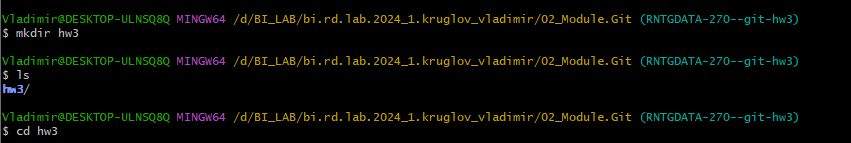
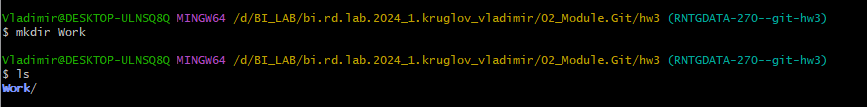
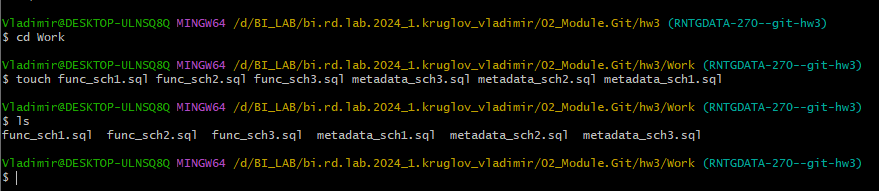
Перейдите в папку «HW3» локального репозитория и создайте в ней папку “Work”.

