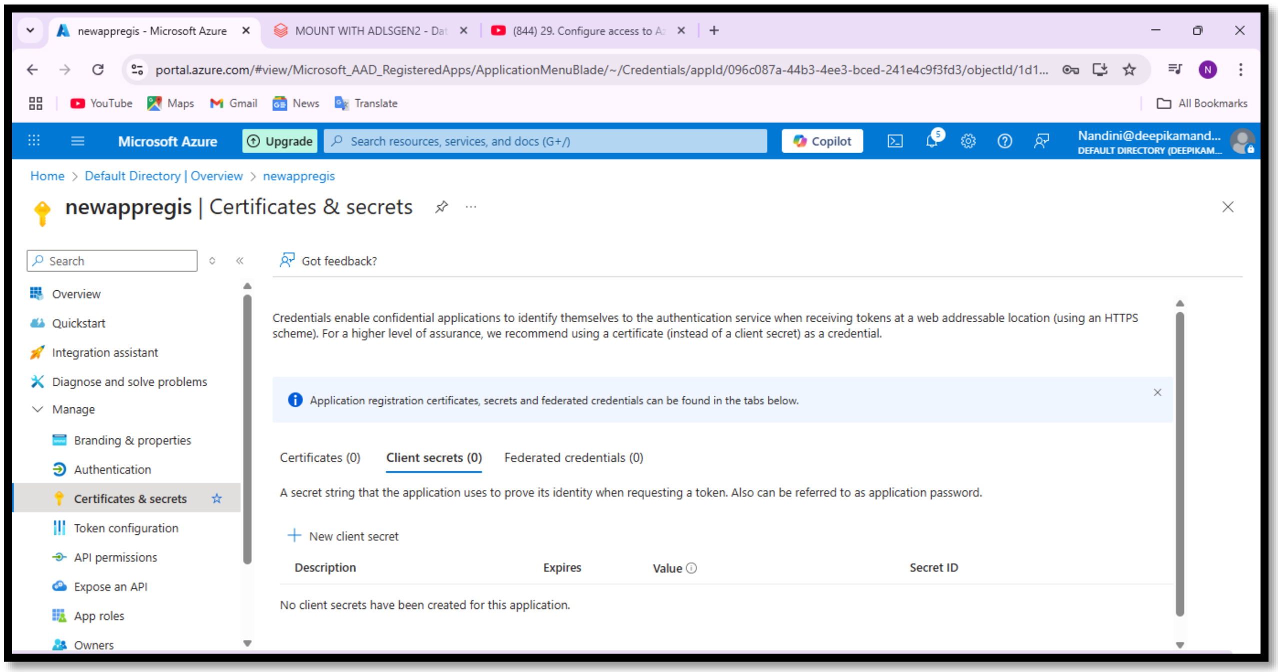
PROJECT-06 (DATABRICKS)

NANDINI RATHORE

1.Python fundamentals: Data types, looping concepts (for, if), functions concept, exception handling (3 examples with project).

2. Get data from ADLSGen2 and load in sql data base (service principle) (key vault) (Storage data contributor role, create scope)

3. Use dynamic method to get data from sql database to adls gen2 (through url copy from another notebook)


newappregis - Microsoft Azure
MOUNT WITH ADLSGEN2 - Dat
(844) 29. Configure access to A
portal.azure.com/#view/Microsoft_AAD_RegisteredApps/ApplicationMenuBlade/~/Credentials/appId/096c087a-44b3-4ee3-bced-241e4c9f3fd3/objectId/1d1...
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Copilot
Home > Default Directory | Overview > newappregis
newappregis | Certificates & secrets
Overview
Quickstart
Integration assistant
Diagnose and solve problems
Manage
Branding & properties
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles
Owners
Got feedback?
Certificates (0)
Client secrets (0)
Federated credentials (0)
New client secret
Description
Expires
Value
Secret ID

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using an HTTPS scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Application registration certificates, secrets and federated credentials can be found in the tabs below.

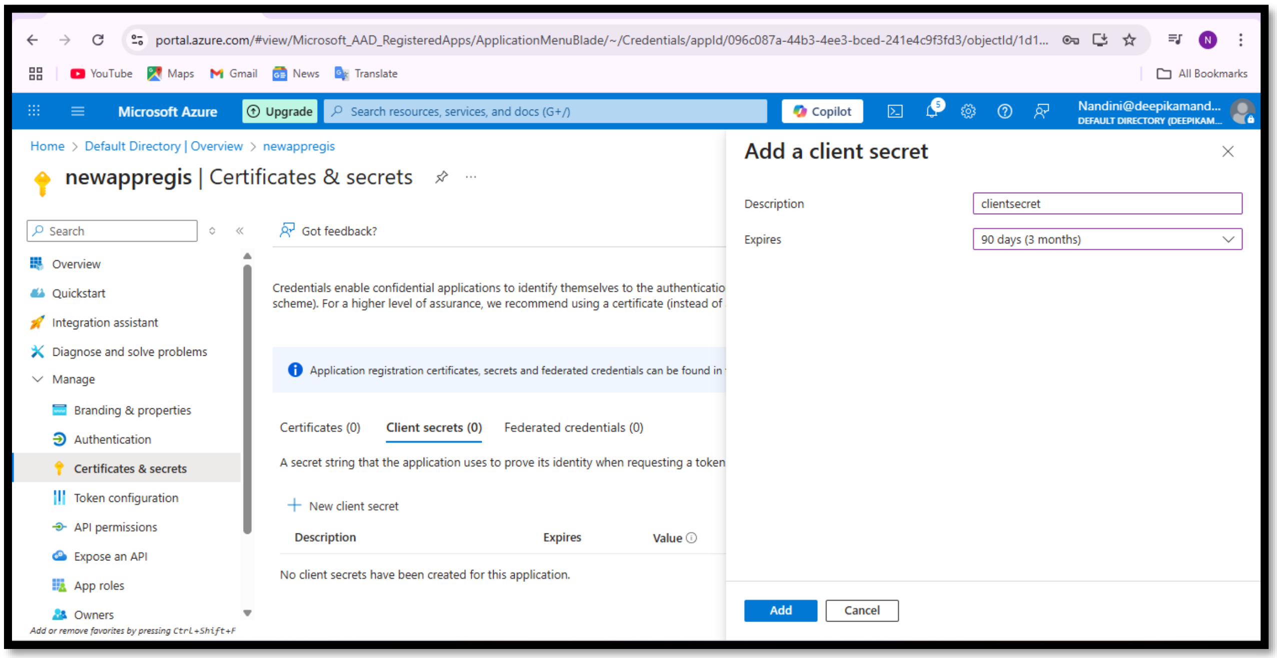
A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

No client secrets have been created for this application.

1. OPEN AZURE→MICROSOFTENTRAID→APP REGISTRATION.

CREATE A NEW APP.

THEN GO TO CERTIFICATES AND SECRETS IN LEFT BAR AND CREATE CLIENT SECRETS AND COPY THAT SAVE IT SOMEWHERE IN DEVICE.

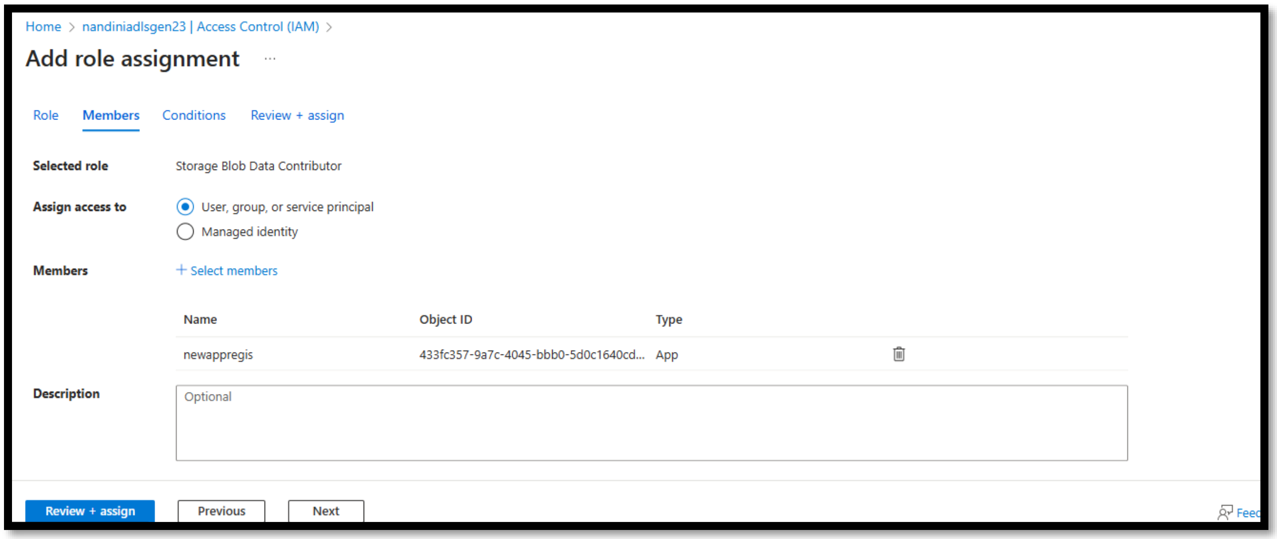


Add a client secret

Description: clientsecret

Expires: 90 days (3 months)

2. CREATE CLIENT SECRET SELECT EXPIRYDATE.


Home > nandiniadlsgen23 | Access Control (IAM) >

Add role assignment

Role | Members | Conditions | Review + assign

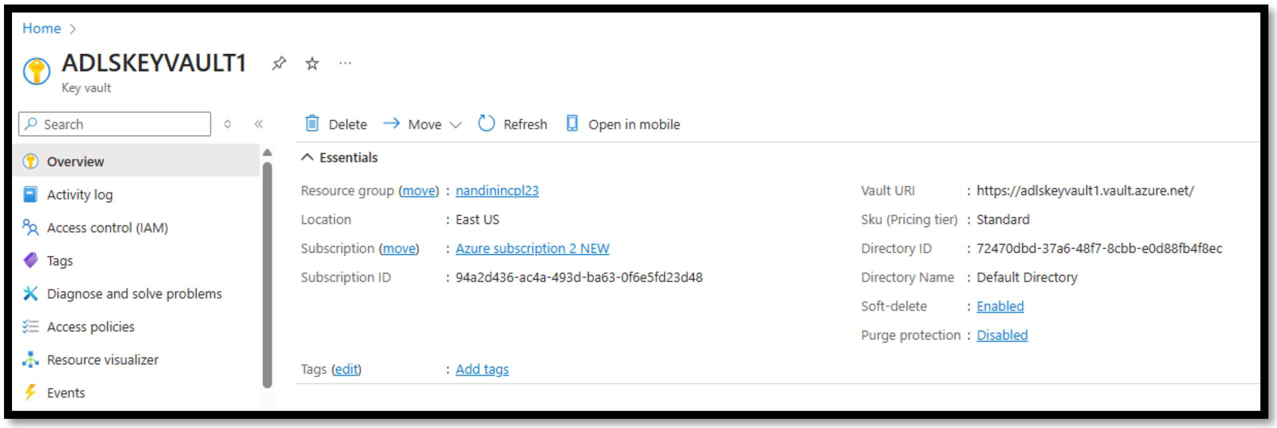
Selected role	Storage Blob Data Contributor
Assign access to	(●) User, group, or service principal ( ) Managed identity
Members	+ Select members

