Tipe Data dan Fungsi Bawaan

Basis Data Lanjut

JTI-Polinema

Dosen Annisa Puspa Kirana, S.Kom, M.Kom

Outline

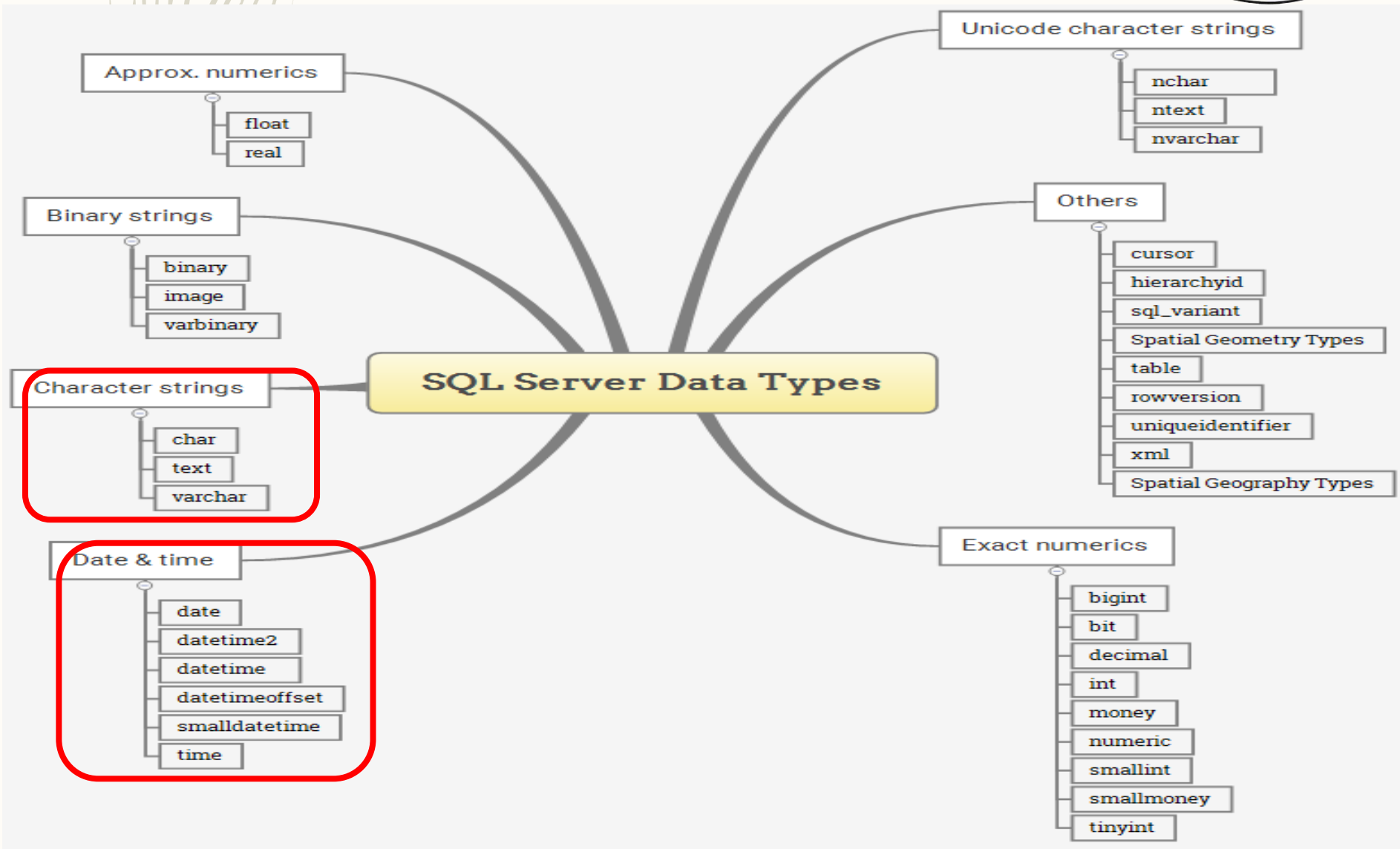
1. Tipe Data
2. Date & Time
3. Fungsi Date & Time
4. Data Karakter
5. Fungsi Karakter

