



14 DAYS

AI CHALLENGE

DAY 09

Topic:

SQL Analytics & Dashboards

Challenge:

- 1.Create SQL warehouse
- 2.Write analytical queries
- 3.Build dashboard
- 4.Add filters & schedule refresh

Sales Analytics Dashboard

Data | Dashboard Page | +

Sales Distribution by Region

1500

1000

500

0

region
East
North
South
West

North

South

East

West

Sum of amount

Total Sales Amount

2.53K

Select a widget to configure

Sales by Category

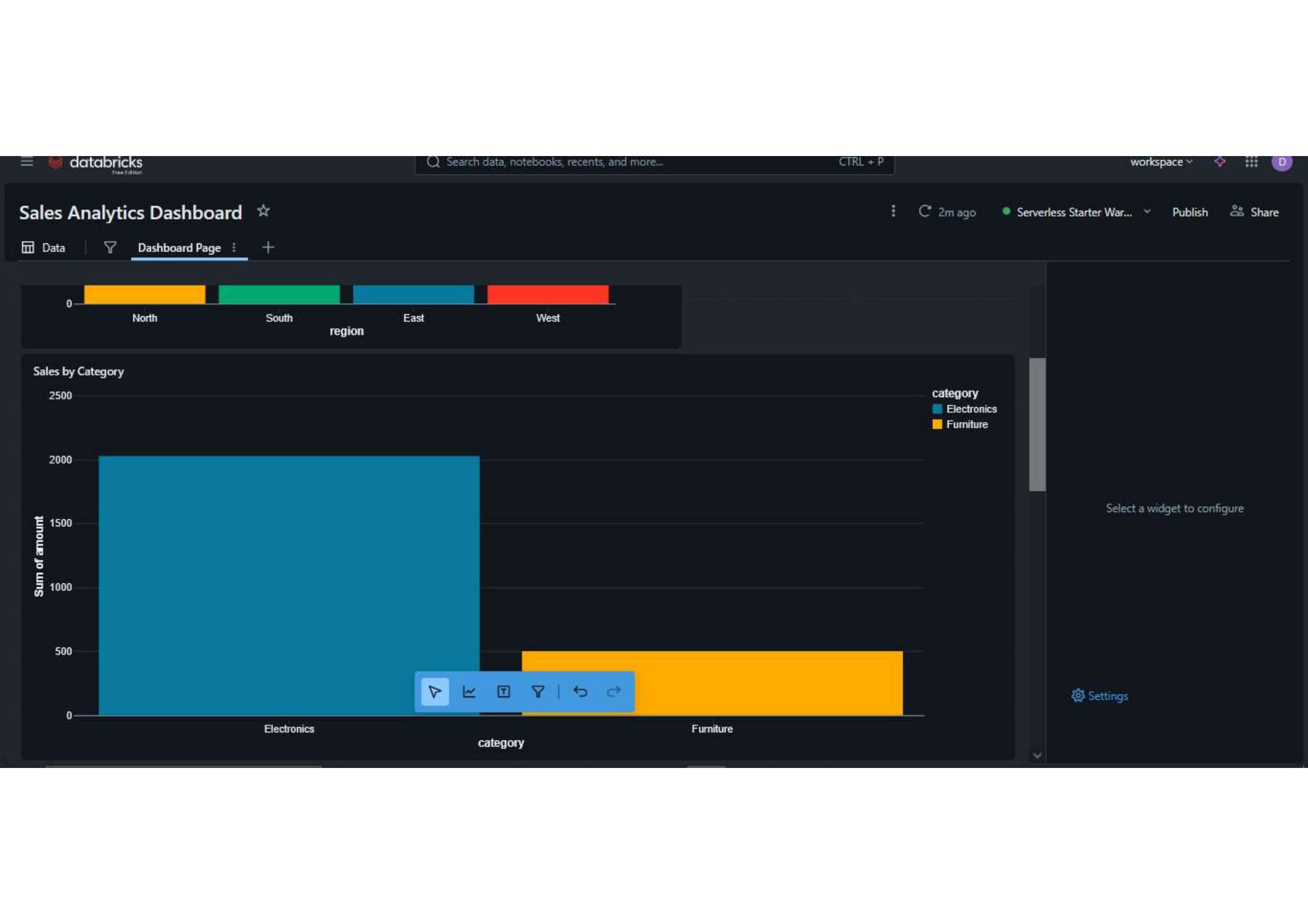
2500

2000

category
Electronics
Furniture



Settings



Sales Analytics Dashboard star

CTRL + P

workspace Serverless Starter War... Publish Share

Data Dashboard Page +

Product Sales Table

order_id	customer_id	product	category	amount	order_date	region
1	101	Laptop	Electronics	1200	15/01/2024	North
2	102	Phone	Electronics	800	16/01/2024	South
3	103	Desk	Furniture	350	17/01/2024	East
4	104	Chair	Furniture	150	18/01/2024	West
5	101	Mouse	Electronics	25	19/01/2024	North

Select a widget to configure

Ask the assistant to create a chart...

