INSTITUTO POLITECNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS

INGENIERIA EN TELEMATICA

BASES DE DATOS DISTRIBUIDAS

EQUIPO 3:

ACOSTA VILLA CRISTIAN ABRAHAM

ARVIZU GÓMEZ ABRAHAM

ENRIQUEZ MELENDEZ JESÚS

GARCÍA ACOSTA ABRAHAM

MONZÓN MOGOLLAN DANIELA

ROSAS PALACIOS ALAN

PRESENTAN:

PROYECTO FINAL

ADVENTURE WORKS 2019

GRUPO:
3TV2

SEMESTRE 2022 – 1

upiita-ipn

Al migrar los registros de la base de datos original (AdventureWorks2019), se mantuvieron parámetros como llaves primarias y foráneas (Solamente relaciones que comparte el esquema), restricciones e índices agrupados. En consecuencia, las consultas podrían ya estar optimizadas.

El siguiente análisis presenta los planes de ejecución de cada una de las consultas y de requerirse la propuesta para su optimización.

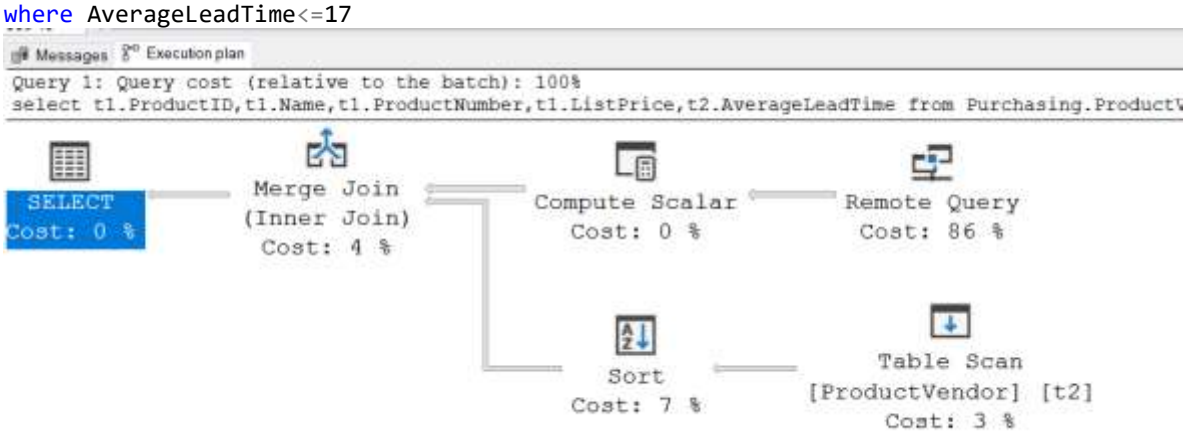
Estas consultas se realizaron en los esquemas compras (pursashing) y producción (production).

Pursashing

Las siguientes consultas fueron ejecutadas en la instancia de SQLServer COMPRAS.

6.- Listar los productos comprados que en promedio tardan entre 17 días o menos en llegar desde que se realizó el pedido de compra

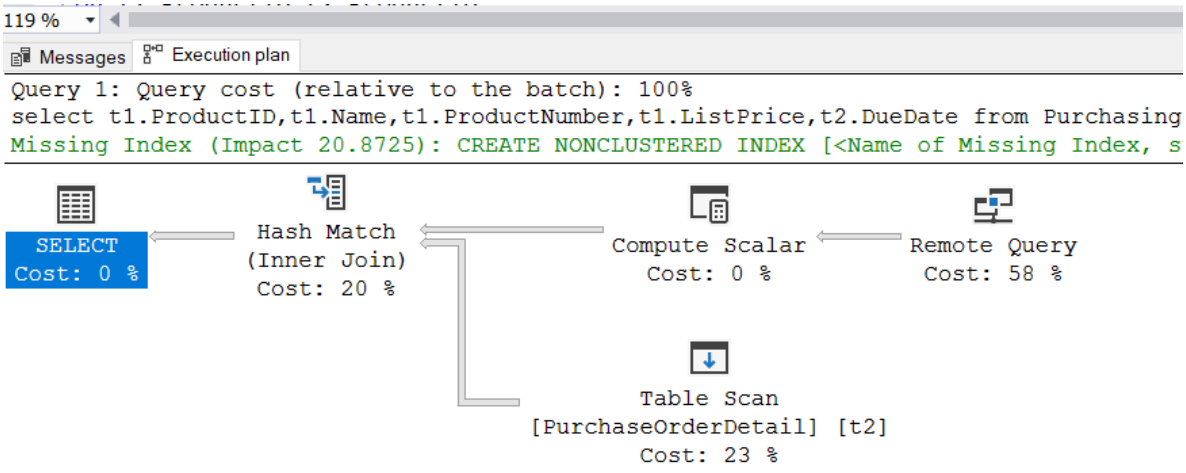
```
select t1.ProductID,t1.Name,t1.ProductNumber,t1.ListPrice,t2.AverageLeadTime
from Purchasing.ProductVendor t2 join LS_PRODUCCION.Produccion.Production.Product t1
on t1.ProductID=t2.ProductID
where AverageLeadTime<=17
```



El plan de ejecución muestra que Remote Query es la operación con mayor costo, al ser esta una operación final, solamente optimizaremos del lado donde se obtienen los datos.

7.- Listar las compras de productos que se esperen lleguen antes del 24 de febrero del 2014;

