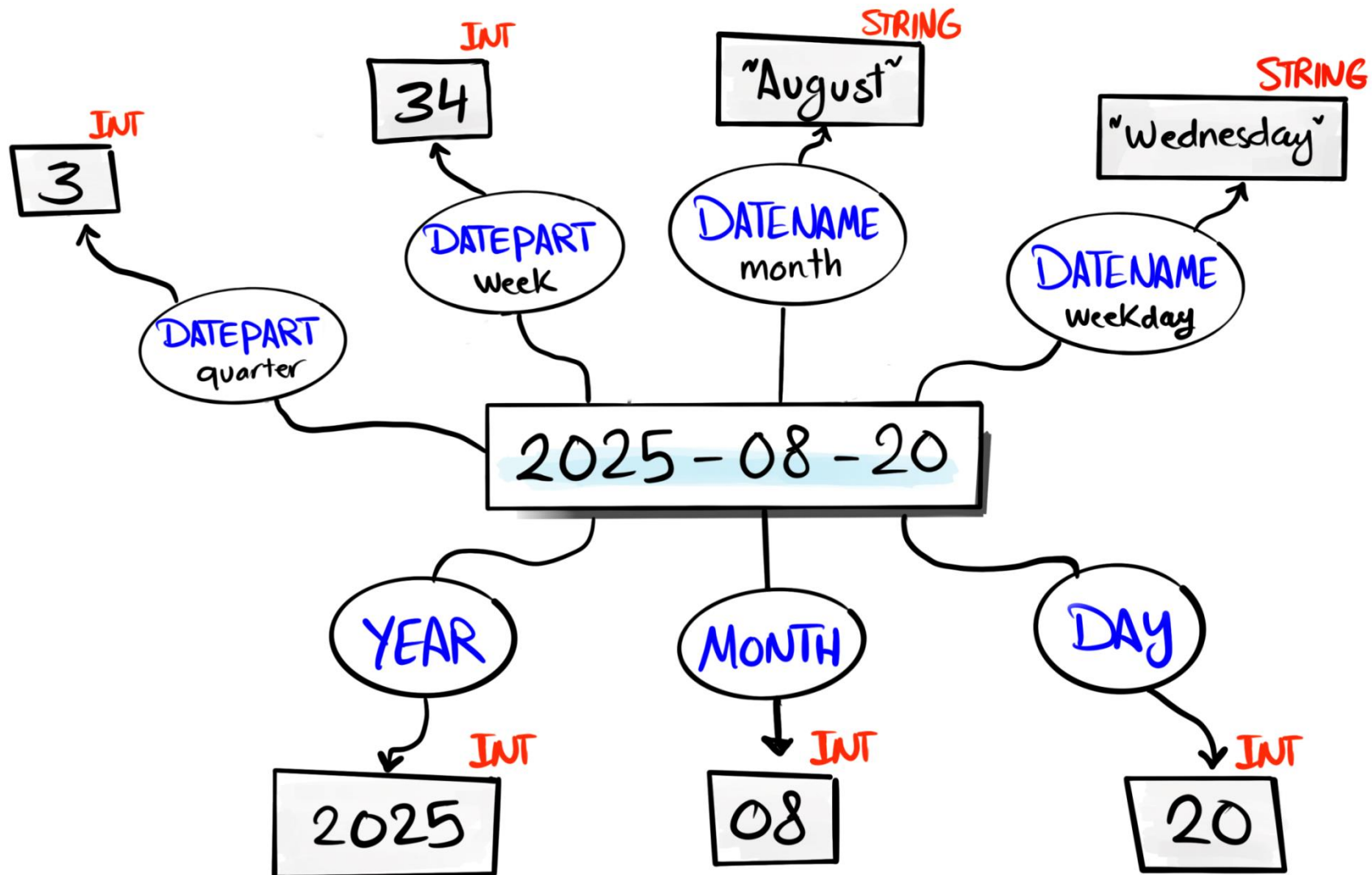
Quick Functions YEAR, MONTH, DAY



DATEPART



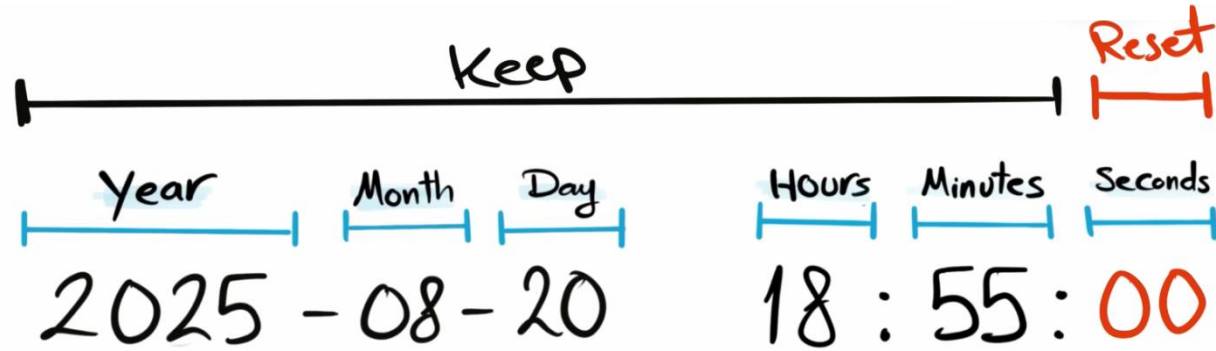
DATENAME



DATETRUNC

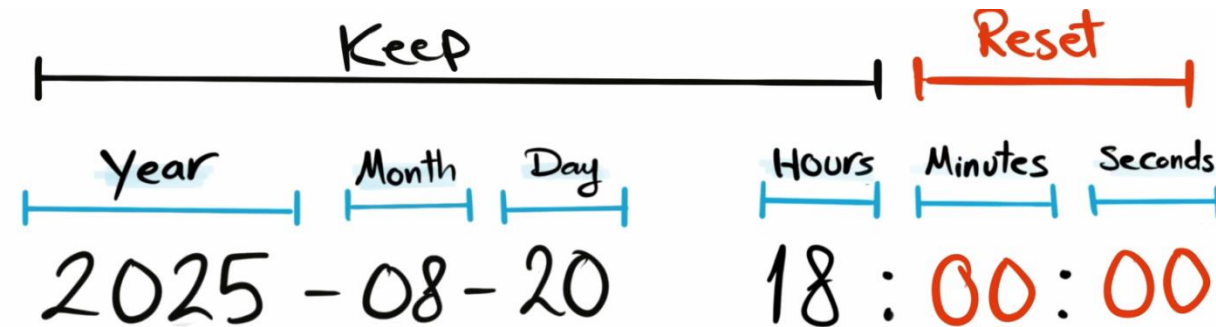
DATETRUNC

minute



DATETRUNC

hour



DATETRUNC

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

DATETRUNC

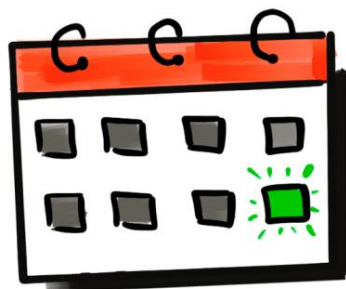
year

Keep			Reset		
Year	Month	Day	Hours	Minutes	Seconds
2025	01	01	00	00	00

Date part resets to 01

Time part resets to 00

EOMONTH



EOMONTH()



Year Month Day

2025 - 08 - 31

2025 - 02 - 28

2025 - 03 - 31

DATE

PART EXTRACTION

Syntax

DAY (*date*)

MONTH (*date*)

YEAR (*date*)

EOMONTH (*date*)

DATEPART (*part*, *date*)

DATENAME (*part*, *date*)

DATETRUNC (*part*, *date*)

DATA TYPES

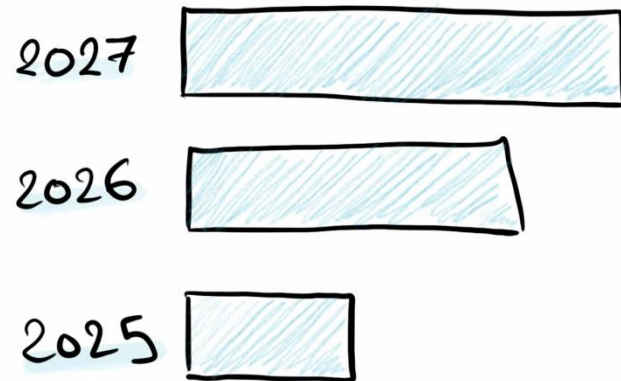
DATA TYPE

DAY MONTH YEAR DATEPART	➔	INT
DATENAME	➔	STRING
DATETRUNC	➔	DATETIME
EOMONTH	➔	DATE

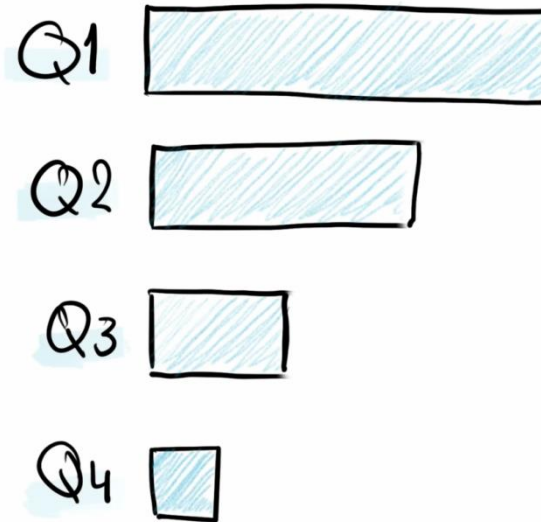
DATE PARTS

Date parts help to aggregate data at different levels of granularity (year, month, day, etc.).

