Транзакции

Транзакции

Предположим, у нас есть база данных, имеющая следующую структуру:

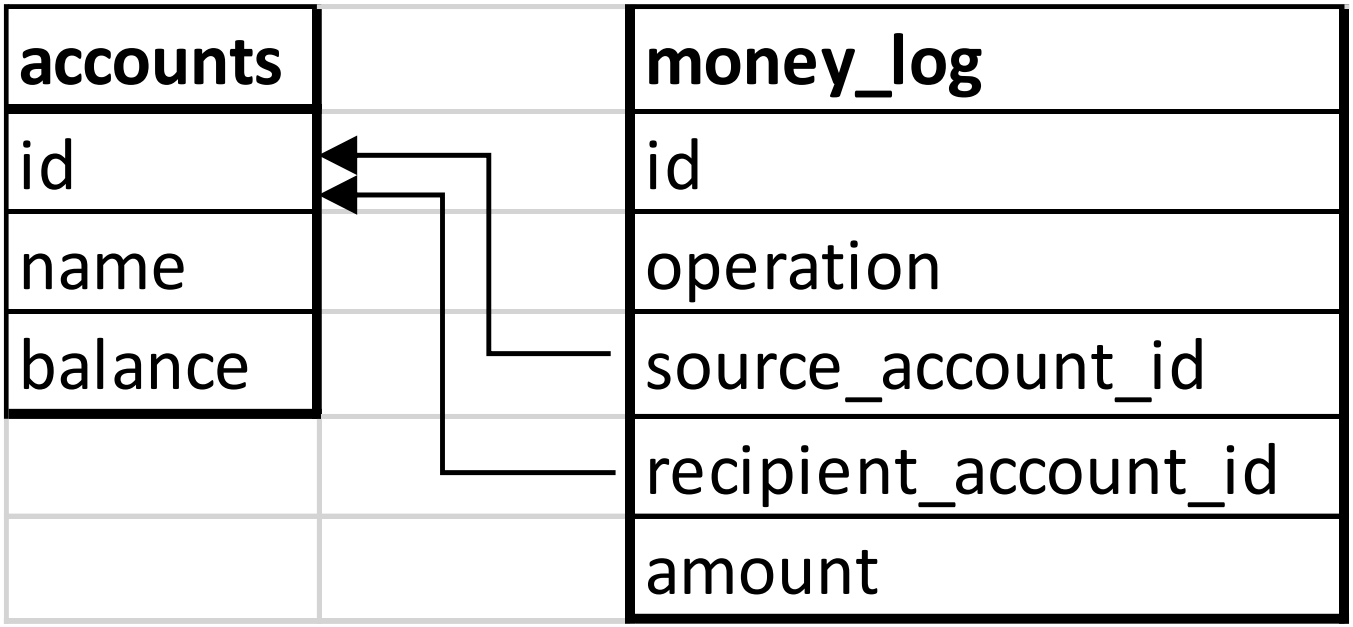

Таблица **money_log** предназначена для хранения данных о совершённых денежных операциях, где: **operation** – название операции, **source_account_id** – id счёта с которого были списаны деньги, **recipient_account_id** – id счёта на который были добавлены деньги, **amount** – сумма операции.

Понятие транзакции

Заполним базу:

ID	name	balance
101	Варя	5000
102	Куралай	5000
103	Фёдор	5000

Перед нами стоит задача:

Перевести деньги, в сумме 1000 единиц, со счёта Вари, на счёт Фёдора.

Алгоритм перевода денег

1. Уменьшить счёт Вари на 1000 единиц;
2. Увеличить счёт Фёдора на 1000 единиц;
3. Записать в таблицу **money_log** информацию о совершённой операции:

   operation – название операции,
   source_account_id – id счёта с которого были списаны деньги,
   recipient_account_id – id счёта на который были добавлены деньги,

   amount – сумма операции.

Понятие транзакции

Заполним базу:

ID	name	balance
101	Варя	4000
102	Куралай	5000
103	Фёдор	5000

Понятие транзакции

Заполним базу:

ID	name	balance
101	Варя	4000
102	Куралай	5000
103	Фёдор	6000

Понятие транзакции

Заполним базу:

ID	name	balance
101	Варя	4000
102	Куралай	5000
103	Фёдор	6000

id	operation	source_account_id	recipient_account_id	amount
1	transfer	101	103	1000

СБОЙ

Заполним базу:

ID	name	balance
101	Варя	4000
102	Куралай	5000
103	Фёдор	5000

Транзакция

- Транзакция - Способ объединения нескольких действий в один шаг изменения данных.

- Автофиксация – режим по умолчанию. Каждое действие фиксируется по своему завершению.

- Явный режим.

Транзакции

Явные - это транзакции, заданные явно.

