CHAPTER 14: STRING FUNCTIONS

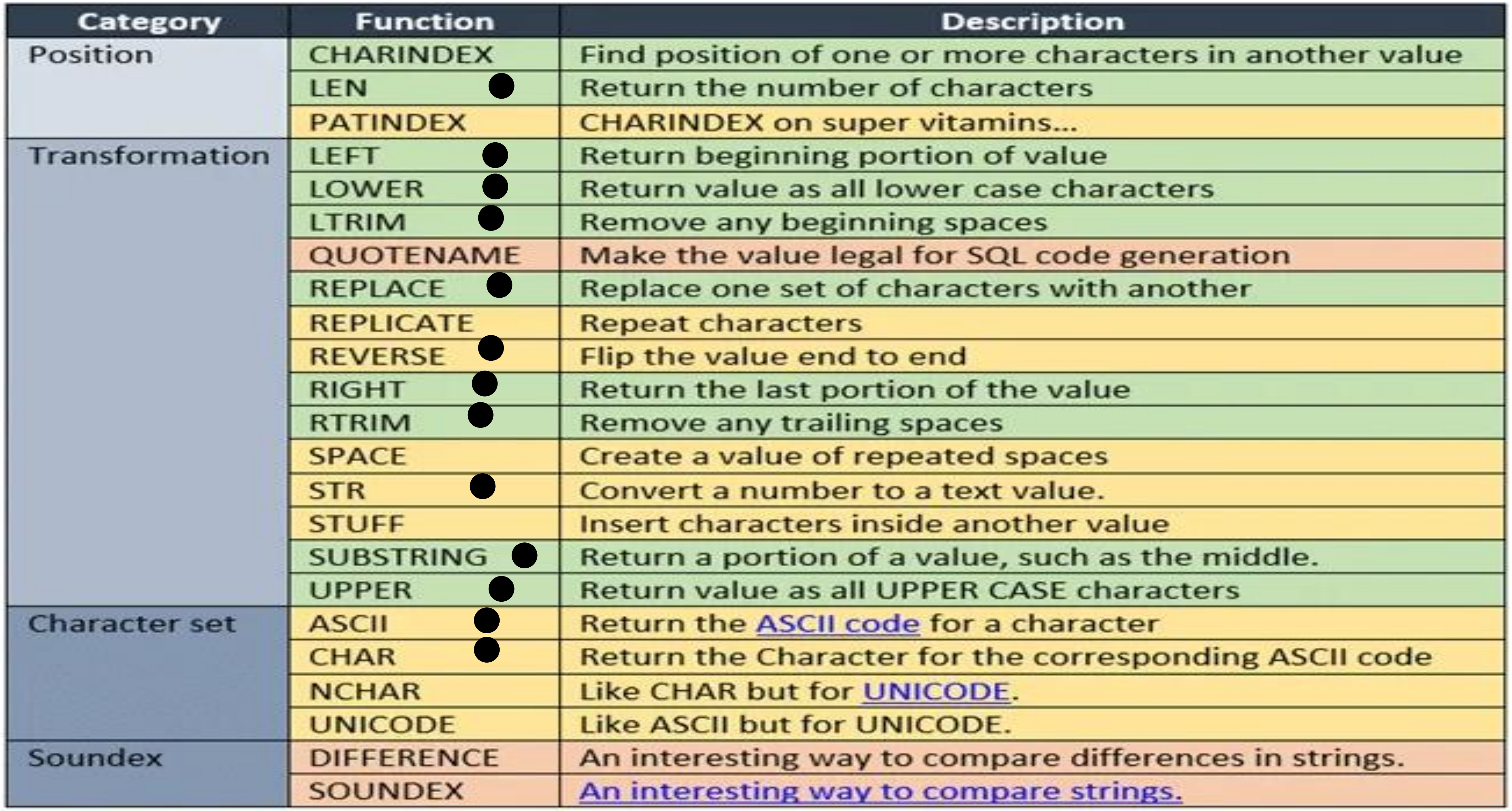
2023

CHAPTER 14: STRING FUNCTIONS

Contents:

MS ACCESS STRING / TEXT FUNCTIONS

These are some string functions and their purpose:



Category	Function	Description
Position	CHARINDEX	Find position of one or more characters in another value
	LEN ●	Return the number of characters
	PATINDEX	CHARINDEX on super vitamins...
Transformation	LEFT ●	Return beginning portion of value
	LOWER ●	Return value as all lower case characters
	LTRIM ●	Remove any beginning spaces
	QUOTENAME	Make the value legal for SQL code generation
	REPLACE ●	Replace one set of characters with another
	REPLICATE	Repeat characters
	REVERSE ●	Flip the value end to end
	RIGHT ●	Return the last portion of the value
	RTRIM ●	Remove any trailing spaces
	SPACE	Create a value of repeated spaces
	STR ●	Convert a number to a text value.
	STUFF	Insert characters inside another value
	SUBSTRING ●	Return a portion of a value, such as the middle.
	UPPER ●	Return value as all UPPER CASE characters
Character set	ASCII ●	Return the ASCII code for a character
	CHAR ●	Return the Character for the corresponding ASCII code
	NCHAR	Like CHAR but for UNICODE.
	UNICODE	Like ASCII but for UNICODE.
Soundex	DIFFERENCE	An interesting way to compare differences in strings.
	SOUNDEX	An interesting way to compare strings.

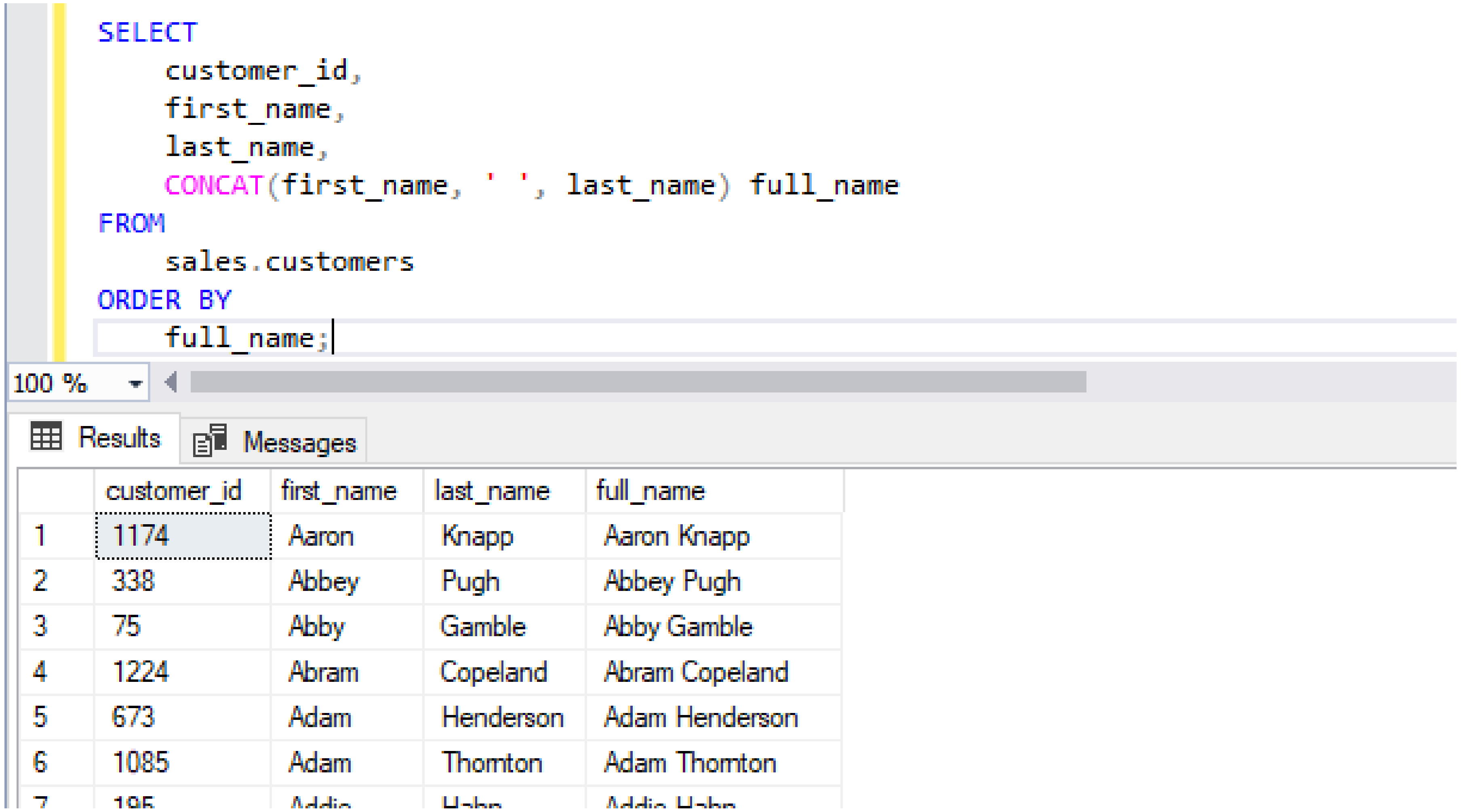
CONCAT FUNCTION

To join two or more strings into one you use the CONCAT() function

The syntax: CONCAT (input_string1, input_string2…)

The CONCAT() function also converts NULL into an empty string with the type VARCHAR(1).

CONCAT FUNCTION EXAMPLE



```sql
SELECT
    customer_id,
    first_name,
    last_name,
    CONCAT(first_name, ' ', last_name) full_name
FROM
    sales.customers
ORDER BY
    full_name;
```

	customer_id	first_name	last_name	full_name
1	1174	Aaron	Knapp	Aaron Knapp
2	338	Abbey	Pugh	Abbey Pugh
3	75	Abby	Gamble	Abby Gamble
4	1224	Abram	Copeland	Abram Copeland
5	673	Adam	Henderson	Adam Henderson
6	1085	Adam	Thornton	Adam Thornton
7	195	Addie	Hahn	Addie Hahn

CONCAT VS CONCAT_WS

The main difference between the two functions is *how they handle the separator* between the concatenated strings.

