

CALIFORNIA STATE UNIVERSITY, FULLERTON

Department of Computer Science

Final Project Submission
for
CPSC 531- Advanced Database Management System

**HADOOP SPARK-BASED REAL-TIME ONLINE
SALES DATA ANALYSIS**

Submitted by:

Meghana Bodapati ThirumalaNaidu, CWID: 885206029
Pooja Honneshwari Ravi, CWID: 885237307

CONTENTS

1. Problem Statement
2. Aim of the project
3. Approach
4. Tools and Technologies used
5. Architecture
6. Dataset Information
7. Steps to run the application
8. Code Link
9. Test Results
10. Future Scope
11. Conclusion

LIST OF FIGURES

Figure Number	Description
Fig 1	Internal working of Kafka
Fig 2	Real-time data analysis application architecture
Fig 3	Dataset: customers.csv
Fig 4	Dataset: orders.csv
Fig 5	Dataset: products.csv
Fig 6	Streaming of Data in Kafka Producer
Fig 7	Streaming of Data in Kafka Consumer
Fig 8	Schema of data frames generated, of DataStream of orders received from Kafka-consumer
Fig 9	Schema of data frames generated, of static customer data from the dataset
Fig 10	Schema of data frames generated, of forked data of customers and orders used for data analysis
Fig 11	Processed data stored in Mysql Database
Fig 12	Raw input data stored in Cassandra Database
Fig 13	Total Sales in different states with Source Organic
Fig 14	Total Sales in different states with Source Affiliate
Fig 15	Total Sales in different states with Source Facebook
Fig 16	Total Sales in different states with Source Google
Fig 17	Total Sales in different states with Source Twitter

PROBLEM STATEMENT

Everyone in the modern world now follows the trend of online purchasing. E-commerce websites have made it easier and is the mainstream market for businesses to connect with clients, regardless of their location, interests, or preferences.

It's crucial to have an examination of solid sales data. Decision-making is aided, sales quality is maintained, and clients are better understood, all of which improve performance going forward.

AIM OF THE PROJECT

The goal of this project is to gather real-time sales information for goods sold across various platforms in several states. The outcome generated depicts the sales of 5 platforms such as Facebook, Twitter , Google, Affiliate and Organic (product's own website) in different states, which aids the enhancement of cross-platform advertising tactics.

APPROACH TO DESIGN THE PROJECT

- Kafka is an open-source distributed stream-processing framework that collects real-time data from the e-commerce website and stores it in Kafka-broker(Cluster).
- This stream of data is directed to the processing engine (Spark) through Kafka-consumer.
- The data streams undergo various processes such as Data injection, Data Storage, Data Processing, and Data Analysis.
- The analysis results are stored in MySQL and the raw data is stored in Cassandra (because for further enhancement in the application, where-in we intend to add prediction models, data of any format can be stored as it is a NO-SQL-based database).
- After analyzing, the visualization outcome is depicted in the Superset application.

Kafka Internal Functioning

- The topic orderstopic is created in the Kafka-Cluster.
- Kafka mainly has two important API's.
- Kafka-Producer collects all the stream of data from the e-commerce website and allows the application to write that into the topic in the cluster.
- From the other side, the kafka-Consumer allows the application to read the stream of data from the topic in the cluster and gives it to spark for further processing.

Apache Spark Structured Streaming with Kafka using Python(PySpark)

