

Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

Location for activities:

Local system for access to:

<http://jenkins.██████████.local>

<http://octopus.██████████.local>

<https://bitbucket.org/██████████/workspace/overview>

Remote Desktop Protocol for access to: redgate.██████████.local

You must use the redgate.██████████.local RDP machine to perform the following functions.

RedGate Operations:

Start by RDP to redgate.██████████.local

Open Visual Studio on the RedGate machine

Select Clone Repository, enter your repo location (for this example we will use the actual location for ██████DB)

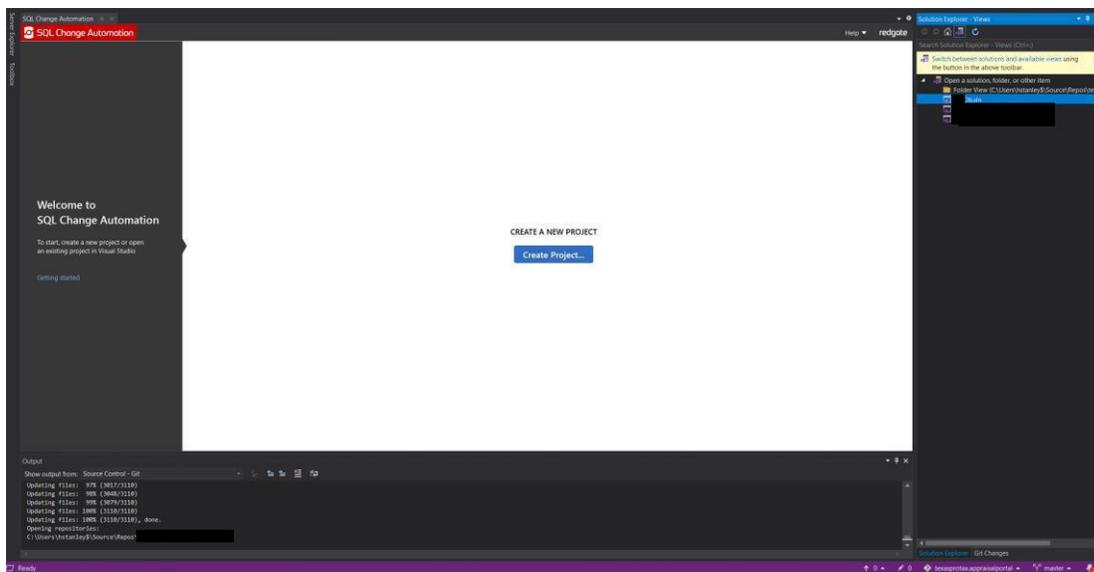
<https://██████████@bitbucket.org/██████████/██████████.git>

Click the Clone button.

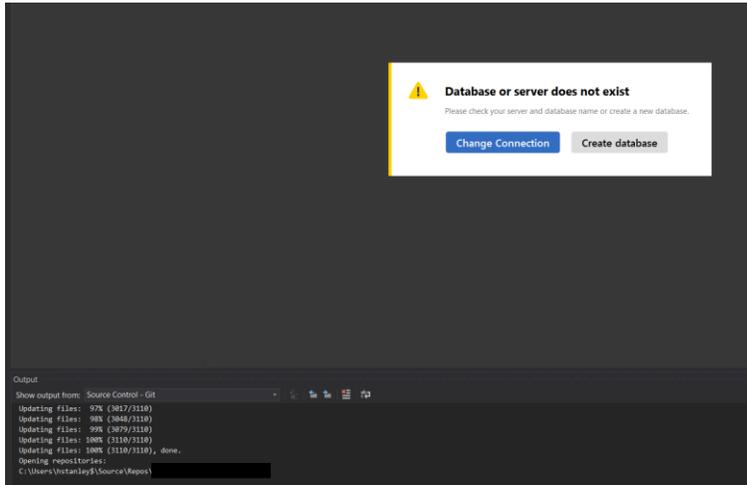
Depending on how your system is setup and which plugin you use for Bitbucket, click the sign in with web browser.

You should now be presented with this screen:

Select the ██████Db.sln to open the project.



You may be presented with the following message:



Click Change Connection,

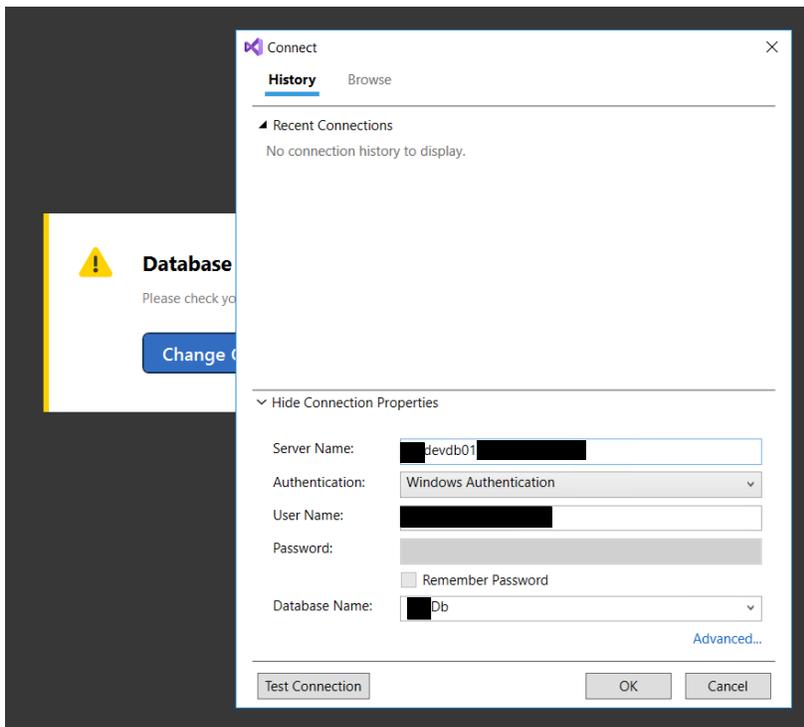
Server Name should be: [redacted]devdb01.[redacted].local

