INDIAN DATA CLUB

CODE BASICS

databricks

14 DAYS AI CHALLENGE

DAY 08

Topic:

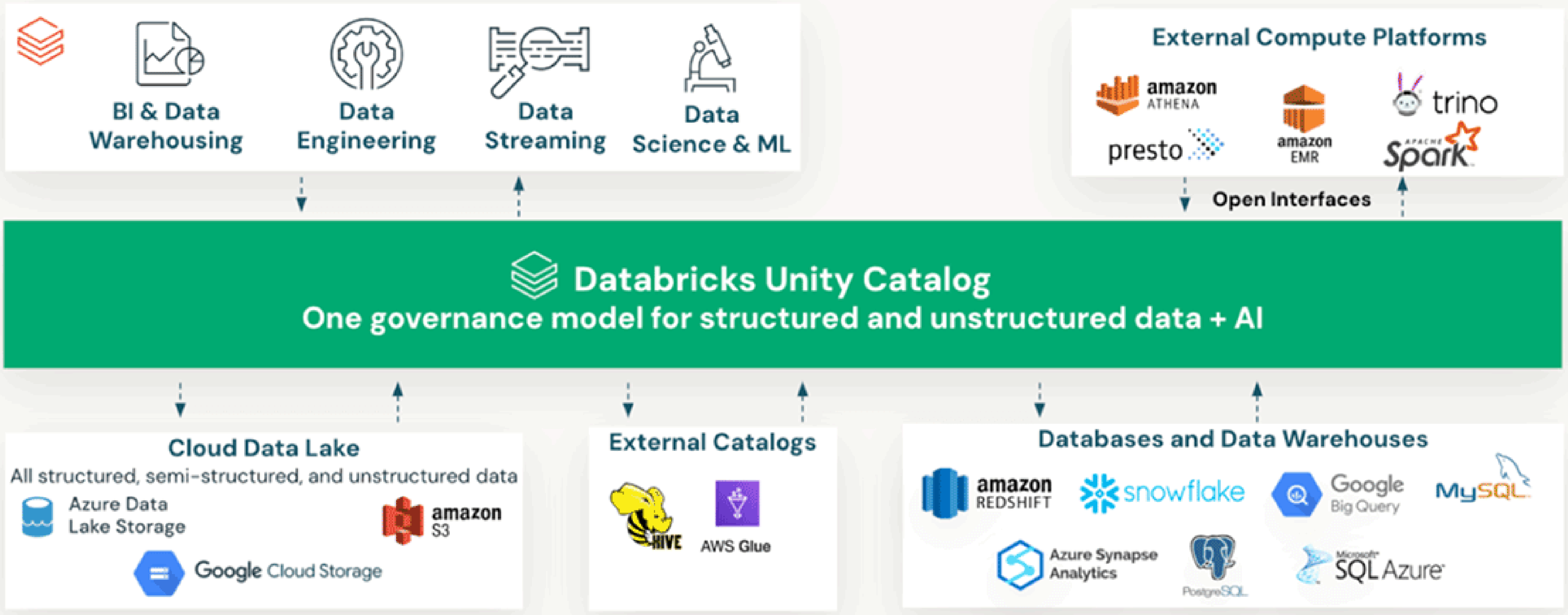
Unity Catalog Governance

Challenge:

1. Create catalog & schemas
2. Register Delta tables
3. Set up permissions
4. Create views for controlled access

#DatabricksWithIDC

Databricks Unifies Data and AI Governance



BI & Data Warehousing

Data Engineering

Data Streaming

Data Science & ML

External Compute Platforms

Open Interfaces

Databricks Unity Catalog

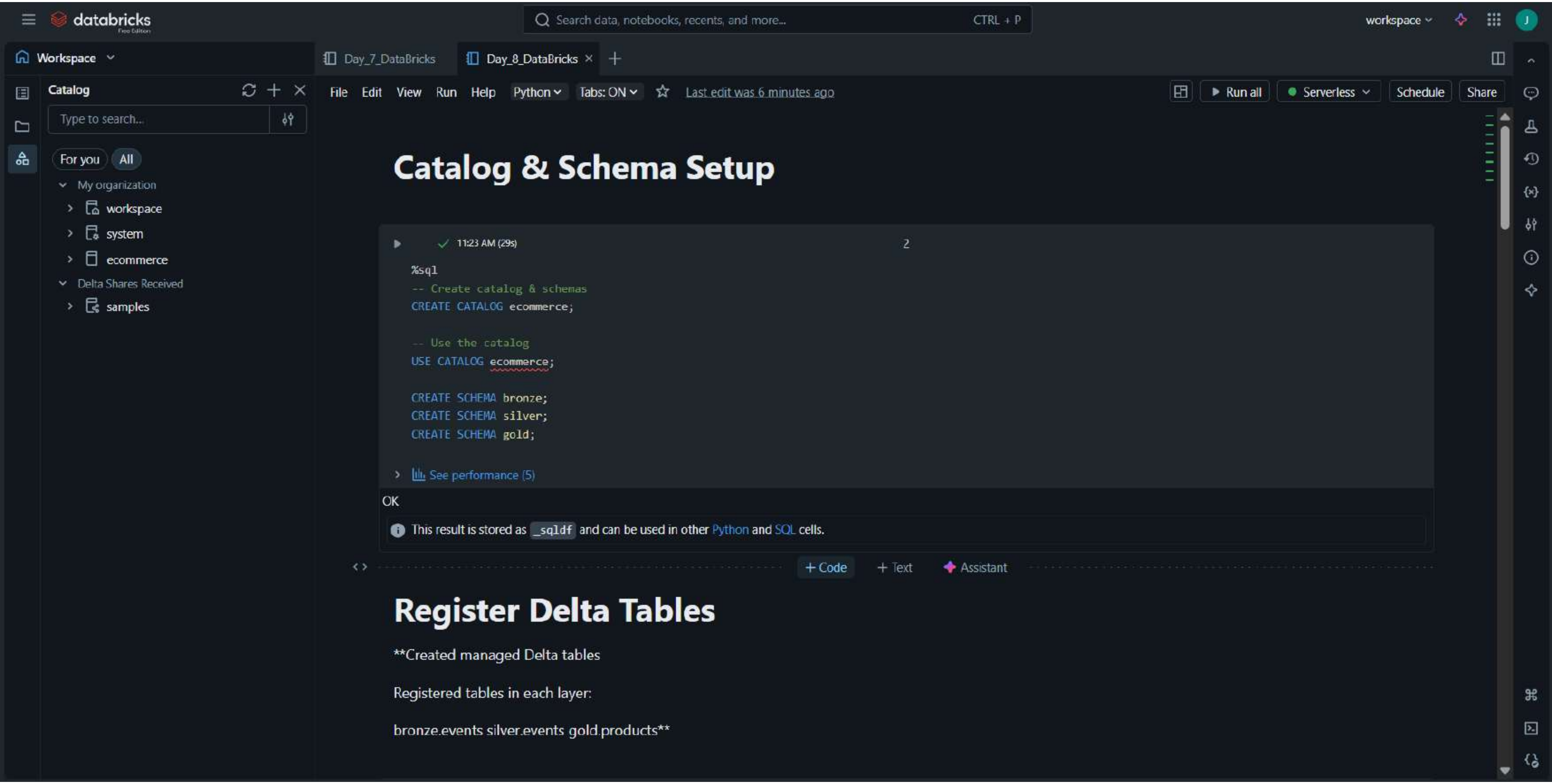
One governance model for structured and unstructured data + AI

Cloud Data Lake

All structured, semi-structured, and unstructured data

External Catalogs

Databases and Data Warehouses



Catalog & Schema Setup

```
%sql
-- Create catalog & schemas
CREATE CATALOG ecommerce;

-- Use the catalog
USE CATALOG ecommerce;

CREATE SCHEMA bronze;
CREATE SCHEMA silver;
CREATE SCHEMA gold;
```

```
OK
```

