Эксплуатация БД. Технология оперативной обработки транзакций.

1. Управление транзакциями
2. Модель транзакции
3. Свойства транзакции
4. Журнализация
5. Проблемы многопользовательских систем
6. Блокировка
7. Тупиковая ситуация

Управление транзакциями

Транзакция — это неделимая с точки зрения воздействия на БД последовательность операторов манипулирования данными (чтения, удаления, вставки, модификации), рассматриваемая СУБД как единое целое.

Если транзакция успешно выполняется, то СУБД фиксирует изменения БД, произведенные этой транзакцией, во внешней памяти. Если транзакция не может закончиться корректно, она должна вернуть БД к исходному состоянию.

Управление транзакциями

Понятие транзакции имеет непосредственную связь с понятиями целостности и безопасности БД.

Последовательность операторов манипулирования данными, представляющая транзакцию, ***определяется разработчиком приложений***, исходя из наличия определенных процессов в данной предметной области.

Пример: поступивший в торговую фирму товар должен быть занесен во все необходимые отношения. данная операция должна:

- Увеличить товарные остатки на складах
- Увеличить долг предприятия поставщику товара
- Увеличить сумму НДС покупки

Управление транзакциями

Группирование операторов в транзакции сообщает СУБД, что вся эта группа должна быть выполнена как единое целое. Причем такое выполнение должно осуществляться автоматически.

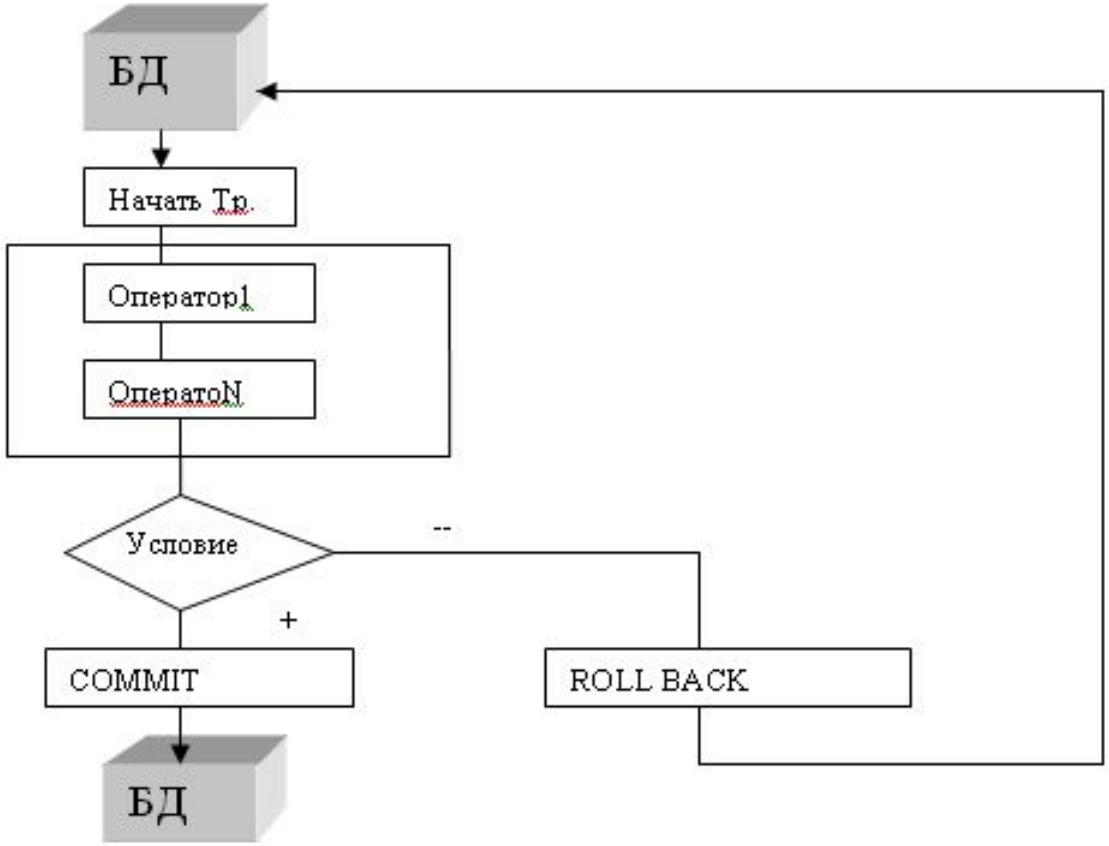
Корректное поддержание транзакций особенно важно в многопользовательских системах.