Authentication should be: Windows Authentication

Username should be your username

Database should be [redacted]Db

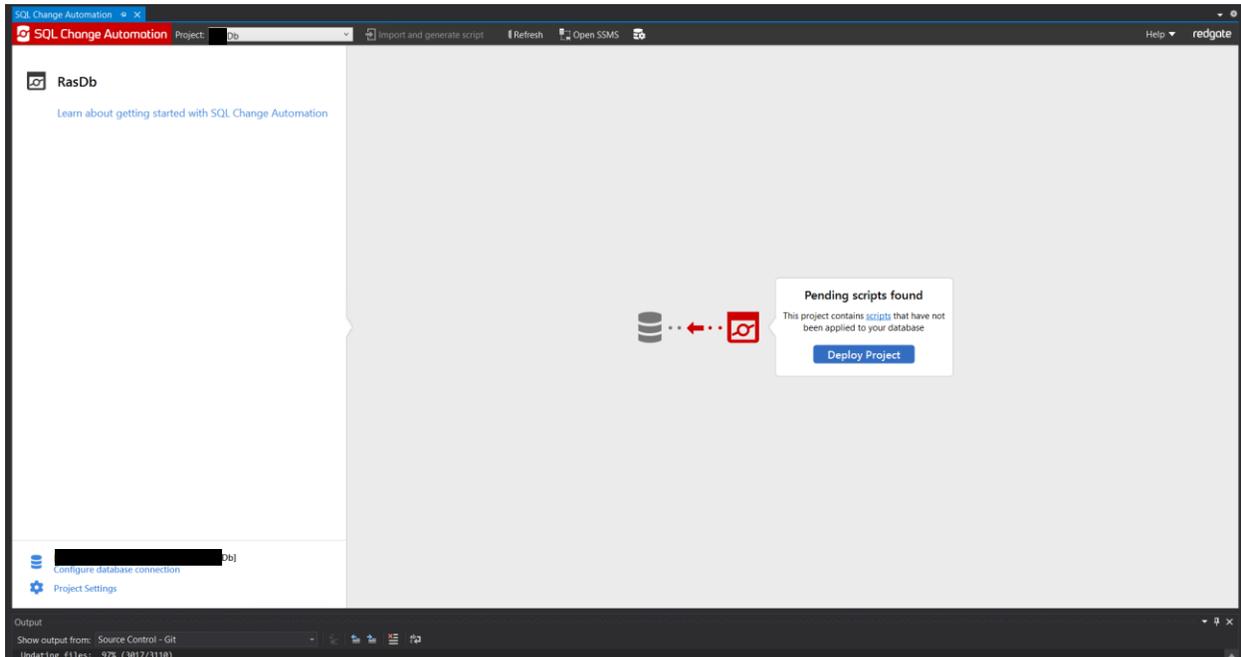
Click Test Connection to verify



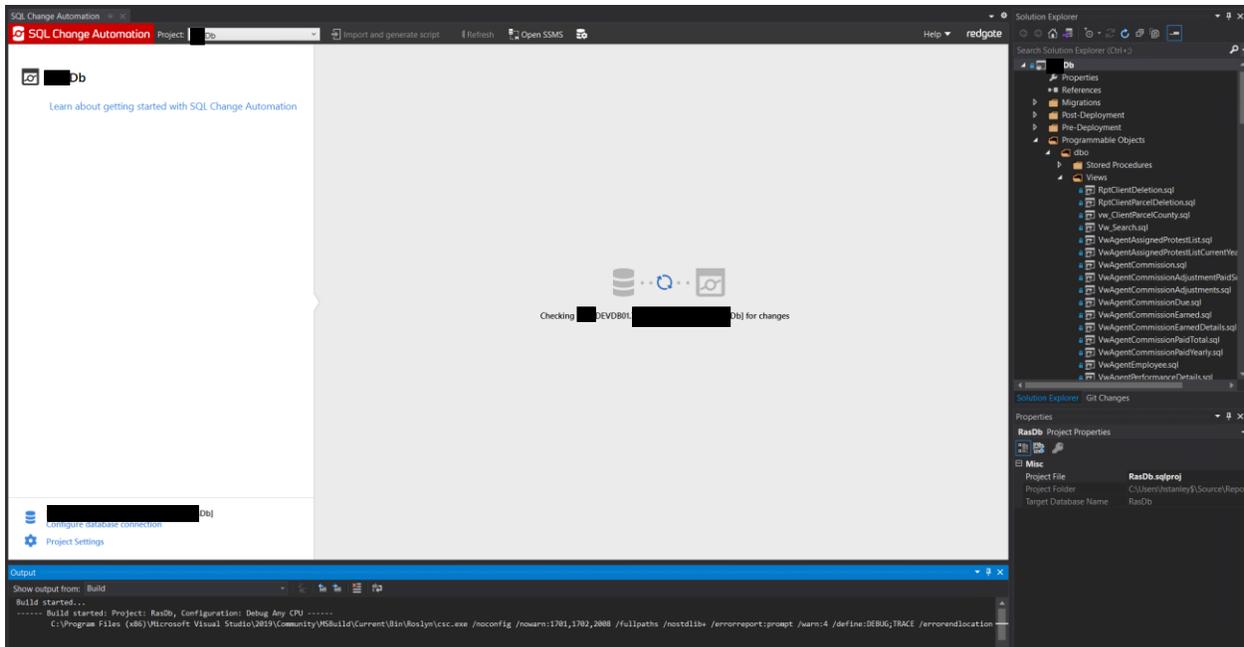
Click OK,

Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

You may be presented with the following screen indicating that there are scripts that have not yet been run on the server but are in the project. They are located in the Error List towards the bottom of the screen:

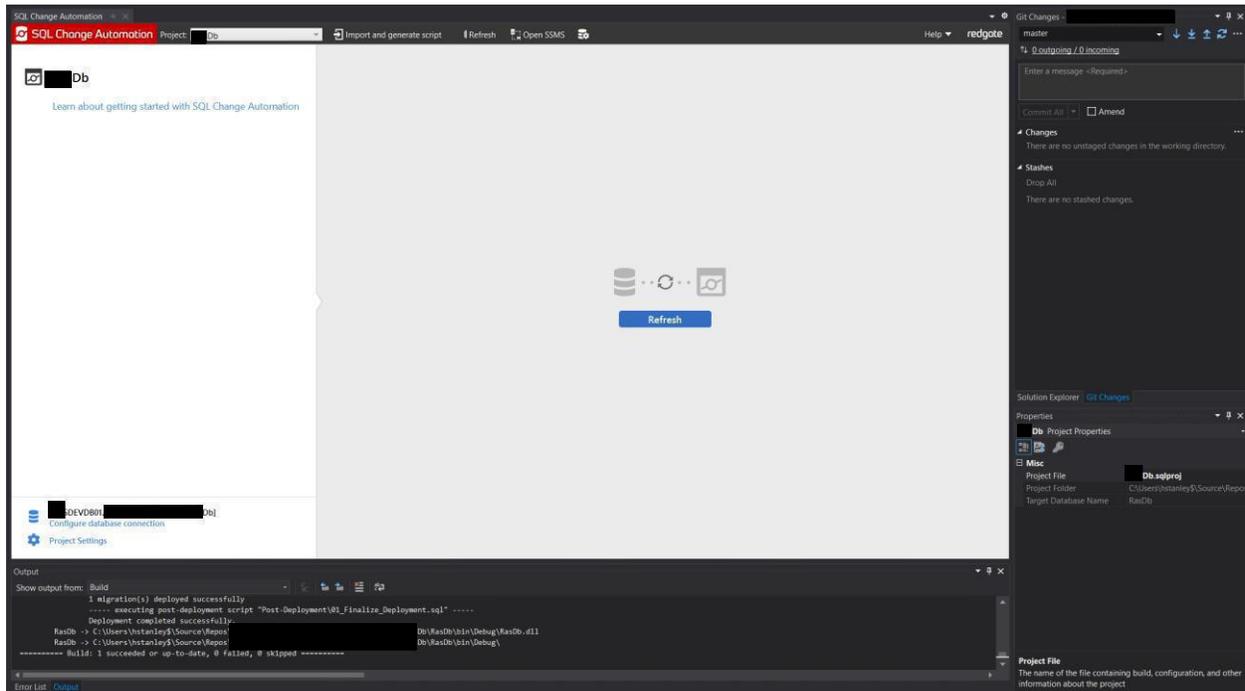


You'll need to resolve the errors by either removing the offending script or by Deploying the Project.



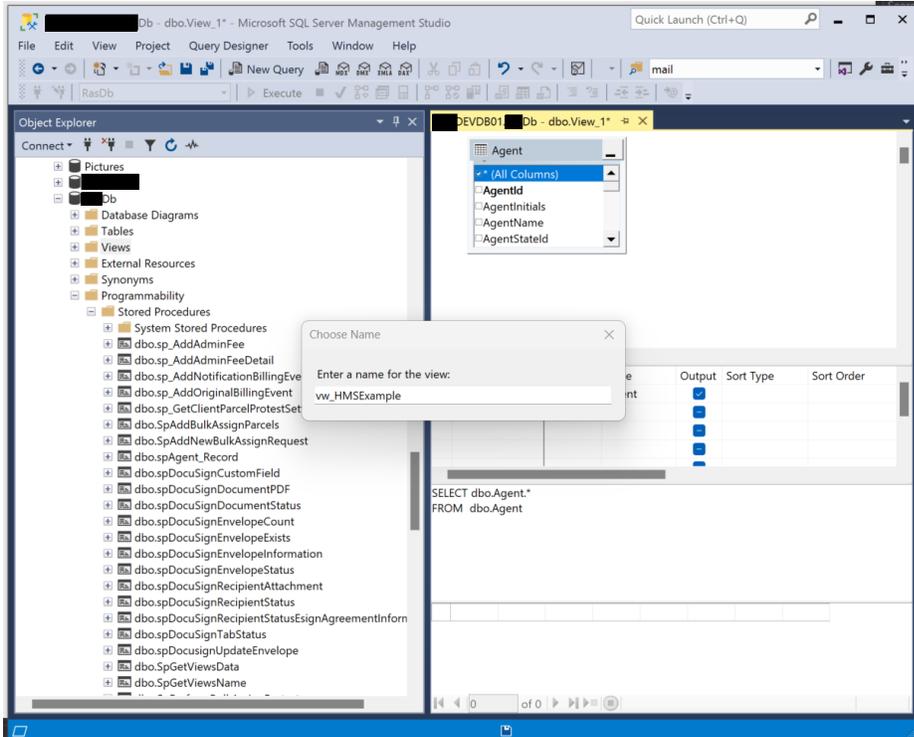
Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

When it's done resolving, you should rest here:

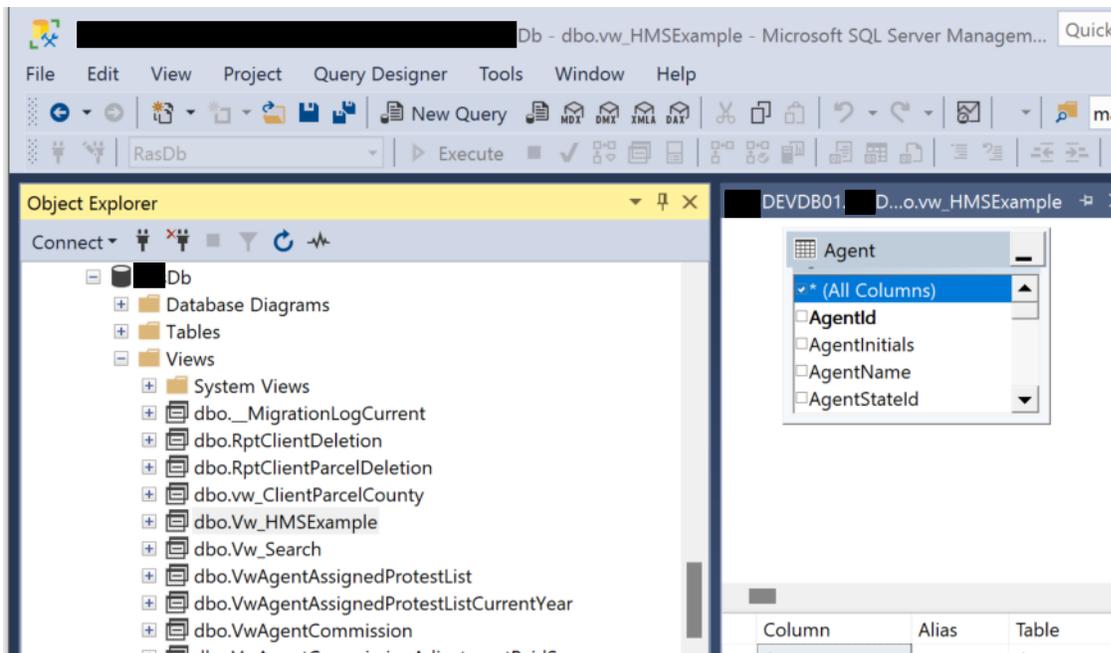


Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

On your machine or on the RedGate Server, open SSMS (Sql Server Management) and create a test table on the ausdevdb01.██████████.local database server. Stay within the database you are working with in visual studio. This example is in ██████DB and I've made a view called Vw_HMSEExample – use Pascal case (not like in the picture!).



