**CASE 1 (**Parameter Approach)

**Senior**

 I have an **on-premises SQL Server**, and I want to run a **SQL Query** on all the **Databases** and want to schedule it base on hourly, Daly, or monthly.

**Approach**

I will create **an ADF** pipeline. Add **For Each** activity to loop over **Database**. Add **Store Procedure** we can run any query using **stores procedure** on **Databased** and then I can **schedule** this pipeline on time-based using ADF Pipeline **Triggers**

**Case 1 Structure**

I will create a Global parameter which will have Array in Array Structure **2 Key:Value** one is **DB name** and second is **SP name**.

Pass this parameter in For Each Activity.in local parameters

**Prerequisite**

* Need a **SQL VM** Create using **Azure Portal** Need **ADF** Create using **Azure Portal**
* Need Some **Database** in your **SQL Server.**

**Step 1** Open **SQL VM** and restore 3 Databases.

In my case, I have databases **DB01, DB02, DB03.**

Graphical user interface, text, application

Description automatically generated

**Step 2** Create a **Store Procedure** in all DB.

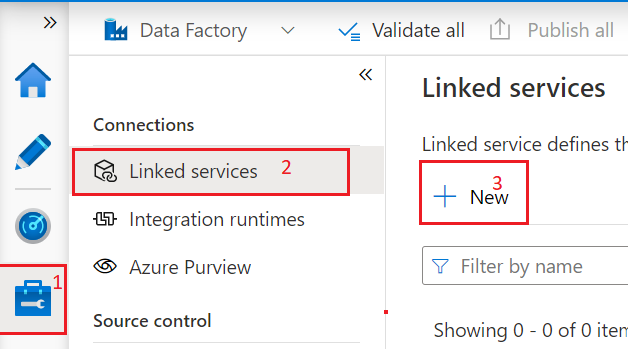
|  |  |
| --- | --- |
| Database | Store procedure |
| DB01 | [dbo].[spdb01] |
| DB02 | [dbo].[spdb02] |
| DB03 | [dbo].[spdb03] |

|  |
| --- |
| --I have the [Address] Table in all 3 DB with same schema.  CREATE PROCEDURE [dbo].[spdb01]  AS  BEGIN  UPDATE [Person].[Address] SET AddressLine2 = '1db01' where AddressID = 1;  END  GO |
| CREATE PROCEDURE [dbo].[spdb02]  AS  BEGIN  UPDATE [Person].[Address] SET AddressLine2 = '2db02' where AddressID = 1;  END  GO |
| CREATE PROCEDURE [dbo].[spdb03]  AS  BEGIN  UPDATE [Person].[Address] SET AddressLine2 = '3db03' where AddressID = 1;  END  GO |

**Step 3** Open ADF and configure **Self Hosted Integration Runtime** with **Link service** to connect **on-primes SQL-Server**.

3.1 Configure NEW **Link Service** follow the below Screenshot.

Click on **New** Button.



3.2 Windows will appear on the right side of the screen select SQL Server and click on Continue Button.

Graphical user interface, application

Description automatically generated

3.3 On this window Configure NEW Integrations runtime. Click on +NEW from the Drop-down.

Graphical user interface, text, application, email

Description automatically generated

3.4 Select Integrations runtime Self-Hosted option on this Windows. And click on Continue Button.

Graphical user interface, application

Description automatically generated

3.5 Click on Create button.

Graphical user interface, text, application, email

Description automatically generated

3.6 On this window Download the Software from the [option 1] link.

Install this Software in your SQL VM. and click on the close button. Graphical user interface, text, application, email

Description automatically generated

3.7 install **Integration runtime** Software on VM Machines. And configure it. Start the Server.

