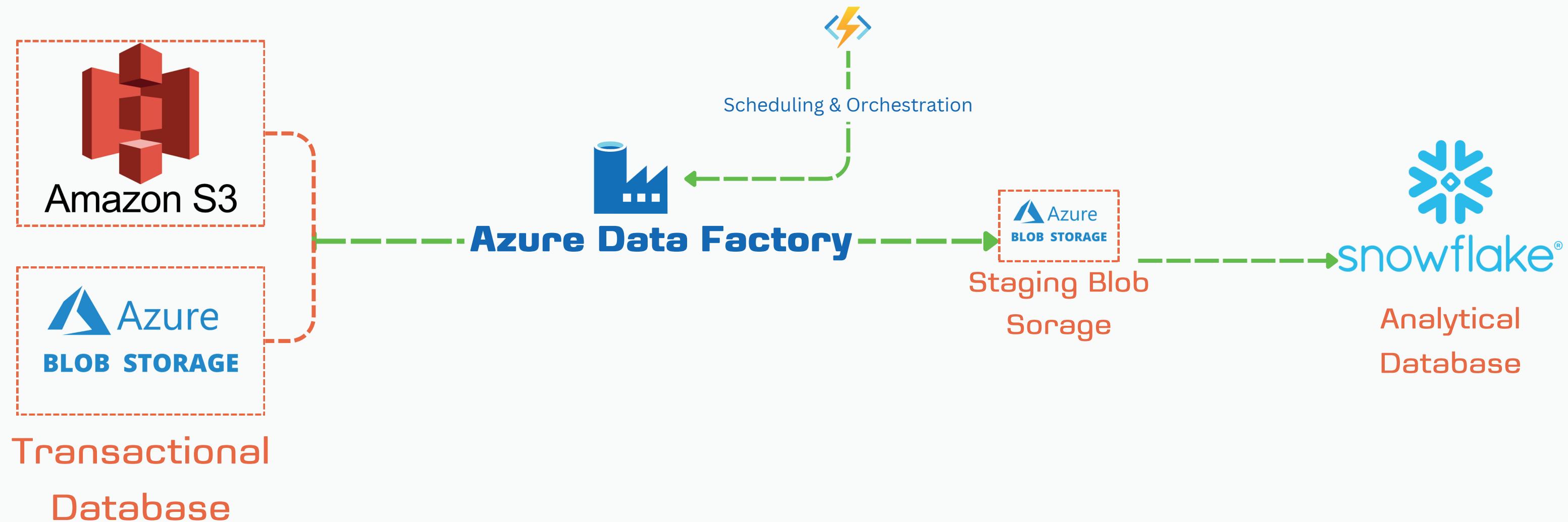


Multi-Cloud Data Integration



[in/prayagv](#)
[/prayag-verma](#)



Prayag Verma
Data Engineer

Probelm Statement

Companies use different cloud platforms to store data, which makes it hard to bring everything together, therefore,

In this project, we'll take “customers” and “orders” data from ‘AWS S3’ and ‘Azure Blob Storage’ and move it into Snowflake Warehouse using Azure Data Factory (ADF) and Trigger. We will integrate data from both clouds into one place for the business analysis.

We need the following to proceed with the plan:

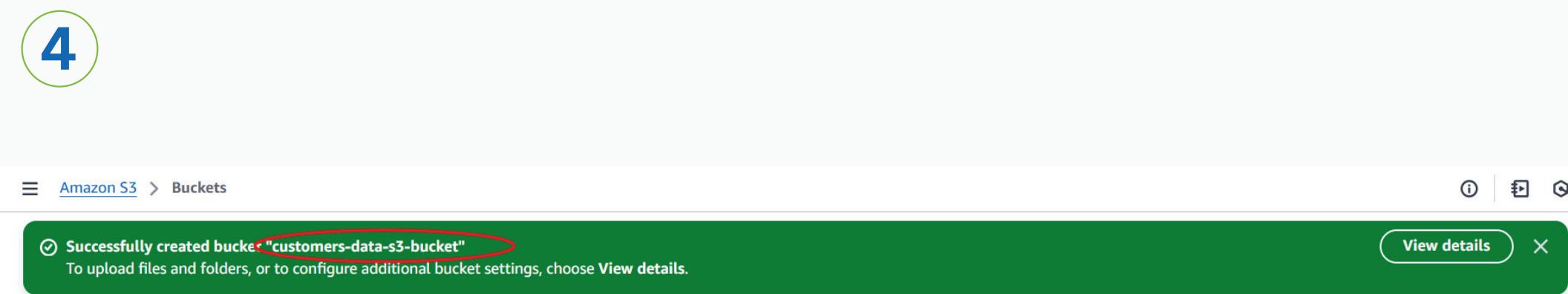
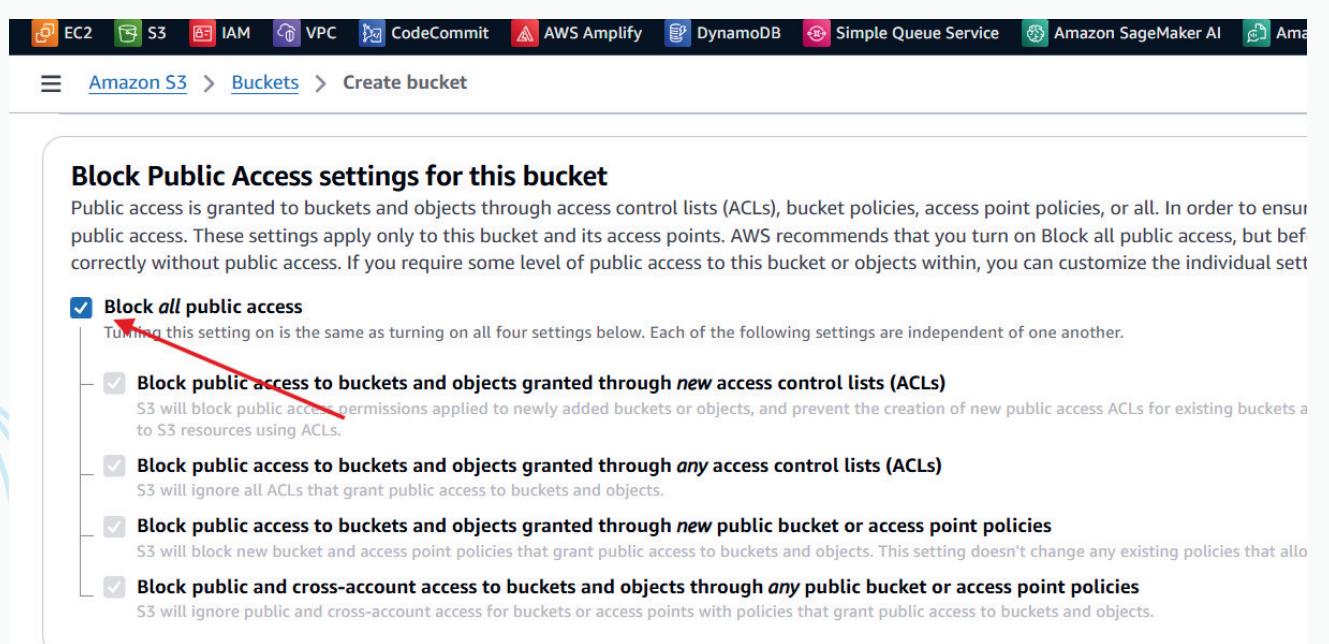
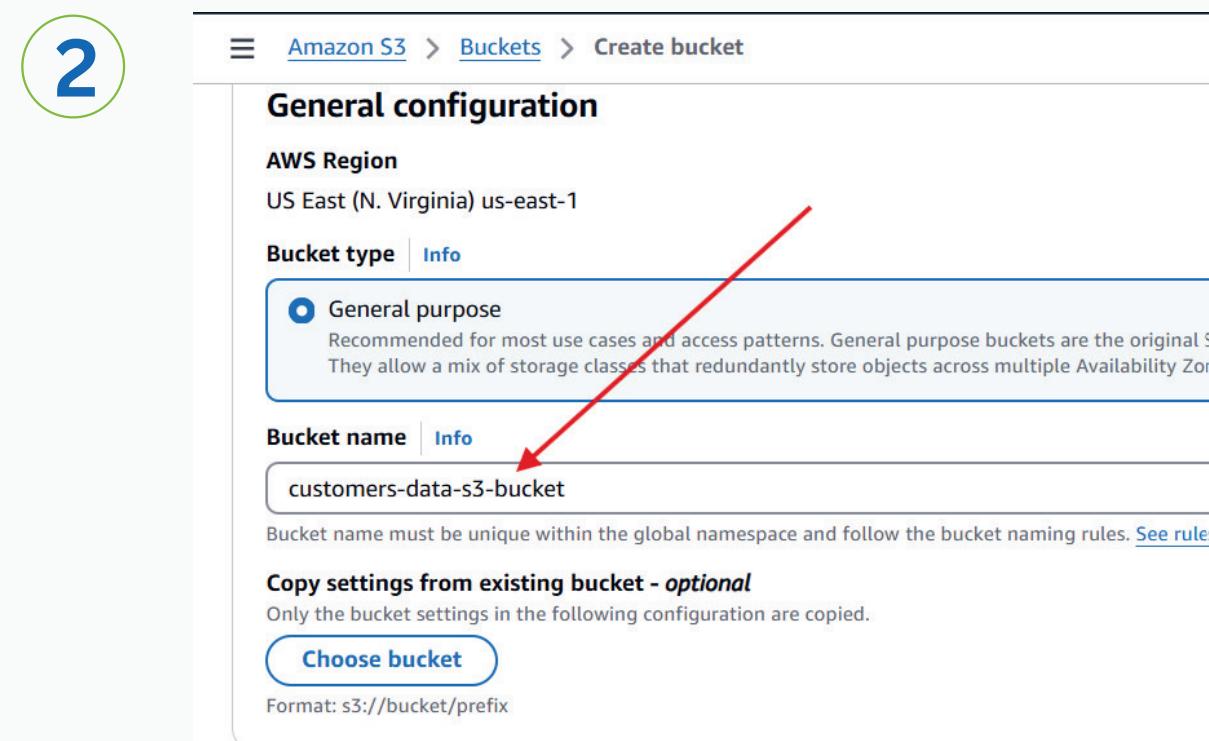
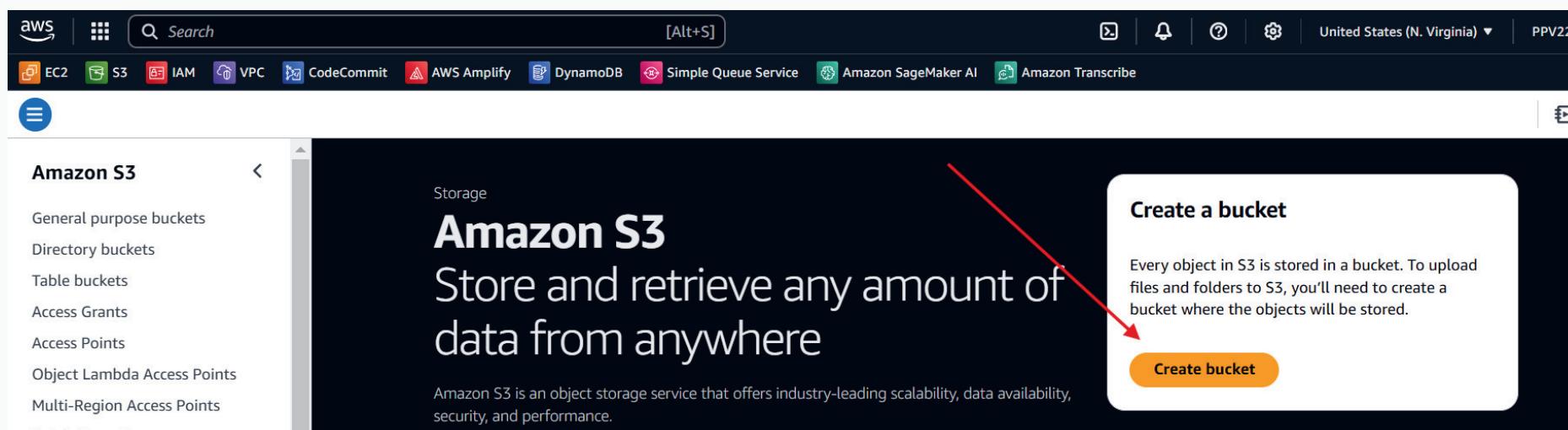
1. **AWS S3 Bucket (For customer data - customers.csv)**
Bucket Name: 'customers-data-s3-bucket'
File name: 'customers.csv'
1. **Azure Blob Storage (For order data - orders.csv)**
Containers Name: 'orders'
File Name: 'orders.csv'
1. **Snowflake Account**
Create tables: 'customers' and 'orders'
1. **Azure Data Factory (ADF)**
2. **ADF Trigger (For scheduling and orchestration)**

Step 2

Create S3 Bucket and Upload File

Step 2a: Create an S3 bucket and upload the customers.csv file.

- Create S3 Bucket:** Go to [AWS dashboard](#), search for 'S3' and select it> Click on 'Create bucket', enter a unique name (e.g., customers-data-s3-bucket), enable 'Block all public access for this bucket' option, and click on 'create bucket' button.
- Upload File:** Click on the bucket just created, then click 'Upload', then click on 'add files' and select 'customers.csv' file, and finely click on 'upload'. **Note:** Ensure the file is uploaded in the bucket.