Name	Object ID	Type
newappregis	433fc357-9a7c-4045-bbb0-5d0c1640cd...	App

Description: Optional

Review + assign | Previous | Next

3. THEN ADLSGEN ACCOUNT→GIVE ROLE ASSIGNMENT TO REGISTERAPP→"STORAGE BLOB DATA CONTRIBUTOR."


Home >

ADLSKEYVAULT1
Key vault

Delete | Move | Refresh | Open in mobile

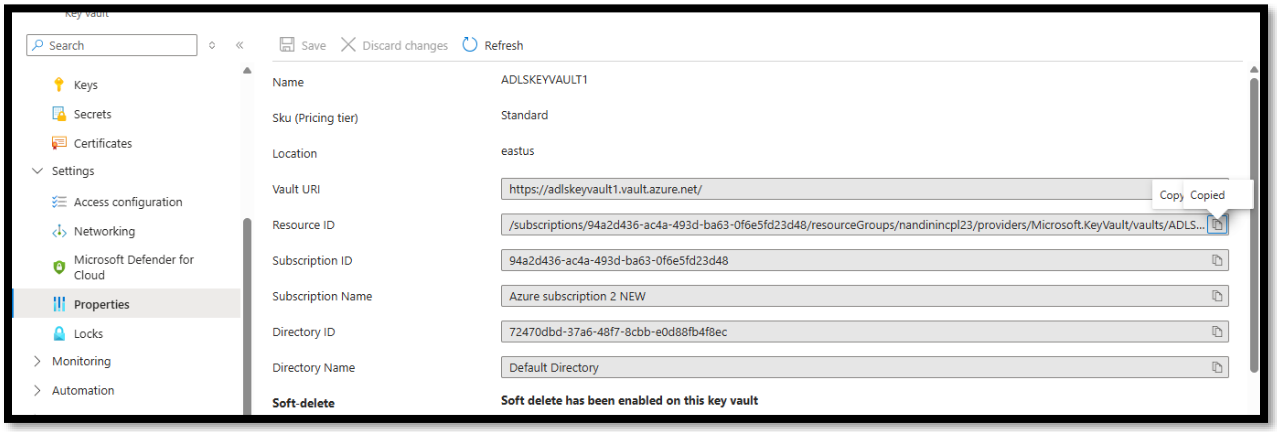
Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Access policies, Resource visualizer, Events

Essentials

Resource group (move)	: nandinincpl23	Vault URI	: https://adlskeyvault1.vault.azure.net/
Location	: East US	Sku (Pricing tier)	: Standard
Subscription (move)	: Azure subscription 2 NEW	Directory ID	: 72470dbd-37a6-48f7-8cbb-e0d88fb4f8ec
Subscription ID	: 94a2d436-ac4a-493d-ba63-0f6e5fd23d48	Directory Name	: Default Directory
		Soft-delete	: Enabled
		Purge protection	: Disabled
Tags (edit)	: Add tags		

4. FROM KEYVAULT,COPY DIRECTORY ID OR TENANT ID.

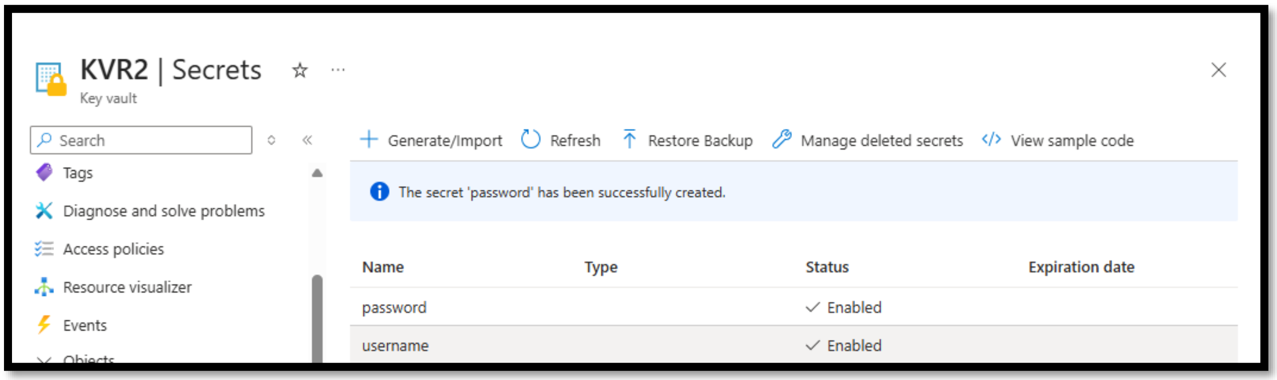
COPY VAULT URI-IT IS THE SPECIPIC ID OF KEY VAULT.


Save | Discard changes | Refresh

Keys, Secrets, Certificates, Settings, Access configuration, Networking, Microsoft Defender for Cloud, Properties, Locks, Monitoring, Automation

Name	ADLSKEYVAULT1
Sku (Pricing tier)	Standard
Location	eastus
Vault URI	https://adlskeyvault1.vault.azure.net/
Resource ID	/subscriptions/94a2d436-ac4a-493d-ba63-0f6e5fd23d48/resourceGroups/nandinincpl23/providers/Microsoft.KeyVault/vaults/ADLS...
Subscription ID	94a2d436-ac4a-493d-ba63-0f6e5fd23d48
Subscription Name	Azure subscription 2 NEW
Directory ID	72470dbd-37a6-48f7-8cbb-e0d88fb4f8ec
Directory Name	Default Directory
Soft-delete	Soft delete has been enabled on this key vault

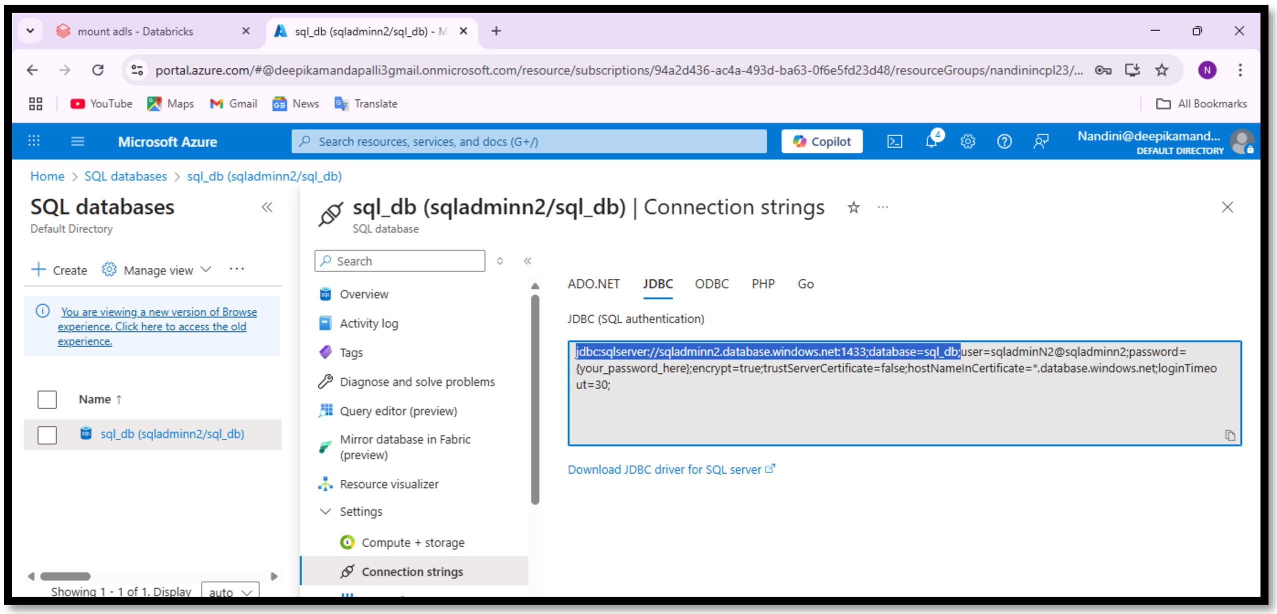
COPY RESOURCE ID .


KVR2 | Secrets

Key vault

The secret 'password' has been successfully created.

Name	Type	Status	Expiration date
password		✓ Enabled	
username		✓ Enabled	

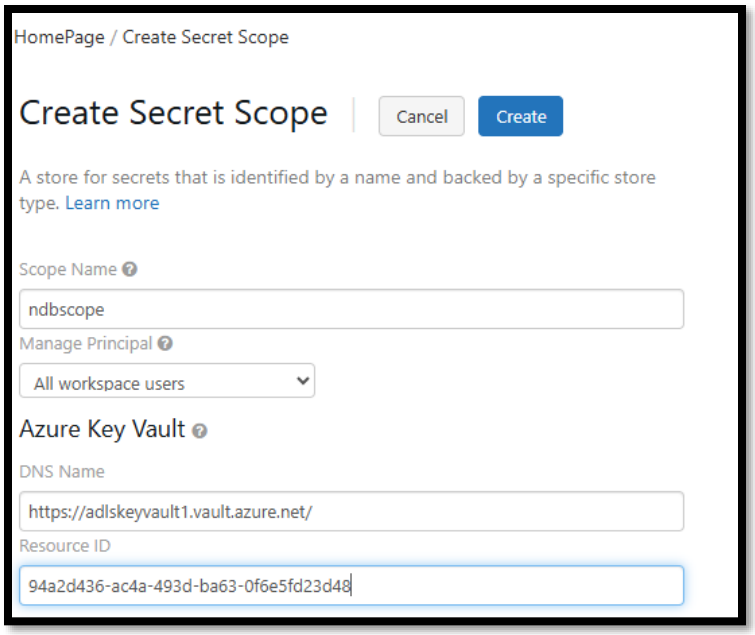

sql_db (sqladminn2/sql_db) | Connection strings

SQL database

JDBC (SQL authentication)

```
jdbc:sqlserver://sqladminn2.database.windows.net:1433;database=sql_db;user=sqladminN2@sqladminn2;password={your_password_here};encrypt=true;trustServerCertificate=false;hostNameInCertificate=*.database.windows.net;loginTimeout=30;
```

Download JDBC driver for SQL server



HomePage / Create Secret Scope

Create Secret Scope

A store for secrets that is identified by a name and backed by a specific store type. Learn more

Scope Name: ndbscope

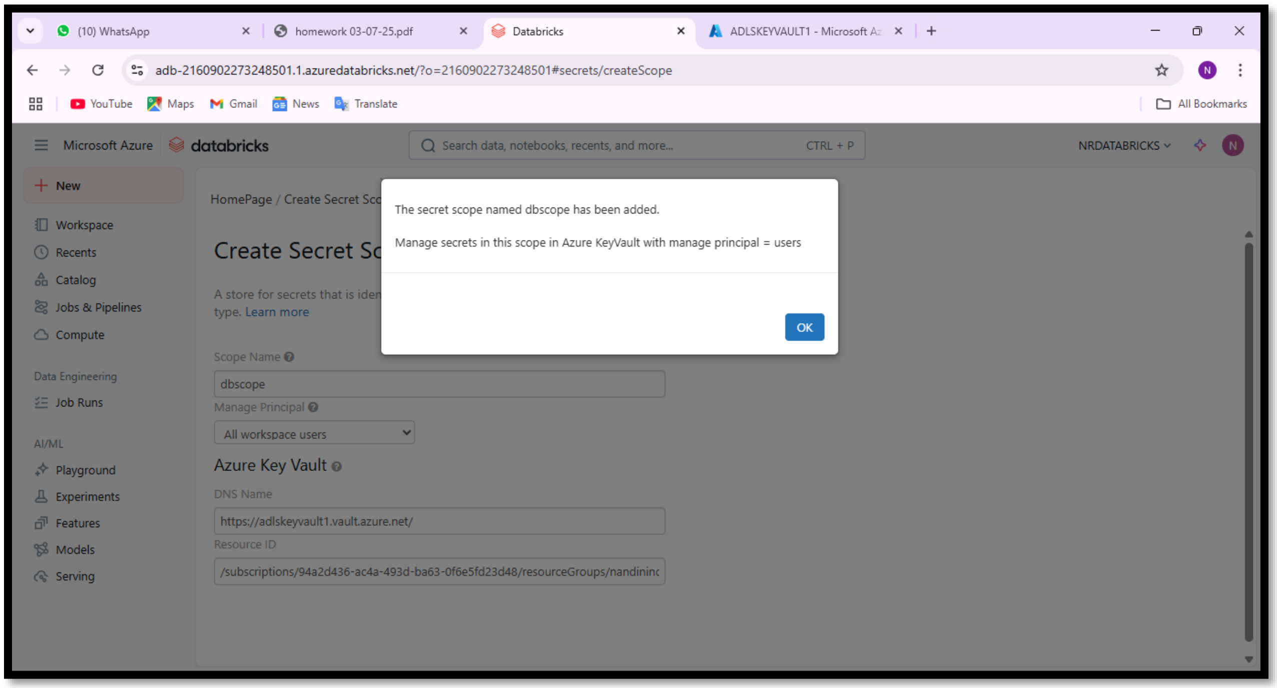
Manage Principal: All workspace users

Azure Key Vault

DNS Name: https://adlskeyvault1.vault.azure.net/

Resource ID: 94a2d436-ac4a-493d-ba63-0f6e5fd23d48

CREATE SCOPE, WRITE NAME,PASTE VAULT URI IN DNS NAME AND PASTE RESOURCE ID THAT WE HAVE COPY FROM KEY VAULT.



adb-2160902273248501.1.azuredatabricks.net/?o=2160902273248501#secrets/createScope

The secret scope named dbscope has been added.