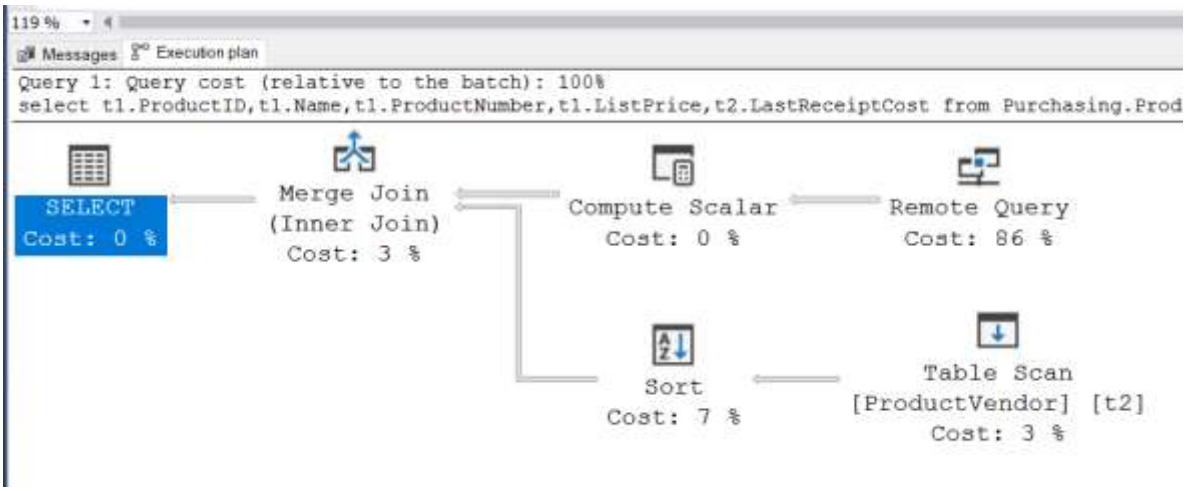
```
select t1.ProductID,t1.Name,t1.ProductNumber,t1.ListPrice,t2.DueDate
from Purchasing.PurchaseOrderDetail t2 join
LS_PRODUCCION.Produccion.Production.Product t1
on t1.ProductID=t2.ProductID
where DueDate<'24-02-2014'
```



El plan de ejecución muestra claramente la necesidad de optimizar las operaciones Table Scan que es muy ineficiente y tiene un gran costo.

9.- Listar los productos que compra la empresa de Adventure Works y le cuestan más de 40 dólares

```
select t1.ProductID,t1.Name,t1.ProductNumber,t1.ListPrice,t2.LastReceiptCost
from Purchasing.ProductVendor t2 join LS_PRODUCCION.Produccion.Production.Product t1
on t1.ProductID=t2.ProductID
where LastReceiptCost>40
```



En el plan de ejecución podemos notar que la operación Remote Query es la de mayor costo y al ser una operación final no se optimizara y optaremos por optimizar del lado donde se obtienen los datos.

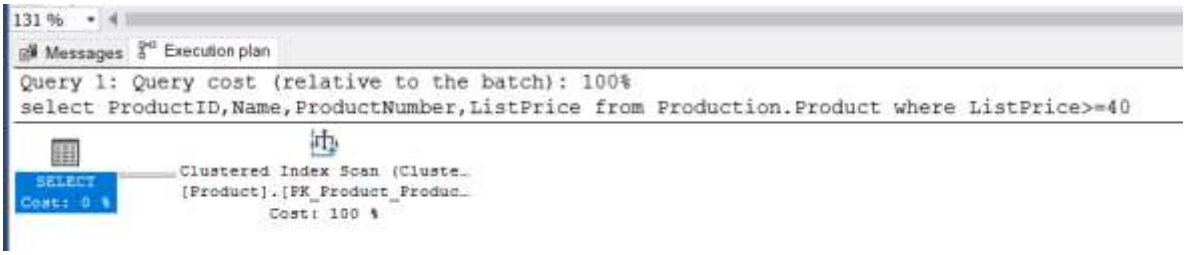
Production

Las siguientes consultas fueron ejecutadas en la instancia de SQLServer PRODUCCION.

1.- Listar todos los productos que tengan un precio de venta mayor o igual a 40 dólares.

```
select ProductID,Name,ProductNumber,ListPrice from Production.Product
```

```
where ListPrice>=40
```

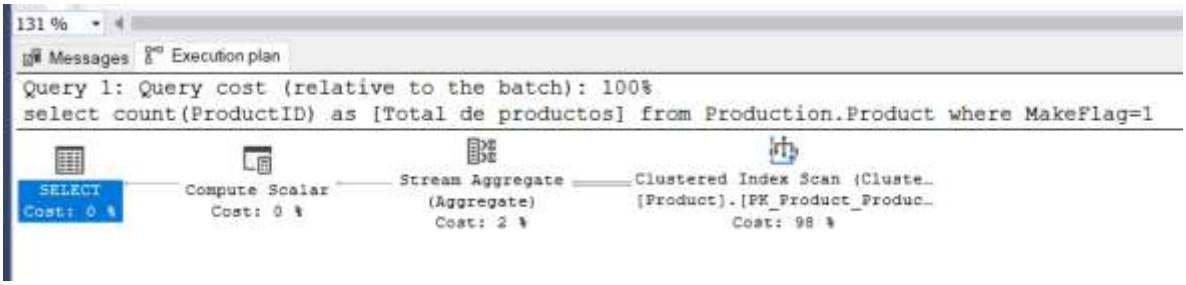


El plan de ejecución muestra que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

Mas adelante analizaremos y existe otra manera de optimizar esta consulta y las que usen Clusterd index scan.

2.- Listar la cantidad de productos que se venden para ensamblar en casa

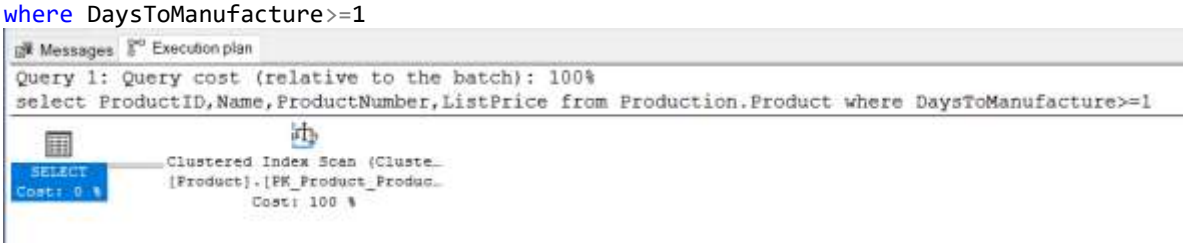
```
select count(ProductID) as [Total de productos] from Production.Product
where MakeFlag=1;
```



El plan de ejecución muestra que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

3.- Listar los productos que les toma un día o más en ensamblar el producto

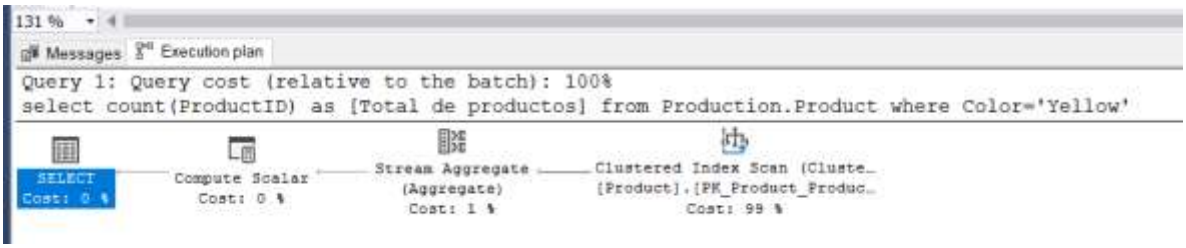
```
select ProductID,Name,ProductNumber,ListPrice from Production.Product
where DaysToManufacture>=1
```



