МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт информационных технологий |
| наименование института (факультета) |
| Кафедра математического и программного обеспечения ЭВМ |
| наименование кафедры  Базы данных |

наименование дисциплины в соответствии с учебным планом

ЛАБОРАТОРНАЯ РАБОТА №4

«Управление транзакциями. Журнализация изменений БД»

|  |  |
| --- | --- |
| Исполнитель |  |
| студент | 1ПИб-02-3оп-22 |
|  | группа |
|  | Маркелов С. А. |
|  | Фамилия, имя, отчество |
| Руководитель | Селяничев О. Л. |
|  | Ф.И.О. преподавателя |
| Оценка |  |
| Подпись |  |

2024 год

Цель:

1. Рассмотреть порядок выполнения транзакций
2. Изучить механизм ведения журнала изменений БД
3. Исследовать способы поддержания целостности БД

Транзакция – это такая неделимая с точки зрения воздействия на БД последовательность операторов манипулирования данными (чтения, удаления, вставки, модификации), что:

* либо результаты всех операторов, входящих в транзакцию, отображаются в БД,
* либо воздействие всех этих операторов полностью отсутствует.

Лозунг транзакций – «Все или ничего»:

* при завершении транзакции оператором COMMIT («зафиксировать») результаты фиксируются во внешней памяти;
* при завершении транзакции оператором ROLLBACK («ликвидировать») результаты отсутствуют во внешней памяти.

В Access завершение транзакции оператором COMMIT происходит в следующих случаях:

* после выполнения операций через SQL-запросы;
* при переходе к другой записи;
* при закрытии таблицы, запроса или формы;
* при удалении записи через интерфейс программы.

В Access завершение транзакции оператором ROLLBACK происходит в следующих случаях:

* при отмене изменений через комбинации Ctrl+Z или кнопки «Отменить» в меню;
* при ошибке в процессе выполнения транзакции.

1. Исследовать работу с одной БД средствами СУБД Access. Изучить, каким образом осуществляется незаметная для пользователей работа для некоторых случаев при одновременном выполнении транзакций разными пользователями.

В многопользовательских системах с одной БД одновременно работают несколько пользователей или прикладных программ. Система должна обеспечивать изолированность пользователей, т. е. создание иллюзии того, что каждый работает с БД в одиночку.

Тогда транзакция выступает единицей изолированности пользователей.

Существуют несколько вариантов изолированности транзакций, которые мы рассмотрим и проверим, как Access с ними работает.

Для работы была подготовлена следующая БД, содержащая данные о студентах – фамилии, даты рождения, оценки по математике, информатике, иностранному языку и пол (рис. 1).

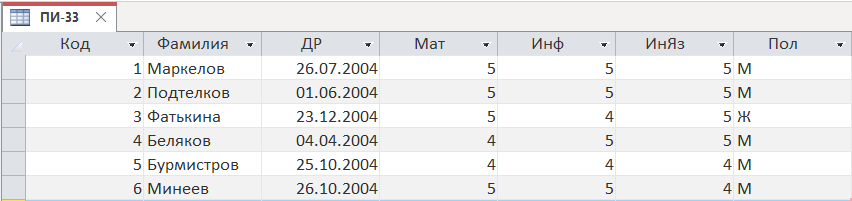


Рис. 1. База данных, над которой будут проводиться транзакции

1. Отсутствие потерянных изменений

Ситуация: Первая транзакция (Т1) изменяет объект БД А. До завершения Т1 вторая транзакция (Т2) также изменяет объект А. Т2 завершается оператором ROLLBACK (например, из-за нарушения ограничения целостности). Тогда при повторном чтении объекта А Т1 не видит изменений объекта, произведенных ранее.

Диагноз: Ситуация потерянных изменений; противоречие требованию изолированности пользователей.

Лечение: До завершения Т1 никакая другая транзакция не может изменять объект А.

Профилактика: Отсутствие потерянных изменений – минимальное требование к СУБД по части синхронизации параллельно выполняемых транзакций.

Т1: Пользователь №1 вносит изменение в БД – исправляет оценку 4 по математике у Бурмистрова на 5, но не сохраняет изменение (рис. 2).

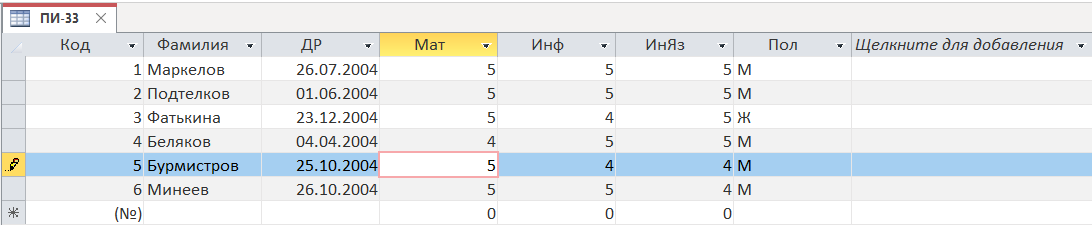


Рис. 2. Т1: изменение данных, не завершена

Т2: В это же время пользователь №2 вносит изменение в БД – также исправляет оценку 4 по математике у Бурмистрова на 5, но затем отменяет изменение, т. е. совершает ROLLBACK (рис. 3).

