Day 3

primary key vs unique key

primary key cannot be null, unique key can
only 1 primary key, multiple unique key
primary key gives clustered index, unique key gives unclustered.

constraints

Not null
between A and B
default()
IDENTITY(seed, increment)

Window functions

rank over

```
--Rank: When ranking, numbers jump if there is a tiebreaker
--Dense_Rank: Numbers dont jump on ties
--Row_Number: just counters by the row number
```

```
Select p.ProductId, p.ProductName, p.UnitPrice,
Rank() Over(Order By UnitPrice asc) "Rank",
Dense_Rank() Over(Order By UnitPrice asc) [Dense],
Row_Number() Over(Order By UnitPrice asc) rownum
From Products p
```

partition

```
--Partition By: "In Relation to each"
Select c.ContactName, c.Country, Count(o.OrderId) [NumofOrd],
Rank() Over( Partition By c.Country Order By Count(o.OrderId)) as "Rank"
From Customers c Join Orders o on c.CustomerId = o.CustomerId
Group By c.ContactName, c.Country
```

113 %

Results | Messages | Execution plan

	ContactName	Country	NumofOrd	Rank
4	Georg Pipps	Austria	10	1
5	Roland Mendel	Austria	30	2
6	Catherine Dewey	Belgium	7	1
7	Pascale Cartrain	Belgium	12	2
8	Pedro Afonso	Brazil	5	1
9	Anabela Domingues	Brazil	6	2
10	Aria Cruz	Brazil	7	3
11	Bernardo Batista	Brazil	9	4
12	Paula Parente	Brazil	9	4
13	André Fonseca	Brazil	9	4
14	Janete Limeira	Brazil	11	7
15	Lúcia Carvalho	Brazil	13	8

Row number

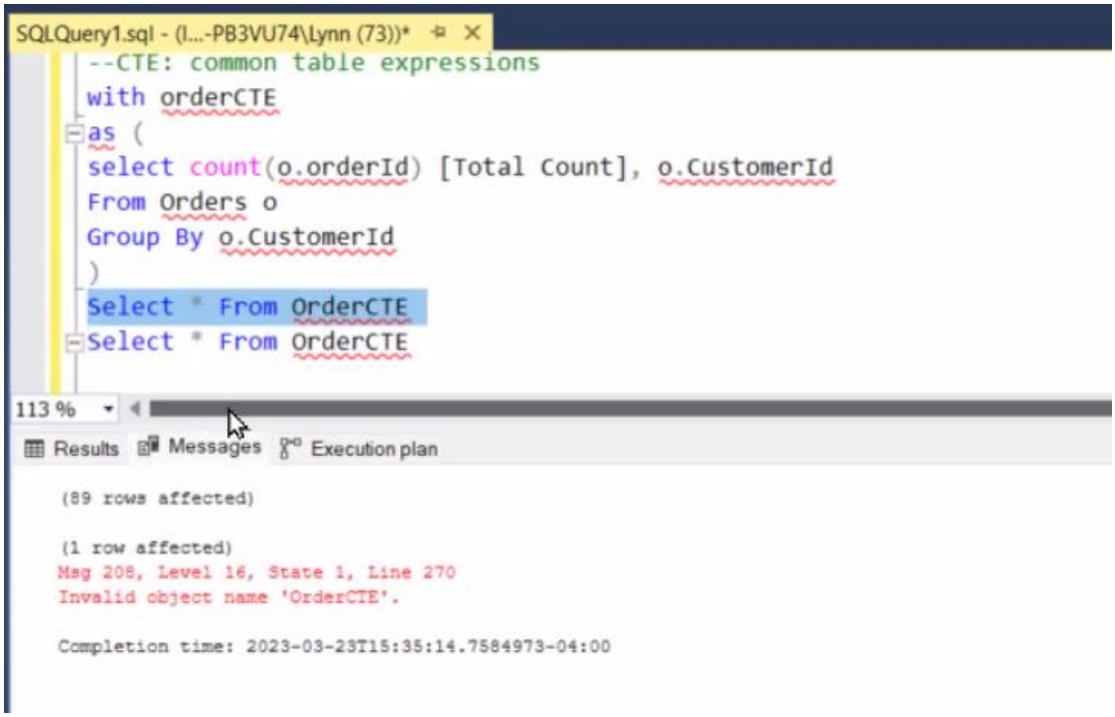
```
Select c.ContactName, c.Country,
Row_Number() Over( Partition By c.Country Order By c.Country) as "Rank"
From Customers c Join (Select distinct CustomerId From Orders) o on o.CustomerId = c.CustomerId
Order By 2, 3
```

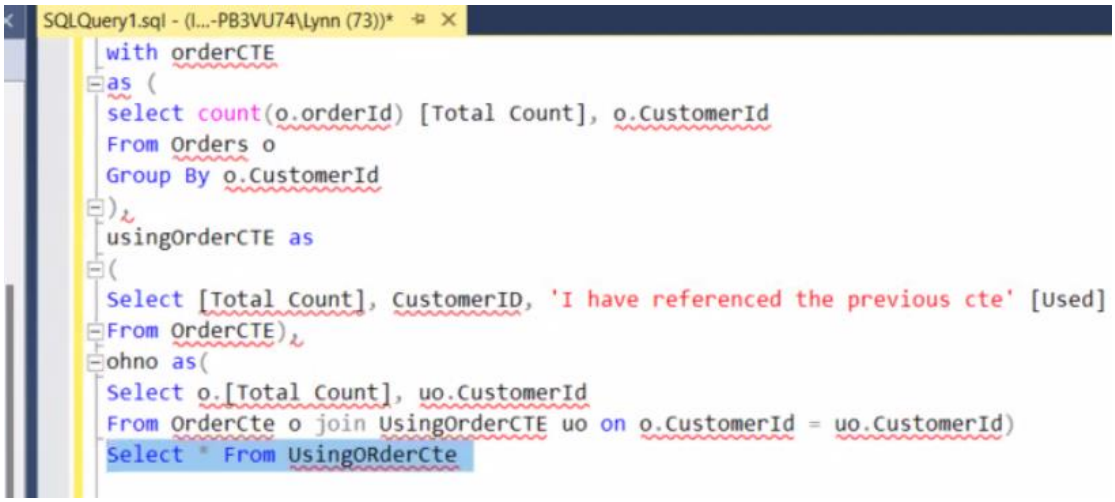
View: virtual table

```
CREATE VIEW SalesEmployees AS
SELECT ID, Salary
FROM Employees
WHERE Dept = 'Sales';
```

CTE: common table expressions

CTE can be used only in the query right after the CTE declared





recursive cte

prevent sql injection