Step 2

Create S3 Bucket and Upload File

Step 2b: Upload 'customers.csv' file to S3 bucket

1. **Upload File:** Click on the bucket just created, then click '**Upload**', then click on '**add files**' and select '**customers.csv**' file, and finely click on '**upload**'. **Note:** Ensure the file is uploaded in the bucket.

The screenshot shows the Amazon S3 console interface. The navigation bar at the top displays 'Amazon S3 > Buckets > customers-data-s3-bucket'. The main content area is titled 'customers-data-s3-bucket' with a 'Info' link. Below the title, there is a navigation bar with tabs: Objects (which is selected), Metadata, Properties, Permissions, Metrics, Management, and Access Points. Under the 'Objects' tab, there is a heading 'Objects (1)'. To the right of this heading are several buttons: a blue 'C' icon, 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Actions', 'Create folder', and an orange 'Upload' button. Below these buttons is a note: 'Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions.' There is also a 'Learn more' link. Below the note is a search bar with the placeholder 'Find objects by prefix'. The main table lists one object: 'customers.csv'. The table has columns: Name, Type, Last modified, Size, and Storage class. The 'customers.csv' entry shows 'customers.csv' in the Name column, 'csv' in the Type column, 'March 1, 2025, 00:38:31 (UTC-06:00)' in the Last modified column, '305.0 B' in the Size column, and 'Standard' in the Storage class column. The 'Name' column has a checkbox and a blue arrow pointing to the 'customers.csv' entry. The 'Type' column has a dropdown arrow. The 'Last modified' column has a dropdown arrow. The 'Size' column has a dropdown arrow. The 'Storage class' column has a dropdown arrow. The bottom right corner of the table has a dropdown arrow.

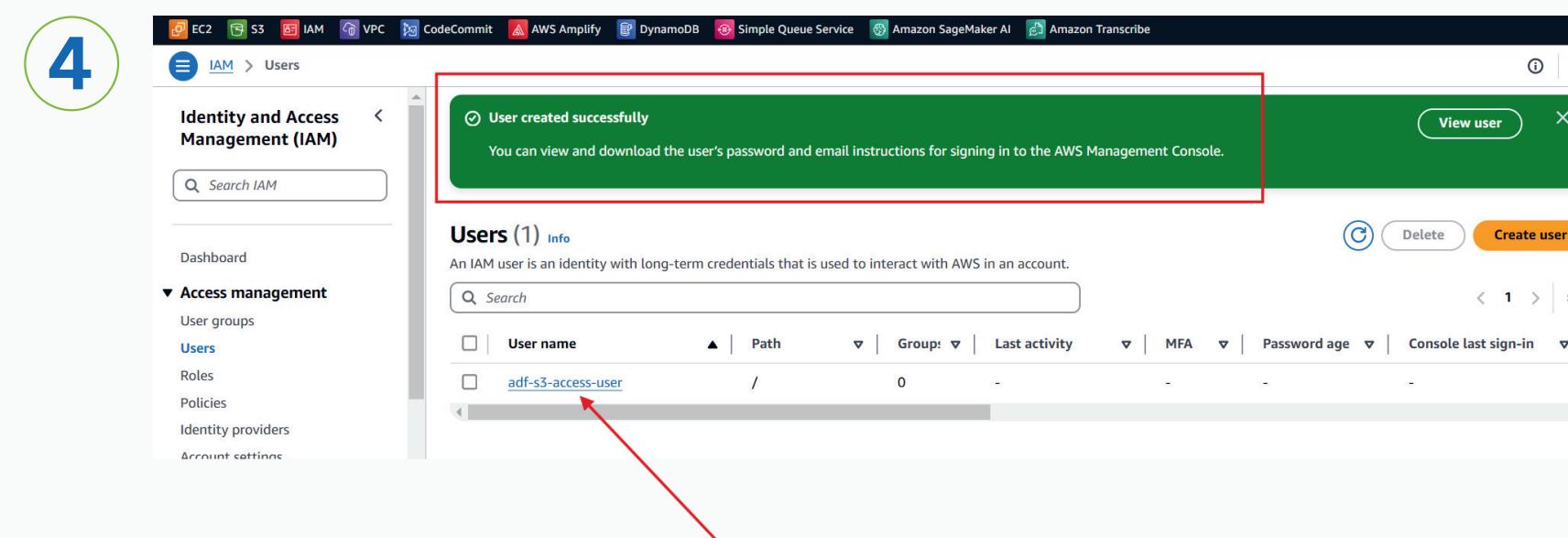
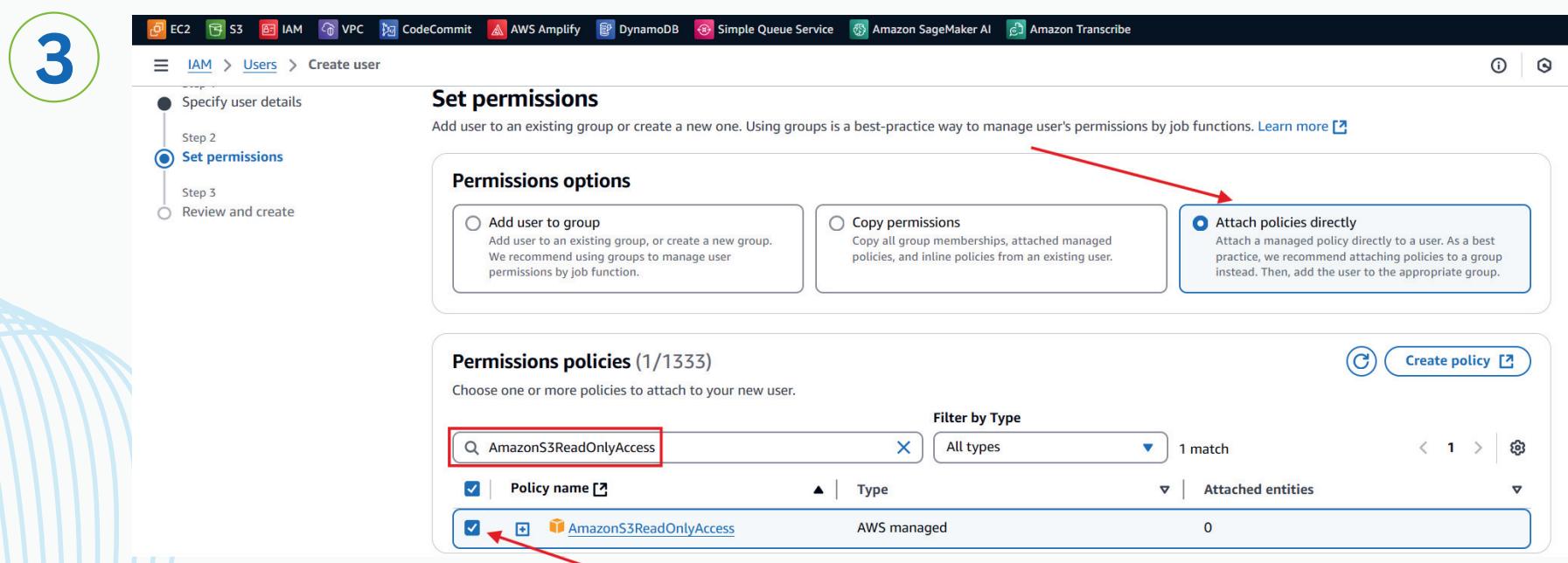
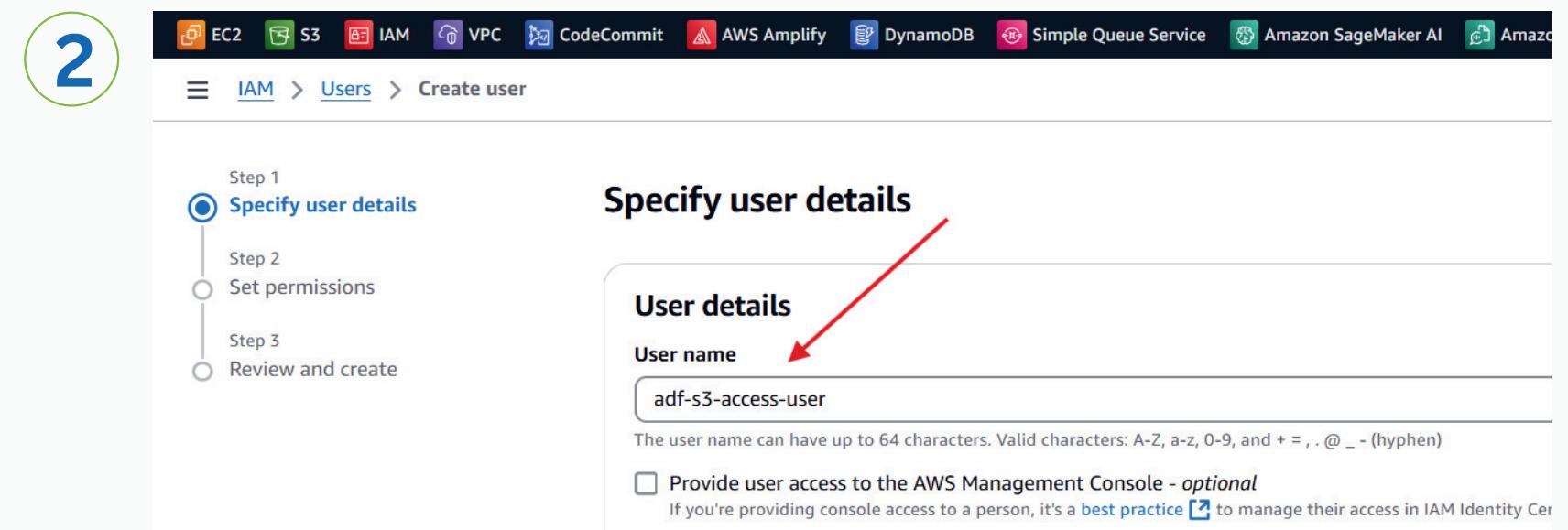
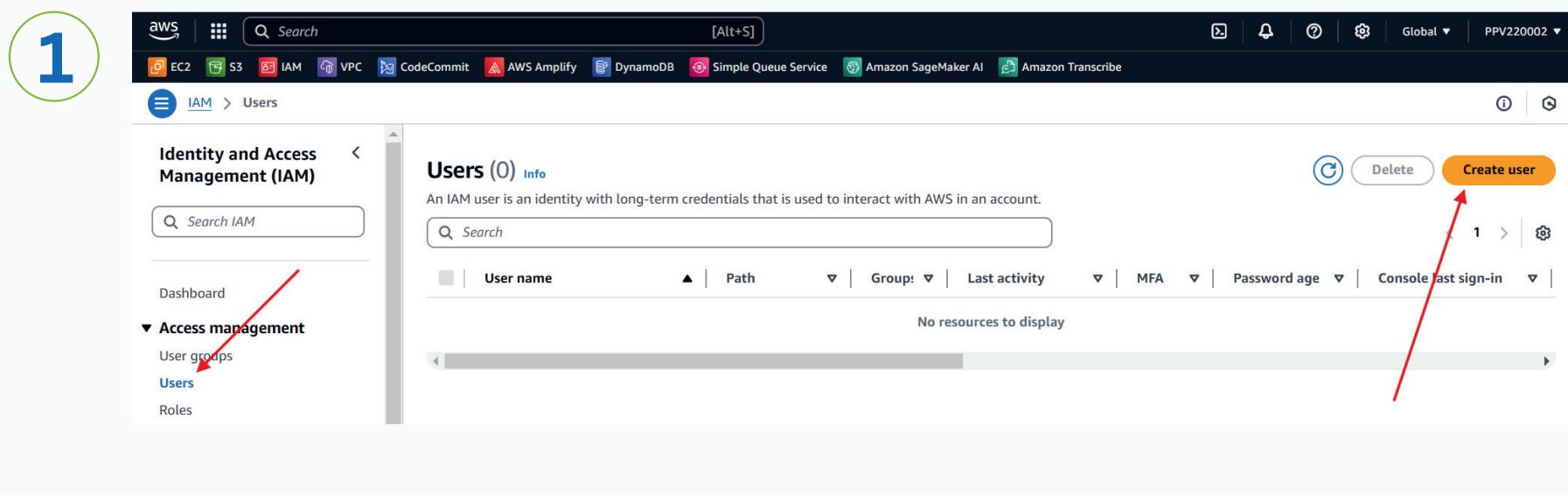
Name	Type	Last modified	Size	Storage class
customers.csv	csv	March 1, 2025, 00:38:31 (UTC-06:00)	305.0 B	Standard

Step 2

Create S3 Bucket and Upload File

Step 2c: To access this bucket from ADF(3rd Party), Create an IAM user credentials and assign a permission to it

1. **Create IAM User:** Go to **AWS dashboard**, search for '**IAM**' and click on in-> '**Users**' > then '**Create user**', enter a name (e.g., **adf-s3-access-user**), go to '**next**' step, Select '**Attach policies directly**' then search and select '**AmazonS3ReadOnlyAccess policy**', and create the user.