Модель транзакции

В стандарте SQL определена следующая модель транзакции:

1. транзакция начинается с первого SQL -оператора;
2. последующие SQL -операторы составляют тело транзакции;
3. оператор COMMIT выполняется в случае успешного завершения обработки информации, объединенной в транзакцию; его выполнение фиксирует изменения, внесенные в базу данных текущей транзакцией;
4. оператор ROLLBACK (откат транзакции) прерывает выполнение транзакции и осуществляет отмену изменений, проведенных в ходе выполнения транзакции.

Модель транзакции



Свойства транзакции

Любая из транзакций должна обладать четырьмя основными свойствами:

- **атомарности** — это свойство означает: либо транзакция выполняется полностью, либо не выполняется совсем;
- **согласованности** — это свойство гарантирует, что транзакция не нарушает согласованность данных;
- **изолированности** — это свойство обеспечивает такую изолированность одной транзакции от другой, что промежуточные результаты незавершенной транзакции не доступны другой транзакции;
- **долговечности** — это свойство гарантирует, что результаты зафиксированной транзакции не могут быть потеряны ни при каких обстоятельствах.

Журнализация

Возможность реализации транзакций предполагает способность системы сохранять промежуточные состояния базы данных, необходимые для отката транзакций.

Сохранение требуемых состояний осуществляется посредством специального механизма, который называется журналом транзакций.

Журнал транзакций — важнейшая часть СУБД — используется не только с целью обеспечения работы механизма транзакций. Он предназначен для поддержки одного из основных требований к СУБД: надежности хранения данных во внешней памяти.

Журнализация

Для восстановления БД нужно располагать некоторой дополнительной информацией, причем та часть данных, которая используется для восстановления, должна храниться особо надежно.

Наиболее распространенным методом поддержания такой избыточной информации является ведение журнала изменений БД.

Журнал — это особая часть БД, недоступная пользователям СУБД и поддерживаемая с особой тщательностью (иногда поддерживаются две копии журнала, располагаемые на разных физических дисках), в которую поступают записи обо всех изменениях основной части БД.

Журнализация

Под *надежностью хранения* понимается то, что СУБД должна быть в состоянии восстановить последнее согласованное состояние БД после любого аппаратного или программного сбоя.

Обычно рассматриваются два возможных вида аппаратных сбоев:

- так называемые ***мягкие сбои***, характеризуемые внезапной потерей содержимого оперативной памяти, наступающие в результате внезапной остановки работы компьютера (например, аварийное выключение питания);
- ***жесткие сбои***, характеризуемые потерей информации на носителях внешней памяти.

Журнализация

Примерами программных сбоев могут быть:

аварийное завершение работы СУБД (по причине ошибки в программе или в результате некоторого аппаратного сбоя);

аварийное завершение пользовательской программы, в результате чего некоторая транзакция остается незавершенной.

Первую ситуацию можно рассматривать как особый вид мягкого аппаратного сбоя; при возникновении последней требуется ликвидировать последствия только одной транзакции.

Журнализация

Если произошел мягкий сбой и содержимое буферов утрачено, для проведения восстановления базы данных необходимо иметь некоторое согласованное состояние журнала и базы данных во внешней памяти.

При мягком сбое во внешней памяти основной части БД могут наблюдаться нежелательные ситуации двух типов:

- *присутствие объектов*, модифицированных транзакциями, не закончившимися к моменту сбоя;
- *отсутствие объектов*, модифицированных транзакциями, которые к моменту сбоя успешно завершились, но по причине использования буферов оперативной памяти не были помещены во внешнюю память.