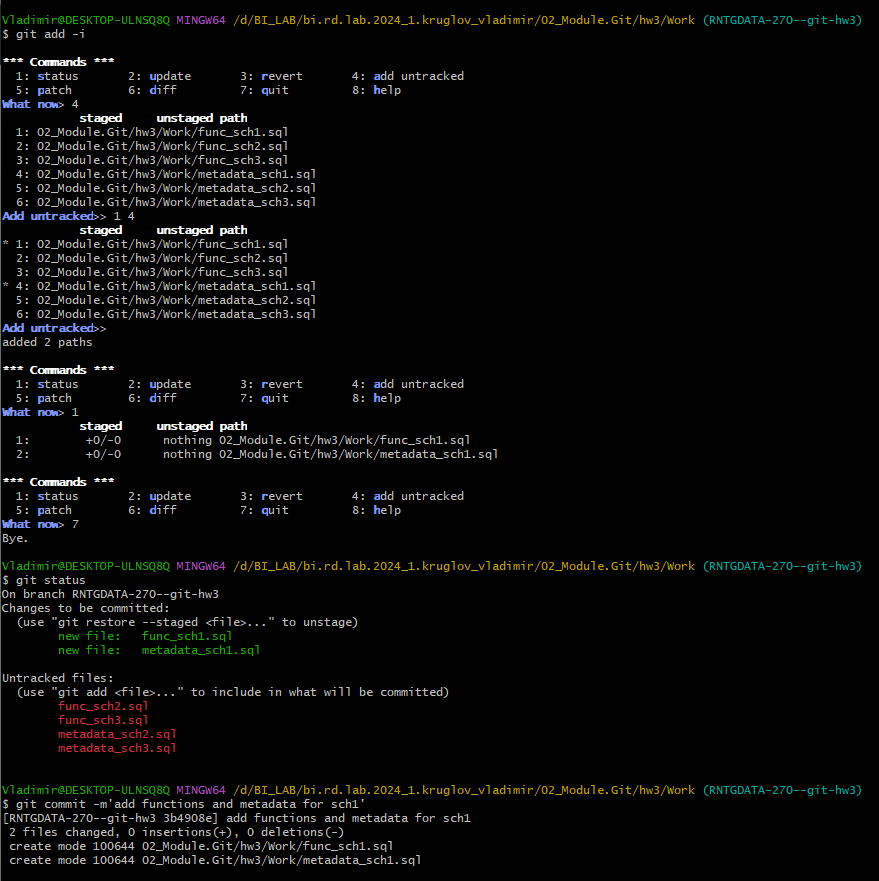


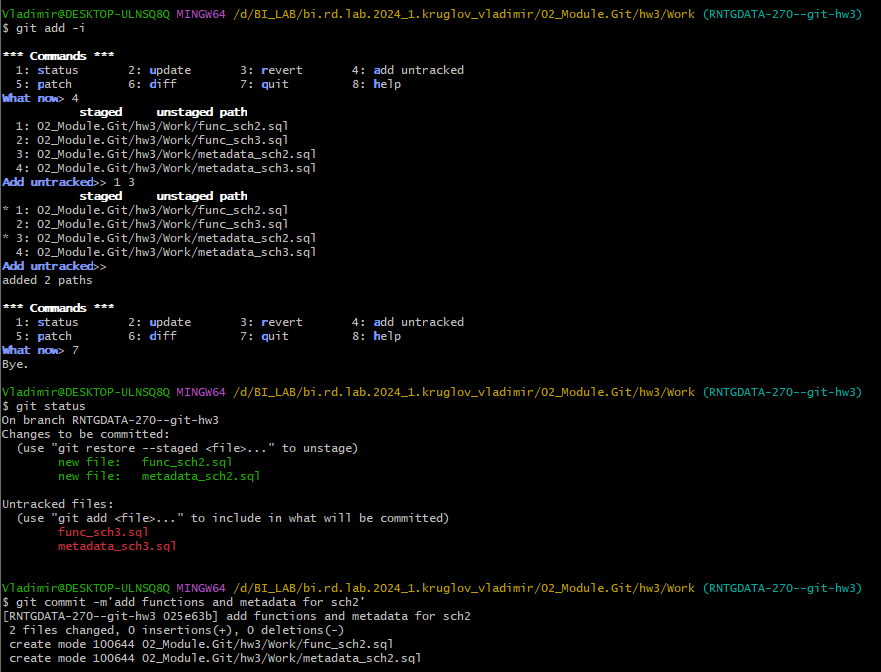


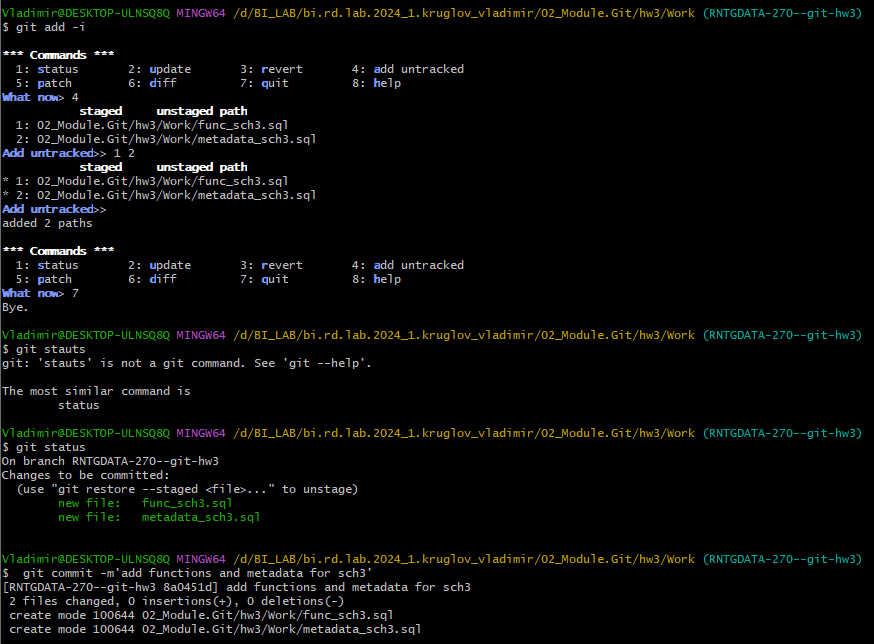
Перейдите в папку “Work” и создайте в ней 6 файлов: “func\_sch1.sql”, “func\_sch2.sql”, “func\_sch3.sql”, “metadata\_sch3.sql”, “metadata\_sch2.sql”, “metadata\_sch1.sql”.



Используя интерактивное индексирование (git add -i) разделите все эти файлы по 3 разным коммитам. Каждый коммит должен содержать файл “func\_” и “metadata\_” для соответствующей схемы (sch1, sch2 и sch3), комментарий к каждому сообщению должен быть “add functions and metadata for название\_схемы”.







Создайте новый файл “func\_ref.txt” и закоммитьте его создание с подходящим комментарием. Скопируйте в файл следующее определение функции:

CREATE [ OR ALTER ] FUNCTION [ schema\_name. ] function\_name( [ { @parameter\_name [ AS ][ type\_schema\_name. ] parameter\_data\_type

[ = default ] [ READONLY ] }

[ ,...n ]

]

)

RETURNS return\_data\_type

[ WITH <function\_option> [ ,...n ] ]

[ AS ]