Manage secrets in this scope in Azure KeyVault with manage principal = users

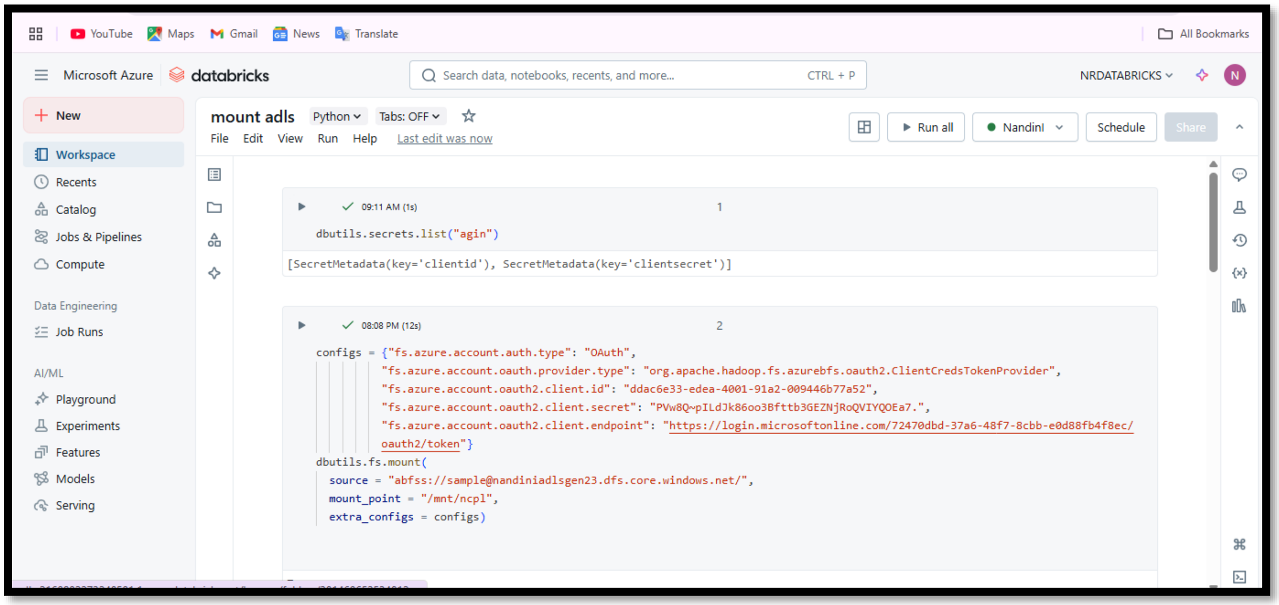
Scope Name: dbscope

Manage Principal: All workspace users

Azure Key Vault

DNS Name: https://adlskeyvault1.vault.azure.net/

Resource ID: /subscriptions/94a2d436-ac4a-493d-ba63-0f6e5fd23d48/resourceGroups/nandininc



```
dbutils.secrets.list("agin")
```

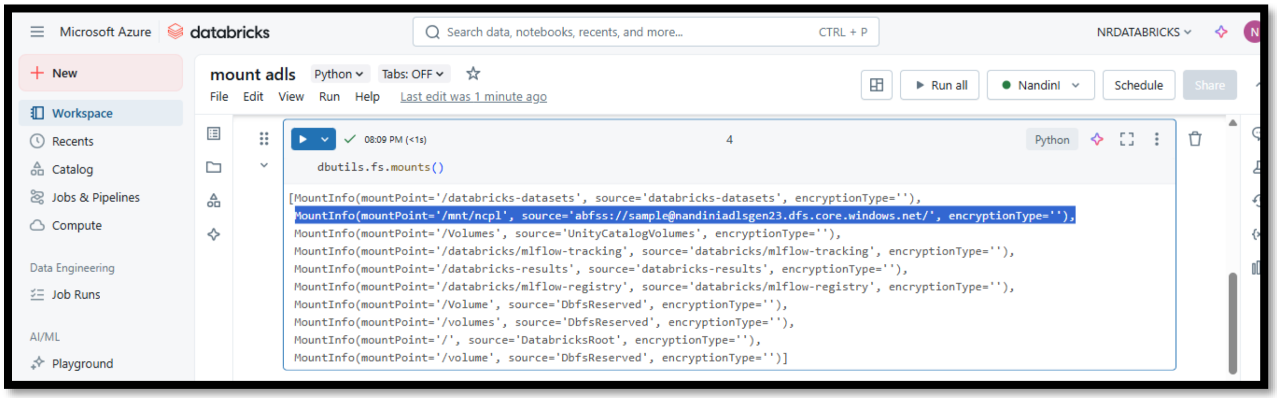
```
[SecretMetadata(key='clientid'), SecretMetadata(key='clientsecret')]
```

```
configs = {"fs.azure.account.auth.type": "OAuth",
          "fs.azure.account.oauth.provider.type": "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
          "fs.azure.account.oauth2.client.id": "ddac6e33-edea-4001-91a2-009446b77a52",
          "fs.azure.account.oauth2.client.secret": "PVw8Q~pILdJk86oo3Bfttb3GEZNjRoQVIYQOEa7.",
          "fs.azure.account.oauth2.client.endpoint": "https://login.microsoftonline.com/72470dbd-37a6-48f7-8cbb-e0d88fb4f8ec/
          oauth2/token"}
dbutils.fs.mount(
  source = "abfss://sample@nandiniadlsgen23.dfs.core.windows.net/",
  mount_point = "/mnt/ncpl",
  extra_configs = configs)
```

5.mount with adls storage account.

Using scope ,to get clientid and client secret value.

Then mount the adls with databricks by using scope, clientid, clientsecret , storage container,storage account and mount point.

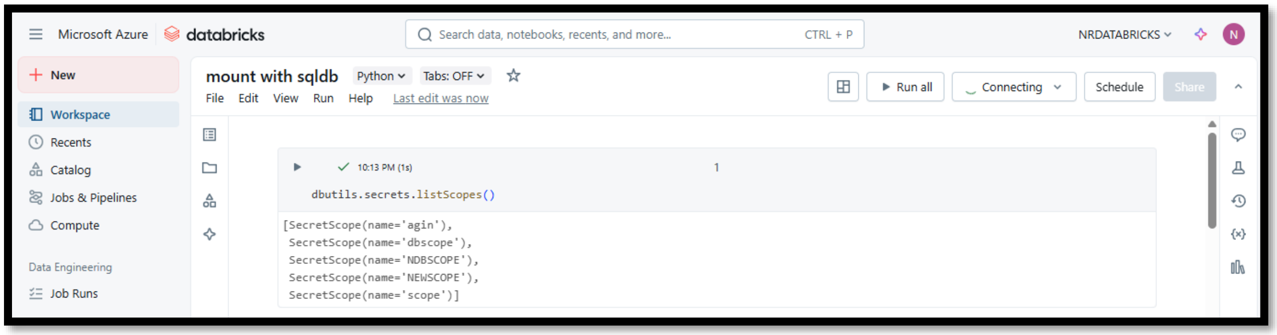


```
dbutils.fs.mounts()
```

```
[MountInfo(mountPoint='/databricks-datasets', source='databricks-datasets', encryptionType=''),
 MountInfo(mountPoint='/mnt/ncpl', source='abfss://sample@nandiniadlsgen23.dfs.core.windows.net/', encryptionType=''),
 MountInfo(mountPoint='/Volumes', source='UnityCatalogVolumes', encryptionType=''),
 MountInfo(mountPoint='/databricks/mlflow-tracking', source='databricks/mlflow-tracking', encryptionType=''),
 MountInfo(mountPoint='/databricks-results', source='databricks-results', encryptionType=''),
 MountInfo(mountPoint='/databricks/mlflow-registry', source='databricks/mlflow-registry', encryptionType=''),
 MountInfo(mountPoint='/Volume', source='DbfsReserved', encryptionType=''),
 MountInfo(mountPoint='/volumes', source='DbfsReserved', encryptionType=''),
 MountInfo(mountPoint='/', source='DatabricksRoot', encryptionType=''),
 MountInfo(mountPoint='/volume', source='DbfsReserved', encryptionType='')]
```

Abfss:→ Azure Blob File System Secure.

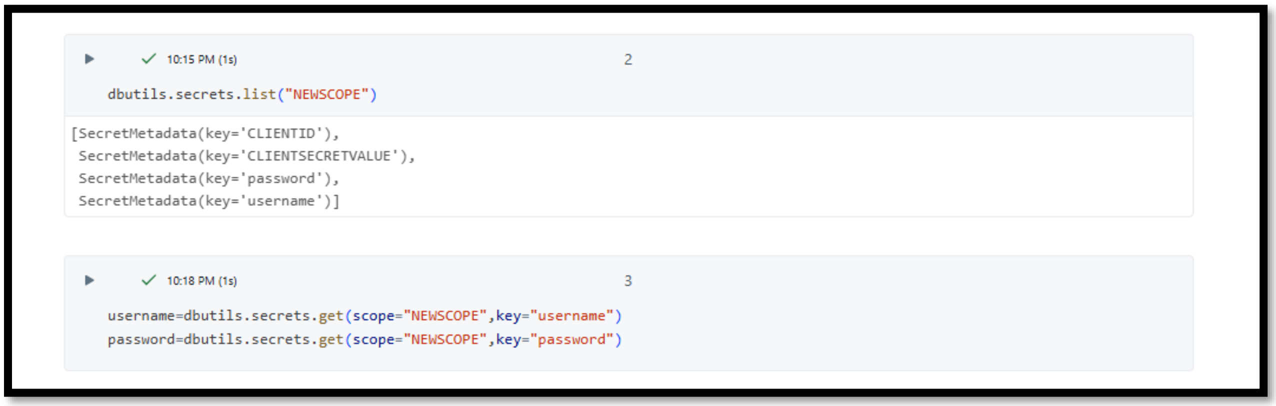
abfss:// is used to securely connect to Azure Data Lake Storage Gen2 from tools like Databricks and Spark.



```
dbutils.secrets.listScopes()
```

```
[SecretScope(name='agin'),
 SecretScope(name='dbscope'),
 SecretScope(name='NDBSCOPE'),
 SecretScope(name='NEWSCOPE'),
 SecretScope(name='scope')]
```

List of scopes in databricks.

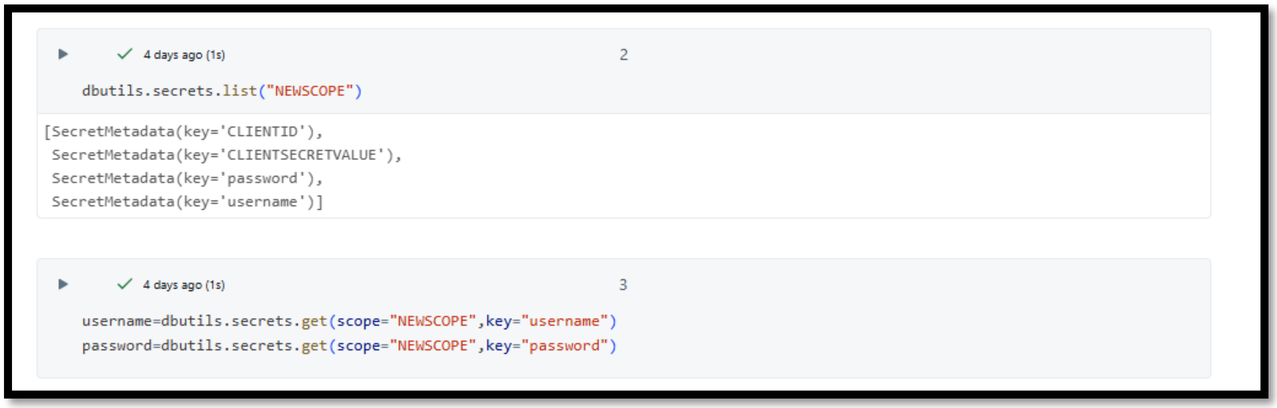


```
dbutils.secrets.list("NEWSCOPE")
```

```
[SecretMetadata(key='CLIENTID'),
 SecretMetadata(key='CLIENTSECRETVALUE'),
 SecretMetadata(key='password'),
 SecretMetadata(key='username')]
```

```
username=dbutils.secrets.get(scope="NEWSCOPE",key="username")
password=dbutils.secrets.get(scope="NEWSCOPE",key="password")
```

Secrets in the scope. Username and password used in sqldb.