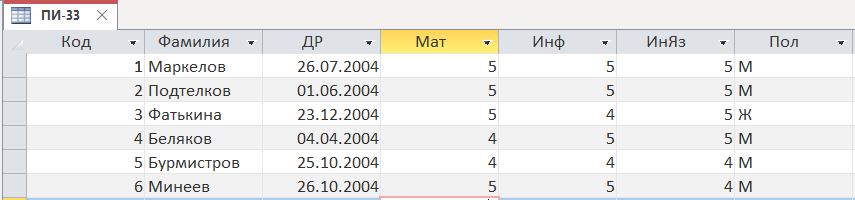


Рис. 3. Т2: изменение данных, отменена

Т1: пользователь №1 решает сохранить изменения (совершить COMMIT), но у него это не получается – всплывает диалоговое окно, сообщающее, что пока он правил эту запись, она была изменена другим пользователем (рис. 4).

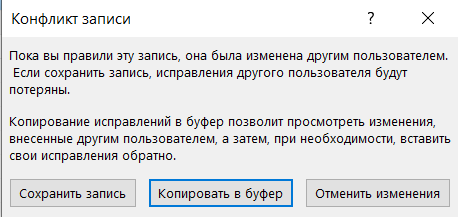


Рис. 4. Ошибка при попытке завершить Т1

Access предлагает либо сохранить свои изменения (совершить COMMIT), но тогда изменения другого пользователя пропадут, либо отменить их (совершить ROLLBACK). Также есть возможность скопировать изменения в буфер. В таком случае также будет совершен ROLLBACK, но пользователь №1 при необходимости быстро сможет повторить транзакцию после ознакомления с изменениями от пользователя №2.

Выберем вариант отмены изменений. После этого таблица будет иметь вид (рис. 5):

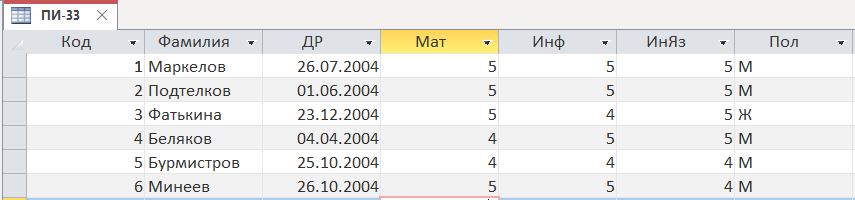


Рис. 5. БД после отмены Т1 и Т2

Если бы не было защиты от отсутствия потерянных изменений, данные пользователя №1 были бы утеряны из-за правки от пользователя №2.

1. Отсутствие чтения «грязных данных»

Ситуация. Т1 изменяет объект БД А. Параллельно с этим Т2 читает объект А. Т. к. операция изменения еще не завершена, Т2 видит несогласованные «грязные» данные.

Диагноз. Несоответствие требованию изолированности пользователей: каждый пользователь начинает свою транзакцию при согласованном состоянии базы данных и вправе видеть согласованные данные.

Профилактика. До завершения Т1, изменившей объект А, никакая другая транзакция не должна читать объект А (минимальным требованием является блокировка чтения объекта А до завершения операции его изменения в Т1).

Т1: Пользователь №1 вносит изменение в БД – исправляет оценку 5 по информатике у Минеева на 4, но не сохраняет изменения (рис. 6).

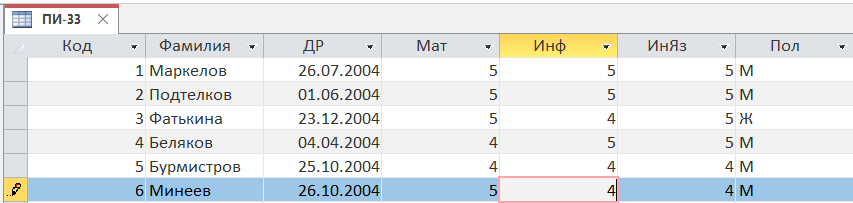


Рис. 6. Т1: изменение данных, не завершена

Т2: Пользователь №2 читает данные и не видит изменений – у Минеева по-прежнему 5 по информатике (рис. 7). Т. е. «грязные данные» от пользователя №1 пользователю №2 не видны.

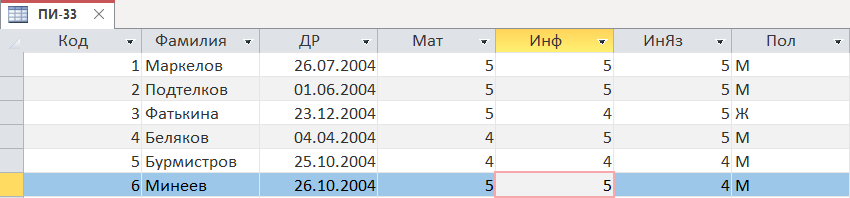


Рис. 7. Т1: чтение данных, изменений нет

Т1: Пользователь №1 отменяет изменения – делает ROLLBACK (рис. 8).

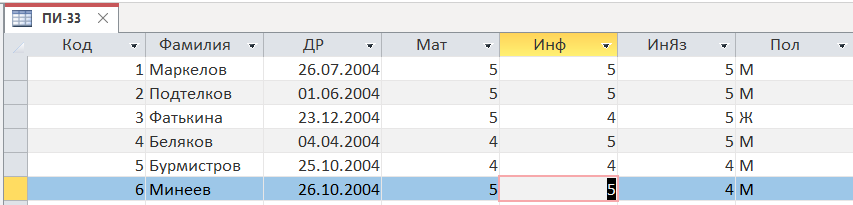


Рис. 8. Т1: изменение данных, отменена

Т2: Пользователь №2 после обновления БД (с помощью нажатия клавиши F5, или по прошествии заданного времени до автообновления) читает данные и также не видит изменений – 5 по информатике у Минеева (рис. 9).