Tipe data

Tentang TIPE DATA

- Himpunan yang dapat ditemui pada data
- Dapat disebut sebagai kelompok data
- Mendefinisikan suatu kolom
- Menentukan batasan/kontrol terhadap data
- Menentukan memori yang digunakan

TIPE DATA PADA SQL



Efisiensi

Tipe Data Sesuai = Efisien

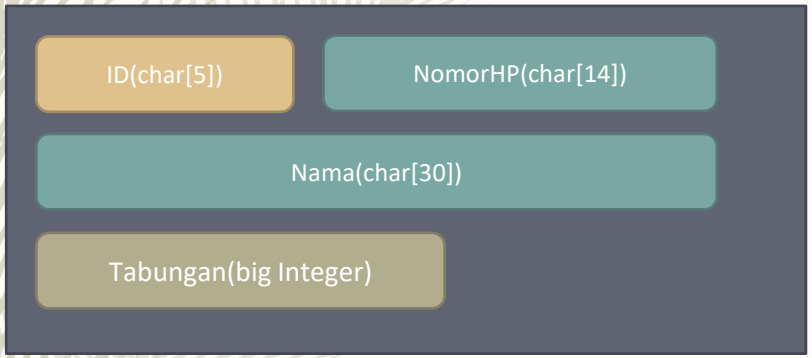
- Ketika membuat sebuah kolom dalam tabel, tipe data yang sesuai akan menghemat space storage
- Selain menghemat storage, juga akan membuat database lebih reliable.
- Tipe data yang sesuai juga akan lebih memastikan data yang masuk sesuai standar.

Tipe Data Tidak Sesuai = Tidak Efisien

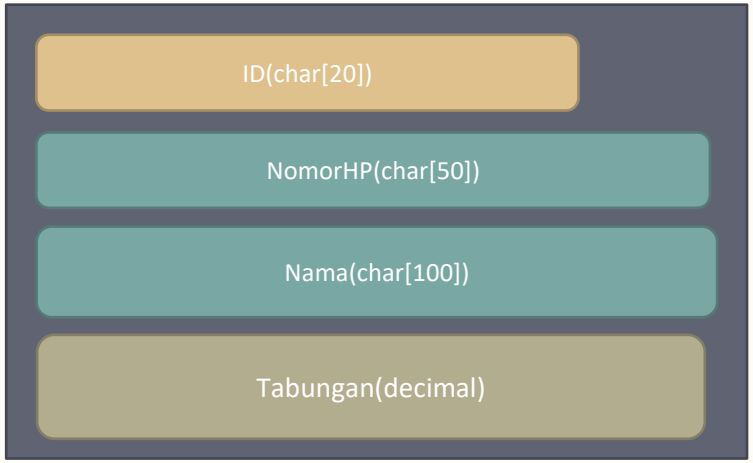
- Contoh :
  - Untuk membuat kolom Nomer Handphone maka hanya diperlukan paling banyak 15 karakter
- Untuk menyimpan tanggal, gunakan tipe data *datetime,* sehingga format penanggalan akan seragam.

Illustrasi tipe data

Tipe Data Tidak Sesuai (exaggerated)



- 1 record data
- Total :
  - char = 5 + 14 + 30 = 49 byte
  - Big integer = 8 byte
  - TOTAL 1 record = 57 byte



- 1 record data
- Total :
  - Char = 20 + 50 + 100 = 170byte
  - Decimal = 17 byte
  - TOTAL 1 record = 187 byte

10 Integer Data Types

Integer Data Types in SQL Server:

Integer Data Types are allowed only to hold integer types of values and this data type can be applied on EmpId, ProductCode, BracnchCode columns, etc. These data types are classified into 4 types based on their range and memory size as shown in the below image

Data Type	Range	Stored Memory
TinyInt	0-255	1byte
SmallInt	-32768 to 32767	2bytes
Int	$-2 * 10^{-8}$ to $2 * 10^{8}$	4 bytes
BigInt	$-9 * 10^{-8}$ to $9 * 10^{8}$	8 bytes

Decimal Data Types in SQL Server:

These data types are allowed decimal point values only. The Decimal Data Type contains two types those are