CONCAT() function takes two or more expressions as arguments and concatenates them together, without adding any separator between them.

CONCAT(*string1, string2, …., string_n*)

SELECT CONCAT('Hello', 'World'); -- Output: HelloWorld

On the other hand, CONCAT_WS() function also concatenates two or more strings, but it allows you to specify a separator between the strings. The first argument of the function is the separator, and subsequent arguments are the strings to concatenate.

CONCAT_WS(*separator, string1, string2, …., string_n*)

SELECT CONCAT_WS(',', 'John', 'Doe', '42'); -- Output: John,Doe,42

CONCAT_WS()

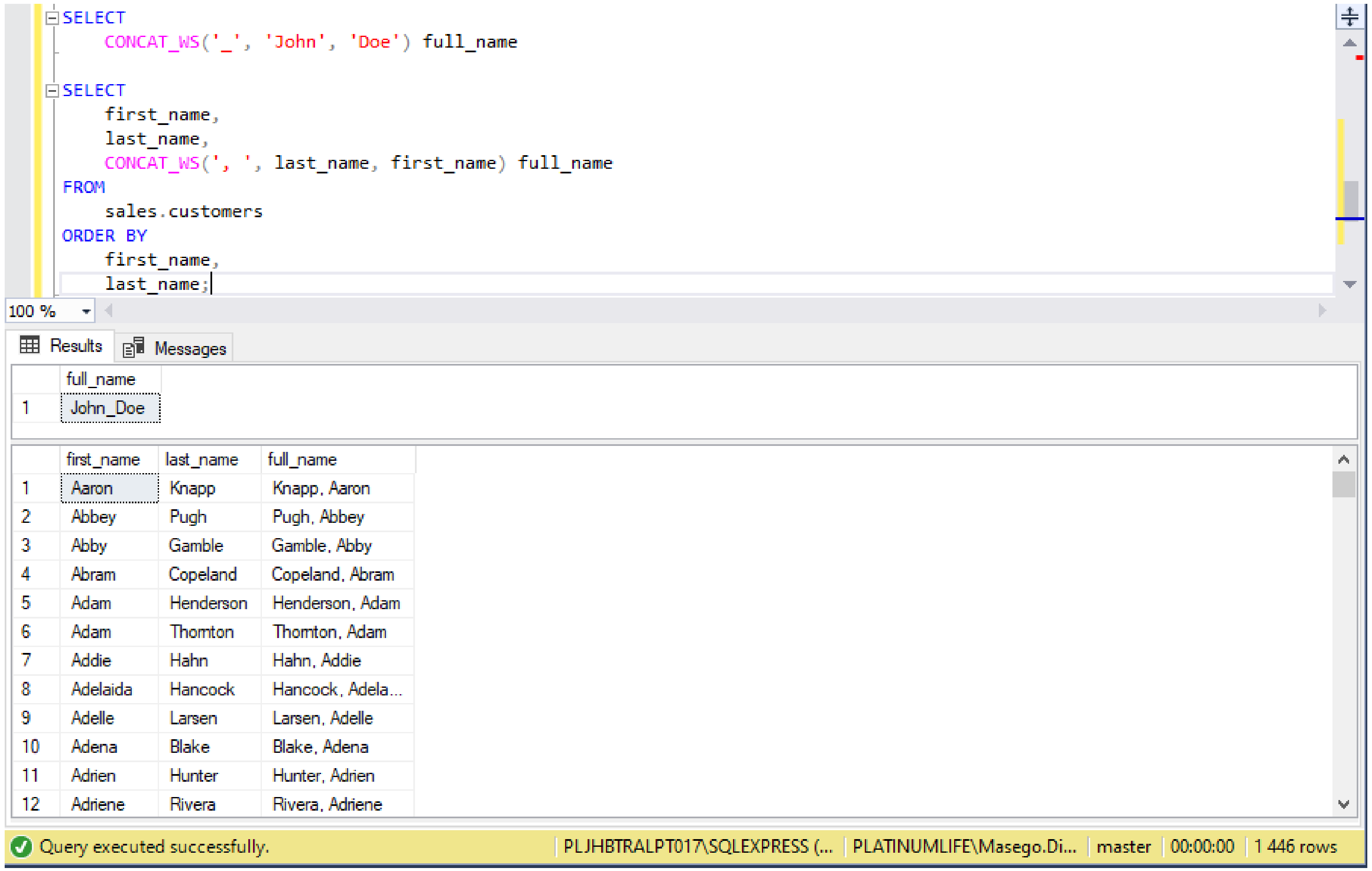
The SQL Server CONCAT_WS() function concatenates two or more strings into one string with a separator.

The syntax: CONCAT_WS(separator, input_string1, input_string2...);

1. The separator is a character-based expression that will be used to separate characters.

2. The input_string1 to input_stringN are expressions of any type. The CONCAT_WS() function implicitly converts values of non-character type to character type before concatenation.

CONCAT_WS() EXAMPLE



```sql
SELECT
    CONCAT_WS('_', 'John', 'Doe') full_name

SELECT
    first_name,
    last_name,
    CONCAT_WS(', ', last_name, first_name) full_name
FROM
    sales.customers
ORDER BY
    first_name,
    last_name;
```

100 %

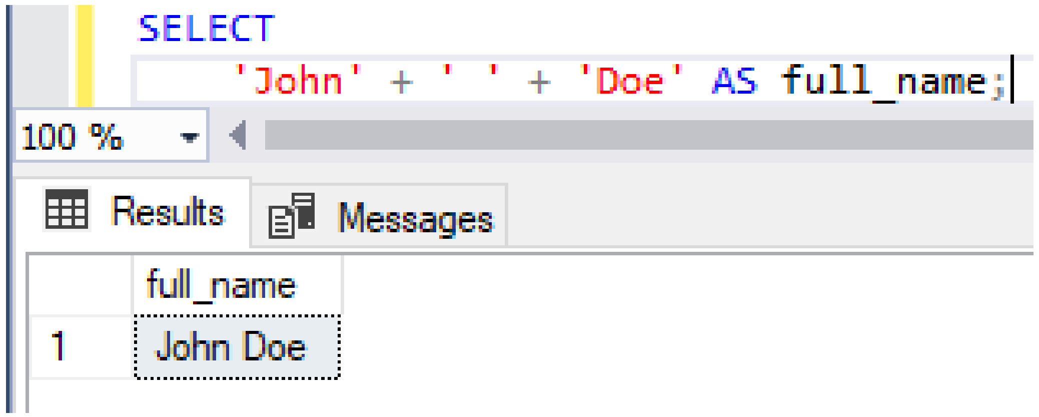
Results | Messages

	full_name
1	John_Doe

	first_name	last_name	full_name
1	Aaron	Knapp	Knapp, Aaron
2	Abbey	Pugh	Pugh, Abbey
3	Abby	Gamble	Gamble, Abby
4	Abram	Copeland	Copeland, Abram
5	Adam	Henderson	Henderson, Adam
6	Adam	Thornton	Thornton, Adam
7	Addie	Hahn	Hahn, Addie
8	Adelaida	Hancock	Hancock, Adela...
9	Adelle	Larsen	Larsen, Adelle
10	Adena	Blake	Blake, Adena
11	Adrien	Hunter	Hunter, Adrien
12	Adriene	Rivera	Rivera, Adriene

Query executed successfully. | PLJHBTRALPT017\SQLEXPRESS (... | PLATINUMLIFE\Masego.Di... | master | 00:00:00 | 1 446 rows

AN ALTERNATIVE OPTION TO CONCATENATE



```sql
SELECT
    'John' + ' ' + 'Doe' AS full_name;
```

```
   full_name
1  John Doe
```

CHARINDEX()

SQL Server CHARINDEX() function searches for a substring inside a string, starting from a specified location.

It returns the position of the substring found in the searched string or zero if the substring is not found. The starting position returned is 1-based not 0-based.

CHARINDEX()

The syntax: CHARINDEX(substring, string, [start_location])

1.The substring is the substring to search for. Its length is limited to 8 000 characters.