Step 2

Create S3 Bucket and Upload File

Step 2d: Next, Generate Access Keys (Programmatic Access)

1. **Generate Access Keys:** Go to 'IAM' dashboard, and select 'adf-s3-access-user' the one you just created, **Next**, Go to the 'Security credentials' tab and scroll to 'Access Keys' section, then click 'Create access key' and select 'Third-party service', and save the 'Access Key' and 'Secret Access Key' or download the **csv** file for your better.

1

Identity and Access Management (IAM) > Users > adf-s3-access-user

adf-s3-access-user info

Summary

ARN: arn:aws:iam::058264557200:user/adf-s3-access-user
Console access: Disabled
Created: March 01, 2025, 00:23 (UTC-06:00)
Last console sign-in: -

Access key 1: Create access key

Permissions | Groups | Tags | **Security credentials** | Last Accessed

Console sign-in

Console sign-in link: https://058264557200.signin.aws.amazon.com/console
Console password: Not enabled

Enable console access

2

Identity and Access Management (IAM) > Users > adf-s3-access-user

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. [Learn more](#)

Create access key

SSH public keys for AWS CodeCommit (0)

Actions | Upload SSH public key

3

IAM > Users > adf-s3-access-user > Create access key

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following

Use case

- Command Line Interface (CLI)
- Local code
- Application running on an AWS compute service
- Third-party service** (selected)
- Application running outside AWS

Step 2 - optional: Set description tag
Step 3: Retrieve access keys

4

Access key created

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 2 - optional: Set description tag
Step 3: Retrieve access keys

Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAQ3EGWS2IBOF4AW3P	E6YXH4ErLFJl/EYagI9arealPB3csT4sfY+/FdY1 Hide

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [best practices for managing AWS access keys](#).

Download .csv file | Done

Step 3

Create Azure Container & Upload File

Note: Create a storage account with name '**awsazstorageac**', and also create a container with name '**orders**' and then upload the '**orders.csv**' file to the same containers!

1

This screenshot shows the 'Containers' blade for the 'awsazstorageac' storage account. The left sidebar has a 'Containers' item highlighted with a red oval. The main area shows a list of containers with '\$logs' and 'orders' listed. A red arrow points from the top-left towards the '\$logs' container.

2

This screenshot shows the 'orders' container within the 'awsazstorageac' storage account. The 'Overview' tab is selected. A red arrow points from the bottom towards the 'orders.csv' blob listed in the table below.

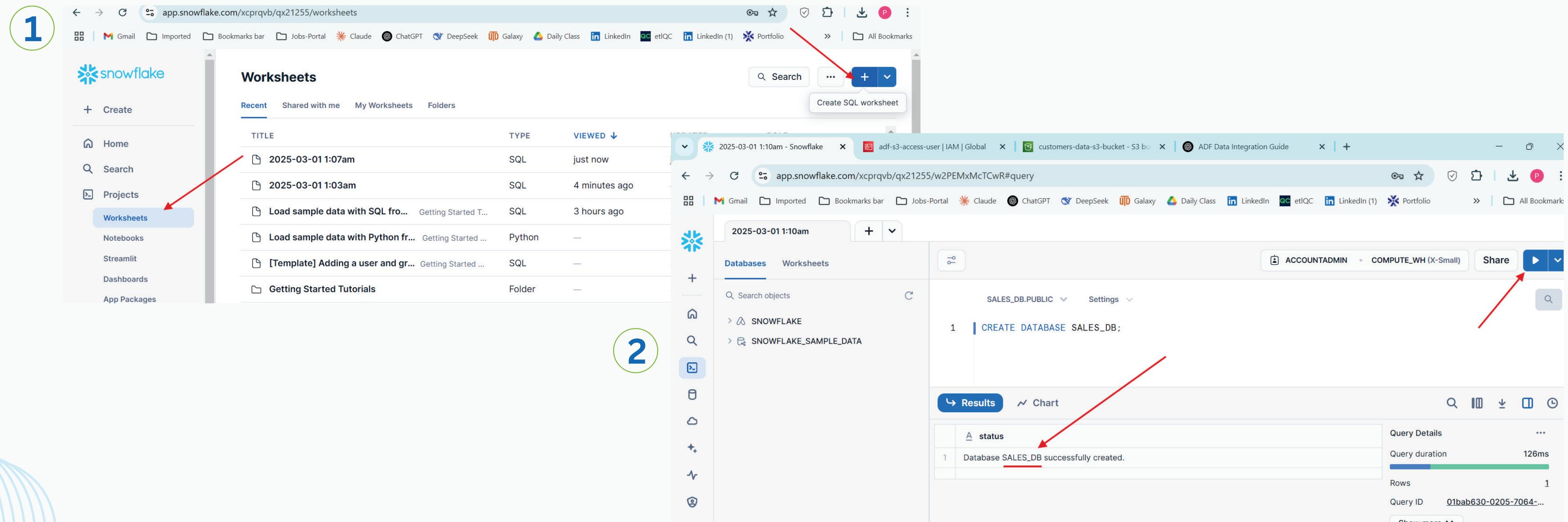
Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
orders.csv	3/1/2025, 12:44:26 AM	Hot (Inferred)		Block blob	149 B	Available

Step 4

Set Up Snowflake (Analytical DB)

Step 4a: Create a ‘Database’

Log in to snowflake - from homepage, click on ‘Project’ then ‘Worksheets’ Next, create a db using ‘CREATE DATABASE SALES_DB;’ and execute by clicking ‘Play’ icon visible at the top right corner position.



NOTE: You can also verify by executing ‘SHOW DATABASES;’

Step 4

Set Up Snowflake (Analytical DB)

Step 4b: Create a ‘Schema’:

A schema is a **logical container** for the **table**. Run the ‘`CREATE SCHEMA SALES_DB.SALES_SCHEMA;`’ SQL query to create a schema.

1

The screenshot shows the Snowflake Worksheet interface. A red box highlights the SQL command `CREATE SCHEMA SALES_DB.SALES_SCHEMA;`. A red arrow points from this command to the results section below, which displays the message "Schema SALES_SCHEMA successfully created." The top right corner of the interface shows the user has "ACCOUNTADMIN" privileges and is using "COMPUTE_WH (X-Small)" compute resources. The "Share" button is also visible.

```
CREATE SCHEMA SALES_DB.SALES_SCHEMA;
```

status

Schema SALES_SCHEMA successfully created.

2 NOTE: Verify the schema was created

The screenshot shows the Snowflake Worksheet interface. A red box highlights the SQL command `SHOW SCHEMAS IN DATABASE SALES_DB;`. A red arrow points from this command to the results section below, which lists the schemas in the database. The results table includes columns: created_on, name, is_defa, is_curre, database_nam, owner, and comment. It shows three rows: INFORMATION_SCHEMA, PUBLIC, and the newly created SALES_SCHEMA.

created_on	name	is_defa	is_curre	database_nam	owner	comment
2025-02-28 23:25:11.893 -0800	INFORMATION_SCHEMA	N	N	SALES_DB		Views describing the contents of schema
2025-02-28 23:12:52.040 -0800	PUBLIC	N	N	SALES_DB	ACCOUNTADMIN	
2025-02-28 23:23:59.640 -0800	SALES_SCHEMA	N	Y	SALES_DB	ACCOUNTADMIN	

Step 4

Set Up Snowflake (Analytical DB)

Step 4c: Create the ‘customers’ Table:

Run the following SQL command to create the customers table as shown in the screenshots

1

The screenshot shows the Snowflake UI interface. In the top navigation bar, the date is 2025-03-01 11:10am, and the user is ACCOUNTADMIN with COMPUTE_WH (X-Small) assigned. The left sidebar shows databases SALES_DB, SNOWFLAKE, and SNOWFLAKE_SAMPLE_DB. The main workspace is titled 'SALES_DB.SALES_SCHEMA'. A red box highlights the SQL code for creating the 'CUSTOMERS' table:

```
CREATE TABLE SALES_DB.SALES_SCHEMA.CUSTOMERS (
    CUSTOMER_ID INT PRIMARY KEY,
    FIRST_NAME STRING,
    LAST_NAME STRING,
    EMAIL STRING,
    PHONE STRING,
    ADDRESS STRING
);
```

The results pane shows the output: "Table CUSTOMERS successfully created."

2 **NOTE:** Verify the table ‘customers’ was created

The screenshot shows the Snowflake UI interface. The date is 2025-03-01 11:10am, and the user is ACCOUNTADMIN with COMPUTE_WH (X-Small) assigned. The left sidebar shows databases SALES_DB, SNOWFLAKE, and SNOWFLAKE_SAMPLE_DB. The main workspace is titled 'SALES_DB.SALES_SCHEMA'. A red box highlights the SQL command 'SHOW TABLES IN SCHEMA SALES_DB.SALES_SCHEMA;'. The results pane shows a table with one row, indicating the successful creation of the 'CUSTOMERS' table.

created_on	name	database_name	schema_name	kind	comment	cluster_size	row_count	by	owner
2025-02-28 23:30:38.685 -0800	CUSTOMERS	SALES_DB	SALES_SCHEMA	TABLE			0	0	ACCOUNTADMIN

Step 4

Set Up Snowflake (Analytical DB)

Step 4d: Create the 'orders' Table:

Run the following SQL command to create the customers table as shown in the screenshots

1

```
CREATE TABLE SALES_DB.SALES_SCHEMA.ORDERS (
    ORDER_ID INT PRIMARY KEY,
    CUSTOMER_ID INT,
    ORDER_DATE DATE,
    TOTAL_AMOUNT FLOAT,
    FOREIGN KEY (CUSTOMER_ID) REFERENCES SALES_DB.SALES_SCHEMA.CUSTOMERS(CUSTOMER_ID)
);
```

status
1 Table ORDERS successfully created.

2

NOTE: Verify the table 'orders' was created

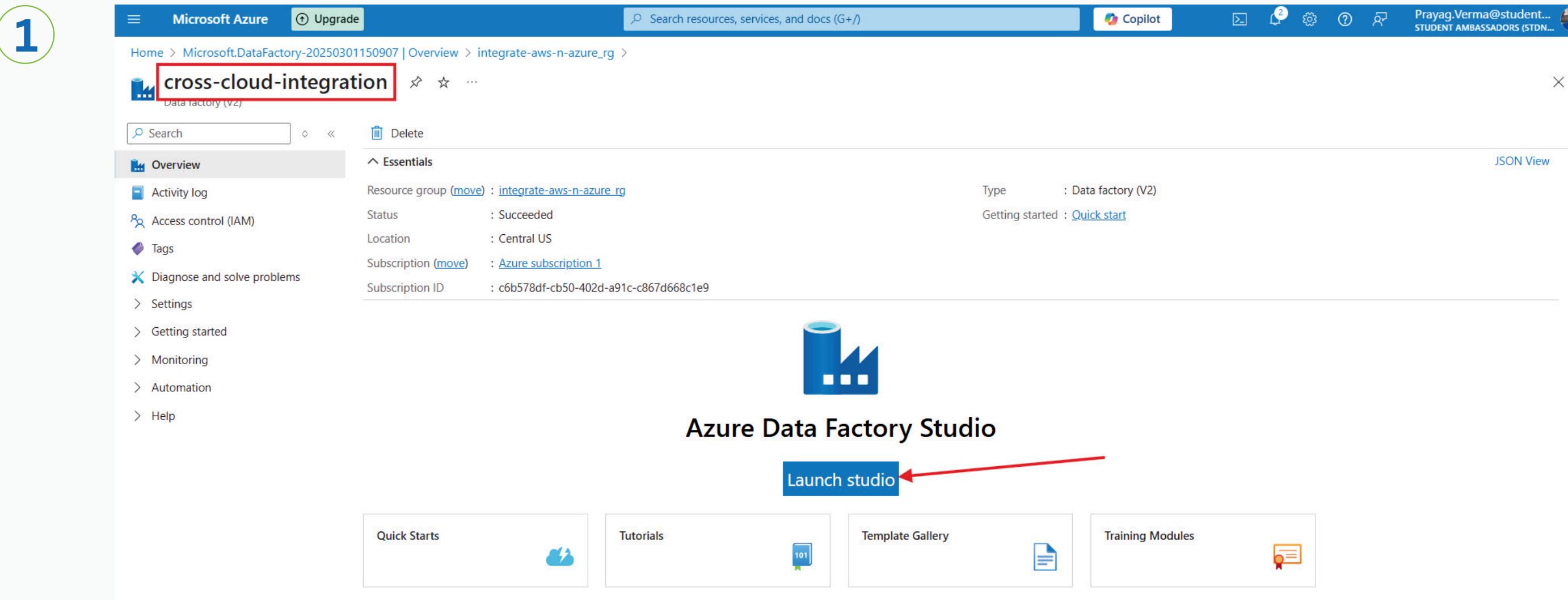
```
SHOW TABLES IN SCHEMA SALES_DB.SALES_SCHEMA;
```

created_on	name	database_name	schema_name	kind	commits	clusters	rows	bytes	owner
2025-02-28 23:30:38.685 -0800	CUSTOMERS	SALES_DB	SALES_SCHEMA	TABLE			0	0	ACCOUNTADMIN
2025-02-28 23:33:38.443 -0800	ORDERS	SALES_DB	SALES_SCHEMA	TABLE			0	0	ACCOUNTADMIN

Step 5

Create Linked Services In ADF

Step 5a: Navigate to the **Azure dashboard**, find **Data Factory**, and **create** a new data factory named '**cross-cloud-integration**.' Launch it and follow the steps provided in the **next slide**.