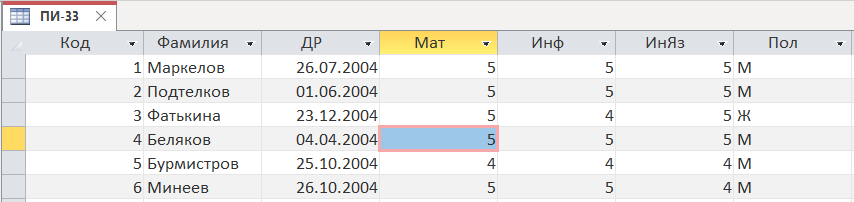


Рис. 9. Т2: чтение данных, изменений нет

Если бы не было защиты от отсутствия чтения «грязных данных», пользователь №2 увидел бы ошибочные данные, которые в итоге не были внесены пользователем №1 в БД.

1. Отсутствие неповторяющихся чтений

Ситуация. Т1 читает объект БД А. До завершения Т1 Т2 изменяет объект А и успешно завершается оператором COMMIT. Т1 повторно читает объект А и видит его измененное состояние.

Диагноз. Неповторяющиеся чтения.

Профилактика. До завершения Т1 никакая другая транзакция не должна изменять объект А. В большинстве систем это является максимальным требованием к синхронизации транзакций.

Т1: Пользователь №1 читает данные и видит, что у Белякова стоит 4 по математике (рис. 10).

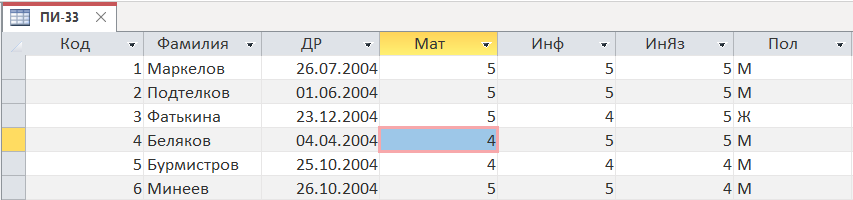


Рис. 10. Т1: чтение данных

Т2: Пользователь №2 вносит изменение в БД – исправляет оценку 4 по математике у Белякова на 5, и сохраняет изменение – делает COMMIT (рис. 11).

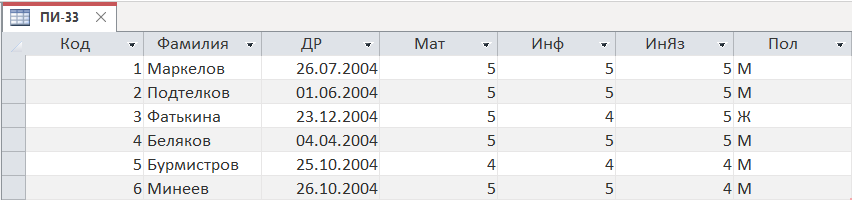


Рис. 11. Т1: изменение данных, завершена

Т1: Пользователь №1 после обновления БД (с помощью нажатия клавиши F5, или по прошествии заданного времени до автообновления) читает данные и видит новые данные – 5 по математике у Белякова (рис. 12).

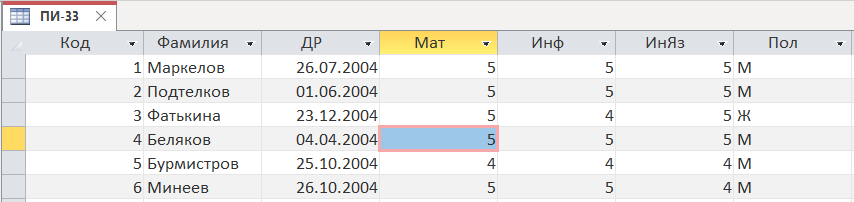


Рис. 12. Т2: чтение данных, изменения появились

Если бы не было защиты от неповторяющихся чтений, пользователь №1 видел бы неактуальную информацию без внесенных пользователем №2 изменений в БД.

1. Проблема кортежей-«фантомов»

Она вызывает ситуации, которые также противоречат изолированности пользователей.

Ситуация. Т1 выполняет оператор А выборки кортежей отношения R с условием выборки S. До завершения Т1 Т2 вставляет в отношение R новый кортеж К, удовлетворяющий условию S, и успешно завершается. Т1 повторно выполняет оператор А, и в результате появляется кортеж, который отсутствовал при первом выполнении оператора.

Диагноз. Противоречие идее изолированности транзакций.

Профилактика. Требуется более высокий уровень синхронизации транзакций. Идеи такой синхронизации (предикатные синхронизационные захваты) известны давно, но в большинстве систем не реализованы.

Т1: Пользователь №1 формирует запрос на выборку студентов, имеющих 5 по иностранному языку, но не сохраняет его (рис. 13-14).

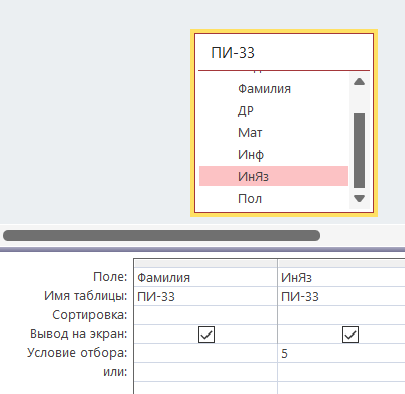


Рис. 13. Т1: создание запроса

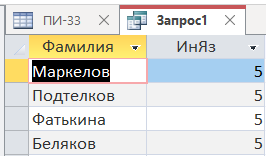


Рис. 14. Т1: результат запроса

Т2: Пользователь №2 добавляет новый кортеж с данными о студенте Цветкове, также имеющем оценку 5 по иностранному языку (рис. 15).

