Транзакции в SQL

Транзакция – последовательность операций над базой данных, которая воспринимается системой как единая операция.

Принципы АСИД (ACID):

- Атомарность (Atomicity) – транзакция выполняется как единая операция и либо результаты всех операций, ее составляющих, отражаются в базе данных, либо они там гарантировано отсутствуют
- Согласованность (Consistency) – транзакция переводит базу данных из одного согласованного состояния в другое согласованное состояние, она успешно завершается в том и только том случае, когда действия ее операций не нарушают целостность БД
- Изоляция (Isolation) – две транзакции, выполняющиеся одновременно (параллельно или квазипараллельно), не должны никоим образом действовать друг на друга (результаты одной транзакции не должны быть видны никакой другой, до тех пор, пока первая транзакция не завершится успешно)
- Долговечность (Durability) – после успешного завершения транзакции все внесенные ею изменения должны быть гарантированно сохранены в БД даже в случае аппаратных или программных сбоев

Инициация транзакций

В SQL транзакции могут образовываться:

- явно с использованием оператора START TRANSACTION
- неявно, когда выполняется оператор, для которого требуется контекст транзакции, а этого контекста не существует

Большинство операторов SQL (за исключением ряда административных операторов) требуют наличие контекста транзакции.

Явная инициация транзакции: START TRANSACTION mode_list

Характеристики транзакции:

1. Режим доступа: READ ONLY или READ WRITE
2. Уровень изоляции: READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ или SERIALIZABLE
3. Размер области диагностики (количество сохраняемых диагностических сообщений): DIAGNOSTIC SIZE value (значение по умолчанию определяется в реализации)

Для характеристик транзакций, инициируемых неявно, используются либо значения по умолчанию, либо значения, определяемые оператором SET TRANSACTION mode_list

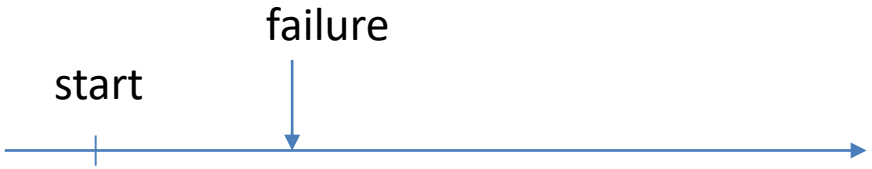
(данный оператор недопустимо выполнять в контексте активной транзакции).

Завершение транзакций

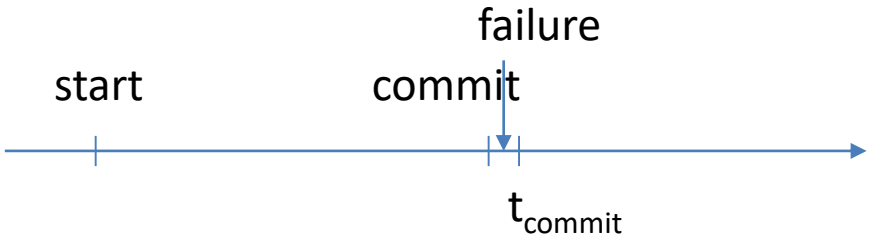
Для завершения стартовавшей транзакции должен быть явно использован один из двух операторов:
COMMIT [WORK] [AND [NO] CHAIN] – фиксация транзакции (завершение с фиксацией результатов в БД)
ROLLBACK [WORK] [AND [NO] CHAIN] – откат транзакции (завершение с возвратом к предыдущему состоянию БД)
Если присутствует раздел AND CHAIN, то по завершении текущей транзакции образуется новая, наследующая все характеристики завершенной.
Оператор COMMIT считается безусловно выполненным только тогда, когда это подтверждает СУБД после выполнения всех действий, необходимых для фиксации результата транзакции. Это делается с целью защиты от сбоев, произошедших в момент выполнения COMMIT.

