Stored Procedures Optimization Tips

Source	http://www.mssqlcity.com/Tips/stored_procedures_optimization.htm

Use stored procedures instead of heavy-duty queries.
This can reduce network traffic, because your client will send to server only stored procedure name (perhaps with some parameters) instead of large heavy-duty queries text. Stored procedures can be used to enhance security and conceal underlying data objects also. For example, you can give the users permission to execute the stored procedure to work with the restricted set of the columns and data.

Include the SET NOCOUNT ON statement into your stored procedures to stop the message indicating the number of rows affected by a Transact-SQL statement.
This can reduce network traffic, because your client will not receive the message indicating the number of rows affected by a Transact-SQL statement.

Call stored procedure using its fully qualified name.
The complete name of an object consists of four identifiers: the server name, database name, owner name, and object name. An object name that specifies all four parts is known as a fully qualified name. Using fully qualified names eliminates any confusion about which stored procedure you want to run and can boost performance because SQL Server has a better chance to reuse the stored procedures execution plans if they were executed using fully qualified names.

Consider returning the integer value as an RETURN statement instead of an integer value as part of a recordset.
The RETURN statement exits unconditionally from a stored procedure, so the statements following RETURN are not executed. Though the RETURN statement is generally used for error checking, you can use this statement to return an integer value for any other reason. Using RETURN statement can boost performance because SQL Server will not create a recordset.

Don't use the prefix "sp_" in the stored procedure name if you need to create a stored procedure to run in a database other than the master database.
The prefix "sp_" is used in the system stored procedures names. Microsoft does not recommend to use the prefix "sp_" in the user-created stored procedure name, because SQL Server always looks for a stored procedure beginning with "sp_" in the following order: the master database, the stored procedure based on the fully qualified name provided, the

stored procedure using dbo as the owner, if one is not specified. So, when you have the stored procedure with the prefix "sp_" in the database other than master, the master database is always checked first, and if the user-created stored procedure has the same name as a system stored procedure, the user-created stored procedure will never be executed.

Use the sp_executesql stored procedure instead of the EXECUTE statement.
The sp_executesql stored procedure supports parameters. So, using the sp_executesql stored procedure instead of the EXECUTE statement improve readability of your code when there are many parameters are used. When you use the sp_executesql stored procedure to executes a Transact-SQL statements that will be reused many times, the SQL Server query optimizer will reuse the execution plan it generates for the first execution when the change in parameter values to the statement is the only variation.

Use sp_executesql stored procedure instead of temporary stored procedures.
Microsoft recommends to use the temporary stored procedures when connecting to earlier versions of SQL Server that do not support the reuse of execution plans. Applications connecting to SQL Server 7.0 or SQL Server 2000 should use the sp_executesql system stored procedure instead of temporary stored procedures to have a better chance to reuse the execution plans.

If you have a very large stored procedure, try to break down this stored procedure into several sub-procedures, and call them from a controlling stored procedure.
The stored procedure will be recompiled when any structural changes were made to a table or view referenced by the stored procedure (for example, ALTER TABLE statement), or when a large number of INSERTS, UPDATES or DELETES are made to a table referenced by a stored procedure. So, if you break down a very large stored procedure into several sub-procedures, you get chance that only a single sub-procedure will be recompiled, but other sub-procedures will not.

Try to avoid using temporary tables inside your stored procedure.
Using temporary tables inside stored procedure reduces the chance to reuse the execution plan.

Try to avoid using DDL (Data Definition Language) statements inside your stored procedure.
Using DDL statements inside stored procedure reduces the chance to reuse the execution plan.

Add the WITH RECOMPILE option to the CREATE PROCEDURE statement if you know that your query will vary each time it is run from the stored procedure.
The WITH RECOMPILE option prevents reusing the stored procedure execution plan, so SQL Server does not cache a plan for this procedure and the procedure is recompiled at run time. Using the WITH RECOMPILE option can boost performance if your query will vary each time it is run from the stored procedure because in this case the wrong execution plan will not be used.

Use SQL Server Profiler to determine which stored procedures has been recompiled too often.
To check the stored procedure has been recompiled, run SQL Server Profiler and choose to trace the event in the "Stored Procedures" category called "SP:Recompile". You can also trace the event "SP:StmtStarting" to see at what point in the procedure it is being recompiled. When you identify these stored procedures, you can take some correction actions to reduce or eliminate the excessive recompilations.

***~~~ End of Article ~~~***