

# Contact Center Data Analysis using MS Excel and SQL

## Understanding inbound communication traffic at a contact center

### Background/Scenario

Contact centers also known as customer service centers operate in a similar way to call centers, however, they handle additional forms of communication channels apart from phone calls. Contact centers also expand to include all other communications as well. Contact centers take an Omni channel approach, incorporating not only voice communication, but also live web chats, text messaging, messenger apps, email, video chat, social media, and are even involved in managing virtual agents and chatbots.

### Project Approach

In this project we use a dataset from '[Real World Fake Data](#)', preparation of the data in the CSV file was done using **Microsoft Excel**, the CSV file was then imported into **MySQL Workbench**, cleaning and analysis of the data was done using SQL. Finally, a visualization of the data was created using **Tableau**. The tutorial/article for this project was offered by [Armonia](#) from [medium.com](#).

### Part 1: Prepare the data

Before the CSV file can be imported into the MySQL Workbench it has to be prepared. Good data preparation allows for efficient data analysis, limits errors and inaccuracies that can occur to data during analysis. This will be done using **Microsoft Excel**.

Below is a snapshot of the dataset;

	A	B	C	D	E	F	G	H	I	J	K
1	id	customer_name	sentiment	csat_score	call_timestamp	reason	city	state	channel	response_time	call duration in minutes
2	DKK-57076809-w-055481-fu	Analise Gairdner	Neutral		7 10/29/2020	Billing Question	Detroit	Michigan	Call-Center	Within SLA	17
3	QGG-72219678-w-102139-KY	Crichton Kidsley	Very Positive		10/05/2020	Service Outage	Spartanburg	South Carolina	Chatbot	Within SLA	23
4	GYJ-30025932-A-023015-LD	Averill Brundrett	Negative		10/04/2020	Billing Question	Gainesville	Florida	Call-Center	Above SLA	45
5	ZJI-96807559-i-620008-m7	Noreen Lafflina	Very Negative	1	10/17/2020	Billing Question	Portland	Oregon	Chatbot	Within SLA	12
6	DDU-69451719-O-176482-Fm	Toma Van der Beken	Very Positive		10/17/2020	Payments	Fort Wayne	Indiana	Call-Center	Within SLA	23
7	JVI-79728660-U-224285-4a	Kaylyn Emlen	Neutral	5	10/28/2020	Billing Question	Salt Lake City	Utah	Call-Center	Within SLA	25
8	AZI-95054097-e-185542-PT	Phillipe Bowring	Neutral	8	10/16/2020	Billing Question	Tyler	Texas	Chatbot	Within SLA	31
9	TWX-27007918-I-608789-Xw	Krysta de Tocqueville	Positive		10/21/2020	Billing Question	New York City	New York	Chatbot	Below SLA	37
10	XNG-44599118-P-344473-ZU	Oran Lifsey	Very Negative		10/03/2020	Billing Question	Dallas	Texas	Email	Below SLA	37
11	RLC-64108207-Z-285141-VS	Port Inggall	Neutral		10/07/2020	Billing Question	Cincinnati	Ohio	Chatbot	Within SLA	12
12	RJF-00263922-O-647027-TB	Ella Cristoforo	Negative		10/09/2020	Billing Question	Everett	Washington	Chatbot	Within SLA	35
13	ZQN-32874873-e-786499-KJ	Aubrey Surcombe	Negative		10/11/2020	Billing Question	Huntington	West Virginia	Web	Within SLA	18
14	JDP-35147568-w-630120-3I	Nicolle Fareweather	Very Positive		10/02/2020	Billing Question	Portland	Oregon	Call-Center	Within SLA	30
15	DPT-56483482-P-371409-CQ	Melesa Ricardot	Positive	7	10/10/2020	Billing Question	Springfield	Massachusetts	Chatbot	Within SLA	20
16	ZOV-95861398-a-333622-9r	Odell Cathesyed	Very Negative		10/06/2020	Payments	Hyattsville	Maryland	Call-Center	Below SLA	22

The CSV file we will work with has over **32,000 records** and **12 fields**. With such a large data set, loading the data into MySQL using the 'Table Data Import' function will take up a lot of time. In such an instance, the 'LOAD DATA FILE' SQL statement is used to load the data set.