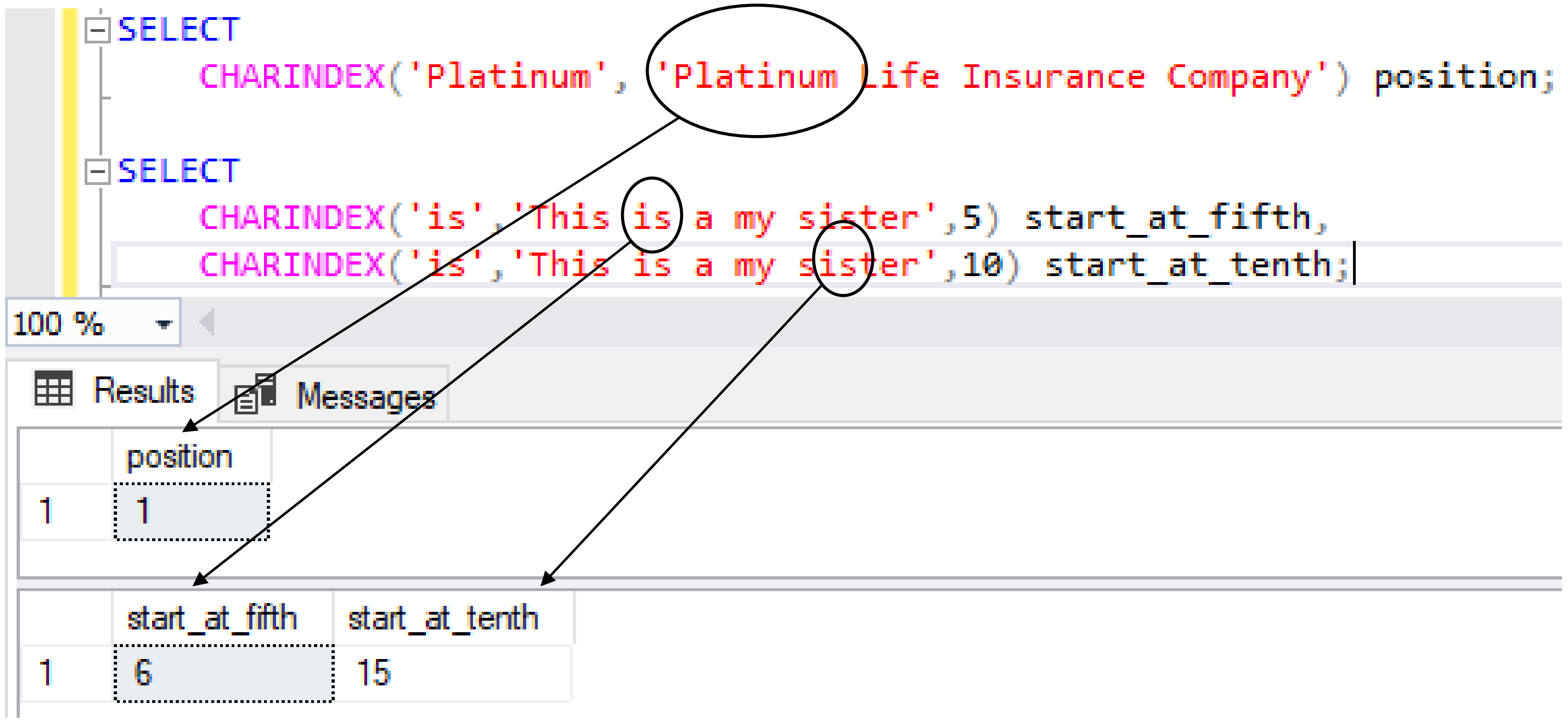
2. The string can be a literal string, expression or column. It is a string to search.

3. The start_location is the location at which the search starts.

4. The start_location parameter is optional. If it is skipped zero or negative value, the search starts at the beginning of the string.

Note that the CHARINDEX() function can perform both case-sensitive and case-insensitive searches based on the specified collation.

CHARINDEX() EXAMPLE



```
SELECT
    CHARINDEX('Platinum', 'Platinum Life Insurance Company') position;

SELECT
    CHARINDEX('is','This is a my sister',5) start_at_fifth,
    CHARINDEX('is','This is a my sister',10) start_at_tenth;
```

```
   position
1  1

   start_at_fifth  start_at_tenth
1  6               15
```

STRING_AGG()

The STRING_AGG() is an aggregate function that concatenate the values of a column into a single string, separated by a specified separator. It does not add the separator at the end of the result string.

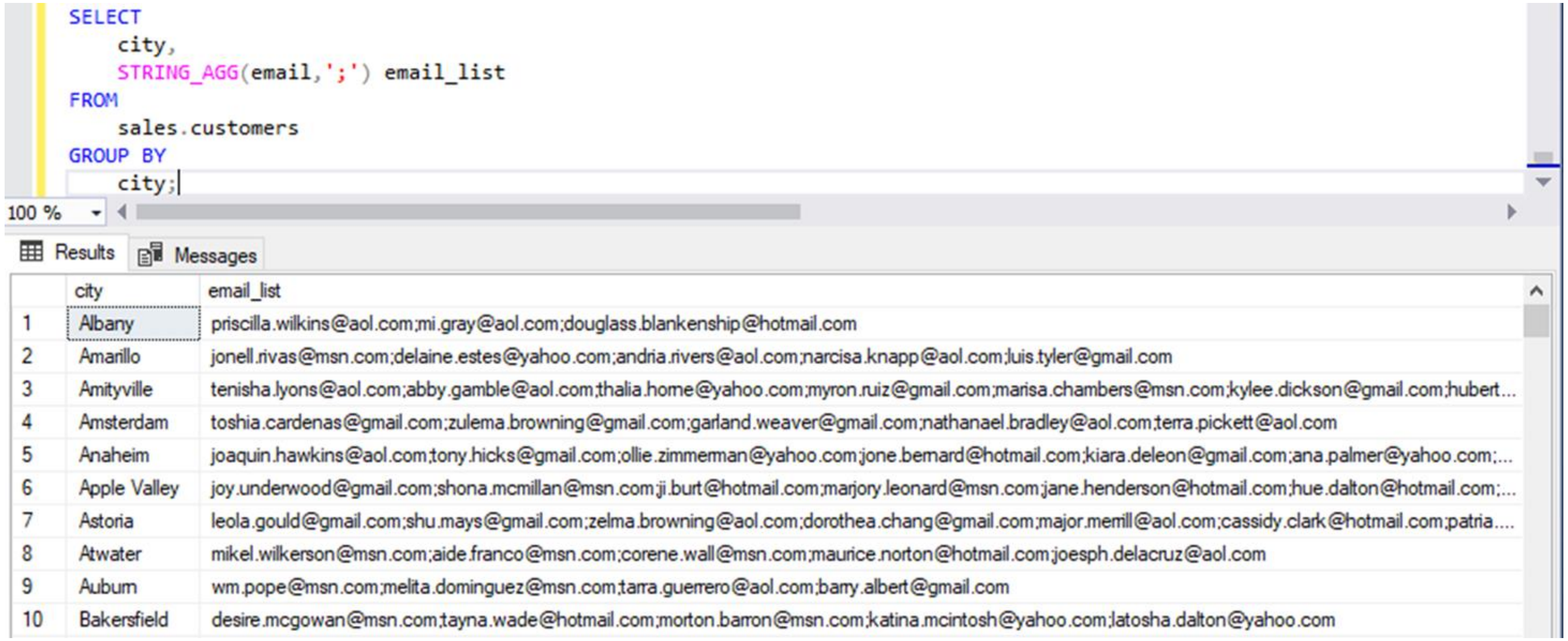
The syntax: STRING_AGG (input_string, separator)

```
Select Tutor, STRING_AGG(Studentname, “,”) Students

From Training

Group by Tutor
```

Tutor	**Students**
Ilze	Jack, Judith, Mary
Fortune	Lisa, George, Peter
Jayna	Elie, Lee, Tanya, Rose



```sql
SELECT
    city,
    STRING_AGG(email,';') email_list
FROM
    sales.customers
GROUP BY
    city;
```

	city	email_list
1	Albany	priscilla.wilkins@aol.com;mi.gray@aol.com;douglass.blankenship@hotmail.com
2	Amarillo	jonell.rivas@msn.com;delaine.estes@yahoo.com;andria.rivers@aol.com;narcisa.knapp@aol.com;luis.tyler@gmail.com
3	Amityville	tenisha.lyons@aol.com;abby.gamble@aol.com;thalia.horne@yahoo.com;myron.ruiz@gmail.com;marisa.chambers@msn.com;kylee.dickson@gmail.com;hubert...
4	Amsterdam	toshia.cardenas@gmail.com;zulema.browning@gmail.com;garland.weaver@gmail.com;nathanael.bradley@aol.com;terra.pickett@aol.com
5	Anaheim	joaquin.hawkins@aol.com;tony.hicks@gmail.com;ollie.zimmerman@yahoo.com;jone.bernard@hotmail.com;kiara.deleon@gmail.com;ana.palmer@yahoo.com;...
6	Apple Valley	joy.underwood@gmail.com;shona.mcmillan@msn.com;ji.burt@hotmail.com;marjory.leonard@msn.com;jane.henderson@hotmail.com;hue.dalton@hotmail.com;...
7	Astoria	leola.gould@gmail.com;shu.mays@gmail.com;zelma.browning@aol.com;dorothea.chang@gmail.com;major.merrill@aol.com;cassidy.clark@hotmail.com;patria....
8	Atwater	mikel.wilkerson@msn.com;aide.franco@msn.com;corene.wall@msn.com;maurice.norton@hotmail.com;joesph.delacruz@aol.com
9	Auburn	wm.pope@msn.com;melita.dominguez@msn.com;tarra.guerrero@aol.com;barry.albert@gmail.com
10	Bakersfield	desire.mcgowan@msn.com;tayna.wade@hotmail.com;morton.barron@msn.com;katina.mcintosh@yahoo.com;latosha.dalton@yahoo.com

LEFT()

The LEFT() function extracts a given number of characters from the left side of a supplied string. For example: LEFT(SQL Server, 3) returns SQL.

