Contents

[1 Overview 2](#_Toc530154869)

[2 Details – Test Case – Code Fix 2](#_Toc530154870)

[3 Details – Test Case – Data Fix 3](#_Toc530154871)

[4 Details – Setting up for Ticket Testing 7](#_Toc530154872)

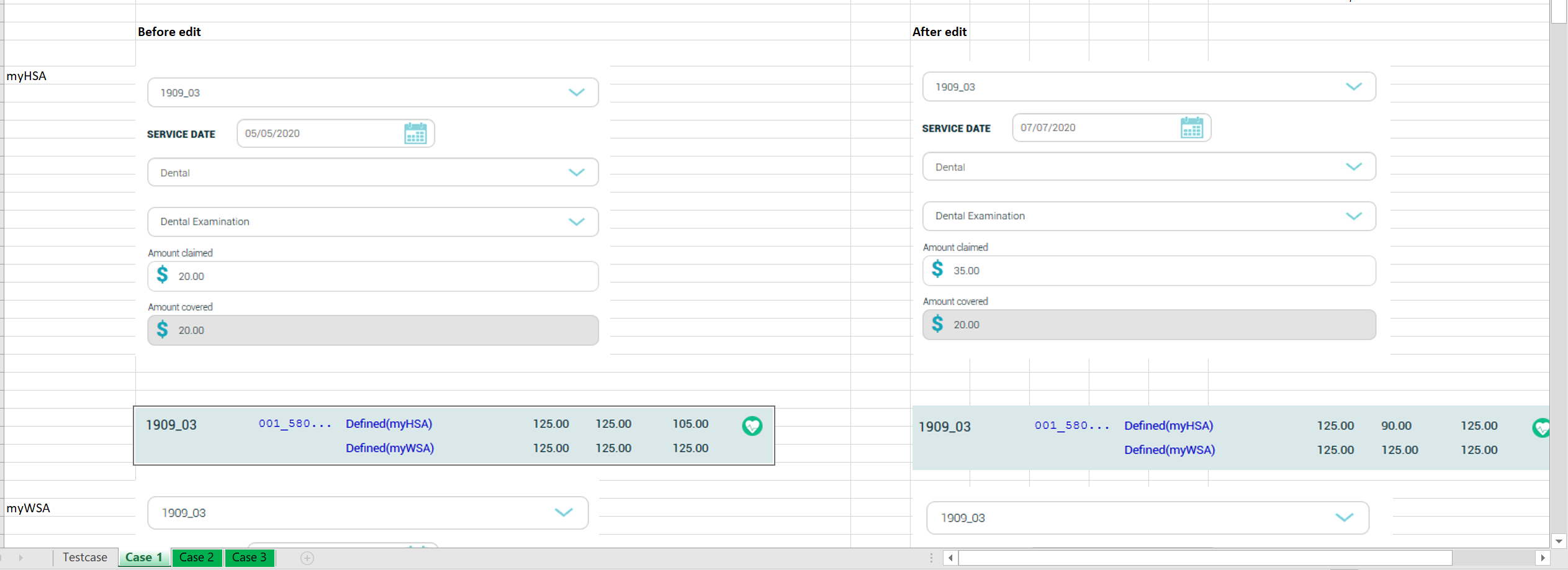
# Details for Meeting on November 19th

# Overview

This is a document we can work through; it provides an overview of what we’ll cover and during the actual meeting we’ll go into a bit more detail and touch on any questions that arise.

# Details – Test Case – Code Fix

Start by looking at one of the more involved tickets as far as testing is concerned. [5802 (Balance Problem - The service date started out in the previous period, and was subsequently updated to the current renewal period)](https://myhsa1.zendesk.com/agent/tickets/5802). Take a look at the Excel TestDoc\5802\_BalanceUpdate\_v(2).xlsx. Notably, testing this ticket is more involved and the tester here moved here date forward so that reserve could be used. Understanding this particular ticket in detail at this point is not really necessary, but in this example the tester shows what happened before and after the edit to the balances. The way the date and amount was changed caused us to expect the reserve that was originally taken out to be put back into the reserve on the update and instead pulled from carryover. Prior to the fix, the amounts were not correct and in some cases did not balance.



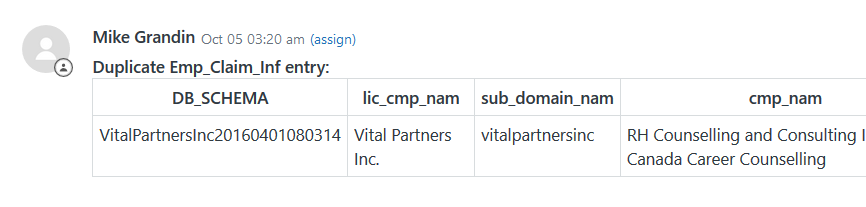
Most tickets are quite a bit simpler. [8744 (Super admin pop up message error grammar)](https://myhsa1.zendesk.com/agent/tickets/8744) is an example of one with a very basic test case. See TestDoc\8744\_v.1.0.xlsx.

