УДК 004

Использование транзакций в базе данных на языке SQL

Кочитов Михаил Евгеньевич
Приамурский государственный университет им. Шолом-Алейхема
студент

Аннотация
В данной статье рассматривается использование транзакций в базе данных на языке SQL. Транзакции необходимы для создания группы последовательных запросов, при выполнении которых будут произведены изменения в таблицах базы данных. Также в статье рассмотрен пример с использованием всех видов транзакций, которые существуют в языке SQL и их применение над данными.
Ключевые слова: транзакция, база данных, SQL, записи, таблица, запросы

Using transactions in an SQL database

Kochitov Mikhail Evgenevich
Sholom-Aleichem Priamursky State University
student

Abstract
This article discusses the use of transactions in an SQL database. Transactions are necessary to create a group of sequential queries, during the execution of which changes will be made in the database tables. Also, the article will consider an example using all types of transactions that exist in the SQL language and their application over data.
Keywords: transaction, database, SQL, records, table, queries

В настоящее время большинство сайтов обладает огромными базами данных с различными данными, которые могут изменяться в различное время. Однако, чтобы изменять записи в базе данных более аккуратно и без всяких потерь, то используют транзакции. Транзакция – это группа последовательных запросов, которая имеет полезную функцию – откат изменений, выполненных запросами, которые были в транзакции. Например, если вы добавите в транзакцию запрос удаления 100 записей, то при откате изменения, все 100 записей вернуться, и вы ничего не потеряете.

Целью данной статьи является рассмотрение транзакций в базе данных на языке SQL. Также будут показаны все виды транзакций и использование их на примере таблицы в базе данных.

В статье И.В. Артамонова рассматриваются бизнес – транзакции и их характеристики и отличительные особенности [1]. Рассматривая статью М.Г. Ерохиной и А.В. Косса можно заметить отдельные вопросы, связанные с

несанкционированными транзакциями при использовании банковских (платежных) карт [2]. А.Ф. Макеев в своей статье рассматривает транзакции в распределенных системах [3]. В статье В.Г. Папинашвили рассматриваются транзакции в базе данных [4]. Рассматривая статью В.В. Кочергина, .О. Антонова В, М.Г. Огура можно увидеть параллельные транзакции в SQL [5].

Для того, чтобы приступить к написанию транзакций на языке SQL, то надо установить локальный сервер «OpenServer» [6] и СУБД (систему управления базами данных) «HeidiSQL» [7]. Данный локальный сервер и СУБД распространяются в интернете в свободном доступе и доступны для использования всем желающим. Далее откроем программу СУБД «HeidiSQL» и создадим таблицу под названием «data_person», которая будет хранить информацию о людях: имя, возраст и профессия (см. рис. 1)

id	name	age	profession
1	Михаил	22	Программист
2	Андрей	20	Оператор
3	Алексей	18	Музыкант
4	Александр	23	Дизайнер
5	Максим	20	Менеджер
6	Артем	17	Музыкант
7	Антон	20	Программист
8	Виктор	22	Оператор
9	Руслан	26	Организатор
10	Олег	21	Программист

Рисунок 1. Таблица «data_person» со столбцами имени, возраста и профессии

Теперь рассмотрим, какие бывают транзакции на языке SQL: транзакция с сохранением изменений (Commit), транзакция с откатом (отменой) изменений (Rollback) и транзакция с созданием точек сохранения (Savepoint). Далее рассмотрим каждую транзакцию на примере более подробно.

Транзакция с сохранением изменений (Commit) предназначена для полного сохранения изменений после выполнения запросов без отмены изменений. Данную транзакцию лучше проводить с теми данными, которыми исходное состояние в дальнейшем не понадобится. Далее рассмотрим пример с использованием данной транзакции.

```
1 DELETE FROM data_person
2   WHERE age = 20;
3 COMMIT;
```

Рисунок 2. SQL запрос с транзакцией сохранения изменений (Commit)

На рисунке 2 показан SQL запрос, в котором из таблицы «data_person» будут удалены записи людей, где их возраст равен 20 лет и при этом

используется ключевое слово «COMMIT», которое создает транзакцию с моментальным сохранением изменений без отката.

id	name	age	profession
1	Михаил	22	Программист
3	Алексей	18	Музыкант
4	Александр	23	Дизайнер
6	Артем	17	Музыкант
8	Виктор	22	Оператор
9	Руслан	26	Организатор
10	Олег	21	Программист

Рисунок 3. Таблица «data_person», в которой были удалены записи людей, у которых возраст 20 лет

Как видно на рисунке 3 результат выполнения запроса с транзакцией сохранения изменений (Commit). В таблице «data_person» были удалены записи людей, у которых был возраст равен 20 лет.

Далее рассмотрим пример с транзакцией отката (отмены) изменений (Rollback)

```
1 DELETE FROM data_person
2   WHERE age = 20;
3 ROLLBACK;
```

Рисунок 4. SQL запрос с транзакцией отмены изменений (Rollback)

На рисунке 4 показан SQL запрос, который также удалит из таблицы «data_person» записи людей, у которых возраст равен 20 лет, но при этом будет использовано слово «ROLLBACK», которое сразу откатит изменения.

id	name	age	profession
1	Михаил	22	Программист
2	Андрей	20	Оператор
3	Алексей	18	Музыкант
4	Александр	23	Дизайнер
5	Максим	20	Менеджер
6	Артем	17	Музыкант
7	Антон	20	Программист
8	Виктор	22	Оператор
9	Руслан	26	Организатор
10	Олег	21	Программист

Рисунок 5. Таблица «data_person» после использования транзакции отмены изменений (Rollback)

Как можно заметить на рисунке 5 таблица «data_person» никак не изменилась и определенные записи в ней не удалились, так как был произведена отмена удаления этих записей.