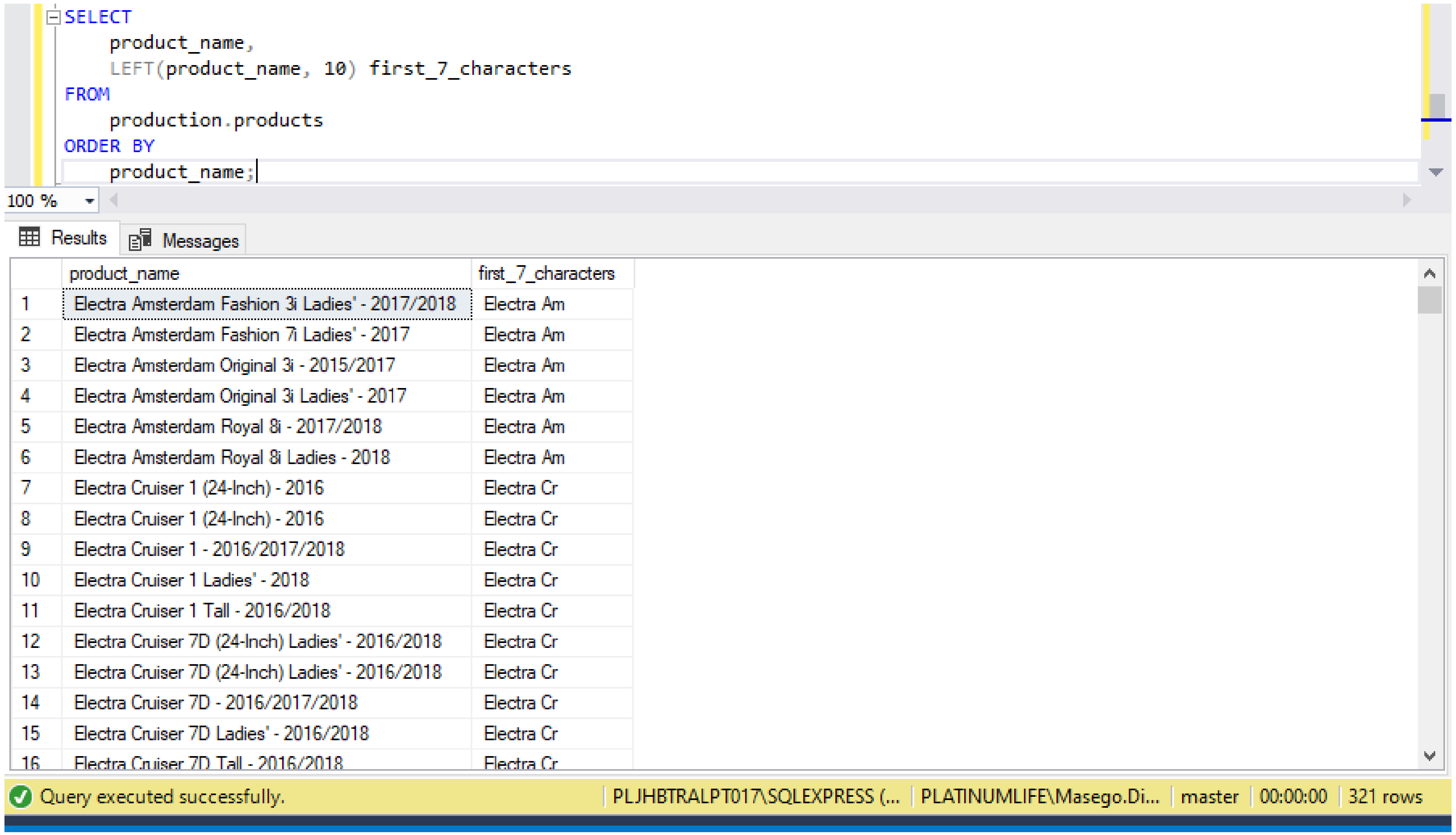
The syntax: LEFT(input_string, number_of_characters)

The number_of_characters is a positive integer that specifies the number of characters of the input_string will be returned.

Exercise:

Use the LEFT() function to return a set of initials of the product name (first letter/character of the product name) and count the number of each product for each initial. *#Hint you may use the group by clause.*

LEFT() EXAMPLE



```
SELECT
    product_name,
    LEFT(product_name, 10) first_7_characters
FROM
    production.products
ORDER BY
    product_name;
```

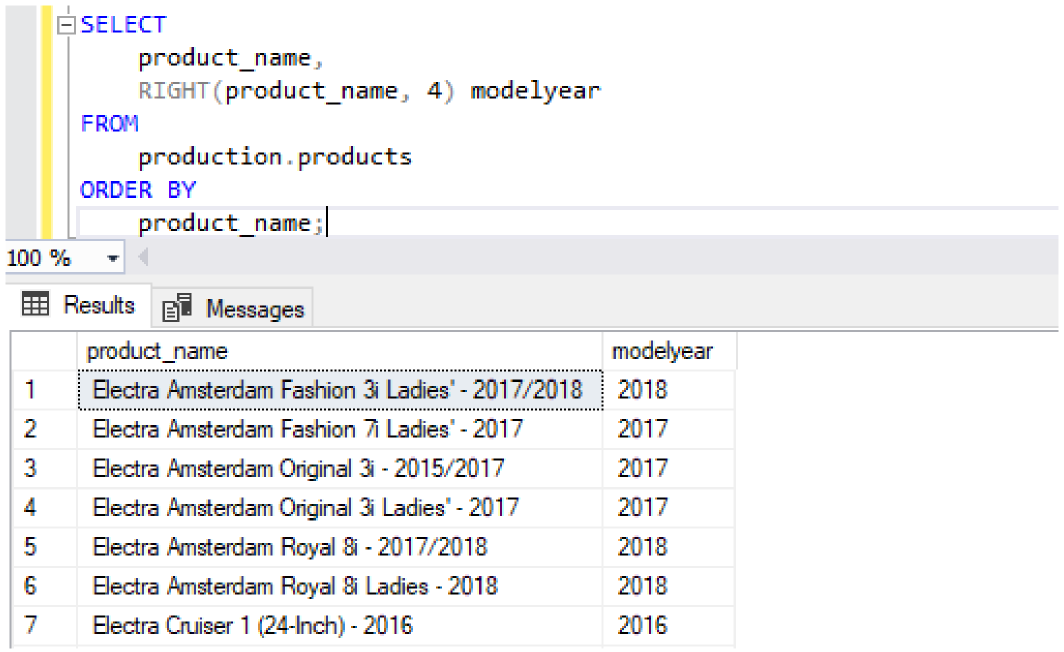
	product_name	first_7_characters
1	Electra Amsterdam Fashion 3i Ladies' - 2017/2018	Electra Am
2	Electra Amsterdam Fashion 7i Ladies' - 2017	Electra Am
3	Electra Amsterdam Original 3i - 2015/2017	Electra Am
4	Electra Amsterdam Original 3i Ladies' - 2017	Electra Am
5	Electra Amsterdam Royal 8i - 2017/2018	Electra Am
6	Electra Amsterdam Royal 8i Ladies - 2018	Electra Am
7	Electra Cruiser 1 (24-Inch) - 2016	Electra Cr
8	Electra Cruiser 1 (24-Inch) - 2016	Electra Cr
9	Electra Cruiser 1 - 2016/2017/2018	Electra Cr
10	Electra Cruiser 1 Ladies' - 2018	Electra Cr
11	Electra Cruiser 1 Tall - 2016/2018	Electra Cr
12	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	Electra Cr
13	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	Electra Cr
14	Electra Cruiser 7D - 2016/2017/2018	Electra Cr
15	Electra Cruiser 7D Ladies' - 2016/2018	Electra Cr
16	Electra Cruiser 7D Tall - 2016/2018	Electra Cr

Query executed successfully. | PLJHBTRALPT017\SQLEXPRESS (... | PLATINUMLIFE\Masego.Di... | master | 00:00:00 | 321 rows

RIGHT()

The RIGHT() function extracts a given number of characters from the right side of a specified character string.

The syntax:RIGHT (input_string   number_of_characters)



```
SELECT
    product_name,
    RIGHT(product_name, 4) modelyear
FROM
    production.products
ORDER BY
    product_name;
```

100 %

Results Messages

	product_name	modelyear
1	Electra Amsterdam Fashion 3i Ladies' - 2017/2018	2018
2	Electra Amsterdam Fashion 7i Ladies' - 2017	2017
3	Electra Amsterdam Original 3i - 2015/2017	2017
4	Electra Amsterdam Original 3i Ladies' - 2017	2017
5	Electra Amsterdam Royal 8i - 2017/2018	2018
6	Electra Amsterdam Royal 8i Ladies - 2018	2018
7	Electra Cruiser 1 (24-Inch) - 2016	2016

SUBSTRING()

The SUBSTRING() extracts a substring with a specified length starting from a location in an input string.

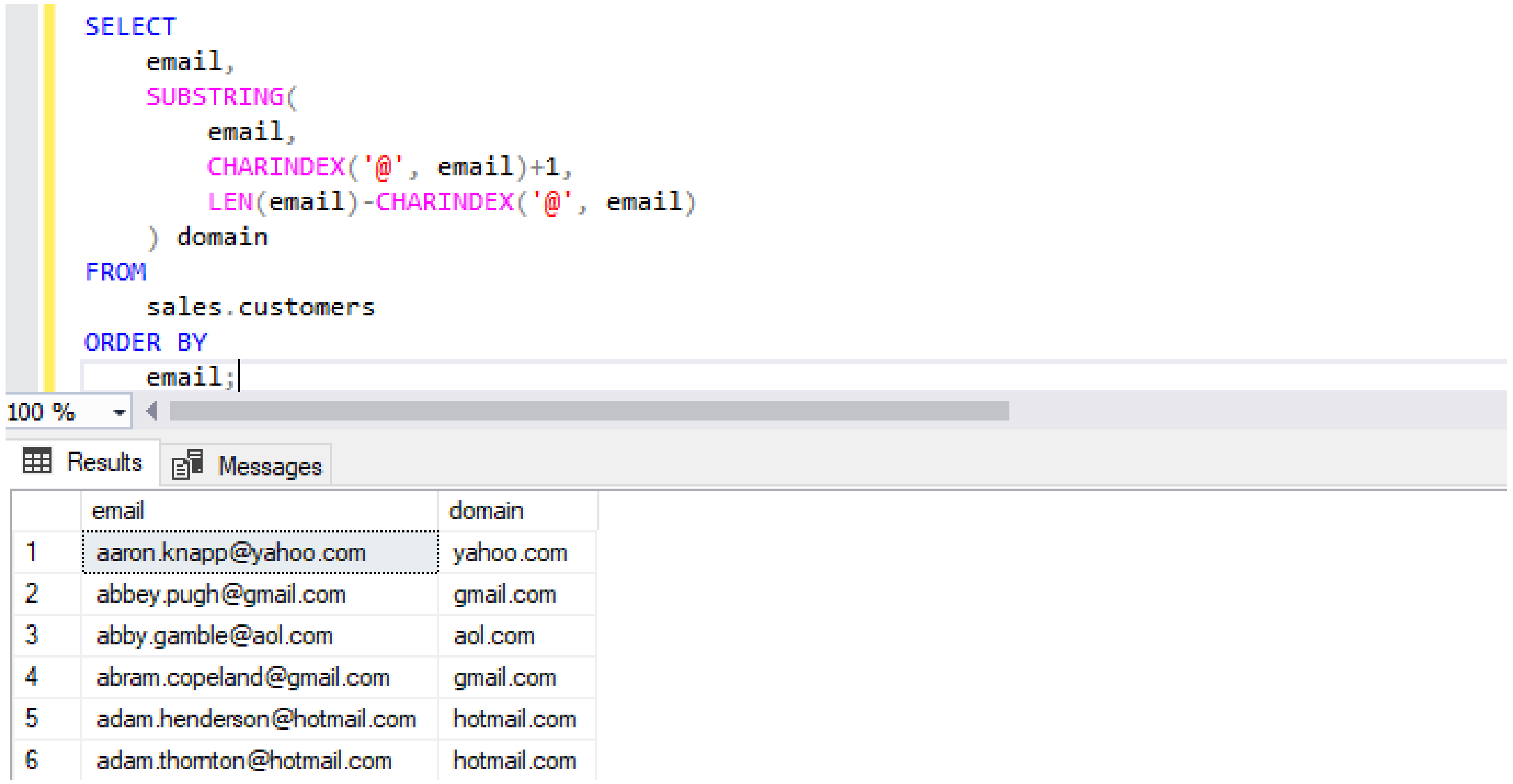
The syntax: SUBSTRING(*string*, *start*, *length*)