Поддержка долговечности

Сбой после фиксации транзакции: Транзакция долговечна (состояние БД восстанавливается по журналу или по журналу и резервной копии)



Сбой во время выполнения транзакции: Возврат к состоянию до начала выполнения транзакции



Сбой в процессе фиксации транзакции: транзакция считается незафиксированной и все ее изменения автоматически удаляются из базы данных при ее восстановлении

Учебный пример

```
CREATE TABLE EMPLOYEE (
…
DEPT_ID INTEGER DEFAULT
NULL REFERENCES
DEPARTMENT ON DELETE SET
NULL,
…
);
```

```
CREATE TABLE DEPARTMENT (
        DEPT_ID INTEGER PRIMARY KEY,
        DEPT_EMP_NUM INTEGER NOT NULL
CHECK( VALUE <= 100 ),
        CHECK ( DEPT_EMP_NUM = ( SELECT
COUNT(*) FROM EMPLOYEE WHERE DEPT_ID =
EMPLOYEE.DEPT_ID )),
…
);
```

Операторы модификации таблицы EMPLOYEE вида:
INSERT INTO EMPLOYEE (…, DEPT_ID, …) VALUES ROW (…, X, …);
UPDATE EMPLOYEE SET DEPT_ID = X WHERE …;
DELETE FROM EMPLOYEE WHERE …;
где X – значение DEPT_ID, отличное от NULL, а во множество удаляемых строк включается хотя бы одна строка, в которой значение столбца DEPT_ID отличается от NULL, нарушают выделенное ограничение целостности

???

Транзакции и ограничения целостности

В контексте выполняемой транзакции каждое ограничение целостности может находиться в одном из двух режимов:
1. Режим немедленной проверки (immediate) – ограничение проверяется сразу после выполнения любой операции, изменяющей состояние БД. Операция отвергается, если она нарушает хотя бы одно ограничение целостности, находящееся в данном режиме.
2. Режим отложенной проверки (deferred) – ограничение проверяется при выполнении операции COMMIT. Транзакция откатывается (COMMIT трактуется как ROLLBACK), если ее операции нарушили хотя бы одно отложенно проверяемое ограничение целостности.

Для указания режима проверки к определению ограничения целостности добавляется следующая синтаксическая конструкция (в качестве заключительной конструкции определения ограничения):
INITIALLY { DEFERRED | IMMEDIATE } [[NOT] DEFERRABLE]
По умолчанию предполагается INITIALLY IMMEDIATE NOT DEFERRABLE
Комбинация INITIALLY DEFERRED NOT DEFERRABLE недопустима.

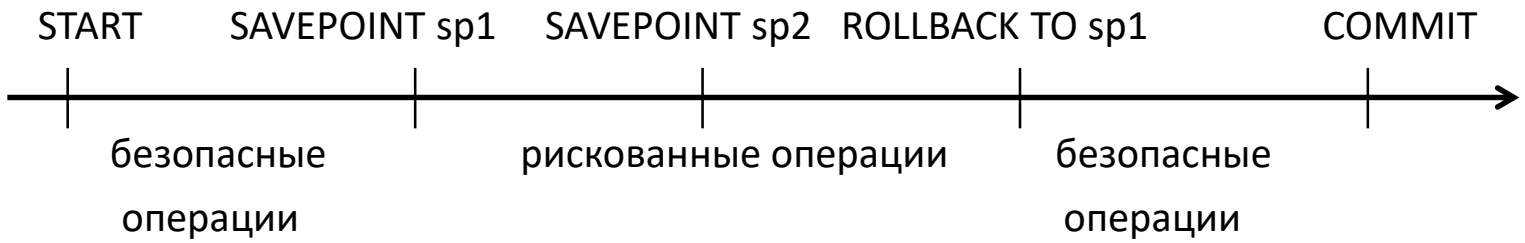
Изменение режима проверки ограничений

Режим проверки ограничений, которые специфицированы с указанием ключевого слова DEFERRABLE, можно изменить в ходе выполнения транзакции с помощью оператора:
SET CONSTRAINTS { ALL | constraint_name_list } { DEFERRED | IMMEDIATE }
Если в списке имен будет явно указано хотя бы одно ограничение, определенное как NOT DEFERRABLE, оператор будет отвергнут.