1. **Decimal (P, S)**
2. **Numeric (P, S)**

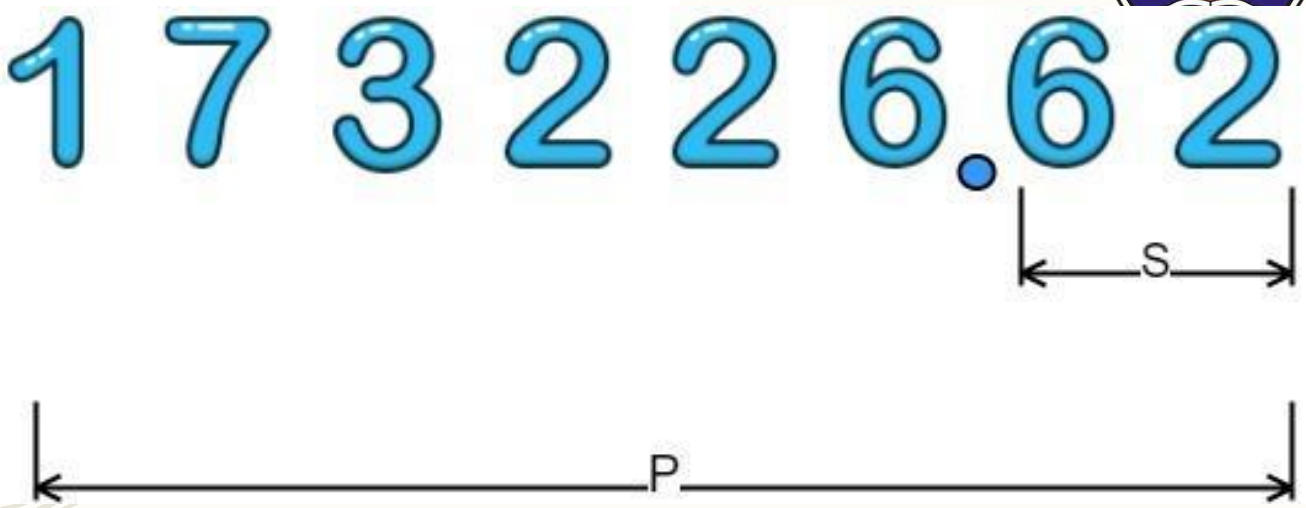
https://dotnettutorials.net/lesson/data-types-sql-server/

Let's look at an example. Suppose that you defined a "`balance`" column as `NUMERIC` with a precision of 8 and a scale of 2.

The DDL would look like this:

```
CREATE TABLE account (
accountNo integer,
balance numeric(8,2)
);
```

The "`balance`" column can safely store



Data type	Range	Storage
BIGINT	-2^{63} (-9,223,372,036,854,775,808) to $2^{63}-1$ (9,223,372,036,854,775,807)	8 Bytes
INT	-2^{31} (-2,147,483,648) to $2^{31}-1$ (2,147,483,647)	4 Bytes
SMALLINT	-2^{15} (-32,768) to $2^{15}-1$ (32,767)	2 Bytes
TINYINT	0 to 255	1 Byte

bigint_col	int_col	smallint_col	tinyint_col
9223372036854775807	2147483647	32767	255

Date & time

TIPE DATA DATE & TIME

Data Type	Storage (bytes)	Date Range	Accuracy	Recommended Entry Format
DATETIME	8	January 1, 1753 to December 31, 9999	3-1/3 milliseconds	'YYMMDD hh:mm:ss:nnn'
SMALLDATETIME	4	January 1, 1900 to June 6, 2079	1 minute	'YYMMDD hh:mm:ss:nnn'
DATETIME2	6 to 8	January 1, 0001 to December 31, 9999	100 nanoseconds	'YYMMDD hh:mm:ss.nnnnnn'
DATE	3	January 1, 0001 to December 31, 9999	1 day	'YYYY-MM-DD'
TIME	3 to 5		100 nanoseconds	'hh:mm:ss:nnnnnnn'
DATETIMEOFFSET	8 to 10	January 1, 0001 to December 31, 9999	100 nanoseconds	'YY-MM-DD hh:mm:ss:nnnnnnn [+ |-]hh:mm'

– Secara umum format DATE adalah YYMMDD, sedangkan TIME adalah hh:mm:ss

Query dengan date

```
SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE orderdate = '20070825';
```