1.The column_name is the field to extract characters from

2. Start is the specific the starting position (starts at 1)

3. The length is the number of characters to return and it s usually optional. If omitted, the SUBSTRING() function returns the rest of the text

SUBSTRING() EXAMPLE



```
SELECT
    email,
    SUBSTRING(
        email,
        CHARINDEX('@', email)+1,
        LEN(email)-CHARINDEX('@', email)
    ) domain
FROM
    sales.customers
ORDER BY
    email;
```

100 %

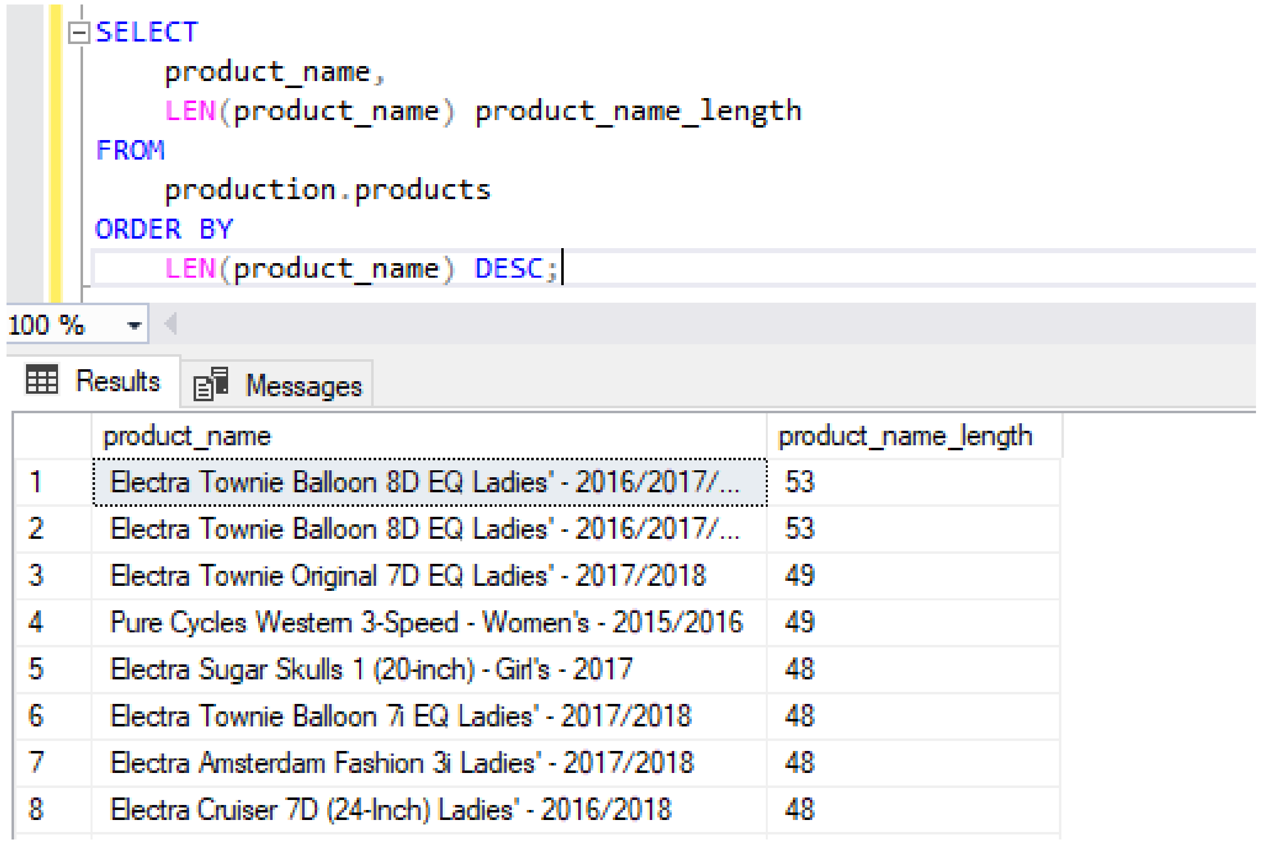
Results | Messages

```
   email                        domain
1  aaron.knapp@yahoo.com        yahoo.com
2  abbey.pugh@gmail.com         gmail.com
3  abby.gamble@aol.com          aol.com
4  abram.copeland@gmail.com     gmail.com
5  adam.henderson@hotmail.com   hotmail.com
6  adam.thornton@hotmail.com    hotmail.com
```

LEN()

The LEN() function returns the number of characters of an input string excluding the trailing blanks.

The syntax: LEN(input_string)



```
SELECT
    product_name,
    LEN(product_name) product_name_length
FROM
    production.products
ORDER BY
    LEN(product_name) DESC;
```

	product_name	product_name_length
1	Electra Townie Balloon 8D EQ Ladies' - 2016/2017/...	53
2	Electra Townie Balloon 8D EQ Ladies' - 2016/2017/...	53
3	Electra Townie Original 7D EQ Ladies' - 2017/2018	49
4	Pure Cycles Western 3-Speed - Women's - 2015/2016	49
5	Electra Sugar Skulls 1 (20-inch) - Girl's - 2017	48
6	Electra Townie Balloon 7i EQ Ladies' - 2017/2018	48
7	Electra Amsterdam Fashion 3i Ladies' - 2017/2018	48
8	Electra Cruiser 7D (24-Inch) Ladies' - 2016/2018	48

LOWER() & UPPER()

The LOWER() function converts a string into lowercase.

The syntax: LOWER(input_string)

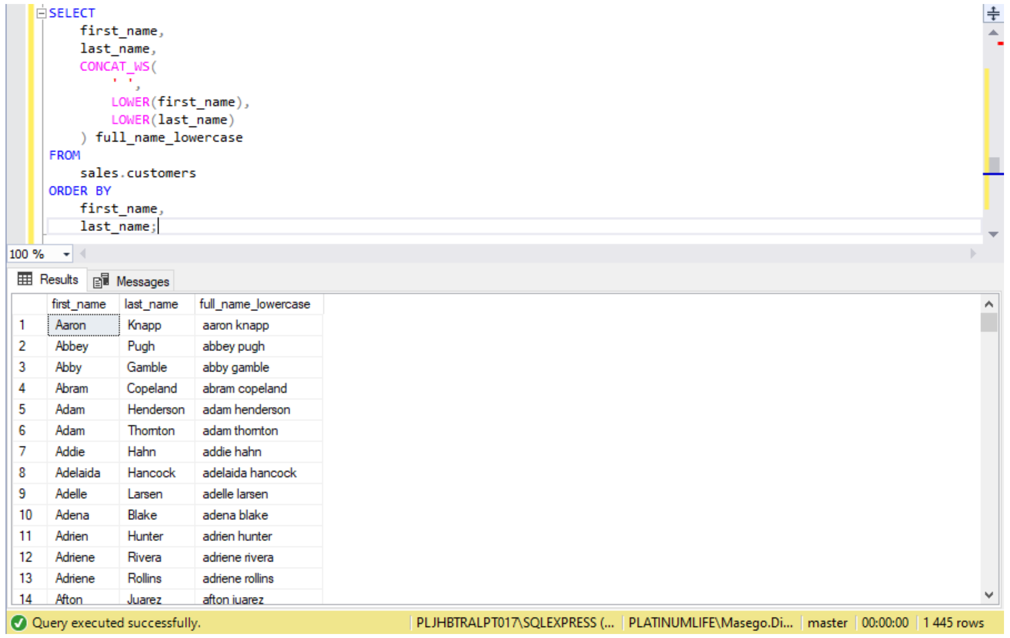
The UPPER() function converts an input string into uppercase.

