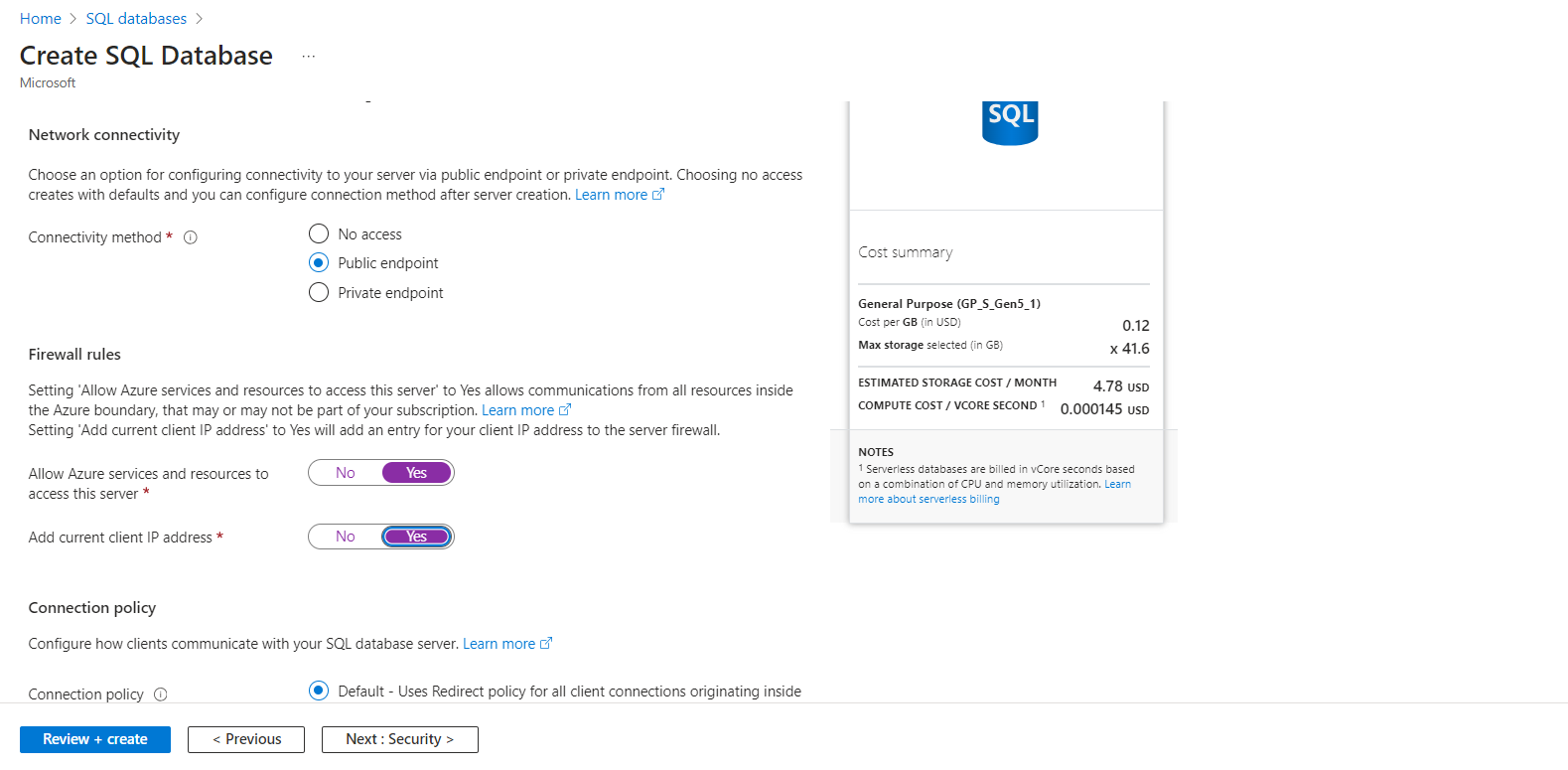
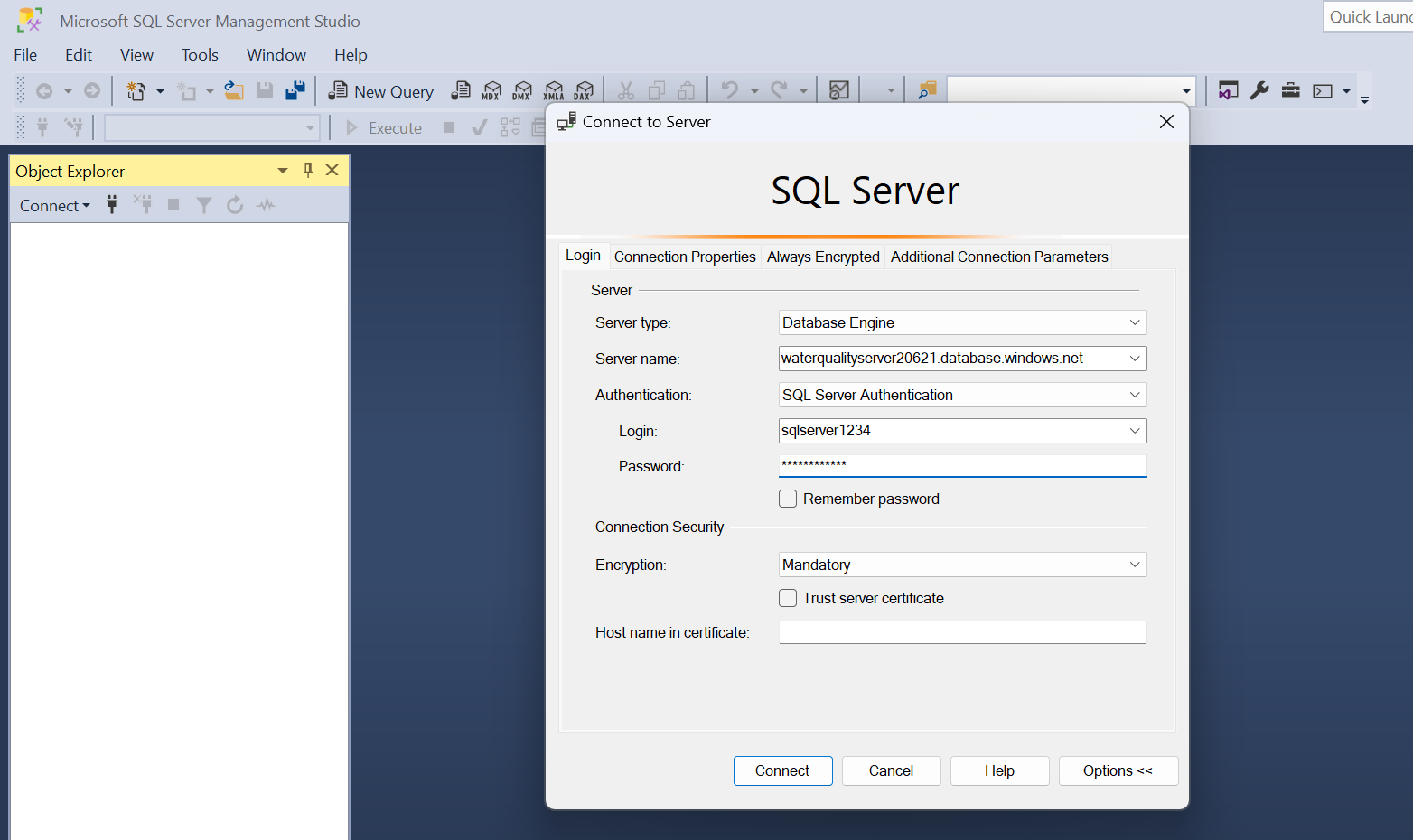
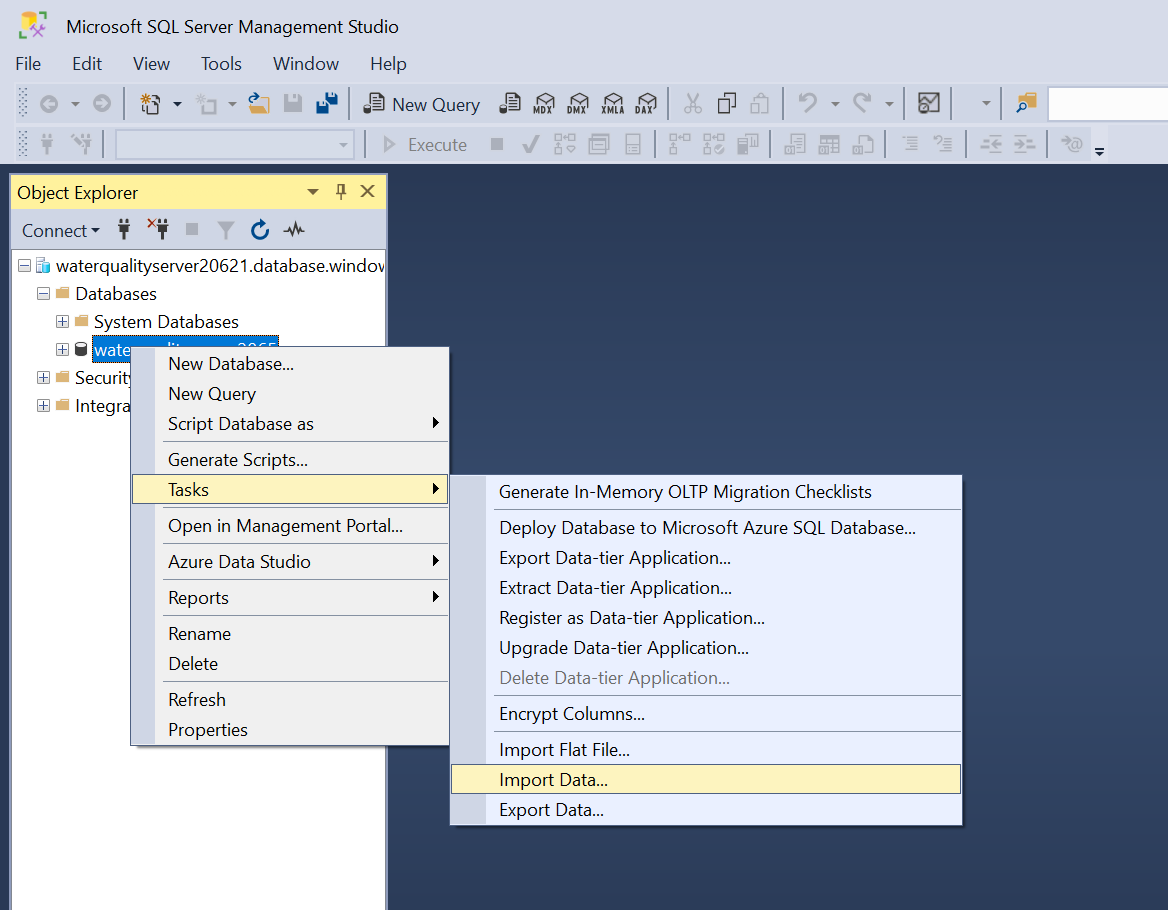
# Ingesting data from the on-premise SQL server

