

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**.

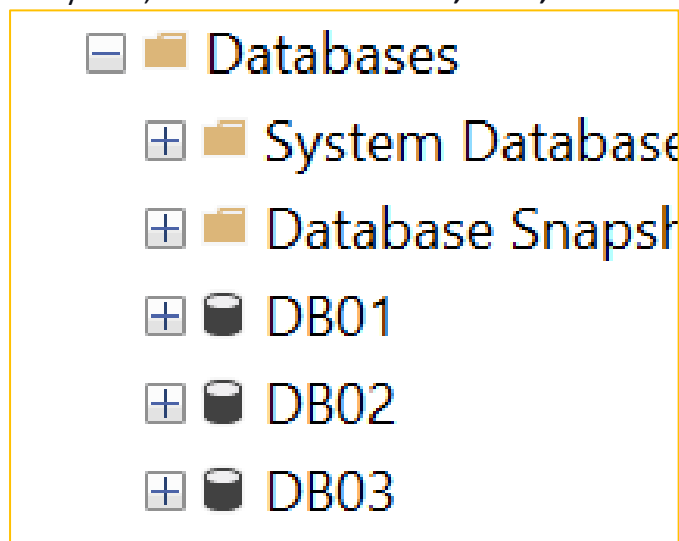
Pass this parameter in For Each Activity.in local parameters

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**.



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;
```

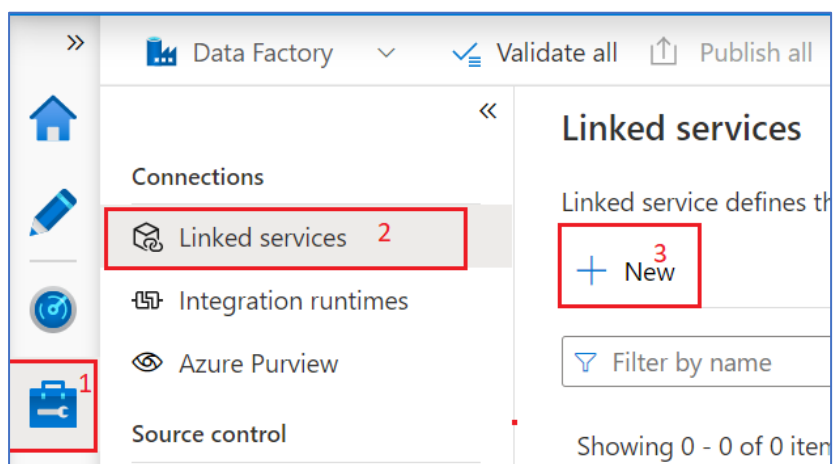
```
END
```

```
GO
```

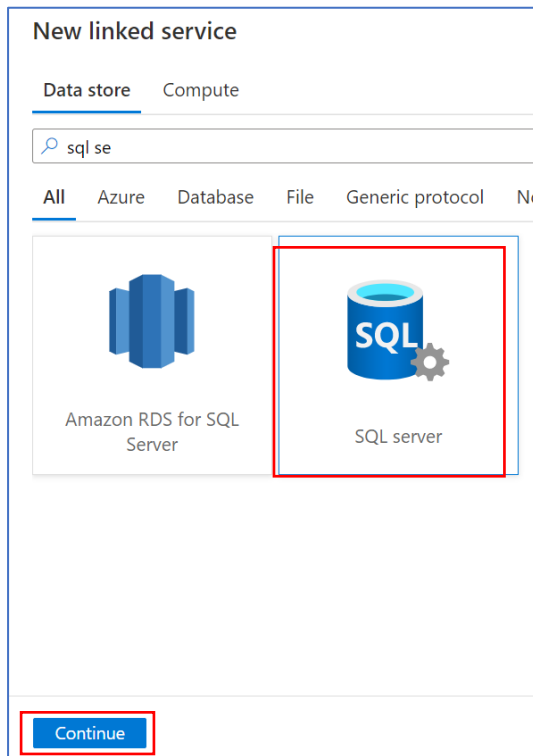
Step 3 Open ADF and configure **Self Hosted Integration Runtime** with **Link service** to connect **on-primers SQL-Server**.

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

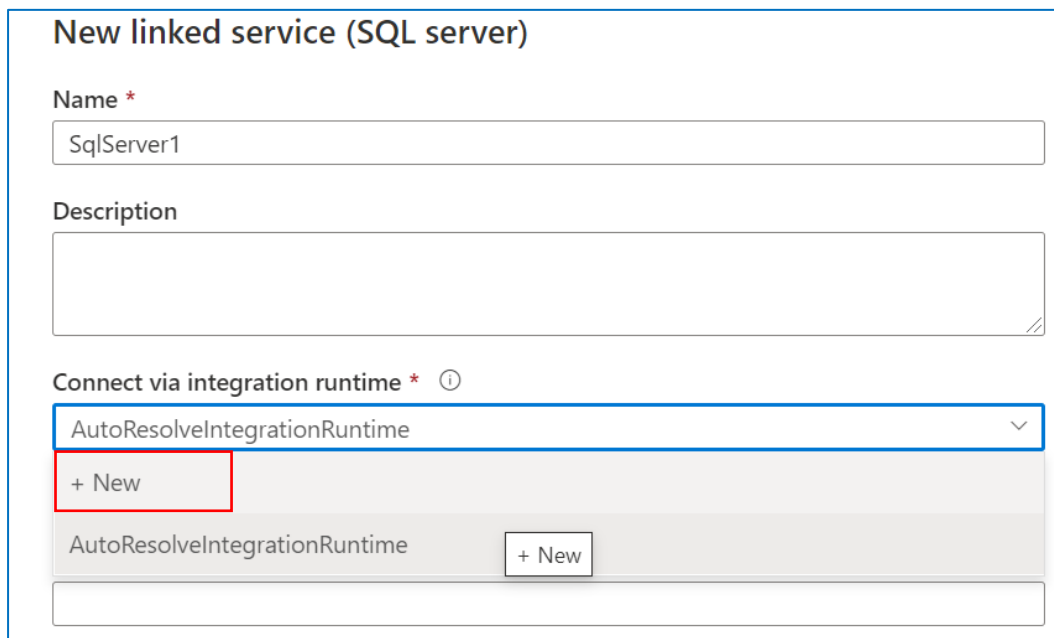
Click on **New** Button.



