

BOOTCAMP PROJECT 4

Project Title: Incremental Data Loading and Automated Notifications using Microsoft Fabric

❖ Problem Statement:

In modern data ecosystems, organizations need to efficiently ingest, transform, and load data from various sources into centralized platforms for analytics, while also ensuring timely monitoring and notification upon successful data refreshes. This project addresses the challenge of incrementally loading data from on-premises sources to Microsoft Fabric Lakehouse, processing it through a structured transformation pipeline, and triggering automated notifications upon successful execution.

❖ Project Objective:

To build an end-to-end data pipeline on Microsoft Fabric that:

1. Ingests structured data from on-premises environments into a Fabric Lakehouse using the On-Prem Gateway
2. Utilizes the AI Bank Dataset as the source
3. Implements Dataflow Gen 1 to join tables, remove duplicates, and clean data
4. Loads the cleansed data into a Fabric Warehouse
5. Applies Slowly Changing Dimension (SCD) Type 1 logic using Fabric Notebooks and writes the results into separate warehouse tables
6. Schedules and monitors the pipeline, sending an automated email notification (via Outlook or Gmail) upon successful pipeline completion

Step 2:

* Source: Sales Excel file from ADLS Gen 2

* Do transformations on this file, remove duplicates, casting, and filter null values

* Bring data into Lake House using a fabric notebook (Tables)

- * While loading, decide which column will go into which table (3-5 dimensions tables)

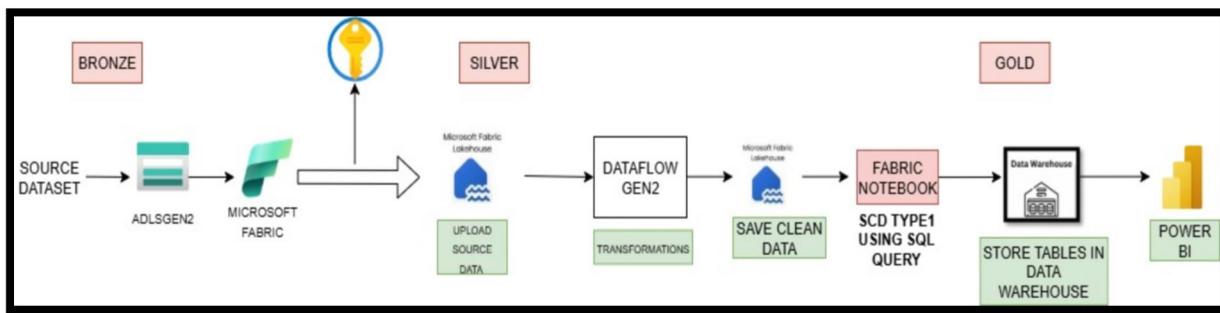
Step 3:

- * Source: Excel file (Sales Return) in ADLS Gen 2
- * Do transformations on this file, remove duplicates, casting, and filter null values
- * Bring the Excel file into Lake House using a fabric notebook (1-dimensional Table)

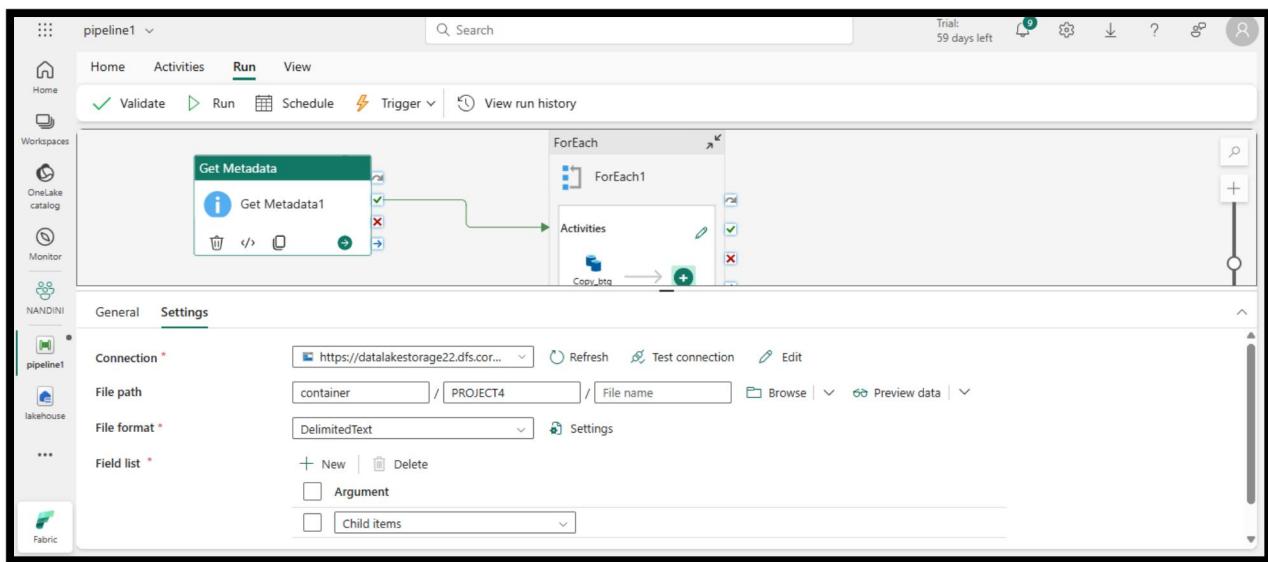
Step 4:

- * Join the dimension tables with Excel files using the Fabric notebook and generate a fact table (keep numerical values in the fact table, also foreign keys from dimension tables)
- * Join the fact table with the dimension tables and display the additional descriptive information using the SQL query, and use that query in Power BI
- * Store facts and dimensions in the warehouse

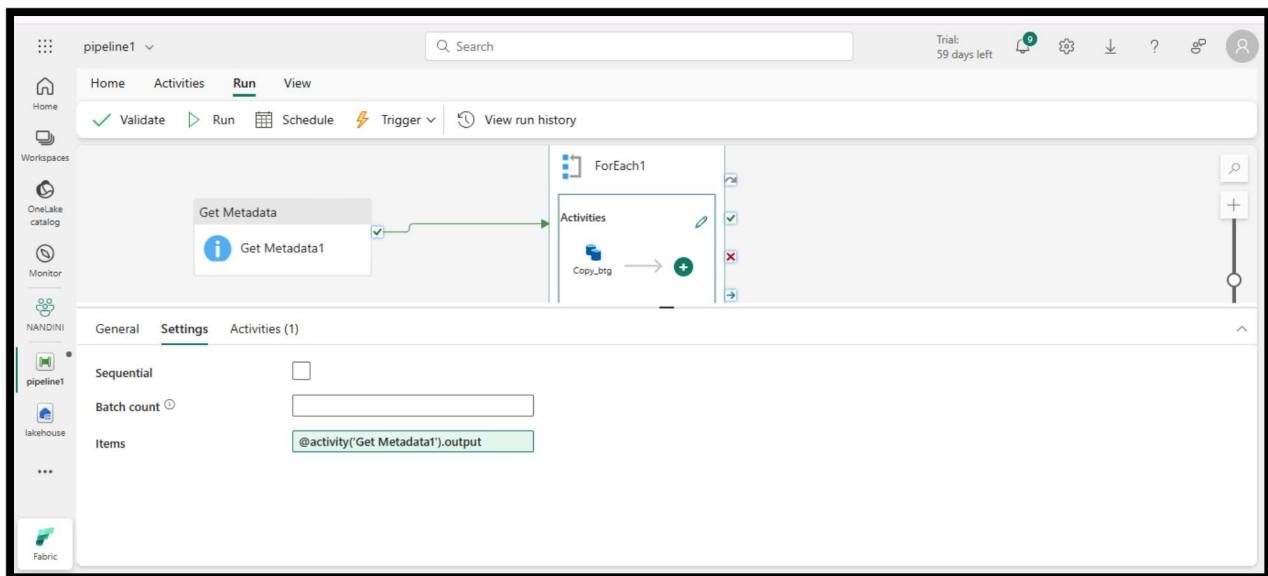
ARCHITECTURE DIAGRAM:-



FIRST STEP OF INGESTION OF DATA FROM ADLS TO LAKEHOUSE FOR TRANSFORMATION.

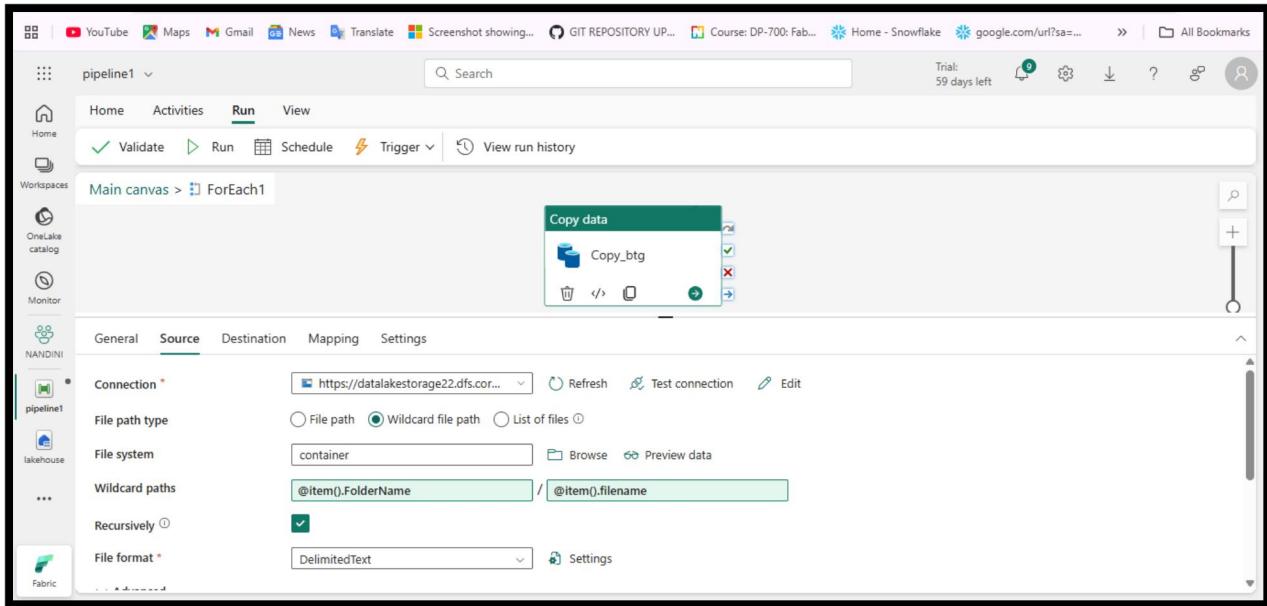


In fabric workspace, I selected the data pipeline and I want to load data dynamically. Therefore I select get metadata and for each activity.

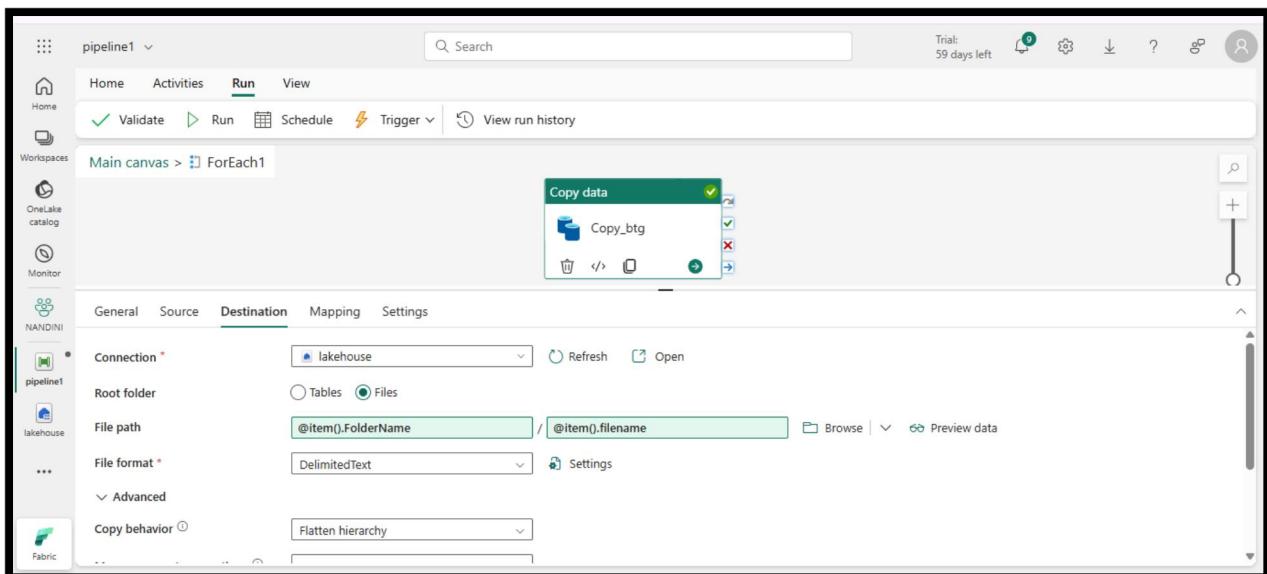


In items ,providing the output of get metadata activity.

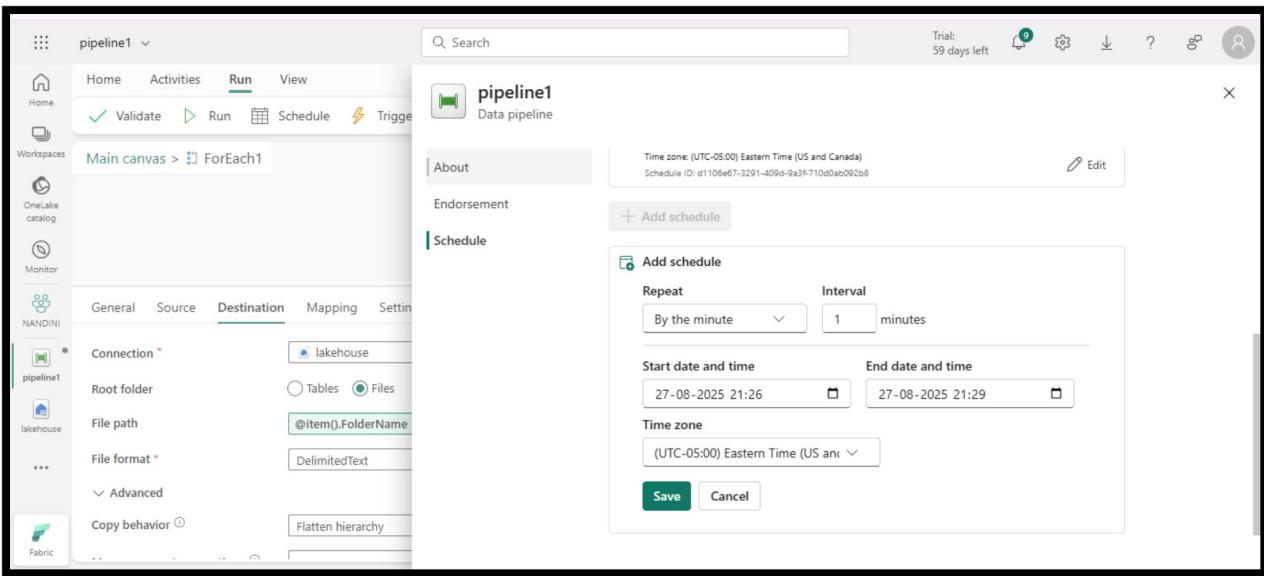
NANDINI RATHORE



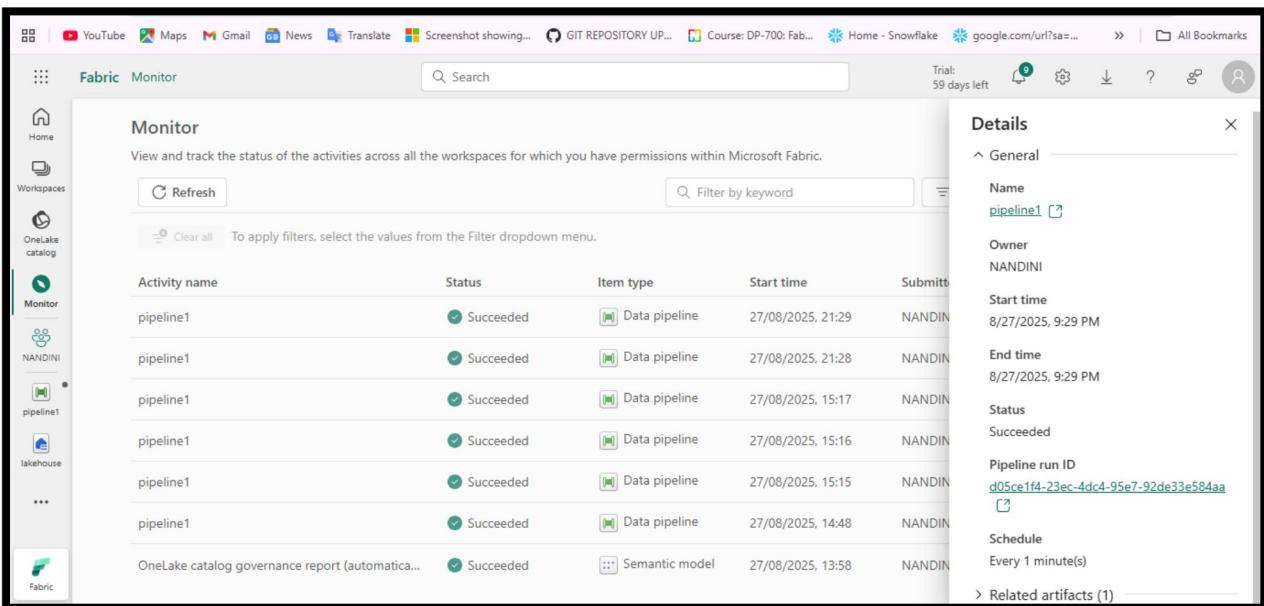
Now inside for each activity ,use copy activity to copy data from adlsgen2 to lakehouse.

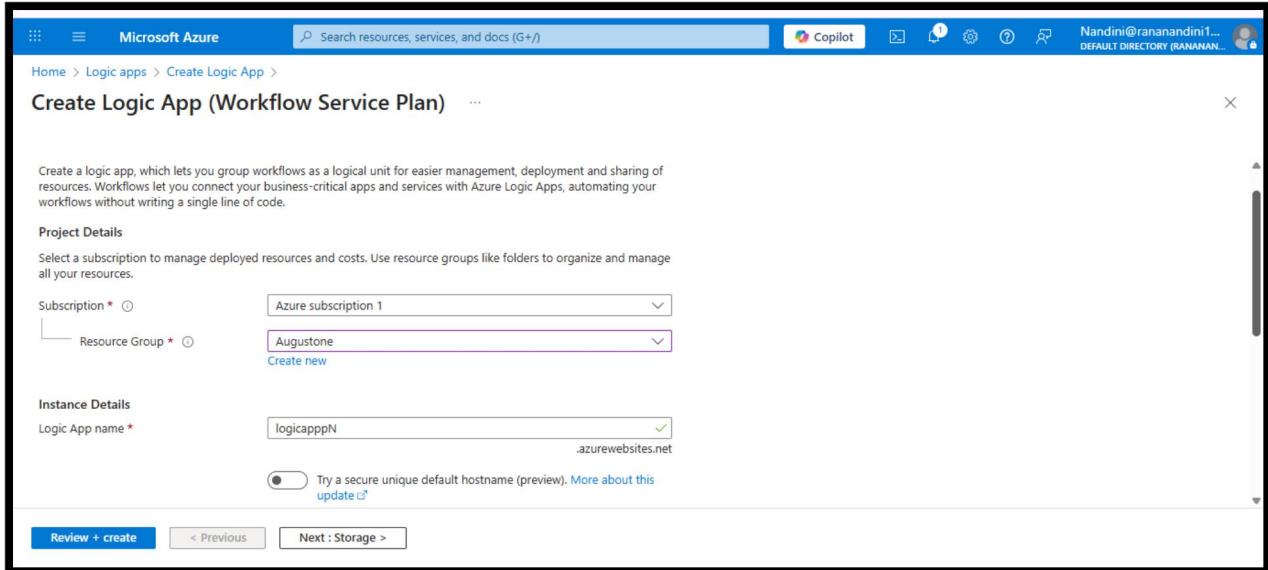


NANDINI RATHORE



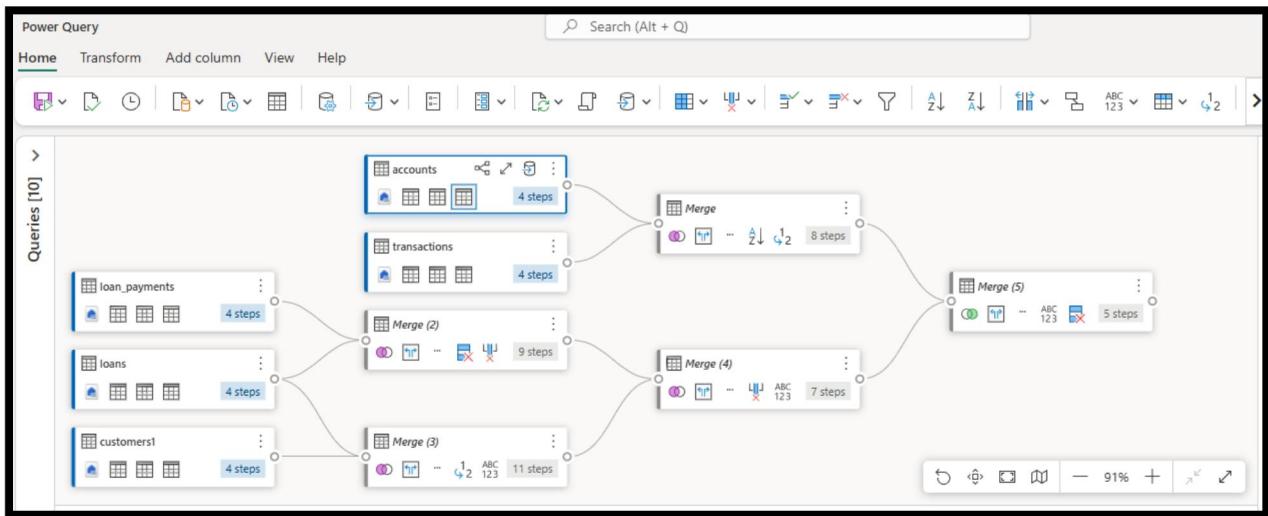
Schedule trigger.



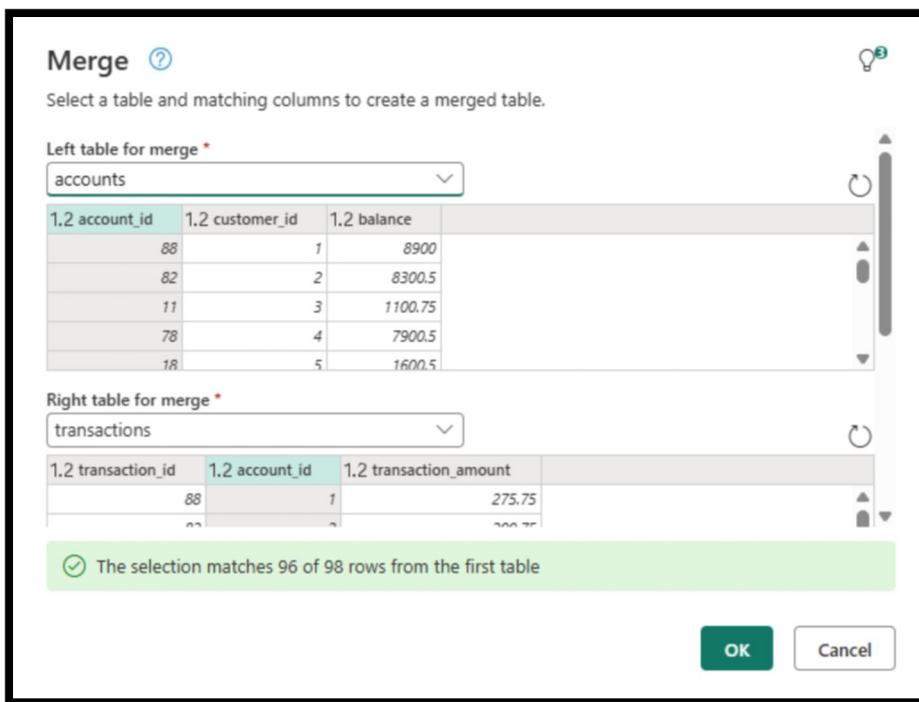


Download logic app for web activity and get mail notifications.

Dataflow transformations.



1 merge :-ACCOUNT AND TRANSACTION TABLE.



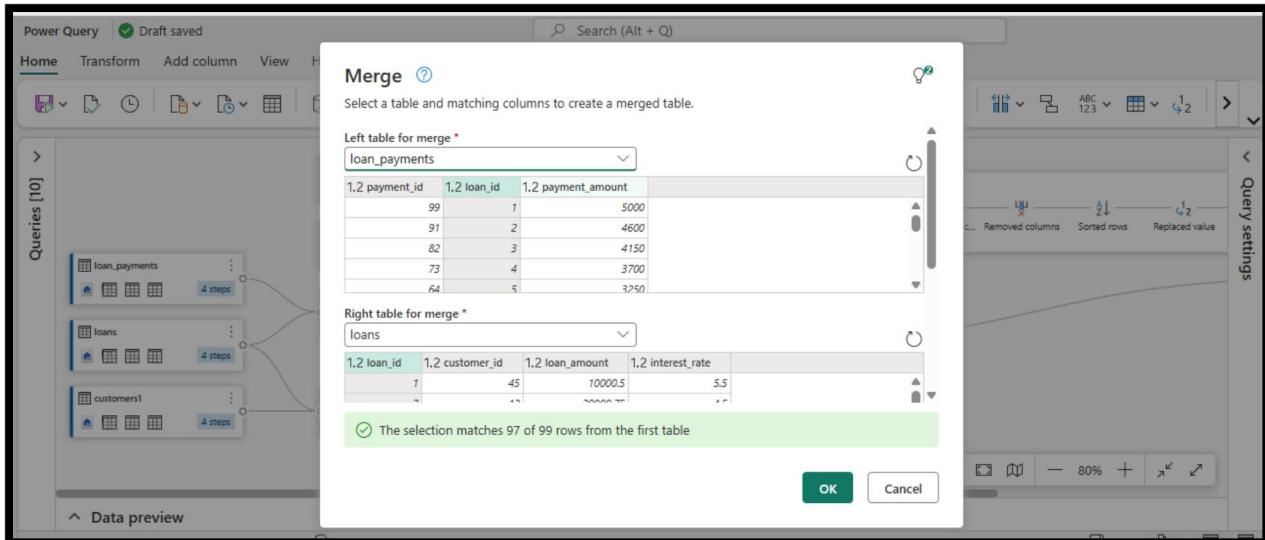
The screenshot shows the Power BI Data Flow editor with a 'Merge' step selected. The source is 'Source' (transactions) and the target is 'Replaced value' (accounts). The 'Applied steps' pane on the right lists the following changes:

- Source
- Expanded transactions (ABC 123)
- Changed column type (123)
- Reordered columns
- Removed duplicates
- Removed columns
- Sorted rows
- Replaced value

The 'Data destination' pane shows 'Lakehouse'.

2. MERGE –LOAN_PAYMENTS & LOANS.

NANDINI RATHORE



The screenshot shows the Power Query Editor with a merged table named 'Merge (2)' containing 99 rows. The table includes columns: payment_id, loan_id, payment_amount, customer_id, loan_amount, and interest_rate. The 'Query settings' pane on the right shows the name 'Merge (2)' and the 'Applied steps' list, which includes 'Source', 'Expanded loans', 'Changed column type', 'Reordered columns', 'Removed duplicates', 'Added custom', 'Removed columns', and 'Removed duplicates 1'. The 'Data destination' section is set to 'Lakehouse'.

ADD A CUSOM COLUMN:-balance_loan_amount.

The screenshot shows the Power Query Editor with the merged table 'Merge (2)'. A new column 'balance_loan_amount' has been added using the formula `Table.AddColumn(#"Removed duplicates", "balance_loan_amount", each [loan_amount])`. The table now includes an additional column 'balance_loan_amount' with values such as 5000, 15401, 10850, etc. The 'Query settings' pane remains the same as in the previous screenshot.

3. MERGE-LOANS & CUSTOMERS.

Merge

Select a table and matching columns to create a merged table.

Left table for merge *

customers1

customer_id	first_name	last_name	address	city	state	zip
1	John	Doe	123 Elm St	Toronto	ON	M4B1B3
2	Jane	Smith	456 Maple Ave	Ottawa	ON	K1A0B1
3	Michael	Johnson	789 Oak Dr	Montreal	QC	H1A1A1
4	Emily	Davis	101 Pine Rd	Calgary	AB	T2A0A1

Right table for merge *

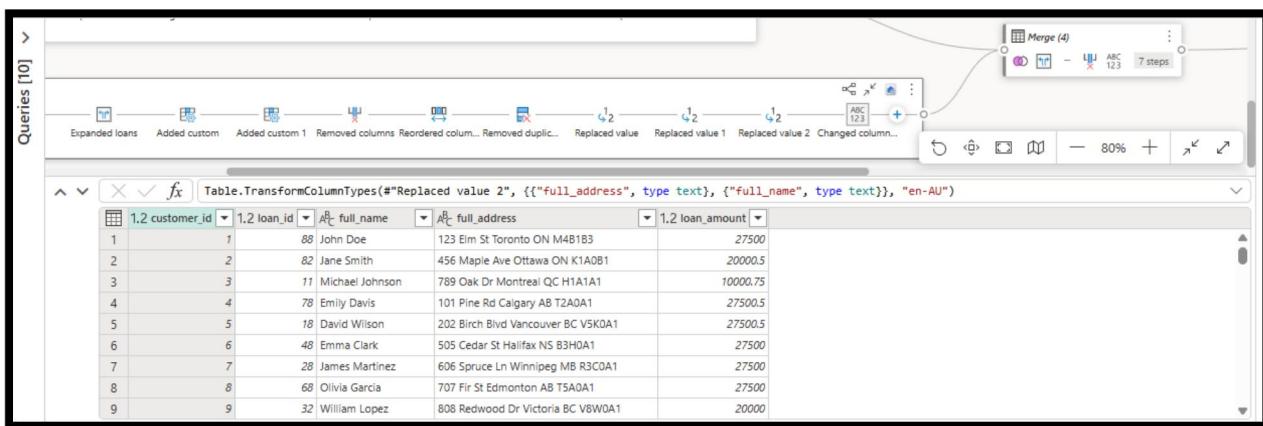
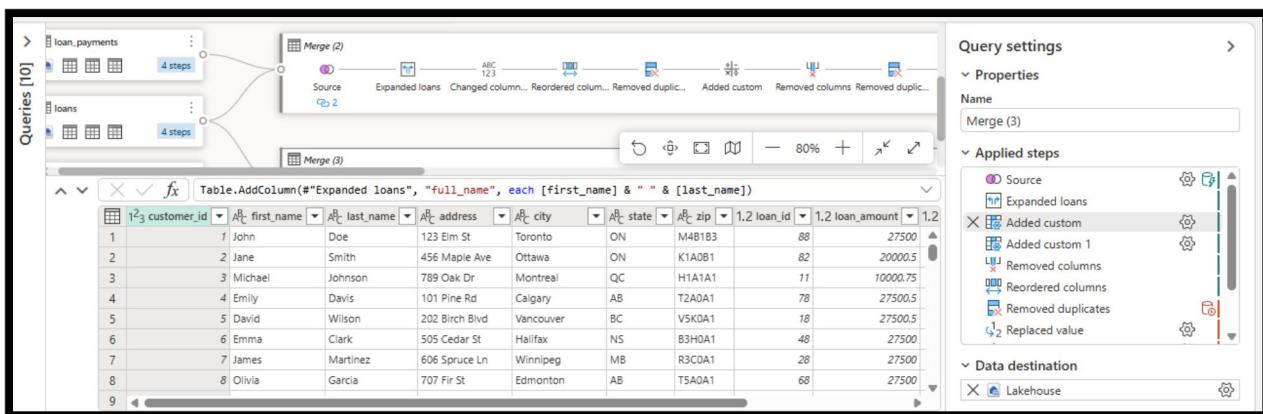
loans

loan_id	customer_id	loan_amount	interest_rate
1	45	10000.5	5.5

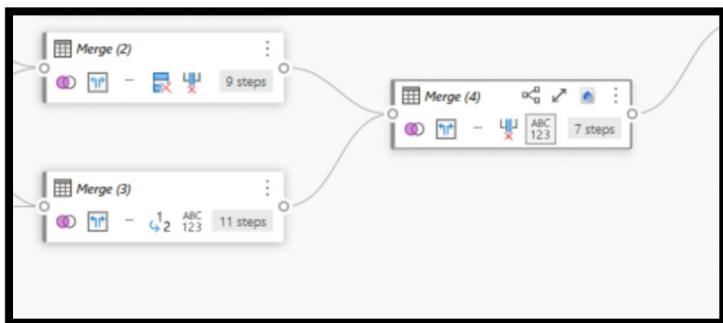
The selection matches 86 of 87 rows from the first table

OK Cancel

NANDINI RATHORE



4.MERGE-



NANDINI RATHORE

Merge ?

Select a table and matching columns to create a merged table.

Left table for merge *

Merge (3)

1.2 customer_id	1.2 loan_id	A _C full_name	A _C full_address	1.2 loan_amount
1	88	John Doe	123 Elm St Toronto ON M4B1B3	275
2	82	Jane Smith	456 Maple Ave Ottawa ON K1A0B1	2000
3	11	Michael Johnson	789 Oak Dr Montreal QC H1A1A1	10000
4	78	Emily Davis	101 Pine Rd Calgary AB T2A0A1	2750

Right table for merge *

Merge (2)

1 ² 3 customer_id	1 ² 3 loan_id	1 ² 3 payment_id	1 ² 3 balance_loan_amount
45	1	99	5000

The selection matches 85 of 87 rows from the first table

OK **Cancel**

Power Query | Draft saved

Home Transform Add column View Help

Queries [10]

Source
Merge

Direct referenced queries
Merge (2)
Merge (3)

Indirect referenced queries
loan_payments
loans
customers1

This step will be evaluated outside the data source.
[Learn more](#)

This source is not supported by fast copy.

Power Query

Queries [10]

Merge (4)

Source Expanded Merge... Changed column... Sorted rows Removed duplicates Removed columns Changed column...

