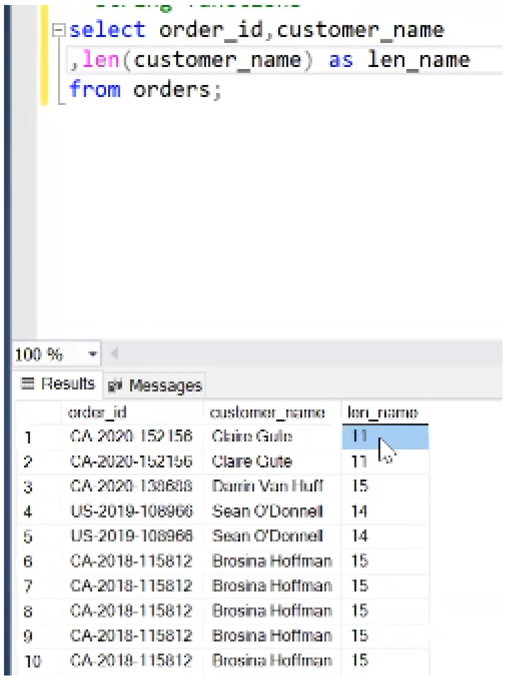
String Functions

1. Here to perform string operations..lets take string related cols from orders
2. First is len function



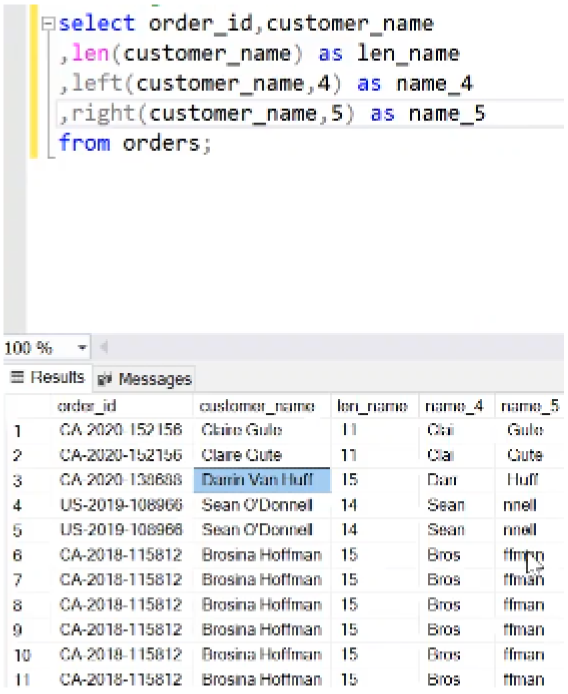
```
select order_id,customer_name
,len(customer_name) as len_name
from orders;
```

100 %

Results | Messages

	order_id	customer_name	len_name
1	CA-2020-152156	Claire Gute	11
2	CA-2020-152156	Claire Gute	11
3	CA-2020-138688	Darrin Van Huff	15
4	US-2019-108966	Sean O'Donnell	14
5	US-2019-108966	Sean O'Donnell	14
6	CA-2018-115812	Brosina Hoffman	15
7	CA-2018-115812	Brosina Hoffman	15
8	CA-2018-115812	Brosina Hoffman	15
9	CA-2018-115812	Brosina Hoffman	15
10	CA-2018-115812	Brosina Hoffman	15

3. here we got length of each customer name..using len function
4. Then next we have **left(col_name,4)** which gives the 4 char's starting from left..
5. Similarly we have **right(col_name,4)** which gives 4 chars starting from right



```
select order_id,customer_name
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
from orders;
```

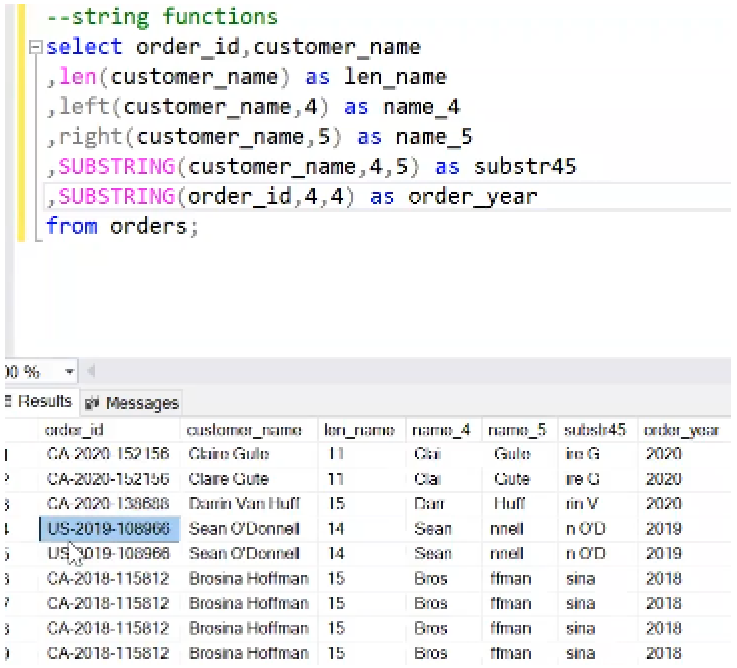
100 %

Results | Messages

	order_id	customer_name	len_name	name_4	name_5
1	CA-2020-152156	Claire Gute	11	Clai	Gute
2	CA-2020-152156	Claire Gute	11	Clai	Gute
3	CA-2020-138688	Darrin Van Huff	15	Darr	Huff
4	US-2019-108966	Sean O'Donnell	14	Sean	nnell
5	US-2019-108966	Sean O'Donnell	14	Sean	nnell
6	CA-2018-115812	Brosina Hoffman	15	Bros	ffman
7	CA-2018-115812	Brosina Hoffman	15	Bros	ffman
8	CA-2018-115812	Brosina Hoffman	15	Bros	ffman
9	CA-2018-115812	Brosina Hoffman	15	Bros	ffman
10	CA-2018-115812	Brosina Hoffman	15	Bros	ffman
11	CA-2018-115812	Brosina Hoffman	15	Bros	ffman