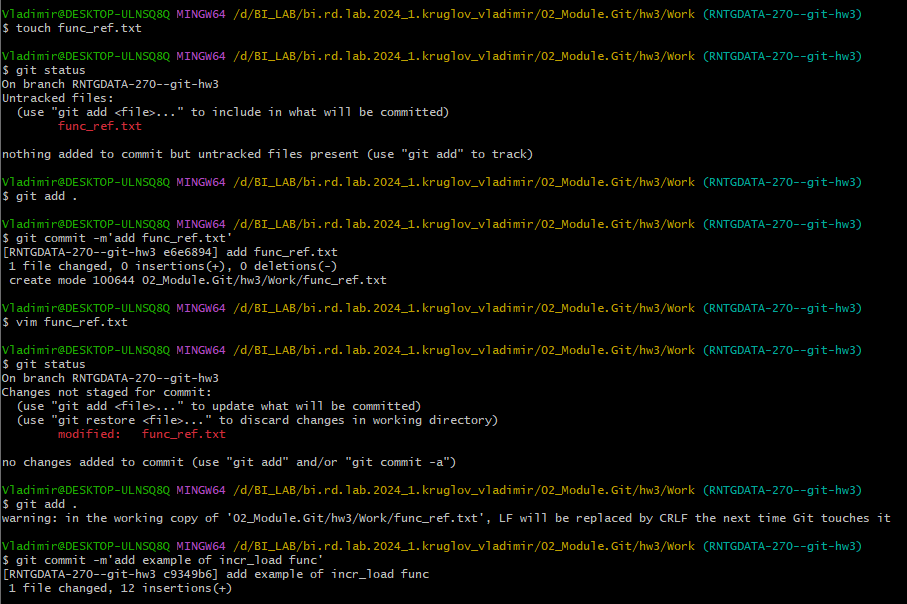
BEGIN

function\_body

RETURN scalar\_expression

END[ ; ]

Сохраните и закоммитьте с комментарием “add example of incr\_load func”.



Упс, эта функция неверно названа. Отредактируйте функцию, чтобы она выглядела таким образом:

CREATE [ OR ALTER ] FUNCTION schema\_name.incr\_load\_table\_name ( [ { @parameter\_name [ AS ][ type\_schema\_name. ] parameter\_data\_type

[ = default ] [ READONLY ] }

[ ,...n ]

]

)

[ WITH <function\_option> [ ,...n ] ]

[ AS ]

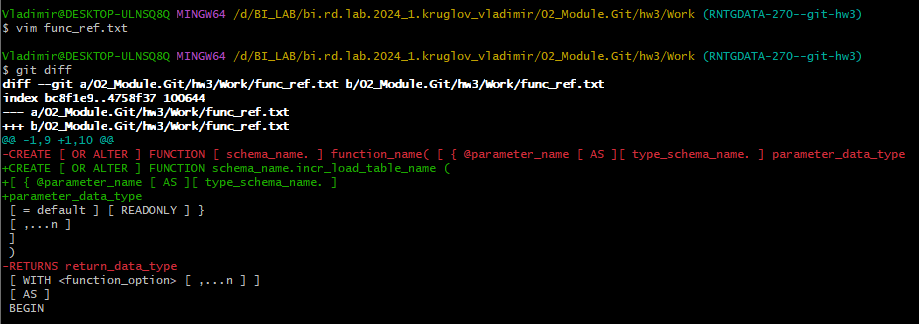
BEGIN

function\_body

RETURN scalar\_expression

END[ ; ]

Сохраните и закройте редактор.