Step 5

Create Linked Services In ADF

Step 5b: For AWS S3 - Go to Azure Data Factory Studio → Manage → Linked Services → + New → Search for Amazon S3 and select it. Then, name it as ‘AmazonS3LinkedService’, enter details Access Key ID, Secret Access Key, Bucket name, Region, and then, Click ‘Test connection’ then Create.

1

Microsoft Azure | Data Factory > cross-cloud-integration

Home Author Monitor **Manage** Learning Center

General Factory settings Connections **Linked services** Integration runtimes Microsoft Purview Source control Git configuration

Search factory and documentation

Validate all Publish all

Linked services

Linked service defines the connection information to a data store or compute. Learn more

+ New Filter by name Annotations : Any

2

Microsoft Azure | Data Factory > cross-cloud-integration

Home Author Monitor **Manage** Learning Center

General Factory settings **Linked services** Integration runtimes Microsoft Purview Source control Git configuration ARM template

Search factory and documentation

Validate all Publish all

Linked services

Linked service defines the connection information to a data store or compute. Learn more

+ New Filter by name Annotations : Any

New linked service

Data store Compute

Amazon S3

All Azure Database File Generic protocol

Amazon S3 Amazon S3 Compatible

3

New linked service

Amazon S3 Learn more

Name * AmazonS3LinkedService

Description

Connect via integration runtime * AutoResolveIntegrationRuntime

Authentication type Access key

Access key ID Azure Key Vault

Access key ID * AKIAQ3EGWS2IBOF4AW3P

Secret access key Azure Key Vault

Secret access key *

Service URL https://s3.amazonaws.com

Test connection To linked service To file path

Connection successful

Create Back Test connection Cancel

Step 5

Create Linked Services In ADF

Step 5c: For Azure Blob Storage - Go to Azure Data Factory Studio → Manage → Linked Services → + New → Search for **Azure Blob Storage** and select it. Then, name it as '**AzureBlobLinkedService**', select Authentication type 'Account Key', Account name, and Account key, and then, Click '**Test connection**' then **Create**.

1

Microsoft Azure | Data Factory > cross-cloud-integration

Search factory and documentation

Home Author Monitor Manage Learning Center

Data Factory Validate all Publish all

General Factory settings Connections Linked services Integration runtimes Microsoft Purview Source control Git configuration

Linked services

Linked service defines the connection information to a data store or compute. Learn more

+ New Filter by name Annotations: Any

2

cross-cloud-integration

Search factory and documentation

Data Factory Validate all Publish all

General Factory settings Connections Linked services Integration runtimes Microsoft Purview Source control Git configuration ARM template

Linked services

Linked service defines the connection information to a data store or compute. Learn more

+ New Filter by name Annotations: Any

Showing 1 - 1 of 1 items

Name Type

AmazonS3LinkedService Amazon S3

Data store Compute

Azure Blob

All Azure Database

Azure Blob Storage

3

New linked service

Azure Blob Storage Learn more

Name * AzureBlobLinkedService

Description

Type ↑

Amazon S3

Connect via integration runtime * ⓘ

AutoResolveIntegrationRuntime

Authentication type

Account key

Connection string Azure Key Vault

Account selection method ⓘ

From Azure subscription Enter manually

Azure subscription ⓘ

Azure subscription 1 (c6b578df-cb50-402d-a91c-c867d668c1e9)

Storage account name *

awsazstorageac

Additional connection properties

+ New

Connection successful

Create Back Test connection Cancel

Step 5

Create Linked Services In ADF

Step 5d: For Snowflake- Go to Azure Data Factory Studio → Manage → Linked Services → + New → Search for **Snowflake** and select it. Then, name it as '**SnowflakeLinkedService**', select name, Account, Database, Schema, Warehouse, Authentication type, and User name, then, Click '**Test connection**' then **Create**.

The screenshot illustrates the process of creating a linked service in Azure Data Factory. It is divided into two main sections: a left panel and a right panel.

Left Panel (Step 1): Shows the 'Linked services' blade. The 'Type' column for the first item is highlighted with a red arrow. The items listed are 'AmazonS3LinkedService' and 'AzureBlobLinkedService'. Below the table, there is a 'New' button.

Right Panel (Step 2): Shows the 'New linked service' configuration page for Snowflake.

- Name:** SnowflakeLinkedService
- Connect via integration runtime:** AutoResolveIntegrationRuntime
- Account name:** XCPRQVB-QX21255
- Database:** SALES_DB
- Warehouse:** COMPUTE_WH
- Authentication type:** Basic
- User name:** PRAYAG
- Password:** (redacted)
- Role:** ACCOUNTADMIN
- Host:** contoso.snowflakecomputing.com

A red arrow points from the 'Test connection' button at the bottom right to a green checkmark icon indicating a successful connection. The 'Create' button is also visible at the bottom.

Note: If you don't know how to find these informations, refer to the next slide **Step 5e!**

Step 5

Create Linked Services In ADF

Step 5e: For Snowflake - Skip this slide if you've already set up & tested the **Snowflake Linked Services**!

Note:

1. To find **Account Name**, Go to **Snowflake Dashboard** → **Profile** → **Account** → **View Account Details**, then Copy '**Account Identifier**' and past in **Account Name**.

The screenshot shows the 'Account Details' section of the Snowflake dashboard. It lists various account settings. A red arrow points to the 'Account Identifier' field, which contains 'XCPRQVB-QX21255'. Another red arrow points to the 'User Name' field, which contains 'PRAYAG'. A third red arrow points to the 'Role' field, which contains 'ACCOUNTADMIN'. The left sidebar shows the user 'Prayag Verma' and their role 'ACCOUNTADMIN'.

3. To find **Warehouse**, Go to **Snowflake Dashboard** → **Admin** → **Warehouses**, then all warehouses is displayed.

The screenshot shows the 'Warehouses' section of the Snowflake dashboard. It lists two warehouses: 'COMPUTE_WH' (Standard, X-Small, Started) and 'SYSTEM\$STREAMLIT_NOTEBOOK...' (Standard, X-Small, Suspended). A red arrow points to the 'COMPUTE_WH' row.

2. To find **Database Name**, Go to **Snowflake Dashboard** → **Admin** → **Data**, then all DBs is displayed.

The screenshot shows the 'Databases' section of the Snowflake dashboard. It lists three databases: 'SALES_DB' (Local, OWNER: ACCOUNTADMIN, CREATED: 14 hours ago), 'SNOWFLAKE' (Share, OWNER: —, CREATED: 18 hours ago), and 'SNOWFLAKE_SAMPLE_DATA' (Share, OWNER: ACCOUNTADMIN, CREATED: 18 hours ago). A red arrow points to the 'SALES_DB' row.

4. To find **Database Schema**, Go to **Snowflake Dashboard** → **Admin** → **Data** → select <your_database> → Click on '**Schemas**' Tab.