Журнализация

Во внешней памяти журнала должны гарантированно находиться записи, относящиеся к операциям модификации обоих видов объектов.

Целью процесса восстановления после мягкого сбоя является состояние внешней памяти основной части БД, которое возникло бы при фиксации во внешней памяти изменений всех завершившихся транзакций, и которое не содержало бы никаких следов незаконченных транзакций.

Журнализация

Для восстановления после мягкого сбоя необходимо:

- произвести откат незавершенных транзакций;
- повторно воспроизвести те операции завершенных транзакций, результаты которых не отображены во внешней памяти.

Для восстановления БД после жесткого сбоя журнала изменений базы данных явно недостаточно.

Основой восстановления последнего согласованного состояния базы данных после жесткого сбоя является журнал и архивная копия БД.

Архивная копия — это полная копия БД к моменту начала заполнения журнала.

Проблемы многопользовательских систем

В многопользовательских системах несколько одновременно работающих пользователей инициируют параллельные транзакции.

При параллельной обработке транзакций возникает ряд проблем. ***Для того чтобы получить корректно работающую транзакцию, недостаточно написать ряд правильно составленных операторов манипулирования данными.***

При обработке правильно составленных операторов манипулирования данными транзакций возникают ситуации, которые могут привести к получению неправильного результата из-за взаимных помех среди некоторых транзакций, вызванных бесконтрольным чередованием операций из двух правильных транзакций.

Проблемы многопользовательских систем

1. ***Проблема потерянных результатов обновления.***

Время	**Транзакция A1**	**Транзакция A2**
T1	Чтение P	—
T2	—	Чтение P
T3	Обновление P	—
T4	—	Обновление P

Проблемы многопользовательских систем

Результат операции обновления, выполненной транзакцией A1, будет утерян, поскольку в момент времени *t*4 она не будет учтена, и потому будет отменена операцией обновления, выполненной транзакцией A2.

Чтобы исключить такую ситуацию требуется, чтобы до завершения транзакции A1 никакая другая транзакция не могла изменять объект P.

Отсутствие потерянных изменений является минимальным требованием к СУБД в области синхронизации параллельно выполняемых транзакций

Проблемы многопользовательских систем

2. ***Проблема несогласованных данных***

Данная проблема появляется, если помощью некоторой транзакции осуществляется извлечение (обновление) некоторого объекта, который в данный момент обновляется другой транзакцией, но это обновление еще не закончено (если обновление не завершено, существует некоторая вероятность того, что оно не будет завершено никогда).

В таком случае в первой транзакции будут принимать участие данные, которые больше не существуют.

Проблемы многопользовательских систем

Проблема несогласованных данных:

Время	**Транзакция A1**	**Транзакция A2**
T1	—	—
T2	—	Чтение P
T3	Обновление P	—
T4	—	Чтение P

Проблемы многопользовательских систем

Транзакция A1 изменяет объект базы данных P. Параллельно с этим транзакция A2, читая объект P, видит, что он изменился, а значит нарушена целостность его транзакции. Произошло это потому, что транзакция A1 смогла изменить кортеж с данными, который прочитала транзакция A2.

Поскольку операция изменения еще не завершена, транзакция A2 видит несогласованные данные.

Чтобы избежать ситуации чтения несогласованных данных до завершения транзакции A1, изменившей объект P, никакая другая транзакция не должна читать объект P.

Проблемы многопользовательских систем

3. ***Проблема несовместимого анализа***

Возникает тогда, когда, например, транзакция A1 осуществляет вычисление некоторой статистической величины, скажем, среднего значения, а транзакция A2 выполняет обновление кортежа P3, который еще только будет использован транзакцией A1.

Причем транзакция A1 не зависит от транзакции A2, так как транзакция A2 выполнила все обновления до того, как транзакция A1 извлекла кортеж P3

Проблемы многопользовательских систем

Несовместимый анализ:

Время	Транзакция A1	Транзакция A2
T1	Чтение P1	—
T2	Чтение P2	Обновление P3
T3	Чтение P3	—
T4	Вывод результата	—

Проблемы многопользовательских систем

Для того, чтобы избежать подобных проблем, в СУБД должны использоваться какие-либо методы регулирования совместного выполнения транзакций.