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




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




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

Integration runtime setup

the integration runtime will connect to for data flows, da

**Azure**


Use this for running data flows, data mo
in a fully managed, serverless compute


**Self-Hosted**

Use this for running activities in an on-p
[View more](#) ▾

External Resources:

You can use an existing self-hosted integration runtime t
you can reuse your existing infrastructure where self-hos

**Linked Self-Hosted**

[Learn more](#) 

Continue

3.5 Click on **Create** button.

Integration runtime setup

Private network support is realized by installing integration runtime to machines
on-premises network/VNET as the resource the integration runtime is connecting
below steps to register and install integration runtime on your self-hosted machi

Name * ⓘ

integrationRuntime1

Description

Enter description here...

Type

Self-Hosted

Create Back

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.

Integration runtime setup

SettingsNodesAuto updateSharingLinks

Install integration runtime on Windows machine or add further nodes using the Authentication Key.

Name ⓘ

integrationRuntime1





Option 1: Express setup

[Click here to launch the express setup for this computer](#)

Option 2: Manual setup

Step 1: [Download and install integration runtime](#)

Step 2: Use this key to register your integration runtime

Name	Authentication key	
Key1	IR@e64f709c-c80a-44fb-997c-fd4806c888ab@adf992@ServiceEndpoin	 
Key2	IR@e64f709c-c80a-44fb-997c-fd4806c888ab@adf992@ServiceEndpoin	 

Close

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

Microsoft Integration Runtime Configuration Manager

HomeSettingsDiagnosticsUpdateHelp

✓

Self-hosted node is connected to the cloud service

Data Factory:

adf911

Integration Runtime:

integrationRuntime1

Node:

TusharXmla

Stop Service

Data Source Credential ⓘ

Credential store:

On-premises

Credential status:

In sync

Last backup time:

N/A

Generate BackupImport Backup

✓ Connected to the cloud service (Data Factory V2)

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

New linked service (SQL server)

Name *

Description

Connect via integration runtime * ⓘ

✓ integrationRuntime1

+ New

AutoResolveIntegrationRuntime

✓ integrationRuntime1

Database name *

Authentication type

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.

Parameters

+ New

Delete

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	<input type="text" value="DBN"/>	<input type="text" value="String"/>	<input type="text" value="Value"/>

4.2 In Database name add this syntax.

@{linkedService().DBN}

Connect via integration runtime * ⓘ

✓ integrationRuntime1

⚠ The credentials are stored in the machines of self-hosted integration runtime if you don't choose to store them in Azure Key Vault.

Connection string Azure Key Vault

Server name *

TusharXmla

Database name *

@{linkedService().DBN}

Authentication type

SQL authentication

User name *

test123

Password Azure Key Vault

Password *

.....

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

Always encrypted ⓘ ☐

Additional connection properties

+ New

Annotations

+ New

Parameters

+ New | Delete

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	DBN	String	Value

> Advanced ⓘ

Create Back Test connection

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

Please provide actual value of the parameters to test connection

Parameters for linked service Server01

Name	Type	Value
DBN	string	<input type="text" value="DB01"/>

4.5 Click on **Create** buttons.

Always encrypted ⓘ ☐

Additional connection properties

+ New

Annotations

+ New

Parameters

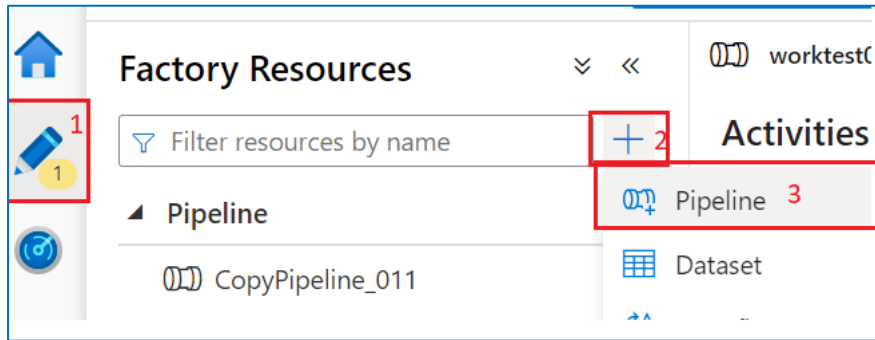
+ New

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	<input type="text" value="DBN"/>	<input type="text" value="String"/> ▼	<input type="text" value="Value"/>

> Advanced ⓘ

☒ Connection successful

Step 5 Create a new pipeline.

