Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет информатики и радиоэлектроники

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Базы данных

Лабораторная работа № 7

«Триггеры. Механизм реализации транзакций»

Выполнил:

ст.гр. 113802

Разумов Д.А.

Проверила:

Василькова А.Н.

Минск 2022

Цель работы – изучить принципы работы триггеров в реляционных СУБД, формирование умений создания триггеров на добавление, изменение и удаление данных для создания пользовательских ограничений целостности данных, изучение механизма реализации транзакций.

Триггер — это специальный тип хранимой процедуры, которая автоматически вызывается, когда данные в определенной таблице добавляются, удаляются или изменяются с помощью SQL-предложений INSERT, DELETE или UPDATE. В зависимости от того, какое из событий, связанных с изменением данных, инициирует запуск триггера, он называется insert trigger, delete trigger или update trigger.

Создать новый триггер позволяет оператор:

CREATE TRIGGER trigger\_name trigger\_time trigger\_event

ON tbl\_name FOR EACH ROW trigger\_stmt;

Оператор создает триггер с именем trigger\_name, привязанный к

таблице tbl\_name. Не допускается привязка триггера к временной таблице или

представлению. Конструкция trigger\_time указывает момент выполнения

триггера и может принимать два значения:

– before – действия триггера производятся до выполнения операции

изменения таблицы;

– after – действия триггера производятся после выполнения

операции изменения таблицы.

Конструкция trigger\_event показывает, на какое событие должен

реагировать триггер, и может принимать три значения:

– insert – триггер привязан к событию вставки новой записи в

таблицу;

– update – триггер привязан к событию обновления записи таблицы;

– delete – триггер привязан к событию удаления записей таблицы.

В таблице drivers создадим триггер, удаляющий записи об исполнителях, информация которых была удалена. Триггер delete\_driver и его результат приведены на рисунке 1.

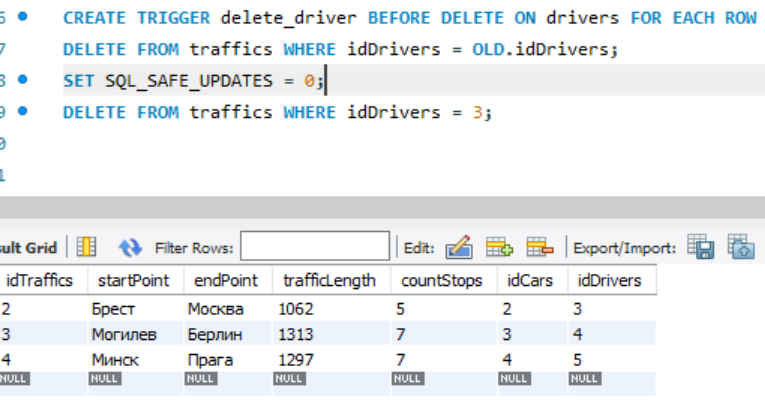


Рисунок 1 – Триггер удаления водителя

В таблице drivers создадим триггер update\_count\_stops, который вызывается после обновления опыта и обновляет количество остановок. Исходный вид таблицы приведен на рисунке 2.

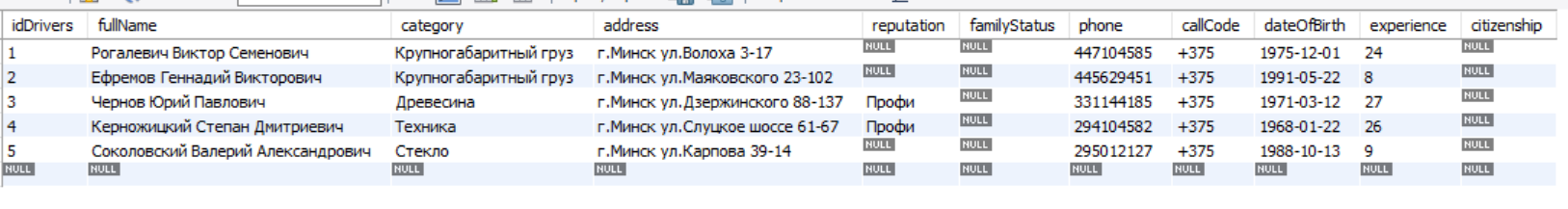


Рисунок 2 – Исходный вид таблицы drivers

Изменение таблицы приведено на рисунке 3.

