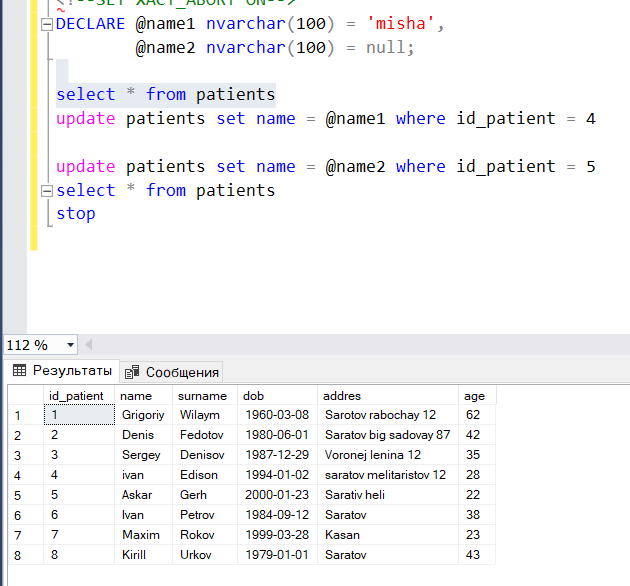
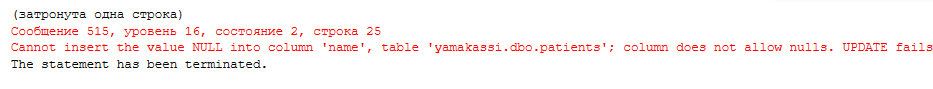
**Лабораторная работа 1. Транзакции и блокировки. Уровни изолированности транзакций. Распределённые транзакции.**

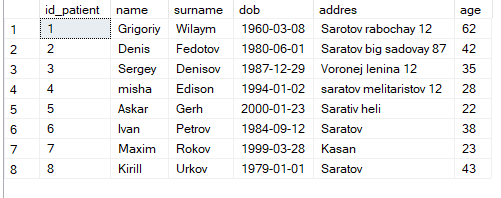
Предметная область БД, на которой необходимо производить упражнения, может быть любой. Рекомендуется использовать БД, которая была разработана в ходе лабораторного практикума по дисциплине «Управление данными».

**Задание:**

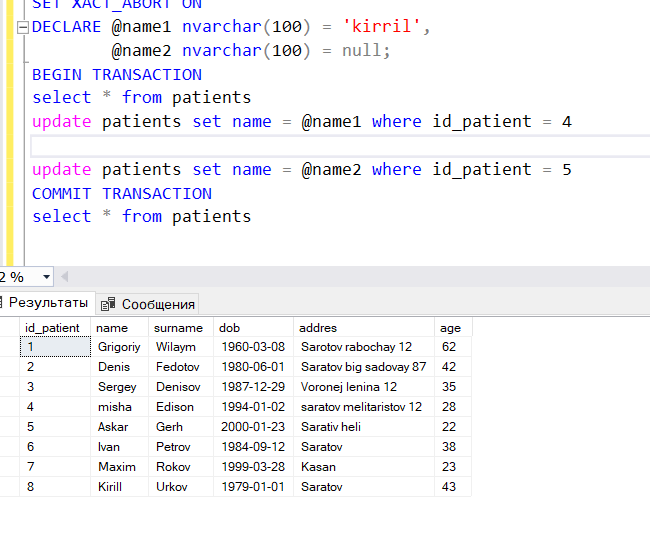
1. Наглядно продемонстрировать влияние параметра xact\_abort.











1. Наглядно продемонстрировать явные, неявные и автоподтверждённые транзакции в обоих режимах работы сервера (режим неявного подтверждения и автоматического подтверждения).

<!--№2==Явные транзакции 1) Запрос явных транзакций в явном режиме транзакций и результат-->

SET IMPLICIT\_TRANSACTIONS OFF

PRINT @@TRANCOUNT

BEGIN TRANSACTION

PRINT @@TRANCOUNT

select \* from diagnosis

insert into diagnosis values

('migren',4,1)

PRINT @@TRANCOUNT

DELETE FROM diagnosis

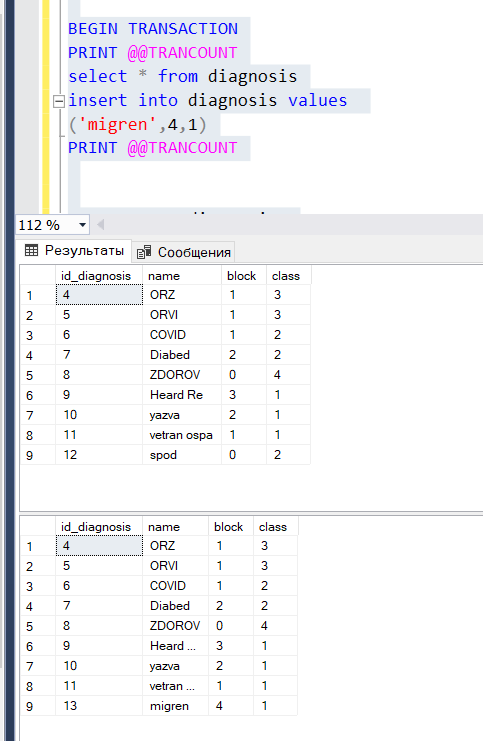
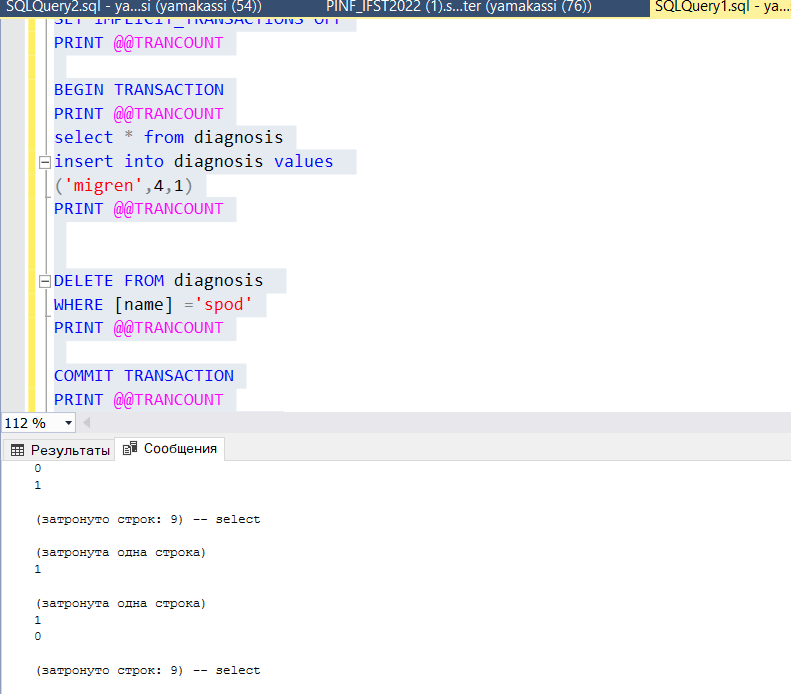
WHERE [name] ='spod'