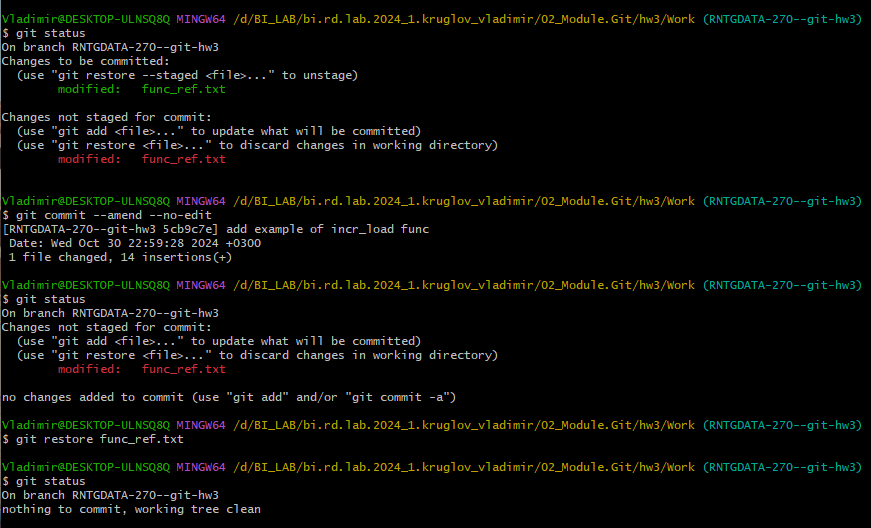


Теперь используйте команду git add -i option patch или git add -p для разделения изменений на фрагменты (hunks) (опция s при выводе сообщения “Stage this hunk [y,n,q,a,d,/,j,J,g,s,e,?]?”); добавьте в индекс изменение первой строки (первый фрагмент) и не добавляйте удаление строки “RETURNS return\_data\_type” (второй фрагмент).

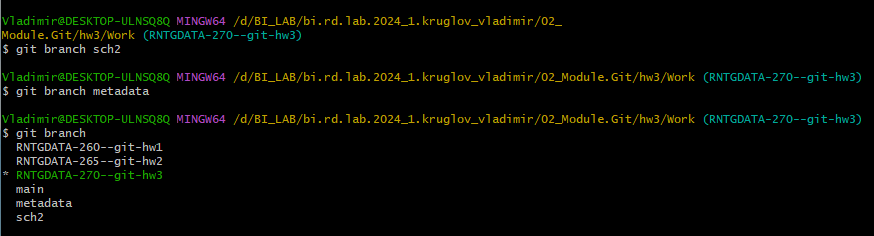


Посмотрите состояние с помощью команды git status.

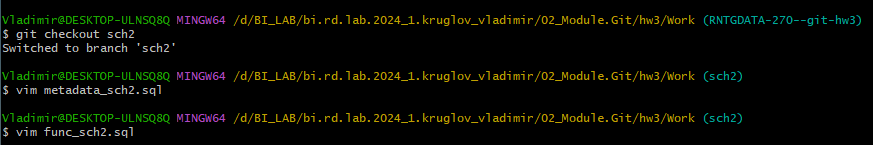
Как закончите, добавьте это изменение (проиндексированный первый фрагмент) в последний коммит используя команду, которую вы уже знаете. После того, как закончите, восстановите файл для удаления ненужных изменений.



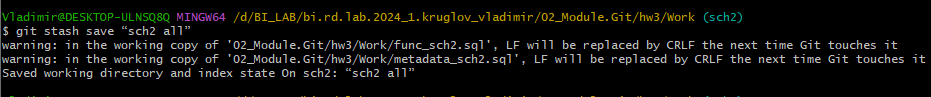
Создайте ветку “sch2” и ветку “metadata” из ветки "{номер задачи в JIRA}--git-hw3".

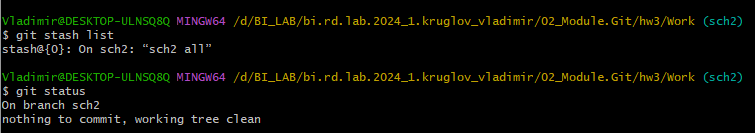


Переключитесь (checkout/switch) на ветку “sch2” и добавьте в файл “metadata\_sch2.sql” строку “added metadata to sch2”, а в файл “func\_sch2.sql” строку “added functions to sch2”.

