



CALL CENTER PERFORMANCE ANALYZATION USING DATABRICKS



Trillia Kumaresan

Data Analyst

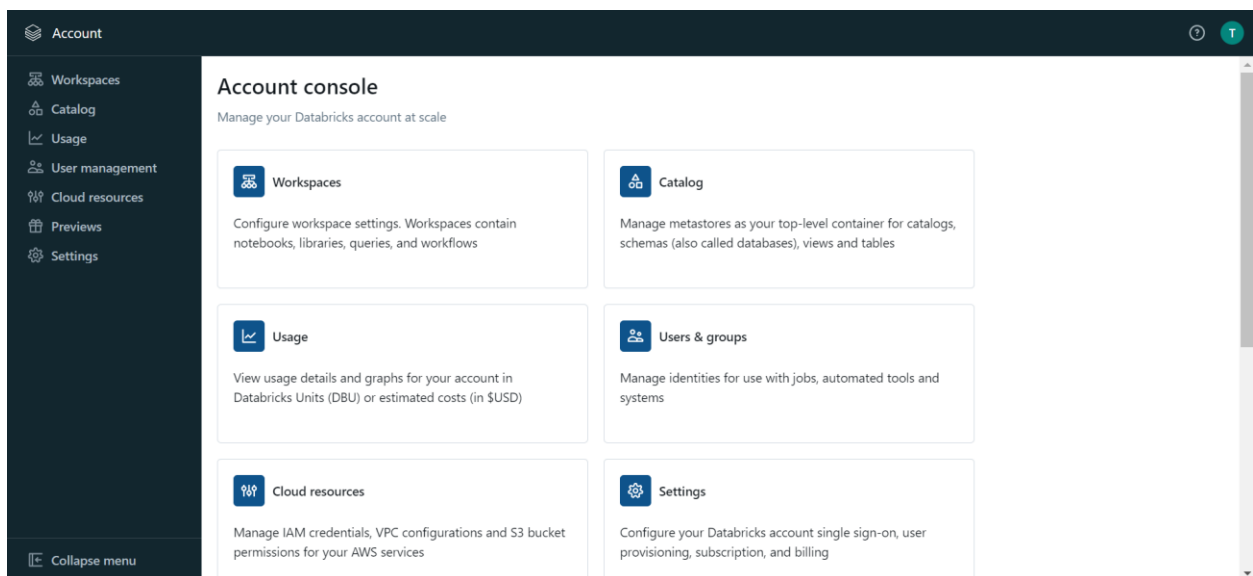
02/09/2024

Exploring Data Analytics on the Databricks Platform

Databricks is a cloud-based data analytics platform that provides a unified environment for data engineering, data science, and machine learning. It was founded by the creators of Apache Spark and is designed to handle large-scale data processing and analytics, making it a popular choice for organizations looking to manage and analyze big data.

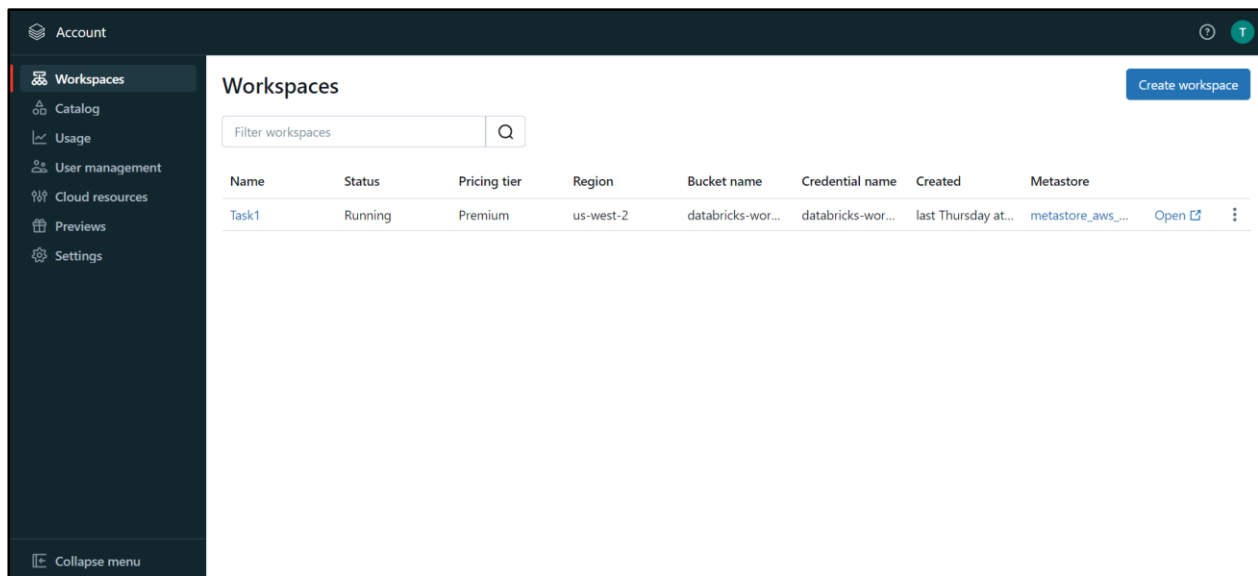
Key Features of Databricks:

1. Unified Data Analytics Platform
2. Apache Spark Integration
3. Lakehouse Architecture
4. Databricks SQL
5. Collaborative Notebooks
6. Machine Learning and AI
7. Data Security and Governance
8. Scalability and Performance



Creating Databricks Workspace

Created a Databricks workspace called **Task1** which involves setting up a cloud environment where workloads can run on Amazon Web Services (AWS).

