Задание к лабораторной работе 6

Преподаватель выбирает запрос одной из предыдущих лабораторных работ с целью проведения оптимизации его выполнения. Задает кол-во строк в каждой таблице и, по необходимости, другие параметры.

Порядок выполнения работы обучающимся:

1. Создать скрипт создания и заполнения таблиц тестовыми данными в объеме, указанном преподавателем. Строки таблиц заполняются в соответствующих циклах. При этом:
* столбцы наименований сущностей заполняются по шаблону <наименование сущности> <порядковый номер> (например, «Изделие 1», «Изделие 2» и т.д.);
* числовые столбцы (вес, цена и т. д.) – случайными числами из выбранного обучающимся диапазона;
* столбцы с датами – датами, начиная с выбранной обучающимся базовой даты (например, 1 января 2010 года) и далее следующими через случайное число дней.
1. Оформить запрос в виде отдельного скрипта, не забыв поставить в его начале команды DBCC DROPCLEANBUFFERS и SET STATISTICS IO ON.
2. Включить показ предполагаемого плана выполнения (Query/Display Estimated Execution Plan).
3. Выполнить скрипт запроса. Записать данные SET STATISTICS IO, выводимые на вкладке Messages.
4. Посмотреть рекомендации по созданию недостающих индексов на вкладке Execution Plan и добавить соответствующие команды в скрипт перед командой запроса. Обратите внимание, что иногда рекомендация относится к индексу для таблицы, для которой вы уже создали индекс. В таких случаях рекомендация обычно предлагает несколько поменять порядок или состав столбцов. Для этого удалите старый индекс и создайте новый по предлагаемому рекомендацией образцу.
Примечание:
В редких случаях рекомендации могут зациклиться: после создания нового индекса опять предлагается создать старый – тот, который вы ранее уже создавали, но удалили. Это означает, что для вашей таблицы и вашего запроса оказалось недостаточно одного индекса. В этом случае не удаляйте существующий индекс, а рекомендуемый новый индекс создайте под другим именем. Зацикливание должно прекратиться.
5. Повторить п.4.
6. Провести сравнительный анализ данных SET STATISTICS IO, полученных в пп.4 и 6: определить, какие параметры и во сколько раз уменьшились/увеличились, что дало добавление недостающих индексов, есть ли от этого польза или нет.
7. Сделать выводы: какие столбцы следует выбирать для индексов программисту, желающему ускорить выполнение своих запросов.
8. Информацию, полученную в пп. 4, 6, 7 и 8, поместить в конец скрипта запроса в виде многострочного комментария.

Скрипты оформляются в виде 2-х текстовых файлов .sql с именами:

*Создание.sql*

*Запрос.sql*

помещаются в архив и посылаются в электронном письме в виде вложения.

Формат темы письма и имени файла архива (без расширения):

*КодГруппы КодДисциплины Фамилия 06 #НомерВерсииИзДвухЦифр*

Пример:

*ИВТ-41-19 СУБД Иванов 06 #01*

Номер версии изначально равен 01 и увеличивается на единицу при каждой сдаче работы.

**Основные ошибки при выполнении лабораторной:**

* Плохое форматирование команд. Программист, читающий скрипт SQL, должен четко видеть, где начинается и заканчивается сама команда, где начинаются и заканчиваются основные предложения внутри команды (для команды SELECT это FROM, WHERE, GROUP BY, HAVING, ORDER BY) и т.д.
* При большом кол-ве добавляемых в цикле строк объем выводимых при этом сообщений легко может достигать миллионов строк. Поэтому следует использовать команду SET NOCOUNT ON, подавляющую вывод сообщения о добавлении каждой строки.
* Если того не требует алгоритм генерации, то порядок команд CREATE TABLE или их содержимое не должно отличаться от порядка или содержимого команд в ранее сданной лабораторной работе по созданию таблиц.
* Написание плохо сопровождаемого скрипта заполнения таблиц тестовыми данными. Например, кол-во строк в каждой таблице – величина непостоянная, сильно зависит от решаемых тестировщиком задач и поэтому может меняться от случая к случаю. Соответственно, становятся неуместны так называемые «магические числа», которые потом придется разыскивать по всему исходному коду и заменять на другие значения. Параметры генератора тестовых данных лучше оформить в виде отдельного смыслового блока с соответствующим комментарием, чтобы программист, пользующийся вашим генератором, легко мог их найти и поменять.
* Объявление всех вспомогательных (не относящихся к параметрам генерации) переменных в начале скрипта. Вспомогательная переменная должна объявляться как можно ближе к месту ее использования (но не внутри цикла).
* Следует с осторожностью подходить к генерации значений ID\_parent в таблицах, где данные организованы в виде дерева. Присвоение случайных значений с большой вероятностью приведет к тому, что появятся зацикленные сами на себя ветви, ветви, которые не ведут к корню дерева, или дерево вообще не будет иметь корней.
* Невнимательное чтение описания функции RAND() в части, касающейся верхней границы генерируемых случайных чисел.
* Использование ROUND() к результатам RAND(). В этом случае границы диапазона встречаются в 2 раза реже, чем остальные числа.
* Если столбец может содержать нецелые числа, то и случайные числа в него тоже должны записываться нецелые.
* Генерация случайных чисел непосредственно в условии проверки выхода из цикла. В этом случае для каждой итерации условие выхода будет новым и случайным, что, скорее всего, приведет к завершению цикла гораздо раньше ожидаемого числа итераций.