
JSON V

Managed Resource Group : [databricks-rg-databricksmaster-5yny0jx53ss](#)

Copy to clipboard

URL : <https://adb-3950914461847041.1.azuredatabricks.net>

Pricing Tier : [Trial \(Premium - 14-Days Free DBUs\) \(Click to change\)](#)

Enable No Public IP : Yes



Add #secrets/createScope

Create Secret Scope

Cancel

Create

A store for secrets that is identified by a name and backed by a specific store type. [Learn more](#)


Scope Name 

aryascope

Manage Principal 

Creator



Azure Key Vault 

DNS Name

https://xxx.vault.azure.net/

Resource ID

/subscriptions/xxxxxx/...

Copy keyvault url and access

masterclasssma01 | Properties ☆ ...

Key vault

arch



Save



Discard changes



Refresh

Access policies

Source visualizer

ents

jects

Keys

Secrets

Certificates

Settings

Access configuration

Networking

Microsoft Defender for Cloud

Properties

Locks

Monitoring

Automation

Help

Name

masterclasssma01

Sku (Pricing tier)

Standard

Location

eastus

Vault URI

https://masterclass

Resource ID

/subscriptions/4089

Subscription ID

4089499a-d7af-4089

Subscription Name

Azure subscription

Directory ID

0058073a-35a8-44

Directory Name

Default Directory

Soft-delete

Soft delete has been

Days to retain deleted vaults

90

Purge protection



Disable purge protection



Enable purge protection

Remove favorites by pressing Ctrl+Shift+F



Copy vault URI and resource ID to create scope page

⚙️ **dbutils.secrets**

```
▶ ✓ 1 minute ago (3s)
dbutils.secrets.list(scope='aryascope')
[SecretMetadata(key='appsecret')]
```

```
⚙️ ▶ ✓ Just now (1s)
▼ dbutils.secrets.get(scope='aryascope', key='appsecret')
'[REDACTED]'
```

Use of secrets scope is you can prevent use of hardcoded secret and tenant id values using Azure Key vaults where you can u store secrets .

IAM-- assing key vault admin role

Grab the secret value which you created previously from the microsoft entraid and paste as below under vault→ secrets

Microsoft Azure

Search resources, services, and docs (G+)

[Home](#) > [masterclasssma01](#) | [Secrets](#) > [appsecret](#) >

c036d0d18d91488cbfdbb16a5d20af990

Secret Version

Properties

Created

4/26/2025, 4:09:31 PM

Updated

4/26/2025, 4:09:31 PM

Secret Identifier

https://masterclasssma01.vault.azure.net/secrets/c036d0d18d91488cbfdbb16a5d20af990/4/26/2025, 4:09:31 PM

Settings

Set activation date

☐

Set expiration date

☐

Enabled

Yes

No

Tags

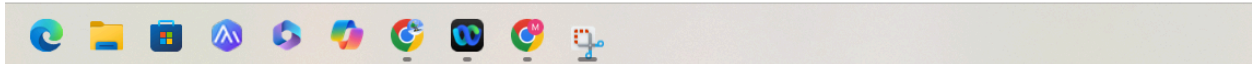
0 tags

Secret content type (optional)

Show Secret Value

Secret value

.....



DELTA LAKE:

Contains transaction log, it will read metadata from transaction log
Format is parquet

Overwrite and next append

ame

8,523 rows | 0.68s runtime

Refreshed 162 days ago

Python

```
%python
# Remove any existing files in the directory
dbutils.fs.rm('abfss://destination@datalakesmas.dfs.core.windows.net/sales', recurse=True)

# Initialize the Delta table
df_sales.write.format('delta')\
  .mode('overwrite')\
  .option('path', 'abfss://destination@datalakesmas.dfs.core.windows.net/sales')\
  .save()

# Append data to the Delta table
df_sales.write.format('delta')\
  .mode('append')\
  .option('path', 'abfss://destination@datalakesmas.dfs.core.windows.net/sales')\
  .save()
```

(2) Spark Jobs

```
%python
# Initialize the Delta table
df_sales.write.format('delta')\
  .mode('overwrite')\
  .option('path', 'abfss://destination@datalakesmas.dfs.core.windows.net/sales')\
  .save()

# Append data to the Delta table
df_sales.write.format('delta')\
  .mode('append')\
  .option('path', 'abfss://destination@datalakesmas.dfs.core.windows.net/sales')\
  .save()
```

This code first initializes the Delta table by writing to it in `overwrite` mode, which creates the necessary transaction log. Then, it appends data to the Delta table.