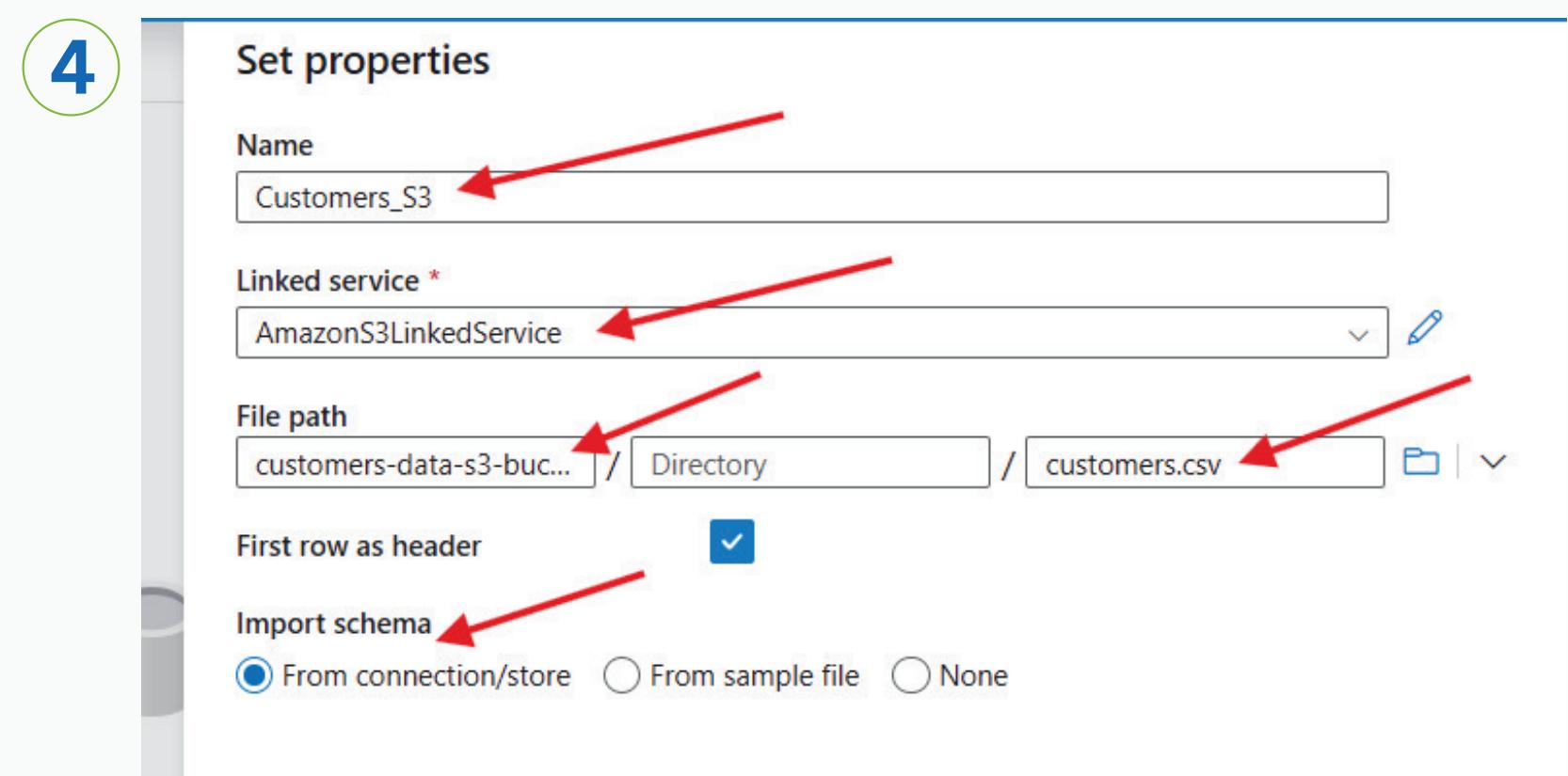
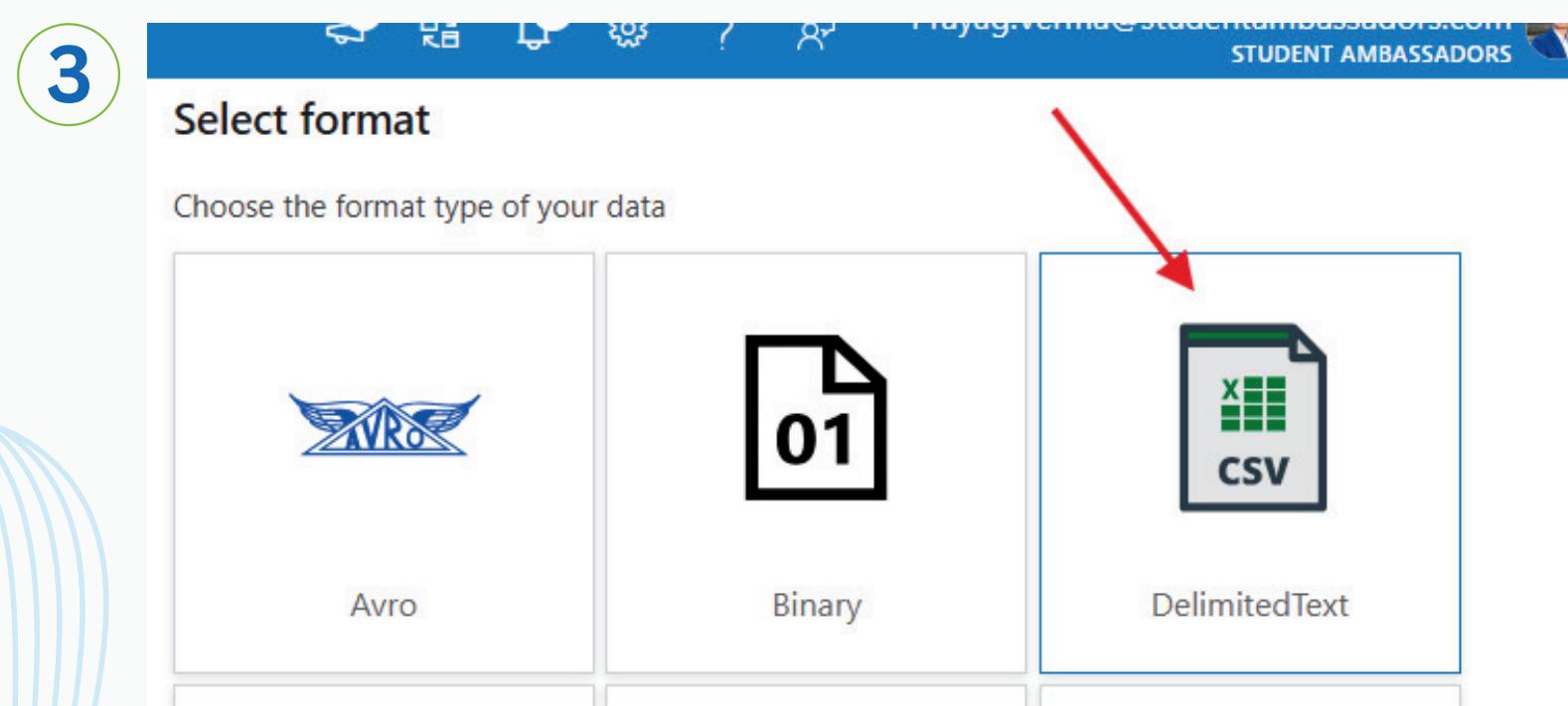
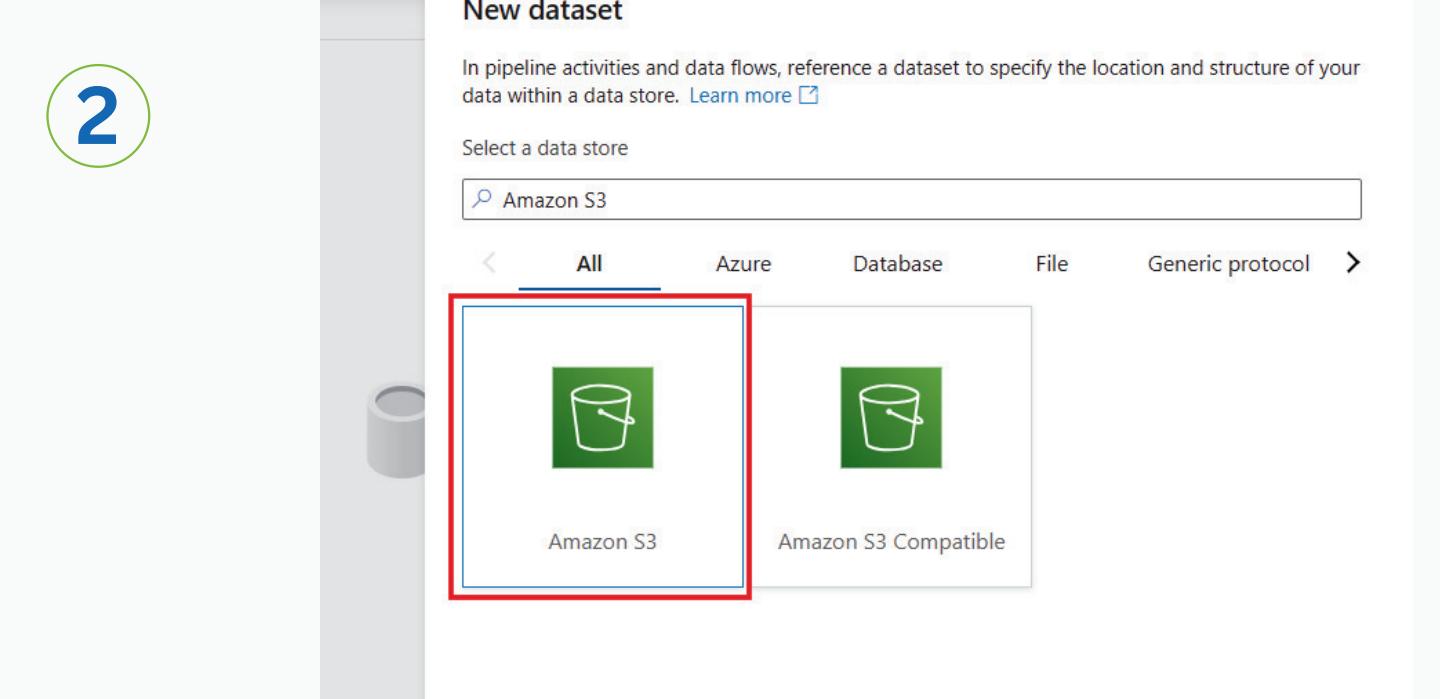
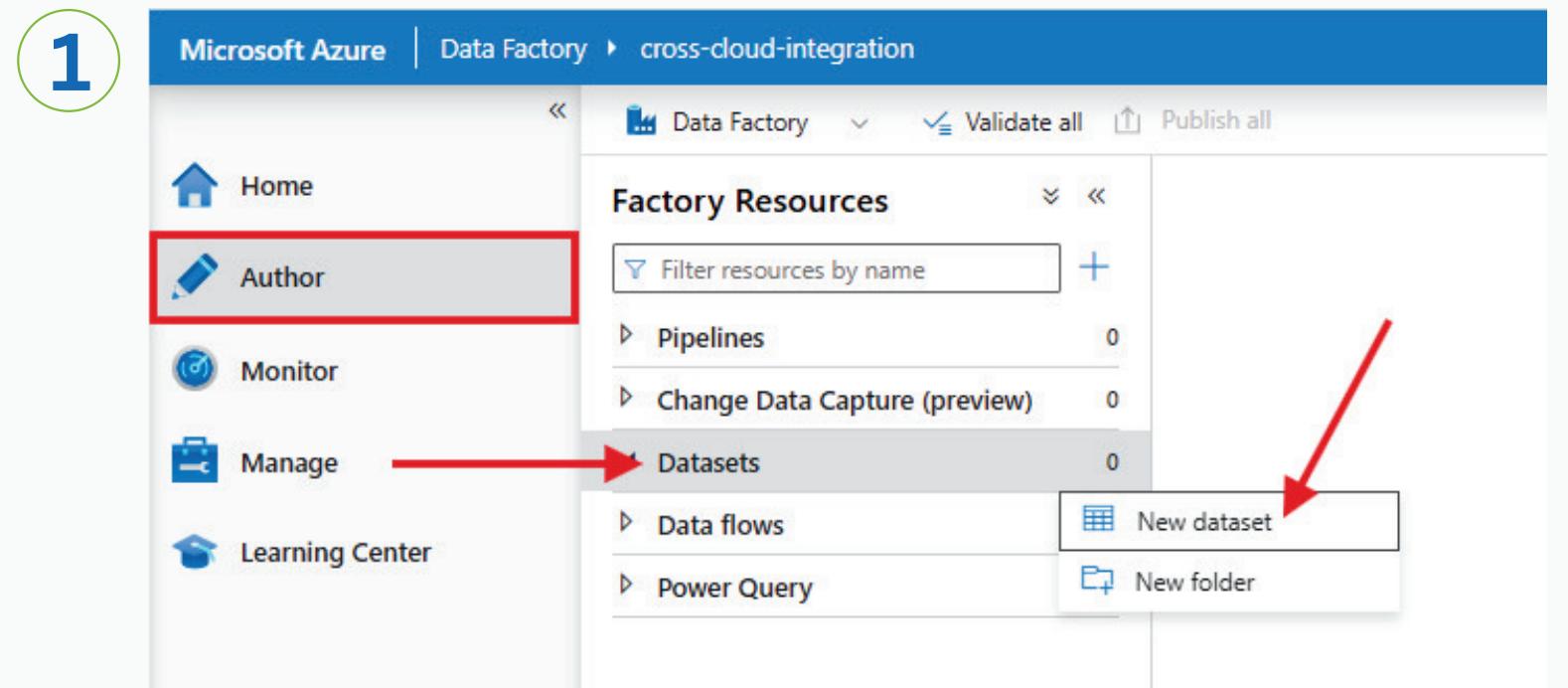
The screenshot shows the 'Schemas' tab for the 'SALES_DB' database. It lists three schemas: 'INFORMATION_SCHEMA', 'PUBLIC', and 'SALES_SCHEMA'. A red arrow points to the 'SALES_SCHEMA' schema. Below it, another red arrow points to the 'Schemas' tab in the navigation bar.