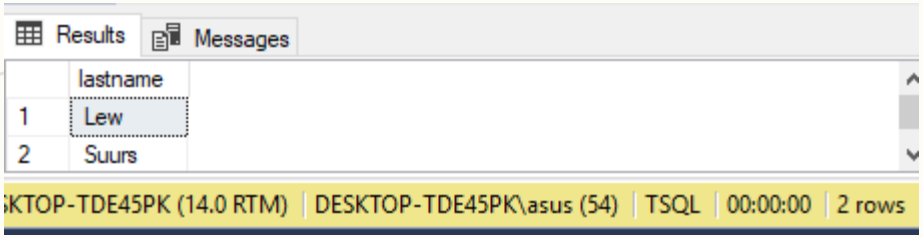
Results | Messages

	orderid	custid	empid	orderdate
1	10643	1	6	2007-08-25 00:00:00.000
2	10644	88	3	2007-08-25 00:00:00.000

SKTOP-TDE45PK (14.0 RTM) | DESKTOP-TDE45PK\asus (54) | TSQL | 00:00:00 | 2 rows

Query dengan date (1)

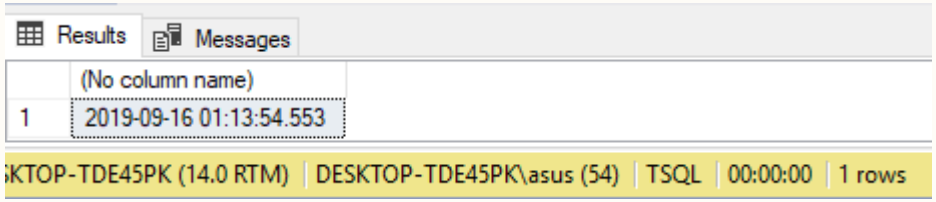
```
SELECT lastname
FROM HR.Employees
WHERE
    YEAR(birthdate) = 1973;
```



FUNGSI DATE & TIME

```
SELECT CURRENT_TIMESTAMP
```

- CURRENT_TIMESTAMP()



- Mengembalikan nilai berupa tanggal dan waktu saat ini
- Merupakan fungsi SQL ANSI
- Fungsi TSQL: GETDATE()

FUNGSI DATE & TIME (1)

Function	Return Type	Remarks
GETDATE()	datetime	Current date and time. No time zone offset.
GETUTCDATE()	datetime	Current date and time in UTC.
CURRENT_TIMESTAMP	datetime	Current date and time. No time zone offset. ANSI standard.
SYSDATETIME()	datetime2	Current date and time. No time zone offset
STSUTCDATETIME()	datetime2	Current date and time in UTC.
SYSDATETIMEOFFSET()	datetimeoffset	Current date and time. Includes time zone offset

- silahkan dicoba masing-masing!

Fungsi DATE & TIME (2)

– DATEADD()

```
SELECT
    lastname, hiredate,
    DATEADD(year, 1, hiredate) AS assesment
FROM HR.Employees
```

Results | Messages

	lastname	hiredate	assesment
1	Davis	2002-05-01 00:00:00.000	2003-05-01 00:00:00.000
2	Funk	2002-08-14 00:00:00.000	2003-08-14 00:00:00.000
3	Lew	2002-04-01 00:00:00.000	2003-04-01 00:00:00.000
4	Peled	2003-05-03 00:00:00.000	2004-05-03 00:00:00.000
5	Buck	2003-10-17 00:00:00.000	2004-10-17 00:00:00.000
6	Suurs	2003-10-17 00:00:00.000	2004-10-17 00:00:00.000
7	King	2004-01-02 00:00:00.000	2005-01-02 00:00:00.000
8	Cameron	2004-03-05 00:00:00.000	2005-03-05 00:00:00.000
9	Dolgopyatova	2004-11-15 00:00:00.000	2005-11-15 00:00:00.000

SKTOP-TDE45PK (14.0 RTM) | DESKTOP-TDE45PK\asus (54) | TSQL | 00:00:00 | 9 rows

- Digunakan untuk penjumlahan tanggal dan waktu

Fungsi DATE & TIME (3)

- DATEDIFF()

```
SELECT lastname,
DATEDIFF(year,YEAR(CURRENT_TIMESTAMP), birthdate) AS age
FROM HR.Employees
```

Results | Messages

	lastname	age
1	Davis	53
2	Funk	57
3	Lew	68
4	Peled	42
5	Buck	60
6	Suurs	68
7	King	65
8	Cameron	63
9	Dolgopyatova	71

Query executed successfully.