You can see it in the list of Views now after a refresh of the Views folder. (Notice that I fixed the error in type case)



Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

Go back to the RedGate server and click that Refresh Button displayed earlier. This will grab any changes between the code from Bitbucket and the **Db** on ausdevdb01. **██████████.local** (also known as the SourceDb and the DatabaseServer):

The screenshot displays the SQL Change Automation interface. The main window shows a table of database objects with columns for Object Type, Owner, Object Name, and Action. The Action column for all objects is 'Add to Project'. The Change Summary column provides details for each object, such as 'Create view [dbo].[VwAgentPortalParcelMainImprovements]' and 'Add level 1 extended property MS_DiagramPaneCount on...'. Below the table, it indicates 'Identical objects on ██████████ (453 objects in-sync)'. The Output window at the bottom shows the build process: 'Build', '1 migration(s) deployed successfully', '----- executing post-deployment script "Post-Deployment_V01_Finalize_Deployment.sql" -----', 'Deployment completed successfully.', and 'Build: 1 succeeded or up-to-date, 0 failed, 0 skipped'.

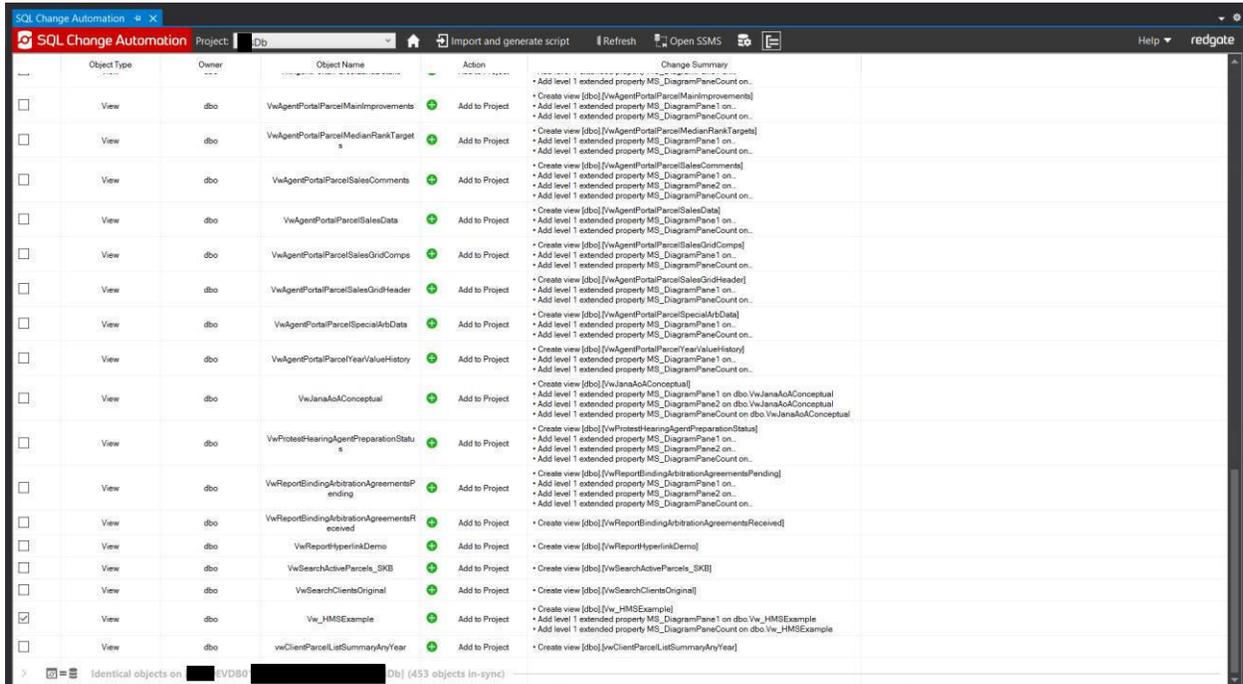
Object Type	Owner	Object Name	Action	Change Summary
View	dbo	VwAgentPortalParcelMainImprovements	Add to Project	• Create view [dbo].[VwAgentPortalParcelMainImprovements] • Add level 1 extended property MS_DiagramPaneCount on... • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPane2 on...
View	dbo	VwAgentPortalParcelMedianRankTargets	Add to Project	• Create view [dbo].[VwAgentPortalParcelMedianRankTargets] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwAgentPortalParcelSalesComments	Add to Project	• Create view [dbo].[VwAgentPortalParcelSalesComments] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPane2 on... • Add level 1 extended property MS_DiagramPane3 on...
View	dbo	VwAgentPortalParcelSalesData	Add to Project	• Create view [dbo].[VwAgentPortalParcelSalesData] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwAgentPortalParcelSalesGridComps	Add to Project	• Create view [dbo].[VwAgentPortalParcelSalesGridComps] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwAgentPortalParcelSalesGridHeader	Add to Project	• Create view [dbo].[VwAgentPortalParcelSalesGridHeader] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwAgentPortalParcelSpecialArbData	Add to Project	• Create view [dbo].[VwAgentPortalParcelSpecialArbData] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwAgentPortalParcelYearValueHistory	Add to Project	• Create view [dbo].[VwAgentPortalParcelYearValueHistory] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwJanaAocConceptual	Add to Project	• Create view [dbo].[VwJanaAocConceptual] • Add level 1 extended property MS_DiagramPane1 on dbo.VwJanaAocConceptual • Add level 1 extended property MS_DiagramPane2 on dbo.VwJanaAocConceptual • Add level 1 extended property MS_DiagramPaneCount on dbo.VwJanaAocConceptual
View	dbo	VwProtestHearingAgmtPreparationStatus	Add to Project	• Create view [dbo].[VwProtestHearingAgmtPreparationStatus] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPane2 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwReportBindingArbitrationAgreementsPending	Add to Project	• Create view [dbo].[VwReportBindingArbitrationAgreementsPending] • Add level 1 extended property MS_DiagramPane1 on... • Add level 1 extended property MS_DiagramPane2 on... • Add level 1 extended property MS_DiagramPaneCount on...
View	dbo	VwReportBindingArbitrationAgreementsReceived	Add to Project	• Create view [dbo].[VwReportBindingArbitrationAgreementsReceived]
View	dbo	VwReportHyperlinkDemo	Add to Project	• Create view [dbo].[VwReportHyperlinkDemo]
View	dbo	VwSearchActiveParcels_SKB	Add to Project	• Create view [dbo].[VwSearchActiveParcels_SKB]
View	dbo	VwSearchClientsOriginal	Add to Project	• Create view [dbo].[VwSearchClientsOriginal]
View	dbo	Vw_HMSEExample	Add to Project	• Create view [dbo].[Vw_HMSEExample] • Add level 1 extended property MS_DiagramPane1 on dbo.Vw_HMSEExample • Add level 1 extended property MS_DiagramPaneCount on dbo.Vw_HMSEExample
View	dbo	VwClientParcelListSummaryAnyYear	Add to Project	• Create view [dbo].[VwClientParcelListSummaryAnyYear]

```
Build
Show output from: Build
1 migration(s) deployed successfully
----- executing post-deployment script "Post-Deployment_V01_Finalize_Deployment.sql" -----
Deployment completed successfully.
RasDb -> C:\Users\hstanley\Source\Repos\
RasDb -> C:\Users\hstanley\Source\Repos\
Db\RasDb\bin\Debug\RasDb.dll
Db\RasDb\bin\Debug\
----- Build: 1 succeeded or up-to-date, 0 failed, 0 skipped -----
```

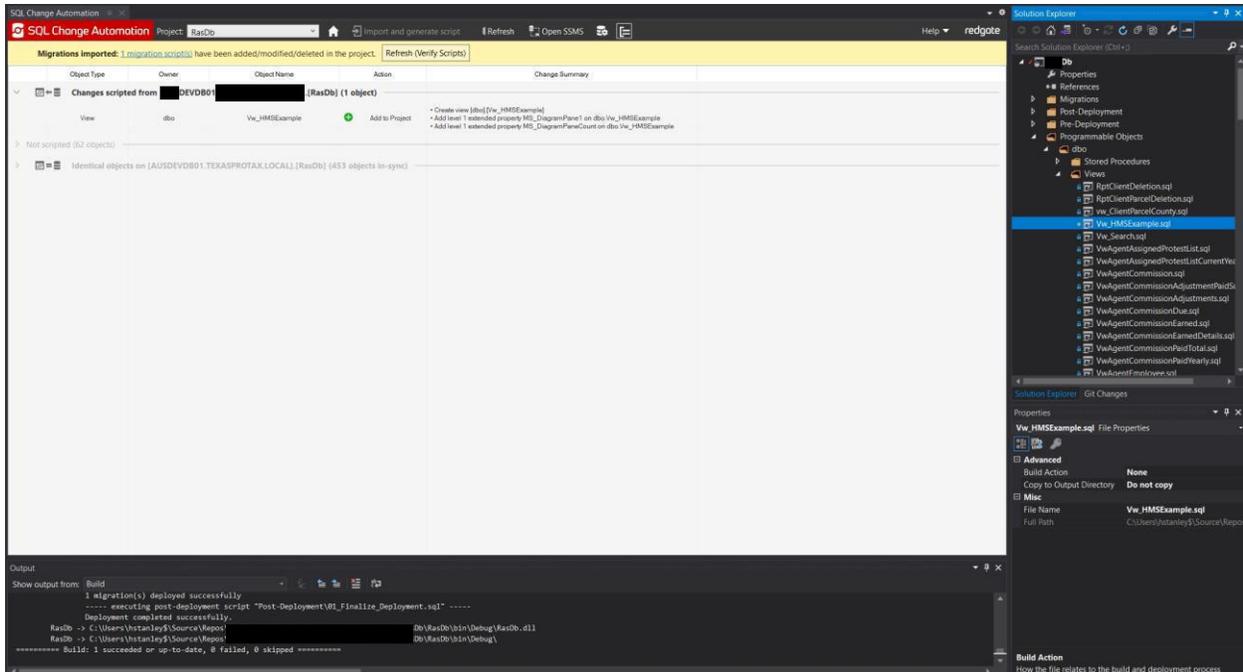
Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