El plan de ejecución muestra que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

4.- Listar la cantidad de productos Amarillos

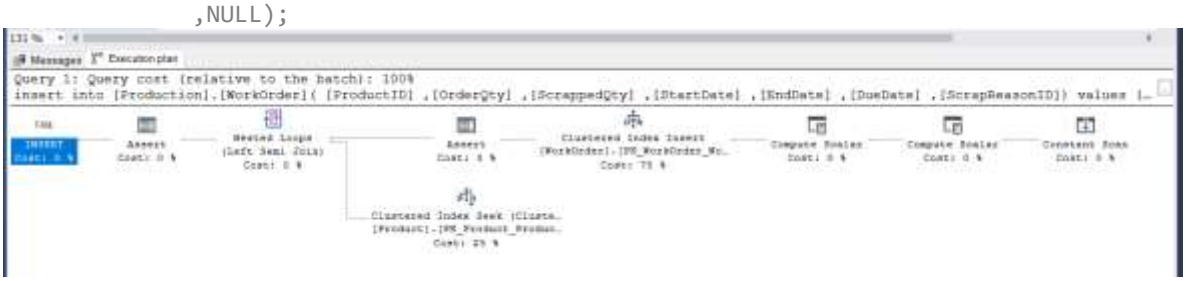
```
select count(ProductID) as [Total de productos] from Production.Product
where Color='Yellow'
```



El plan de ejecución muestra que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

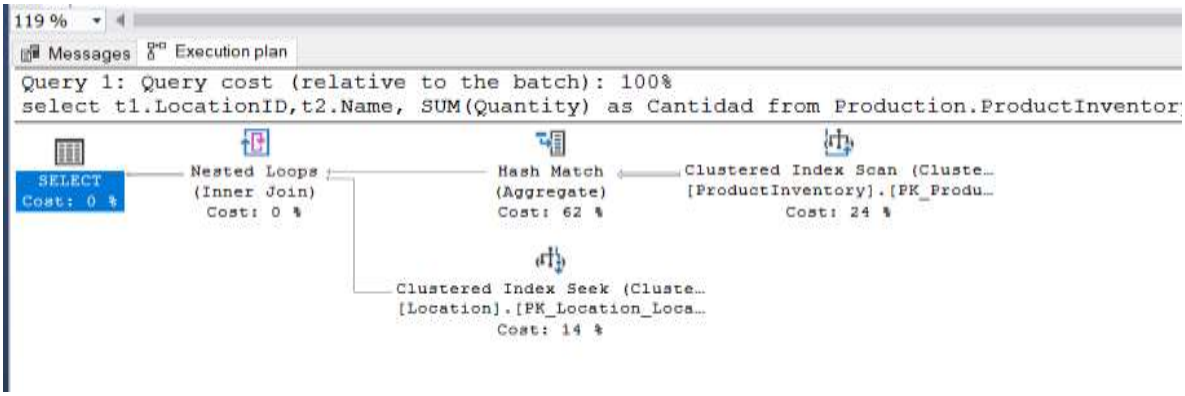
5.- Registrar una nueva orden de fabricación de 4 piezas de un mismo producto.

```
insert into [Production].[WorkOrder](
            [ProductID]
            ,[OrderQty]
            ,[ScrappedQty]
            ,[StartDate]
            ,[EndDate]
            ,[DueDate]
                    ,[ScrapReasonID])
values ( 730
              ,4
              ,0
              ,CONVERT(DATE, GETDATE(),120)
              ,NULL
              ,DATEADD(day,11,CONVERT(DATE, GETDATE(),120))
              ,NULL);
```



8.- Listar la cantidad total de productos por el área en el que se encuentran dentro de la planta de producción.

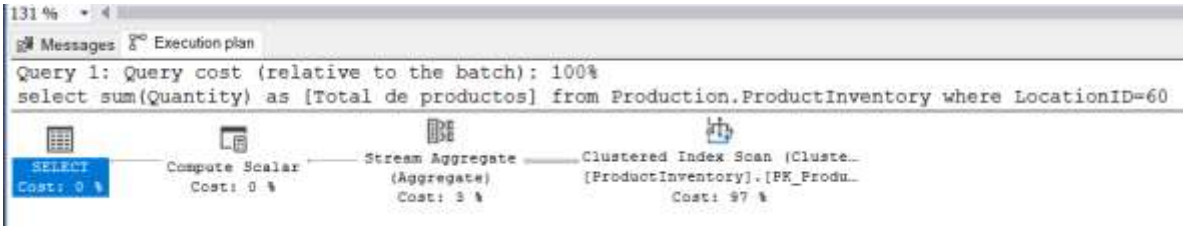
```
select t1.LocationID,t2.Name, SUM(Quantity) as Cantidad
from Production.ProductInventory t1 join
Production.Location t2
on t1.LocationID=t2.LocationID
group by t1.LocationID,t2.Name
```



El plan de ejecución muestra el uso de las operaciones NonClustered Index Seek la cual es bastante eficiente.