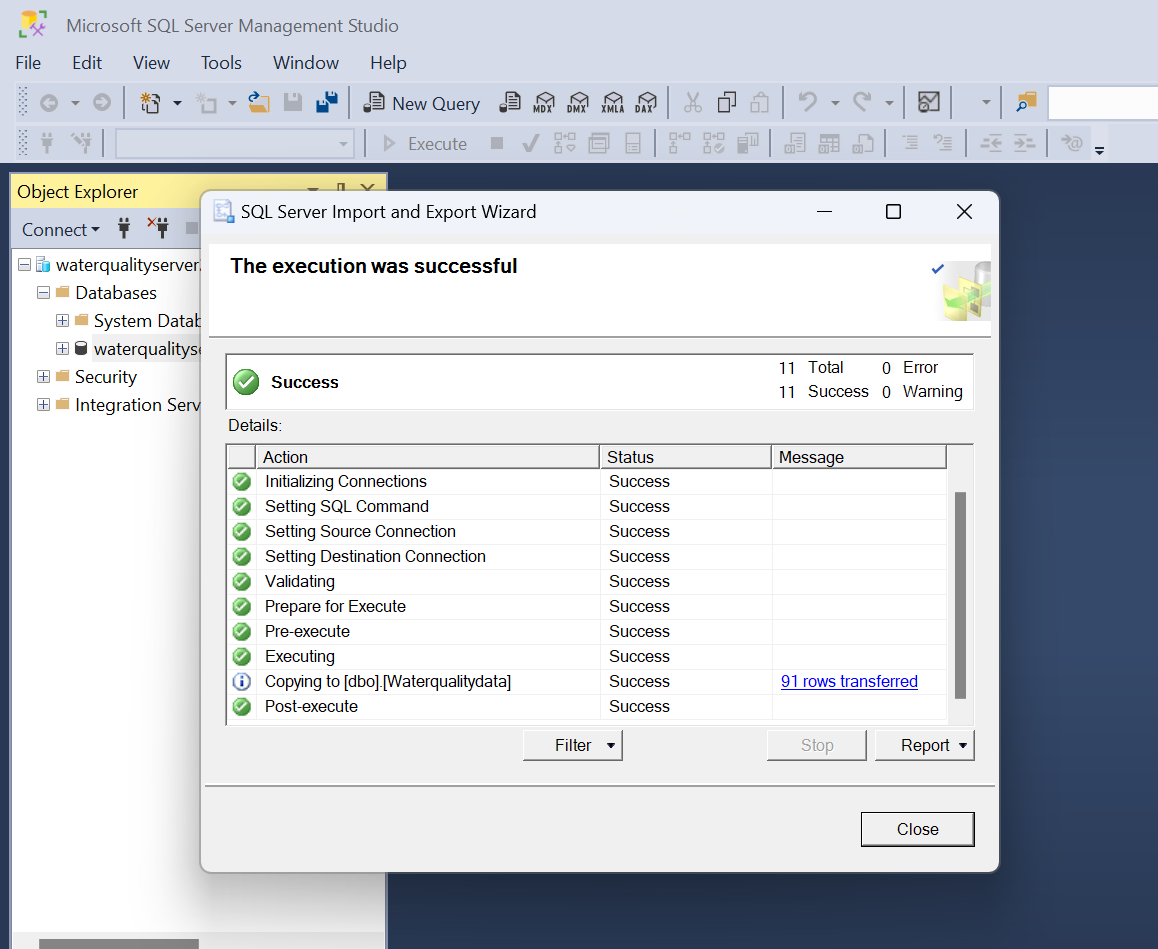
Note: When creating Azure SQL server and database, kindly consider this network connectivity issue. It is recommended to mark it as a public endpoint

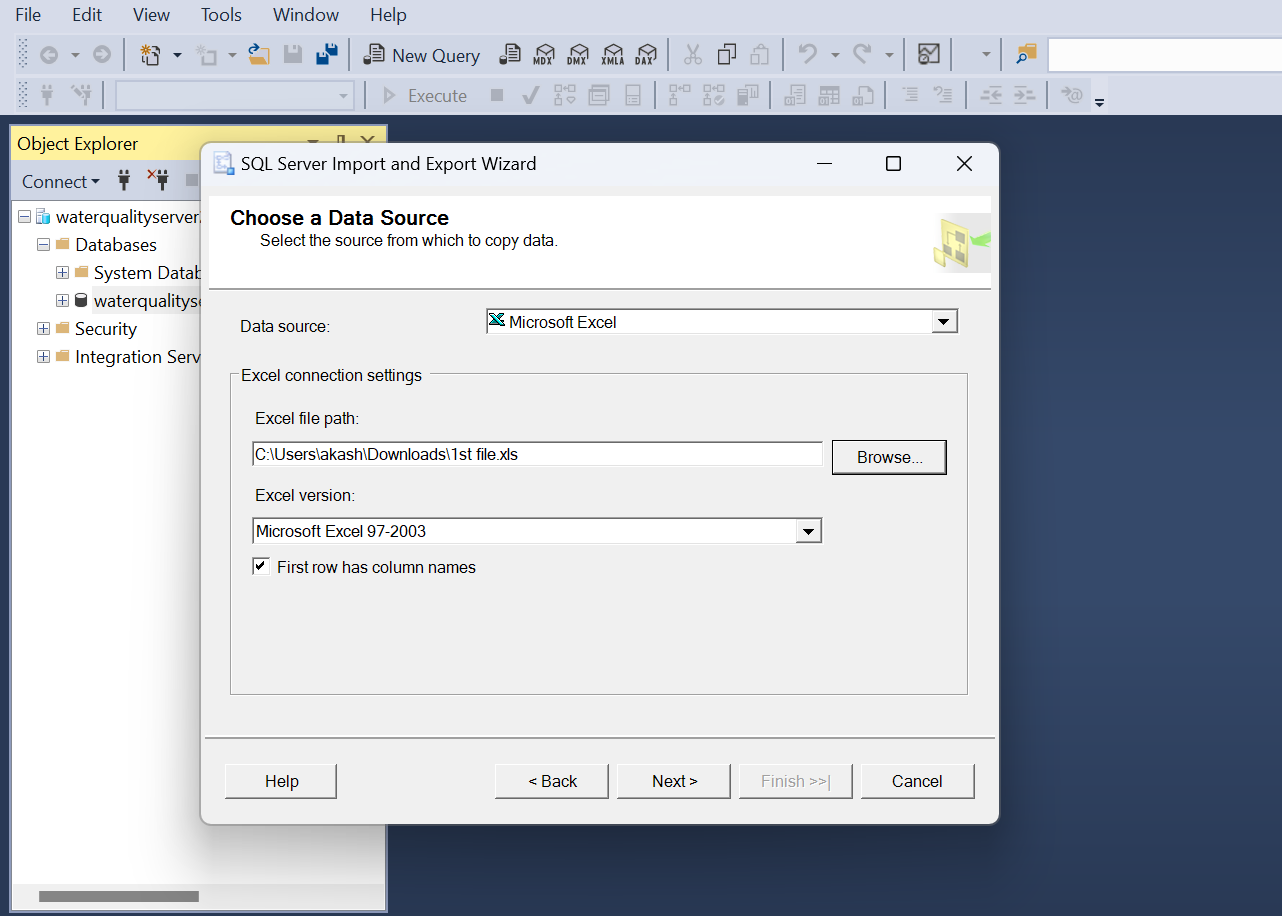


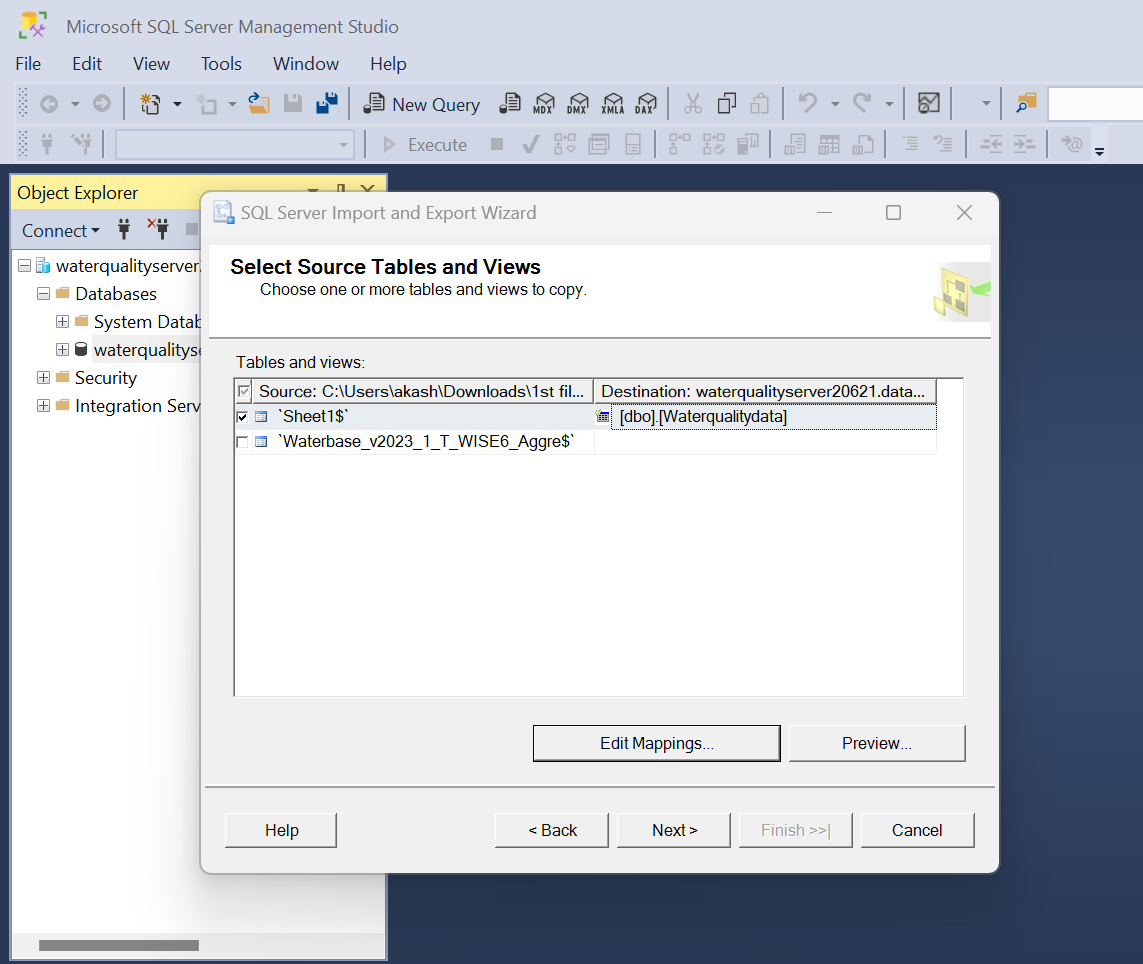
Step1: Authenticate the on-premise SQL server with Azure SQL database