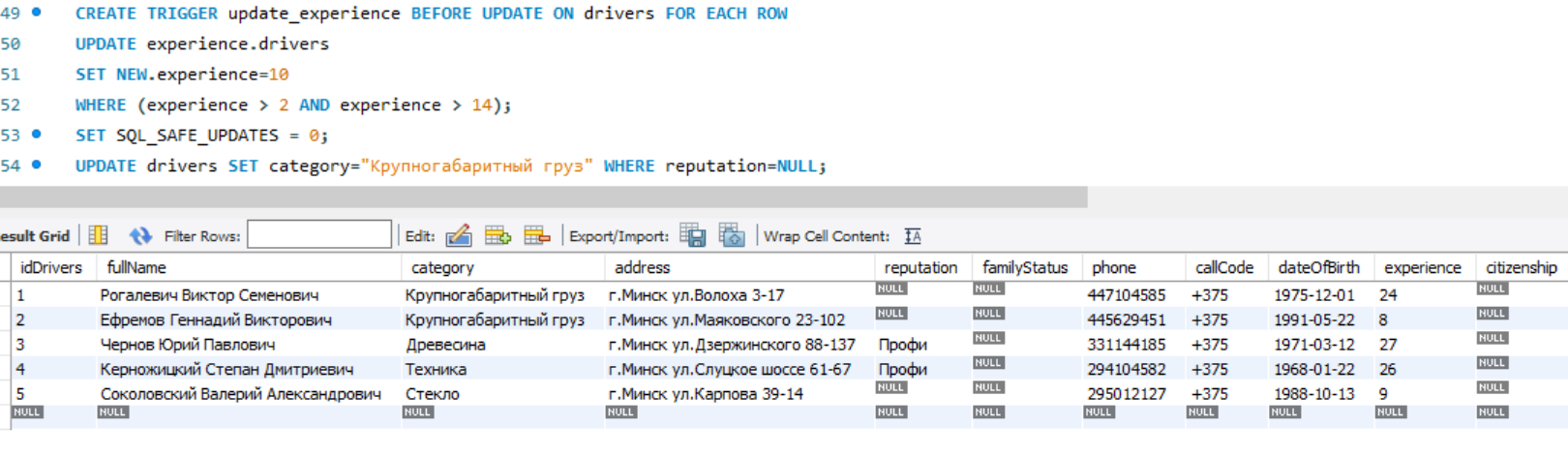


Рисунок 3 – Триггер изменения опыта

В таблице traffics создадим триггер, удаляющий записи о маршрутах, информация которых была удалена. Исходная таблица traffics приведена на рисунке 4.

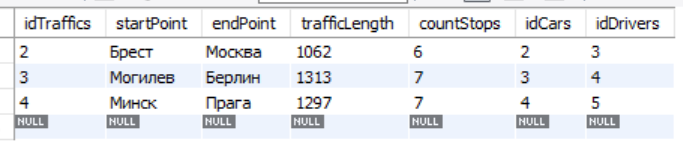


Рисунок 4 – Исходная таблица traffics

Исходный вид таблицы drivers приведен на рисунке 5.

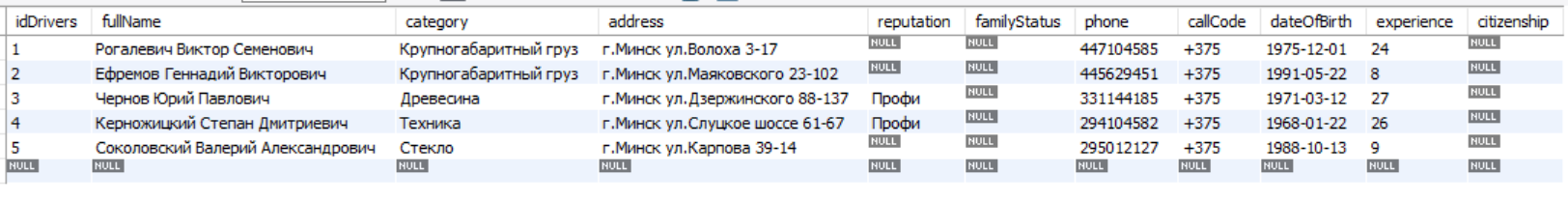


Рисунок 5 – Исходный вид таблицы drivers

Триггер delete\_driver и его результат приведены на рисунке 6.