Select the item(s) you want deployed out of the Development Database (SourceDB) by selecting the checkbox (You can select them all by clicking on the group header then selecting a checkbox to any object. Deselect all items by clicking on a whitespace next to an object)

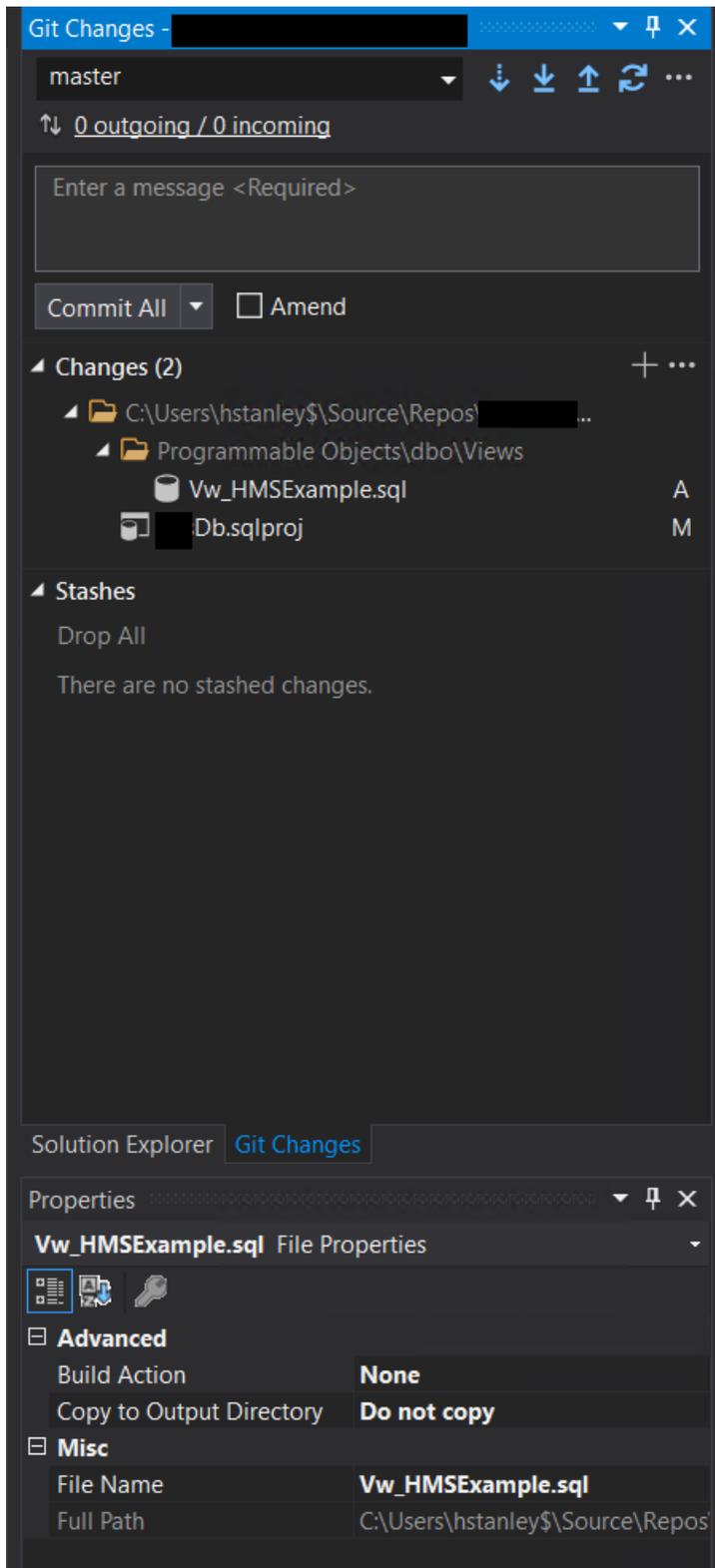
Here I've selected only the change created in this demonstration, then click the Import and generate script at the top of this screen.:



Now you can see, as indicated by the green plus symbol, that we have a new SQL Script Vw_HMSEExample.sql under the Views Folder.

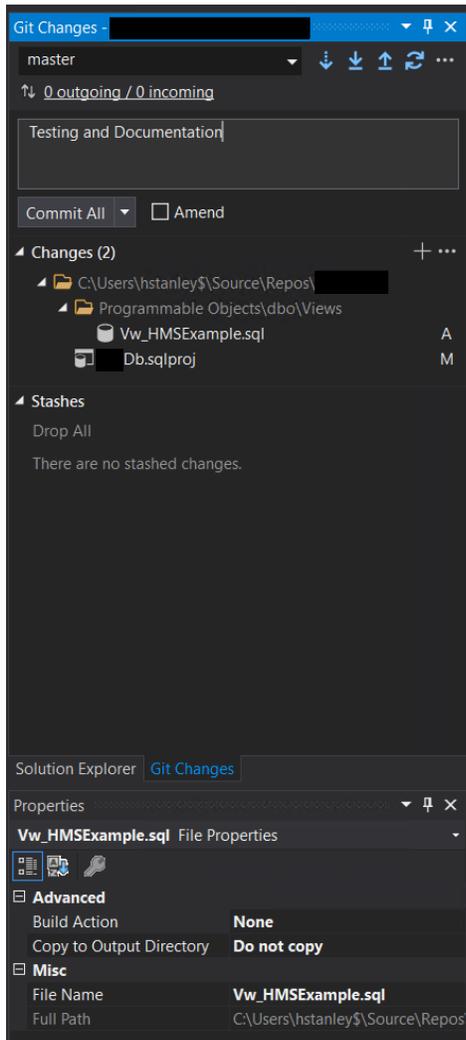


Let's click Git Changes,

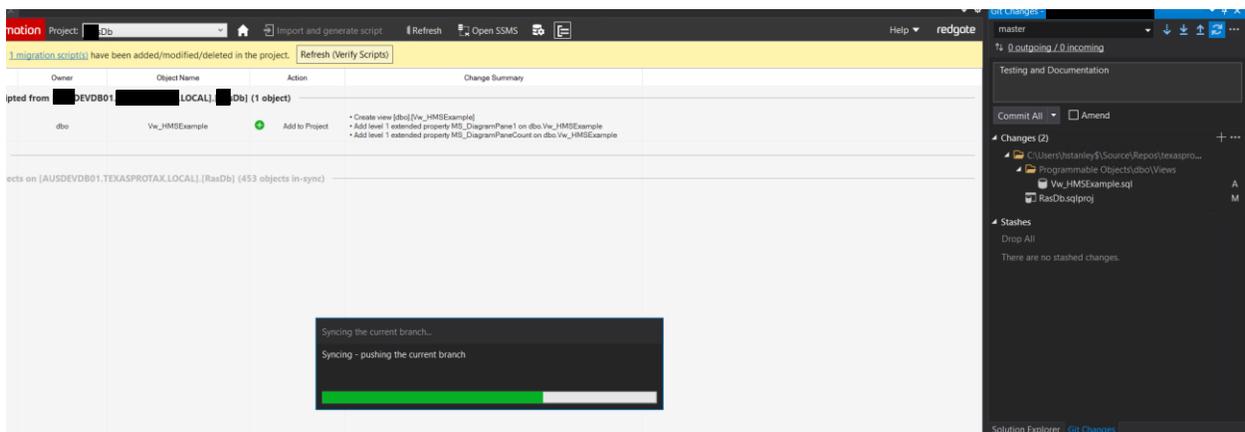


Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

Enter your Commit message, click Commit All, then select the Sync button (Pull, then Push)

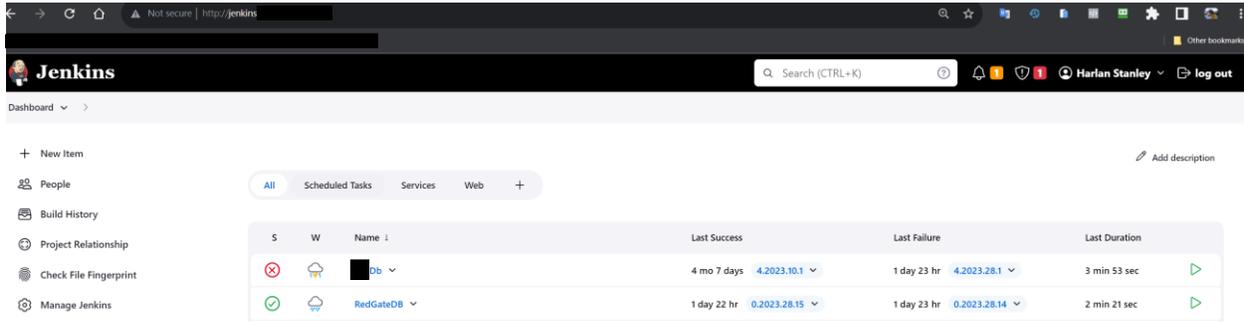


This will send the new script up to Bitbucket and we're done with the RedGate server, be sure to logout.