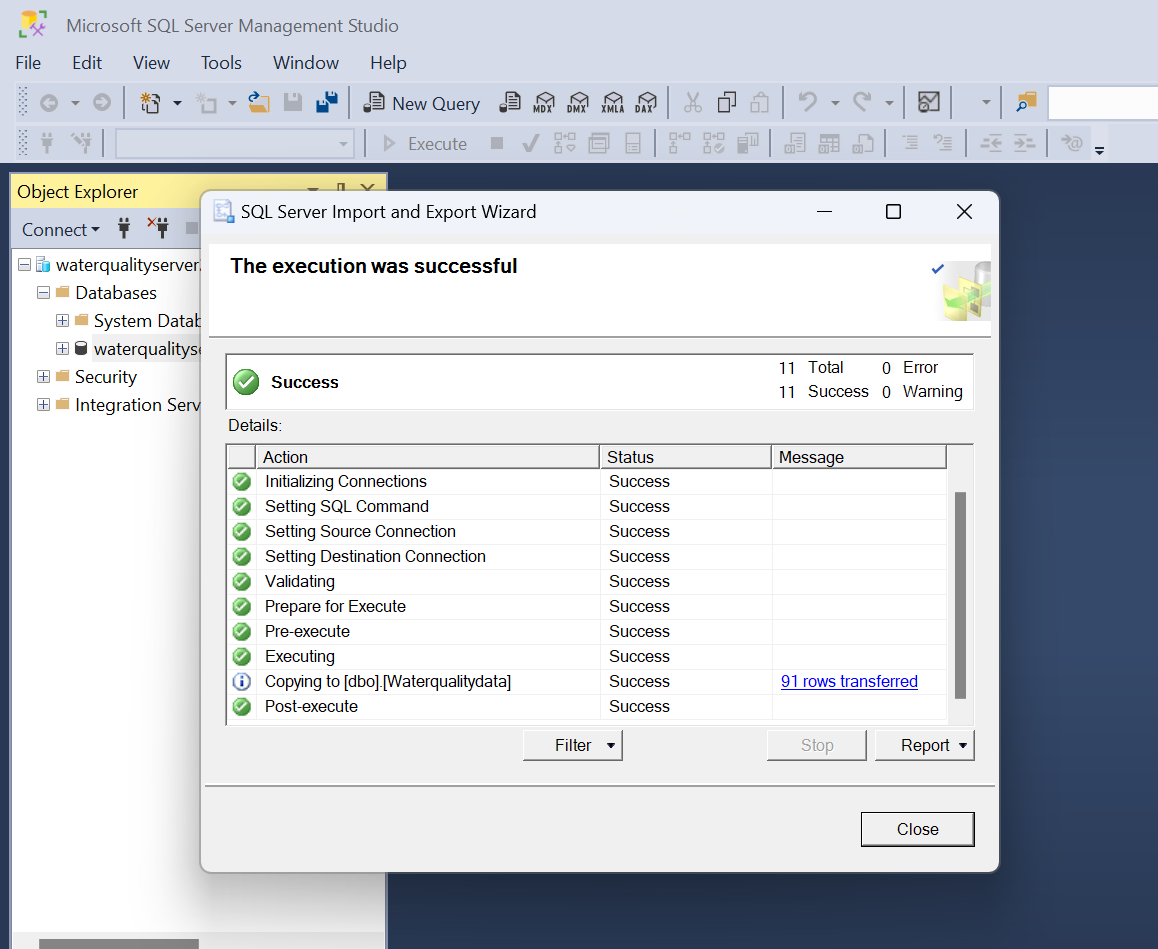
****

****

****

****

* Importing 1st File to Azure SQL Database

****

In the first run, after loading the data into the Azure SQL database, run this query in the query editor to create a Unique key primary column which is (Id)

**Run this query in the query editor of the Azure SQL database:**

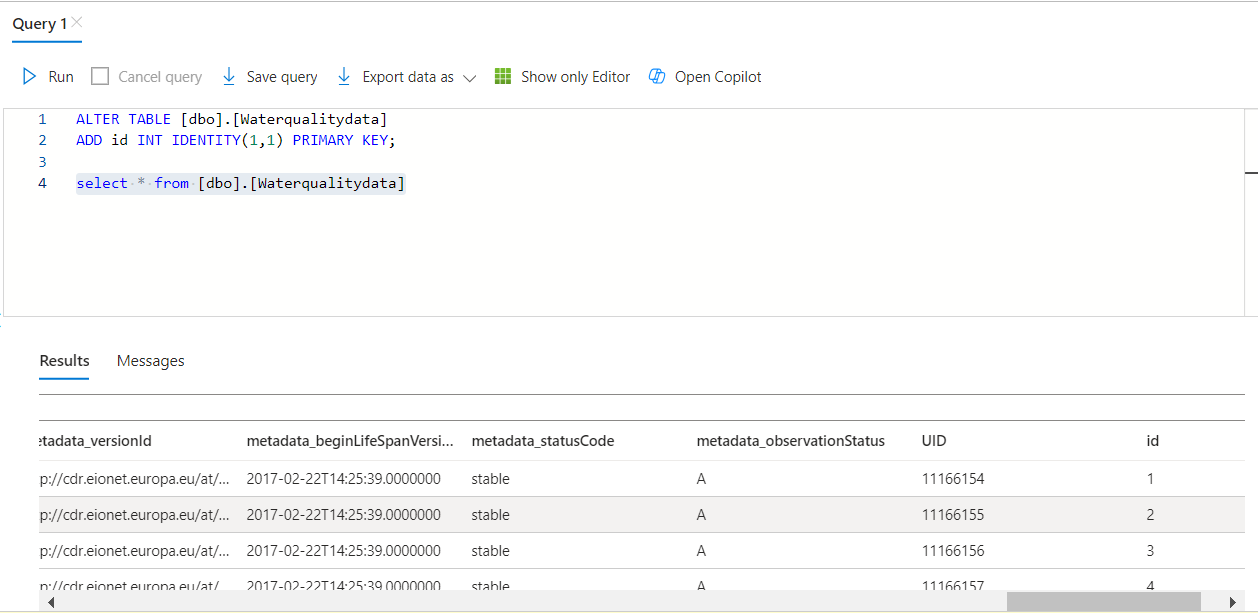
ALTER TABLE [dbo].[waterqualitydata]

ADD Id INT IDENTITY(1,1) PRIMARY KEY;

**Check whether the ID column is attached to the table and has values starting from 1.**

**Run this query:**

Select \* from [dbo].[Waterqualitydata]