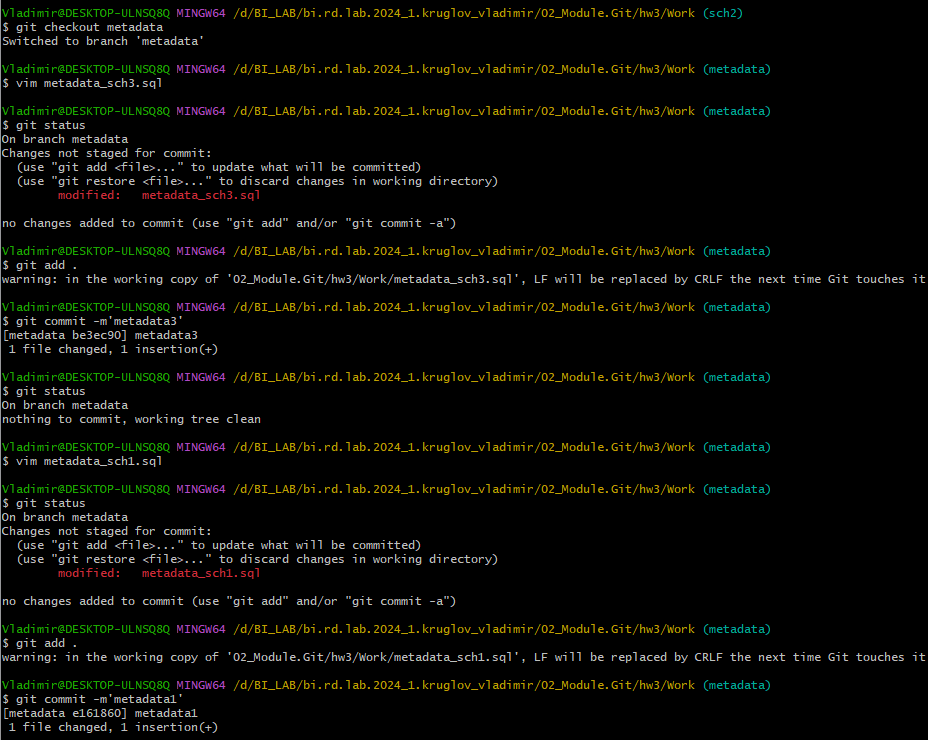


Не добавляйте эти изменения в индекс, вместо это спрячьте (stash) их с комментарием “sch2 all” – используйте для этого команду git stash save “comment”. Посмотрите список ваших ‘спрятанных’ объектов.

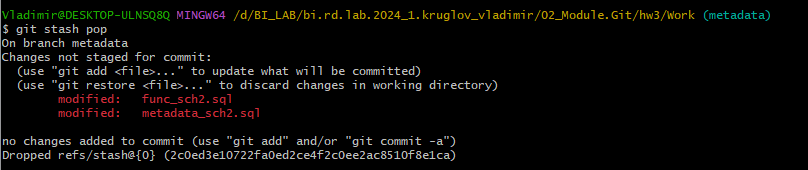




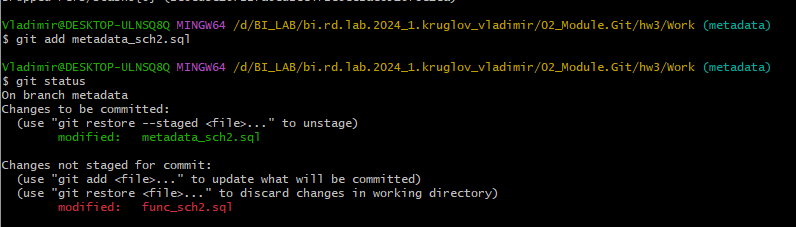
Переключитесь (checkout/switch) на ветку “metadata”. Добавьте в файл “metadata\_sch3.sql” строку “added metadata to sch3” и закоммитьте с комментарием “metadata3”. Добавьте в файл “metadata\_sch1.sql” строку “added metadata to sch1” и закоммитьте его с комментарием “metadata1”.



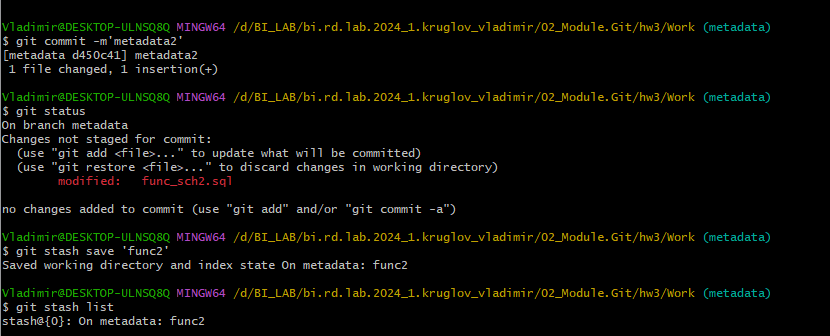
Восстановите спрятанный объект (stash) из ветки “sch2” в ветку “metadata” с удалением спрятанного объекта (stash), используя команду git stash pop.



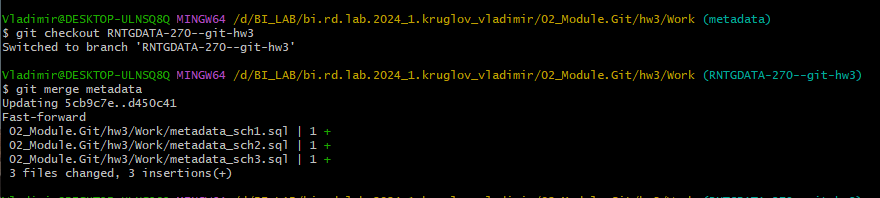
Добавьте в индекс изменения файла “metadata\_sch2.sql.

