INDIAN DATA CLUB

CODE BASICS

databricks

14 DAYS

AI CHALLENGE

DAY 03

Topic:

PySpark Transformations Deep Dive

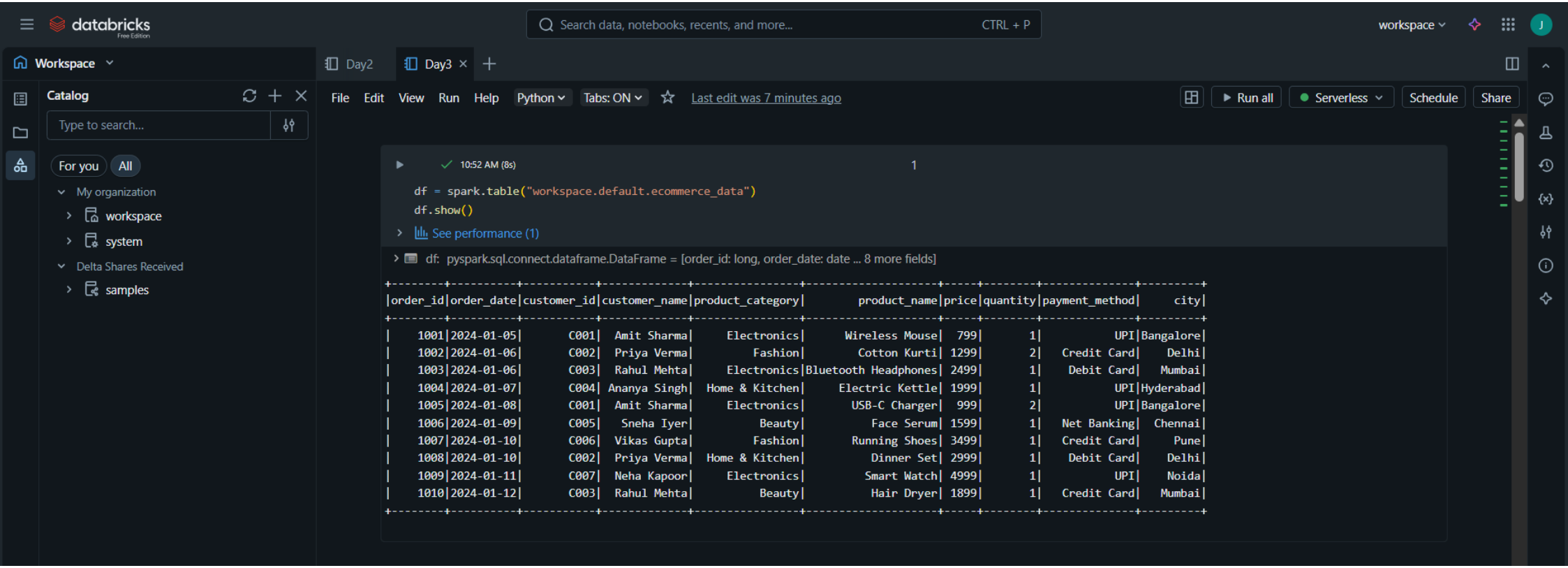
Challenge:

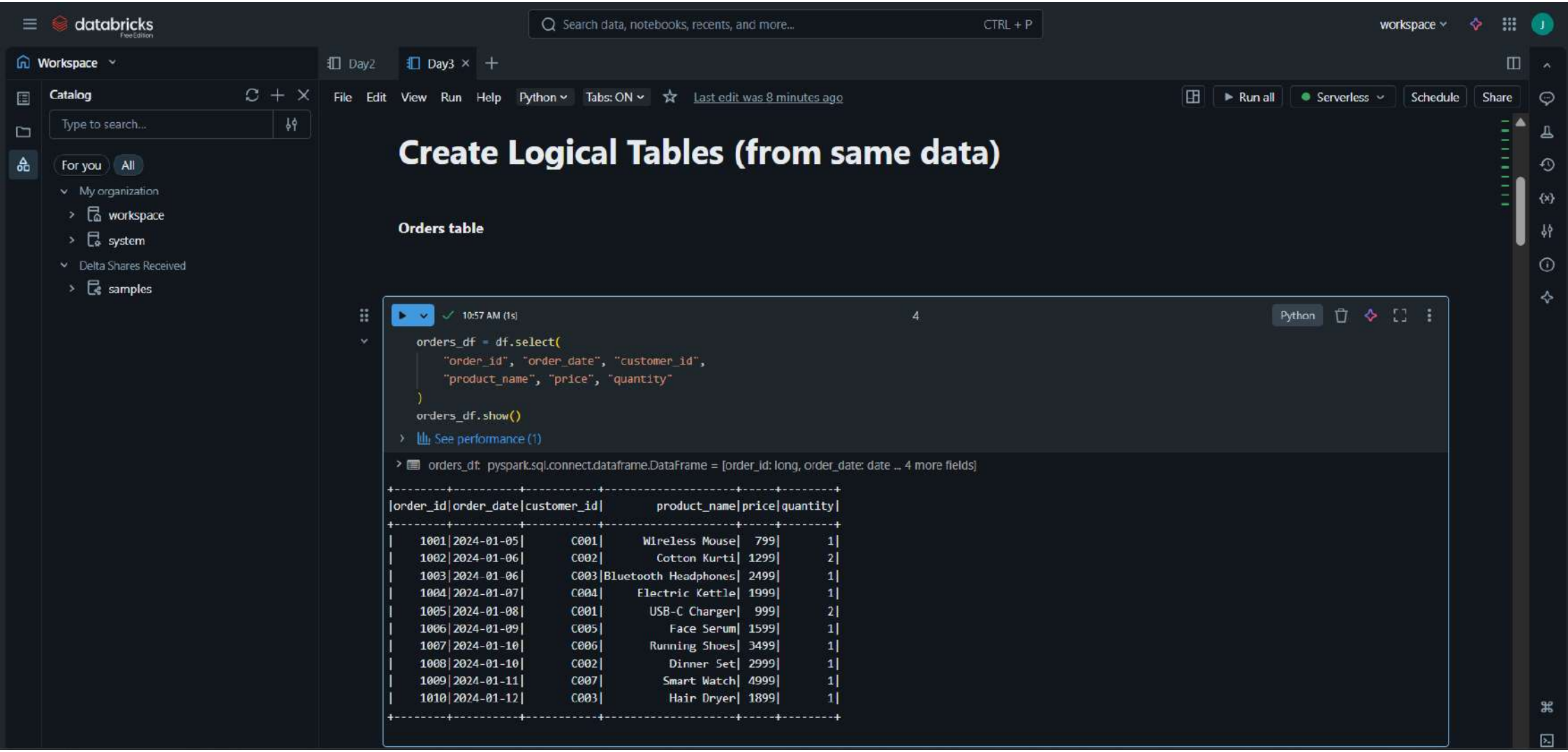
1. Load full e-commerce dataset
2. Perform complex joins
3. Calculate running totals with window functions
4. Create derived features