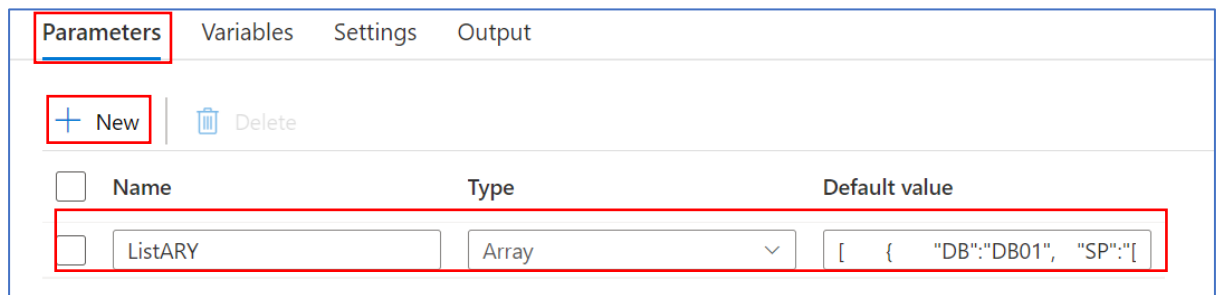


5.1 Create new **parameter**. With

Name= "**ListARY**"

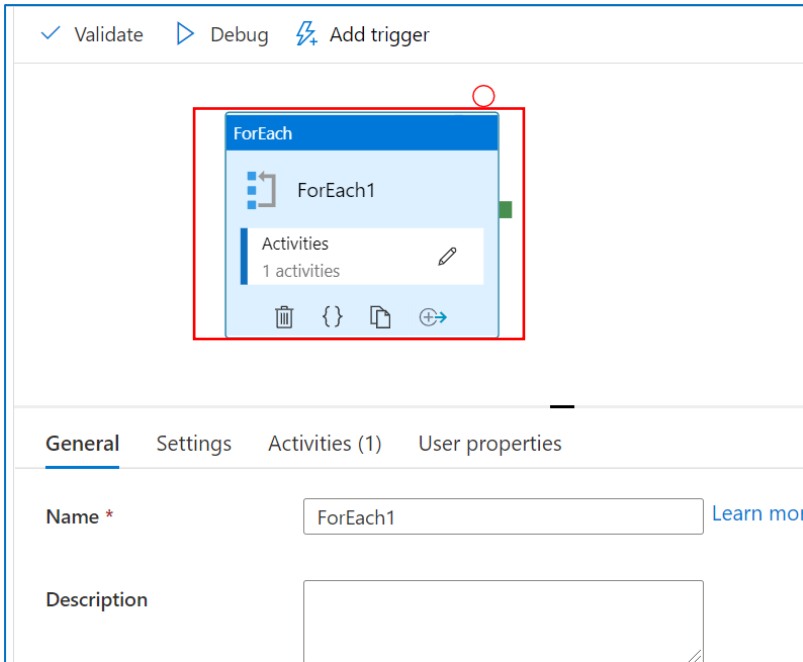
Type = **Array**

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



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

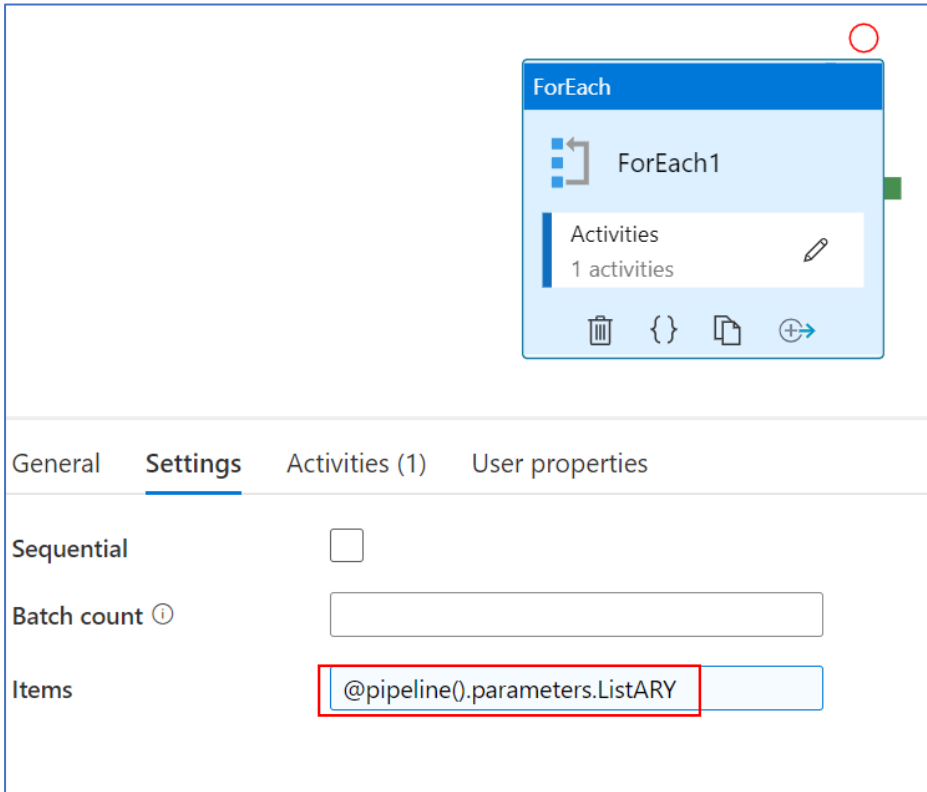
5.2 Add **ForEach** activity.



The screenshot shows the Azure DevOps pipeline editor interface. At the top, there are buttons for 'Validate', 'Debug', and 'Add trigger'. Below this, a 'ForEach' activity is being added to the pipeline. The activity is represented by a blue box with a 'ForEach' title and a 'ForEach1' icon. Below the icon, there is a section for 'Activities' showing '1 activities'. At the bottom of the activity box, there are icons for deleting, adding, and saving. Below the activity box, there are tabs for 'General', 'Settings', 'Activities (1)', and 'User properties'. The 'General' tab is selected, showing the 'Name' field with the value 'ForEach1' and a 'Description' field.