- Digunakan untuk mendapatkan selisih tanggal dan waktu

20 CAST / CONVERT

1. CAST dan CONVERT adalah fitur dari server SQL yang diperlukan untuk konversi ekspresi dari satu jenis ke tipe lainnya.
2. CAST lebih user-friendly daripada CONVERT karena lebih mudah digunakan untuk konversi.
3. CONVERT, bagaimanapun, terbukti lebih kuat dan fleksibel daripada CAST.
4. CAST disarankan untuk konversi dasar. CONVERT disarankan untuk rutinitas spesifik datetime.
5. CAST mempunyai waktu loading data yang lebih cepat dibandingkan dengan menggunakan CONVERT.

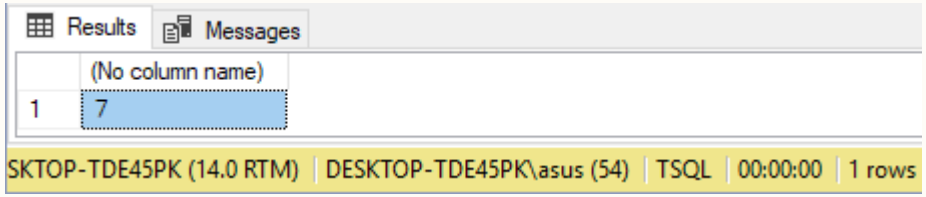
21 CAST / convert

– Kapan tipe data harus dikonversi?

- ketika data dipindahkan, dibandingkan, atau digabungkan dengan data lain

eg.

```
DECLARE @somechar CHAR(5) = '6'
DECLARE @someint INT = 1
SELECT @somechar + @someint;
```



Kenapa char otomatis di convert ke integer?

22 CAST

- Digunakan pada klausa SELECT dan WHERE
- Merupakan fungsi standar ANSI
- Syntax:

```
CAST(<value> AS <datatype>)
```

23 CONVERT

- Digunakan pada klausa SELECT dan WHERE
- Untuk konversi:
  - Date
  - Time
  - Numeric
  - XML
  - dll
- Merupakan fungsi bawaan SQL SERVER
- Syntax:

```
CONVERT(<datatype>,
<value>,<optional_style_number>)
```

Convert date format

Without century	With century	Input/Output	Standard
0	100	mon dd yyyy hh:miAM/PM	Default
1	101	mm/dd/yyyy	US
2	102	yyyy.mm.dd	ANSI
3	103	dd/mm/yyyy	British/French
4	104	dd.mm.yyyy	German
5	105	dd-mm-yyyy	Italian
6	106	dd mon yyyy	-
7	107	Mon dd, yyyy	-
8	108	hh:mm:ss	-
9	109	mon dd yyyy hh:mi:ss:mmmAM (or PM)	Default + millisec

https://www.w3schools.com/sql/func_sqlserver_convert.asp

CAST / CONVERT (2)

– Konversi tipe data secara eksplisit

```
SELECT CAST('20190916' AS DATE)

SELECT CONVERT(DATE, '09/16/2019')
```

	(No column name)
1	2019-09-16

SKTOP-TDE45PK (14.0 RTM) | DESKTOP-TDE45PK\asus (54) | TSQL | 00:00:0

Data Karakter

Tipe data character

- Regular: CHAR, VARCHAR
  - 256 character set
  - 1 byte per character
  - single quote ‘ ’
- Unicode: NCHAR, NVARCHAR
  - 2 bytes per character
  - N prefix N‘ ’
- TEXT, NTEXT
  - VARCHAR(MAX) dan NVARCHAR(MAX)

CONcatenation

- Menggabungkan karakter
- Menggunakan tanda +

```
SELECT empid, firstname + N ‘ ’ +  lastname AS FullName
FROM HR.employees;
```

- Menggunakan CONCAT()

```
SELECT empid,
        CONCAT (firstname, ‘ ’, lastname) AS FullName
FROM HR.employees;
```

Results | Messages

	empid	FullName
1	1	Sara Davis
2	2	Don Funk
3	3	Judy Lew
4	4	Yael Peled
5	5	Sven Buck