Table.Distinct(#"Sorted rows", {"customer_id"})

1 ² 3 customer_id	1.2 loan_id	A _C full_name	A _C full_address	1.2 loan_amount	1 ² 3 payment_id	1 ² 3 balance_loan_amount
1	1	John Doe	123 Elm St Toronto ON M4B1B3	27500	17	5000
2	2	Jane Smith	456 Maple Ave Ottawa ON K1A0B1	20000.5	71	20000.5
3	3	Michael Johnson	789 Oak Dr Montreal QC H1A1A1	10000.75	10	10000.75
4	4	Emily Davis	101 Pine Rd Calgary AB T2A0A1	27500.5	8	27500.5
5	5	David Wilson	202 Birch Blvd Vancouver BC V5K0A1	27500.5	47	27500.5
6	6	Emma Clark	505 Cedar St Halifax NS B3H0A1	27500	77	27500
7	7	James Martinez	606 Spruce Ln Winnipeg MB R3C0A1	27500	57	27500
8	8	Olivia Garcia	707 Fir St Edmonton AB T5A0A1	27500	97	27500

Query settings

Name: Merge (4)

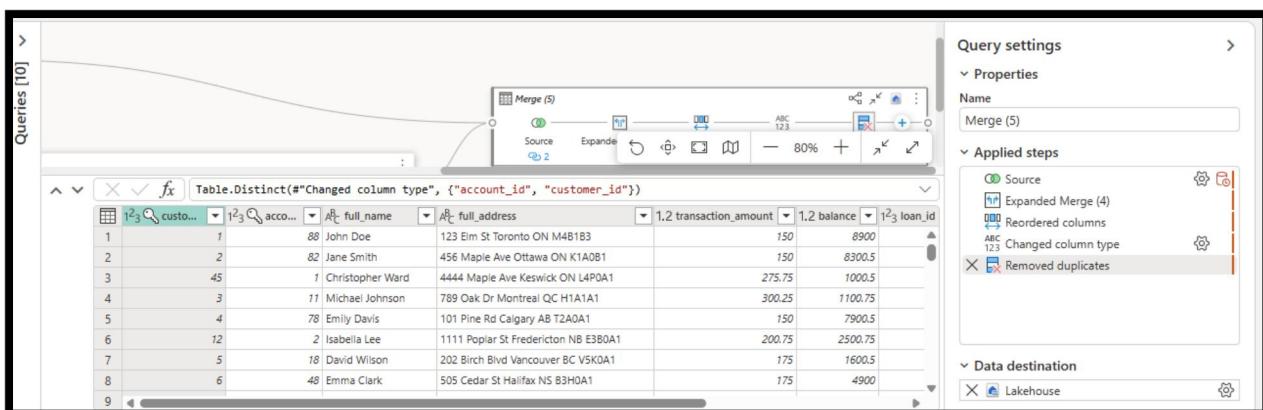
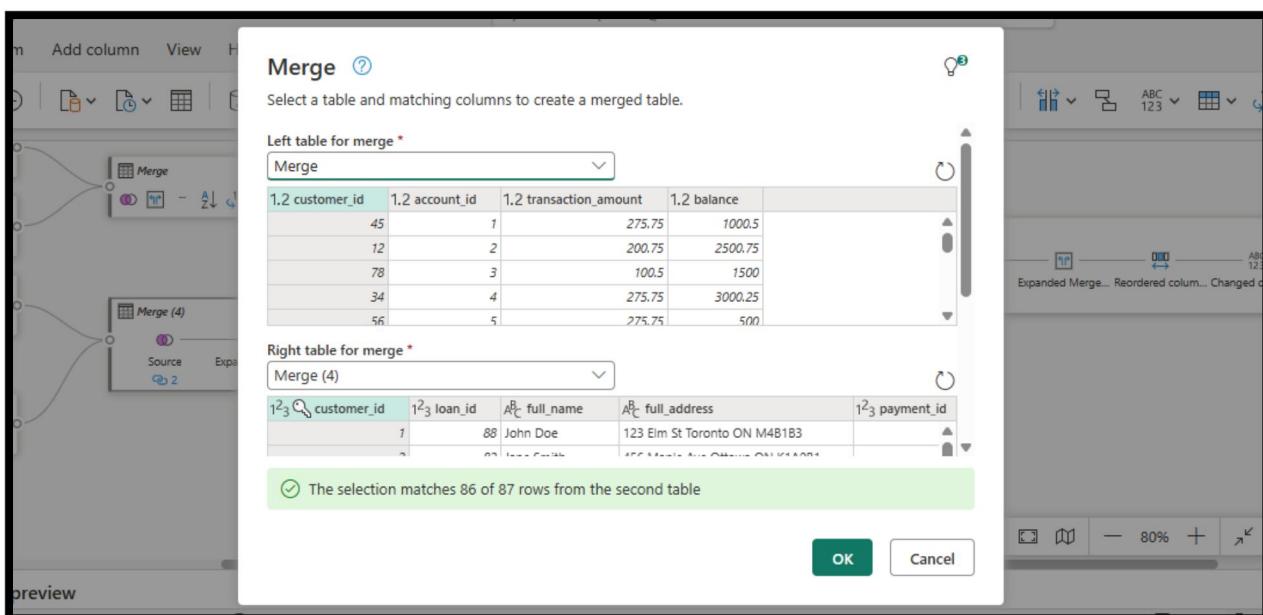
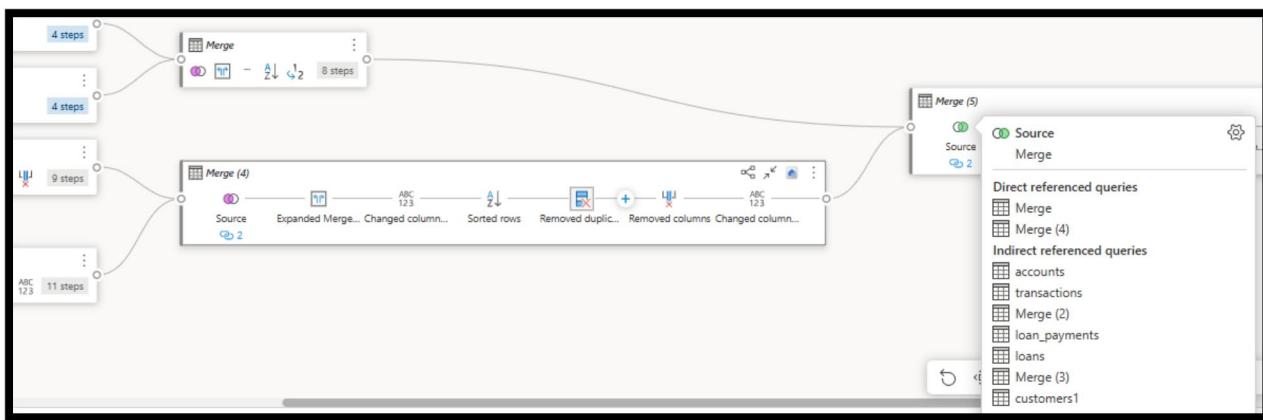
Applied steps

- Source
- Expanded Merge (2)
- ABC 123
- Changed column type
- Z↓ Sorted rows
- X Removed duplicates
- L_U Removed columns
- ABC 123 Changed column type 1

Data destination

Lakehouse

5.MERGE-ALL TABLES.



Now transformations are done and I get 5 merge tables here and load them into lakehouse.

NANDINI RATHORE

Now for scd type1 , i selected the sql endpoints where tables are automatically created if present in lakehouse. We can't create or alter tables in lakehouse. So I use warehouse to create,alter tables.

The screenshot shows the Power BI Data Studio interface. On the left, the 'Explorer' pane is open under the 'Warehouses' tab, showing a list of tables and objects: 'createdby', 'createddate', 'updatedby', 'updateddd...', 'hashkey', 'table_scd_typ1', 'Views', 'Functions', and 'Stored Procedures'. The main area displays a T-SQL query for SCD Type1:

```
1 SELECT TOP (100) [customer_id],  
2     [loan_id],  
3     [full_name],  
4     [full_address],  
5     [payment_id],  
6     [balance_loan_amount]  
7 FROM [bronze_lakehouse].[dbo].[table4_Merge _accpayloan]
```

The status bar at the bottom indicates: 2 sec - Command executed using warehouse in 12 sec by NANDINI on 6:08:39 PM, 8/30/25.

Right click on the table and selected sql query.

```
1 ✓ create table TARGET_TABLE2  
2 ✓ (customer_id int,loan_id int,full_name varchar(100),full_address varchar(100),payment_id int,balance_loan_amount in  
3 DATETIME2(3),hashkey VARBINARY(32))  
4 select * from TARGET_TABLE2  
5
```

Creating target table.

```

1  select src.customer_id,
2      src.loan_id,
3      src.full_name,
4      src.full_address,
5      src.payment_id,
6      src.balance_loan_amount,
7      CONVERT(VARBINARY(32),
8      HASHBYTES('SHA2_256',
9      CONCAT(
10         CAST(src.customer_id AS VARCHAR),
11         CAST(src.loan_id AS VARCHAR),
12         CAST(src.payment_id AS VARCHAR),
13         CAST(src.balance_loan_amount AS VARCHAR),
14         src.full_name,
15         src.full_address
16     )))
17 ) AS src_hashkey
18 FROM [bronze_lakehouse].[dbo].[table4_Merge _accpayloan] src
19

```

✓ 1 sec - Command executed using warehouse in 1 sec by NANDINI on 12:31:59 AM, 8/31/25

T-SQL

Generating hashkey.i use Varbinary (32),so that in the end I didn't get any negative value.

Crc32 is not supported in fabric that's why I used SHA2_256.

- **CRC32** is fast but **not cryptographically strong**; it has a higher chance of collisions (different inputs producing the same output).
- **SHA2_256** is slower but **collision-resistant and secure**, making it reliable for **change detection, deduplication, and sensitive data hashing**.

We use SHA2_256 instead of CRC32 because it's **much safer and reliable for detecting changes and ensuring data integrity**.

	12L customer_id	12L loan_id	ANY full_name	ANY full_address	12L payment_id	12L balance_loan_amount	ANY src_hashkey
1	1	88	John Doe	123 Elm St Toron...	17	26600	0xAAA56555E743...
2	2	82	Jane Smith	456 Maple Ave ...	71	16400	0x50CC3FBB2A7...
3	3	11	Michael Johnson	789 Oak Dr Mon...	10	9451	0x3B001365D8E...
4	4	78	Emily Davis	101 Pine Rd Calg...	8	27050	0xCE21FE56D1A...
5	5	18	David Wilson	202 Birch Blvd V...	47	25100	0x87E1EAC7AC0...
6	6	48	Emma Clark	505 Cedar St Hal...	77	23600	0x7A3B985819F0...
7	7	28	James Martinez	606 Spruce Ln W...	57	24600	0xA5917AB9DEA...
8	8	68	Olivia Garcia	707 Fir St Edmon...	97	22600	0xCDCEDEE3873...
9	9	32	William Lopez	808 Redwood Dr...	21	18900	0x934FE7918C76...
10	10	52	Ava Anderson	909 Cypress Ave ...	41	17900	0xBF87D40ACD1...
11	11	24	Alexander Tho...	1010 Willow Rd ...	93	25300	0x85D9702FED0...
12	12	2	Isabella Lee	1111 Poplar St Fr...	91	15401	0x674A87D22B2...
13	13	44	Daniel Harris	1212 Ash Blvd C...	13	29300	0xD9E295261B8...
14	14	9	Sophia Young	1313 Beech Dr Y...	28	31050	0xB54631C9AC2...
15	15	38	Matthew King	1414 Cedar Ln W...	67	24100	0x2B94BABEBE5...
16	16	58	Charlotte Scott	1515 Elm St Iqal...	87	23100	0xF69D77FC941...

This is a alias of source table with target table .src.customer_id=tgt.customer_id and
Src.hashkey=tgt.hashkey.

- ✓ When doing LEFT JOIN with an alias, you **cannot just use src.hashkey** if it's not a real column in the source.
- ✓ You need to **compute the hash in the query** so the join can accurately detect new or changed rows.

```

1  select src.customer_id,
2      src.loan_id,
3      src.full_name,
4      src.full_address,
5      src.payment_id,
6      src.balance_loan_amount,
7      CONVERT(VARBINARY(32),
8      HASHBYTES('SHA2_256',
9      CONCAT(
10         CAST(src.customer_id AS VARCHAR),
11         CAST(src.loan_id AS VARCHAR),
12         CAST(src.payment_id AS VARCHAR),
13         CAST(src.balance_loan_amount AS VARCHAR),
14         src.full_name,
15         src.full_address
16     )))
17 ) AS src_hashkey
18 FROM [bronze_lakehouse].[dbo].[table4_Merge_accpayloan] src
19 left join [TARGET_TABLE2] tgt
20 on src.customer_id=tgt.customer_id
21 AND CONVERT(VARBINARY(32),
22 HASHBYTES('SHA2_256',
23     CONCAT(
24         CAST(src.customer_id AS VARCHAR),
25         CAST(src.loan_id AS VARCHAR),
26         CAST(src.payment_id AS VARCHAR),
27         CAST(src.balance_loan_amount AS VARCHAR),
28         src.full_name,
29         src.full_address
30     )))
31 )=tgt.hashkey;
32

```

```

19 left join [TARGET_TABLE2] tgt
20 on src.customer_id=tgt.customer_id
21 AND CONVERT(VARBINARY(32),
22 HASHBYTES('SHA2_256',
23     CONCAT(
24         CAST(src.customer_id AS VARCHAR),
25         CAST(src.loan_id AS VARCHAR),
26         CAST(src.payment_id AS VARCHAR),
27         CAST(src.balance_loan_amount AS VARCHAR),
28         src.full_name,
29         src.full_address
30     )))
31 )=tgt.hashkey;
32

```

left join

```

1   update tgt
2     SET
3       tgt.customer_id=src.customer_id,
4       tgt.loan_id=src.loan_id,
5       tgt.full_name=src.full_name,
6       tgt.full_address=src.full_address,
7       tgt.payment_id=src.payment_id,
8       tgt.balance_loan_amount=src.balance_loan_amount,
9       tgt.hashkey=src.src_hashkey,
10      updateddate=getdate(),
11      updatedby='fabric-updated'
12  FROM [TARGET_TABLE2] tgt
13  join (
14    SELECT *,
15      CONVERT(VARBINARY(32),
16      HASHBYTES('SHA2_256',
17      CONCAT(
18        CAST(customer_id AS VARCHAR),

```

```

7          tgt.payment_id=src.payment_id,
8          tgt.balance_loan_amount=src.balance_loan_amount,
9          tgt.hashkey=src.src_hashkey,
10         updateddate=getdate(),
11         updatedby='fabric-updated'
12    FROM [TARGET_TABLE2] tgt
13    join (
14      SELECT *,
15        CONVERT(VARBINARY(32),
16        HASHBYTES('SHA2_256',
17        CONCAT(
18          CAST(customer_id AS VARCHAR),
19          CAST(loan_id AS VARCHAR),
20          CAST(payment_id AS VARCHAR),
21          CAST(balance_loan_amount AS VARCHAR),
22          full_name,
23          full_address
24        )))
25        )AS src_hashkey
26    FROM [bronze_lakehouse].[dbo].[table4_Merge_accpayloan]) AS src
27  on tgt.customer_id=src.customer_id
28  where tgt.hashkey <> src.src_hashkey
29  OR tgt.hashkey IS NULL;
30

```

```

1  INSERT INTO TARGET_TABLE2
2  (customer_id, loan_id, full_name, full_address, payment_id, balance_loan_amount,
3  SELECT customer_id, loan_id, full_name, full_address, payment_id, balance_loan_am
4  CONVERT(VARBINARY(32),
5      HASHBYTES('SHA2_256',
6      CONCAT(
7          CAST(customer_id AS VARCHAR),
8          CAST(loan_id AS VARCHAR),
9          CAST(payment_id AS VARCHAR),
10         CAST(balance_loan_amount AS VARCHAR),
11         full_name,
12         full_address
13     )))
14   ),
15   'fabric', GETDATE(), 'fabric', GETDATE()
16 FROM bronze_lakehouse.dbo.[table4_Merge_accpayloan] src
17 WHERE NOT EXISTS (
18     SELECT 1 FROM TARGET_TABLE2 tgt
19     WHERE tgt.customer_id = src.customer_id
20 );
21

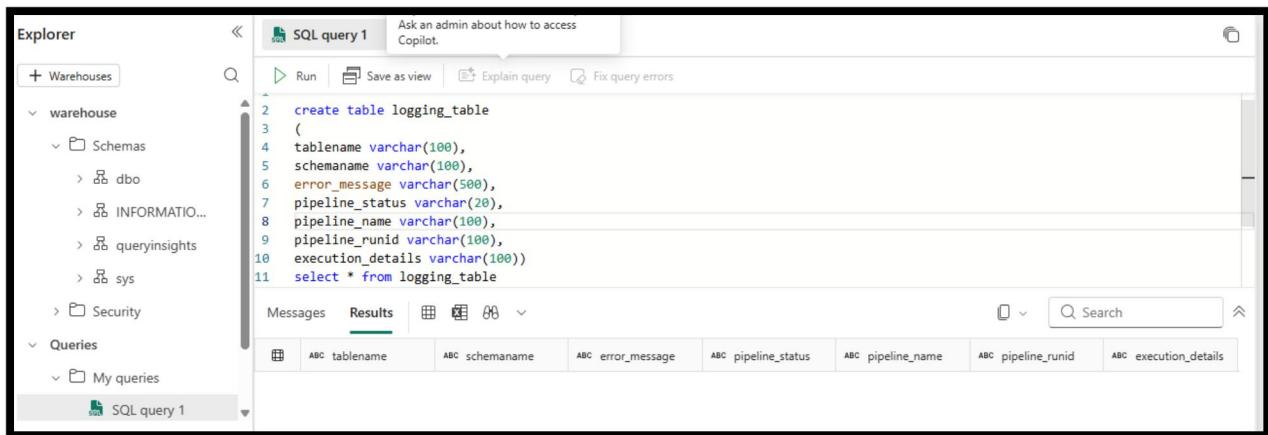
```

Datetime2 or datetime is not supported here that's why I used getdate() here.

The screenshot shows a database interface with a query results table. The table has 11 columns and 87 rows. The columns are labeled: customer_id, loan_id, full_name, full_address, payment_id, balance_loan_amount, createdby, and a timestamp column. The 'createdby' column consistently contains the value 'fabric'. The 'full_address' column contains various street addresses. The 'balance_loan_amount' column contains numerical values ranging from 16400 to 26600. The 'payment_id' column contains numerical values ranging from 7 to 97. The 'loan_id' column contains numerical values ranging from 88 to 52. The 'customer_id' column contains numerical values ranging from 1 to 10. The 'full_name' column contains names such as John Doe, Jane Smith, Michael Johnson, Emily Davis, David Wilson, Emma Clark, James Martinez, Olivia Garcia, William Lopez, and Ava Anderson.

	customer_id	loan_id	full_name	full_address	payment_id	balance_loan_amount	createdby	
1	1	88	John Doe	123 Elm St Toron...	17	26600	fabric	
2	2	82	Jane Smith	456 Maple Ave ...	71	16400	fabric	
3	3	11	Michael Johnson	789 Oak Dr Mon...	10	9451	fabric	
4	4	78	Emily Davis	101 Pine Rd Calg...	8	27050	fabric	
5	5	18	David Wilson	202 Birch Blvd V...	47	25100	fabric	
6	6	48	Emma Clark	505 Cedar St Hal...	77	23600	fabric	
7	7	28	James Martinez	606 Spruce Ln W...	57	24600	fabric	
8	8	68	Olivia Garcia	707 Fir St Edmon...	97	22600	fabric	
9	9	32	William Lopez	808 Redwood Dr...	21	18900	fabric	
10	10	52	Ava Anderson	909 Cypress Ave ...	41	17900	fabric	

LOGGING TABLE :- creating in warehouse.



The screenshot shows the Azure Data Studio interface. On the left, the Explorer sidebar is open, showing a tree structure with 'warehouse' selected. Under 'warehouse', there are nodes for 'Schemas', 'dbo', 'INFORMATION...', 'queryinsights', 'sys', 'Security', and 'Queries'. In the main area, a SQL query window titled 'SQL query 1' is active. It contains the following T-SQL code:

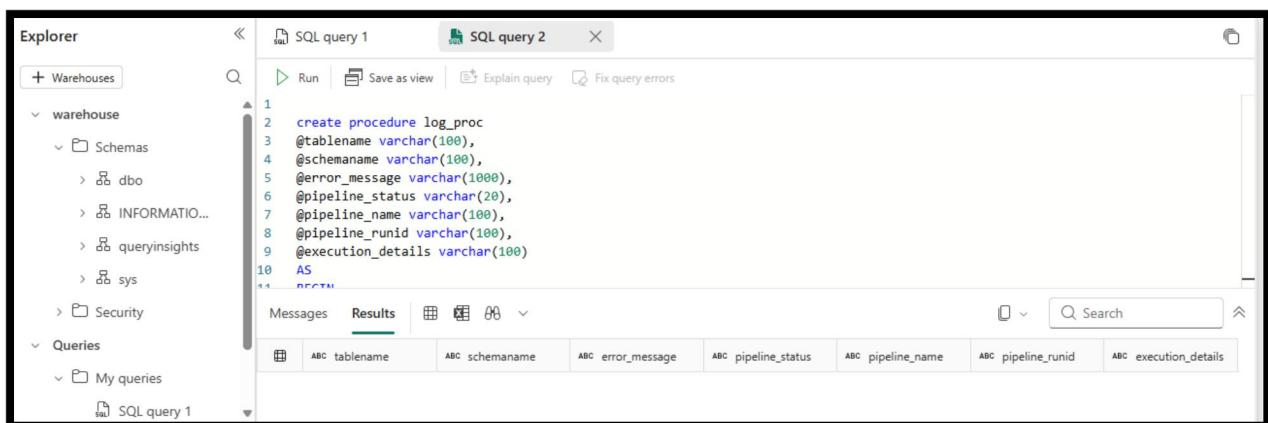
```

1 create table logging_table
2 (
3     tablename varchar(100),
4     schemaname varchar(100),
5     error_message varchar(500),
6     pipeline_status varchar(20),
7     pipeline_name varchar(100),
8     pipeline_rnid varchar(100),
9     execution_details varchar(100)
10 select * from logging_table
11

```

Below the code, the 'Results' tab is selected, showing a table header with columns: ABC tablename, ABC schemaname, ABC error_message, ABC pipeline_status, ABC pipeline_name, ABC pipeline_rnid, and ABC execution_details. A message bar at the top right says 'Ask an admin about how to access Copilot.'

STORED PROCEDURE



The screenshot shows the Azure Data Studio interface. The Explorer sidebar is open, showing the same tree structure as the previous screenshot. In the main area, a SQL query window titled 'SQL query 1' is active. It contains the following T-SQL code:

```

1 create procedure log_proc
2     @tablename varchar(100),
3     @schemaname varchar(100),
4     @error_message varchar(1000),
5     @pipeline_status varchar(20),
6     @pipeline_name varchar(100),
7     @pipeline_rnid varchar(100),
8     @execution_details varchar(100)
9
10 AS
11     BEGIN
12         SET NOCOUNT ON
13         insert into logging_table VALUES
14             (@tablename,
15             @schemaname,
16             @error_message,
17             @pipeline_status,
18             @pipeline_name,
19             @pipeline_rnid,
20             @execution_details
21         )
22     END
23
24     select * from logging_table

```

Below the code, the 'Results' tab is selected, showing a table header with columns: ABC tablename, ABC schemaname, ABC error_message, ABC pipeline_status, ABC pipeline_name, ABC pipeline_rnid, and ABC execution_details. A message bar at the top right says 'Ask an admin about how to access Copilot.'



The screenshot shows the Azure Data Studio interface. The Explorer sidebar is open, showing the same tree structure. In the main area, a SQL query window titled 'SQL query 1' is active. It contains the following T-SQL code:

```

1 BEGIN
2     SET NOCOUNT ON
3     insert into logging_table VALUES
4         (@tablename,
5         @schemaname,
6         @error_message,
7         @pipeline_status,
8         @pipeline_name,
9         @pipeline_rnid,
10        @execution_details
11    )
12
13 END
14
15 select * from logging_table

```

Below the code, the 'Results' tab is selected, showing a table header with columns: ABC tablename, ABC schemaname, ABC error_message, ABC pipeline_status, ABC pipeline_name, ABC pipeline_rnid, and ABC execution_details. A message bar at the top right says 'Ask an admin about how to access Copilot.'

UPDATE SOURCE

For update the values in scd type 1 target table:- I inserted new rows in source table and update few rows.

```

warehouse  v   Search   Trial: 57 days left
Home Management Help
Get data New SQL query SQL templates Query activity New semantic model Open in New API for GraphQL ...
Explorer logging table stored procedure table4_Merge _accpayloan1 update table in ...
Run Save as view Explain query Fix query errors
1  SELECT TOP (100) [customer_id],
2      [loan_id],
3      [full_name],
4      [full_address],
5      [payment_id],
6      [balance_loan_amount]
7  FROM [warehouse].[dbo].[table4_Merge _accpayloan1]
8  insert into [warehouse].[dbo].[table4_Merge _accpayloan1] values (88,20,'nandini','123 abc xyz',11,200),(89,10,'abcd','321zyx cba',40,1000)

```

```

9  update  [warehouse].[dbo].[table4_Merge _accpayloan1]
10 set full_name='joy smith'
11 where customer_id=1
12 update  [warehouse].[dbo].[table4_Merge _accpayloan1]
13 set full_address='4800 Spruce Ln Beaverton ON L0K0B2'
14 where customer_id=49

```

86	49	98	Joshua Bennett	4800 Spruce Ln Beaverton ON L...	27	26100
87	1	88	joy smith	123 Elm St Toronto ON M4B1B3	17	26600
88	88	20	nandini	123 abc xyz	11	200
89	89	10	abcd	321zyx cba	40	1000

UPDATE SOURCE TABLE.

```

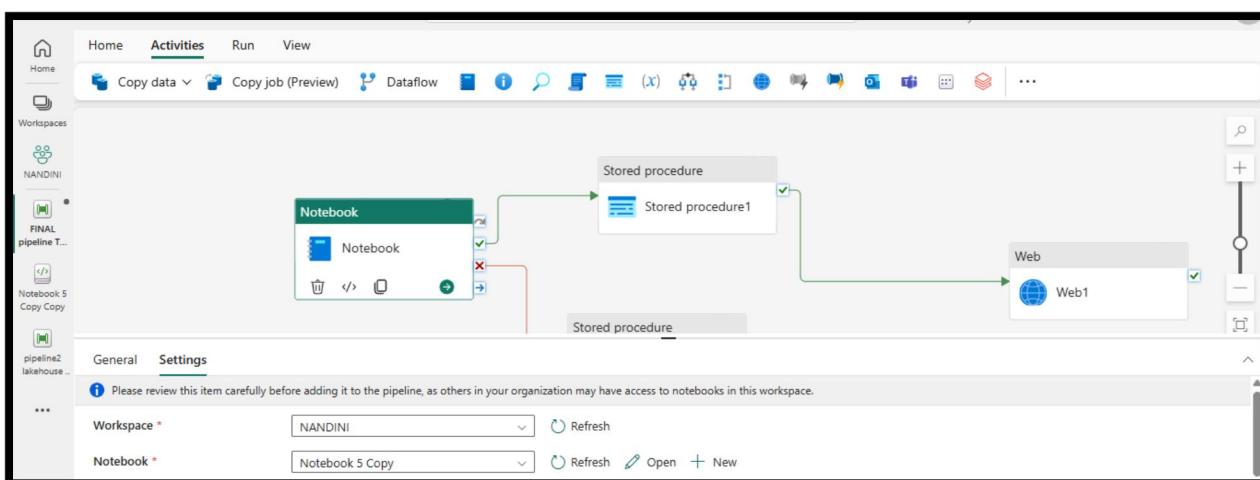
1  SELECT TOP (100) [customer_id],
2      [loan_id],
3      [full_name],
4      [full_address],
5      [payment_id],
6      [balance_loan_amount]
7  FROM [dbo].[table4_Merge _accpayloan1]

```

✓ 9 sec - Command executed using warehouse in 1 sec by NANDINI on 2:33:54 PM, 8/31/25

NANDINI RATHORE

123 loan_id	ANY full_name	ANY full_address	123 payment_id	123 balance_loa...	ANY createdby	ANY createddate	ANY update...
/1	Andrew Hami...	/272 Maple Ave ...	/0	6451	fabric	2025-08-31 0...	fabric
39	Harper Graham	7373 Oak Dr Bal...	58	29551	fabric	2025-08-31 0...	fabric
59	Joshua Sullivan	7474 Pine Rd Bra...	78	28551	fabric	2025-08-31 0...	fabric
19	Evelyn Wallace	7575 Birch Blvd ...	38	30551	fabric	2025-08-31 0...	fabric
91	Daniel Woods	7676 Spruce Ln ...	90	5451	fabric	2025-08-31 0...	fabric
3	Abigail Cole	7777 Fir St Sundr...	82	10850	fabric	2025-08-31 0...	fabric
93	James West	7878 Redwood ...	72	11350	fabric	2025-08-31 0...	fabric
99	Emily Jordan	7979 Cypress Av...	18	31551	fabric	2025-08-31 0...	fabric
12	Michael Owens	8080 Willow Rd ...	5	19700	fabric	2025-08-31 0...	fabric
83	Elizabeth Rey...	8181 Poplar St St...	62	11851	fabric	2025-08-31 0...	fabric
43	David Fisher	8282 Ash Blvd V...	22	13851	fabric	2025-08-31 0...	fabric
63	Sophia Ellis	8383 Beech Dr Fi...	42	12851	fabric	2025-08-31 0...	fabric
33	John Harrison	8484 Cedar Ln Te...	12	14350	fabric	2025-08-31 0...	fabric
53	Olivia Gibson	8585 Elm St New...	32	13350	fabric	2025-08-31 0...	fabric
73	William McD...	N/A	52	12350	fabric	2025-08-31 0...	fabric
88	joy smith	123 Elm St Toron...	17	26600	fabric	2025-08-31 0...	fabric-updated
98	Joshua Bennett	4800 Spruce Ln ...	27	26100	fabric	2025-08-31 0...	fabric-updated



Final pipeline having notebook ,stored procedure and web activity run successfully.

General Settings

Stored procedure name * [dbo].[log_proc] Refresh

Stored procedure parameters ①

Import New Delete

Name	Type *	Value	Treat as null
error_message	String	Value	<input checked="" type="checkbox"/>
execution_details	String	pipeline().TriggerTime	<input type="checkbox"/>
increment_column	String	@activity('Notebook').output	<input type="checkbox"/>
Ipv	String	@activity('Notebook').output	<input type="checkbox"/>
pipeline_name	String	pipeline().Pipeline	<input type="checkbox"/>
pipeline_runid	String	pipeline().RunID	<input type="checkbox"/>
pipeline_status	String	successful	<input type="checkbox"/>
schemaname	String	dbo	<input type="checkbox"/>

Stored procedure for success pipeline.

General Settings

Connection * warehouse Refresh Open

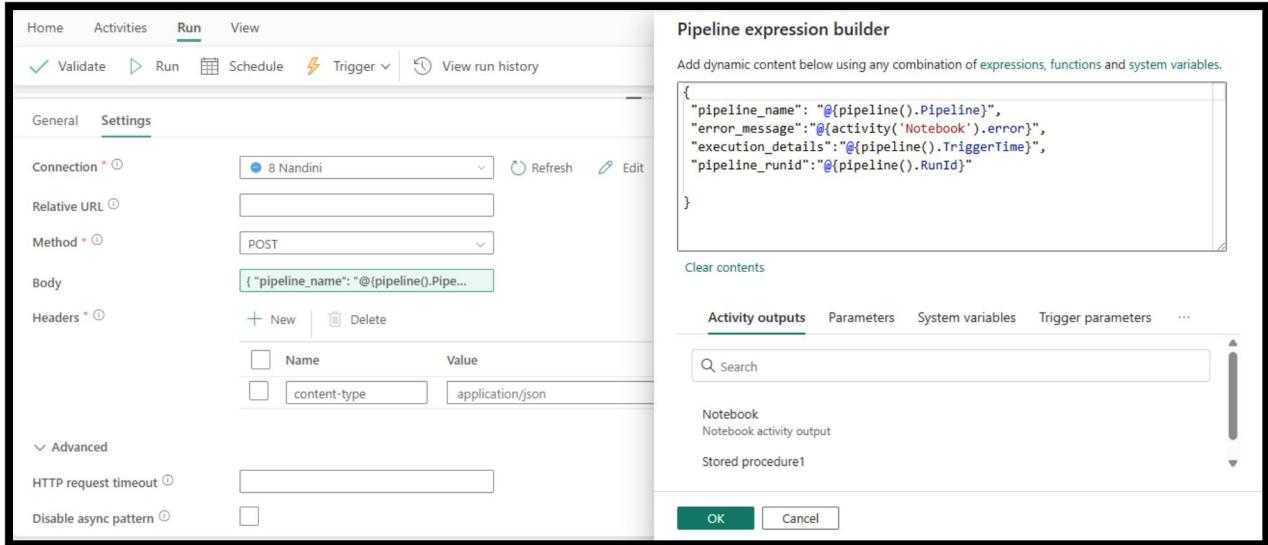
Stored procedure name * [dbo].[log_proc] Refresh

Stored procedure parameters ①

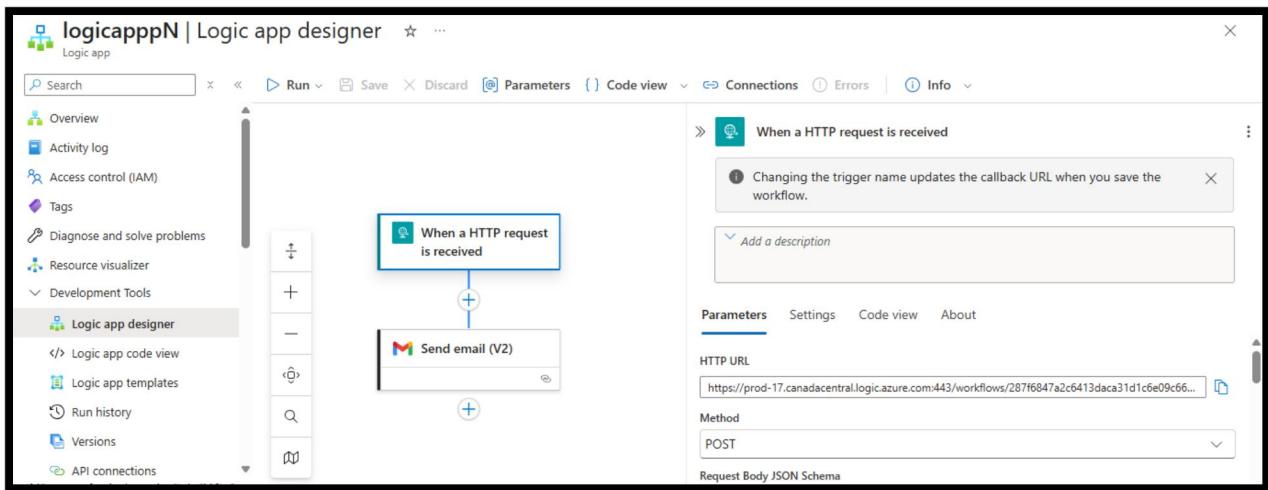
Import New Delete

Name	Type *	Value	Treat as null
error_message	String	fail	<input type="checkbox"/>
execution_details	String	pipeline().TriggerTime	<input type="checkbox"/>
increment_column	String	@activity('Notebook').output	<input type="checkbox"/>
Ipv	String	@activity('Notebook').output	<input type="checkbox"/>
pipeline_name	String	pipeline().Pipeline	<input type="checkbox"/>
pipeline_runid	String	pipeline().RunID	<input type="checkbox"/>

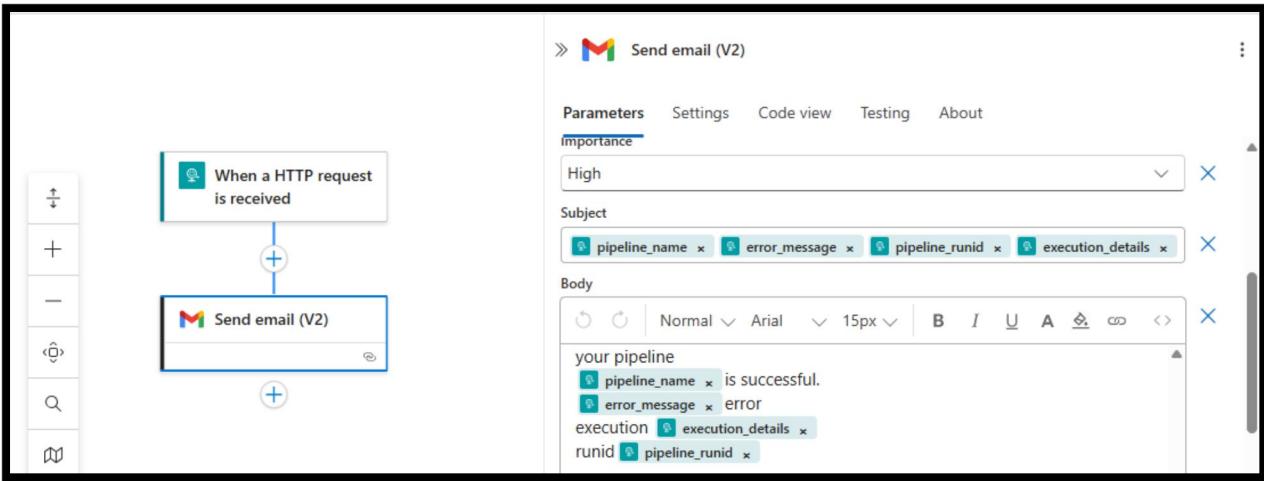
Stored procedure for fail pipeline.