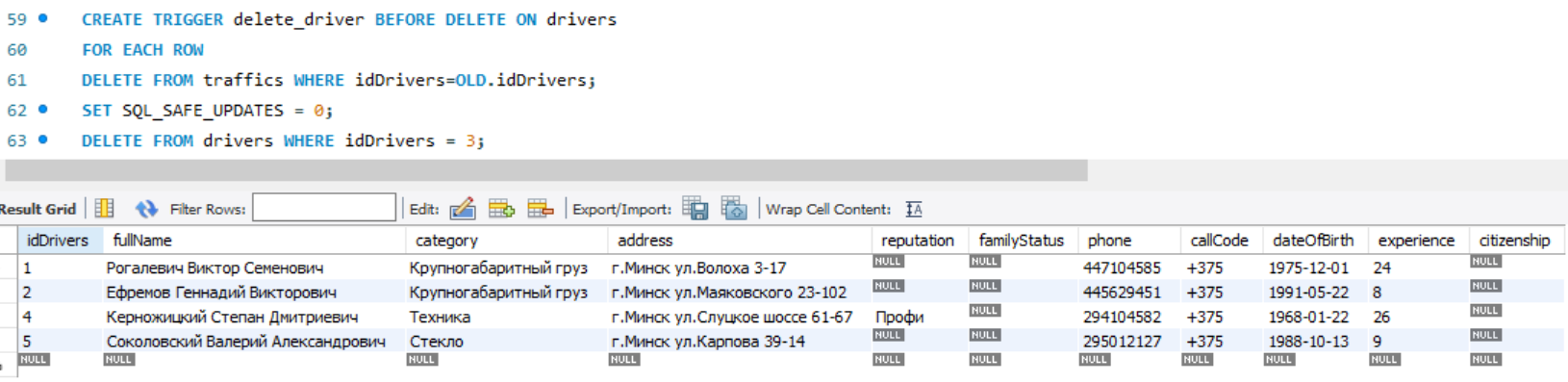


Рисунок 6 – Триггер delete\_driver и его результат

Удаления триггеров. Удалить существующий триггер позволяет оператор

DROP TRIGGER trigger\_name;

Триггеры можно использовать для проверки корректности ввода данных.

**Транзакции**

Транзакция является рабочей единицей работы с базой данных. Это последовательность операций, выполняемых в логическом порядке пользователем, либо программой, которая работает с БД.

Транзакция представляет собой группу запросов SQL, обрабатываемых атомарно, то есть как единое целое. Если подсистема базы данных может выполнить всю группу запросов, она делает это, но, если какой-либо запрос не может быть выполнен в результате сбоя или по иной причине, ни один запрос группы не будет выполнен.

Начинается транзакция ключевым словом BEGIN.

Завершается либо словом COMMIT (применить изменения), либо ROLLBACK (откатить изменения). Пример и результаты транзакции с командой COMMIT приведены на рисунках 7-8.

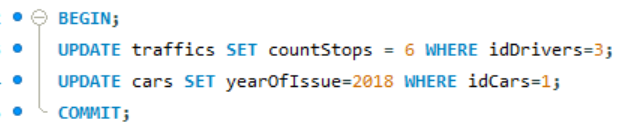


Рисунок 7 – Код транзакции

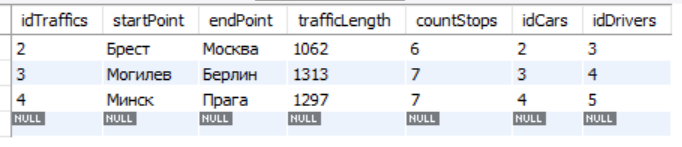


Рисунок 8 – Результат транзакции

После завершения транзакции будет гарантировано, что оба запроса либо выполнились успешно, либо, если хотя бы один из них выполнить не удалось, не будет выполнен ни один из них.