5. **User Name** and **Password** are the one you used to login to **Snowflake**

Step 6

Create AWS S3 Datasets In ADF

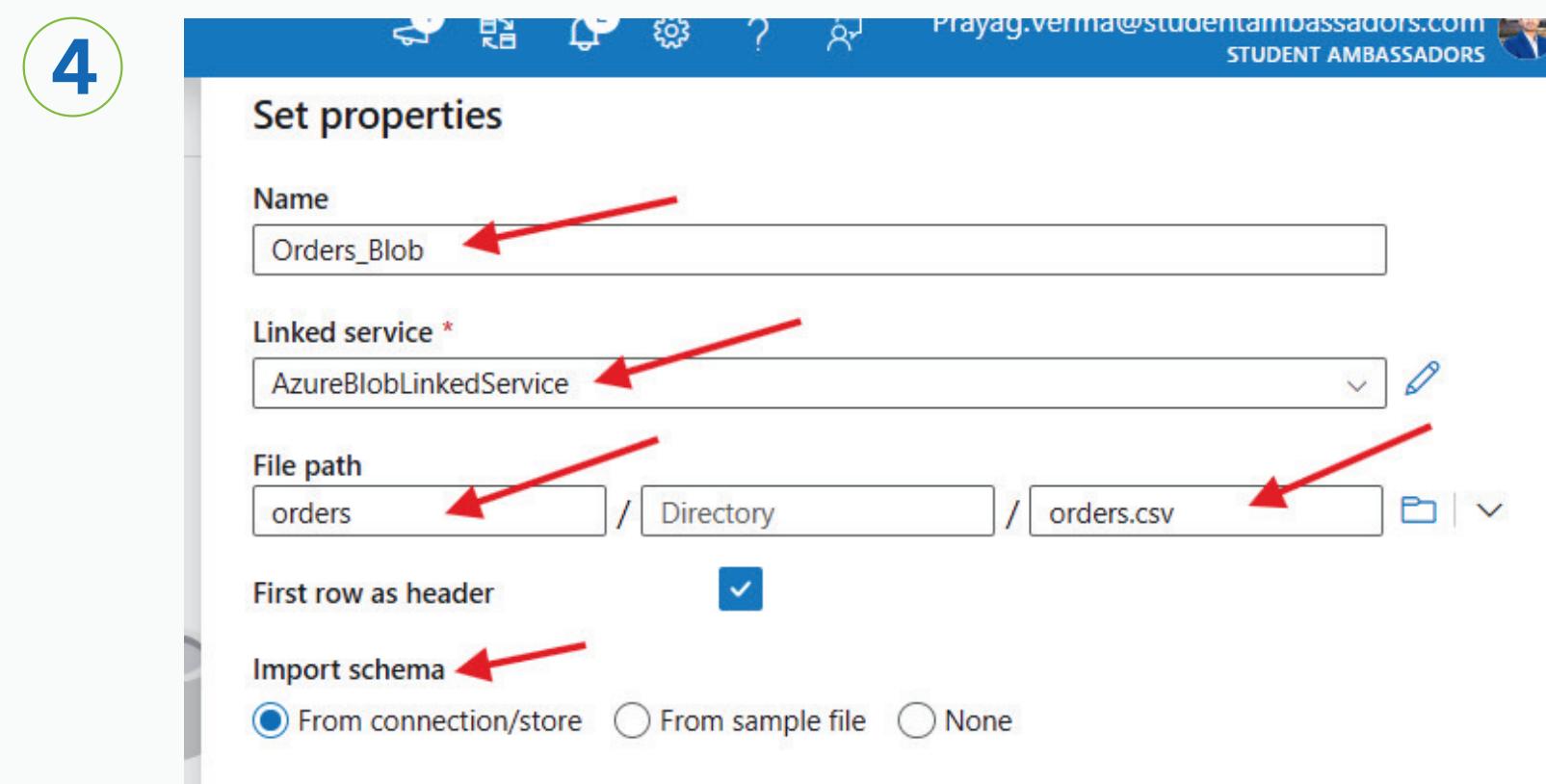
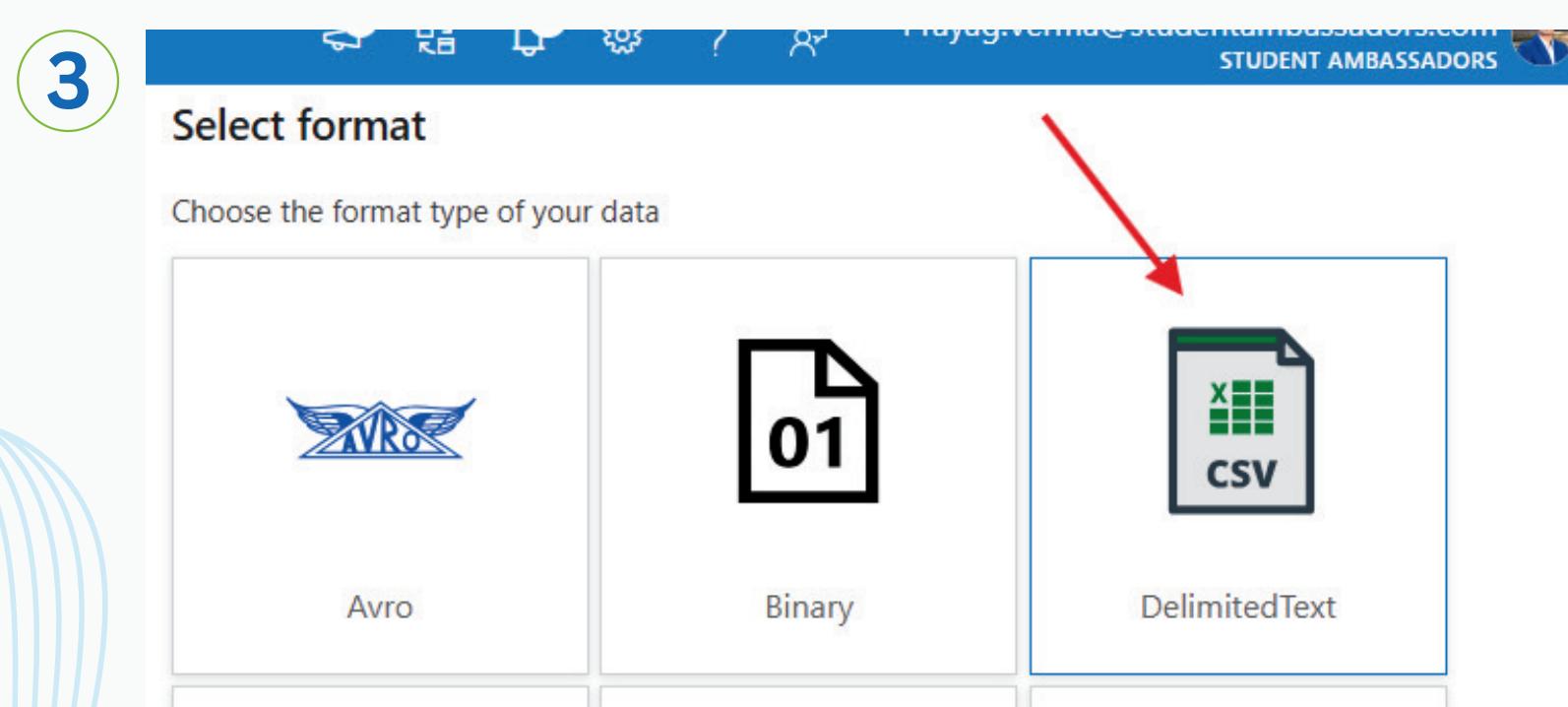
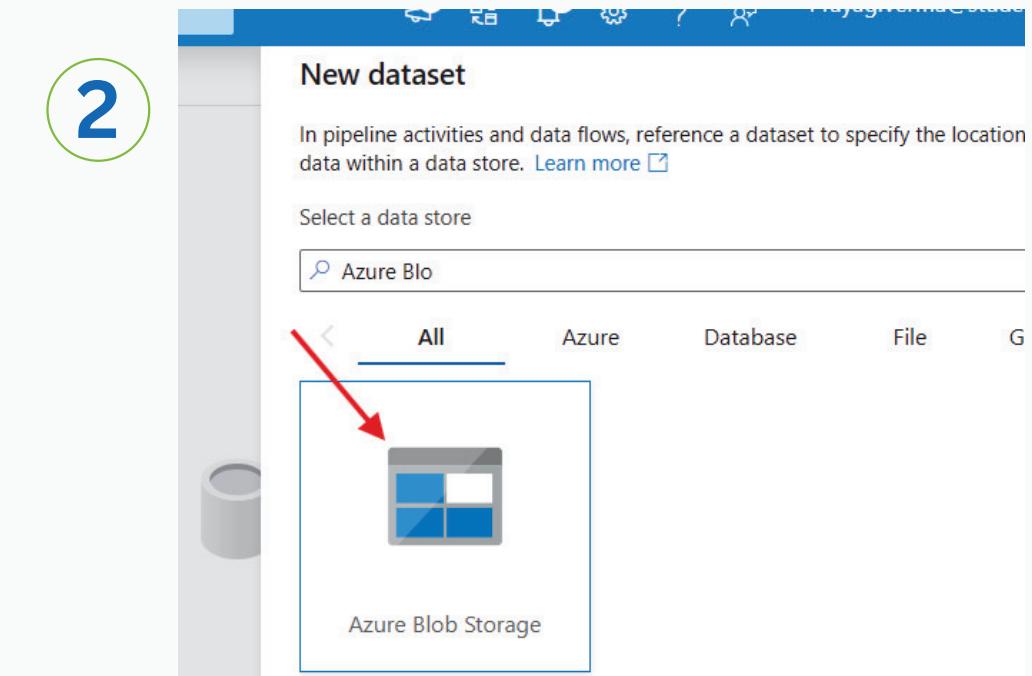
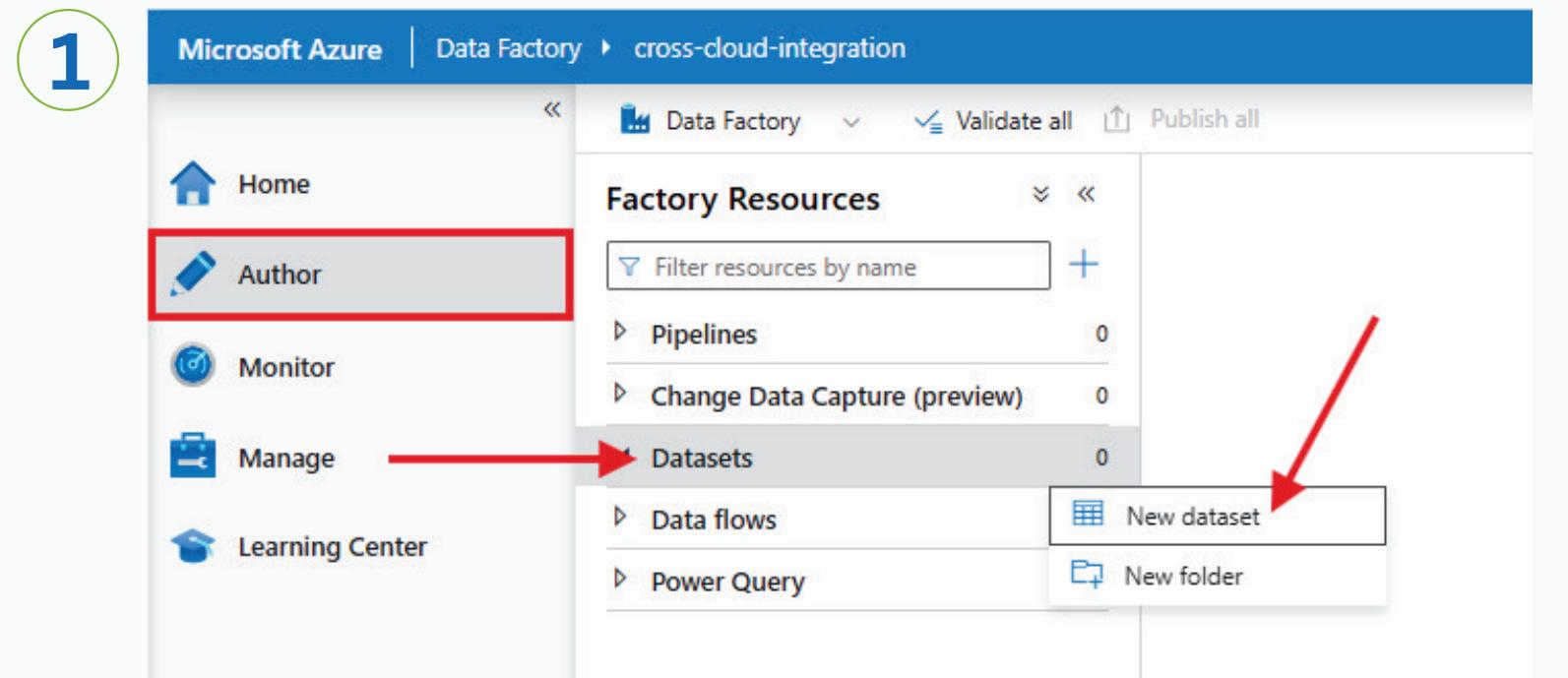
Step 6a: Go to Data Factory dashboard → Author → Datasets → New dataset → Search & select Amazon S3 → Select ‘Delimited Text’ CSV → and fill in all the informations as shown in screenshot 4 and Click OK.



Step 6

Create Azure Blob Datasets In ADF

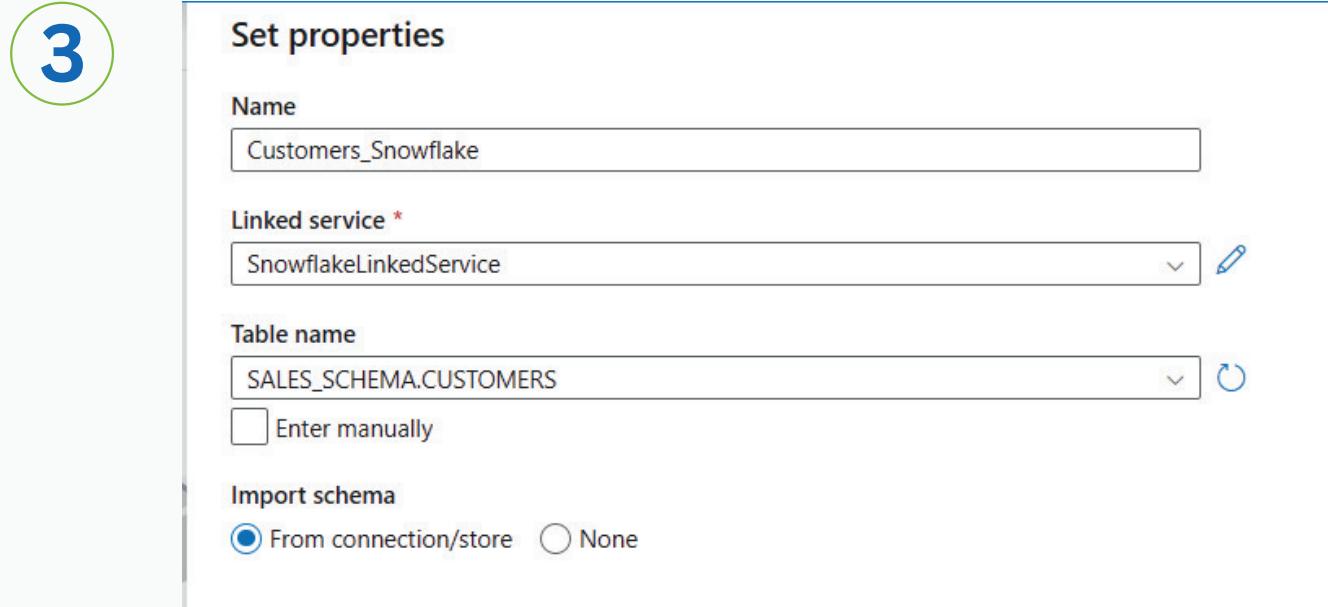
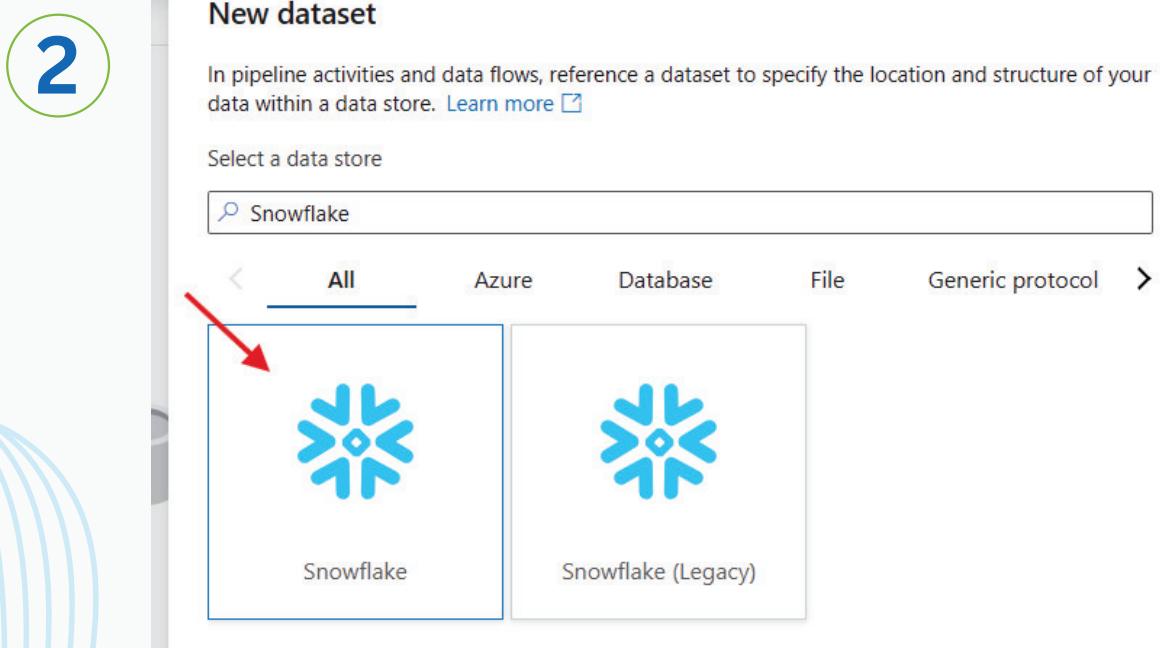
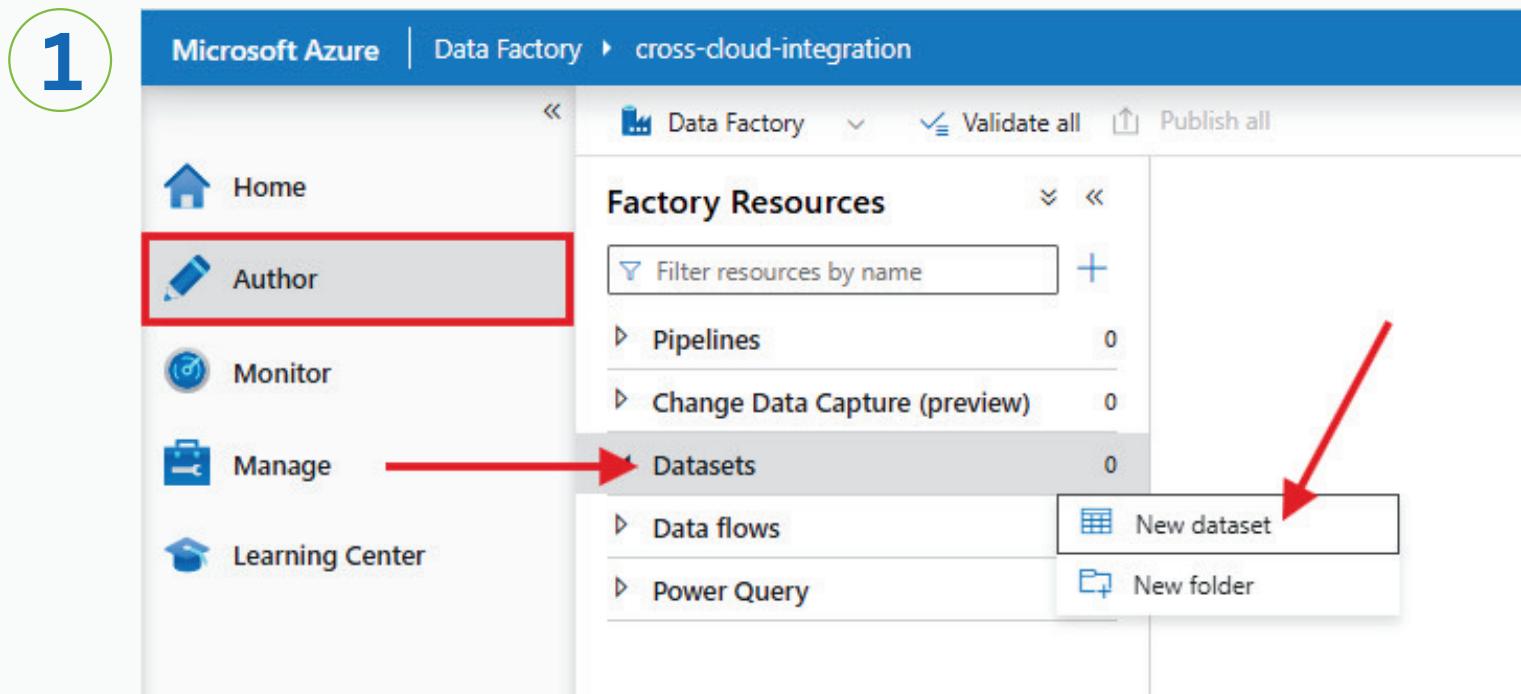
Step 6b: Go to Data Factory dashboard → Author → Datasets → New dataset → Search & select Amazon S3 → Select ‘Delimited Text’ CSV → and fill in all the informations as shown in screenshot 4 and Click OK.