6.
7. Next we have substring..which takes col_name and a point to start at and len of chars

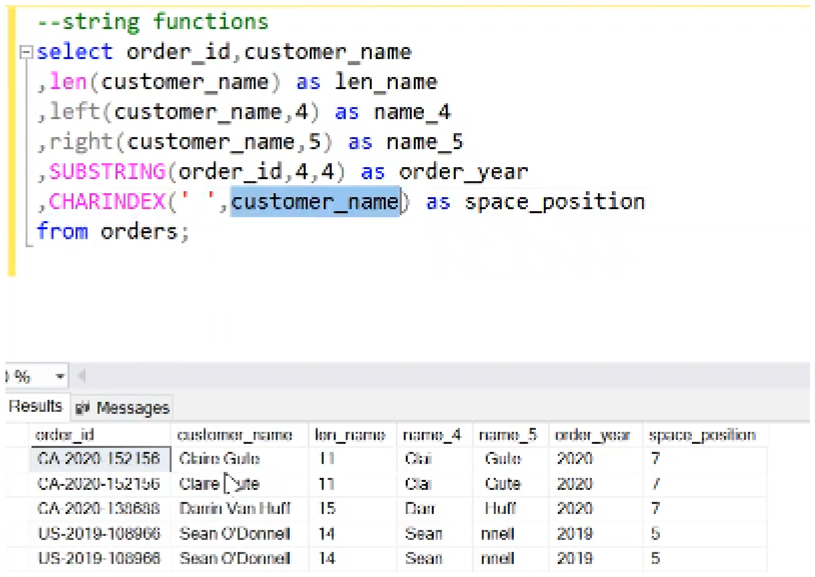
8. For example **substring(order_id,4,4)** ..here in the order_id col..it will retrieves char from index 4 till the length 4…see code pic



```
--string functions
select order_id,customer_name
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
,SUBSTRING(customer_name,4,5) as substr45
,SUBSTRING(order_id,4,4) as order_year
from orders;
```

order_id	customer_name	len_name	name_4	name_5	substr45	order_year
CA-2020-152156	Claire Gute	11	Clai	Gute	ire G	2020
CA-2020-152156	Claire Gute	11	Clai	Gute	ire G	2020
CA-2020-138688	Darrin Van Huff	15	Darr	Huff	rin V	2020
US-2019-108966	Sean O'Donnell	14	Sean	nnell	n O'D	2019
US-2019-108966	Sean O'Donnell	14	Sean	nnell	n O'D	2019
CA-2018-115812	Brosina Hoffman	15	Bros	ffman	sina	2018
CA-2018-115812	Brosina Hoffman	15	Bros	ffman	sina	2018
CA-2018-115812	Brosina Hoffman	15	Bros	ffman	sina	2018
CA-2018-115812	Brosina Hoffman	15	Bros	ffman	sina	2018

9. Next we have is **charindex** ..it takes a char and a column name…and gives us the index of that character in that column..see pic

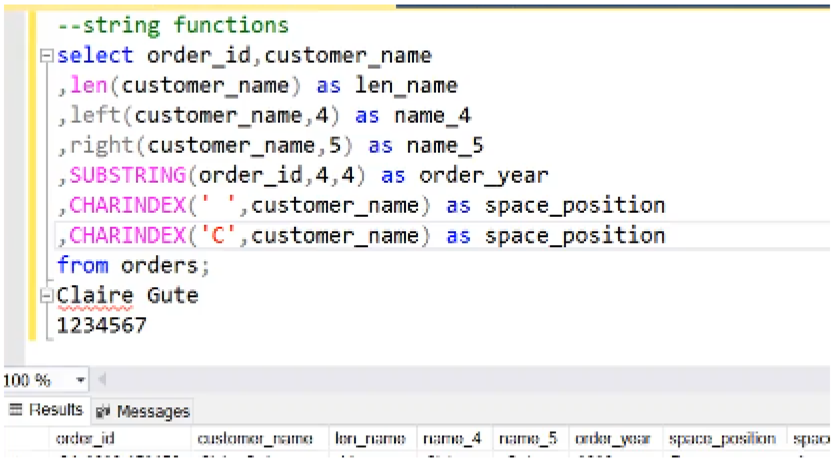


```
--string functions
select order_id,customer_name
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
,SUBSTRING(order_id,4,4) as order_year
,CHARINDEX(' ',customer_name) as space_position
from orders;
```

order_id	customer_name	len_name	name_4	name_5	order_year	space_position
CA-2020-152156	Claire Gute	11	Clai	Gute	2020	7
CA-2020-152156	Claire Gute	11	Clai	Gute	2020	7
CA-2020-138688	Darrin Van Huff	15	Darr	Huff	2020	7
US-2019-108966	Sean O'Donnell	14	Sean	nnell	2019	5
US-2019-108966	Sean O'Donnell	14	Sean	nnell	2019	5

10.