Ограничения, находящиеся в режиме отложенной проверки, будут проверены сразу при переводе их в режим немедленной проверки (неявно – при выполнении COMMIT или явно – при выполнении SET CONSTRAINTS … IMMEDIATE).

Точки сохранения в транзакциях

Точка сохранения – пометка в последовательности операций транзакции, которую можно использовать для ее частичного отката с сохранением жизнеспособности и результатов операций, выполненных до точки сохранения.
Установление точки сохранения: SAVEPOINT savepoint_name
Удаление точки сохранения: RELEASE SAVEPOINT savepoint_name
Откат до точки сохранения: ROLLBACK TO SAVEPOINT savepoint_name
В одной транзакции возможно установить несколько последовательных точек сохранения. При откате до некоторой точки сохранения неявно выполняется RELEASE для всех точек, определенных в транзакции после данной.

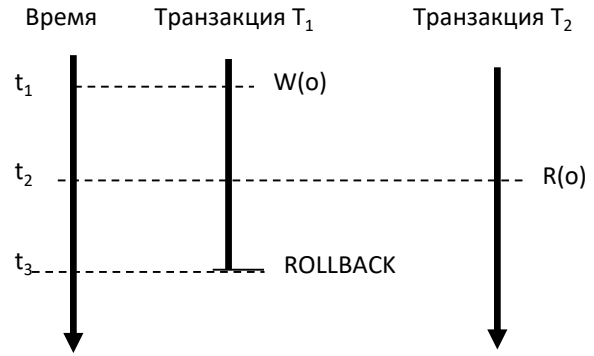


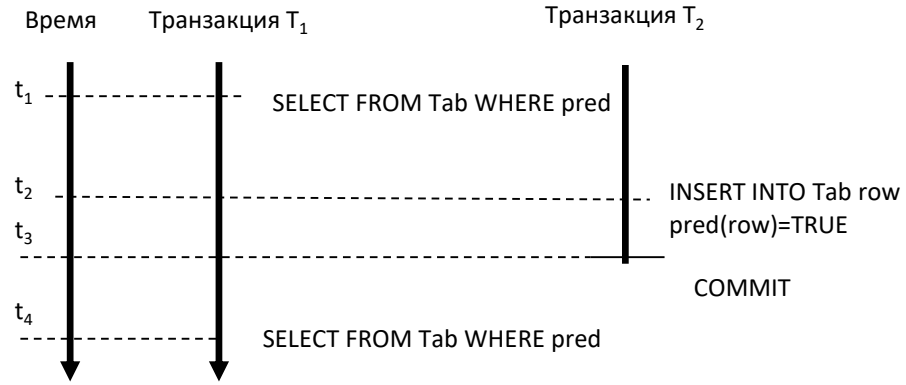
Точки сохранения не нарушают принцип атомарности, поскольку извне транзакции она все равно рассматривается как единая операция (точки сохранения не видны).

Феномены выполнения транзакций

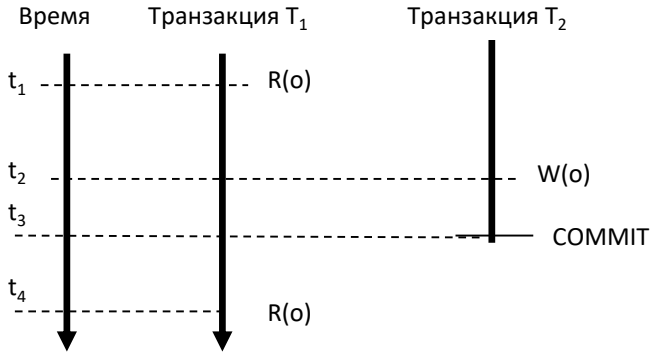
При одновременном (параллельном или квазипараллельном) выполнении нескольких транзакций могут возникнуть следующие феномены.