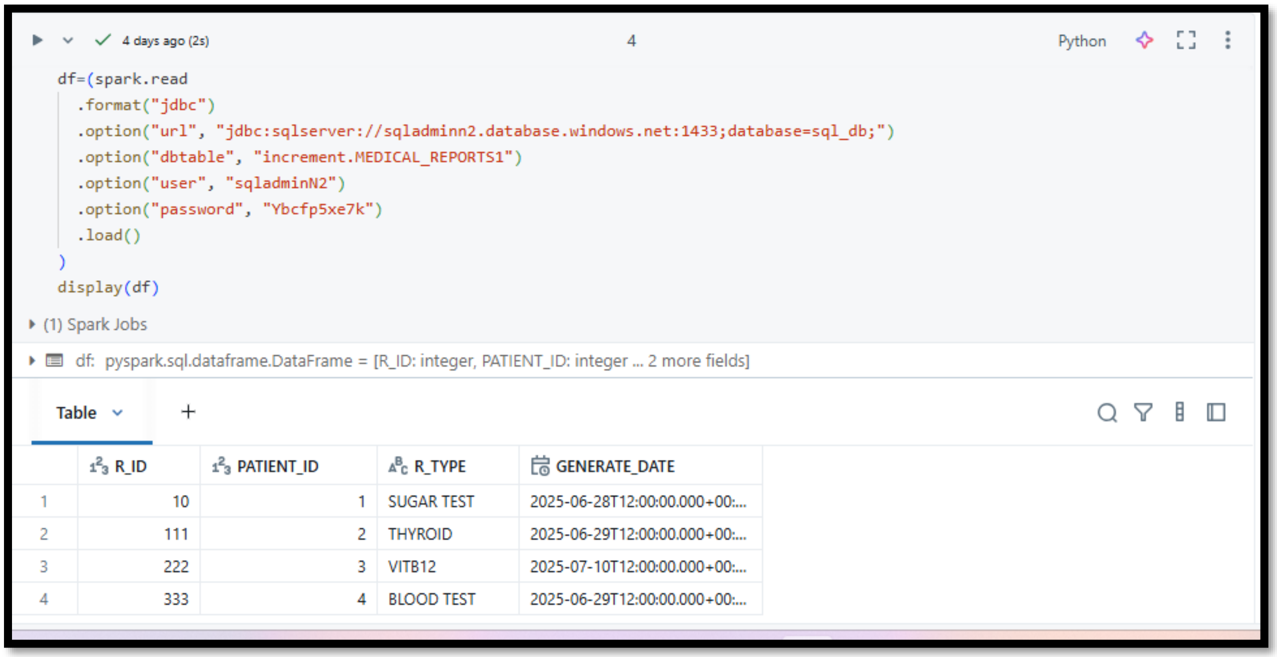
```
dbutils.secrets.list("NEWSCOPE")
```

```
[SecretMetadata(key='CLIENTID'),
 SecretMetadata(key='CLIENTSECRETVALUE'),
 SecretMetadata(key='password'),
 SecretMetadata(key='username')]
```

```
username=dbutils.secrets.get(scope="NEWSCOPE",key="username")
password=dbutils.secrets.get(scope="NEWSCOPE",key="password")
```

1.DATA:- INCREMENT.MEDICAL_REPORTS1

READ DATA FROM ADLSGEN2 TO SQLDB.



4 days ago (2s) 4 Python

```
df=(spark.read
    .format("jdbc")
    .option("url", "jdbc:sqlserver://sqladminn2.database.windows.net:1433;database=sql_db;")
    .option("dbtable", "increment.MEDICAL_REPORTS1")
    .option("user", "sqladminN2")
    .option("password", "Ybcfp5xe7k")
    .load()
)
display(df)
```

(1) Spark Jobs

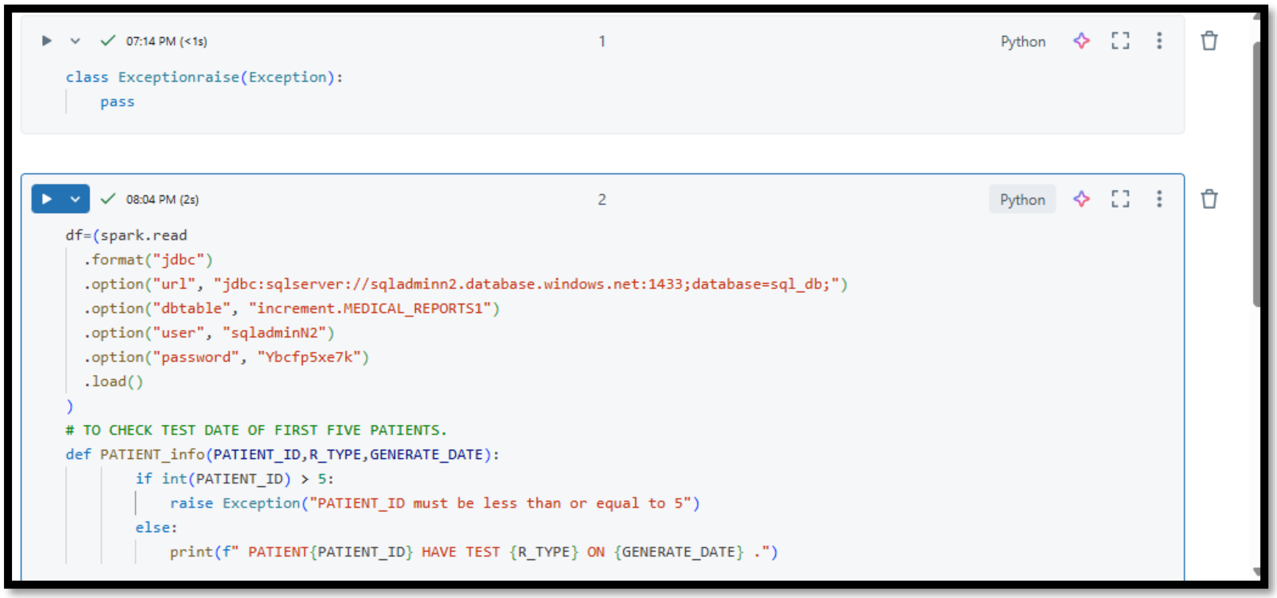
df: pyspark.sql.dataframe.DataFrame = [R_ID: integer, PATIENT_ID: integer ... 2 more fields]

Table

	R_ID	PATIENT_ID	R_TYPE	GENERATE_DATE
1	10	1	SUGAR TEST	2025-06-28T12:00:00.000+00:...
2	111	2	THYROID	2025-06-29T12:00:00.000+00:...
3	222	3	VITB12	2025-07-10T12:00:00.000+00:...
4	333	4	BLOOD TEST	2025-06-29T12:00:00.000+00:...

JDBC (Java Database Connectivity) is a connecting string to connect with database. Used in spark, databricks, java, scala.

IN OTHER NOTEBOOK **DEFINE FUNCTIONS**.



07:14 PM (<1s) 1 Python

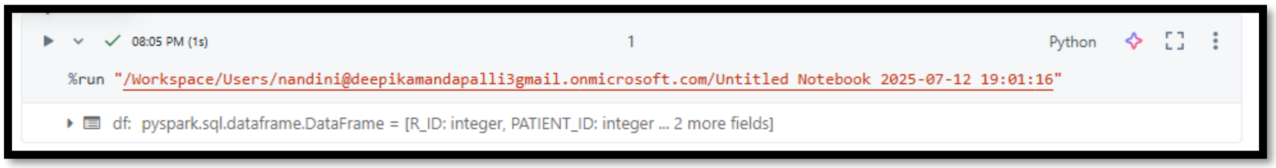
```
class Exceptionraise(Exception):
    pass
```

08:04 PM (2s) 2 Python

```
df=(spark.read
  .format("jdbc")
  .option("url", "jdbc:sqlserver://sqladminn2.database.windows.net:1433;database=sql_db;")
  .option("dbtable", "increment.MEDICAL_REPORTS1")
  .option("user", "sqladminN2")
  .option("password", "Ybcfp5xe7k")
  .load()
)
# TO CHECK TEST DATE OF FIRST FIVE PATIENTS.
def PATIENT_info(PATIENT_ID,R_TYPE,GENERATE_DATE):
        if int(PATIENT_ID) > 5:
            raise Exception("PATIENT_ID must be less than or equal to 5")
        else:
            print(f" PATIENT{PATIENT_ID} HAVE TEST {R_TYPE} ON {GENERATE_DATE} .")
```

CREATE NEW NOTEBOOK AND USE

%run is Databricks magic command to include another notebook.



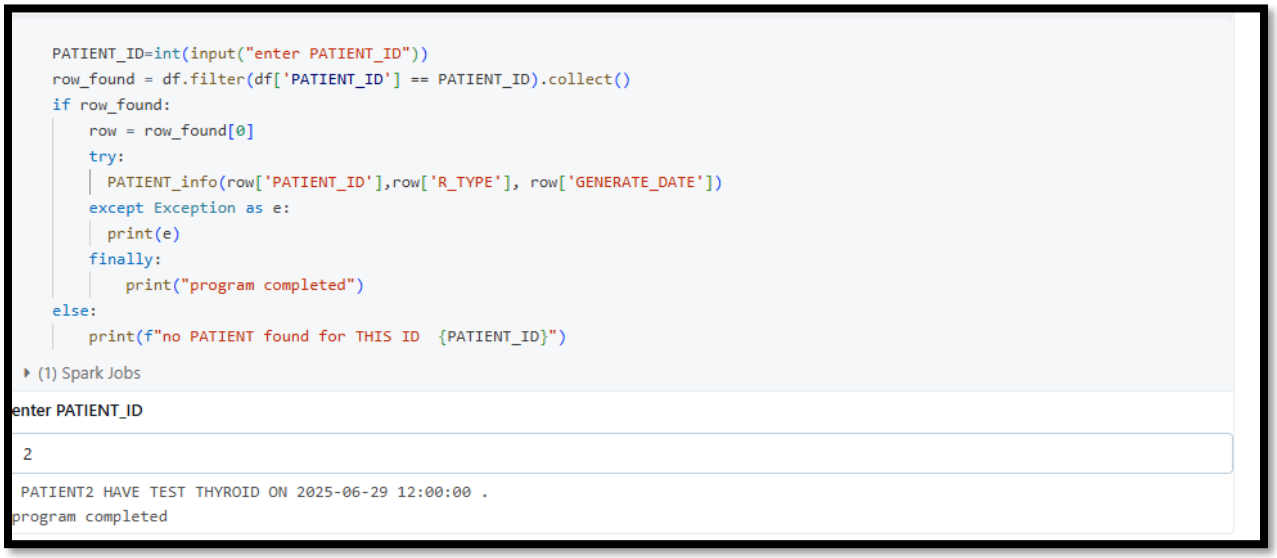
```
%run "/Workspace/Users/nandini@deepikamandapalli3gmail.onmicrosoft.com/Untitled Notebook 2025-07-12 19:01:16"
```

```
df: pyspark.sql.dataframe.DataFrame = [R_ID: integer, PATIENT_ID: integer ... 2 more fields]
```

This command tells Databricks to **run another notebook** located at the path.

