Azure Databricks Coding Challenge
Name : Mitushi Vishwakarma

Question 2 :Explain Overview of 3 level namespace and creating Unity Catalog objects.

Unity Catalog is a powerful tool in Databricks that offers centralized access control, auditing, lineage, and data discovery capabilities across all your workspaces. Think of it as the single source of truth for your data governance needs.

A three-level namespace in Unity Catalog consists of three levels of hierarchy — catalog, schema, and objects. Catalogs are like databases, schemas are like folders or directories, and objects can be tables, views, functions, or other entities.

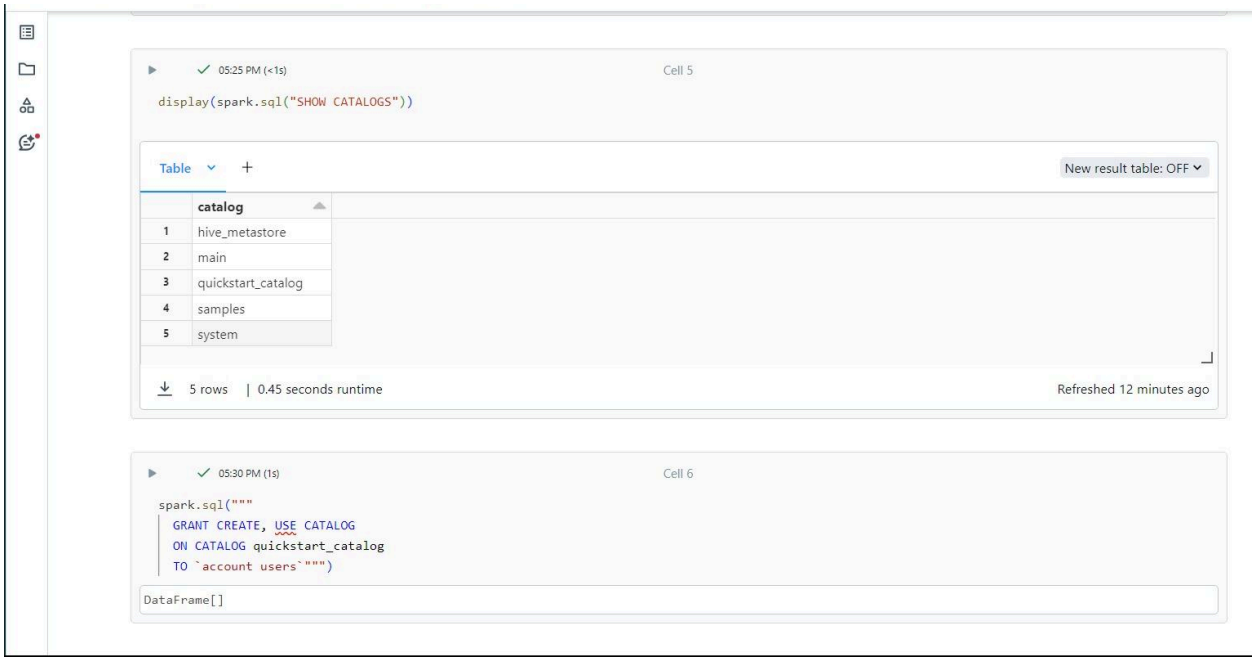
1st level :
Catalog: Catalogs are the highest level in the data hierarchy (catalog > schema > table/view/volume) managed by the Unity Catalog metastore. They are intended as the primary unit of data isolation in a typical Databricks data governance model.

2nd level :
Schema (Database): Schemas, also known as databases, are logical groupings of tabular data (tables and views), non-tabular data (volumes), functions, and machine learning models. They give you a way to organize and control access to data that is more granular than catalogs. Typically they represent a single use case, project, or team sandbox.

3rd level :
Tables: Tables reside in the third layer of Unity Catalog's three-level namespace. They contains rows of data.
Views: A view is a read-only object derived from one or more tables and views in a metastore.
Volumes: Volumes reside in the third layer of Unity Catalog's three-level namespace. They manage non-tabular data.
Models and functions: Although they are not, strictly speaking, data assets, registered models and user-defined functions can also be managed in Unity Catalog and reside at the lowest level in the object hierarchy.

