通常情况下，建立索引是加快查询速度的有效手段。但索引不是万能的，靠索引并不能实现对所有数据的快速存取。事实上，如果索引策略和数据检索需求严重不符的话，建立索引反而会降低查询性能。因此在实际使用当中，应该充分考虑到索引的开销，包括**磁盘空间的开销及处理开销**（如资源竞争和加锁）。例如，**如果数据频繁的更新或删加，就不宜建立索引。**

本文简要讨论一下**聚簇索引的特点及其与非聚簇索引的区别**。

- 建立索引：

在 SQL 语言中，建立聚簇索引使用 CREATE INDEX 语句，格式为：

```
CREATE CLUSTER INDEX index_name ON table_name(column_name1,column_name2,...);
```

- 存储特点：

1. 聚集索引。表数据按照索引的顺序来存储的，也就是说索引项的顺序与表中记录的物理顺序一致。对于聚集索引，叶子结点即存储了真实的数据行，不再有另外单独的数据页。 **在一张表上最多只能创建一个聚集索引，因为真实数据的物理顺序只能有一种。**
2. 非聚集索引。表数据存储顺序与索引顺序无关。对于非聚集索引，叶结点包含索引字段值及指向数据页数据行的逻辑指针，其行数量与数据表行数据量一致。

总结一下：聚集索引是一种稀疏索引，数据页上一级的索引页存储的是页指针，而不是行指针。而对于非聚集索引，则是密集索引，在数据页的上一级索引页它为每一个数据行存储一条索引记录。

- 更新表数据

1、向表中插入新数据行

如果一张表没有聚集索引，那么它被称为“堆集”（Heap）。这样的表中的数据行没有特定的顺序，所有的新行将被添加到表的末尾位置。而建立了聚簇索引的数据表则不同：最简单的情况下，插入操作根据索引找到对应的数据页，然后通过挪动已有的记录为新数据腾出空间，最后插入数据。如果数据页已满，则需要拆分数据页，调整

索引指针（且如果表还有非聚集索引，还需要更新这些索引指向新的数据页）。而类似于自增列为聚集索引的，数据库系统可能并不拆分数据页，而只是简单的新添数据页。

2、从表中删除数据行

对删除数据行来说：删除行将导致其下方的数据行向上移动以填充删除记录造成的空白。如果删除的行是该数据页中的最后一行，那么该数据页将被回收，相应的索引页中的记录将被删除。对于数据的删除操作，可能导致索引页中仅有一条记录，这时，该记录可能会被移至邻近的索引页中，原索引页将被回收，即所谓的“索引合并”。

聚簇索引确定表中数据的物理顺序。聚簇索引类似于电话簿，后者按姓氏排列数据。由于聚簇索引规定数据在表中的物理存储顺序，因此一个表只能包含一个聚簇索引。但该索引可以包含多个列（组合索引），就像电话簿按姓氏和名字进行组织一样。汉语字典也是聚簇索引的典型应用， 在汉语字典里，索引项是字母+声调，字典正文也是按照先字母再声调的顺序排列。

聚簇索引对于那些经常要搜索范围值的列特别有效。使用聚簇索引找到包含第一个值的行后，便可以确保包含后续索引值的行在物理相邻。例如，如果应用程序执行的一个查询经常检索某一日期范围内的记录，则使用聚集索引可以迅速找到包含开始日期的行，然后检索表中所有相邻的行，直到到达结束日期。这样有助于提高此类查询的性能。同样，如果对从表中检索的数据进行排序时经常要用到某一列，则可以将该表在该列上聚簇（物理排序），避免每次查询该列时都进行排序，从而节省成本。

建立聚簇索引的思想

1、大多数表都应该有聚簇索引或使用分区来降低对表尾页的竞争，在一个高事务的环境中，对最后一页的封锁严重影响系统的吞吐量。

2、在聚簇索引下，数据在物理上按顺序排在数据页上，重复值也排在一起，因而在那些包含范围检查 (between、<、<=、>、>=)或使用 group by 或 orderby 的查询时，一旦找到具有范围中第一个键值的行，具有后续索引值的行保证物理上毗连在一起而不必进一步搜索，避免了大范围扫描，可以大 大提高查询速度。

3、在一个频繁发生插入操作的表上建立聚簇索引时，不要建在具有单调上升值的列(如IDENTITY)上，否则会经常引起封锁冲突。

4、在聚簇索引中不要包含经常修改的列，因为码值修改后，数据行必须移动到新的位置。

5、选择聚簇索引应基于 where 子句和连接操作的类型。