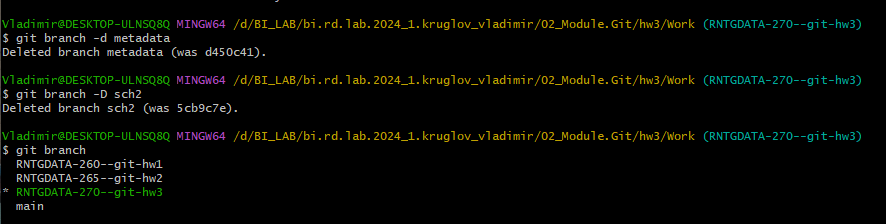


Закоммитьте изменения файла “metadata\_sch2.sql” с комментарием “metadata2”, после этого спрячьте (stash) снова для того, чтобы сохранить изменения в файле “func\_sch2.sql” с комментарием “func2”.



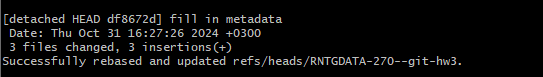
Переключитесь (checkout/switch) на ветку "{номер задачи в JIRA}--git-hw3" и объедините (merge) с веткой “metadata”. После объединения (merge) удалите ветки “metadata” и “sch2”.

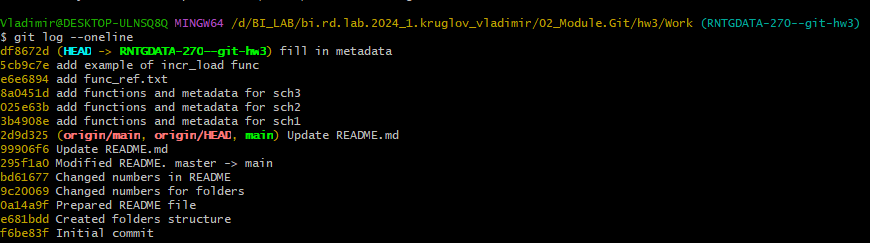




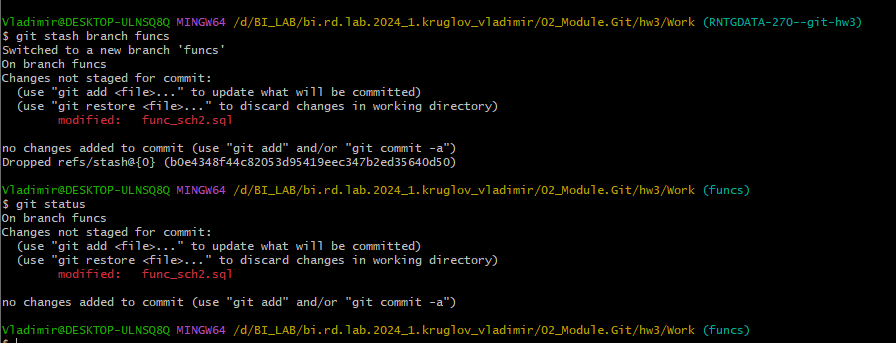
Используйте команду rebase -i для объединения всех коммитов, которые пришли после слияния (merge) в один коммит (последние 3 коммита). Переименуйте этот коммит на “fill in metadata”.



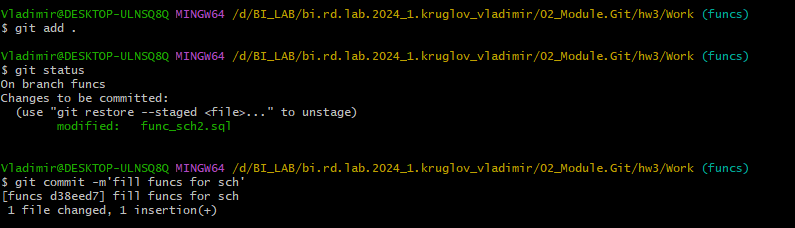




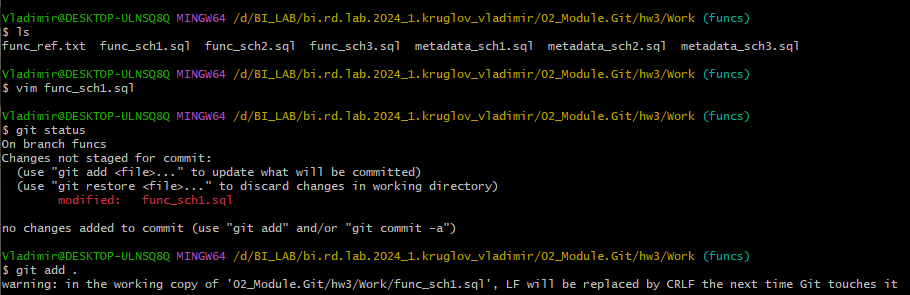
Создайте новую ветку “funcs” из спрятанных объектов (stash), которые вы сохраняли в ветке “metadata” в п.5.