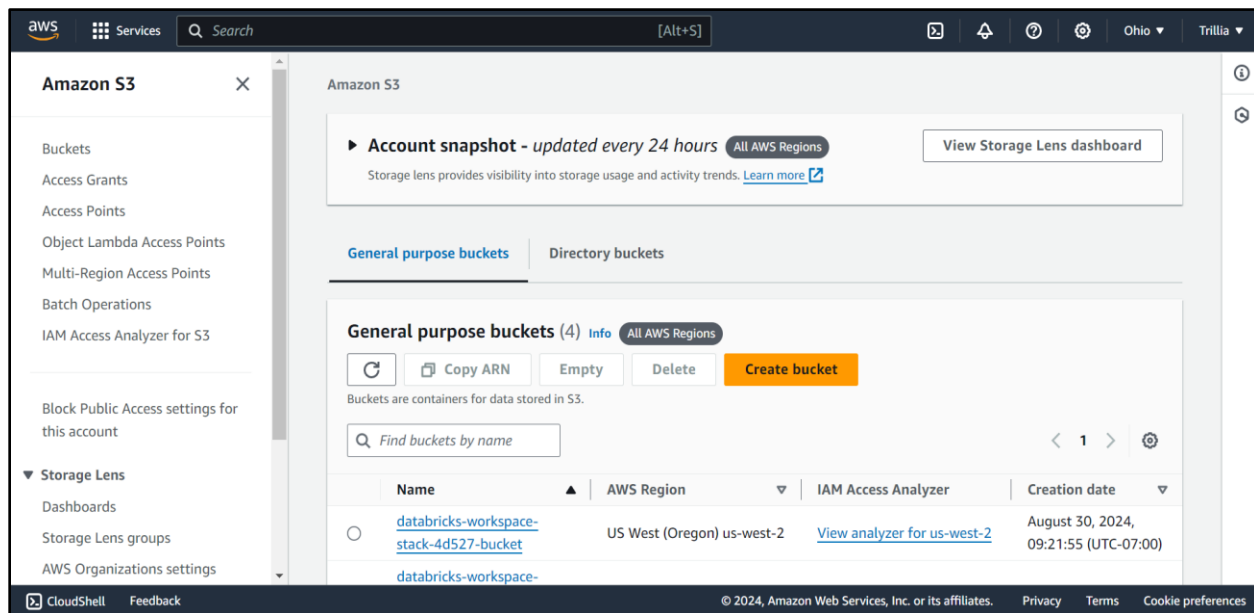


Integration with AWS Services

Databricks integrates seamlessly with Amazon Web Services (AWS) to provide a powerful, scalable platform for big data processing, analytics, and machine learning. By combining Databricks with AWS, we can leverage the strengths of both platforms to manage and analyze large datasets efficiently.

Databricks integrates with a wide range of AWS services, including:

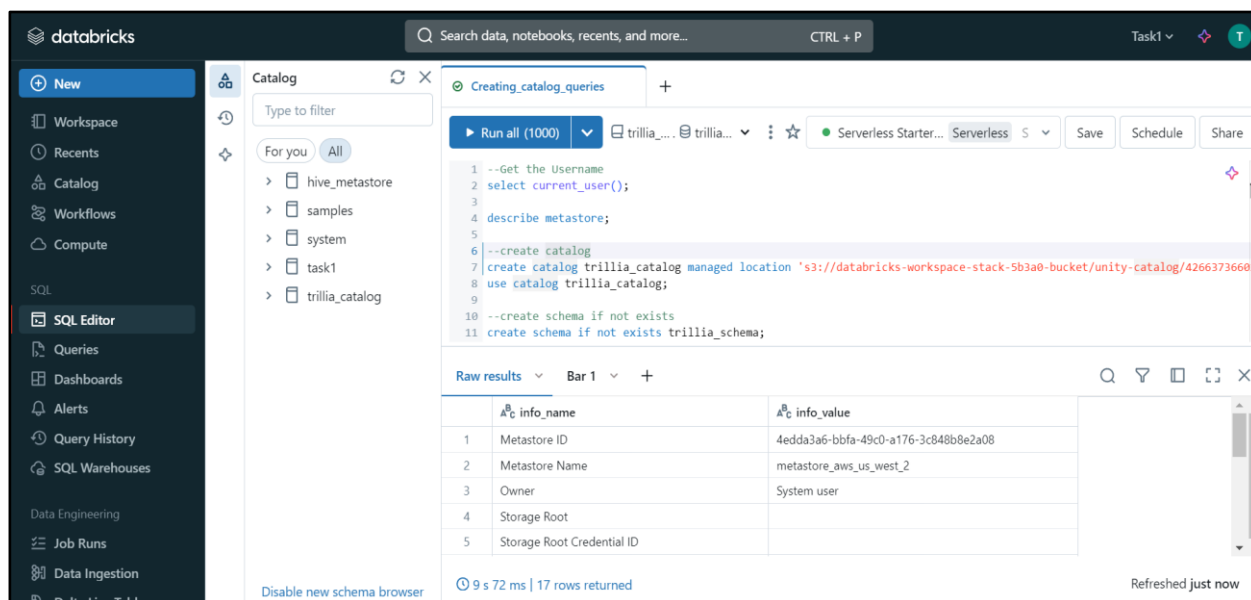
- **Amazon S3:** For scalable, secure storage of data in data lakes.
- **AWS Glue:** For data cataloging and ETL (Extract, Transform, Load) processes.
- **Amazon Redshift:** For data warehousing and SQL analytics.
- **Amazon RDS:** For managing relational databases.
- **AWS Lambda:** For serverless computing and event-driven processing.
- **AWS Identity and Access Management (IAM):** For secure access control and authentication.



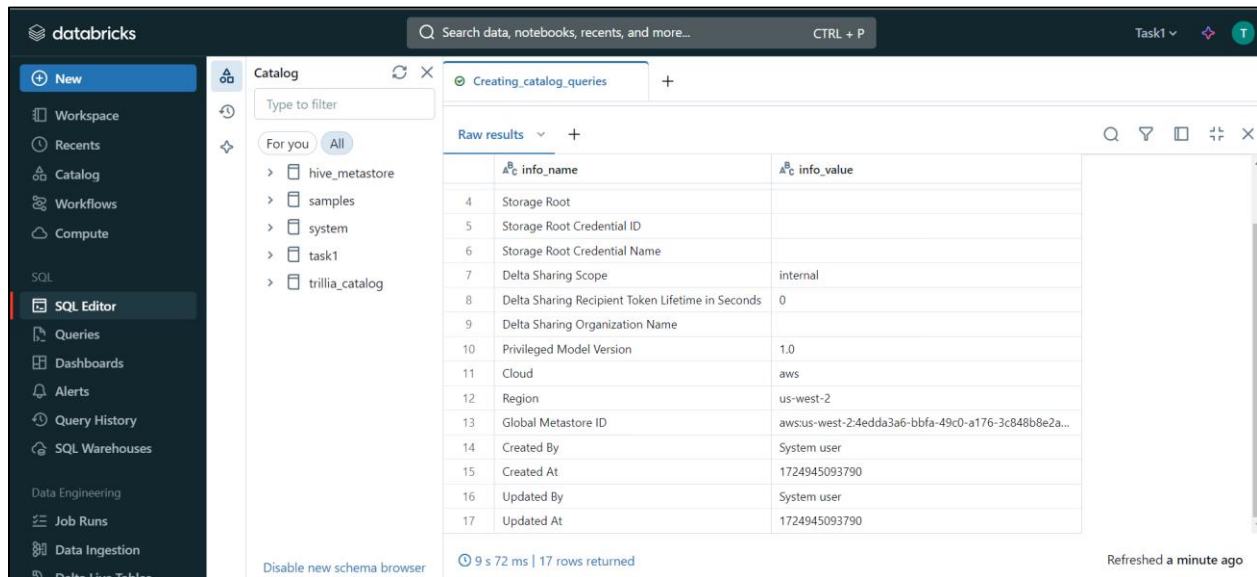
Getting Started with Databricks SQL

It involves the basics of setting up a SQL working environment. This includes setting up a catalog and schema, which is crucial for organizing data within the Databricks Lakehouse platform.

Step 1: Getting the username of the console that is integrated with Amazon Web Services(AWS)



Step 2: By describing a metastore in Databricks can interact with the metadata stored in the metastore.



