**Индекс** – это объект базы данных, позволяющий **ускорить поиск** в определенной таблице, так как при этом данные организуются в виде сбалансированного бинарного дерева поиска.

Как и любой другой объект базы данных, индекс может быть создан с помощью оператора CREATE, модифицирован с помощью ALTER и удален с помощью оператора DROP. Для одной таблицы возможно построение нескольких индексов.

Индексы бывают кластеризованные, некластеризованные, уникальные, неуникальные и др.

Индекс в MS SQL Server - это структура данных, которая ускоряет операции поиска и сортировки записей в таблицах базы данных. Индекс представляет собой отдельный объект базы данных, который содержит отсортированные значения ключевых столбцов таблицы и указатель на соответствующую строку в таблице.

Индекс может быть создан на одном или нескольких столбцах таблицы, включая первичный ключ, внешние ключи и другие столбцы, часто используемые в запросах. Поиск записей с использованием индекса происходит намного быстрее, чем без него, так как SQL Server может искать записи по отсортированным значениям ключевого столбца, используя алгоритмы быстрого поиска.

Однако следует иметь в виду, что создание индексов может замедлять операции вставки, обновления и удаления данных, так как для каждой операции необходимо обновлять значения индекса. Поэтому при создании индексов необходимо учитывать баланс между быстродействием операций чтения и затратами на операции изменения данных.

**Кластеризованные индексы**

Обычно *кластеризованные* индексы создаются автоматически при создании таблицы если в ней присутствует первичный ключ (ограничение PRIMARY KEY).

Кластеризованные индексы физически упорядочены в соответствии со значениями индексируемых столбцов. В таблице может быть только один кластеризованный индекс.

С помощью системной процедуры SP\_HELPINDEX можно получить перечень индексов, связанных с заданной таблицей:



Кластеризованный индекс в MS SQL Server - это особый тип индекса, при котором строки таблицы физически упорядочиваются в соответствии с значениями ключевого столбца индекса. Каждая таблица в базе данных может иметь только один кластеризованный индекс, который определяет физический порядок хранения строк в таблице.

При использовании кластеризованного индекса SQL Server может быстро определять расположение нужной строки в таблице и выполнять запросы, использующие ключевой столбец индекса, с высокой скоростью. Кластеризованные индексы обеспечивают быстрый доступ к большим объемам данных, но при этом они могут занимать больше места на диске и могут вызывать затруднения при обновлении данных.

Кластеризованный индекс может быть создан на столбце, содержащем уникальные значения (например, первичный ключ), или на столбце, который содержит часто используемые значения в запросах. Важно знать, что при создании кластеризованного индекса таблица перестраивается, что может занять много времени при большом объеме данных.

Если создать кластеризованный индекс:

**CREATE clustered index #EXPLRE\_CL on #EXPLRE(TIND asc)**

**Некластеризованный индекс**

*Некластеризованные* индексы не влияют на физический порядок строк в таблице.

Можно создавать индексчы по нескольким столбцам – такие индексы называются *составными*. Можно создать составной неуникальный, некластеризованный индекс **#EX\_NONCLU** по двум столбцам **TKEY** и **CC** таблицы **#EX** с помощью оператора:

**CREATE index #EX\_NONCLU on #EX(TKEY, CC)**

В Microsoft SQL Server не кластеризованный индекс (Nonclustered Index) - это структура данных, которая содержит отдельные значения столбцов и ссылки на соответствующие строки таблицы, чтобы ускорить выполнение запросов на поиск, сортировку и фильтрацию данных.

Основное отличие некластеризованного индекса от кластеризованного индекса заключается в том, что данные не упорядочены в соответствии с ключом индекса. Вместо этого некластеризованный индекс содержит отдельную структуру данных, которая указывает на соответствующие строки в таблице.

Некластеризованный индекс можно создать на любом столбце таблицы, включая первичный ключ и внешний ключ. Он может содержать один или несколько столбцов, и каждый столбец может быть отсортирован в порядке возрастания или убывания.

Некластеризованный индекс может быть создан для ускорения поиска данных по любым столбцам таблицы, которые используются в условиях WHERE, JOIN, GROUP BY или ORDER BY запросов. При наличии некластеризованного индекса SQL Server может использовать его вместо сканирования всей таблицы для выполнения запросов на поиск и выборку данных.

**Некластеризованный индекс покрытия**

*Некластеризованный индекс покрытия* запроса позволяет включить в состав индексной строки значения одного или нескольких неиндексируемых столбцов.