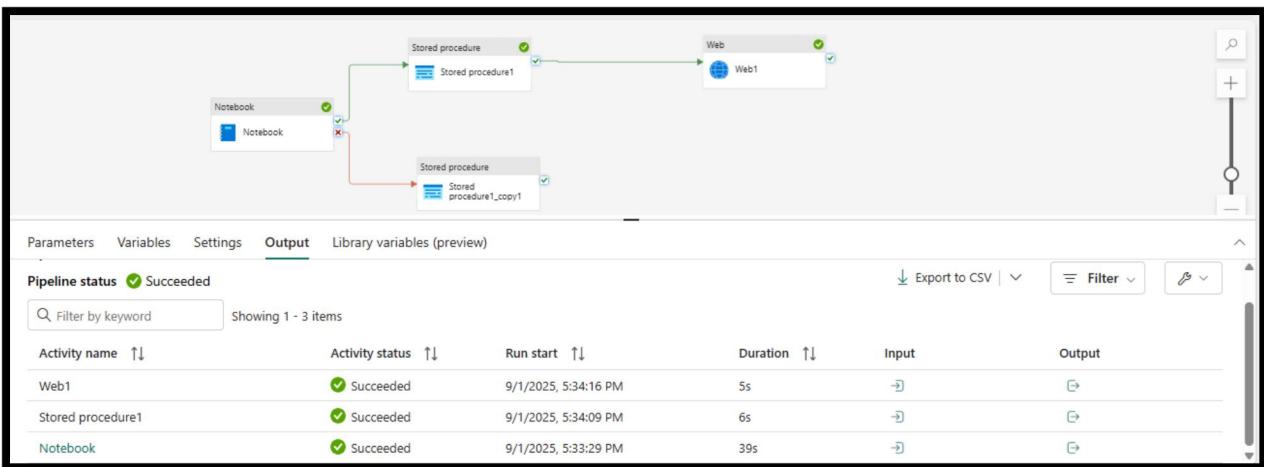
In web activity,in body I wrote this json code to get pipeline details in mail notifications.



Paste that json code in logic app body and save it ,then copy this url from here.



Here in the email, write the message in the body that I want in mail notification.



Paste that url here in web activity connection, run the pipeline.



This is the mail notification I get by using logic app and web activity in fabric.

The screenshot shows a SQL query window with the following content:

```
1 create table increment_watermark(tablename varchar(100),schemaname varchar(100),lpv int,increment_column varchar(100))
2 insert into increment_watermark values('TARGET_TABLE3','dbo',0,'customer_id')
3 select * from increment_watermark
```

The results pane displays the data inserted into the watermark table:

	tablename	schemaname	lpv	increment_column
1	TARGET_TABLE3	dbo	0	customer_id

This is the watermark table to get last processed value .

In stored procedure.

The screenshot shows a SQL query window with the following content:

```
1 increment_column varchar(100),
2 execution_details varchar(100)
3 select * from logging_table
4 drop table logging_table
5
6
7
8
```

The results pane displays the data from the logging table after running the stored procedure:

	tablename	schemaname	error_mes...	pipeline_st...	pipeline_n...	pipeline_r...	lpv	increment...	execution...
1	table4_Merge...	dbo	NULL	successful	7e607389-917...	9f91cc1b-bf87...	System.Col	System.Collections...	pipeline().Trigg...

The result here in stored procedure after running pipeline.

SCD TYPE1 OF SECOND TABLE.

The screenshot shows the Snowflake UI interface. In the top navigation bar, there are links for Home, Help, New SQL query, Query activity, New semantic model, Download SQL database project, Open in, New API for GraphQL, and Copilot. On the left, the Explorer sidebar shows the schema structure under the 'bronze_lakehouse' database, including the 'dbo' folder which contains tables like 1_table_acc_transaction, 2_table_loan_loanpayment, 3_table_Merge_cust_loans, accounts, clean_acc_trans, clean_loan_payment, clean_Merge_3_cust_loan, and clean Merge 4 cust loan. The '3_table_Merge_cust_loans' table is currently selected. The main area displays the 'Data preview - 3_table_Merge_cust_loans' with a message 'Showing 1000 rows'. The table data is as follows:

	customer_id	loan_id	full_name	full_address	loan_amount
1	1	88	John Doe	123 Elm St Toronto O...	27500
2	45	1	Christopher Ward	4444 Maple Ave Kesw...	10000.5
3	12	2	Isabella Lee	1111 Poplar St Freder...	20000.75
4	2	82	Jane Smith	456 Maple Ave Ottaw...	20000.5
5	78	3	Abigail Cole	7777 Fir St Sundridge ...	15000
6	34	4	Olivia Reed	3333 Birch Blvd Orillia...	30000.25
7	3	11	Michael Johnson	789 Oak Dr Montreal ...	10000.75
8	56	5	Elizabeth Long	5555 Beech Dr East G...	25000
9	23	6	Joshua Turner	2222 Redwood Dr Wi...	17500.5
10	4	78	Emily Davis	101 Pine Rd Calgary A...	27500.5
11	67	8	Matthew Russell	6666 Willow Rd Penet...	27500
12	14	9	Sophia Young	1313 Beech Dr Yellow...	32500.25
13	5	18	David Wilson	202 Birch Blvd Vancou...	27500.5

The screenshot shows the T-SQL command window with the following code:

```

1  SELECT TOP (100) [customer_id],
2      [loan_id],
3      [full_name],
4      [full_address],
5      [loan_amount]
6  FROM [bronze_lakehouse].[dbo].[3_table_Merge_cust_loans]

```

The command was executed successfully. Below the command window is a 'Table view' preview showing the same data as the previous screenshot, with 5 columns and 87 rows.

The screenshot shows the T-SQL command window with the following code:

```

1  create table NEW_TARGET_TABLE
2  (customer_id int,loan_id int,full_name varchar(100),full_address varchar(100),loan_amount int,createdby varchar(100),createddate DATETIME2(3),hashkey VARBINARY(32))
3
4  select * from NEW_TARGET_TABLE
5

```

The command was executed successfully. Below the command window is a 'Table view' preview showing the same data as the previous screenshots, with 5 columns and 87 rows.

```

1  select src.customer_id,
2    src.loan_id,
3    src.full_name,
4    src.full_address,
5    src.loan_amount,
6    CONVERT(VARBINARY(32),
7    HASHBYTES('SHA2_256',
8    CONCAT(
9      CAST(src.customer_id AS VARCHAR),
10     CAST(src.loan_id AS VARCHAR),
11     CAST(src.loan_amount AS VARCHAR),
12     src.full_name,
13     src.full_address
14   )))
15  ) AS src_hashkey
16  FROM [bronze_lakehouse].[dbo].[3_table_Merge_cust_loans] src
17

```

✓ - Command executed using warehouse in 1 sec by NANDINI on 12:31:59 AM, 8/31/25

T-SQL

alias

```

1  select src.customer_id,
2    src.loan_id,
3    src.full_name,
4    src.full_address,
5    src.loan_amount,
6    CONVERT(VARBINARY(32),
7    HASHBYTES('SHA2_256',
8    CONCAT(
9      CAST(src.customer_id AS VARCHAR),
10     CAST(src.loan_id AS VARCHAR),
11     CAST(src.loan_amount AS VARCHAR),
12     src.full_name,
13     src.full_address
14   )))
15  ) AS src_hashkey
16  FROM [bronze_lakehouse].[dbo].[3_table_Merge_cust_loans] src
17  left join [NEW_TARGET_TABLE] tgt
18  on src.customer_id=tgt.customer_id
19  AND CONVERT(VARBINARY(32),
20  HASHBYTES('SHA2_256',

```

```

20  HASHBYTES('SHA2_256',
21  CONCAT(
22    CAST(src.customer_id AS VARCHAR),
23    CAST(src.loan_id AS VARCHAR),
24    CAST(src.loan_amount AS VARCHAR),
25    src.full_name,
26    src.full_address
27  )))
28  )=tgt.hashkey;
29
30

```

✓ - Command executed using warehouse in 1 sec by NANDINI on 12:22:14 AM, 8/31/25

> Messages

Table view

Download Save as

	12F customer_id	12F loan_id	ANY full_name	ANY full_address	12F loan_amount	ANY src_hashkey
1	23	6	Joshua Turner	2222 Redwood ...	17500.5	0x7007C679B01...
2	8	68	Olivia Garcia	707 Fir St Edmon...	27500	0x8760359978A...
3	56	5	Elizabeth Long	5555 Beech Dr E...	25000	0x4661BA82252...

left join

```

1  update tgt
2    SET
3      tgt.customer_id=src.customer_id,
4      tgt.loan_id=src.loan_id,
5      tgt.full_name=src.full_name,
6      tgt.full_address=src.full_address,
7
8      tgt.loan_amount=src.loan_amount,
9      tgt.hashkey=src.src_hashkey,
10     updateddate=getdate(),
11     updatedby='fabric-updated'
12   FROM [NEW_TARGET_TABLE] tgt
13   join (
14     SELECT *,
15       CONVERT(VARBINARY(32),
16           HASHBYTES('SHA2_256',
17           CONCAT(
18             CAST(customer_id AS VARCHAR),
19             CAST(loan_id AS VARCHAR),
20             CAST(loan amount AS VARCHAR),
21
22
23             CAST(customer_id AS VARCHAR),
24             CAST(loan_id AS VARCHAR),
25             CAST(loan_amount AS VARCHAR),
26             full_name,
27             full_address
28           )))
29         )AS src_hashkey
30   FROM [bronze_lakehouse].[dbo].[3_table_Merge_cust_loans]) AS src
31   on tgt.customer_id=src.customer_id
32   where tgt.hashkey <> src.src_hashkey
33     OR tgt.hashkey IS NULL;
34

```

```

1  INSERT INTO NEW_TARGET_TABLE
2  (customer_id, loan_id, full_name, full_address, loan_amount, hashkey, createdby, createddate, updatedby, updateddate)
3  SELECT customer_id, loan_id, full_name, full_address,loan_amount,
4  CONVERT(VARBINARY(32),
5      HASHBYTES('SHA2_256',
6      CONCAT(
7        CAST(customer_id AS VARCHAR),
8        CAST(loan_id AS VARCHAR),
9        CAST(loan_amount AS VARCHAR),
10       full_name,
11       full_address
12     )));
13   ),
14   'fabric', GETDATE(), 'fabric', GETDATE()
15   FROM bronze_lakehouse.dbo.[3_table_Merge_cust_loans] src
16   WHERE NOT EXISTS (
17     SELECT 1 FROM NEW_TARGET_TABLE tgt
18     WHERE tgt.customer_id = src.customer_id
19   );
20

```

UPDATE SOURCE TABLE.

```

1  SELECT TOP (100) [customer_id],
2      [loan_id],
3      [full_name],
4      [full_address],
5      [payment_id],
6      [balance_loan_amount]
7  FROM [dbo].[3_table_Merge_cust_loans1]

1] ✓ - Command executed using warehouse in 1 sec by NANDINI on 2:33:54 PM, 8/31/25

```

```

SELECT TOP (100) [customer_id],
[loan_id],
[full_name],
[full_address],
[loan_amount]
FROM [warehouse].[dbo].[3_table_Merge_cust_loans1]
insert into [warehouse].[dbo].[3_table_Merge_cust_loans1] values(88,74,'nandini','oj123456',2100),(89,20,'xyxsdfc','cvf564789')
update [warehouse].[dbo].[3_table_Merge_cust_loans1] set full_name='angellia' where loan_id=82
update [warehouse].[dbo].[3_table_Merge_cust_loans1] set full_name='monika' where loan_id=1

```

AFTER RUN ALL THE CODE ,WE GET UPDATED ROWS AND NEW INSERT ROWS IN A TABLE.

Olivia Gibson	8585 Elm St New...	32	13350	fabric	2025-08-31 04:3...	fabric	2025-08-31 04:3...	0xC6437E79...
William McDo...	N/A	52	12350	fabric	2025-08-31 04:3...	fabric	2025-08-31 04:3...	0xB159CDA9...
joy smith	123 Elm St Toron...	17	26600	fabric	2025-08-31 04:3...	fabric-updated	2025-08-31 18:4...	0x0E07CC0B...
Joshua Bennett	4800 Spruce Ln ...	27	26100	fabric	2025-08-31 04:3...	fabric-updated	2025-08-31 18:4...	0x86F35E101...

Name	Type *	Value	Treat as null
error_message	String	Value	<input checked="" type="checkbox"/>
execution_details	String	@pipeline().TriggerTime	
increment_column	String	@activity('Notebook1').output	
Ipv	String	@activity('Notebook1').output	
pipeline_name	String	@pipeline().Pipeline	
pipeline_runid	String	@pipeline().RunID	
pipeline_status	String	successful	<input type="checkbox"/>
schemaname	String	dbo	<input type="checkbox"/>
tablename	String	3_table_Merge_cust_lo...	<input type="checkbox"/>

General Settings

Connection * warehouse **Stored procedure name *** [dbo].[log_proc]

Stored procedure parameters

Import **New** **Delete**

<input type="checkbox"/> Name	Type *	Value
error_message	String	<input type="text"/> Value <input checked="" type="checkbox"/> Treat as null
execution_details	String	<input type="text"/> @pipeline().TriggerTime
increment_column	String	<input type="text"/> @activity('Notebook1').output

Run

Validate **Run** **Schedule** **Trigger** **View run history**

Notebook **Stored procedure**

General Settings

Connection * 9 Nandini **Refresh** **Edit**

Relative URL **Method *** POST

Body {"pipeline_name": "@{pipeline().Pipe..."}

Headers *

Activity outputs

```
{
  "pipeline_name": "@{pipeline().Pipeline}",
  "error_message": "@{activity('Notebook1').error}",
  "execution_details": "@{pipeline().TriggerTime}",
  "pipeline_runid": "@{pipeline().RunId}"
}
```

Parameters

System variables

Trigger parameters

Clear contents

Search

Notebook1
Notebook1 activity output

Stored procedure1

OK **Cancel**

Explore

SQL query 3 **stored procedure** **storedprocedur...**

Run **Save as view** **Explain query** **Fix query errors**

```

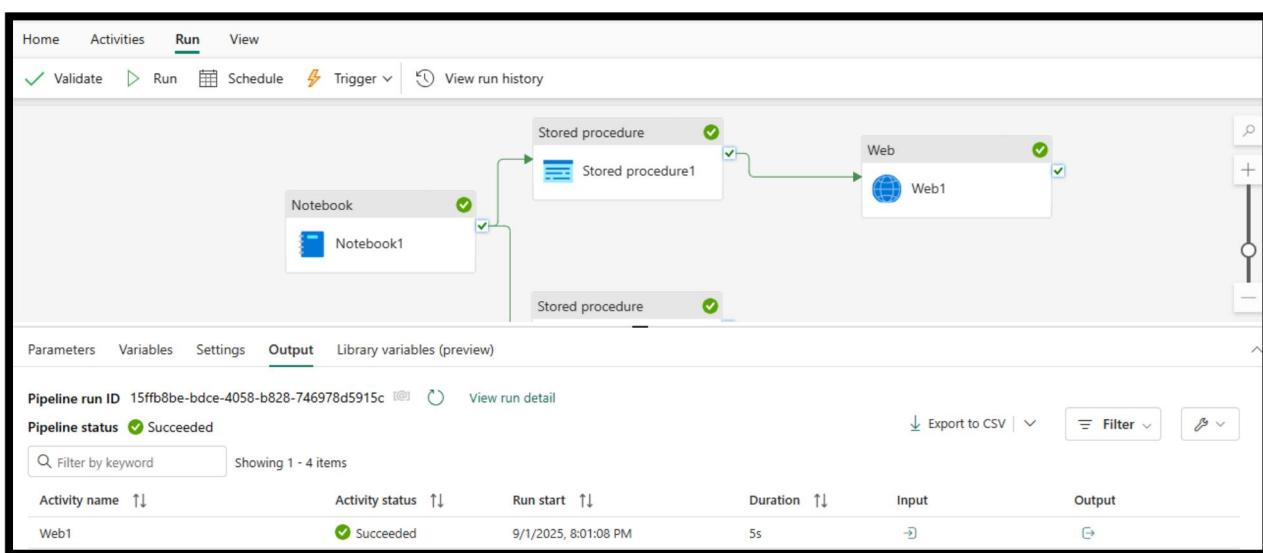
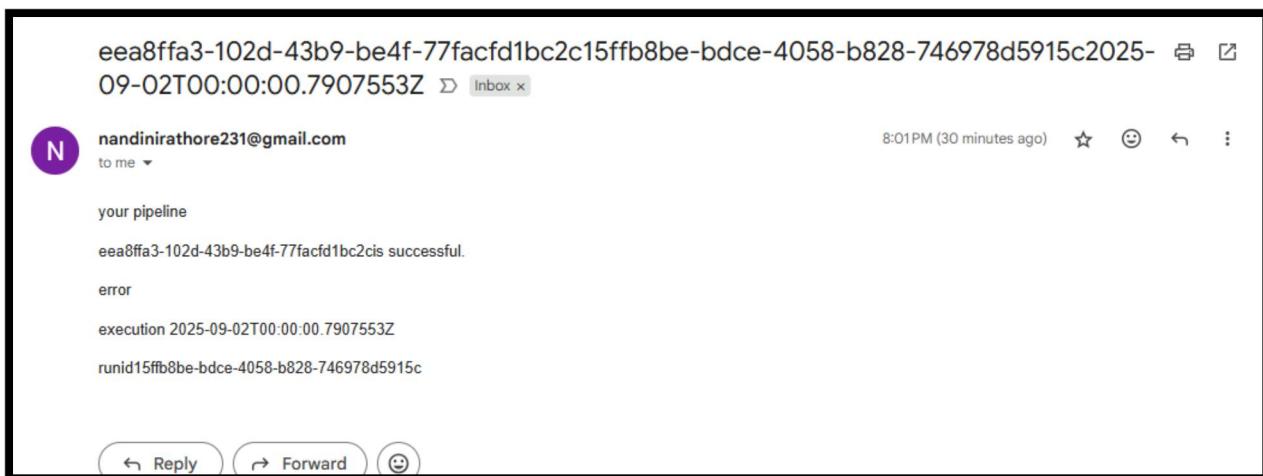
21  @pipeline_runid,
22  @lpv,
23  @increment_column,
24  @execution_details
25  )
26  END
27
28  select * from logging_table

```

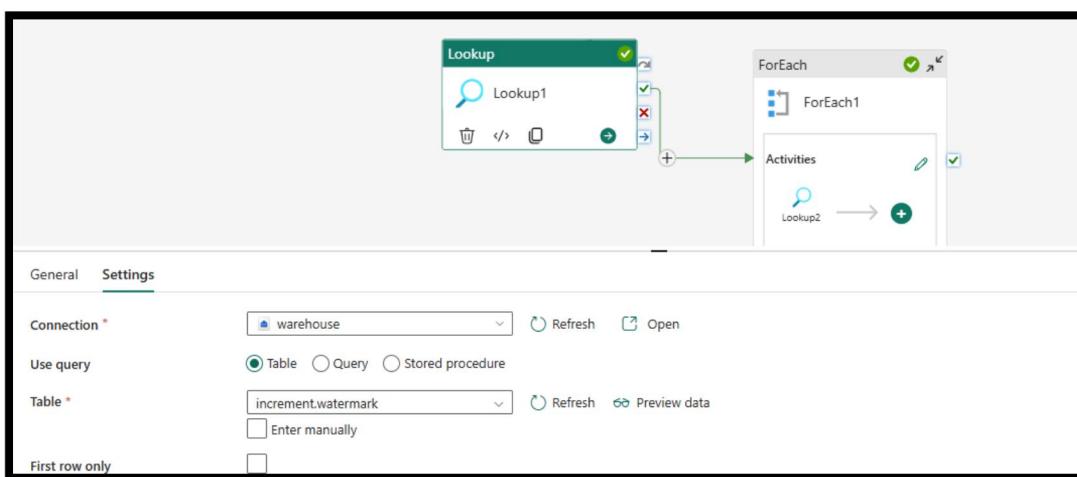
Messages Results

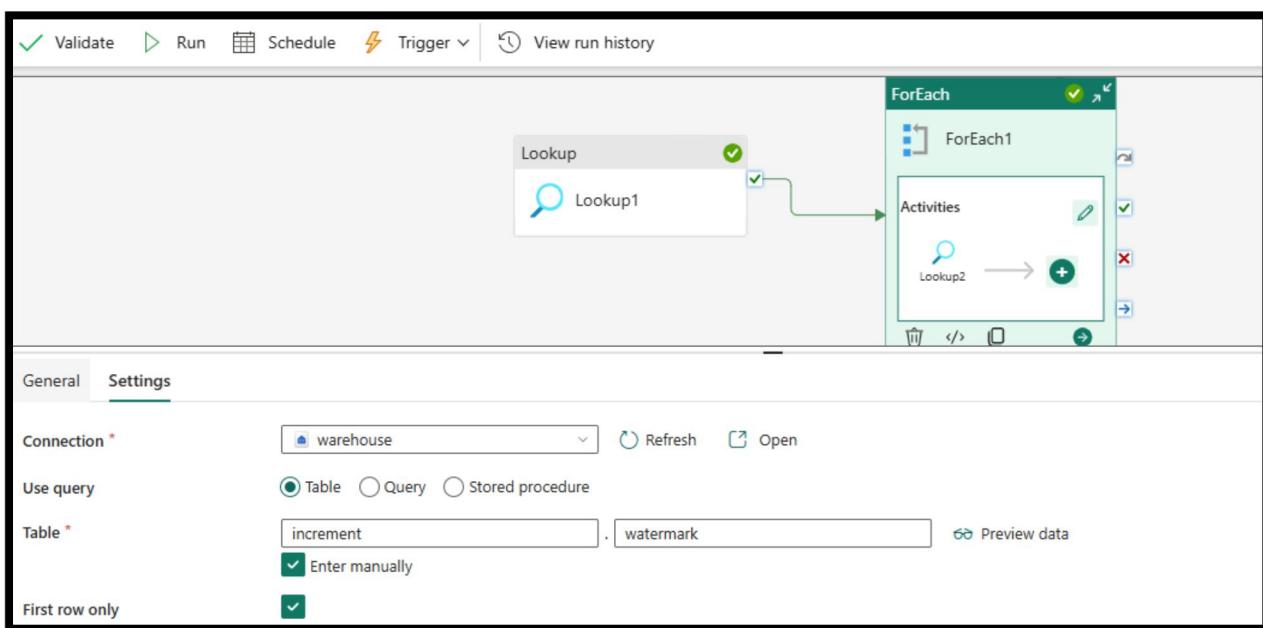
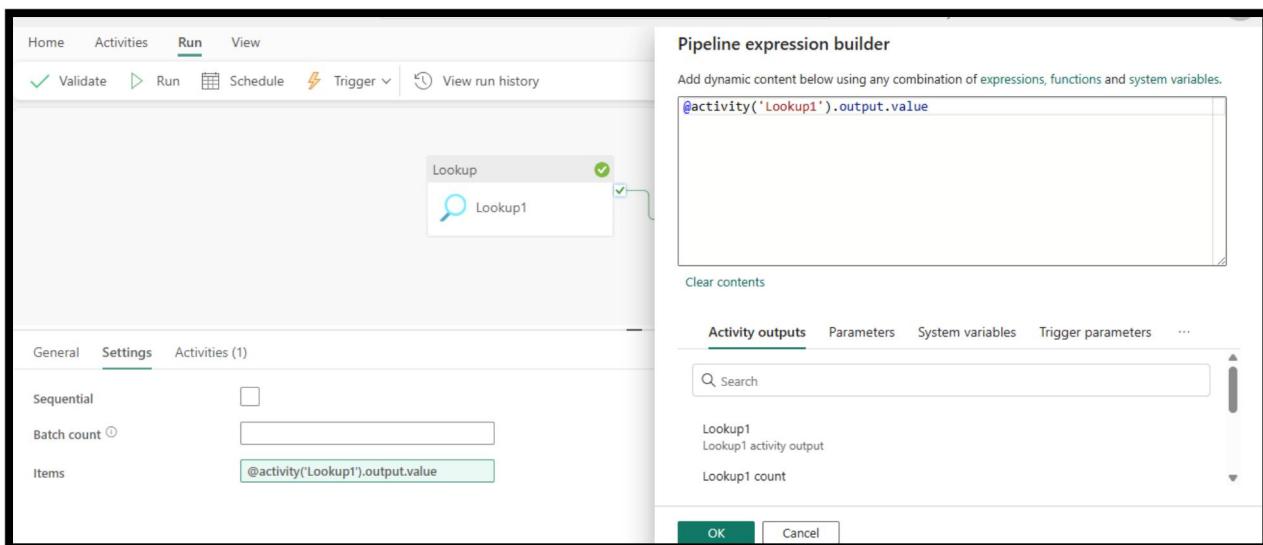
#	ABC tablename	ABC schemaname	ABC error_message	ABC pipeline_status	ABC pipeline_name	ABC pipeline_runid	ABC lpv	ABC increment_colu...	ABC execution_deta...
1	3_table_Merge_cust...	dbo	NULL	successful	eea8ffa3-102d-43b...	15ffb8be-bdce-405...	System.Col	System.Collections....	2025-09-02T00:00:...
2	3_table_Merge_cust...	dbo	NULL	successful	eea8ffa3-102d-43b...	15ffb8be-bdce-405...	System.Col	System.Collections....	2025-09-02T00:00:...
3	table4_Merge_accep...	dbo	NULL	successful	7e607389-917a-46...	9f91cc1b-bf87-455...	System.Col	System.Collections....	pipeline().TriggerTi...

NANDINI RATHORE



Incremental loading check max value of incremental_column.





The screenshot shows the 'Pipeline status' page. The top navigation bar includes 'Parameters', 'Variables', 'Settings', 'Output' (selected), and 'Library variables (preview)'. Below the navigation, it says 'Pipeline status' with a green checkmark and 'Succeeded'. A search bar with 'Filter by keyword' and a message 'Showing 4 items' are present. A table lists four activities: 'Lookup1' (Status: Succeeded, Run start: 9/1/2025, 11:33:45 PM, Duration: 6s), 'ForEach1' (Status: Succeeded, Run start: 9/1/2025, 11:33:52 PM, Duration: 9s), 'Lookup2' (Status: Succeeded, Run start: 9/1/2025, 11:33:52 PM, Duration: 6s), and another 'Lookup2' (Status: Succeeded, Run start: 9/1/2025, 11:33:52 PM, Duration: 6s). To the right of the table, an 'Output' pane displays a JSON object:

```
{
  "firstRow": {
    "tablename": "table4_Merge_accpayloan",
    "schemaname": "dbo",
    "Ipv": 85,
    "increment_column": "customer_id"
  }
}
```

NANDINI RATHORE

```
6 select max(customer_id) as maxvalues from [dbo].[3_table_Merge_cust_loans1] WHERE customer_id>85
```

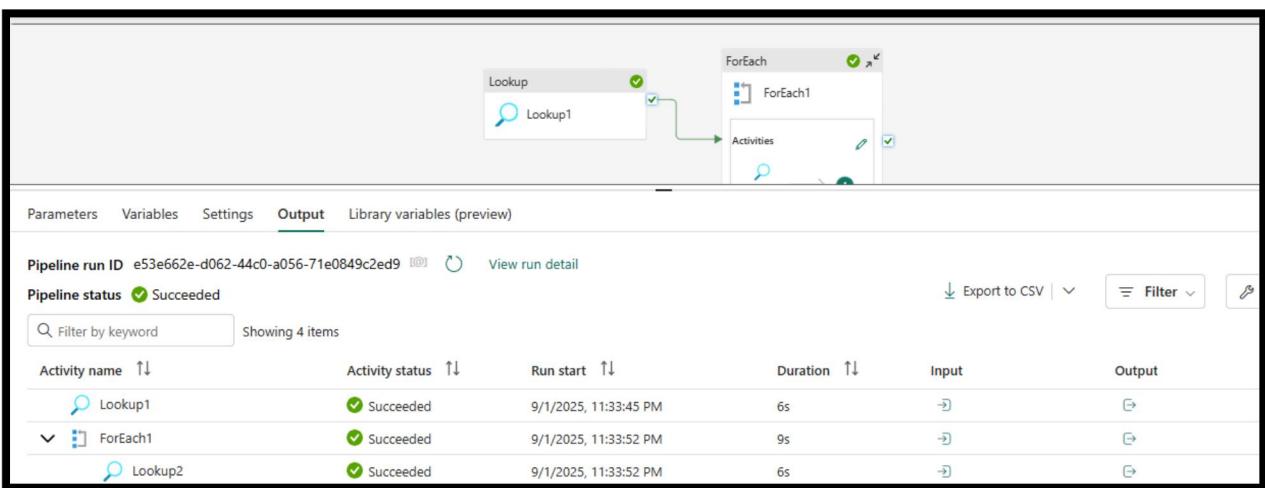
Messages Results 00 Search

	12F maxvalues
1	89

```
1 SELECT TOP (100) [customer_id],  
2 [loan_id],  
3 [full_name],  
4 [full_address],  
5 [loan_amount]  
6 FROM [warehouse].[dbo].[3_table_Merge_cust_loans1]  
7 insert into [warehouse].[dbo].[3_table_Merge_cust_loans1] values(90,25,'raghav','5874 bhggfrt',22)
```

Messages Results 00 Search

	12F customer_id	12F loan_id	ABC full_name	ABC full_address	12F loan_amount
1	90	25	raghav	5874 bhggfrt	22

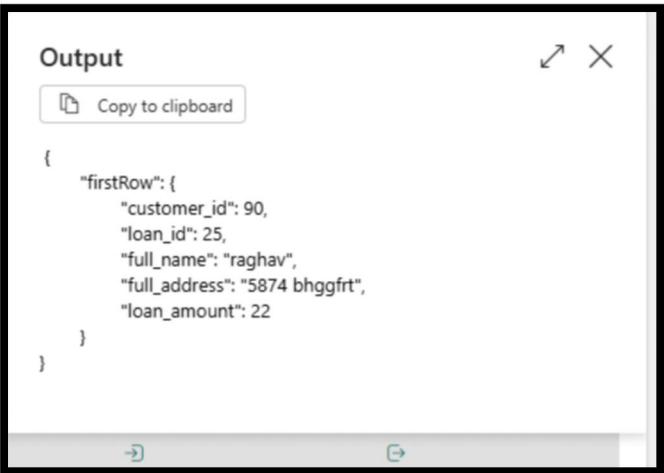


After increment loading

```
6 select max(customer_id) as maxvalues from [dbo].[3_table_Merge_cust_loans1] WHERE customer_id>85  
7  
8
```

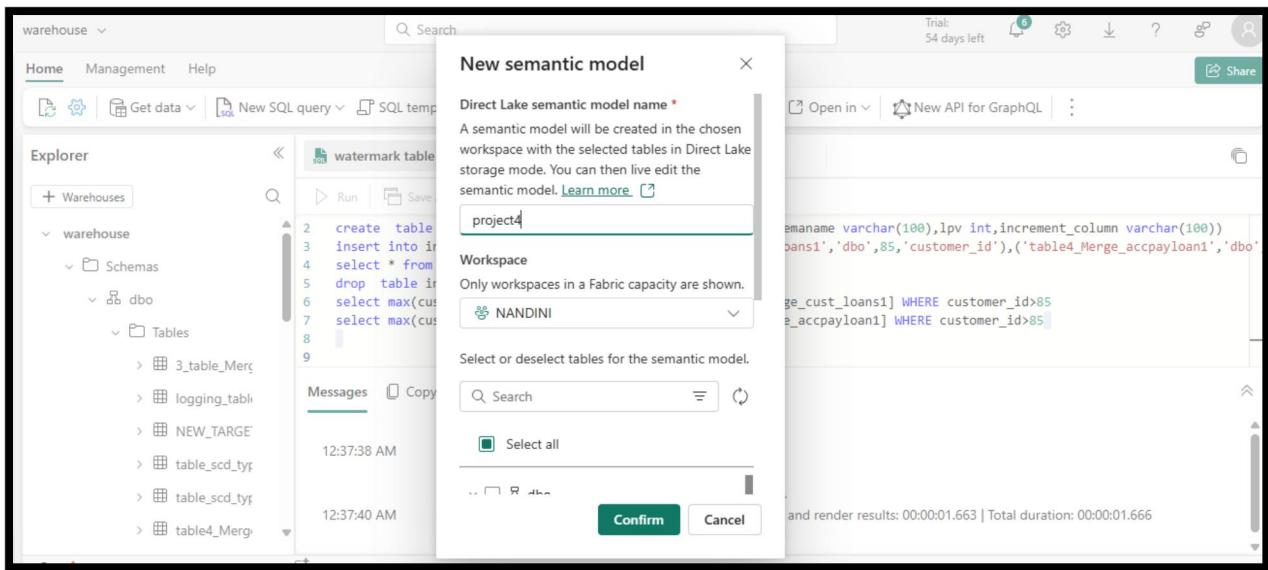
Messages Results 00 Search

	12F maxvalues
1	90



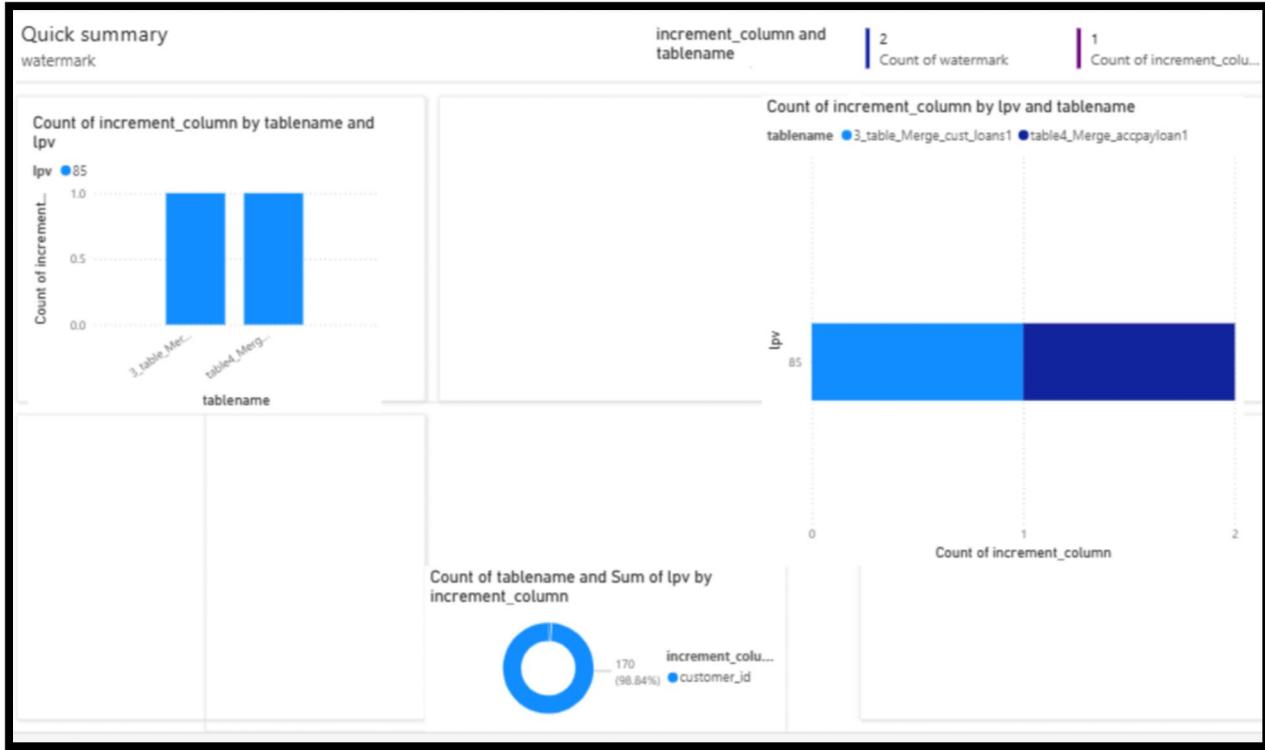
The screenshot shows a browser window with the title 'Output'. It contains a JSON object:

```
{ "firstRow": { "customer_id": 90, "loan_id": 25, "full_name": "raghav", "full_address": "5874 bhggfrt", "loan_amount": 22 } }
```



The screenshot shows the Snowflake UI interface. On the left, the 'Explorer' sidebar shows a warehouse named 'warehouse' with a schema 'dbo' containing tables like '3_table_Merg', 'logging_table', 'NEW_TARG', 'table_scd_ty', 'table_scd_ty', and 'table4_Merg'. In the center, a modal dialog titled 'New semantic model' is open. The 'Semantic model name' field is filled with 'project4'. Below it, a message states: 'A semantic model will be created in the chosen workspace with the selected tables in Direct Lake storage mode. You can then live edit the semantic model.' A 'Learn more' link is provided. To the right of the modal, the main workspace shows some SQL code and its execution results.