Далее рассмотрим пример с транзакцией создания точек сохранения (Savepoint)

```
1 SAVEPOINT SP1
2 DELETE FROM data_person
3   WHERE age = 20;
4 SAVEPOINT SP2;
5 DELETE FROM data_person
6   WHERE profession = "Музыкант";
7 SAVEPOINT SP3;
8 DELETE FROM data_person
9   WHERE profession = "Программист";
```

Рисунок 6. SQL запрос с транзакцией создания точек сохранения (Savepoint)

На рисунке 6 представлен SQL запрос, в котором используется транзакция создания трех контрольных точек сохранения (SP1, SP2, SP3) перед каждым запросом удаления записей: первый запрос» – удаление записей людей, у которых возраст равен 20 лет, второй запрос – удаление записей людей, у которых профессия «Музыкант» и третий запрос – удаление записей людей, у которых профессия «Программист». Далее выполним транзакцию трех запросов и глянем результат.

id	name	age	profession
4	Александр	23	Дизайнер
8	Виктор	22	Оператор
9	Руслан	26	Организатор

Рисунок 7. Таблица «data_person» после выполнения транзакции создания сохранения точек и запросов удаления указанных записей

На рисунке 7 показан результат выполнения SQL запроса транзакции создания контрольных точек и удаления записей. Теперь чтобы убедиться, что контрольные точки были созданы транзакцией, то рассмотрим следующий SQL запрос.

```
1 ROLLBACK TO SP2;
```

Рисунок 8. SQL запрос с откатом изменений (Rollback) до контрольной точки (SP2)

Рисунок 8 показал SQL запрос, в котором использована транзакция отмены (отката) изменений (Rollback) до контрольной точки сохранения SP2. Далее выполним этот SQL запрос и глянем результат таблицы.

id	name	age	profession
1	Михаил	22	Программист
3	Алексей	18	Музыкант
4	Александр	23	Дизайнер
6	Артем	17	Музыкант
8	Виктор	22	Оператор
9	Руслан	26	Организатор
10	Олег	21	Программист

Рисунок 9. Таблица «data_person» после отката изменений до контрольной точки SP2

На рисунке 9 показана таблица, в котором вернулись удаленные записи людей, у которых были профессии «Музыкант» и «Программист». Однако полностью таблица не восстановилась со всеми записями, так как SQL запрос был выполнен транзакцией отмены изменений до контрольной точки SP2. Если бы использовалась точка SP1, то все три запроса удаления записей были бы отменены, и таблица полностью бы восстановила все свои записи людей.

Также имеется SQL запрос транзакции удаления точки сохранения.

```
1 RELEASE SAVEPOINT SP1;
```

Рисунок 10. SQL запрос транзакции удаления контрольной точки SP1

Как можно предположить SQL запрос удаляет точку сохранения SP1, которая отменяет запрос и возвращает записи людей, у которых возраст равен 20 лет. При удалении этой контрольной точки, полностью таблицу со всеми записями уже не вернуть, так как первый запрос (удаление записей людей, где возраст равен 20 лет) не подвергается откату и изменения в нем полностью сохранились.

Таким образом было рассмотрено использование транзакций в базе данных на языке SQL. Можно предположить, что SQL транзакции очень необходимы в базе данных, так как благодаря им можно удаленные данные вернуть обратно с помощью откатов изменений. Также можно с транзакцией делать обратное – добавить новые записи в таблицу и при откате, они снова удалятся.

Библиографический список

1. Артамонов И.В. Бизнес-транзакции: характеристики и отличительные особенности // Бизнес-информатика. 2012. № 2 (20). С. 29-34.
2. Ерохина М.Г., Косс А.В. Отдельные вопросы, связанные с

несанкционированными транзакциями при использовании банковских (платежных) карт // Вестник Калининградского филиала Санкт-Петербургского университета МВД России. 2017. № 4 (50). С. 83-86.
3. Макеев А.Ф. Транзакции в распределенных системах // В сборнике: От земских учреждений к местному самоуправлению в России: традиции, опыт, перспективы (к 150-летию Земской реформы) Материалы Международной научно-практической конференции. 2014. С. 109-110.
4. Папинашвили В.Г. Транзакции в базах данных // Молодой ученый. 2017. № 12 (146). С. 29-30.
5. Кочергин В.В., Антонов В.О., Огур М.Г. Параллельные транзакции в SQL // В сборнике: Фундаментальные и прикладные научные исследования Сборник статей Международной научно-практической конференции. Ответственный редактор: Сукиасян Асатур Альбертович. 2015. С. 56-58.
6. Open Sever Panel URL: https://ospanel.io/ (дата обращения 15.08.2019)
7. HeidiSQL URL: https://www.heidisql.com/ (дата обращения 15.08.2019)