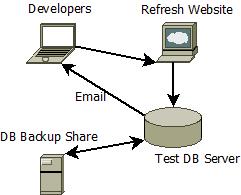
**A Self-Serve System to Refresh Databases - Part 1**

[Ben Kubicek](https://www.sqlservercentral.com/author/bkubicek), 2019-05-02

FROM: <https://www.sqlservercentral.com/articles/a-self-serve-system-to-refresh-databases-part-1>

I am not sure I could count the number of times I have restored a backup from production to a test or dev server. It has happened enough times I finally wrote a system so the developers could do it themselves. This series of articles will review this system in detail. In this article (Part 1), I will give an overview of the design and the technology used to implement this system. In Part 2 I will go in depth on the setup of the database side. Finally, in Part 3 I will describe in detail the web portion of the refresh system. My hope is that you will find some or all of what I have done helpful in your own final solution to automate production refreshes on your test systems.

I have found throughout my career that there is often a need to restore a production backup to a test server. This process can be time consuming, when there are other things you probably should be focusing your attention on. I started thinking about building a system that would allow developers to restore databases themselves without having to actually give them security rights to do so. One of the things I needed to consider is that multiple developers could be using the same database in test, so there needed to be some way to alert all the developers that a refresh was about to happen.



The solution I came up with would use a web site front end (internal IIS server) to allow developers to select which database they wanted to refresh. At my current company we have 40+ manufacturing plants. Each plant has its own test server. The plants have multiple databases that support different systems. So the user would select which location (plant) they want to do a refresh for and then they would see a list of database that could be refreshed in a grid. Then they could update a row in the grid and that database would be marked for restore in test.

As soon as the user marks a database as needing a refresh an email is sent out to the entire developer group letting them know such and such database will be refreshed in 30 minutes. This gives other developers the opportunity to finish whatever they might be doing, or in some cases stop the refresh from happening if they are in the middle of something and need more time to finish.

There is database, called DBCopy, that exists on each of the test database servers. Its only purpose is to help in the process of restoring databases. It has a single table that contains all the information necessary to restore a database to test. This table contains the data that is displayed in the grid when a plant is selected from the drop down on the web site. The table has a needprocess column that is set to 1 when the user marks the database and needing a refresh via the website. It also contains a processtime column. That column is auto updated to 30 minutes in the future when the row is updated.

There is a SQL Agent job that runs once a minute and checks the table for rows that have needprocess set to 1 and has a processtime that is in the past. When a record is found, the restore database procedure is called. Once the database is successfully restored an email is sent out to the developer email group. If the backup fails for some reason and email is sent to a DBA to investigate.

There are some things to note about this system. First, it is entirely based off of backups existing on a file share. The process does not initiate a backup. Since it is assumed this is running on a test server, some liberties are taken. One example is database mail is turned on. Another is extended stored procedures are turned on so the restore procedure can query the backup folder and get the latest backup. These are things that you may not want to have turned on in a production environment because of security concerns.

In this case since it is assumed a test server is being used, there is less risk with having these options enabled. Also, in my situation, all the data is manufacturing related so we don’t have any need to alter data to protect the innocent. This also means things like Always Encrypted is not being used. I believe these issues can be overcome with this solution, but extra steps would be needed to properly deal with them.

Stay tuned for Part 2 of this series as I will go into detail of the design and implementation of the DBCopy database and sql job that runs the refresh system.

[TestRefreshScripts](https://www.sqlservercentral.com/wp-content/uploads/2019/04/TestRefreshScripts.zip)