PRINT @@TRANCOUNT

COMMIT TRANSACTION

PRINT @@TRANCOUNT

select \* from diagnosis



Внутри блока выполнения всей транзакции @@TRANCOUNT =1

*2***) явные транзакции в неявном режиме транзакций**

SET IMPLICIT\_TRANSACTIONS ON

select @@TRANCOUNT

BEGIN TRANSACTION

select @@TRANCOUNT

select \* from diagnosis

insert into diagnosis values

('migren wrc',4,1)

select @@TRANCOUNT

DELETE FROM diagnosis

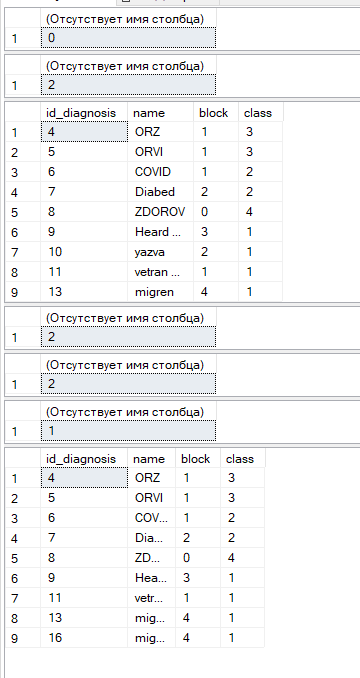
WHERE [name] ='yazva'

select @@TRANCOUNT

COMMIT TRANSACTION

select @@TRANCOUNT

select \* from diagnosis



Режим неявных транзакций плохо работает с явными транзакциями, т.к. неправильно увеличилось значение @@TRANCOUNT на 2 в начале транзакции и уменьшилось на 1 после коммита

Неявные транзакции

неявные транзакции в явном режиме транзакций

SET IMPLICIT\_TRANSACTIONS OFF

select @@TRANCOUNT

select \* from diagnosis

insert into diagnosis values

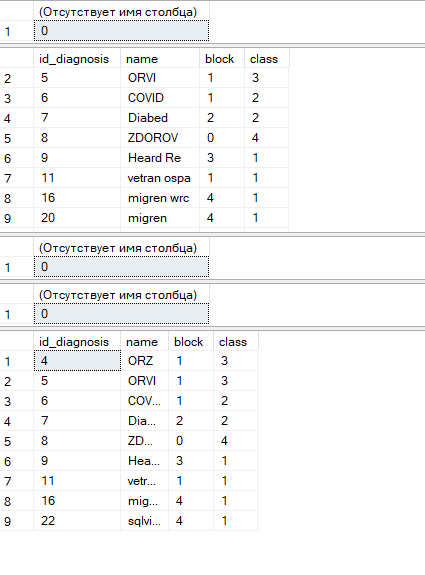
('sqlvitoz',4,1)

select @@TRANCOUNT

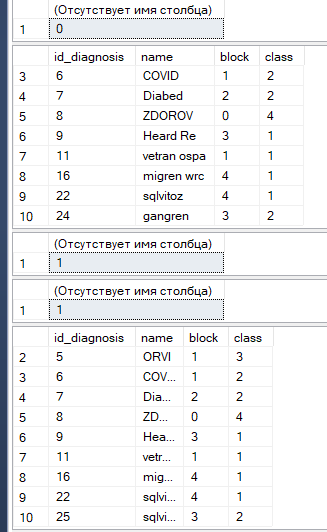
DELETE FROM diagnosis

WHERE [name] ='migren'

select @@TRANCOUNT PRINT @@TRANCOUNT select \* from diagnosis



выключен режим неявных транзакций, то значение @TRANCOUNT не изменилось

2) неявные транзакции в неявном режиме транзакций

SET IMPLICIT\_TRANSACTIONS on

select @@TRANCOUNT

select \* from diagnosis

insert into diagnosis values

('sqlvitoz 2',3,2)

select @@TRANCOUNT

DELETE FROM diagnosis

WHERE [name] ='gangren'

select @@TRANCOUNT

PRINT @@TRANCOUNT

select \* from diagnosis

При неявном режиме работы транзакций каждая операция представляла собой отдельную транзакцию

**Автоподтверждённые транзакции**

1. неявные транзакции в режиме автоподтверждённых транзакций

select \* from diagnosis

insert into diagnosis values

('fibroz2',6,5)