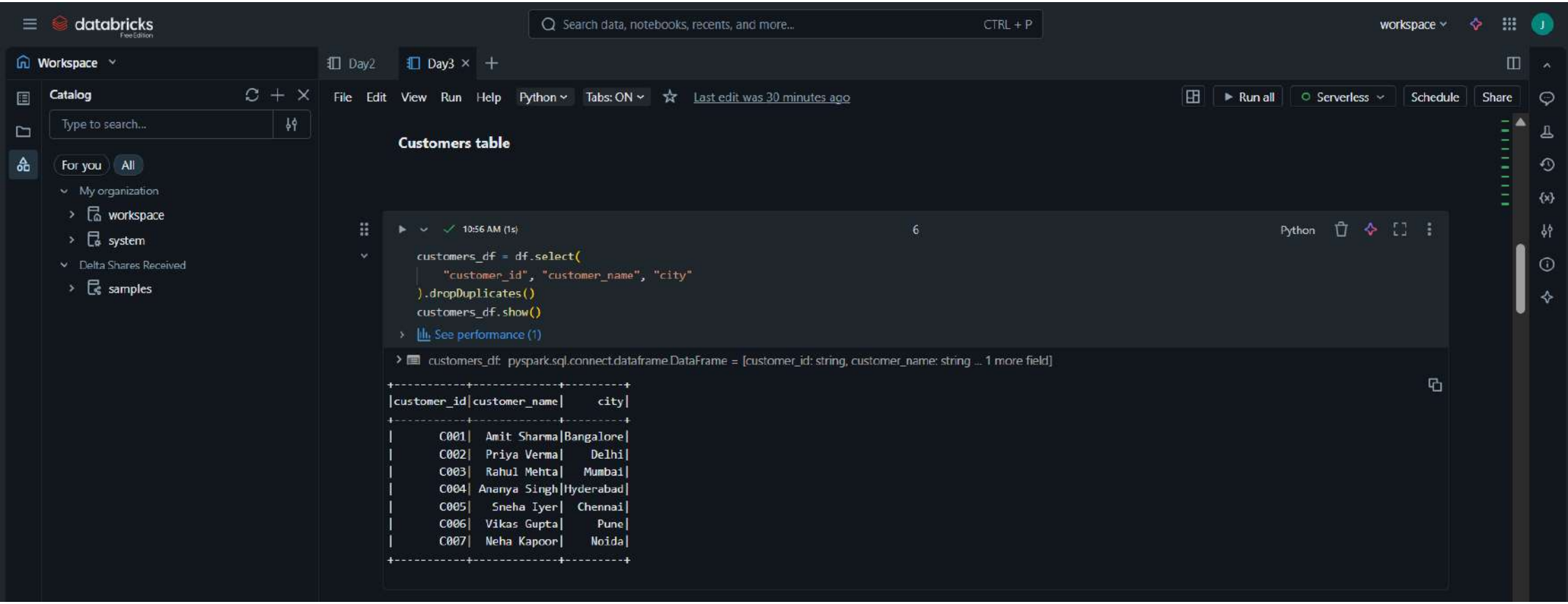
#DatabricksWithIDC

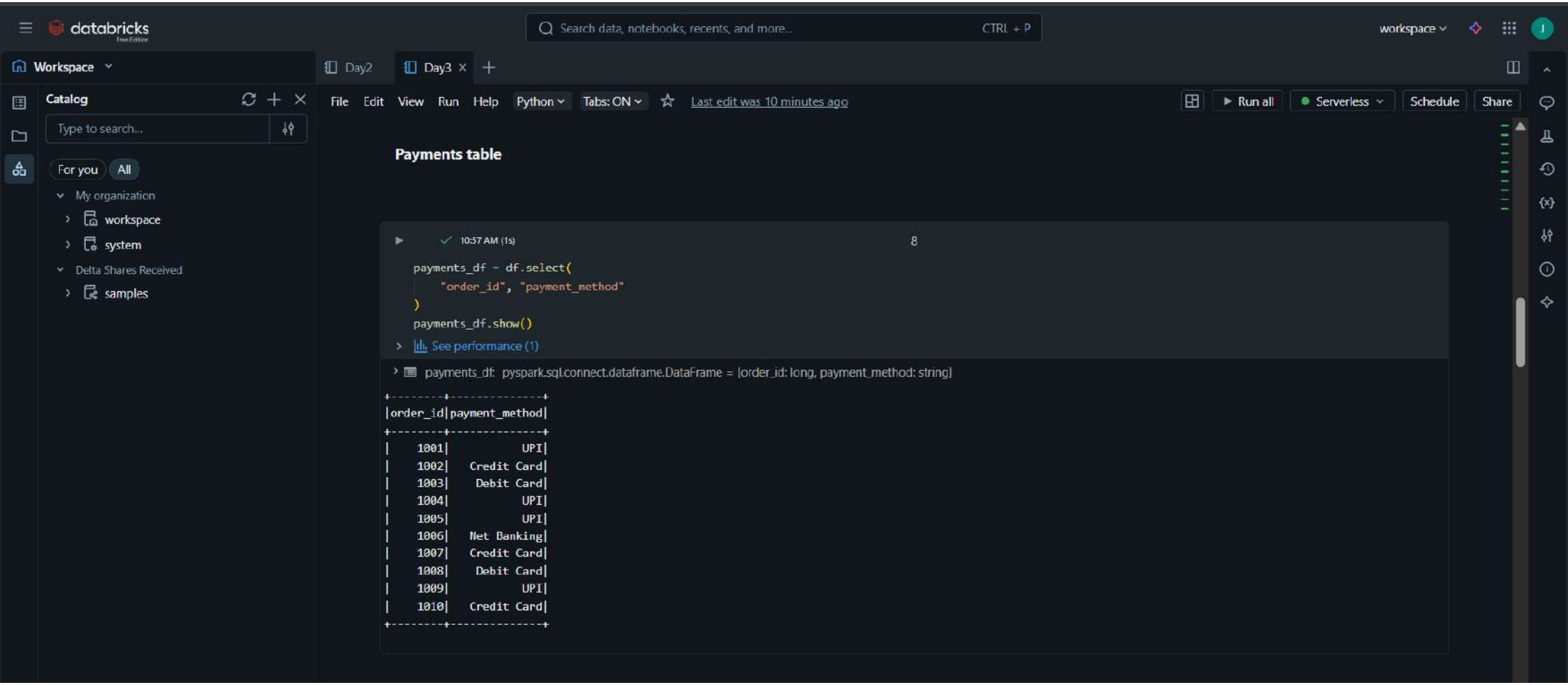
Pandas vs PySpark

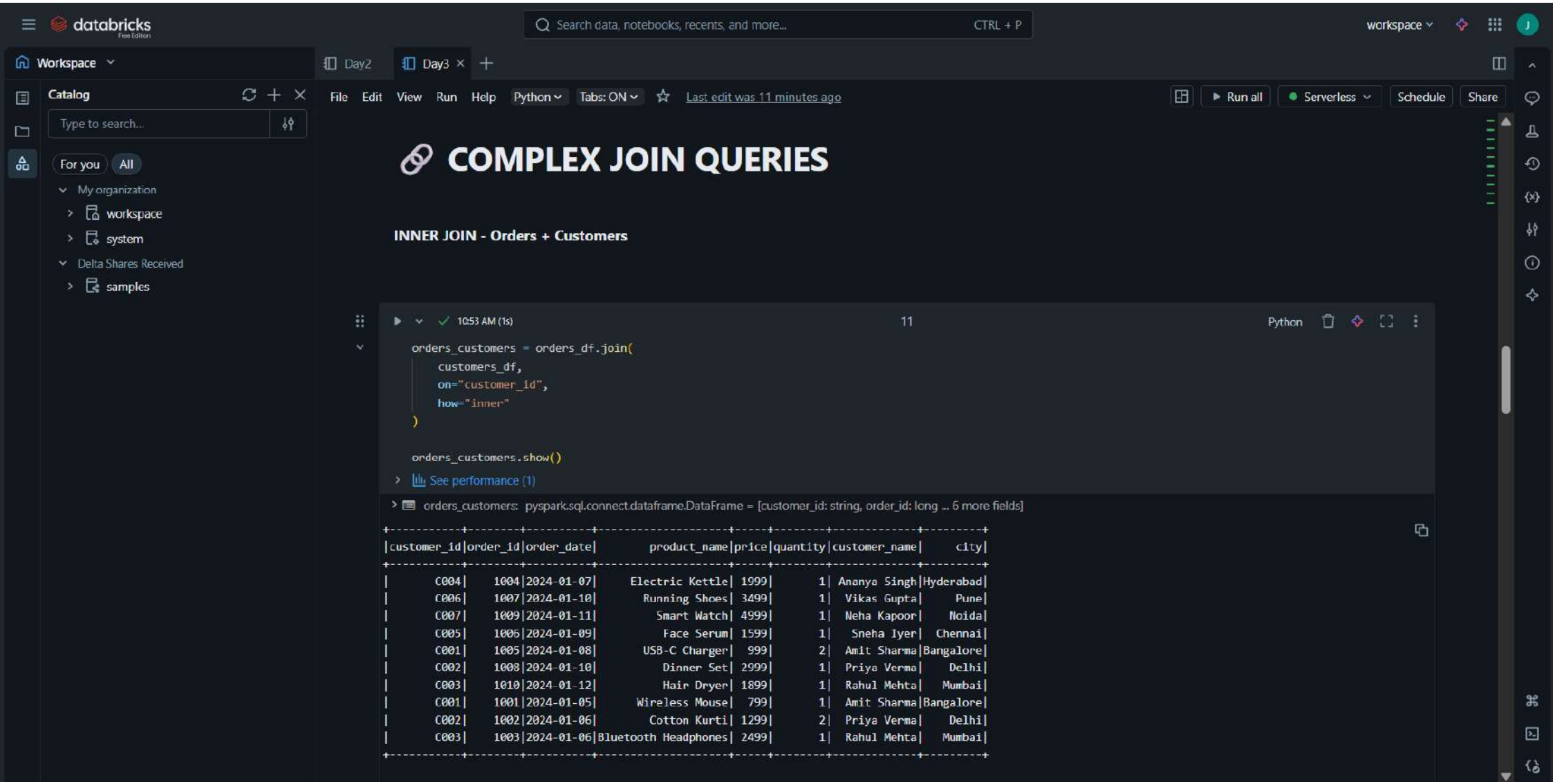
Operation	Pandas	PySpark
View Data	df.head()	df.show()
Data Shape	df.shape	df.count(), len(df.columns)
View Schema	df.info()	df.printSchema()
Select Columns	df[['col1','col2']]	df.select('col1','col2')
Filter Rows	df[df['col] > value]	df.filter(df.col > value)
Multiple Conditions	df.query('col1 >10 & col2 == "A")	df.filter((df.col1 >10) & (df.col2 == 'A'))
Sort Rows	df.sort_values('col')	df.orderBy('col')
Group By + Aggregate	df.groupby('col').sum()	df.groupBy('col').sum()
Count Unique	df['col'].nunique()	df.select('col').distinct().count()
Get Unique Values	df['col'].unique()	df.select('col').distinct()
Check Missing Values	df.isnull().sum()	df.select([F.count(F.when(F.col(c).isNull(), c)).alias(c) for c in df.columns])
Drop Missing Values	df.dropna()	df.dropna()
Fill Missing Values	df.fillna(value)	df.fillna(value)
Join DataFrames	pd.merge(df1, df2, on='key')	df1.join(df2, on='key', how='inner')
Remove Duplicates	df.drop_duplicates()	df.dropDuplicates()
Add New Column	df['new']=df['col1']+df['col2']	df.withColumn('new', df.col1 + df.col2)
Rename Column	df.rename(columns={'old':new'})	df.withColumnRenamed('old', 'new')