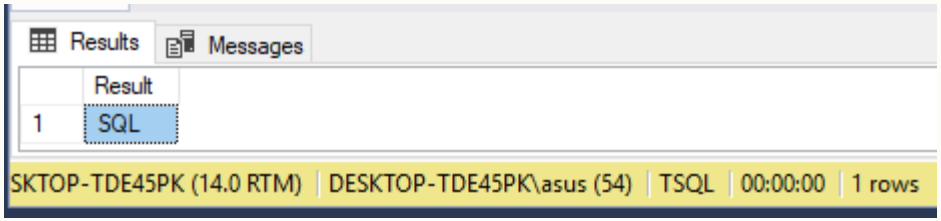
Fungsi String

- SUBSTRING()

  SUBSTRING(EXPRESSION, START, LENGTH)

  eg:

```
SELECT
SUBSTRING ('Microsoft SQL SERVER', 11,3)
AS Result;
```



- Mengembalikan sebagian string berdasarkan *starting point* dan jumlah karakter yang dikembalikan

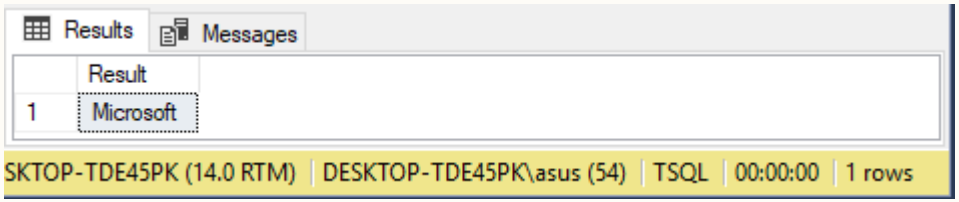
Fungsi String (1)

- LEFT(), RIGHT()

  LEFT(EXPRESSION, integer_value)

  eg:

```
SELECT
LEFT ('Microsoft SQL SERVER', 9)
AS Result;
```



- Mengembalikan sebagian string dalam jumlah tertentu, dimulai dari karakter string paling kiri/kanan

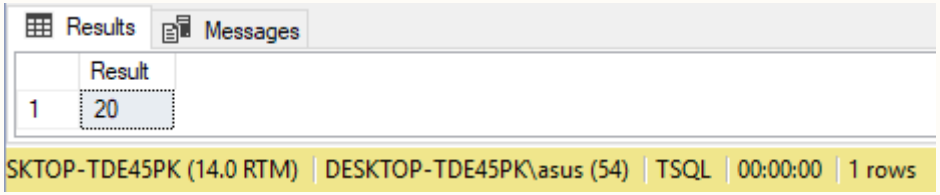
Fungsi String (2)

– LEN(), DATALENGHT()

LEN(STRING), DATALENGHT(STRING)

eg:

```
SELECT LEN('Microsoft SQL SERVER')AS Result;
SELECT
    DATALENGHT('Microsoft SQL SERVER')AS Result;
```



- Mengembalikan jumlah karakter pada string tertentu

32 LEN vs datalength

Name	Name_Length	Name_Datalength
'AAA'	3	3
' BBB'	4	4
'CCC '	3	5

Name_in_Regional_Lang	Length	Datalength
'அஅஅ'	3	6
'ஆஆஆ'	3	6
'இஇஇஇ'	4	8

LIKE

- Digunakan untuk membandingkan data berdasarkan pola tertentu
- Digunakan pada statemen WHERE
- Simbol % merepresentasikan string
- eg.

```
SELECT categoryid, description
 FROM Production.Categories
 WHERE description LIKE '%ee%';
```

Results | Messages

	categoryid	description
1	1	Soft drinks, coffees, teas, beers, and ales
2	2	Sweet and savory sauces, relishes, spreads, and ...
3	3	Desserts, candies, and sweet breads
4	4	Cheeses
5	8	Seaweed and fish

SKTOP-TDE45PK (14.0 RTM) | DESKTOP-TDE45PK\asus (54) | TSQL | 00:00:00 | 5 rows

Working with NULLs
NULL Values

- NULL merepresentasikan nilai yang kosong
- Perlakuan ANSI untuk NULL values:
  - Hasil semua ekspresi yang bernilai NULL adalah NULL
    - *2 + NULL = NULL*
    - *'MyString: ' + NULL = NULL*
  - Perbandingan nilai
    - *NULL = NULL returns false*
    - *NULL **IS NULL** returns true*

Working with NULLs
NULL Functions

- ISNULL menggantikan nilai NULL dengan suatu nilai

ISNULL elemen	Deskripsi
Ekspresi/ kolom yang ingin di cek	Mengembalikan ekspresi tersebut jika tidak bernilai NULL
Nilai pengganti	Menjadi kembalian jika ekspresi/kolom bernilai NULL