Tutorial (1)

@ for objects or / for commands

Deltag log:

Upload | Add directory | Refresh | \ | Rename | Delete | Change dir | Request read | Break read | Save feedback

Authentication method: Access key (Switch to Microsoft Entra user account)
Location: destination / sales / _delta_log

Search blobs by prefix (case-sensitive)

Show deleted objects

Name	Modified	Access tier	Archive status	Blob type	Size	Leas
[.]						
__tmp_path_dir	4/27/2025, 9:53:25 PM					-
_commits	4/27/2025, 9:53:22 PM					-
00000000000000000000.crc	4/27/2025, 9:53:26 PM	Hot (Inferred)		Block blob	4.42 KiB	Avail
00000000000000000000.json	4/27/2025, 9:53:25 PM	Hot (Inferred)		Block blob	3.41 KiB	Avail
0000000000000000000001.crc	4/27/2025, 9:53:27 PM	Hot (Inferred)		Block blob	5.79 KiB	Avail
0000000000000000000001.json	4/27/2025, 9:53:27 PM	Hot (Inferred)		Block blob	1.98 KiB	Avail

Metadata is in json file.

Upload

Add Directory

Refresh

Rename

Delete

Change tier

Acquire lease

Break lease







Give feedback

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

Location: [destination](#) / [salesDB](#) / [exttable](#)

Search blobs by prefix (case-sensitive)

Show deleted objects

Name	Modified	Access tier	Archive status	Blob type	Size	Lease
<input type="checkbox"/>  [-]						
<input type="checkbox"/>  _delta_log	5/3/2025, 2:34:53 PM					-
<input type="checkbox"/>  deletion_vector_3c40c204-00cf-40f4-b619-ca4e43...	5/3/2025, 3:00:35 PM	Hot (Inferred)		Block blob	43 B	Availa
<input checked="" type="checkbox"/>  part-00000-62f64820-726b-4b47-a3f5-ff06e0d661...	5/3/2025, 2:56:38 PM	Hot (Inferred)		Block blob	1.08 KiB	Availa
<input type="checkbox"/>  part-00000-71a9629c-6a8e-4658-a870-70bd0f588...	5/3/2025, 3:00:43 PM	Hot (Inferred)		Block blob	1.16 KiB	Availa
<input type="checkbox"/>  part-00000-e36df7ef-9aa4-4d82-a263-b5d03e35a...	5/3/2025, 2:36:55 PM	Hot (Inferred)		Block blob	1.08 KiB	Availa

Tombstoning

When you perform delete, it copies existing data,
Tombstoning (ghost mode existing partitions) and read only one partition

existing parquets remaining it does not delete

salesDB/exttable/_delta_log/00000000000000000004.json

Blob

Save Discard Download Refresh Delete

Overview Versions Edit Generate SAS

1 {"commitInfo":{"timestamp":1746298843737,"userId":"111888533905749","userName":"mith
2 {"remove":{"path":"part-00000-e36df7ef-9aa4-4d82-a263-b5d03e35a459-c000.snappy.parque
3 {"remove":{"path":"part-00000-62f64820-726b-4b47-a3f5-ff06e0d661fa-c000.snappy.parque
4 {"add":{"path":"part-00000-71a9629c-6a8e-4658-a870-70bd0f5881c2-c000.snappy.parquet",
5

Search blobs by prefix (cas
Show deleted objects

Name

- [.]
- __tmp_path_dir
- __commits
- 00000000000000000000.crc
- 00000000000000000000.json
- 00000000000000000001.crc
- 00000000000000000001.json
- 00000000000000000002.crc
- 00000000000000000002.json
- 00000000000000000003.crc
- 00000000000000000003.json
- 00000000000000000004.crc

Json Preview

Forbidding concept

Accept and run Ctrl + Enter Accept Ctrl + Enter Reject Ctrl + Enter

DESCRIBE HISTORY salesdb.exttable;

Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [version: long, timestamp: timestamp ... 13 more fields]

table +

1 2 3 ve...	timestamp	userId	userName	operation	operationParameters
4	2025-05-03T19:00:43.000+00:...	1118885339057...	mithelesharya@gmail.co...	OPTIMIZE	> {"predicate":[""],"aut...
3	2025-05-03T19:00:37.000+00:...	1118885339057...	mithelesharya@gmail.co...	DELETE	> {"predicate":["\"(id#9...
2	2025-05-03T18:56:38.000+00:...	1118885339057...	mithelesharya@gmail.co...	WRITE	> {"mode":"Append","s...
1	2025-05-03T18:36:56.000+00:...	1118885339057...	mithelesharya@gmail.co...	WRITE	> {"mode":"Append","s...
0	2025-05-03T18:34:53.000+00:...	1118885339057...	mithelesharya@gmail.co...	CREATE TABLE	> {"partitionBy":["\"(id#...