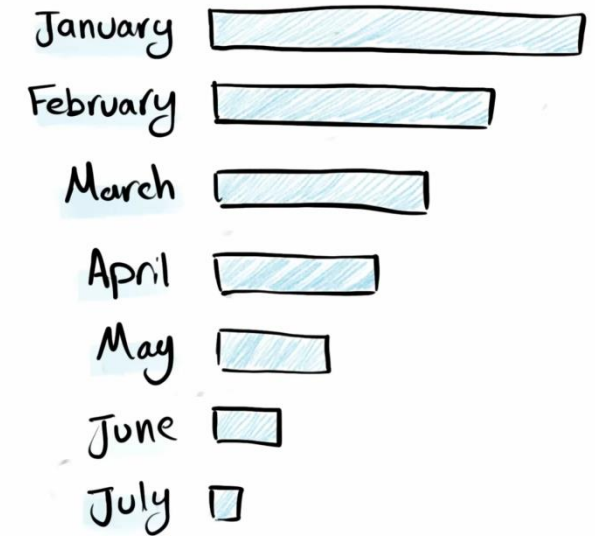
Report: Sales By year



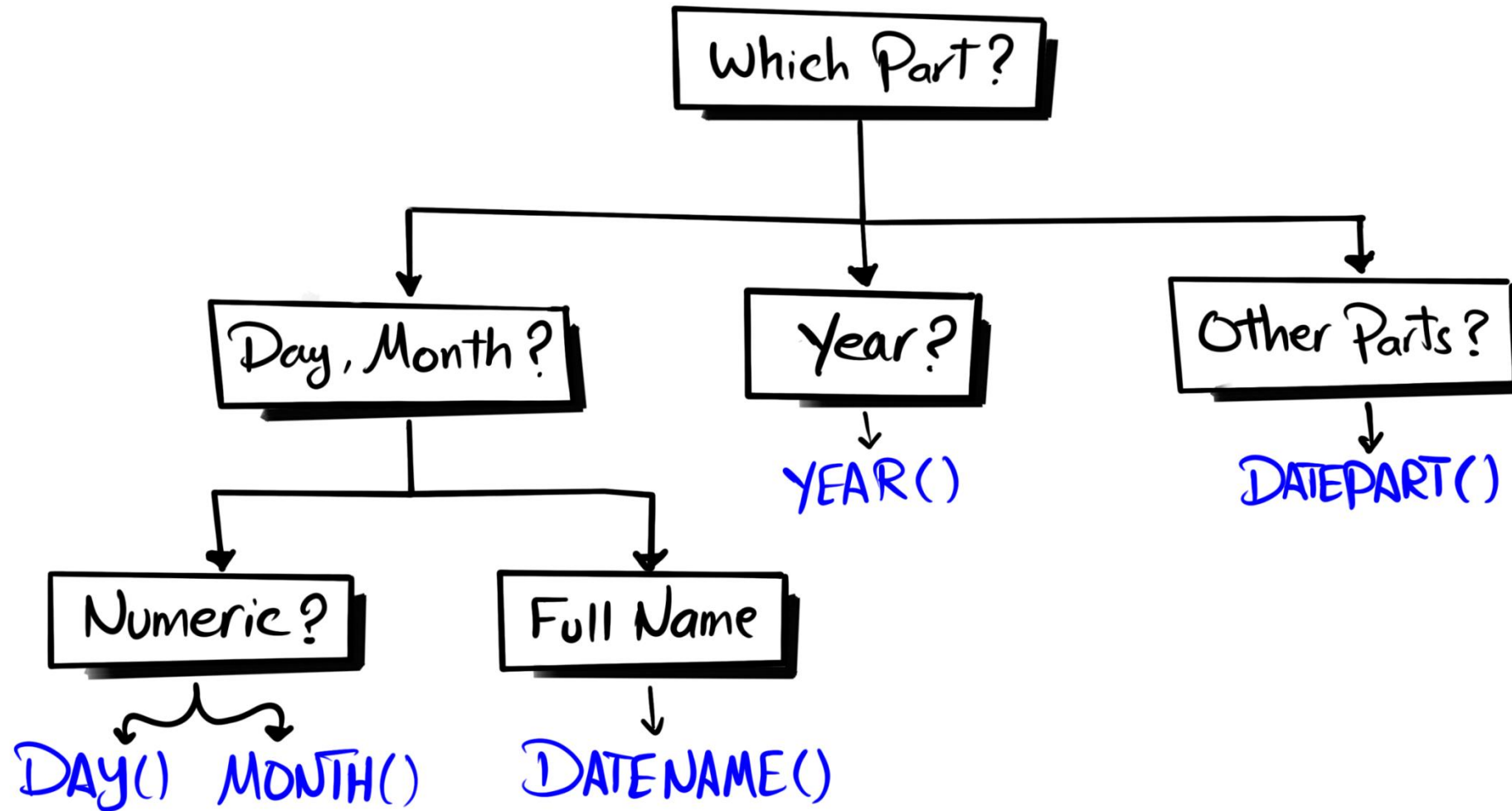
Report: Sales By year



Report: Sales By Month



How to Choose the Right Function?



DATEPART

DATENAME

DATETUNC

Part	Abbre.	INT	String	Datetime2
		DATEPART	DATENMAME	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

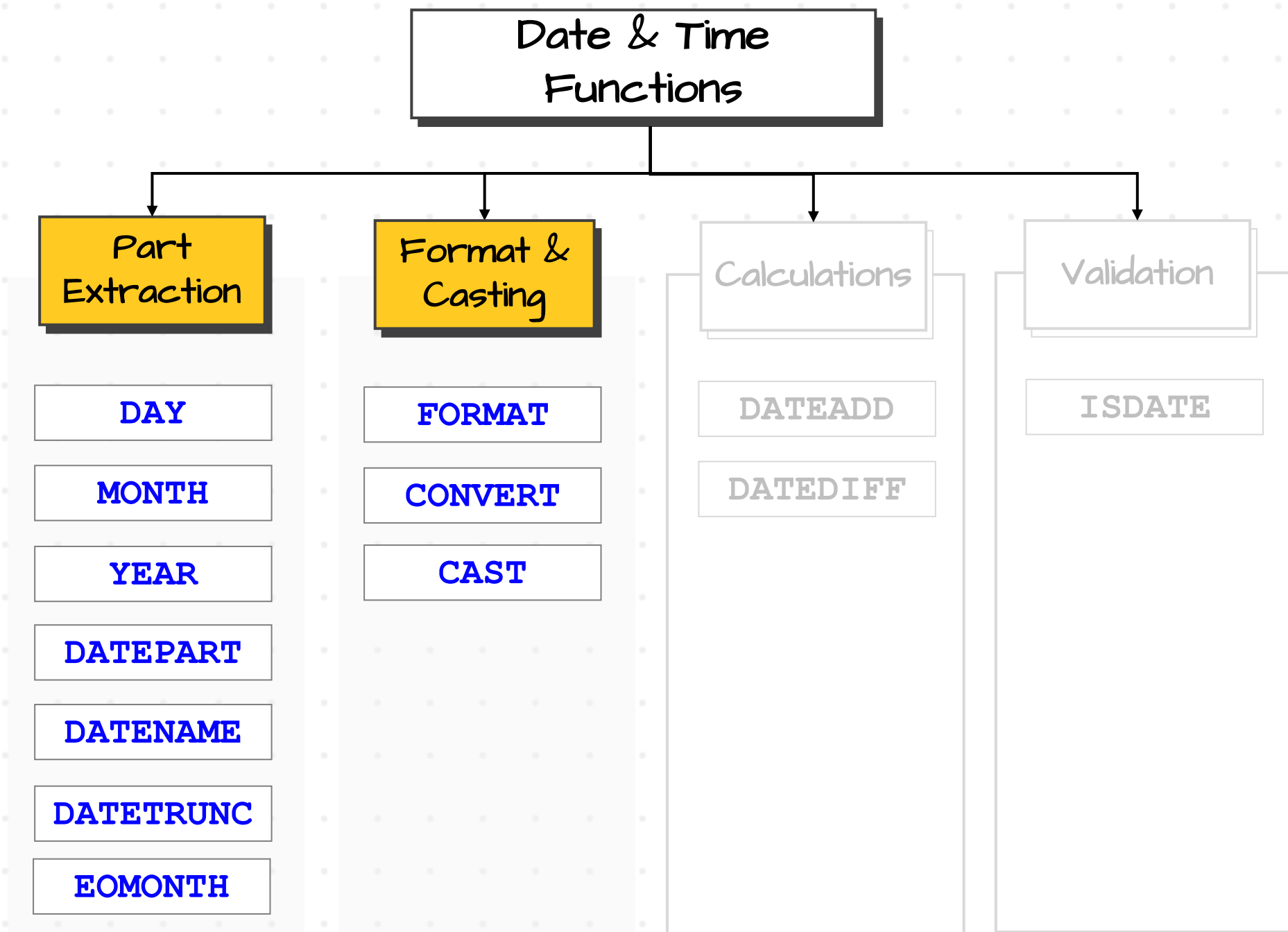


DATA WITH BARAA

DATE FORMATS

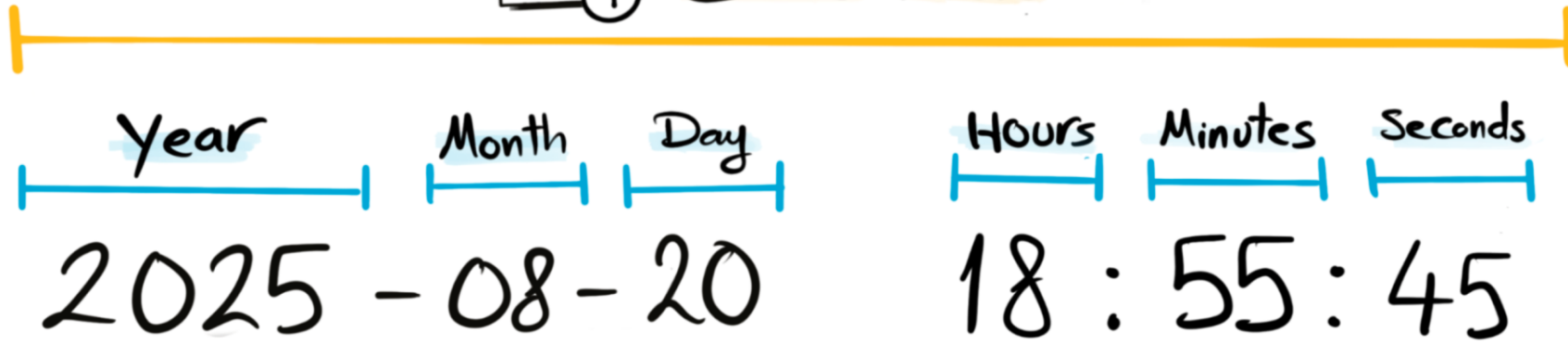
Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | Date & Time Functions







Datetime2



YYYY-MM-dd

HH:mm:ss

Format Specifier

Date & Time Format



2025 - 08 - 20
YYYY - MM - dd



International Standard (Iso 8601)