# Details – Test Case – Data Fix

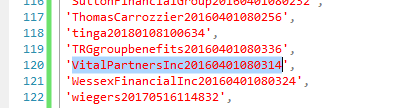
We often run into issues where a data fix is required. Sometimes the level of testing is not that great, especially if we’ve seen the issue before. One recent example is [8730](https://myhsa1.zendesk.com/agent/tickets/8730), which had a data fix applied as we’ve seen this a few times before but the code fix remains outstanding. However, an example where more testing was required can be seen under [8653 (Data Integrity Check - Duplicate Emp\_Claim\_Inf entry)](https://myhsa1.zendesk.com/agent/tickets/8653). In this case, the testing was a bit more thorough as we just verified this specific user could setup a claim. So, I want to go over the process of how we get our test environment setup to handle this particular case.

### First, if you haven’t reviewed “Supportdoc\SetupDevelopmentToRunMultipleSubdomains-v1.0.docx” let’s just briefly go over that. Once you are running with IIS, the first step is to grab a recent version of the database if you do not already have that.

### We see from the ticket the issue relates to this database and subdomain:

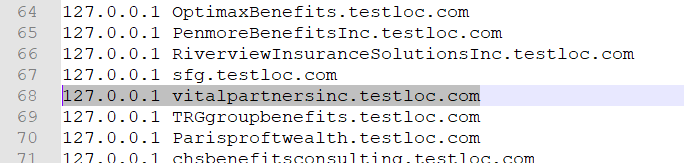


### The way I typically do this is to login to the production server and open the following script: “E:\myHSA\_all\_DB\PostRolloutBackupForTest\ PostRolloutBackupsForTest.sql”. Scroll down and see if the database is in the list (this is a list of frequently used databases). In this case, it’s already there:

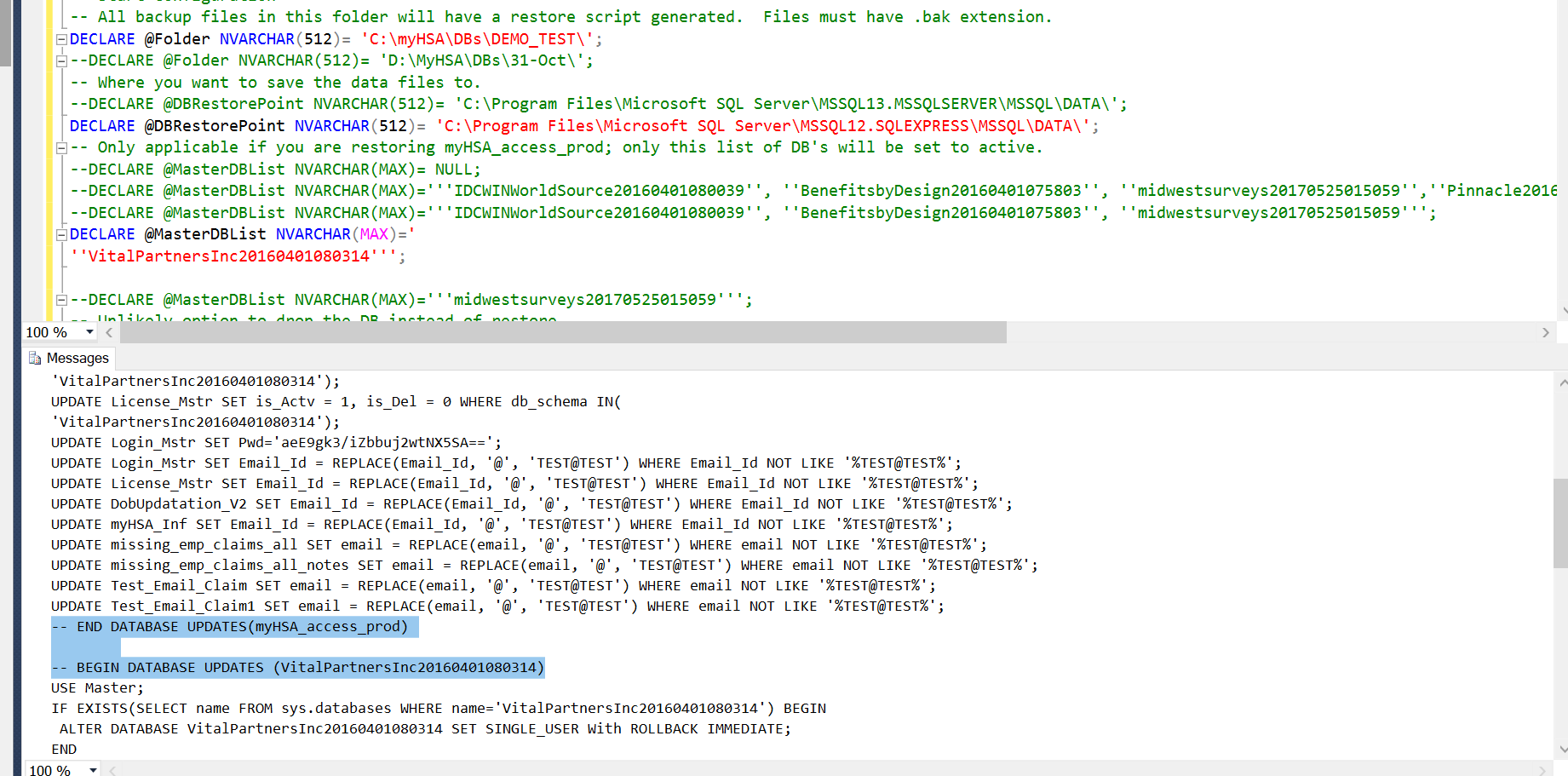


### Run the script. It’ll backup all the databases in the list, but you can just grab what you need which will be put into a folder such as “E:\myHSA\_all\_DB\PostRolloutBackupForTest\16-Nov”. Typically, I just 7z the database I need and copy that to my local. Often, I’ll also just 7z the whole set and copy and set that up later so I have a full updated set of commonly used databases to work with. Also of note is you may need to grab the myhsa\_access\_prod database, if you haven’t updated it recently. Let’s assume that is the case so you get both myhsa\_acess\_prod and navigate20160620061743\_\*.BAK to your local environment. I have placed these databases in “E:\myHSA\_all\_DB\PostRolloutBackupForTest\ DEMO\_TEST.7z” on the production server so you can try this at a later time if you want as well.

### Make sure your hosts file is up to date. It should have this line:



### Now, you need to setup this database on your local environment. The script that we need to run is under \proc\_subdomain\dev\_scripts\ GenerateTestDatabasesRestoreScript.sql. I have updated to include only the database we are looking to test.

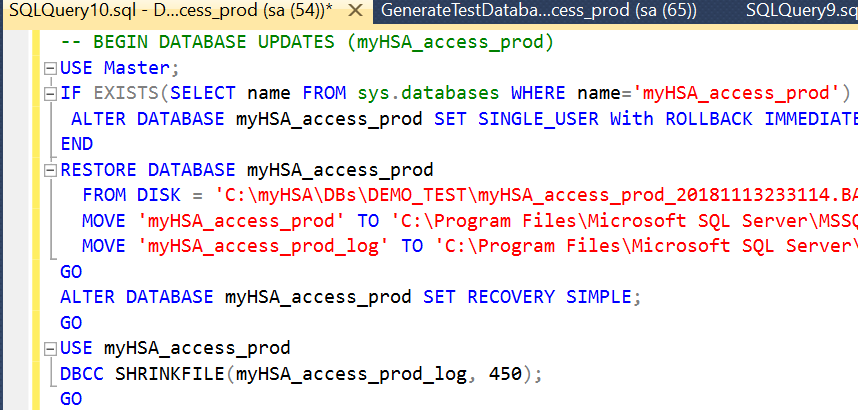


#### @Folder is the location of your database backups

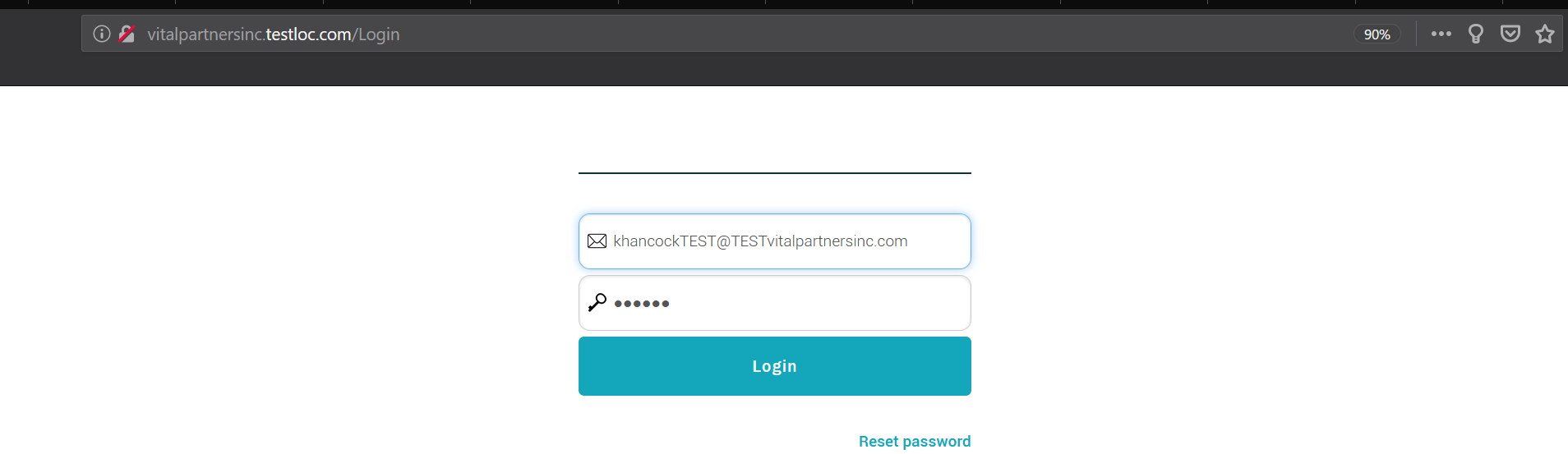
#### @DBRestorePoint is where your MS SQL backups are located.

