一、索引是什么  
  
索引是与表或视图关联的磁盘上结构，可以加快从表或视图中检索行的速度。索引包含由表或视图中的一列或多列生成的键。这些键存储在一个结构（B树）中，使 SQL Server 可以快速有效地查找与键值关联的行。  
  
表或视图可以包含以下类型的索引：

   \* 聚集  
         o聚集索引根据数据行的键值在表或视图中排序和存储这些数据行。索引定义中包含聚集索引列。每个表只能有一个聚集索引，因为数据行本身只能按一个顺序排序。  
         o只有当表包含聚集索引时，表中的数据行才按排序顺序存储。如果表具有聚集索引，则该表称为聚集表。如果表没有聚集索引，则其数据行存储在一个称为堆的无序结构中。  
    \* 非聚集  
         o 非聚集索引具有独立于数据行的结构。非聚集索引包含非聚集索引键值，并且每个键值项都有指向包含该键值的数据行的指针。  
         o从非聚集索引中的索引行指向数据行的指针称为行定位器。行定位器的结构取决于数据页是存储在堆中还是聚集表中。对于堆，行定位器是指向行的指针。对于聚集表，行定位器是聚集索引键。  
         o 您可以向非聚集索引的叶级添加非键列以跳过现有的索引键限制（900 字节和 16 键列），并执行完整范围内的索引查询。  
  
聚集索引和非聚集索引都可以是唯一的。这意味着任何两行都不能有相同的索引键值。另外，索引也可以不是唯一的，即多行可以共享同一键值。  
  
每当修改了表数据后，都会自动维护表或视图的索引。  
  
索引和约束  
  
对表列定义了 PRIMARY KEY 约束和 UNIQUE 约束时，会自动创建索引。例如，如果创建了表并将一个特定列标识为主键，则数据库引擎自动对该列创建 PRIMARY KEY 约束和索引。有关详细信息，请参阅创建索引（数据库引擎）。  
  
  
二、索引有什么用  
  
与书中的索引一样，数据库中的索引使您可以快速找到表或索引视图中的特定信息。索引包含从表或视图中一个或多个列生成的键，以及映射到指定数据的存储位置的指针。通过创建设计良好的索引以支持查询，可以显著提高数据库查询和应用程序的性能。索引可以减少为返回查询结果集而必须读取的数据量。索引还可以强制表中的行具有唯一性，从而确保表数据的数据完整性。  
  
设计良好的索引可以减少磁盘 I/O 操作，并且消耗的系统资源也较少，从而可以提高查询性能。对于包含SELECT、UPDATE、DELETE 或 MERGE 语句的各种查询，索引会很有用。例如，在 AdventureWorks数据库中执行的查询 SELECT Title, HireDate FROM HumanResources.Employee WHEREEmployeeID =250。执行此查询时，查询优化器评估可用于检索数据的每个方法，然后选择最有效的方法。可能采用的方法包括扫描表和扫描一个或多个索引（如果有）。  
  
扫描表时，查询优化器读取表中的所有行，并提取满足查询条件的行。扫描表会有许多磁盘 I/O操作，并占用大量资源。但是，如果查询的结果集是占表中较高百分比的行，扫描表会是最为有效的方法。  
  
查询优化器使用索引时，搜索索引键列，查找到查询所需行的存储位置，然后从该位置提取匹配行。通常，搜索索引比搜索表要快很多，因为索引与表不同，一般每行包含的列非常少，且行遵循排序顺序。  
  
查询优化器在执行查询时通常会选择最有效的方法。但如果没有索引，则查询优化器必须扫描表。您的任务是设计并创建最适合您的环境的索引，以便查询优化器可以从多个有效的索引中选择。SQLServer 提供的数据库引擎优化顾问可以帮助分析数据库环境并选择适当的索引。  
  
三、索引怎么用  
  
索引其实关键目的是为了加快检索速度而建立的，所以，怎么用索引是数据库系统本身的事情，作为数据库设计或使用者，设计并创建好索引然后体验加上索引后的查询变快的感觉就行了。所以，索引怎么用就变为了“怎么创建合适的索引”，以下说明这个问题：  
  