Preview ▶

Sales Over Time

Settings

Databricks Free Edition

Search data, notebooks, recents, and more...

CTRL + P

workspace Serverless Starter War... Publish Share

Data Dashboard Page +

Product Sales Table

order_id	customer_id	product	category	amount	order_date	region
1	101	Laptop	Electronics	1200	15/01/2024	North
2	102	Phone	Electronics	800	16/01/2024	South
3	103	Desk	Furniture	350	17/01/2024	East
4	104	Chair	Furniture	150	18/01/2024	West
5	101	Mouse	Electronics	25	19/01/2024	North

Select a widget to configure

Ask the assistant to create a chart...

Preview ▶

Sales Over Time

Settings

databricks

Search data, notebooks, recents, and more... CTRL + P workspace D

DAY 9 × +

File Edit View Run Help Python Tabs: ON Last edit was 1 hour ago

Run all Serverless Schedule Share

DAY 9 (17/01/26) – SQL Analytics & Dashboards

Markdown

Sales Dataset

Open focus mode (Ctrl + Alt + O)

```
%sql
-- Create a sample sales table
CREATE TABLE IF NOT EXISTS sales_data (
    order_id INT,
    customer_id INT,
    product STRING,
    category STRING,
    amount DECIMAL(10,2),
    order_date DATE,
    region STRING
);

-- Insert sample data
INSERT INTO sales_data VALUES
(1, 101, 'Laptop', 'Electronics', 1200.00, '2024-01-15', 'North'),
(2, 102, 'Phone', 'Electronics', 800.00, '2024-01-16', 'South'),
(3, 103, 'Desk', 'Furniture', 350.00, '2024-01-17', 'East'),
```

File Edit View Run Help Python ▾ Tabs: ON Last edit was 1 hour ago

Run all Serverless Schedule Share

10:50 AM (42s) 3 SQL

```
);  
  
-- Insert sample data  
INSERT INTO sales_data VALUES  
(1, 101, 'Laptop', 'Electronics', 1200.00, '2024-01-15', 'North'),  
(2, 102, 'Phone', 'Electronics', 800.00, '2024-01-16', 'South'),  
(3, 103, 'Desk', 'Furniture', 350.00, '2024-01-17', 'East'),  
(4, 104, 'Chair', 'Furniture', 150.00, '2024-01-18', 'West'),  
(5, 101, 'Mouse', 'Electronics', 25.00, '2024-01-19', 'North');
```

See performance (2)

Table + Q Y E

	num_affected_rows	num_inserted_rows
1	5	5

1 row | 42.09s runtime Refreshed 1 hour ago

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

Write Analytical Queries Query 1: Total Sales by Category

```
▶ ✓ 10:51 AM (2s) 5
%sql
SELECT
    category,
    COUNT(*) as total_orders,
    SUM(amount) as total_revenue,
    AVG(amount) as avg_order_value
FROM sales_data
GROUP BY category
ORDER BY total_revenue DESC;
> See performance (1) Optimize
```

Table Visualization 1 +

New charts: ON

The screenshot shows a data analysis environment with a dark theme. At the top, there's a header bar with icons for file, search, and user. Below it is a toolbar with various icons. The main area has a title "Write Analytical Queries Query 1: Total Sales by Category". Underneath, a code editor window displays an SQL query. The query selects data from a "sales_data" table, grouping by "category" and calculating "total_orders", "total_revenue", and "avg_order_value". It then orders the results by "total_revenue" in descending order. Below the code, there's a link to "See performance (1)" and an "Optimize" button. The interface then transitions into a visualization section. It shows a table with two rows and a chart below it. The table has columns for "category" (Category A and Category B), "total_orders" (2025 and 500), and "total_revenue" (2025 and 500). To the right of the table is a legend titled "total_revenue" with entries for "2025" (blue square) and "500" (orange square). Below the table and legend is a large, mostly empty chart area.

category	total_orders	total_revenue
Category A	2025	2025
Category B	500	500

total_revenue

- 2025
- 500

Query 2: Sales Trend by Date

✓ 10:52 AM (2s)

```
%sql  
SELECT  
    order_date,  
    COUNT(*) as orders,  
    SUM(amount) as daily_revenue  
FROM sales_data  
GROUP BY order_date  
ORDER BY order_date;
```