08 - 20 - 2025
MM - dd - YYYY



USA Standard

20 - 08 - 2025
dd - MM - YYYY



European Standard

FORMATING

Date

2025-08-20

FORMAT

MM/dd/yy

08/20/25

MMM yyyy

Aug 2025

CONVERT

6

20 Aug 25

style

112

20250820

String

Number

1234567.89

FORMAT

N

1,234,567.89

C

\$ 1,234,567.89

P

123,456,789.00%

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

Syntax

Syntax

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

Optional

Examples

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

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

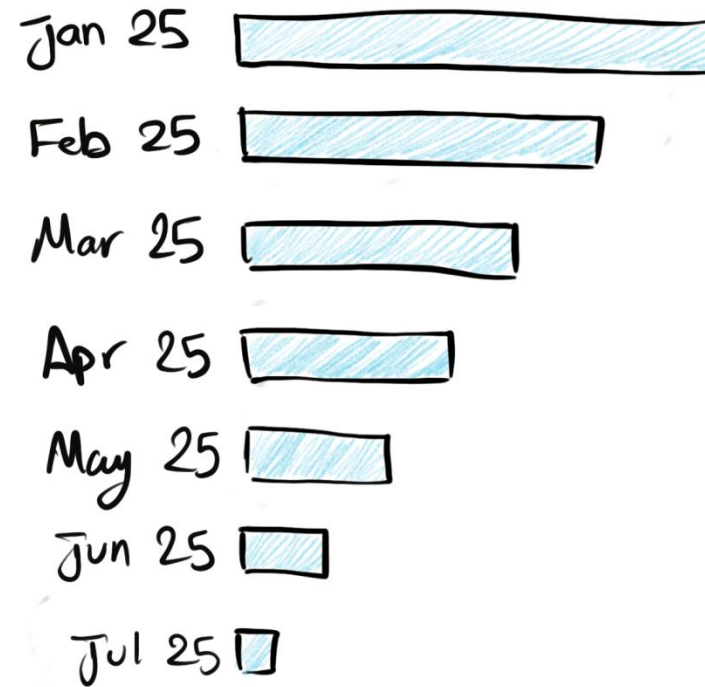
```
FORMAT (1234.56, 'D', 'fr-FR')
```

Default Cultuer = **en-US**

FORMAT

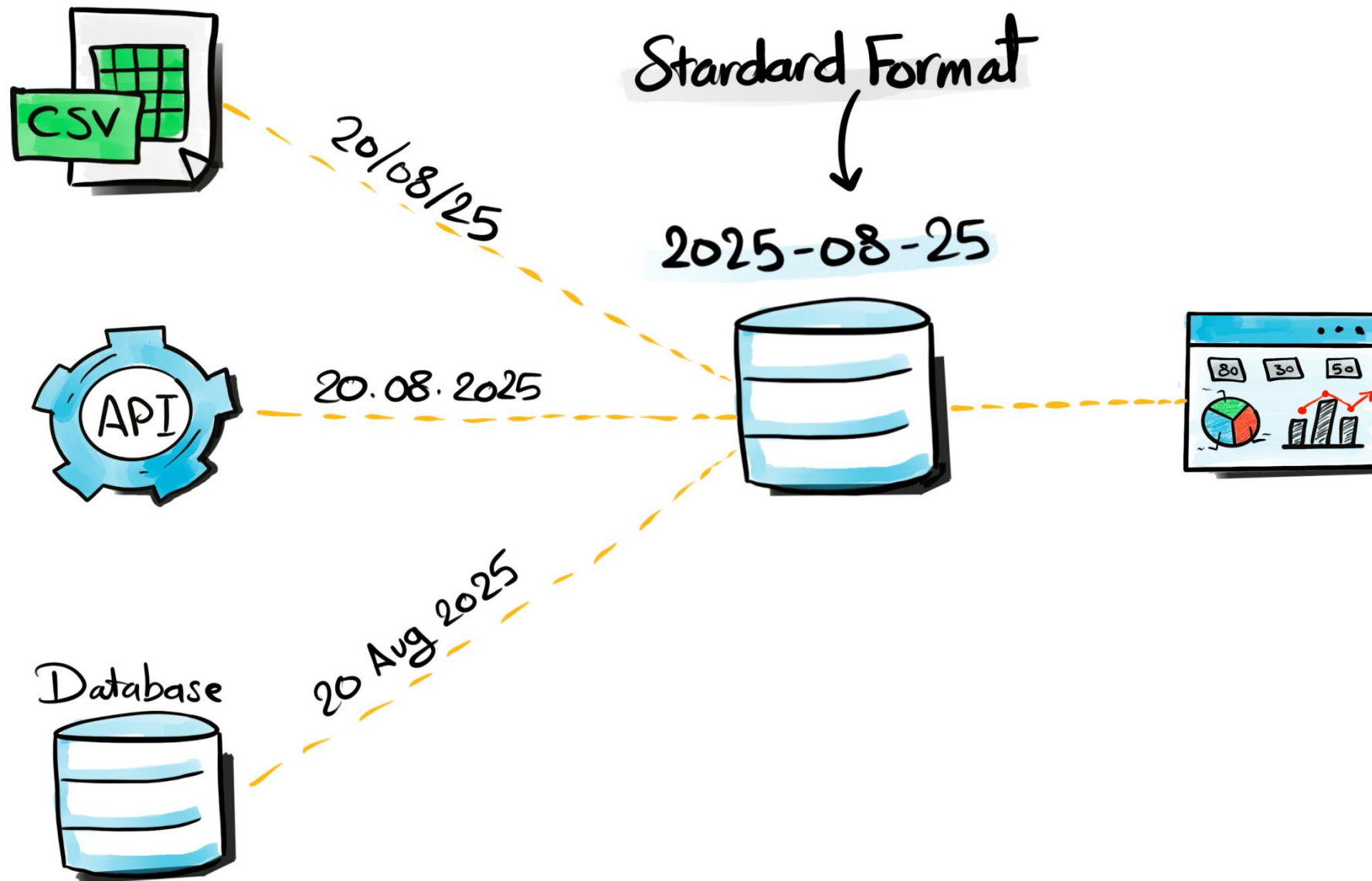
Use Case

Report: Sales By Month



FORMAT

Use Case



2025-08-20

18:55:45

Date & Time Format Specifiers

FORMAT

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

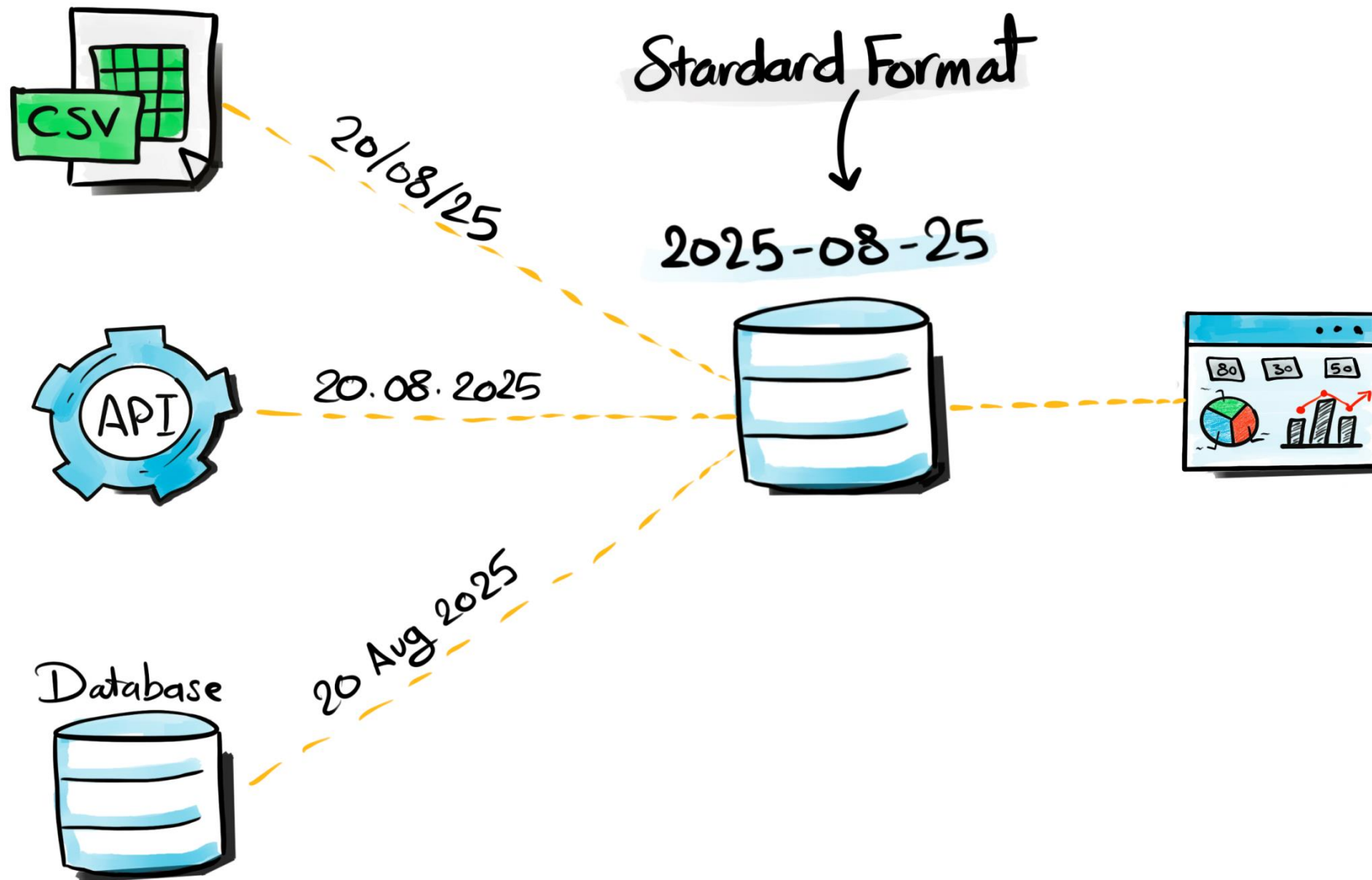
Number Format Specifiers

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

FORMAT

Use Case



CONVERT

Syntax

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

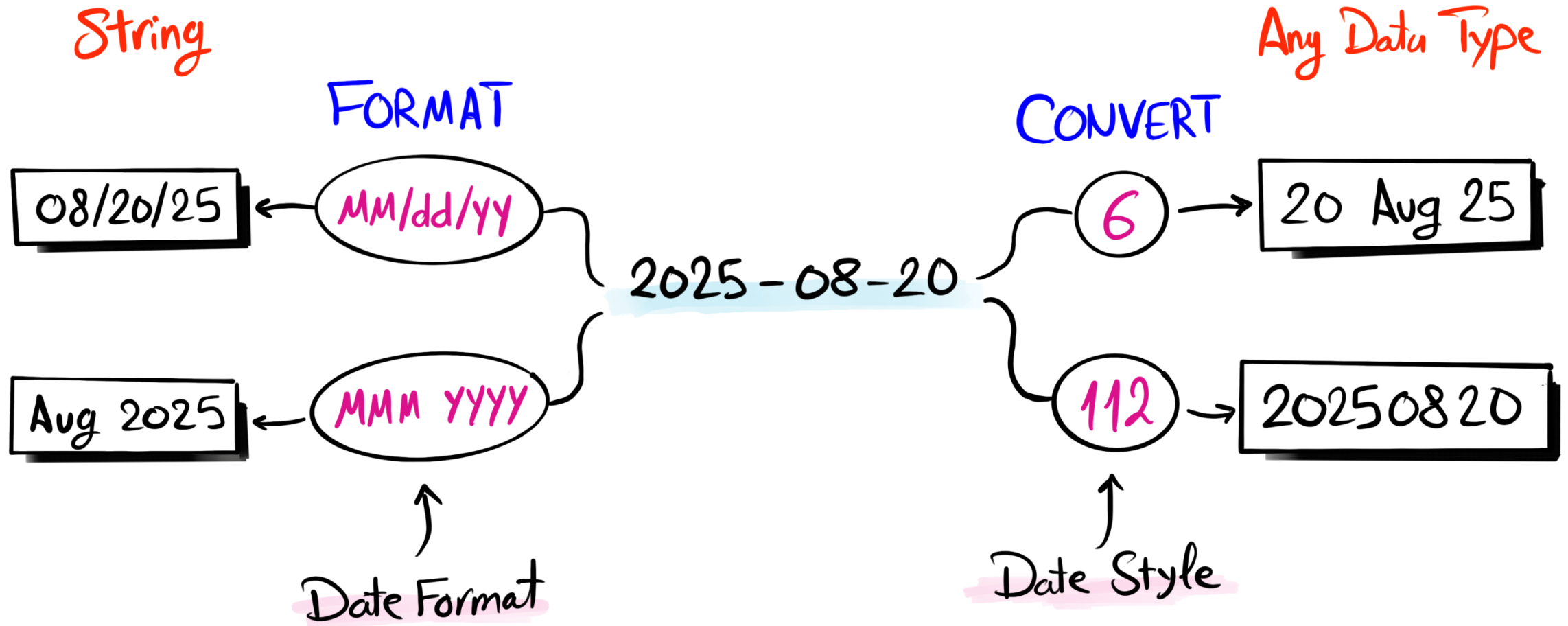
Optional

Examples

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

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

Default Style = 0



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

Syntax

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

Examples

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

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

No format can be specified

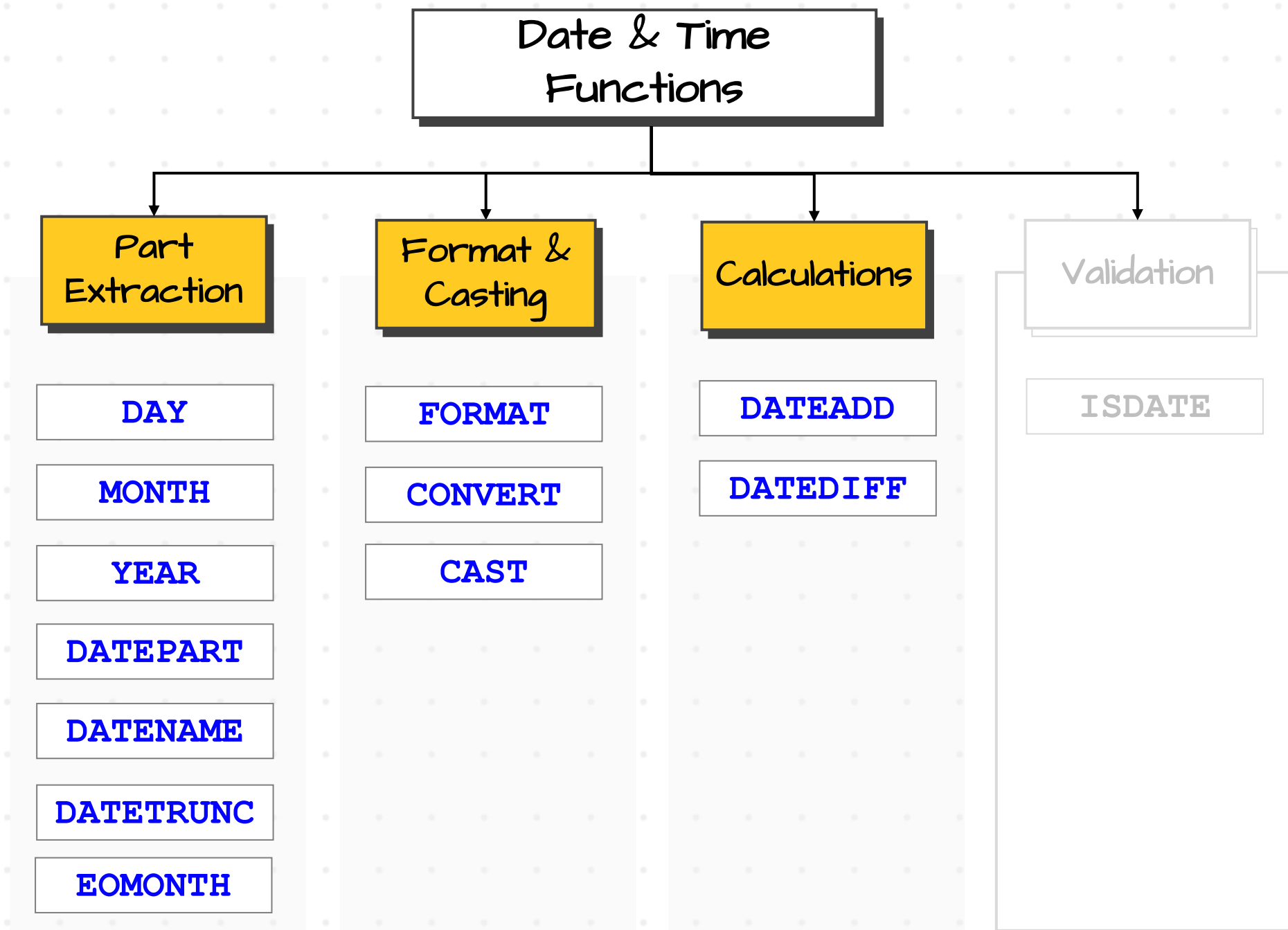
	CASTING	FORMATING
CAST	Any Type to Any Type	X No Formating
CONVERT	Any Type to Any Type	Formats only Date & Time
FORMAT	Any Type to Only String	Formats $\begin{matrix} \swarrow \searrow \\ \text{Date \& Time} \\ \text{Numbers} \end{matrix}$