Creating Unity Catalog Object :


05:21 PM (<1s)
Cell 1
from pyspark.sql import SparkSession
05:21 PM (<1s)
Cell 2
spark=SparkSession.Builder().appName('Unity').getOrCreate()
05:21 PM (5s)
Cell 3
spark.sql("CREATE CATALOG IF NOT EXISTS quickstart_catalog")
DataFrame[]
05:25 PM (1s)
Cell 4
spark.sql("USE CATALOG quickstart_catalog")
DataFrame[]


05:25 PM (<1s)
Cell 5
display(spark.sql("SHOW CATALOGS"))
Table
New result table: OFF
catalog
1 hive_metastore
2 main
3 quickstart_catalog
4 samples
5 system
5 rows | 0.45 seconds runtime
Refreshed 12 minutes ago
05:30 PM (1s)
Cell 6
spark.sql("""
GRANT CREATE, USE CATALOG
ON CATALOG quickstart_catalog
TO `account users`""")
DataFrame[]

Cell 7

05:31 PM (1s)

```
display(spark.sql("SHOW GRANT ON CATALOG quickstart_catalog"))
```

Table +

New result table: OFF

	Principal	ActionType	ObjectType	ObjectKey
1	account users	CREATE SCHEMA	CATALOG	quickstart_catalog
2	account users	USE CATALOG	CATALOG	quickstart_catalog

2 rows | 0.61 seconds runtime

Refreshed 7 minutes ago

Cell 8

05:31 PM (1s)

```
spark.sql("""
  CREATE SCHEMA IF NOT EXISTS quickstart_schema
  COMMENT 'A new Unity Catalog schema called quickstart_schema'""")
```

```
DataFrame[]
```

Cell 9

05:31 PM (1s)

```
display(spark.sql("SHOW SCHEMAS"))
```

Table +

New result table: OFF

	databaseName
1	default
2	information_schema
3	quickstart_schema

3 rows | 0.59 seconds runtime

Refreshed 7 minutes ago

Cell 10

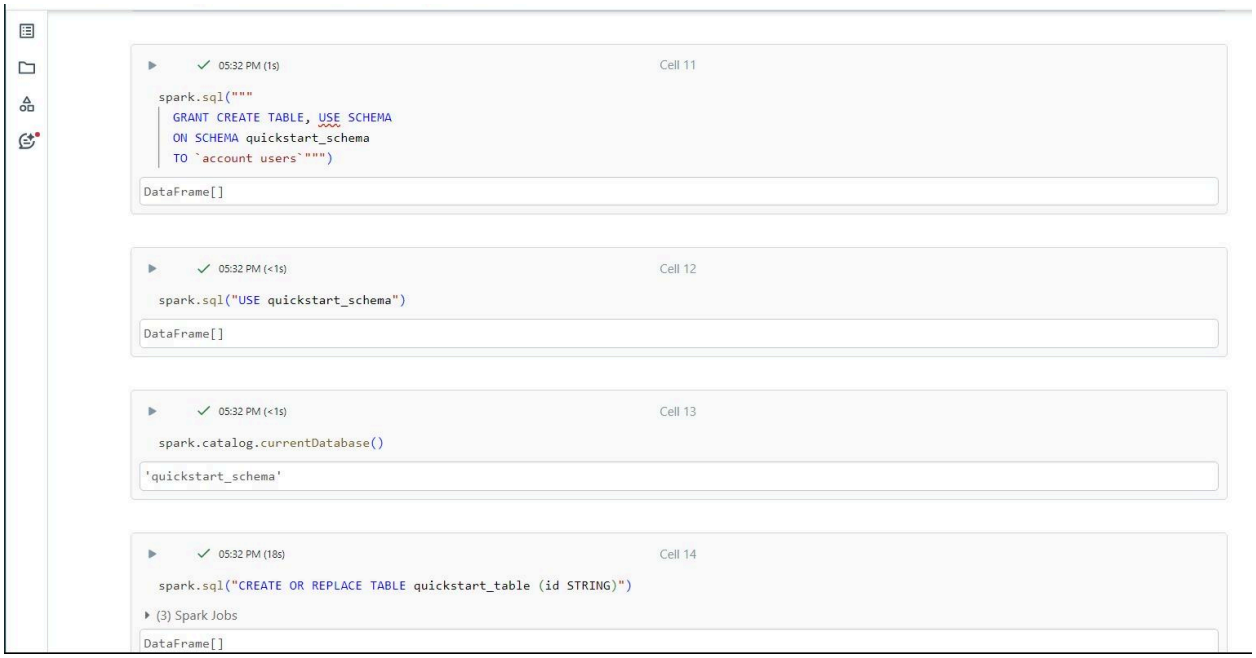
05:31 PM (<1s)