10.- Obtener la cantidad de productos que ya se encuentran en el área de ensamblaje final

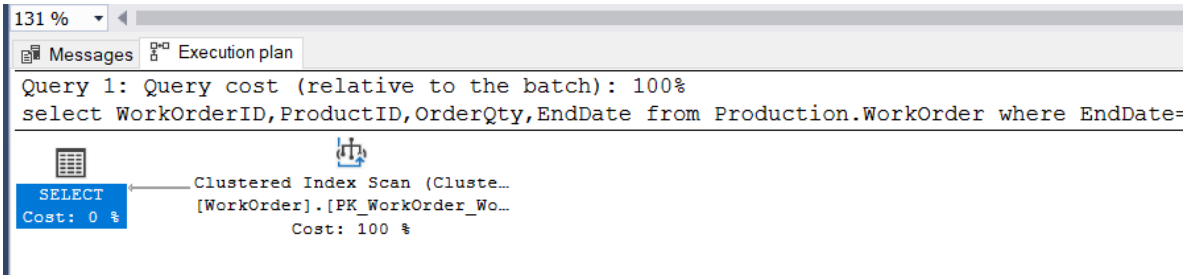
```
select sum(Quantity) as [Total de productos] from Production.ProductInventory
where LocationID=60
```



El plan de ejecución muestra que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

11.- Listar las ordenes de producción que terminaron de fabricarse el 1 de enero del año 2013

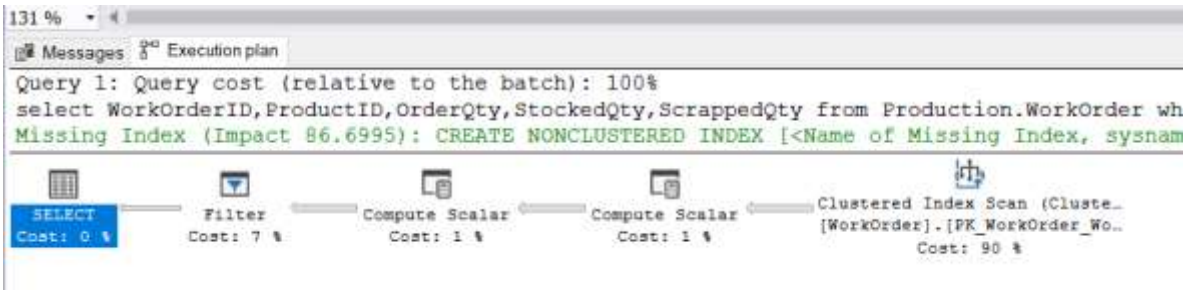
```
select WorkOrderID,ProductID,OrderQty,EndDate from Production.WorkOrder
where EndDate='2013-01-01'
```



El plan de ejecución muestra que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

12.- Listar las ordenes de manufactura que no cumplieron con su meta de stock

```
select WorkOrderID,ProductID,OrderQty,StockedQty,ScrappedQty from
Production.WorkOrder
where ScrappedQty!=0
```



El plan de ejecución muestra el uso de las operaciones Filter necesarias para filtrar los datos requeridos en la consulta. Además, que se utiliza un índice agrupado de la tabla product, el cual nos indica que quizás estamos retornando más registros de los necesarios.

