SQL Server Exercises Advanced Performance

HO GENT

Installation of test database Stack Overflow

All data of the famous forum stackoverflow is stored in a SQL Server database. We will use a fragment of this database (years 2008-2010). Because of the size we don't deliver a backup file but we restore the database directly from the datafile `StackOverflow2010.mdf` and the logfile `StackOverflow2010.ldf`.

Download the mdf- and logfile from:
http://downloads.brentozar.com.s3.amazonaws.com/StackOverflow2010.7z

Caution the zipped size is about 1 Gigabyte, unzipped the Database is about 10 Gigabyte. You can unzip the .7z file using 7 zip.

Copy the files to your SQL Server data directory

- SQL Server 2017: C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA
- SQL Server 2019: C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\DATA

Now, in SQL Server Management Studio, right mouse click on Databases, Attach, Add and browse to the mdf file. You now have a database `StackOverflow2010`.

1. Explore the data

Draw the database diagram. Guess, from the tables names, which table is the largest (in terms of number of records). How many records does this table have?

2. Exercises

Make a plan for each exercise:

1. *Write the a functional correct query that also takes into account the 10 "rules of thumb".*
2. *Check the EXECUTION PLAN before undertaking any actions. Interpret it. Find possible areas of improvement.*
    a. *TABLE SCAN and SORT in execution plan always have to be avoided.*
    b. *INDEX SCAN is only allowed when just sorting, not when searching.*
3. *Create and index for fields on which we search (WHERE, GROUP BY, HAVING, JOIN .. ON) or sort (ORDER BY). Use a covering index (INCLUDE clause of index) if fields occur in SELECT that do not need indexing.*
4. *Execute the query again and recheck the EXECUTION PLAN. Are there still areas of possible improvement? If so, repeat steps 3 and 4.*
    a. *Only INDEX SEEK = optimal situation, no improvement possible.*
    b. *Only INDEX SCAN = only allowed when just sorting.*

1. Consider following query. How can we speed up this query?

```
select u.id,u.displayname,u.Reputation,u.UpVotes
from Users u
order by u.Reputation desc,u.UpVotes asc;
```

2. We know the index is not used when the indexed field is used in a function (like year(creationdate)). Some database systems (like Oracle) provide function based indexes to cope with this, but MS doesn't. What technique could you use to simulate the behaviour of function based indexes? You can use this technique as one of the approaches in the next exercise.

3. Show all posts created in the year 2010 in an efficient way. Show id, title and viewcount.
Give two approaches.

4. Count the number of votes per year (year of creationdate of post) in an efficient way. What's the downside of this approach? Is it recommended in the use case of Stack Overflow? How often will the query have to be executed.

5. Make a ranking of all users (id, displayname,number of badges) based on the number of badges in an efficient way.

6. Import the file readings.csv using the **SQL Server 2019 Import and Export Data** tool (from the Windows menu) in whatever database to a table **readings**. This table contains readings from energy monitors on different locations. Choose the right target data types in the wizard. Explore the data in the table after import.

   See some of the most common queries:

   ```
   select (select total_consumption_day+total_consumption_night from readings where
   location = %s and timestamp =(select max(timestamp) from readings where location = %s)) -
   (select total_consumption_day+total_consumption_night from readings where location = %s
   and timestamp = (select max(timestamp) from readings where location = %s and timestamp
   < current_date))
   ```

   ```
   select -actual_injection from readings where timestamp = (select max(timestamp) from
   readings where location = %s ) and location = %s
   (%s is a bind variable in Python)
   ```

   Create a useful partitioning scheme.