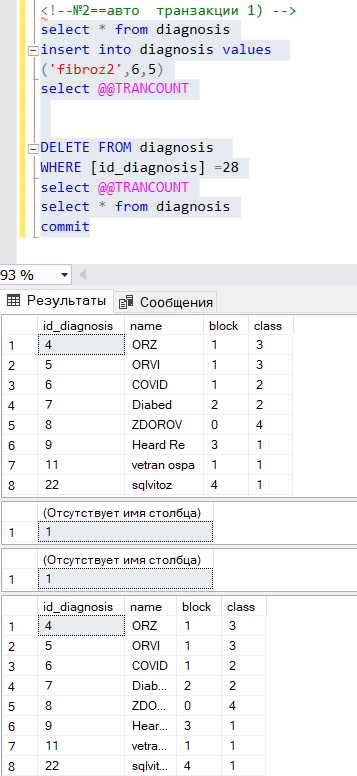
select @@TRANCOUNT

DELETE FROM diagnosis

WHERE [id\_diagnosis] =28

select @@TRANCOUNT

select \* from diagnosis



Каждая операция представляла собой транзакцию

1. явные транзакции в режиме автоподтверждённыех транзакци

select @@TRANCOUNT

BEGIN TRANSACTION

select @@TRANCOUNT

select \* from diagnosis

insert into diagnosis values

('rtg',3,4)

select @@TRANCOUNT

DELETE FROM diagnosis

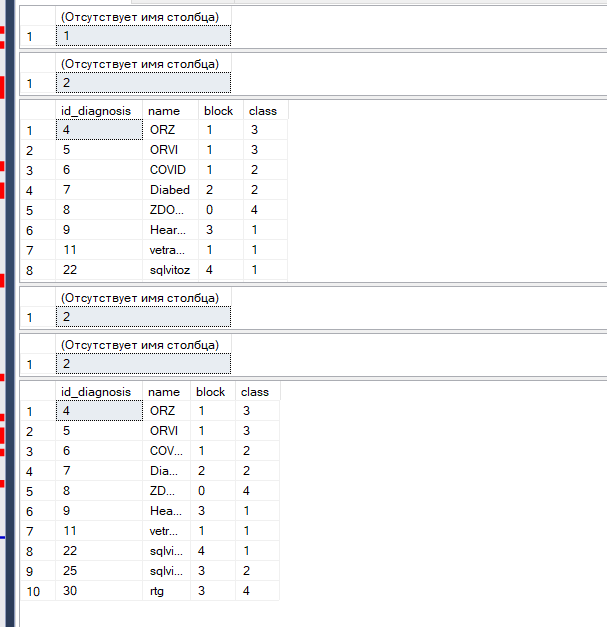
WHERE [name] ='fibroz2'

select @@TRANCOUNT

COMMIT TRANSACTION

PRINT @@TRANCOUNT

select \* from diagnosis

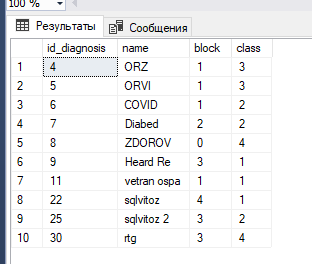


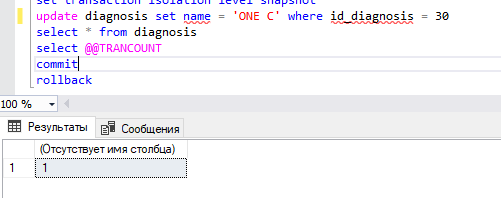
работает аналогично явным транзакция в режиме явных транзакций

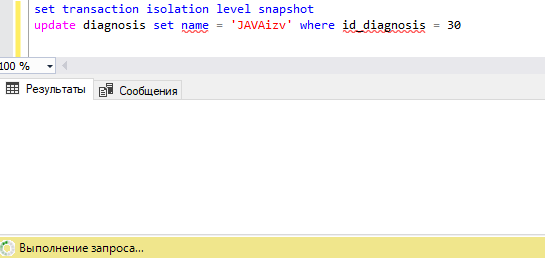
1. **Создать несколько подключений и наглядно продемонстрировать на них разницу между различными уровнями изолированности транзакций (5 уровней).**

**1)**

**До**



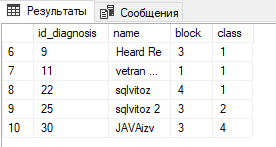




попытались выполнить lost update - но бд нам не позволила ( пока не будет закоммичена 1 транзакция) - на всех уровнях изоляции запрещен lost update

**2)** отличи uncommitted от committed ( на committed запрещено “грязное чтение”)

Данные до:

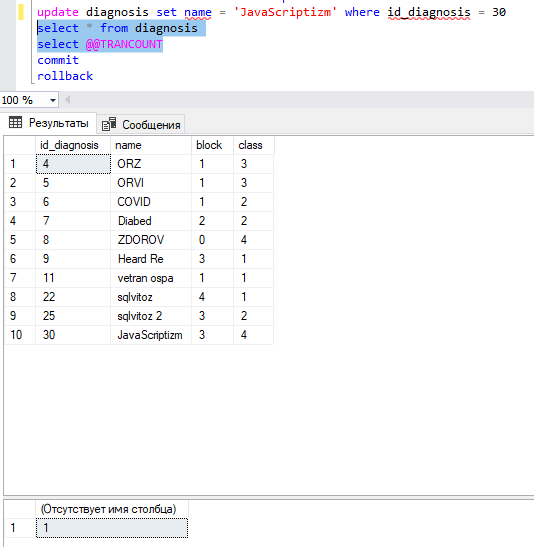


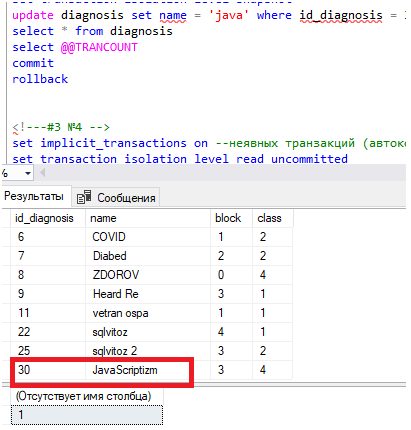
1 транзакция

update diagnosis set name = 'JavaScriptizm' where id\_diagnosis = 30

select \* from diagnosis

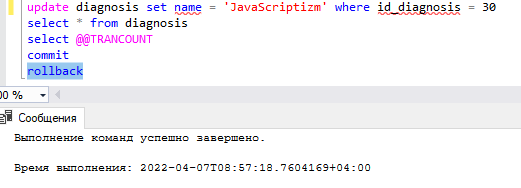
select @@TRANCOUNT

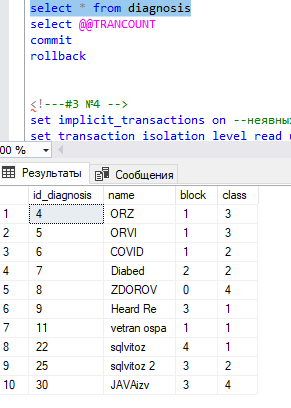




Сейчас были прочитаны данные, которые не закоммитчены 1 транзакции и их можем отменить

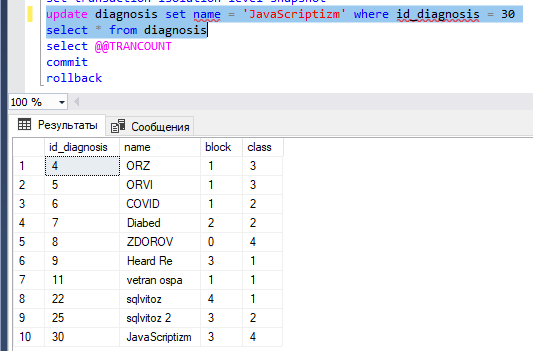
отменим 1 транзак

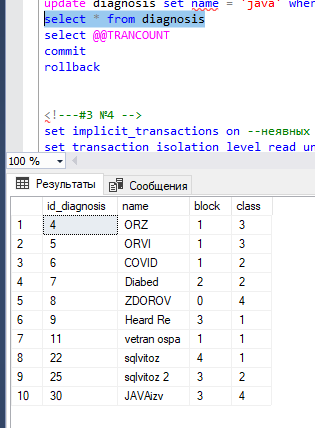




И теперь выходит, что были взяты данные которых не существует.

теперь включи commited





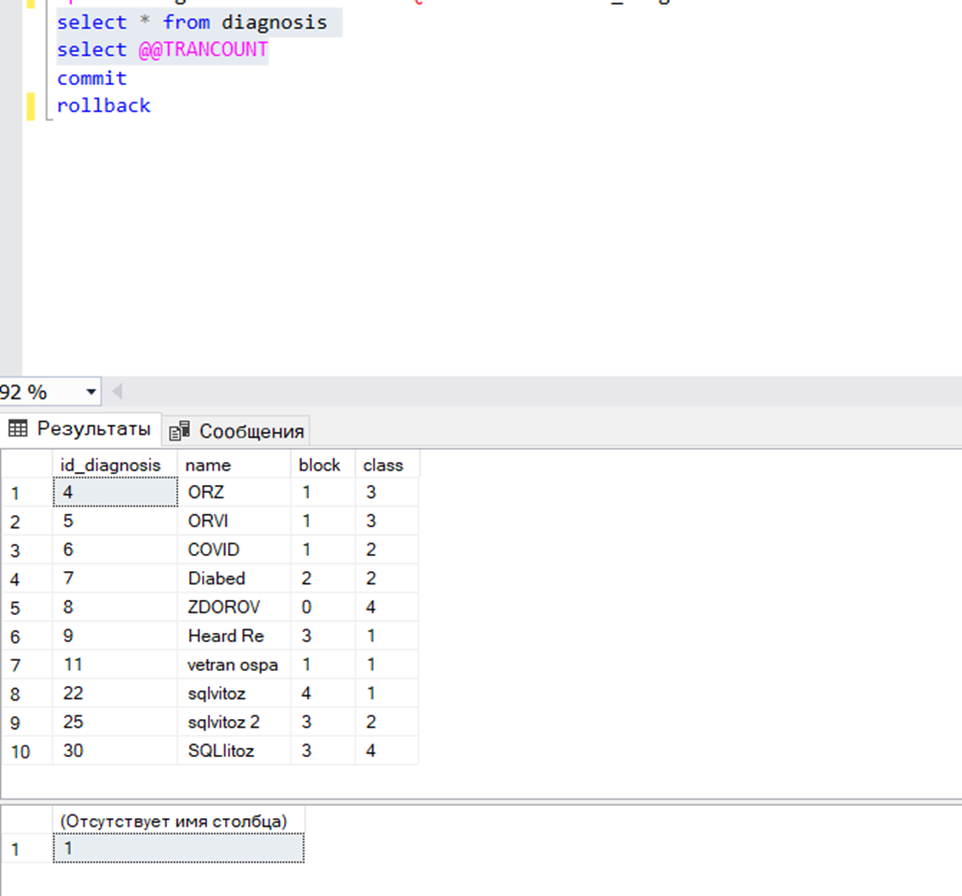
теперь данные прочитаны правильно ( данные до начало 1 транзакции)