#### @MasterDBList is the list of databases that will be set as active. In this example then, only the navigate database will be active on your local environment after running this.

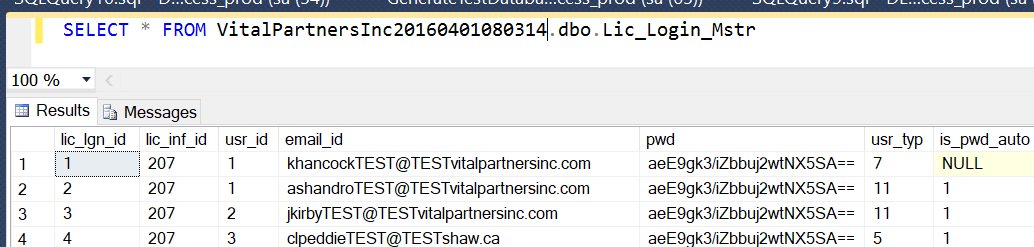
### You need to copy the results of the above into another window, and execute it. This will restore the database.



### Once that completes, you can pull up the site on your local environment and login:

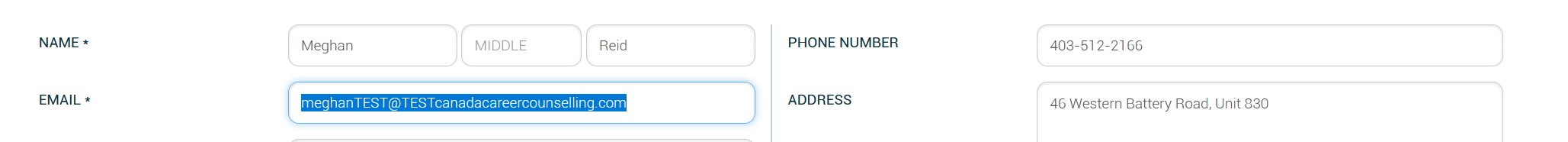


#### We get the user by querying Lic\_Login\_Mstr on the database as such:



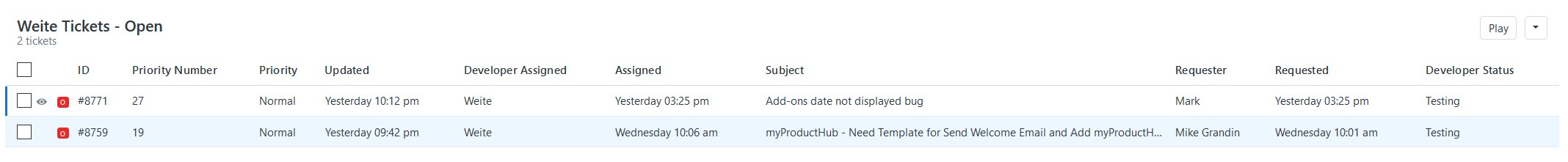
### For test databases, notice that the email\_id is modified (TEST@TEST) and the password is set to 123456. This is rather important that everyone restores databases using this script, as we do not want to accidently fire an email to a user during testing.

### We’ll look at the script to apply the ticket updates to the database in more detail in the next section. Assuming the ticket update script was applied, you’d then be at a point where the data fix could be verified. You could find that employee and login as them.