The syntax: UPPER(input_string)

LOWER() & UPPER() EXAMPLE

Lower()

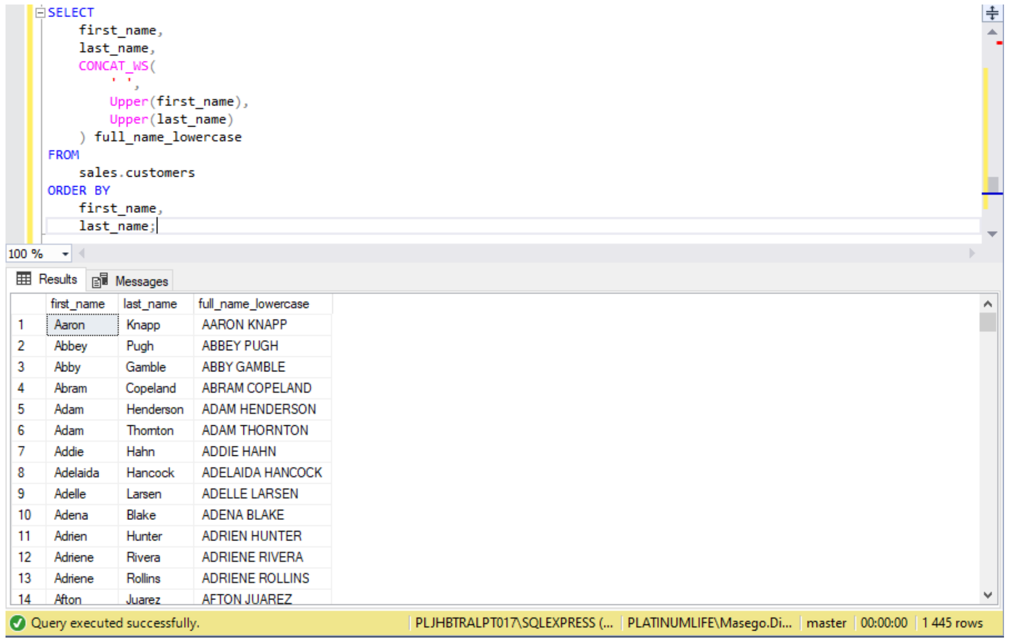
Upper()



```
SELECT
    first_name,
    last_name,
    CONCAT_WS(
        ' ',
        LOWER(first_name),
        LOWER(last_name)
    ) full_name_lowercase
FROM
    sales.customers
ORDER BY
    first_name,
    last_name;
```

	first_name	last_name	full_name_lowercase
1	Aaron	Knapp	aaron knapp
2	Abbey	Pugh	abbey pugh
3	Abby	Gamble	abby gamble
4	Abram	Copeland	abram copeland
5	Adam	Henderson	adam henderson
6	Adam	Thornton	adam thornton
7	Addie	Hahn	addie hahn
8	Adelaida	Hancock	adelaida hancock
9	Adelle	Larsen	adelle larsen
10	Adena	Blake	adena blake
11	Adrien	Hunter	adrien hunter
12	Adriene	Rivera	adriene rivera
13	Adriene	Rollins	adriene rollins
14	Afton	Juarez	afton juarez

Query executed successfully. | PLJHBTRALPT017\SQLEXPRESS (... | PLATINUMLIFE\Masego.Di... | master | 00:00:00 | 1 445 rows



```
SELECT
    first_name,
    last_name,
    CONCAT_WS(
        ' ',
        Upper(first_name),
        Upper(last_name)
    ) full_name_lowercase
FROM
    sales.customers
ORDER BY
    first_name,
    last_name;
```

	first_name	last_name	full_name_lowercase
1	Aaron	Knapp	AARON KNAPP
2	Abbey	Pugh	ABBEY PUGH
3	Abby	Gamble	ABBY GAMBLE
4	Abram	Copeland	ABRAM COPELAND
5	Adam	Henderson	ADAM HENDERSON
6	Adam	Thornton	ADAM THORNTON
7	Addie	Hahn	ADDIE HAHN
8	Adelaida	Hancock	ADELAIDA HANCOCK
9	Adelle	Larsen	ADELLE LARSEN
10	Adena	Blake	ADENA BLAKE
11	Adrien	Hunter	ADRIEN HUNTER
12	Adriene	Rivera	ADRIENE RIVERA
13	Adriene	Rollins	ADRIENE ROLLINS
14	Afton	Juarez	AFTON JUAREZ

Query executed successfully. | PLJHBTRALPT017\SQLEXPRESS (... | PLATINUMLIFE\Masego.Di... | master | 00:00:00 | 1 445 rows

LTRIM() & RTRIM() & TRIM()

The LTRIM() function returns a string after removing leading blanks.

The syntax: LTRIM(input_string)

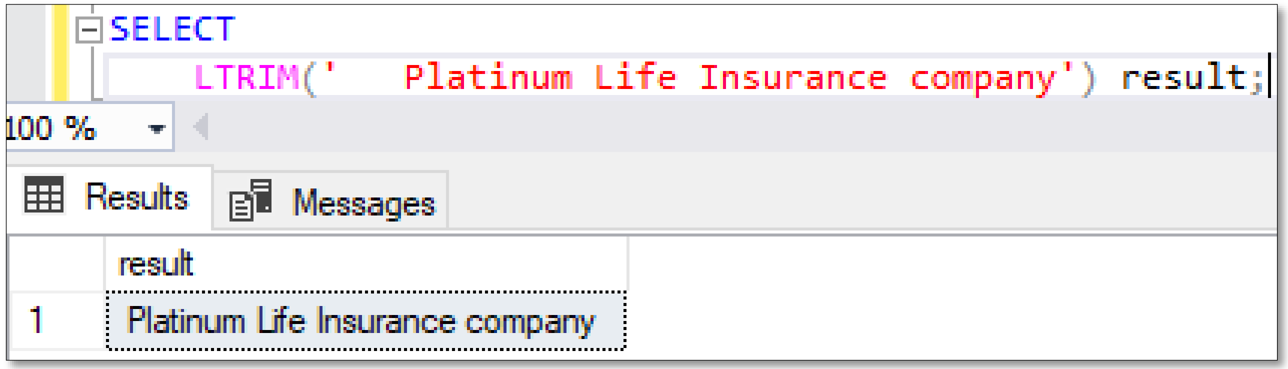
The RTRIM() function returns a string after truncating all trailing blanks.

the syntax: RTRIM(input_string)

The TRIM() function removes spaces or specified characters from both ends of a string.

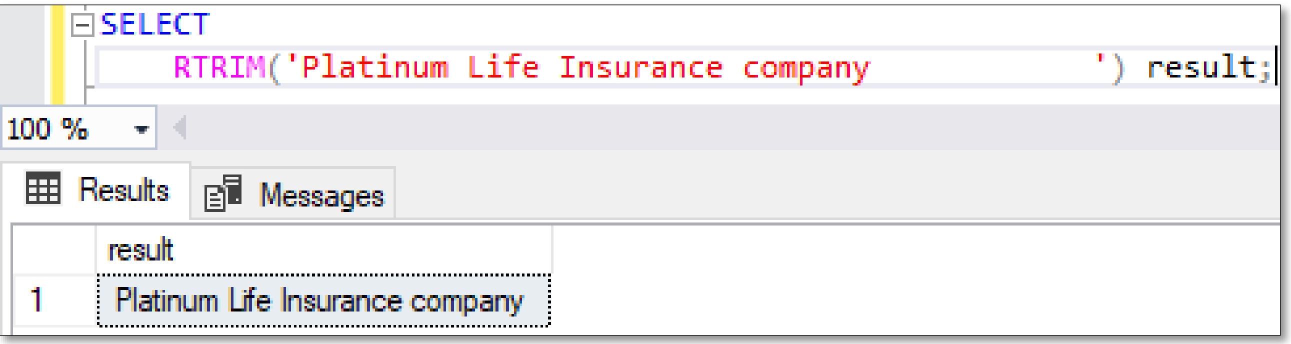
The syntax: TRIM([removed_characters FROM] input_string)

LTRIM() & RTRIM() & TRIM() EXAMPLES



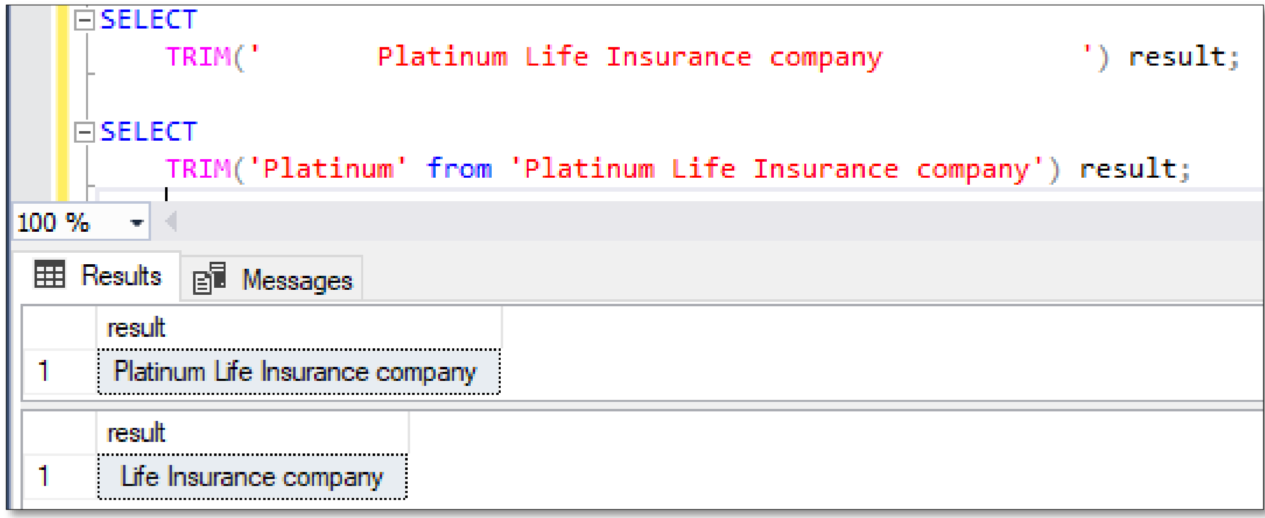
```
SELECT
    LTRIM('    Platinum Life Insurance company') result;
```

```
100 %
Results   Messages
   result
1  Platinum Life Insurance company
```



```
SELECT
    RTRIM('Platinum Life Insurance company              ') result;
```

```
100 %
Results   Messages
   result
1  Platinum Life Insurance company
```



```
SELECT
    TRIM('        Platinum Life Insurance company              ') result;

SELECT
    TRIM('Platinum' from 'Platinum Life Insurance company') result;
```

```
100 %
Results   Messages
   result
1  Platinum Life Insurance company

   result
1  Life Insurance company
```