The requirements of the 'LOAD DATA INFILE' statement is that, any columns containing numeric data with currency, percentage, thousand comma separator or any other symbol need to be removed as MySQL will treat such as data as string. Additionally, any columns containing 'dates' should be changed to the MySQL format of 'YYYY-MM-DD'. 'Blank' cells should equally be filled up with 'text/numbers' or set to 'NULL'.

### Step 1: Change all columns with dates to 'yyyy-mm-dd' format.

This task is carried out using the 'text to columns' MS Excel feature.

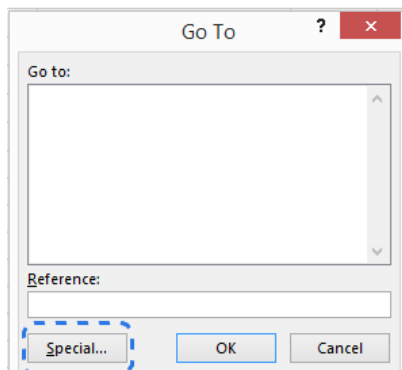
E	F
call_timestamp	reason
2020-10-29	Billing Question
2020-10-05	Service Outage
2020-10-04	Billing Question
2020-10-17	Billing Question
2020-10-17	Payments
2020-10-28	Billing Question
2020-10-16	Billing Question
2020-10-21	Billing Question
2020-10-03	Billing Question
2020-10-07	Billing Question

### Step 2: Fill all blank cells in columns with numeric data with zeros.

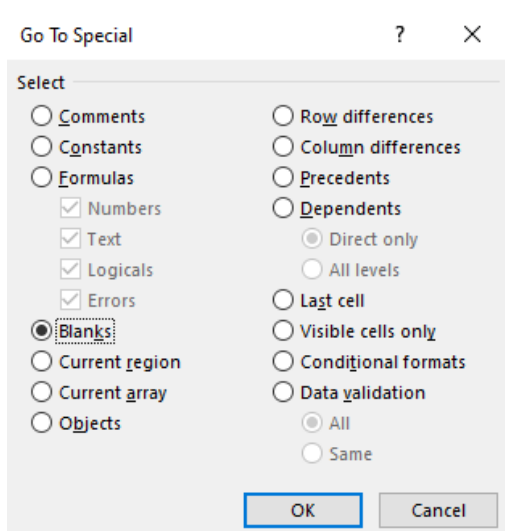
The 'csat\_score' is one such column with blank cells. The entire column is selected by typing the range of cells needed in the 'Name Box', in this instance, 'D2:D3242' is typed in the Name Box.

C	D
sentiment	csat_score
Neutral	7
Very Positive	
Negative	
Very Negative	1
Very Positive	
Neutral	5
Neutral	8
Positive	
Very Negative	

Thereafter, use the keyboard shortcut 'Ctrl + G' to display the 'Go To' dialog box. On the 'Go To' dialog box, click on the 'Special' button.



Next, select the Blanks radio button and click OK.



To enter 0 in the blank cells, press F2 to enter a value in the active cell.

C	D
sentiment	csat_score
Neutral	7
Very Positive	0
Negative	
Very Negative	1
Very Positive	
Neutral	5
Neutral	8

Type in the number or text you want.

Press Ctrl + Enter and all the selected blank cells are populated with 0s in this case.

D
csat_score
7
0
0
1
0
5
8

### Step 3: Save the CSV file in the folder containing the database.

In this case, the Call Center CSV file is saved in the following directory:

'C:\Program Data\MySQL\MySQL Server 8.0\Data\call\_center\_project\call center.csv'

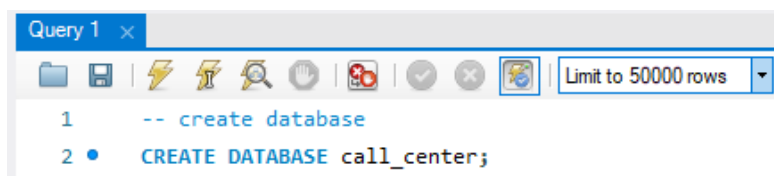
## Part 2: Import the Data

The call-center dataset basically has over 32,900 records and 12 fields of data that describes calls made to various call centers. It includes the ID of the call, duration of the call in minutes, the name of the person who called, their satisfaction score and many other attributes.

After the dataset has been downloaded and prepared, it's now time to import it into MySQL Workbench. For Part 2, the tasks will be carried out using **SQL**.

### Step 1: Create a database

First off, there's need to have a database where the data will be imported into. The database can be created from scratch or an existing one can be used. For this project, a new database will be created.