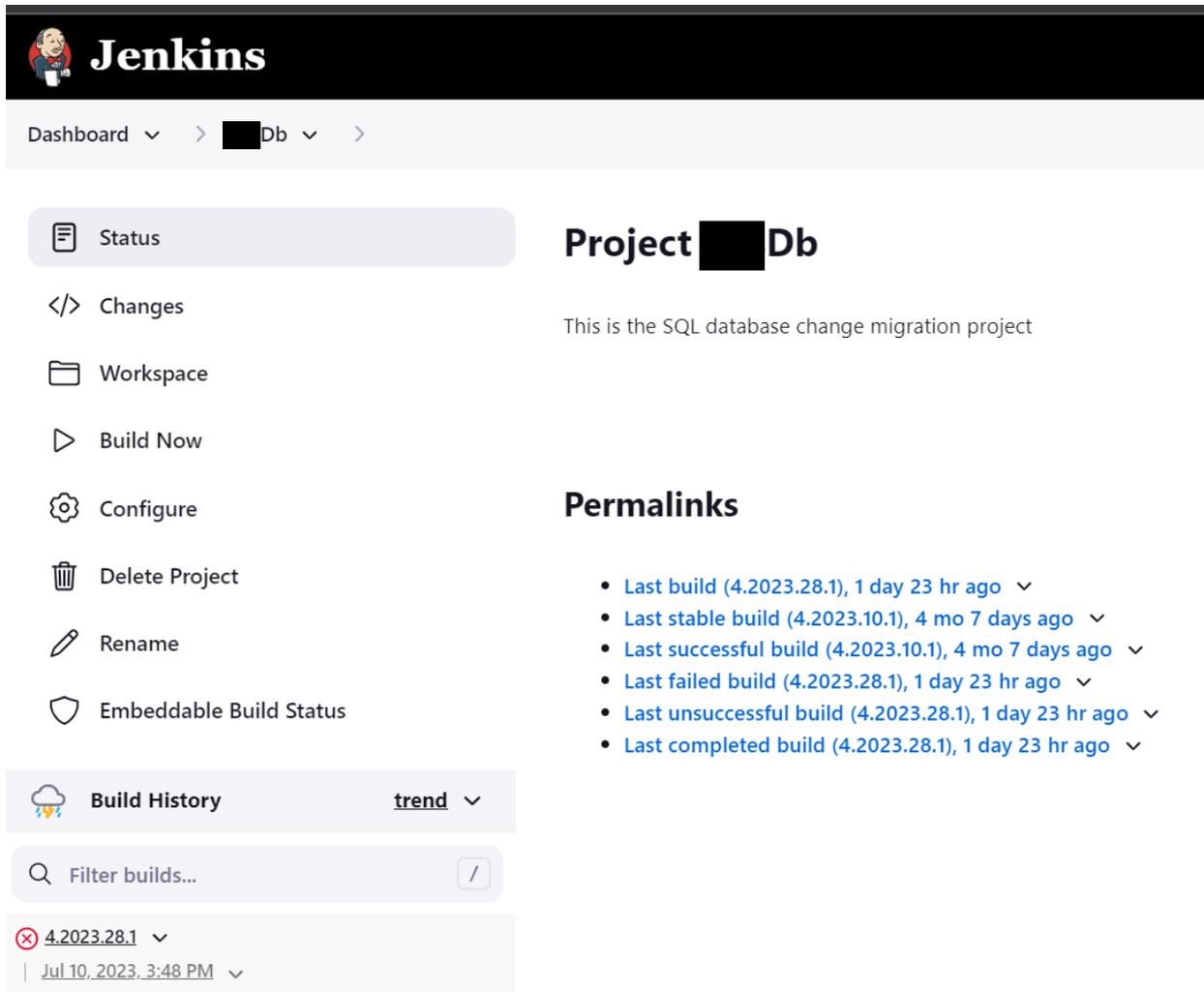
On your local machine let's switch over to Jenkins at <http://jenkins.██████████.local>

Jenkins and Octopus Operation:



Select **Db** from the menu

Click “Build Now”



It takes a moment, but you should see your build start, and once the version name updates, If you click on the date link, you will be able to open the console stream where you can see the status of everything happening.

- Status
- Changes
- Workspace
- Build Now
- Configure
- Delete Project
- Rename
- Embeddable Build Status

Project ██████D

This is the SQL database

Permalinks

- [Last build \(4.2023](#)
- [Last stable build \(](#)
- [Last successful bu](#)
- [Last failed build \(](#)
- [Last unsuccessful](#)
- [Last completed b](#)

 **Build History** trend ▼

Filter builds...

✘ [4.2023.28.1](#) ▼ ✘

Jul 12, 2023, 2:58 PM ▼

✘ 4.2023.28.1 ...

Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

The screenshot shows the Jenkins web interface. At the top, the browser address bar displays 'http://jenkins.[redacted]/job/RasDb/'. The Jenkins logo and name are visible. Below the navigation bar, the breadcrumb path is 'Dashboard > [redacted] Db >'. The main content area is titled 'Project [redacted] Db' and includes a description: 'This is the SQL database change migration project'. On the left, a sidebar contains several menu items: Status, Changes, Workspace, Build Now, Configure, Delete Project, Rename, and Embeddable Build Status. The 'Build History' section is expanded, showing a list of builds. The most recent build is '#129' with a green checkmark, dated 'Jul 12, 2023, 5:01 PM'. Below it, build '4.2023.28.1' is also marked as successful with a green checkmark and dated 'Jul 12, 2023, 4:50 PM'. To the right of the build history, there is a 'Permalinks' section with a list of links for various build types, such as 'Last build (4.2023.28.1), 10 min ago', 'Last stable build (4.2023.28.1), 10 min ago', 'Last successful build (4.2023.28.1), 10 min ago', 'Last failed build (4.2023.28.1), 14 min ago', 'Last unsuccessful build (4.2023.28.1), 14 min ago', and 'Last completed build (4.2023.28.1), 10 min ago'.

A green Check mark indicates that the build was successful. You can click on the Octopus icon, navy takes you to the Release screen and the light blue drops you in to the actual deployment page. We are done with Jenkins, let's move on to Octopus.

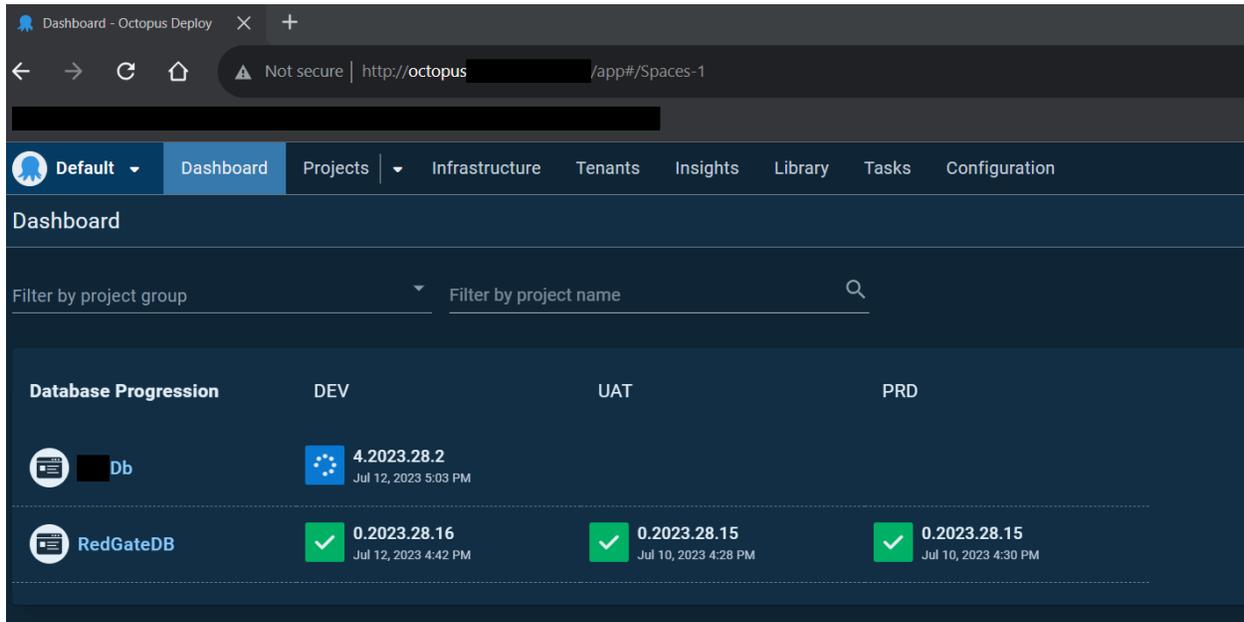
Alternatively, you can go to [http://octopus.\[redacted\].local](http://octopus.[redacted].local)

The screenshot shows the Octopus Deploy web interface. The browser address bar displays 'http://octopus.[redacted].local/app#/Spaces-1/projects/rasdb/deployments/releases/4.2023.28.2'. The interface has a dark theme. The top navigation bar includes 'Default', 'Dashboard', 'Projects', 'Infrastructure', 'Tenants', 'Insights', 'Library', 'Tasks', and 'Configuration'. The main content area is titled 'Projects Database Progression'. On the left, there is a sidebar with a 'CREATE RELEASE' button and a 'Deployments' section with sub-items: Overview, Process, Channels, Releases, Triggers, and Settings. The main content area shows a 'Release 4.2023.28.2' with a 'DEPLOY TO...' button and a 'DEPLOY TO DEV...' button. Below this, there is a 'Progression' section with a table showing the deployment process. The table has columns for 'Lifecycle', 'Task', and 'When'. The first row shows 'DEV' with a task 'Deploy to DEV' and a status of 'Deploying' (indicated by a blue box with a white octopus icon). The second row shows 'UAT' and 'PRD' with a status of 'Completed' (indicated by a green box). Below the table, there is a 'Release notes' section with the text 'Release created by Build [redacted] Db #129'.