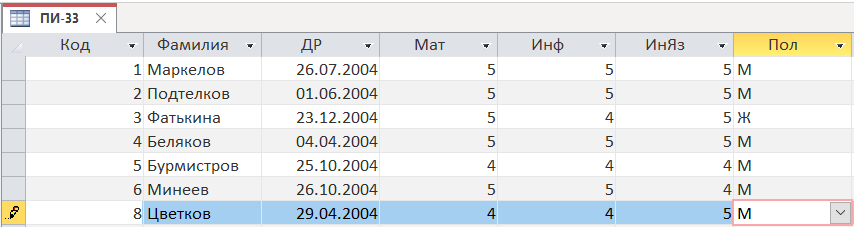


Рис. 15. Т2: добавление нового кортежа

Т1: Пользователь №1 повторно выполняет запрос с теми же условиями. В результатах запроса появился студент Цветков, внесенный пользователем №2 (рис. 16).

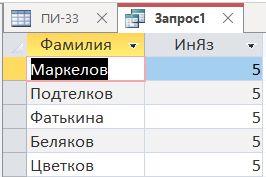


Рис. 16. Т1: результат повторного запроса

Такой результат говорит о том, что в Access нет защиты от кортежей-«фантомов».

2. На примере эксплуатации конкретной БД рассмотреть поведение СУБД, направленное на поддержание целостности БД, для различных случаев.

Одно из требований к СУБД – надежность хранения БД. Для восстановления согласованного состояния БД после аппаратных и программных сбоев поддерживается журнал изменений БД.

В основе поддержания целостного состояния БД – механизм транзакций. Общие принципы восстановления:

* результаты зафиксированных транзакций должны быть сохранены в восстановленном состоянии БД;
* результаты незафиксированных транзакций должны отсутствовать в восстановленном состоянии БД.

1. Откат транзакции

Тривиальной ситуацией отката является ее явное завершение оператором ROLLBACK. Возможны также ситуации, когда откат инициируется системой. Примерами могут быть возникновение исключительной ситуации в прикладной программе (например, деление на ноль) или выбор транзакции в качестве жертвы при обнаружении синхронизационного тупика. Для восстановления согласованного состояния БД при индивидуальном откате транзакции нужно устранить последствия операторов модификации БД, которые выполнялись в этой транзакции.

Внесем изменения в БД – исправим оценку 4 по информатике у Цветкова на 5 (рис. 17).

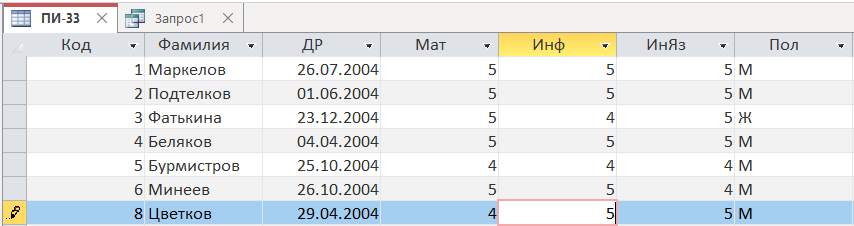


Рис. 17. Внесение изменений в БД

Затем совершим оператор ROLLBACK, т. е. откатим изменения – с помощью нажатия комбинации клавиш Ctrl+Z либо кнопки «Отмена изменений» в меню (рис. 18). СУБД восстановила предыдущее значение в данной ячейке.

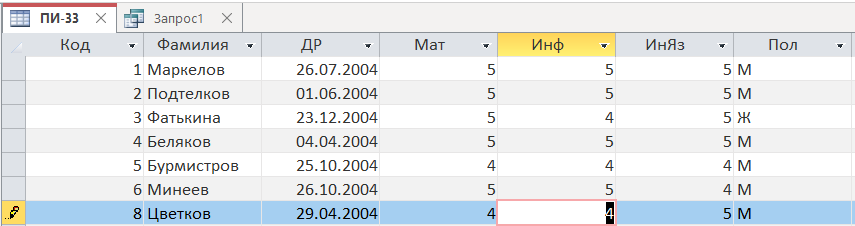


Рис. 18. Ручной откат изменений в БД

Теперь попробуем внести данные, тип которых не соответствует данному полю. Например, вместо 4 по информатике у Фатькиной попробуем написать слово «Зачтено» в числовом поле «Инф» и попытаемся совершить COMMIT (рис. 19).

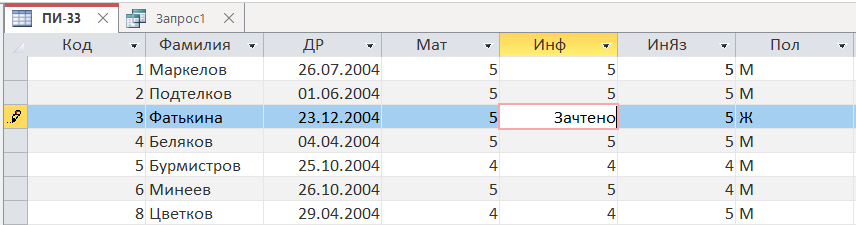


Рис. 19. Попытка внесения изменений в БД

Программа выдает сообщение об ошибке и автоматически завершает транзакцию оператором ROLLBACK (рис. 20).