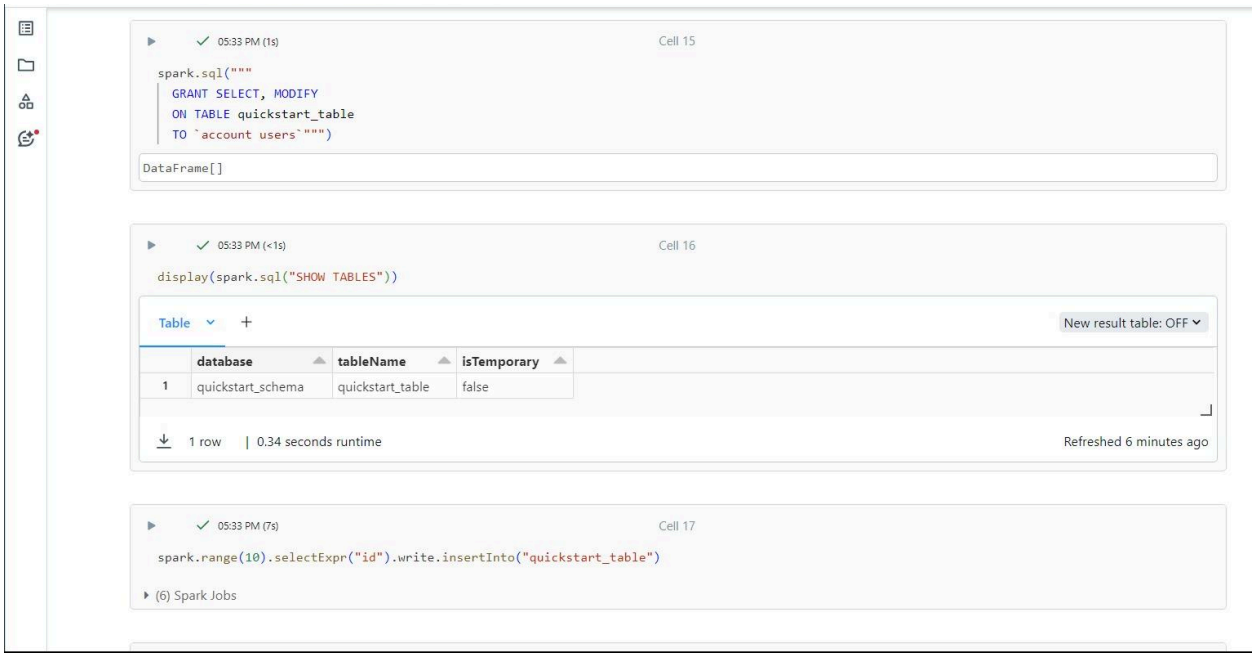
```
display(spark.sql("DESCRIBE SCHEMA EXTENDED quickstart_schema"))
```