A new database named 'call\_center' has been created. The two – hyphens at the beginning of line 1 indicate that it is a comment and it doesn't affect the code.

## Step 2: Activate the database

Next, to get to work on a specific database there's need to select it, and to do that the 'use' keyword is typed.

```
3
4  -- activate database
5 •  USE call_center;
```

## Step 3: Create a table

After this, we need to create a table in the database that will fit the data and match it. The SQL statement below is used to create a table named 'caller';

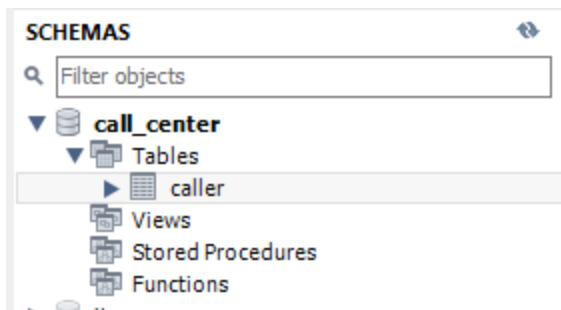
```
7  -- create table caller
8 •  CREATE TABLE caller (
9      ID char (50),
10     Cust_Name char (50),
11     Sentiment char (20),
12     Csat_Score int,
13     Call_time_Stamp char (10),
14     Reason char (20),
15     City char (20),
16     State char (20),
17     Comm_Channel char (20),
18     Response_Time char (20),
19     Call_Duration_Mins int,
20     Call_Center char (20)
21 );
```

Here a table called 'caller' is created, and designed in a way to fit the CSV dataset by matching the columns and data types. It's done column by column as in the CSV file and in the same order when building the 'caller' table.

The columns are created in this manner (in example above): first off, specify the column name, then add the data type of that column and in the parenthesis the size of the variable is entered.

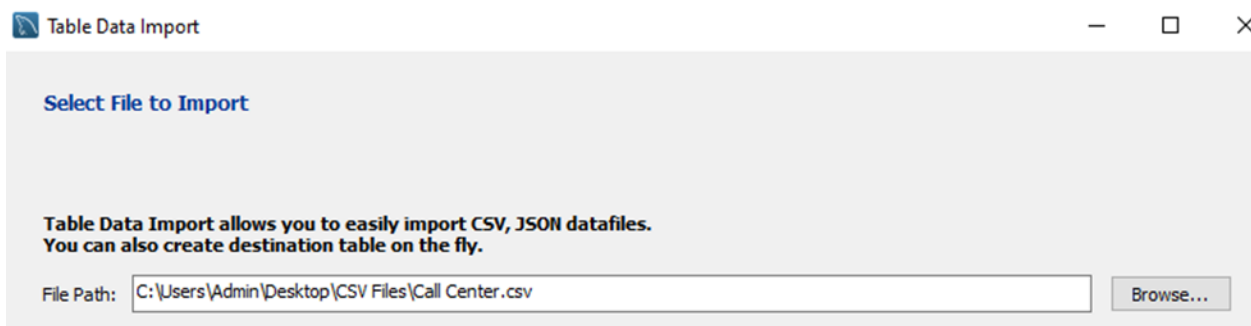
It's imperative to check the data in the CSV file before specifying the field sizes in the database table. For example, by specifying that city, or state, or whatever field it is, is of max size of 20 means that rows which contain fields that have more than 20 characters will not be imported.

Go to the left panel of the MySQL Workbench; double click on the database name in the left panel and then right click on the table recently created: If the created table does not appear in the left panel, click the refresh button next to 'SCHEMAS'.