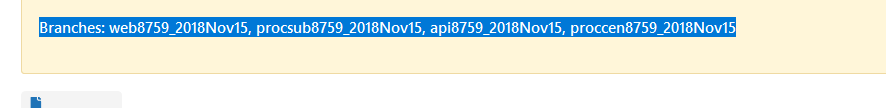


# Details – Setting up for Ticket Testing

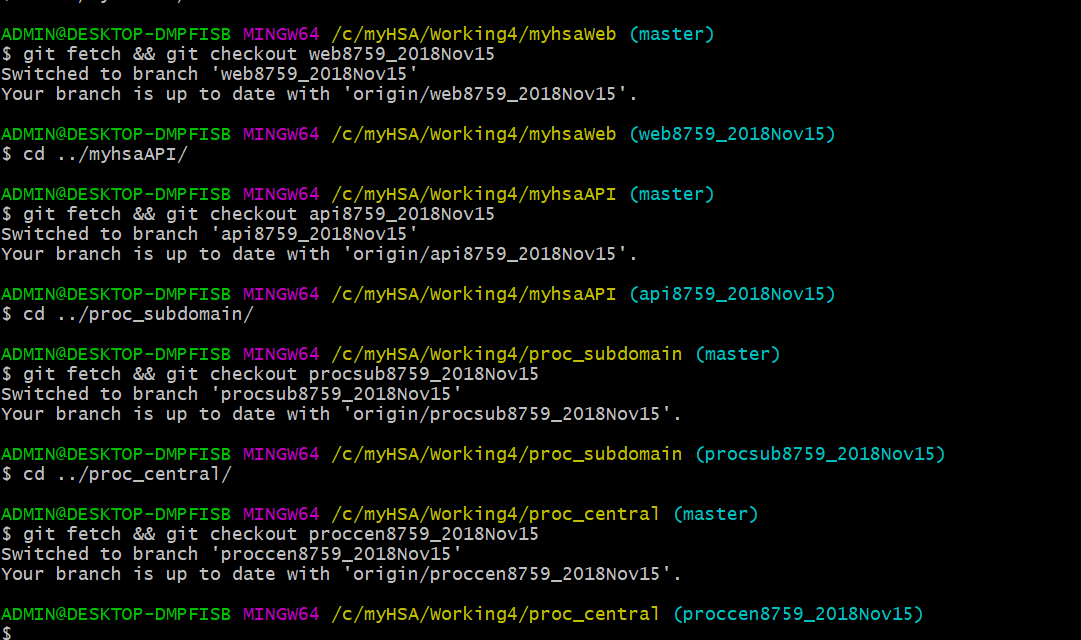
Let’s take a look at a recent ticket. Notably, the scripts we use later take a comparison against the master branch, so it’s best to look at a ticket that is current rather than historical. This particular ticket is also good as it touched the web, api, proc\_subdomain, and proc\_central repositories. In fact, this ticket also touched the Job Executor so we can touch on that as well. Open up [8759 (myProductHub - Need Template for Send Welcome Email and Add myProductHub Add Plan Admin auto create logic)](https://myhsa1.zendesk.com/agent/tickets/8759).



We find and see the branches:

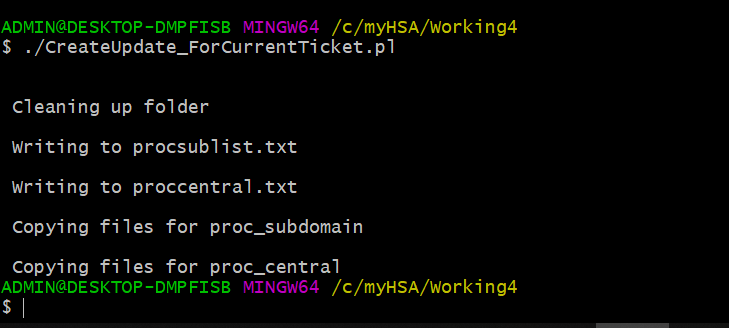


Next, open git bash and checkout these branches. So we have:

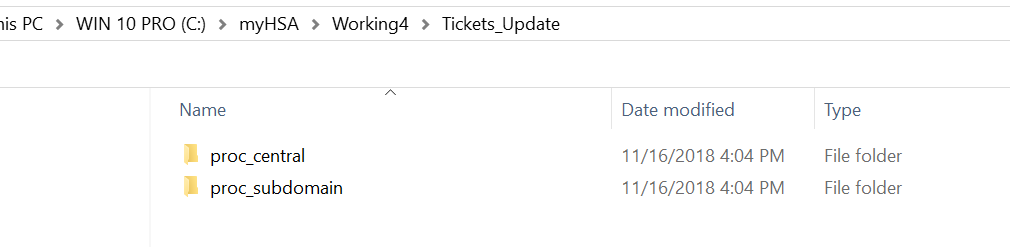


Copy “\Git-Bash-Scripts\CreateUpdate\_ForCurrentTicket.pl” to your working folder.