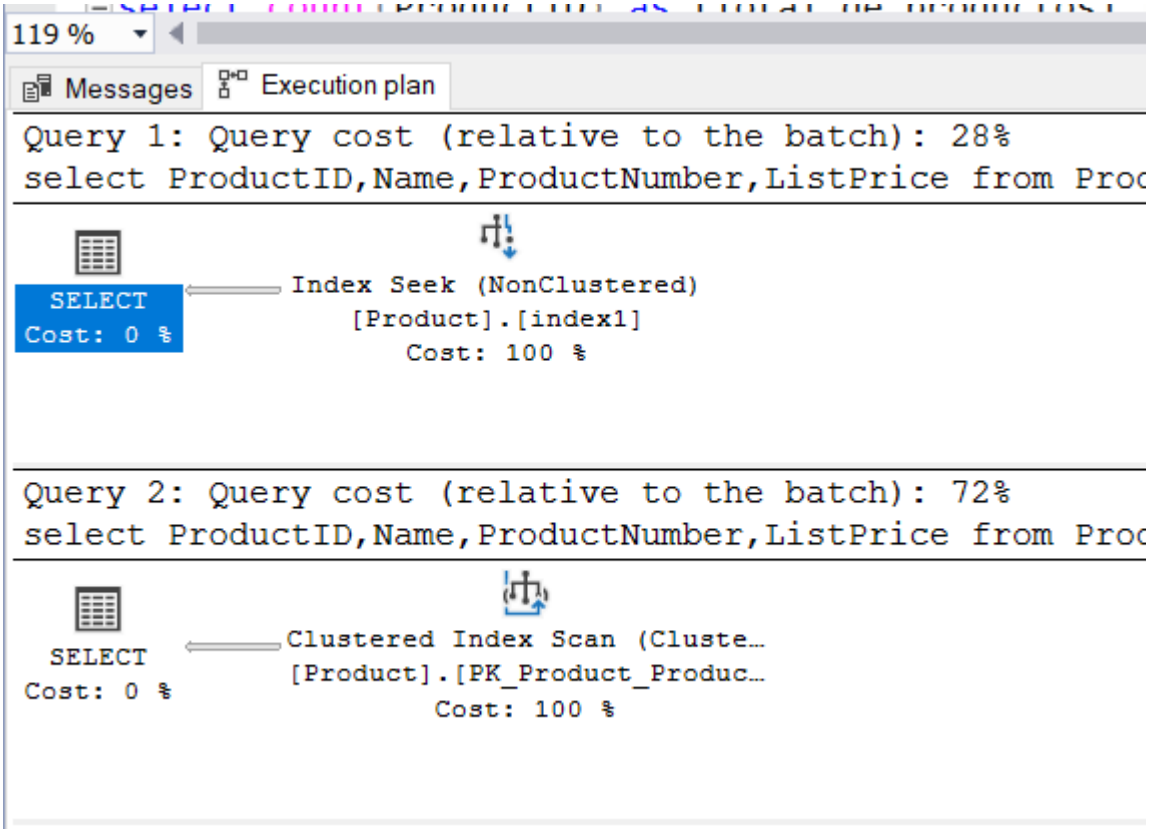
Análisis de optimización de consultas

Consulta numero 1: Para esta consulta se creó el siguiente índice

```
create nonclustered index index1
```

```
on Production.Product(ListPrice)
INCLUDE(ProductID,Name, ProductNumber)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

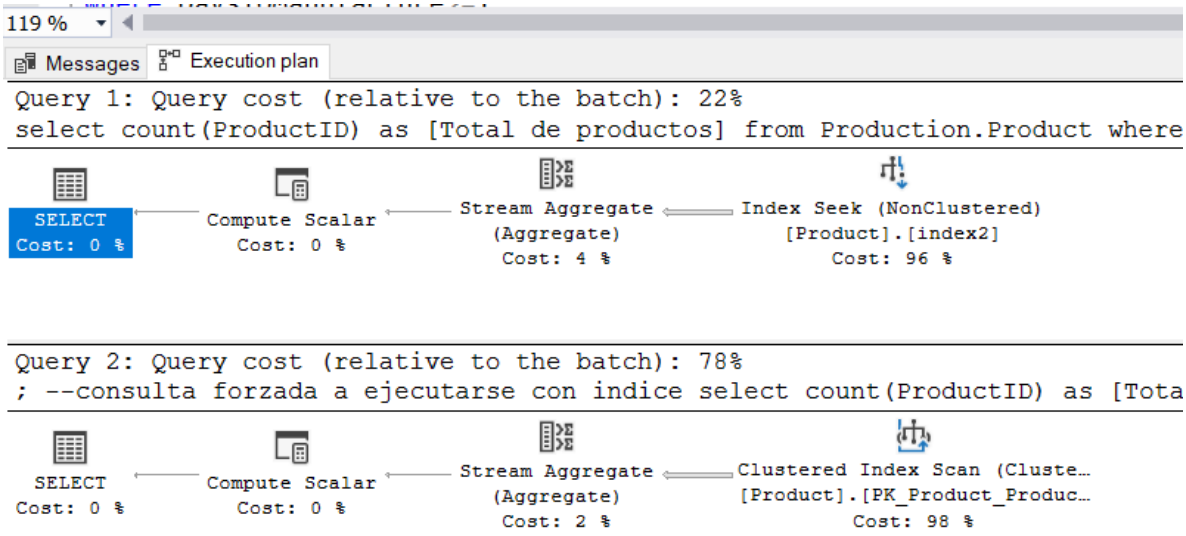


Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.

Consulta numero 2: Para esta consulta se creó el siguiente índice

```
create nonclustered index index2
on Production.Product(MakeFlag)
INCLUDE(ProductID)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

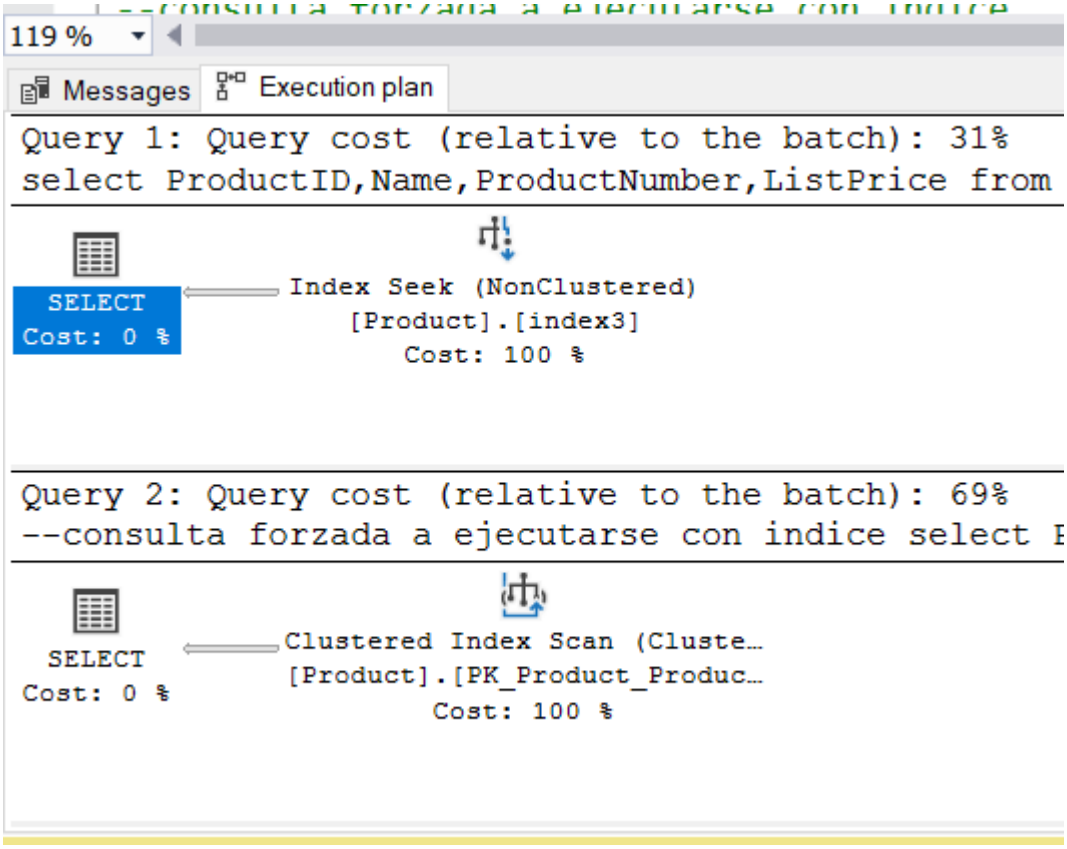


Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.

Consulta numero 3: Para esta consulta se creó el siguiente índice

```
create nonclustered index index3
on Production.Product(DaysToManufacture)
INCLUDE(ProductID,Name,ProductNumber,ListPrice)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

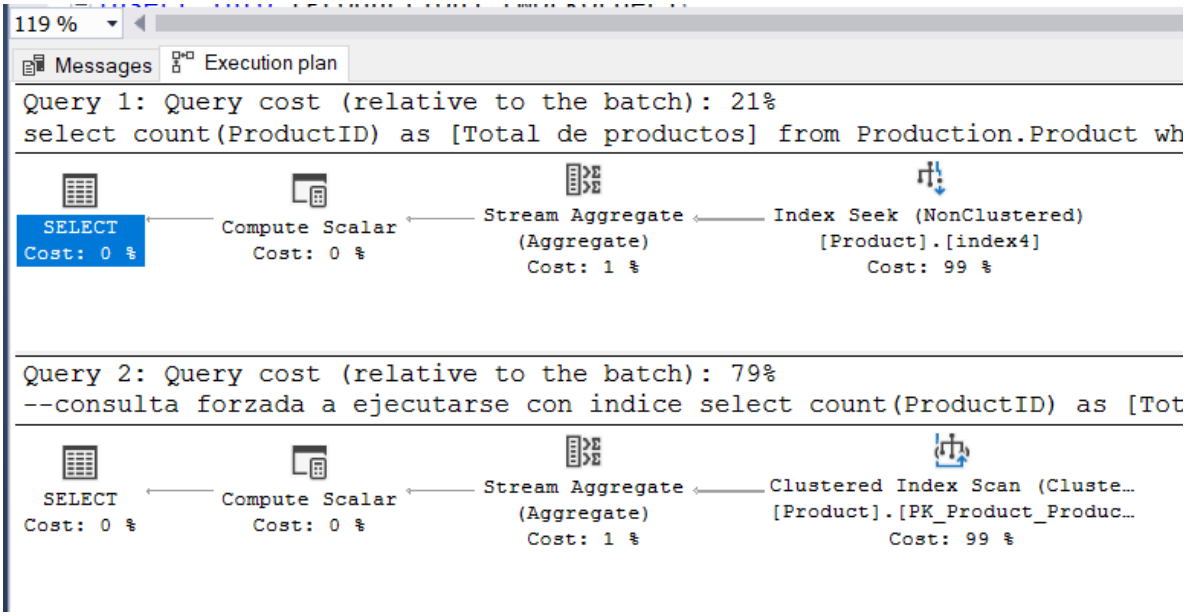


Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.

Consulta numero 4: Para esta consulta se creó el siguiente índice