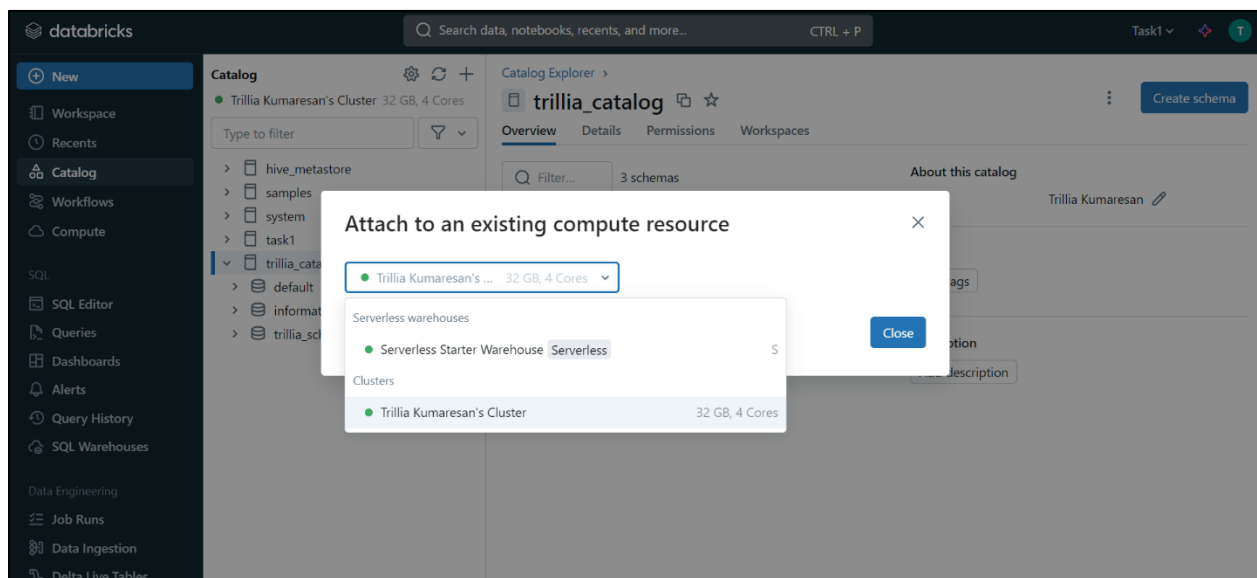
The screenshot shows the Databricks SQL Editor interface. On the left, the 'Catalog' sidebar is open, displaying a tree view with 'hive_metastore', 'samples', 'system', 'task1', and 'trillia_catalog'. The main panel shows a query titled 'Creating_catalog_queries' with 'Raw results' displayed. The results table has two columns: 'info_name' and 'info_value'. The table contains 17 rows of metadata information.

	info_name	info_value
4	Storage Root	
5	Storage Root Credential ID	
6	Storage Root Credential Name	
7	Delta Sharing Scope	internal
8	Delta Sharing Recipient Token Lifetime in Seconds	0
9	Delta Sharing Organization Name	
10	Privileged Model Version	1.0
11	Cloud	aws
12	Region	us-west-2
13	Global Metastore ID	awsus-west-2:4edda3a6-bbfa-49c0-a176-3c848b8e2a...
14	Created By	System user
15	Created At	1724945093790
16	Updated By	System user
17	Updated At	1724945093790

9 s 72 ms | 17 rows returned

Step 3: Creating Databricks Catalog and Schema

Created and used a catalog & schema in a Databricks environment, specifically leveraging AWS S3 as the storage location.



The screenshot shows the Databricks Catalog Explorer interface. The 'trillia_catalog' is selected, and the 'Overview' tab is active. A dialog box titled 'Attach to an existing compute resource' is open, showing a list of available compute resources. The dialog has a 'Close' button.

Attach to an existing compute resource

- Trillia Kumaresan's ... 32 GB, 4 Cores

Serverless warehouses

- Serverless Starter Warehouse Serverless

Clusters

- Trillia Kumaresan's Cluster 32 GB, 4 Cores

Step 4: Creating View

The `satisfied_customers_view` will contain all records from `call_center_data` where the `Customer_Sat_Score` is a valid number greater than 7. Any records where `Customer_Sat_Score` is non-numeric, null, or less than or equal to 7 will be excluded from this view.

The screenshot shows the Databricks SQL Editor interface. On the left is a sidebar with navigation options like Workspace, Recents, Catalog, Workflows, Compute, SQL, SQL Editor, Queries, Dashboards, Alerts, Query History, SQL Warehouses, Data Engineering, Job Runs, Data Ingestion, and Delta Live Tables. The main area displays a SQL query titled 'Creating_catalog_queries' with the following code:

```
--create view
create view satisfied_customers_view as
select *
from call_center_data
where try_cast(customer_sat_score as BIGINT) > 7;
select * from satisfied_customers_view;
--TIME TRAVEL
--History of the table
```

Below the query, the 'Raw results' section shows a table with 104 rows. The first four rows are visible:

	Customer_ID	Customer_Name	Sentiment	Customer_Sat_Score	Call_Timesta...	Date_o
1	AZI-95054097-e-185542-PT	Phillipe Bowring	Neutral	8	10/01/2020	
2	WRN-02286567-g-819640-IF	Inessa Trippitt	Very Positive	9	10/01/2020	
3	RTW-93566842-a-737480-ex	Luca Castel	Positive	9	10/01/2020	
4	NEC-52763410-8-812910-pl	Dare Ropcke	Positive	8	10/01/2020	

At the bottom, it indicates '4 s 129 ms | 104 rows returned' and 'Refreshed 23 minutes ago'.