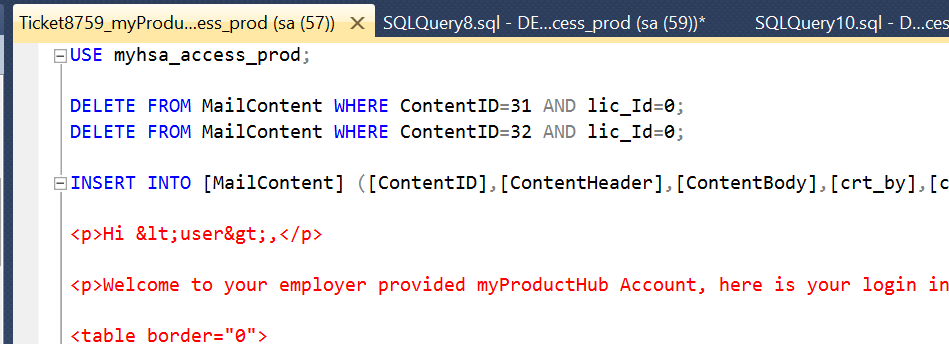
Run it – what this does is copy all the SQL update scripts you need to get your environment up to date with the ticket to another folder (it just works out what you need which is quite helpful):



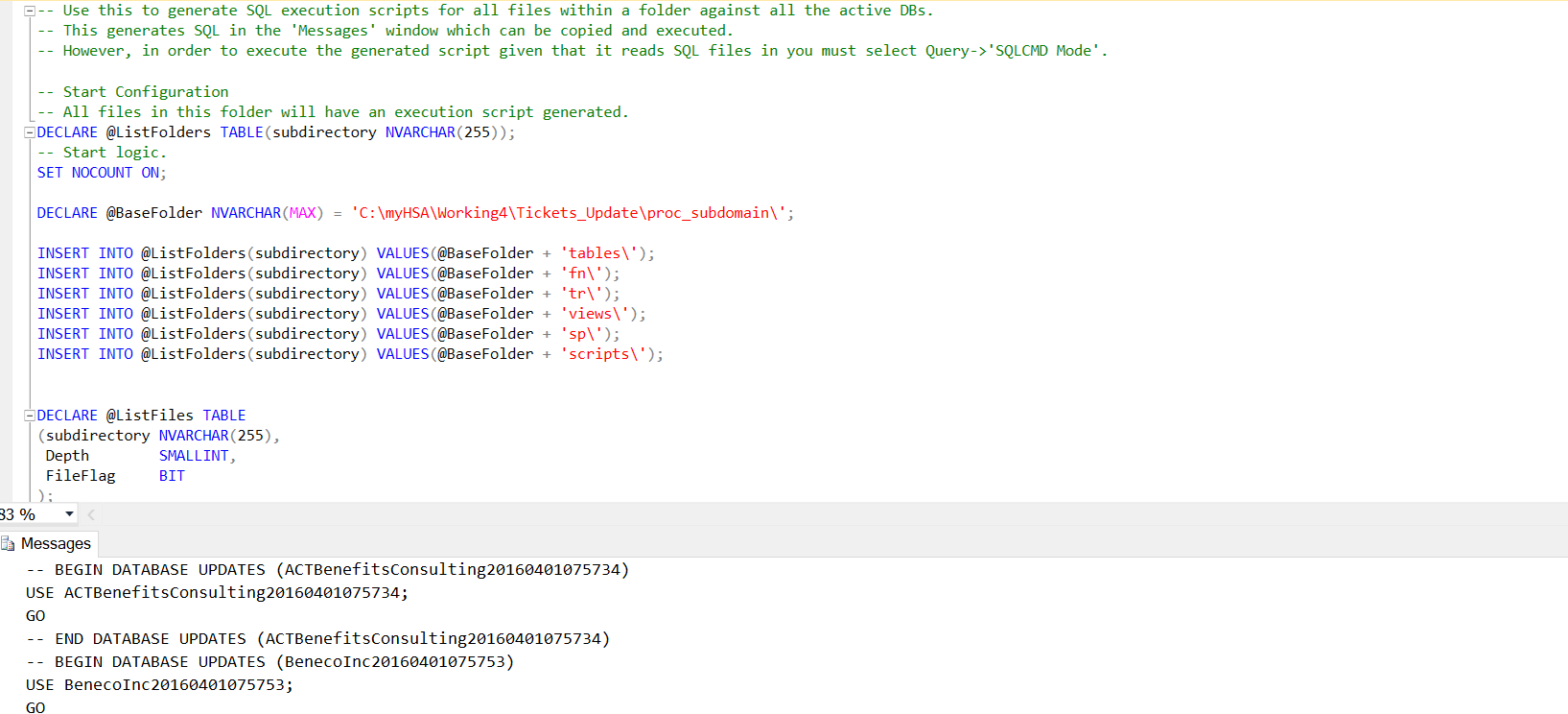
Now, we check the folder it created:



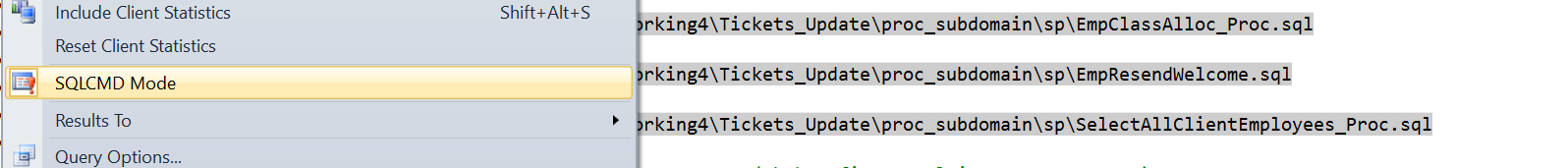
These are the scripts we need to run to get our databases in sync with the ticket. Go to the proc\_central database first, and manually run the scripts on your myhsa\_access\_prod database one by one. Here is the first one:



Now, for the proc\_subdomain, use the \proc\_subdomain\dev\_scripts\ GenerateTestDatabasesRestoreScript.sql script.

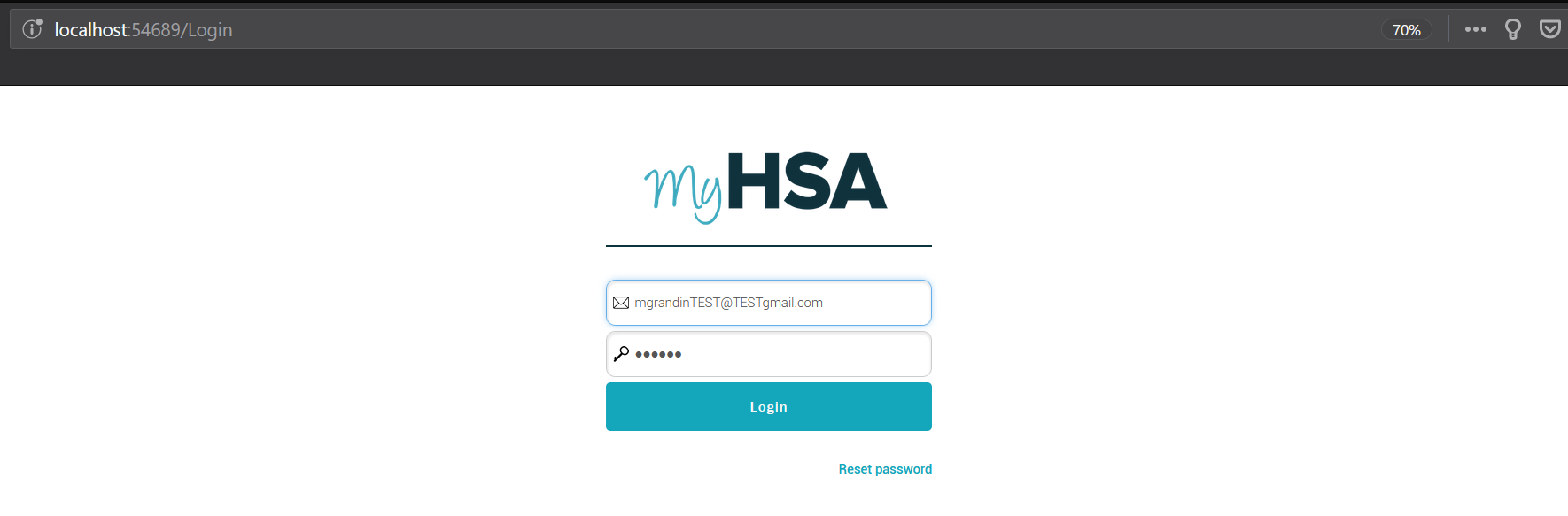


You need to copy the “Messages” to another query window. By default, it generates the update script for all your active database. In order to run it, you need to first turn on SQLCMD Mode.

