[External] [In-Progress] Database Migration PowerShell Commands

A. Initial Setup

	To-do	PS Command
1	Enable remoting	`Enable-PSRemoting`
2	One-to-one remote	`Enter-PSSession -computerName <Server-R2>`

B. Auditing Server Properties

	Server Property	PS Command
1	Localhost	`$env:computername`
2	IP address	`Get-NetIPAddress -AddressFamily IPv4`
3	PowerShell version	`Get-Host | Select-Object -Property Version`

C. dbatools

dbatools require a minimum PowerShell version of 3.0: https://www.powershellgallery.com/packages/dbatools/1.0.135

See command B3 for checking the version, as well as the following articles on updating PowerShell:

- https://docs.microsoft.com/en-us/powershell/scripting/windows-powershell/install/installing-windows-powershell?view=powershell-7.1#upgrading-existing-windows-powershell
- https://docs.microsoft.com/en-us/powershell/scripting/windows-powershell/wmf/setup/install-configure?view=powershell-7.1

If PowerShell is greater or equal to 3.0 but less than 5.0, dbatools will have to be installed with an offline method: https://dbatools.io/offline/

If step 3 returns an error like “PowerShell Gallery is currently unavailable,” try running the commands here and restart PowerShell:

https://dev.to/darksmile92/powershell-disabled-support-for-tls-1-0-for-the-gallery-update-module-and-install-module-broken-1oii

	To-do	PS Command	Example
1	Get list of installed modules and versions	`Get-InstalledModule`	
2	Execution Policy	`Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser`	
3	Trust PowerShell Gallery	`Set-PSRepository -Name PSGallery -InstallationPolicy Trusted`	
4	Install dbatools	`Install-Module -Name dbatools`	
5	Update dbatools	`Update-Module dbatools`	
6	Test DB connection	`Connect-DbaInstance`	`Connect-DbaInstance -SqlInstance EBENGUNADI255E\MSSQLSERVER01`

7	Disable SQL Agent Jobs	`Set-DbaAgentJob -Disable`	`Get-DbaAgentJob -SqlInstance localhost |`
			`Set-DbaAgentJob -SqlInstance localhost -Disable`

8	Enable running SQL Agent Jobs	`Start-DbaAgentJob -Enable`	`Get-DbaAgentJob -SqlInstance localhost|`
			`Set-DbaAgentJob -SqlInstance localhost -Enable`

9	Create Credential object	`Get-Credential`	`$cred = Get-Credential sa`

10	Copy all non-system Logins	`Copy-DbaLogin -ExcludeSystemLogins`	`` Copy-DbaLogin -Source $replica -Destination $replicatocopy ` ``
			`` -SourceSqlCredential $scred -DestinationSqlCredential $dcred ` ``
			`-ExcludeLogin realcajun -ExcludeSystemLogins -EnableException`

11	Copy SQL Jobs
	⚠ Some jobs have hard-coded IPs, hostnames, database names, or directories. Be sure to manually check for these.
		`Copy-DbaAgentJob`	`` Copy-DbaAgentJob -Source sqlserver2014a -Destination sqlcluster ` ``
			`-SourceSqlCredential $scred -DestinationSqlCredential $dcred`

12	Visualize SQL Job schedules	`Find-DbaAgentJob | Get-DbaAgentJobHistory | ConvertTo-DbaTimeline`	`$oneweekago = [datetime]::Today.AddDays(-7)`
			`Find-DbaAgentJob -SqlInstance localhost  |`
			`Get-DbaAgentJobHistory -StartDate $oneweekago |`
			`ConvertTo-DbaTimeline |`
			`Out-File -FilePath c:\temp\jobs.html -Encoding ASCII`

13	Check for active connections to the database	`Get-DbaProcess`	`Get-DbaProcess -SqlInstance SourceServer -Database MigratingDatabase |`
			`Select Host, login, Program`

14	Remove connections to the database	`Stop-DbaProcess`	`Get-DbaProcess -SqlInstance SourceServer -Database MigratingDatabase |`
			`Stop-DbaProcess`

15	Copy databases	`Copy-DbaDatabase`	`Copy-DbaDatabase -Source sql2014a -Destination sql2014b -Database TestDB -BackupRestor`
16	Restore databases	`Restore-DbaDatabase`	`Restore-DbaDatabase -SqlInstance dbatoolslab\sql2017 -Path 'C:\dbatoolslab\Backup\Wide`

17	Install Standard Maintenance Plan	`Install-DbaMaintenanceSolution`	`$params = @{`
			`    SqlInstance = 'SQLSERVER01'`
			`    Database = 'medical'`
			`    ReplaceExisting = $true`
			`    InstallJobs = $true`
			`    CleanupTime = 72`
			`    Verbose = $true`
			`}`
			`Install-DbaMaintenanceSolution @params`

18	Automatically calculate and set maximum memory	`Set-DbaMaxMemory`	`Set-DbaMaxMemory -SqlInstance $SQLInstance`