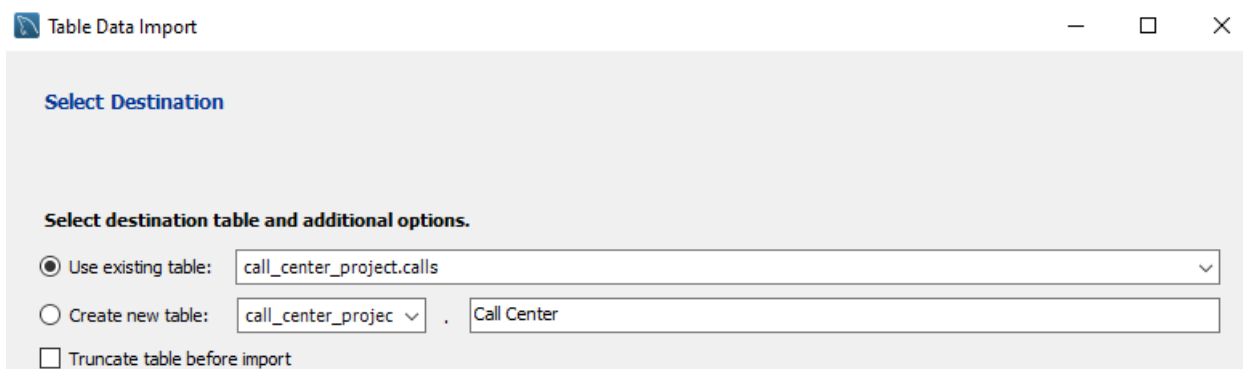


#### Step 4: Import the csv file

After right clicking, there will be an option called 'Table Data Import Wizard'. Click on it, it will open a window to browse for the csv file that has to be imported, search for it and then when the file is located click on next button:



After clicking next, it will ask for the destination where the data will go to:



As the data needs to go to the table we recently created choose the 'Use existing table' and then press next.

Next, ensure the source and destination columns match. Where necessary make changes using the drop down arrow. When the columns match, click on the next button and finally the finish button when the dataset has been loaded into MySQL.

**Table Data Import**

**Configure Import Settings**

Detected file format: csv

Encoding: utf-8

Columns:

Source Column	Dest Column
<input checked="" type="checkbox"/> id	ID
<input checked="" type="checkbox"/> customer_name	Cust_Name
<input checked="" type="checkbox"/> sentiment	Sentiment
<input checked="" type="checkbox"/> csat_score	Csat_Score
<input checked="" type="checkbox"/> call_timestamp	Call_time_Stamp
<input checked="" type="checkbox"/> reason	Reason

In the event that 'Table Data Import' function in MySQL takes too long to import the CSV, the '**LOAD DATA INFILE**' statement can be used.

```
--
21  -- import the csv file using LOAD DATA INFILE
22  •  LOAD DATA INFILE "Call_Center.csv" INTO TABLE calls
23  FIELDS TERMINATED BY ','
24  IGNORE 1 LINES;
25
26  •  SELECT * FROM calls;
```

After successful execution of the statement above all the records are imported successfully. As, the table is way too big to see all of it, so let's see first 10 rows using the SQL statement below.

```
1  •  SELECT * FROM calls LIMIT 10;
```

Below is a screenshot of the imported CSV file records:

ID	Cust_Name	Sentiment	Csat_Score	Call_time_Stamp	Reason	City	State
DKK-57076809-w-055481-fJ	Analise Gairdner	Neutral	7	2020-10-29	Billing Question	Detroit	Michig
QGK-72219678-w-102139-KY	Crichton Kidsley	Very Positive	0	2020-10-05	Service Outage	Spartanburg	South
GYJ-30025932-A-023015-LD	Averill Brundrett	Negative	0	2020-10-04	Billing Question	Gainesville	Florida
ZJI-96807559-i-620008-m7	Noreen Lafflina	Very Negative	1	2020-10-17	Billing Question	Portland	Orego
DDU-69451719-O-176482-Fm	Toma Van der Beken	Very Positive	0	2020-10-17	Payments	Fort Wayne	Indian
JVI-79728660-U-224285-4a	Kaylyn Emlen	Neutral	5	2020-10-28	Billing Question	Salt Lake City	Utah
AZI-95054097-e-185542-PT	Phillipe Bowring	Neutral	8	2020-10-16	Billing Question	Tyler	Texas
TWX-27007918-I-608789-Xw	Krysta de Tocqueville	Positive	0	2020-10-21	Billing Question	New York City	New Y

## Part 3: Clean the data.

### Step 1: Convert datatypes

Convert the 'call\_timestamp' field has a string datatype in the database, It has to be converted to date.

```

28
29 • SELECT * FROM caller;
30
31 -- Convert the 'call_timestamp' field has a string datatype in the database, It has to be converted to date
32 • SET SQL_SAFE_UPDATES = 0;
33 • UPDATE caller SET Call_time_Stamp = str_to_date(str_to_date, '%d/%m/%Y');
34
35 • SET SQL_SAFE_UPDATES = 1;
36
37 • SELECT * FROM caller;
--

```

There's a need to set the 'SQL\_SAFE\_UPDATES' off before making any change the column. The reason is because we don't specify a 'where' clause that uses a 'KEY' column. That is why it's set off before the query, and then set it back on after. and the results?