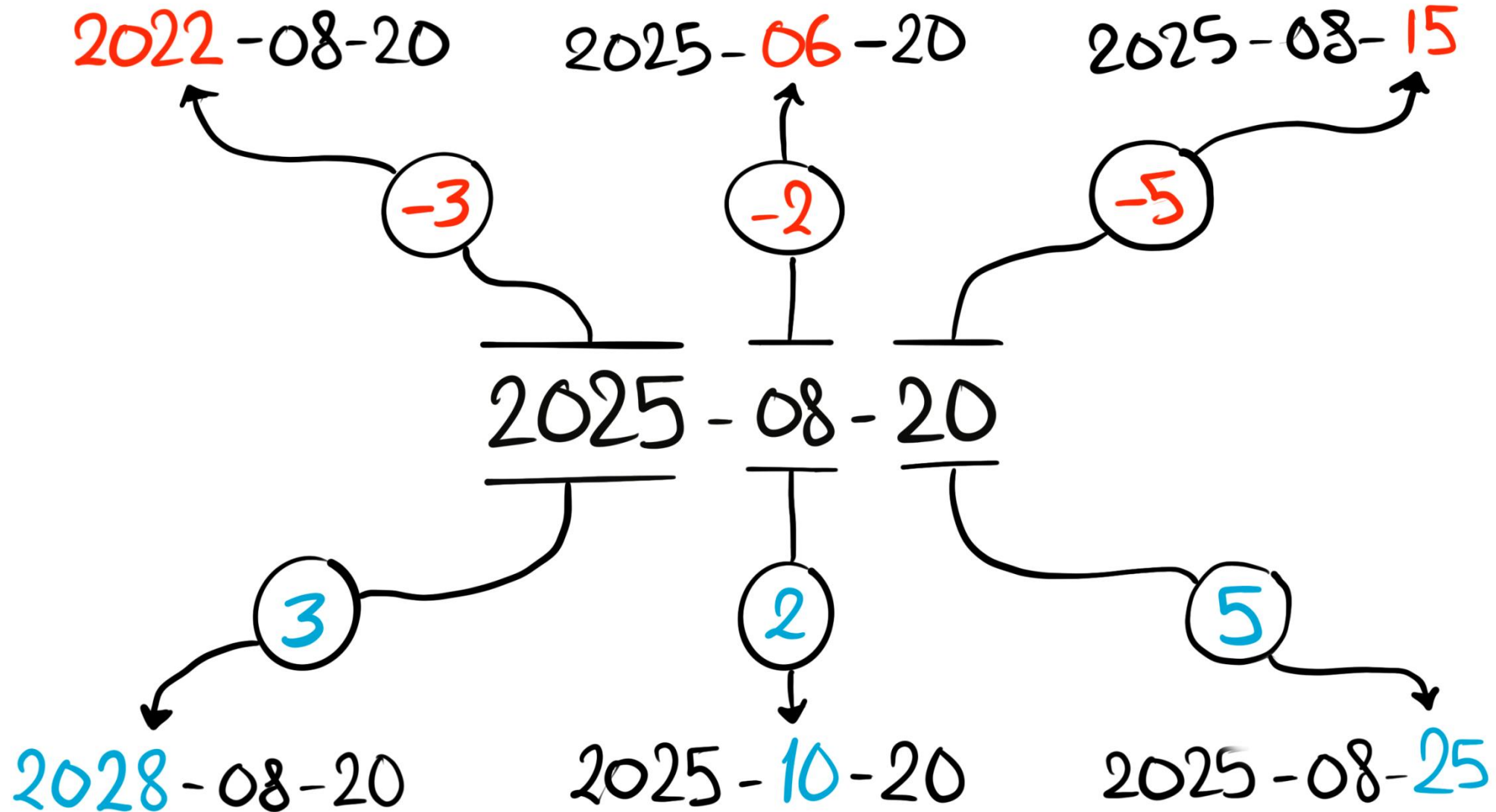


DATE CALCULATIONS

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | Date & Time Functions







DATEADD

Syntax

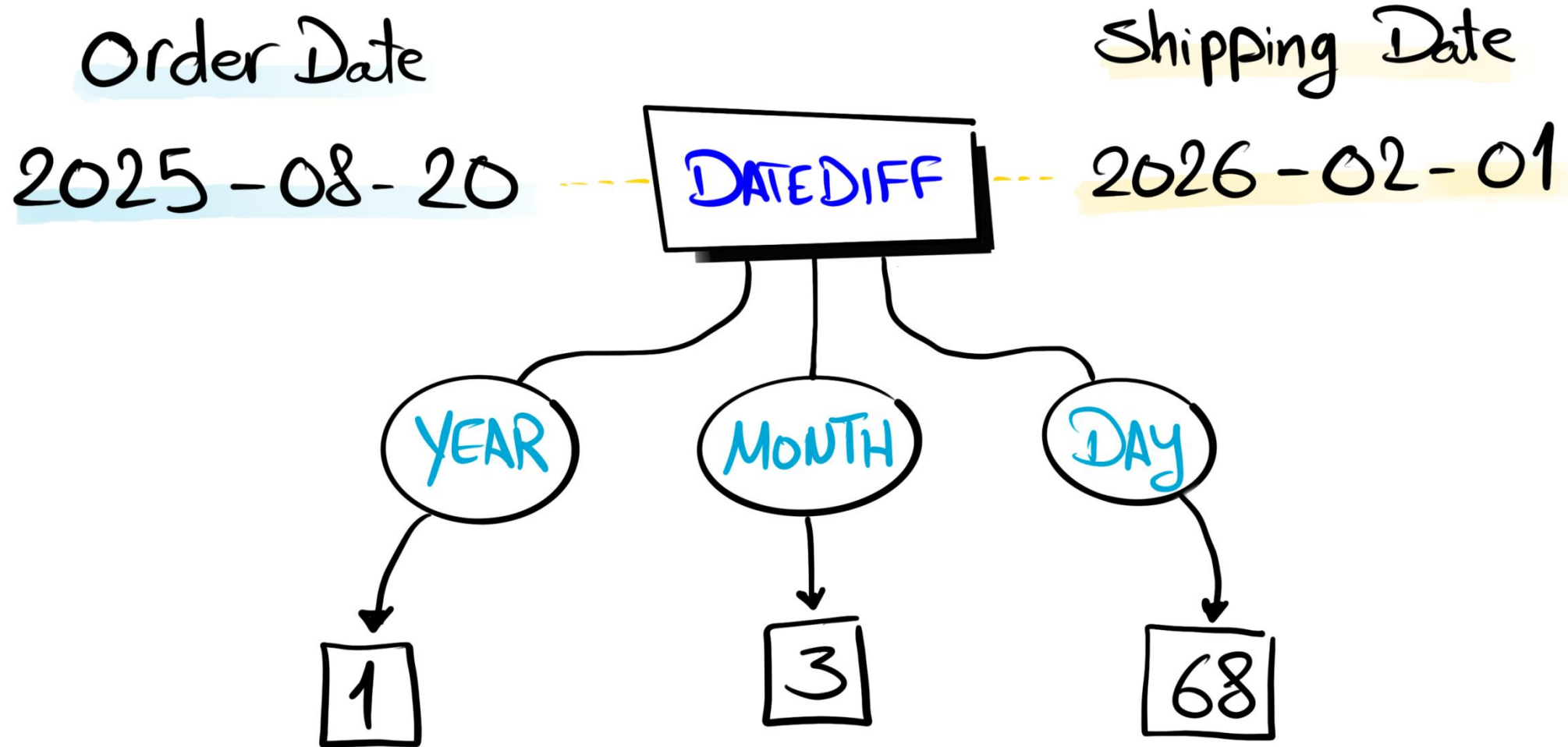
```
DATEADD (part, interval, date)
```

Examples

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

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


DATEDIFF



DATEDIFF

Syntax

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

Examples

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

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

ISDATE

Check if a value is a date

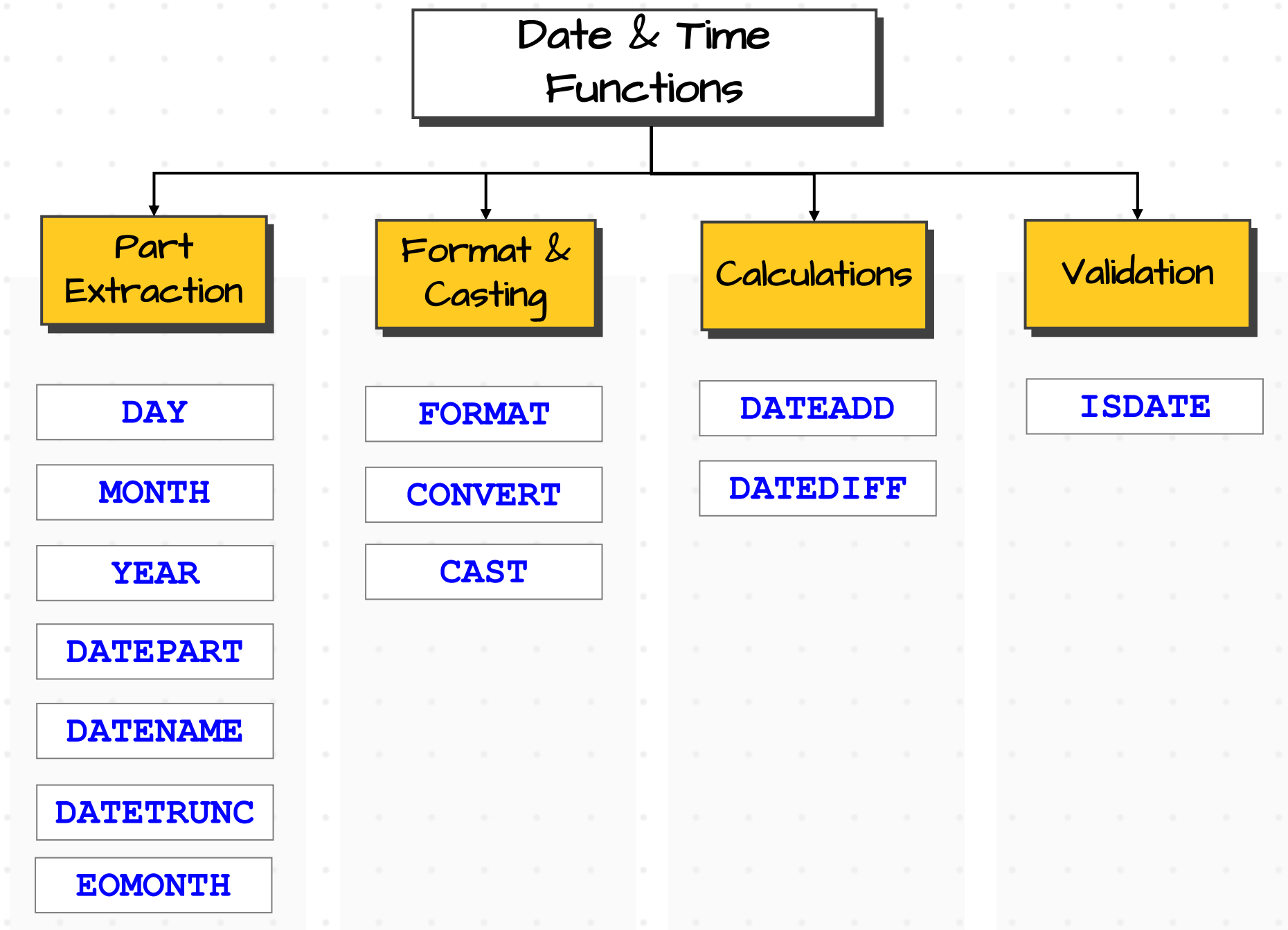
Returns 1 if the string value is a valid date