```
create nonclustered index index4
on Production.Product(Color)
INCLUDE(ProductID)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

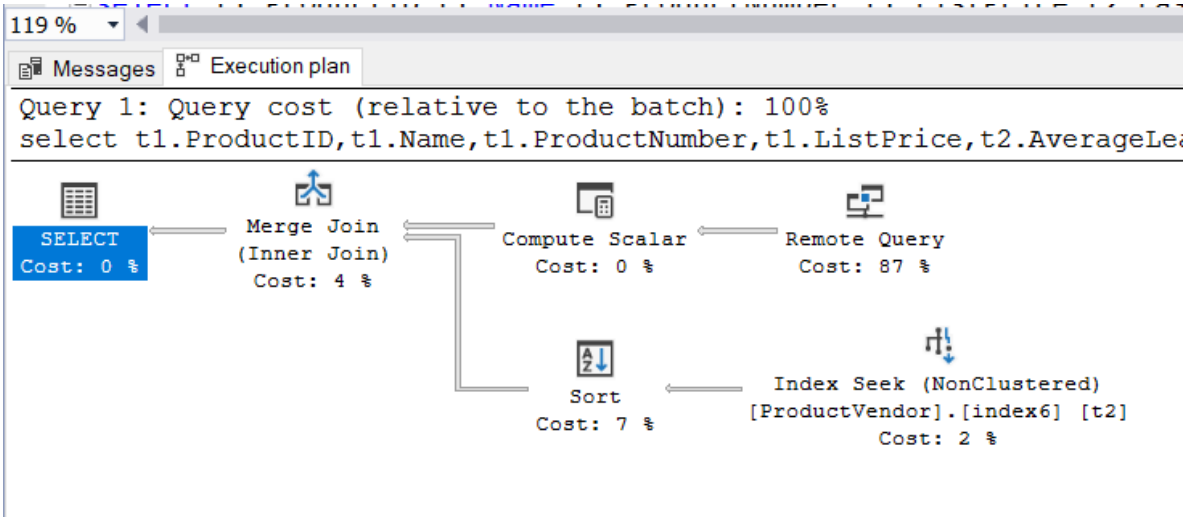


Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.

Consulta numero 5: En el caso de la consulta uno esta usa el índice agrupado que viene por defecto, además de ejecutar operaciones eficientes, por lo cual no se optimizara más.

Consulta numero 6: Para esta consulta se creó el siguiente índice no agrupado

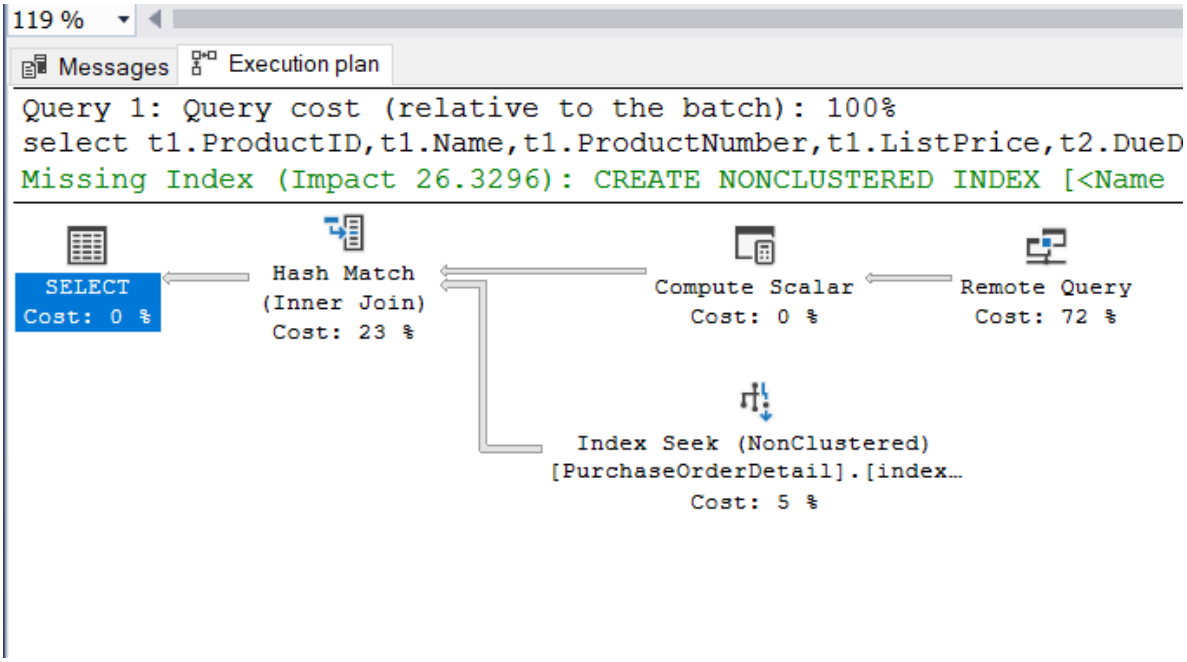
```
CREATE NONCLUSTERED INDEX [index6]
ON [Purchasing].[ProductVendor] ([AverageLeadTime])
INCLUDE ([ProductID])
```



Sin embargo, los resultados obtenidos no difieren mucho con respecto a los anteriores ya presentados dado que las operaciones que consumen el plan de ejecución en su gran mayoría son del tipo Remote Query.