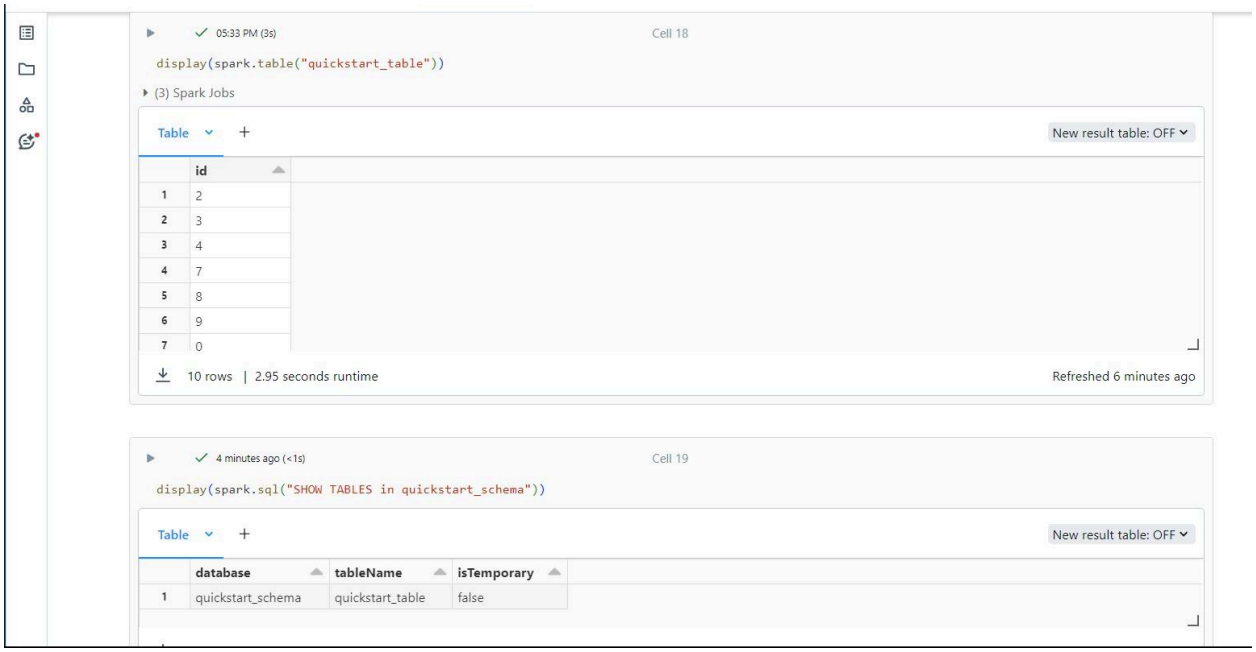
Table +

New result table: OFF

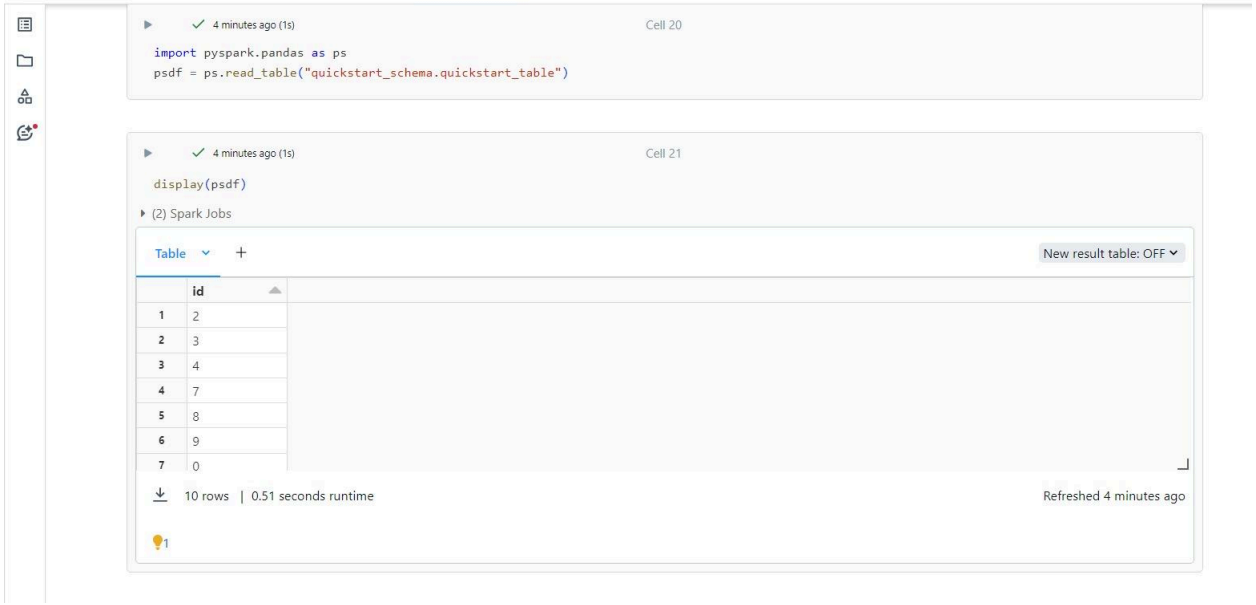
	database_description_item	database_description_value
1	Catalog Name	quickstart_catalog
2	Namespace Name	quickstart_schema
3	Comment	A new Unity Catalog schema called quickstart_schema
4	Location	
5	Owner	pratikwani116@outlook.com
6	Properties	