Syntax

```
ISDATE (value)
```

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

```
ISDATE (2025)
```



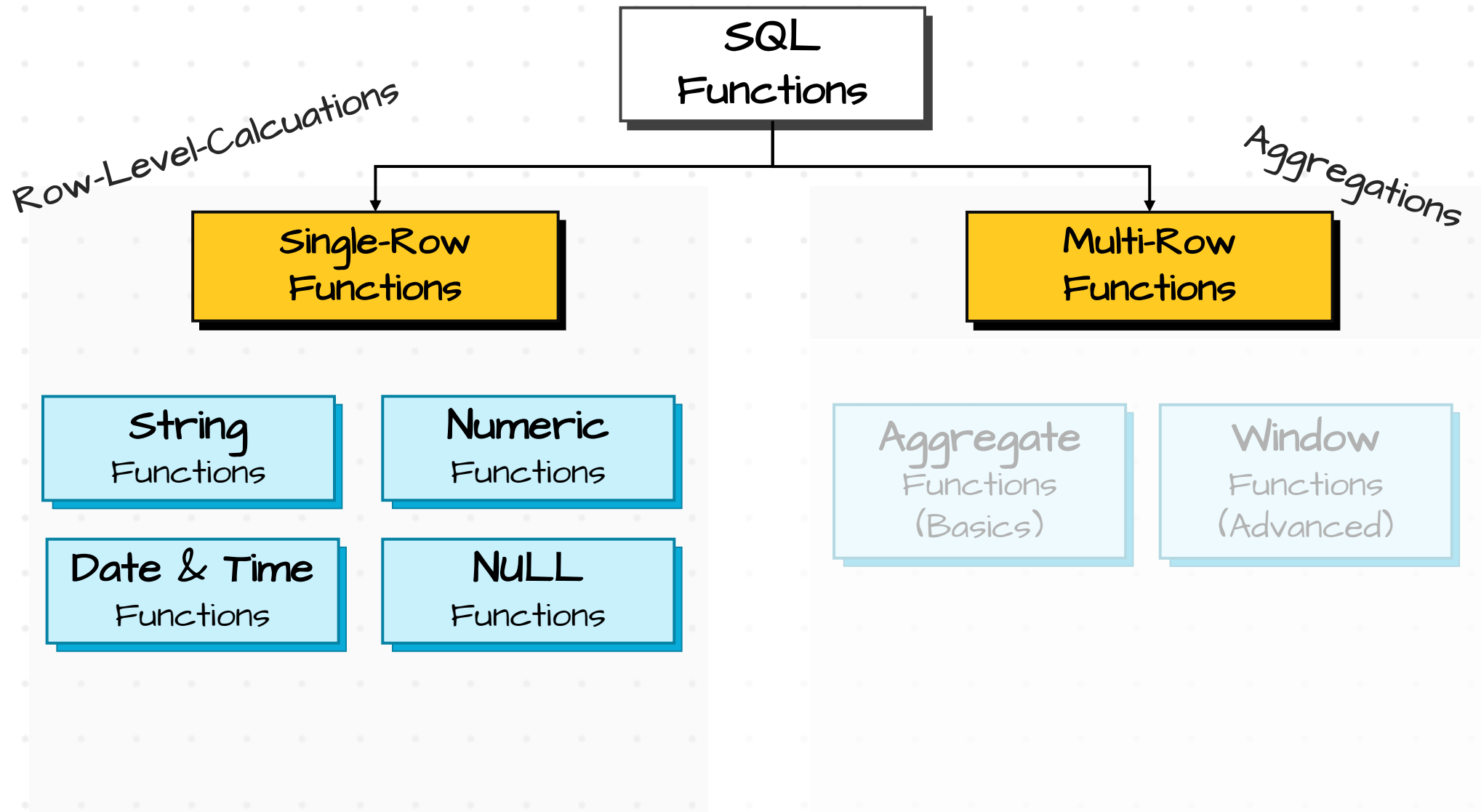


DATA WITH BARAA

NULL FUNCTIONS

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | NULL Functions



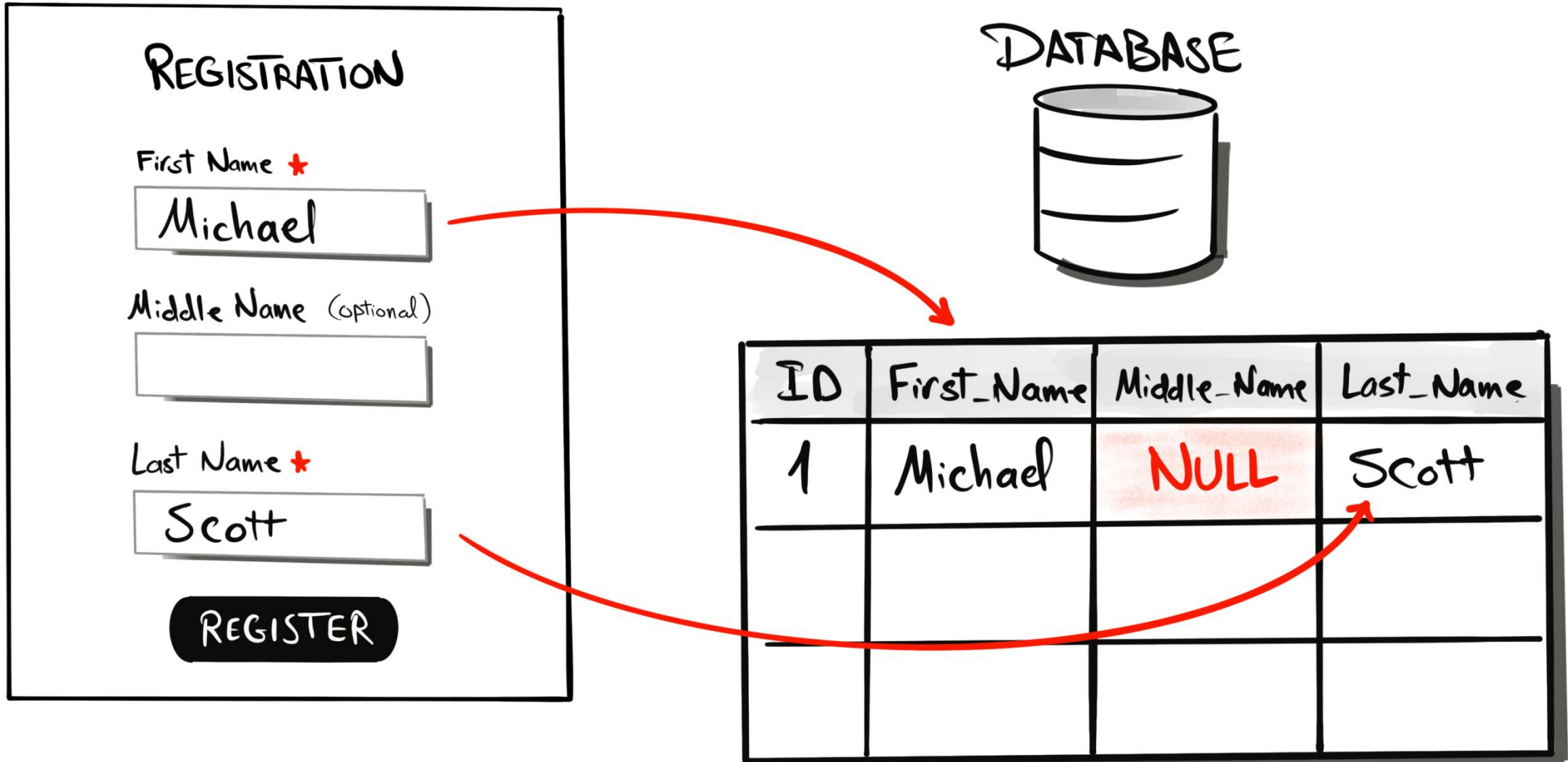


What are NULLS?

ID	Name	Country	Score
1	Maria	NULL	300
2	NULL	DE	NULL
3	John	NULL	800

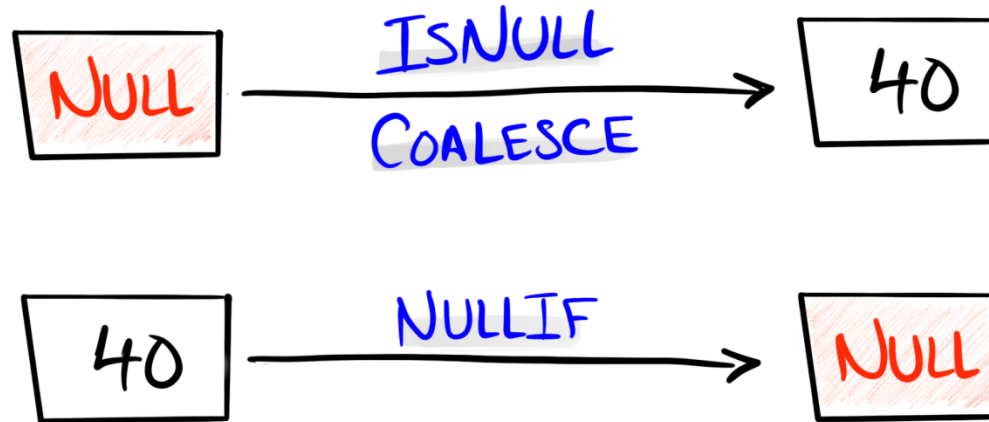

Unknown
Information

Where NULLS Come From?