Without isnull

```
Select custid, city, region, country
FROM Sales.Customers;
```

	custid	city	region	country
1	1	Berlin	NULL	Germany
2	2	México D.F.	NULL	Mexico
3	3	México D.F.	NULL	Mexico
4	4	London	NULL	UK
5	5	Luleå	NULL	Sweden
6	6	Mannheim	NULL	Germany
7	7	Strasbourg	NULL	France
8	8	Madrid	NULL	Spain
9	9	Marseille	NULL	France
10	10	Tsawassen	BC	Canada
11	11	London	NULL	UK
12	12	Buenos Aires	NULL	Argentina

37 With isnull

```
Select custid, city, ISNULL(region, 'N/A') AS region, country
FROM Sales.Customers;
```

	custid	city	region	country
1	1	Berlin	N/A	Germany
2	2	México D.F.	N/A	Mexico
3	3	México D.F.	N/A	Mexico
4	4	London	N/A	UK
5	5	Luleå	N/A	Sweden
6	6	Mannheim	N/A	Germany
7	7	Strasbourg	N/A	France
8	8	Madrid	N/A	Spain
9	9	Marseille	N/A	France
10	10	Tsawassen	BC	Canada
11	11	London	N/A	UK
12	12	Buenos Aires	N/A	Argentina
13	13	México D.F.	N/A	Mexico

38 coalesce

- Coalesce mengembalikan nilai pertama yang tidak null dari list kolom/variable yang diberikan
- Dengan hanya 2 argumen, COALESCE bertindak seperti ISNULL
- Jika semua argument NULL, maka COALESCE mengembalikan nilai NULL
- Syntax:

```
SELECT COALESCE(<expression_1>[,
...<expression_n>];
```

39 Coalesce

```
Select custid, city, region, country+','+ COALESCE(region,' ')+','+city as location
FROM Sales.Customers;
```

	custid	city	region	location
1	1	Berlin	NULL	Germany, ,Berlin
2	2	México D.F.	NULL	Mexico, ,México D.F.
3	3	México D.F.	NULL	Mexico, ,México D.F.
4	4	London	NULL	UK, ,London
5	5	Luleå	NULL	Sweden, ,Luleå
6	6	Mannheim	NULL	Germany, ,Mannheim
7	7	Strasbourg	NULL	France, ,Strasbourg
8	8	Madrid	NULL	Spain, ,Madrid
9	9	Marseille	NULL	France, ,Marseille
10	10	Tsawassen	BC	Canada,BC,Tsawassen
11	11	London	NULL	UK, ,London
12	12	Buenos Aires	NULL	Argentina, ,Buenos Aires
13	13	México D.F.	NULL	Mexico, ,México D.F.

Logical Function

Logical function

- Seleksi kondisi dengan IIF
- Memilih Item pada List dengan CHOOSE

Seleksi kondisi dengan **IIF**

- Fungsi logika baru pada SQL Server 2012
- Seperti ekspresi CASE dengan 2 kemungkinan kembalian
- Syntax:

```
SELECT IIF(<Boolean expression>,<value_if_TRUE>,
<value_if_FALSE_or_UNKNOWN);
```

43 Seleksi kondisi dengan **IIF**

– Example

```
SELECT productid, unitprice,
IIF(unitprice > 50, 'high','low') AS
pricepoint
FROM Production.Products;
```

productid	unitprice	pricepoint
7	30.00	low
8	40.00	low
9	97.00	high
17	39.00	low
18	62.50	high

Memilih Item pada List dengan **CHOOSE**

- CHOOSE mengembalikan sebuah item pada list yang dipilih berdasarkan nilai index
- Syntax:

```
SELECT CHOOSE(<index_value>,<item1>,
<item2>[,...]);
```

Elemen CHOOSE	Deskripsi
Index_value	Nilai integer yang merepresentasikan posisi item pada list
Value_List **(<item1>, <item2>,…)**	List item dengan tipe data apa saja yang akan dikembalikan nilainya

45 Memilih Item pada List dengan **CHOOSE**

```
SELECT CHOOSE (3, 'Beverages', 'Condiments', 'Confections') AS
choose_result;
```

```
choose_result
-------------
Confections
```

Terimakasih