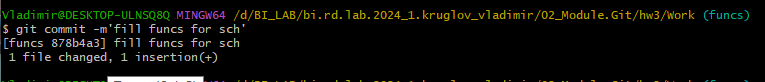


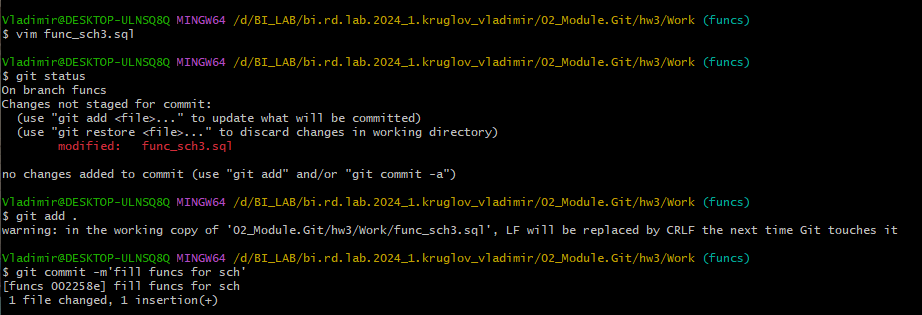
Переключитесь (checkout/switch) на ветку “funcs”. Добавьте изменения в индекс и закоммитьте изменения в файле “func\_sch2.sql” с комментарием “fill funcs for sch”.



Сделайте те же изменения, что вы делали в файле “func\_sch2.sql” в других 2 func файлах и закоммитьте в отдельных коммитах с комментариями, аналогичными предыдущему.

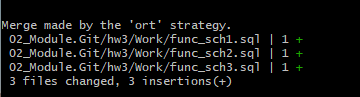


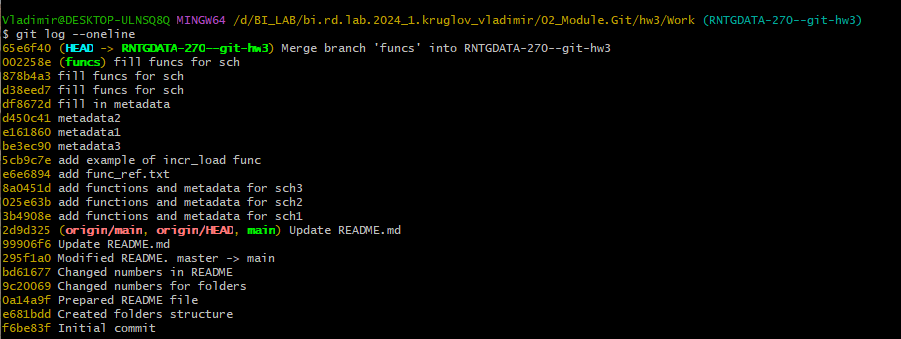


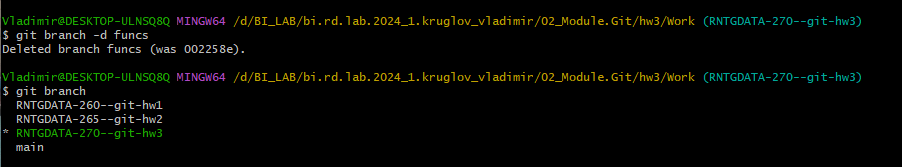


Переключитесь (checkout/switch) на ветку "{номер задачи в JIRA}--git-hw3" и объедините (merge) с веткой “funcs”. Удалите ветку “funcs”.

