PASS SUMMIT 2015

POWERSHELL AND THE ART OF SQL SERVER DEPLOYMENT

Mike Fal

PASS

Please silence cell phones

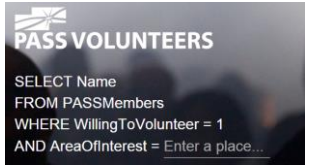
Explore Everything PASS Has to Offer

24 HOURS OF PASS

FREE ONLINE WEBINAR EVENTS

PASS SQL saturday

FREE 1-DAY LOCAL TRAINING EVENTS



VOLUNTEERING OPPORTUNITIES

PASS CHAPTERS

LOCAL USER GROUPS AROUND THE WORLD

PASS VIRTUAL CHAPTERS

ONLINE SPECIAL INTEREST USER GROUPS

PASS CommunityCONNECTOR

PASS COMMUNITY NEWSLETTER

FREE ONLINE RESOURCES

PASS BUSINESS ANALYTICS CONFERENCE 2016

BUSINESS ANALYTICS TRAINING

BA INSIGHTS NEWSLETTER

Session Evaluations

Your feedback is important and valuable.

3 ways to access

Submit by 5pm Friday November 6th to **WIN** prizes

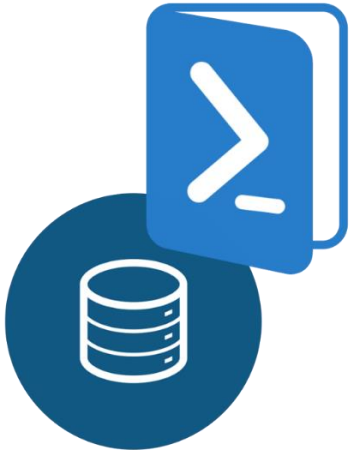
Go to **passSummit.com**

Download the GuideBook App and search: **PASS Summit 2015**

Follow the QR code link displayed on session signage throughout the conference venue and in the program guide

www.mikefal.net

IDERA
ACE



Mike Fal

Microsoft
CERTIFIED
Solutions Expert
Data Platform

Get-Agenda

- Why and How of Automation
- Tools to Deploy SQL Server
- The Build Plan
- The Build Demo

```
    $trns = Get-ChildItem $dir -recurse | Where-Object {$_.name -like "*.trn"} | sort-object LastWriteTime
}
else{
    $full = Get-ChildItem $dir | Where-Object {$_.name -like "*.bak"} | Sort-Object LastWriteTime -desc | Select-Object -first 1
    $diff = Get-ChildItem $dir | Where-Object {$_.name -like "*.dff"} | sort-object LastWriteTime -desc | select-object -first 1
    $trns = Get-ChildItem $dir | Where-Object {$_.name -like "*.trn"} | sort-object LastWriteTime
}

#initialize and process full backup
$outputfile = Join-Path -Path $outputdir -ChildPath "restore_$database.sql"
$restore = Get-RestoreObject $database $full
$hfull = Get-Header $restore $smosrv
if($database.Length -eq 0)
{
    $database = $hfull.DatabaseName
    $restore.Database=$database
}

$LSNCheck = $hfull.CheckpointLSN
$files = $restore.ReadFileList($smosrv)
foreach($file in $files){
    $pfile = $file.PhysicalName
    if($newdata.Length -gt 0 -and $file.Type -eq "D"){
        $pfile=$newdata + $pfile.Substring($pfile.LastIndexOf("\"))
    }

    if($newdata.Length -gt 0 -and $file.Type -eq "L"){
        $pfile=$newlog + $pfile.Substring($pfile.LastIndexOf("\"))
    }

    $newfile = New-Object("Microsoft.SqlServer.Management.Smo.RelocateFile") ($file.LogicalName,$pfile)
    $restore.RelocateFiles.Add($newfile) | out-null
}

$sqlout += "/*****************************************************"
$sqlout += "Restore Database Script Generated $(Get-Date)"
$sqlout += "Database: "+$database
$sqlout += "*****************************************************/"
$sqlout += "--FULL RESTORE"
If($Owner){$sqlout += "EXECUTE AS LOGIN = '$Owner';"}
$sqlout += $restore.Script($smosrv)

#process differential backups
if($diff -ne $null){
    $restore = Get-RestoreObject $database $diff
    $hdiff = Get-Header $restore $smosrv

    if($hdiff.DatabaseBackupLSN -eq $LSNCheck){
        $sqlout += "--DIFF RESTORE"
        $sqlout += $restore.Script($smosrv)
        $LSNCheck = $hdiff.LastLSN
    }
    else{
```

DO NOT PANIC

Automation: Why and How

Why?