索引设计不佳和缺少索引是提高数据库和应用程序性能的主要障碍。设计高效的索引对于获得良好的数据库和应用程序性能极为重要。为数据库及其工作负荷选择正确的索引是一项需要在查询速度与更新所需开销之间取得平衡的复杂任务。如果索引较窄，或者说索引关键字中只有很少的几列，则需要的磁盘空间和维护开销都较少。而另一方面，宽索引可覆盖更多的查询。您可能需要试验若干不同的设计，才能找到最有效的索引。可以添加、修改和删除索引而不影响数据库架构或应用程序设计。因此，应试验多个不同的索引而无需犹豫。  
  
SQL Server中的查询优化器可在大多数情况下可靠地选择最高效的索引。总体索引设计策略应为查询优化器提供可供选择的多个索引，并依赖查询优化器做出正确的决定。这在多种情况下可减少分析时间并获得良好的性能。若要查看查询优化器对特定查询使用的索引，请在SQL Server Management Studio 中的“查询”菜单上选择“包括实际的执行计划”。  
  
不要总是将索引的使用等同于良好的性能，或者将良好的性能等同于索引的高效使用。如果只要使用索引就能获得最佳性能，那查询优化器的工作就简单了。但事实上，不正确的索引选择并不能获得最佳性能。因此，查询优化器的任务是只在索引或索引组合能提高性能时才选择它，而在索引检索有碍性能时则避免使用它。  
  
建议的索引设计策略包括以下任务：  
  
   1.了解数据库本身的特征。例如，它是频繁修改数据的联机事务处理 (OLTP) 数据库，还是主要包含只读数据的决策支持系统 (DSS)或数据仓库 (OLAP) 数据库？  
   2.了解最常用的查询的特征。例如，了解到最常用的查询联接两个或多个表将有助于决定要使用的最佳索引类型。  
   3.了解查询中使用的列的特征。例如，某个索引对于含有整数数据类型同时还是唯一的或非空的列是理想索引。筛选索引适用于具有定义完善的数据子集的列。  
   4.确定哪些索引选项可在创建或维护索引时提高性能。例如，对现有某个大型表创建聚集索引将会受益于 ONLINE 索引选项。ONLINE选项允许在创建索引或重新生成索引时继续对基础数据执行并发活动。  
   5.确定索引的最佳存储位置。非聚集索引可以与基础表存储在同一个文件组中，也可以存储在不同的文件组中。索引的存储位置可通过提高磁盘 I/O性能来提高查询性能。例如，将非聚集索引存储在表文件组所在磁盘以外的某个磁盘上的一个文件组中可以提高性能，因为可以同时读取多个磁盘。  
     或者，聚集索引和非聚集索引也可以使用跨越多个文件组的分区方案。在维护整个集合的完整性时，使用分区可以快速而有效地访问或管理数据子集，从而使大型表或索引更易于管理。有关详细信息，请参阅已分区表和已分区索引。在考虑分区时，应确定是否应对齐索引，即，是按实质上与表相同的方式进行分区，还是单独分区。  
  
#  设计索引。  
索引设计是一项关键任务。索引设计包括确定要使用的列，选择索引类型（例如聚集或非聚集），选择适当的索引选项，以及确定文件组或分区方案布置。  
  
# 确定最佳的创建方法。按照以下方法创建索引：  
  
    \* 使用 CREATETABLE 或 ALTER TABLE 对列定义 PRIMARY KEY 或 UNIQUE 约束  
     SQL Server 数据库引擎自动创建唯一索引来强制 PRIMARY KEY 或 UNIQUE约束的唯一性要求。默认情况下，创建的唯一聚集索引可以强制 PRIMARY KEY约束，除非表中已存在聚集索引或指定了唯一的非聚集索引。默认情况下，创建的唯一非聚集索引可以强制 UNIQUE约束，除非已明确指定唯一的聚集索引且表中不存在聚集索引。  
     还可以指定索引选项和索引位置、文件组或分区方案。  
     创建为 PRIMARY KEY 或 UNIQUE 约束的一部分的索引将自动给定与约束名称相同的名称。  
  