7

See performance (1)

Optimize

Table Visualization 1 +

New charts: ON

1200

1000

800

600

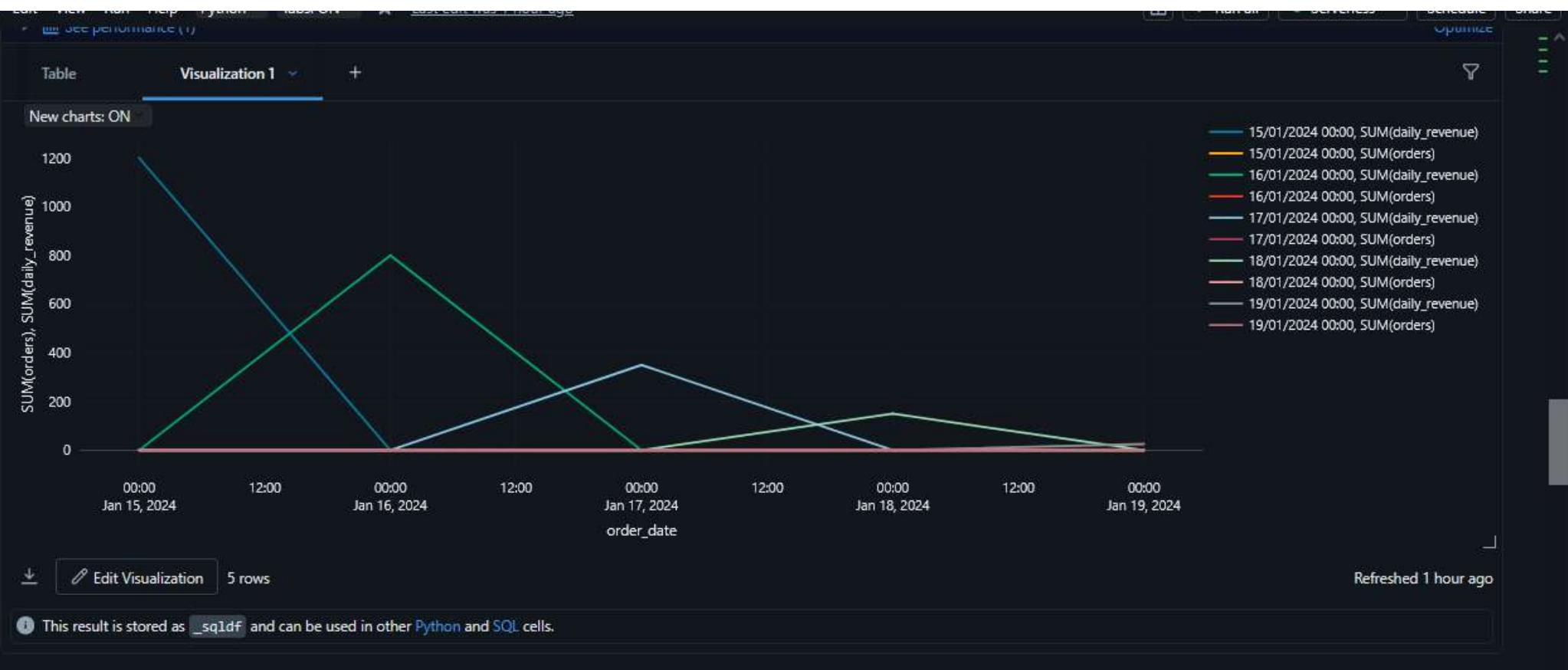
400

200

0



- 15/01/2024 00:00, SUM(daily_revenue)
- 15/01/2024 00:00, SUM(orders)
- 16/01/2024 00:00, SUM(daily_revenue)
- 16/01/2024 00:00, SUM(orders)
- 17/01/2024 00:00, SUM(daily_revenue)
- 17/01/2024 00:00, SUM(orders)
- 18/01/2024 00:00, SUM(daily_revenue)
- 18/01/2024 00:00, SUM(orders)
- 19/01/2024 00:00, SUM(daily_revenue)
- 19/01/2024 00:00, SUM(orders)



Day 9: SQL Analytics & Dashboards Challenge



1. Create SQL warehouse

Set up the foundational compute resource for running SQL queries.

2. Write analytical queries

Develop SQL queries to extract insights from your data.

3. Build dashboard

Visualize the results of your queries in a new dashboard.

4. Add filters & schedule refresh

Enhance interactivity and ensure your data stays current.