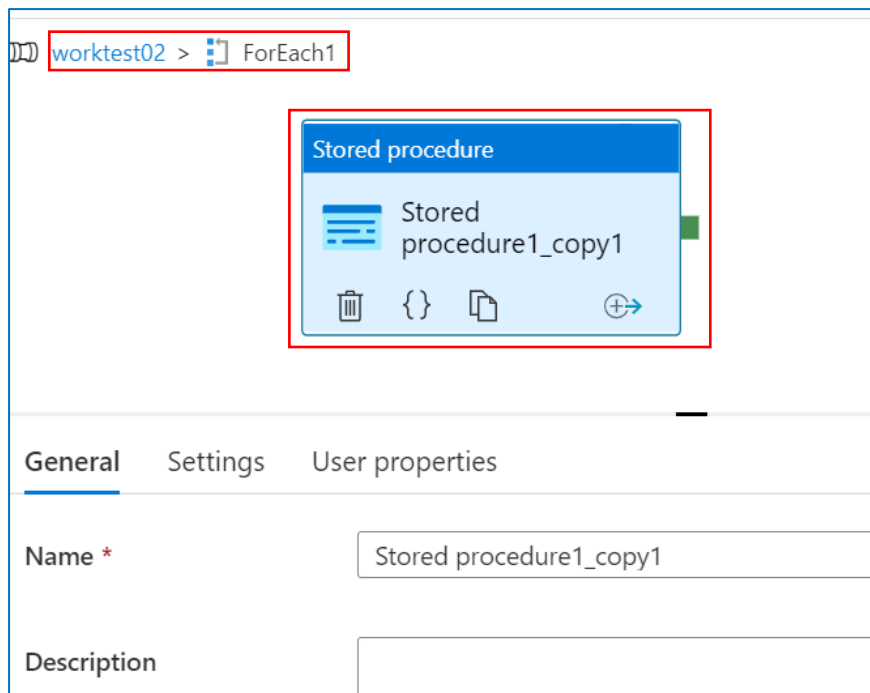
5.3 Click on **setting** and add this syntax.

`@pipeline().parameters.ListARY`

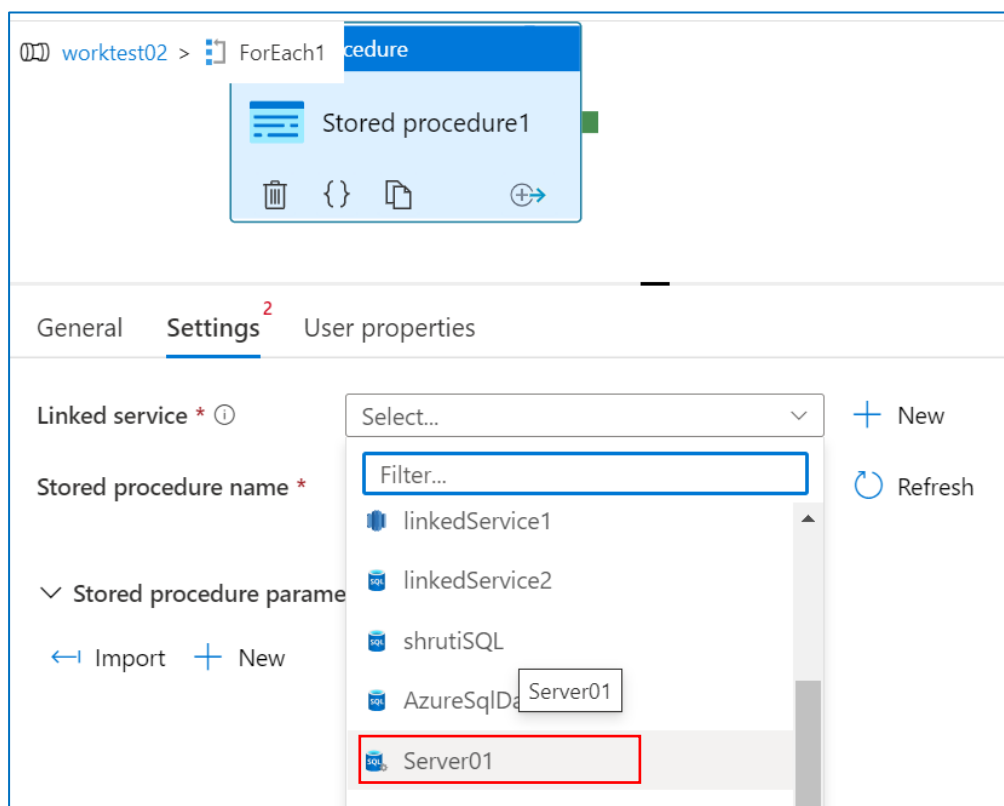


The screenshot shows the Azure DevOps pipeline editor interface with the 'ForEach' activity selected. The 'Settings' tab is active, showing the 'Sequential' checkbox, the 'Batch count' field, and the 'Items' field. The 'Items' field contains the syntax '@pipeline().parameters.ListARY', which is highlighted with a red box. The 'ForEach' activity box is also visible, showing the 'ForEach1' icon and the 'Activities' section with '1 activities'.

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



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



5.5 Add value in parameter as below.

DBN = @item().DB

Store procedure name = @item().SP

Loop_DB_Apply_SP_copy2 > ForEach1

Stored procedure1_copy1_copy1

General Settings User properties

Linked service * Server01 Test c

▼ Linked service properties ⓘ

Name	Value
DBN	@item().DB

Integration runtime * integrationRuntime1 Edit

Stored procedure name @item().SP

▼ Stored procedure parameters ⓘ

5.6 Click on **Debug** Button.

✓ Validate Debug Add trigger

ForEach

ForEach1

Parameters Variables Settings Output

+ New | Delete

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	ListARY	Array	[{ "DB":"DB01", "SP":["

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.

Pipeline run

Parameters

Name	Type	Value
ListARY	array	[{ "DB":"DB01", "SP"...

OK

Cancel

5.8 pipeline ran successfully.