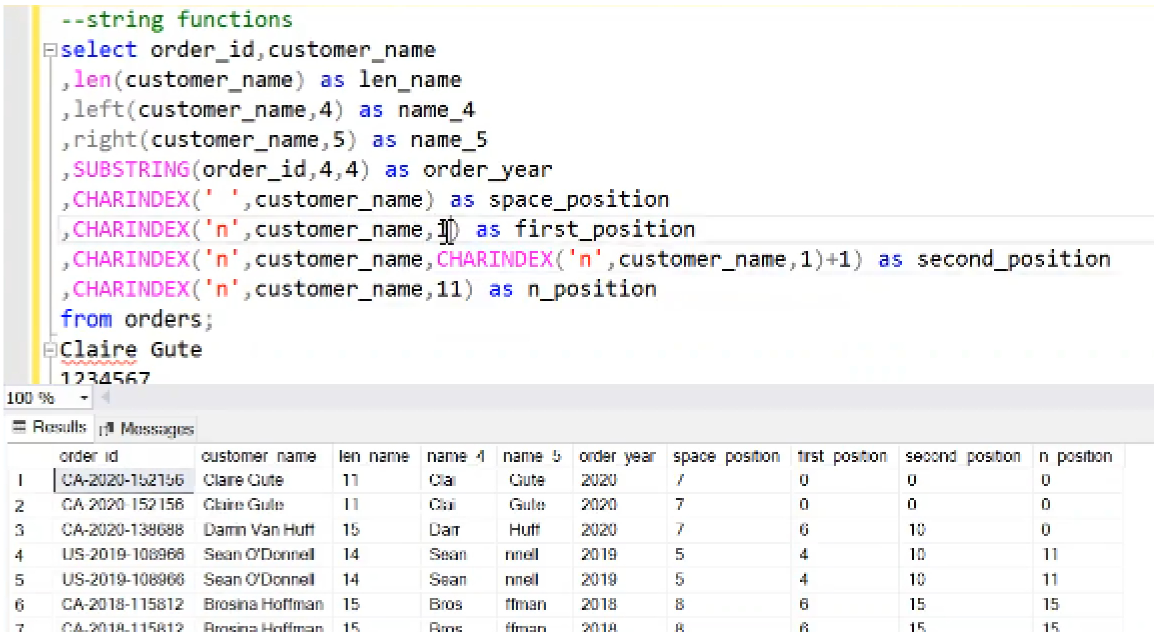
11. If the char is not present…then it results in 0



```
--string functions
select order_id,customer_name
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
,SUBSTRING(order_id,4,4) as order_year
,CHARINDEX(' ',customer_name) as space_position
,CHARINDEX('C',customer_name) as space_position
from orders;
Claire Gute
1234567
```

```
order_id   customer_name   len_name   name_4   name_5   order_year   space_position   spac
```

12. To find multiple occurrences of char..we used(see pic)…but there also work arounds for this

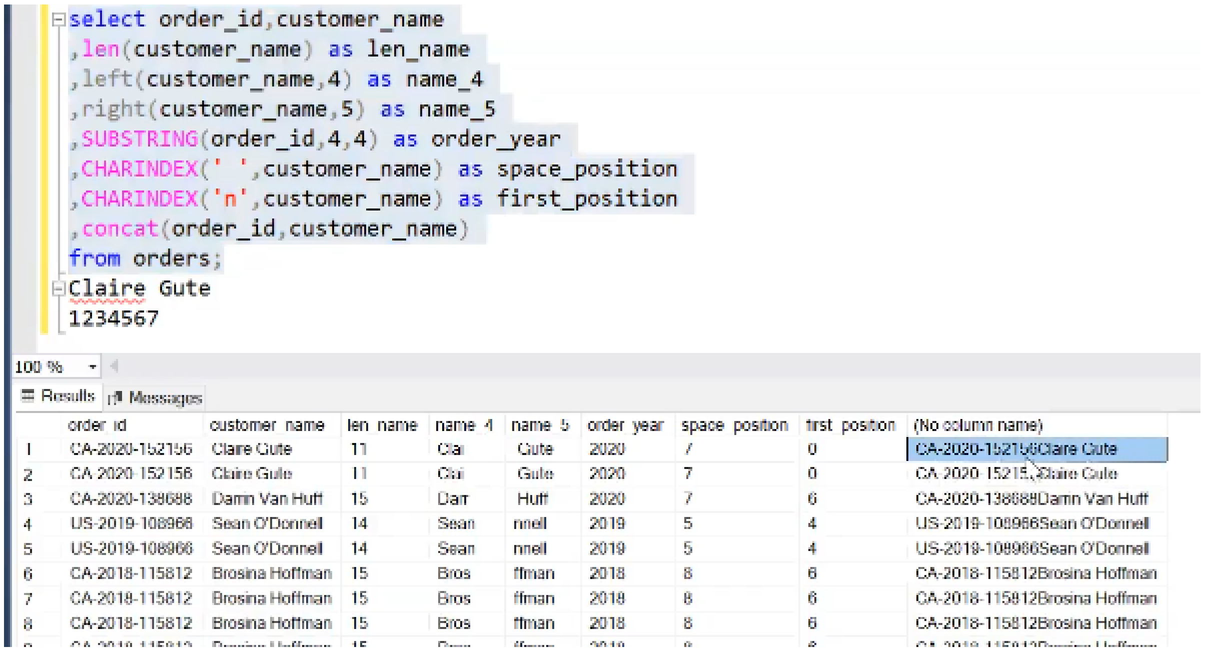


```
--string functions
select order_id,customer_name
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
,SUBSTRING(order_id,4,4) as order_year
,CHARINDEX(' ',customer_name) as space_position
,CHARINDEX('n',customer_name,1) as first_position
,CHARINDEX('n',customer_name,CHARINDEX('n',customer_name,1)+1) as second_position
,CHARINDEX('n',customer_name,11) as n_position
from orders;
Claire Gute
1234567
```