Consistency	Reliability
Same way, every time	Errors are reduced
Everything gets done	Speed is the side effect

Downtime and instability happen when we have variance, not because we try to do things quickly. -Me

How?

Have a process!

- Runbooks
- Checklists

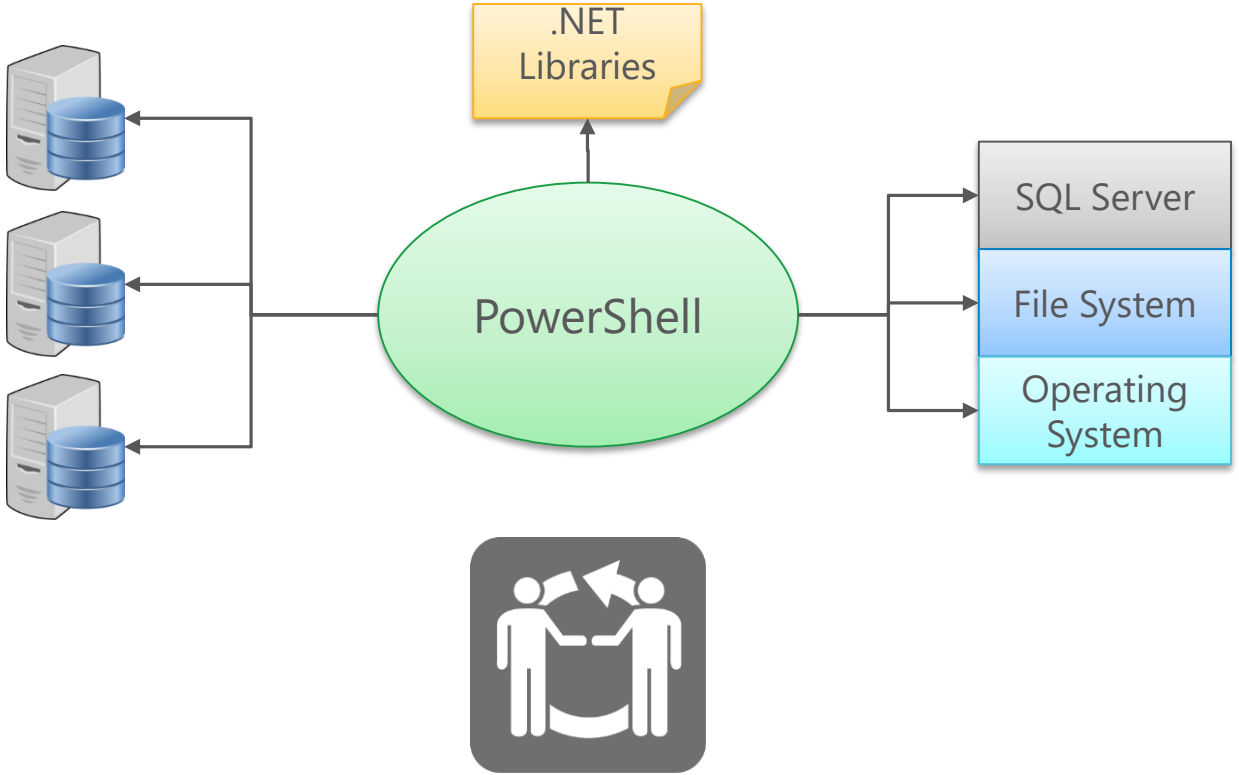
Have a tool!

- Powershell
- T-SQL

Have a script!

Tooling

Why PowerShell?



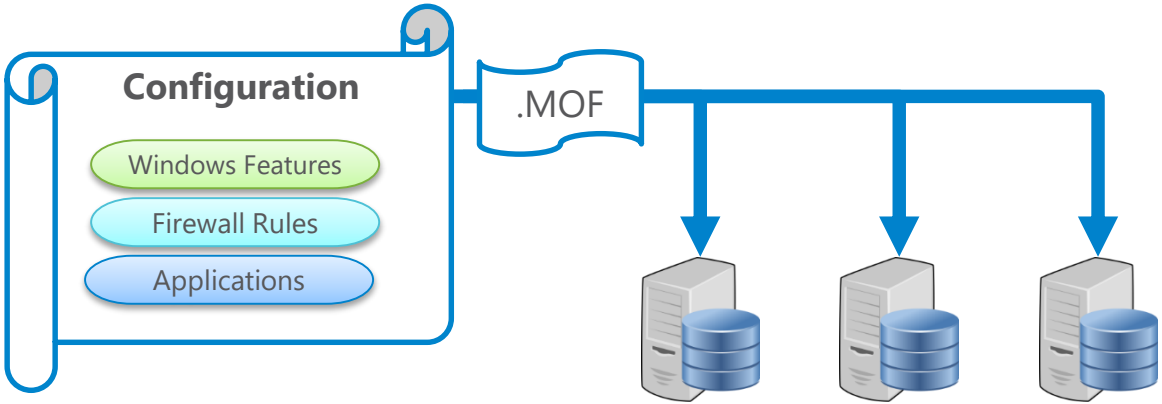
The Tools – Powershell DSC

"Killer Feature" in Powershell v4.0

Declarative **Configurations**

Resources define *what* and *how*

Create. Deploy. Execute.



