How we made Configurable Pester Tests

for SQL Server

Rob Sewell, MVP, DBA
Chrissy LeMaire, MVP, DBA

Rob Sewell

Consultant, Sewells Consulting Ltd

in /in/robsewellsqldba

@sqldbawithbeard

robsewell.info

Sewells Consulting Limited

PowerShell Automator, PowerShell Trainer, Database DevOps

MVP, Speaker and Organizer

Half a Best Speaker!!, PowerShell and SQL user groups, national and international events

dbatools dbareports dbachecks

Loves Pester. Always available for help via the usual social media channels

Chrissy LeMaire

Sr. Systems Engineer
GDIT @ NATO Special Ops HQ

in /in/chrissylemaire

@cl

netnerds.net

SQL Data Pro – Since 1999

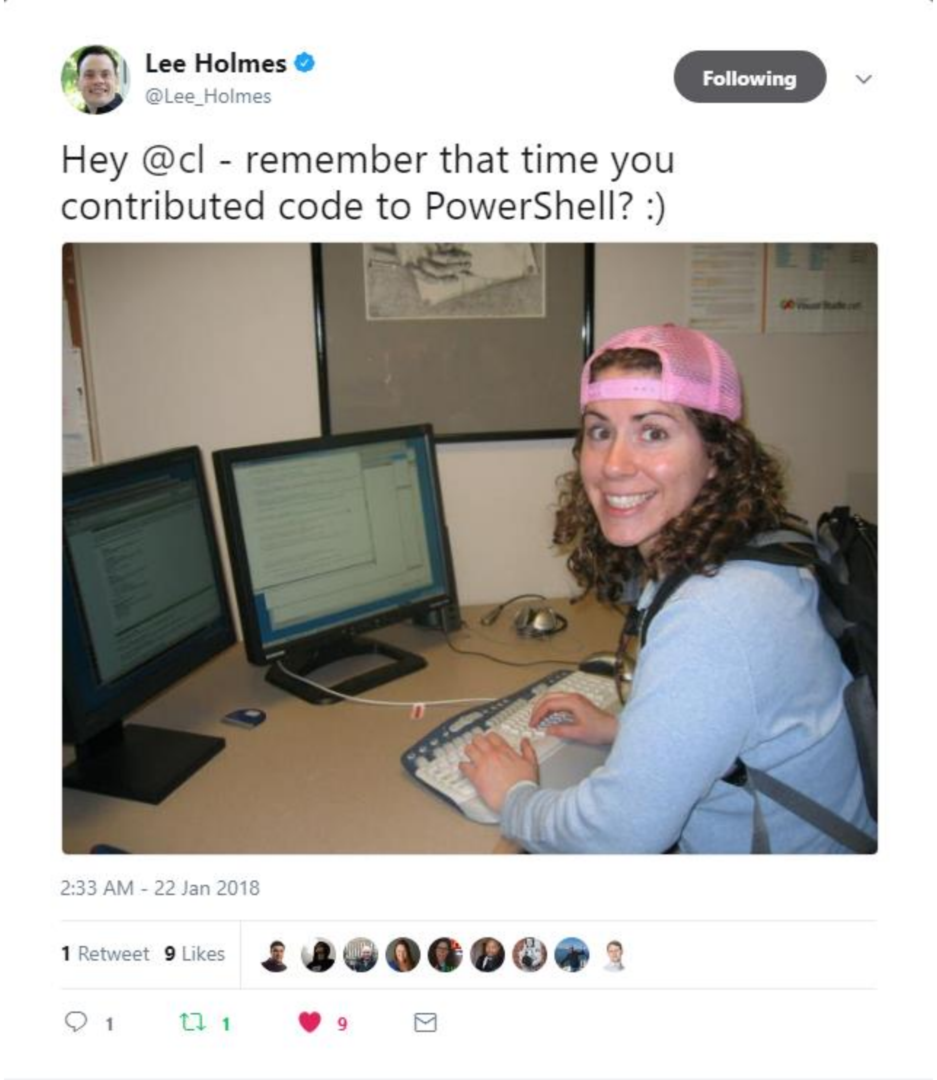
DBA, Developer and Architect

PowerShell MVP – Since 2015

Most PowerShell work revolves around SQL Server with a bit of VMware & SharePoint

PASS DevOps Virtual Chapter

Co-lead



Lee Holmes
@Lee_Holmes

Hey @cl - remember that time you contributed code to PowerShell? :)

2:33 AM - 22 Jan 2018

1 Retweet 9 Likes

1 1 9

PowerShell since 2005

Agenda

- Background
- Goals & Challenges
- Solutions
- Intro to module
- Demo

Background

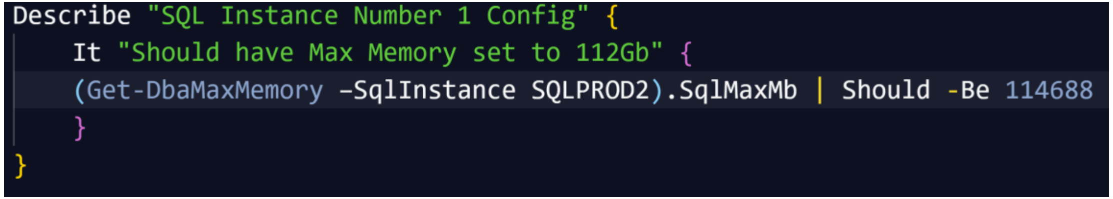
dbatools

- Community module founded by Chrissy
- Over 100 contributors
- 5 billion commands to work with SQL Server (no point putting an actual number in here it will change!!)
- Many commands to get information or check best practices
- Rob needed to validate estates at work
- Wrote Pester tests using dbatools
- More than a year discussing wondering the best way to enable configuration