	order id	customer name	len name	name 4	name 5	order year	space position	first position	second position	n position
1	CA-2020-152156	Claire Gute	11	Clai	Gute	2020	7	0	0	0
2	CA-2020-152156	Claire Gute	11	Clai	Gute	2020	7	0	0	0
3	CA-2020-138688	Darrin Van Huff	15	Darr	Huff	2020	7	6	10	0
4	US-2019-108966	Sean O'Donnell	14	Sean	nnell	2019	5	4	10	11
5	US-2019-108966	Sean O'Donnell	14	Sean	nnell	2019	5	4	10	11
6	CA-2018-115812	Brosina Hoffman	15	Bros	ffman	2018	8	6	15	15
7	CA-2018-115812	Brosina Hoffman	15	Bros	ffman	2018	8	6	15	15

13.
14. Here in the above pic..we have found occurrences of n..see pic and understand
15. Also refer documentation\

16. Next we have concat..which concatenates two strings of different columns



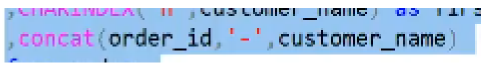
```
select order_id,customer_name
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
,SUBSTRING(order_id,4,4) as order_year
,CHARINDEX(' ',customer_name) as space_position
,CHARINDEX('n',customer_name) as first_position
,concat(order_id,customer_name)
from orders;
Claire Gute
1234567
```

100 %

Results | Messages

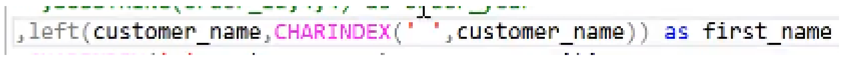
	order id	customer name	len name	name 4	name 5	order year	space position	first position	(No column name)
1	CA-2020-152156	Claire Gute	11	Clai	Gute	2020	7	0	CA-2020-152156Claire Gute
2	CA-2020-152156	Claire Gute	11	Clai	Gute	2020	7	0	CA-2020-152156Claire Gute
3	CA-2020-138688	Darrin Van Huff	15	Darr	Huff	2020	7	6	CA-2020-138688Darrin Van Huff
4	US-2019-108966	Sean O'Donnell	14	Sean	nnell	2019	5	4	US-2019-108966Sean O'Donnell
5	US-2019-108966	Sean O'Donnell	14	Sean	nnell	2019	5	4	US-2019-108966Sean O'Donnell
6	CA-2018-115812	Brosina Hoffman	15	Bros	ffman	2018	8	6	CA-2018-115812Brosina Hoffman
7	CA-2018-115812	Brosina Hoffman	15	Bros	ffman	2018	8	6	CA-2018-115812Brosina Hoffman
8	CA-2018-115812	Brosina Hoffman	15	Bros	ffman	2018	8	6	CA-2018-115812Brosina Hoffman

17. We can also add anything in the concat statement..to concatenate ..here we have concatenated our strings with a hyphen



```
,concat(order_id,'-',customer_name)
```

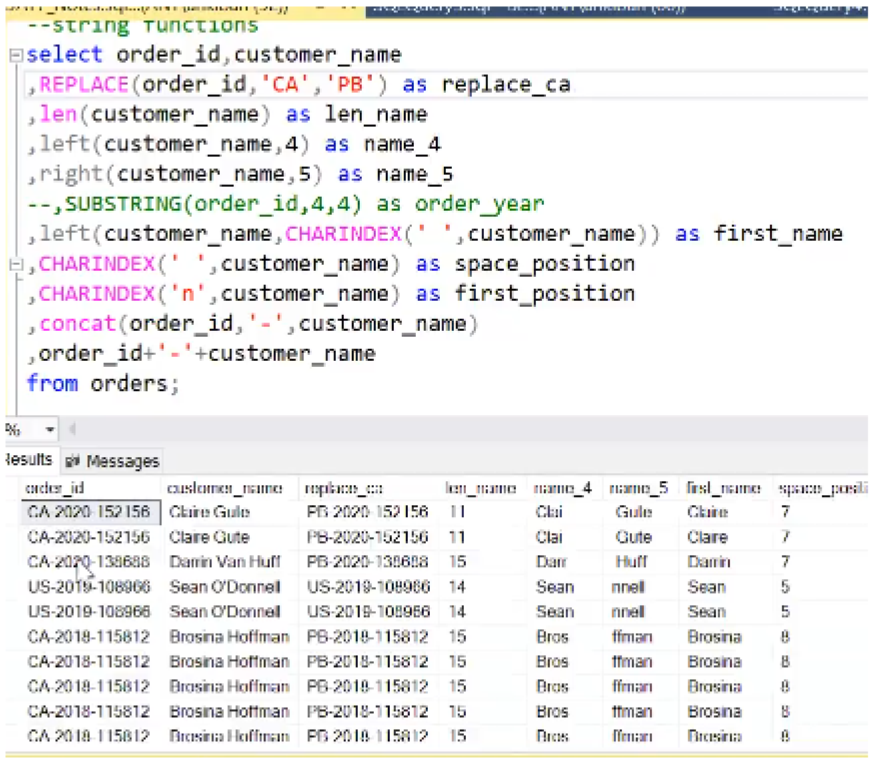
18. To find the first name of customer ..we can use **left** and **CharIndex** functions of a string.



```
,left(customer_name,CHARINDEX(' ',customer_name)) as first_name
```

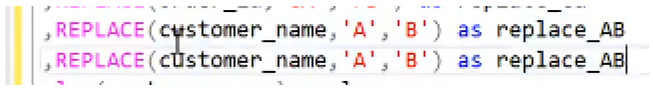
19. Also it can be done by using..**SubString** and **CharIndex** (try yourself)

