**Транзакция** − это механизм базы данных, позволяющий таким образом объединять несколько операторов, изменяющих базу данных, чтобы 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.

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).

Транзакция начинается внутри TRY-блока и в случае успешного выполнения завершается оператором COMMIT. Если при выполнении возникла ошибка, то в CATCH-блоке формируется и выводится соответствующее сообщение, а затем выполняется откат (ROLLBACK).

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.

Изменения в таблице будут осуществлены только в случае отсутствия ошибок в операторах удаления и вставки.

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

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.

Для моделирования процесса в этом и следующих заданиях следует выполнять сценарии в три этапа: последовательно оба сценария до отметки t1; последовательно оба сценария c отметки t1 до отметки t2; последовательно оба сценария c отметки t2 и до конца.

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.

**Грязное чтение** – это когда данные, которые я прочитал, кто-то может откатить еще до того, как я завершу свою транзакцию.

**Неповторяющееся чтение** – это когда данные, которые я прочитал, кто-то может изменить еще до того, как я завершу свою транзакцию.

**Фантомное чтение** – когда ряд данных, которые я прочитал, кто-то может изменить до того, как я завершу свою транзакцию.

READ COMMITED – может читать только те данные, которые были закоммичены

Спасает от грязного чтения

REPEATABLE READ – пока транзакция не завершится, никто не может изменять и удалять строки

Спасает от грязного чтения и неповторяющегося чтения

SERIALIZABLE – самый жесткий, но самый тяжелый для БД и медленный для обработки запросов уровень. В этом случае транзакции идут друг за другом.

Избавляет от всех аномалий.

**Грязное чтение** – это когда данные, которые я прочитал, кто-то может откатить еще до того, как я завершу свою транзакцию.

**Неповторяющееся чтение** – это когда данные, которые я прочитал, кто-то может изменить еще до того, как я завершу свою транзакцию. (возможно добавление и изменение)

**Фантомное чтение** – когда ряд данных, которые я прочитал, кто-то может изменить до того, как я завершу свою транзакцию. (возможно удаление)

READ COMMITED – может читать только те данные, которые были закоммичены

Спасает от грязного чтения

REPEATABLE READ – пока транзакция не завершится, никто не может изменять и удалять строки

Спасает от грязного чтения и неповторяющегося чтения

SERIALIZABLE – самый жесткий, но самый тяжелый для БД и медленный для обработки запросов уровень. В этом случае транзакции идут друг за другом.

Избавляет от всех аномалий.