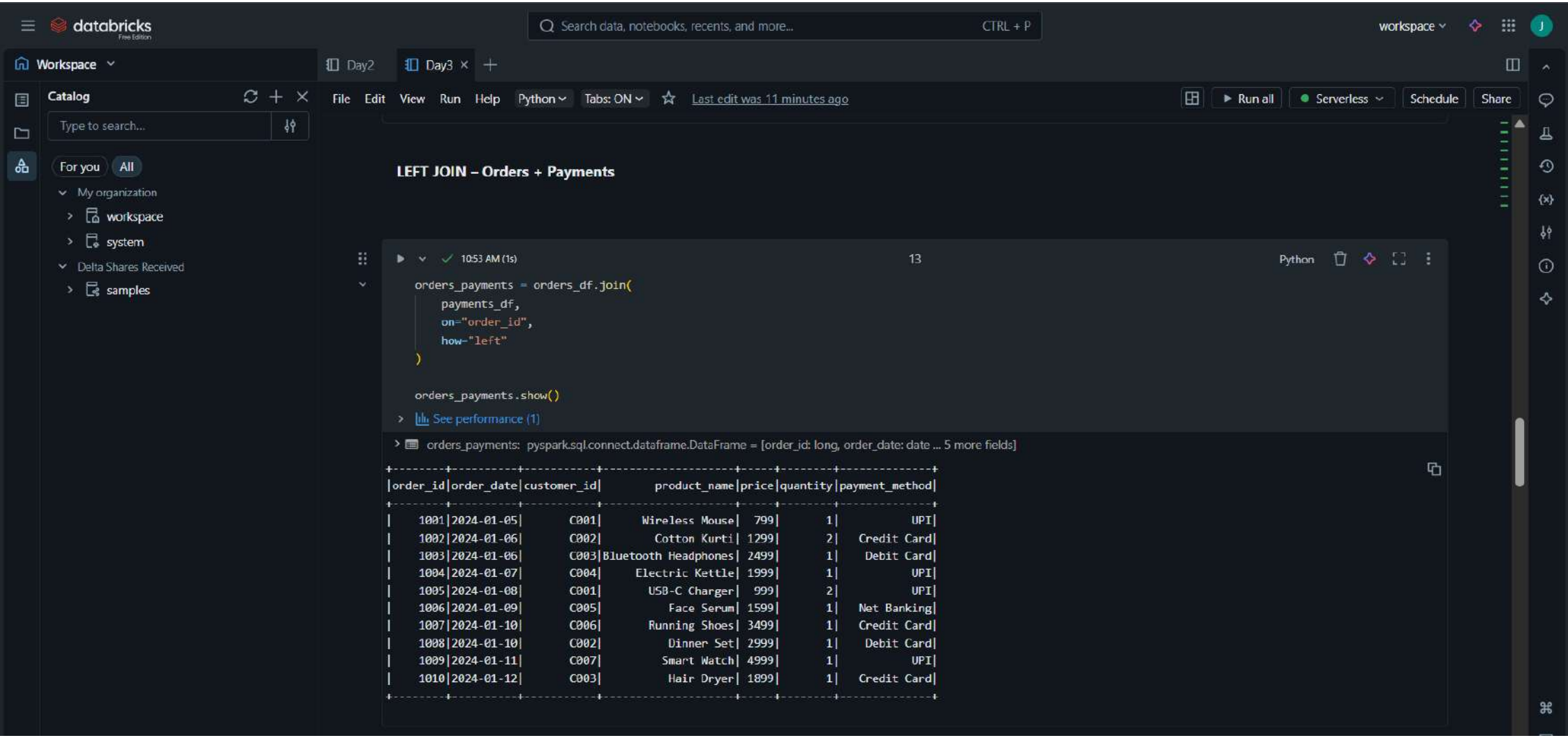

databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File
Edit
View
Run
Help
Python
Tabs: ON
Last edit was 7 minutes ago
Run all
Serverless
Schedule
Share
10:52 AM (8s)
1
df = spark.table("workspace.default.ecommerce_data")
df.show()
See performance (1)
df: pyspark.sql.connect.dataframe.DataFrame = [order_id: long, order_date: date ... 8 more fields]
+--------+----------+-----------+-------------+----------------+--------------------+-----+--------+--------------+---------+
|order_id|order_date|customer_id|customer_name|product_category| product_name|price|quantity|payment_method| city|
+--------+----------+-----------+-------------+----------------+--------------------+-----+--------+--------------+---------+
1001	2024-01-05	C001	Amit Sharma	Electronics	Wireless Mouse	799	1	UPI	Bangalore
1002	2024-01-06	C002	Priya Verma	Fashion	Cotton Kurti	1299	2	Credit Card	Delhi
1003	2024-01-06	C003	Rahul Mehta	Electronics	Bluetooth Headphones	2499	1	Debit Card	Mumbai
1004	2024-01-07	C004	Ananya Singh	Home & Kitchen	Electric Kettle	1999	1	UPI	Hyderabad
1005	2024-01-08	C001	Amit Sharma	Electronics	USB-C Charger	999	2	UPI	Bangalore
1006	2024-01-09	C005	Sneha Iyer	Beauty	Face Serum	1599	1	Net Banking	Chennai
1007	2024-01-10	C006	Vikas Gupta	Fashion	Running Shoes	3499	1	Credit Card	Pune
1008	2024-01-10	C002	Priya Verma	Home & Kitchen	Dinner Set	2999	1	Debit Card	Delhi
1009	2024-01-11	C007	Neha Kapoor	Electronics	Smart Watch	4999	1	UPI	Noida
1010	2024-01-12	C003	Rahul Mehta	Beauty	Hair Dryer	1899	1	Credit Card	Mumbai
+--------+----------+-----------+-------------+----------------+--------------------+-----+--------+--------------+---------+