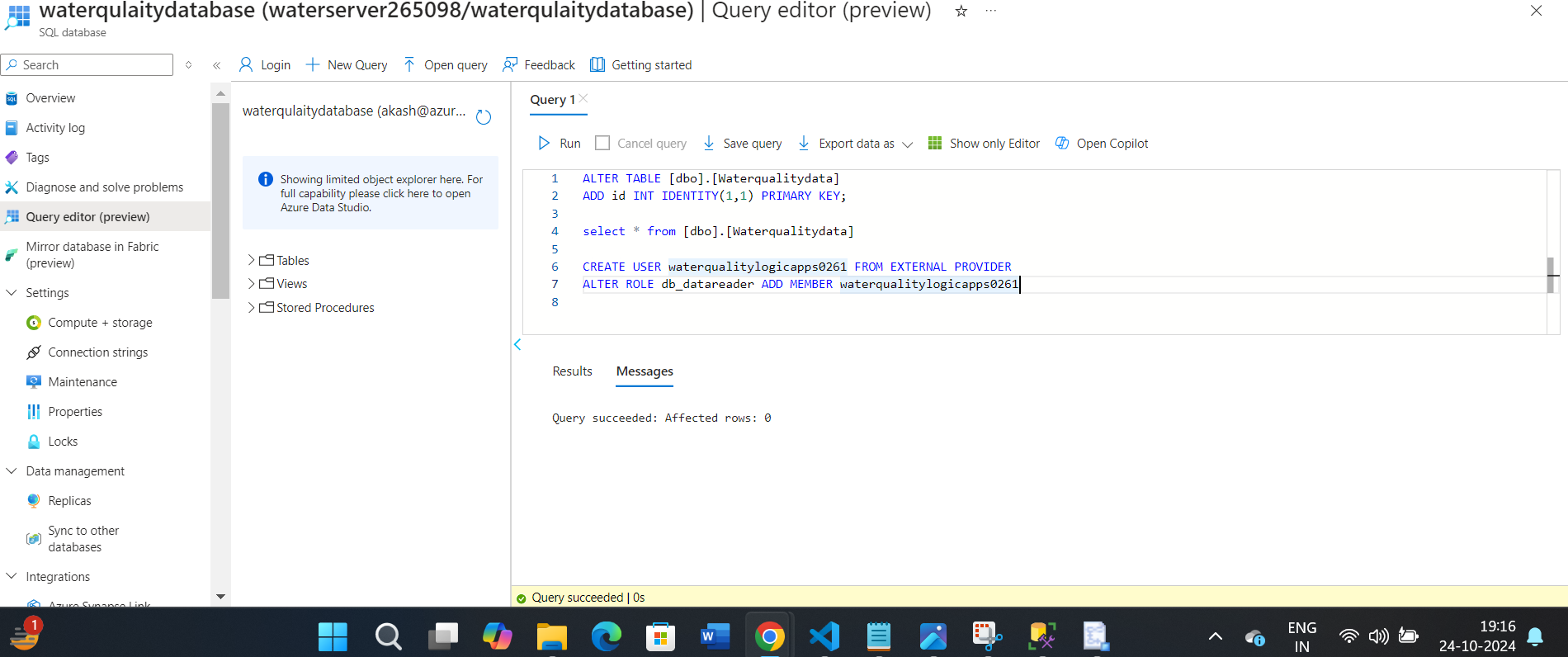


# Data Ingestion workflow using Azure Logic App

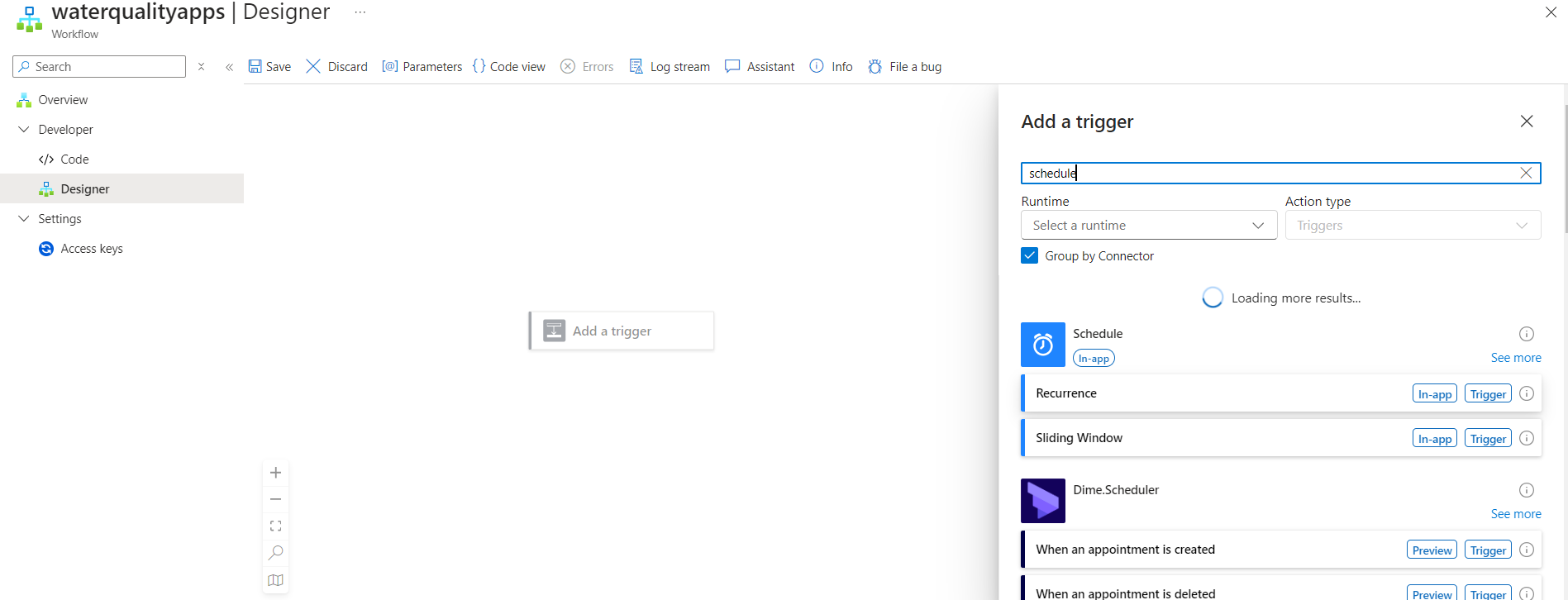
1. Create Logic App
2. Create Workflow and choose stateful in the workflow
3. Take the name of the Logic app and run this query in the query editor of the SQL Database to allow Logic app to have access to the Database.

CREATE USER <LOGIC-APP-NAME> FROM EXTERNAL PROVIDER

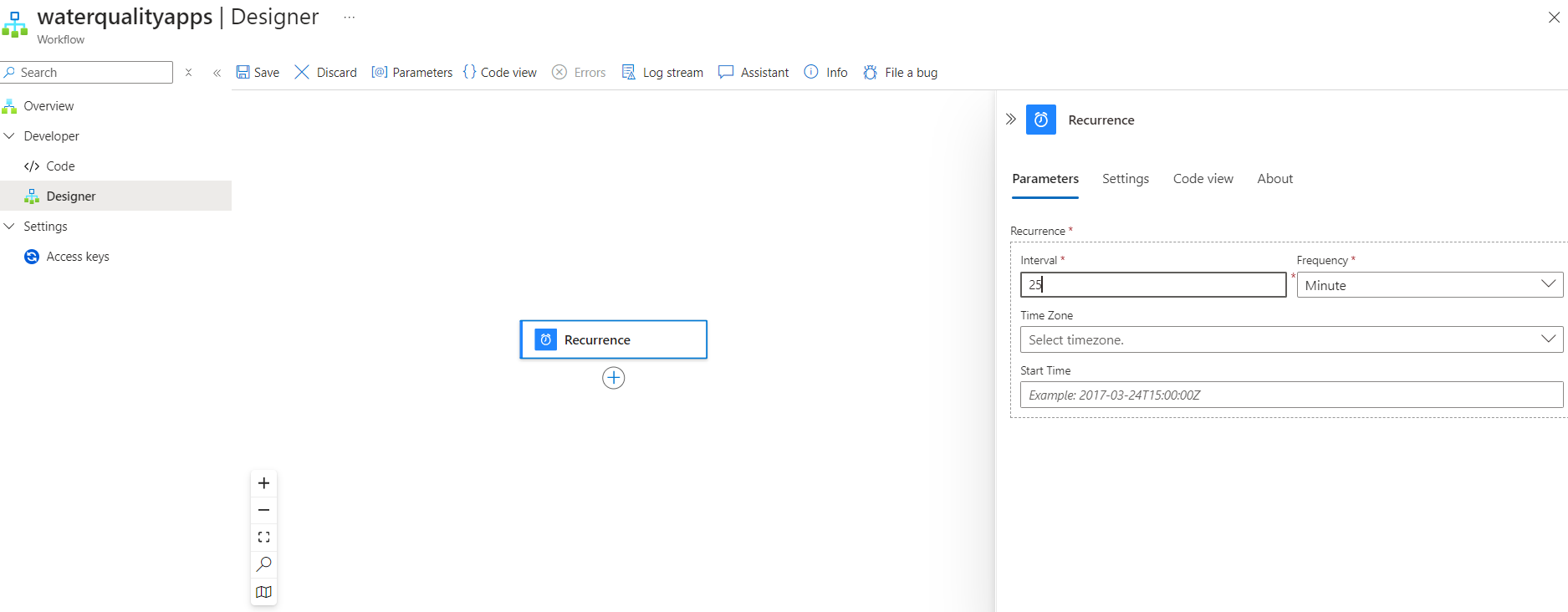
ALTER ROLE db\_datareader ADD MEMBER <LOGIC-APP-NAME>



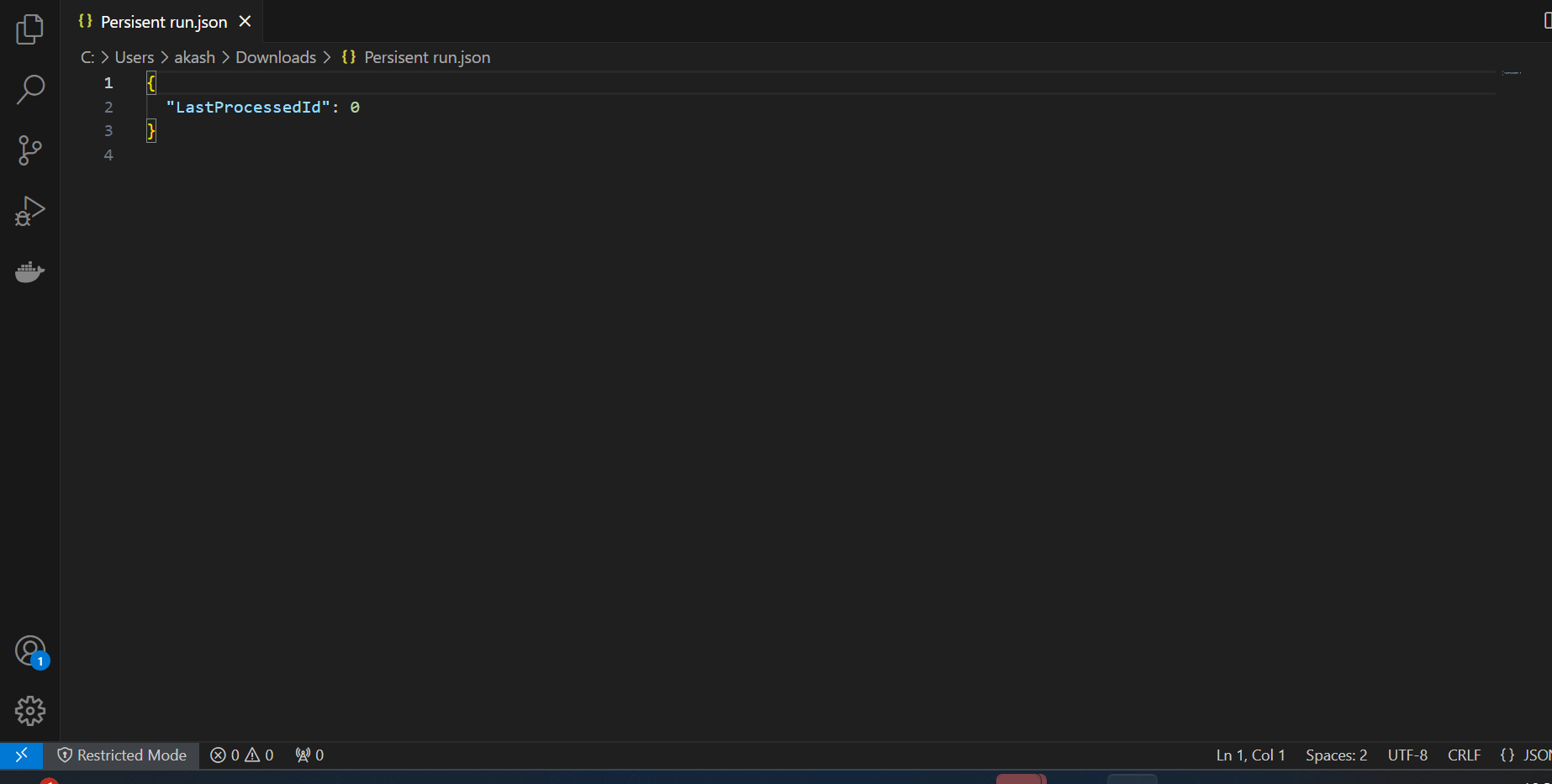
1. Start creating the design for the workflow using the designer in the workflow of the designer app
2. Add recurrence trigger using schedule action. Search for the schedule and select recurrence action