    \* 使用 CREATEINDEX 语句或 SQL Server Management Studio对象资源管理器中的“新建索引”对话框创建独立于约束的索引  
     必须指定索引的名称、表以及应用该索引的列。还可以指定索引选项和索引位置、文件组或分区方案。默认情况下，如果未指定聚集或唯一选项，将创建非聚集的非唯一索引。若要创建筛选索引，请使用可选的WHERE 子句。  
  
# 创建索引。  
要考虑的一个重要因素是对空表还是对包含数据的表创建索引。对空表创建索引在创建索引时不会对性能产生任何影响，而向表中添加数据时，会对性能产生影响。  
对大型表创建索引时应仔细计划，这样才不会影响数据库性能。对大型表创建索引的首选方法是先创建聚集索引，然后创建任何非聚集索引。在对现有表创建索引时，请考虑将ONLINE 选项设置为 ON。该选项设置为 ON 时，将不持有长期表锁以继续对基础表的查询或更新。  
  
简单的创建索引，可采用如下语句：  
CREATE INDEX IX\_ProductVendor\_VendorID  
    ONPurchasing.ProductVendor (VendorID, VendorName);

数据库中索引的优缺点  
为什么要创建索引呢？这是因为，创建索引可以大大提高系统的性能，**优点**:  
第一，通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。  
第二，可以大大加快数据的检索速度，这也是创建索引的最主要的原因。  
第三，可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义。  
第四，在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。  
第五，通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。   
  
也许会有人要问：增加索引有如此多的优点，为什么不对表中的每一个列创建一个索引呢？  
这种想法固然有其合理性，然而也有其片面性。虽然，索引有许多优点，但是，为表中的每一个列都增加索引，是非常不明智的。  
这是因为，增加索引也有许多不利的一个方面, **缺点**：  
第一，创建索引和维护索引要耗费时间，这种时间随着数据量的增加而增加。  
第二，索引需要占物理空间，除了数据表占数据空间之外，每一个索引还要占一定的物理空间，如果要建立聚簇索引，那么需要的空间  
就会更大。  
第三，当对表中的数据进行增加、删除和修改的时候，索引也要动态的维护，这样就降低了数据的维护速度。   
  
索引是建立在数据库表中的某些列的上面。因此，在创建索引的时候，应该仔细考虑在哪些列上可以创建索引，在哪些列上不能创建索  
引。一般来说，应该在这些列上创建索引，  
例如：在经常需要搜索的列上，可以加快搜索的速度；  
在作为主键的列上，强制该列的唯一性和组织表中数据的排列结构；  
在经常用在连接的列上，这些列主要是一些外键，可以加快连接的速度；  
在经常需要根据范围进行搜索的列上创建索引，因为索引已经排序，其指定的范围是连续的；  
在经常需要排序的列上创建索引，因为索引已经排序，这样查询可以利用索引的排序，加快排序查询时间；  
在经常使用在WHERE子句中的列上面创建索引，加快条件的判断速度。   
  
同样，对于有些列不应该创建索引。一般来说，不应该创建索引的的这些列具有下列特点：  
第一，对于那些在查询中很少使用或者参考的列不应该创建索引。这是因为，既然这些列很少使用到，因此有索引或者无索引，  
并不能提高查询速度。相反，由于增加了索引，反而降低了系统的维护速度和增大了空间需求。  
第二，对于那些只有很少数据值的列也不应该增加索引。这是因为，由于这些列的取值很少，例如人事表的性别列，  
在查询的结果中，结果集的数据行占了表中数据行的很大比例，即需要在表中搜索的数据行的比例很大。增加索引，  
并不能明显加快检索速度。第三，对于那些定义为text, image和bit数据类型的列不应该增加索引。这是因为，这些列的数据量要么  
相当大，要么取值很少。第四，当修改性能远远大于检索性能时，不应该创建索引。这是因为，修改性能和检索性能是互相矛盾的。  
当增加索引时，会提高检索性能，但是会降低修改性能。当减少索引时，会提高修改性能，降低检索性能。因此，当修改性能远远大  
于检索性能时，不应该创建索引。