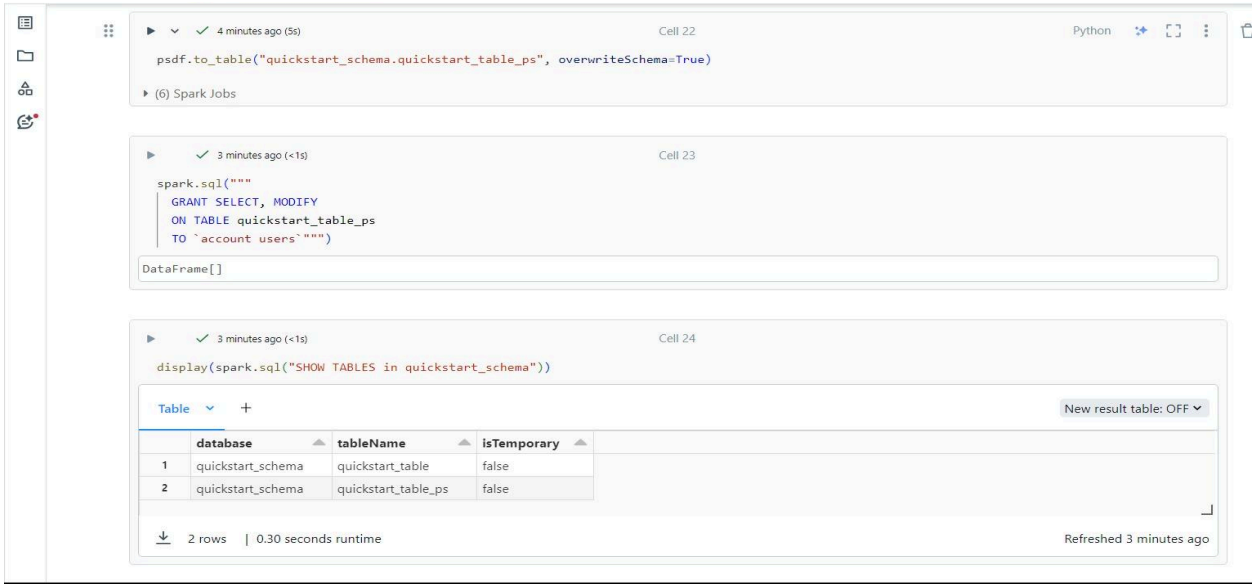

05:32 PM (1s)
Cell 11
spark.sql("""
GRANT CREATE TABLE, USE SCHEMA
ON SCHEMA quickstart_schema
TO `account users`""")
DataFrame[]
05:32 PM (<1s)
Cell 12
spark.sql("USE quickstart_schema")
DataFrame[]
05:32 PM (<1s)
Cell 13
spark.catalog.currentDatabase()
'quickstart_schema'
05:32 PM (18s)
Cell 14
spark.sql("CREATE OR REPLACE TABLE quickstart_table (id STRING)")
(3) Spark Jobs
DataFrame[]