1. **Nonrepeatable reads**

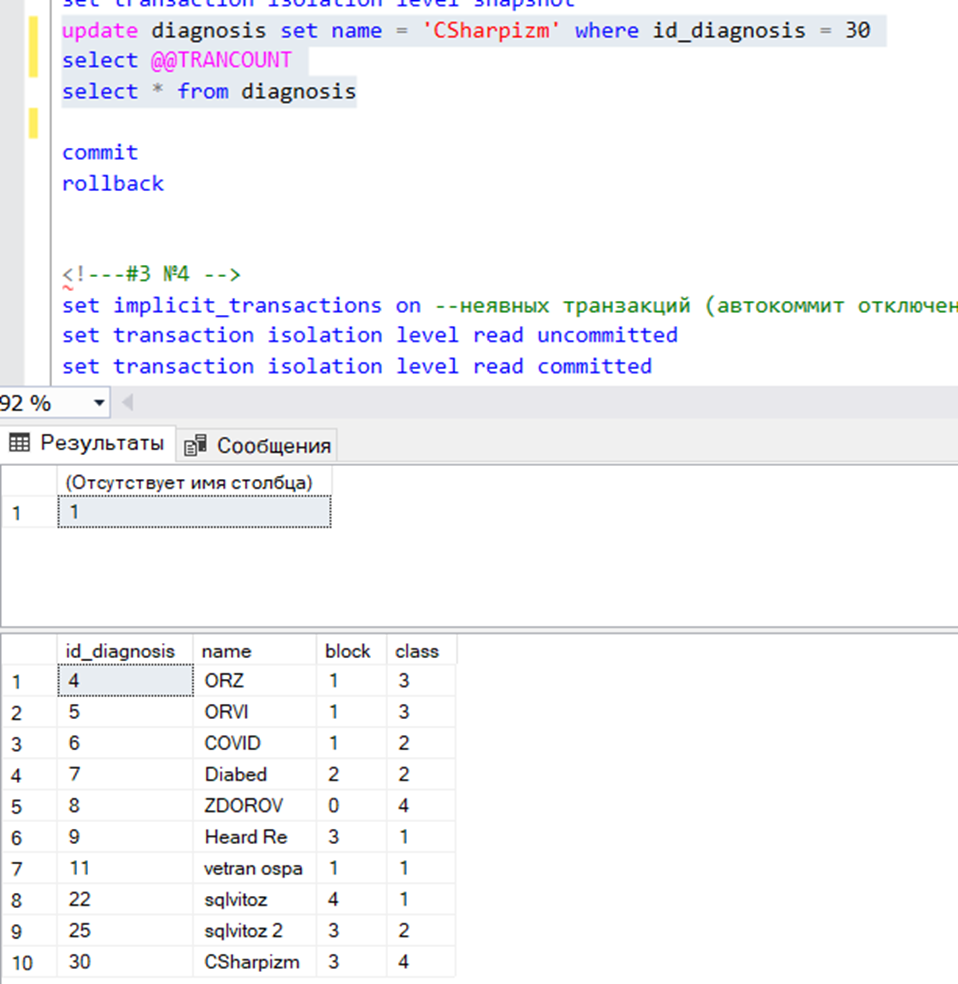
Транзакцией 1 считываем

Транзакцией 2 изменяем счит.данные ( она не за коммитчена)

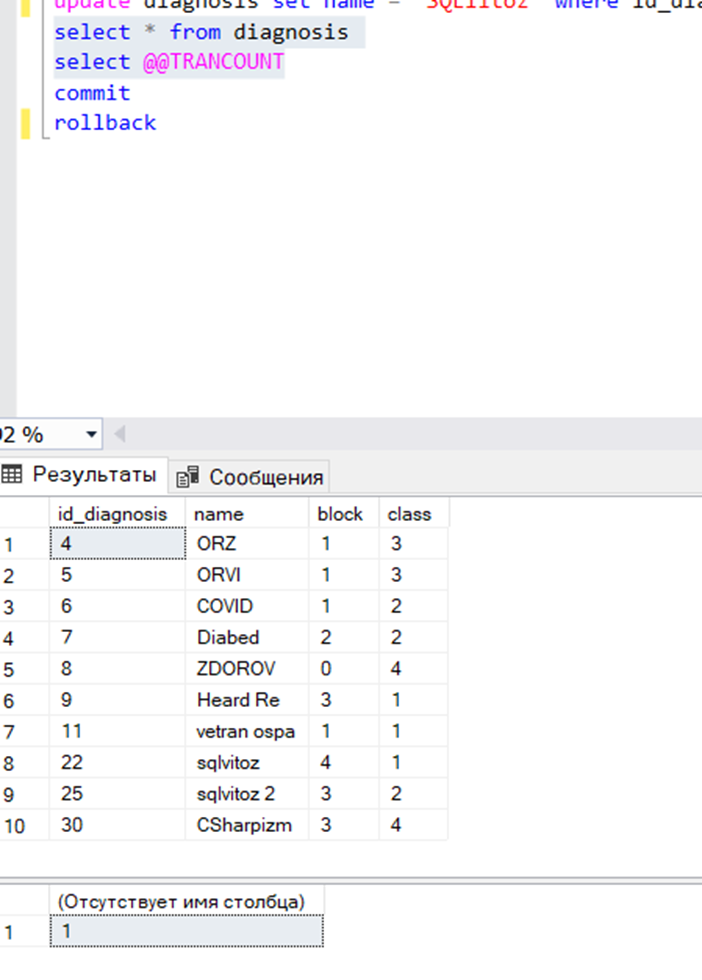
Транзакцией 1 снова считываем

1)

2)