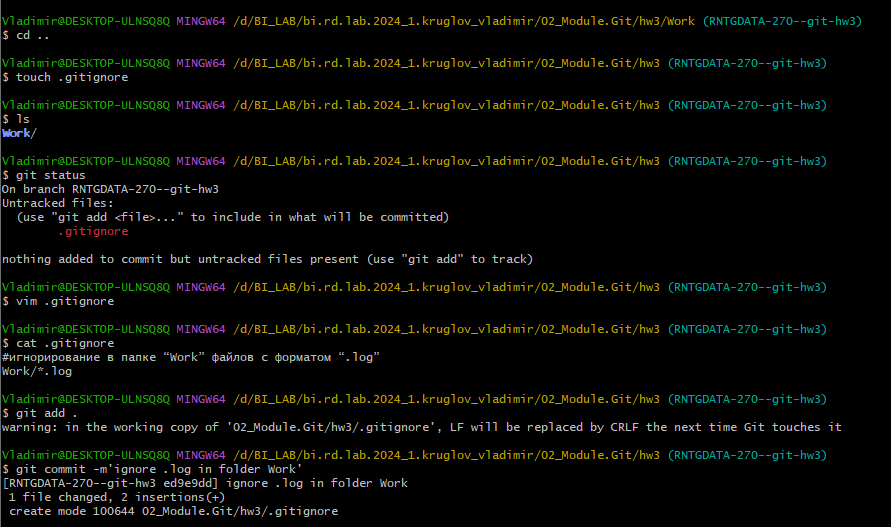
  

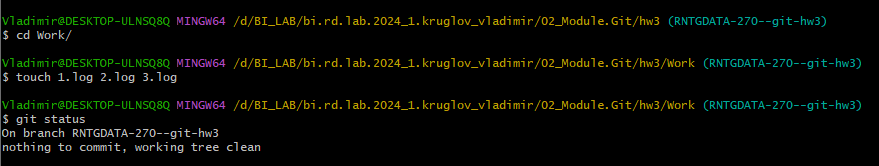


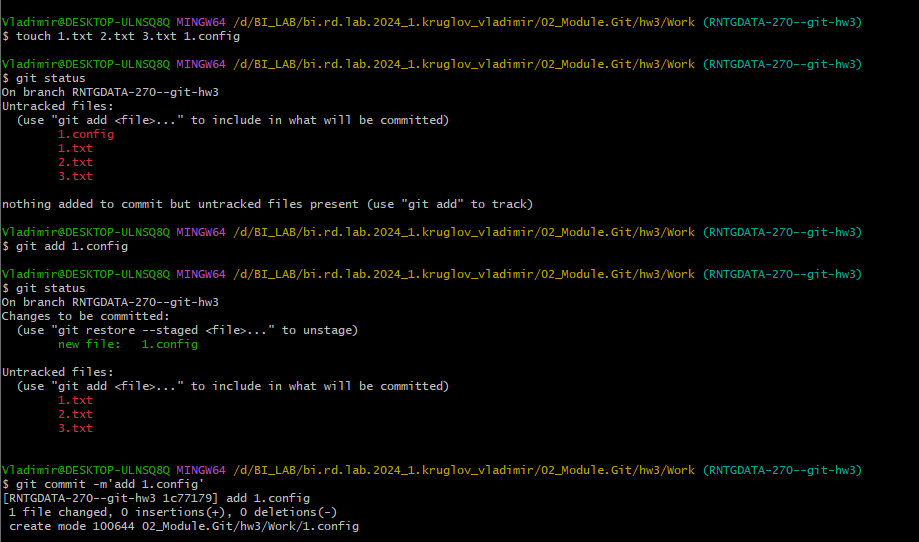


Перейдите в основную папку локального репозитория и настройте игнорирование в папке “Work” файлов с форматом “.log” – закоммитьте с подходящим комментарием.

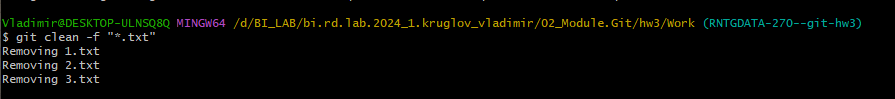


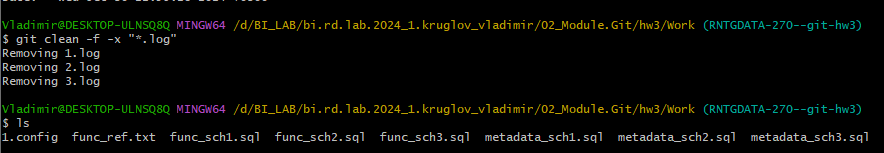
Перейдите в папку “Work” и создайте несколько файлов с форматом “.log” (1.log, 2.log, 3.log), форматом “.txt” (1.txt, 2.txt, 3.txt), и 1 файл – “1.config”. Добавьте файл “1.config” в индекс и закоммитьте его.





Используйте команду git clean с нужными параметрами для удаления всех “.txt” файлов, после этого используйте ее для удаления всех “.log” файлов.





Используйте команду git reset с нужным параметром для отмены фиксации (committing) и индексирования файла “1.config” и очисткой директории со спрятанным объектами (stashing) одной командой.