- Any **functions, variables, or imports** defined in that notebook (NOTEBOOK)will become available in your **current notebook**.
- It's like copy-pasting all the code from NOTEBOOK into your current notebook — but cleaner and modular.

Call function and exception handling:

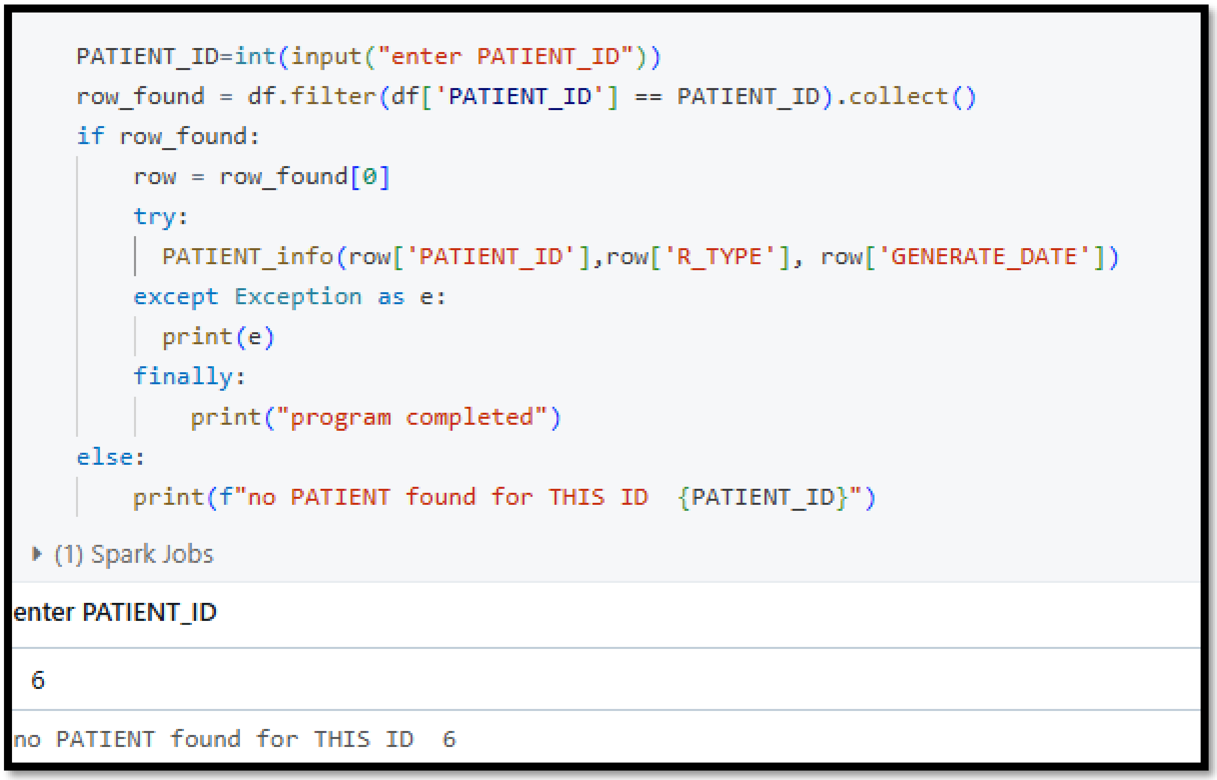


```
PATIENT_ID=int(input("enter PATIENT_ID"))
row_found = df.filter(df['PATIENT_ID'] == PATIENT_ID).collect()
if row_found:
    row = row_found[0]
    try:
        PATIENT_info(row['PATIENT_ID'],row['R_TYPE'], row['GENERATE_DATE'])
    except Exception as e:
        print(e)
    finally:
        print("program completed")
else:
    print(f"no PATIENT found for THIS ID  {PATIENT_ID}")
```

```
enter PATIENT_ID
2
PATIENT2 HAVE TEST THYROID ON 2025-06-29 12:00:00 .
program completed
```

I define function in such a way that it ask to user for input and it define that which patient have test on which day.

If error occur then also it gives a mssg:-



```
PATIENT_ID=int(input("enter PATIENT_ID"))
row_found = df.filter(df['PATIENT_ID'] == PATIENT_ID).collect()
if row_found:
    row = row_found[0]
    try:
        PATIENT_info(row['PATIENT_ID'],row['R_TYPE'], row['GENERATE_DATE'])
    except Exception as e:
        print(e)
    finally:
        print("program completed")
else:
    print(f"no PATIENT found for THIS ID  {PATIENT_ID}")
```

▸ (1) Spark Jobs

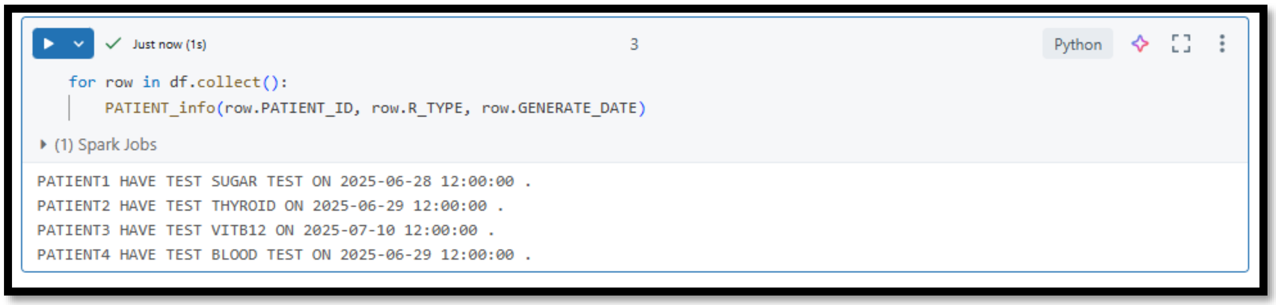
enter PATIENT_ID

```
6
```

```
no PATIENT found for THIS ID  6
```

It gives an exception raise message " That no patient with id 6."

By using for loop:-



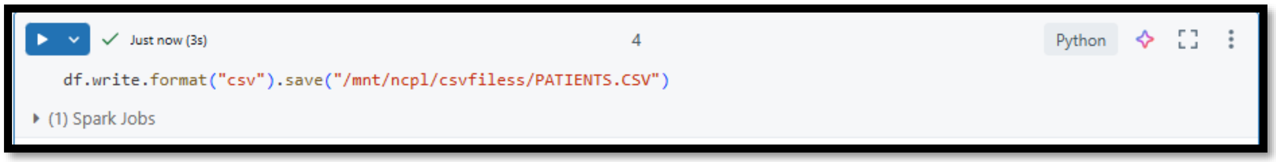
```
for row in df.collect():
    PATIENT_info(row.PATIENT_ID, row.R_TYPE, row.GENERATE_DATE)
```

▸ (1) Spark Jobs

```
PATIENT1 HAVE TEST SUGAR TEST ON 2025-06-28 12:00:00 .
PATIENT2 HAVE TEST THYROID ON 2025-06-29 12:00:00 .
PATIENT3 HAVE TEST VITB12 ON 2025-07-10 12:00:00 .
PATIENT4 HAVE TEST BLOOD TEST ON 2025-06-29 12:00:00 .
```

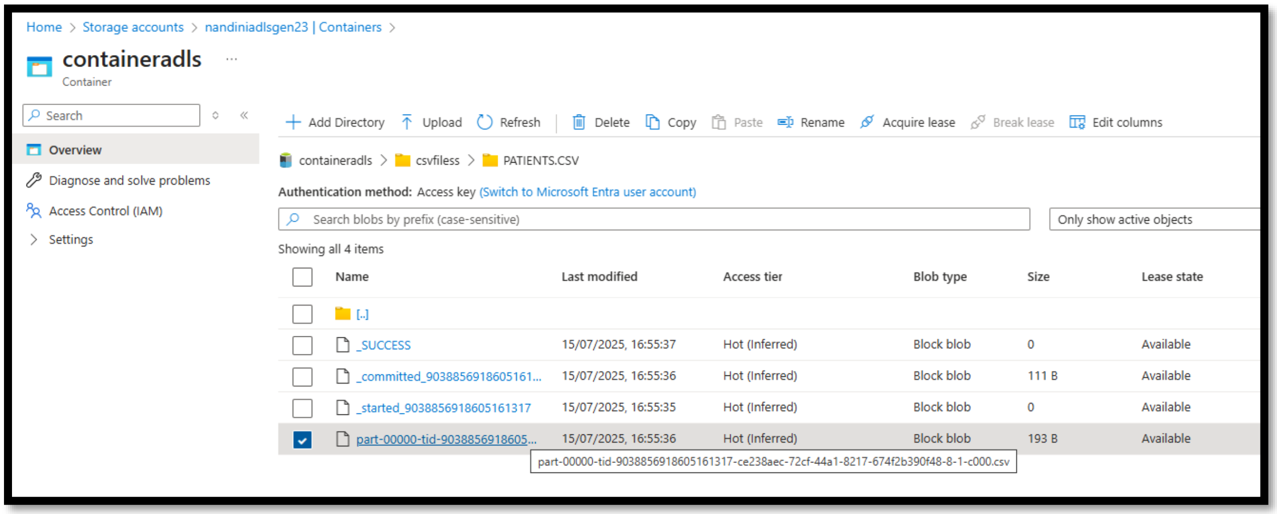
This command means it will "Go through each row in the Spark DataFrame df, and for each row, call the function patient_info()".

Write table from DATABRICKS TO ADLS



```
df.write.format("csv").save("/mnt/ncpl/csvfiless/PATIENTS.CSV")
```

▸ (1) Spark Jobs

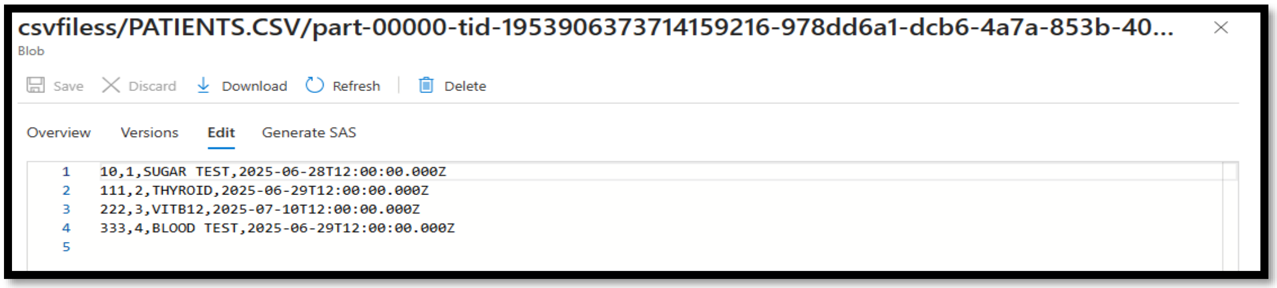


Authentication method: Access key (Switch to Microsoft Entra user account)

Showing all 4 items

Name	Last modified	Access tier	Blob type	Size	Lease state
[..]					
_SUCCESS	15/07/2025, 16:55:37	Hot (Inferred)	Block blob	0	Available
_committed_9038856918605161...	15/07/2025, 16:55:36	Hot (Inferred)	Block blob	111 B	Available
_started_9038856918605161317	15/07/2025, 16:55:35	Hot (Inferred)	Block blob	0	Available
part-00000-tid-9038856918605...	15/07/2025, 16:55:36	Hot (Inferred)	Block blob	193 B	Available