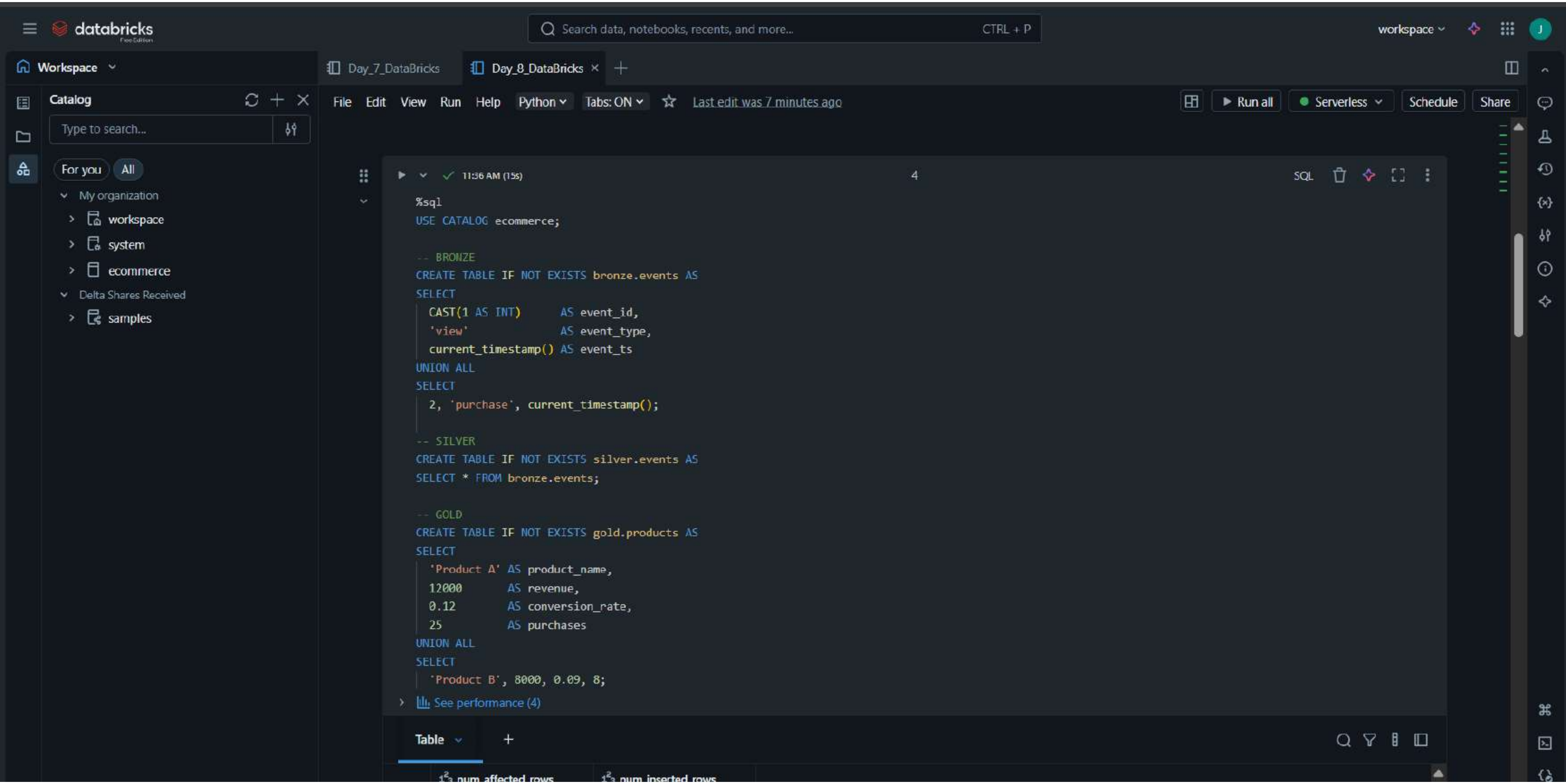
This result is stored as `_sqldf` and can be used in other Python and SQL cells.

Register Delta Tables

**Created managed Delta tables

Registered tables in each layer:

bronze.events silver.events gold.products**

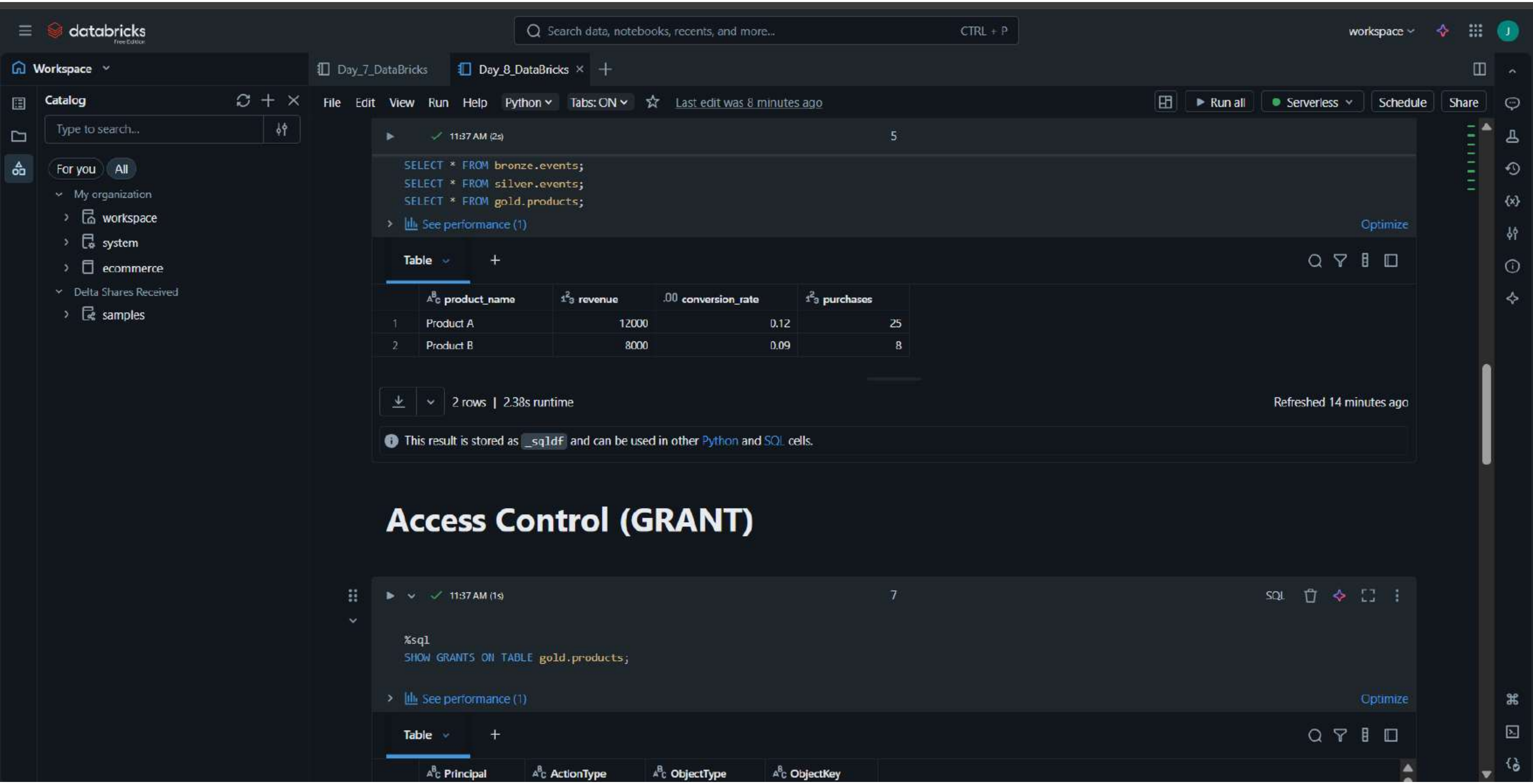


```
%sql
USE CATALOG ecommerce;

-- BRONZE
CREATE TABLE IF NOT EXISTS bronze.events AS
SELECT
  CAST(1 AS INT)      AS event_id,
  'view'              AS event_type,
  current_timestamp() AS event_ts
UNION ALL
SELECT
  2, 'purchase', current_timestamp();


-- SILVER
CREATE TABLE IF NOT EXISTS silver.events AS
SELECT * FROM bronze.events;

-- GOLD
CREATE TABLE IF NOT EXISTS gold.products AS
SELECT
  'Product A' AS product_name,
  12000       AS revenue,
  0.12        AS conversion_rate,
  25          AS purchases
UNION ALL
SELECT
  'Product B', 8000, 0.09, 8;
```



11:37 AM (2s) 5

```
SELECT * FROM bronze.events;
SELECT * FROM silver.events;
SELECT * FROM gold.products;
```

	product_name	revenue	conversion_rate	purchases
1	Product A	12000	0.12	25
2	Product B	8000	0.09	8