Step 5: Time Travel

Time Travel concept in Databricks allows to query and restore previous versions of a Delta Lake table. Delta Lake is a storage layer that brings ACID transactions to Apache Spark and big data workloads. One of its key features is the ability to track and maintain the history of changes made to the data.

```
--TIME TRAVEL
--History of the table
describe history call_center_data;

---update a row in a table
select * from call_center_data where Customer_ID = 'FAK-09576309-s-864637-wb';
update call_center_data set Customer_ID = 'FAK-09576309-s-864637-wb' where Customer_ID = 'FAK-09576309';

--version selection
select Customer_ID from call_center_data version as of 2 where Customer_Name='Lexy Cradey';
```

databricks Search data, notebooks, recents, and more... CTRL + P Task1

New Workspace Recents Catalog Workflows Compute

SQL SQL Editor Queries Dashboards Alerts Query History SQL Warehouses Data Engineering Job Runs Data Ingestion Delta Live Tables

Catalog Type to filter

For you All

- hive_metastore
 - samples
 - system
 - task1
 - trillia_catalog
 - default
 - information_schema
 - trillia_schema
 - call_center_data
 - satisfied_cust...
 - satisfied_cust...

Disable new schema browser

Creating_catalog_queries

Raw results

	version	timestamp	userid	username	operation	operationParameters
1	3	2024-08-31T...	22305204...	trillia.kumaresa...	UPDATE	("predicate":["(Customer_ID#2570 =
2	2	2024-08-31T...	22305204...	trillia.kumaresa...	UPDATE	("predicate":["(Customer_ID#1399 =
3	1	2024-08-30T...	22305204...	trillia.kumaresa...	SET TBLPROPERTIES	("properties":["comment":"The 'exc
4	0	2024-08-30T...	22305204...	trillia.kumaresa...	CREATE TABLE ...	("partitionBy":[""],"description":"Creat

15 s 75 ms | 4 rows returned Refreshed 2 minutes ago

databricks Search data, notebooks, recents, and more... CTRL + P Task1

New Workspace Recents Catalog Workflows Compute

SQL SQL Editor Queries Dashboards Alerts Query History SQL Warehouses Data Engineering Job Runs Data Ingestion Delta Live Tables

Catalog Type to filter

For you All

- hive_metastore
 - samples
 - system
 - task1
 - trillia_catalog
 - default
 - information_schema
 - trillia_schema
 - call_center_data
 - satisfied_cust...
 - satisfied_cust...

Disable new schema browser

Creating_catalog_queries

Run selected (1000) trillia... trillia... Serverless Starter... Serverless S Save Schedule

```

25 --TIME TRAVEL
26 --History of the table
27 describe history call_center_data;
28
29 ---update a row in a table
30 select * from call_center_data where Customer_ID = 'FAK-09576309-s-864637-wb';
31 update call_center_data set Customer_ID = 'FAK-09576309-s-864637-wb' where Customer_ID = 'FAK-09576309';
32
33 --version selection
34 select Customer_ID from call_center_data version as of 2 where Customer_Name='Lexy Cradey';
35

```

Raw results

	Customer_ID
1	FAK-09576309

Step 6: Restore Table

Creating_catalog_queries

Run selected (1000) trillia... trillia... Serverless Starter... Serverless S Save Schedule

```

28
29 ---update a row in a table
30 select * from call_center_data where Customer_ID = 'FAK-09576309-s-864637-wb';
31 update call_center_data set Customer_ID = 'FAK-09576309-s-864637-wb' where Customer_ID = 'FAK-09576309';
32
33 --version selection
34 select Customer_ID from call_center_data version as of 2 where Customer_Name='Lexy Cradey';
35
36 --restore
37 restore table call_center_data to version as of 2;
38

```

This command is used to revert a Delta Lake table to a previous state or version. It's part of Delta Lake's Time Travel feature, which allows you to access and restore historical versions of a table.

Data Analysis on Databricks

To perform data analysis on a call center dataset using Databricks, you would typically go through a series of steps, from data exploration and cleaning to more advanced analytics and visualization. Below is a detailed outline of how you might approach this, assuming the dataset includes columns like **Customer_ID**, **Customer_Name**, **Sentiment**, **Customer_Sat_Score**, **Call_Timestamp**, **Reason**, **City**, **State**, **Channel**, **Response_Time**, **Call_Duration_in_Minutes**, etc.

Questions:

1. What is the average customer satisfaction score across different call centers?

An average customer satisfaction score for each call center within the call_center_data table.