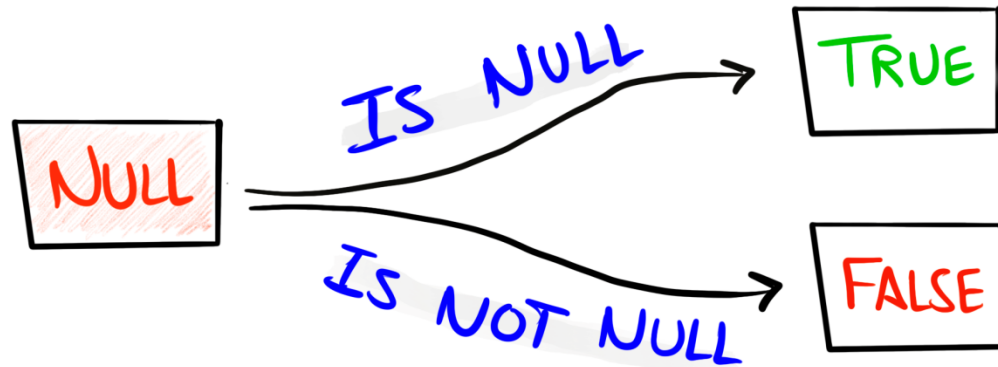


NULL FUNCTIONS

Replace
values



Check
for NULLs



ISNULL

Limited to two values

Fast

SQL Server → ISNULL
Oracle → NVL
MySQL → IFNULL

COALESCE

Unlimited

Slow

Available in All Databases

ISNULL

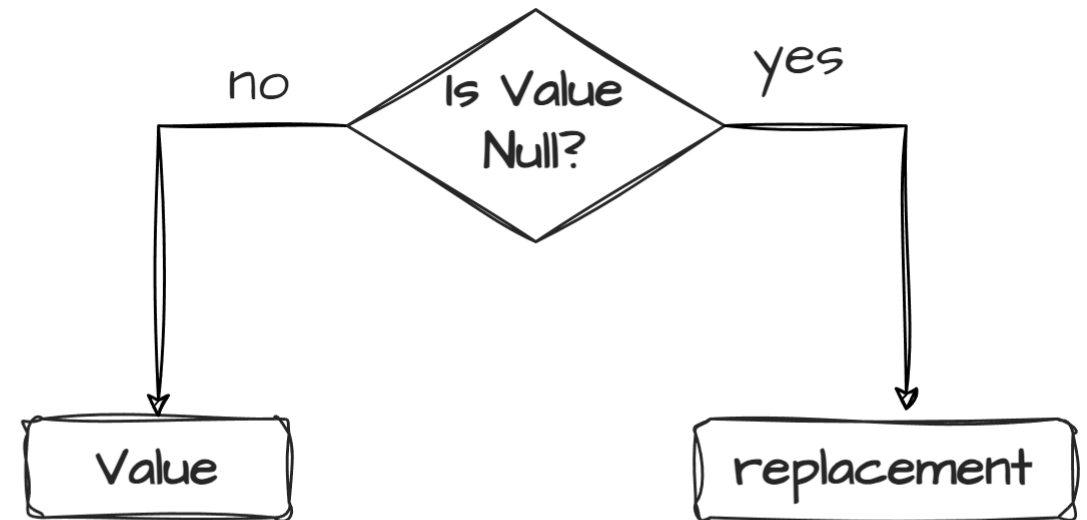
replaces NULL with the specified replacement value.

SYNTAX

ISNULL(value, replacement)

ISNULL(ShippingAddress, 'N/A')

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



ISNULL

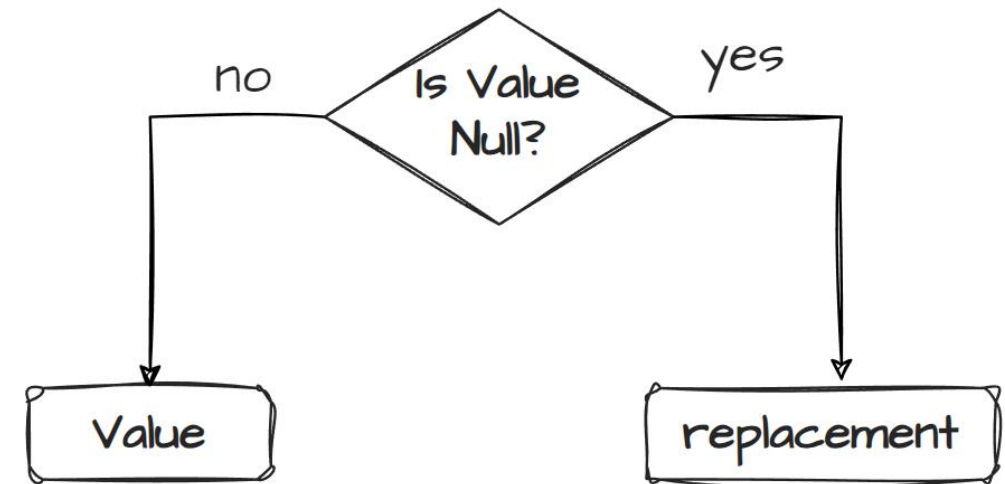
replaces NULL with the specified replacement value.

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

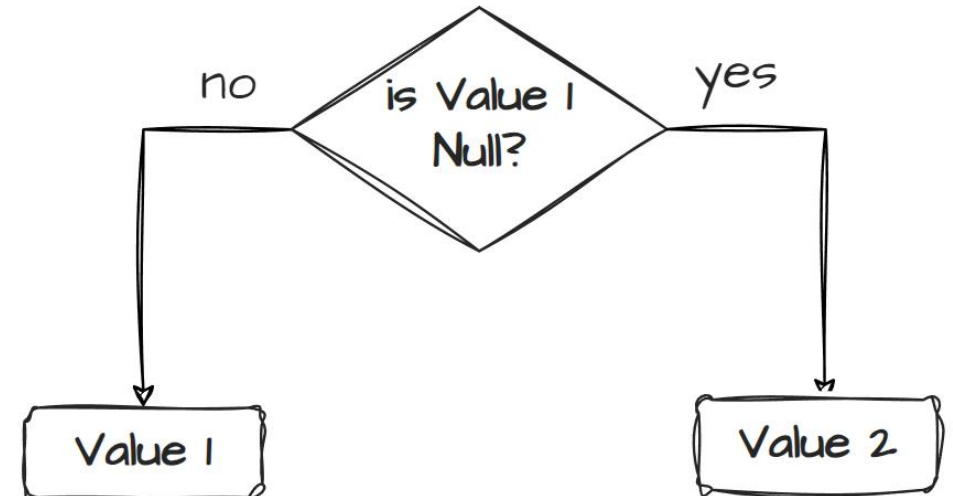
returns the first non-NULL value from the given expressions.

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



COALESCE

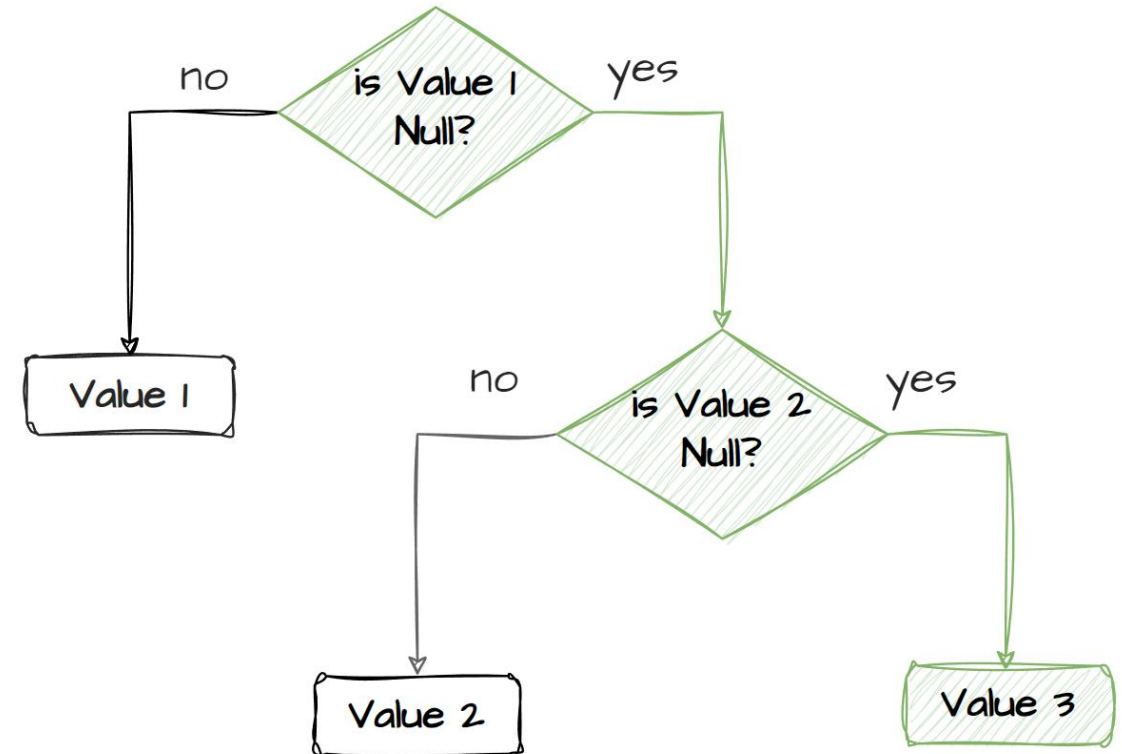
returns the first non-NULL value from the given expressions.

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



NULLIF

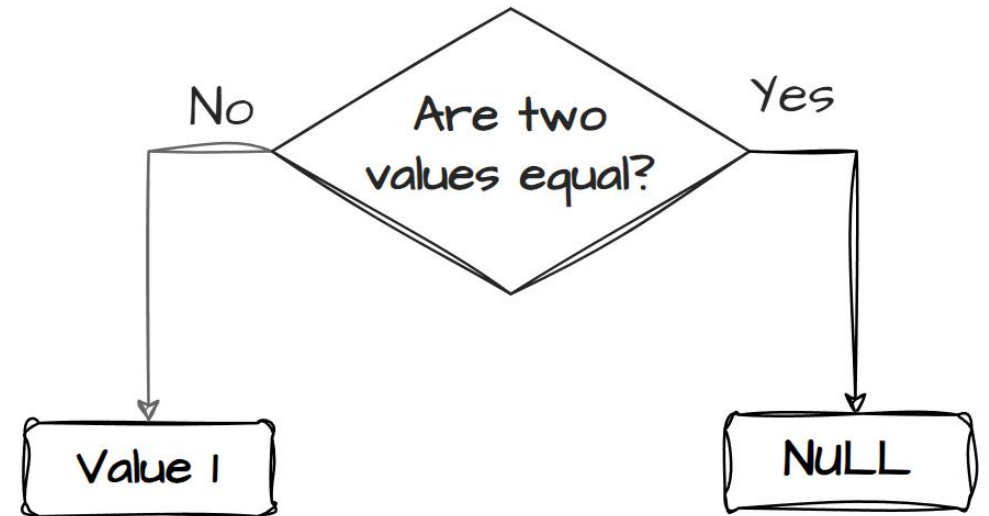
returns NULL if both values are equal; otherwise, it returns first value.

SYNTAX

```
NULLIF(value1, value2)
```

```
NULLIF(Original_Price, Discount_Price)
```

OrderID	Original_Price	Discount_Price	NULLIF
1	150	50	150
2	250	250	NULL

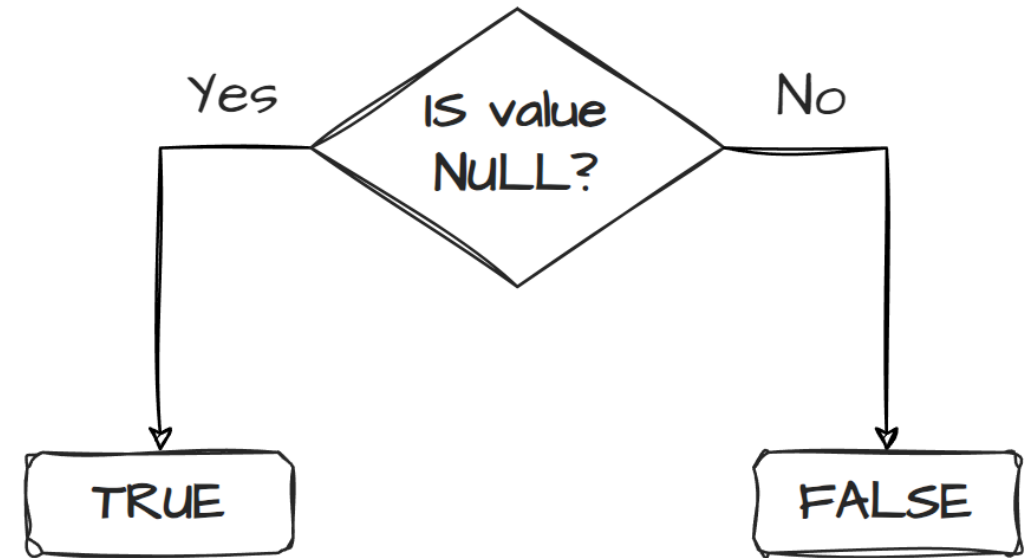


IS NULL

check if a value is NULL.