SAVE IT AS SEMANTIC MODEL.OPEN POWER BI,SELECT PROJECT4 THE SEMANTIC MODEL WHICH IS SAVED AND VIEW ITS REPORT.



Report in Power BI.

PROJECT-03

Design a dynamic pipeline that facilitates the workflow

NANDINI RATHORE

INDEX

1. Extract data from 5 tables from an on-premises SQL Server.....**2-8**
2. Load the extracted data into Azure Data Lake Storage Gen2 in a structured format.....**8-17**
3. Copy the data from ADLS Gen2 back to a designated file system folder on the local C: drive...**18-27**
4. Implement a metadata-driven approach using a control table that stores details such as:.....**27-32**
1. Source table names
2. Target file paths
5. Implement the email configuration task by using logic apps and send an email for any failure...**32-38**

Design a dynamic pipeline that facilitates the following workflow:

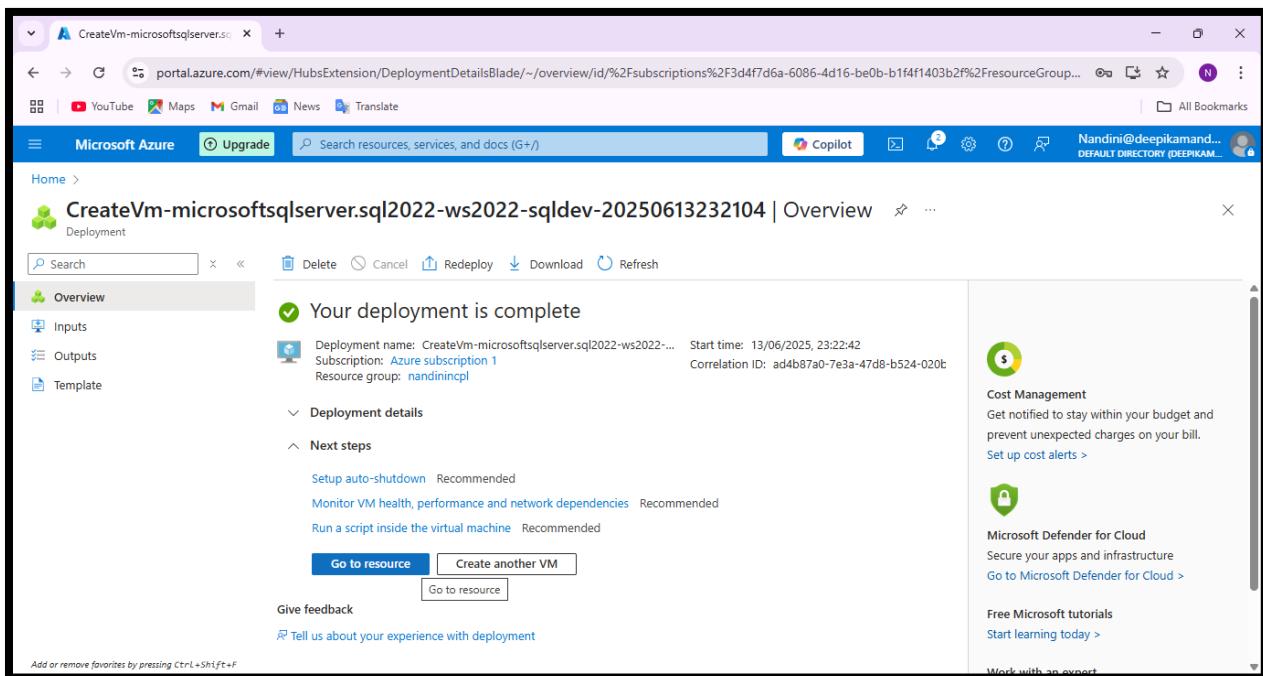
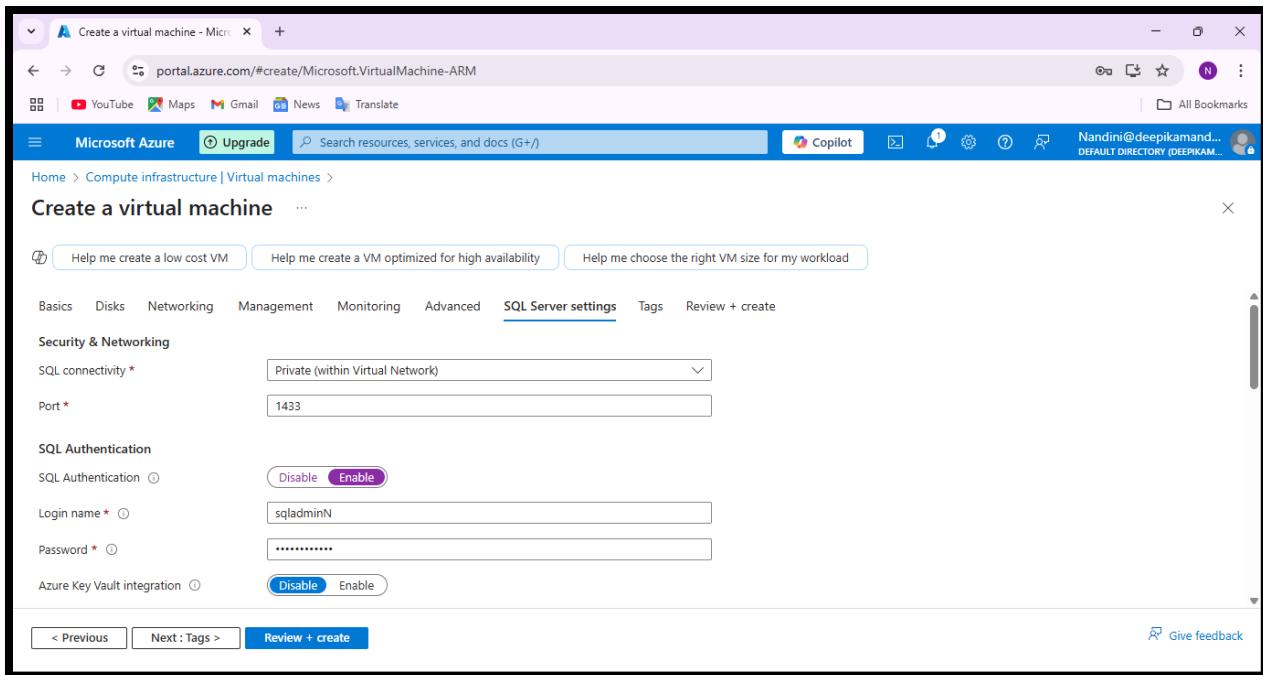
1. Extract data from 5 tables from an on-premises SQL Server.

The screenshot shows the Microsoft Azure portal interface. The user is on the 'Compute infrastructure | Virtual machines' page. The left sidebar has 'Virtual machines' selected. The main content area says 'No virtual machines to display' and includes a 'Create' button. There are also filter options and a note about viewing a new version of the experience.

1. Create VM.

The screenshot shows the 'Create a virtual machine' wizard. The user is on step 1, 'Adding details'. The form includes fields for Subscription (selected: Azure subscription 1), Resource group (selected: nandinincpl), Virtual machine name (VMsqlN), Region (selected: (US) West US 2), Availability options (selected: No infrastructure redundancy required), Security type (selected: Standard), and Image (selected: Free SQL Server License: SQL Server 2022 Developer on Windows Server 2022). At the bottom, there are navigation buttons (< Previous, Next : Disks >) and a 'Review + create' button.

2. ADDING DETAILS.



3. GO TO RESOURCES.

VMsqliN | Overview

Connect

Resource group (move) : nandinincpl

Status : Running

Location : West US 2

Subscription (move) : Azure subscription 1

Subscription ID : 3d4f7d6a-6086-4d16-be0b-b1f4f1403b2f

Operating system : Windows (Windows Server 2022 Datacenter)

Size : Standard D2s v5 (2 vcpus, 4 GiB memory)

Public IP address : 20.114.52.169

Virtual network/subnet : VMsqliN-vnet/default

DNS name : Not configured

Health state : -

Time created : 14/06/2025, 03:22 UTC

Tags (edit) : Add tags

Properties Monitoring Capabilities (8) Recommendations Tutorials

Virtual machine Networking

4.CONNECT VM.

VMsqliN | Connect

Port (change) 3389 Check access

Just-in-time policy Unsupported by plan

Most common

Native RDP Local machine

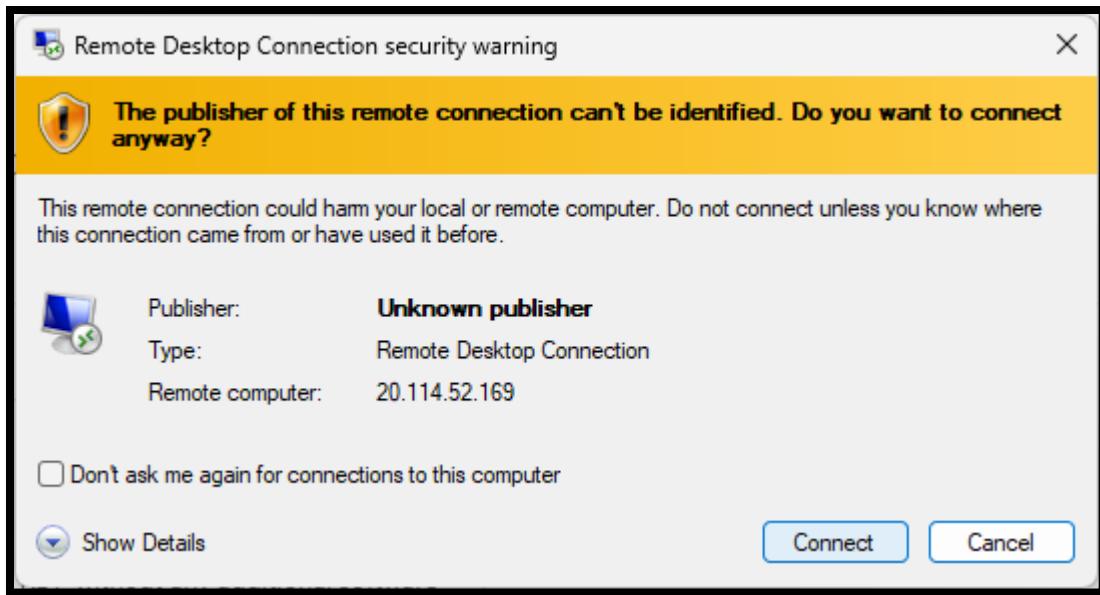
Connect via native RDP without any additional software needed. Recommended for testing only.

Public IP address (20.114.52.169)

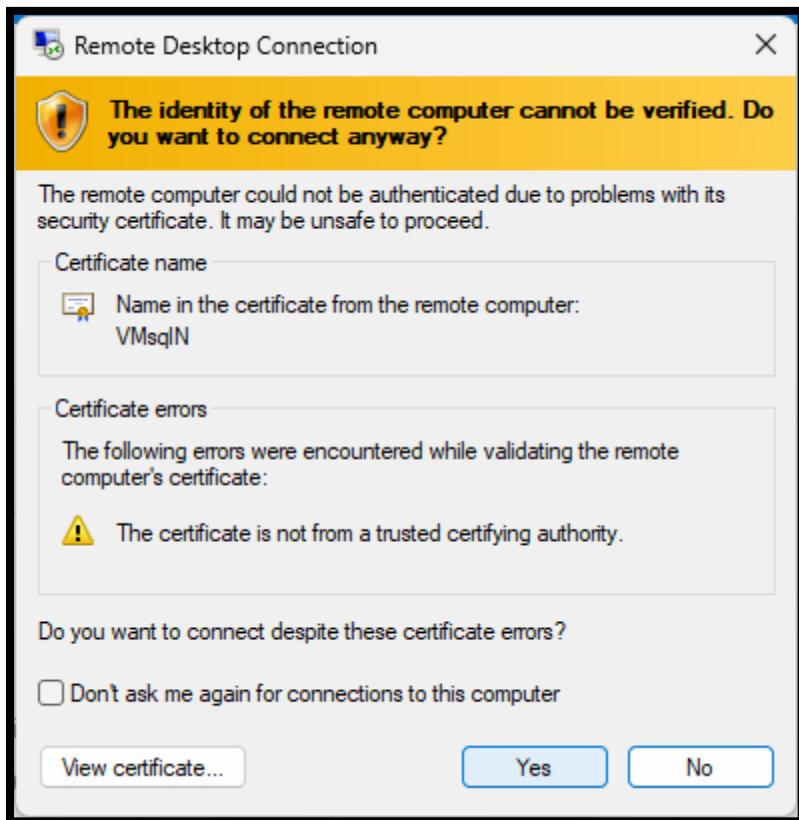
Select Download RDP file

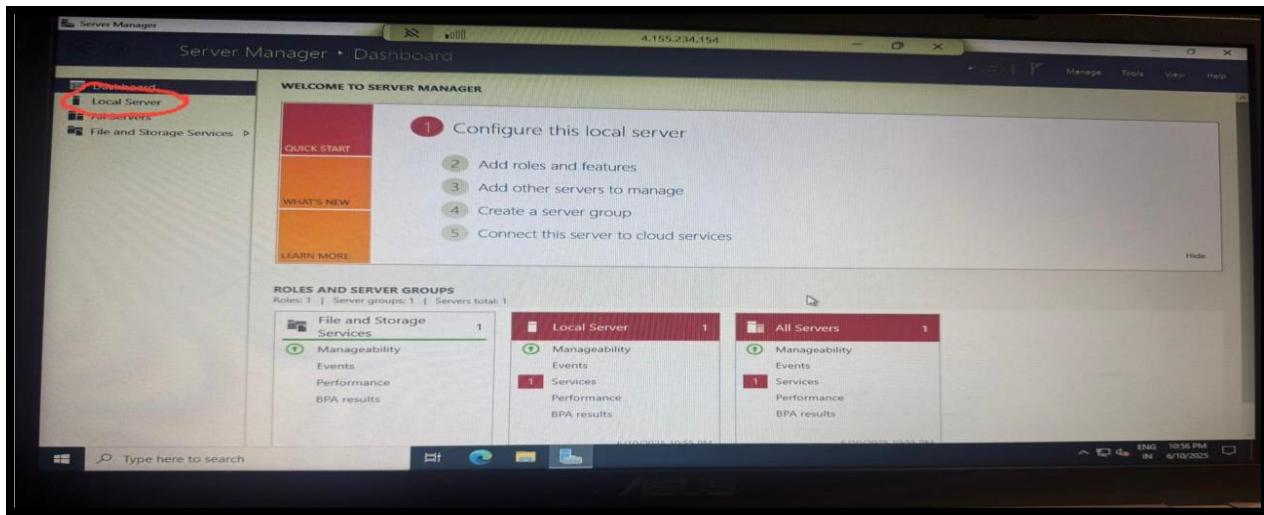
More ways to connect (4)

4. DOWNLOAD RDP FILE.

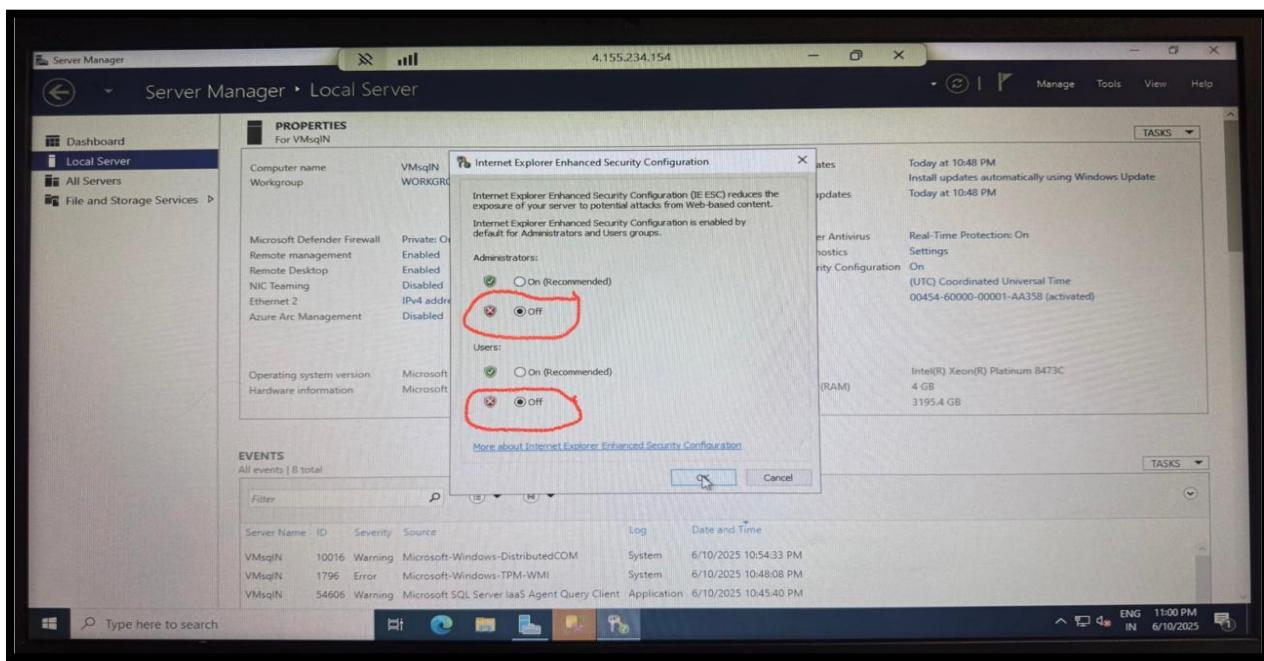


5. CONNECT IT WITH SERVER.

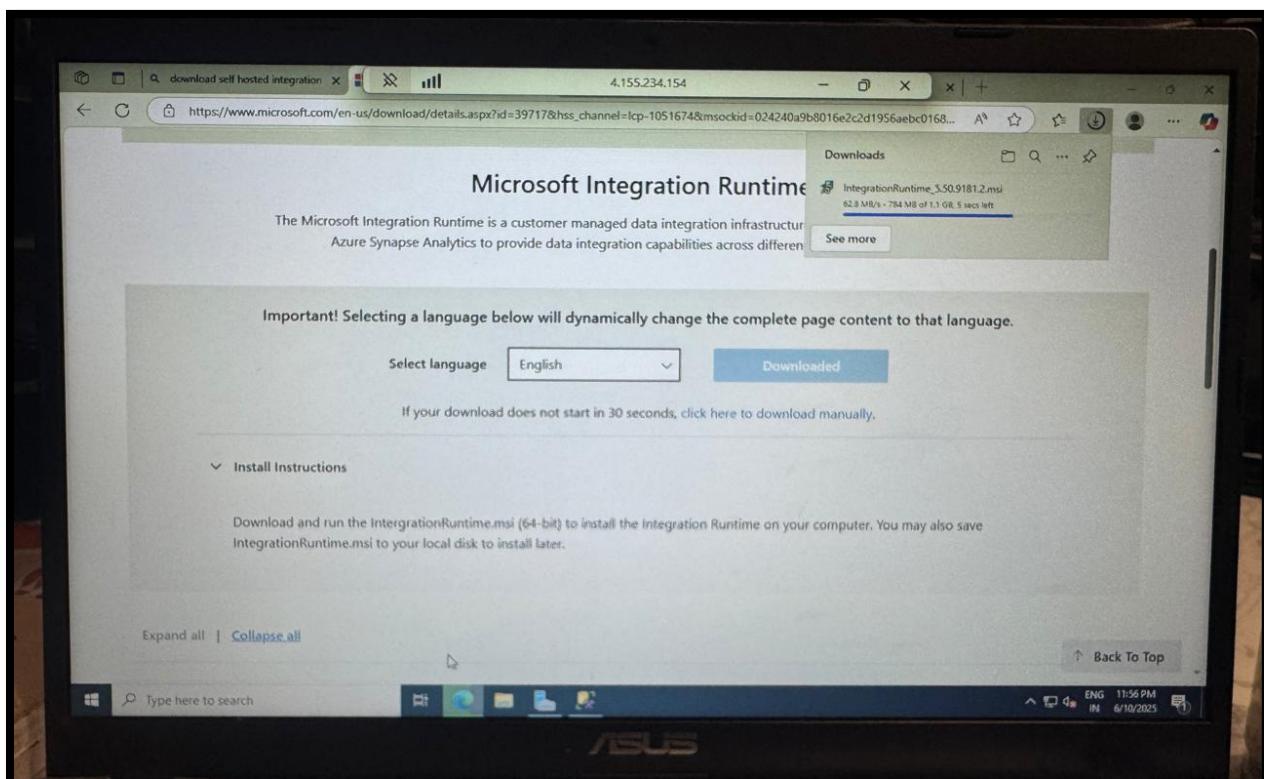
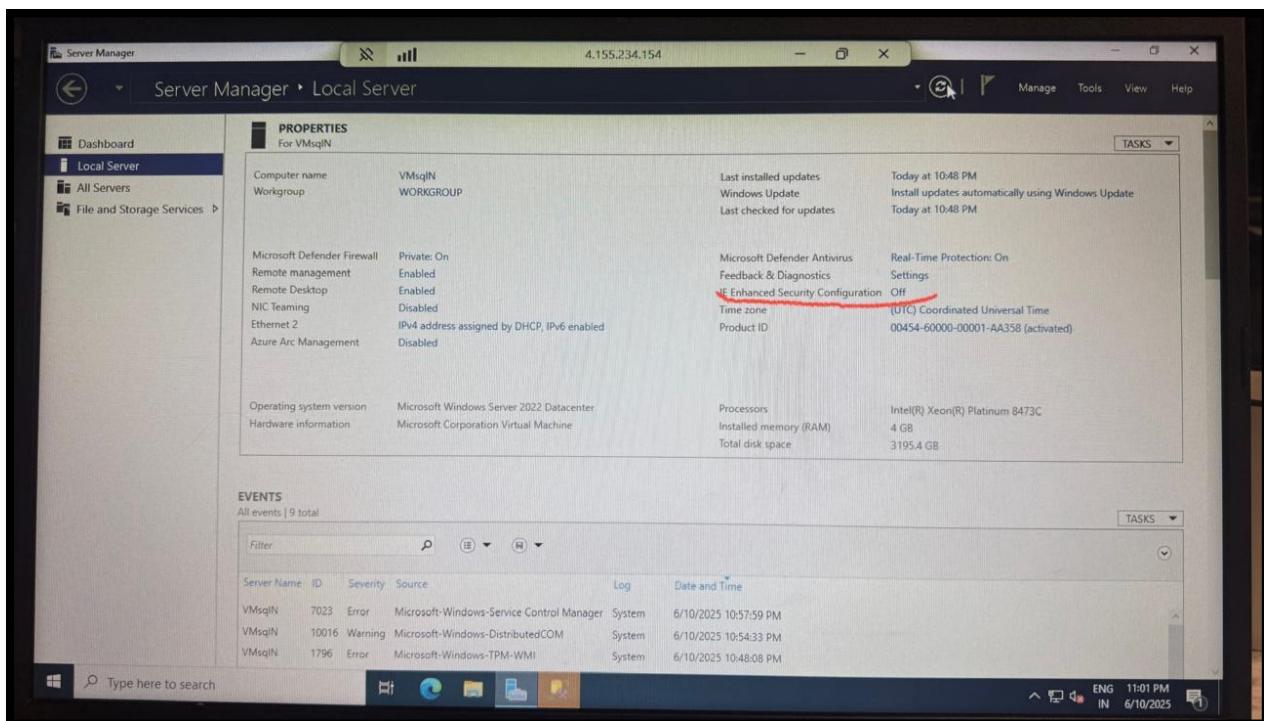




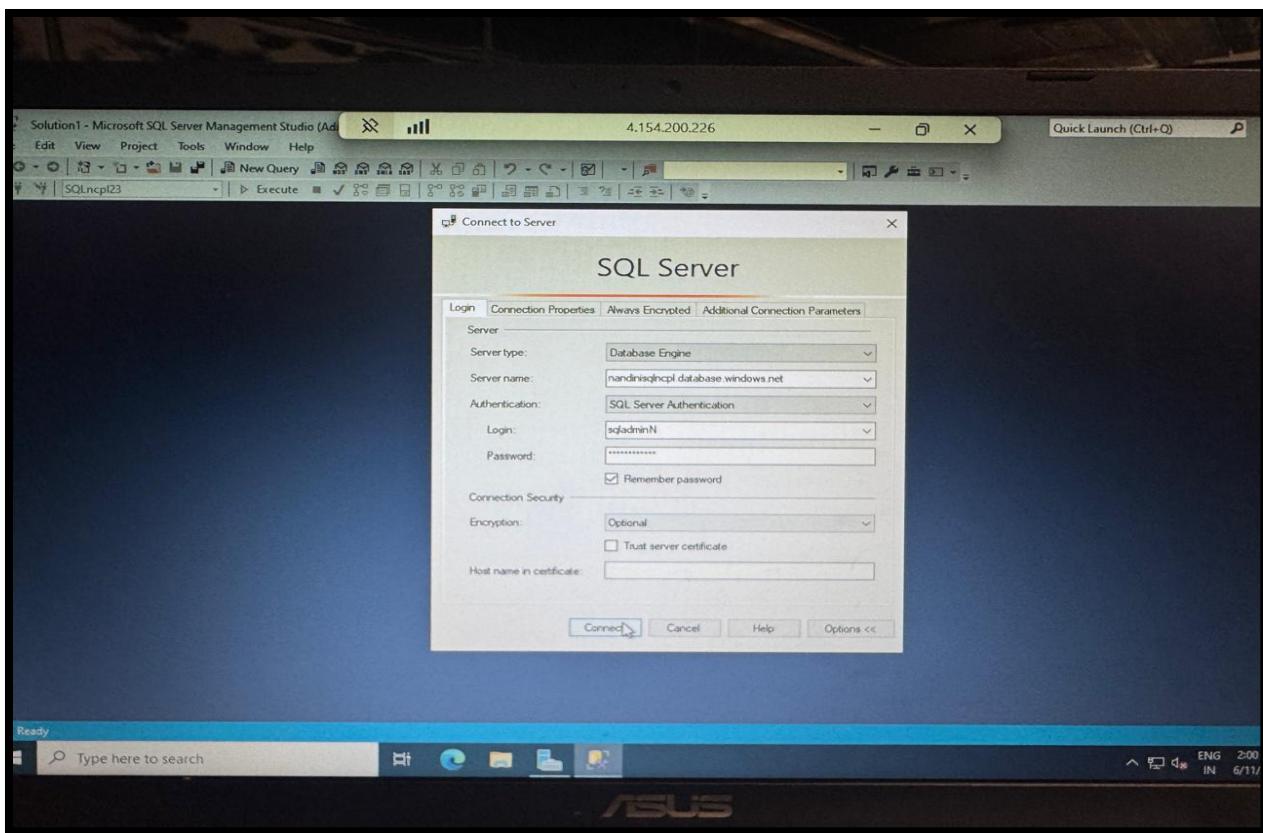
6.AFTER CONNECT WITH VM THIS SCREEN APPEARED ,SELECT LOCAL SERVER.



7.NOW SELECT 'OFF' FOR BOTH ADMINISTRATOR AND USERS.



7.DOWNLOAD SELF-HOSTED IR.



8. OPEN SSMS ,CONNECT IT WITH SQL SERVER.

```

ALTER TABLE [school].[metadata2] ADD targetlocalfilepath nvarchar(200);
UPDATE school.metadata2 SET targetlocalfilepath = 'c:\school\students2\students2.csv' WHERE tablename = 'students2';
UPDATE school.metadata2 SET targetlocalfilepath = 'c:\school\teachers2\teachers2.csv' WHERE tablename = 'teachers2';
UPDATE school.metadata2 SET targetlocalfilepath = 'c:\school\enrollment2\enrollment2.csv' WHERE tablename = 'enrollment2';
UPDATE school.metadata2 SET targetlocalfilepath = 'c:\school\attendance3\attendance3.csv' WHERE tablename = 'attendance3';
UPDATE school.metadata2 SET targetlocalfilepath = 'c:\school\fee\fee.csv' WHERE tablename = 'fee';

```

	id	tablename	schemaname	foldename	filename	TARGETFILEPATH	targetlocalfilepath
1	1	STUDENTS2	SCHOOL	SCHOOL\STUDENTS2	STUDENTS2.CSV	/RAW/SCHOOL/STUDENTS2.CSV	c:\school\students2\students2.csv
2	2	TEACHERS2	SCHOOL	SCHOOL\TEACHERS2	TEACHERS2.CSV	/RAW/SCHOOL/TEACHERS2.CSV	c:\school\teachers2\teachers2.csv
3	4	ENROLLMENT2	SCHOOL	SCHOOL\ENROLLMENT2	ENROLLMENT2.CSV	/RAW/SCHOOL/ENROLLMENT2.CSV	c:\school\enrollment2\enrollment2.csv
4	4	ATTENDANCE3	SCHOOL	SCHOOL\ATTENDANCE3	ATTENDANCE3.CSV	/RAW/SCHOOL/ATTENDANCE3.CSV	c:\school\attendance3\attendance3.csv
5	5	FEE	SCHOOL	SCHOOL\FEE	FEEL.CSV	/RAW/SCHOOL/FEE.CSV	c:\school\fee\fee.csv

9. CREATE A METADATA TABLE AND ALSO ADD FILE PATH FOR BOTH CLOUD AND ON-PREMISE .

2. Load the extracted data into Azure Data Lake Storage Gen2 in a structured format.

The screenshot shows the Microsoft Azure Synapse Analytics interface. The left sidebar has a tree view with 'Integration' expanded, showing 'Triggers', 'Integration runtimes' (which is selected and highlighted in grey), 'Security', 'Access control', and 'Credentials'. The main content area is titled 'Integration runtimes' and contains the following information:

- Description: The integration runtime (IR) is the compute infrastructure to provide the following data integration capabilities across different network environment.
- Buttons: '+ New' and 'Refresh'.
- Filter: 'Filter by name'.
- Table header: Name, Type, Sub-type, Status, Related, Region, Version.
- Table data (5 items):

Name	Type	Sub-type	Status	Related	Region	Version
AutoResolveIntegrationRuntime	Azure	Public	Running	11	Auto Resolve	---
IntegrationRuntime2310	Self-Hosted	---	Unavail...	More 1	---	5.50.9181.2
IntegrationRuntime2311	Self-Hosted	---	Unavail...	More 0	---	---
SELFHOSTED-IR	Self-Hosted	---	Unavail...	More 0	---	5.50.9181.2
selfhostedintegrationruntime0506	Self-Hosted	---	Unavail...	More 3	---	5.50.9181.2

9. NOW CREATE INTEGRATION TUNTIME.

10.SELECT

SELF

HOSTED

The screenshot shows the 'Integration runtime setup' page. The left sidebar is identical to the previous screenshot. The main content area has two large boxes side-by-side:

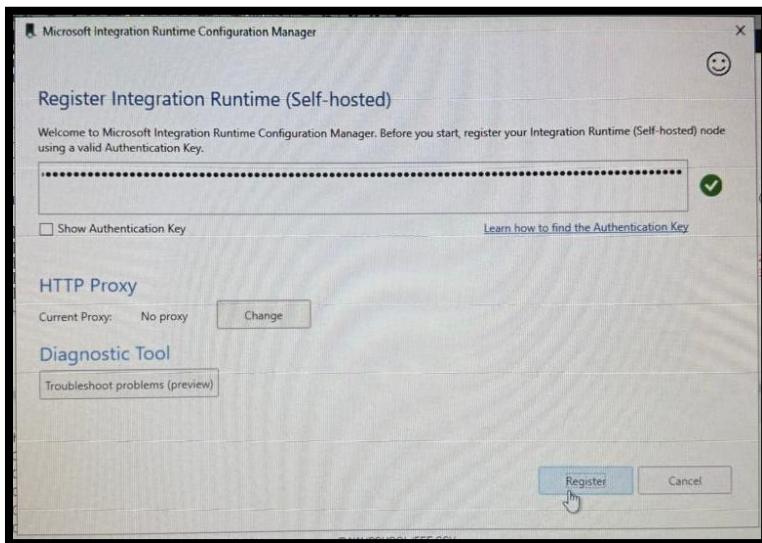
- SELF:** 'Azure, Self-Hosted' icon. Description: Perform data flows, data movement and dispatch activities to external compute.
- HOSTED:** 'Azure-SSIS' icon. Description: Lift-and-shift existing SSIS packages to execute in Azure.

