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-02-2оп-22 |
|  | группа |
|  | Зернов В.А. |
|  | Фамилия, имя, отчество |
| Руководитель | Селяничев О.Л. |
|  | Ф.И.О. преподавателя |
| Оценка |  |
| Подпись |  |

2024 год

ЗАДАНИЯ

1. Исследовать работу с одной БД средствами СУБД Access. Изучить, каким образом осуществляется незаметная для пользователей работа для случаев:

* **потерянных изменений**;

Tx1 изменяет объект «А» БД. До завершения Tx1 Tx2 также изменяет объект. Tx2 завершается оператором «rollback», тогда при повторном чтении объекта «А» Tx1 не видит изменение объекта, произведенное ранее.

Когда первый пользователь начинает изменять запись, второй также начинает ее редактировать. Результатом возникнет конфликт записи, из-за чего программа предложит изменения либо сохранить, либо отменить, либо копировать в буфер (рис. 1).

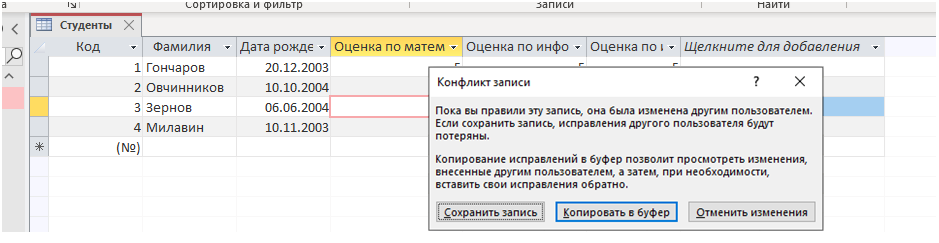


Рис. 1. Конфликт записи

Для вызова оператора «Rollback» необходимо отменить изменения, тогда возникнет ситуация потерянных изменений.

Вывод: Произошла потеря изменений. В программе «Access» такая ситуация предусмотрена, в связи с чем появляется соответствующее диалоговое окно. Чтобы при изменении объекта tx1 этого избежать, другой объект (tx) не должен вносить в него изменения.

* **чтение грязных данных**;

Tx1 изменяет объект «А», параллельно tx2 читает «А», т.к. операция изменения не была завершена, tx2 видит несогласованные «грязные» данные.

Первый пользователь вносит изменения в запись, но не завершает транзакцию. В это же время второй пользователь пытается прочитать эту запись и не видит изменений, которые начал вносить первый пользователь.

Вывод: была создана ситуация, в которой данные считывались некорректно. Чтобы предотвратить это до завершения транзакции, другой транзакции не следует считывать этот объект.

* **неповторяющихся чтений**;

Tx1 читает объект. До завершения tx1 tx2 изменяет объект и успешно завершается оператором «commit». Tx1 повторно читает объект и видит его измененное состояние.

Первый пользователь читает запись. Затем второй пользователь вносит в неё изменения, и первый пользователь видит их после повторного чтения. В программе «Access» можно увидеть изменения, если перейти к другой записи или подождать обновления данных (согласно настройкам), сохранить файл и т. д.

Вывод: удалось воспроизвести ситуацию неповторяющихся чтений. Чтобы избежать этой ситуации до завершения чтения объекта никакая другая tx не должна изменять этот объект. Завершить чтение объекта можно перейдя на другой объект и т.д.

* **кортежей-фантомов.**

Tx1 выполняет оператор выборки кортежа отношения «R» с условием выборки «S». До завершения tx1 tx2 вставляет в «R» новый кортеж, удовлетворяющий «S» и успешно завершается. Tx1 повторно выполняет оператор и в результате видит новый кортеж.

Первый пользователь выполняет запрос на сортировку данных. Затем второй пользователь добавляет новую запись, но первый пользователь видит её только после повторного выполнения запроса.

Вывод: удалось воспроизвести ситуацию кортежей-фантомов. Чтобы избежать этой ситуации никакая tx не должна добавлять новых кортежей, пока другая вызывает запрос.

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

- **отката транзакции**;

Откат транзакции происходит при завершении ее оператором «rollback».

Возможность отката транзакции удалось достичь при конфликте записей (рис. 1).

Кроме того, удалось достичь отката транзакции системой при одновременном изменении записи (рис. 2).

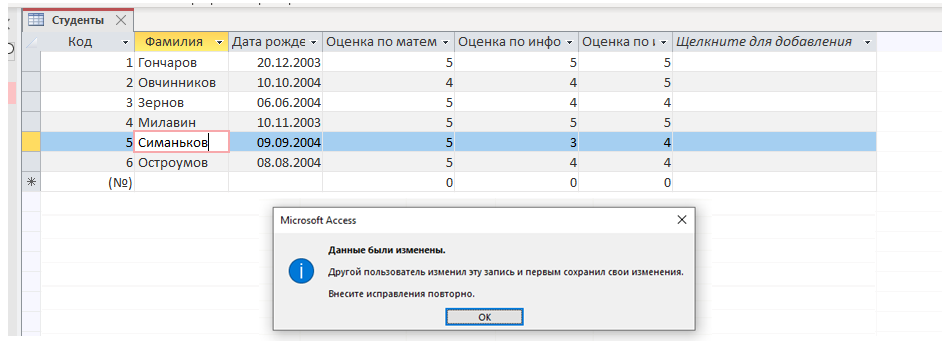


Рис. 2. Откат транзакции

- **мягкого сбоя**;

Мягкий сбой – потеря данных в оперативной памяти.

Мягкий сбой можно вызвать, выключив питание компьютера от электросети, зажать кнопку «Power» и так далее.

Если транзакция завершилась до мягкого сбоя, то ее можно повторить, если после, то результат транзакции не может быть сохранен.

- **жесткого сбоя** (без его реализации).

Жесткий сбой – потеря данных в внешней памяти.

Жесткий сбой можно вызвать, сломав или отключив жесткий диск, удалив файл и т.д. Чтобы восстановится после жесткого сбоя необходима резервная копия.

Заключение.

Была проведена одновременная работа двух пользователей над одной БД, такая работа возможна за счет изолированности пользователей, где транзакция одного пользователя изолирована от другого. Были рассмотрены и воспроизведены различные ситуации изолированности и поведения транзакций в СУБД «MS Access».