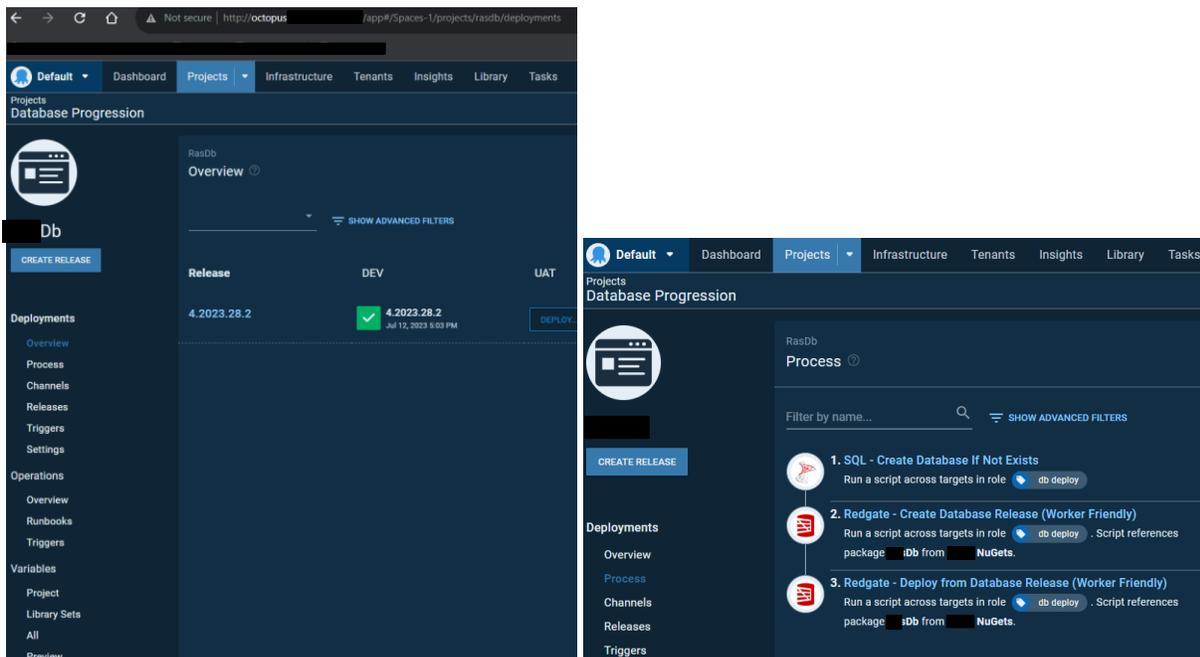
That blue box indicates that a deployment is happening. Green indicates that it was a success!

Octopus Configuration Overview:

Login to <http://octopus.██████████.local>



Select the ██████Db Project, then on the next page, Click Processes.



█████Db Database Progression takes only three steps in Octopus. Remember, of course to select save after each operation. – Or cancel if you’re just looking around.

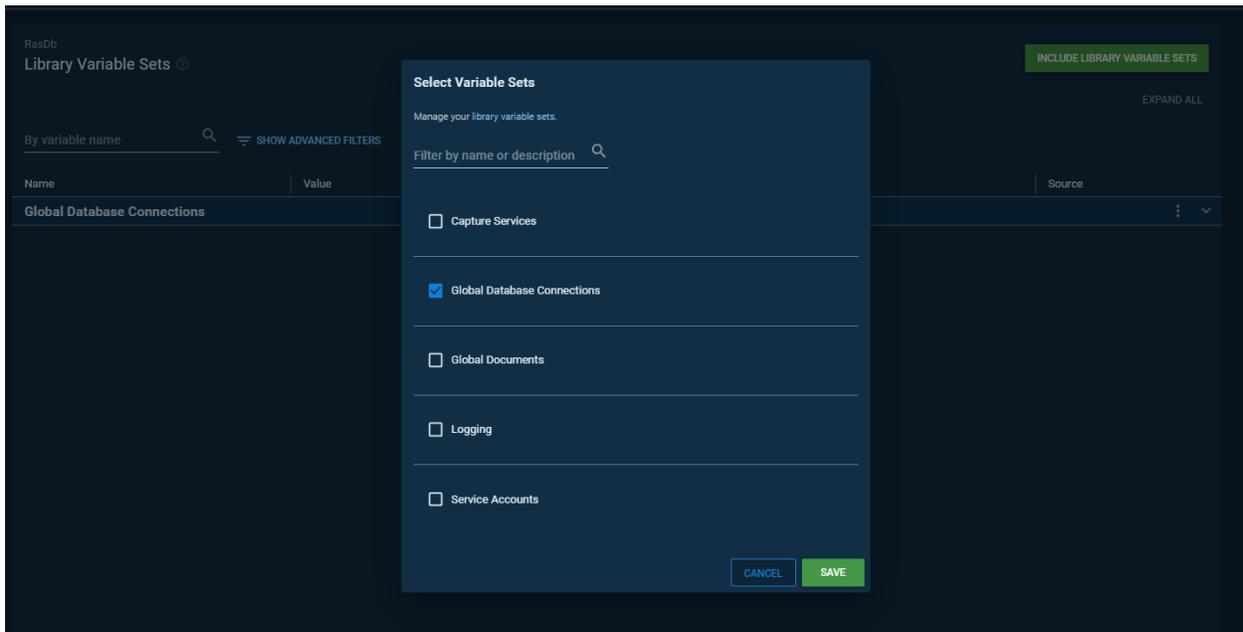
1. SQL – Create Database If Not Exists
2. Redgate – Create Database Release (Worker Friendly)
3. Redgate – Deploy from Database Release (Worker Friendly)

Database CI/CD with Bitbucket, Jenkins, Octopus, RedGate, and Visual Studio

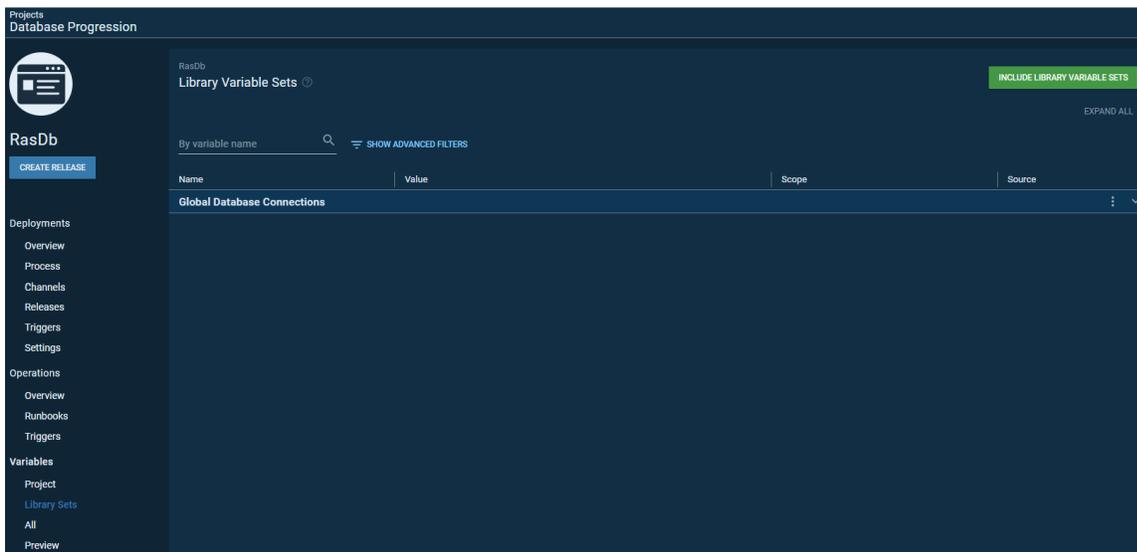
Octopus Variables must be set beforehand, #SourceDb, and #TargetDb are set in the Project Variables location.



#DatabaseServer is set in the Library Set “Global Database Connections” and must be subscribed to:



Click Save.



Each of these tasks have variable sets and configuration as described below.