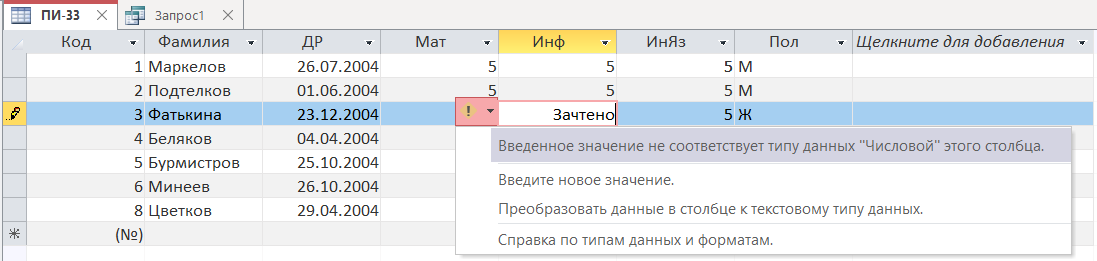


Рис. 20. Автоматический откат изменений в БД

1. «Мягкий» сбой

«Мягкий» сбой – это потеря содержимого оперативной памяти.

Такая ситуация может возникнуть при аварийном выключении электрического питания, при возникновении неустранимого сбоя процессора (например, срабатывании контроля оперативной памяти). Ситуация характеризуется потерей той части БД, которая к моменту сбоя содержалась в буферах оперативной памяти.

Инициируем «мягкий» сбой различными методами

Внесем изменение в БД – исправим оценку 4 по информатике у Бурмистрова на 5, но не сохраним изменения (рис. 21).

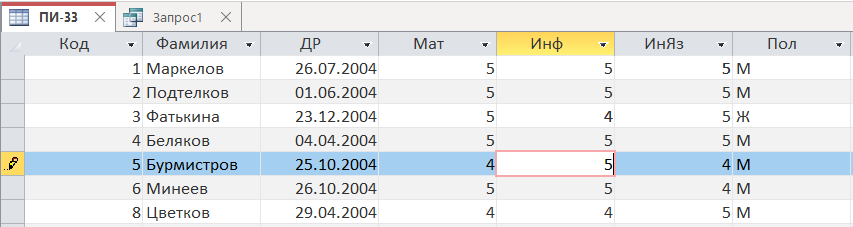


Рис. 21. Внесение изменений в БД

Зайдем в Диспетчер задач Windows через комбинацию клавиш Ctrl+Alt+Del и принудительно завершим процесс программы Access (рис. 22).

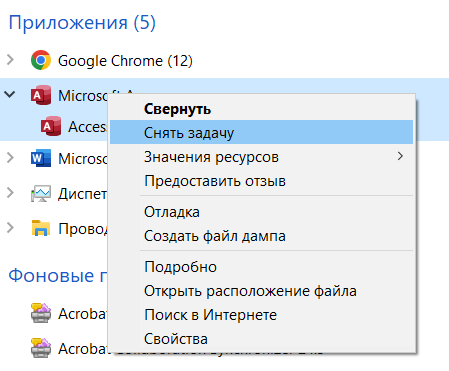


Рис. 22. Завершение процесса «Access» через Диспетчер задач

Заново открываем базу данных. Внесенное изменение осталось – Access сохранил его перед закрытием (рис. 23).

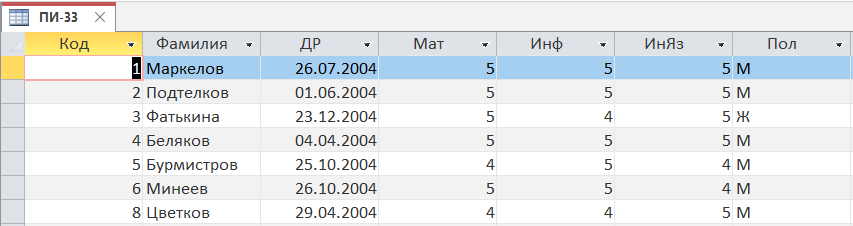


Рис. 23. БД после завершения процесса

Попробуем другой метод. Внесем изменение в БД – исправим дату рождения у Минеева, но не сохраним изменения (рис. 24).

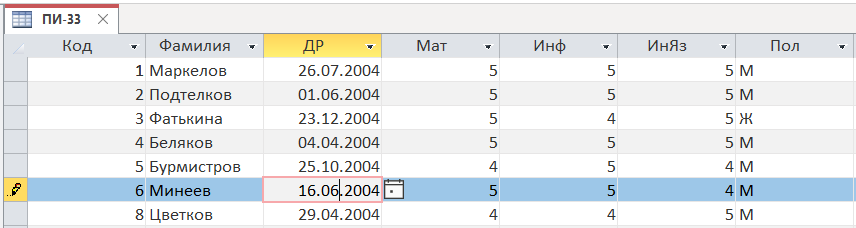


Рис. 24. Внесение изменений в БД

Далее выключим компьютер через кнопку «Завершение работы» в меню «Пуск» (рис. 25).