20. Next we have **replace** function



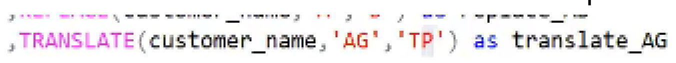
```
--string functions
select order_id,customer_name
,REPLACE(order_id,'CA','PB') as replace_ca
,len(customer_name) as len_name
,left(customer_name,4) as name_4
,right(customer_name,5) as name_5
--,SUBSTRING(order_id,4,4) as order_year
,left(customer_name,CHARINDEX(' ',customer_name)) as first_name
,CHARINDEX(' ',customer_name) as space_position
,CHARINDEX('n',customer_name) as first_position
,concat(order_id,'-',customer_name)
,order_id+'-'+customer_name
from orders;
```

order_id	customer_name	replace_ca	len_name	name_4	name_5	first_name	space_positi
CA-2020-152156	Claire Gute	PB-2020-152156	11	Clai	Gute	Claire	7
CA-2020-152156	Claire Gute	PB-2020-152156	11	Clai	Gute	Claire	7
CA-2020-138688	Darrin Van Huff	PB-2020-138688	15	Darr	Huff	Darrin	7
US-2019-108966	Sean O'Donnell	US-2019-108966	14	Sean	nnell	Sean	5
US-2019-108966	Sean O'Donnell	US-2019-108966	14	Sean	nnell	Sean	5
CA-2018-115812	Brosina Hoffman	PB-2018-115812	15	Bros	ffman	Brosina	8
CA-2018-115812	Brosina Hoffman	PB-2018-115812	15	Bros	ffman	Brosina	8
CA-2018-115812	Brosina Hoffman	PB-2018-115812	15	Bros	ffman	Brosina	8
CA-2018-115812	Brosina Hoffman	PB-2018-115812	15	Bros	ffman	Brosina	8
CA-2018-115812	Brosina Hoffman	PB-2018-115812	15	Bros	ffman	Brosina	8

21. Here we have replaced **CA** with **PB** using replace function on column order_id
22. 

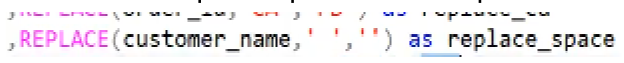
```
,REPLACE(customer_name,'A','B') as replace_AB
,REPLACE(customer_name,'A','B') as replace_AB
```

try this command with small letters as well and see
23. Next we have translate ..which is similar lo replace..

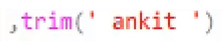


```
,TRANSLATE(customer_name,'AG','TP') as translate_AG
```

but here..A will be replaced with T and G will be replaced with P…but in replace..it replaces entire AG with TP…in Translate it replaces each letter..with its corresponding letter..see code and understand
24. We can also replace space with no space as well



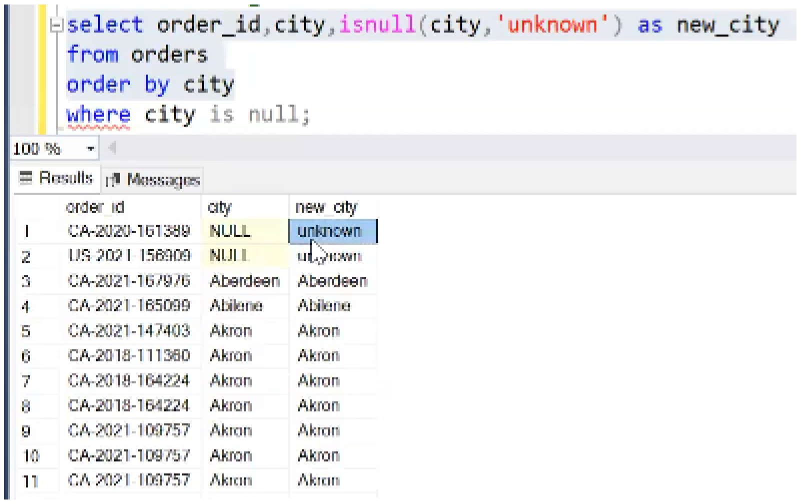
```
,REPLACE(customer_name,' ','') as replace_space
```

25. Trim helps us to remove and trailing spaces 

```
,trim(' ankit ')
```

NULL Handling

1. We have isNULL function ..which changes our null values..into the names which we provide..see pic for example



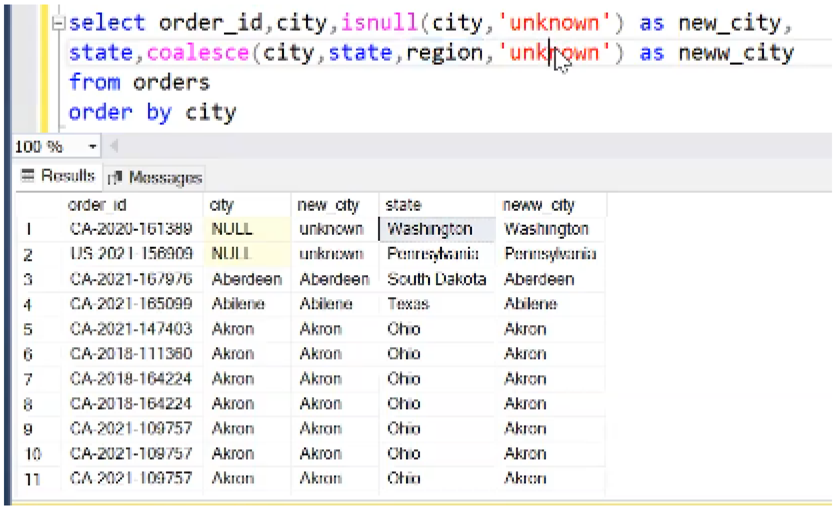
```
select order_id,city,isnull(city,'unknown') as new_city
from orders
order by city
where city is null;
```