2 rows | 2.38s runtime

Refreshed 14 minutes ago

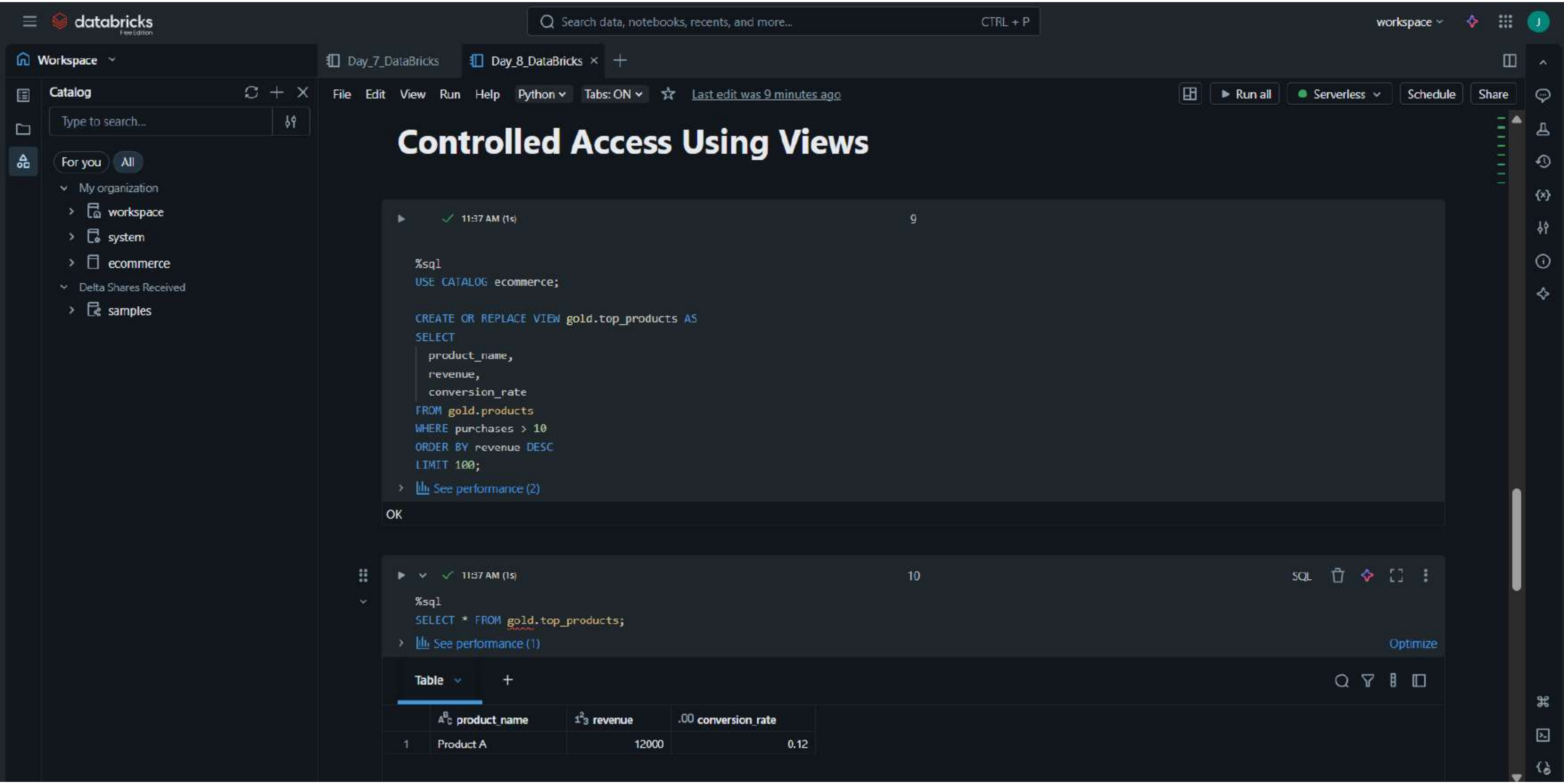
This result is stored as `_sqldf` and can be used in other Python and SQL cells.

Access Control (GRANT)