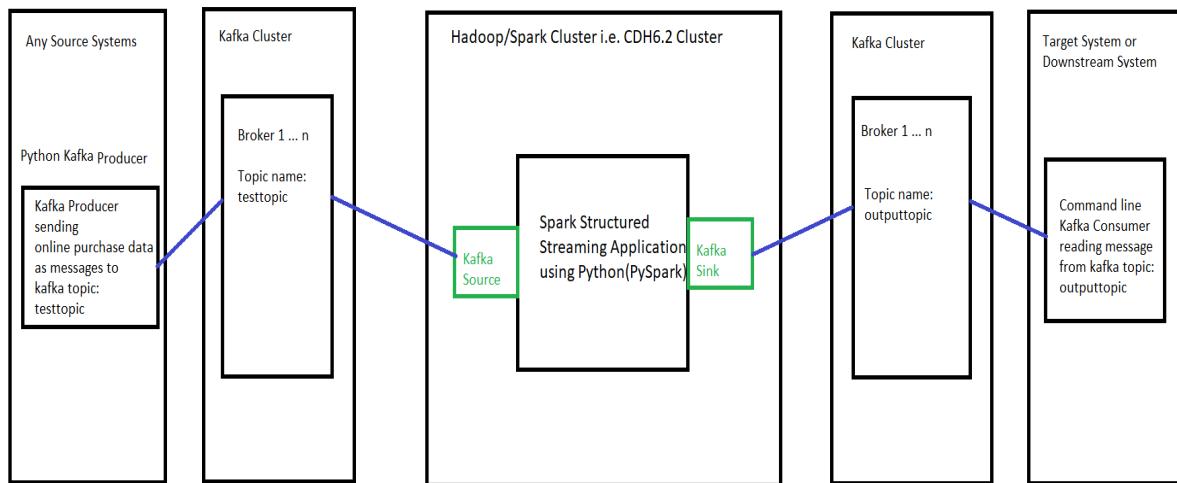


Fig 1: Internal working of Kafka

TOOLS AND TECHNOLOGIES USED

- Hadoop – open-source software utility used for storage and computation of data.
- Kafka – used for capturing and storing streams of real-time data into categories called topics and later gives it to a processing engine for analysis.
- Spark – a data processing engine used for collecting data-stream from Kafka and further storing, processing, and analyzing the data.

- MySQL – is an open-source RDMS used to store the analyzed data.
- Cassandra – NoSQL database management system used to store raw data.
- Superset- an open-source software application for data exploration and data visualization.
- Python (Anaconda, PyCharm) – The programming language used for the analysis.

ARCHITECTURE

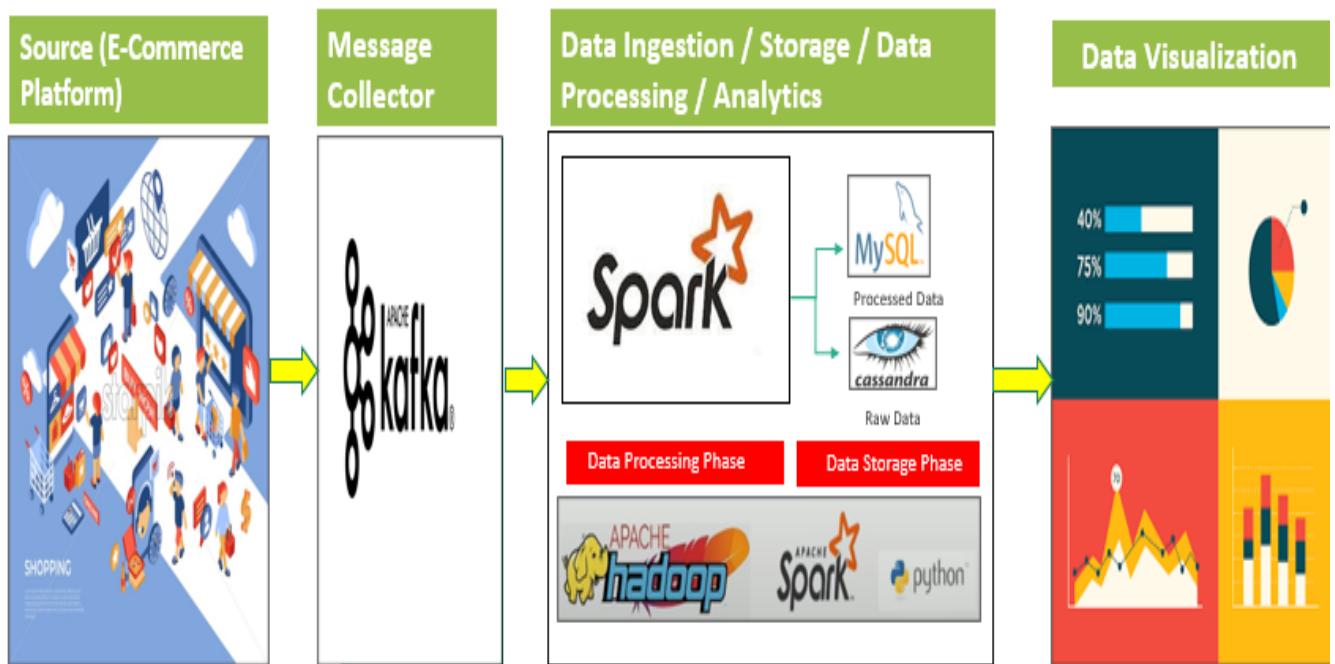


Fig: Real-Time Data Analysis Application Architecture

DATASET INFORMATION

Three datasets were used:

- Customers dataset
- Orders dataset
- Products dataset

Data Attributes: customers.csv

- ID – customer's id
- Name – customer's name
- Address – customer's address
- Birth Date – customer's date of birth
- City – customer's city he lives
- Created At - customer's account created date
- Email - customer's email address
- Latitude - customer's location details
- Longitude - customer's location details
- Password - customer's account password
- Source - customer's source through which he made an order
- State - customer's state
- Zip - customer's area zip code

ID	Name	Address	Birth Date	City	Created At	Email	Latitude	Longitude	Password	Source	State	Zip
1	Hudson Borer	9611-9809 West Rosedale Ro	12/12/1986	Wood River	2017-10-07T01:34:35	borer-hudson@yahoo.com	40.7131489	-98.5259864	ccca881f-3e4	Twitter	NE	68883
2	Domenica Williamson	101 4th Street	6/10/1967	Searsboro	2018-04-09T12:10:05	williamson-domenica@yaho	41.5813224	-92.6991321	eafcf45bf-cf8	Affiliate	IA	50242
3	Lina Heaney	29494 Anderson Drive	12/18/1961	Sandstone	2017-06-27T06:06:20	lina.heaney@yahoo.com	46.1197304	-92.8416108	36f6f7891-34e	Facebook	MN	55072
4	Arnold Adams	2-7900 Cuerno Verde Road	8/12/1992	Rye	2019-02-21T13:59:15	adams.arnold@gma	37.9202933	-104.972691	537a727b-75	Google	CO	81069
5	Dominique Leffler	761 Fish Hill Road	4/20/1974	Beaver Dams	2017-09-05T03:36:44	leffler.dominique@	42.348954	-77.056681	6a802b6c-4d	Twitter	NY	14812
6	Rene Muller	1243 West Whitney Street	3/27/1983	Morse	2016-09-22T10:08:29	rene.muller@gmail.c	30.1514772	-92.4861786	760495fb-9c3	Google	LA	70559
7	Roselyn Bosco	630 Coaker Road	1/19/1996	Leakesville	2018-05-24T06:18:20	bosco.roselyn@hot	31.2341055	-88.5856948	43adf-055	Facebook	MS	39451
8	Aracely Jenkins	1167 East 570th Avenue	6/5/1973	Pittsburg	2017-08-18T02:55:11	aracely.jenkins@gm	37.4347209	-94.6426865	6bb01b7f-64	Facebook	KS	66762
9	Anais Ward	5816-5894 280th Street	10/16/1999	Ida Grove	2018-10-13T23:52:56	ward.anais@gmail.c	42.2979021	-95.4673587	a3de5208-2f	Twitter	IA	51445
10	Tressa White	13081-13217 Main Street	1/13/1968	Upper Sandusky	2018-08-16T12:16:30	white.tressa@yahoo.com	40.8006673	-83.2838391	81052233-b3	Google	OH	43351
11	Lolita Schaefer	495 Juniper Road	8/20/1982	Pilot Mound	2018-03-19T07:17:22	schaefer-lolita@hot	42.1394217	-93.982366	cddd766-92	Facebook	IA	50223
12	Ciara Larson	16701-16743 449th Avenue	12/16/1982	Florence	2018-06-27T17:25:24	ciara-larson@hotmail.c	44.9564152	-97.2287266	eb925d11-ea	Facebook	SD	57235
13	Mustafa Thiel	2993 Hoskins Ranch Road	7/20/1963	Santa Ysabel	2017-09-21T16:21:06	mustafa.thiel@hotn	33.08172	-116.661853	bb85c7fa-431	Facebook	CA	92070
14	Lavonne Senger	3964 Chico River Road	9/22/1963	Chico	2018-06-17T23:29:39	senger.lavonne@ya	39.6485802	-121.934332	8bee8bc9-a3	Facebook	CA	95928
15	Bertrand Romaguera	258 Opal Road	2/14/2000	El Paso	2018-12-20T17:50:32	romaguera.bertrand@	35.1080351	-92.0101859	2734ae7a-aa	Twitter	AR	72045
16	Khalid Pouros	3234 County Road 7	9/16/1961	Greensboro	2016-05-14T23:44:56	khalid.pouros@yahoo.com	32.7042678	-87.5413596	e60e4c1e-c9	Google	AL	36744
17	Lora Cronin	6111 Rogue Riv	2/16/1990	Gold Beach	2017-03-05T20:38:57	lora-cronin@gmail.c	42.5048529	-124.187837	1b414387-e9	Organic	OR	97444
18	Arely Pollich	13116 Northeast Sandy Boule	6/27/1990	Portland	2017-04-25T20:11:39	pollich-arely@yahoo.com	45.554885	-122.527744	251cd4bd-d7	Affiliate	OR	97230
19	Wyman Kertzmann	1460 Grays Creek Road	7/23/1975	Grants Pass	2018-12-03T07:27:47	wyman-kertzmann@	42.3222669	-123.324017	89ef82c4-55e	Affiliate	OR	97527