✓ Validate

⌛ Cancel options

⚡ Add trigger

ForEach

ForEach1

Activities
1 activities

Parameters

Variables

Settings

Output

Pipeline run ID: 2cf7e231-718f-402e-bc35-3a411f820b86

🔄

📄

Name	Type	Run start	Duration	Status	Integration runtime
Stored procedure1_copy1	Stored proce...	2021-11-04T07:00:49.414	00:00:19	✓ Succeeded	integrationRuntime1
Stored procedure1_copy1	Stored proce...	2021-11-04T07:00:47.883	00:00:11	✓ Succeeded	integrationRuntime1
Stored procedure1_copy1	Stored proce...	2021-11-04T07:00:47.399	00:00:08	✓ Succeeded	integrationRuntime1
ForEach1	ForEach	2021-11-04T07:00:45.898	00:00:24	✓ Succeeded	

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

DB01

	AddressID	AddressLine1	AddressLine2
1	1	1970 Napa Ct.	1db01
2	2	9833 Mt. Dias Blv.	NULL
3	3	7484 Roundtree Drive	NULL
4	4	9539 Glenside Dr	NULL
5	5	1226 Shoe St.	NULL

DB02

	AddressID	AddressLine1	AddressLine2
1	1	1970 Napa Ct.	2db02
2	2	9833 Mt. Dias Blv.	NULL
3	3	7484 Roundtree Drive	NULL
4	4	9539 Glenside Dr	NULL
5	5	1226 Shoe St.	NULL
6	6	1399 Firestone Drive	NULL
7	7	5672 Hale Dr	NULL

DB03

	AddressID	AddressLine1	AddressLine2
1	1	1970 Napa Ct.	3db03
2	2	9833 Mt. Dias Blv.	NULL
3	3	7484 Roundtree Drive	NULL
4	4	9539 Glenside Dr	NULL
5	5	1226 Shoe St.	NULL
6	6	1399 Firestone Drive	NULL

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.

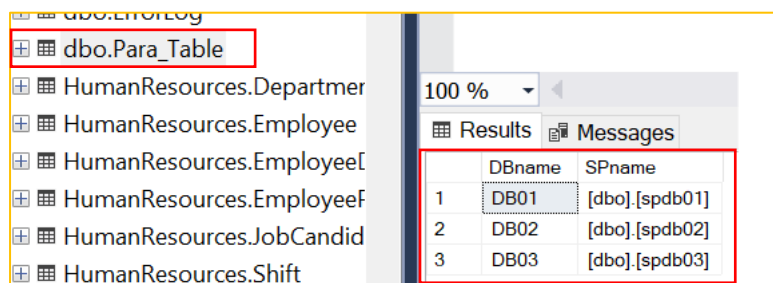
I'll pass the **parameter** Value in **For Each** activity from **Lookup** activity.

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.

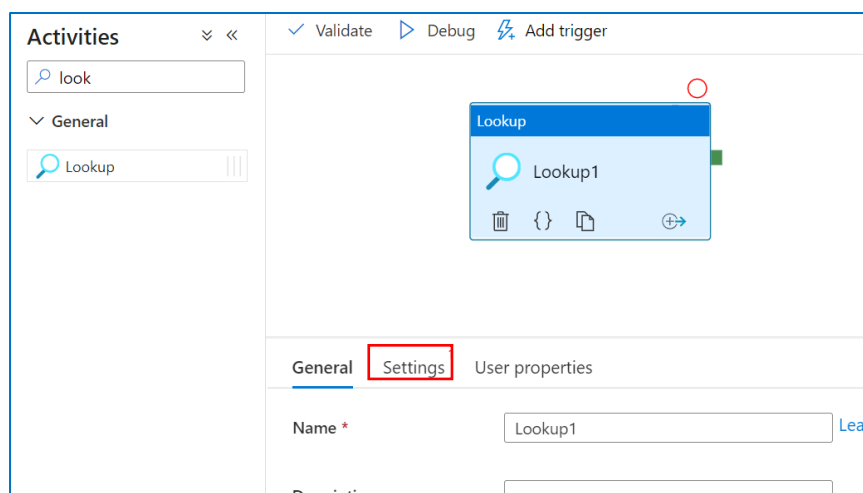
Table name = **dbo.Para_Table**



The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'dbo.Para_Table' table is highlighted in the tree view. On the right, the 'Results' pane displays the table's data. The table has two columns: 'DBname' and 'SPname'. It contains three rows of data.

	DBname	SPname
1	DB01	[dbo].[spdb01]
2	DB02	[dbo].[spdb02]
3	DB03	[dbo].[spdb03]

Step 2 Add **lookup** activity.



The screenshot shows the Microsoft Power Automate interface. The 'Activities' pane on the left has a search bar with 'look' and a 'Lookup' activity is selected. The main workspace shows the 'Lookup' activity icon. Below the workspace, the 'Settings' tab is selected, showing the 'Name' field with the value 'Lookup1'.