5 rows | 1.58s runtime

Why we need history:

To get the databack , like rollback using concept of time travel we bring the databack

Run suggested ✓ 3 minutes ago (36s) 32

```
%sql RESTORE TABLE salesdb.exttable TO VERSION AS OF 2;
%sql
```

Accept and run CTRL + ⌘ Accept CTRL + ` Reject ESC

```
RESTORE TABLE salesdb.exttable TO VERSION AS OF 2;
```

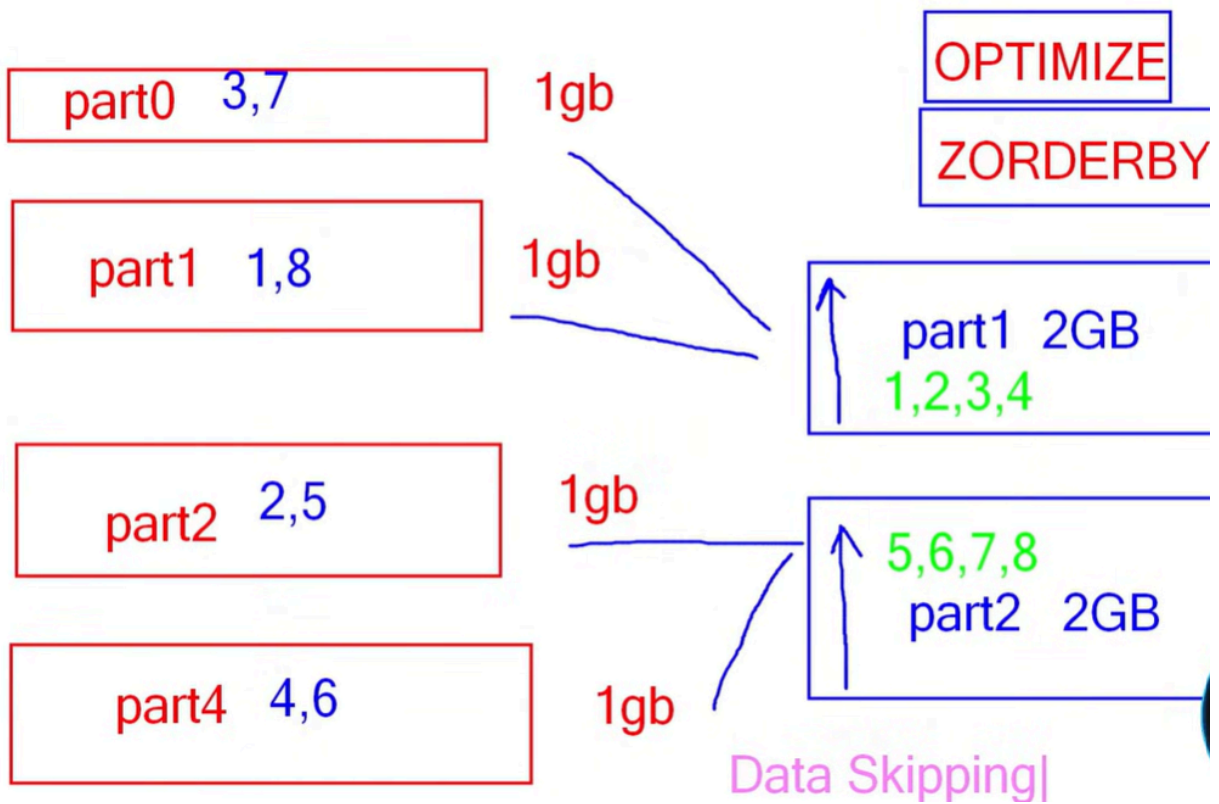
▶ (18) Spark Jobs

▶ _sqlidf: pyspark.sql.dataframe.DataFrame = [table_size_after_restore: long, num_of_files_after_restore: long ... 4 more fields]

Table +

	¹ ₃ table_size_after_restore	¹ ₃ num_of_files_after_restore	¹ ₃ num_removed_files	¹ ₃ num_restored_files
1	2210	2	0	0

DATA SKIPPING:



```
%sql
```

```
OPTIMIZE salesdb.exttable ZORDER BY (id)
```

From 1.65 to 0.62 sec ,performance increased

▼

```
%sql
select * from salesDB.exttable;
```

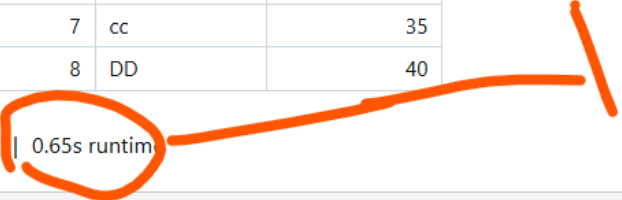
► (1) Spark Jobs

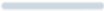
► `_sqldf`: pyspark.sql.dataframe.DataFrame = [id: integer, name: string ... 1 more field]


Table ▼ +

	¹ ₃ id	¹ ₃ name	¹ ₃ marks
1	1	aa	30
2	2	bb	33
3	3	cc	35
4	4	DD	40
5	5	aa	30
6	6	bb	33
7	7	cc	35
8	8	DD	40

↓ 8 rows | 0.65s runtime





 This result is stored as `_sqldf` and can be used in other [Python](#) and [SQL](#) cells.