100 %

Results | Messages

	order_id	city	new_city
1	CA-2020-161389	NULL	unknown
2	US-2021-156909	NULL	unknown
3	CA-2021-167976	Aberdeen	Aberdeen
4	CA-2021-165099	Abilene	Abilene
5	CA-2021-147403	Akron	Akron
6	CA-2018-111360	Akron	Akron
7	CA-2018-164224	Akron	Akron
8	CA-2018-164224	Akron	Akron
9	CA-2021-109757	Akron	Akron
10	CA-2021-109757	Akron	Akron
11	CA-2021-109757	Akron	Akron

2. Next we have is **coalesce** it works same like **isnull** ..but main difference is that…**coalesce** will take many columns and if every column's value is null in that row….then it replaces it with the name we provide



```
select order_id,city,isnull(city,'unknown') as new_city,
state,coalesce(city,state,region,'unknown') as neww_city
from orders
order by city
```

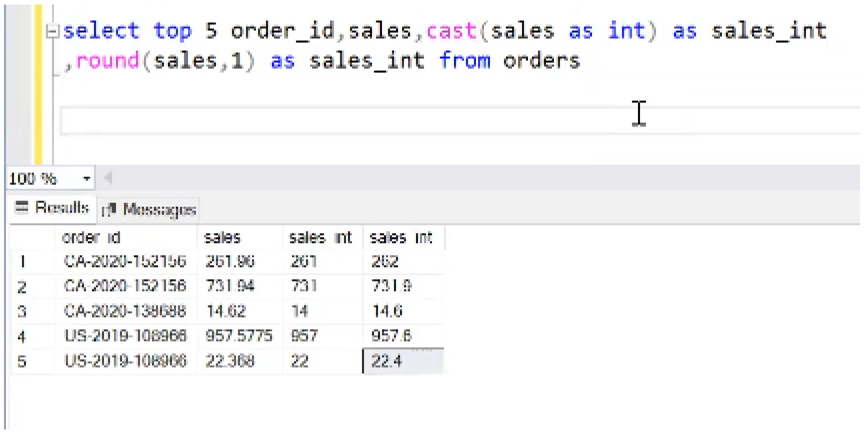
100 %

Results | Messages

	order_id	city	new_city	state	neww_city
1	CA-2020-161389	NULL	unknown	Washington	Washington
2	US-2021-156909	NULL	unknown	Pennsylvania	Pennsylvania
3	CA-2021-167976	Aberdeen	Aberdeen	South Dakota	Aberdeen
4	CA-2021-165099	Abilene	Abilene	Texas	Abilene
5	CA-2021-147403	Akron	Akron	Ohio	Akron
6	CA-2018-111360	Akron	Akron	Ohio	Akron
7	CA-2018-164224	Akron	Akron	Ohio	Akron
8	CA-2018-164224	Akron	Akron	Ohio	Akron
9	CA-2021-109757	Akron	Akron	Ohio	Akron
10	CA-2021-109757	Akron	Akron	Ohio	Akron
11	CA-2021-109757	Akron	Akron	Ohio	Akron

3.
4. Here ..our city is null..then it replaced the city value with the state value(as state was not null) ..if state was null…then it uses region value..even if region is null…it replaces with the name we provide
5. Then we have **CAST** ….instead of using alter table and changing the data type of column…we can just use CAST …it changes the data type of specified column and

create a new column for it



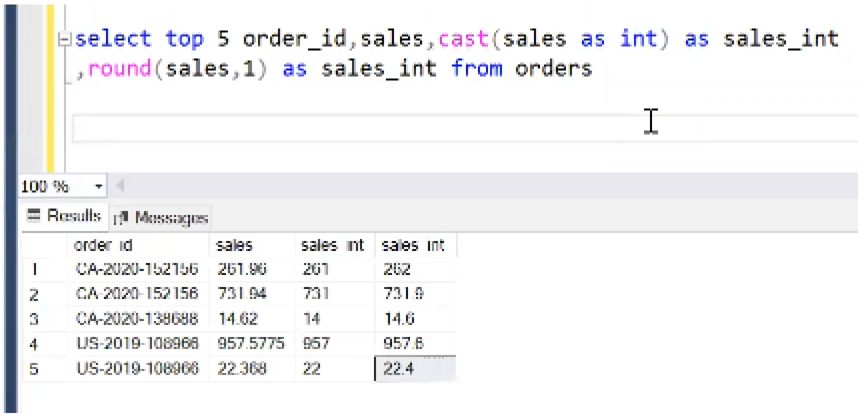
```
select top 5 order_id,sales,cast(sales as int) as sales_int
,round(sales,1) as sales_int from orders
```

100 %

Results Messages

	order id	sales	sales int	sales int
1	CA-2020-152156	261.96	261	262
2	CA-2020-152156	731.94	731	731.9
3	CA-2020-138688	14.62	14	14.6
4	US-2019-108966	957.5775	957	957.6
5	US-2019-108966	22.368	22	22.4

6. Next we have **ROUND** which rounds-off our value to specified points



```
select top 5 order_id,sales,cast(sales as int) as sales_int
,round(sales,1) as sales_int from orders
```

