Title:
SQL Server Indexes

Word Count:
388

Summary:
SQL Server Two Kinds Of Indexes Clustered Non-Clustered Clustered Indexes Require Data Table Physically Sorted Order Non Clustered Index Not Data Physically Sorted Enterprise Manager Create Index Wizard Index Fragmented SQL Server Indexes Introduction SQL Server Indexes

Keywords:
databse design nj, It Consulting nj, custom programming

Article Body:
"SQL Server Indexes
A database index is similar to an index in a book – it is comprised of a lookup value, and a number identifier that corresponds to the row number in a table. In SQL Server, there are two kinds of indexes – clustered and non-clustered. Clustered Indexes require that the data in the table is physically sorted in the order of the index. Because the data in a table can be physically sorted only one way, there can be at most only one clustered index per table. Non clustered index do not require that data be physically sorted, so there can be more that one non-clustered index per table. In fact SQL Server allows up to 249 non-clustered indexes per table. Because data is not physically sorted, range searches using a non clustered index are not very efficient.

The command for creating an index in T-SQL is

CREATE [UNIQUE] [CLUSTERED | NONCLUSTERED] INDEX index_name ON { table | view } (column [ASC | DESC] [,...n]) [WITH < index_option > [,...n]] [ON filegroup] < index_option > :: = { PAD_INDEX | FILLFACTOR = fillfactor | IGNORE_DUP_KEY | DROP_EXISTING | STATISTICS_NORECOMPUTE | SORT_IN_TEMPDB }

PAD_INDEX specifies the percentage of space left free on the non-leaf levels of the index. FILLFACTOR specifies the percentage to fill the leaf pages. SORT_IN_TEMPDB specifies that intermediate results of the sort will be stored in tempdb. This increases disk space requirement but affects speed index creation. STATISTICS_NO_RECOMPUTE tells the system not to automatically update index statistics.

Of course, indexes can also be created and managed using the Enterprise Manager. They can be created using the Create Index Wizard, from the Database Diagram, or by modifying fields in the Table Designer.

There is a trade off with indexes. While they speed up execution of queries immensely, there is overhead associated with them. They consume additional disk space, and require additional time to update themselves whenever data is updated or appended to a table. When loading large amounts of data it may pay to drop the index prior to the loading, then recreate the index after the new records have been appended to the table. Indexes can be dropped using the Table Designed, or by using the DROP INDEX command.

Indexes can also become fragmented. To defrag an index, either drop and recreate the index, or issue the command dbcc indexdefrag.

"