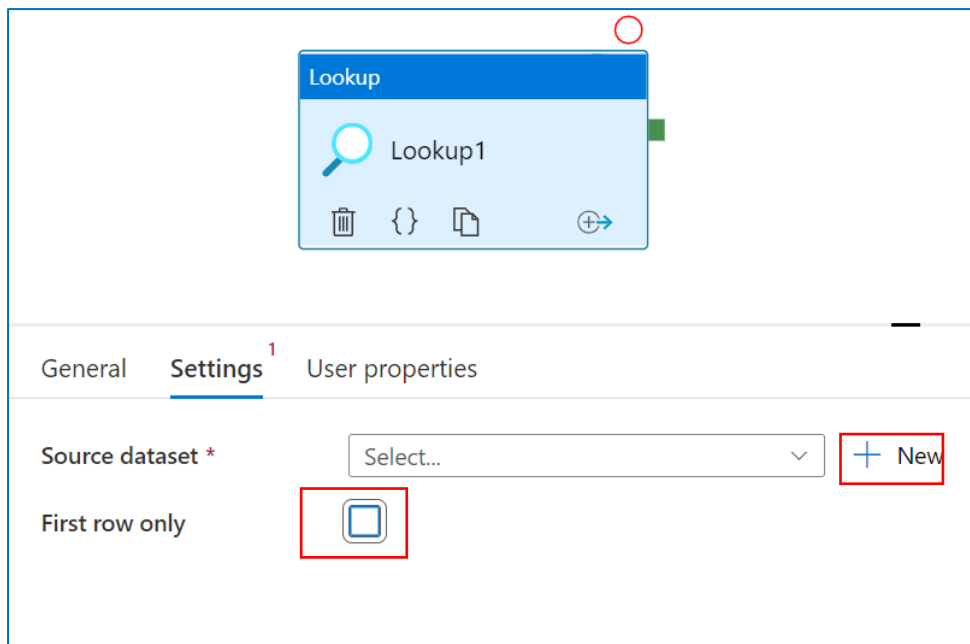
Activities: look

General: Lookup

Lookup activity: Lookup1

Settings: Name * Lookup1

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



Lookup

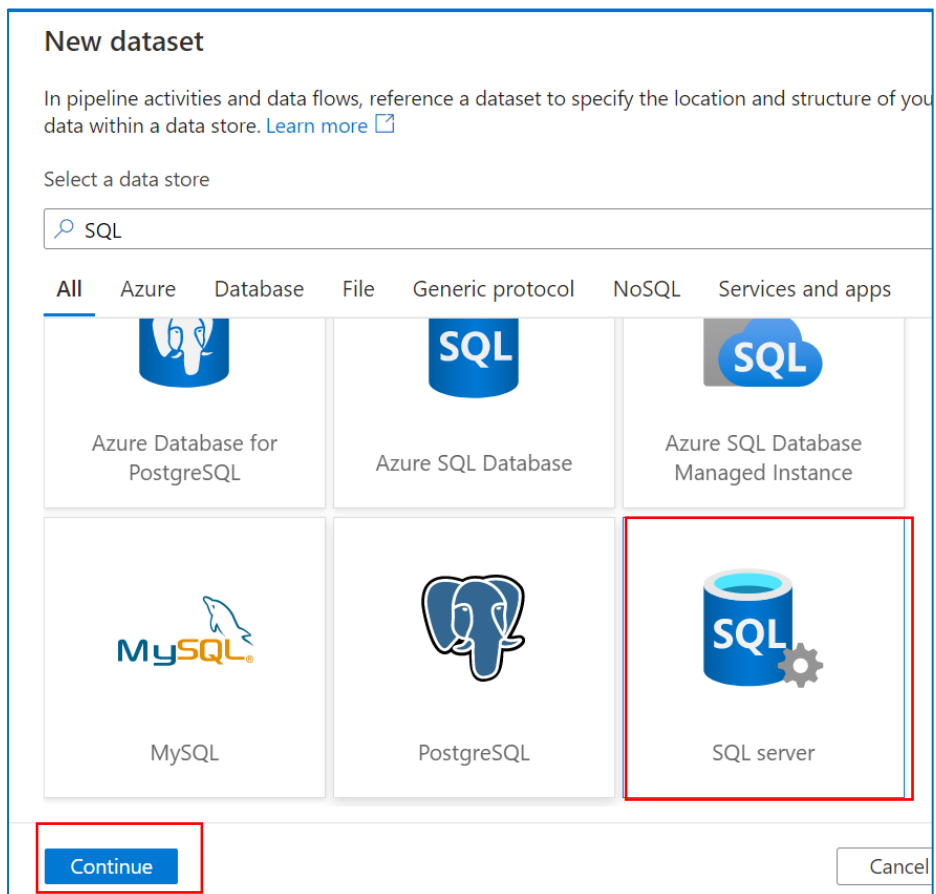
Lookup1

General **Settings** ¹ User properties

Source dataset * Select... + New

First row only ☐

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









New dataset

In pipeline activities and data flows, reference a dataset to specify the location and structure of your data within a data store. [Learn more](#)

Select a data store

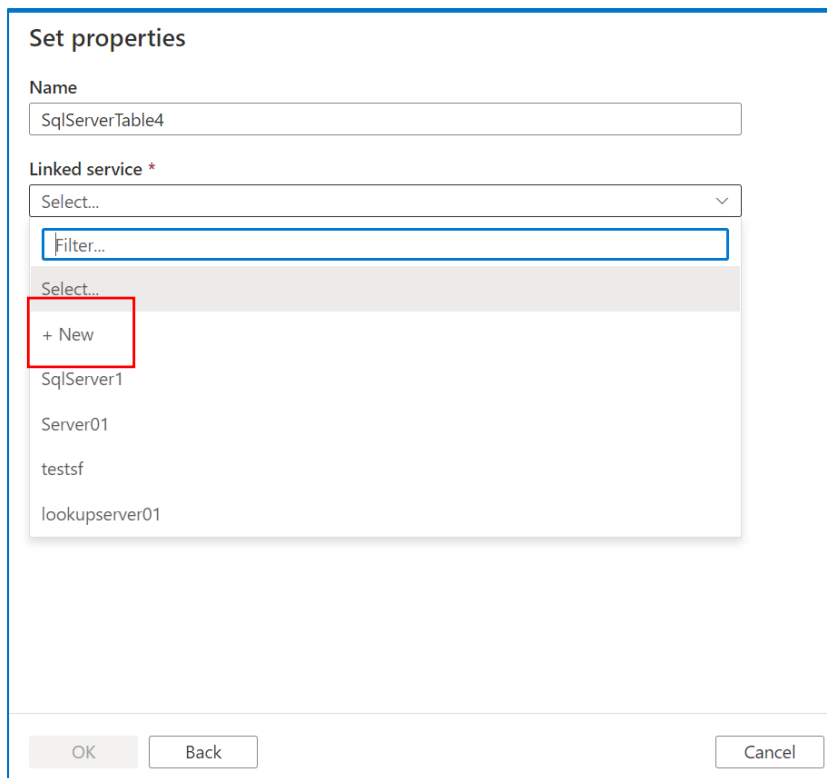
SQL

All Azure Database File Generic protocol NoSQL Services and apps

 Azure Database for PostgreSQL	 Azure SQL Database	 Azure SQL Database Managed Instance
 MySQL	 PostgreSQL	 SQL server

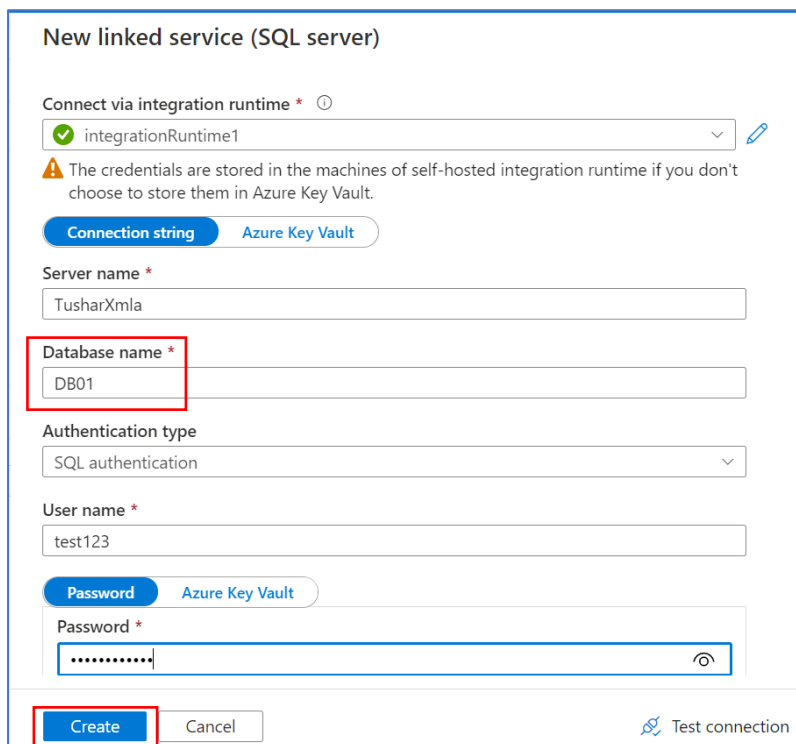
Continue Cancel

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



The screenshot shows the 'Set properties' dialog box. The 'Name' field is set to 'SqlServerTable4'. The 'Linked service *' dropdown menu is open, showing a search filter 'Filter...' and a list of existing services: 'SqlServer1', 'Server01', 'testsf', and 'lookupserver01'. A red box highlights the '+ New' option at the top of the list. At the bottom of the dialog are 'OK', 'Back', and 'Cancel' buttons.

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**.



The screenshot shows the 'New linked service (SQL server)' dialog box. The 'Connect via integration runtime *' dropdown is set to 'integrationRuntime1'. A warning message states: 'The credentials are stored in the machines of self-hosted integration runtime if you don't choose to store them in Azure Key Vault.' Below this are two tabs: 'Connection string' (selected) and 'Azure Key Vault'. The 'Server name *' field contains 'TusharXmla'. The 'Database name *' field contains 'DB01' and is highlighted with a red box. The 'Authentication type' dropdown is set to 'SQL authentication'. The 'User name *' field contains 'test123'. Below this are two tabs: 'Password' (selected) and 'Azure Key Vault'. The 'Password *' field contains masked characters and is highlighted with a red box. At the bottom are 'Create' and 'Cancel' buttons, with the 'Create' button highlighted by a red box. A 'Test connection' button with a lock icon is also present.

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

Set properties

Name
SqlServerTable4

Linked service *
SqlServer2

Connect via integration runtime * ⓘ
✓ integrationRuntime1

Table name
dbo.Para_Table

☐ Edit

Import schema
☒ From connection/store ☐ None

> Advanced

OK Back Cancel

Step 8 Add **ForEach** Activity.

