Artigo
Kyle Baxter · jan 18, 2021 1min de leitura

Melhore o desempenho do SQL para consultas de intervalo de datas

As consultas utilizando intervalo de datas estão muito lentas para você? O desempenho do SQL te desanima? Eu tenho um estranho truque que pode te ajudar! (Desenvolvedores de SQL odeiam isso!)*

Se você tiver uma classe que registra os timestamps quando os dados são adicionados, esses dados estarão em sequência com seus valores IDKEY - *isto é, $TimeStamp_1 < TimeStamp_2$ se e somente se $ID_1 < ID_2$ para todos os IDs e valores de timestamps na tabela* - então, você pode usar esse conhecimento para aumentar o desempenho de consultas para intervalos de timestamps. Considere a seguinte tabela:

```
Class User.TSOrder extends %Persistent
{

Property TS as %TimeStamp;

Property Data as %String (MAXLEN=100, MINLEN=200);

Index TSIdx on TS;

Index Extent [type=bitmap, extent];

}
```

Popular isso com 30.000.000 de linhas aleatórias com datas dos últimos 30 dias, resultará em 1.000.000 de linhas por dia. Agora, se quisermos consultar as informações de um determinado dia, você pode escrever o seguinte

```
SELECT ID, TS, Data
FROM TSOrder
WHERE
     TS >= '2016-07-01 00:00:00.00000' AND
     TS <= '2016-07-01 23:59:59.999999'
```

Uma consulta razoável, com certeza. No meu sistema, no entanto, isso realizou 2.172.792 referências a globais em 7,2 segundos. Mas, sabendo que os IDs e TimeStamps estão na mesma ordem, podemos usar os TimeStamps para obter um intervalo de ID. Considere a seguinte consulta:

```
SELECT ID, TS, Data
FROM TSOrder
WHERE
     ID >= (SELECT TOP 1 ID FROM TSOrder WHERE TS >='2016-07-01 00:00:00.00000' ORDER
 BY TS ASC) AND
     ID <= (SELECT TOP 1 ID FROM TSOrder WHERE TS <='2016-07-01 23:59:59.999999' ORDE
R BY TS DESC)
```

A nova consulta é concluída em 5,1 segundos e realiza apenas 999.985 referências a globais**!

Essa técnica pode ser aplicada de forma mais pragmática a tabelas com mais campos indexados e consultas que possuem várias cláusulas WHERE. O intervalo de ID gerado a partir das subconsultas pode ser colocado no formato de bitmap, gerando uma velocidade incrível quando você tem uma solução de vários índices. A tabela Ens.MessageHeader é um ótimo exemplo onde você pode colocar para funcionar esse truque.

Vamos ser claros - este é um *EXEMPLO* de uma vitória. Se você tiver muitas condições na cláusula WHERE na mesma tabela (e elas estão indexadas), Então esta técnica pode lhe dar **GRANDES** vitórias! Experimente em suas consultas!

- Desenvolvedores de SQL não odeiam isso, mas se a Internet nos ensinou alguma coisa é que bordões obtêm mais tráfego.

** Ao testar as consultas que retornam muitas linhas, o SMP não consegue lidar com isso, e a maior parte do tempo é gasto na exibição dos dados. A maneira adequada de testar é com o SQL integrado ou dinâmico, examinando os resultados, mas gerando a saída deles por um tempo, e usando o SQL Shell para suas contagens de globais. Você também pode usar o SQL Stats para isso.

#Code Snippet #SQL