Транзакции

Под транзакцией понимается неделимая с точки зрения воздействия на БД последовательность операторов манипулирования данными

Основное назначение транзакций в базе данных — переводить ее из одного согласованного состояния в другое

Транзакции также являются *единицами восстановления* данных после сбоев

Транзакции позволяют реализовывать *откладываемые ограничения целостности.*

Такие ограничения проверяются в конце транзакции, в случае их выполнения транзакция завершится оператором COMMIT.

Если ограничения будут нарушены, транзакция может быть откачена оператором ROLLBACK.

Откат транзакции

- В случае отката транзакции аннулируются все изменения, произведенные в ее рамках
  - Как это реализуется?
  - Кто должен выдавать команду ROLLBACK?
  - Автономные транзакции – зачем нужны?

Транзакции в многопользовательских системах

Транзакция рассматривается как последовательность элементарных атомарных операций

Элементарные операции различных транзакций могут выполняться в произвольной очередности

Определение 1

Набор из нескольких транзакций, элементарные операции которых чередуются друг с другом, называется ***смесью транзакций***

Определение 2

Последовательность, в которой выполняются элементарные операции заданного набора транзакций, называется ***графиком запуска*** набора транзакций

Если транзакции выполняются одна за другой, то график запуска называется ***последовательным***

Определение 3

Транзакции называются ***конкурирующими***, если они пересекаются по времени и обращаются к одним и тем же данным

Транзакция A

1. Читать значение из X
2. Уменьшить значение на 2
3. Записать значение в X

Транзакция B

1. Читать значение из X
2. Увеличить значение на 3
3. Записать значение в X

График 1

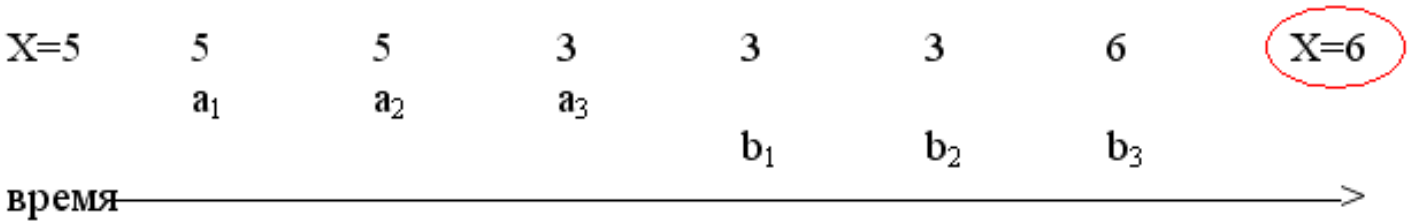


График 2

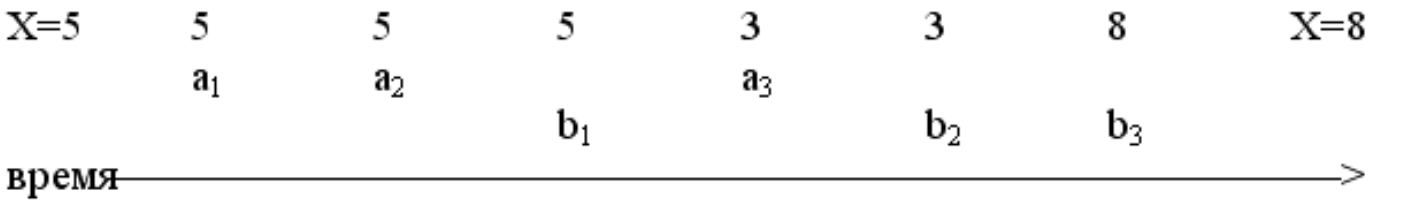
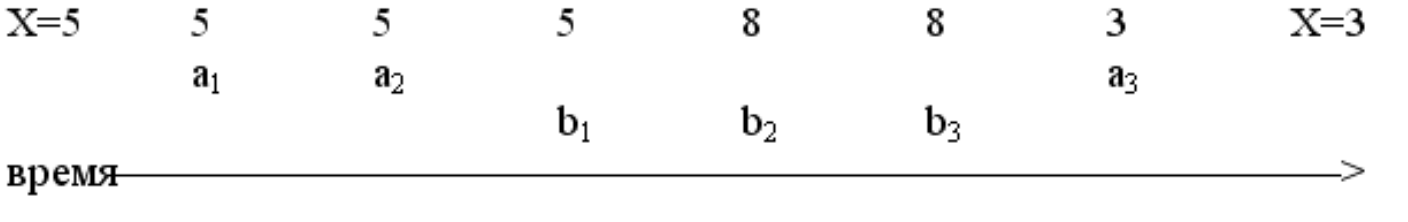


График 3



Определение 4

План (график) выполнения набора транзакций называется ***сериальным***, если результат совместного выполнения транзакций эквивалентен результату некоторого последовательного графика выполнения этих же транзакций.

Требования к транзакциям (ACID)

- **Неделимость (Atomicity).** Транзакция либо выполняется, либо не выполняется полностью
- **Согласованность (Consistency).** Транзакция переводит базу данных из одного согласованного состояния в другое
- **Изолированность (Isolation).** Результаты транзакции становятся доступны для других транзакций только после ее фиксации
- **Продолжительность (Durability**). После фиксации транзакции изменения становятся постоянными

Конфликты между транзакциями

Утраченные изменения (*конфликт типа W-W)*

Транзакция A	X	Транзакция B
$R(x_0)$	x_0	
		$R(x_0)$
$W(x_a)$	x_a	
COMMIT	x_a	
	x_b	$W(x_b)$
	x_0	ROLLBACK

Незафиксированные зависимости (*конфликт типа W-R)*

Транзакция A	X	Транзакция B
	x_0	$R(x_0)$
	x_0	$W(x_0)$
$R(x_0)$		
обработка (x_0)	x_0	
	x_0	ROLLBACK
COMMIT		

Несовместный анализ (*конфликт типа R-W)*

Случай 1
Неповторяемое считывание

Транзакция A	X	Транзакция B
$R(x_0)$	x_0	
		$R(x_0)$
	x_0	$W(x_0)$
		COMMIT
$R(x_0)$		
COMMIT		

Случай 2
Фиктивные элементы (фантомы)

Пишущая транзакция

```
insert into T(Id, val1)
          values(1000,1);
commit;
```

Читающая транзакция

```
select count(*) from T;



select count(*) from T;
commit;
```

Случай 3
Нарушение бизнес-логики

Пишущая транзакция	Читающая транзакция
	select sum(val) from T;
update T set val = val-1 where id=1;	
	select sum(val) from T;
update T set val = val+1 where id=2;	
commit;	

Изоляция транзакций

- СУБД должна иметь механизмы, для обеспечения требований ACID
- Эти механизмы могут обеспечивать разную степень (уровень) изоляции транзакций
- СУБД должна всегда обеспечивать недопустимость конфликта W-W. **Недопустимы** утраченные изменения

Стандартом SQL предусматриваются четыре уровня изоляции транзакций:

a) *READ UNCOMMITTED* – чтение незафиксированных данных;

b) *READ COMMITTED* – чтение зафиксированных данных;

c) *REPEATABLE READ* – повторяющееся чтение;

d) *SERIALIZABLE* –

Уровень изоляции	Проблема			
	потерянные обновления	«грязное» считывание	неповторяющееся чтение	Фантомы
READ UNCOMMITTED	нет	да	да	да
READ COMMITTED	нет	нет	да	да
REPEATABLE READ	нет	нет	нет	да
SERIALIZABLE	нет	нет	нет	нет

Механизмы сериализации

-обеспечить, чтобы конкурирующие транзакции выполнялись в разное время

- *Блокировки*
- *Временные метки*

- обеспечить, чтобы конкурирующие транзакции работали с разными версиями данных

Блокировки

- Монопольные (X) eXclusive locks
- Разделяемые (S) Shared locks

Правила

- Перед чтением накладывается S блокировка
- Перед изменением накладывается X блокировка
- Если блокировка отвергается, транзакция переходит в состояние ожидания
- Двухфазный протокол синхронизационных захватов (2PL) – сначала наложить слабую блокировку S, если удалось – пытаться наложить X блокировку

Совместимость блокировок

Транзакция A наложила	Транзакция B пытается наложить	
	S-блокировку	X-блокировку
S-блокировку	Да	НЕТ (Конфликт R-W)
X-блокировку	НЕТ (Конфликт W-R)	НЕТ (Конфликт W-W)

Решение проблем с помощью блокировок

- Потеря результатов – да, *но возможны тупиковые ситуации*
- Чтение «грязных данных» - да
- Неповторяемое считывание – да
- «Фантомы» - нет
- Несовместный анализ– да, *но возможны тупиковые ситуации*

Решение проблем тупиков

- Транзакция может задать максимально возможное время ожидания и выполнить откат, если оно превышено
- СУБД может обнаружить тупик и принести одну из транзакций в жертву

Что является объектом блокировки?

Логические

- Поле данных
- Кортеж
- Группа кортежей
- Таблица
- Группа таблиц
- База данных

Физические

- Блок
- Сектор
- Файл
- Группа файлов

Метод временных меток

- «Оптимистические» блокировки
- Используется, если вероятность возникновения конфликтов невелика
- Каждая транзакция получает временную метку (момент начала)
- Если транзакция блокирует объект, она помечает его своей меткой

Метод временных меток

- Прежде, чем заблокировать объект, транзакция анализирует его временную метку и конфликтность ситуации
- При отсутствии конфликта объект помечается наименьшей временной меткой и транзакции продолжают работу
- Если обнаружен конфликт, то более старая транзакция продолжит работу, пометив своим временем объект, а более молодая будет откачена и автоматически стартует вновь

Версии данных

- Отказ от блокировок для транзакций, читающих данные
- Транзакциям, читающим данные, представляется своя версия данных, существовавшая в тот момент, когда читающая транзакция начала свою работу
- Для транзакций, меняющих данные, по-прежнему используется механизм блокировок

Читающая транзакция ищет среди версий данных те, которые существовали на момент ее старта, т.е. помеченные системным номером меньшим, чем у нее самой

Это гарантирует, что транзакция не увидит изменений, произошедших в данных, начиная с того момента как она стартовала

Если транзакция меняет данные (фиксирует изменение), то она помечает эти данные своим системным номером, не уничтожая старые версии данных, помеченные другими системными номерами

При использовании версий данных постоянно накапливаются устаревшие версии строк

От СУБД требуется следить за этими версиями и, как только они становятся неактуальными, предпринимать меры по их удалению

Проблема тупиков по-прежнему существует