Fig 3: Dataset: customers.csv

Data Attributes: orders.csv

- ID – order id
- Created At – order created date
- Discount – order discount
- Product ID – product id of the product the customer ordered
- Quantity – quantity of the product ordered
- Subtotal – product amount
- Tax – tax for the product
- Total – total amount including tax
- User Id – customer's id

ID	Created At	Discount	Product ID	Quantity	Subtotal	Tax	Total	User ID
1	2019-02-11T21:40:27.892+05:30			14	2	37.64814539	2.07	39.71815
2	2018-05-15T08:04:04.580+05:30			123	3	110.9314565	6.1	117.0377
3	2019-12-06T22:22:48.544+05:30	6.416679209		105	2	52.72352144	2.9	55.62209
4	2019-08-22T16:30:42.392+05:30			94	6	109.2186416	6.01	115.2207
5	2018-10-10T03:34:47.309+05:30			132	5	127.8819703	7.03	134.9419
6	2019-11-06T16:38:50.134+05:30			60	3	29.80214752	1.64	31.44168
7	2018-09-11T11:22:26.521+05:30			55	5	95.77128576	5.27	101.0611
8	2019-06-17T02:37:41.693+05:30	8.65395293		65	7	68.22769726	3.75	71.9646
9	2017-05-03T16:00:54.923+05:30	3.594742155		184	3	77.39827487	4.26	81.67427
10	2020-01-17T01:44:37.233+05:30			6	2	97.43621265	5.36	102.7706
11	2018-07-22T20:31:01.969+05:30			76	6	63.82421061	3.51	67.30536
12	2018-06-26T23:21:13.271+05:30			7	7	148.2290053	10.19	158.4454
13	2019-04-06T01:04:43.973+05:30	2.117341034		70	2	57.49300381	3.95	61.42339
14	2017-05-25T20:50:37.137+05:30			139	4	51.18512213	3.52	54.67137
15	2018-06-26T02:24:38.715+05:30			116	5	114.4248513	7.87	122.3092
16	2017-12-14T11:28:30.031+05:30			68	1	76.82895922	5.28	82.15965
17	2020-02-06T12:14:36.282+05:30			48	1	123.2088425	8.47	131.6599
18	2020-04-10T23:29:46.804+05:30			12	2	116.0142758	7.98	123.9353
19	2019-02-14T07:28:31.104+05:30			136	1	105.2040232	7.23	112.5304

Fig 4: Dataset: orders.csv

Data Attributes: products.csv

- ID – product id
- Category – product category
- Created At – a product created date
- EAN (European Article Number) – unique number for the product
- Price – product amount
- Rating – product rating
- Title – name of the product
- Vendor – vendor name