1. Select the frequency as “Day” and Interval as 1



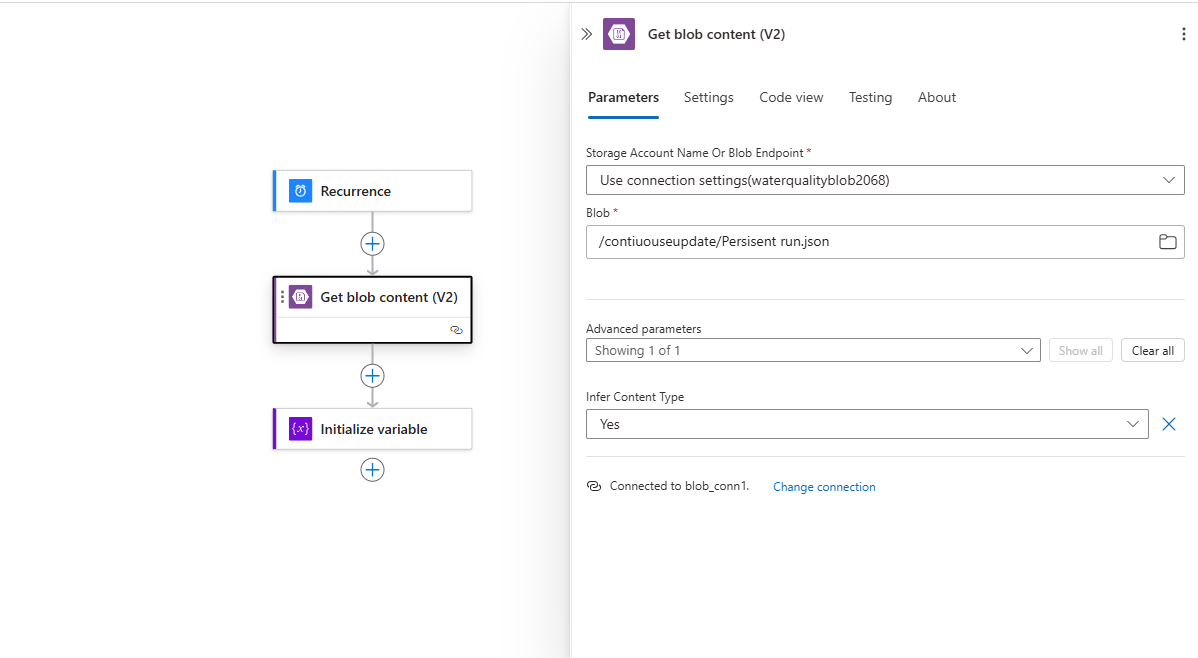
1. Then create a json file in VS code with content like this and name it as (persistent.json)



Then create a container in the blob storage account and upload this JSON file from local to the container using Azure UI.

1. Then in the logic app, add this action (Get blob content V2) mention the blob container detail, and specify this file(persistent.json) in the configurations

In the blob section here, please specify the correct location of the persistent.json which is in the blob container using the file icon



(continuous update is the name of a blob container where we stored our JSON file)

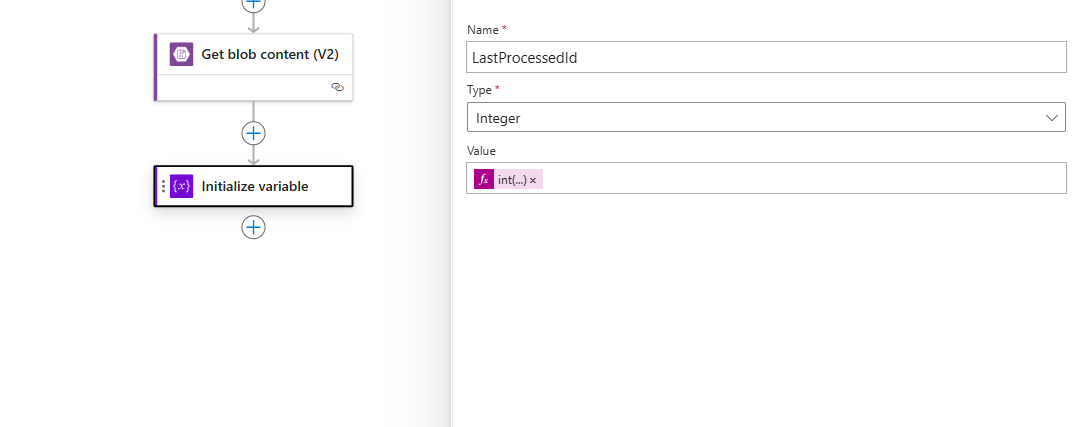
1. Add action initialized Variable and then mention these details:

Name: LastProcessedId

Type: Integer

Value: @int(json(base64ToString(body('Get\_blob\_content\_(V2)')?

['$content']) ) ['LastProcessedId' ])



1. Then add action Execute query and then firstly mention details to connect with the Azure SQL server database and then mention these details.

**Query:**

DECLARE @LastProcessedId INT;

SET @LastProcessedId = @{variables('LastProcessedId')};

SELECT \* FROM [dbo].[Waterqualitydata] WHERE Id > @LastProcessedId;

(This is used to set a variable (LastProcessedId) in the Execute Query action.

**Add SQL parameter from advanced parameters:**

**Content in SQL parameters:**

[

{

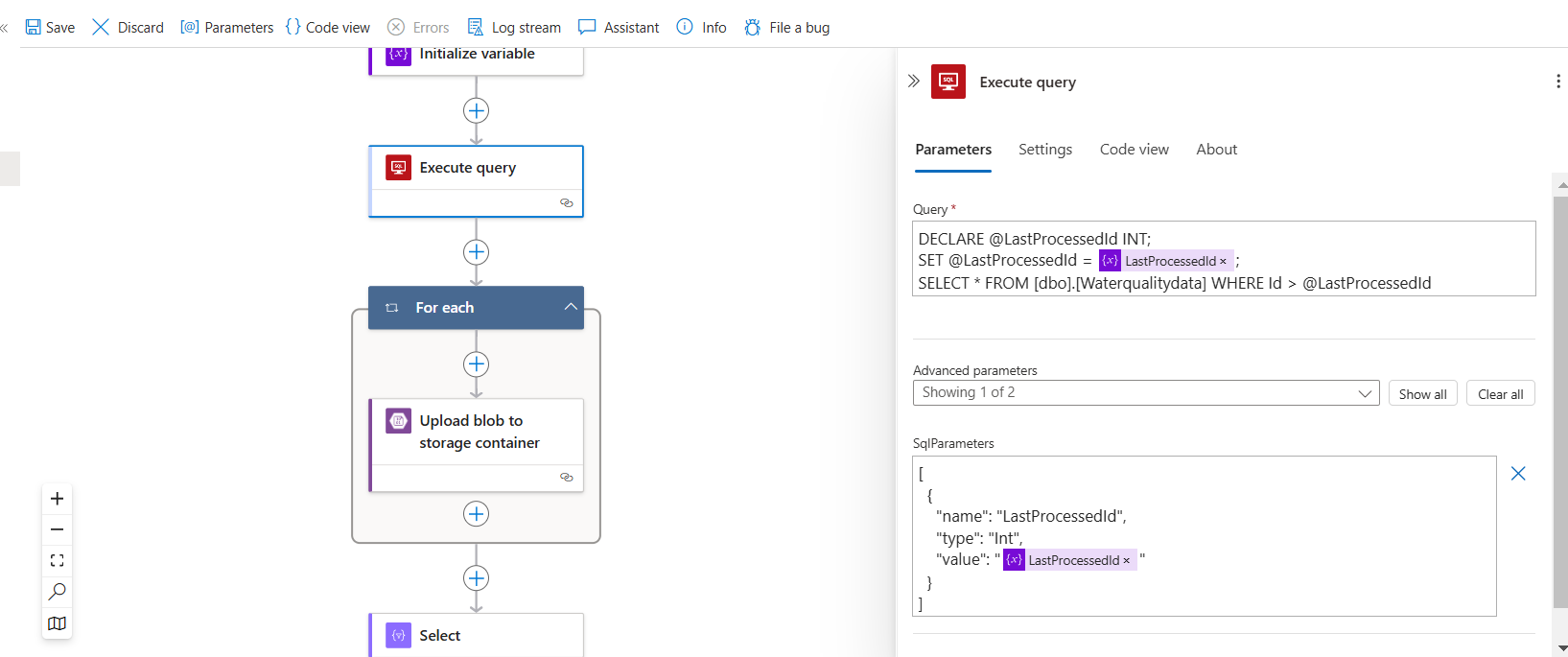
"name": "LastProcessedId",

"type": "Int",

"value": "@{variables('LastProcessedId')}"

}

]