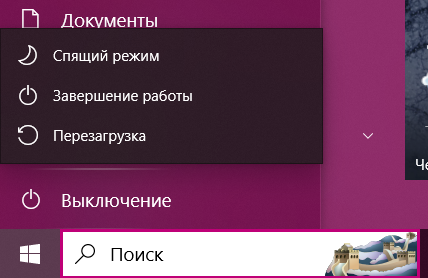


Рис. 25. «Завершение работы» через меню «Пуск»

После перезапуска компьютера и повторного открытия БД мы видим, что изменения также остались. Access сохранил их перед выключением компьютера (рис. 26).

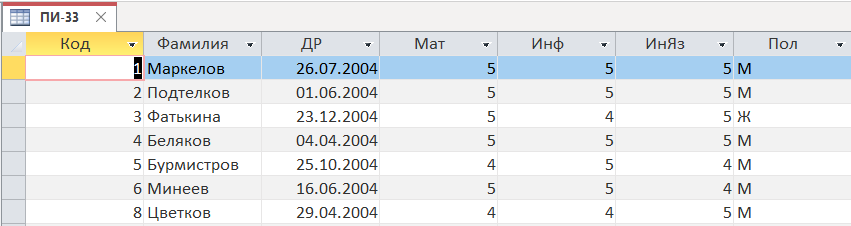


Рис. 26. БД после выключения компьютера через «Пуск»

Попробуем еще один метод. Внесем изменение в БД – исправим дату рождения у Бурмистрова, но не сохраним изменения (рис. 27).

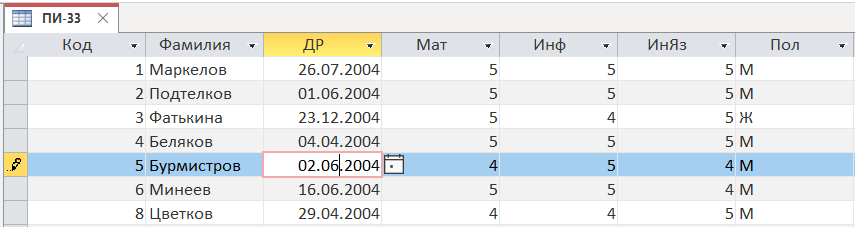


Рис. 27. Внесение изменений в БД

Теперь попробуем выключить компьютер не через меню «Пуск», а путем однократного нажатия на кнопку питания на системном блоке.

После перезапуска компьютера вновь открываем БД. Access вновь сохранил изменения (рис. 28)

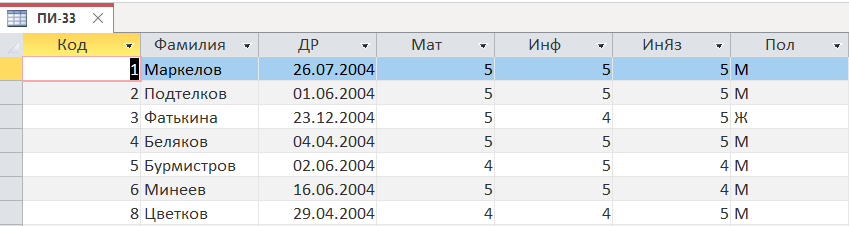


Рис. 28. БД после выключения компьютера через кнопку питания

На самом деле, ждать в данном случае иных результатов не имело смысла. Выключение компьютера через меню «Пуск» абсолютно аналогично выключению компьютера через одинарное нажатие на кнопку. Система точно также завершает все процессы и автоматически сохраняет данные.

Но если кнопку питания нажать и удерживать, то произойдет принудительное отключение компьютера, уже без участия ОС. Это полезно в тех случаях, если ОС «намертво» зависла и не откликается ни на какие действия. Проверим, как поведет себя СУБД в данном случае.

Внесем изменение в БД – исправим дату рождения у Фатькиной, но не сохраним изменение (рис. 29).

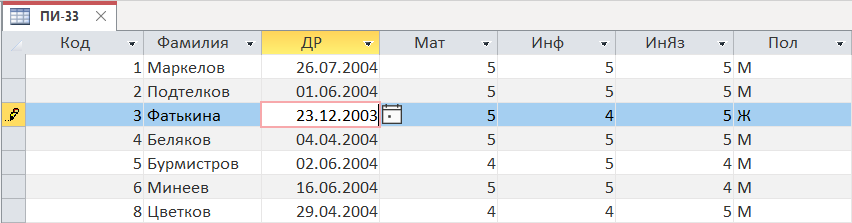


Рис. 29. Внесение изменений в БД

Теперь принудительно выключим компьютер, нажав и удерживая кнопку питания.

После перезапуска мы видим, что внесенные изменения не сохранились – БД находится в том состоянии, в каком была до их внесения (рис. 30).

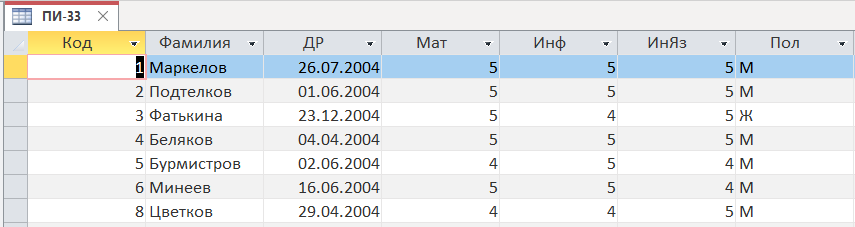


Рис. 30. БД после принудительного выключения компьютера через кнопку питания

Повторно внесем изменения даты рождения у Фатькиной (рис. 31).