Challenges & Goals

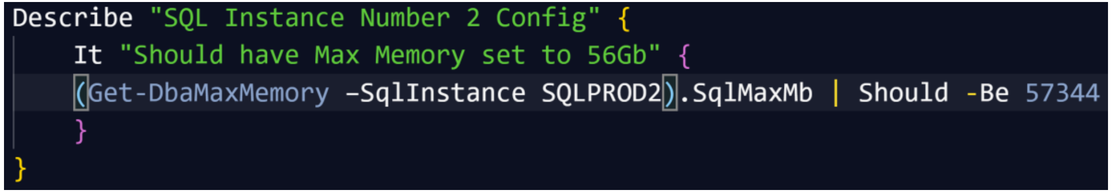
Main Challenge - Configuration

- Writing Pester Tests for one SQL instance is easy



```
Describe "SQL Instance Number 1 Config" {
    It "Should have Max Memory set to 112Gb" {
    (Get-DbaMaxMemory -SqlInstance SQLPROD2).SqlMaxMb | Should -Be 114688
    }
}
```

- Writing slightly different Pester Tests for slightly different instances is copy and paste

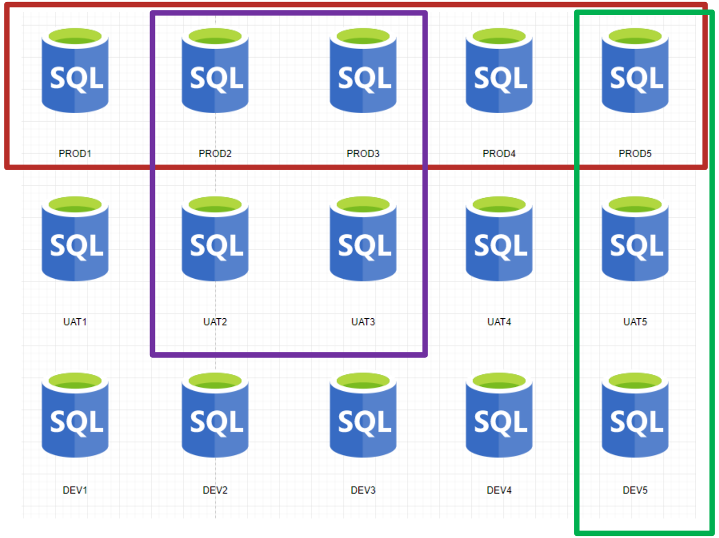


```
Describe "SQL Instance Number 2 Config" {
    It "Should have Max Memory set to 56Gb" {
    (Get-DbaMaxMemory -SqlInstance SQLPROD2).SqlMaxMb | Should -Be 57344
    }
}
```

- It is possible to parameterize Pester tests (but not so easy to say!)

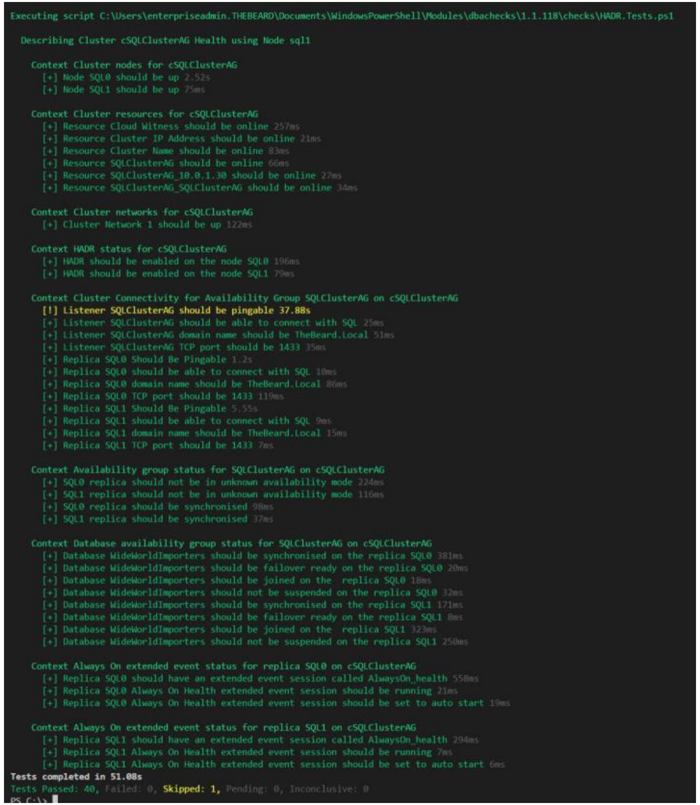
Main Challenge - Configuration

- We wanted to be able to Pester test a SQL environment like Production, UAT, DEV - horizontal
- We wanted to be able to Pester test a whole applications SQL environments – vertical
- We wanted to be able to Pester test the SQL estate for a solution – block



Challenge - Output

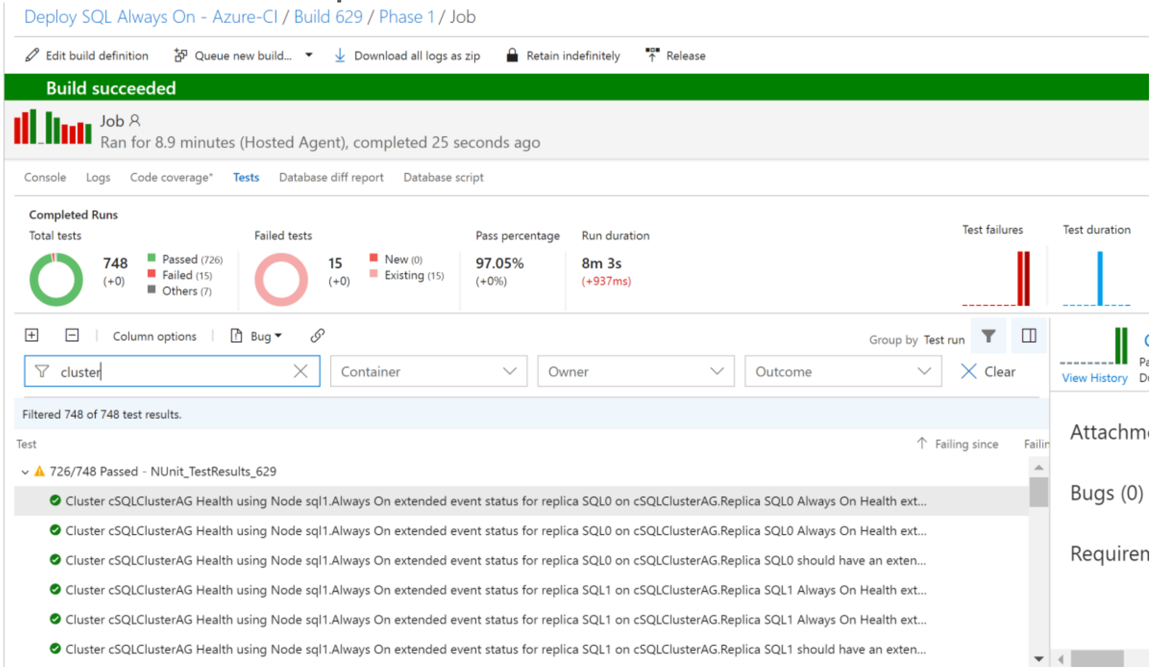
- DBAs may need output instantly



```
Executing script C:\Users\enterpriseadmin.THEBEARD\Documents\WindowsPowerShell\Modules\dbachecks\1.1.118\checks\HADR.Tests.ps1

  Describing Cluster cSQLClusterAG Health using Node sql1

    Context Cluster nodes for cSQLClusterAG
      [+] Node SQL0 should be up 2.52s
      [+] Node SQL1 should be up 75ms

    Context Cluster resources for cSQLClusterAG
      [+] Resource Cloud Witness should be online 257ms
      [+] Resource Cluster IP Address should be online 21ms
      [+] Resource Cluster Name should be online 83ms
      [+] Resource SQLClusterAG should be online 66ms
      [+] Resource SQLClusterAG_10.0.1.30 should be online 27ms
      [+] Resource SQLClusterAG_SQLClusterAG should be online 34ms

    Context Cluster networks for cSQLClusterAG
      [+] Cluster Network 1 should be up 122ms

    Context HADR status for cSQLClusterAG
      [+] HADR should be enabled on the node SQL0 196ms
      [+] HADR should be enabled on the node SQL1 79ms

    Context Cluster Connectivity for Availability Group SQLClusterAG on cSQLClusterAG
      [!] Listener SQLClusterAG should be pingable 37.88s
      [+] Listener SQLClusterAG should be able to connect with SQL 25ms
      [+] Listener SQLClusterAG domain name should be TheBeard.Local 51ms
      [+] Listener SQLClusterAG TCP port should be 1433 35ms
      [+] Replica SQL0 Should Be Pingable 1.2s
      [+] Replica SQL0 should be able to connect with SQL 10ms
      [+] Replica SQL0 domain name should be TheBeard.Local 86ms
      [+] Replica SQL0 TCP port should be 1433 119ms
      [+] Replica SQL1 Should Be Pingable 5.55s
      [+] Replica SQL1 should be able to connect with SQL 9ms
      [+] Replica SQL1 domain name should be TheBeard.Local 15ms
      [+] Replica SQL1 TCP port should be 1433 7ms

    Context Availability group status for SQLClusterAG on cSQLClusterAG
      [+] SQL0 replica should not be in unknown availability mode 224ms
      [+] SQL1 replica should not be in unknown availability mode 116ms
      [+] SQL0 replica should be synchronised 98ms
      [+] SQL1 replica should be synchronised 37ms

    Context Database availability group status for SQLClusterAG on cSQLClusterAG
      [+] Database WideWorldImporters should be synchronised on the replica SQL0 381ms
      [+] Database WideWorldImporters should be failover ready on the replica SQL0 20ms
      [+] Database WideWorldImporters should be joined on the  replica SQL0 18ms
      [+] Database WideWorldImporters should not be suspended on the replica SQL0 32ms
      [+] Database WideWorldImporters should be synchronised on the replica SQL1 171ms
      [+] Database WideWorldImporters should be failover ready on the replica SQL1 8ms
      [+] Database WideWorldImporters should be joined on the  replica SQL1 323ms
      [+] Database WideWorldImporters should not be suspended on the replica SQL1 250ms

    Context Always On extended event status for replica SQL0 on cSQLClusterAG
      [+] Replica SQL0 should have an extended event session called AlwaysOn_health 558ms
      [+] Replica SQL0 Always On Health extended event session should be running 21ms
      [+] Replica SQL0 Always On Health extended event session should be set to auto start 19ms

    Context Always On extended event status for replica SQL1 on cSQLClusterAG
      [+] Replica SQL1 should have an extended event session called AlwaysOn_health 294ms
      [+] Replica SQL1 Always On Health extended event session should be running 7ms
      [+] Replica SQL1 Always On Health extended event session should be set to auto start 6ms
Tests completed in 51.08s
Tests Passed: 40, Failed: 0, Skipped: 1, Pending: 0, Inconclusive: 0
PS C:\>
```

Challenge - Output

- DBAs may want to automate and integrate with other solutions (DevOps, Daily Checks, Incident Response, Maintenance Windows)



Deploy SQL Always On - Azure-CI / Build 629 / Phase 1 / Job

Edit build definition | Queue new build... | Download all logs as zip | Retain indefinitely | Release

Build succeeded

Job
Ran for 8.9 minutes (Hosted Agent), completed 25 seconds ago

Console | Logs | Code coverage* | Tests | Database diff report | Database script

Completed Runs

Total tests		Failed tests		Pass percentage	Run duration
748 (+0)	🟩 Passed (726) 🟥 Failed (15) ■ Others (7)	15 (+0)	🟥 New (0) 🟥 Existing (15)	97.05% (+0%)	8m 3s (+937ms)

Column options | Bug | Group by Test run

cluster | Container | Owner | Outcome | Clear

Filtered 748 of 748 test results.

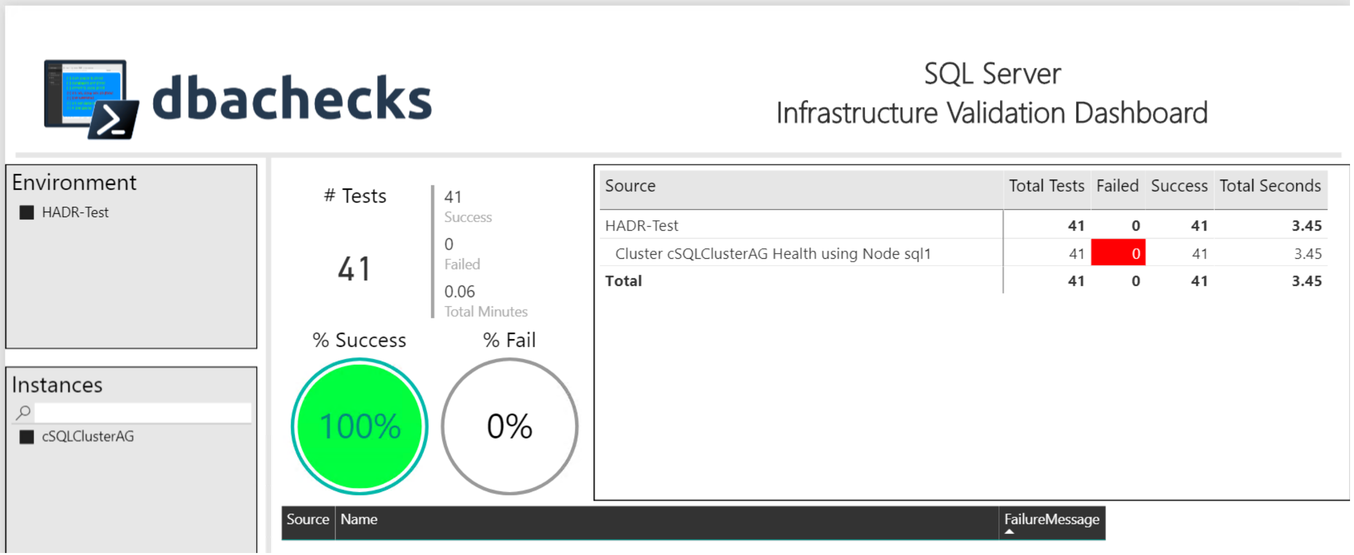
Test | Failing since | Failin

726/748 Passed - NUnit_TestResults_629

- Cluster cSQLClusterAG Health using Node sql1.Always On extended event status for replica SQL0 on cSQLClusterAG.Replica SQL0 Always On Health ext...
- Cluster cSQLClusterAG Health using Node sql1.Always On extended event status for replica SQL0 on cSQLClusterAG.Replica SQL0 Always On Health ext...
- Cluster cSQLClusterAG Health using Node sql1.Always On extended event status for replica SQL0 on cSQLClusterAG.Replica SQL0 should have an exten...
- Cluster cSQLClusterAG Health using Node sql1.Always On extended event status for replica SQL1 on cSQLClusterAG.Replica SQL1 Always On Health ext...
- Cluster cSQLClusterAG Health using Node sql1.Always On extended event status for replica SQL1 on cSQLClusterAG.Replica SQL1 Always On Health ext...
- Cluster cSQLClusterAG Health using Node sql1.Always On extended event status for replica SQL1 on cSQLClusterAG.Replica SQL1 should have an exten...

Challenge - Output

- Management want output they understand



dbachecks

SQL Server
Infrastructure Validation Dashboard

Environment

- ■ HADR-Test

Instances

- ■ cSQLClusterAG

# Tests	
41	41 Success 0 Failed 0.06 Total Minutes

% Success: 100%

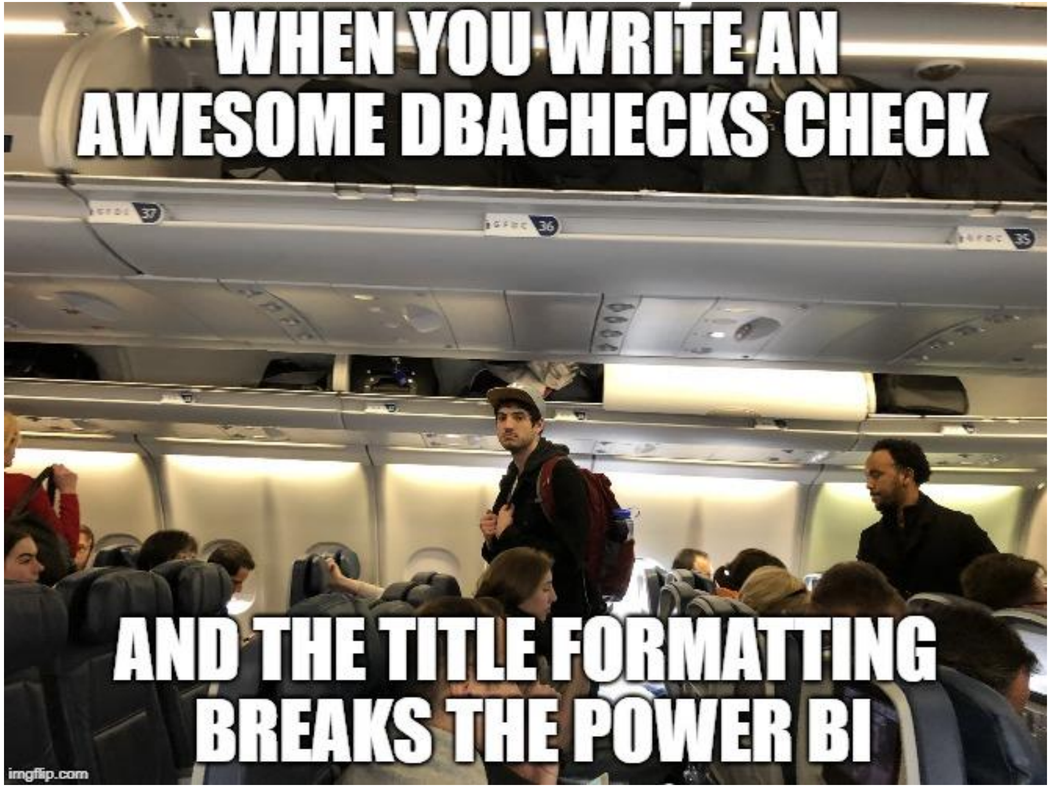
% Fail: 0%

Source	Total Tests	Failed	Success	Total Seconds
HADR-Test	**41**	**0**	**41**	**3.45**
Cluster cSQLClusterAG Health using Node sql1	41	0	41	3.45
Total	**41**	**0**	**41**	**3.45**

Source	Name	FailureMessage

Challenge - Output

- The Power Bi template requires a specific title configuration


WHEN YOU WRITE AN AWESOME DBACHECKS CHECK
AND THE TITLE FORMATTING BREAKS THE POWER BI
imgflip.com

Challenge – User Simplicity

End Users

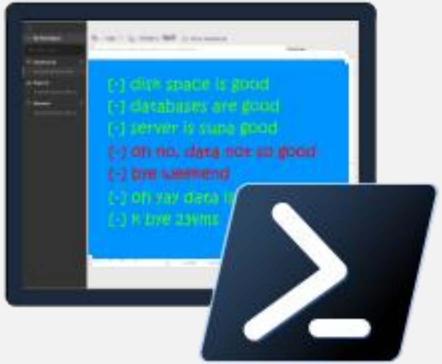
- Need simplicity to enable easy adoption
- Need index for Checks
- Need index for Configuration
- Simplified output options
- File system access to work across many differing user environments and permissions

Goals

- Create redistributable, easily configurable Pester tests using industry leaders checklists
- Enable output to suit the requirements of different types of end users human and machine
- Capability to provide response/resolution ?

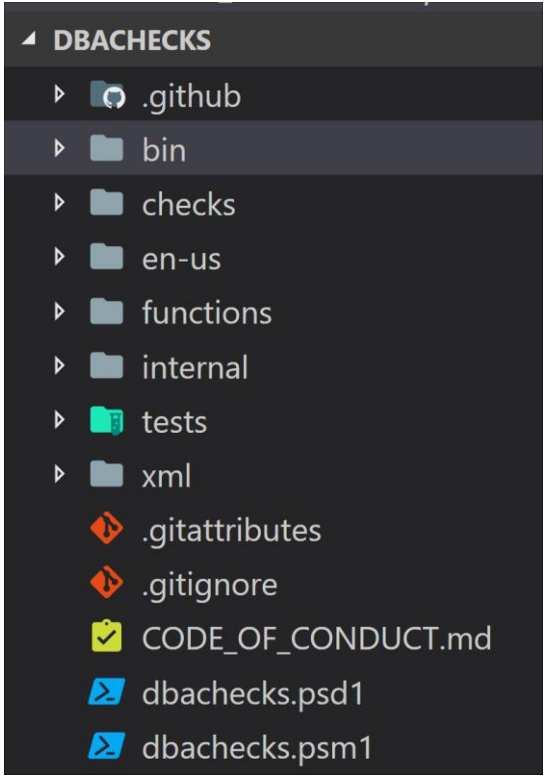
Solution

dbachecks



```
[-] disk space is good
[-] databases are good
[-] server is supa good
[-] oh no, data not so good
[-] bye weekend
[-] oh yay data is
[-] k bye 23hrs
```

dbachecks



DBACHECKS

- .github
- bin
- checks
- en-us
- functions
- internal
- tests
- xml
- .gitattributes
- .gitignore
- CODE_OF_CONDUCT.md
- dbachecks.psd1
- dbachecks.psm1

Dbachecks – Configuration

- Using PSFramework to create configuration items

```
Set-PSFConfig -Module dbachecks -Name app.sqlinstance -Value $null
-Initialize -Description "List of SQL Server instances"
```

- Stored in registry

Dbachecks – Configuration

- Enabling Users to set configuration

```
Set-DbcConfig -Name app.sqlinstance -Value sql2016, sql2017

Set-DbcConfig -Name app.sqlinstance -Value sqlcluster –Append
```

- Over 120 configuration items available right now (Apr 2018)

Dbachecks – Configuration

Using Configuration in Pester Tests

```
Describe "Network Latency" -Tag NetworkLatency, Connectivity, $filename {
 $max = Get-DbcConfigValue policy.network.latencymaxms
 @(Get-Instance).ForEach{
   Context "Testing Network Latency on $psitem" {
     @(Test-DbaNetworkLatency -SqlInstance $psitem).ForEach{
       It "network latency Should Be less than $max ms on $($psitem.SqlInstance)" {
         $psitem.Average.TotalMilliseconds | Should -BeLessThan $max -Because 'You dont want to be waiting on the network'
       }
   }
}
```

Dbachecks – Configuration

Export and Import the config

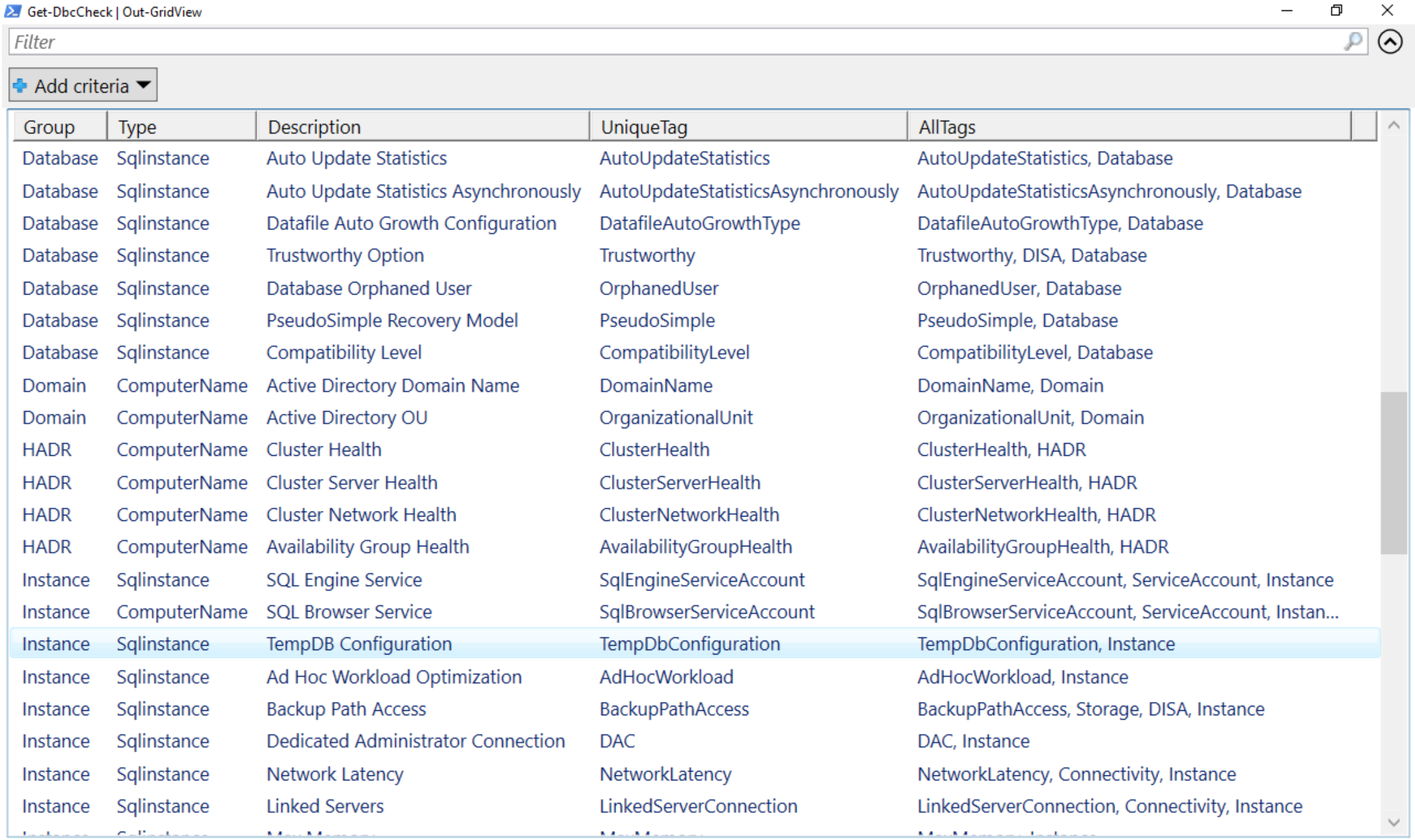
```
Export-DbcConfig -Path C:\Users\Beard\git\PesterConfigs\Application1_PROD.json
Export-DbcConfig -Path C:\Users\Hair\git\PesterConfigs\Client1_System2_Quick.json

Import-DbcConfig -Path Git:\PesterConfigs\Application1_PROD.json
Invoke-DbcCheck

Import-DbcConfig -Path Git:\PesterConfigs\Client1_System2_Quick.json
Invoke-DbcCheck
```

Dbachecks – User Simplicity

End user needs index

- Get-DbcCheck
- Get-DbcConfig
- Get-DbcTagCollection

Dbachecks – User Simplicity



Group	Type	Description	UniqueTag	AllTags
Database	Sqlinstance	Auto Update Statistics	AutoUpdateStatistics	AutoUpdateStatistics, Database
Database	Sqlinstance	Auto Update Statistics Asynchronously	AutoUpdateStatisticsAsynchronously	AutoUpdateStatisticsAsynchronously, Database
Database	Sqlinstance	Datafile Auto Growth Configuration	DatafileAutoGrowthType	DatafileAutoGrowthType, Database
Database	Sqlinstance	Trustworthy Option	Trustworthy	Trustworthy, DISA, Database
Database	Sqlinstance	Database Orphaned User	OrphanedUser	OrphanedUser, Database
Database	Sqlinstance	PseudoSimple Recovery Model	PseudoSimple	PseudoSimple, Database
Database	Sqlinstance	Compatibility Level	CompatibilityLevel	CompatibilityLevel, Database
Domain	ComputerName	Active Directory Domain Name	DomainName	DomainName, Domain
Domain	ComputerName	Active Directory OU	OrganizationalUnit	OrganizationalUnit, Domain
HADR	ComputerName	Cluster Health	ClusterHealth	ClusterHealth, HADR
HADR	ComputerName	Cluster Server Health	ClusterServerHealth	ClusterServerHealth, HADR
HADR	ComputerName	Cluster Network Health	ClusterNetworkHealth	ClusterNetworkHealth, HADR
HADR	ComputerName	Availability Group Health	AvailabilityGroupHealth	AvailabilityGroupHealth, HADR
Instance	Sqlinstance	SQL Engine Service	SqlEngineServiceAccount	SqlEngineServiceAccount, ServiceAccount, Instance
Instance	ComputerName	SQL Browser Service	SqlBrowserServiceAccount	SqlBrowserServiceAccount, ServiceAccount, Instan...
Instance	Sqlinstance	TempDB Configuration	TempDbConfiguration	TempDbConfiguration, Instance
Instance	Sqlinstance	Ad Hoc Workload Optimization	AdHocWorkload	AdHocWorkload, Instance
Instance	Sqlinstance	Backup Path Access	BackupPathAccess	BackupPathAccess, Storage, DISA, Instance
Instance	Sqlinstance	Dedicated Administrator Connection	DAC	DAC, Instance
Instance	Sqlinstance	Network Latency	NetworkLatency	NetworkLatency, Connectivity, Instance
Instance	Sqlinstance	Linked Servers	LinkedServerConnection	LinkedServerConnection, Connectivity, Instance

Dbachecks – User Simplicity

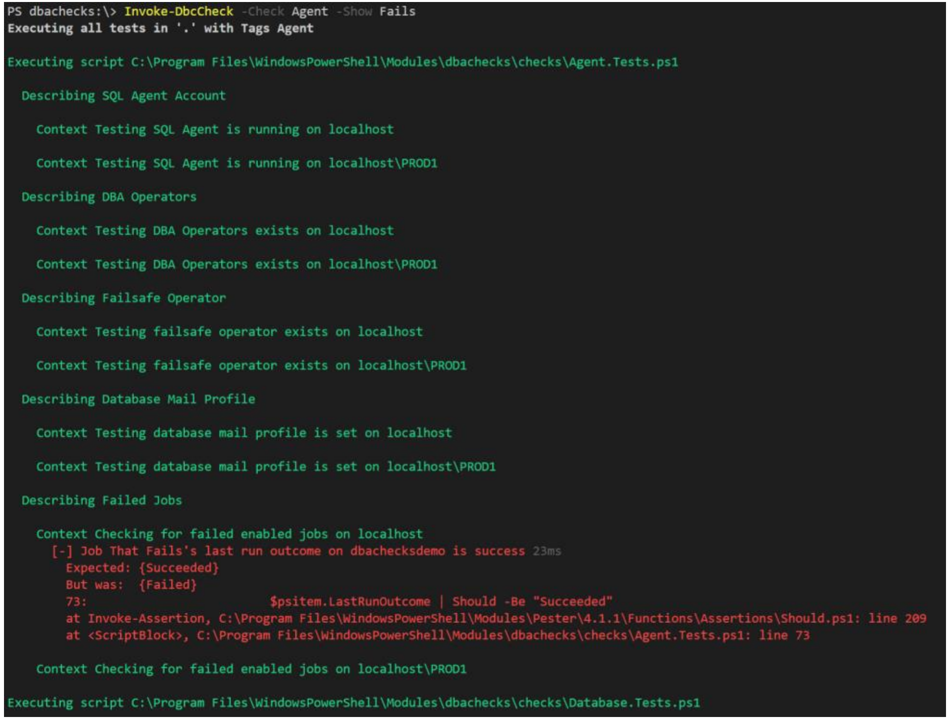
Simplified output

- Send-DbcMailMessage
- Update-DbcPowerBiDataSource
- Start-DbcPowerBi

Dbachecks – Output

Invoke-DbcCheck wraps Invoke-Pester so results available at command-line

Invoke-DbcCheck -Show Fails



```
PS dbachecks:\> Invoke-DbcCheck -Check Agent -Show Fails
Executing all tests in '.' with Tags Agent

Executing script C:\Program Files\WindowsPowerShell\Modules\dbachecks\checks\Agent.Tests.ps1

  Describing SQL Agent Account

    Context Testing SQL Agent is running on localhost

    Context Testing SQL Agent is running on localhost\PROD1

  Describing DBA Operators

    Context Testing DBA Operators exists on localhost

    Context Testing DBA Operators exists on localhost\PROD1

  Describing Failsafe Operator

    Context Testing failsafe operator exists on localhost

    Context Testing failsafe operator exists on localhost\PROD1

  Describing Database Mail Profile

    Context Testing database mail profile is set on localhost

    Context Testing database mail profile is set on localhost\PROD1

  Describing Failed Jobs

    Context Checking for failed enabled jobs on localhost
      [-] Job That Fails's last run outcome on dbachecksdemo is success 23ms
        Expected: {Succeeded}
        But was:  {Failed}
        73:                         $psitem.LastRunOutcome | Should -Be "Succeeded"
        at Invoke-Assertion, C:\Program Files\WindowsPowerShell\Modules\Pester\4.1.1\Functions\Assertions\Should.ps1: line 209
        at <ScriptBlock>, C:\Program Files\WindowsPowerShell\Modules\dbachecks\checks\Agent.Tests.ps1: line 73

    Context Checking for failed enabled jobs on localhost\PROD1

Executing script C:\Program Files\WindowsPowerShell\Modules\dbachecks\checks\Database.Tests.ps1
```

Dbachecks – Output

Invoke-DbcCheck can output XML (Just like Invoke-Pester can)

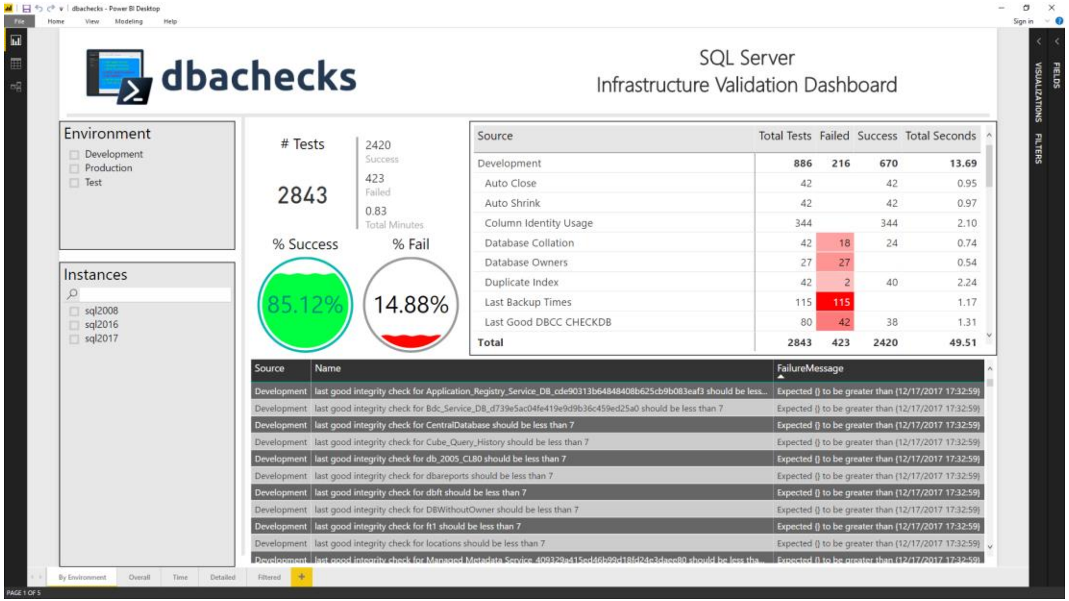
```
 Import-DbcConfig -Path
$(System.WorkingDirectory)\PesterConfigs\Application.json

Invoke-DbcCheck -Show Summary -PassThru -OutputFile
$(System.WorkingDirectory)\Test-Results.xml
```

Dbachecks – Output

Invoke-DbcCheck for multiple scenarios all in one PowerBi ☺

```
Import-DbcConfig –path Git:\PesterConfigs\Application1_PROD.json
Invoke-DbcCheck -Show Summary -PassThru | Update-DbcPowerBiDataSource –Environment App1_Prod

Import-DbcConfig –path Git:\PesterConfigs\Application2_PROD.json
Invoke-DbcCheck -Show Summary -PassThru | Update-DbcPowerBiDataSource –Environment App2_Prod

Import-DbcConfig –path Git:\PesterConfigs\Application3_PROD.json
Invoke-DbcCheck -Show Summary -PassThru | Update-DbcPowerBiDataSource –Environment App3_Prod

Start-DbcPowerbi
```

Dashboards galore



dbachecks - Power BI Desktop

dbachecks

SQL Server
Infrastructure Validation Dashboard

Environment

- □ Development
- □ Production
- □ Test

Instances

- □ sql2008
- □ sql2016
- □ sql2017

Tests

2843

2420 Success

423 Failed

0.83 Total Minutes

% Success: 85.12%

% Fail: 14.88%

Source	Total Tests	Failed	Success	Total Seconds
Development	**886**	**216**	**670**	**13.69**
Auto Close	42		42	0.95
Auto Shrink	42		42	0.97
Column Identity Usage	344		344	2.10
Database Collation	42	18	24	0.74
Database Owners	27	27		0.54
Duplicate Index	42	2	40	2.24
Last Backup Times	115	115		1.17
Last Good DBCC CHECKDB	80	42	38	1.31
Total	**2843**	**423**	**2420**	**49.51**

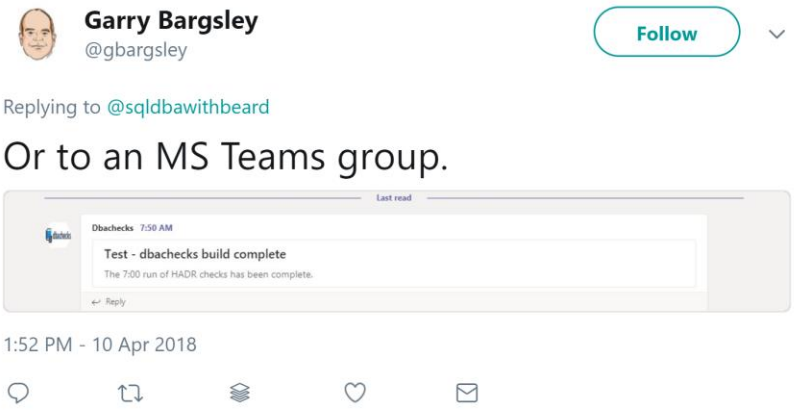
Source	Name	FailureMessage
Development	last good integrity check for Application_Registry_Service_DB_cde90313b64848408b625cb9b083eaf3 should be less...	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for Bdc_Service_DB_d739e5ac04fe419e9d9b36c459ed25a0 should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for CentralDatabase should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for Cube_Query_History should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for db_2005_CL80 should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for dbareports should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for dbft should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for DBWithoutOwner should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for ft1 should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for locations should be less than 7	Expected {} to be greater than (12/17/2017 17:32:59)
Development	last good integrity check for Managed Metadata Service_409329a415ed46b99d18fd24e3daee80 should be less tha...	Expected {} to be greater than (12/17/2017 17:32:59)

We use C:\Windows\Temp\dbachecks to simplify enabling instant refreshes

Dbachecks – What we didn't do

Enable Output to

- Database
- MS Teams
- Slack
- Twitter
- JIRA



Garry Bargsley
@gbargsley

Replying to @sqldbawithbeard

Or to an MS Teams group.

Last read

Dbachecks 7:50 AM

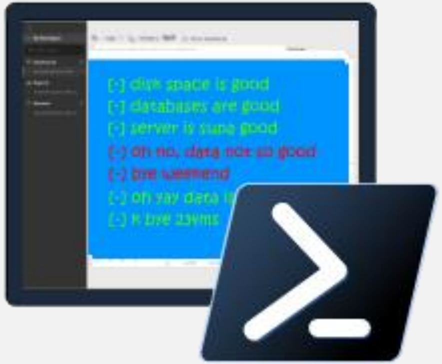
Test - dbachecks build complete

The 7:00 run of HADR checks has been complete.

1:52 PM - 10 Apr 2018

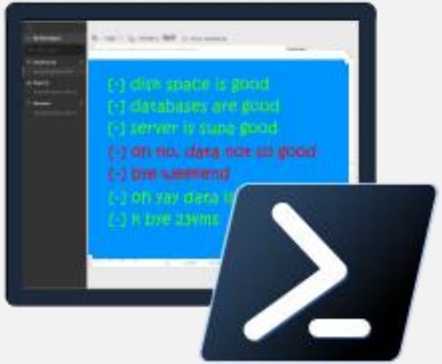
Because it's PowerShell so Invoke-DbcCheck –PassThru | ANYTHING

demo time!



```
[-] disk space is good
[-] databases are good
[-] server is supa good
[-] oh no, data not so good
[-] bye weekend
[-] oh yay data is
[-] k bye 23yms
```

questions

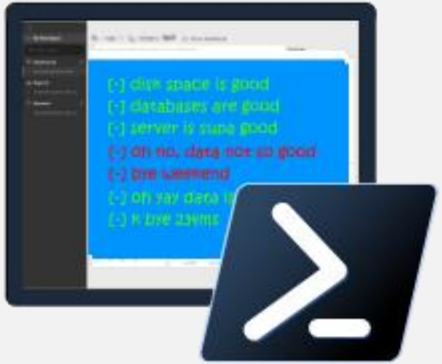


```
[-] disk space is good
[-] databases are good
[-] server is supa good
(-) oh no, data not so good
(-) bye weekend
[-] oh yay data is
[-] k bye 23vms
```

Blog posts & Twitter

- dbachecks.io/intro
- dbachecks.io/deepdive
- dbachecks.io/blog
- dbachecks.io/twitter

thank you!



```
[-] disk space is good
[-] databases are good
[-] server is supa good
[-] oh no, data not so good
[-] bye weekend
[-] oh yay data is
[-] k bye 23yms
```

Install is easy

POWERSHELL GALLERY

```
Install-Module dbachecks

Install-Module dbachecks –Scope CurrentUser
```

* Automatically installs required modules

Join our Slack channel

- Invite yourself to Slack
    - dbatools.io/slack
- Join #dbachecks and #dbatools
- Ask questions, possibly get answers in real time ;)

Pester

♡

SQL Server