Price **IS NULL**

OrderID	Price	IS NULL	IS NOT NULL
1	90	FALSE	TRUE
2	NULL	TRUE	FALSE



IS NULL

In SQL, use IS NULL instead of = NULL to correctly filter rows with NULL values.

ID	Sales
1	100
2	200
3	NULL

→ WHERE Sales = 100 ⇒

ID	Sales
1	100

→ WHERE Sales = NULL ⇒ No Results

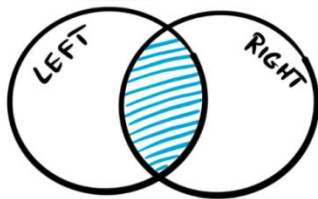
→ WHERE Sales IS NULL ⇒

ID	Sales
3	NULL

JOINS & IS NULL

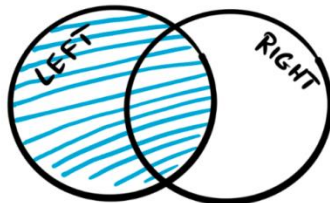
Only matching Rows

Inner Join



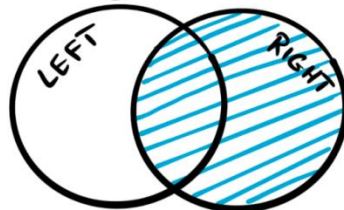
All Rows from Left and matching rows

Left Join



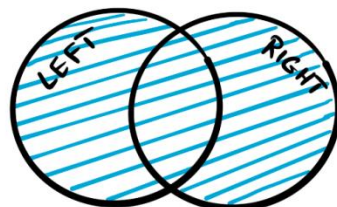
All Rows from right and matching rows

Right Join

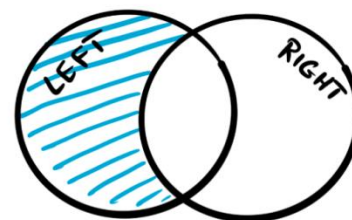


All Rows

Full Join

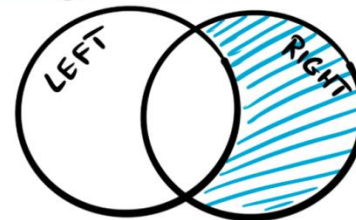


Left Anti Join



All Rows from Left without matching rows
(Left Join + IS NULL)

Right Anti Join



All Rows from Right without matching rows
(Right Join + IS NULL)

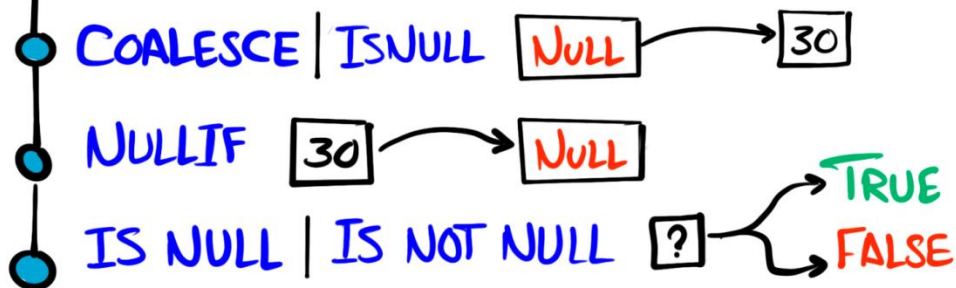
NULL vs Empty vs Blank

	<u>NULL</u>	<u>Empty String</u>	<u>Blank Space</u>
Representation	NULL	''	' '
Meaning	unknown	Known, Empty Value	Known, Space value
Data Type	Special Marker	String (0)	String (1 or more)
Storage	Very minimal	occupies memory	occupies memory (each space)
Performance	Best	Fast	Slow
Comparison	IS NULL	= ''	= ' '

NULL Functions

- Nulls special markers means missing value.
- Using Nulls can optimize storage and performance.

Functions



USE CASES

- Handle Nulls - Data Aggregation
- Handle Nulls - Mathematical operations
- Handle Nulls - Joining Tables
- Handle Nulls - Sorting Data
- Finding unmatched data - Left Anti Join
- Data Policies
 - Nulls
 - Default Value



DATA WITH BARAA

CASE STATEMENT

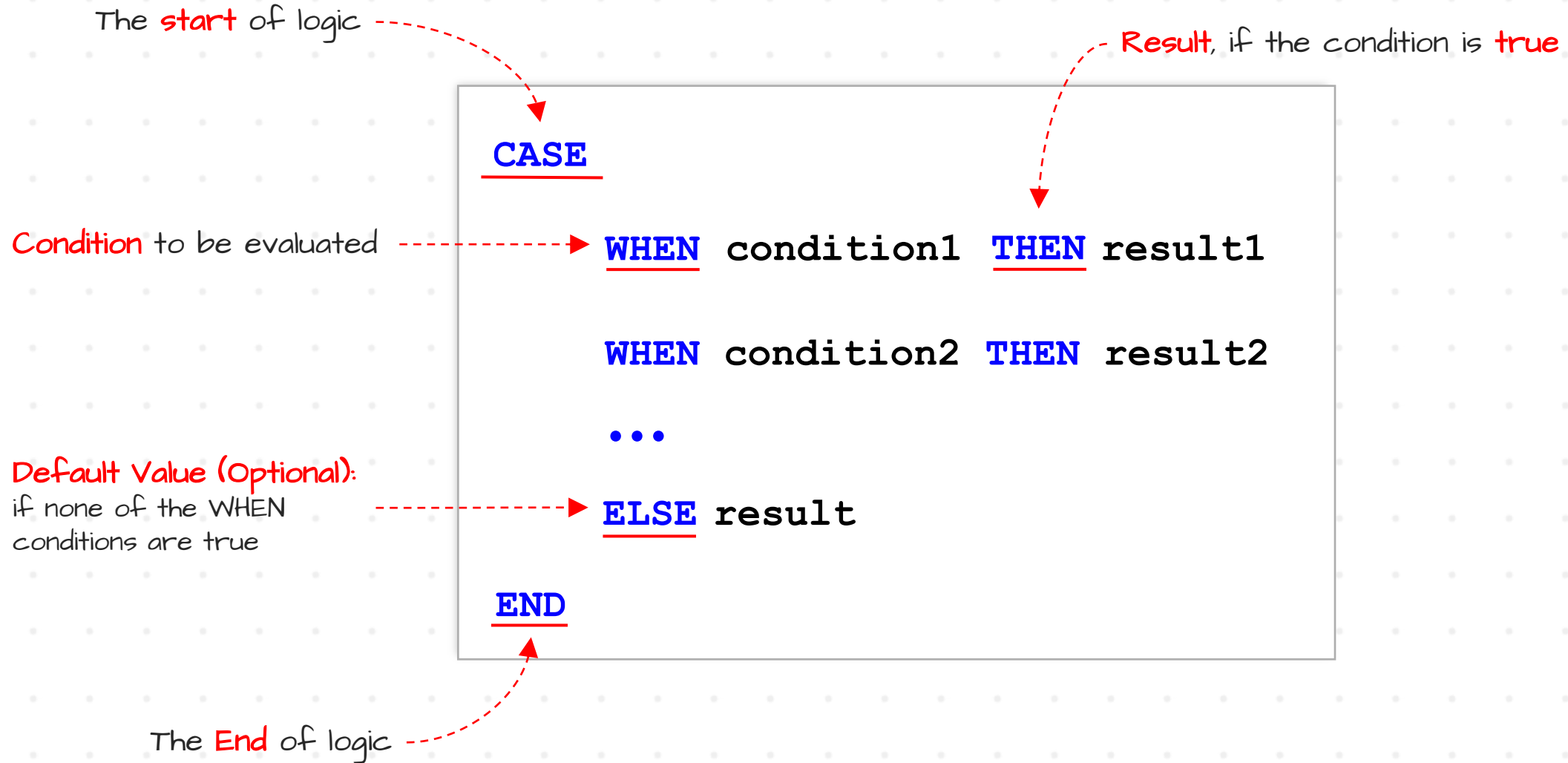
CASE WHEN

Baraa Khatib Salkini
YouTube | **DATA WITH BARAA**
SQL Course | Case Statement



CASE STATEMENT

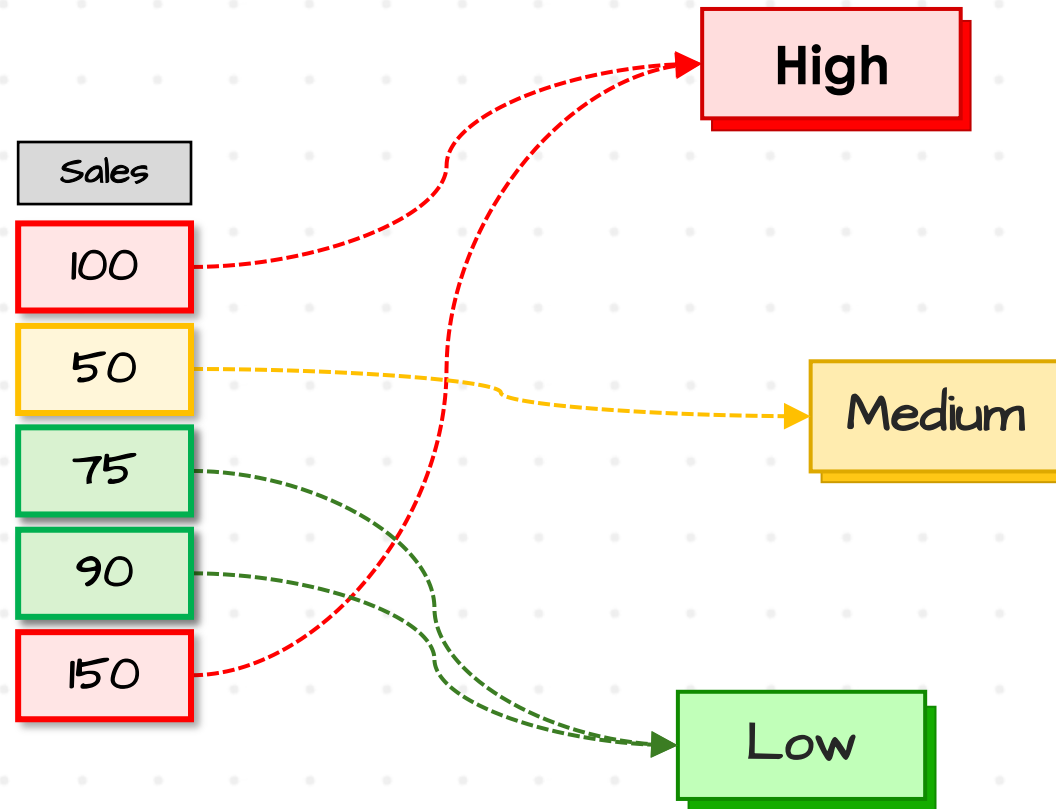
Syntax



Use Case: Derive New Columns

The CASE statement in SQL categorizes values based on conditions

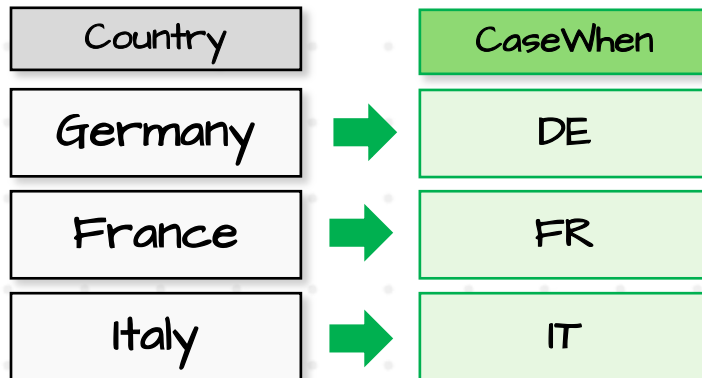
```
CASE
  WHEN Sales >= 100 THEN 'High'
  WHEN Sales >= 50 THEN 'Medium'
  ELSE 'Low'
END
```