Result Grid								
Filter Rows:								
Export: Wrap Cell Content: Fetch rows:								
	ID	Cust_Name	Sentiment	Csat_Score	Call_time_Stamp	Reason	City	State
▶	DKK-57076809-w-055481-fU	Analise Gairdner	Neutral	7	2020-10-29	Billing Question	Detroit	Michigan
	QKG-72219678-w-102139-KY	Crichton Kidsley	Very Positive	0	2020-10-05	Service Outage	Spartanburg	South Carolina
	GYJ-30025932-A-023015-LD	Averill Brundrett	Negative	0	2020-10-04	Billing Question	Gainesville	Florida
	ZJI-96807559-i-620008-m7	Noreen Lafflina	Very Negative	1	2020-10-17	Billing Question	Portland	Oregon
	DDU-69451719-O-176482-Fm	Toma Van der Beken	Very Positive	0	2020-10-17	Payments	Fort Wayne	Indiana
	JVI-79728660-U-224285-4a	Kaylyn Emlen	Neutral	5	2020-10-28	Billing Question	Salt Lake City	Utah
	AZI-95054097-e-185542-PT	Phillipe Bowring	Neutral	8	2020-10-16	Billing Question	Tyler	Texas
	THX-37007010-T-600700-Yu	Krista de Tressenville	Positive	0	2020-10-21	Billing Question	New York City	New York

### Step 2:

The minimum customer satisfaction score (Csat\_Score) has to 1 and not 0, this is because a value of 0 might mess up the aggregations. There are two options: either the field data type is set them to NULL or we just leave them be and then when the table is queried a WHERE clause is added indicating that Csat\_Score != 0.

But the data type will be set to 'NULL' in this way:

```

-- Convert the 'call_timestamp' field has a string datatype in the database, It has to be converted to date
SET SQL_SAFE_UPDATES = 0;
UPDATE caller SET Call_time_Stamp = str_to_date(str_to_date, '%d/%m/%Y');

-- Set the Csat_Score to NULL if the value is 0
UPDATE caller SET Csat_Score = NULL WHERE Csat_Score = 0;
SET SQL_SAFE_UPDATES = 1;

SELECT * FROM caller;

```



### Step 3: Replace vague data with appropriate term.

In the 'Comm\_Channel' field, the term 'Call-Center' needs to be replaced with 'Calls'. Calls is an appropriate term under communication channels instead of Call-Center.

```

45  -- replace Call-Center with Calls in Comm_Channel field
46  •  UPDATE caller
47    SET Comm_Channel = 'Calls'
48    WHERE Comm_Channel = 'Call-Center';

```

time_stamp	Reason	City	State	Comm_Channel	Response_Time	Call_Duration_Mins	Call_Center
10-06	Payments	Hyattsville	Maryland	Calls	Below SLA	22	Baltimore/MD
10-18	Billing Question	New York City	New York	Chatbot	Within SLA	28	Denver/CO
10-11	Billing Question	Huntsville	Alabama	Email	Above SLA	36	Baltimore/MD
10-30	Billing Question	Wichita	Kansas	Calls	Above SLA	37	Baltimore/MD

Our data has been cleaned and is now ready to be explored for insight generation.

## Part 4: Exploratory Data Analysis (EDA)

### Step1: Number of columns and rows in the 'callers' table

What is the shape of the database table, i.e., the number of columns and rows? The following SQL statements are used:

```

45  -- Exploring our data
46  -- Check the number of rows
47  •  SELECT COUNT(*) AS rows_num FROM caller;
48  -- check the number of columns
49  •  SELECT COUNT(*) AS cols_num FROM information_schema.columns WHERE table_name = 'caller';

```

The screenshots below are the output after executing the SQL statements in lines 47 and 49.

rows_num
38934

cols_num
12

So the 'caller' table has **38,934 records** and **12 columns**.