The screenshot shows the Databricks SQL Editor interface. On the left is a sidebar with navigation options like Workspace, Recents, Catalog, Workflows, Compute, SQL Editor, Queries, Dashboards, Alerts, Query History, and SQL Warehouses. The main area displays a SQL query and its results. The query is as follows:

```
1 --1. What is the average customer satisfaction score across different call centers?
2 SELECT Call_Center, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
3 FROM call_center_data
4 GROUP BY Call_Center;
5
6 --2. How does customer sentiment correlate with customer satisfaction scores?
7 SELECT Sentiment, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
8 FROM call_center_data
9 GROUP BY Sentiment;
10
11 --3. Are there any patterns in customer satisfaction scores based on the Channel used?
```

Below the query, the 'Raw results' section shows a table with 4 rows and 2 columns: 'Call_Center' and '1.2 Average_Satisfaction'.

	Call_Center	1.2 Average_Satisfaction
1	Denver	6.444444444444445
2	Chicago	5.522388059701493
3	Baltimore	5.887931034482759
4	Los Angeles	5.775510204081633

At the bottom of the results section, it indicates '14 s 137 ms | 4 rows returned' and 'Refreshed 6 minutes ago'.

2. How does customer sentiment correlate with customer satisfaction scores?

This selects the Sentiment column from the call_center_data table. The Sentiment column likely contains categorical values that represent the sentiment expressed by the customer, such as "Positive," "Negative," or "Neutral."

The screenshot shows the Databricks SQL Editor interface. The left sidebar contains navigation options like Workspace, Recents, Catalog, Workflows, Compute, SQL, SQL Editor, Queries, Dashboards, Alerts, Query History, SQL Warehouses, Data Engineering, Job Runs, Data Ingestion, and Delta Live Tables. The main panel displays a SQL query with three questions and their corresponding SQL statements. The query is executed, and the results are shown in a table.

```
1 --1. What is the average customer satisfaction score across different call centers?
2 SELECT Call_Center, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
3 FROM call_center_data
4 GROUP BY Call_Center;
5
6 --2. How does customer sentiment correlate with customer satisfaction scores?
7 SELECT Sentiment, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
8 FROM call_center_data
9 GROUP BY Sentiment;
10
11 --3. Are there any patterns in customer satisfaction scores based on the Channel used?
```

	Sentiment	1.2 Average_Satisfaction
1	Positive	7.8979591836734695
2	Very Positive	9.62
3	Very Negative	2.435483870967742
4	Negative	4.61344537815126
5	Neutral	6.597402597402597

812 ms | 5 rows returned

3. Are there any patterns in customer satisfaction scores based on the Channel used?

It helps in understanding how customer satisfaction varies across different communication channels. For example, by finding customers who interact via chat tend to have higher satisfaction scores compared to those who use the phone or email. Such insights can guide the

The screenshot shows the Databricks SQL Editor interface. The left sidebar contains navigation options like Workspace, Recents, Catalog, Workflows, Compute, SQL, SQL Editor, Queries, Dashboards, Alerts, Query History, SQL Warehouses, Data Engineering, Job Runs, Data Ingestion, and Delta Live Tables. The main panel displays a SQL query with three questions and their corresponding SQL statements. The query is executed, and the results are shown in a table.

```
1 --1. What is the average customer satisfaction score across different call centers?
2 SELECT Call_Center, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
3 FROM call_center_data
4 GROUP BY Call_Center;
5
6 --2. How does customer sentiment correlate with customer satisfaction scores?
7 SELECT Sentiment, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
8 FROM call_center_data
9 GROUP BY Sentiment;
10
11 --3. Are there any patterns in customer satisfaction scores based on the Channel used?
12 SELECT Channel, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
13 FROM call_center_data
14 GROUP BY Channel;
15
```

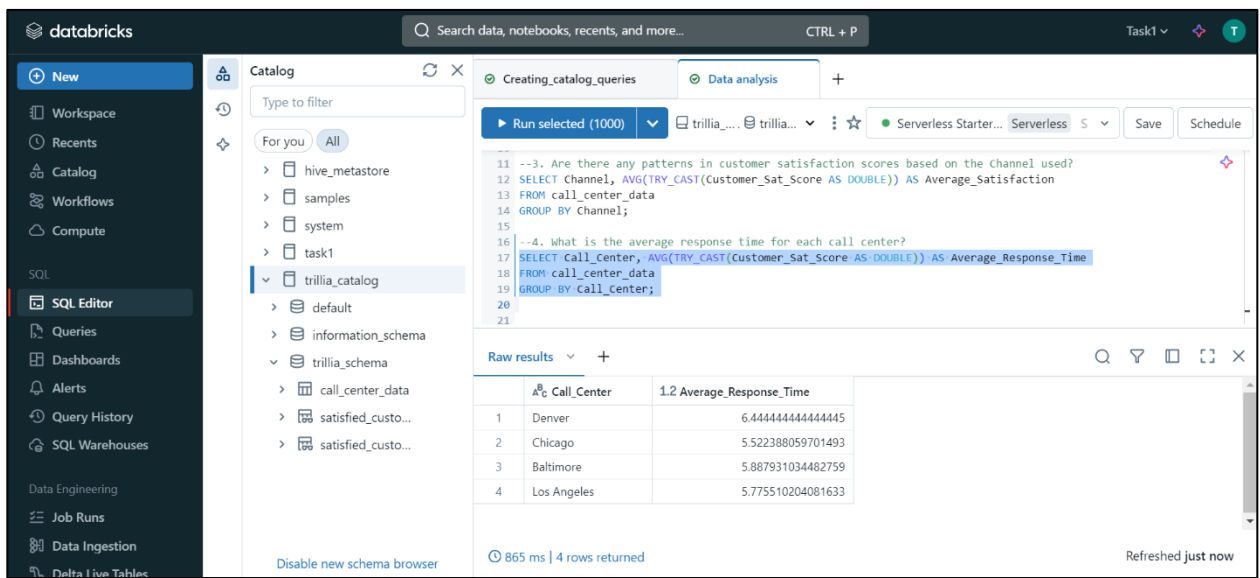
	Channel	1.2 Average_Satisfaction
1	Chatbot	5.689655172413793
2	Web	5.552631578947368
3	Call-Center	6.08256880733945
4	Email	5.8352941176470585