TRIM

```
SELECT TRIM('    Hello    ') AS Result; -- Output: 'Hello'
SELECT TRIM(LEADING '0' FROM '0000123') AS Result; -- Output: '123'
SELECT TRIM(TRAILING '!' FROM 'Hello!!!') AS Result; -- Output: 'Hello'
```

REPLACE()

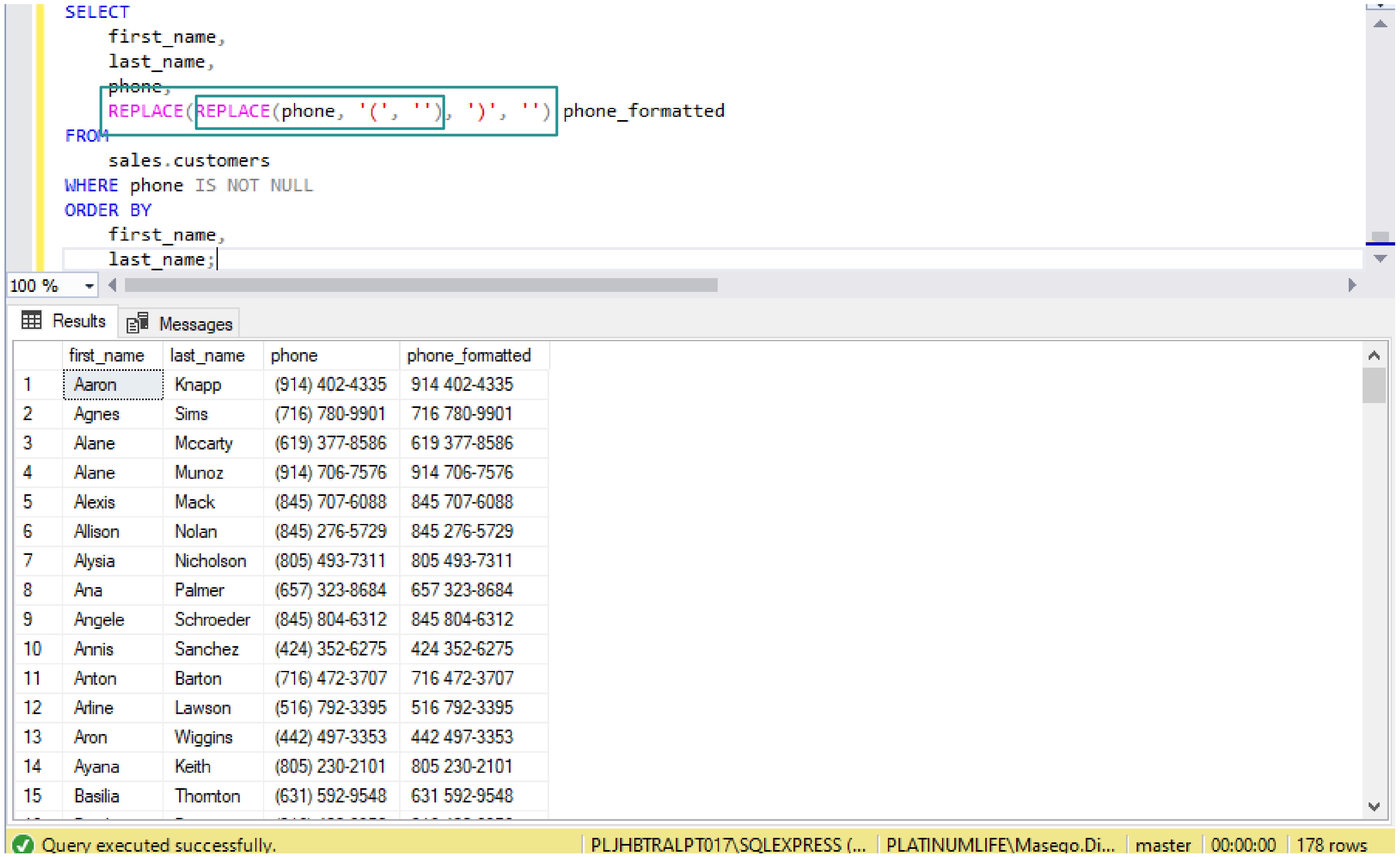
REPLACE() function to replace all occurrences of a substring by a new substring within a string.

The syntax: REPLACE(input_string, substring, new_substring);

1. The input_string is any string expression to be searched.
2. The substring is the substring to be replaced.
3. The new_substring is the replacement string.

The REPLACE() function returns a new string in which all occurrences of the substring are replaced by the new_substring. It returns NULL if any argument is NULL.

REPLACE() EXAMPLE



```
SELECT
    first_name,
    last_name,
    phone,
    REPLACE(REPLACE(phone, '(', ''), ')', '') phone_formatted
FROM
    sales.customers
WHERE phone IS NOT NULL
ORDER BY
    first_name,
    last_name;
```

100 %

Results | Messages

	first_name	last_name	phone	phone_formatted
1	Aaron	Knapp	(914) 402-4335	914 402-4335
2	Agnes	Sims	(716) 780-9901	716 780-9901
3	Alane	Mccarty	(619) 377-8586	619 377-8586
4	Alane	Munoz	(914) 706-7576	914 706-7576
5	Alexis	Mack	(845) 707-6088	845 707-6088
6	Allison	Nolan	(845) 276-5729	845 276-5729
7	Alysia	Nicholson	(805) 493-7311	805 493-7311
8	Ana	Palmer	(657) 323-8684	657 323-8684
9	Angele	Schroeder	(845) 804-6312	845 804-6312
10	Annis	Sanchez	(424) 352-6275	424 352-6275
11	Anton	Barton	(716) 472-3707	716 472-3707
12	Arline	Lawson	(516) 792-3395	516 792-3395
13	Aron	Wiggins	(442) 497-3353	442 497-3353
14	Ayana	Keith	(805) 230-2101	805 230-2101
15	Basilia	Thornton	(631) 592-9548	631 592-9548

Query executed successfully. | PLJHBTRALPT017\SQLEXPRESS (... | PLATINUMLIFE\Masego.Di... | master | 00:00:00 | 178 rows

Practice STRING FUNCTIONS

Using the L&D Company database

1. Return the full name of the employees by combining the first name and last name
2. Return the full name of the employees by concatenating the first name and last names of all the accountants by using joins
3. Encode all the email addresses for all the employees using the appropriate function.
4. Decode all the email addresses for all the employees using the appropriate function.

L&D SHOP

COALESCE

The SQL Coalesce function lets you return an alternative value if an expression is NULL → in this case a 0

```
 SELECT ProductName,  UnitPrice * (UnitsInStock +
COALESCE(UnitsOnOrder,  0)) FROM Products;
```

```
 SELECT ProductName,  UnitPrice * (UnitsInStock +
IFNULL(UnitsOnOrder,  0)) FROM Products;
```

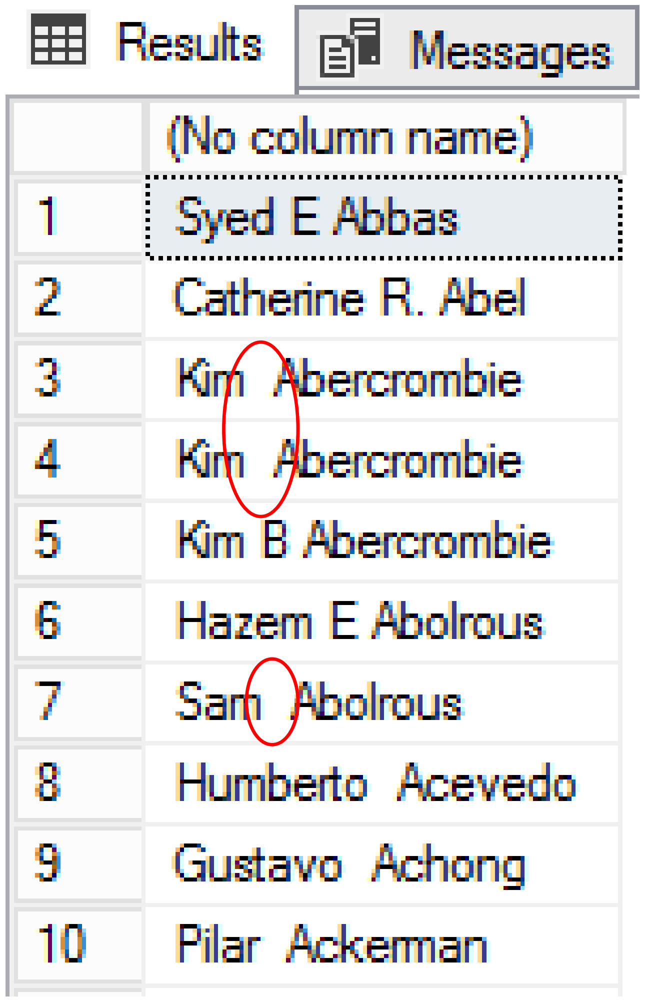
COALESCE

```
SELECT firstName + '   ' + MiddleName + '     ' +
LastName FROM Person
```

If they don t have a middle name (if middle name column = null) then this will result in a null value for this calculation however:

```
SELECT firstName + ' ' +COALESCE(MiddleName) +'    '
+ LastName  FROM Person
```