Грязное чтение (некорректные данные в транзакции T_2)





Неповторяющееся чтение (несовпадение результатов операций чтения одного и того же объекта в транзакции T_1)



Фантомы (появление фантомных строк в таблице Tab при повторном запросе в транзакции T_1)

Уровни изоляции транзакций в SQL

На уровне изоляции READ UNCOMMITTED могут проявляться все три феномена. Рекомендуется его использовать только в тех транзакциях, где точные данные не обязательны (статистическая обработка). Одновременное задание READ UNCOMMITTED и режима доступа READ WRITE не допускается. Каждый следующий уровень устраняет проявление одного из феноменов.

Уровень изоляции	Описание	Грязное чтение	Неповт. чтение	Фантомы
READ UNCOMMITTED	Возможность видеть изменения, произ-водимые еще не зафиксированными параллельными транзакциями	+	+	+
READ COMMITTED	Допускаются изменения объектов, прочитанных другими одновременно выполняющимися транзакциями	-	+	+
REPEATABLE READ	Допускается добавление строк к таблицам, которые удовлетворяют условиям выборки, выполненной в других параллельных транзакциях	-	-	+
SERIALIZABLE	Предельная изолированность одно-временно выполняющихся транзакций	-	-	-

Поддержка авторизации доступа к данным в SQL

Метод авторизации доступа, используемый в SQL относится к мандатным видам защиты данных: с каждым зарегистрированным в СУБД пользователем или ролью (субъектом) и каждым защищаемым объектом БД связывается мандат (или привилегия), определяющий действия, которые может выполнять данный субъект над данным объектом.

В SQL поддерживается принцип сокрытия информации об объектах, содержащихся в схеме БД, от субъектов, которые лишены доступа к этим объектам. Если некоторый субъект не обладает привилегиями доступа к некоторой таблице, то при попытке выполнить какое-либо действие над ней он получит такое же диагностическое сообщение, как если бы данная таблица не существовала.

Создатель объекта базы данных автоматически становится владельцем этого объекта, который обладает полным набором привилегий для выполнения действий над объектом, в том числе привилегией на передачу всех или части своих привилегий другим субъектам.

Привилегии доступа к объектам

Действие	Привилегия	Объекты
Просмотр	SELECT	Таблицы, столбцы, хранимые процедуры
Вставка	INSERT	Таблицы, столбцы
Модификация	UPDATE	Таблицы, столбцы
Удаление	DELETE	Таблицы
Ссылка	REFERENCES	Таблицы, столбцы
Использование	USAGE	Домены, типы и проч. определения
Инициирование	TRIGGER	Таблицы
Выполнение	EXECUTE	Хранимые процедуры
Типизация	UNDER	Определяемые пользователем типы
Передача	GRANT/ADMIN	Привилегии/роли

Привилегии над представлениями основываются на привилегиях по отношению к базовым таблицам данных представлений.

Пользователи и роли

Привилегии доступа к объектам предоставляются пользователям, а также ролям, выполнение которых, в свою очередь, может предоставляться пользователям. С каждым пользователем и каждой ролью связывается уникальный идентификатор авторизации (authID).

Роль – динамически образуемая группа пользователей СУБД, каждый из которых обладает привилегией на исполнение данной роли, а также всеми привилегиями данной роли для доступа к объектам БД. Роли упрощают построение и администрирование системы авторизации доступа.

В стандарте SQL не определяются средства создания и ликвидации идентификаторов пользователей. Для создания и ликвидации ролей поддерживаются операторы CREATE ROLE/DROP ROLE. В стандарте также поддерживается концепция идентификатора псевдопользователя PUBLIC, который соответствует любому пользователю, зарегистрированному в СУБД. Пользователю PUBLIC могут предоставляться привилегии доступа к объектам базы данных, как и любому другому пользователю, при этом они будут распространяться автоматически и на всех вновь создаваемых пользователей.