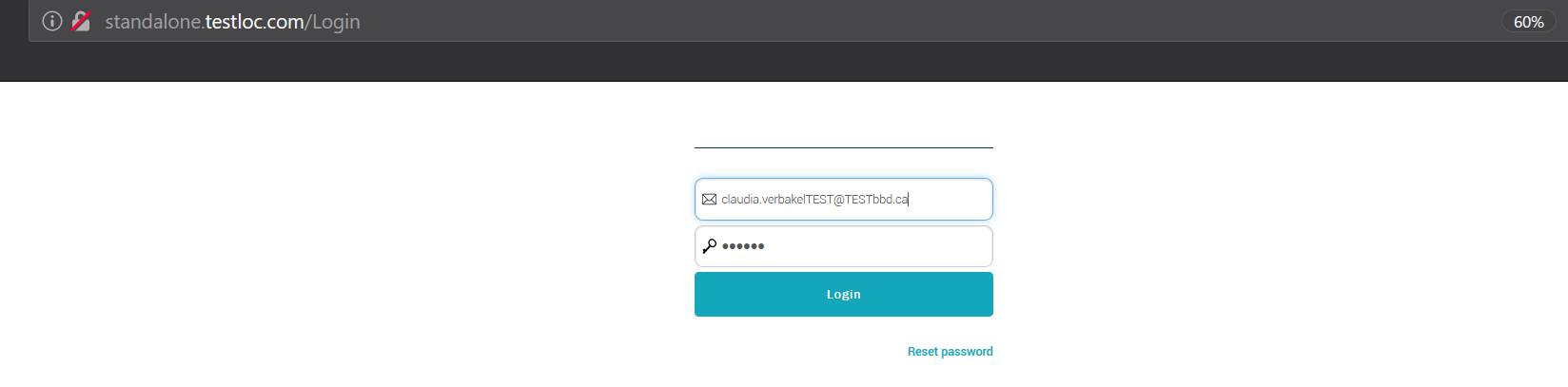


Now, that your databases are up to date you can build the API and Web layers and go ahead with the tests.

### Run the web project in Debug mode (superadmin testing):



### Go to our testloc sites to test the advisor side:

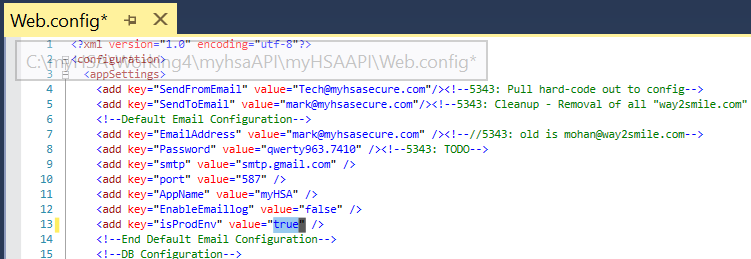


Other notes on this process:

### I will use the “GenerateTestDatabasesRestoreScript.sql” and “GenerateExecuteSQLScriptsForAllActiveDBs.sql” fairly frequently when testing tickets just to ensure I’m properly in sync. Often times, we need to have a clean setup and ensure past ticket database updates do not overlap or cause a problem with the next ticket we test. We don’t always need a clean database restore between ticket testing and certainly a lot of this is dependent on whether the tickets may have overlap in the database updates. Certainly, many tickets will not have overlap so restoring a clean database is not always necessary but if there is some uncertainty it’s better to just go with a clean restore to be sure.

API Email update

### It’s worth noting here that if you want emails to work on your local environment, you must update the API config option as follows:



### The default on this is “false”. You should really only have this on when you are testing an email feature, as there are emails that will get fired off. Here are some of those other email config options:

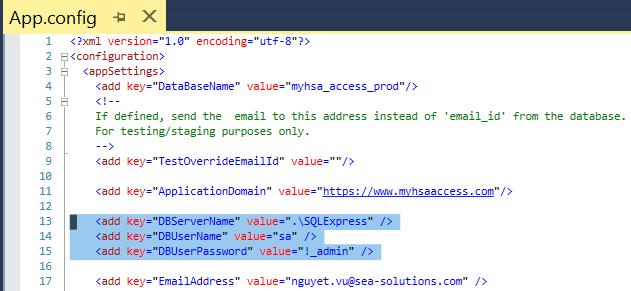


### Nothing here would be highly problematic. Just if you left it on and did not adjust these keys as well, I’d get an email for instance every time you added a company. Additionally, Mark may get emails if you were playing with the account move logic. That’s certainly not a big problem but it’s best to just turn it on when needed to avoid unnecessary emails.

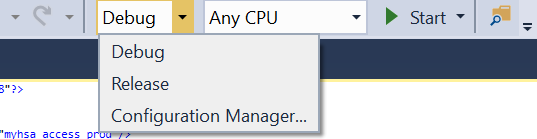
Job Executor

### Open the solution in \myhsaWeb\myHSAJobExecutor.

### The app.config likely needs update for your database:

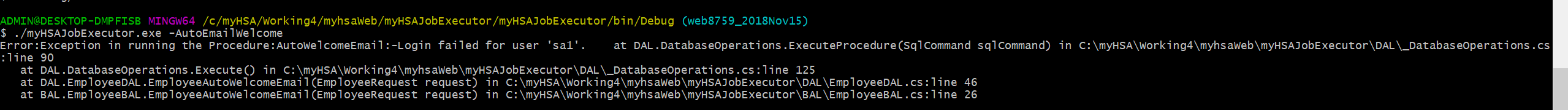


### You may want to change it to debug mode:



### Build the solution

### Now, you can run it. I intentionally setup my database config incorrectly so this would trigger an error:



### There are many switches for the job executor. If and when we update them, we’ll certainly provide further details but as this ticket touched on this logic, I thought it was worth noting here the general setup and execution for this.