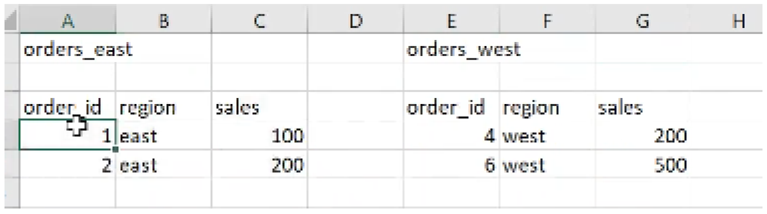
100 %

Results Messages

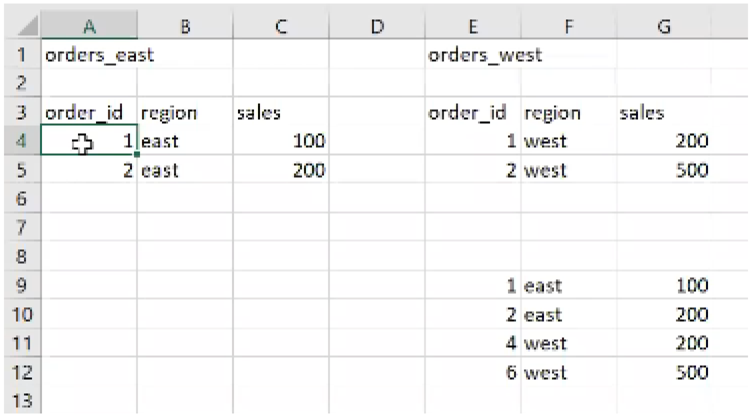
	order id	sales	sales int	sales int
1	CA-2020-152156	261.96	261	262
2	CA-2020-152156	731.94	731	731.9
3	CA-2020-138688	14.62	14	14.6
4	US-2019-108966	957.5775	957	957.6
5	US-2019-108966	22.368	22	22.4

7. Here round function …round-off our sales col values to single digit point..see above pic and understand

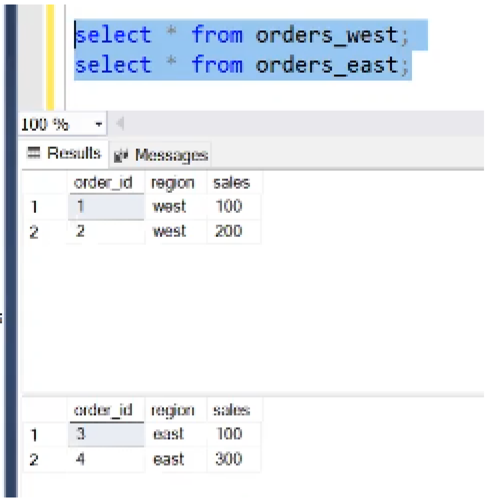
SET queries

1. 

A	B	C	D	E	F	G	H
orders_east				orders_west			
order_id	region	sales		order_id	region	sales	
1	east	100		4	west	200	
2	east	200		6	west	500	

2. 

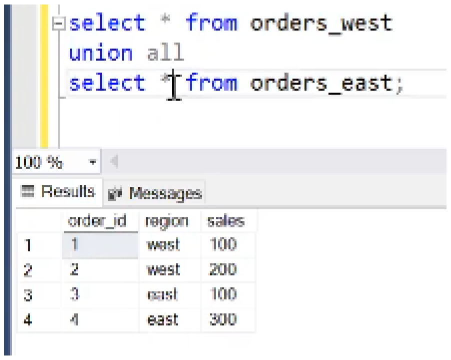
	A	B	C	D	E	F	G
1	orders_east				orders_west		
2							
3	order_id	region	sales		order_id	region	sales
4	1	east	100		1	west	200
5	2	east	200		2	west	500
6							
7							
8							
9					1	east	100
10					2	east	200
11					4	west	200
12					6	west	500
13							

3. Next we are creating two sample tables(east and west0..which looks like above
4. And inserting some sample values into the table
5. 

```
select * from orders_west;
select * from orders_east;
```

```
   order_id  region  sales
1  1         west    100
2  2         west    200

   order_id  region  sales
1  3         east    100
2  4         east    300
```

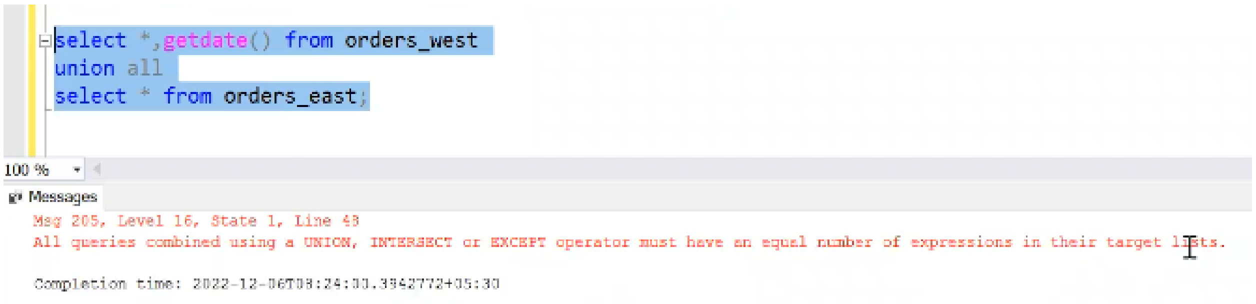
Here we can see two table with data inserted



```
select * from orders_west
union all
select * from orders_east;
```

```
   order_id  region  sales
1  1         west    100
2  2         west    200
3  3         east    100
4  4         east    300
```

6. And if we perform **union-all** on them we get

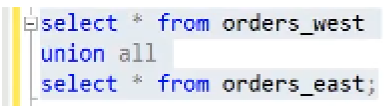


```
select *,getdate() from orders_west
union all
select * from orders_east;
```

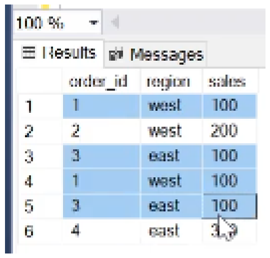
```
Msg 205, Level 16, State 1, Line 43
All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists.

Completion time: 2022-12-06T08:24:33.3942772+05:30
```

7.