Создание и ликвидация роли

Создание роли:
CREATE ROLE role_name [WITH ADMIN { CURRENT_USER | CURRENT_ROLE }]
Имя роли должно отличаться от любого идентификатора авторизации (authID), уже определенного в СУБД. При наличии раздела WITH ADMIN привилегию на исполнение данной роли вместе с правом передачи привилегии получает либо текущий пользователь, либо текущая роль SQL сессии. По умолчанию привилегия на исполнение создаваемой роли передается текущему пользователю. Если SQL сессия не имеет пользователя – то текущей роли. Привилегии, требуемые для выполнения CREATE ROLE определяются в реализациях (как правило, выполнение разрешается только администраторам).

Ликвидация роли:
DROP ROLE role_name
Для выполнения DROP ROLE текущий authID SQL сессии (пользователь или роль) должен являться владельцем данной роли. При ликвидации роли автоматически ликвидируются привилегии на ее исполнение у всех пользователей и ролей, которым эта привилегия ранее передавалась.

Передача привилегий

GRANT { ALL PRIVILEGES | privilege_list } ON object_name TO { PUBLIC | authID_list } [WITH GRANT OPTION] [GRANTED BY { CURRENT_USER | CURRENT_ROLE }]

Привилегии передаются от текущего authID сессии (пользователя или роли) к указанным в списке authID (пользователям или ролям). Для этого он должен обладать привилегией на передачу всех или части привилегий из списка, указанного в операторе GRANT.
Если текущий authID обладает правом на передачу только части привилегий из этого списка, то передается именно эта часть и выдается предупреждение, а если он не имеет прав на передачу ни одной из перечисленных привилегий, то фиксируется ошибка.
Если указан раздел WITH GRANT OPTION, то привилегии из списка передаются с правом дальнейшей передачи.
Раздел GRANTED BY позволяет явно указать, от какого authID (текущего пользователя или текущей роли) передаются привилегии.
Избыточная дублирующая передача привилегий от имени одного и того же authID другому (тому же самому) authID игнорируется.

Передача ролей

GRANT role_name_list TO { PUBLIC | authID_list } [WITH ADMIN OPTION] [GRANTED BY { CURRENT_USER | CURRENT_ROLE }]

Оператор позволяет передавать произвольное число ролей произвольному числу пользователей или ролей.
Если текущий authID обладает правом на передачу только части ролей из этого списка, то передается именно эта часть и выдается предупреждение, а если он не имеет прав на передачу ни одной из перечисленных ролей, то фиксируется ошибка.
Если указан раздел WITH ADMIN OPTION, то привилегии на исполнение ролей из списка передаются с правом дальнейшей передачи.
Раздел GRANTED BY позволяет явно указать, от какого authID (текущего пользователя или текущей роли) передаются привилегии на исполнение ролей.

Аннулирование привилегий

REVOKE [GRANT OPTION FOR] privilege_list ON object_name FROM { PUBLIC | authID_list } [GRANTED BY { CURRENT_USER | CURRENT_ROLE }] { RESTRICT | CASCADE }

RESTRICT – действие оператора отвергается, если хотя бы одна из указанных в списке привилегий была передана другому тем authID, у которого привилегия должна быть аннулирована (сочетание с GRANT OPTION FOR означает, что аннулируется привилегия на передачу привилегий из указанного списка, сама привилегия при этом остается).
CASCADE – указанные привилегии аннулируются у всех authID, прямо или косвенно (через промежуточные authID) получивших привилегии от текущего authID, выполняющего данную операцию.
Для успешного выполнения REVOKE необходимо, чтобы текущий authID обладал всеми или частью привилегий из указанного списка. В последнем случае аннулируется именно эта часть (с выдачей предупреждения).