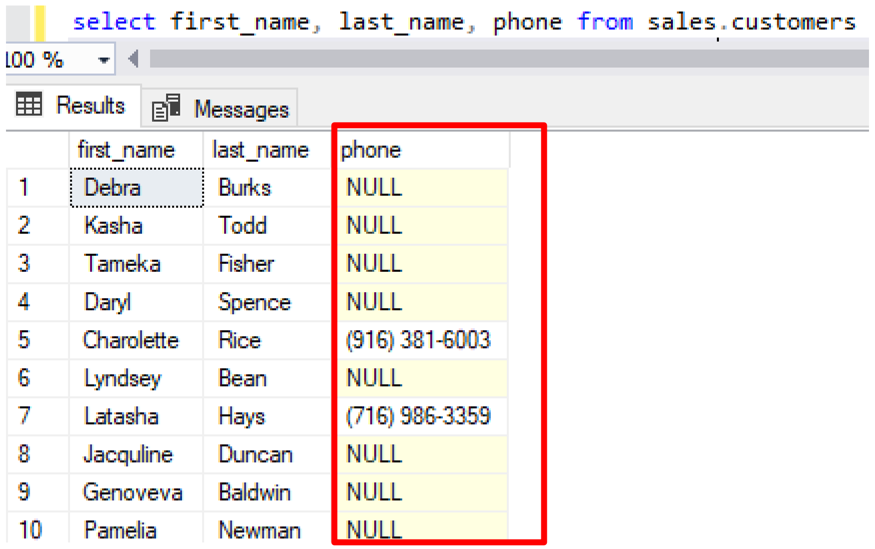
This will now only add the middle name if there is one and a space if they dont have a middle name


Results | Messages

	(No column name)
1	Syed E Abbas
2	Catherine R. Abel
3	Kim Abercrombie
4	Kim Abercrombie
5	Kim B Abercrombie
6	Hazem E Abolrous
7	Sam Abolrous
8	Humberto Acevedo
9	Gustavo Achong
10	Pilar Ackerman

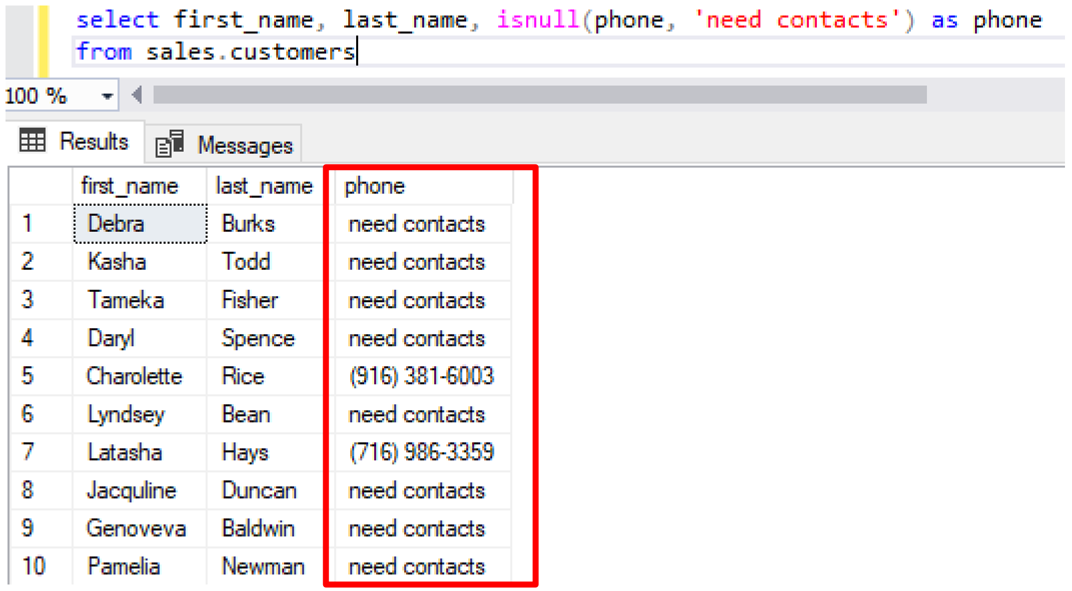
NULL FUNCTIONS

The NULL or ISNULL() function is used to replace the NULL values with a value or text.



```
select first_name, last_name, phone from sales.customers
```

	first_name	last_name	phone
1	Debra	Burks	NULL
2	Kasha	Todd	NULL
3	Tameka	Fisher	NULL
4	Daryl	Spence	NULL
5	Charolette	Rice	(916) 381-6003
6	Lyndsey	Bean	NULL
7	Latasha	Hays	(716) 986-3359
8	Jacquline	Duncan	NULL
9	Genoveva	Baldwin	NULL
10	Pamelia	Newman	NULL



```
select first_name, last_name, isnull(phone, 'need contacts') as phone
from sales.customers
```

	first_name	last_name	phone
1	Debra	Burks	need contacts
2	Kasha	Todd	need contacts
3	Tameka	Fisher	need contacts
4	Daryl	Spence	need contacts
5	Charolette	Rice	(916) 381-6003
6	Lyndsey	Bean	need contacts
7	Latasha	Hays	(716) 986-3359
8	Jacquline	Duncan	need contacts
9	Genoveva	Baldwin	need contacts
10	Pamelia	Newman	need contacts

NULL FUNCTION CONTINUED..

Original:

```
SELECT ProductName,  UnitPrice * (UnitsInStock + UnitsOnOrder)
FROM Products;
```

Example of using the isnull:

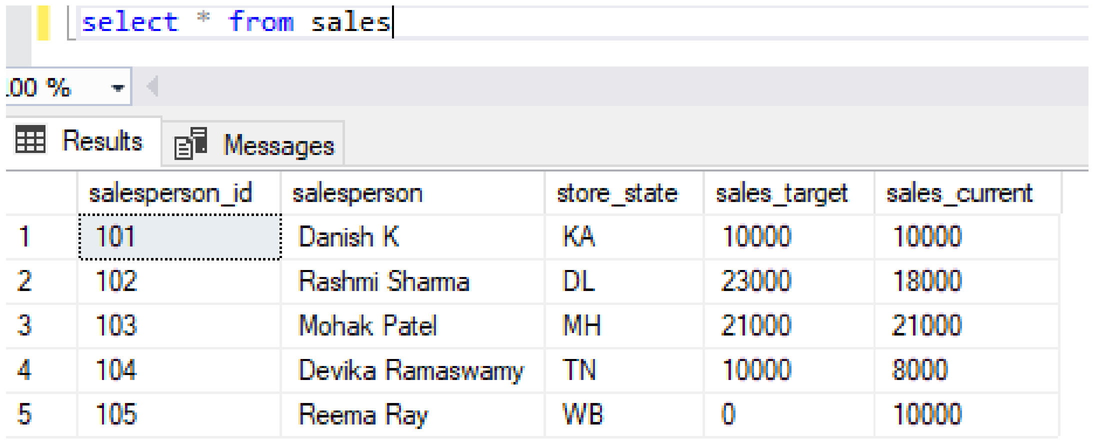
```
 SELECT ProductName,  UnitPrice * (UnitsInStock +
IF(IsNull(UnitsOnOrder),  0,  UnitsOnOrder))
FROM Products;
```

NULLIF

When two expressions are compared the NULLIF() method is used.

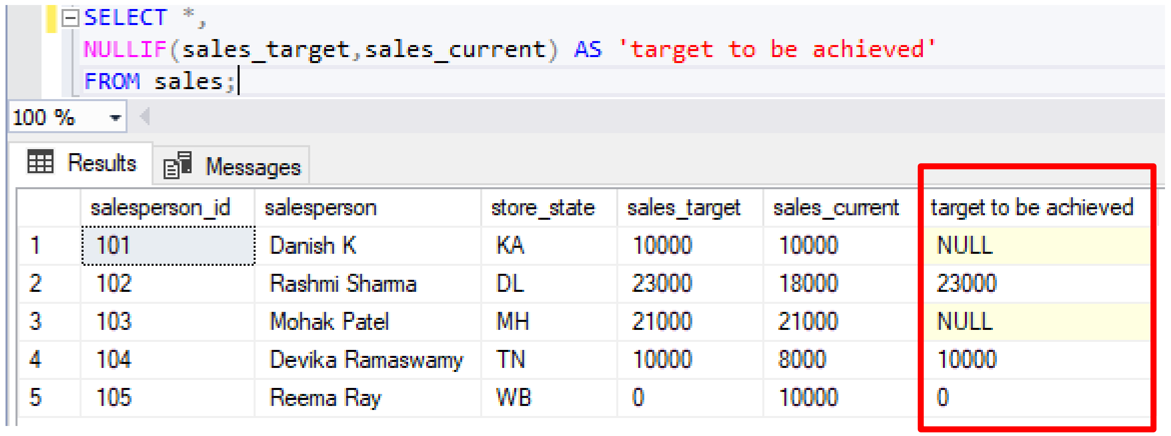
If both expressions are equal the NULLIF() method returns NULL;

If both expressions are not equal it returns the first expression.



```
select * from sales
```

	salesperson_id	salesperson	store_state	sales_target	sales_current
1	101	Danish K	KA	10000	10000
2	102	Rashmi Sharma	DL	23000	18000
3	103	Mohak Patel	MH	21000	21000
4	104	Devika Ramaswamy	TN	10000	8000
5	105	Reema Ray	WB	0	10000



```
SELECT *,
NULLIF(sales_target,sales_current) AS 'target to be achieved'
FROM sales;
```

	salesperson_id	salesperson	store_state	sales_target	sales_current	target to be achieved
1	101	Danish K	KA	10000	10000	NULL
2	102	Rashmi Sharma	DL	23000	18000	23000
3	103	Mohak Patel	MH	21000	21000	NULL
4	104	Devika Ramaswamy	TN	10000	8000	10000
5	105	Reema Ray	WB	0	10000	0