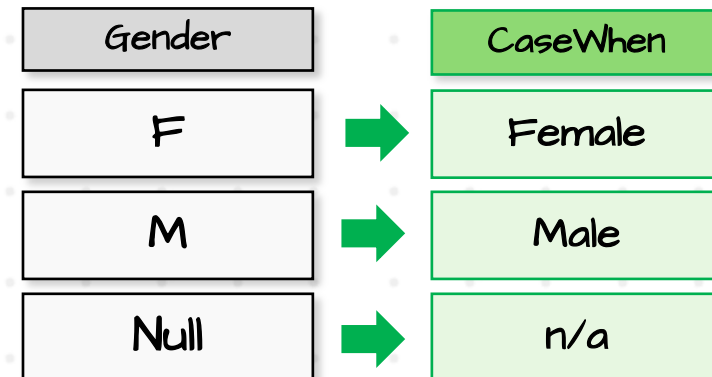
Transformation & Standardization

The CASE statement in SQL is used for data transformation and standardization by mapping specific values to standardized formats.

```
CASE
  WHEN Country = 'Germany' THEN 'DE'
  WHEN Country = 'France'  THEN 'FR'
  WHEN Country = 'Italy'   THEN 'IT'
ELSE 'n/a'
END
```



```
CASE
  WHEN Country = 'F' THEN 'Female'
  WHEN Country = 'M' THEN 'Male'
ELSE 'n/a'
END
```



CASE

```
WHEN Country = 'Germany' THEN 'DE'  
WHEN Country = 'India' THEN 'IN'  
WHEN Country = 'United States' THEN 'US'  
WHEN Country = 'France' THEN 'FR'  
WHEN Country = 'Italy' THEN 'IT'  
ELSE 'n/a'
```

END

Full Form

Column Name
to be evaluated (Only One)

Column Value
To be compared

CASE Country

```
WHEN 'Germany' THEN 'DE'  
WHEN 'India' THEN 'IN'  
WHEN 'United States' THEN 'US'  
WHEN 'France' THEN 'FR'  
WHEN 'Italy' THEN 'IT'  
ELSE 'n/a'
```

END

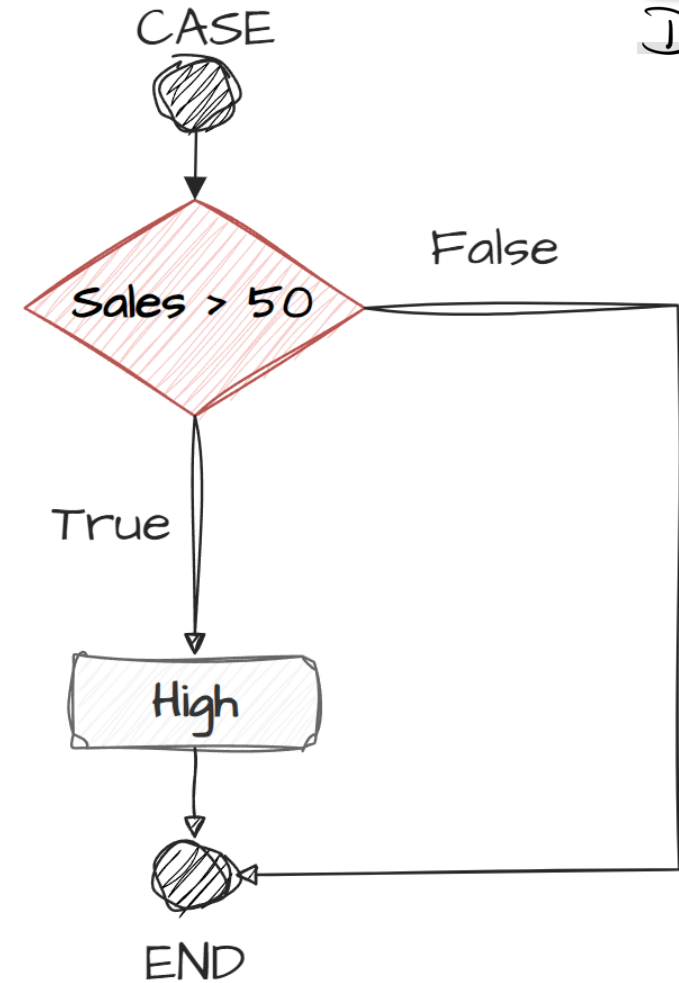
Quick Form

CASE

WHEN Sales > 50 **THEN** 'High'

END

Sales	CASE
60	High
30	NULL
15	NULL
NULL	NULL



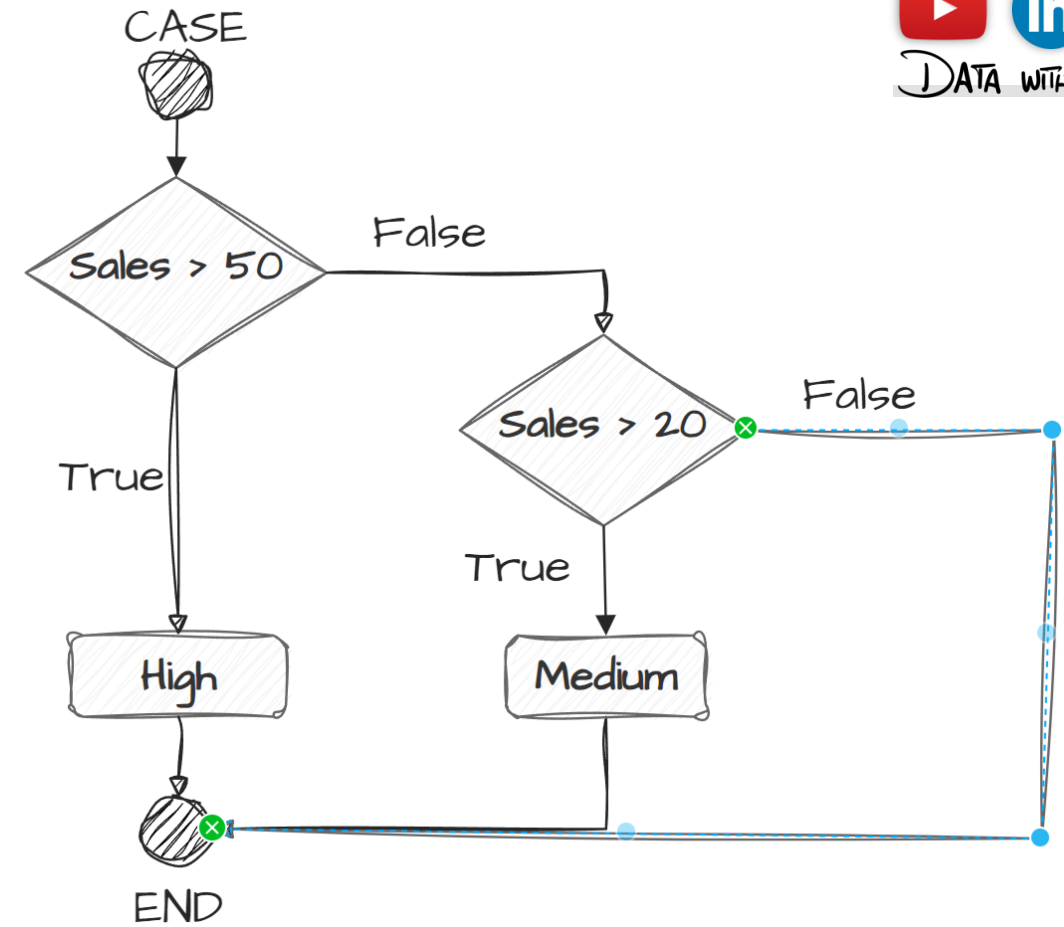
CASE

WHEN Sales > 50 **THEN** 'High'

WHEN Sales > 20 **THEN** 'Medium'

END

Sales	CASE
60	High
30	Medium
15	NULL
NULL	NULL

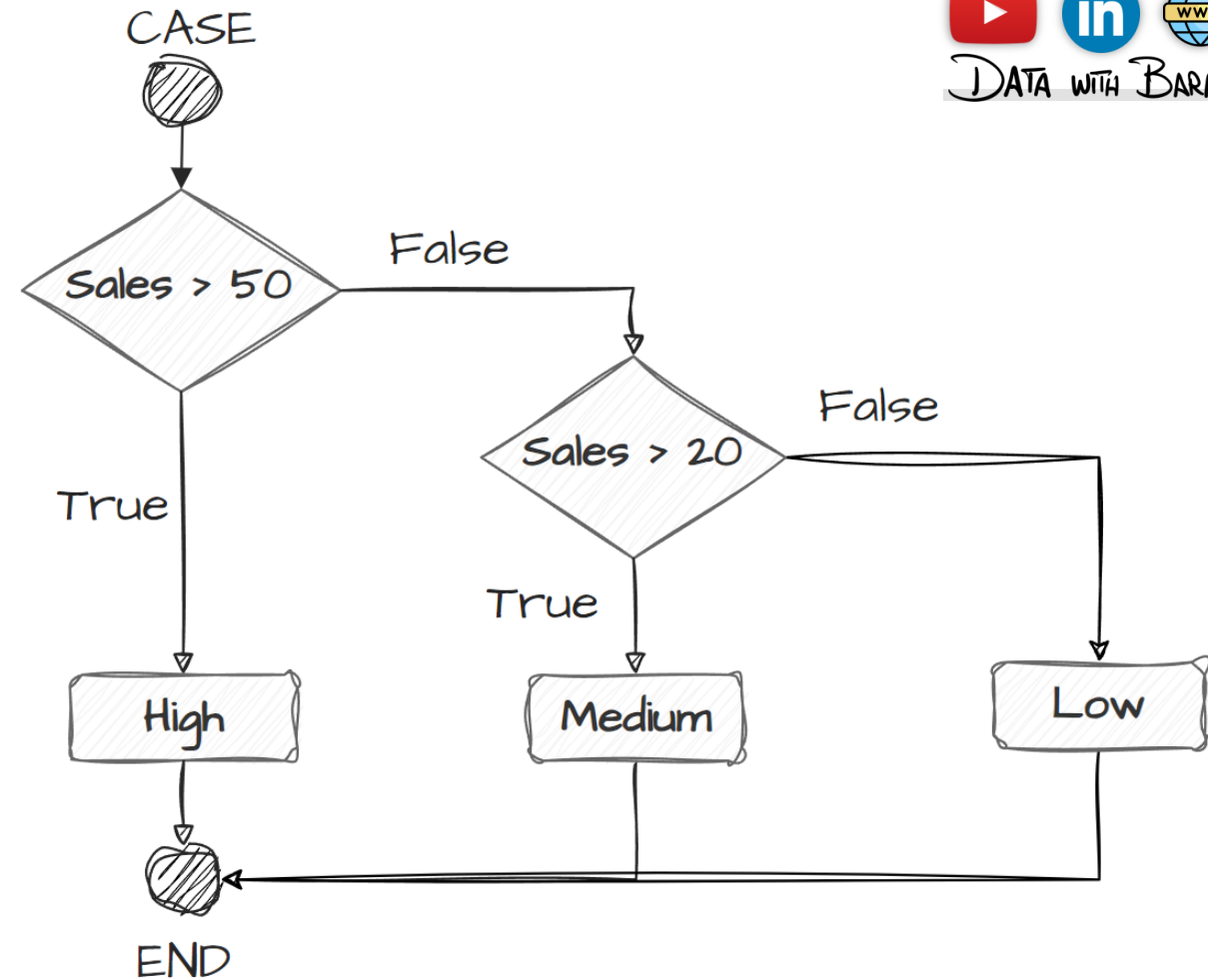


CASE

```

WHEN Sales > 50 THEN 'High'
WHEN Sales > 20 THEN 'Medium'
ELSE 'Low'
END
  
```

Sales	CASE
60	High
30	Medium
15	Low
NULL	Low



CASE STATEMENT

Evaluates a list of conditions and returns a value when the first Condition is met.

Rules

The data type of the results must be matching.

USE CASES

Derive New Information

- Categorizing Data
- Mapping Values
- Handling NULLs
- Conditional Aggregations