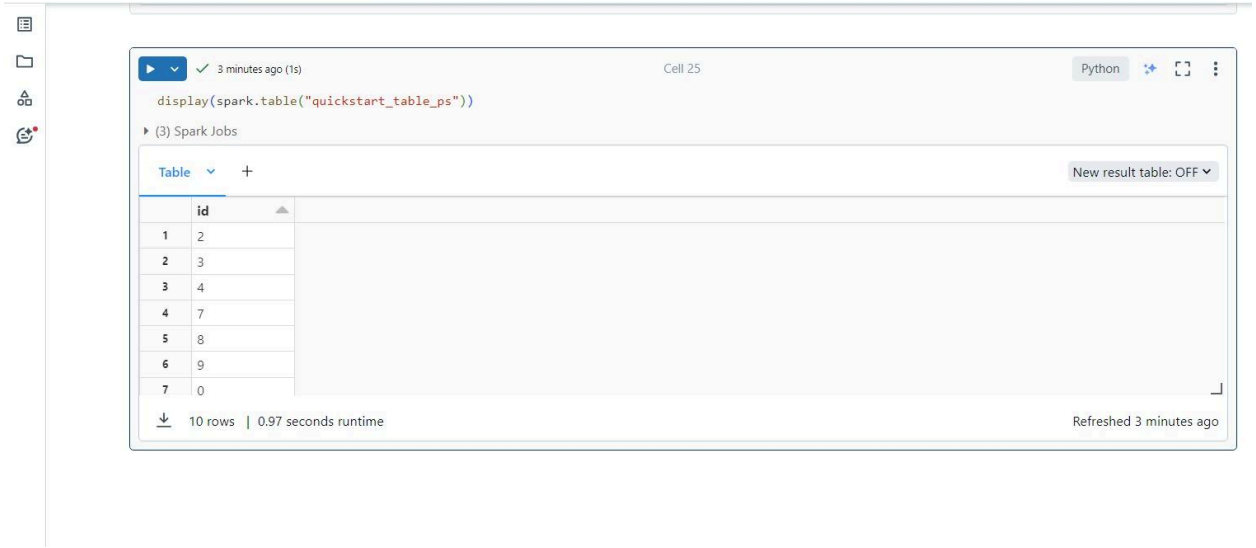

05:33 PM (1s)
Cell 15
spark.sql("""
GRANT SELECT, MODIFY
ON TABLE quickstart_table
TO `account users`""")
DataFrame[]
05:33 PM (<1s)
Cell 16
display(spark.sql("SHOW TABLES"))
Table
New result table: OFF
database
tableName
isTemporary
1
quickstart_schema
quickstart_table
false
1 row | 0.34 seconds runtime
Refreshed 6 minutes ago
05:33 PM (7s)
Cell 17
spark.range(10).selectExpr("id").write.insertInto("quickstart_table")
(6) Spark Jobs


05:33 PM (3s)
Cell 18
display(spark.table("quickstart_table"))
(3) Spark Jobs
Table
New result table: OFF
id
1 2
2 3
3 4
4 7
5 8
6 9
7 0
10 rows | 2.95 seconds runtime
Refreshed 6 minutes ago

4 minutes ago (<1s)
Cell 19
display(spark.sql("SHOW TABLES in quickstart_schema"))
Table
New result table: OFF
database
tableName
isTemporary
1 quickstart_schema quickstart_table false


4 minutes ago (1s)
Cell 20
import pyspark.pandas as ps
psdf = ps.read_table("quickstart_schema.quickstart_table")
4 minutes ago (1s)
Cell 21
display(psdf)
(2) Spark Jobs
Table
New result table: OFF
id
1 2
2 3
3 4
4 7
5 8
6 9
7 0
10 rows | 0.51 seconds runtime
Refreshed 4 minutes ago


4 minutes ago (5s)
Cell 22
Python
psdf.to_table("quickstart_schema.quickstart_table_ps", overwriteSchema=True)
(6) Spark Jobs
3 minutes ago (<1s)
Cell 23
spark.sql("""
GRANT SELECT, MODIFY
ON TABLE quickstart_table_ps
TO `account users`""")
DataFrame[]
3 minutes ago (<1s)
Cell 24
display(spark.sql("SHOW TABLES in quickstart_schema"))
Table
New result table: OFF
database
tableName
isTemporary
1 quickstart_schema quickstart_table false
2 quickstart_schema quickstart_table_ps false
2 rows | 0.30 seconds runtime
Refreshed 3 minutes ago


3 minutes ago (1s)
Cell 25
Python
display(spark.table("quickstart_table_ps"))
(3) Spark Jobs
Table
New result table: OFF
id
1 2
2 3
3 4
4 7
5 8
6 9
7 0
10 rows | 0.97 seconds runtime
Refreshed 3 minutes ago