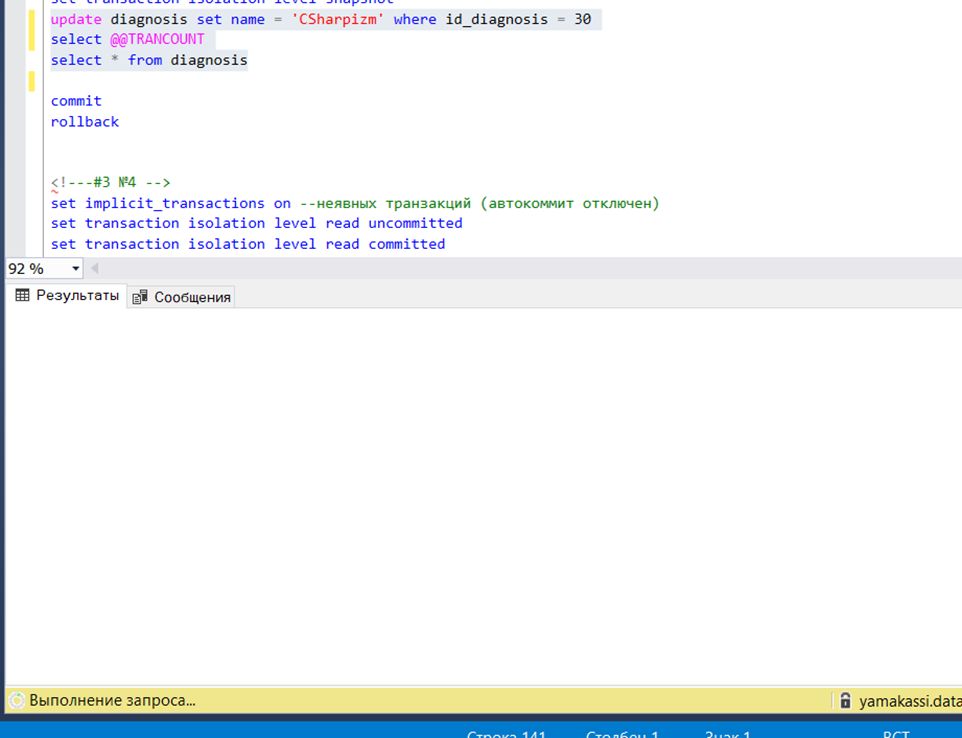


3)

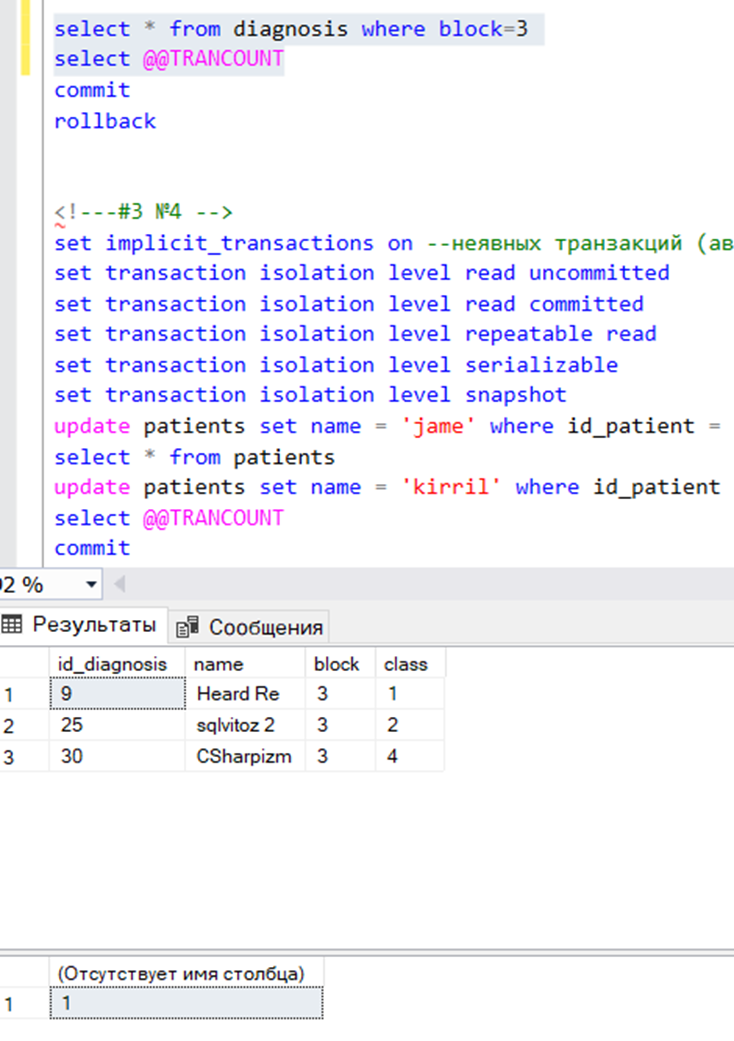


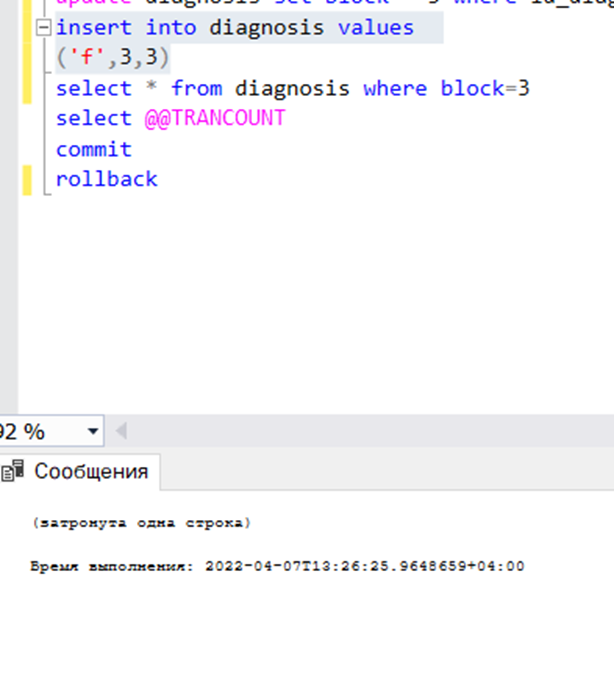
Во время 2 чтения(транзакция1) мы увидели обновленные данные не закоммитченой транзакции( транзакция 2 – может быть отменена и так будут увидены данные, которых НЕТ)

