ТЕХНИЧЕСКИЕ НАУКИ

УДК 004.65

Обеспечение требований ACID для высоконагруженных СУБД

И.П. Костенко, М.В. Ступина
Донской государственный технический университет (г. Ростов-на-Дону, Российская Федерация)

Аннотация. Рассматриваются ключевые принципы ACID, которые являются основой надежных и целостных баз данных. ACID — это аббревиатура, обозначающая Atomicity, Consistency, Isolation и Durability, то есть Атомарность, Согласованность, Изоляция и Устойчивость. Дано подробное описание каждого из этих принципов, проанализировано их влияние на надежность и целостность данных, отмечены ограничения, которые необходимо соблюдать для их обеспечения.

Ключевые слова: транзакция, СУБД, атомарность, согласованность, изолированность, устойчивость, ACID.

Ensure Acid Requirements for High-Load Dbms

Igor P Kostenko, Mariya V Stupina
Don State Technical University (Rostov-on-Don, Russian Federation)

Abstract. The article discusses the key principles of ACID, which are the basis of reliable and consistent databases. ACID is an acronym for Atomicity, Consistency, Isolation, and Durability. The article describes in detail each of these principles and their impact on the reliability and integrity of data, as well as what restrictions must be observed in order to implement them.

Keywords: transaction, DBMS, atomicity, consistency, isolation, durability, ACID.

Введение. Требования ACID были разработаны для обеспечения надежности и целостности данных в СУБД (СУБД — это совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных). Эти требования гарантируют, что операции с базой данных будут выполняться правильно и безопасно даже в случае сбоя или ошибки в системе.

Требования ACID являются стандартом в индустрии баз данных и важны для многих приложений, которые работают с критическими данными, такими как финансовые или медицинские записи. Понимание и соблюдение этих требований может помочь разработчикам создать более надежные и безопасные приложения для своих пользователей.

Реализация требований ACID имеет существенное влияние на производительность СУБД, особенно при работе с большими объемами данных. В связи с этим некоторые СУБД предлагают различные уровни гарантии выполнения требований ACID, которые могут быть настроены в зависимости от конкретных потребностей приложения.

Например, многие СУБД поддерживают несколько уровней изоляции транзакций, которые определяют, насколько изолированы транзакции друг от друга. Более высокий уровень изоляции может обеспечивать более строгое соблюдение требований ACID, но может также приводить к большей нагрузке на СУБД и снижению производительности.

Кроме того, при использовании распределенных СУБД могут возникать дополнительные проблемы в обеспечении согласованности и изолированности данных. В таких случаях могут использоваться различные технологии, такие как блокировки, репликация данных и транзакционные протоколы, чтобы обеспечить правильное функционирование системы.

Важно отметить, что ACID не являются единственными требованиями, которые могут быть установлены для СУБД. Например, некоторые приложения могут требовать поддержки транзакций с распределенными ресурсами (XA) или же возможности выполнения операций без блокировки (lock-free). В этих случаях разработчики могут использовать другие технологии и подходы для обеспечения нужных характеристик базы данных.

Важность использования ACID заключается в том, что эти требования позволяют создавать надежные и безопасные приложения, которые работают со сложными и критическими данными. Несоблюдение требований ACID может привести к неожиданным результатам, таким как потеря данных, некорректные результаты запросов и прочее. Это может вызвать серьезные последствия, включая финансовые потери и снижение доверия пользователей.

Требования ACID особенно важны для систем, которые работают с финансовыми, медицинскими или другими критическими данными, где ошибки могут привести к катастрофическим последствиям. Например, в банковских системах, системах здравоохранения, управления авиационным трафиком, а также в других системах, где надежность и точность данных являются критическими факторами, требования ACID являются обязательными [1].

Цель данной статьи — рассмотреть все требования ACID, выяснить, как они влияют на целостность данных, какие накладывает ограничения, какие проблемы могут возникнуть вследствие применения каждого из требований, а также установить связь между требованиями, чтобы понимать, как они взаимосвязаны.

Основная часть. Атомарность (Atomicity) является одним из ключевых требований ACID для обеспечения целостности и надежности транзакций в базах данных. Она означает, что каждая транзакция должна быть выполнена полностью или не выполнена вообще, то есть транзакция не может быть выполнена частично. Если в процессе выполнения транзакции происходит сбой, то все изменения, внесенные в базу данных, должны быть отменены, чтобы сохранить ее состояние до начала транзакции.

