SQL SERVER Performance Tuning Notes

http://msbiskills.com/

How to avoid/eliminate/remove ORDER BY from SQL Queries?

Great so today we are going to discuss what are the options we have to avoid sort operation in the execution plan. Sort is a blocking operator and can be very expensive at times hence as a developer first we need to ask question to ourselves whether do we really order by not. So by enlarge if we don't need it please remove order by from the Top most outer query? Now let’s say there a requirement and we need order by there is not resort, we have to apply a particular order. Now let’s check out what are the options we have-

- **Rewrite Query - Multiple smaller Sorts instead of a Large Sort**
  First and most important thing is consider if we can convert the sort operation on larger set of data to multiple sorts on smaller set of data. This is mathematically provable that performing sort on smaller set of data is much better than one large sort. The algorithmic cost of sort is O (N*Log (N)) – N here is the number of input rows. So mathematically we can prove that doing multiple mini sorts is much better than performing a sort on a large dataset.

  Example – Converting big sort into smaller multiple sorts

```
/*          QUERY 1             */

SELECT ProductID , MAX(SalesOrderID) SalesOrderID
FROM [Sales].[SalesOrderDetail] s
GROUP BY ProductID
ORDER BY SalesOrderID

/*       Rewritten Query 1 as  QUERY 2     */

SELECT p.ProductID,tr.SalesOrderID
FROM [Production].[Product] p
CROSS APPLY
    (
      SELECT TOP 1 SalesOrderID
      FROM [Sales].[SalesOrderDetail] s
```

Pawan Kumar Khowal / MSBISkills.com

```
        WHERE s.ProductID = p.ProductID
        ORDER BY SalesOrderID DESC
      ) tr
```

What we are doing in the above query is we are reading all product ids from products master table and then for each product id we are hitting orders table to find out the latest salesorderid which works perfectly in our case since we need only 1 salesorderid per product that too we are getting with the help of nonclustered index seek. In the first query we are reading whole SalesOrderDetail table which is very expensive as the order details tables are normally very big in size and in this case we consume lot of IO and CPU.

- **Move Order by in the application tier or web tier.**

  SQL SERVER is now a day's very expensive software. You can check out the pricing @ http://www.microsoft.com/en-in/server-cloud/products/sql-server/purchasing.aspx (7K per Core I think). With this kind of money we can buy multiple app servers and we can easily handle sorting in there!

  Example – Sorting data via C# function in app tier.

```
private void btnSort_Click(object sender, EventArgs e)
        {
                string conn = "Server=.
                                   ;Database=Pawan;Trusted_Connection=true";
                DataSet ds = new DataSet();

                SqlConnection connection = new SqlConnection(conn);
              SqlCommand command = new SqlCommand(" SELECT Id FROM
                                                [dbo].[MissingNumbers] ",
                                                  connection);

                SqlDataAdapter adapter = new SqlDataAdapter(command);
                adapter.Fill(ds);

                DataView dv = ds.Tables[0].DefaultView;
                dv.Sort = "Id";

                dataGridView1.DataSource = dv;
        }
```

Pawan Kumar Khowal/MSBISkills.com

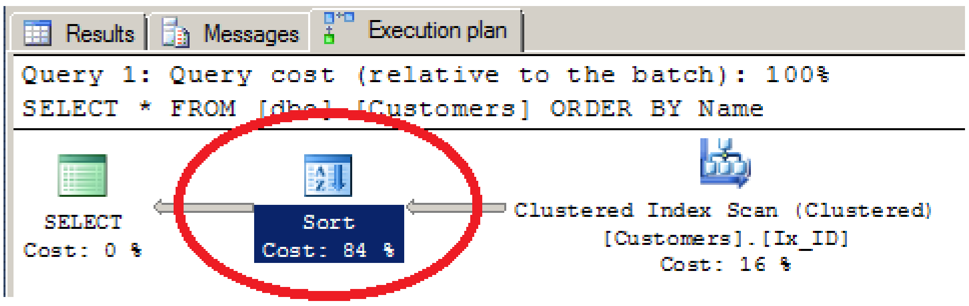
- **By Creating an Index**

  The quickest and easiest way to avoid a SORT operator is by creating an Index. As we know, indexes are ordered by the columns so if you create an index covering your query, the Query Optimizer identifies this index and uses it to avoid a SORT operation. Here please remember that you are changing your DB. There should be a rule of thumb like how many indexes you wanted to have per table. E.g. for my case I go with maximum 5 indexes per table. This should be your last resort.

  Example – Removing Sort operator via Index

```
/* Query 1 - Without Index */

SELECT * FROM [dbo].[Customers]
ORDER BY Name
```



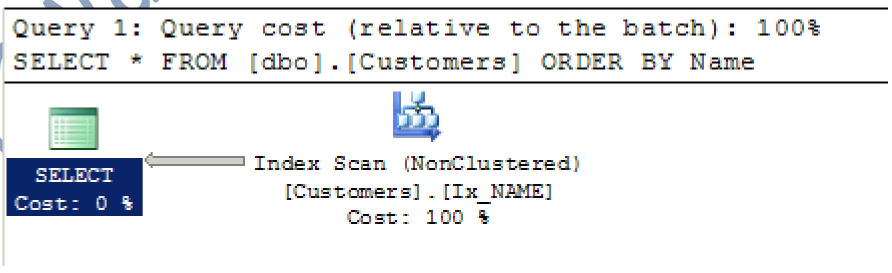
```
Query 1: Query cost (relative to the batch): 100%
SELECT * FROM [dbo].[Customers] ORDER BY Name
```

```
/* Lets create an Index first */

CREATE INDEX [Ix_NAME] ON [dbo].[Customers](NAME)
GO

/* Query 2 - With Index */

SELECT * FROM [dbo].[Customers]
ORDER BY Name
```



```
Query 1: Query cost (relative to the batch): 100%
SELECT * FROM [dbo].[Customers] ORDER BY Name
```

Pawan Kumar
BISkills.com

Thanks for Reading!

Pawan Khowal

MSBISkills.com

Pawan Kumar Khowal / MSBISkills.com

Copy protected with Online-PDF-NoCopy.com