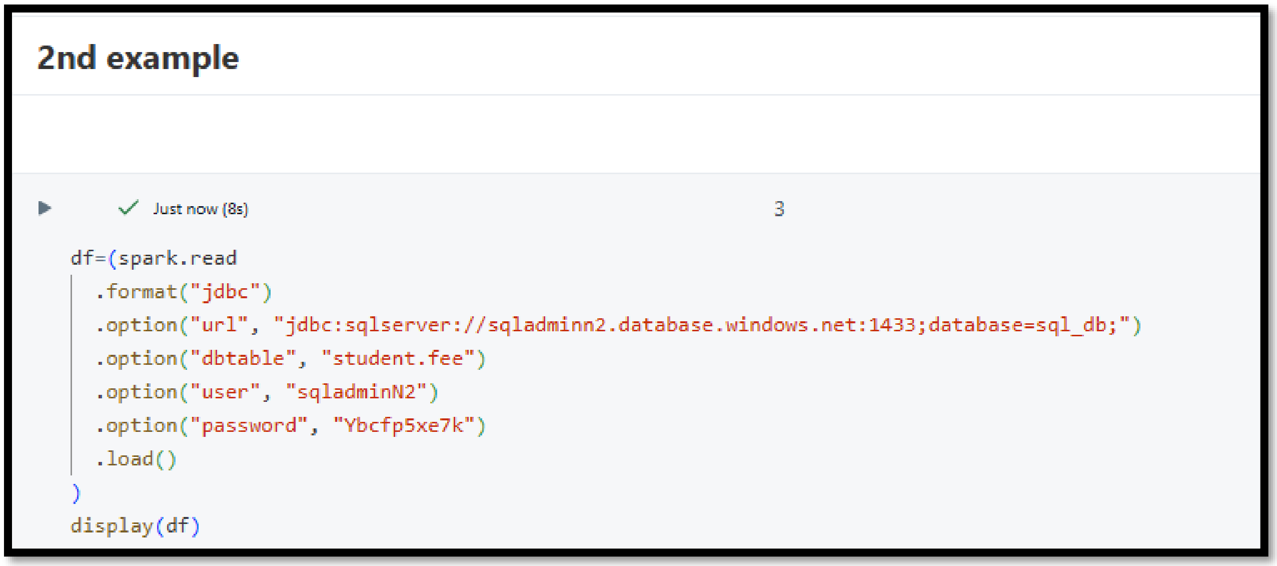
part-00000-tid-9038856918605161317-ce238aec-72cf-44a1-8217-674f2b390f48-8-1-c000.csv


csvfiless/PATIENTS.CSV/part-00000-tid-1953906373714159216-978dd6a1-dcb6-4a7a-853b-40...
Blob
Save
Discard
Download
Refresh
Delete
Overview
Versions
Edit
Generate SAS

```
10,1,SUGAR TEST,2025-06-28T12:00:00.000Z
111,2,THYROID,2025-06-29T12:00:00.000Z
222,3,VITB12,2025-07-10T12:00:00.000Z
333,4,BLOOD TEST,2025-06-29T12:00:00.000Z
```

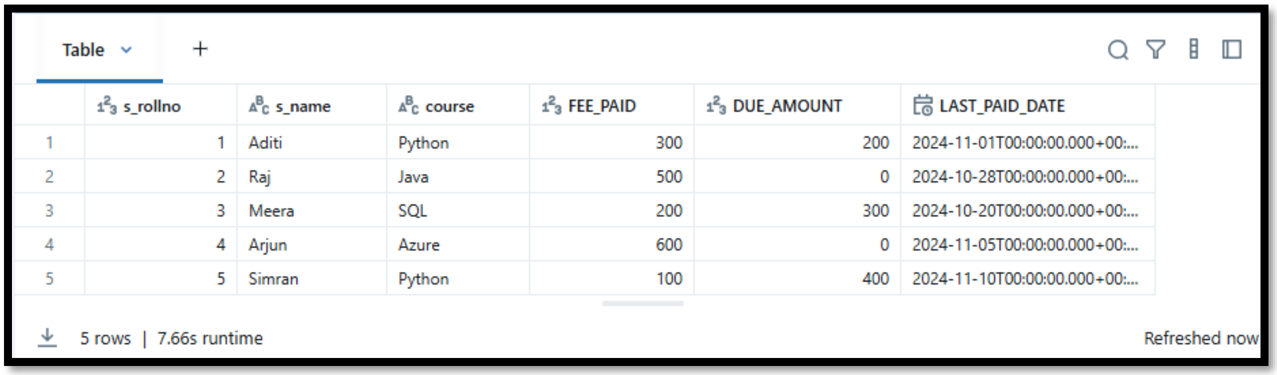
Table stored in csv file in adls storage account.

<u>**Second data:-STUDENT.FEE**</u>


2nd example

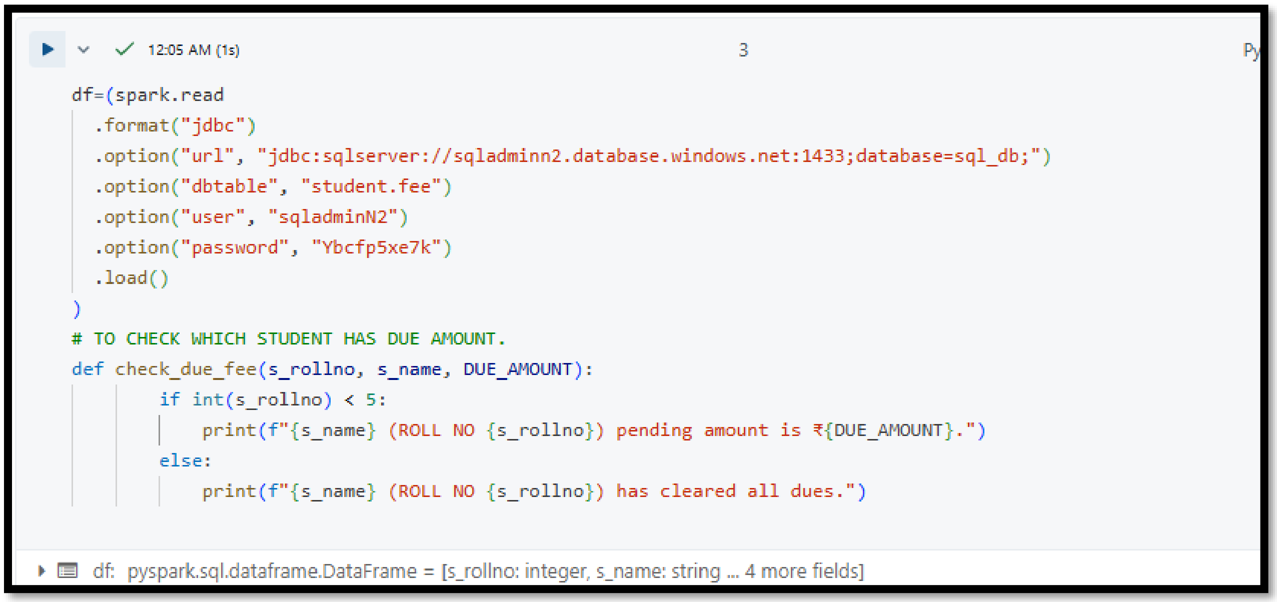
```
df=(spark.read
    .format("jdbc")
    .option("url", "jdbc:sqlserver://sqladminn2.database.windows.net:1433;database=sql_db;")
    .option("dbtable", "student.fee")
    .option("user", "sqladminN2")
    .option("password", "Ybcfp5xe7k")
    .load()
)
display(df)
```

Read table from sqldb in databricks.



	s_rollno	s_name	course	FEE_PAID	DUE_AMOUNT	LAST_PAID_DATE
1	1	Aditi	Python	300	200	2024-11-01T00:00:00.000+00:...
2	2	Raj	Java	500	0	2024-10-28T00:00:00.000+00:...
3	3	Meera	SQL	200	300	2024-10-20T00:00:00.000+00:...
4	4	Arjun	Azure	600	0	2024-11-05T00:00:00.000+00:...
5	5	Simran	Python	100	400	2024-11-10T00:00:00.000+00:...

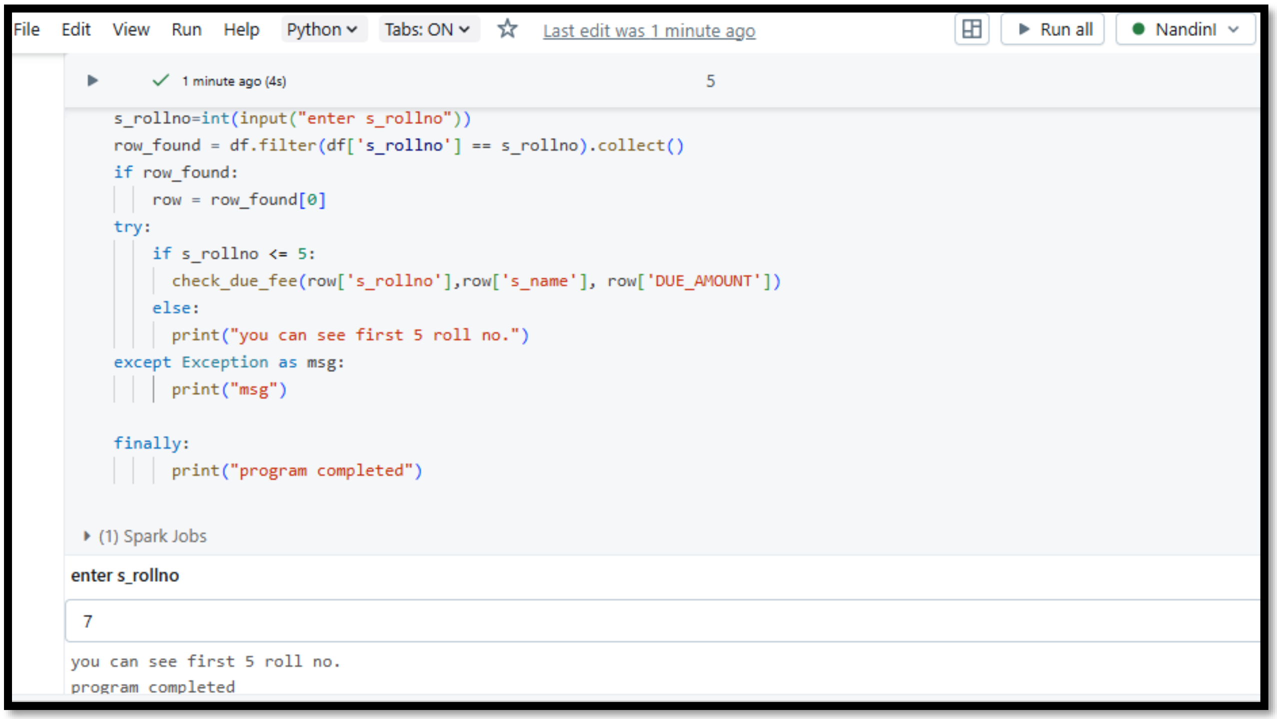
5 rows | 7.66s runtime Refreshed now



```
df=(spark.read
    .format("jdbc")
    .option("url", "jdbc:sqlserver://sqladminn2.database.windows.net:1433;database=sql_db;")
    .option("dbtable", "student.fee")
    .option("user", "sqladminN2")
    .option("password", "Ybcfp5xe7k")
    .load()
)
# TO CHECK WHICH STUDENT HAS DUE AMOUNT.
def check_due_fee(s_rollno, s_name, DUE_AMOUNT):
        if int(s_rollno) < 5:
            print(f"{s_name} (ROLL NO {s_rollno}) pending amount is ₹{DUE_AMOUNT}.")
        else:
            print(f"{s_name} (ROLL NO {s_rollno}) has cleared all dues.")
```

df: pyspark.sql.dataframe.DataFrame = [s_rollno: integer, s_name: string ... 4 more fields]

Define a function , to check which student fee is pending.