Атомарность является важным требованием для обеспечения целостности данных в многопользовательских системах, где несколько пользователей могут пытаться изменить одни и те же данные одновременно. Если транзакции не являются атомарными, то может возникнуть ситуация, когда одна транзакция изменяет данные, а другая транзакция использует неправильные данные, что приводит к некорректным результатам и нарушению целостности данных.

Чтобы обеспечить атомарность транзакций, СУБД используют механизмы транзакций, которые позволяют выполнить несколько операций как единое целое. Если при выполнении одной из операций возникает ошибка, то все операции, выполненные ранее в рамках транзакции, откатываются и содержание базы данных возвращается к состоянию до начала транзакции.

Атомарность также является важным требованием для обеспечения надежности системы в целом. Если транзакции не являются атомарными, то могут возникнуть ошибки, которые способны привести к некорректным результатам и нарушению целостности данных. Например, если транзакция изменяет баланс счета клиента, то необходимо убедиться, что все изменения, связанные с этой транзакцией, будут выполнены атомарно, чтобы избежать некорректного изменения баланса счета.

Кроме того, атомарность также помогает обеспечить консистентность данных в базе данных. Если транзакции не являются атомарными, то может возникнуть ситуация, когда данные находятся в неконсистентном состоянии, что приводит к ошибкам и неправильным результатам запросов.

С другой стороны, выполнение атомарных транзакций может привести к повышению нагрузки на систему, так как выполнение каждой транзакции требует дополнительных ресурсов. Поэтому при проектировании баз данных необходимо учитывать тот факт, что выполнение транзакций может существенно влиять на производительность системы, особенно если в ней много пользователей или большое количество транзакций.

Тем не менее преимущества атомарных транзакций гораздо важнее, чем небольшое увеличение нагрузки на систему. Они позволяют обеспечить надежность и целостность данных, что особенно ценно для критически важных систем: финансовых, управления запасами, здравоохранения и др.

В целом атомарность является важным требованием ACID для обеспечения надежности и целостности данных в СУБД. Она позволяет выполнить транзакции как единое целое, что помогает избежать ошибок и некорректных результатов запросов. При проектировании баз данных необходимо учитывать, что выполнение атомарных транзакций может существенно влиять на производительность системы, но в целом, преимущества выполнения атомарных транзакций гораздо важнее, чем небольшое увеличение нагрузки на систему [2].

Согласованность (Consistency) гарантирует, что данные в базе данных всегда находятся в состоянии, которое соответствует определенным правилам и ограничениям.

Это требование означает, что, если транзакция изменяет данные в базе данных, то она должна обеспечить их целостность, сохраняя при этом все ограничения и правила, заданные в базе данных. Если транзакция не может выполниться в полном объеме, то она должна быть отменена, чтобы сохранить согласованность базы данных.

Примером нарушения согласованности может быть ситуация, когда транзакция изменяет две связанные таблицы, но изменение в одной таблице происходит, а изменение в другой таблице не происходит из-за ошибки в программном коде. Это приведет к неконсистентности данных и возможности ошибочных результатов запросов.

Для обеспечения согласованности в базе данных используются ограничения целостности данных, которые задают правила, которым должны соответствовать данные в базе данных. Например, ограничения могут определять, что в каком-то столбце таблицы не могут быть пустые значения или что внешний ключ должен ссылаться на существующую запись в другой таблице.

При выполнении транзакции СУБД проверяет соответствие изменений существующим ограничениям целостности данных и откатывает транзакцию в случае несоответствия. Это гарантирует, что данные в базе всегда находятся в согласованном состоянии.

Нарушение согласованности данных может привести к трудно обнаруживаемым ошибкам в работе приложения, которые могут оказаться критическими, особенно для систем, где данные обрабатываются в реальном времени, например в финансовых или банковских системах.

В целом согласованность является важным требованием ACID для обеспечения надежности и целостности данных в СУБД. Она позволяет гарантировать, что данные в базе данных всегда находятся в состоянии, которое соответствует определенным правилам и ограничениям, что помогает избежать ошибок и недейственной работы приложения. Важно также отметить, что согласованность не является самостоятельным требованием, а тесно связана с другими требованиями ACID, такими как атомарность, изоляция и устойчивость.