8. Here theses two tables must have same columns and same data types
9. Here if table 1 has 100 rows and table has 200 rows...now if we perform union-all..then we will get 300 rows..
10. Union is also same thing..but we if have 2 same values..then it deletes one(removes duplicates)
11. Here we have performed union-all

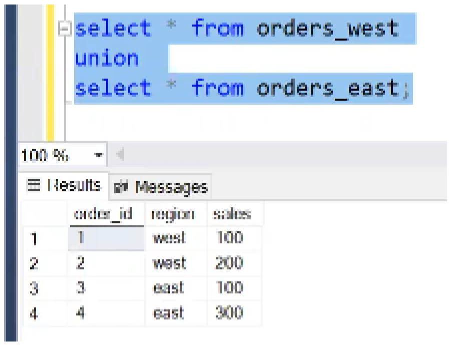


```
select * from orders_west
union all
select * from orders_east;
```



	order_id	region	sales
1	1	west	100
2	2	west	200
3	3	east	100
4	1	west	100
5	3	east	100
6	4	east	300

12. Here we have performed union

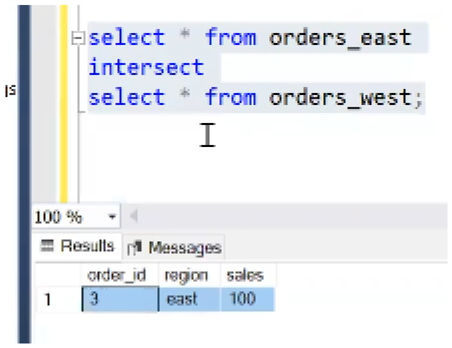


```
select * from orders_west
union
select * from orders_east;
```

	order_id	region	sales
1	1	west	100
2	2	west	200
3	3	east	100
4	4	east	300

..try union-all using distinct(task for myself)

13. And if we want to see common rows between two tables..we can use intersect

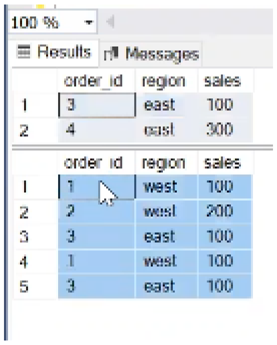


```
select * from orders_east
intersect
select * from orders_west;
```

	order_id	region	sales
1	3	east	100

14. Basically while doing intersection..two tables must have same number of columns and data types
15. Next we have is except..

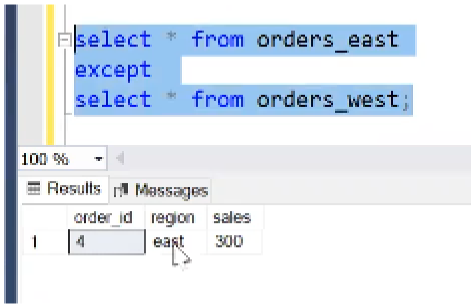
> The EXCEPT clause in SQL helps users combine two SELECT statements and returns distinct rows from the first SELECT statement that are not available in the second SELECT statement.



	order_id	region	sales
1	3	east	100
2	4	east	300

	order_id	region	sales
1	1	west	100
2	2	west	200
3	3	east	100
4	1	west	100
5	3	east	100

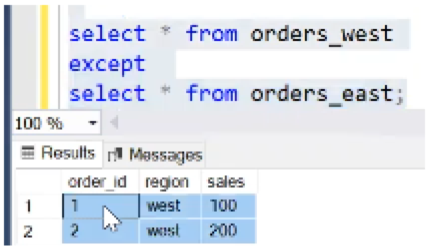
16. If we run our tables we get



```
select * from orders_east
except
select * from orders_west;
```

	order_id	region	sales
1	4	east	300

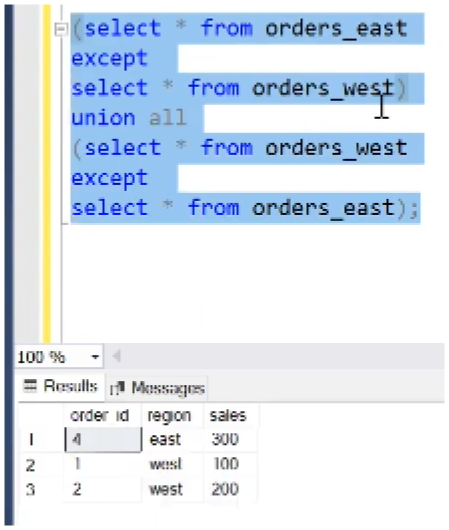
17. If we run except on east and west …we get



```
select * from orders_west
except
select * from orders_east;
```

	order_id	region	sales
1	1	west	100
2	2	west	200

18. If we run except on west and east …we get
19. Here only union-all will give all the values…remaining every functions…removes duplicates



```
(select * from orders_east
except
select * from orders_west)
union all
(select * from orders_west
except
select * from orders_east);
```

	order_id	region	sales
1	4	east	300
2	1	west	100
3	2	west	200

20. if we run this query we get this