965 ms | 4 rows returned

call center in optimizing their customer service strategies by focusing on the channels that lead to higher satisfaction or improving those with lower satisfaction.

4. What is the average response time for each call center?

The alias `Average_Response_Time` suggests that the query is calculating the average response time, but the actual calculation is based on `Customer_Sat_Score`. The query should either calculate the average of a column representing response time or use an alias that correctly reflects the data being averaged.



The screenshot shows the Databricks SQL Editor interface. On the left is a sidebar with navigation options like Workspace, Recents, Catalog, Workflows, Compute, and SQL. The main area displays a SQL query in the 'Data analysis' tab. The query is as follows:

```
--3. Are there any patterns in customer satisfaction scores based on the Channel used?
12 SELECT Channel, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Satisfaction
13 FROM call_center_data
14 GROUP BY Channel;
15
16 --4. What is the average response time for each call center?
17 SELECT Call_Center, AVG(TRY_CAST(Customer_Sat_Score AS DOUBLE)) AS Average_Response_Time
18 FROM call_center_data
19 GROUP BY Call_center;
20
21
```

Below the query, the 'Raw results' section shows a table with 4 rows returned. The table has two columns: 'Call_Center' and '1.2 Average_Response_Time'.

Call_Center	1.2 Average_Response_Time
1 Denver	6.444444444444445
2 Chicago	5.522388059701493
3 Baltimore	5.887931034482759
4 Los Angeles	5.775510204081633

At the bottom of the results section, it indicates '865 ms | 4 rows returned' and 'Refreshed just now'.

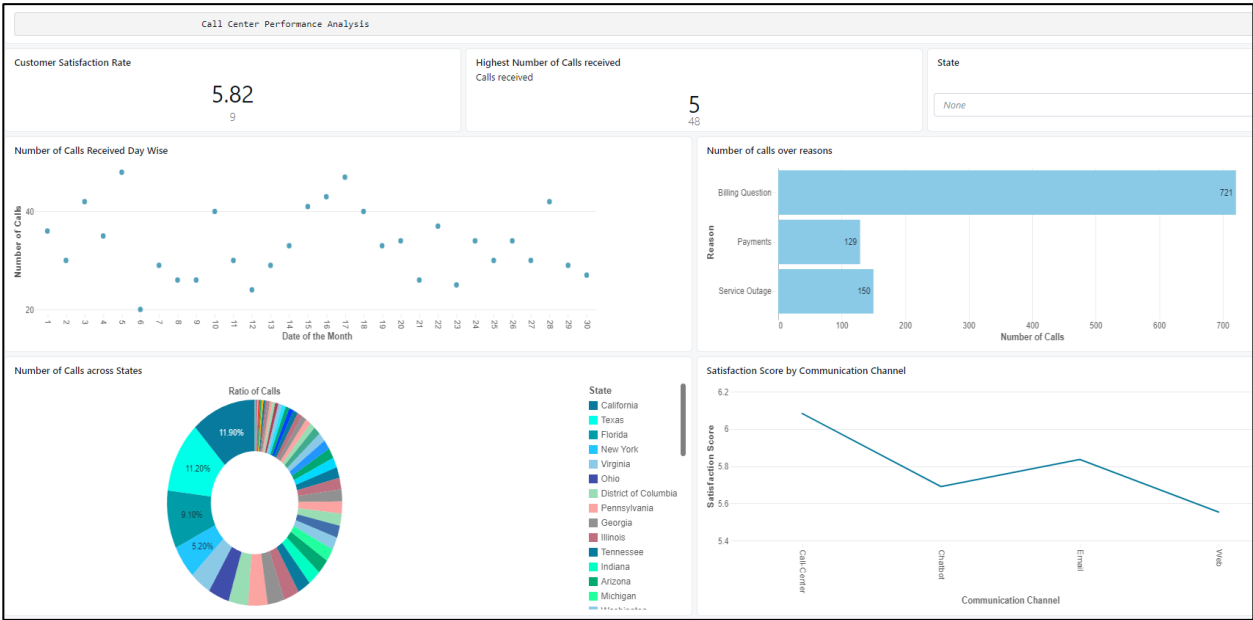
Call Center Performance Dashboard

This dashboard provides a comprehensive view of various metrics related to a call center's performance and customer satisfaction built on Databricks platform.