The Tools – Configuration.INI

```
; SQL Server 2014 Configuration File
[OPTIONS]

; Specifies a Setup work flow, like INSTALL, UNINSTALL, or UPGRADE. This is a required
ACTION="Install"

;Set quiet mode ON, send log to command window
QUIET=True
INDICATEPROGRESS=True

; Specifies features to install, uninstall, or upgrade. The lists of features include
FEATURES=SQLENGINE

; Windows account(s) to provision as SQL Server system administrators.
; Domain/computer and account should match your environment
SQLSYSADMINACCOUNTS="SDF\Administrator"
```

The Tools – The SMO and SQLPS

The SMO

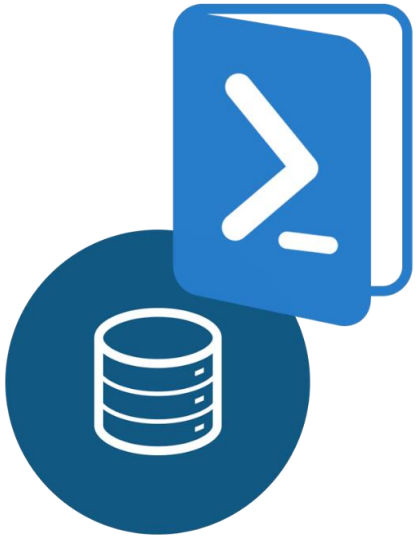
- .NET library for SQL Server

```
[System.Reflection.Assembly]::LoadWithPartialName
                ('Microsoft.SqlServer.SMO')*
```

SQLPS

- Powershell module for SQL Server
- Includes cmdlets and the provider

```
Import-Module SQLPS
```



Deploying SQL Server

The Goal

- ✓SQL Server 2014 SP1 CU2 installed on Server 2012 R2 Core (X 2)
- ✓SQL Server configured
- ✓Create an Availability Group
- ✓AdventureWorks2012 deployed to the Availability Group

The Process – Build the OS

OS Type – Windows Server 2012 R2 Core

Windows Features

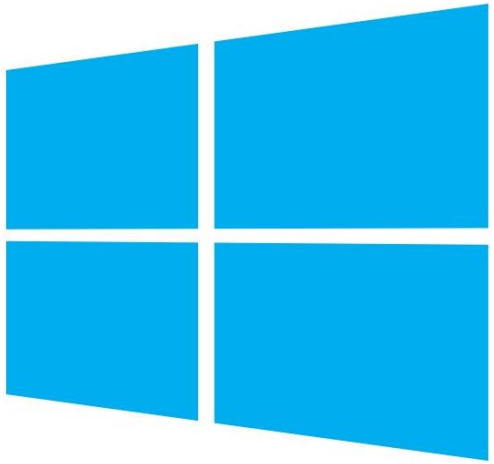
- .NET Core (required by SQL Server)
- Failover Clusters (to support Availability Groups)

Firewall Ports

- Allow 1433 (default SQL Server port)
- Allow 5022 (default HADR endpoint port)

Create directories for database files

- C:\DBFiles\Data
- C:\DBFiles\Log
- C:\DBFiles\TempDB



The Process – Install SQL Server

SQL Server – SQL 2014 SP1 CU 2 (Developer Edition)

Features

- SQL Engine

TCP Enabled

Service Accounts

- Same account for Engine and Agent

Default database directories

Agent Set to Auto Start

The Process – Configure SQL Server

Min and Max Memory

- Min = 0
- Max = 4 GB

Configurations

- MAXDOP = 2
- Optimize for Ad Hoc Workloads

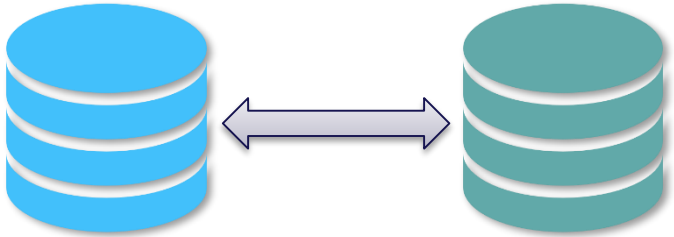
SQL Agent Retention

- Max Rows Per Job
- Max Total Rows

The Process – The Availability Group

Failover Cluster

- Two SQL Servers
- Fileshare Witness



Availability Group

- Two Nodes, Synchronous Commit
- Automatic Failover
- AdventureWorks2012

The Demo

Making Use of It

Physical Hardware

Reliable and consistent builds

Quick turn around after delivery

Build can be handed off to non-DBAs

Virtual Hardware

…all of that plus:

Deploy on demand

Treat servers as “disposable”

Expand and migrate quickly

Build is the same

- VMWare
- Azure
- AWS

DevOps

What is it?