**CREATE index #EX\_TKEY\_X on #EX(TKEY) INCLUDE (CC)**

Некластеризованный индекс покрытия (Covering Nonclustered Index) - это некластеризованный индекс, который включает в себя не только ключевые столбцы, но и другие столбцы, необходимые для выполнения запроса, без необходимости обращаться к основной таблице данных.

Представим, что у нас есть таблица с несколькими столбцами, и мы создали некластеризованный индекс на одном из столбцов. Если запрос содержит условия WHERE, которые соответствуют этому столбцу, тогда использование индекса ускорит выполнение запроса. Однако если этот же запрос требует вывода других столбцов, то база данных должна обратиться к основной таблице данных для получения значений этих столбцов. Это может замедлить выполнение запроса.

Создание некластеризованного индекса покрытия может улучшить производительность запросов, так как индекс будет включать все необходимые столбцы. Когда запрос будет выполняться, движок базы данных может использовать только индекс для получения значений, что позволяет избежать поиска в основной таблице данных.

**Некластеризованный фильтруемый индекс**

Если запросы основаны на WHERE-фильтрации строк, то может быть эффективным применение *фильтруемых некластеризованных* индексов.

**CREATE index #EX\_WHERE on #EX(TKEY) where (TKEY>=15000 and**

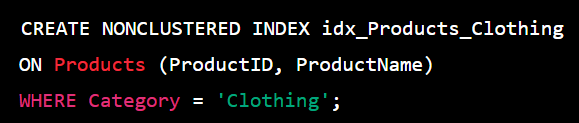
**TKEY < 20000);**

Здесь фильтруемый индекс создается только для строк таблицы **#EX,** которыеудовлетворяют логическому условию. Стоимость запросов уменьшится.

Некластеризованный фильтруемый индекс (Filtered Non-Clustered Index) - это специальный вид индекса в Microsoft SQL Server, который позволяет создать индекс только на определенную часть данных таблицы, которая удовлетворяет определенному условию.

Для создания некластеризованного фильтруемого индекса используется оператор CREATE INDEX с указанием ключевого слова WHERE, после которого указывается логическое выражение, определяющее, какие строки таблицы будут включены в индекс.

Пример создания некластеризованного фильтруемого индекса на таблице Products, который будет содержать только те строки, у которых значение столбца Category равно 'Clothing':



Такой индекс будет содержать только те строки таблицы Products, которые относятся к категории 'Clothing'. Это позволяет уменьшить размер индекса и ускорить выполнение запросов, которые используют этот индекс.

**Фрагментация диска**

Операции добавления и изменения строк базы данных могут повлечь образование неиспользуемых фрагментов в области памяти индекса. Процесс образования неиспользуемых фрагментов памяти называется *фрагментацией*. Фрагментация индексов снижает эффект от их применения. Пусть создан индекс:

**CREATE index #EX\_TKEY ON #EX(TKEY);**

Получить информацию о степени фрагментации индекса можно с помощью операторов:

**SELECT name [Индекс], avg\_fragmentation\_in\_percent [Фрагментация (%)]**

**FROM sys.dm\_db\_index\_physical\_stats(DB\_ID(N'TEMPDB'),**

**OBJECT\_ID(N'#EX'), NULL, NULL, NULL) ss JOIN sys.indexes ii on ss.object\_id = ii.object\_id and ss.index\_id = ii.index\_id WHERE name is not null;**

Фрагментация индексов - это процесс, при котором данные в индексе разбиваются на фрагменты, которые располагаются на разных страницах диска. В результате фрагментации индекса производительность запросов, использующих этот индекс, может ухудшаться.

Фрагментация индексов может возникать из-за множества причин, таких как удаление, вставка или обновление данных в таблице, что может привести к перемещению данных в индексе и, следовательно, к его фрагментации.

Для устранения фрагментации индексов в MS SQL Server существует несколько способов, например, можно использовать встроенные функции и процедуры, такие как ALTER INDEX REORGANIZE или ALTER INDEX REBUILD, которые позволяют перестроить индексы и уменьшить их фрагментацию.

Уменьшение фрагментации индексов может значительно повысить производительность запросов, которые используют эти индексы, и улучшить общую производительность базы данных.

Для избавления от фрагментации индекса предусмотрены две специальные операции: реорганизация и перестройка индекса.

**Реорганизация индекса**

*Реорганизация* (REORGANIZE) выполняется быстро, но после нее фрагментация будет убрана только на самом нижнем уровне.