## Step 2: Distinct values

```

51  -- Look up distinct values in some of the columns.
52  • SELECT DISTINCT Sentiment FROM caller;
53  • SELECT DISTINCT Call_Center FROM caller;
54  • SELECT DISTINCT Reason FROM caller;
55  • SELECT DISTINCT Comm_Channel FROM caller;
56  • SELECT DISTINCT Response_Time FROM caller;

```

The results are as follows:

Sentiment	Call_Center	Reason	Comm_Channel	Response_Time
Neutral	Los Angeles/CA	Billing Question	Call-Center	Within SLA
Very Positive	Baltimore/MD	Service Outage	Chatbot	Above SLA
Negative	Denver/CO	Payments	Email	Below SLA
Very Negative	Chicago/IL		Web	
Positive				

The distinct values for the five columns are as displayed in their respective screenshots.

## Step 3: Answering business questions.

### Step 3.1: What are the sentiments expressed by clients?

```

65  -- sentiments expressed by clients
66  • SELECT Sentiment, COUNT(*), ROUND((COUNT(*) / (SELECT COUNT(*) FROM caller)) * 100, 1) AS pct
67  FROM caller GROUP BY 1 ORDER BY 3 DESC;

```

Sentiment	COUNT(*)	pct
Negative	13400	33.5
Neutral	10663	26.6
Very Negative	7295	18.2
Positive	4789	12.0
Very Positive	3870	9.7

Based on the data in the screenshot above, it is evident that most of the sentiments expressed by callers are '**Negative**', as can be seen by the range topping '**33.5%**'. The '**Very Positive**' sentiment is least among callers amounting to '**9.7%**' of the calls.

### Step 3.2: Which day had the highest number of calls?

```

74 -- Which day has the most calls
75 • SELECT DAYNAME(Call_time_Stamp) AS Day_of_Call, COUNT(*) Num_of_Calls FROM caller
76 GROUP BY 1 ORDER BY 2 DESC;

```

Day_of_Call	Num_of_Calls
Friday	6754
Thursday	6687
Wednesday	5400
Saturday	5341
Tuesday	5335
Monday	5309
Sunday	5191

Based on the screenshot above, **Friday** has the **highest** number of calls amounting to **6,754** while **Sunday** has the **least** with **5,191**.

### Step 3.3: What are the reasons for clients calling?

```

68 -- reason for contacting the call center
69 • SELECT Reason, COUNT(*), ROUND((COUNT(*) / (SELECT COUNT(*) FROM caller)) * 100, 1) AS pct
70 FROM caller GROUP BY 1 ORDER BY 3 DESC;

```

Reason	COUNT(*)	pct
Billing Question	28557	71.4
Service Outage	5766	14.4
Payments	5694	14.2

With regards to why clients contact the call center, it is evident that most of the times it's for **Billing Questions** which amounts to **71.4%**. On the other hand, **Service Outage** and **Payments** are more or less the same with **14.4%** and **14.2%** respectively.

### Step 3.4: Which communication channels are frequently used by clients?

```

70 -- frequently used communication channels
71 • SELECT Comm_Channel, COUNT(*), ROUND((COUNT(*) / (SELECT COUNT(*) FROM caller)) * 100, 1) AS pct
72 FROM caller GROUP BY 1 ORDER BY 3 DESC;

```

Comm_Channel	COUNT(*)	pct
Calls	12837	32.1
Chatbot	10080	25.2
Email	9079	22.7
Web	8021	20.0

From the screenshot above, it is apparent that the majority of inbound communication is taken up by **Calls** with **32.1%** and the least being **Web** communication with **20.0%**.

### Step 3.5: Which SLA response time category was the highest?

```

79  -- response time
80  • SELECT Response_Time, COUNT(*), ROUND((COUNT(*) / (SELECT COUNT(*) FROM caller)) * 100, 1) AS pct
81  FROM caller GROUP BY 1 ORDER BY 3 DESC;

```

Response_Time	COUNT(*)	pct
Within SLA	25090	62.7
Below SLA	9864	24.6
Above SLA	5063	12.7

A Service Level Agreement (SLA) response time is the time it takes a provider to respond to an inquiry or request from a client. Based on the screenshot, **62.7%** of the inquiries were handled **within SLA** time while **24.6%** of inquiries were **Below SLA**, finally, **12.7%** were **Above SLA**.