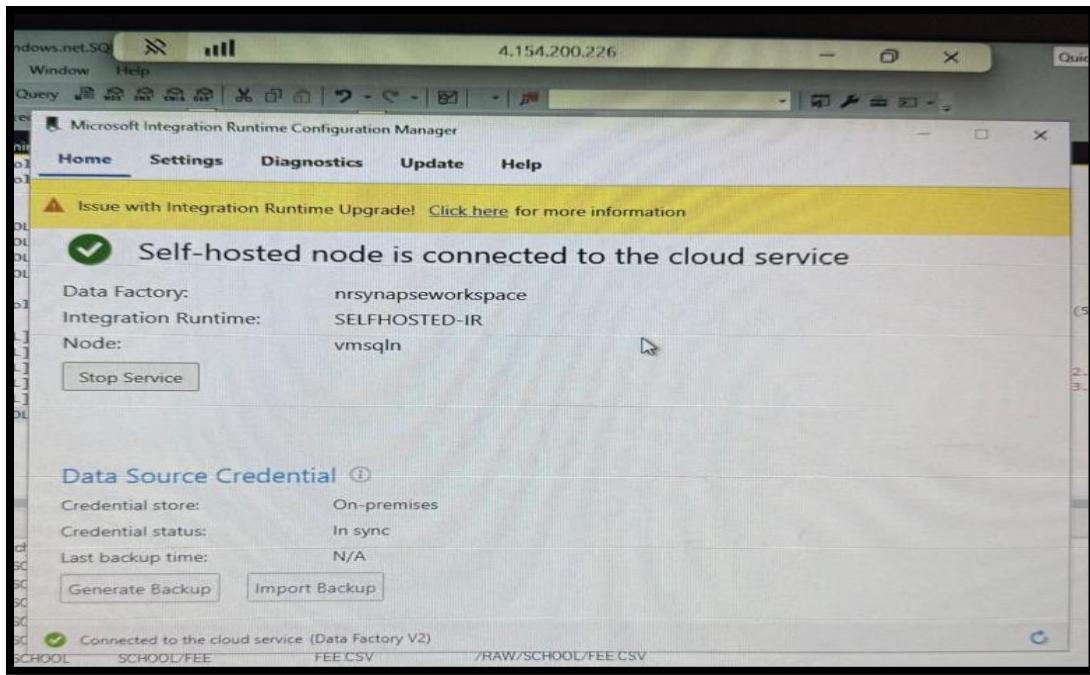
At the bottom right are 'Continue' and 'Cancel' buttons.

The screenshot shows the Microsoft Azure Synapse Analytics Integration runtime setup page. On the left, there's a sidebar with options like Synapse live, Connector upgrade advisor, Analytics pools, SQL pools, Apache Spark pools, Data Explorer pools, External connections, Linked services, Microsoft Purview, Integration, Triggers, and Integration runtimes. The Integration runtimes option is selected. The main area is titled "Integration runtime setup" and has a section for "Network environment". It shows two options: "Azure" (selected) and "Self-Hosted". The "Azure" option is described as "Use this for running data flows, data movement, external and pipeline activities in a fully managed, serverless compute in Azure." The "Self-Hosted" option is described as "Use this for running data movement, external and pipeline activities in an on-premises / private network by installing the integration runtime." A note says "Note: Data flows are only supported on Azure integration runtime. You can use self-hosted integration runtime to stage the data on cloud storage and then use data flows to transform it." At the bottom, there are "Continue", "Back", and "Cancel" buttons.

This screenshot shows the same Microsoft Azure Synapse Analytics Integration runtime setup page, but now the "Self-Hosted" option is selected. The "Name" field is filled with "shis". The "Description" field has the placeholder "Enter description here...". The "Type" field is set to "Self-Hosted". At the bottom, a progress bar indicates "Creating..." and there are "Back" and "Cancel" buttons. The rest of the interface is identical to the previous screenshot, including the sidebar and the Azure environment option.

11. COPY THIS KEY1 URL AND PASTE TO ON PREMISE FOR CONNECTION OF SH-IR.



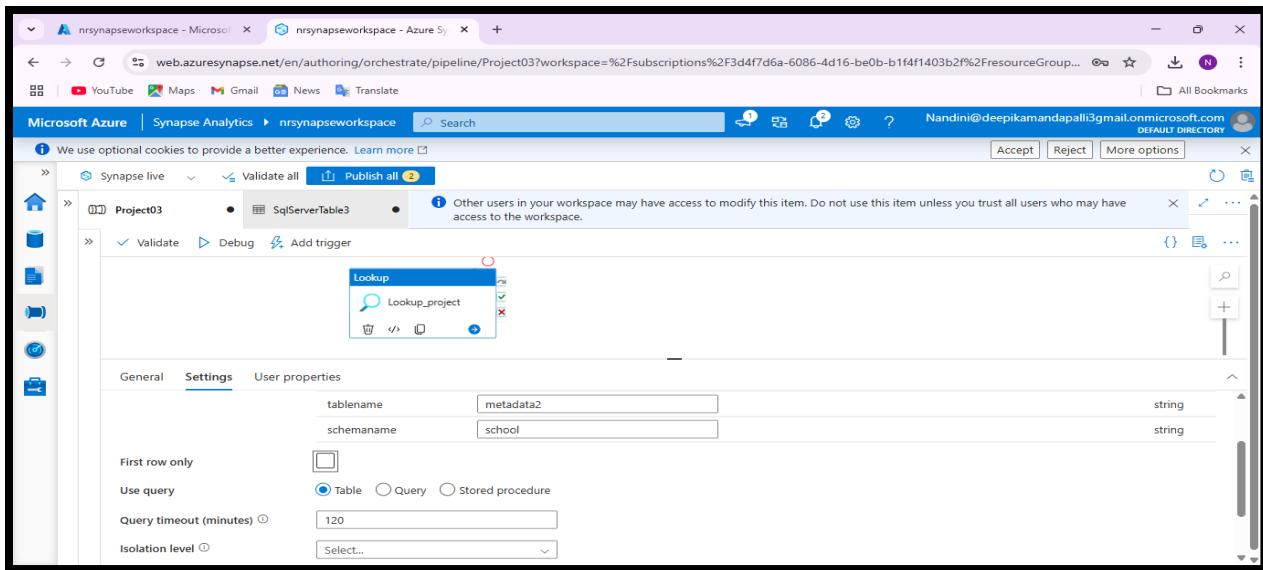


The screenshot shows the Microsoft Azure Synapse Analytics portal. The left sidebar navigation includes options like Synapse live, Validate all, Publish all, Connector upgrade advis..., Analytics pools, SQL pools, Apache Spark pools, Data Explorer pools (prev...), External connections, Linked services, Microsoft Purview, Integration, Triggers, Integration runtimes (which is currently selected), Security, Access control, and Credentials.

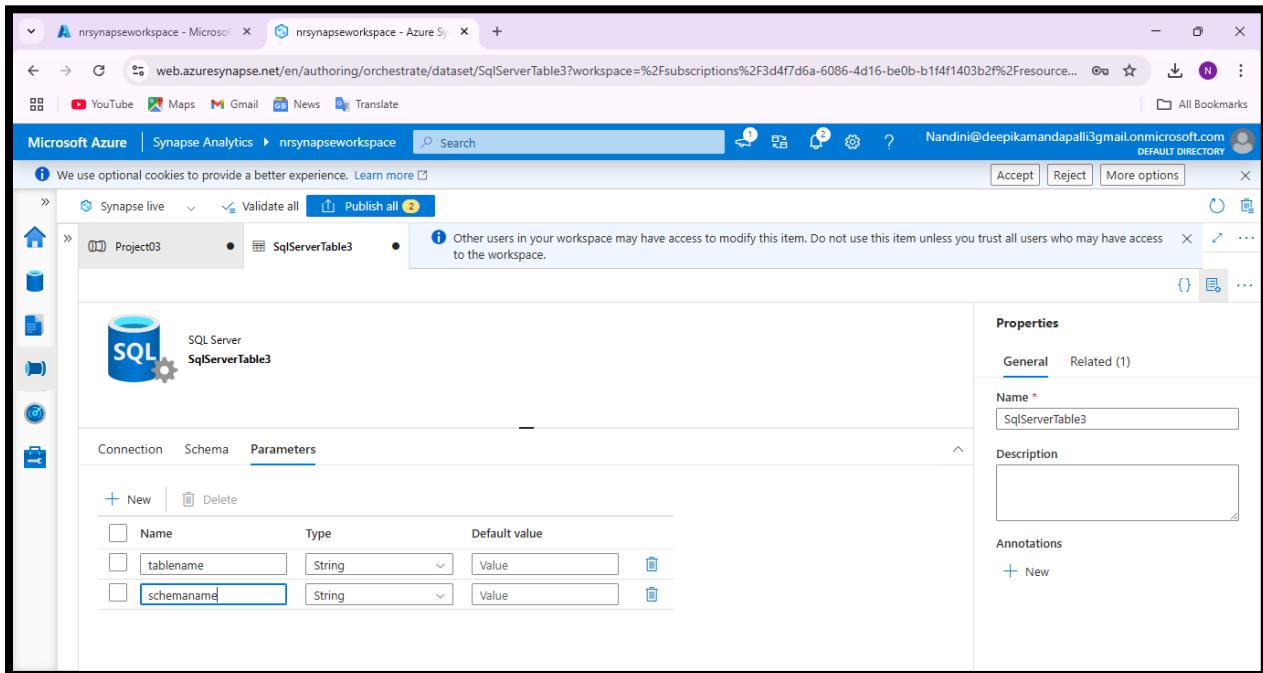
The main content area is titled "Integration runtimes" and describes them as the compute infrastructure for data integration. It includes a "New" button and a "Refresh" button. A "Filter by name" input field is available. The table below lists six items:

Name	Type	Sub-type	Status	Related	Region	Version
AutoResolveIntegrationRuntime	Azure	Public	Running	11	Auto Resolve	---
IntegrationRuntime2310	Self-Hosted	---	Unavail...	1	---	5.50.9181.2
IntegrationRuntime2311	Self-Hosted	---	Unavail...	0	---	---
SELFHOSTED-IR	Self-Hosted	---	Unavail...	0	---	5.50.9181.2
selfhostedintegrationruntime0506	Self-Hosted	---	Unavail...	3	---	5.50.9181.2
shis	Self-Hosted	---	Running	0	---	5.50.9181.2

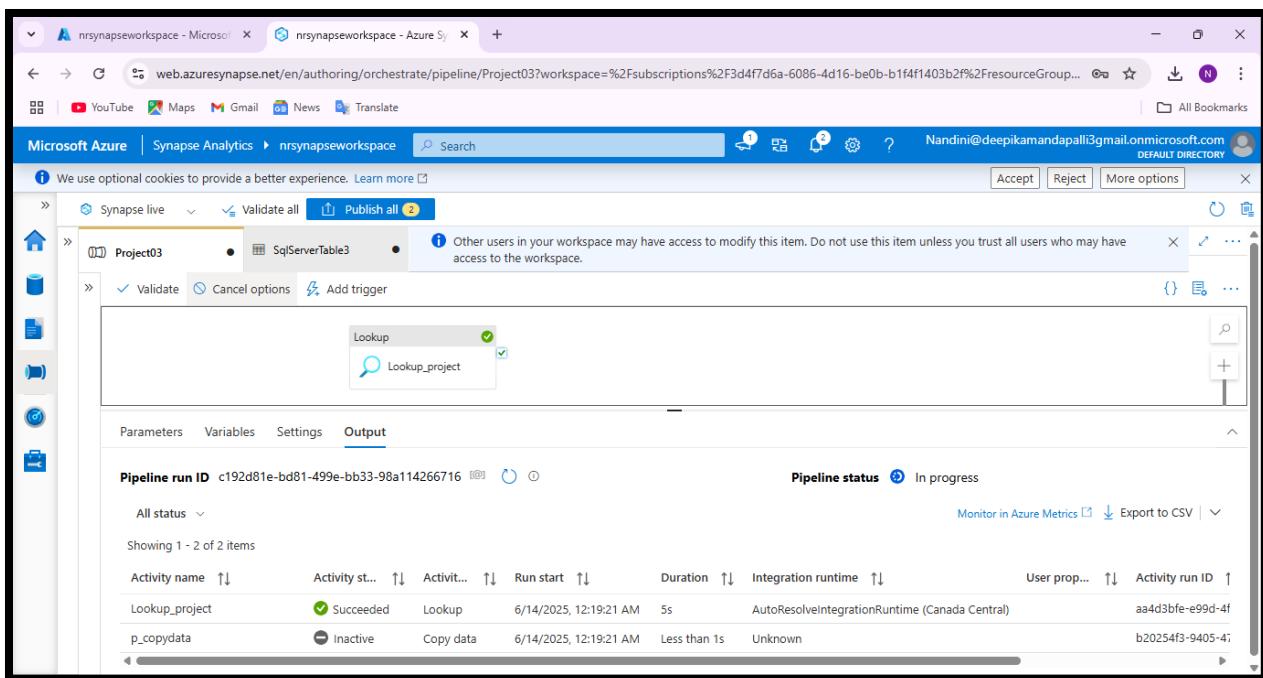
12. SELF HOSTED INTEGRATION RUNTIME IS RUNNING.



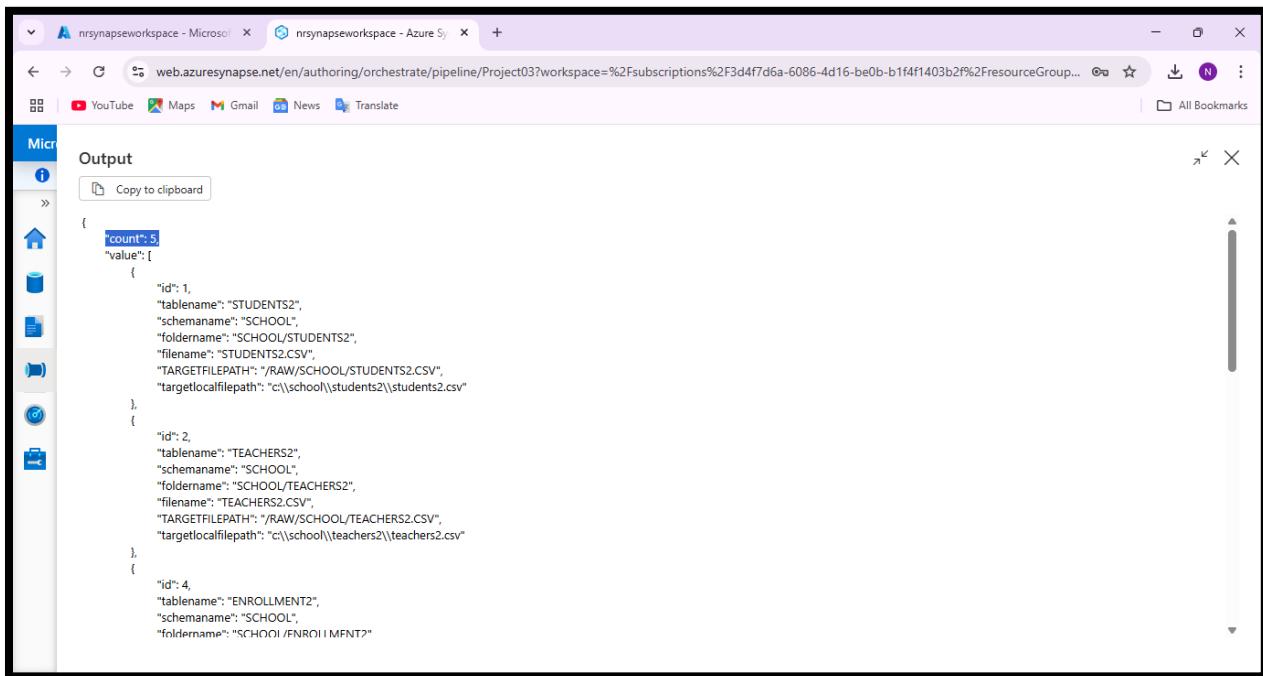
13. NOW GO TO DATA ON 'INTEGRATE ' ON LEFT SIDE OF PAGE ,CREATE A PIPELINE,NAME IT AND DRAG A LOOKUP ACTIVITY TO LOOKUP FOR TABLES PRESENT IN METADATA TABLE IN SQL SERVER.



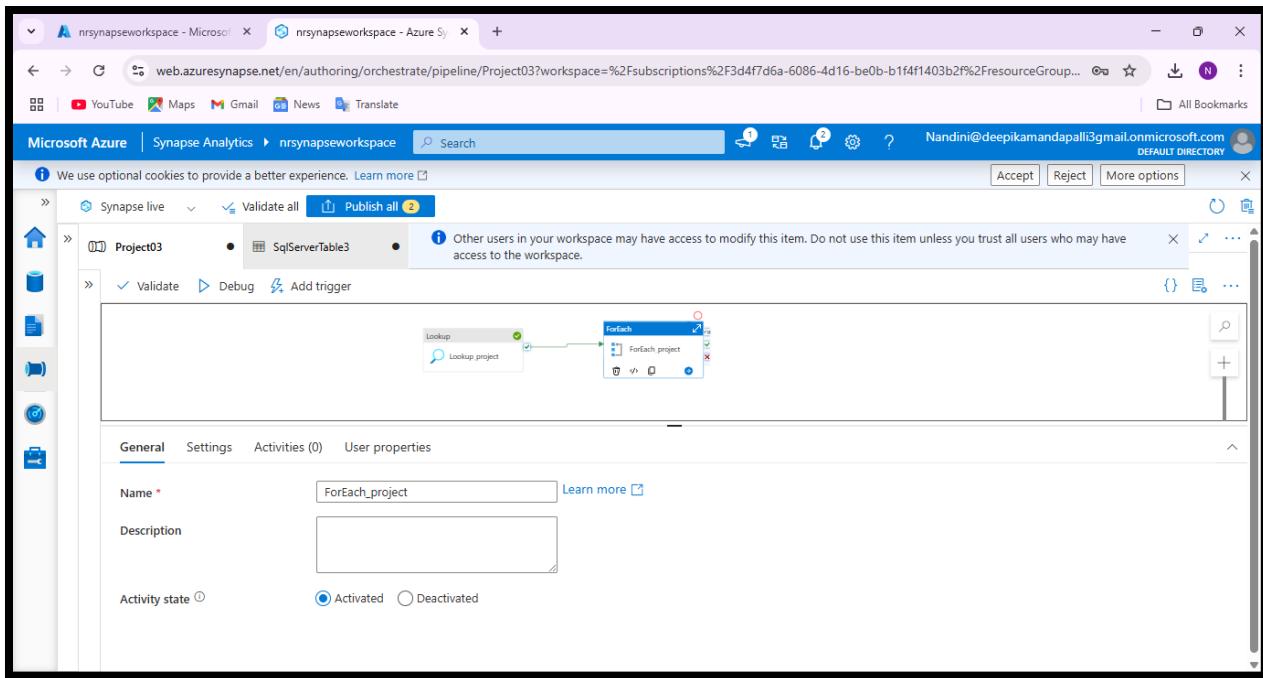
14. CREATE PARAMETERS.



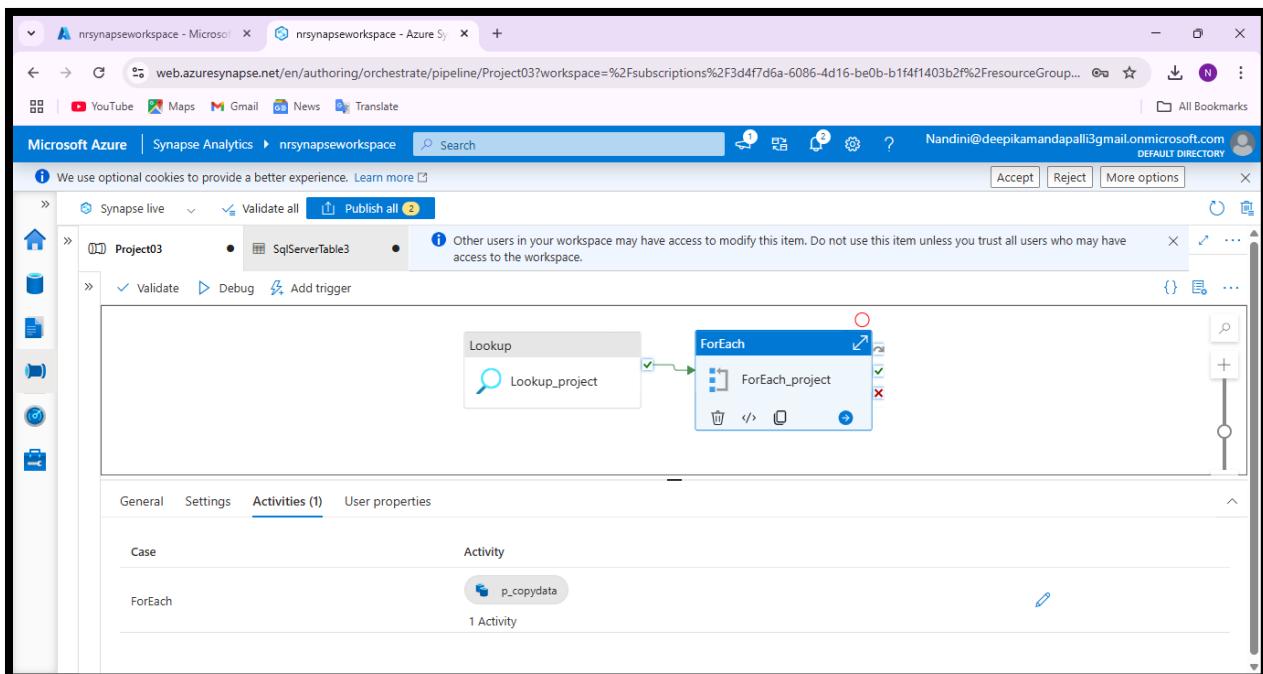
15. THIS ACTIVITY IS SUCCEEDED.

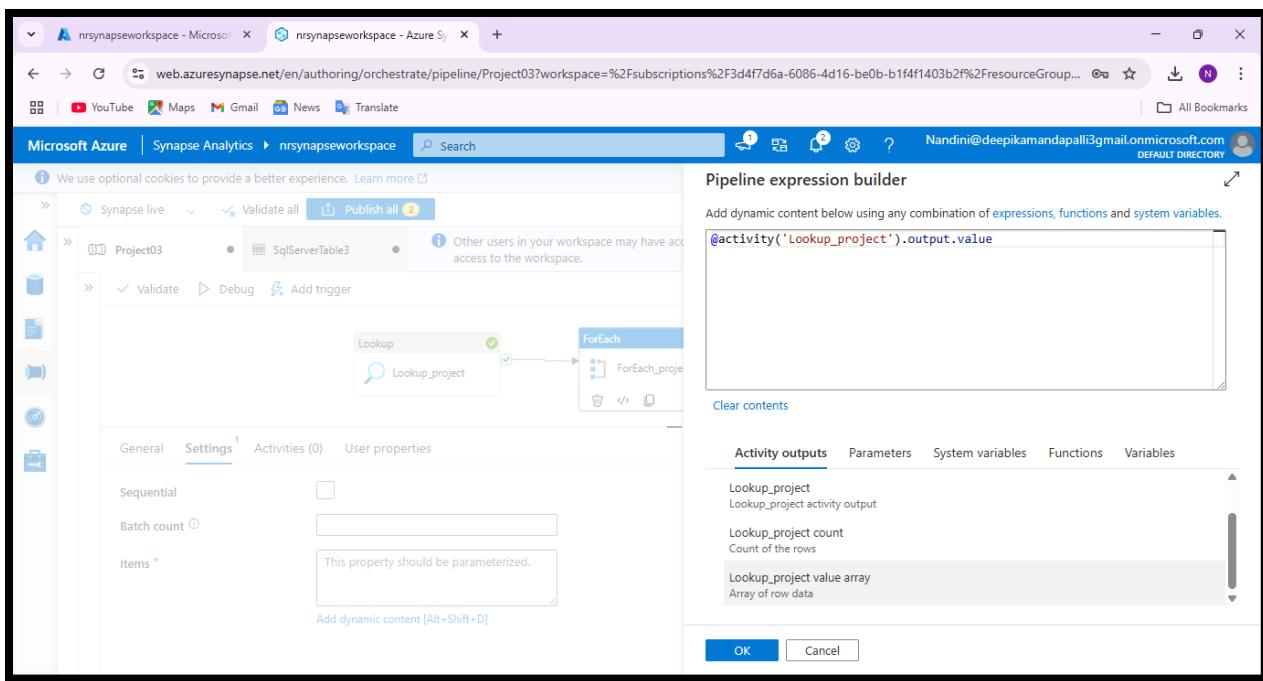


14. THIS SHOWS NO. OF TABLES IN METADATA TABLE,COUNT=5.

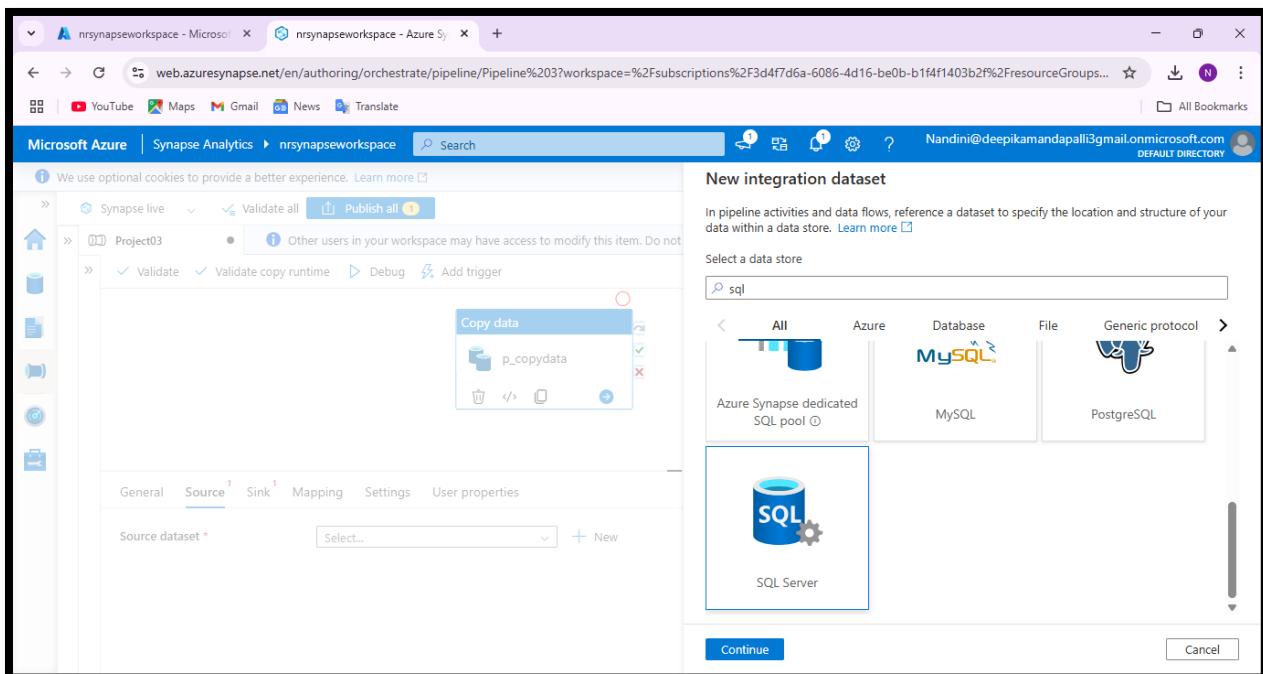


16. AFTER THAT CREATE FOR EACH LOOP ,NAME IT AND CONNECT IT WITH LOOKUP DATA SO THAT IT WILL ITERATE THE ARRAYS OF METADATA TABLES OR OUTPUT OF LOOKUP ACTIVITY.

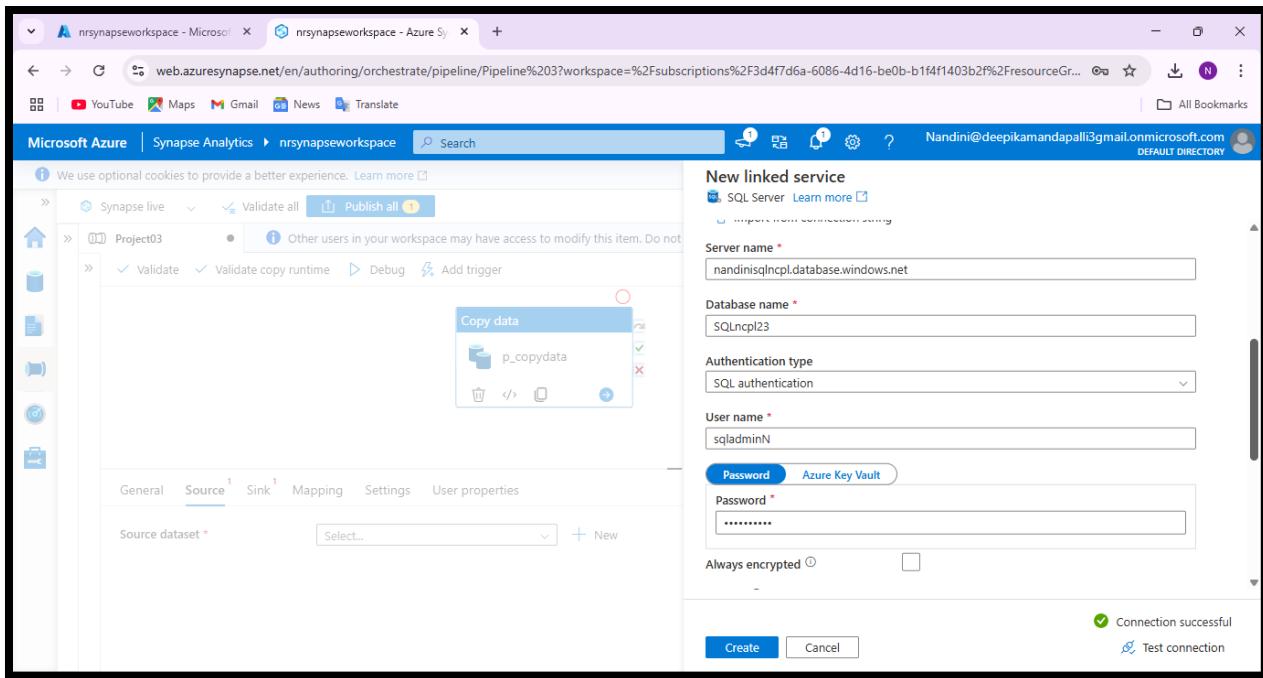




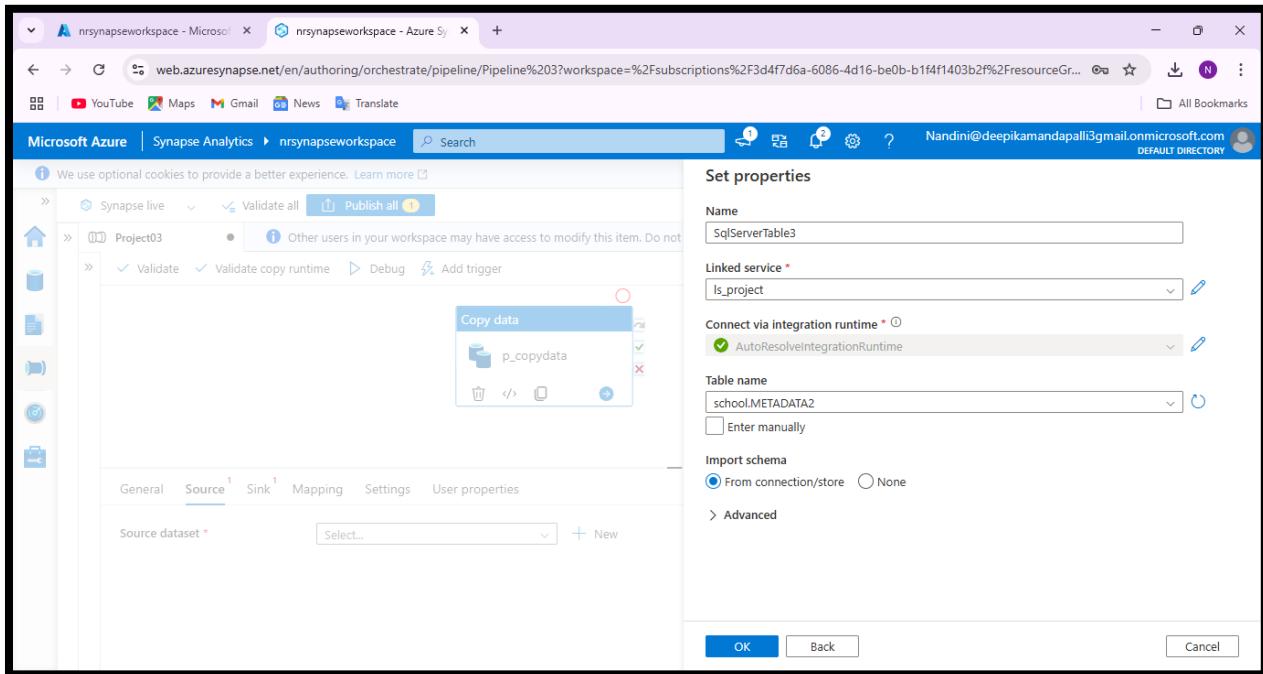
17. INSIDE FOREACH ACTIVITY CREATE COPY DATA ACTIVITY, HERE WE USE @ACTIVITY , WHICH WILL COPY OUTPUT OF LOOKUP ACTIVITY.