Эти методы должны опираться на следующие правила:

- в ходе выполнения транзакции пользователь видит только согласованные данные;
- результаты параллельно выполняемых транзакций должны быть такими же, как если бы вначале выполнялась одна транзакция, а потом — вторая.

Проблемы многопользовательских систем

Реализация этих методов управления транзакциями в многопользовательской СУБД опирается на такие важные понятия, как:

- ***сериализация транзакций*** и
- ***сериальный план выполнения смеси транзакций***.

Блокировка

Под *сериализацией параллельно выполняющихся транзакций* понимается такой порядок планирования их работы, при котором суммарный эффект смеси транзакций эквивалентен эффекту их некоторого последовательного выполнения.

Сериальный план выполнения смеси транзакций — это такой план, который приводит к сериализации транзакций.

Блокировка

Наиболее распространенным механизмом сериализации транзакций, который используется коммерческими СУБД, являетс ***механизм блокировок***, или, иначе, механизм синхронизационных захватов, позволяющий разрешить описанные проблемы.

Данная методика предполагает блокировку в течение некоторой транзакции тех объектов, которые на протяжении этой транзакции должны оставаться неизменными.

Эффект блокировки состоит в том, чтобы заблокировать доступ к этому объекту со стороны других транзакций, а значит предотвратить непредсказуемое изменение этого объекта.

Блокировка

Различают два типа блокировок:

X-блокировка — блокировка без взаимного доступа (монопольная блокировка);

S-блокировка — с взаимным доступом.

Блокировка

Правила применения блокировок состоят в следующем:

- Если транзакция A блокирует кортеж P без возможности взаимного доступа (X-блокировка), то запрос другой транзакции B с блокировкой этого кортежа P будет отменен.

- Если транзакция A блокирует кортеж P с возможностью взаимного доступа, то запрос со стороны некоторой транзакции B на X-блокировку кортежа будет отвергнут, а запрос со стороны некоторой транзакции B на S-блокировку кортежа P будет принят.

Блокировка

На основе введения данных правил для избежания возникновения указанных выше проблем параллельной работы нескольких пользователей необходимо придерживаться следующей стратегии:

- транзакция, предназначенная для извлечения кортежа, должна наложить S-блокировку на этот кортеж;

- транзакция, предназначенная для обновления кортежа, должна наложить X-блокировку на этот кортеж.

Блокировка

Если запрашиваемая блокировка со стороны транзакции B отвергается из-за конфликта с некоторой другой блокировкой со стороны транзакции A, то транзакция B переходит в состояние ожидания.

Транзакция B будет находиться в состоянии ожидания до тех пор, пока не будет снята блокировка, заданная транзакцией A.

Блокировка

X-блокировки сохраняются вплоть до конца выполнения транзакции.

S-блокировки также обычно сохраняются вплоть до этого момента, однако при работе с ними есть свои особенности.

Проблемы параллельного выполнения двух транзакций:

- Проблема потери результатов обновления.
- Проблема незафиксированной зависимости.
- Проблема несовместимого анализа.

Блокировка

1. **Проблема потери результатов обновления.**
   С учетом применения протокола блокировки для чередующихся операций складывается следующая ситуация:

Время	Транзакция A1	Транзакция A2
T1	Чтение P Задание S-блокировки P	—
T2	—	Чтение P Задание S-блокировки P
T3	Обновление P Задание X - блокировки P	—
T4	Ожидание	Обновление P Задание X -блокировки P
T5	Ожидание	Ожидание

Блокировка

Чтение объекта P в момент времени tl транзакцией A1 вызывает наложение на этот объект S-блокировки.

Такую же блокировку объекта P устанавливает транзакция A2 в момент времени t2. Операция обновления для транзакции A1 в момент времени t3 не будет выполнена, поскольку она является неявным запросом с заданием X-блокировки для объекта P, а этот запрос вступает в конфликт с S-блокировкой, уже заданной транзакцией A2.