ID	Category	Created At	Ean	Price	Rating	Title	Vendor
1	Gizmo	2017-07-19T19:44:56.582+05:30	1.01895E+12	29.46326	4.6	Rustic Paper Wallet	Swaniawski, Casper and Hill
2	Doohickey	2019-04-11T08:49:35.932+05:30	7.66352E+12	70.0799	0	Small Marble Shoes	Balistreri-Ankunding
3	Doohickey	2018-09-08T22:03:20.239+05:30	4.96628E+12	35.38874	4	Synergistic Granite Ch	Murray, Watsica and Wunsch
4	Doohickey	2018-03-06T02:53:09.937+05:30	4.1345E+12	73.99178	3	Enormous Aluminum	Regan Bradtke and Sons
5	Gadget	2016-10-03T01:47:39.147+05:30	5.49974E+12	82.7451	4	Enormous Marble Wa	Price, Schultz and Daniel
6	Doohickey	2017-03-29T05:43:40.150+05:30	2.29334E+12	64.95748	3.8	Small Marble Hat	Nolan-Wolff
7	Doohickey	2017-06-03T03:07:28.061+05:30	1.57967E+11	98.81934	4.3	Aerodynamic Linen C	Little-Pagac
8	Doohickey	2018-04-30T15:03:53.193+05:30	1.07877E+12	65.89216	4.1	Enormous Steel Wat	Senger-Stamm
9	Widget	2019-02-07T08:26:25.647+05:30	7.21747E+12	58.31312	4.2	Practical Bronze Comp	Keely Stehr Group
10	Gizmo	2017-01-09T09:51:20.352+05:30	1.80796E+12	31.78622	4.3	Mediocre Wooden Ta	Larson, Pfeffer and Klocko
11	Gadget	2018-05-28T08:02:54.482+05:30	3.64241E+12	88.30453	0	Ergonomic Silk Coat	Upton, Kovacek and Halvorson
12	Gizmo	2017-11-12T14:51:16.221+05:30	9.48247E+12	77.34285	4.4	Sleek Paper Toucan	Mueller-Dare
13	Gizmo	2016-05-24T23:09:46.392+05:30	3.99569E+11	75.08617	0	Synergistic Steel Chai	Mr. Tanya Stracke and Sons
14	Widget	2017-12-31T14:41:56.870+05:30	8.83342E+12	25.09876	4	Awesome Concrete SI	McClure-Lockman
15	Widget	2016-09-08T14:42:57.264+05:30	5.88165E+12	25.09876	4	Aerodynamic Paper C	Friesen-Anderson
16	Gadget	2017-01-30T22:00:25.015+05:30	9.63314E+12	44.07353	4	Incredible Bronze Par	Okuneva, Kutch and Monahan
17	Doohickey	2017-01-31T11:05:14.863+05:30	6.70464E+12	53.29072	4.3	Enormous Wool Car	Hilpert, Jacobs and Hauck
18	Gadget	2017-02-11T03:45:40.957+05:30	8.90936E+12	54.60205	4.4	Lightweight Paper Bo	Wilkinson-Gottlieb
19	Doohickey	2016-11-06T23:27:12.642+05:30	4.30772E+12	42.67117	4.2	Rustic Concrete Bott	Gibson, Turner and Douglas

Fig 5: Dataset: products.csv

CODE LINK

<https://github.com/meghanabt/HADOOP-SPARK-BASED-REAL-TIME-ONLINE-SALES-DATA-ANALYSIS>

STEPS TO RUN THE APPLICATION

Pre-requisites:

In an Ubuntu system, install Hadoop, Zookeeper and Kafka, Spark, Anaconda, Pycharm, Mysql , Cassandra and Superset.

Steps to Configure Kafka:

1. Go to the folder where the Pycharm Community application is located, then right click and select ‘open in terminal’. Type the command ‘./pycharm.sh’.
2. In the Pycharm application, open the Kafka Producer code which is configured to read the orders dataset and produce streams of data.
3. To configure the kafka consumer to consume and pass the stream data to spark, open a new terminal and change directory to kafka path and then type the command ‘/bin/db-project/kafka/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic orderstopic(*topics name*)’. Keep both ready.
4. Now, first run the kafka_producer to produce stream data and then run the consumer command on the terminal. You can see the streaming of data on pycharm platform which will reflect as it is on the terminal as it is consumed and given as output from the kafka consumer.
5. This stream data from the consumer is fed to the pyspark application via configured jar files .On running this application on pyspark platform the processed output is stored in the mysql database and raw data is stored in cassandra.
6. The processed output is later visualized on Apache superset application in which we initially login using the superset admin USERID and password.
7. Connect the mysql database from superset using the database name and mysql jdbc url.
8. After connecting to the database create charts using the charts tab and select the table in which the processed data is stored.
9. Based on the columns generated a query can be created to obtain the desired graphs.