18. SELECT 'SQL SERVER' AS SOURCE DATASET.



19. CREATE NEW LINKED SERVICE.

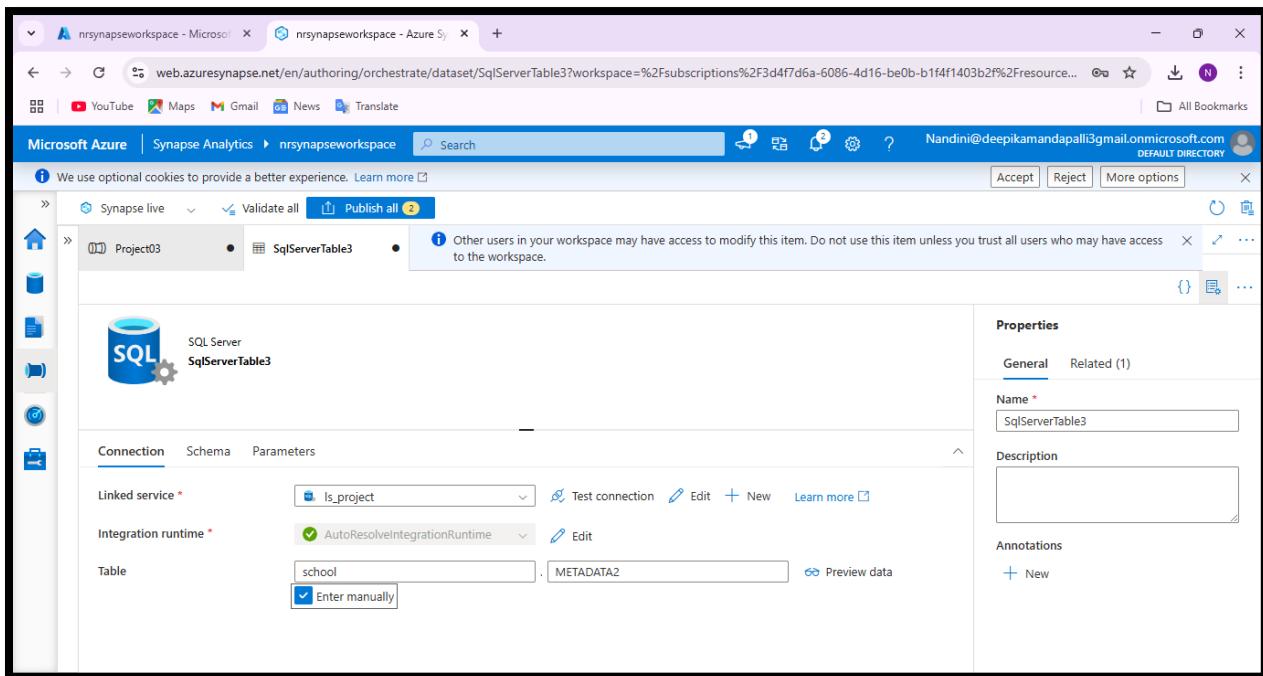


20. SET PROPERTIES AND SELECT TABLE NAME.'SCHOOL.METADATA2'

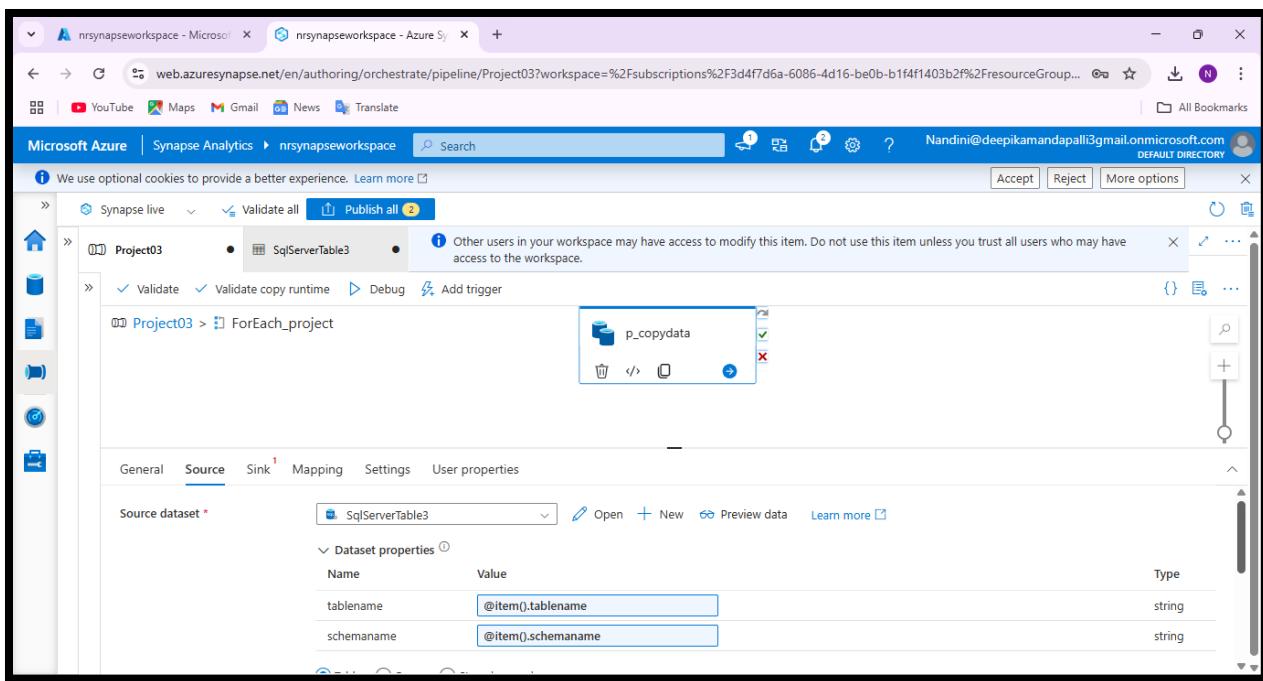
4. *Implement a metadata-driven approach using a control table that stores details such as:*

1. *Source table names*

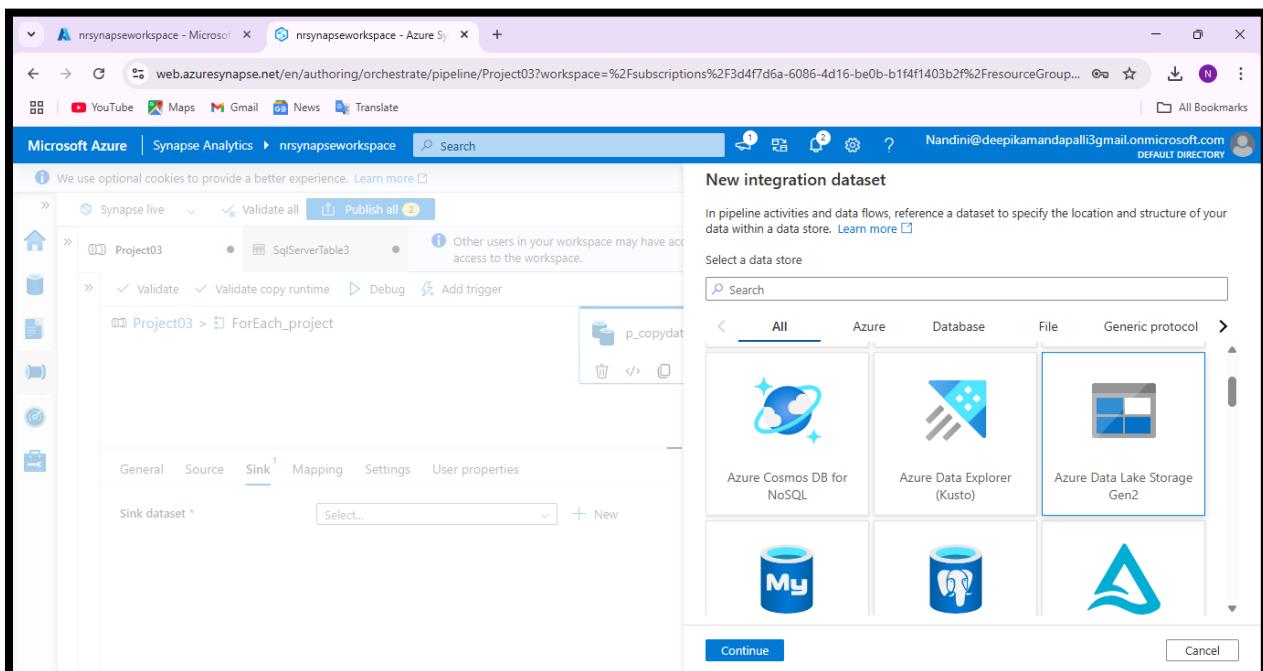
2. *Target file paths*



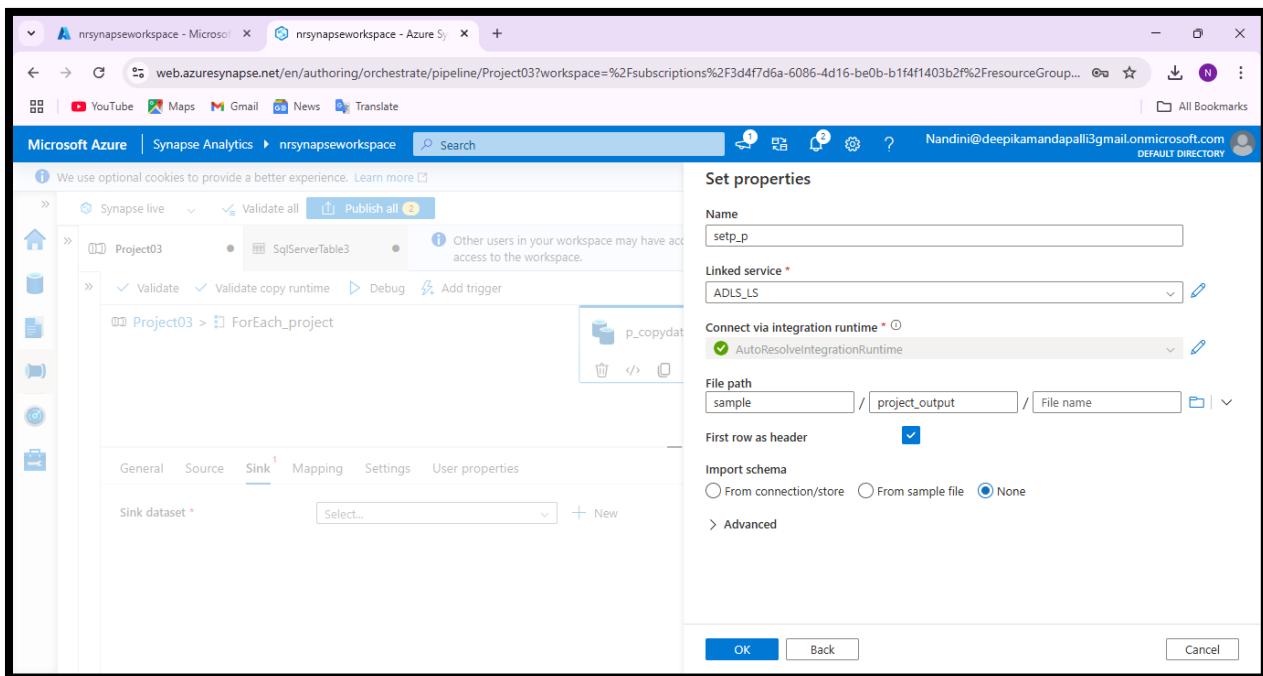
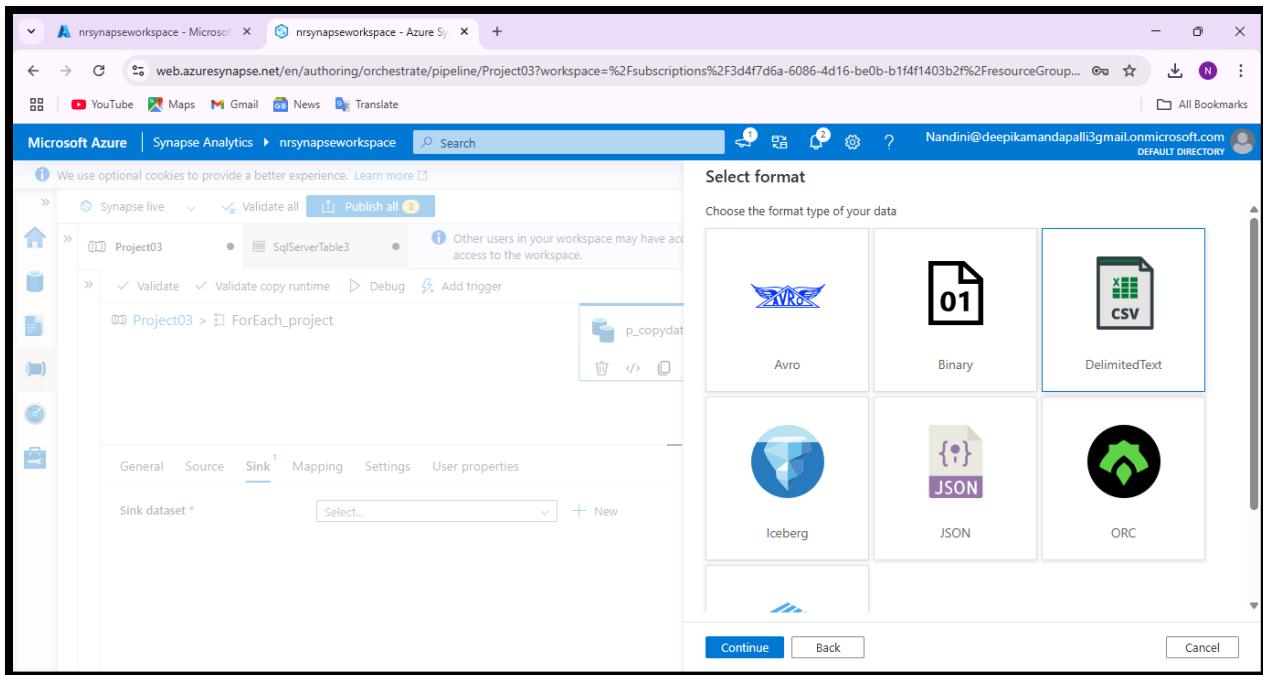
21. CREATE PARAMETERS SCHEMANAME AND TABLENAME.



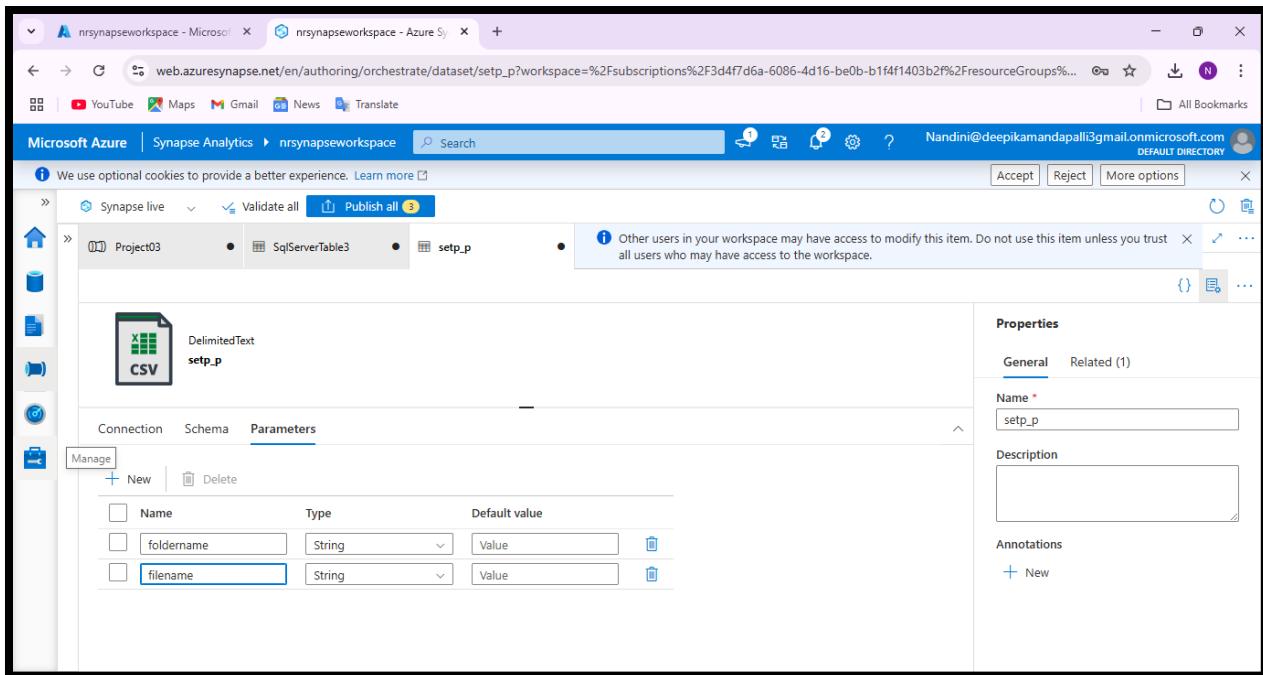
22. USE @ITEM(). FUNCTION ,THIS WILL ITERATE EVERY ITEM OF THE TABLE.



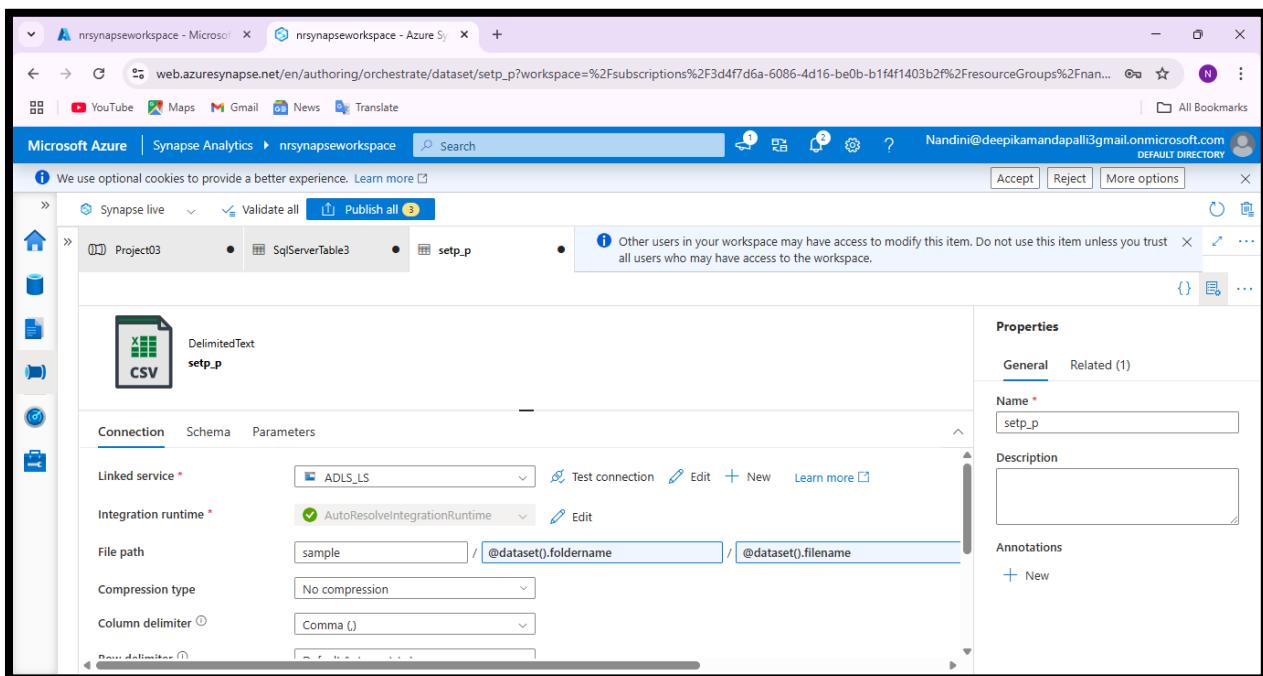
23. NOW FOR SINK SELECT ADLSGEN2,THEN CSV FORMAT.

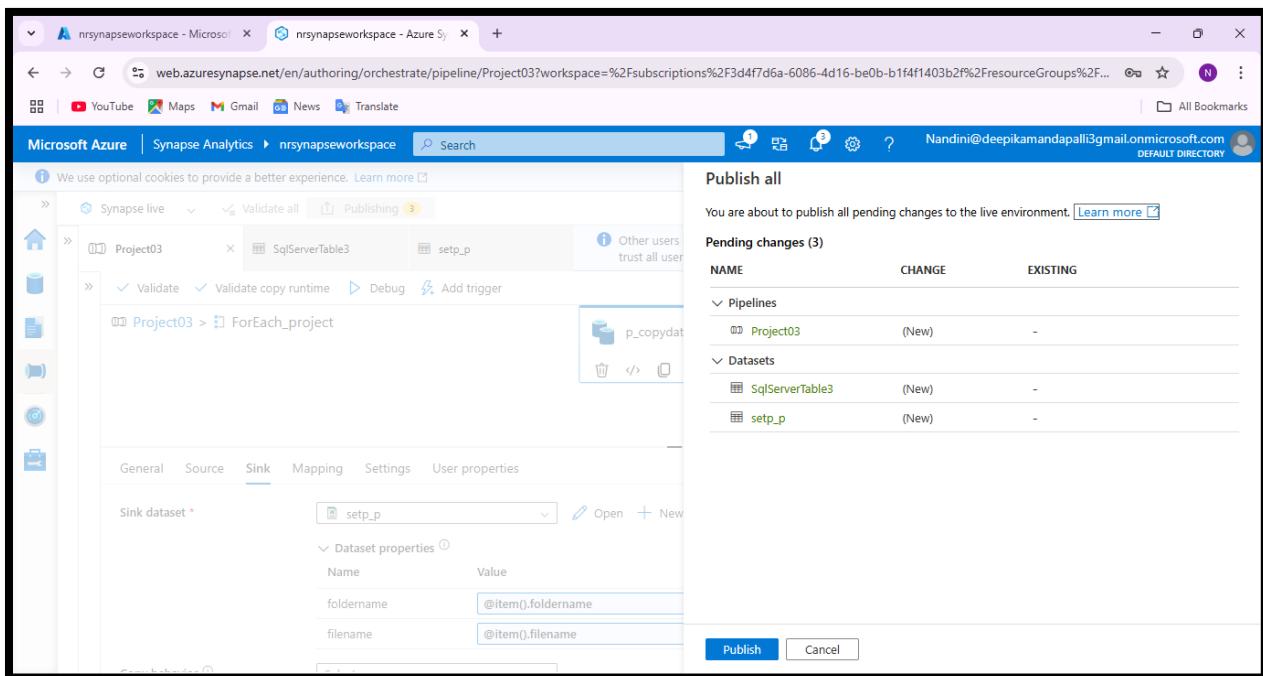
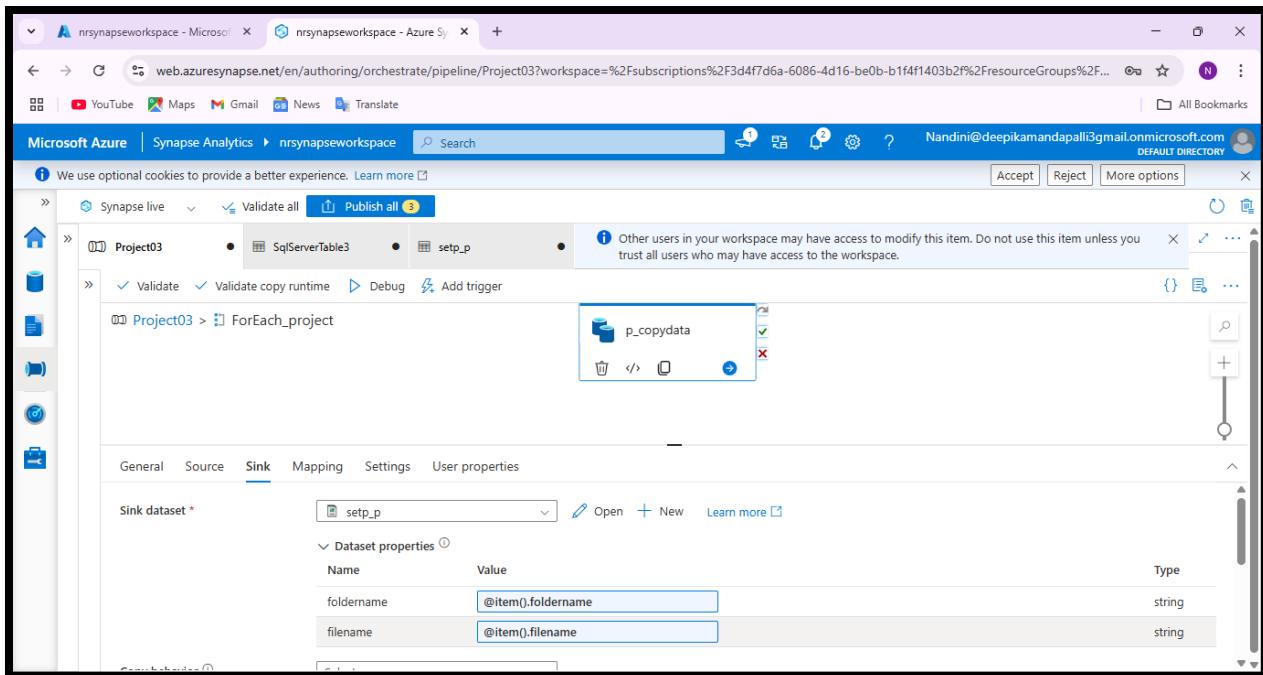


24. SET PROPERTIES AND SELECT FILE PATH WHERE WE WANT TO COPY DATA IN ADLSGEN2 ACCOUNT.

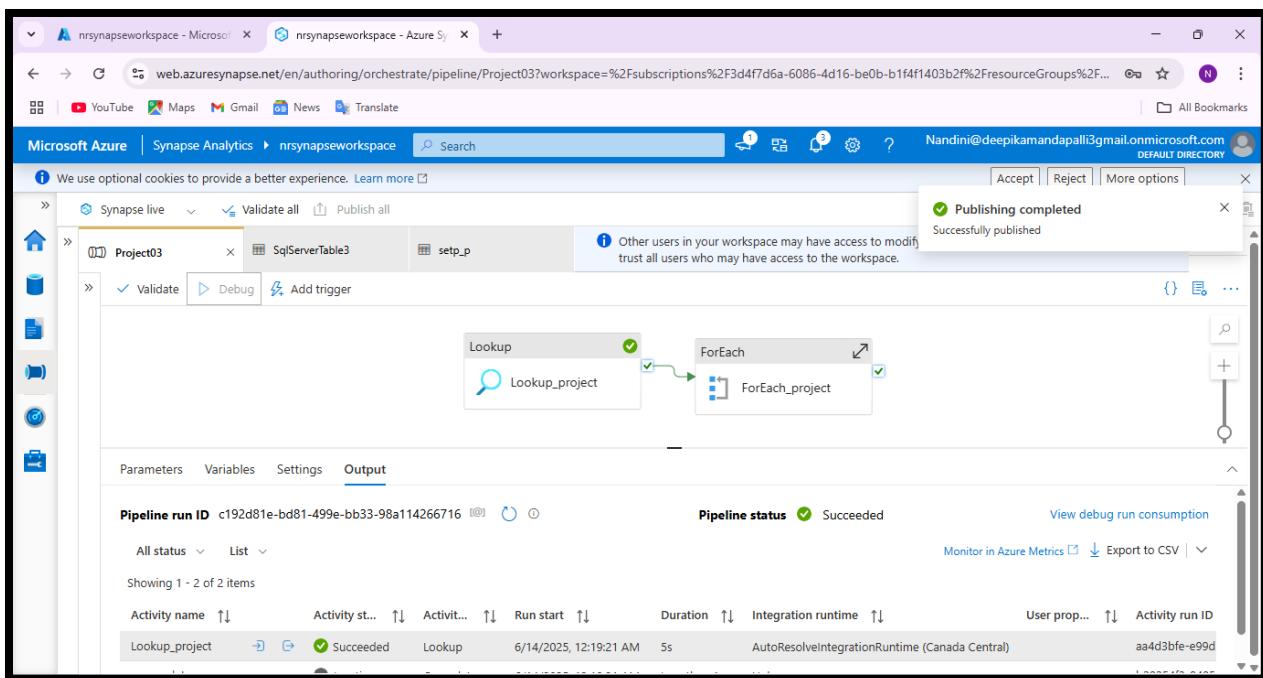


25. CREATE PARAMETERS FOR ADLSGEN2 → FOLDERNAME,FILENAME.

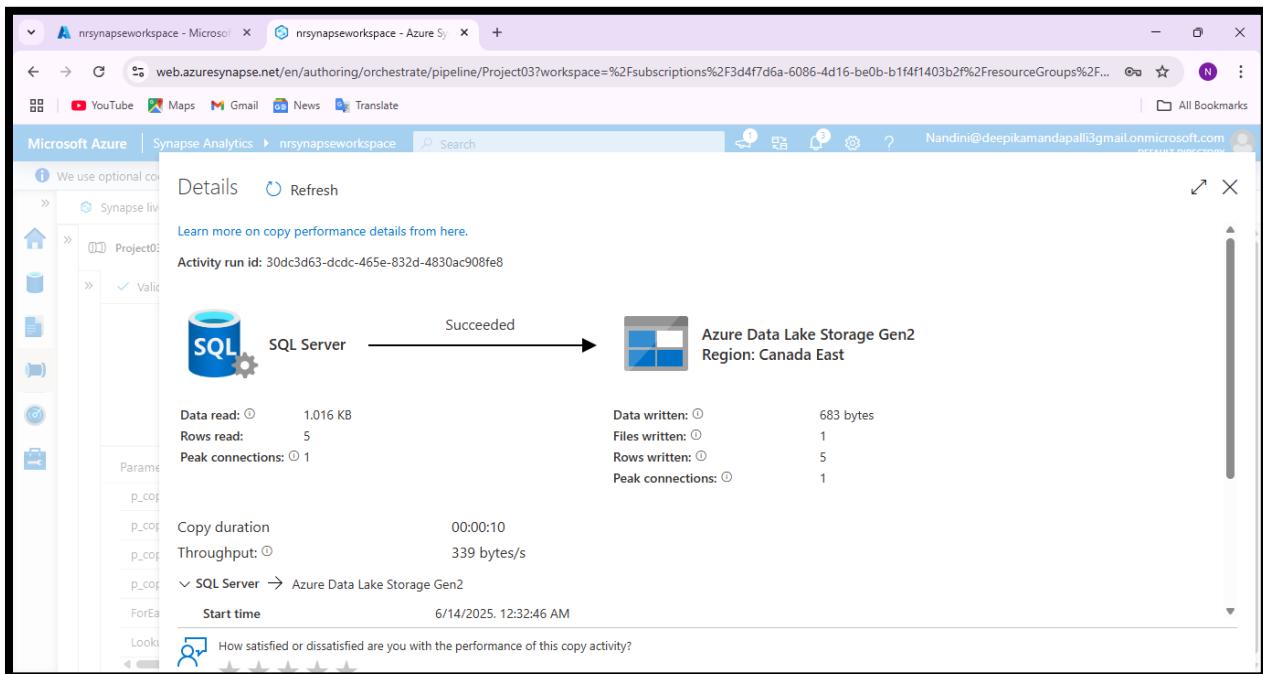




26.PUBLISH PIPELINE.



27.PIPELINE SUCCEEDED.



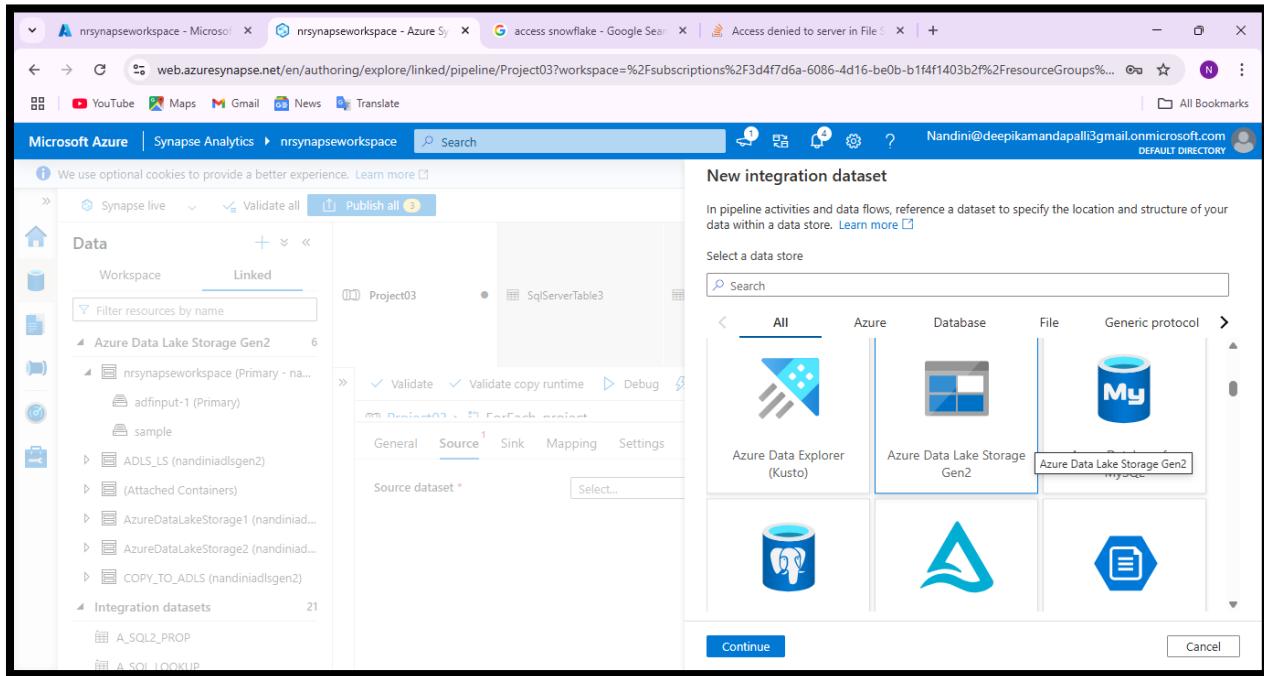
28. DATA SUCCESSFULLY LOADED FROM SQL SERVER TO ADLSGEN2.