Graphical user interface, text, application, email

Description automatically generated

3.8 Select the **integration runtime** from the drop-down witch we have configured just now.

Graphical user interface, text, application, email

Description automatically generated

Important Step

**Step 4** On the same windows add all details of SQL VM.

4.1 Create a new parameter name “**DBN**” or any name from the bottom of this same window.

Graphical user interface, application

Description automatically generated

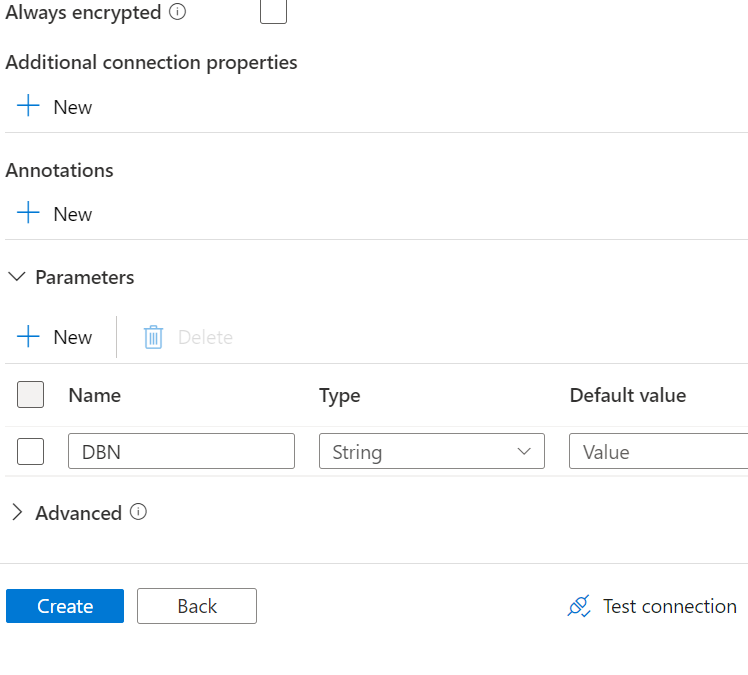
4.2 In Database name add this syntax.

|  |
| --- |
| @{linkedService().DBN} |

Graphical user interface, text, application, chat or text message, email

Description automatically generated

4.3 Check Connection Click on the **Test Connection** button.



4.4 Below windows will appear.Add database name in value Box “**DB01**” and click on Ok Button.

Graphical user interface, text, application

Description automatically generated

4.5 Click on **Create** buttons.

Graphical user interface, text, application

Description automatically generated

**Step 5** Create **a new** pipeline.Graphical user interface, text, application, email

Description automatically generated

5.1 Create new **parameter**. With

Name= “**ListARY**”

Type = **Array**

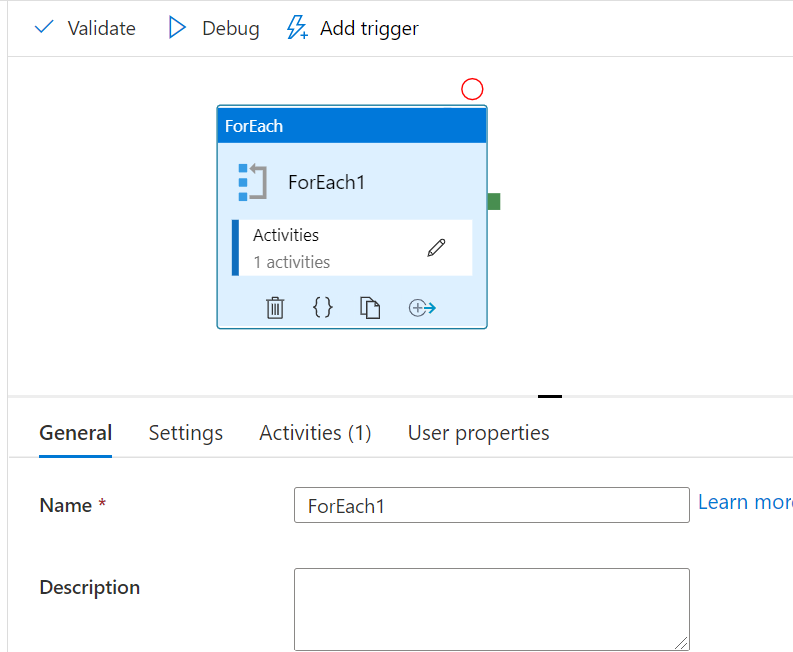
Default value = Value will be Array in Array examples below.

Graphical user interface, application

Description automatically generated

|  |
| --- |
| [  {  "DB":"DB01",  "SP":"[dbo].[spdb01]"  },  {  "DB":"DB02",  "SP":"[dbo].[spdb02]"  },  {  "DB":"DB03",  "SP":"[dbo].[spdb03]"  }  ] |

5.2 Add **ForEach** activity.



5.3 Click on **setting** and add this syntax.

|  |
| --- |
| @pipeline().parameters.ListARY |

Graphical user interface, text, application

Description automatically generated

5.3 Open **ForEach** activity add **Store Procedure** activity in it.

Graphical user interface, text, application

Description automatically generated

5.4 Click on **settings**. Add linked service click on the drop-down and select link service which we have configure above as “**Server01**”. Graphical user interface

Description automatically generated

5.5 Add value in parameter as below.

**DBN = @item().DB**

**Store procedure name = @item().SP**

Graphical user interface, text, application, email

Description automatically generated

5.6 Click on **Debug** Button.

Graphical user interface, application

Description automatically generated

5.7 you can pass a list of **databases** in which you have to execute the store procedure query.