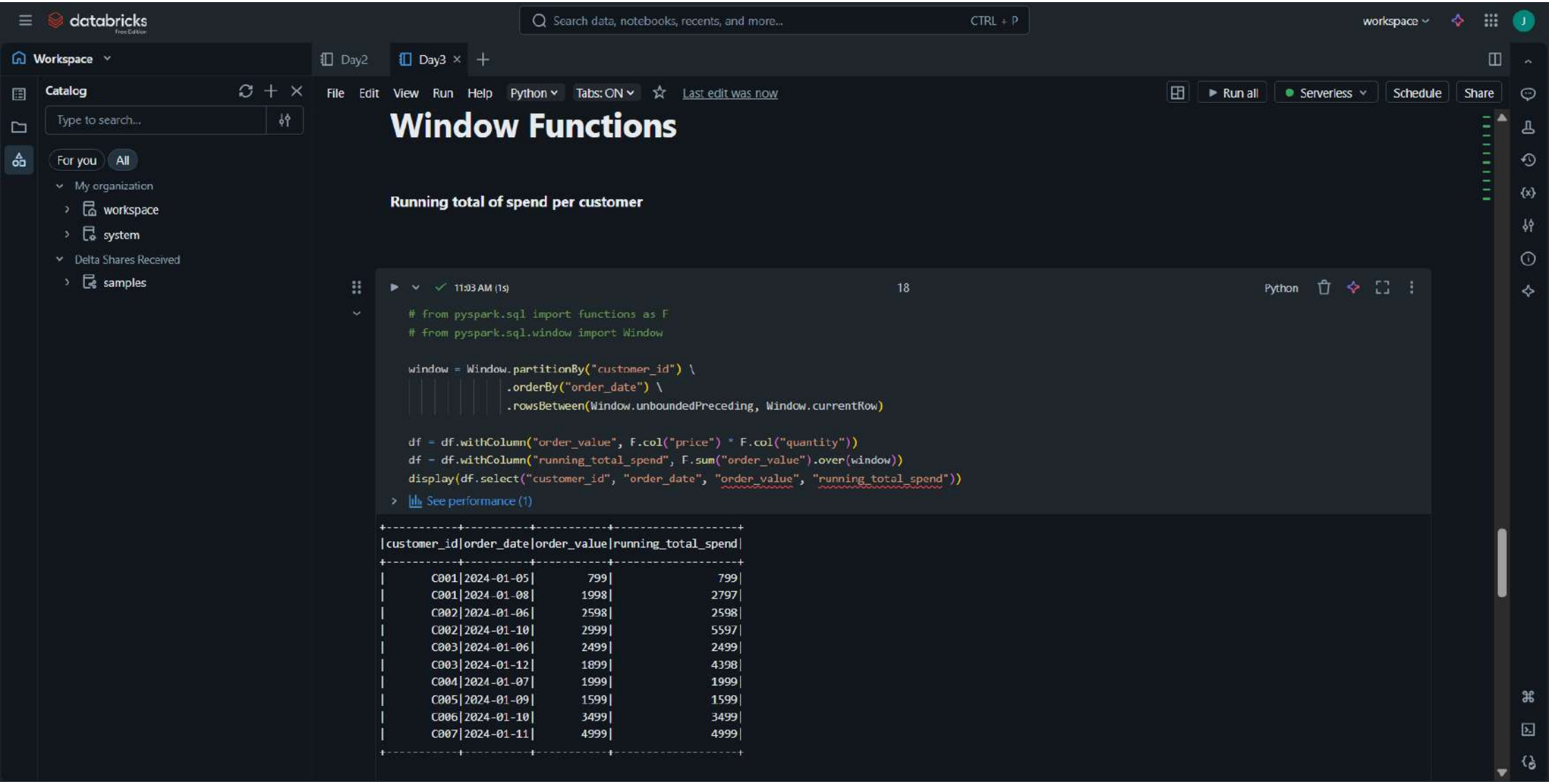

databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File
Edit
View
Run
Help
Python
Tabs: ON
Last edit was 8 minutes ago
Run all
Serverless
Schedule
Share
Create Logical Tables (from same data)
Orders table
10:57 AM (1s)
4
Python
orders_df = df.select(
"order_id", "order_date", "customer_id",
"product_name", "price", "quantity"
)
orders_df.show()
See performance (1)
orders_df: pyspark.sql.connect.dataframe.DataFrame = [order_id: long, order_date: date ... 4 more fields]
+--------+----------+-----------+--------------------+-----+--------+
|order_id|order_date|customer_id| product_name|price|quantity|
+--------+----------+-----------+--------------------+-----+--------+
1001	2024-01-05	C001	Wireless Mouse	799	1
1002	2024-01-06	C002	Cotton Kurti	1299	2
1003	2024-01-06	C003	Bluetooth Headphones	2499	1
1004	2024-01-07	C004	Electric Kettle	1999	1
1005	2024-01-08	C001	USB-C Charger	999	2
1006	2024-01-09	C005	Face Serum	1599	1
1007	2024-01-10	C006	Running Shoes	3499	1