File Edit View Run Help Python Tabs: ON Last edit was 1 minute ago Run all Nandini
1 minute ago (4s)
5
(1) Spark Jobs

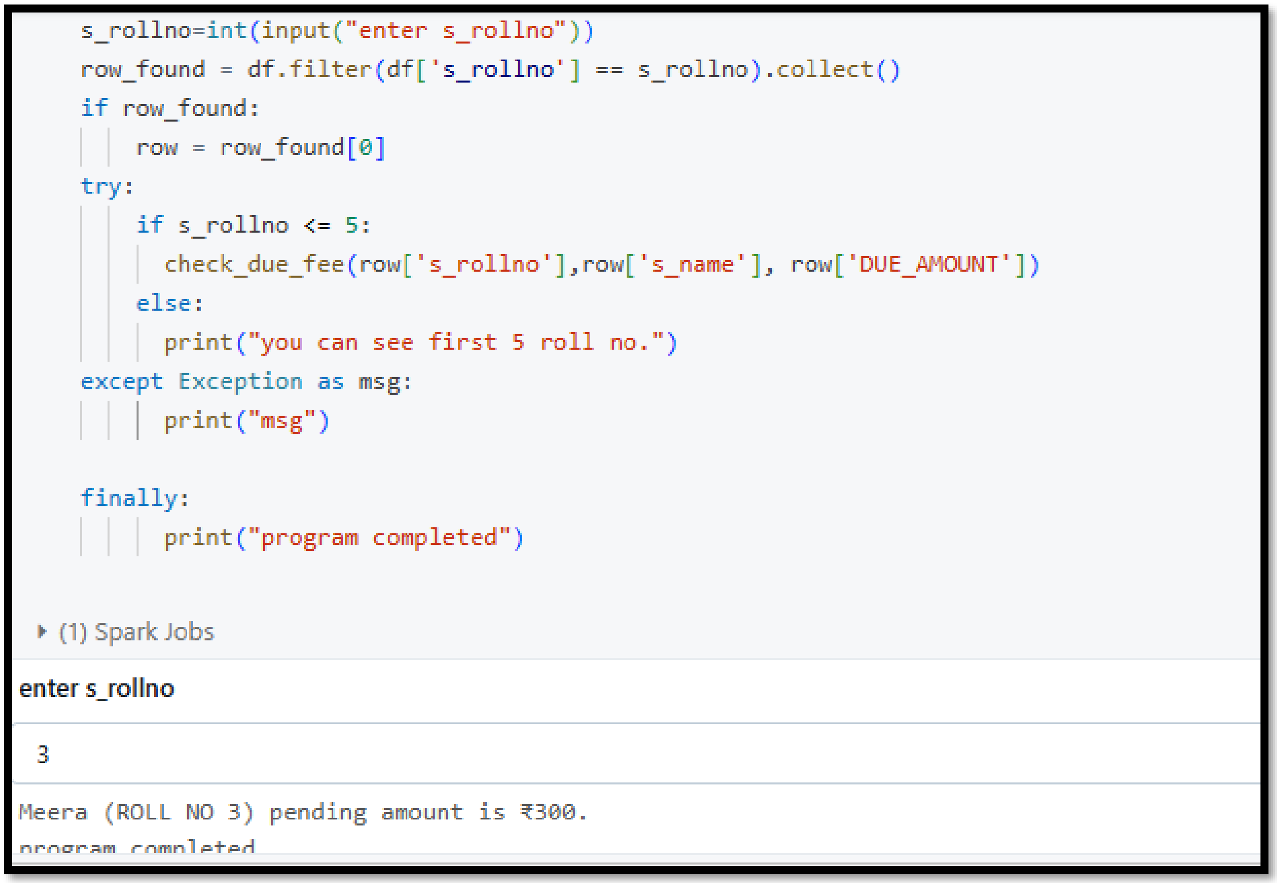
```
s_rollno=int(input("enter s_rollno"))
row_found = df.filter(df['s_rollno'] == s_rollno).collect()
if row_found:
    row = row_found[0]
try:
    if s_rollno <= 5:
        check_due_fee(row['s_rollno'],row['s_name'], row['DUE_AMOUNT'])
    else:
        print("you can see first 5 roll no.")
except Exception as msg:
        print("msg")

finally:
        print("program completed")
```

```
enter s_rollno
7
you can see first 5 roll no.
program completed
```

Now call that function in other notebook having % run “path of previous notebook” and exception handling by using try ,except and final commands.

This screenshot is showing an **exception handling.**



```
s_rollno=int(input("enter s_rollno"))
row_found = df.filter(df['s_rollno'] == s_rollno).collect()
if row_found:
    row = row_found[0]
try:
    if s_rollno <= 5:
      check_due_fee(row['s_rollno'],row['s_name'], row['DUE_AMOUNT'])
    else:
      print("you can see first 5 roll no.")
except Exception as msg:
      print("msg")

finally:
      print("program completed")
```

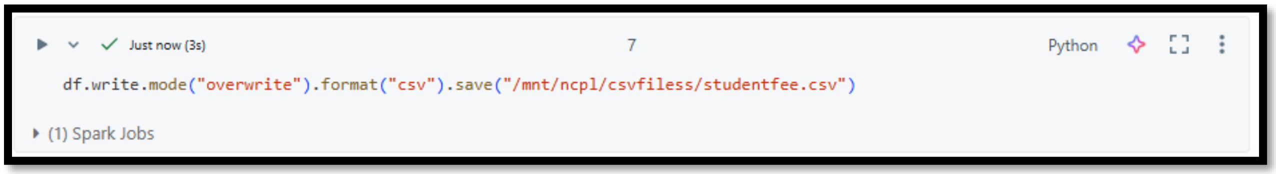
(1) Spark Jobs

enter s_rollno

3

```
Meera (ROLL NO 3) pending amount is ₹300.
```

It will ask for input and show result about that id.if that id is not present that raise an exception.

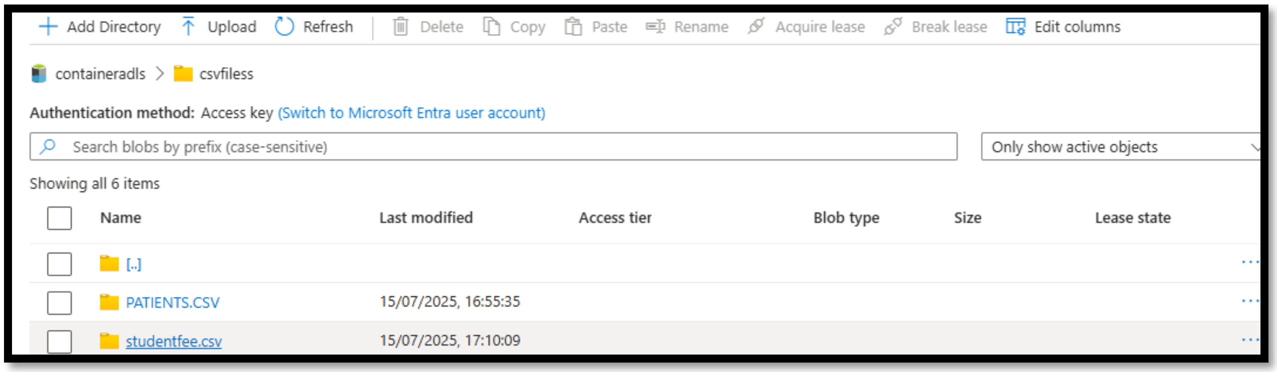


```
df.write.mode("overwrite").format("csv").save("/mnt/ncpl/csvfiless/studentfee.csv")
```

(1) Spark Jobs

Write table to adls as studentfee.csv file.

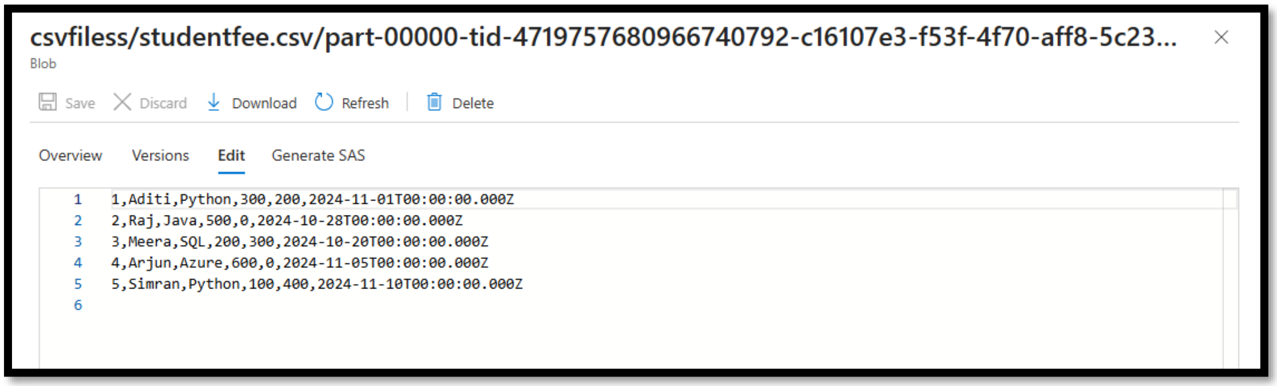
You can choose to store in csv ,parquet files as per requirements.



Authentication method: Access key (Switch to Microsoft Entra user account)

Showing all 6 items

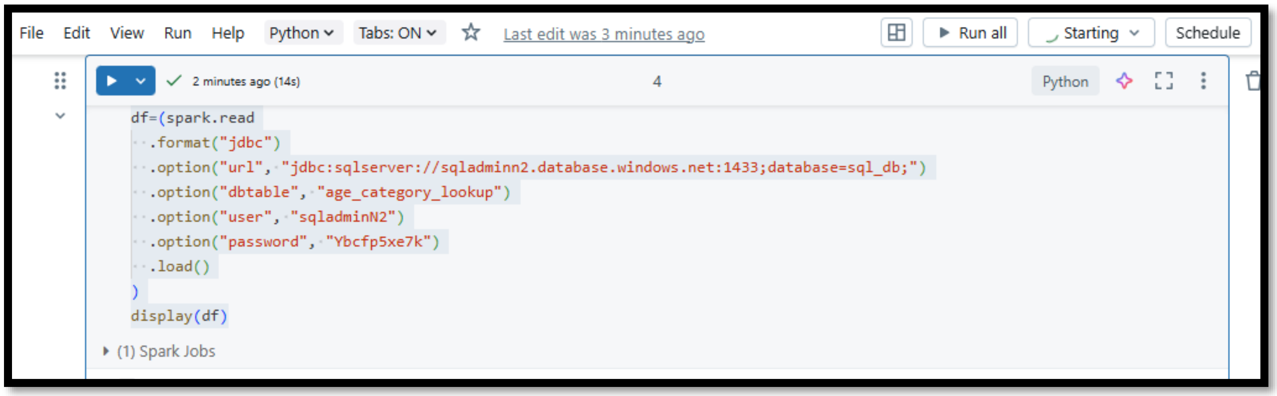
Name	Last modified	Access tier	Blob type	Size	Lease state
[..]					
PATIENTS.CSV	15/07/2025, 16:55:35				
studentfee.csv	15/07/2025, 17:10:09				



csvfiless/studentfee.csv/part-00000-tid-4719757680966740792-c16107e3-f53f-4f70-aff8-5c23...