11:37 AM (1s) 7

```
%sql
SHOW GRANTS ON TABLE gold.products;
```

Principal	ActionType	ObjectType	ObjectKey



Controlled Access Using Views

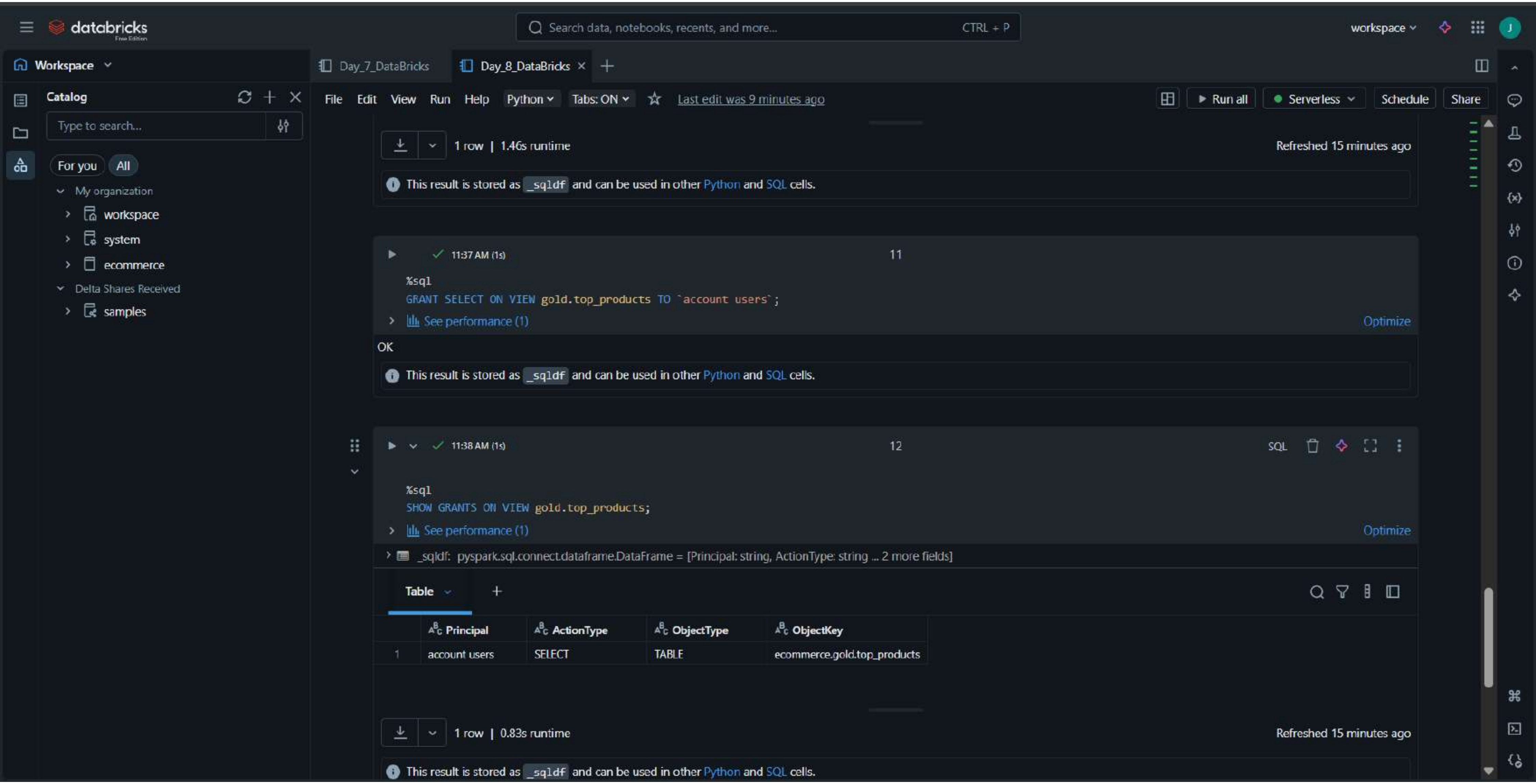
```
%sql
USE CATALOG ecommerce;

CREATE OR REPLACE VIEW gold.top_products AS
SELECT
  product_name,
  revenue,
  conversion_rate
FROM gold.products
WHERE purchases > 10
ORDER BY revenue DESC
LIMIT 100;
```

OK

```
%sql
SELECT * FROM gold.top_products;
```

	product_name	revenue	conversion_rate
1	Product A	12000	0.12



1 row | 1.46s runtime

Refreshed 15 minutes ago

This result is stored as `_sqldf` and can be used in other Python and SQL cells.

11:37 AM (1s) 11

```
%sql
GRANT SELECT ON VIEW gold.top_products TO `account users`;
```

See performance (1) Optimize

```
OK
```

This result is stored as `_sqldf` and can be used in other Python and SQL cells.

11:38 AM (1s) 12 SQL

```
%sql
SHOW GRANTS ON VIEW gold.top_products;
```

See performance (1) Optimize

_sqldf: pyspark.sql.connect.dataframe.DataFrame = [Principal: string, ActionType: string ... 2 more fields]

Table

	Principal	ActionType	ObjectType	ObjectKey
1	account users	SELECT	TABLE	ecommerce.gold.top_products

1 row | 0.83s runtime

Refreshed 15 minutes ago

This result is stored as `_sqldf` and can be used in other Python and SQL cells.