1008	2024-01-10	C002	Dinner Set	2999	1
1009	2024-01-11	C007	Smart Watch	4999	1
1010	2024-01-12	C003	Hair Dryer	1899	1
+--------+----------+-----------+--------------------+-----+--------+

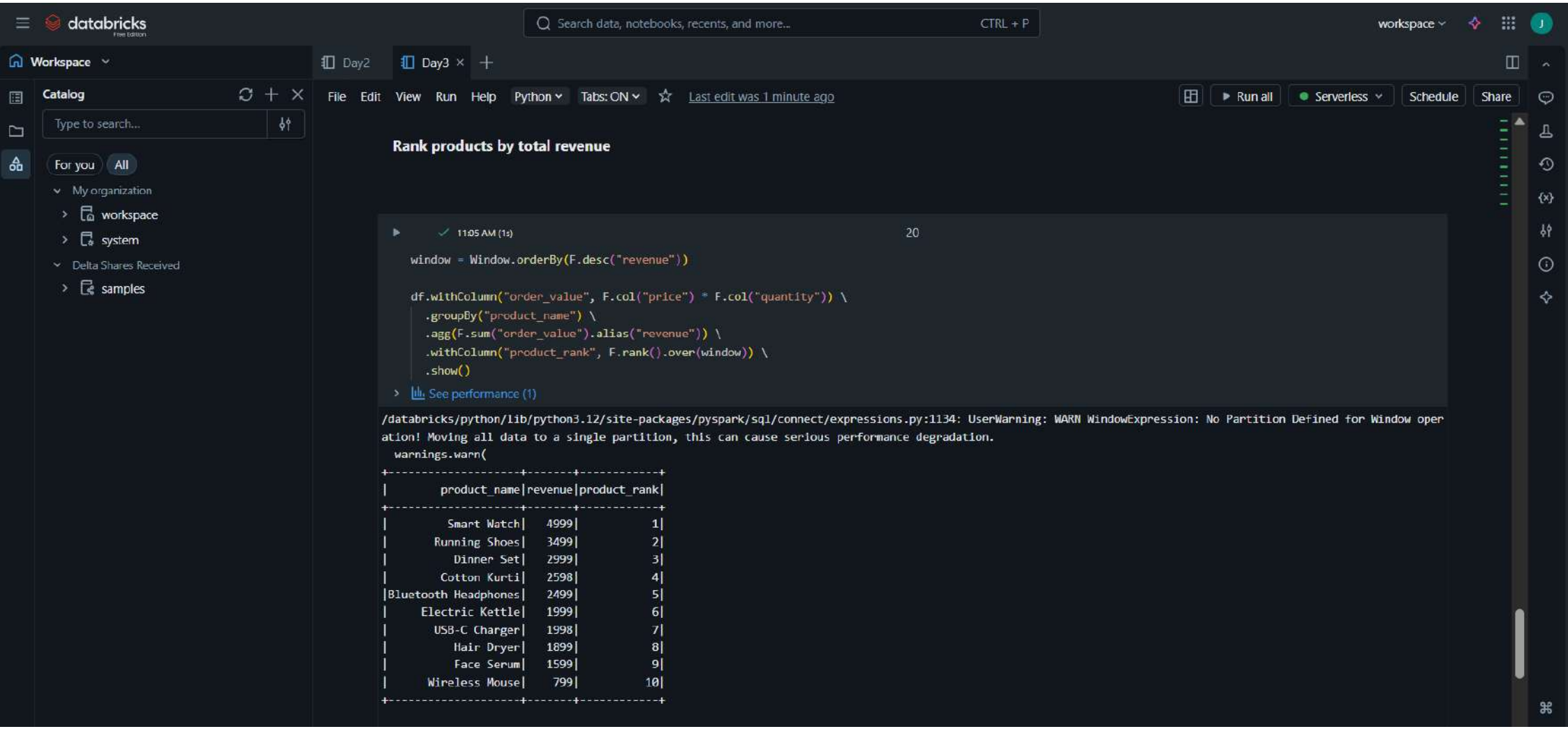

databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File Edit View Run Help Python Tabs: ON
Last edit was 30 minutes ago
Run all
Serverless
Schedule
Share
Customers table
10:56 AM (1s)
6
Python
customers_df = df.select(
"customer_id", "customer_name", "city"
).dropDuplicates()
customers_df.show()
See performance (1)
customers_df: pyspark.sql.connect.dataframe.DataFrame = [customer_id: string, customer_name: string ... 1 more field]
+-----------+-------------+---------+
|customer_id|customer_name| city|
+-----------+-------------+---------+
C001	Amit Sharma	Bangalore
C002	Priya Verma	Delhi
C003	Rahul Mehta	Mumbai
C004	Ananya Singh	Hyderabad
C005	Sneha Iyer	Chennai
C006	Vikas Gupta	Pune
C007	Neha Kapoor	Noida
+-----------+-------------+---------+