And click on the ok button.

Graphical user interface, application, Word

Description automatically generated

5.8 pipeline ran successfully.

Graphical user interface, text, application, email

Description automatically generated

**Step 6** check the table data has updated as per my update query. In all the **Databases**.

DB01

Graphical user interface, text, application, table

Description automatically generated

DB02

Graphical user interface, table, Excel

Description automatically generated

DB03

Graphical user interface, application, table

Description automatically generated

**CASE 2 (**Lookup Approach)

**Senior**

#Same as CASE 1

**Approach**

#Same as CASE 1

**Case 2 Structure**

I will create a Table that will have **2 columns** one with **DB name and** the second with **SP name**.

I’ll load this table in the **lookup** activity.

I’ll pass the **parameter** Value in **For Each** activity from **Lookup** activity.

**Prerequisite**

* Need Some Database More than one (I have 3 DB in this example)
* Need Some Stored Procedure (I have one Store Procedure in each)
* Need to configure Self-hosted integration runtime (Reference Case 1 to do this)

**Step 1** Create a table with a column of **database names** and **Store Procedures** as given below.

Table name = **dbo.Para\_Table**

Graphical user interface, text, application

Description automatically generated

**Step 2** Add **lookup** activity.

Graphical user interface, application

Description automatically generated

**Step 3** Open **Settings** Configured New **Link Service**. And uncheck **First Row**.

Graphical user interface, text, application

Description automatically generated

**Step 4** Select **SQL Server** and Click on **Continue** Button.

Graphical user interface, application

Description automatically generated

**Step 5** create a new **link service** to get the Databasefrom **SQL Server.**

**Graphical user interface, text, application, email

Description automatically generated**

**Step 6** Insert all the Required Fields in **Database name** select the **database** where you have your main witch we have created in **Step 1.**

**Graphical user interface, text, application, email

Description automatically generated**

**Step 7** Select table from drop-down and click on **ok** button.

Graphical user interface, text, application, email

Description automatically generated

**Step 8** Add **ForEach** Activity.

Graphical user interface, text, application

Description automatically generated\

**8.1** Connect **lookup** with **For Each** and in **Settings** add below Syntax.

|  |
| --- |
| @activity('Lookup1').output.value |

Graphical user interface

Description automatically generated

**Step 9** Open **Foreach** and add **Store procedure** activity inside.

Graphical user interface, text, application, email

Description automatically generated

**9.1** Open **Settings** and configured New **Linked Service**.

Graphical user interface, text, application, email

Description automatically generated

**9.2** Create **parameter** as below.

Graphical user interface, text, application, email

Description automatically generated

**9.3** **Parameterized** the **Database** **name** as below with this **syntax**. and click on **Create** Button.

|  |
| --- |
| @{linkedService().DBN} |

Graphical user interface, text, application, email

Description automatically generated

**9.4** Add **parameters** value as bellow **Syntax** for **DBN** and **Store procedure.**

Graphical user interface, text, application, email

Description automatically generated

**Step 10** Click on **Debug** button.

Graphical user interface, text, application, email

Description automatically generated

**Step 11** check the table data has updated as per my update query. In all the **Databases**.

**DB01**

Graphical user interface, text, application, table

Description automatically generated

**DB02**

Graphical user interface, table, Excel

Description automatically generated

**DB03**

Graphical user interface, application, table

Description automatically generated

|  |  |
| --- | --- |
| **Case 1** | **Case 2** |
| **If you want to go with case 1?**  You need to have a good understanding of an array  To pass the parameter we can create it once and save it as a default.  In the future, when you have to change the database or stored procedure then you need to redesign the array for parameters value. | **If you want to go with case 2?**  You must create a table that will have all the database names and store procedure names and must be stored somewhere in a specific Database   or  if you cannot create a table in the available Database, you can create a new database to store this table.   if you want to apply this approach. |
| **Why this?**   IF you don’t want to create a table in any of the Database | **Why this?**  If you are not comfortable with parameter values that will have an array of array structure  Called as dictionary which have {key: value} representation. |
| If you go with case 1. If any update comes, then you need to edit the parameter array insides ADF Pipeline | If you go with case 2 you do not have to make change in ADF anything .if in future, you have to change any database or add removed some database or store procedure  you can simply update the Table in SQL Server |