Consulta numero 7: Para esta consulta se creó el siguiente índice no agrupado

```
CREATE NONCLUSTERED INDEX [index7]
ON [Purchasing].[PurchaseOrderDetail] ([DueDate])
INCLUDE ([ProductID])])
```

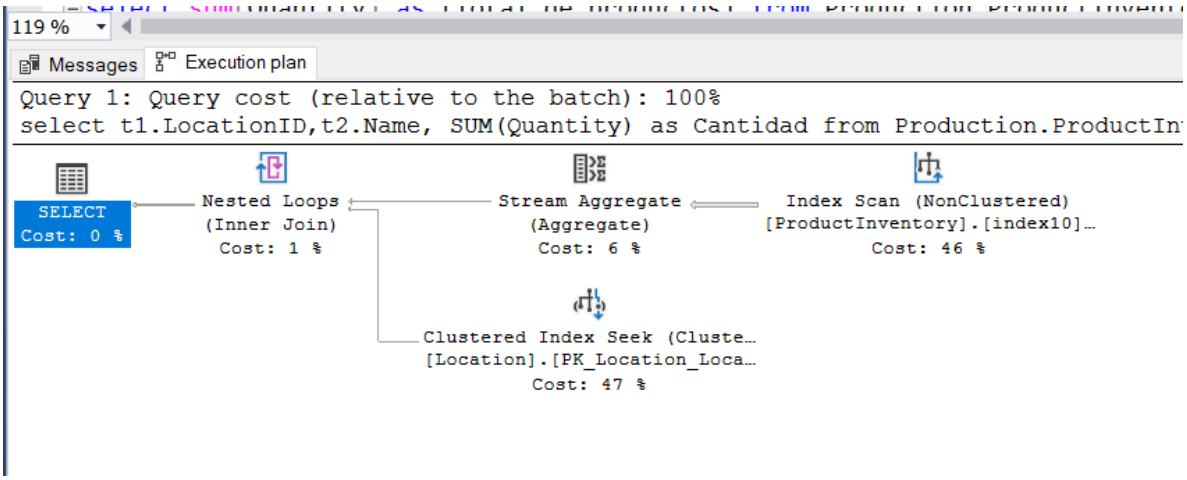


Sin embargo, los resultados obtenidos no difieren mucho con respecto a los anteriores ya presentados dado que las operaciones que consumen el plan de ejecución en su gran mayoría son del tipo Remote Query.

Consulta numero 8: Esta consulta hizo uso del siguiente index para mejorar su rendimiento

```
create nonclustered index index10
on Production.ProductInventory(LocationID)
INCLUDE(Quantity)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

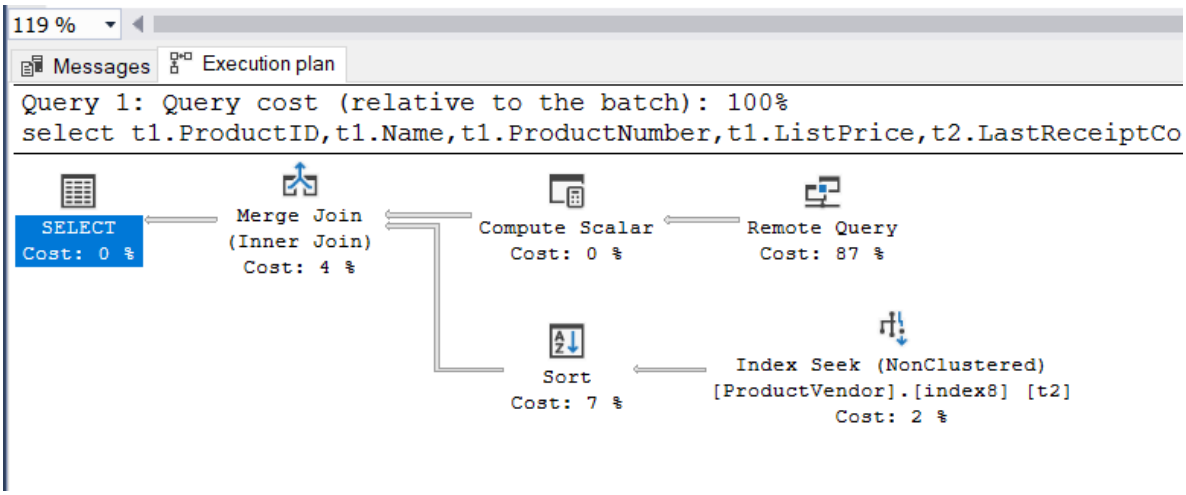


Como podemos ver ahora se realizan mayor numero de operaciones index seek, que son mas eficientes.

Consulta numero 9: Para esta consulta se creó el siguiente índice no agrupado

```
CREATE NONCLUSTERED INDEX [index8]
ON [Purchasing].[ProductVendor] ([LastReceiptCost])
INCLUDE ([ProductID])
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

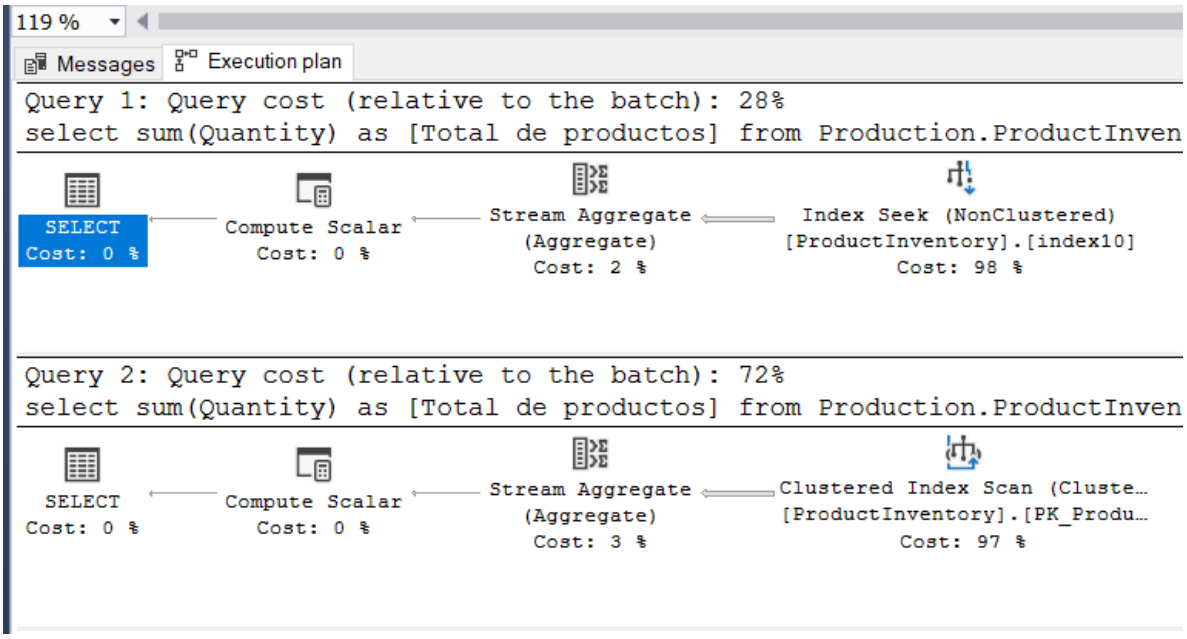


Sin embargo, los resultados obtenidos no difieren mucho con respecto a los anteriores ya presentados dado que las operaciones que consumen el plan de ejecución en su gran mayoría son del tipo Remote Query.

