Automation

Design Document

for

<Feature:Pre/Post Checklist for Server Reboot and PATCHING AND any other maintenance activity>

<Volvo >

Release Date: April 2022

Revision Cycle: Annually

Document Owner: Neha Verma

# Introduction

## Purpose

Purpose of this document is to outline the high-level design and architecture details of the proposed automation solution for **Feature “Pre/Post Checklist for Server Reboot and Patching and any other maintenance activity**”.

## Document Scope

This Document primarily covers following details.

Pre/Post Checklist for Server Reboot and Patching and any other maintenance activity

It will check SQL server services

Such as

* MS SQL Server Service
* SQL Server Agent
* MSDTC
* SQL browser service
* MS SQL Server OLAP Service
* Report Server service

## Definitions and Acronyms

### Definitions

| # | Term | Definition |
| --- | --- | --- |
| 1 | Python | Python is language which we can use for checking the services of SQL server |
| 2 | Python modules pypyodbc,wmi | Pypyodbc modules is used to connect with SQL server and WMI module is used for connect to windows server |

### Acronyms

| # | Acronym | Full Form |
| --- | --- | --- |
| 1 | SSMS | SQL Server management Studio |

# Automation Overview

## Objective

The Purpose of this solution is to automate

1. Save time and after server reboot post and pre checks of services will be easy to track the maintenance activity of the servers.
2. Reduce team effort for Pre-Post check after server reboot /patching.
3. Zero chance to miss any critical checks

## Functions

you can automate following functions/processes

* Function 1 <SQL server services checks before server reboot/patching >
* Function 2 < SQL server services checks after server reboot/patching >

## Users/Beneficiaries

This will be used and leveraged by following teams

* Wintel team
* DBA team
* Patching team

## Benefits

Following are the benefits of this Automation solution.