Такие транзакции начинаются с ключевого слова:

- ✓ **BEGIN [WORK | TRANSACTION]**

Для успешного завершения работы, транзакции должны заканчиваться ключевым словом:

- ✓ **COMMIT [WORK | TRANSACTION]**
- ✓ для полного отката транзакции
- ✓ **ROLLBACK [WORK | TRANSACTION]**

Транзакции

- Транзакция не всегда заканчивается правильно. В PostgreSQL можно совершать только безошибочные транзакции.

- После возникновения ошибки инструкция больше не принимается, даже если она семантически и грамматически верна. Затем, даже если команда COMMIT выполнена, транзакция откатывается.


БЫСТРОЕ СОХРАНЕНИЕ
GTA V
trollix72
10:01
Сохранение

SAVEPOINT

- Внутри транзакции можно установить точку сохранения и, в случае ошибки, можно откатить транзакцию до точки восстановления (отменить только действия совершённые до точки восстановления).

- Точек восстановления, внутри одной транзакции может быть много.

SAVEPOINT

Дляустановки точки восстановления используется ключевое слово:

SAVEPOINT имя_точки_восстановления;

Для отката к определённое точке транзакции используется ключевое слово:

ROLLBACK TO имя_точки_восстановления;

ACID

ТРЕБОВАНИЯ К ТРАНЗАКЦИОННОЙ СИСТЕМЕ

ACID

A tomicity - Атомарность

C onsistency - Согласованность

I solation - Изолированность

D urability - Прочность

Atomicity (атомарность) — выражается в том, что транзакция должна быть выполнена в целом или не выполнена вовсе.

Consistency (согласованность) — гарантирует, что по мере выполнения транзакций, данные переходят из одного согласованного состояния в другое, то есть транзакция не может разрушить взаимной согласованности данных.

Isolation (изолированность) — локализация пользовательских процессов означает, что конкурирующие за доступ к БД транзакции физически обрабатываются последовательно, изолированно друг от друга, но для пользователей это выглядит, как будто они выполняются параллельно.

Durability (долговечность) — устойчивость к ошибкам — если транзакция завершена успешно, то те изменения в данных, которые были ею произведены, не могут быть потеряны ни при каких обстоятельствах.

В высоконагруженной СУБД, работающей в многопользовательском режиме, может возникать ситуация **КОНКУРЕНЦИИ** транзакций за данные.

КОНКУРЕНЦИЕЙ – в ТБД, называется ситуация, когда запросы из разных транзакций (от разных пользователей) пытаются получить доступ к одним и тем же данным. При этом разные транзакции, одновременно, могут пытаться прочитать, изменить или удалить одни и те же данные.

В процессе конкуренции транзакций за одни и те же данные в СУБД могут возникать следующие эффекты:

Название эффекта	Описание
«ГРЯЗНОЕ» ЧТЕНИЕ	Транзакция читает данные, записанные, но неподтверждённые незавершённой, конкурирующей транзакцией.
НЕПОВТОРЯЕМОЕ ЧТЕНИЕ	Транзакция повторно читает те же данные, что и раньше, и обнаруживает, что они были изменены другой транзакцией (которая завершилась после первого чтения).

Название эффекта	Описание
ФАНТОМНОЕ ЧТЕНИЕ	Одна транзакция в ходе своего выполнения несколько раз выбирает множество строк по одним и тем же критериям и получает разные выборки, так как конкурирующая транзакция добавила, удалила или изменила строки.
ПОТЕРЯННОЕ ОБНОВЛЕНИЕ (аномалия сериализации)	При одновременном изменении одного блока данных разными транзакциями теряются все изменения, кроме последнего.

Для управления эффектами конкуренции, транзакции могут иметь различные уровни изоляции друг от друга.

Наименование уровня	Описание
READ COMMITTED	каждый оператор SQL создаёт новый моментальный снимок базы данных, поэтому каждый оператор всегда будет видеть изменения, внесённые параллельными транзакциями в то же время, как только они будут зафиксированы.
REPEATABLE READ	первый оператор в транзакции создаёт моментальный снимок базы данных, который сохраняется для всей транзакции, поэтому все операторы видят одно и то же состояние базы данных.

Наименование уровня	Описание
READ UNCOMMITTED	Низший уровень изоляции. Чтение "грязных" (неподтверждённых) данных, от незавершённых транзакций.
SERIALIZABLE	Высший уровень изоляции. Каждая транзакция выполняется по очереди. Одна транзакция выполняется – остальные ждут в очереди.

Поведение при различных уровнях изолированности

Уровень изоляции	Фантомное чтение	Неповторяющееся чтение	«Грязное» чтение	Потерянное обновление
SERIALIZABLE	+	+	+	+
REPEATABLE READ	-	+	+	+
READ COMMITTED	-	-	+	+
READ UNCOMMITTED	-	-	-	+

«+» — предотвращает, «-» — не предотвращает.