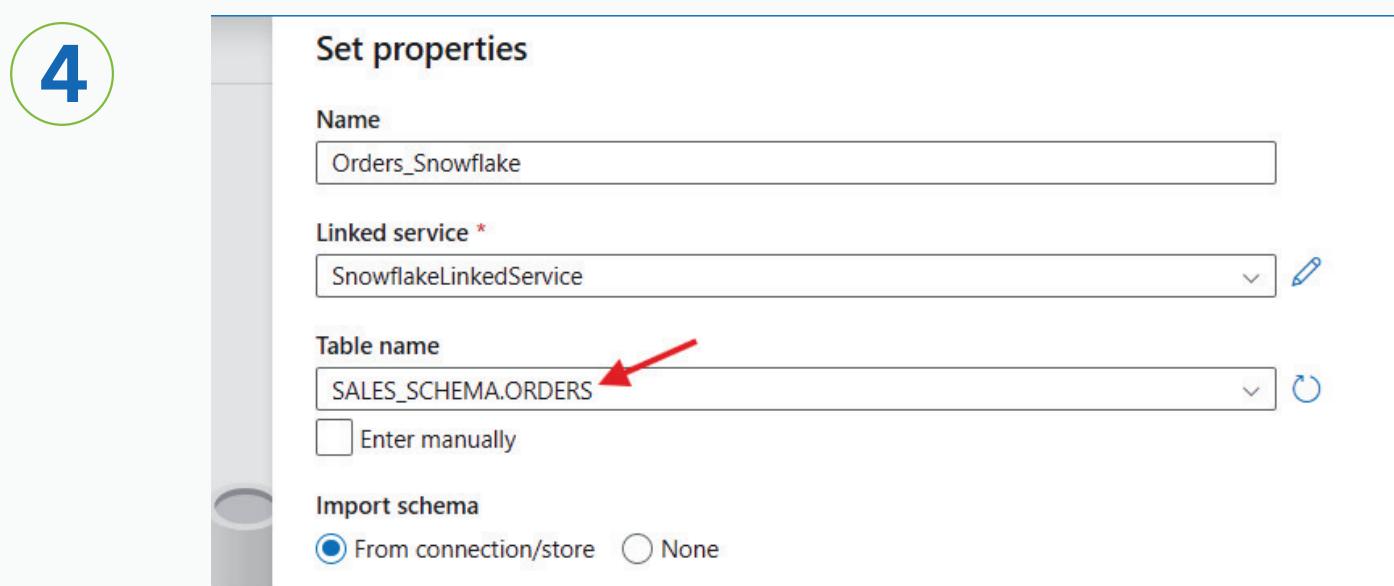
Step 6

Create Snowflake Datasets In ADF

Step 6c: Go to Data Factory dashboard → Author → Datasets → New dataset → Search & select **Snowflake** → enter Name ‘**Customers_Snowflake**’ → Select ‘**SnowflakeLinkedService**’ → Select Table name ‘**Customers**’ then Click OK.



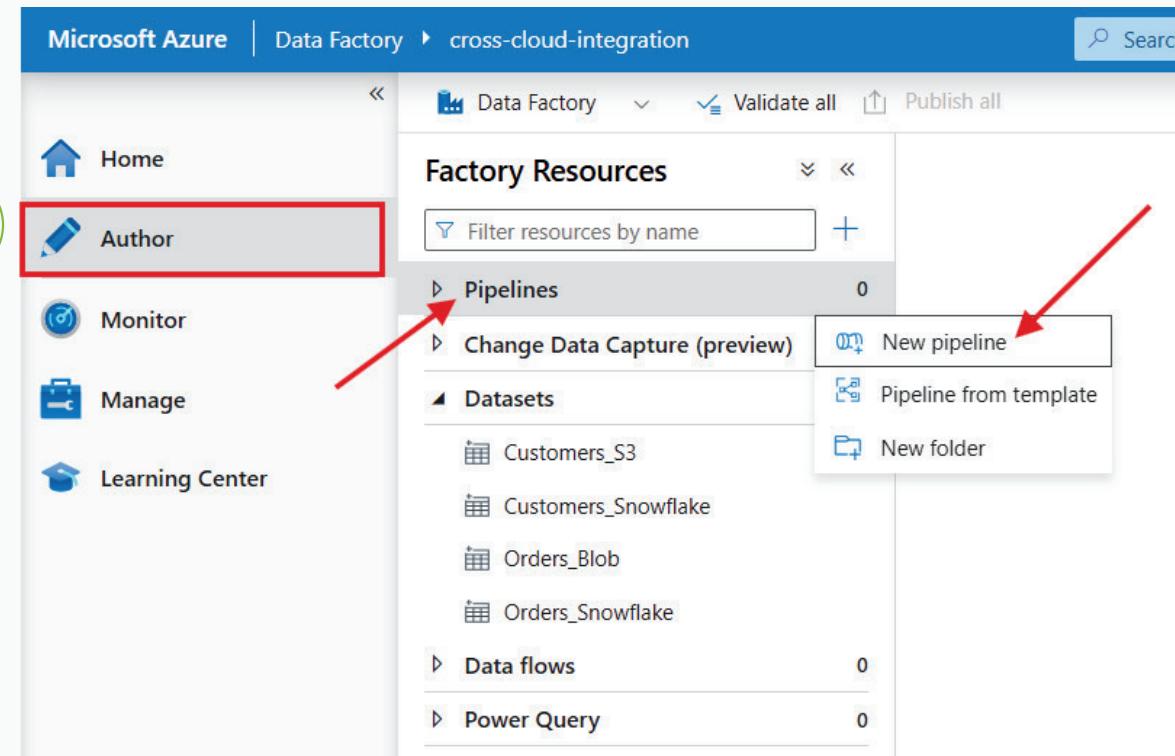
Step 6d: Repeat the same steps and create another **Datasets** with name ‘**Orders_Snowflake**’ and select **Orders** table in Table Name



Step 7

Create Azure Pipeline

Step 7c: Go to Data Factory dashboard → Author → Pipelines → New pipeline → Name it ‘S3_n_Azure_Blob_to_Snowflake’ → Add ‘Copy data’ activity as shown below.



The screenshot shows the 'Create Pipeline' interface. Step 1 is completed, and step 2 is shown. The 'Activities' pane on the left lists 'Copy data' under 'Move and transform'. The main area shows a 'Copy data' activity named 'Copy_Customers_s3'. The 'Properties' pane on the right shows the activity details: Name is 'S3_n_Azure_Blob_to_Snowflake', Description is 'Migrating flat files from AWS S3 Bucket and Azure Blob Storage into Snowflake warehouse!', and Activity state is 'Activated'. Red arrows point to the 'Name' field, the 'Description' field, and the 'Activity state' section.

Step 7

Create Azure Pipeline

Step 7c: Under Source tab, Select Customer_S3

1

The screenshot shows the Azure Data Factory pipeline editor. A 'Copy data' activity is selected. The 'Source' tab is active. In the 'Source dataset' dropdown, 'Customers_S3' is selected. A red arrow points to this dropdown. Other tabs visible include General, Sink, Mapping, Settings, and User properties.

2

The screenshot shows the Azure Data Factory pipeline editor. A 'Copy data' activity is selected. The 'Sink' tab is active. In the 'Sink dataset' dropdown, 'Customers_Snowflake' is selected. A red arrow points to this dropdown. Other tabs visible include General, Source, Mapping, Settings, and User properties.

Note:

- Under “Mapping” tab, Schemas should be imported if not import it and make sure it’s mapped properly.
- Here, data moves from AWS S3 **Customers.csv** → Snowflake’s **Customers** table.
- Now, do not close the current browser tab, and open a new tab in the same browser and see the next slide.

Step 7

Create Azure Pipeline

Step 7c: Now, go to [Azure dashboard](#) → resource group → select your container → Data storage → Containers → + Containers, name it as ‘staging’, once container is created go to the next slide:

The screenshot shows the Azure Storage account 'awsazstorageac' with the 'Containers' blade open. The left sidebar has 'Data storage' expanded, with 'Containers' selected. A red arrow points from the 'Containers' link in the sidebar to the '+ Container' button in the top bar. Another red arrow points from the '+ Container' button to the 'Name' field in the 'New container' dialog. The dialog shows 'staging' entered in the 'Name' field. The table below lists two existing containers: '\$logs' and 'orders', both with 'Last modified' dates of 2/28/2025.

Name	Last modified	Anonymous access level
\$logs	2/28/2025, 9:00:56 PM	Private
orders	2/28/2025, 11:04:46 PM	Private

Step 7

Create Azure Pipeline

Step 7c: Now click & open the container ‘staging’ the one you just created, then click on → **Shared access tokens** → select **read, Write, and List** permissions, then select → **Start and expiry date/time** → now Click on ‘**Generate SAS token and URL**’.

Note:

1. Copy ‘**Blob SAS token, Blob SAS URL**’ and **store** them somewhere as you will not be able to access it later.

1

The screenshot shows the 'Containers' blade for the 'awsazstorageac' storage account. A red arrow points to the 'staging' container in the list. The blade includes a search bar, a 'Container' button, and a 'Change access level' button. On the left, there's a sidebar with links like Overview, Activity log, Tags, Diagnose and solve problems, Access Control (IAM), Data migration, and Events.

The screenshot shows the 'Shared access tokens' blade for the 'staging' container. A red box highlights the 'Shared access tokens' tab. Another red box highlights the 'Permissions' section where 'Read', 'Write', and 'List' checkboxes are checked. A third red box highlights the 'Generate SAS token and URL' button at the bottom. Below the blade, a modal window also displays the generated SAS token and URL.

Home > integrate-aws-n-azure_rg > awsazstorageac | Containers > staging

staging | Shared access tokens

Search

Overview

Diagnose and solve problems

Access Control (IAM)

Settings

Shared access tokens

Signing method

Account key (selected) User delegation key

Key 1

Stored access policy

None

Access policy

Properties

Metadata

Permissions * 3 selected

Read (checked)

Add

Create

Write (checked)

Delete

List (checked)

Immutable storage

8:27:48 PM (US & Canada)

4:27:48 AM (US & Canada)

Allowed IP addresses

for example, 168.1.5.65 or 168.1.5.65-168.1....