Наряду с этим согласованность является важным требованием для многих приложений, где целостность данных имеет критическое значение. Например, в банковских системах, где любые ошибки или несоответствия могут привести к значительным финансовым потерям и негативным последствиям для клиентов.

Также для соблюдения требований согласованности в СУБД используются различные механизмы, например блокировки данных и транзакционная изоляция. Блокировки позволяют управлять доступом к данным и гарантировать, что одновременный доступ к данным не нарушает целостности базы данных. Транзакционная изоляция позволяет предотвращать ошибки в работе приложения и несоответствия, связанные с одновременным доступом к данным [2].

Изоляция (Isolation) — это одно из ключевых требований ACID, которое определяет, каким образом транзакции должны работать в СУБД. Изоляция гарантирует, что каждая транзакция работает в отдельной области памяти и не видит изменений, произведенных другими транзакциями, пока они не будут подтверждены. Это позволяет избежать конфликтов при одновременном доступе к данным, что, в свою очередь, обеспечивает целостность и надежность базы данных.

Одним из важных аспектов изоляции является концепция уровней изоляции, которые определяют, насколько транзакции должны быть изолированы друг от друга. Существует несколько уровней изоляции, каждый из которых определяет свой набор правил и ограничений, в том числе уровень изоляции чтения нефиксируемых данных (READ UNCOMMITTED), уровень изоляции чтения фиксируемых данных (READ COMMITTED), уровень изоляции повторяемого чтения (REPEATABLE READ) и уровень сериализации (SERIALIZABLE).

Уровень изоляции READ UNCOMMITTED позволяет транзакции читать данные, которые еще не были подтверждены другой транзакцией, что может привести к неконсистентности данных. Уровень READ COMMITTED гарантирует, что транзакции видят только те данные, которые были подтверждены другими транзакциями. Уровень REPEATABLE READ гарантирует, что транзакции видят только те данные, которые были считаны в начале транзакции, и не видят изменений, произведенных другими транзакциями. Уровень SERIALIZABLE гарантирует максимальную изоляцию и предотвращает любые конфликты при одновременном доступе к данным.

Важно отметить, что более высокий уровень изоляции может привести к повышению нагрузки на базу данных и снижению производительности, поскольку приходится использовать дополнительные ресурсы для управления транзакциями. Поэтому выбор уровня изоляции должен зависеть от требований конкретного приложения и балансировать между надежностью и производительностью.

В целом изоляция является важным требованием ACID, которое гарантирует, что транзакции работают в отдельной области памяти и не мешают друг другу при одновременном доступе к данным. Это позволяет обеспечить надежность и целостность базы данных, исключив возможность ошибок и противоречий в них. Однако необходимо учитывать, что изоляция может повлиять на производительность СУБД.

Кроме того, некоторые СУБД могут поддерживать дополнительные механизмы изоляции, такие как блокировки, чтение снимков и многоверсионность. Блокировки используются для закрытия доступа к ресурсам базы данных во время выполнения транзакции. Чтение снимков позволяет транзакции читать данные, не блокируя их для других транзакций. Многоверсионность позволяет транзакциям работать с разными версиями данных, что может повысить производительность СУБД.

Наконец, при разработке приложений, которые работают с базами данных, необходимо учитывать, что изоляция влияет на конкурентность доступа к данным, а также на производительность СУБД в целом. Поэтому

важно выбирать наиболее подходящий уровень изоляции в зависимости от конкретных требований приложения и настройки СУБД для оптимальной производительности [3].

Устойчивость (Durability) гарантирует, что результаты транзакции будут сохранены в базе данных и не будут потеряны в случае сбоев. Другими словами, если транзакция была подтверждена, то изменения, внесенные в базу данных в рамках этой транзакции, будут сохранены даже в случае отказа системы или сбоев в оборудовании.

Это достигается путем записи всех изменений в журнал транзакций или журнал изменений перед сохранением этих изменений в базу данных. Журнал транзакций представляет собой файл, который содержит информацию обо всех транзакциях, которые были выполнены в базе данных. Журнал изменений содержит информацию обо всех изменениях, внесенных в базу данных. Если система перезагружается или происходит сбой, журналы могут быть использованы для восстановления базы данных в точности до состояния, которое было до сбоя.