Пусть выполнена реорганизация с помощью оператора ALTER для индекса **#EX\_TKEY.**

**ALTER index #EX\_TKEY on #EX reorganize;**

Тогда выполнение соответствующего запроса покажет, что уровень фрагментации значительно снизился, но не до конца.

Реорганизация индекса в базе данных Microsoft SQL Server является процессом оптимизации индексов путем перестройки фрагментированных страниц индекса и сжатия непустых страниц, что улучшает производительность запросов и снижает фрагментацию. Реорганизация индекса выполняется с использованием встроенной процедуры ALTER INDEX и может быть выполнена как онлайн (позволяющая обращаться к индексу во время реорганизации), так и офлайн (когда индекс заблокирован для чтения и записи).

Во время реорганизации индекса страницы перестраиваются таким образом, чтобы они следовали в порядке увеличения ключей индекса, что уменьшает время, необходимое для поиска данных и повышает производительность запросов. Реорганизация также уменьшает количество страниц, занимаемых индексом, и уменьшает использование дискового пространства.

В целом, реорганизация индекса является важной процедурой для оптимизации производительности базы данных, и ее следует регулярно выполнять для обеспечения эффективной работы системы.

**Перестройка индекса**

Операция *перестройки* (REBUILD) затрагивает все узлы дерева, поэтому после ее выполнения степень фрагментации равна нулю.

Пусть выполнена перестройка с помощью оператора ALTER для индекса **#EX\_TKEY** в режиме **OFFLINE**.

**ALTER index #EX\_TKEY on #EX rebuild with (online = off);**

Выполнением запроса о фрагментации можно оценить ее уровень.

Перестройка индекса (rebuild index) - это операция в MS SQL Server, которая удаляет существующий индекс и создает его заново, чтобы сбалансировать уровни фрагментации и улучшить производительность запросов. В отличие от реорганизации индекса, перестройка полностью перестраивает индекс, а не только перемещает и перестраивает его фрагменты.

Перестройка индекса обычно выполняется при очень высокой степени фрагментации индекса, что может привести к значительному снижению производительности запросов. Она также может быть полезной при добавлении большого количества новых данных в таблицу или при изменении структуры таблицы, так как это может привести к изменению оптимального плана выполнения запросов.

Перестройка индекса может занять довольно много времени и потребовать большого количества ресурсов, поэтому ее следует выполнять осторожно и только в необходимых случаях.

**FILLFACTOR**

Уровнем фрагментации можно в некоторой степени управлять, если при создании или изменении индекса использовать параметры FILLFACTOR и PAD\_INDEX. Параметр FILLFACTOR указывает процент заполнения индексных страниц нижнего уровня.

Пусть индекс пересоздан со значением параметра FILLFACTOR равным 65:

**DROP index #EX\_TKEY on #EX;**

**CREATE index #EX\_TKEY on #EX(TKEY)**

**with (fillfactor = 65);**

Параметр fillfactor в индексах в SQL Server определяет процент заполнения страниц индекса данными. По умолчанию значение fillfactor равно 100, что означает, что страницы индекса заполняются данными на 100%. Однако при создании индексов можно указать другое значение fillfactor, которое определяет, какой процент страниц индекса должен быть заполнен данными.

Значение fillfactor может быть установлено меньше 100, что означает, что страницы индекса заполняются данными на меньшее количество процентов, чем 100%. Например, при значении fillfactor 80 страницы индекса заполняются данными на 80% от своей полной емкости. Это означает, что на странице индекса будет оставлено 20% места для будущих записей.

Указание значения fillfactor менее 100 позволяет снизить степень фрагментации индекса, что в свою очередь уменьшает количество операций реорганизации или перестройки индексов и повышает производительность запросов. Однако, при слишком малом значении fillfactor может возникнуть проблема низкой эффективности использования памяти, так как страницы индекса будут заполнены менее эффективно. Поэтому необходимо выбирать значение fillfactor, основываясь на конкретных требованиях к индексу и на особенностях хранения данных в конкретной таблице.

**PAD\_INDEX**

Параметр PAD\_INDEX в SQL Server позволяет указать, будет ли использоваться дополнительное выравнивание страниц в индексе.

По умолчанию SQL Server заполняет каждую страницу индекса полностью, без дополнительного выравнивания. Если параметр PAD\_INDEX установлен в ON, то на страницах индекса может быть оставлено место для дополнительных записей, что может увеличить производительность операций вставки данных.

Однако использование дополнительного выравнивания может занимать дополнительное место на диске и уменьшать эффективность кэширования в памяти. Поэтому при установке параметра PAD\_INDEX следует балансировать между производительностью и использованием ресурсов системы.