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
p_copydata	Succeeded	Copy data	6/14/2025, 12:32:44 AM	15s	AutoResolveIntegrationRuntime (Canada East)		599a1de4-120c
p_copydata	Succeeded	Copy data	6/14/2025, 12:32:43 AM	15s	AutoResolveIntegrationRuntime (Canada East)		30dc3d63-dcdc
p_copydata	Succeeded	Copy data	6/14/2025, 12:32:43 AM	15s	AutoResolveIntegrationRuntime (Canada East)		a1c905e0-1c0b
p_copydata	Succeeded	Copy data	6/14/2025, 12:32:43 AM	15s	AutoResolveIntegrationRuntime (Canada East)		1453add3-14f2
p_copydata	Succeeded	Copy data	6/14/2025, 12:32:43 AM	21s	AutoResolveIntegrationRuntime (Canada East)		f8c72699-7edb
ForEach_project	Succeeded	ForEach	6/14/2025, 12:32:43 AM	24s			2ad4c0d4-7aa3
Lookup_project	Succeeded	Lookup	6/14/2025, 12:32:46 AM	6s	AutoResolveIntegrationRuntime (Canada Central)		3dfb8257-64b1

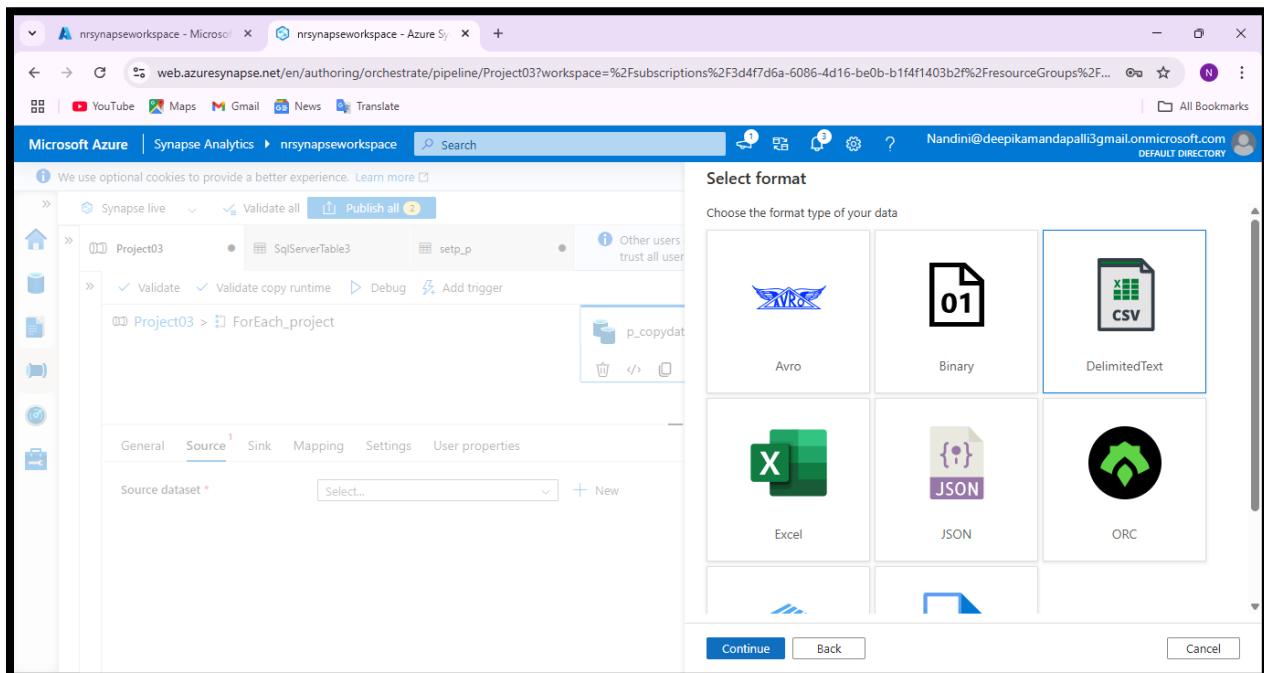
Name	Last Modified	Content Type	Size
ATTENDANCE3	14/06/2025, 00:33:01	Folder	
ENROLLMENT2	14/06/2025, 00:32:55	Folder	
FEE	14/06/2025, 00:32:56	Folder	
STUDENTS2	14/06/2025, 00:32:56	Folder	
TEACHERS2	14/06/2025, 00:32:56	Folder	

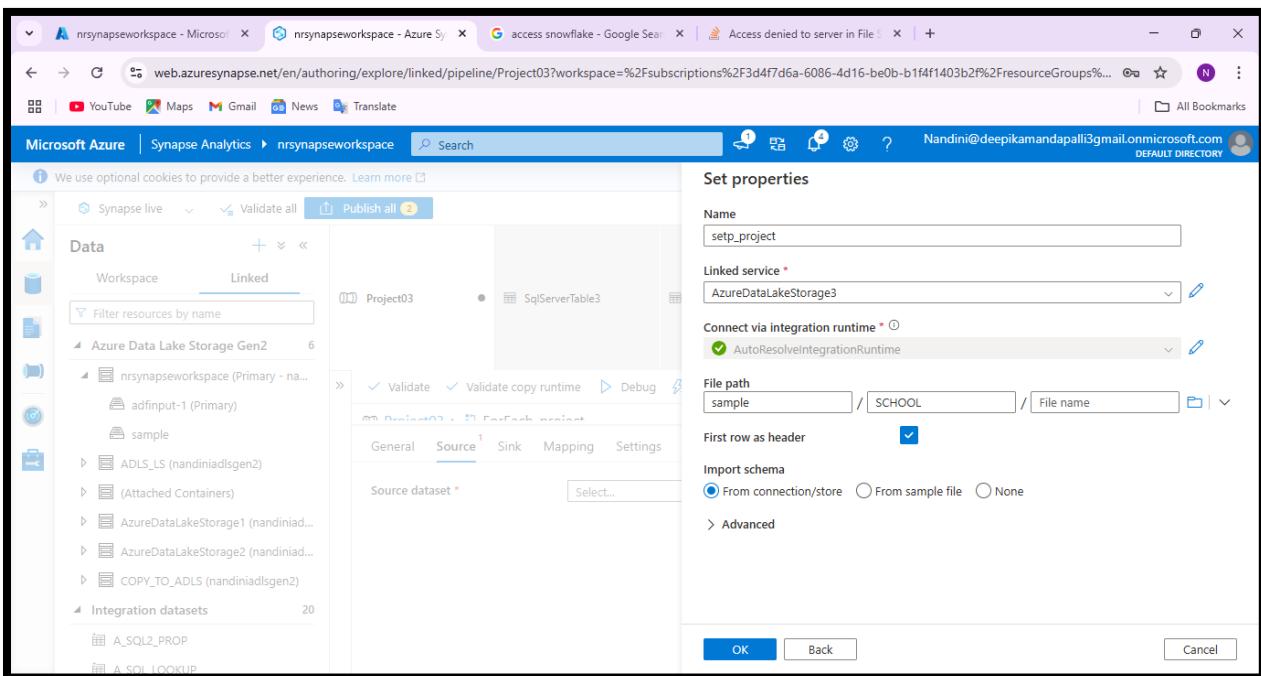
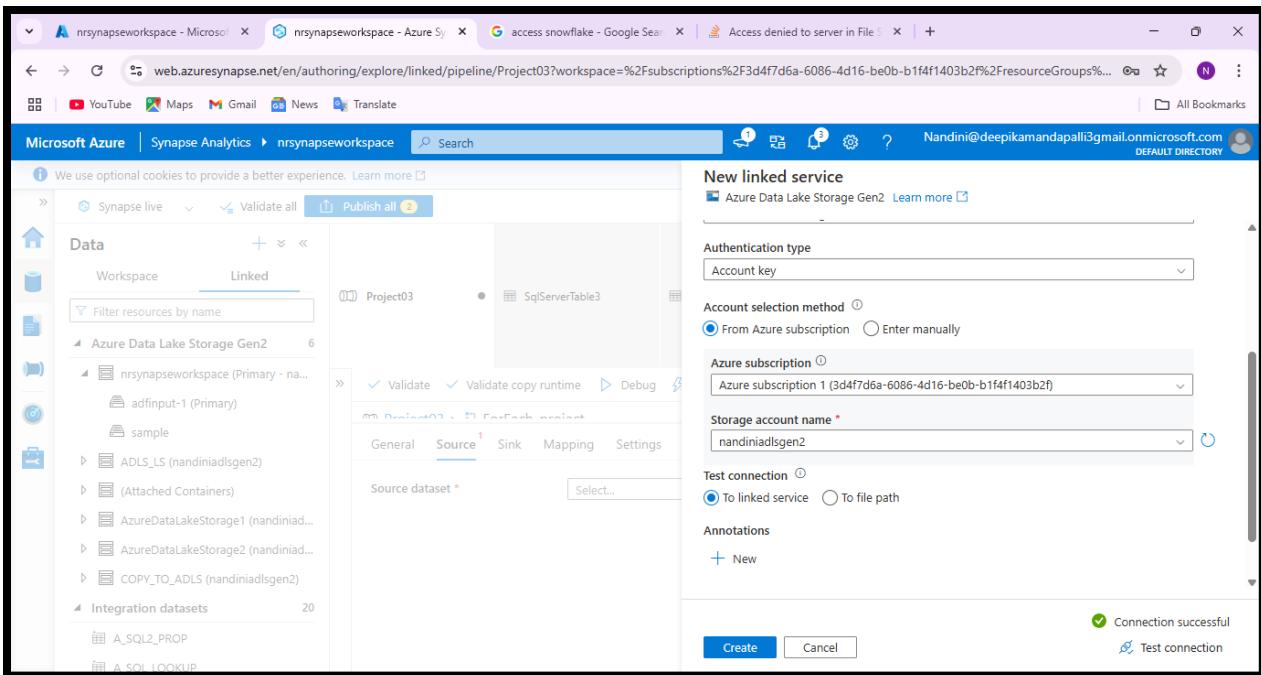
29. LOADED DATA PRESENT IN CLOUD.

3. Copy the data from ADLS Gen2 back to a designated file system folder on the local C: drive.

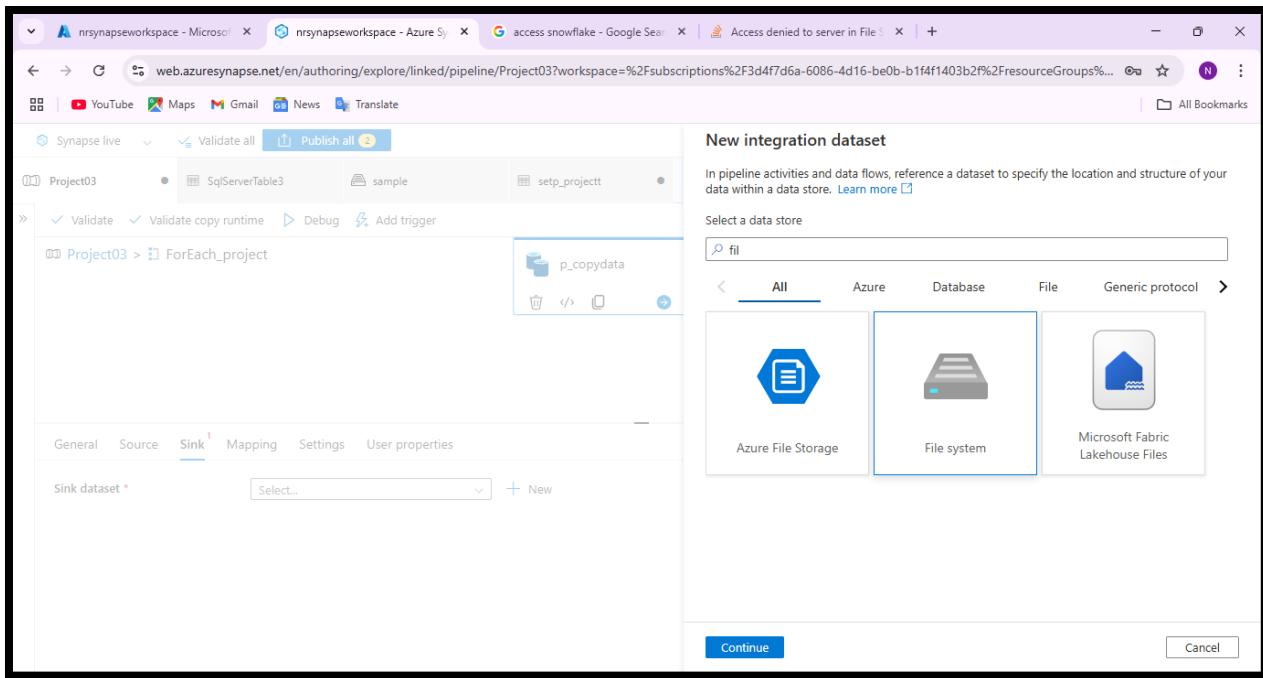


30. NOW FOR LOADING OF DATA FROM CLOUD TO SQL SERVER, USE COPY ACTIVITY IN THE SAME PIPELINE HERE FOR SOURCE DATASET → ADLSGEN2, SINK → FILE SERVER.

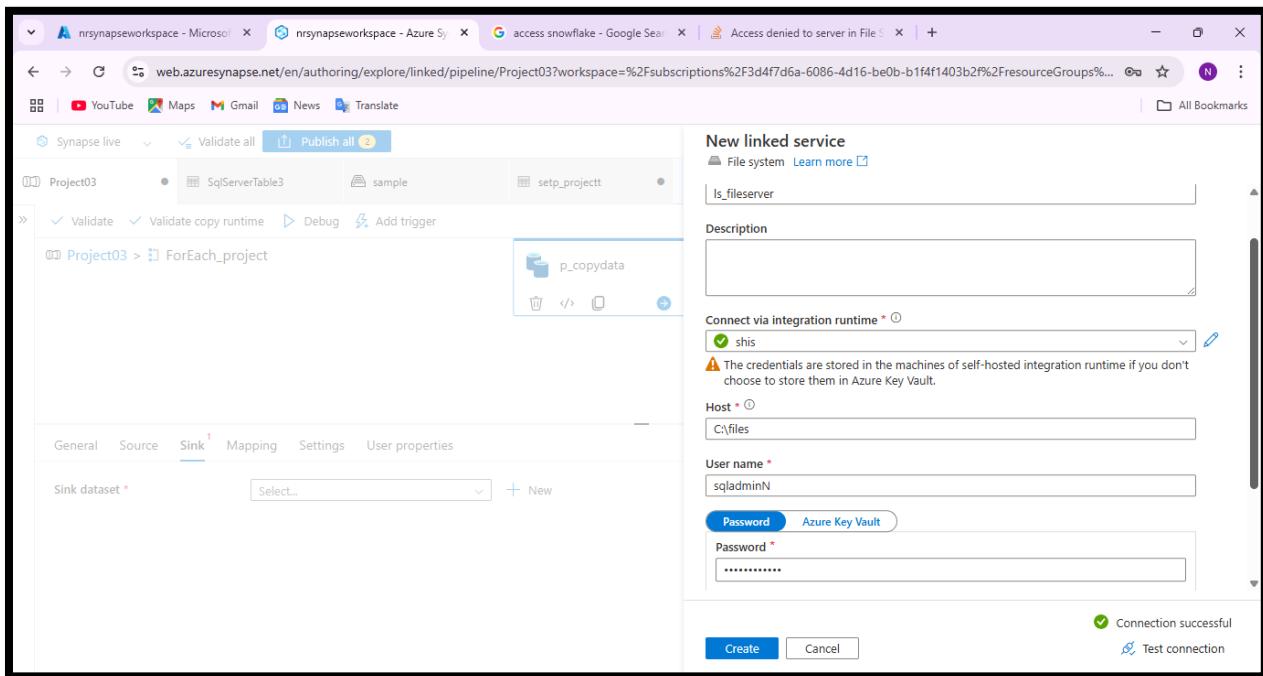




31.SELECT FOLDER WHICH YOU WANT TO LOAD ON-PREMISE.

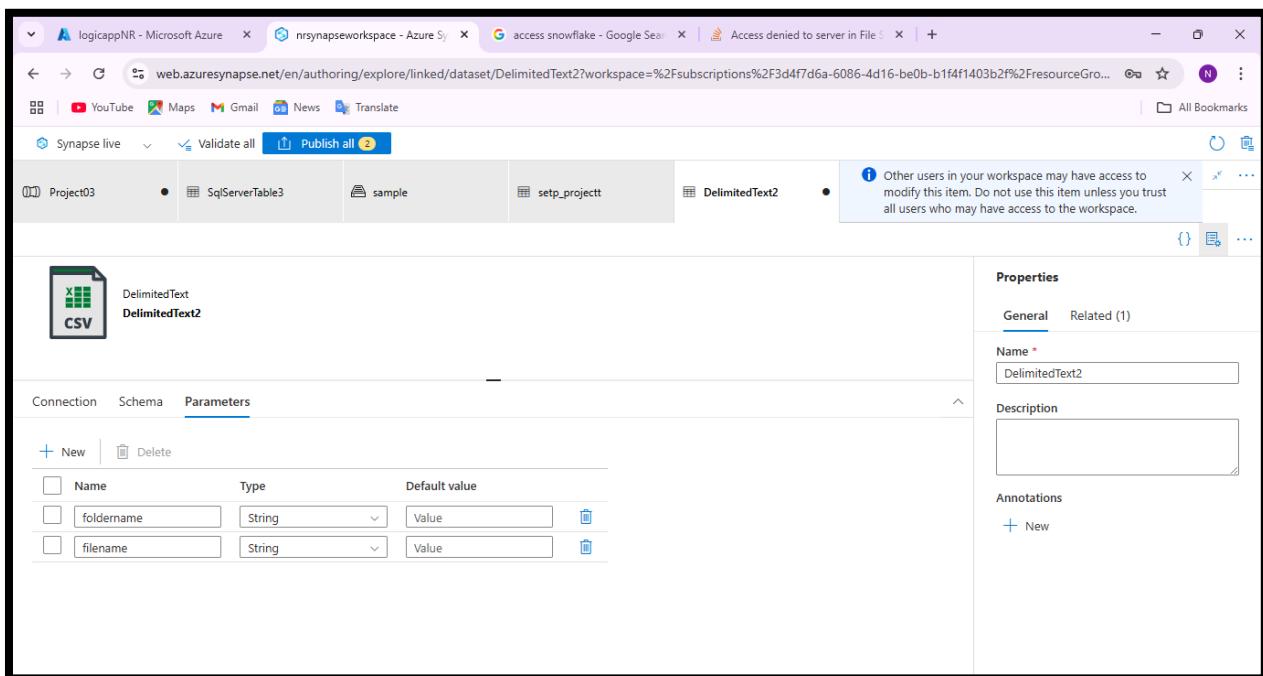
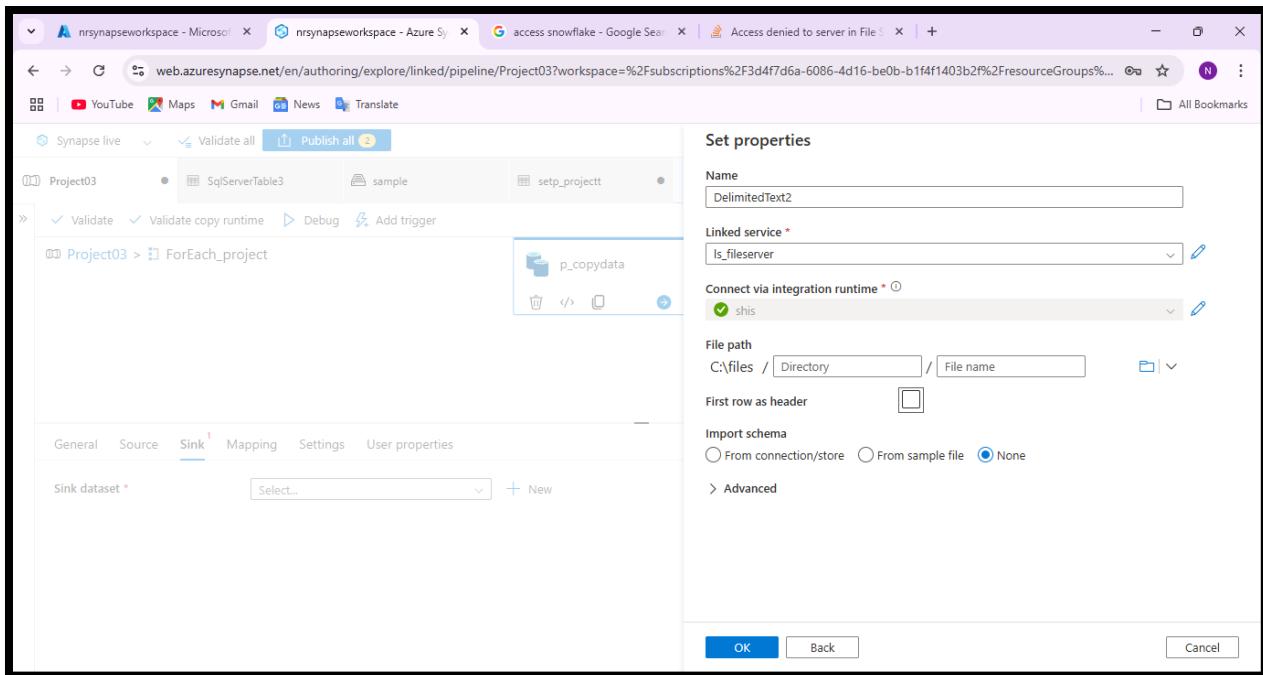


33.FOR SINK→FILE SYSTEM.

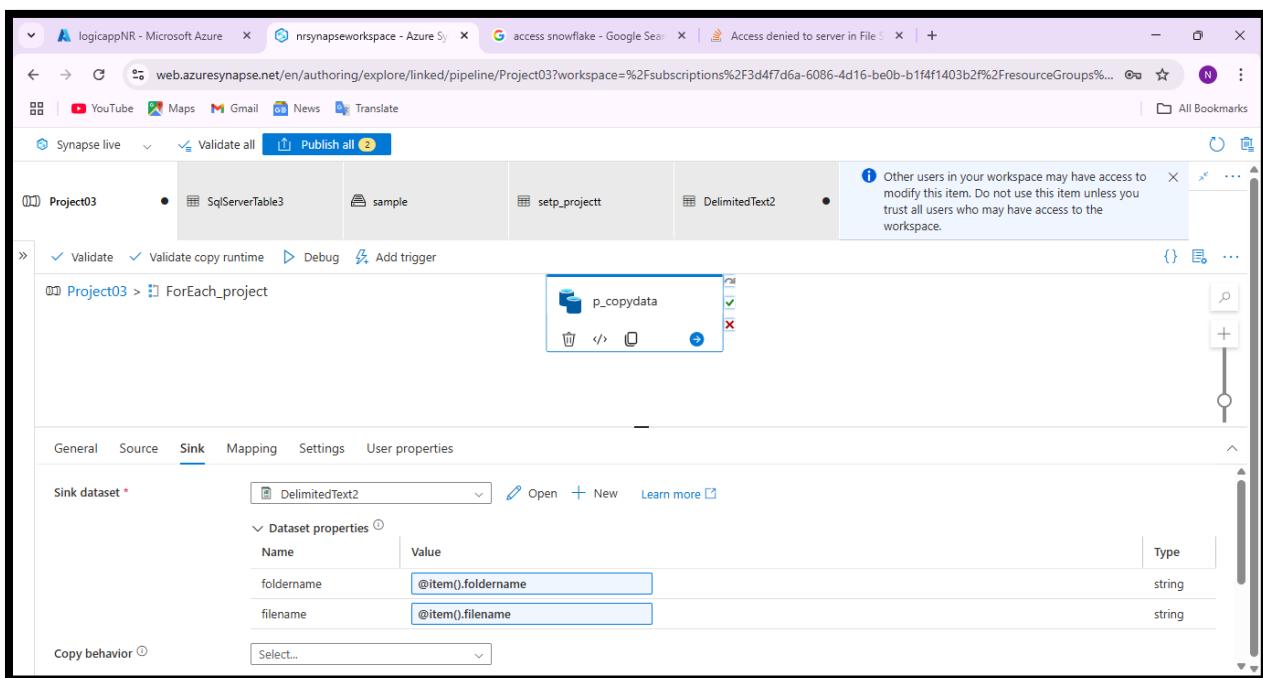
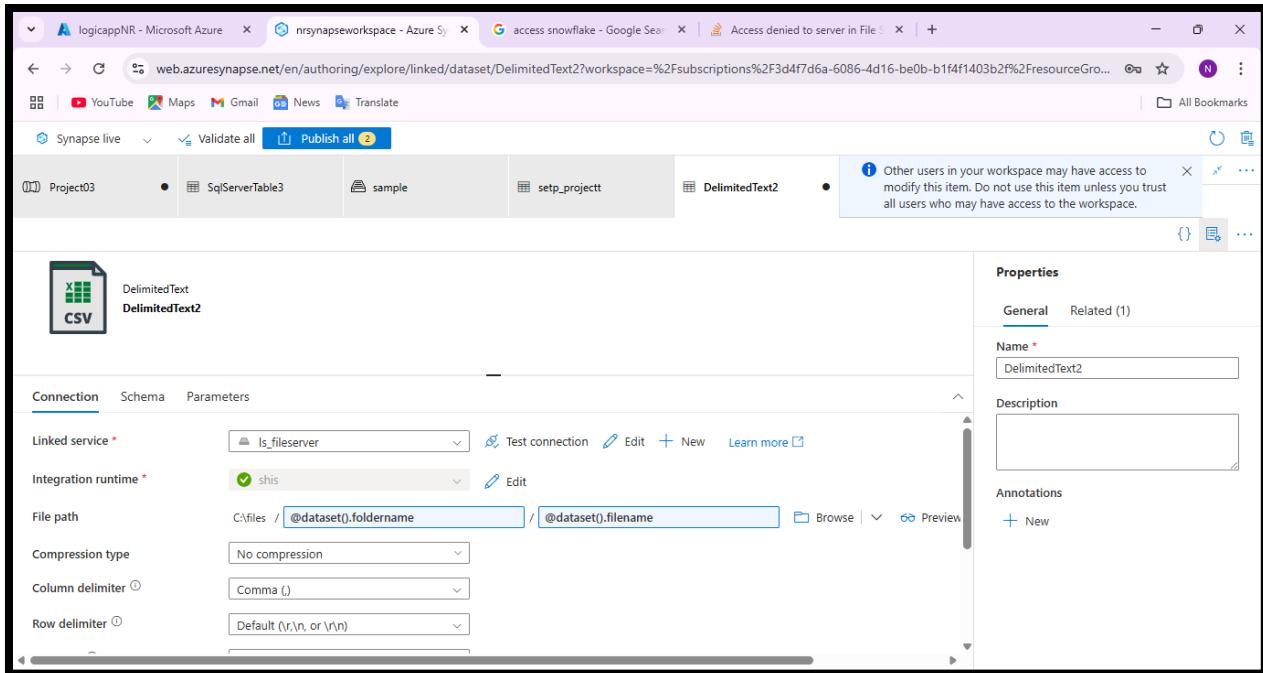


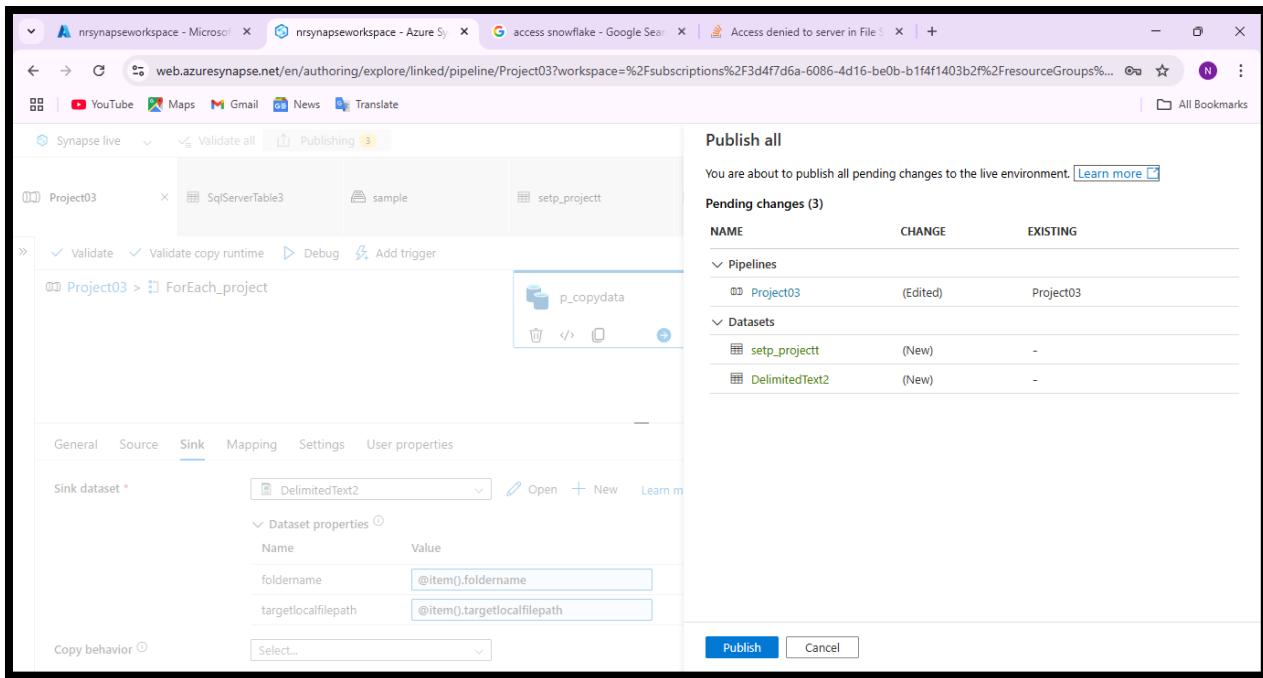
34.SELECT SHIR,HOST ,CREATE LINKED SERVICE.

HERE I WANT TO LOAD FROM CLOUD TO ON-PREMISE THAT'S WHY I USE "SHIR". THAT I HAVE CREATED IN VM.

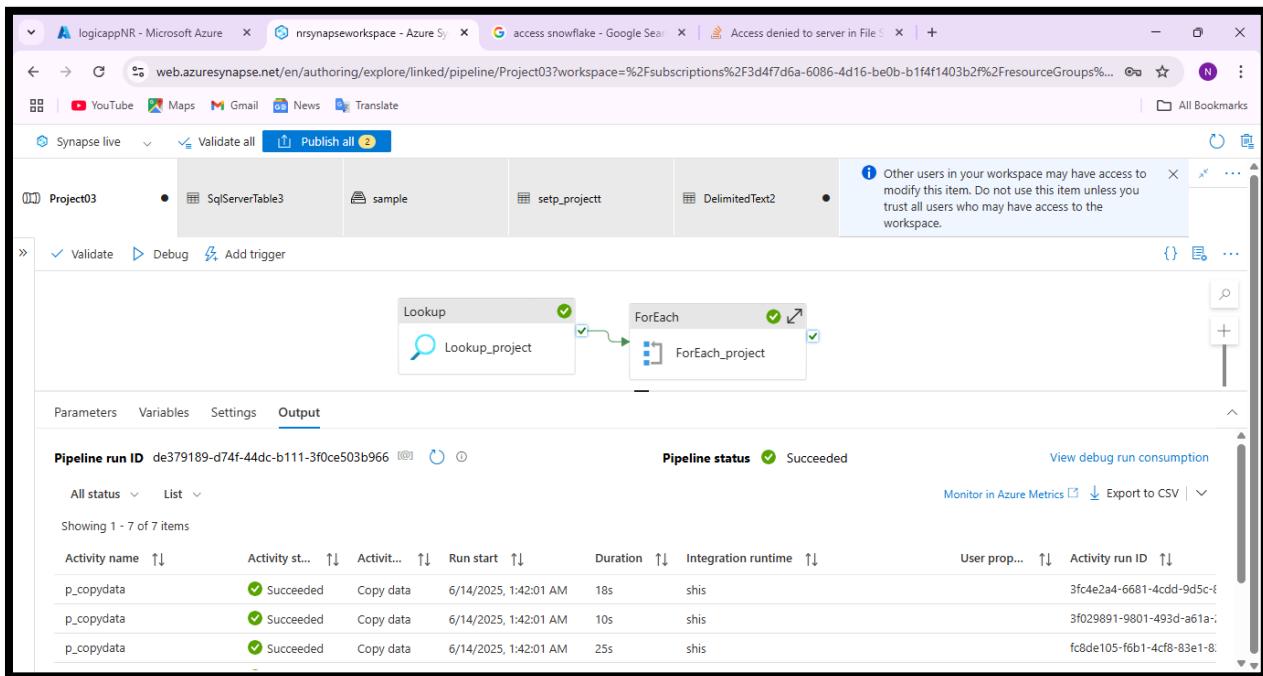


.CREATE PARAMETERS FOR FOLDERNAME AND FILENAME.

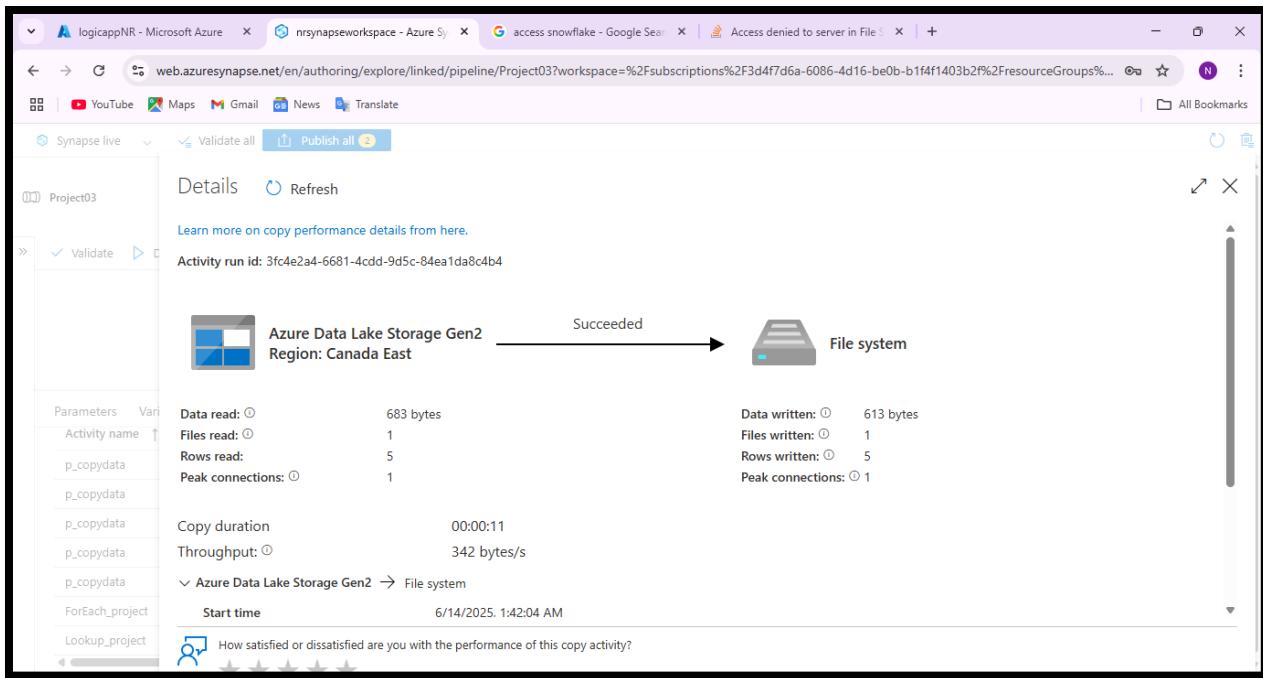




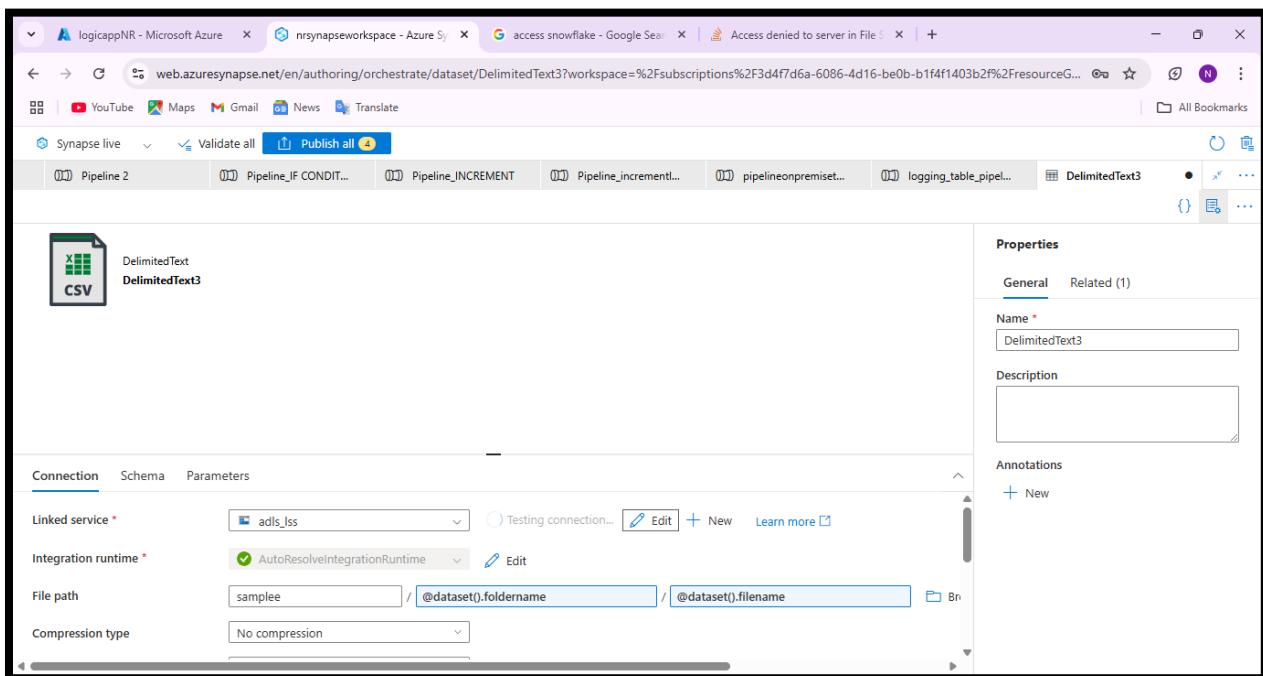
34.PUBLISH PIPELINE.