Также для обеспечения устойчивости могут применяться различные методы хранения данных, такие как репликация и резервное копирование. Репликация может быть использована для создания дополнительных копий базы данных на других серверах или устройствах хранения. Резервное копирование используется для создания резервных копий базы данных, которые могут быть использованы для восстановления базы данных в случае сбоя.

Важно понимать, что устойчивость не означает, что транзакции будут завершены успешно. Она гарантирует только то, что результаты транзакции будут сохранены в базе данных в случае сбоев. Таким образом, в случае ошибок или исключений транзакция может быть отменена, и изменения, внесенные в базу данных, будут также отменены.

Для обеспечения устойчивости также используются различные техники хранения данных. Например, используется кэширование данных для ускорения доступа к ним, однако при этом возникает риск потери данных, так как кэш может быть утерян при сбое системы. Для снижения риска потери данных применяются различные стратегии сброса кэша на диск, например сброс после каждой транзакции.

Важно отметить, что устойчивость не является абсолютной. Возможны ситуации, когда данные могут быть потеряны в результате сбоя оборудования или системы, несмотря на применение всех мер безопасности. Поэтому важно регулярно проверять наличие резервных копий и журналов транзакций, чтобы быть готовым к возможным проблемам [3].

Заключение. В заключение следует подчеркнуть важность требований ACID для обеспечения надежности и целостности данных в СУБД. ACID является фундаментальным понятием в области баз данных и используется практически во всех современных СУБД. Каждый из компонентов ACID — атомарность, согласованность, изоляция и устойчивость — играет важную роль в обеспечении надежности и целостности данных.

Атомарность гарантирует, что транзакция будет выполнена целиком или не выполнена вообще. Согласованность гарантирует, что данные будут находиться в согласованном состоянии после завершения транзакции. Изоляция гарантирует, что параллельное выполнение транзакций не повлияет на их результаты. Устойчивость гарантирует, что данные будут сохранены даже в случае сбоя системы. На основе проведенного анализа можно сделать вывод, что соблюдение требований ACID важно для обеспечения надежности и устойчивости работы систем управления базами данных.

Несмотря на то, что требования ACID обеспечивают высокую надежность данных, использование этих требований также может иметь негативные последствия. Например, параллельное выполнение транзакций с высокой степенью изоляции может привести к блокировкам, что может снизить производительность системы. Кроме того, ACID не подходит для систем с высокой скоростью потока данных, где производительность является более важной, чем надежность.

В итоге следует подчеркнуть, что применение требований ACID зависит от конкретных ситуации и приложения. При проектировании базы данных необходимо учитывать, как требования к надежности, так и требования к производительности. Использование механизмов, таких как индексы и оптимизация запросов, может помочь повысить производительность при сохранении требований к надежности данных [4].

Дальнейшие исследования в этой области могут быть направлены на изучение возможностей повышения производительности высоконагруженных СУБД при применении требований ACID, а также на разработку новых методов управления данными, которые могут быть более эффективными в условиях высокой нагрузки.

Библиографический список

1. Требования ACID на простом языке. *Habr*. URL: https://habr.com/ru/post/555920/ (дата обращения: 24.03.2023).

2. ACID. Что под капотом у транзакции. *Habr*. URL: https://habr.com/ru/company/simbirsoft/blog/572540/ (дата обращения: 24.03.2023).

3. Кучеренко Н.Ю. Проблема целостности данных в микросервисной архитектуре. *Столыпинский вестник*. 2022;4:1974–1983. URL: https://cyberleninka.ru/article/n/problema-tselostnosti-dannyh-v-mikroservisnoy-arhitekture (дата обращения: 24.03.2023).

4. Посконин А. Web-приложения и данные: проблемы абстракции и масштабируемости. *Труды Института системного программирования РАН*. 2012;23:159–172.

Об авторах:

Ступина Мария Валерьевна, доцент кафедры «Информационные технологии» Донского государственного технического университета (344003, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), кандидат педагогических наук, masamvs@bk.ru

Костенко Игорь Павлович, магистрант кафедры «Информационные технологии» Донского государственного технического университета (344003, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), igor.kostenko.1999@mail.ru

About the Authors:

Mariya V Stupina, associate professor of the Information Technologies Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, RF), Cand. Sci. (Pedagog.), masamvs@bk.ru

Igor P Kostenko, Master's degree student of the Information Technologies Department, Don State Technical University (1, Gagarin Sq., Rostov-on-Don, 344003, RF), igor.kostenko.1999@mail.ru