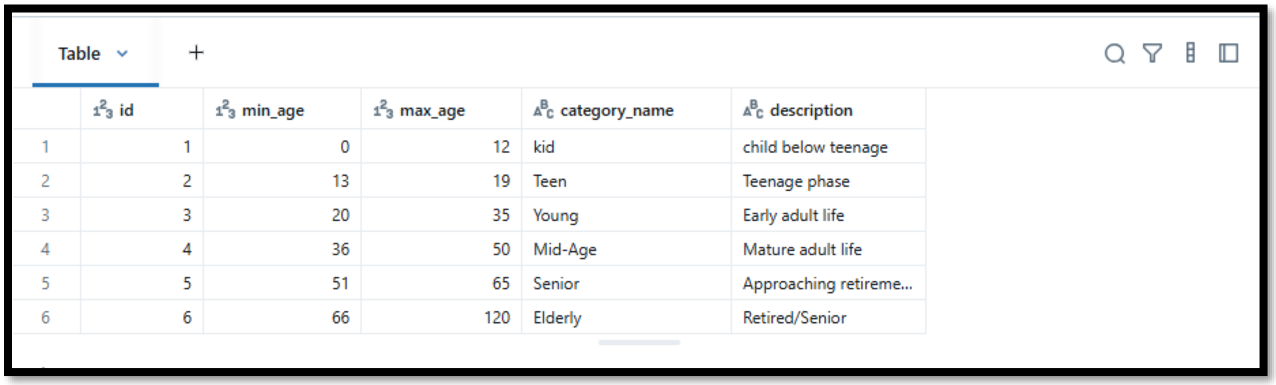
```
1,Aditi,Python,300,200,2024-11-01T00:00:00.000Z
2,Raj,Java,500,0,2024-10-28T00:00:00.000Z
3,Meera,SQL,200,300,2024-10-20T00:00:00.000Z
4,Arjun,Azure,600,0,2024-11-05T00:00:00.000Z
5,Simran,Python,100,400,2024-11-10T00:00:00.000Z
```

<u>3 Data:-Age_category_lookup:-</u>



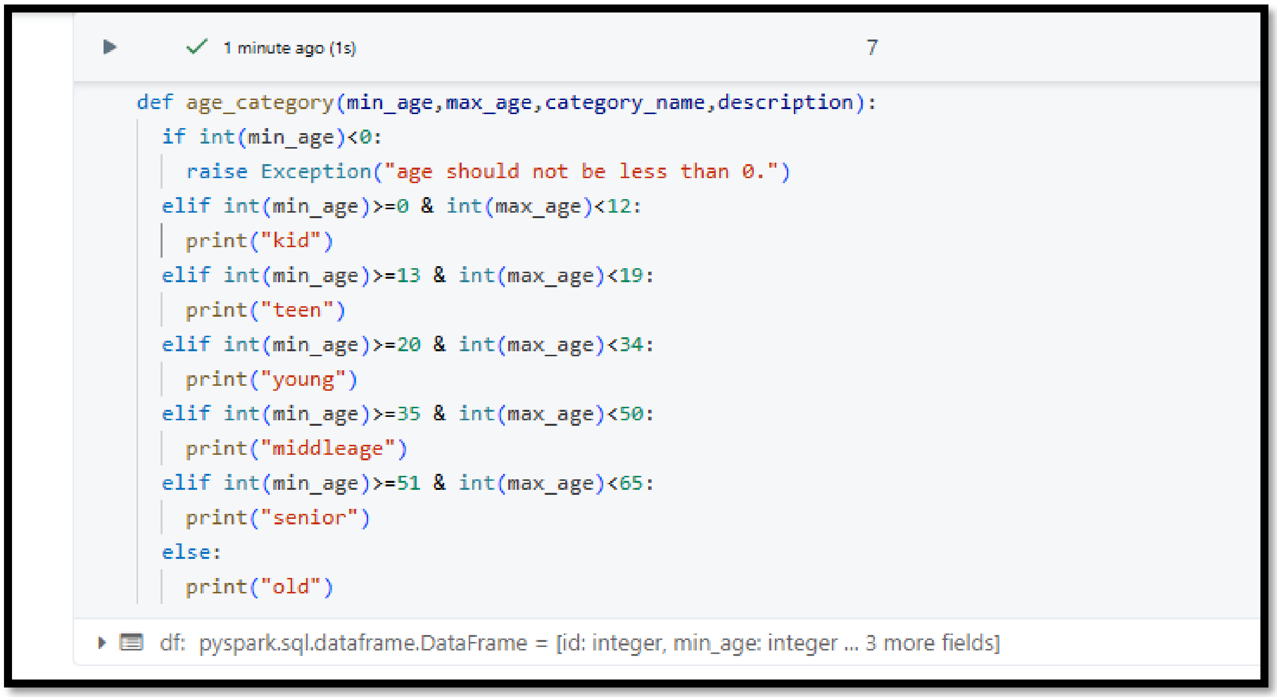
```
df=(spark.read
    .format("jdbc")
    .option("url", "jdbc:sqlserver://sqladminn2.database.windows.net:1433;database=sql_db;")
    .option("dbtable", "age_category_lookup")
    .option("user", "sqladminN2")
    .option("password", "Ybcfp5xe7k")
    .load()
)
display(df)
```

Read table age_category_lookup in table.



Table

	id	min_age	max_age	category_name	description
1	1	0	12	kid	child below teenage
2	2	13	19	Teen	Teenage phase
3	3	20	35	Young	Early adult life
4	4	36	50	Mid-Age	Mature adult life
5	5	51	65	Senior	Approaching retireme...
6	6	66	120	Elderly	Retired/Senior



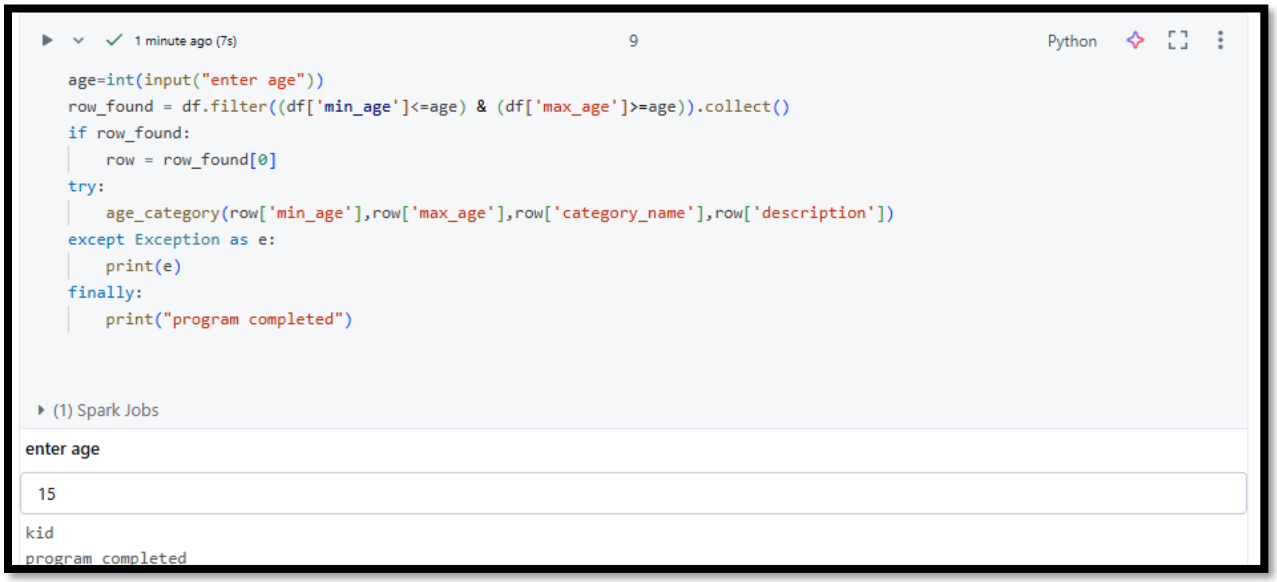
1 minute ago (1s) 7

```
def age_category(min_age,max_age,category_name,description):
  if int(min_age)<0:
    raise Exception("age should not be less than 0.")
  elif int(min_age)>=0 & int(max_age)<12:
    print("kid")
  elif int(min_age)>=13 & int(max_age)<19:
    print("teen")
  elif int(min_age)>=20 & int(max_age)<34:
    print("young")
  elif int(min_age)>=35 & int(max_age)<50:
    print("middleage")
  elif int(min_age)>=51 & int(max_age)<65:
    print("senior")
  else:
    print("old")
```

df: pyspark.sql.dataframe.DataFrame = [id: integer, min_age: integer ... 3 more fields]

Define function and using if-else.

I use if-elif-else here because it has multiple conditions.



```
age=int(input("enter age"))
row_found = df.filter((df['min_age']<=age) & (df['max_age']>=age)).collect()
if row_found:
    row = row_found[0]
try:
    age_category(row['min_age'],row['max_age'],row['category_name'],row['description'])
except Exception as e:
    print(e)
finally:
    print("program completed")
```

```
kid
program completed
```

Then calling function in another notebook and doing exception handling.

Row_found=df.filter((df[‘min_age’]<=age) &(df[‘max_age’]>=age)).collect()

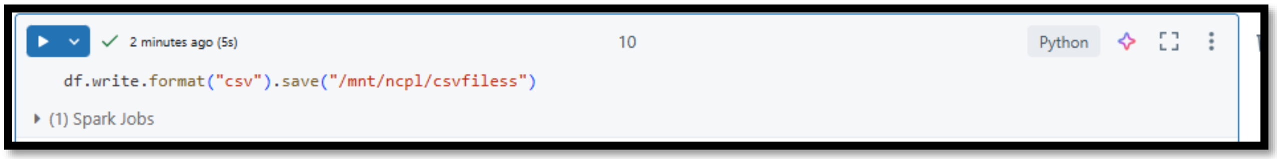
Each row in the Spark DataFrame df, and for each row, **call the function category_age().**

.collect() :- Converts the whole Spark DataFrame into a list of Row objects .

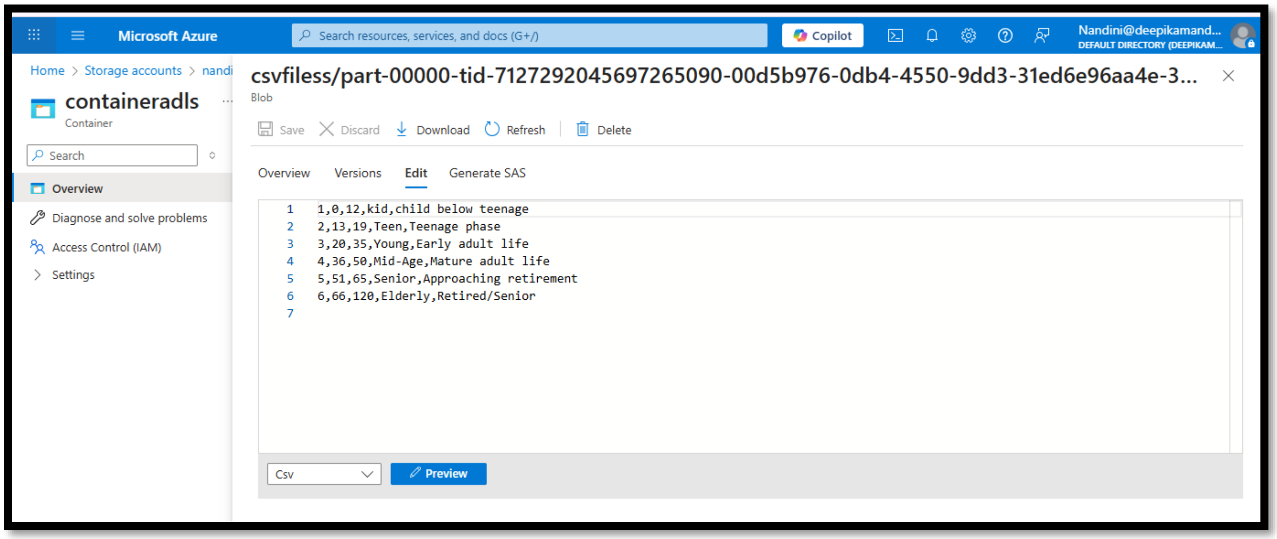
For row:- Loop over each row[min_age & max_age] .

Age_category is a function that we are calling it **once per row.**

WRITE SQL TABLE IN ADLS IN CSV FILES.



```
df.write.format("csv").save("/mnt/ncpl/csvfiless")
```


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
Nandini@deepikamand...
DEFAULT DIRECTORY (DEEPIKAM...
Home > Storage accounts > nandi
containeradls
Container
Search
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
csvfiless/part-00000-tid-7127292045697265090-00d5b976-0db4-4550-9dd3-31ed6e96aa4e-3...
Blob
Save
Discard
Download
Refresh
Delete
Overview
Versions
Edit
Generate SAS
Csv
Preview

```
1,0,12,kid,child below teenage
2,13,19,Teen,Teenage phase
3,20,35,Young,Early adult life
4,36,50,Mid-Age,Mature adult life
5,51,65,Senior,Approaching retirement
6,66,120,Elderly,Retired/Senior
```

Csv file in adls storage account.

CONCLUSION:-

1. To ensure scalability, reusability, and maintainability, this project followed a modular approach by dividing the workflow into separate Databricks notebooks.
2. The first notebook was responsible for securely connecting to and reading data from Azure SQL Database using JDBC.
3. A second notebook was dedicated to defining reusable transformation functions, allowing for cleaner code and easier debugging.
4. The final notebook brought everything together by calling the other two notebooks. This method made the project more organized, easier to understand, and better for future updates. If anything needs to change later, we can do it in one place without changing the whole project.