Управление транзакцией

Следующие команды используются для управления операциями.

– COMMIT – для сохранения изменений.

– ROLLBACK – откат изменений.

– SAVEPOINT – создает точки внутри групп операций, которые следует откатить.

– SET TRANSACTION – размещает имя транзакции.

**Команда ROLLBACK**

Команда ROLLBACK является транзакционной командой и используется для отмены операций, которые еще не были сохранены в базе данных. Эта команда может быть использована только для отката транзакции после последних команд commit или rollback.

Применим команду rollback после удаления информации. Синтаксис команды и результат показан на рисунке 9.

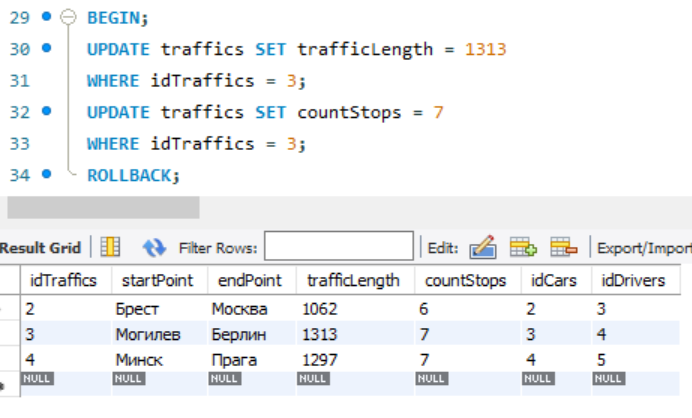


Рисунок 9 – Код и результат выполнения команды ROLLBACK

**Команда SAVEPOINT**

SAVEPOINT это точка в транзакции, когда вы можете откатить транзакцию до определенной точки без отката всех транзакций.

На рисунке 10 приведен пример, в котором мы собираемся обновить 3 записи из таблицы traffics. Мы хотим создать SAVEPOINT перед каждым обновлением, так что мы можем откатить к любому SAVEPOINT в любое время, чтобы вернуть соответствующие данные в исходное состояние.

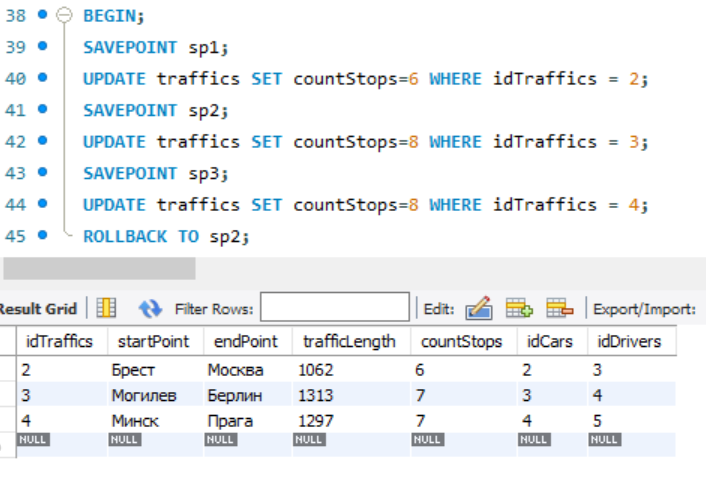


Рисунок 10 – Код и результат выполнения команды ROLLBACK

**Команда RELEASE SAVEPOINT**

Команда RELEASE SAVEPOINT используется для удаления SAVEPOINT, который вы создали.

Синтаксис команды RELEASE SAVEPOINT выглядит следующим образом:

BEGIN;

RELEASE SEVEPOINT sp;

После того, как SAVEPOINT был опубликован, вы больше не можете использовать команду ROLLBACK, чтобы отменить транзакции, выполненные с момента последнего SAVEPOINT.

**Команда SET TRANSACTION**

Команда SET TRANSACTION может быть использована для инициирования транзакции базы данных. Эта команда используется для определения характеристик для транзакции, которая следует. Например, вы можете указать транзакцию только для чтения или чтения и записи.