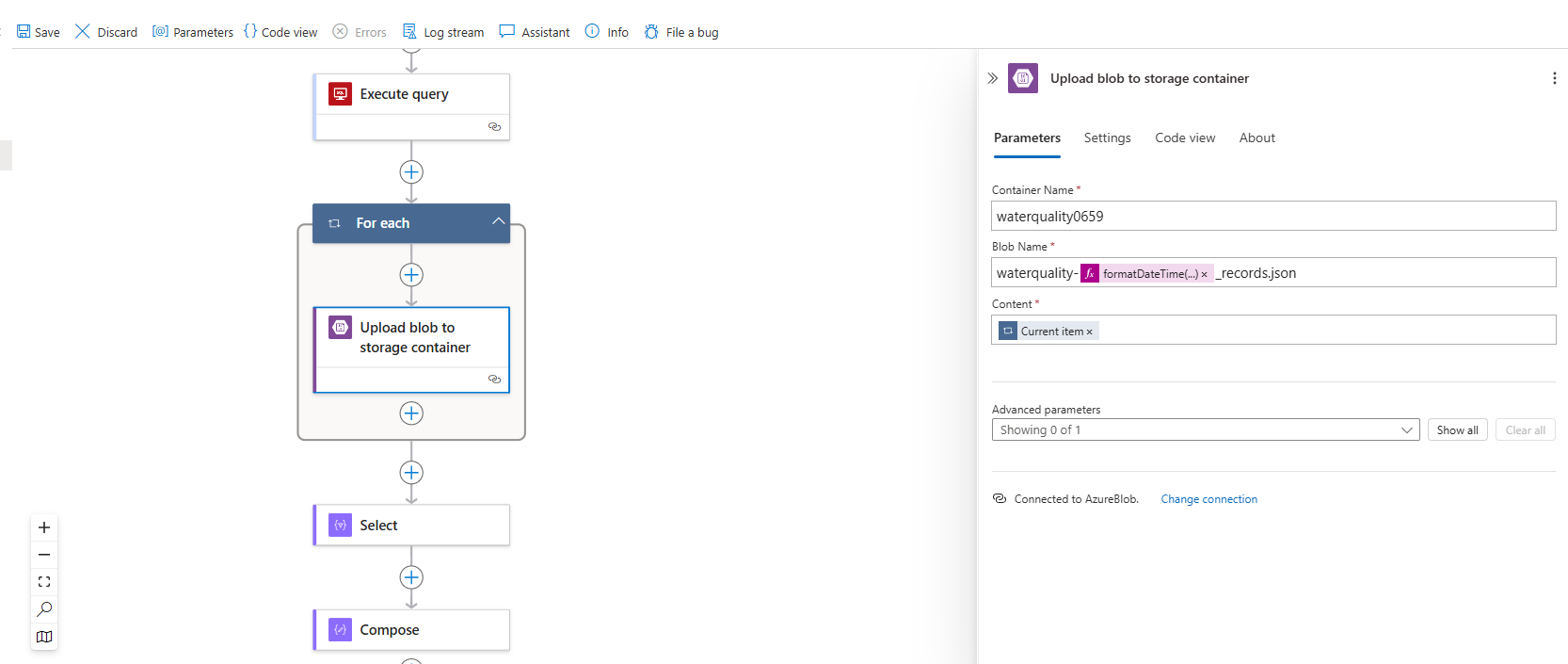
Create a blob container in the blob storage account ( which is also our main container which is used to store real-time records through which we will put the files into the ADLS container)

1. Then add the action “Upload blob to a storage container” and configure it with the connection string of the blob storage account

**Container name:**<name of container>

**Blob name:** waterquality-@{formatDateTime(utcNow(),'yyyyMMddTHHmmss')}\_records.json

**Content:** Select Dynamic results from the Execute query



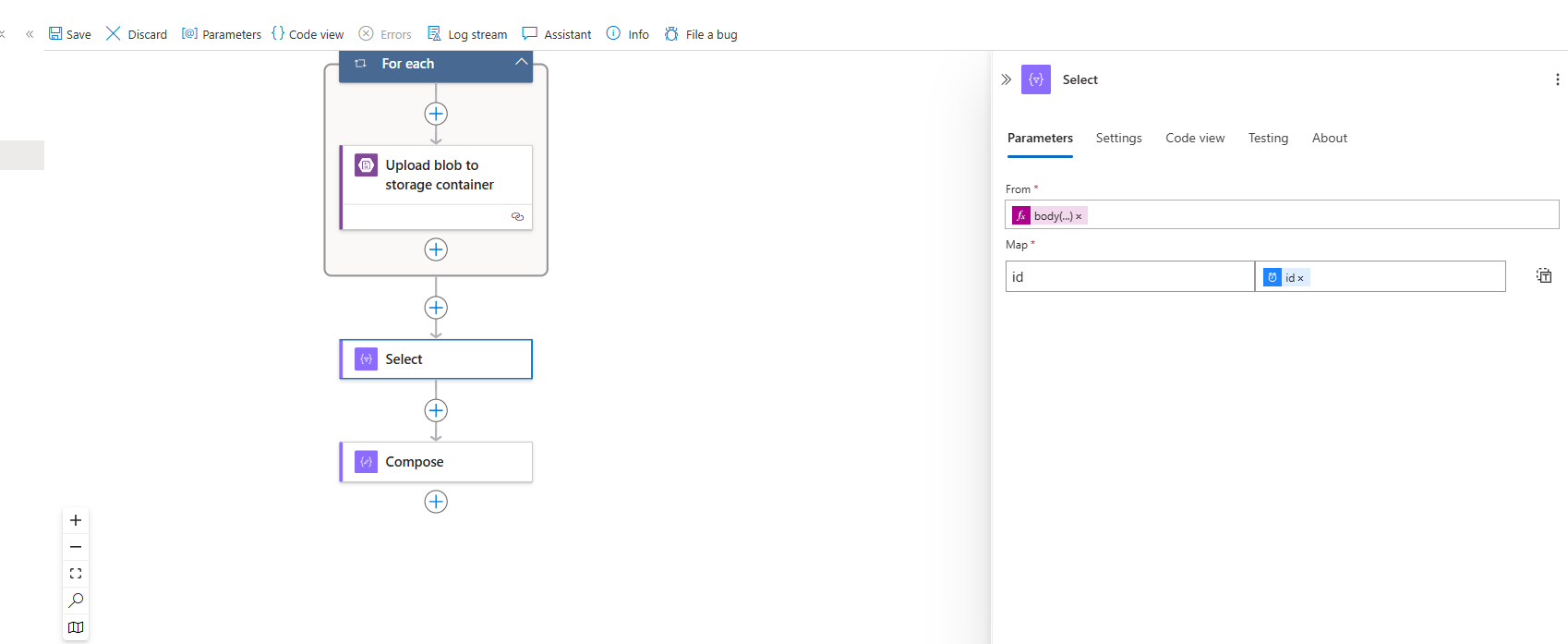
1. Then add the action “Select” and mention the details:

**From**: @body('Execute\_query')[0]

**In Map**

Key = id

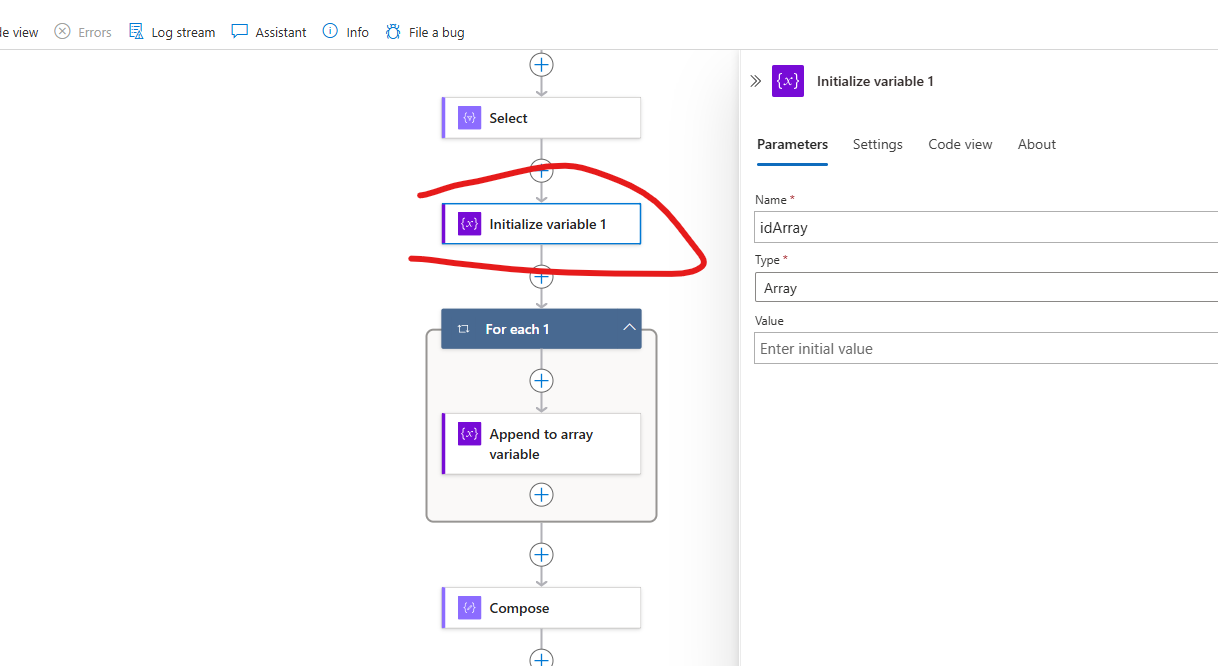
value = @item()?['id']