Activities for

Move & transform
Copy data
Data flow

Iteration & conditionals
ForEach

Lookup
Lookup1

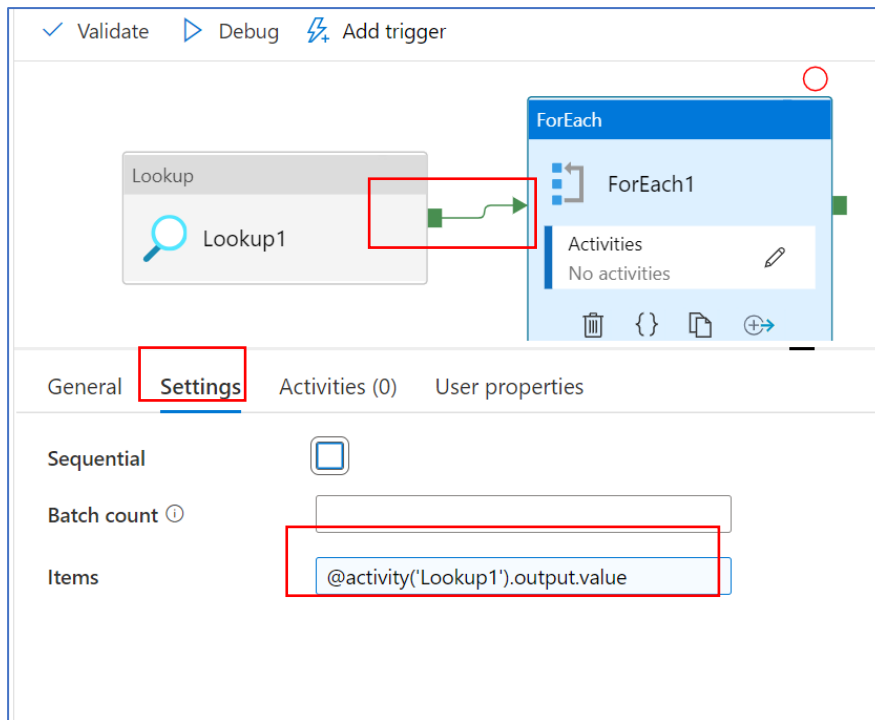
ForEach
ForEach1
Activities
No activities

General Settings¹ Activities (0) User properties

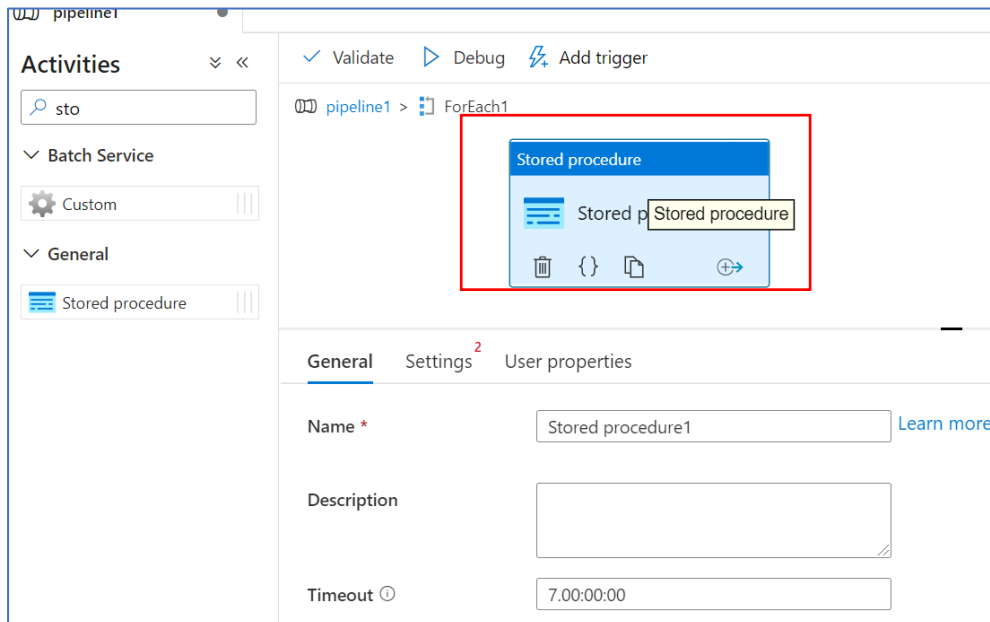
Name * ForEach1 [Learn more](#)

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

`@activity('Lookup1').output.value`



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



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

Validate Debug Add trigger

pipeline1 > ForEach1

Stored procedure

Stored procedure1

General **Settings²** User properties

Linked service * ⓘ Select... **+ New**

Stored procedure name * Select... Refresh

☐ Edit ⓘ

Stored procedure parameters ⓘ

Import + New

9.2 Create **parameter** as below.

Edit linked service (SQL server)

Password *

Add dynamic content [Alt+Shift+D]

Always encrypted ⓘ ☐

Additional connection properties

+ New

Annotations

+ New

Parameters

+ New Delete

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	DBN	String	Value

> Advanced ⓘ

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

@{linkedService().DBN}

New linked service

Connect via integration runtime * ⓘ
integrationRuntime1

⚠ The credentials are stored in the machines of self-hosted integration runtime if you don't choose to store them in Azure Key Vault.

Connection string Azure Key Vault

Server name *
TusharXmIa

Database name *
@linkedService().DBN

Authentication type
SQL authentication

User name *
test123

Add dynamic content [Alt+Shift+D]

Password Azure Key Vault

Password *
.....

Create Cancel Test connection

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

Loop_DB_Apply_lookup_main02 > ForEach1

Stored procedure1_copy1_c...

General Settings User properties

Linked service * ⓘ
Server01 Test connection Edit +

Linked service properties ⓘ

Name	Value
DBN	@item().DBname

Integration runtime *
integrationRuntime1 Edit

Stored procedure name
@item().SPname

Stored procedure parameters ⓘ

Step 10 Click on **Debug** button.

Validate Debug Add trigger

Lookup1

ForEach1

Activities
1 activities

Parameters Variables Settings **Output**

Pipeline run ID: 304abafa-235a-48b7-b781-3f3d750cc89a [@] View debug r

Name	Type	Run start	Duration	Status	Integration runtime
Stored procedure1_copy1_copy1	Stored proce...	2021-11-06T10:45:18.272	00:00:07	✓ Succeeded	integrationRuntime1
Stored procedure1_copy1_copy1	Stored proce...	2021-11-06T10:45:18.256	00:00:03	✓ Succeeded	integrationRuntime1
Stored procedure1_copy1_copy1	Stored proce...	2021-11-06T10:45:18.209	00:00:07	✓ Succeeded	integrationRuntime1
ForEach1	ForEach	2021-11-06T10:45:17.366	00:00:09	✓ Succeeded	

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

DB01

	AddressID	AddressLine1	AddressLine2
1	1	1970 Napa Ct.	1db01
2	2	9833 Mt. Dias Blv.	NULL

DB02

	AddressID	AddressLine1	AddressLine2
1	1	1970 Napa Ct.	2db02
2	2	9833 Mt. Dias Blv.	NULL

DB03

	AddressID	AddressLine1	AddressLine2
1	1	1970 Napa Ct.	3db03
2	2	9833 Mt. Dias Blv.	NULL

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