Allowed protocols

HTTPS only (selected) HTTPS and HTTP

Generate SAS token and URL

Generate SAS token and URL

Blob SAS token

sp=rwl&st=2025-03-04T02:27:48Z&se=2025-03-07T10:27:48Z&spr=https&sv=2022-11-02&sr=c&sig=Upcx9Wh8qo0jmHYy3suaNQfqn9SNicgiaQ%2BCRejTnQE%3D

Blob SAS URL

<https://awsazstorageac.blob.core.windows.net/staging?sp=rwl&st=2025-03-04T02:27:48Z&se=2025-03-07T10:27:48Z&spr=https&sv=2022-11-02&sr=c&sig=Upcx9Wh8qo0jmHYy3suaNQfqn9SNicgiaQ%2BCRejTnQE%3D>

Step 7

Create Azure Pipeline

Step 7c: Now, Return to the Azure Data Factory tab. Under Settings → ensure Enable staging is selected → enter ‘Storage path’ as ‘staging’ → Click ‘+ New’, name it **Ls_staging_blob_storage**, and choose **SAS URI** for authentication type. Paste the **Blob SAS URL** (saved earlier refer to [page no 24](#)) into the SAS URL field → Click on **Test connection** if success click on ‘Create’ else re confirm the **SAS URL** properly.

The screenshot shows the 'Settings' tab of a pipeline configuration. Key settings include:

- Maximum data integration unit: Auto
- Degree of copy parallelism: Auto
- Fault tolerance: Auto
- Enable logging: Unchecked
- Enable staging: Checked
- Staging account linked service: Ls_staging_blob_storage
- Storage path: staging
- Enable compression: Unchecked

The screenshot shows the 'New linked service' dialog for Azure Blob Storage. The configuration includes:

- Name: Ls_staging_blob_storage
- Type: Azure Blob Storage
- Connect via integration runtime: AutoResolveIntegrationRuntime
- Authentication type: SAS URI
- SAS URL: https://awsazstorageac.blob.core.windows.net/staging?sp=rwl&st=2025-03-04T02:27:48Z

Now, Click on ‘Publish all’ and save the changes.

Step 7

Create Azure Pipeline

Step 7c: Similarly add another ‘Copy Data’ activity for ‘Orders’

1 Under Source tab, Select Orders_Blob

The screenshot shows the Azure Pipeline interface with the pipeline name 'pl_S3_n_Azure_Blob...' at the top. The 'Activities' pane on the left lists 'Copy data' and 'Data flow'. The main canvas shows two 'Copy data' activities connected by a green arrow. The bottom section is the 'Source' tab for the second 'Copy data' activity. It has fields for 'Name' (set to 'Copy Orders Azure Blob'), 'Source dataset' (set to 'Orders_Blob'), and other settings like 'File path type' (set to 'File path in dataset'). A red arrow points to the 'Source dataset' field.

2 Under Sink tab, Select Orders_Snowflake

The screenshot shows the 'Sink' tab for the 'Copy Orders Azure Blob' activity. It has fields for 'Sink dataset' (set to 'Orders_Snowflake'), 'Pre-copy script', 'Storage integration', and 'Additional Snowflake copy options'. A red arrow points to the 'Sink dataset' field.

Note:

- Under “Mapping” tab, Schemas should be imported if not, import and verify mapping properly.
- Here, data moves from Azure Blob Container Storage **Orders.csv** → Snowflake’s **Orders** table.

Step 7

Create Azure Pipeline

Step 7c: Under **Settings** → ensure **Enable staging** is selected → enter ‘**Storage path**’ as ‘**staging**’ → Select ‘**Ls_staging_blob_storage**’ → **Test connection** if success click on ‘**Publish all**’.

The screenshot shows the Azure Data Factory pipeline editor for a pipeline named 'pl_S3_n_Azure_Blo...'. On the left, the 'Activities' sidebar is open, showing categories like 'Move and transform', 'Copy data', and 'Data flow'. The main workspace displays two 'Copy data' activities connected sequentially. The top activity is 'Copy Customers S3' and the bottom activity is 'Copy Orders Azure Blob'. Both activities have green checkmarks indicating they are valid. Below the activities, the pipeline settings are visible. The 'Settings' tab is selected, highlighted with a red circle. The 'Enable staging' checkbox is checked. The 'Storage path' field contains the value 'staging'. The 'Staging account linked service' dropdown is set to 'Ls_staging_blob_storage'. A red arrow points from the 'staging' text input to the dropdown. Another red arrow points from the 'Ls_staging_blob_storage' text in the dropdown to the 'Test connection' button. The 'Test connection' button has a green checkmark and the text 'Connection successful'. Other tabs in the settings include 'General', 'Source', 'Sink', 'Mapping', and 'User properties'.

Step 8

Create ADF Trigger

Step 8a: ADF Studio → Author → Select Pipeline → Add Trigger → Configure Schedule → Publish & Test.

The image contains three screenshots of the Azure Data Factory (ADF) interface:

- Screenshot 1: ADF Studio - Author - New trigger**

This screenshot shows the "New trigger" dialog. The "Name" field is set to "cross-cloud-integration". The "Type" dropdown is set to "Schedule". The "Start date" is set to "3/4/2025, 5:33:47 AM". The "Recurrence" section shows "Every 15 Minute(s)". The "End On" field is set to "3/4/2025, 12:05:00 AM". The "OK" button is highlighted in blue.
- Screenshot 2: ADF Studio - All pipeline runs - Activity runs**

This screenshot shows the "Activity runs" section for a pipeline run. It lists two activities: "Copy Orders Azure Blob" and "Copy Customers S3", both marked as "Succeeded". A red box highlights the "Status" column for these activities.
- Screenshot 3: Microsoft Azure - Data Factory - Pipeline runs**

This screenshot shows the "Pipeline runs" list. It displays one run for the pipeline "pl_S3_n_Azure_Blob_to_Snowflake_dw". The run started at "3/3/2025, 11:40:54 PM" and ended at "3/3/2025, 11:42:28 PM" with a duration of "1m 34s". The "Triggered by" column shows "Manual trigger" and the "Status" column shows "Succeeded". A red arrow points from the "Status" column in Screenshot 2 to the "Status" column in Screenshot 3.

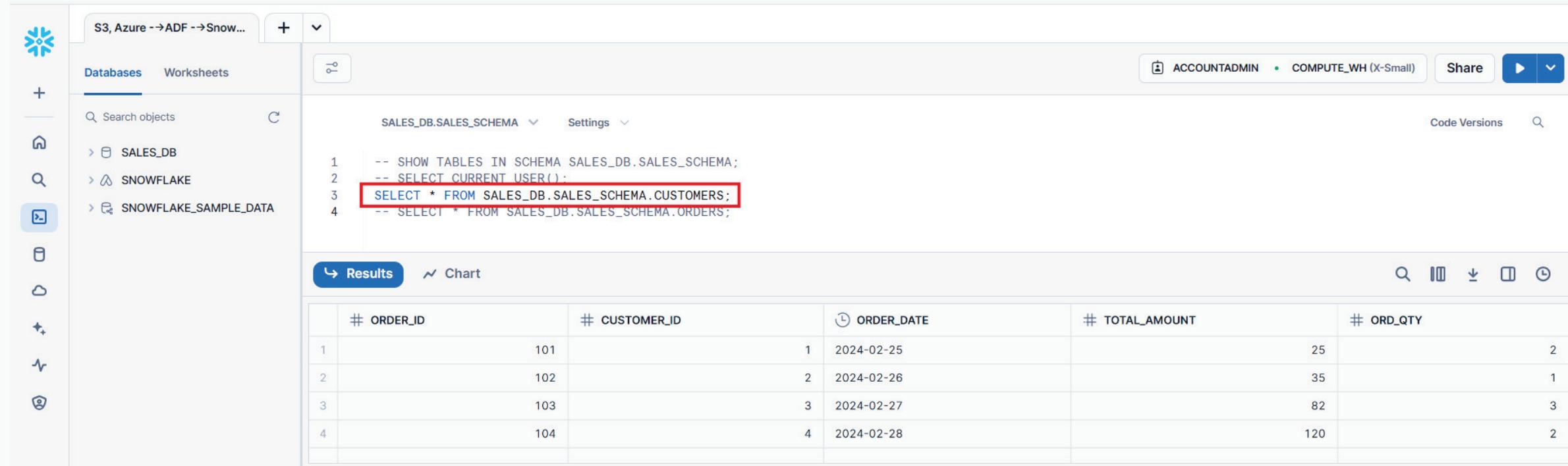
This screenshot shows the "Monitor" section of the Azure Data Factory interface. It displays a table of pipeline runs:

Pipeline name	Run start	Run end	Duration	Triggered by	Status
pl_S3_n_Azure_Blob_to_Snow...	3/3/2025, 11:40:54 PM	3/3/2025, 11:42:28 PM	1m 34s	Manual trigger	Succeeded

Step 9

Data Load Review

SELECT * FROM SALES_DB.SALES_SCHEMA.CUSTOMERS;



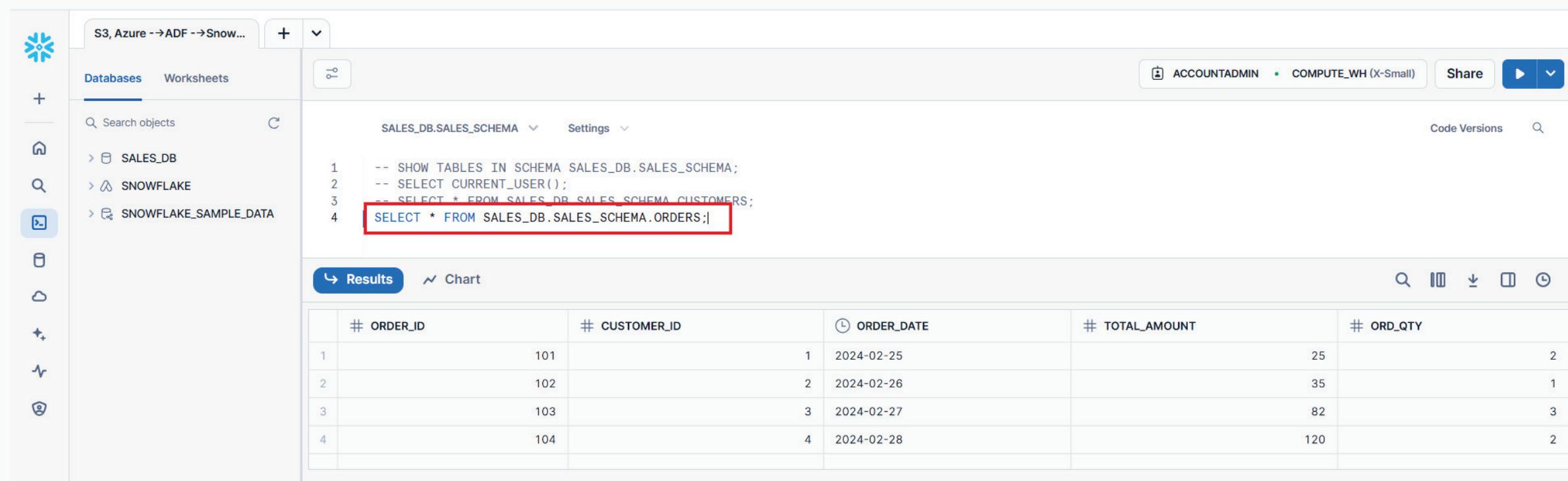
Screenshot of a Snowflake worksheet titled "S3, Azure ->ADF ->Snow...". The code editor shows the following SQL query:

```
-- SHOW TABLES IN SCHEMA SALES_DB.SALES_SCHEMA;
-- SELECT CURRENT_USER();
SELECT * FROM SALES_DB.SALES_SCHEMA.CUSTOMERS;
-- SELECT * FROM SALES_DB.SALES_SCHEMA.ORDERS;
```

The results table displays four rows of data from the CUSTOMERS table:

# ORDER_ID	# CUSTOMER_ID	(L) ORDER_DATE	# TOTAL_AMOUNT	# ORD_QTY
1	101	1 2024-02-25	25	2
2	102	2 2024-02-26	35	1
3	103	3 2024-02-27	82	3
4	104	4 2024-02-28	120	2

SELECT * FROM SALES_DB.SALES_SCHEMA.ORDERS;



Screenshot of a Snowflake worksheet titled "S3, Azure ->ADF ->Snow...". The code editor shows the following SQL query:

```
-- SHOW TABLES IN SCHEMA SALES_DB.SALES_SCHEMA;
-- SELECT CURRENT_USER();
-- SELECT * FROM SALES_DB.SALES_SCHEMA.CUSTOMERS;
SELECT * FROM SALES_DB.SALES_SCHEMA.ORDERS;
```

The results table displays four rows of data from the ORDERS table:

# ORDER_ID	# CUSTOMER_ID	(L) ORDER_DATE	# TOTAL_AMOUNT	# ORD_QTY
1	101	1 2024-02-25	25	2
2	102	2 2024-02-26	35	1
3	103	3 2024-02-27	82	3
4	104	4 2024-02-28	120	2



- THANK YOU -

Prayag Verma | Data Engineer

[in/prayagv](https://www.linkedin.com/in/prayagv) [/prayag-verma](https://github.com/prayag-verma)

Follow Me To Learn More!