Infrastructure as Code

- Consistency
- Control
- History

Code to Automation

Finish Line

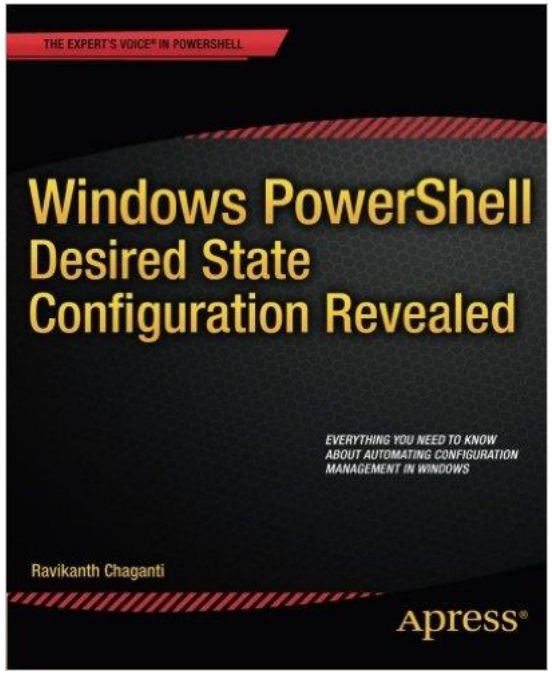
Scripting Server Builds

- Consistency
- Repeatability

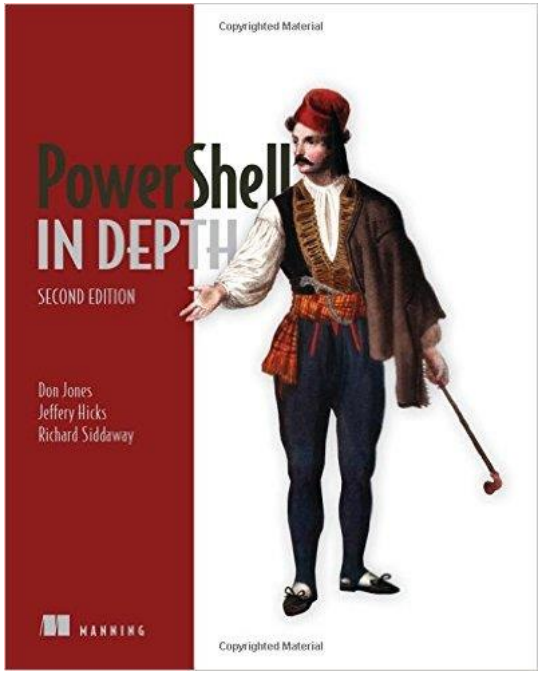
Speed through Automation

Automation Allows Rapid Growth

Resources

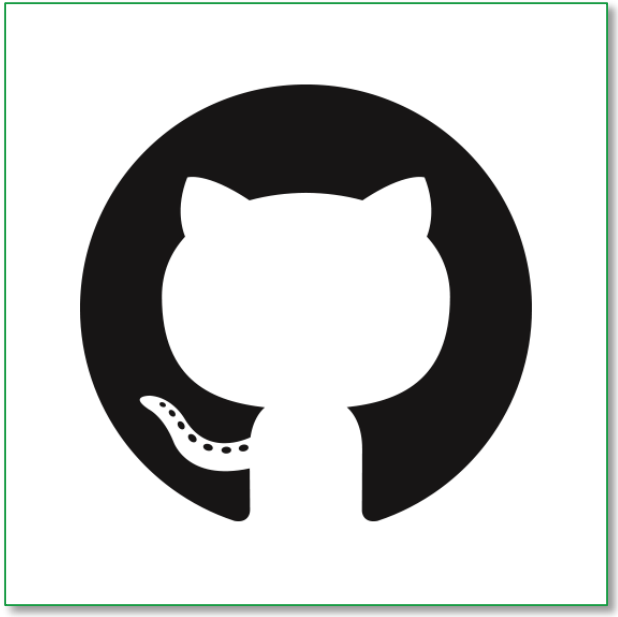


PowerShell.org



Script location

https://github.com/MikeFal/PresentationScripts/



mike@mikefal.net

www.mikefal.net

@Mike_Fal