**Performance Techniques:**

Best performance is the main concern to develop a successful application. Like a coin database is the tail side (back-end) of an application. A good database design provides best performance during data manipulation which results into the best performance of an application.

During database designing and data manipulation we should consider the following key points:

1. Choose Appropriate Data Type

Choose appropriate SQL Data Type to store your data since it also helps in to improve the query performance. Example: To store strings use varchar in place of text data type since varchar performs better than text. Use text data type, whenever you required storing of large text data (more than 8000 characters). Up to 8000 characters data you can store in varchar.

1. Avoid nchar and nvarchar

Practice to avoid nchar and nvarchar data type since both the data types takes just double memory as char and varchar. Use nchar and nvarchar when you required to store Unicode (16-bit characters) data like as Hindi, Chinese characters etc.

1. Avoid NULL in fixed-length field

Practice to avoid the insertion of NULL values in the fixed-length (char) field. Since, NULL takes the same space as desired input value for that field. In case of requirement of NULL, use variable-length (varchar) field that takes less space for NULL.

1. Avoid \* in SELECT statement

Practice to avoid \* in Select statement since SQL Server converts the \* to columns name before query execution. One more thing, instead of querying all columns by using \* in select statement, give the name of columns which you required.

* 1. ***-- Avoid***
  2. **SELECT \* FROM tblName**
  3. ***--Best practice***
  4. **SELECT col1,col2,col3 FROM tblName**

1. Use EXISTS instead of IN

Practice to use EXISTS to check existence instead of IN since EXISTS is faster than IN.

* 1. ***-- Avoid***
  2. **SELECT Name,Price FROM tblProduct**
  3. **where ProductID IN (Select distinct ProductID from tblOrder)**
  4. ***--Best practice***
  5. **SELECT Name,Price FROM tblProduct**
  6. **where ProductID EXISTS (Select distinct ProductID from tblOrder)**

1. Avoid Having Clause

Practice to avoid Having Clause since it acts as filter over selected rows. Having clause is required if you further wish to filter the result of an aggregations. Don't use HAVING clause for any other purpose.

1. Create Clustered and Non-Clustered Indexes

Practice to create clustered and non clustered index since indexes helps in to access data fastly. But be careful, more indexes on a tables will slow the INSERT,UPDATE,DELETE operations. Hence try to keep small no of indexes on a table.

1. Keep clustered index small

Practice to keep clustered index as much as possible since the fields used in clustered index may also used in nonclustered index and data in the database is also stored in the order of clustered index. Hence a large clustered index on a table with a large number of rows increase the size significantly. Please refer the article [Effective Clustered Indexes](http://www.simple-talk.com/sql/learn-sql-server/effective-clustered-indexes/)

1. Avoid Cursors

Practice to avoid cursor since cursor are very slow in performance. Always try to use SQL Server cursor alternative. Please refer the article [Cursor Alternative](http://www.dotnet-tricks.com/Tutorial/sqlserver/IT5G180512-SQL-Server-Cursor-Alternatives.html).

1. Use Table variable inplace of Temp table

Practice to use Table varible in place of Temp table since Temp table resides in the TempDb database. Hence use of Temp tables required interaction with TempDb database that is a little bit time taking task.

1. Use UNION ALL inplace of UNION

Practice to use UNION ALL in place of UNION since it is faster than UNION as it doesn't sort the result set for distinguished values.

1. Use Schema name before SQL objects name

Practice to use schema name before SQL object name followed by "." since it helps the SQL Server for finding that object in a specific schema. As a result performance is best.

* 1. ***--Here dbo is schema name***
  2. **SELECT col1,col2 from dbo.tblName**
  3. ***-- Avoid***
  4. **SELECT col1,col2 from tblName**

1. Keep Transaction small

Practice to keep transaction as small as possible since transaction lock the processing tables data during its life. Some times long transaction may results into deadlocks. Please refer the article [SQL Server Transactions Management](http://www.dotnet-tricks.com/Tutorial/sqlserver/c2XF120412-SQL-Server-Transactions-Management.html)

1. SET NOCOUNT ON

Practice to set NOCOUNT ON since SQL Server returns number of rows effected by SELECT,INSERT,UPDATE and DELETE statement. We can stop this by setting NOCOUNT ON like as:

* 1. **CREATE PROCEDURE dbo.MyTestProc**
  2. **AS**
  3. **SET NOCOUNT ON**
  4. **BEGIN**
  5. **.**
  6. **.**
  7. **END**

1. Use TRY-Catch

Practice to use TRY-CATCH for handling errors in T-SQL statements. Sometimes an error in a running transaction may cause deadlock if you have no handle error by using TRY-CATCH. Please refer the article [Exception Handling by TRY…CATCH](http://www.dotnet-tricks.com/Tutorial/sqlserver/O3P3120412-SQL-Server-Exception-Handling-by-TRY%E2%80%A6CATCH.html)

1. Use Stored Procedure for frequently used data and more complex queries

Practice to create stored procedure for quaery that is required to access data frequently. We also created stored procedure for resolving more complex task.

1. Avoid prefix "sp\_" with user defined stored procedure name

Practice to avoid prefix "sp\_" with user defined stored procedure name since system defined stored procedure name starts with prefix "sp\_". Hence SQL server first search the user defined procedure in the master database and after that in the current session database. This is time consuming and may give unexcepted result if system defined stored procedure have the same name as your defined procedure.

Summary

In this article, I expose some key point to improve your SQL Server database performance. I hope after reading this article you will be able to use these tips with in your Sql Server database designing and manipulation. I would like to have feedback from my blog readers. Please post your feedback, question, or comments about this article.