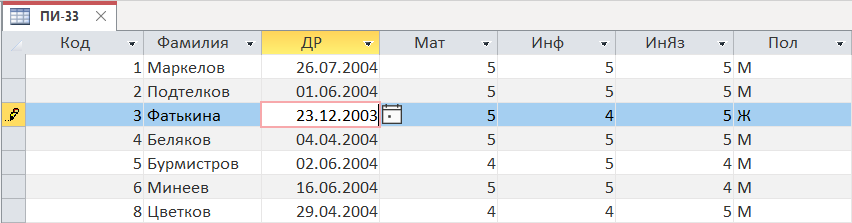


Рис. 31. Внесение изменений в БД

Теперь попробуем принудительно перезагрузить компьютер путем нажатия кнопки «Reset» на системном блоке.

Изменения вновь не сохранились – БД опять находится в том состоянии, в каком была до их внесения (рис. 32).

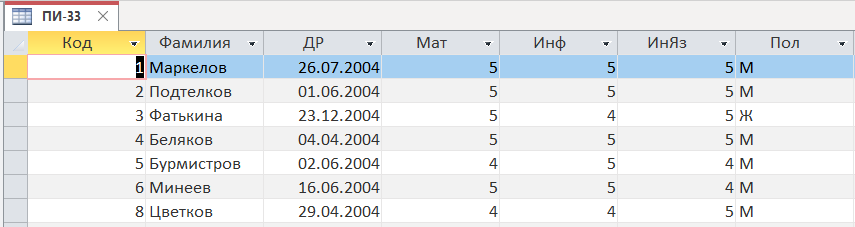


Рис. 32. БД после принудительной перезагрузки компьютера через кнопку «Reset»

Перейдем к более «радикальным» мерам. Вновь внесем такое изменение в БД (рис. 33).

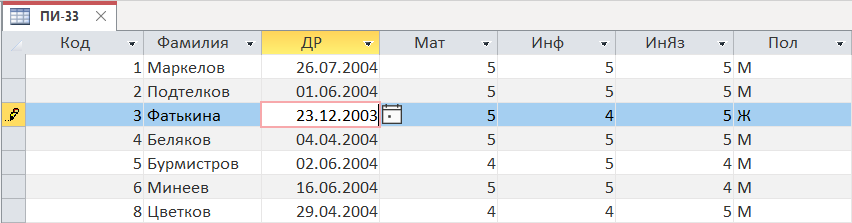


Рис. 33. Внесение изменений в БД

Отключим питание компьютера с помощью кнопки на блоке питания.

После повторного включения компьютера и запуска БД мы видим – изменения опять не сохранились (рис. 34).

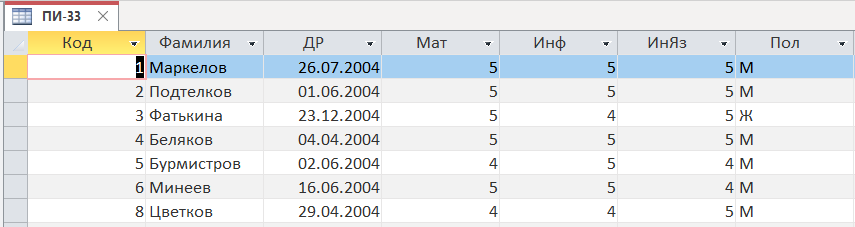


Рис. 34. БД после аварийного выключения компьютера через кнопку на БП

Еще раз внесем изменение в БД (рис. 35).

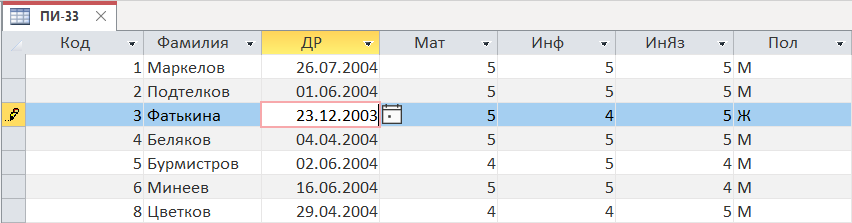


Рис. 35. Внесение изменений в БД

Теперь выключим компьютер из розетки.

После повторного включения компьютера в розетку и запуска БД изменения вновь не сохранились (рис. 36).

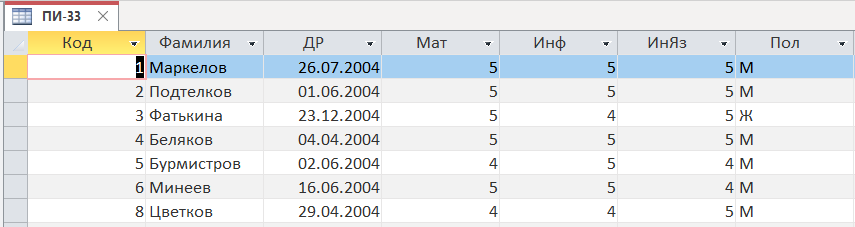


Рис. 36. БД после выключения компьютера из розетки

Таким образом, Access может автоматически сохранить изменения при закрытии программы или «традиционных» способах выключения компьютера. Это поможет спасти данные, если пользователь забыл сохраниться перед закрытием/выключением. Но при аварийных отключениях питания Access ничего сделать не сможет – данные будут утеряны. При этом сама БД не будет повреждена – она будет иметь такой же вид, какой был до внесения изменений.

1. «Жесткий» сбой

«Жесткий» сбой – это поломка основного внешнего носителя БД.