### Step 3.6: Which contact center had the highest communication traffic?

```

68  -- contact center with highest communication traffic
69  • SELECT Call_Center, COUNT(*), ROUND((COUNT(*) / (SELECT COUNT(*) FROM caller)) * 100, 1) AS pct
70  FROM caller GROUP BY 1 ORDER BY 3 DESC;

```

Call_Center	COUNT(*)	pct
Los Angeles/CA	16700	41.7
Baltimore/MD	13342	33.3
Chicago/IL	6595	16.5
Denver/CO	3380	8.4

The **Los Angeles/CA** Call Center had the highest communication traffic of **16,700** which comprised of **41.7%** of the total. The center with the least traffic was **Denver/CO** having **3,380** a representation of **8.4%** of communication traffic.

**Step 3.7: What is the minimum and maximum customer satisfaction score?**

```

90  -- minimum and maximum customer satisfaction score
91  •  SELECT MIN(Csatscore) AS min_score, MAX(Csatscore) AS max_score,
92     ROUND(AVG(Csatscore), 1) AS avg_score FROM caller WHERE Csatscore != 0;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	min_score	max_score	avg_score
▶ 1	10	5.6	

The minimum and maximum customer scores are **5.6** and **10** respectively.

**Step 3.8: How long were the calls to the service center?**

```

96  •  SELECT MIN(Call_Duration_Mins) AS min_call_duration, MAX(Call_Duration_Mins) AS
97     max_call_duration, ROUND(AVG(Call_Duration_Mins), 2) AS avg_call_duration FROM caller;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	min_call_duration	max_call_duration	avg_call_duration
▶ 5	45	25.02	

The call durations were as follows; minimum: **5mins**, average: **25.02mins** and maximum: **45 mins**.

**Step 3.9 What were the average call durations based on call centers?**

```

101 •  SELECT Call_Center, ROUND(AVG(Call_Duration_Mins), 2) FROM caller GROUP BY 1
102     ORDER BY 2 DESC;

```

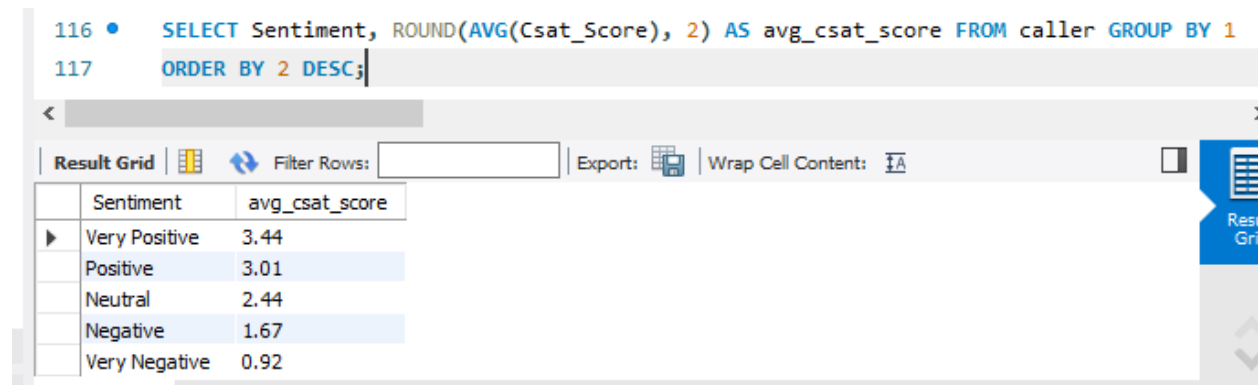
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	Call_Center	ROUND(AVG(Call_Duration_Mins), 2)
▶	Chicago/IL	25.06
	Los Angeles/CA	25.03
	Baltimore/MD	24.99
	Denver/CO	24.95

The call center in **Chicago/IL** had the highest average call duration at **25.0mins** whereas the one in **Denver/CO** had the lowest average call duration of **24.95mins**.

**Step 3.10: What was the relation between the sentiments expressed by the customer and average customer satisfaction score?**

```
116 • SELECT Sentiment, ROUND(AVG(Csatscore), 2) AS avg_csatscore FROM caller GROUP BY 1
117 ORDER BY 2 DESC;
```



Sentiment	avg_csatscore
Very Positive	3.44
Positive	3.01
Neutral	2.44
Negative	1.67
Very Negative	0.92

As expected, the average customer satisfaction score of **3.44** was in correlation to the '**Very Positive**' sentiment expressed by the customers. On the other hand, customers who expressed a '**Very Negative**' sentiment, resulted in an average customer satisfaction score of **0.92**.

## Part 5: Data Visualization

Finally, a dashboard was created using Tableau, the link to the visualization is [here](#).

Sneak peek:

