**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).

Транзакция в MS SQL - это последовательность операций, которые выполняются как единое целое. Она представляет собой логически связанный набор операций базы данных, который должен быть выполнен целиком или не выполнен вовсе. Транзакция может быть выполнена успешно, что означает, что все ее операции были успешно завершены и внесены в базу данных, или завершена с ошибкой, что означает, что все операции транзакции будут отменены и база данных будет возвращена к состоянию до начала транзакции.

Транзакции обычно используются для гарантии целостности данных в базе данных. Например, если при выполнении транзакции произошла ошибка, все изменения будут отменены, и база данных вернется к своему исходному состоянию. Это гарантирует, что данные в базе данных всегда будут находиться в согласованном состоянии и не будут испорчены из-за ошибок при обработке данных.

Транзакции могут быть запущены с помощью оператора BEGIN TRANSACTION и завершены с помощью операторов COMMIT TRANSACTION или ROLLBACK TRANSACTION. Оператор COMMIT TRANSACTION фиксирует все изменения, внесенные в базу данных в рамках транзакции, а оператор ROLLBACK TRANSACTION отменяет все изменения и возвращает базу данных к ее исходному состоянию.

**Режим неявной 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.

Режим неявной транзакции в MS SQL - это режим работы, при котором транзакции начинаются автоматически, без явного указания программистом. Этот режим включается по умолчанию для большинства операций в MS SQL Server, если явная транзакция не была начата вручную.

В режиме неявной транзакции, любые изменения данных в базе данных считаются частью текущей транзакции, которая автоматически завершается при выполнении COMMIT или ROLLBACK. Если транзакция завершается успешно, то изменения сохраняются в базе данных, если же возникает ошибка, то все изменения откатываются.

Режим неявной транзакции может быть удобным для простых операций, таких как выборка данных из таблицы, когда не требуется явно начинать и заканчивать транзакции. Однако, в более сложных сценариях, таких как многопоточные приложения или транзакции, работающие с несколькими таблицами, лучше использовать явные транзакции для более точного управления процессом.

**Явная транзакция**

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).

Явная транзакция в MS SQL - это блок операций, которые выполняются как единое целое, то есть все операции будут либо выполнены, либо не выполнены вообще. Это позволяет обеспечить целостность данных и избежать ошибок, если одна из операций не удалась.

Явная транзакция начинается с оператора BEGIN TRANSACTION и завершается командой COMMIT или ROLLBACK. Внутри транзакции можно выполнять различные операции с данными, например, добавление, изменение или удаление записей в таблицах.

Если оператор COMMIT выполнен успешно, то все изменения, сделанные в рамках транзакции, будут сохранены в базе данных. Если же возникла ошибка и транзакция не может быть завершена, то оператор ROLLBACK откатывает все изменения в базе данных до начала транзакции.

Явная транзакция может быть полезна в случаях, когда необходимо обеспечить целостность данных при выполнении нескольких операций в рамках одной 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.

**Контрольная точка транзакции**

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

Контрольная точка транзакции (checkpoint) в MS SQL Server - это механизм, который записывает все изменения, внесенные в буфер журнала транзакций, на диск. Таким образом, контрольная точка помогает уменьшить объем транзакционных журналов и ускоряет процесс восстановления после сбоя системы.

По умолчанию, контрольные точки в MS SQL Server происходят автоматически. Однако, их можно выполнить и вручную, используя команду CHECKPOINT. При выполнении контрольной точки, все изменения, которые еще не записаны на диск, будут записаны, и соответствующие буферы освобождаются для переиспользования.

Контрольные точки важны для обеспечения надежности и целостности данных, а также для управления производительностью базы данных. Однако, слишком частая запись контрольных точек может привести к ухудшению производительности базы данных, поэтому необходимо подбирать оптимальную частоту выполнения контрольных точек в зависимости от характера работы конкретной базы данных.

**Параллельные транзакции**

Параллельные транзакции в MS SQL - это механизм, который позволяет выполнять несколько транзакций одновременно, используя несколько потоков. Это повышает эффективность работы с базой данных, так как позволяет выполнять операции над данными более быстро, используя ресурсы процессора более эффективно.

В параллельных транзакциях каждая транзакция имеет свой уровень изоляции, что гарантирует, что изменения, внесенные одной транзакцией, не будут видны другой транзакцией до тех пор, пока изменения не будут зафиксированы (commit) или отменены (rollback).

Однако, параллельные транзакции могут привести к блокировкам и конфликтам, особенно при доступе к одним и тем же данным. Поэтому важно тщательно проектировать и оптимизировать запросы, чтобы минимизировать возможность блокировок и конфликтов при работе с параллельными транзакциями.

системная функция @@SPID возвращает системный идентификатор процесса, назначенный сервером текущему подключению.

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.

**уровнем изолированности READ COMMITED**

"Read committed" - это уровень изоляции транзакций в MS SQL Server, при котором каждая транзакция "видит" только изменения, которые были закоммичены другими транзакциями. То есть, изменения, которые находятся в процессе выполнения других транзакций, не видны.

При уровне изоляции "read committed" блокировки могут быть применены для предотвращения конфликтов параллельных транзакций. Например, если две транзакции пытаются изменить одну и ту же строку в таблице одновременно, то одна из них будет заблокирована до тех пор, пока другая не завершится.

Однако, уровень изоляции "read committed" также может приводить к проблеме "фантомных строк". Это происходит, когда транзакция видит изменения в одной строке таблицы, но не видит изменения в других строках, которые были сделаны другими транзакциями.

Кроме того, из-за того, что изменения, сделанные в других транзакциях, могут быть невидимы для текущей транзакции, могут возникать проблемы с повторяемостью чтения. Это означает, что при повторном выполнении запроса, результаты могут быть разными, что может привести к непредсказуемому поведению приложения.

Наконец, уровень изоляции "read committed" также может приводить к проблемам блокировки, особенно в сценариях с большим количеством параллельных транзакций и низким временем выполнения. Если одна транзакция заблокировала какой-то ресурс, другие транзакции могут быть вынуждены ждать дольше, что может снижать производительность системы в целом.

**уровнем изолированности READ UNCOMMITED**

Уровень изоляции READ UNCOMMITTED (недопустимое чтение) в SQL Server позволяет одной транзакции читать данные, которые были изменены другой транзакцией, но еще не зафиксированы (не подтверждены COMMIT). Это означает, что транзакция может видеть "грязные" данные, которые могут быть изменены или удалены другой транзакцией, прежде чем они будут зафиксированы.

Уровень изоляции READ UNCOMMITTED является самым низким уровнем изоляции и обеспечивает самый высокий уровень параллелизма в транзакционных приложениях, поскольку не блокирует доступ к данным другим транзакциям. Однако, это может привести к проблемам согласованности данных и непредсказуемому поведению приложений. Поэтому, использование уровня изоляции READ UNCOMMITTED должно быть ограничено и использоваться только в тех случаях, когда без этого невозможно обеспечить необходимый уровень производительности.

**уровнем изолированности REPEATABLE READ**

Уровень изоляции REPEATABLE READ означает, что любые чтения данных в течение текущей транзакции будут видеть те же самые данные, которые были прочитаны при первом чтении, независимо от того, были ли данные изменены другими транзакциями. Это достигается путем блокировки данных, которые читаются транзакцией, до ее завершения.

Этот уровень изоляции предотвращает аномалии, такие как "Dirty reads" и "Non-repeatable reads", которые могут возникнуть при использовании более низких уровней изоляции.

Однако при использовании этого уровня изоляции возможны проблемы с блокировками, которые могут привести к длительным задержкам в выполнении запросов и конкуренции за ресурсы базы данных.

**уровнем изолированности SERIALIZABLE**

Уровень изолированности SERIALIZABLE в MS SQL Server обеспечивает наивысший уровень изоляции транзакций, гарантируя, что транзакции выполняются, как если бы они были последовательными, то есть так, как если бы каждая транзакция выполнялась после завершения предыдущей. Это означает, что транзакции, выполняющиеся в режиме SERIALIZABLE, полностью изолированы друг от друга и не могут влиять друг на друга.

Для достижения этого уровня изоляции MS SQL Server использует блокировку таблиц, строк и диапазонов индексов, что может привести к возможным проблемам с производительностью при большой нагрузке на сервер. Кроме того, этот уровень изоляции может привести к проблемам с блокировкой при обновлении, удалении или вставке данных, если другие транзакции работают с тем же диапазоном данных.

**Вложенные транзакции**

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.

В MS SQL Server вложенная транзакция - это транзакция, запущенная внутри другой транзакции. Вложенные транзакции позволяют выполнять группу операций, которые нужно либо выполнить полностью, либо не выполнить вообще. Это может быть полезно, например, при написании сложных приложений, где нужно выполнять несколько операций в рамках одного блока кода.

Однако следует помнить, что вложенные транзакции могут повлиять на целостность данных. Если внешняя транзакция откатывается, все вложенные транзакции также будут отменены. Если же одна из вложенных транзакций завершается неудачно, это может привести к непредсказуемым последствиям для внешней транзакции.

Свойства вложенных транзакций в MS SQL Server:

- Вложенные транзакции можно использовать только в том случае, если установлен уровень совместимости 90 или более новый.

- Вложенная транзакция не может быть сохранена в базе данных, она может быть только сохранена во временном хранилище транзакций.

- Если вложенная транзакция откатывается, это не влияет на состояние внешней транзакции. Внешняя транзакция может продолжаться даже после отката вложенной транзакции.

- Если вложенная транзакция завершается неудачно, это может привести к автоматическому откату внешней транзакции.

Вложенная транзакция является транзакцией, которая запускается внутри другой транзакции. В MS SQL Server есть несколько свойств вложенных транзакций, которые важно учитывать:

- Уровень изоляции транзакции наследуется от родительской транзакции. Это означает, что если родительская транзакция работает на уровне изоляции SERIALIZABLE, то и вложенная транзакция будет работать на этом же уровне.

- Вложенная транзакция может быть зафиксирована только после того, как родительская транзакция будет зафиксирована. Если родительская транзакция откатится, то и все вложенные транзакции будут автоматически откатываться.

- Вложенная транзакция может откатиться независимо от родительской транзакции. Это происходит в случае, когда вложенная транзакция выполнена некорректно или обнаружена ошибка. В этом случае вложенная транзакция откатывается, а родительская транзакция может продолжать работать.

- Если вложенная транзакция зафиксирована, то изменения в этой транзакции станут видимы в родительской транзакции. Если же вложенная транзакция откатится, то изменения не будут видимы в родительской транзакции.

- Вложенные транзакции могут быть использованы для более гранулярного управления транзакциями в приложении. Например, если внутри одной транзакции выполняются несколько операций, то можно использовать вложенные транзакции для того, чтобы откатить только часть операций, а не всю транзакцию целиком.

- Вложенные транзакции могут использоваться только в случае, если используется режим управления транзакциями "Явный" (Explicit Transaction Management), когда транзакции создаются и управляются приложением. В режиме "Неявный" (Implicit Transaction Management) вложенные транзакции не поддерживаются.