Таким образом, транзакция **A1** переходит в состояние ожидания. Транзакция A2 переходит в состояние ожидания в момент времени **t4.** Возникает конфликтная ситуация, которая получила название *тупик.*

Блокировка

2. **Проблема незафиксированной зависимости**.

   Они демонстрируют чередующееся выполнение операций согласно описанному выше протоколу блокировки:

Время	Транзакция A1	Транзакция A2
T1	Обновление P Задание X -блокировки P	—
t2	—	Чтение P Запрос на S-блокировку P
T3	—	Ожидание
T4	Снятие X-блокировки P	Чтение P Задание S-блокировки P

Блокировка

Операция для транзакции A2 в момент времени t2 не будет выполнена.

Дело в том, что она является неявным запросом с заданием S-блокировки для объекта P, а этот запрос вступает в конфликт с X-блокировкой, уже заданной транзакцией A1.

Таким образом, транзакция A2 переходит в состояние ожидания до тех пор, пока не будет прекращено выполнение транзакции A1. Тогда заданная транзакцией A1 блокировка будет снята и транзакция A2 может быть выполнена.

Ее результаты в любом случае не будут зависеть от незафиксированного обновления.

Блокировка

3. Проблема несовместимого анализа.

Ситуация, рассмотренная ранее, с учетом блокировок будет развиваться следующим образом:

Время	Транзакция A1	Транзакция A2
T1	Чтение P1 Задание S-блокировки P1	Чтение P3 Задание S-блокировки P3
T2	Чтение P2 Задание S-блокировки P2	Обновление P3 Задание X-блокировки P3
T3	Чтение P3 Запрос на S-блокировку P3	—
T4	Ожидание	Завершение транзакции Снятие X-блокировки P3
T5	Чтение P3 Задание S-блокировки P3	—

Блокировка

Операция чтения кортежа Р3 для транзакции А1 в момент времени t3 не будет выполнена, так как для ее реализации необходимо задать S-блокировку для этого кортежа, которая не может быть установлена поскольку на этот кортеж транзакцией А2 уже наложена X-блокировка.

Транзакция А1 переходит в состояние ожидания. После завершения транзакции А2 и снятия с кортежа Р3 X-блокировки транзакция А1 продолжит свою работу, однако полученный в итоге транзакции А1 результат будет неверен, так как транзакция А2 внесла свои коррективы в работу транзакции А1. Иными словами, транзакция А1 встретилась с несовместимым состоянием, которое блокировка в таком виде не смогла разрешить.

Тупиковая ситуация

Тупиковая ситуация возникает тогда, когда две или более транзакции одновременно находятся в состоянии ожидания, причем для продолжения работы каждая из транзакций ожидает прекращения выполнения другой транзакции.

Можно предположить существование более сложных ситуаций, например, количество заблокированных транзакций больше двух, но на практике никогда не встречаются тупиковые ситуации с участием более чем двух транзакций.

Тупиковая ситуация

Поскольку тупик сама транзакция обнаружить не может, его должна обнаружить и разрешить система.

Поиск выхода из тупиковой ситуации состоит *в выборе одной из заблокированных транзакций в качестве жертвы и отмене ее выполнения*. Таким образом, с нее снимается блокировка, а выполнение другой транзакции может быть возобновлено.

Критерием выбора жертвы является ***стоимость транзакции***, которая учитывает многие факторы (время выполнения, число накопленных захватов, приоритет), и в качестве жертвы выбирается ***самая дешевая транзакция***.

Тупиковая ситуация

Для выбранной транзакции-жертвы осуществляется откат, во время которого снимаются ее блокировки, и у других транзакций появляется возможность продолжить работу.

На практике не все системы в состоянии обнаружить тупиковую ситуацию.

Например, в некоторых из них используется *хронометраж выполнения транзакций*, и сообщение о возникновении тупиковой ситуации поступает, если транзакция не выполняется за некоторое предписанное заранее время.