Эта ситуация при достаточно высокой надежности современных устройств внешней памяти может возникать сравнительно редко, но тем не менее, СУБД должна быть в состоянии восстановить БД даже и в этом случае. Основой восстановления является архивная копия и журнал изменений БД.

Для восстановления согласованного состояния БД после жесткого сбоя журнала изменений БД недостаточно – нужны журнал и архивная копия БД. Восстановление начинается с обратного копирования БД из архивной копии. Затем для всех закончившихся транзакций выполняется redo, т.е. операции выполняются повторно.

Более точно, происходит следующее:

* по журналу в прямом направлении выполняются все операции;
* для транзакций, которые не закончились к моменту сбоя, выполняется откат.

Устраивать «жесткий» сбой в прямом смысле, т. е. ломать диск не будем. Вместо этого вручную повредим файл БД. Для этого откроем его через Блокнот (рис. 37).

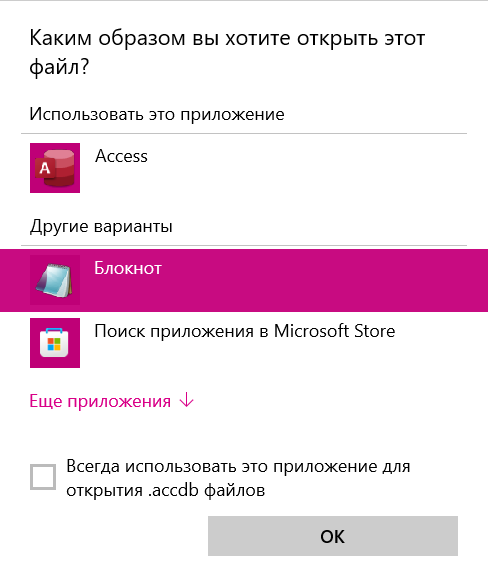


Рис. 37. Открытие файла БД через Блокнот

Перед нами откроется, на первый взгляд, бессмысленный набор символов (рис. 38). На самом деле это бинарное представление файла БД.



Рис. 38. Бинарное представление файла БД

Сотрем часть данных и сохраним файл. Теперь попробуем открыть поврежденный файл в Access.

Access не смог открыть поврежденную БД и выдал ошибку (рис. 39).

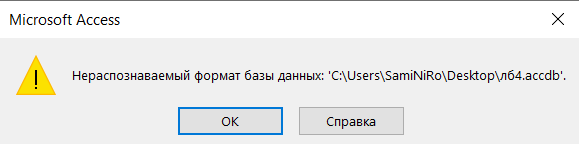


Рис. 39. Ошибка после повреждения файла БД

Попытаемся восстановить БД с помощью встроенных инструментов Access. Перейдем в раздел «Работа с базами данных» и нажимаем на кнопку «Сжать и восстановить базу данных» (рис. 40).

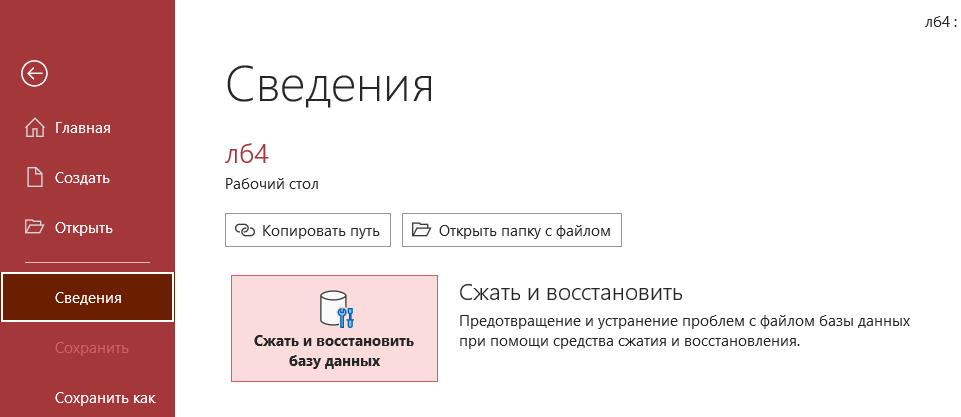


Рис. 40. Инструмент «Сжать и восстановить базу данных»

После выбора файла для восстановления Access вновь выдал ошибку (рис. 41). Тем самым, стандартных средств Access недостаточно для восстановления файла после «жесткого» сбоя.

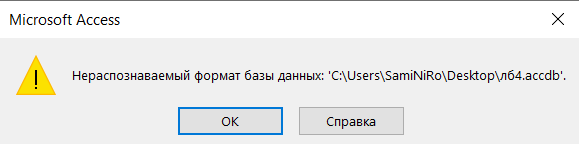


Рис. 41. Ошибка после повреждения файла БД

В некоторых случаях, если повреждение файла незначительное, Access может открыть базу данных, но при этом могут возникать ошибки, например, невозможность открыть определенные таблицы или запросы. В таком случае инструмент «Сжать и восстановить базу данных» может помочь восстановить нормальную работу БД.

Другие варианты восстановления после «жесткого» сбоя:

* Восстановление из резервной копии – это наиболее надежный способ восстановления данных. Копию БД следует хранить на другом носителе информации или в облачном хранилище.
* Использование утилит восстановления данных – существуют специализированные утилиты для восстановления поврежденных файлов, включая файлы баз данных Access. Эти утилиты могут попытаться восстановить данные даже из сильно поврежденных файлов, но 100%-ной гарантии восстановления все равно БД нет. Восстановленные данные могут быть неполными или же файл так и не получится открыть.