هنگام WHERE, JOIN, ORDERBY, SELECT استفاده از ترتیب ایجاد ایندکس ها

ترتیبی که ستون ها در ایندکس های شما تعیین می شوند، بر این موضوع تاثیر دارد که هنگامی که SQL Optimizer، query شما را تجزیه می کند، آیا کل ایندکس مورد استفاده قرار می گیرد یا نه.

توضیحات:

با نگاه کردن به explain plan برای یک query متوجه خواهید شد که SQL Optimizer ابتدا عبارت WHERE را تجزیه کرده و سپس عبارت JOIN که با عبارت ORDRERED BY دنبال می شود و در نهایت داده ی انتخاب شده را پردازش می کند. براساس این واقعیت اگر بخواهید کل ایندکس مورد استفاده قرار بگیرد،نیاز شما به تعیین ستون هایی در ایندکس در این ترتیب ضروری است. این مسئله به ویژه در هنگام ایجاد یک ایندکس پوشاننده (covering index) درست می باشد. اجازه بدهید به query ساده ی زیر به عنوان یک مثال نگاهی داشته باشیم.

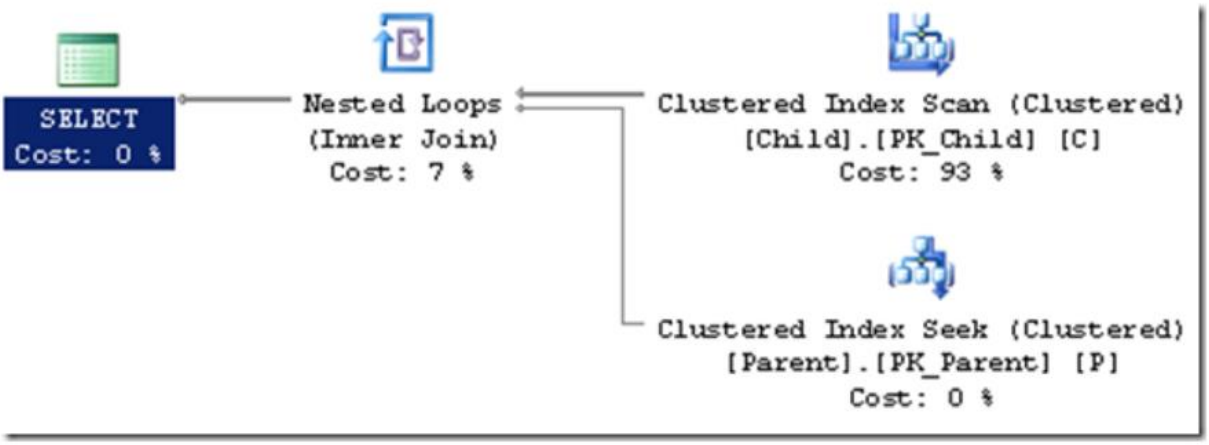
```
SELECT P.ParentID,C.ChildID,C.IntDataColumn,C.VarcharDataColumn
 FROM [dbo].[Parent] P INNER JOIN
    [dbo].[Child] C ON P.ParentID=C.ParentID
WHERE C.IntDataColumn=32433
ORDER BY ChildID
```

و از وضعیت ایندکس زیر برای نشان دادن چگونگی افزودن ستون ها به ایندکس در ترتیبی که در بالا ذکر کردیم، WHERE- JOIN-ORDER BY-SELECT ، استفاده می کنیم که این امر اجرای query را بهبود خواهد بخشید. دو نکته قابل ذکر وجود دارد: اول اینکه من کل وضعیت ایندکس را در اینجا وارد کرده ام، اما شما می توانید ستون ها را همزمان وارد کنید تا تفاوت ها را در هر مرحله مشاهده کنید. نکته دوم اینکه دومین وضعیت ایندکس (index statement) ایجاد شده فقط یک گزینه برای افزودن مستقیم ستون های SELECT به ایندکس می باشد و در عوض آنها بخشی از عبارت INCLUDE می باشند:

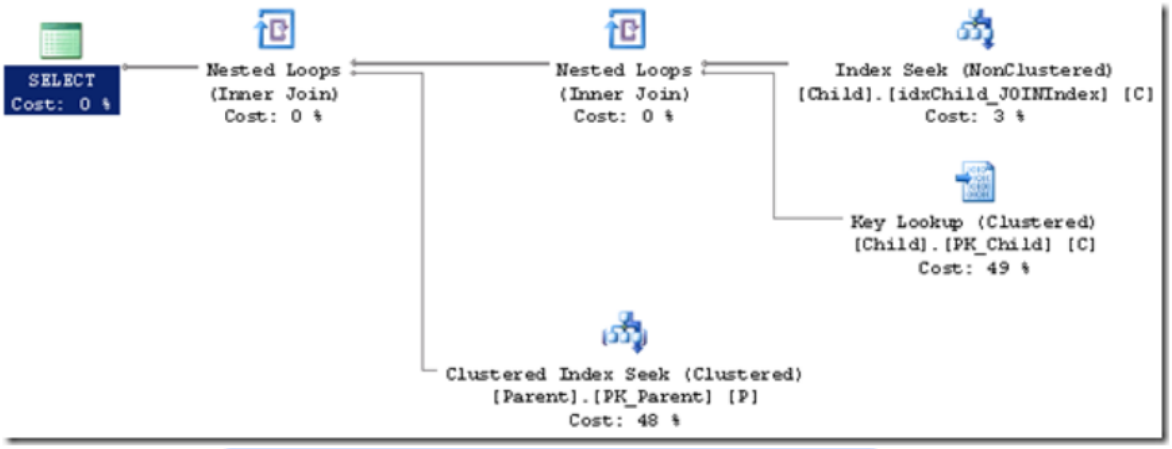
```
CREATE NONCLUSTERED INDEX idxChild_JOINIndex
ON [dbo].[Child] ([IntDataColumn],[ParentID],[ChildID],[VarcharDataColumn])

CREATE NONCLUSTERED INDEX idxChild_JOINIndex
ON [dbo].[Child] ([IntDataColumn],[ParentID],[ChildID]) INCLUDE ([VarcharDataColumn])

-- cleanup statements
DROP INDEX Child.idxChild_JOINIndex
```

اجازه بدهید نگاهی به explain plans برای هرکدام از این query ها داشته باشیم هنگامیکه ستون هایی را به ایندکس اضافه می کنیم.