* Effort Savings
* Eliminate Human Errors.
* Faster Turn Around Time

# Requirements

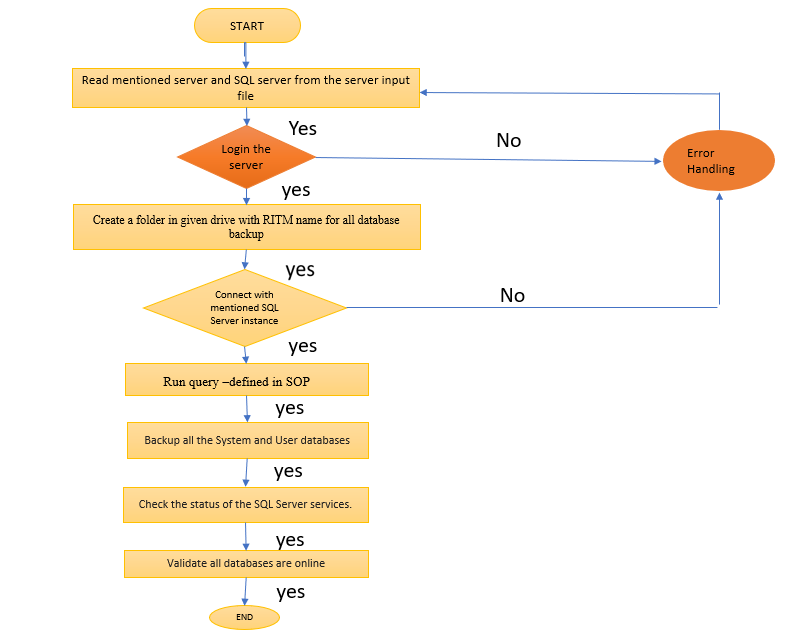
## Functional Requirements

This explains user’s explicit automation requirement

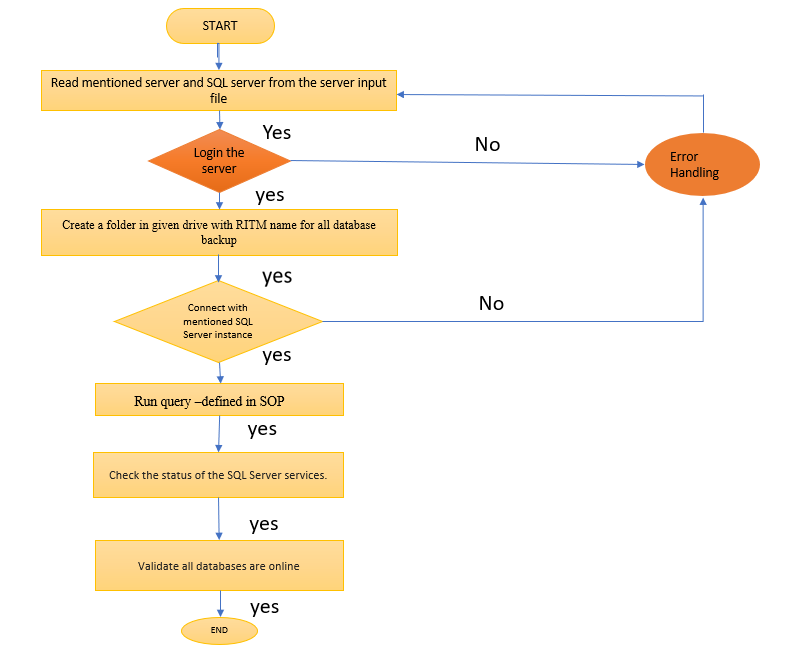
| Ref | Functionality | Description |
| --- | --- | --- |
| F1 | PYTHON VERSION | Python 3.8 and above |
| F2 | Python module | Wmi, pypyodbc |
| F3 | Create a folder in C drive | Folder name pyt |
| F4 | Inputserverlist.txt file | Contains windows server name and sql server name which will be act as an input for the script |

## Functional diagram

3.2.1 Functional diagram for Pre Validation Check



3.2.2 Functional Diagram for Post Validation Check



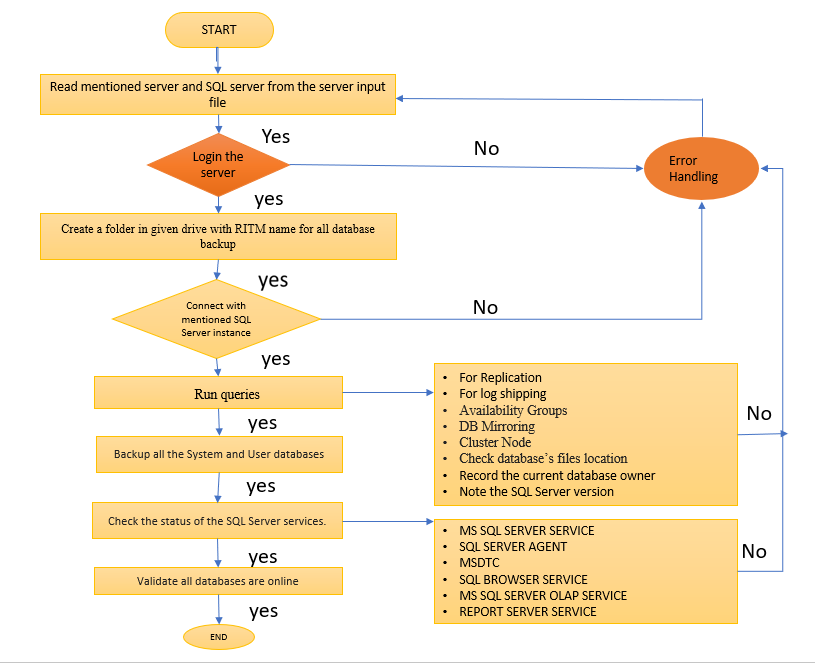
## Out-Of-Scope

* Any product, version, release difference than the automation offered.
* If not provide server list in inputserverlist.txt file
* Python version 2.x and below