1. Number of Calls Received Day Wise

Scatter Plot: This chart shows the distribution of the number of calls received each day throughout a month. The points represent the number of calls, and the x-axis corresponds to

the day of the month. It helps to identify patterns in call volume across different days. It reveals that certain days consistently have higher or lower call volumes.



2. Number of Calls Across States

Pie Chart: This chart represents the proportion of calls received from different states. Each segment of the pie shows the percentage of total calls attributed to a specific state. It indicates the geographical distribution of calls, highlighting states with higher customer engagement.

3. Number of Calls Over Reasons

Bar Chart: This chart breaks down the number of calls by the reason for the call, such as "Billing Question," "Payments," and "Service Outage." It helps to identify the most common issues customers are calling about. Here, "Billing Question" has the highest call volume in this case, indicating a potential area of concern.

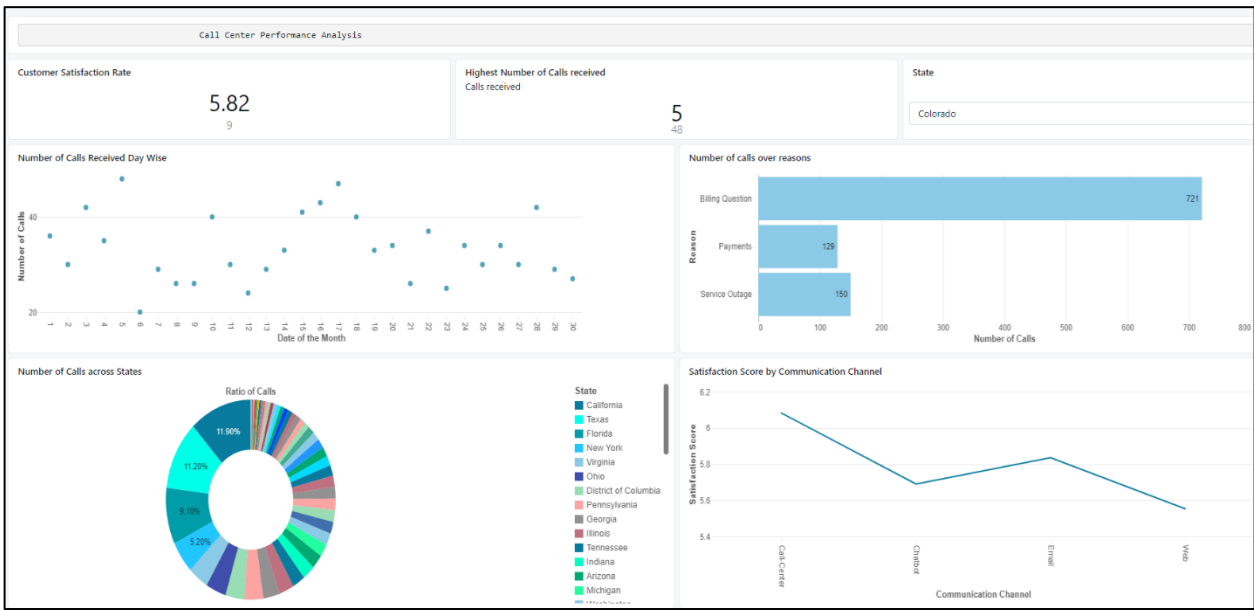
4. Filter – States of the Customers

The inclusion of a state filter will enhance the dashboard's interactivity, enabling users to drill down into specific geographic areas and analyze trends, satisfaction scores, and call volumes

for those regions. Unfortunately, in the Databricks free subscription version, the filter is not working in the dashboard.

5. Satisfaction Score by Communication Channel

Line Chart: This chart shows the average customer satisfaction score across different communication channels, such as "Call Center," "Chatbot," "Email," and "Web." It provides insights into which channels customers find most satisfying. For example, the Call Center might have the highest satisfaction score, while the Web channel has the lowest.



6. Scorecards:

Customer Satisfaction Rate: Shows the current average customer satisfaction score (5.82) against the target (9).

Highest Number of Calls Received: Indicates the highest number of calls received on a single day, specifically noting 48 calls on October 5th.

Reference

- 1."Databricks Essentials" by Databricks
- 2."Advanced Analytics with Databricks" by Databricks
3. Real World Fake Dataset for Practice -
<https://www.kaggle.com/datasets/mesumraza/real-world-fake-dataset-for-practice>