**Лабораторная работа № 11. Особенности использования транзакций**

**Вариант 12**

Цель: изучить транзакции, назначение и способы их совершения; отработать навыки работы с транзакциями.

Транзакция − это механизм базы данных, позволяющий таким образом объединять несколько операторов, изменяющих базу данных, чтобы при выполнении этой совокупности операторов они или все выполнились или все не выполнились.

Основные свойства транзакции: атомарность (операторы изменения БД, включенные в транзакцию, либо выполнятся все, либо не выполнится ни один); согласованность (транзакция должна фиксировать новое согласованное состояние БД); изолированность (отсутствие взаимного влияния параллельных транзакций на результаты их выполнения); долговечность (изменения в БД, выполненные и зафиксированные транзакцией, могут быть отменены только с помощью новой транзакции).

Режим неявной транзакции может быть включен для текущего соединения с сервером БД с помощью специальной инструкции: SET IMPLICIT\_TRANSACTIONS ON

Обратное переключение осуществляется с использованием ключевого слова OFF вместо ON.

Неявная транзакция начинается, если выполняется один из следующих операторов:

CREATE, DROP; ALTER TABLE; INSERT, DELETE, UPDATE, SELECT, TRUNCATE TABLE; OPEN, FETCH; GRANT (выдача разрешений), REVOKE (запрещение разрешений).

Неявная транзакция продолжается до тех пор, пока не будет выполнен оператор фиксации (COMMIT) или оператор отката (ROLLBACK) транзакции.

Первое задание ­ разработка сценария с использованием неявной транзакции – рисунок 1.1.

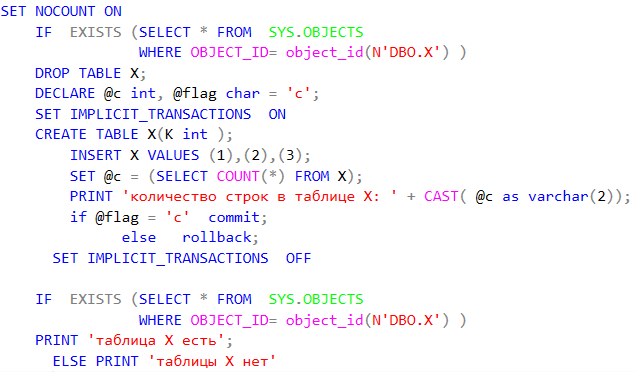


Рисунок 1.1 ­– Задание 1

Переключение в режим явной транзакции осуществляется с помощью оператора BEGIN TRANSACTION. Транзакцию должен завершать один из операторов: COMMIT TRAN или ROLLBACK TRAN. После завершения явной транзакции происходит возврат в исходный режим (автофиксации или неявной транзакции).

Системная функция @@TRANCOUNT возвращает уровень вложенности транзакции. (если значение больше нуля, то транзакция не завершена).

Встроенная функция PATINDEX определяет в строке позицию первого символа подстроки, заданную шаблоном. С помощью этой функции в тексте сообщения об ошибке отыскивается имя ограничения целостности.

Второе задание – разработка сценария, демонстрирующего свойство атомарности явной транзакции – рисунок 1.2.

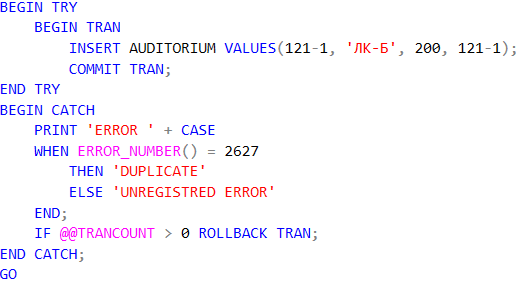


Рисунок 1.2 ­– Явная транзакция

Если транзакция состоит из нескольких независимых блоков операторов T-SQL, изменяющих базу данных, то может быть использован оператор SAVE TRANSACTION, формирующий контрольную точку транзакции.

Сценарий с SAVE TRAN – рисунок 1.3.

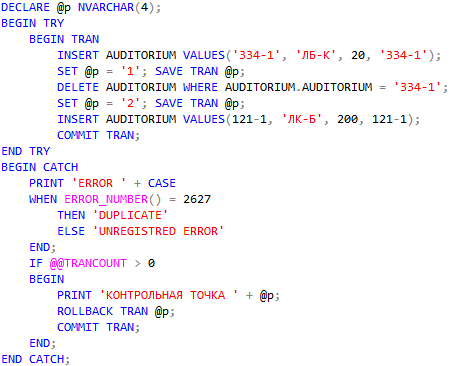


Рисунок 1.3 ­– SAVE TRAN

При параллельных транзакциях могут возникать три проблемы.

Неподтвержденное чтение. До момента t1 транзакцией B выполняются два оператора: INSERT и UPDATE. Эти операторы изменяют таблицы БД, но до момента времени t2 не фиксируют и не откатывают эти изменения. После момента t1 транзакция A считывает содержимое таблиц, измененных транзакцией B и «видит» измененные или добавленные строки. При этом изменения остаются до момента t2 в неподтвержденном состоянии, т. е. могут быть как зафиксированными, так и отмененными.

Неповторяющееся чтение. Одна транзакция читает данные несколько раз, а другая изменяет те же данные между двумя операциями чтения в первом процессе. По этой причине данные, прочитанные в различных операциях, будут разными.

Фантомное чтение. Две последовательные операции чтения могут получать различные значения, т. к. дополнительные строки, называемые фантом­ными, могут добавляться другими транзакциями.

Чтобы такие проблемы не возникали, определяются различные уровни изолированности: READ COMMITED, REPEATABLE READ, SERIALIZABLE и др.

Четвертое задание – Разработать два сценария A и B. Сценарий A представляет собой явную транзакцию с уровнем изолированности READ UNCOMMITED, сценарий B – явную транзакцию с уровнем изолированности READ COMMITED (по умолчанию). Сценарий представлен на рисунке 1.4.

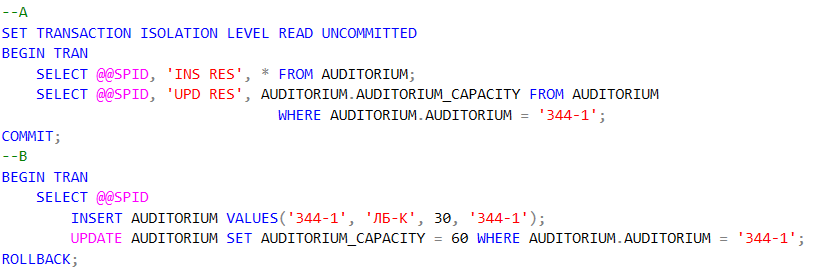


Рисунок 1.4 ­– Итоговый сценарий

Аналогичный запрос на с READ COMMITTED – рисунок 1.5.

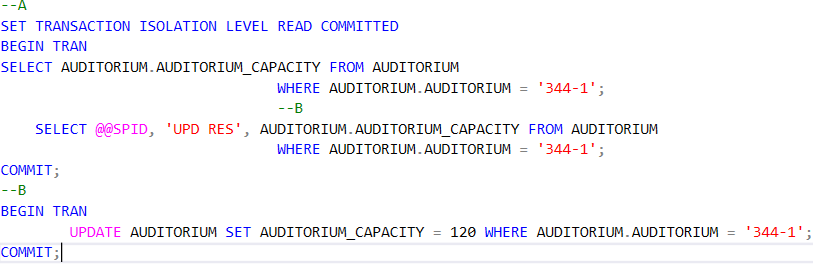


Рисунок 1.5 ­– Задание 5

Также только с фантомным чтением REPEATEBLE READ– рисунок 1.6.

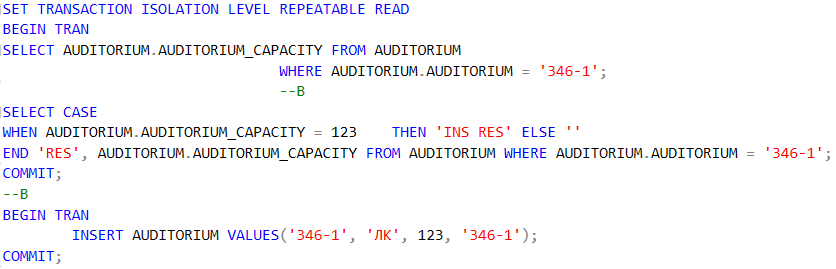


Рисунок 1.6 ­– Задание 6

Сценарий с SERIALIZABLE, который не допускает фантомное, неподтвержденное и неповторяющееся чтение – рисунок 1.7.

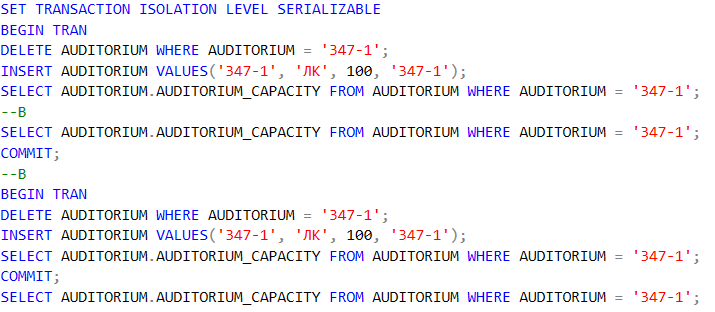


Рисунок 1.7 – Результат задания 7

Транзакция, выполняющаяся в рамках другой транзакции, называется *вложенной*.

При работе с вложенными транзакциями нужно учитывать следующее:

− оператор COMMIT вложенной транзакции действует только на внутренние операции вложенной транзакции;

− оператор ROLLBACK внешней транзакции отменяет зафиксированные операции внутренней транзакции;

− оператор ROLLBACK вложенной транзакции действует на операции внешней и внутренней транзакции, а также завершает обе транзакции;

− уровень вложенности транзакции можно определить с помощью системной функции @@TRANCOUT.

Задание 8 - сценарий, демонстрирующий свойства вложенных транзакций - Рисунок 1.8.

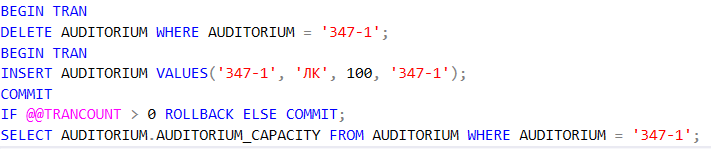


Рисунок 1.8 – Восьмое задание

**Вывод**

В ходе лабораторной работы были изучены транзакции, а также способы содания взаимодействия с ними; были получены навыки по созданию, использованию и взаимодействию с транзакциями; были изучены виды транзакций и их отличия.