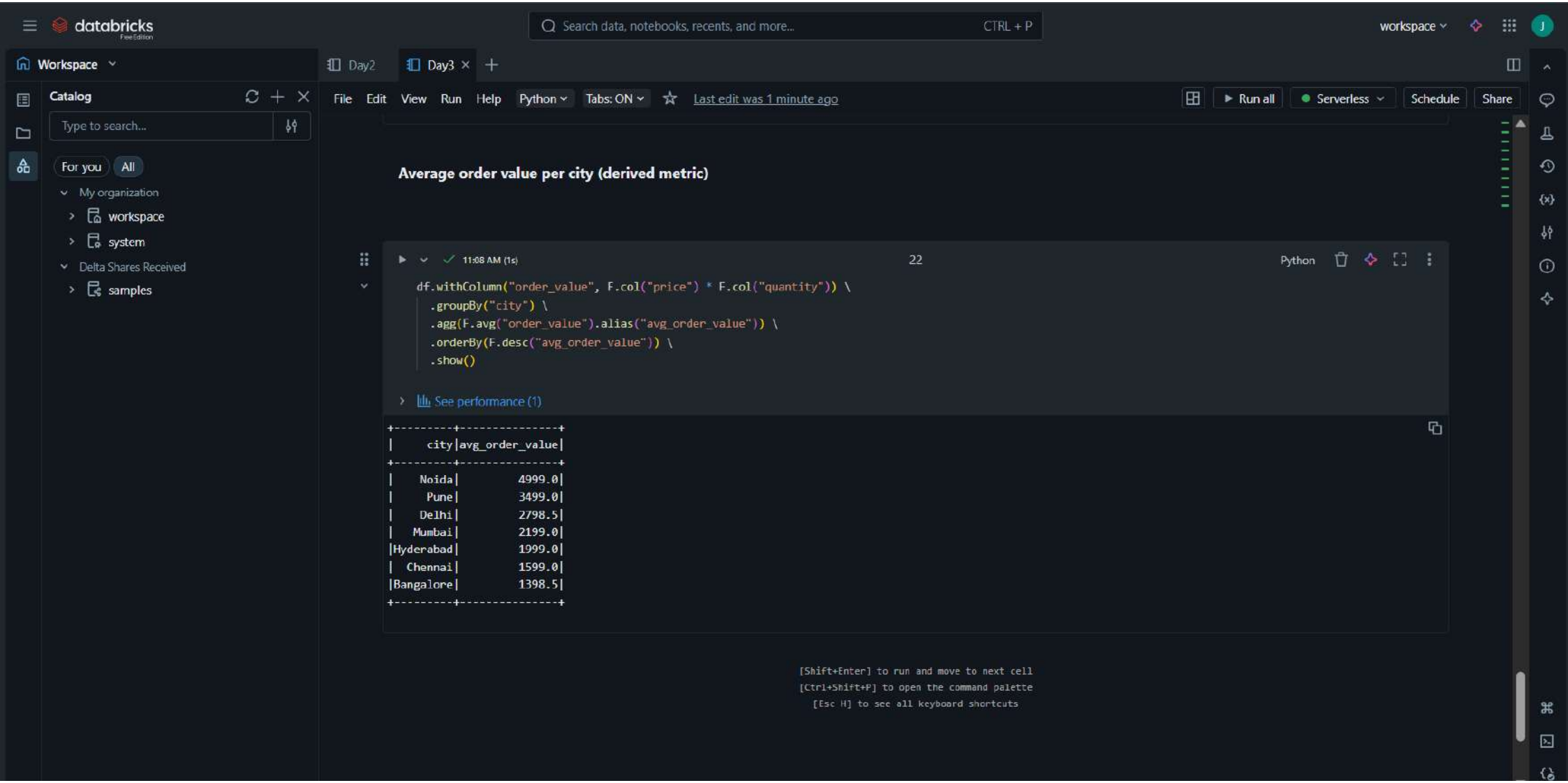

databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File
Edit
View
Run
Help
Python
Tabs: ON
Last edit was 10 minutes ago
Run all
Serverless
Schedule
Share
Payments table
10:57 AM (1s)
payments_df = df.select(
"order_id", "payment_method"
)
payments_df.show()
See performance (1)
payments_df: pyspark.sql.connect.dataframe.DataFrame = [order_id: long, payment_method: string]
+--------+--------------+
|order_id|payment_method|
+--------+--------------+
1001	UPI
1002	Credit Card
1003	Debit Card
1004	UPI
1005	UPI
1006	Net Banking
1007	Credit Card
1008	Debit Card
1009	UPI
1010	Credit Card
+--------+--------------+


databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File Edit View Run Help Python Tabs: ON
Last edit was 11 minutes ago
Run all
Serverless
Schedule
Share
COMPLEX JOIN QUERIES
INNER JOIN - Orders + Customers
10:53 AM (1s)
11
Python
orders_customers = orders_df.join(
customers_df,
on="customer_id",
how="inner"
)
orders_customers.show()
See performance (1)
orders_customers: pyspark.sql.connect.dataframe.DataFrame = [customer_id: string, order_id: long ... 6 more fields]
+-----------+--------+----------+--------------------+-----+--------+-------------+---------+
|customer_id|order_id|order_date| product_name|price|quantity|customer_name| city|
+-----------+--------+----------+--------------------+-----+--------+-------------+---------+
C004	1004	2024-01-07	Electric Kettle	1999	1	Ananya Singh	Hyderabad
C006	1007	2024-01-10	Running Shoes	3499	1	Vikas Gupta	Pune
C007	1009	2024-01-11	Smart Watch	4999	1	Neha Kapoor	Noida
C005	1006	2024-01-09	Face Serum	1599	1	Sneha Iyer	Chennai
C001	1005	2024-01-08	USB-C Charger	999	2	Amit Sharma	Bangalore
C002	1008	2024-01-10	Dinner Set	2999	1	Priya Verma	Delhi
C003	1010	2024-01-12	Hair Dryer	1899	1	Rahul Mehta	Mumbai
C001	1001	2024-01-05	Wireless Mouse	799	1	Amit Sharma	Bangalore
C002	1002	2024-01-06	Cotton Kurti	1299	2	Priya Verma	Delhi
C003	1003	2024-01-06	Bluetooth Headphones	2499	1	Rahul Mehta	Mumbai
+-----------+--------+----------+--------------------+-----+--------+-------------+---------+


databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File
Edit
View
Run
Help
Python
Tabs: ON
Last edit was 11 minutes ago
Run all
Serverless
Schedule
Share
LEFT JOIN – Orders + Payments
10:53 AM (1s)
13
Python
orders_payments = orders_df.join(
payments_df,
on="order_id",
how="left"
)
orders_payments.show()
See performance (1)
orders_payments: pyspark.sql.connect.dataframe.DataFrame = [order_id: long, order_date: date ... 5 more fields]
+--------+----------+-----------+--------------------+-----+--------+--------------+
|order_id|order_date|customer_id| product_name|price|quantity|payment_method|
+--------+----------+-----------+--------------------+-----+--------+--------------+
1001	2024-01-05	C001	Wireless Mouse	799	1	UPI
1002	2024-01-06	C002	Cotton Kurti	1299	2	Credit Card
1003	2024-01-06	C003	Bluetooth Headphones	2499	1	Debit Card
1004	2024-01-07	C004	Electric Kettle	1999	1	UPI
1005	2024-01-08	C001	USB-C Charger	999	2	UPI
1006	2024-01-09	C005	Face Serum	1599	1	Net Banking
1007	2024-01-10	C006	Running Shoes	3499	1	Credit Card
1008	2024-01-10	C002	Dinner Set	2999	1	Debit Card
1009	2024-01-11	C007	Smart Watch	4999	1	UPI
1010	2024-01-12	C003	Hair Dryer	1899	1	Credit Card
+--------+----------+-----------+--------------------+-----+--------+--------------+


databricks
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File Edit View Run Help Python Tabs: ON
Last edit was now
Run all
Serverless
Schedule
Share
Window Functions
Running total of spend per customer
11:03 AM (1s)
18
Python
# from pyspark.sql import functions as F
# from pyspark.sql.window import Window
window = Window.partitionBy("customer_id") \
.orderBy("order_date") \
.rowsBetween(Window.unboundedPreceding, Window.currentRow)
df = df.withColumn("order_value", F.col("price") * F.col("quantity"))
df = df.withColumn("running_total_spend", F.sum("order_value").over(window))
display(df.select("customer_id", "order_date", "order_value", "running_total_spend"))
See performance (1)
+-----------+----------+-----------+-------------------+
|customer_id|order_date|order_value|running_total_spend|
+-----------+----------+-----------+-------------------+
C001	2024-01-05	799	799
C001	2024-01-08	1998	2797
C002	2024-01-06	2598	2598
C002	2024-01-10	2999	5597
C003	2024-01-06	2499	2499
C003	2024-01-12	1899	4398
C004	2024-01-07	1999	1999
C005	2024-01-09	1599	1599
C006	2024-01-10	3499	3499
C007	2024-01-11	4999	4999
+-----------+----------+-----------+-------------------+


databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File
Edit
View
Run
Help
Python
Tabs: ON
Last edit was 1 minute ago
Run all
Serverless
Schedule
Share
Rank products by total revenue
11:05 AM (1s)
20
window = Window.orderBy(F.desc("revenue"))
df.withColumn("order_value", F.col("price") * F.col("quantity")) \
.groupBy("product_name") \
.agg(F.sum("order_value").alias("revenue")) \
.withColumn("product_rank", F.rank().over(window)) \
.show()
See performance (1)
/databricks/python/lib/python3.12/site-packages/pyspark/sql/connect/expressions.py:1134: UserWarning: WARN WindowExpression: No Partition Defined for Window operation! Moving all data to a single partition, this can cause serious performance degradation.
warnings.warn(
+--------------------+-------+------------+
| product_name|revenue|product_rank|
+--------------------+-------+------------+
Smart Watch	4999	1
Running Shoes	3499	2
Dinner Set	2999	3
Cotton Kurti	2598	4
Bluetooth Headphones	2499	5
Electric Kettle	1999	6
USB-C Charger	1998	7
Hair Dryer	1899	8
Face Serum	1599	9
Wireless Mouse	799	10
+--------------------+-------+------------+


databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
Workspace
Day2
Day3
Catalog
Type to search...
For you
All
My organization
workspace
system
Delta Shares Received
samples
File
Edit
View
Run
Help
Python
Tabs: ON
Last edit was 1 minute ago
Run all
Serverless
Schedule
Share
Average order value per city (derived metric)
11:08 AM (1s)
22
Python
df.withColumn("order_value", F.col("price") * F.col("quantity")) \
.groupBy("city") \
.agg(F.avg("order_value").alias("avg_order_value")) \
.orderBy(F.desc("avg_order_value")) \
.show()
See performance (1)
+---------+---------------+
| city|avg_order_value|
+---------+---------------+
Noida	4999.0
Pune	3499.0
Delhi	2798.5
Mumbai	2199.0
Hyderabad	1999.0
Chennai	1599.0
Bangalore	1398.5
+---------+---------------+
[Shift+Enter] to run and move to next cell
[Ctrl+Shift+P] to open the command palette
[Esc H] to see all keyboard shortcuts