Consulta numero 10: Para esta consulta se creó el siguiente índice

```
create nonclustered index index10
on Production.ProductInventory(LocationID)
INCLUDE(Quantity)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

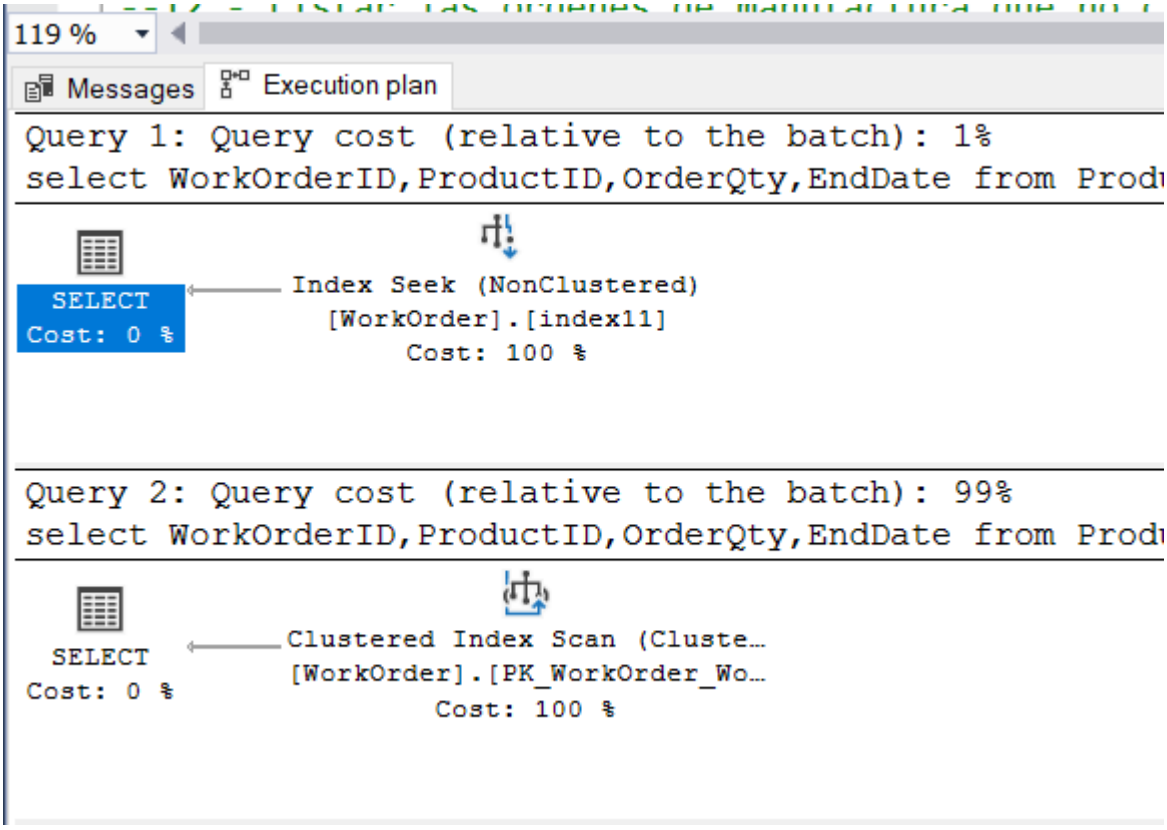


Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.

Consulta numero 11: Para esta consulta se creó el siguiente índice

```
create nonclustered index index11
on Production.WorkOrder(EndDate)
INCLUDE(WorkOrderID,ProductID,OrderQty)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.

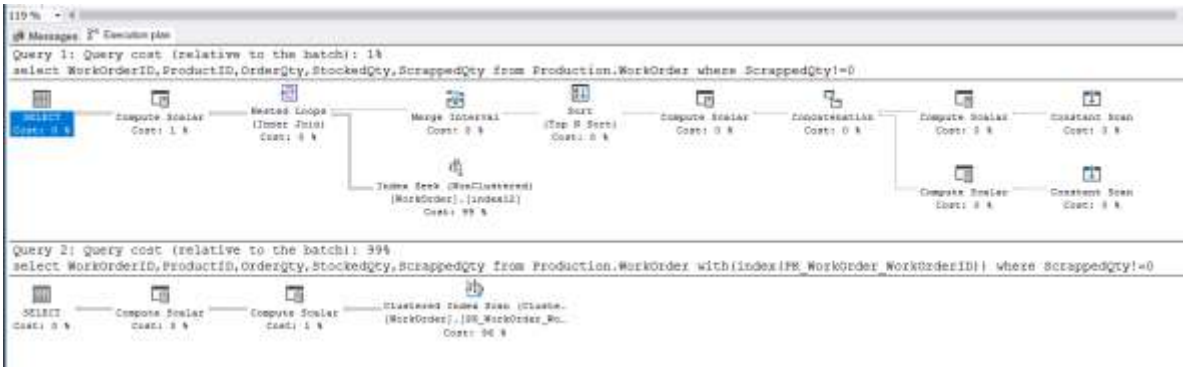


Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.

Consulta numero 12: Para esta consulta se creó el siguiente índice

```
create nonclustered index index12
on Production.WorkOrder(ScrappedQty)
INCLUDE(WorkOrderID,ProductID,OrderQty,StockedQty)
```

Ahora al mostrar el plan de ejecución de la consulta usando el índice y comparándola con la que no usa el índice.



Podemos observar que la consulta se a optimizado considerablemente y ahora hace uso de las operaciones Index Seek que son muy eficientes.