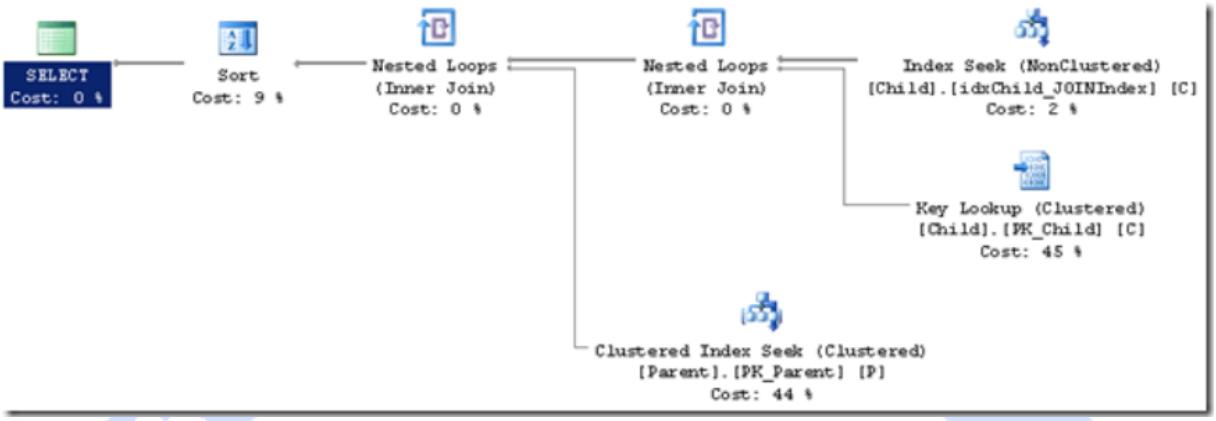
No Index



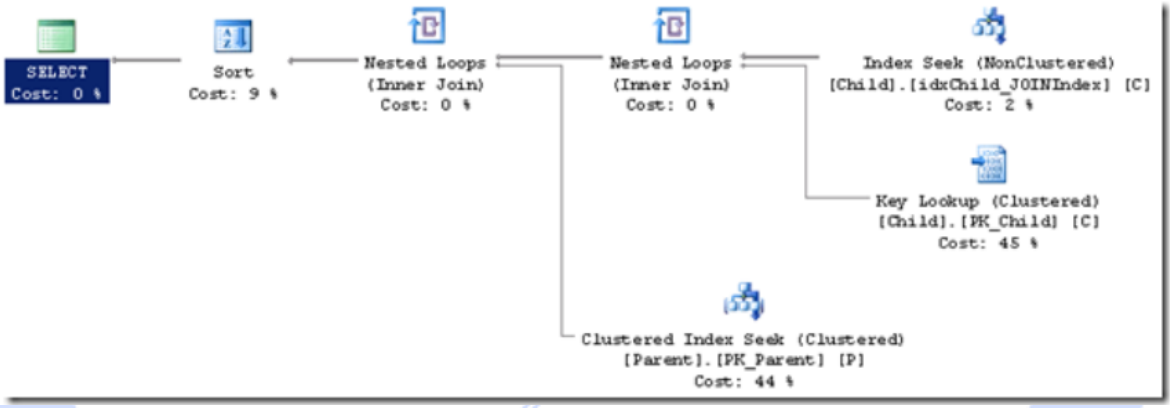
WHERE Index



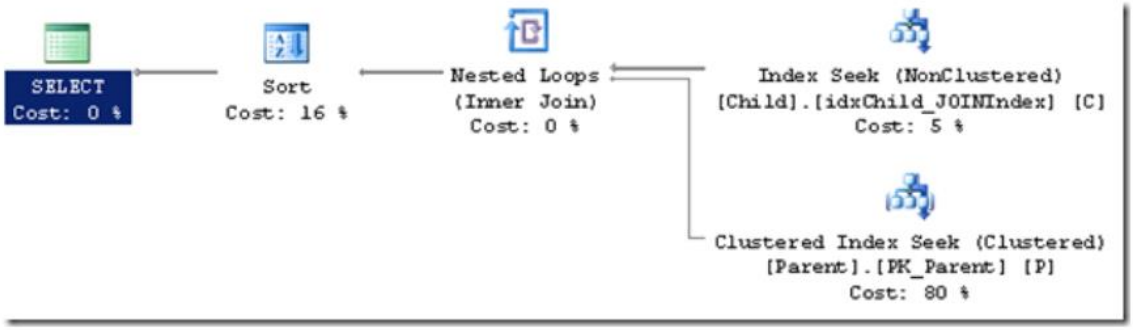
WHERE,JOIN Index



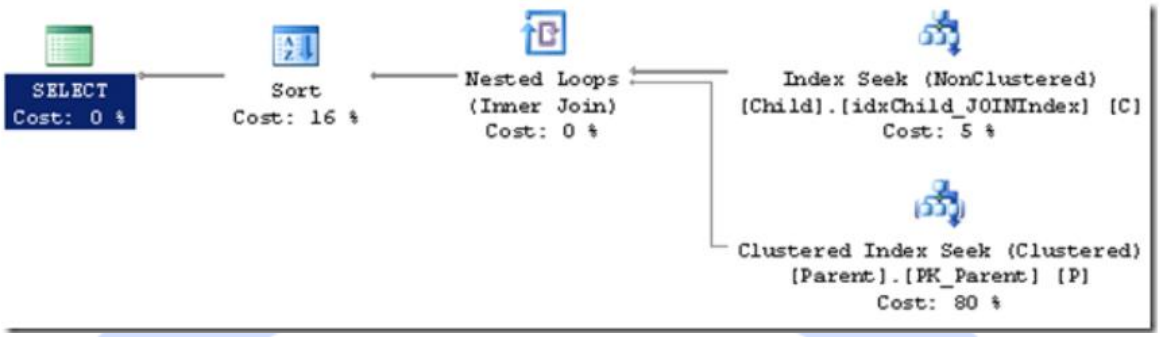
WHERE,JOIN,ORDER BY Index



WHERE,JOIN,ORDER BY, SELECT Index



WHERE,JOIN,ORDER BY, INCLUDE Index



بیان این امر که آیا در هر مرحله بهبودی حاصل می شود یا نه تنها از طریق explain plans دشوار می باشد، یا فقط برای افزودن ایندکس اولیه که منجر به حذف ایندکس اسکن می شود، اجازه بدهید نگاهی به نتایج SQL Profiler داشته باشیم تا مزیت اجرای حقیقی را مشاهده کنیم.

Table Type	CPU	Reads	Writes	Duration
No Index	110	14271	0	103
WHERE Index	0	129	0	2
WHERE, JOIN Index	0	117	0	0
WHERE, JOIN, ORDER BY Index	0	117	0	0
WHERE, JOIN, ORDER BY, SELECT Index	0	60	0	0
WHERE, JOIN, ORDER BY, INCLUDE Index	0	60	0	0

از این نتایج مشاهده می کنیم که با افزودن هر ستون موتور SQL سرور باید خواندن های کمتری را اجرا کند تا query را کمی سریعتر اجرا کند. تنها استثنا آن مرحله ای است که در آن ORDER BY را به ایندکس اضافه کردیم اما این امر می تواند به این واقعیت نسبت داده شود که ما در حال مرتب کردن براساس ChildID می باشیم که یک کلید اولیه می باشد، بنابراین تقریبا منظم شده است. نکته ی دیگری که باید به آن توجه کنیم این است که در واقع هیچ تفاوت اجرایی بین افزودن مستقیم ستون SELECT به ایندکس در مقابل استفاده از عبارت INCLUDE نمی باشد.