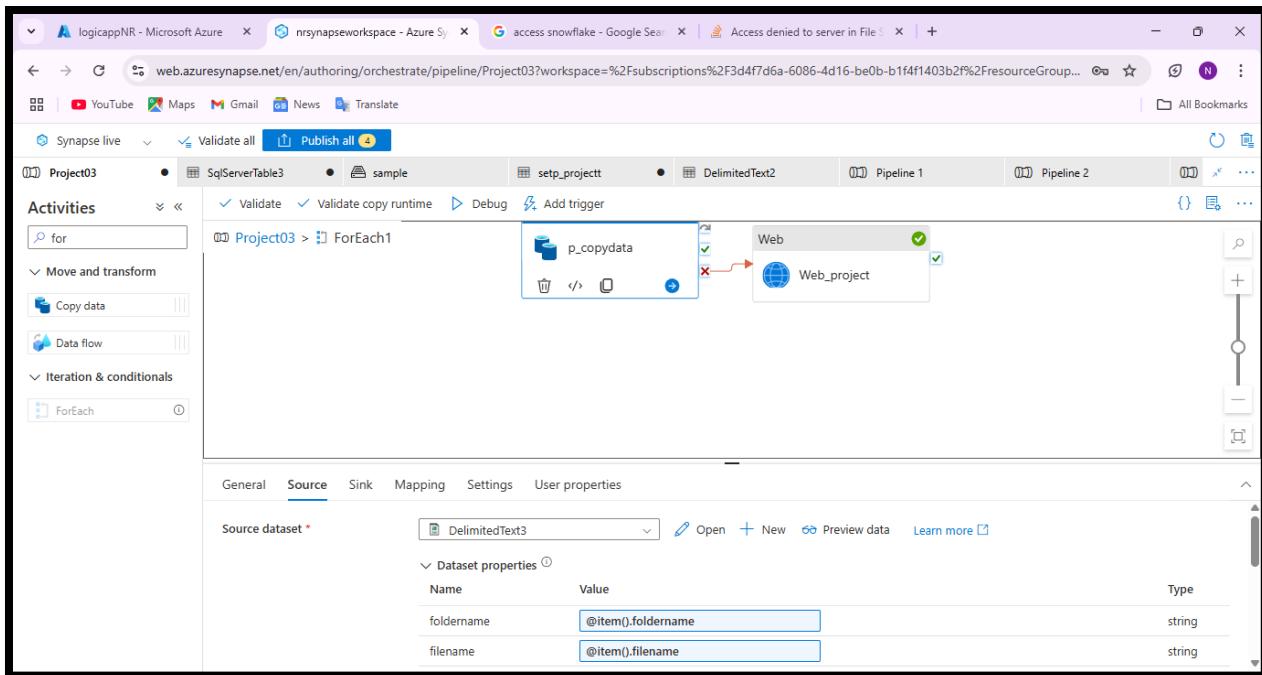


34.PIPELINE STATUS "SUCCEEDED".

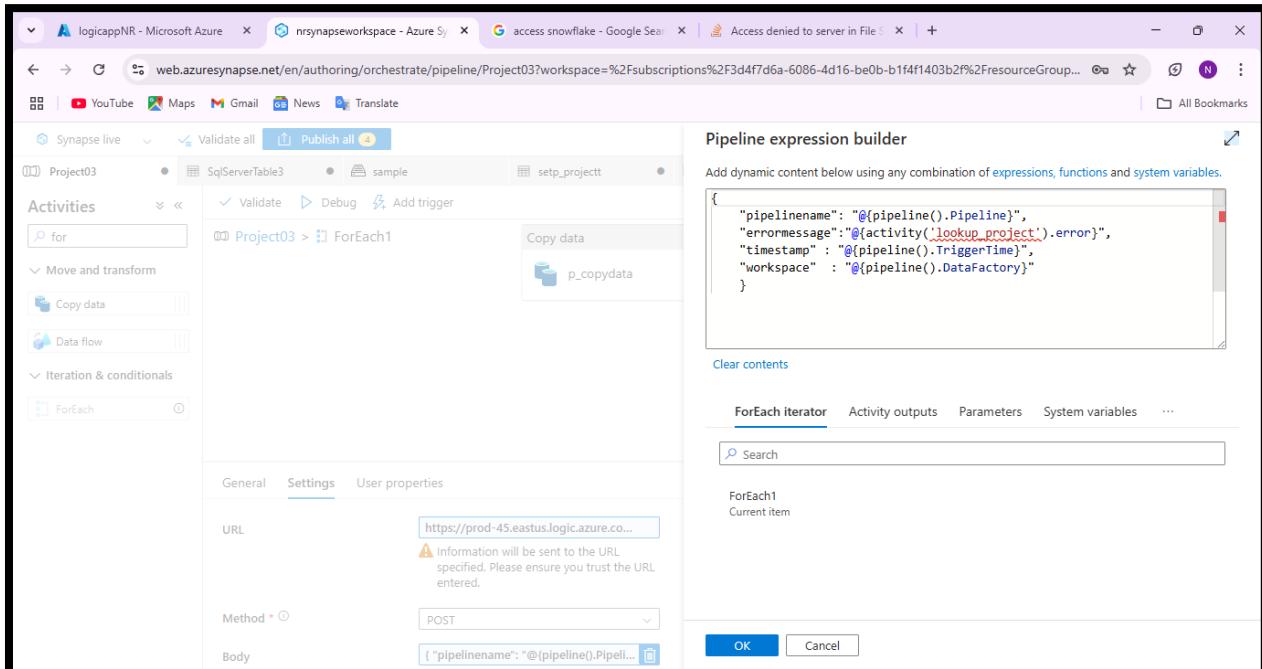


35. DATA LOADED FROM CLOUD TO ON-PREMISE.

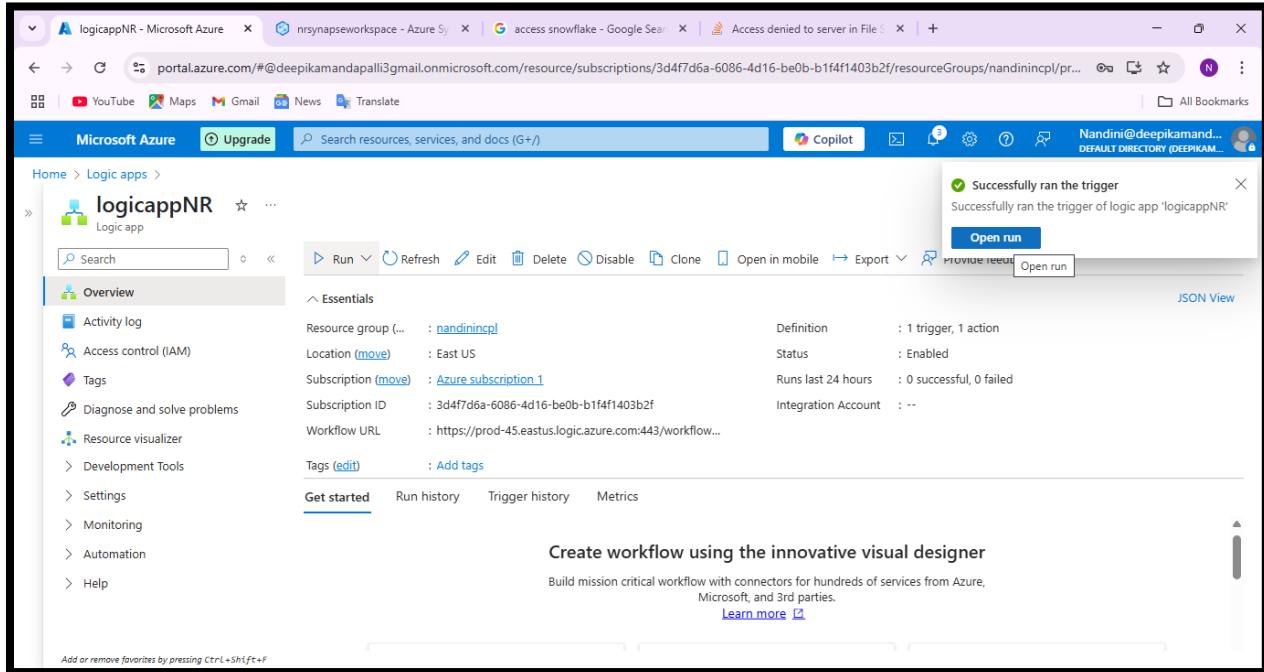




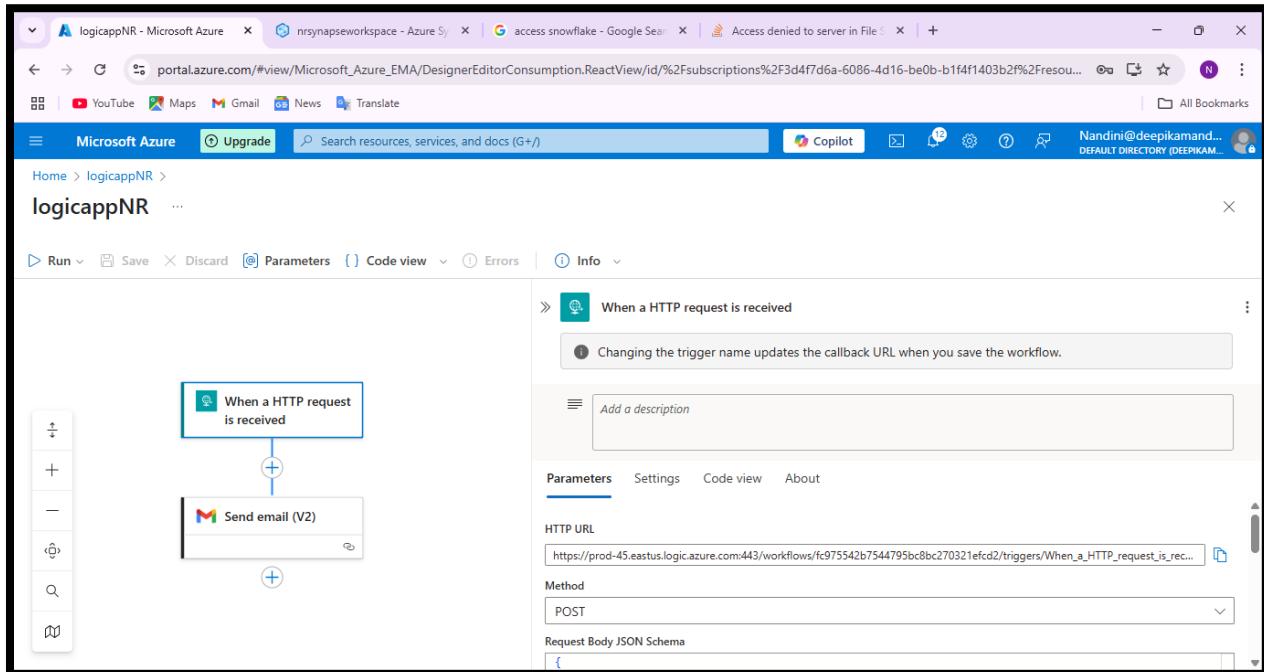
36. NOW CONNECT WEB ACTIVITY WITH COPY DATAIN FOREACH ACTIVITY. WE CONNECTED WEB ACTIVITY BECAUSE IF COPY DATA FAILS IT WILL SEND MAIL ABOUT PIPELINE STATUS BY CONNECTING SYNAPSE PIPELINE WITH LOGIC APPS.



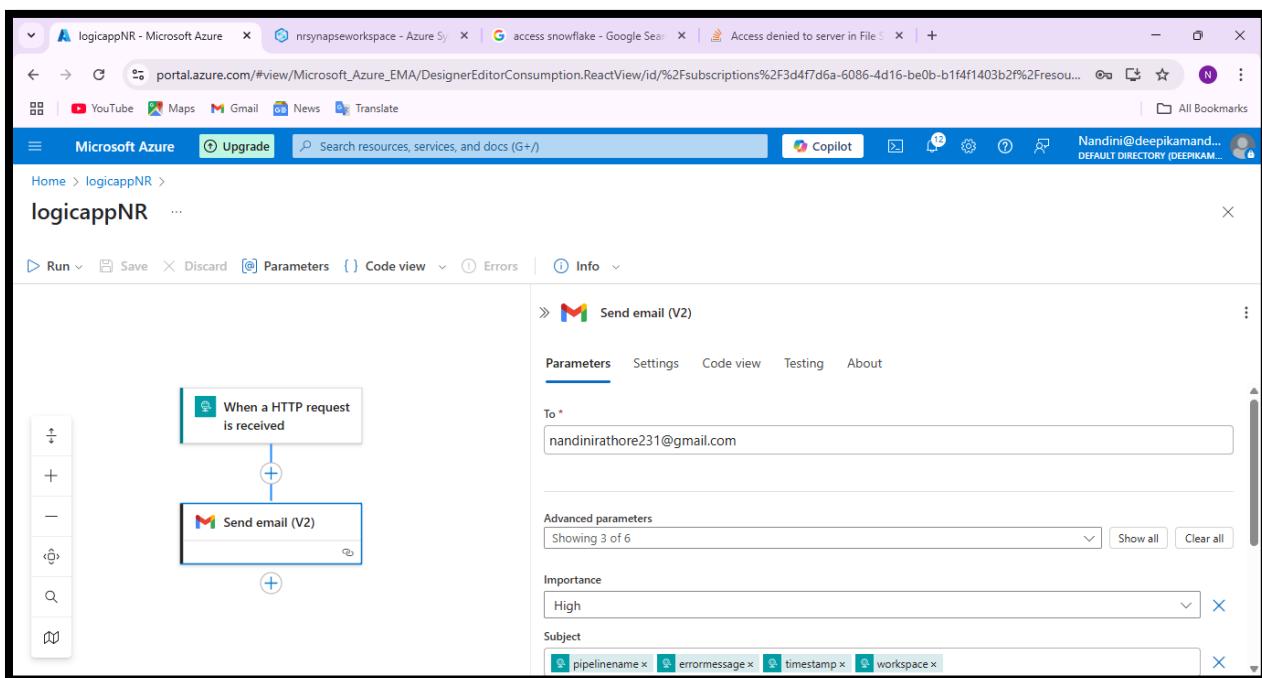
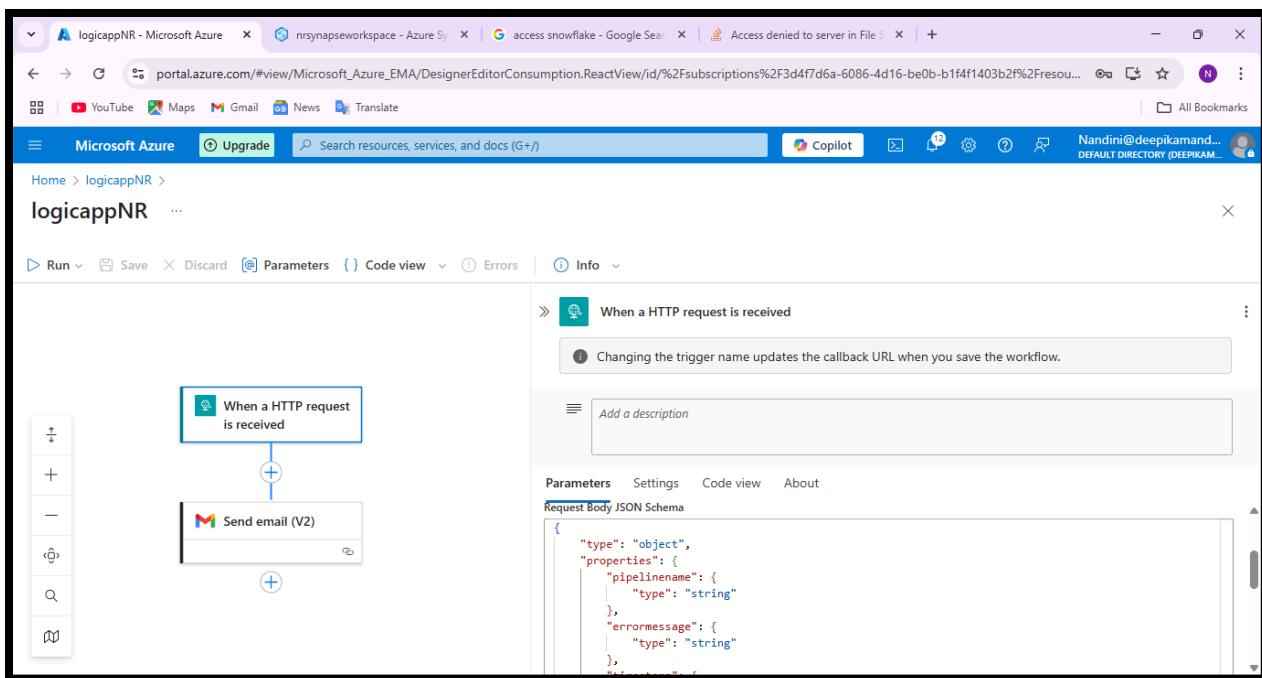
5. Implement the email configuration task by using logic apps and send an email for any failure.



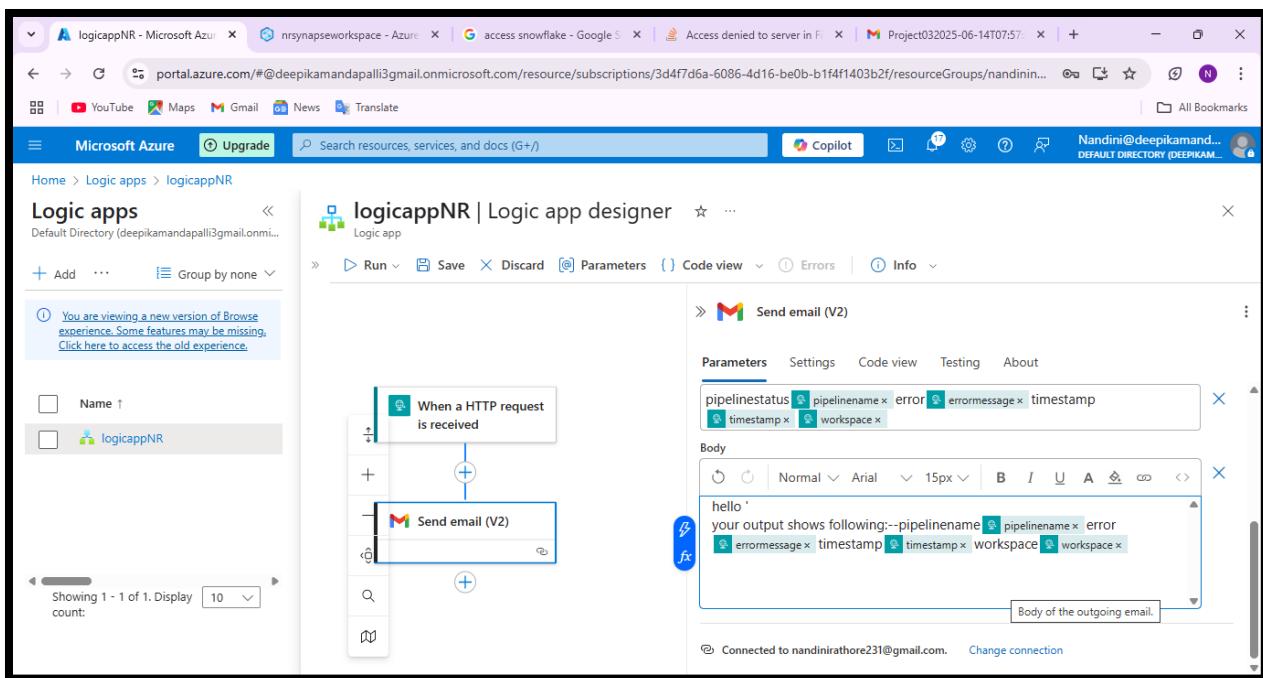
37. OPEN LOGIC APP, RUN IT.



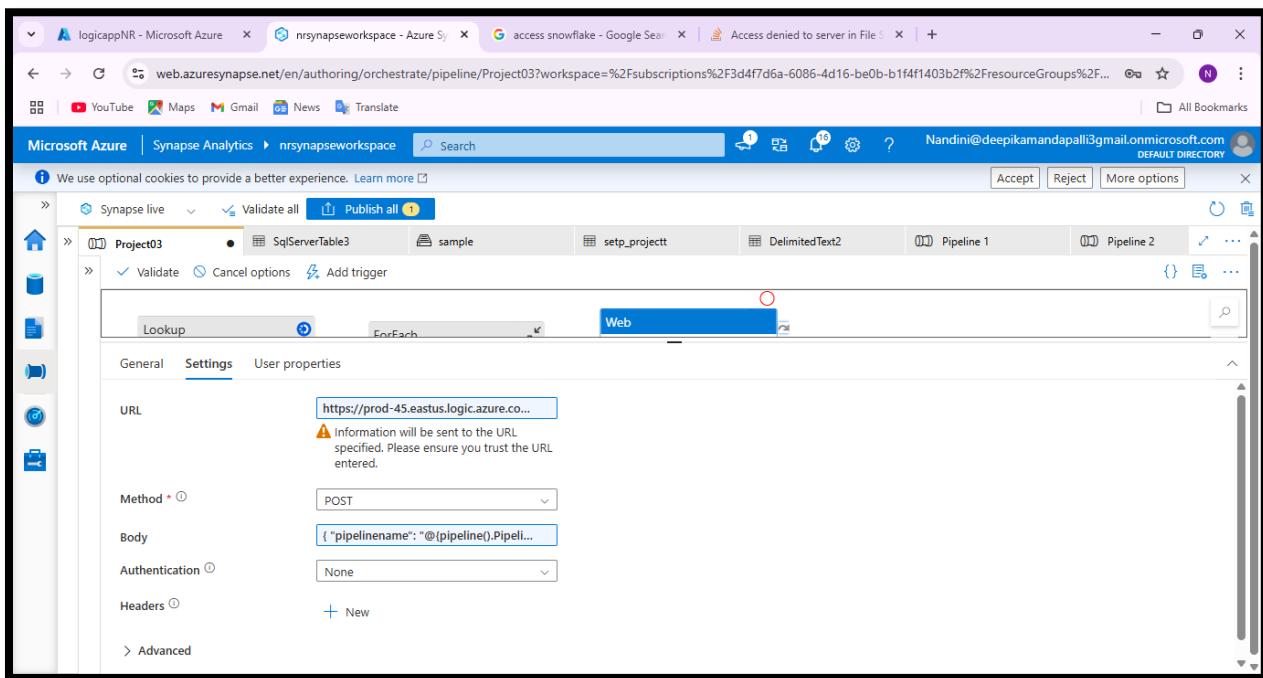
37. IN HTTP, ADD POST METHOD ,IN REQUEST BODY JSON SCHEMA → WRITE
PIPELINENAME,ERRORMESSAGE,TIMESTAMP,WORKSPACE IN JSON FORMAT THAT YOU WANT TO SEE IN
MESSAGE.



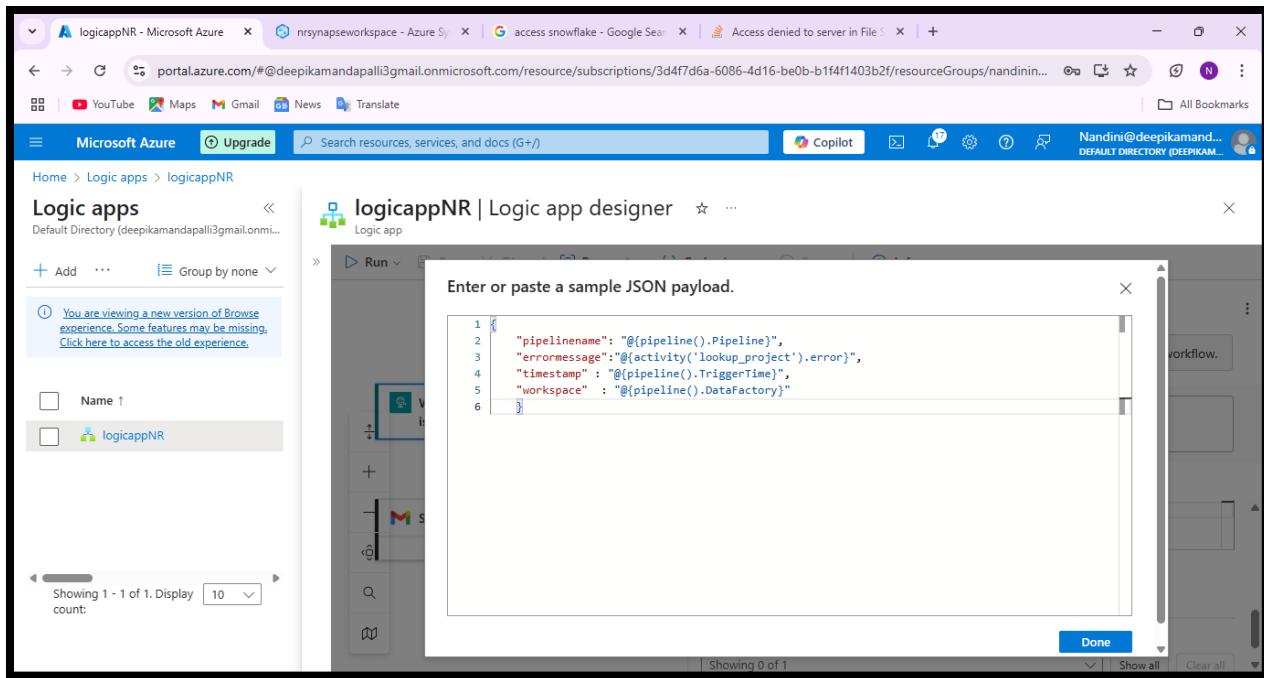
WRITE EMAIL ADDRESS TO WHOM YOU WANT WANT TO SEND A MESSAGE.SELECT PARAMETERS;IMPORTANCE,SUBJECT AND BODY.

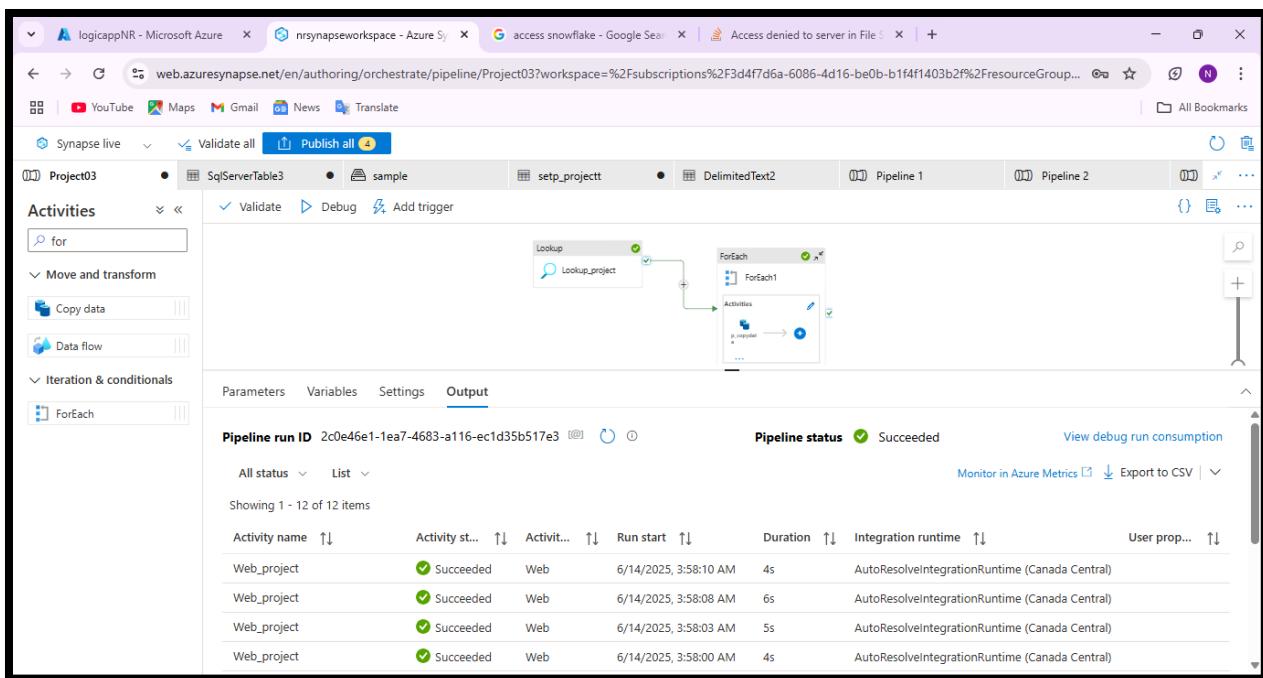


WRITE WHAT YOU WANT IN YOUR MESSAGE IN BODY. NOW SAVE THIS WORKFLOW. GO TO AZURE SYNAPSE.

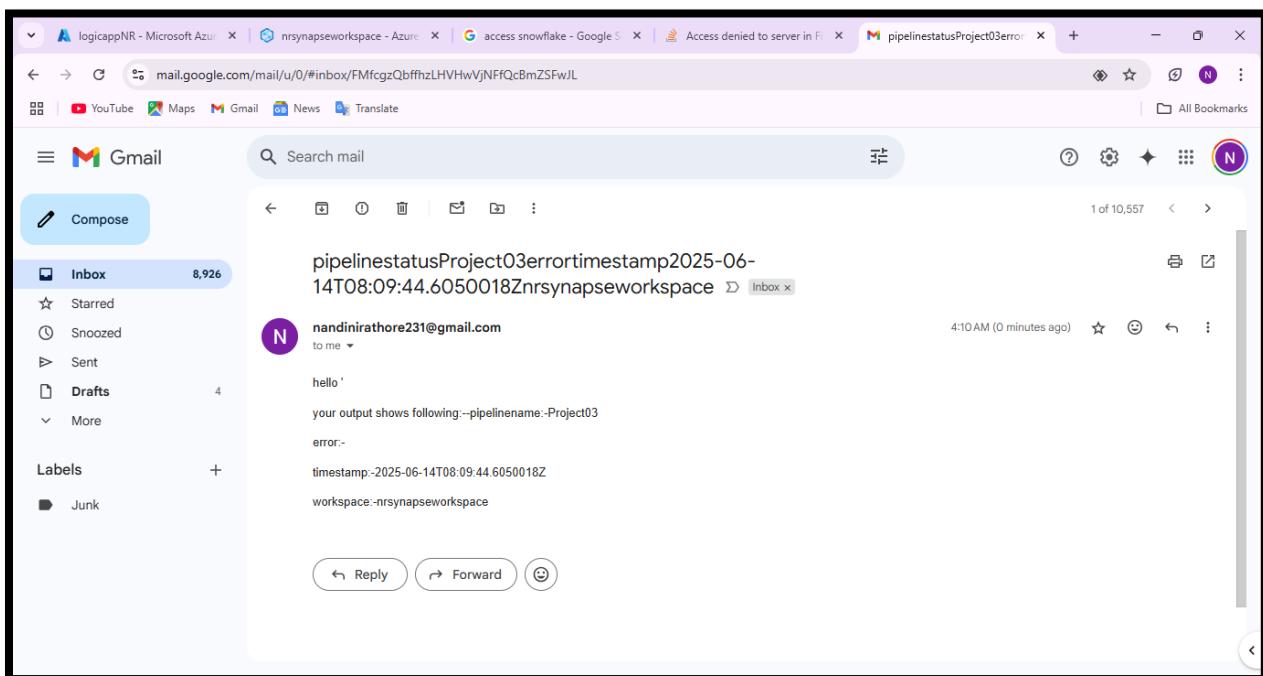


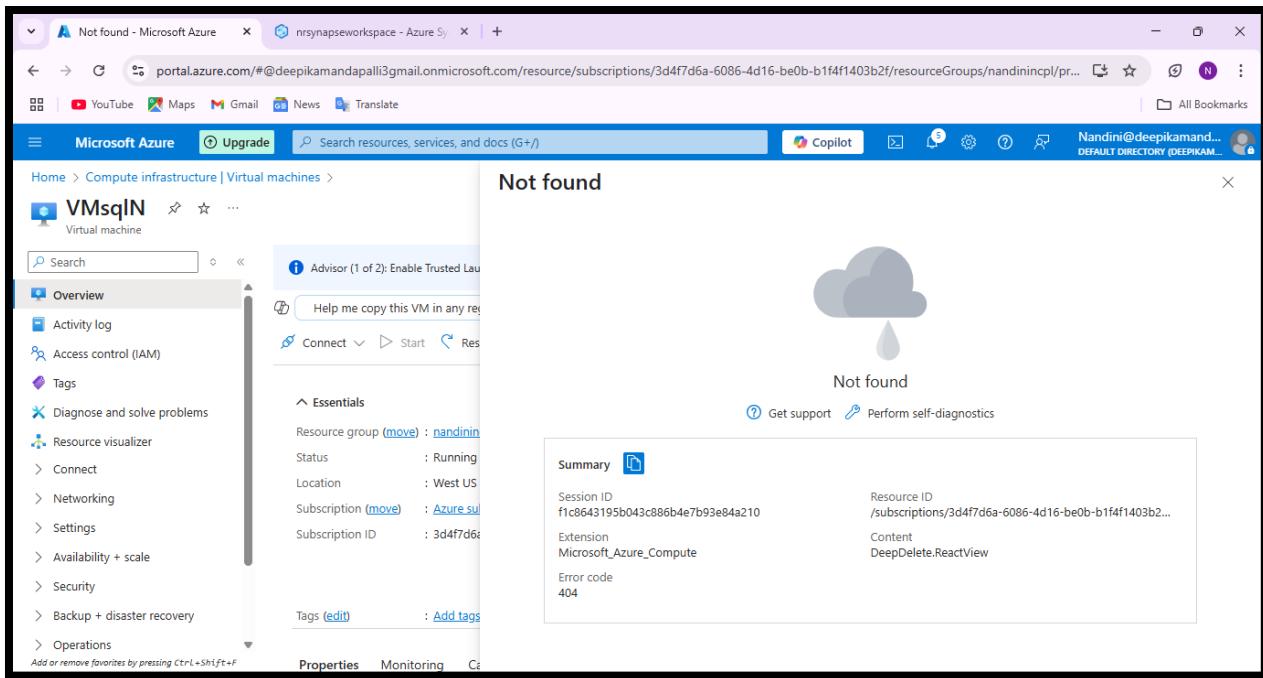
PASTE IN HTTP REQUEST WHERE JSON SCHEMA IS WRITTEN AND FROM THERE COPY URL AND PASTE THAT URL INTO URL OPTION OF WEB ACTIVITY.





I DID SOME MISTAKE MANUALLY SO THAT COPY ACTIVITY FAILS AND IT WILL SEND ME MAIL BY USING WEB ACTIVITY. SO HERE WEB ACTIVITY IS SUCCESSFUL AND SEND ME AN EMAIL OF PIPELINE FAILURE SHOWING PIPELINENAME,ERRORMESSAGE,TIMESTAMP ,WORKSPACE.





AFTER DOING ALL THE WORK DELETE VM.