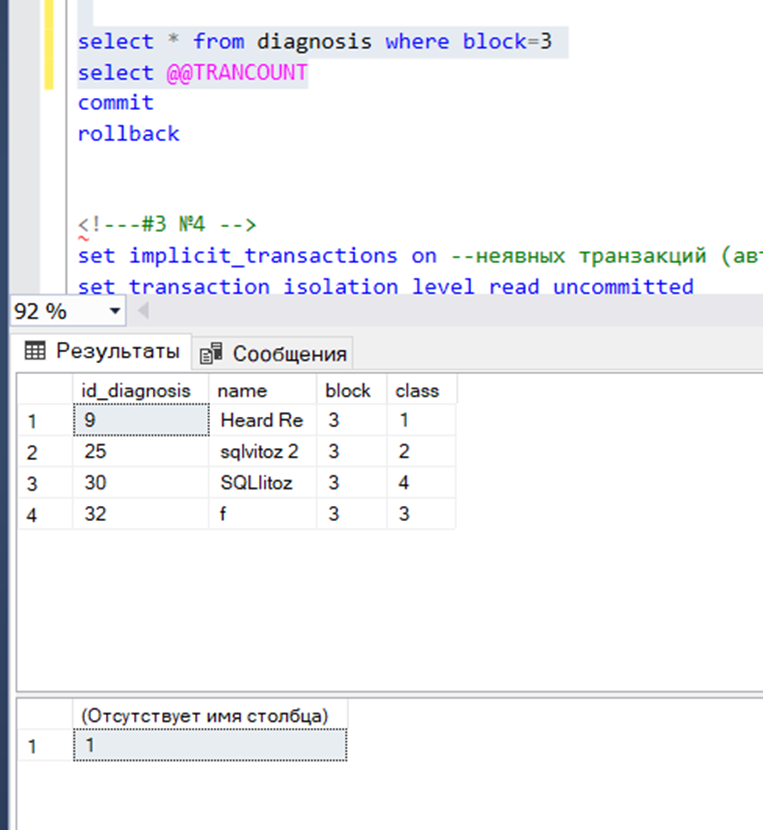
Повысим уровень изоляции до set transaction isolation level repeatable read – на этом уровне данные, которые не были зафиксированы не могут быть прочитаны, а также то, что другие транзакции не могут изменять данные, которые читает данная транзакция



**4)Phantoms ( различие уровней** REPEATABLE и SERIALIZABLE)



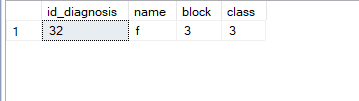




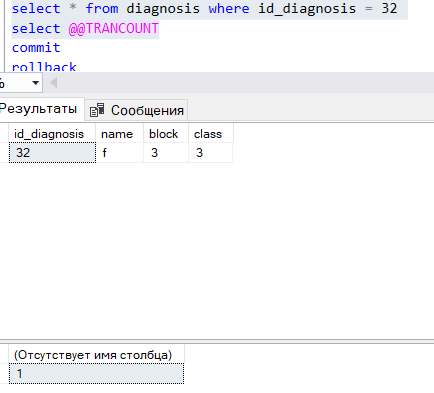
При выполнении двух запросов получается, что запрос с INSERT не дождется окончания другого запроса и вставит свое значение и мы увидим разные списки значений.

Если повысить уровень, до SERIALIZABLE – то будут блокироваться любые изменений

1. уровень SNAPSHOT

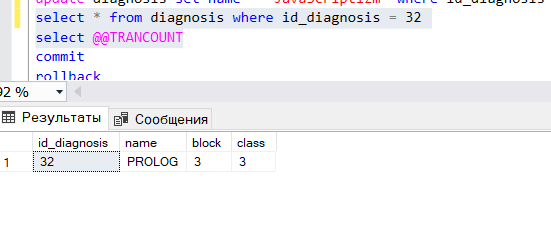


В другом окне выполним update diagnosis set name = 'PROLOG' where id\_diagnosis = 32

В первом 

Фиксируем данные

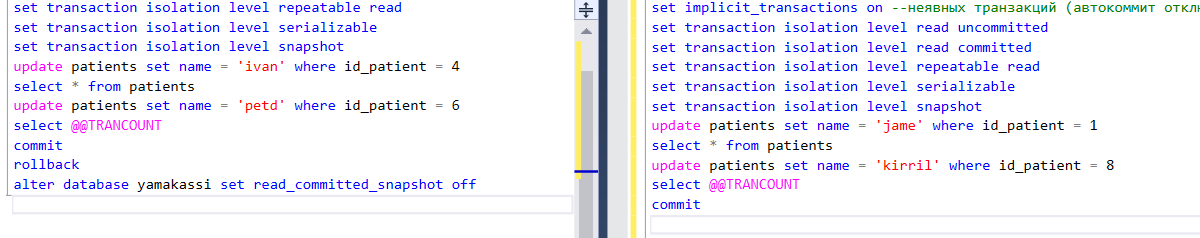
Снова посмотрим данные в первом окне и видим, что данные обновлены



При уровне изоляции SNAPSHOT любая другая транзакция будет использовать подтвержденные изменения, которые существовали непосредственно перед началом работы транзакции SNAPSHOT

Чтобы пользоваться этим уровнем нужно активировать этот уровень для базы данных командой SET ALLOW\_SNAPSHOT\_ISOLATION ON