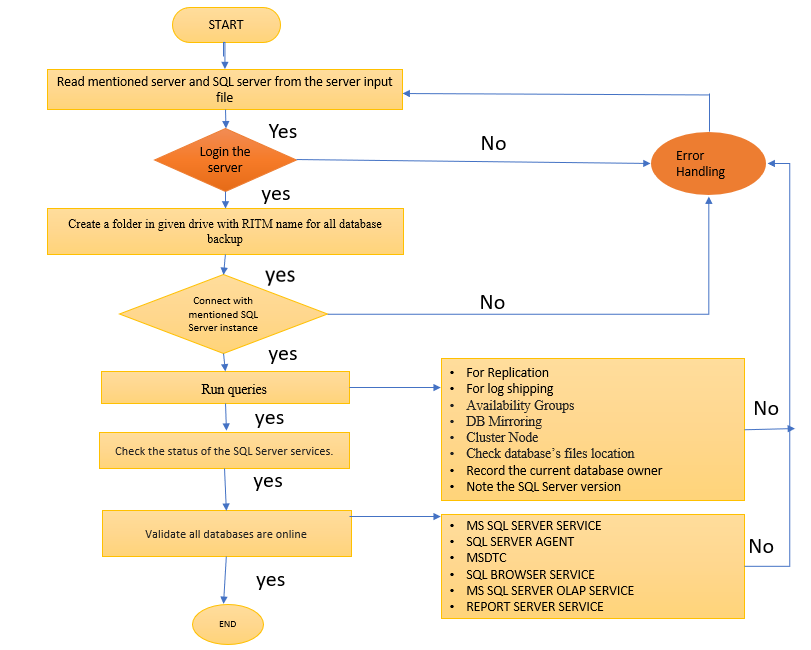
# Logical Architecture

Following diagram depicts the Logical Architecture of Automation Solution. For Pre/Post Validation Checklist for server reboot/patching and any other maintenance activity.

**4.1 Logical Architecture for Pre-Validation Checklist:**



## Logical architecture for Post-Validation Checklist



### Tools & Technologies Used

|  |  |  |
| --- | --- | --- |
| SL No | Requirement | Technology Used |
| 1 | Create pyt folder in C: drive | Python 3.8 and above |
| 2 | Create inputserverlist.txt file in pyt folder | SQL Server,windows server |
| 3 | Python Modules | wmi, pypyodbc |

# Pre-Requisites/Dependencies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Category | Prerequisites Description | Non Availability impact | Availability | Timelines for Availability | Owner |
| Python version and module | Python version must be 3.8 and above. Needed wmi and pypyodbc module |  | Delivery Confirmation needed |  |  |
| Pyt folder in C drive | Create a pyt folder in and create inputserverlist.txt file in pyt folder which contains windows server and sql server name exp windowserver , sqlserver |  | Delivery Confirmation needed |  |  |

### Assumptions/Constraints

|  |  |
| --- | --- |
| Sr No. | Description |
| 1 | Provide windows server name and SQL server name inputserverlist.txt file and create it in pyt folder |
| 2 | Passwords can be store in Windows credential manager on the RAS / Jump server itself |
| 3 | Allow all database access of SQL server |

# Technical Workflow

Technical Workflow of the components as follow

**8.1 Pre- Validation Check list**

1.Login to server

2. Create a folder in E:\drive with RITM name for all database backups.

3. Connect with mentioned SQL server/instance (mentioned inputserverlist.txt file).

4. Run query –Pre-Check

I. Review Replication, Log Shipping, Availability Groups, DB Mirroring and Cluster Node.

FOR Availability Groups

* If IsHadrEnabled = 1, Always on Availability Groups is enabled. Then inform the user by email.
* If IsHadrEnabled = 0, Always on Availability Groups is disabled.

II. Check the database's file’s locations.

III. Record the current database owner.

IV. Note the SQL Server Service Pack (SQL Server version)

5. Backup all the System and User databases in Created folder.

6. Check the status of the SQL Server services.

* MS SQL Server Service
* SQL Server Agent
* MSDTC
* SQL browser service
* MS SQL Server OLAP Service
* Report Server service

7. Validate all databases are online

8. Check free disk space

**8.2 Post Validation Checklist**

All the steps (expect backup step) we follow in the **“Pre Validation Checklist” should** be validated in **‘Post –Validation Checklist’**

# Logging:

Create a logging file of Pre-Post Validation checklist name precheck.log and postcheck.log.

Create Result file which stores all the data after executing the queries

* For Pre Validation check Creates Precheckresult and Prechecknotresult file.

Precheckresult gives all the information about access SQL server and instance

Postchecknotresult gives information about not access servers

* For post Validation Check create postcheckresult and postchecknotresult file.

Precheckresult gives all the information about access SQL server and instance

Postchecknotresult gives information about not access servers

# . Exception Handling:

Use try and Except for handling all type of Exceptions.

# 11. Solution Acceptance Criteria

* Are all critical checks covered?
* > Disk space checks
* > SQL services checks
* > Database Status
* Are summary reports generated for servers?