1. Then again add the action “Initialize variable” and mention these details

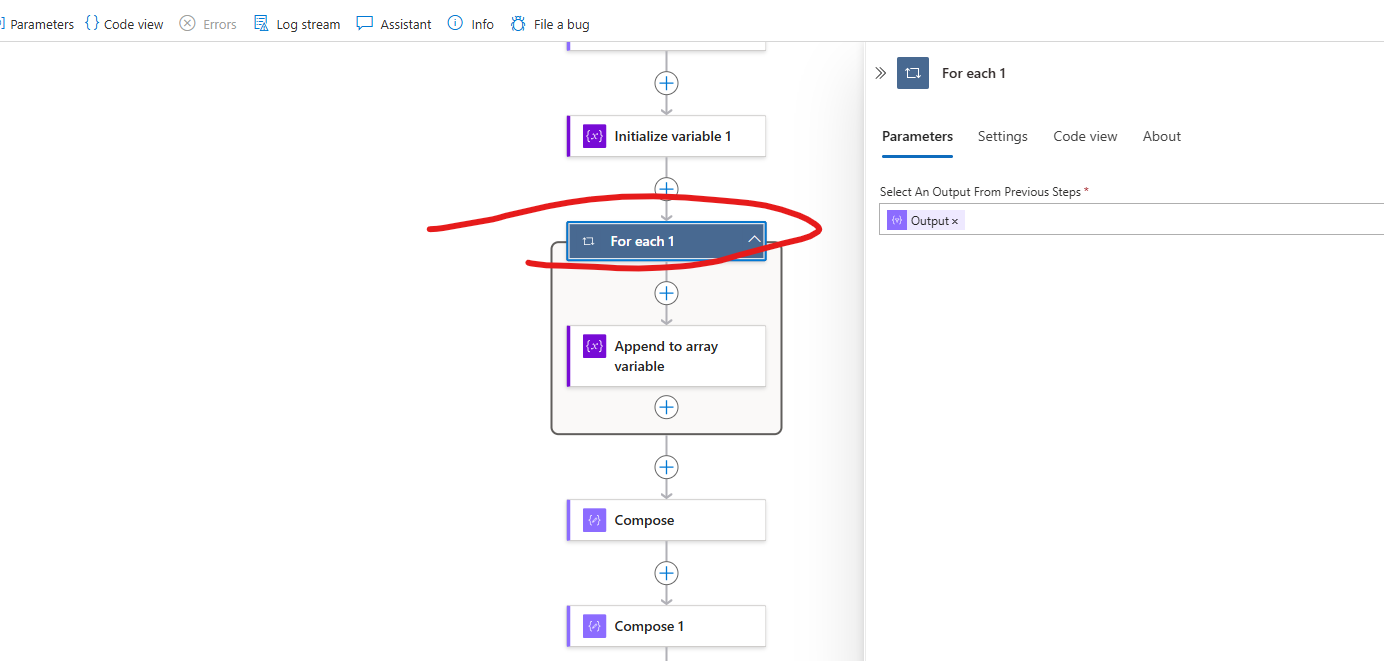
Name: idArray

Type: Array



1. Then add the action “For each” and mention the details:

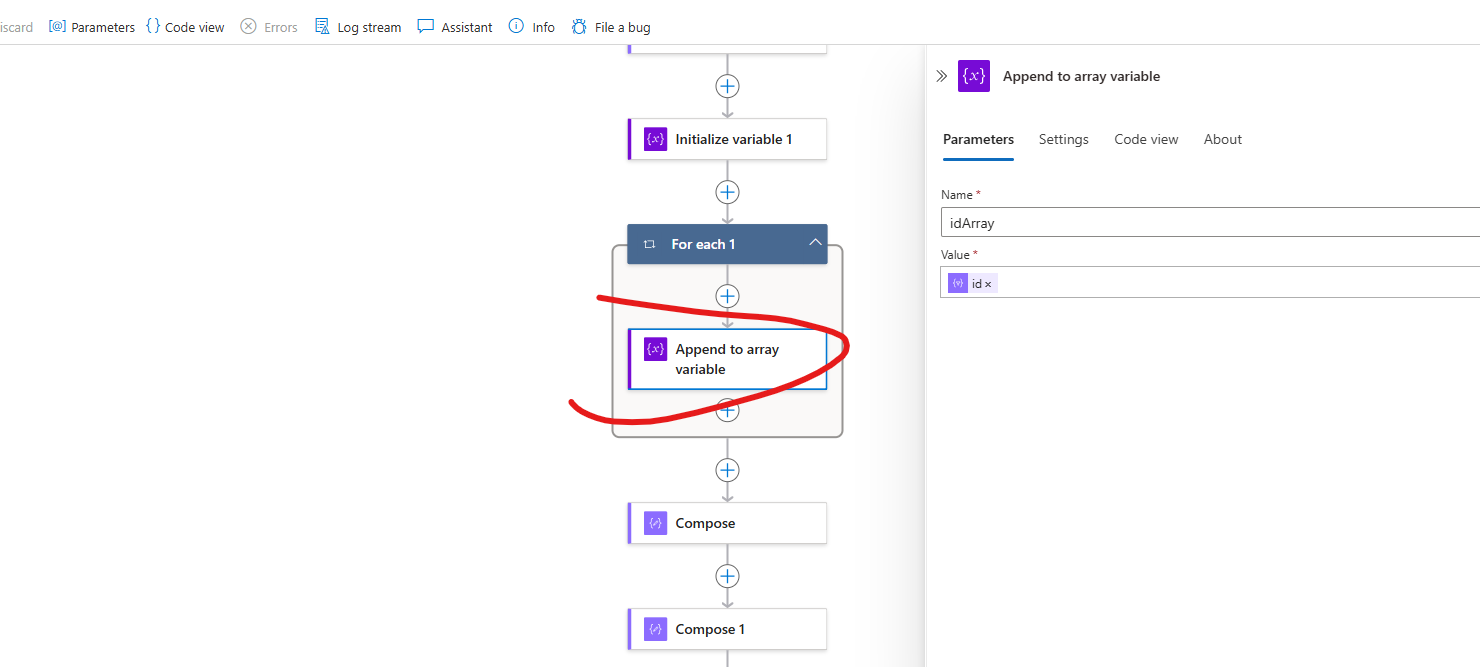
**Select an output from the previous step:** @body(‘Select’)



* Inside the Foreach loop, click on + icon to "Add an Action".
* Search for "Append to array variable" and select it.
* Configure the "Append to array variable" action and mention these details

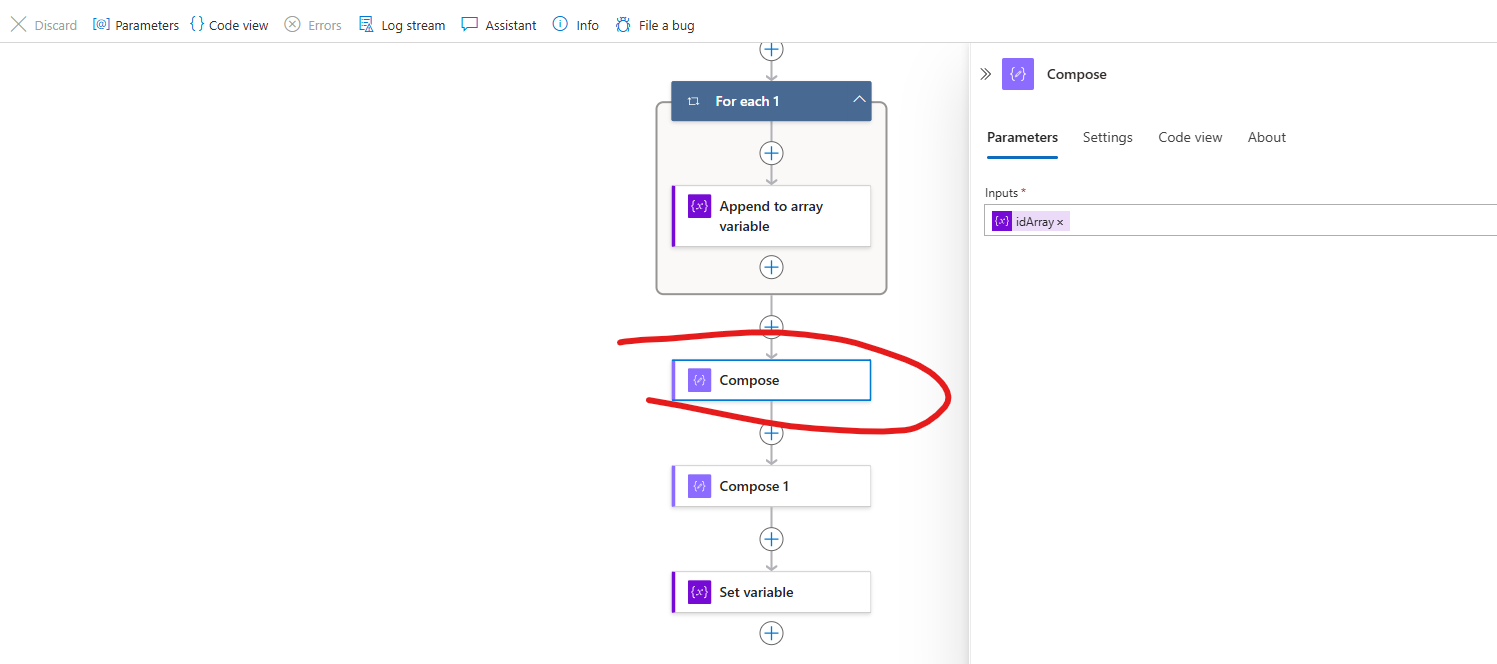
**Name:** idArray

**Value:** @item()?['id']



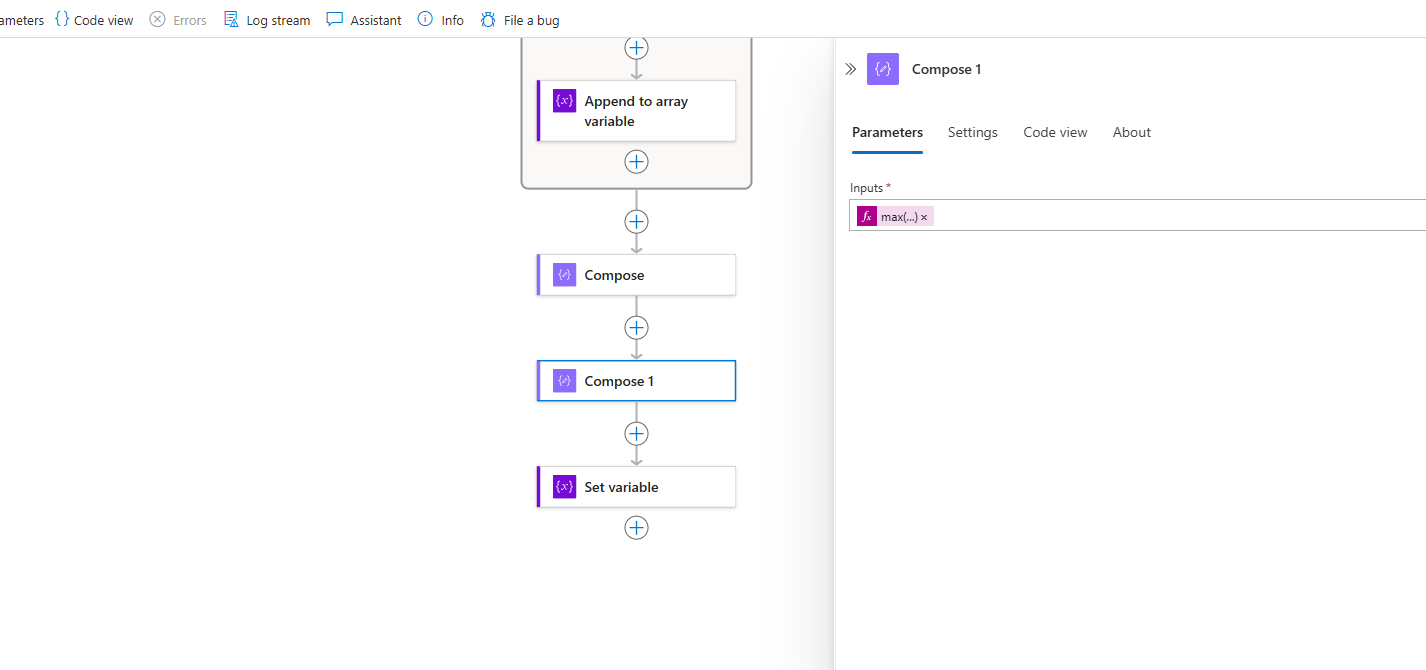
1. Then add another action “Compose’ and add these details:

**Inputs:** @variables('idArray')



1. Then also add another action “Compose” and then mention these details:

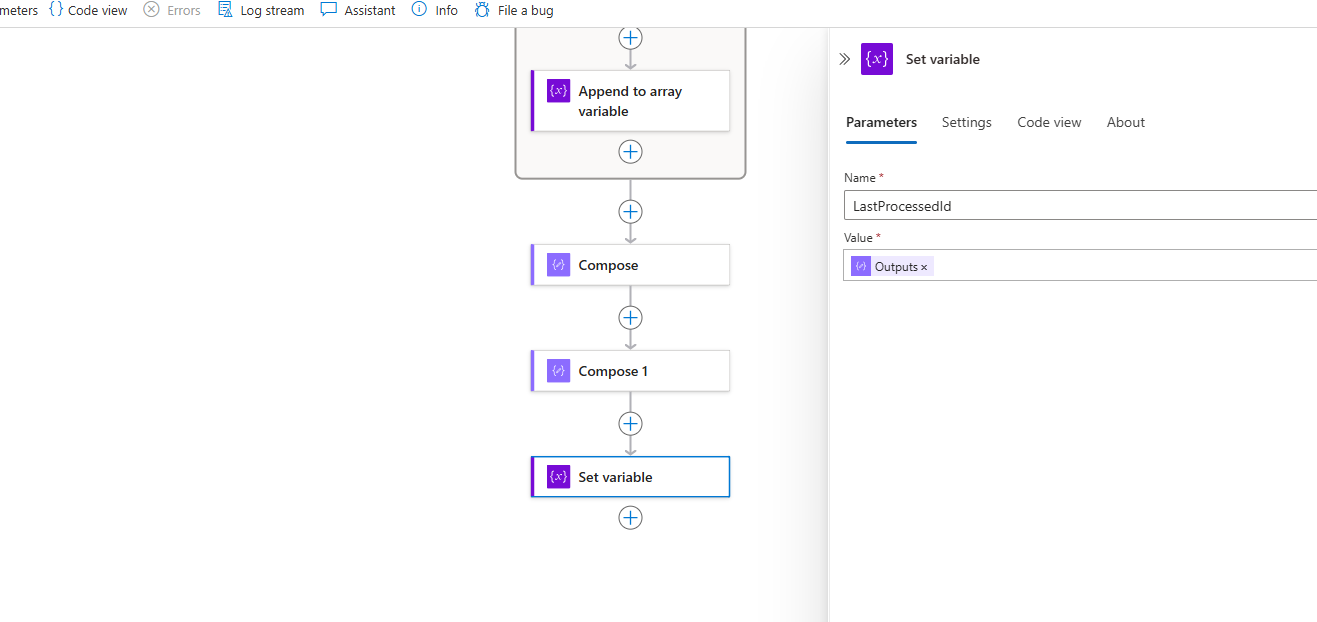
**Inputs:** @max(outputs('Compose'))



k) Then add another action “Set Variable” and then mention these details:

**Name:** LastProcessedId

**Value:** @outputs('Compose\_1')



L) Then add another action “Update Blob” and then mention these details:

**Storage account name: <** Name of the storage account>

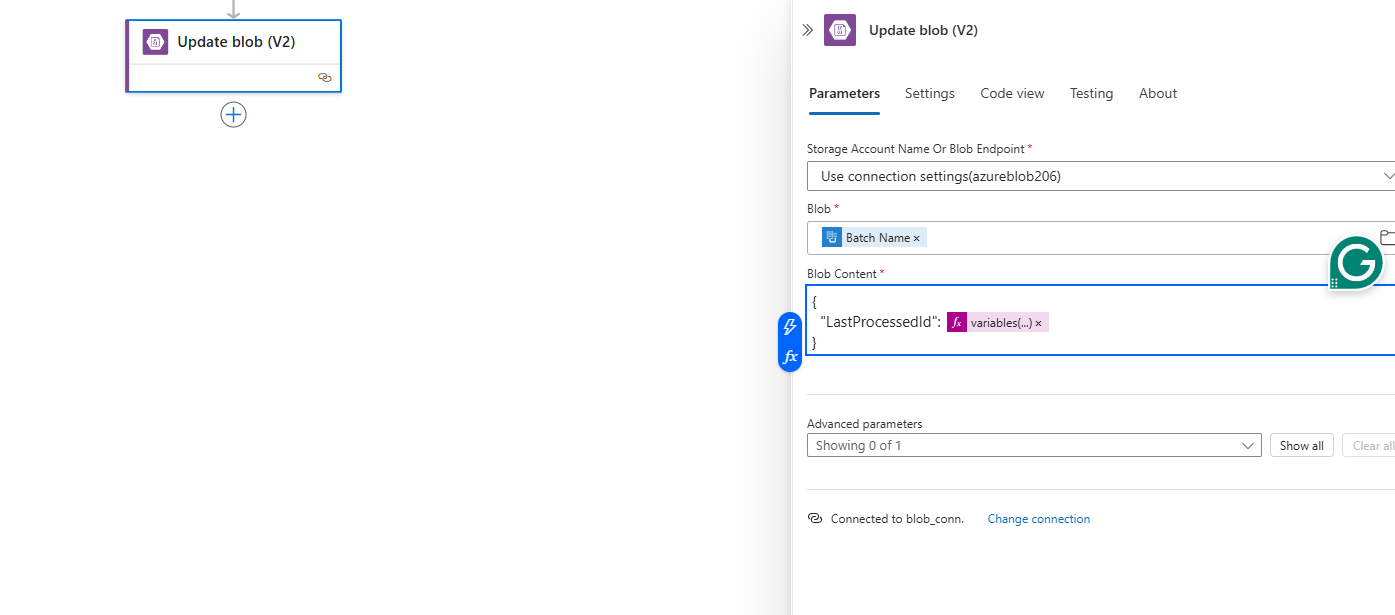
**Blob:** Browse the JSON file in the blob container that we created in the container

**Content:**

{

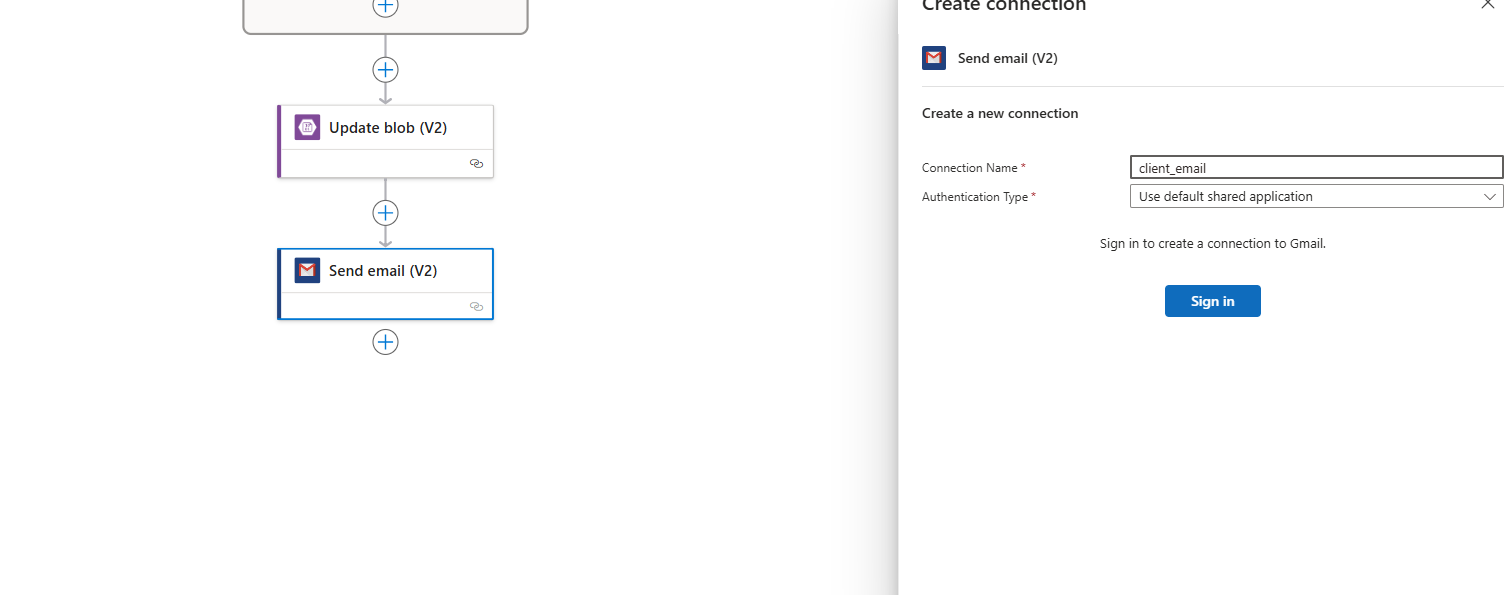
"LastProcessedId": @{variables('LastProcessedId')}

}



M) After this action add any “email” action like this and choose use default as an authentication option

And then specify whom to send emails to with the subject “New Records are fetched from the Azure SQL database”

****