1. SQL – Create Database If Not Exists

Be sure to select the Target Role to “db deploy” SQL Server is set to the variable `#{DataBaseServer}` and Database to create is set to `#{TargetDb}`

RUN A SCRIPT
1. SQL - Create Database If Not Exists

Step Name	SQL - Create Database If Not Exists <code>sql-create-database-if-not-exists</code>
Execution Location	This step will run on each deployment target
On Targets in Roles	<input checked="" type="checkbox"/> db deploy
SQL - Create Database If Not Exists	
This step is based on a community SQL - Create Database If Not Exists step template.	
SQL Server	<code>#{DataBaseServer}</code>
SQL Login	Provide SQL Login
SQL Password	Provide SQL Password
Database to create	<code>#{TargetDb}</code>
Command timeout	30
Azure database edition	Provide Azure database edition
Azure Backup Storage Redundacy	Provide Azure Backup Storage Redundacy
Retry database creation attempts	0
Conditions	
Environments	This step will run for all applicable Lifecycle environments (default)
Run Condition	Success: only run when previous steps succeed (or is first step) (default)
Package Requirement	Let Octopus decide (default)
Required	This step is not required and can be skipped

2. Redgate – Create Database Release (Worker Friendly)

Again, set Target Role to “db deploy”

Export path is set to C:\Octopus\Applications\#{Octopus.Environment.Name}\#{Octopus.Project.Name}

(This path is located on the Redgate Server, redgate.██████████.local)

Target SQL Server is set to the variable #{DataBaseServer} and Database to create is set to #{TargetDb}

RUN A SCRIPT
2. Redgate - Create Database Release (Worker Friendly)

Step Name	Redgate - Create Database Release (Worker Friendly) redgate-create-database-release-worker-friendly
Execution Location	This step will run on each deployment target
On Targets in Roles	db deploy
Redgate - Create Database Release (Worker Friendly)	
This step is based on a community Redgate - Create Database Release (Worker Friendly) step template.	
Export path	C:\Octopus\Applications\#{Octopus.Environment.Name}\#{Octopus.Project.Name}
Package	Package: Db from feed NuGets
Delete files in export folder	True
Target SQL Server instance	#{DataBaseServer}
Target database name	#{TargetDb}
Username (optional)	Provide Username (optional)
Password (optional)	Provide Password (optional)
Filter path (optional)	Provide Filter path (optional)
SQL Compare options (optional)	Provide SQL Compare options (optional)
SQL Data Compare options (optional)	Provide SQL Data Compare options (optional)
Transaction isolation level (optional)	Serializable
Ignore static data	Provide Ignore static data
Include identical objects in the change report	False
SQL Change Automation version (optional)	Provide SQL Change Automation version (optional)
SQL Change Automation Install Location (optional)	Provide SQL Change Automation Install Location (optional)
Conditions	
Environments	This step will run for all applicable Lifecycle environments (default)
Run Condition	Success: only run when previous steps succeed (default)
Start Trigger	Wait for the previous step to complete, then start (default)
Required	This step is not required and can be skipped

Pay attention to the Package selection you will need to select the dropdown to get **NuGets** and then pick the **Db** from the Package ID dropdown:

The screenshot shows the configuration interface for a step named "2. Redgate - Create Database Release (Worker Friendly)". The interface is dark-themed and includes the following sections:

- Step Name:** Redgate - Create Database Release (Worker Friendly)
- Execution Location:** This step will run on each deployment target
- On Targets in Roles:** A dropdown menu is set to "db deploy".
- Export path:** C:\Octopus\Applications\#{Octopus.Environment.Name}\#{Octopus.Project.Name}
- Package:** This section is titled "The name of the package to extract". It contains a description: "Packages can be uploaded directly to the Octopus built-in repository, or external package feeds can be configured in Library. Learn more about what your packages should contain, and how to create them." Below this, there are two dropdown menus:
 - Package feed:** Set to "NuGets".
 - Package ID:** Set to "Db".

Additional text at the bottom of the Package section reads: "Enter the ID of the package. The name used to identify this package reference is `DLMAutomation.Package.Name`. Learn more about Accessing Package References from a Custom Script."

3. Redgate – Deploy from Database Release (Worker Friendly)

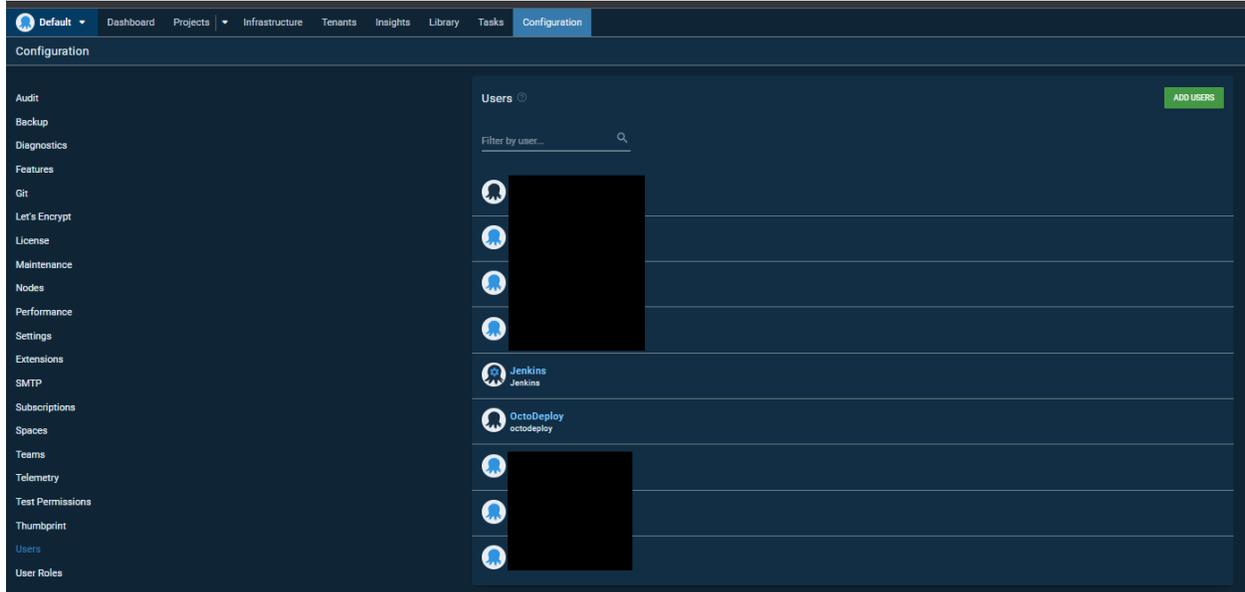
This is setup much the same way as Step 2, set the Roles, Export Path, Target SQL and Target Database, and finally the Package selection.

3. Redgate - Deploy from Database Release (Worker Friendly)

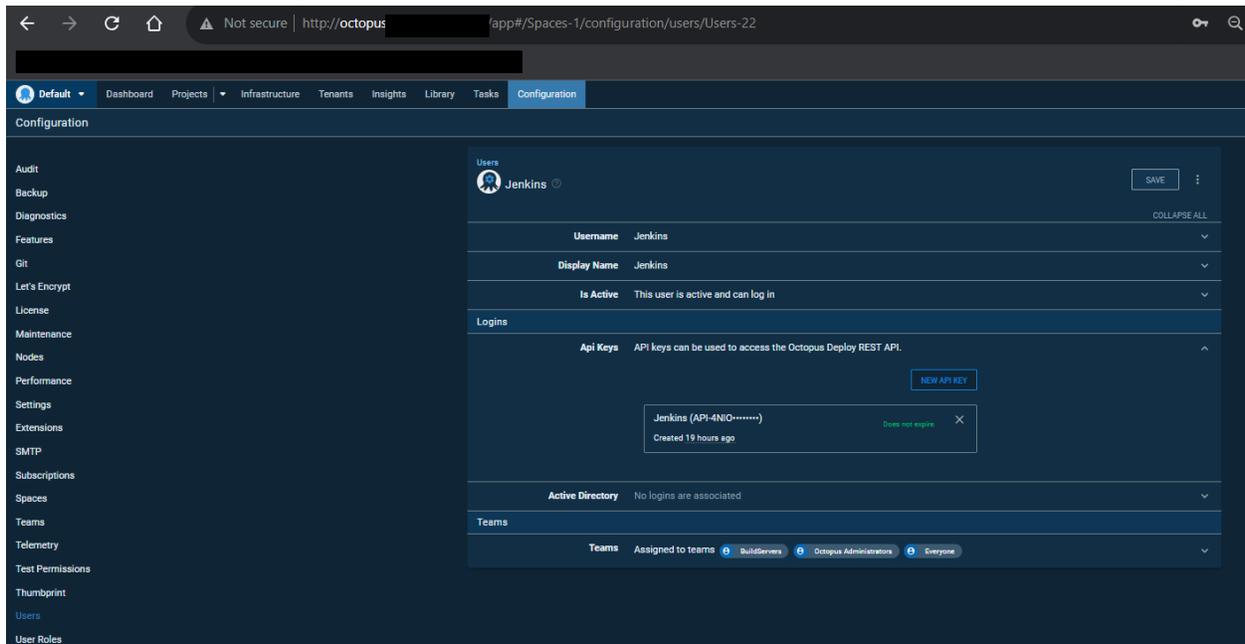
Step Name	Redgate - Deploy from Database Release (Worker Friendly) redgate-deploy-from-database-release-worker-friendly
Execution Location	This step will run on each deployment target
On Targets in Roles	db deploy
Redgate - Deploy from Database Release (Worker Friendly)	
This step is based on a community Redgate - Deploy from Database Release (Worker Friendly) step template.	
Export path	C:\Octopus\Applications\#{Octopus.Environment.Name}\#{Octopus.Project.Name}
Target SQL Server instance	#{DataBaseServer}
Target database name	#{TargetDb}
Username (optional)	Provide Username (optional)
Password (optional)	Provide Password (optional)
Query batch timeout (in seconds)	30
Skip post update schema check	False
SQL Change Automation version (optional)	Provide SQL Change Automation version (optional)
Package	Package RasDb from feed TXPT NuGets
SQL Change Automation Install Location (optional)	Provide SQL Change Automation Install Location (optional)
Conditions	
Environments	This step will run for all applicable Lifecycle environments (default)
Run Condition	Success: only run when previous steps succeed (default)
Start Trigger	Wait for the previous step to complete, then start (default)
Required	This step is not required and can be skipped