**№4)** Промоделировать ситуацию взаимоблокировки транзакций. Объяснить полученные результаты



Оба запроса имеют две операции, но они обращаются к заданным таблицам перекрестно

Выполнить первые операции в двух окнах запроса

Дальше попробуем выполнить вторые команды и закрытие транзакций в каждом окне

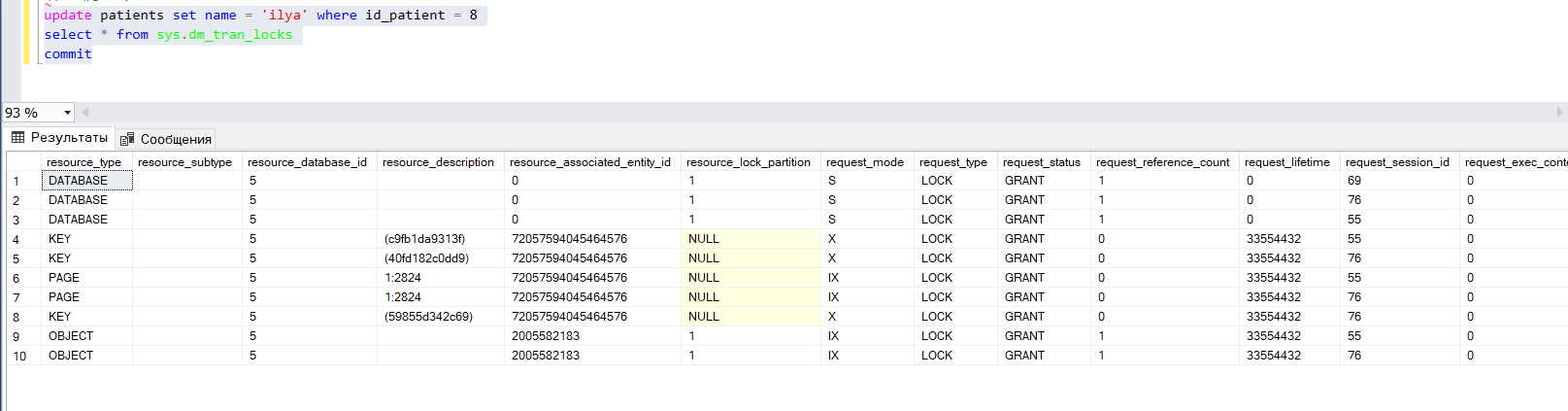
После выполнения второго запроса в первой транзакции база данных встала в режим ожидания

После выполнения второго запроса во второй транзакции база данных выдала ошибку о взаимоблокировке, и вторая инструкция первой транзакции и не выполнилась

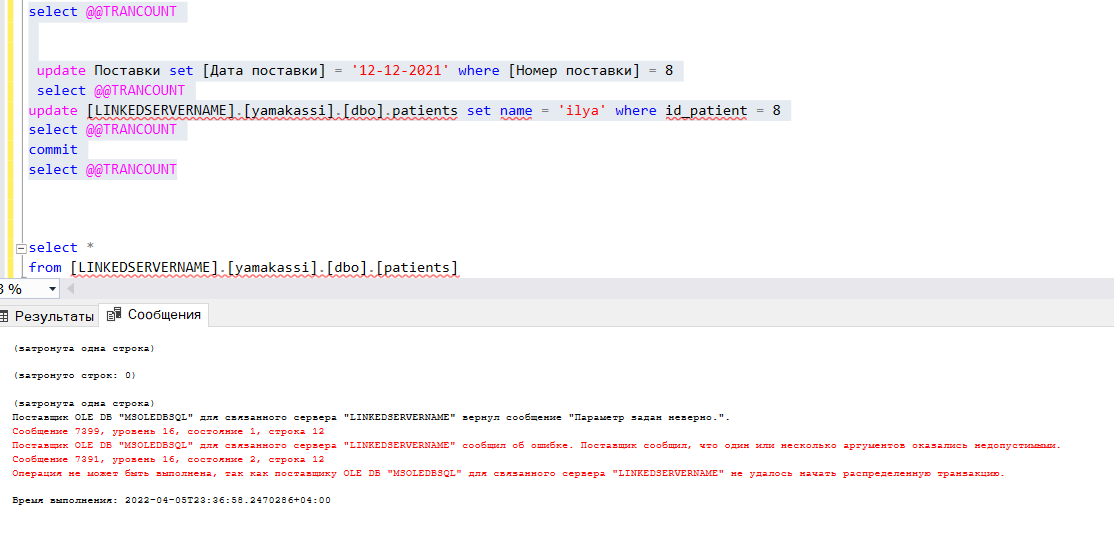


Такая ошибка произошла по причине того, что транзакции выполнили запросы к таблицам, тем самым закрыв доступ к ним

5\_Наглядно продемонстрировать получение информации о текущих транзакциях и блокировках, исходя из данных системного словаря.



6)\* Добавить связанный удалённый сервер и организовать распределённую транзакцию между БД на локальном сервере и БД на привязанном сервере.



**Вопросы к отчёту:**

1. Сущность и предназначение транзакции;
2. Базовые свойства транзакции;
3. Типы транзакций. Режимы подтверждения транзакций сервера;
4. Уровни изолированности транзакций. Read uncommitted;
5. Уровни изолированности транзакций. Read commited;
6. Уровни изолированности транзакций. Repeatable read;
7. Уровни изолированности транзакций. Snapshot;
8. Уровни изолированности транзакций. Serializable;
9. Блокировки. Вид блокировок. Концепция менеджера блокировок;
10. Взаимоблокировка. Примеры;
11. Понятие связанного сервера и распределённого запроса;
12. Структура файлов журналов. Понятие логического и физического журналов, LSN, minLSN, контрольной точки;
13. Распределённые транзакции. Двухфазная фиксация. Компонент DTC;