**Лабораторная работа № 9. Создание и применение индексов**

**Вариант 12**

Цель: изучить индексы, виды индексов; отработать навыки работы с индексами.

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.

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.

Первое задание: с помощью SSMS определить все индексы, которые имеются в БД UNIVER. Создать временную локальную таблицу. Заполнить ее данными (не менее 1000 строк). Разработать SELECT-запрос. Получить план запроса и определить его стоимость. Создать кластеризованный индекс, уменьшающий стоимость SELECT-запроса.

С помощью системной процедуры SP\_HELPINDEX можно получить перечень индексов, связанных с заданной таблицей – рисунок 1.1.

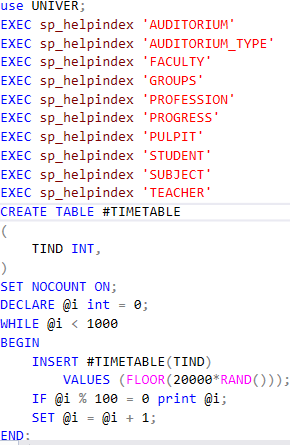


Рисунок 1.1 ­– Задание 1

План запроса позволяет посмотреть, предполагаемую стоимость данного запроса – рисунок 1.2.

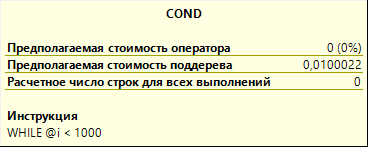


Рисунок 1.2 ­– Стоимость запроса без кластеризованного индекса

Однако при создании кластризованного индекса стоимость запроса уменьшается – рисунок 1.3.

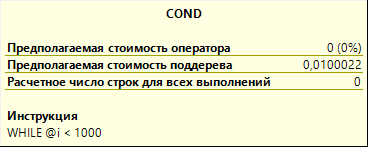


Рисунок 1.3 ­– Стоимость запроса с кластеризованным индексом

Второе задание похоже на первое, но вместо создания кластеризованного индекса, требуется создать некластеризованный неуникальный составной индекс, а также оценить процедуры поиска информации.

Некластеризованные индексы не влияют на физический порядок строк в таблице.

MSS допускает создавать индексы по нескольким столбцам – такие индексы называются составными. Этот индекс не применяется оптимизатором ни при фильтрации, ни при сортировке строк, но, если хотя бы одно из индексируемых значений зафиксировать (задать одно значение), то оптимизатор применит индекс.

Скрипт представлен на рисунке 1.4.

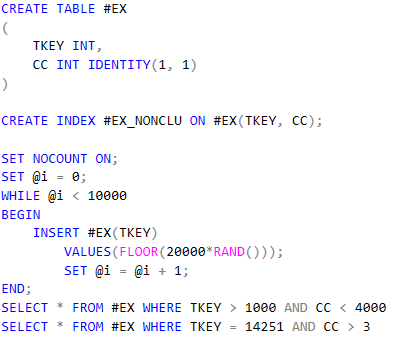


Рисунок 1.4 ­– Итоговый скрипт

В третьем задании нужно создать некластеризованный индекс покрытия.

Некластеризованный индекс покрытия запроса позволяет включить в состав индексной строки значения одного или нескольких неиндексируемых столбцов – рисунок 1.5.



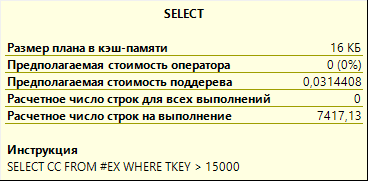


Рисунок 1.5 ­– Задание 3

В 4 задании надо создать некластеризованный фильтруемый индекс, который позволяет уменьшить стоимость SELECT-запроса. – рисунок 1.6.



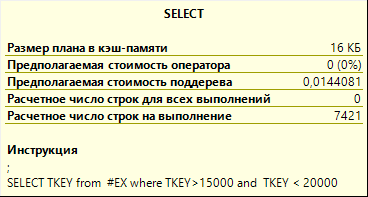


Рисунок 1.6 ­– Задание 4

Пятое задание – оценка уровня фрагментации индекса, выполнение процедуры реорганизации индекса и перестройки индекса.

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.

Для избавления от фрагментации индекса предусмотрены две специальные операции: реорганизация и перестройка индекса. Реорганизация (REORGANIZE) выполняется быстро, но после нее фрагментация будет убрана только на самом нижнем уровне. Операция перестройки (REBUILD) затрагивает все узлы дерева, поэтому после ее выполнения степень фрагментации равна нулю – рисунок 1.7.

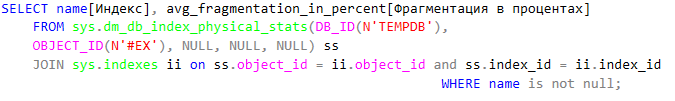


Рисунок 1.7 – Фрагментация индекса

В таблице «#EX» мы создали два индекса, и в силу малого количества данных фрагментация равна нулю. Если с помощью конструкции INSERT-SELECT вставить в таблицу >100000 строк, то степень фрагментации значительно увеличится. При 100000 она приближается к 98% - Рисунок 1.8.

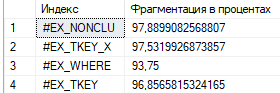


Рисунок 1.8 – Фрагментация индекса

Реорганизация и ребилд индекса для избавления от фрагментации – рисунок 1.9.



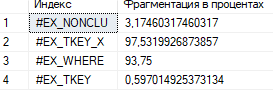


Рисунок 1.9 – Реорганизация и ребилд

В седьмом задании нужно было использовать FILFACTOR. 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 – рисунок 1.8.



Рисунок 1.8 – Использование FILFACTOR

**Вывод**

В ходе лабораторной работы были изучены индексы таблиц баз данных; Были реализованы кластеризованные индексы и некластеризованные: покрытия, фильтруемый, составной для оптимизации запросов и сокращения затрат на выполнение. Рассмотрена степень фрагментации индексов и способы её сокращения – операторы REBUILD и REORGANIZE, а также управление степенью фрагментации при помощи filfactor.

.