Octopus API User Configuration Overview:

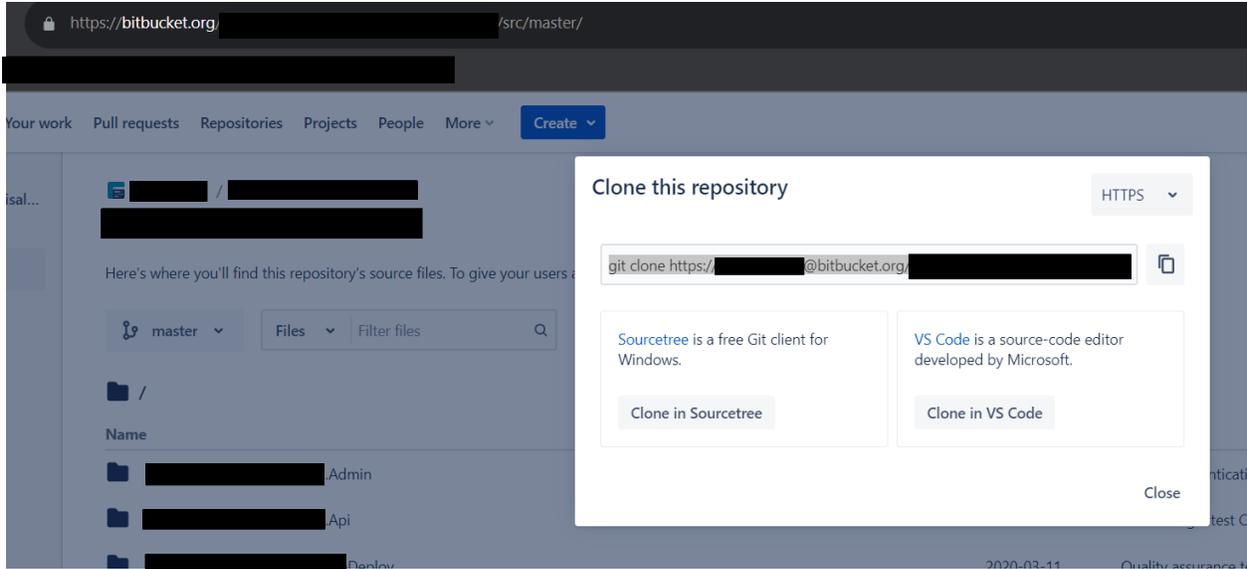
Creating an API Key for Jenkins to login to Octopus is done through the Octopus Configuration Window:



Once the Key is created, it is no longer viewable, you can create multiple keys for a user, the key for Jenkins is recorded at the end of this document. Do not change or delete the key for the Jenkins user!



BitBucket Git Clone Configuration:



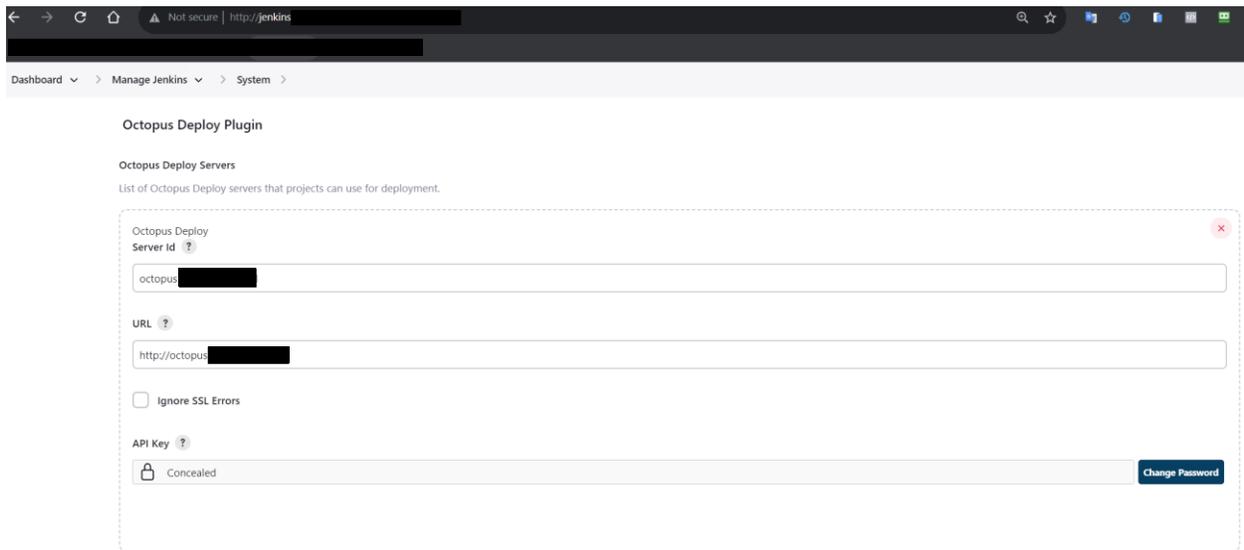
Jenkins Configuration Overview:



Please reference the above attached **Db Config [Jenkins].pdf** it has the configuration pages printed out from Jenkins.

Octopus API Key for Jenkins Authentication to Octopus NugetServer:

The following screenshot is for the Octopus Deploy Plugin on Jenkins. It requires an API key related to the Jenkins account within the Octopus application. The Key is below the screenshot.

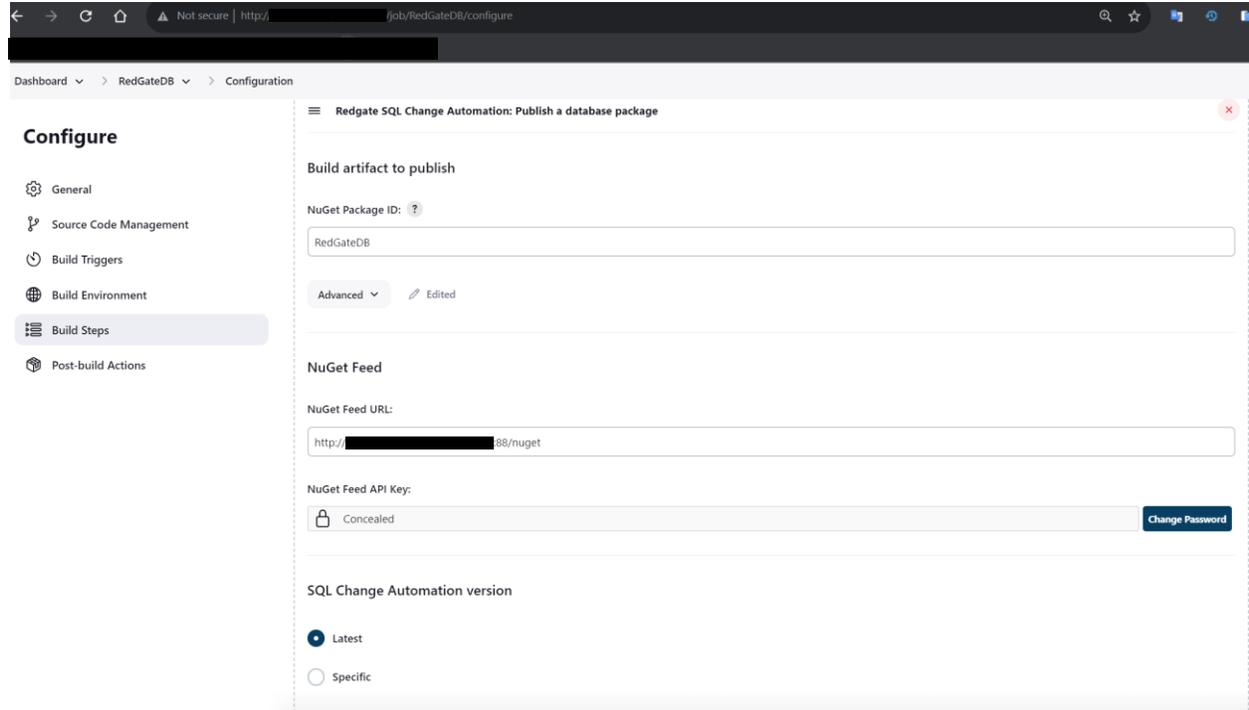


Octopus Jenkins Key: API-

Another thing to keep in mind is that this API key is located in D:\Build_Tools\Scripts\OctoBuild.ps1 on devbox01.local and is used for all the other various applications within the CI/CD system.

Jenkins builds will not work if this key is changed on the Octopus server.

There is another API key which is specific for the NuGet package manager loaded on [redacted]devbox01, it uses the API key below the screenshot. This key is generated by the Octopus Nuget Package Manager and is located within the configuration file for the IIS web application. “[redacted]devbox01 - D:\nuget_site\web.config (yeah... I know)



NuGet Server API:

Also keep in mind is that this API key is also located in D:\Build_Tools\Scripts\DotNetBuildWeb.ps1 on [redacted]devbox01.[redacted].local and is used for all the other various applications that use the [redacted] Nuget Package Manager within the CI/CD system.

Jenkins builds will not work if this key is changed on the octopus server.

Octopus JSON Files for [redacted]DB:

