



The SQL Server Experts  
Discover. Optimise. Mentor.

# Troubleshooting SQL Server with PowerShell

James Boother

[james@coeo.com](mailto:james@coeo.com)

[@jimmyboo](#)





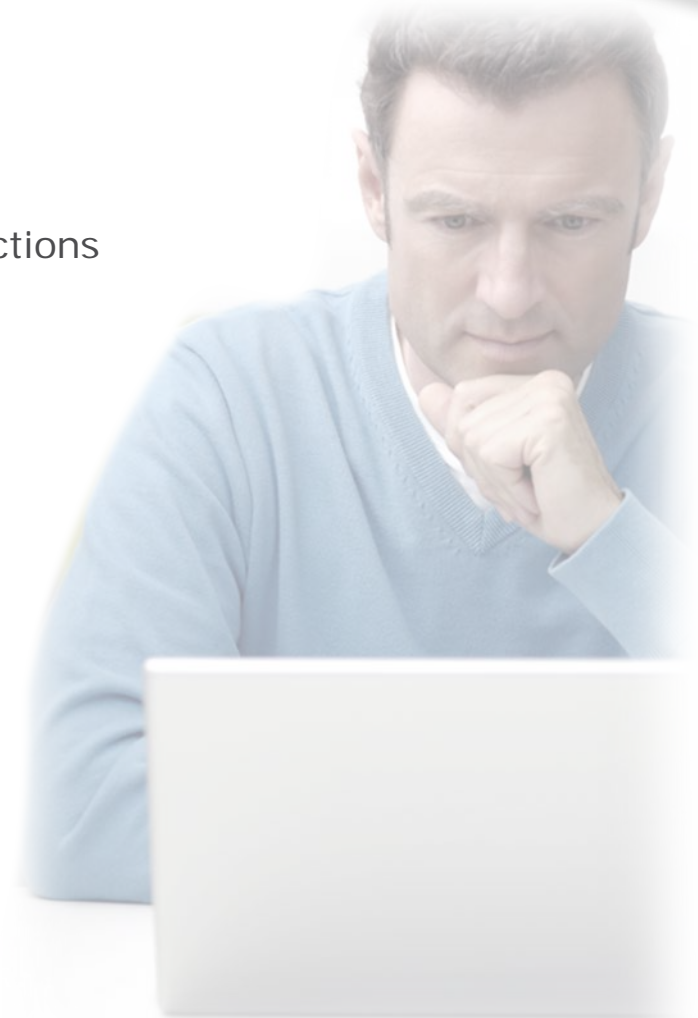
# Agenda

- PowerShell
- SQLPSX
- Conclusions
- Questions



# Why PowerShell?

- Powerful Scripting Environment
  - Automate regularly executed tasks
  - Extend the framework with new modules and functions
- Access to WMI
- Access to File System
- Access to COM
- Access to the .NET Framework
- Access to Specialised Storage



# Access to Windows Management Instrumentation (WMI)

```
Get-WmiObject Win32_logicaldisk `
| Where-Object { $_.DriveType -eq 3 } `
| Out-GridView
```



# Access to the File System

```
Get-ChildItem . -Include *.trn -Recurse `
| Where-Object { $_.LastWriteTime -lt (get-date).AddDays(-8) } `
| Remove-Item

Get-ChildItem . -Include *.dif -Recurse `
| Where-Object { $_.LastWriteTime -lt (get-date).AddDays(-31) } `
| Remove-Item

Get-ChildItem . -Include *.bak -Recurse `
| Where-Object { $_.LastWriteTime -lt (get-date).AddDays(-91) } `
| Remove-Item
```



# Access to COM

```
$a = New-Object -comobject Excel.Application
$a.Visible = $False

$b = $a.Workbooks.Add()
$c = $b.Worksheets.Item(1)

$c.Cells.Item(1,1) = "Hello Cambridge"
$b.SaveAs("C:\Presentations\SQLSaturday162\Demo.xls")

$a.Quit()
```



# Access to the .NET Framework

```
[System.Guid]::NewGuid()
```

# Troubleshooting SQL Server

PowerShell Troubleshooting Techniques



**The SQL Server Experts**  
Discover. Optimise. Mentor.



# Finding Failed Jobs

```
[reflection.assembly]::LoadWithPartialName("Microsoft.SqlServer.Smo")  
  
$Server = New-Object "Microsoft.SqlServer.Management.Smo.Server"  
$Server.jobserver.jobs |  
    where-object {$_.lastrunoutcome -eq "Failed" -and $_.isenabled -eq $true}
```



# SQL PSX

Open Source SQL PowerShell Extensions



**The SQL Server Experts**  
Discover. Optimise. Mentor.

# Download from Codeplex

The screenshot shows a web browser window displaying the CodePlex project page for SQLPSX. The browser's address bar shows the URL `http://sqlpsx.codeplex.com/relea`. The CodePlex header includes navigation links for Register, Sign In, and a search bar. The main content area features the SQLPSX logo and a navigation menu with 'DOWNLOADS' highlighted. The current release is identified as '2.3.2.1 Production'. A summary box provides details: Rating (5 stars based on 2 ratings), Reviewed (1 review), Downloads (15100), Change Set (58705), Released (Mar 13 2011), Updated (Mar 13 2011 by cmille19), and Dev status (Stable). A 'RECOMMENDED DOWNLOAD' section lists 'SQLPSX.msi' as an application, 1105K in size, uploaded on Mar 13 2011, with 10295 downloads. An 'OTHER DOWNLOADS' section lists previous releases: 2.3.2.1 Production (Mar 13 2011, Stable, 5 stars), 2.3.1 Production (Dec 11 2010, Stable), 2.3 Production (Nov 6 2010, Stable), 2.2.3 Beta (Jun 1 2010, Beta), and 2.2.2 Production.

`http://sqlpsx.codeplex.com/releases/view/62503`

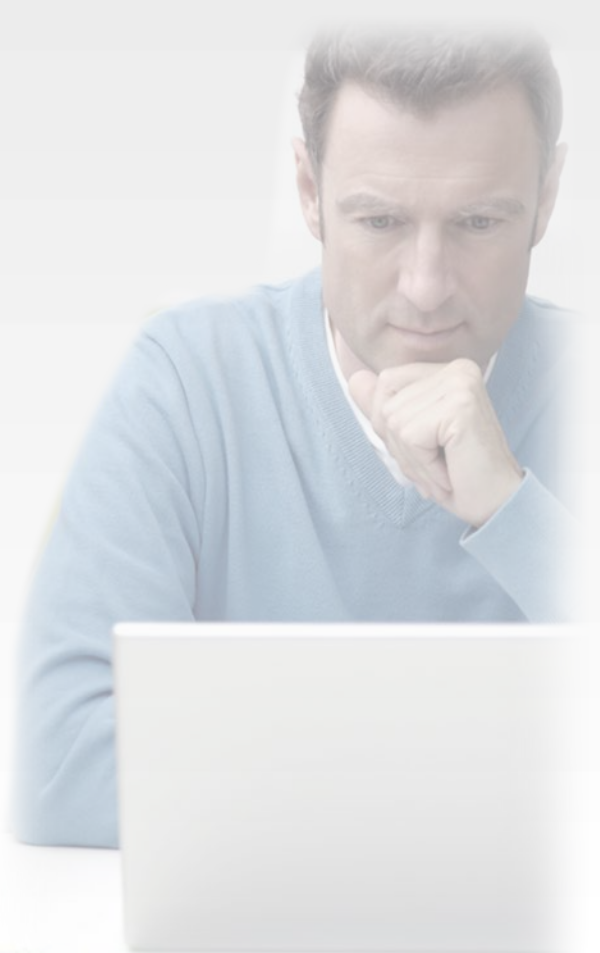
# SQLPSX Modules

- import-module adolib
- import-module SQLServer
- import-module Agent
- import-module Repl
- import-module SSIS
- import-module SQLParser
- import-module Showmbrs
- import-module SQLMaint
- import-module SQLProfiler
- import-module PerfCounters



import-module SQLPSX





# Demo

Gathering Performance Counter  
Data with PowerShell & SQLPSX



**The SQL Server Experts**  
Discover. Optimise. Mentor.

# Setting up PerfCounters

```
Import-Module PerfCounters
```

```
Get-PerfCounterCategory -CategoryName "SQLServer*" | Sort-Object Category_Name `
| Get-PerfCounterInstance -InstanceName 'MSSQL' `
| Get-PerfCounterCounters `
| Save-ConfigPerfCounter -PathConfigFile 'C:\PowerShell\SQLServerPerfcounter.xml' -NewFile
```

```
Get-PerfCounterCategory -CategoryName "*Buffer*" `
| Get-PerfCounterInstance `
| Get-PerfCounterCounters -CounterName "*Hit Ratio*" `
| Save-ConfigPerfCounter -PathConfigFile 'C:\PowerShell\SQLServerPerfcounter_[ComputerName].xml'
```

```
Get-PerfCounterCategory -CategoryName "Process" `
| Get-PerfCounterInstance -InstanceName "sqlservr" `
| Get-PerfCounterCounters -CounterName "% Processor Time" `
| Save-ConfigPerfCounter -PathConfigFile 'C:\PowerShell\SQLServerPerfcounter_[ComputerName].xml'
```



# Capture Data

```
Import-Module PerfCounters
```

```
Set-CollectPerfCounter -DateTimeStart "09/08/2012 08:00:00" -DateTimeEnd `
"09/08/2012 22:00:00" -Interval 10 -PathConfigFile `
C:\PowerShell\SQLServerPerfcounter_[ComputerName].xml -PathOutputFile `
C:\PowerShell\SQLServerPerfcounterData.txt -RunAsJob
```



# Load the data captured

```
Save-PerfCounterSQLTable -DatabaseName Master -NewTable `
-PathConfigFile C:\PowerShell\SQLServerPerfcounter_[ComputerName].XML -PathOutputFile `
C:\PowerShell\SQLServerPerfcounterData_[ComputerName].txt
```



# Conclusions

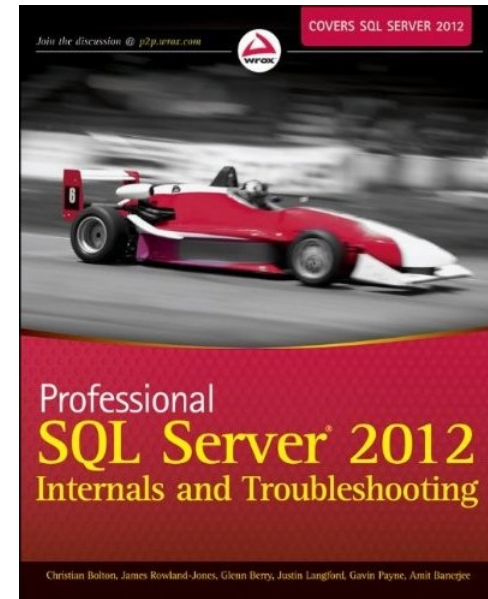
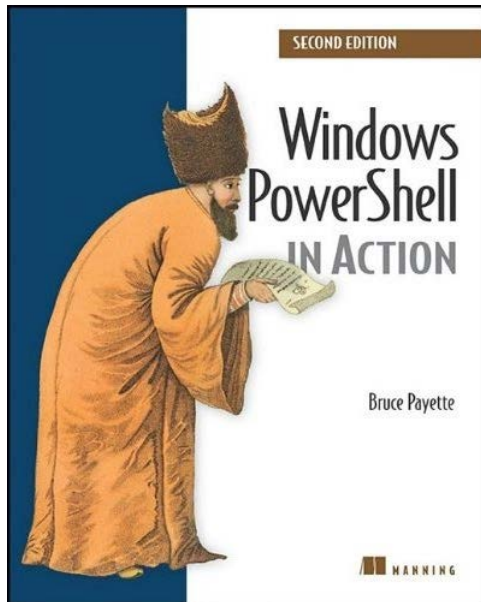
PowerShell is a modern object based scripting engine can be used to automate recurring tasks

PowerShell provides access to File System; Registry; WMI; COM; .NET Objects and Specialist Data Stores such as SQL Server

SQL PSX – Extends PowerShell by adding 13 modules with 163 advanced functions, 2 cmdlets and 7 scripts to help with SQL Related tasks



# Further Reading



# Questions

Thank you – Please feel free to contact me

James Boother

[james@coeo.com](mailto:james@coeo.com)

[@jimmyboo](#)



**The SQL Server Experts**  
Discover. Optimise. Mentor.