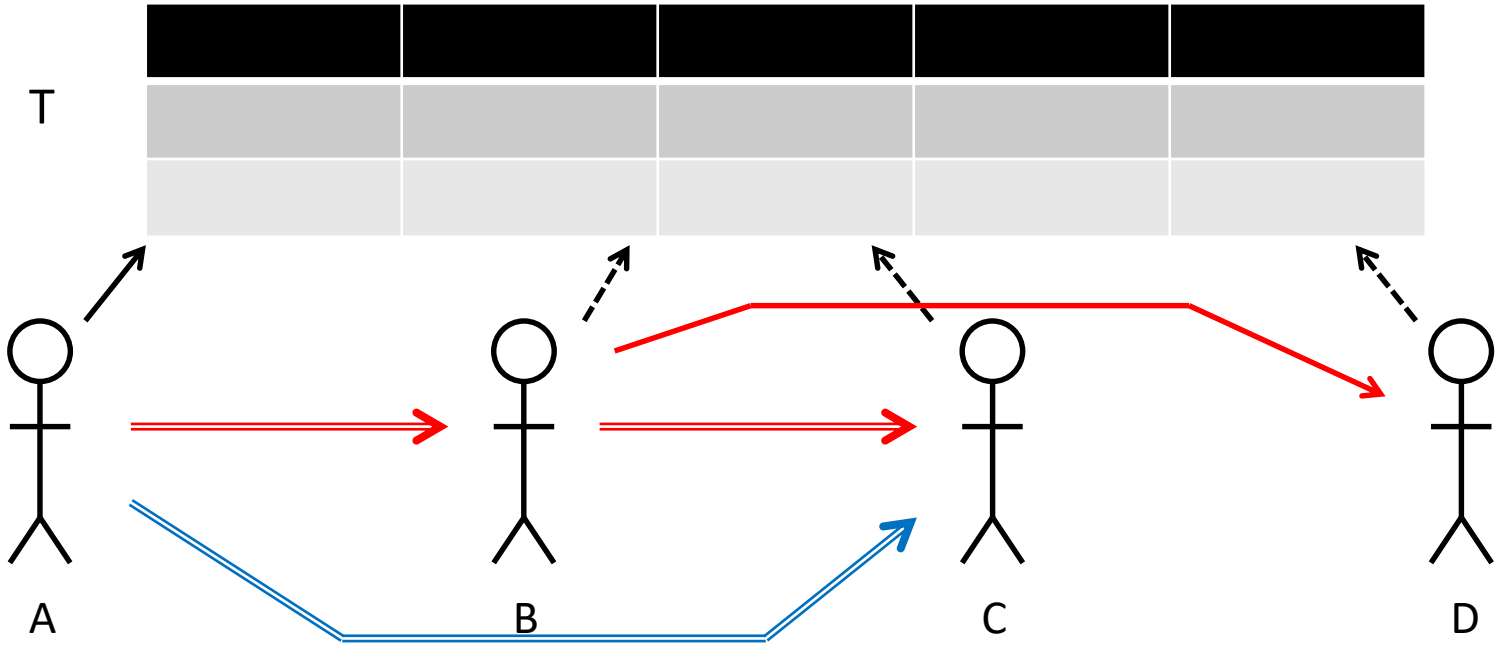
Аннулирование привилегий

При аннулировании привилегий отслеживается путь их передачи по графу идентификаторов авторизации и объектов БД. Поэтому возможны ситуации, когда у некоторого authID останется привилегия даже после ее аннулирования (при условии, что он получил ее несколькими путями).

A – текущий пользователь SQL сессии и владелец таблицы T.

REVOKE priv ON T FROM B CASCADE;

REVOKE priv ON T FROM C RESTRICT;



Аннулирование ролей

REVOKE [ADMIN OPTION FOR] role_name_list FROM { PUBLIC | authID_list } [GRANTED BY { CURRENT_USER | CURRENT_ROLE }] { RESTRICT | CASCADE }

Действие операции аннулирования ролей очень похоже на действие операции аннулирования привилегий. Отличия состоят в том, что аннулируются не привилегии, а роли, а также в том, что для аннулирования привилегии на передачу роли используется раздел ADMIN OPTION FOR.

Если некоторая привилегия или роль была передана PUBLIC, то ей обладают все пользователи. Но нет возможности аннулировать такую привилегию или роль у отдельно указываемого пользователя. Привилегия или роль была передана всем, и аннулировать ее можно только сразу у всех, то есть у псевдопользователя PUBLIC.