* Steps to configure kafka, mysql and cassandra is mentioned in a separate document and is added in the git repository *

TEST RESULTS

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** kafka_producer
- File:** main.py
- Code:**

```

if __name__ == "__main__":
    print("Kafka Producer Application Started....")
    kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
                                         value_serializer=lambda x: dumps(x).encode('utf-8'))
    file_path = '/home/hadoop/Downloads/orders.csv'
    orders_nd_pf = pd.read_csv(file_path)
    if __name__ == "__main__":
        ID      Created At  Discount ... Tax      Total User ID
        0  2019-02-11T21:40:27.892+05:30   NaM ...  2.07  39.718145
        [1 rows x 9 columns]
        [ID: 1, 'Created At': '2019-02-11T21:40:27.892+05:30', 'Discount': nan, 'Product ID': 14, 'Quantity': 2, 'Subtotal': 37.448145389078345, 'Tax': 2.07, 'Total': 39.718145389078346, 'User ID': 1]
        Message to be sent: ('ID': 1, 'Created At': '2019-02-11T21:40:27.892+05:30', 'Discount': nan, 'Product ID': 14, 'Quantity': 2, 'Subtotal': 37.448145389078345, 'Tax': 2.07, 'Total': 39.718145389078346, 'User ID': 1)
        Message to be sent: ('ID': 2, 'Created At': '2018-05-15T08:04:04.580+05:30', 'Discount': nan, 'Product ID': 123, 'Quantity': 3, 'Subtotal': 110.9315648834248, 'Tax': 6.1, 'Total': 117.0376564082765, 'User ID': 1)
        Message to be sent: ('ID': 3, 'Created At': '2018-05-15T08:22:23.044+05:30', 'Discount': 0.416679208847959, 'Product ID': 105, 'Quantity': 2, 'Subtotal': 52.72352144261952, 'Tax': 2.9, 'Total': 55.2208881961842, 'User ID': 1)
        Message to be sent: ('ID': 4, 'Created At': '2018-08-22T16:30:42.392+05:30', 'Discount': 0.19816456655384, 'Product ID': 94, 'Quantity': 6, 'Subtotal': 109.2186456655384, 'Tax': 6.01, 'Total': 115.220751540632995, 'User ID': 1)
        Message to be sent: ('ID': 5, 'Created At': '2018-10-10T01:34:47.309+05:30', 'Discount': nan, 'Product ID': 152, 'Quantity': 5, 'Subtotal': 127.8919708933712, 'Tax': 7.63, 'Total': 134.9419293529647, 'User ID': 1)
        Message to be sent: ('ID': 6, 'Created At': '2018-11-06T01:36:59.154+05:30', 'Discount': nan, 'Product ID': 60, 'Quantity': 5, 'Subtotal': 29.0821475189149, 'Tax': 1.04, 'Total': 31.4416791533797817, 'User ID': 1)
        Message to be sent: ('ID': 7, 'Created At': '2018-11T21:22:02.212+05:30', 'Discount': 0.19816456655384, 'Product ID': 85, 'Quantity': 5, 'Subtotal': 95.7712857953437, 'Tax': 5.27, 'Total': 101.081051022136, 'User ID': 1)
        Message to be sent: ('ID': 8, 'Created At': '2019-02-17T02:42:03.939+05:30', 'Discount': 0.635952930280674, 'Product ID': 65, 'Quantity': 7, 'Subtotal': 58.2279572040804, 'Tax': 3.75, 'Total': 71.0440858804807, 'User ID': 1)
        Message to be sent: ('ID': 9, 'Created At': '2017-05-08T13:06:54.223+05:30', 'Discount': 3.594742155259162, 'Product ID': 184, 'Quantity': 3, 'Subtotal': 77.3982748679465, 'Tax': 4.26, 'Total': 81.6742695904106, 'User ID': 1)
        Message to be sent: ('ID': 10, 'Created At': '2020-01-17T01:44:23+05:30', 'Discount': nan, 'Product ID': 6, 'Quantity': 2, 'Subtotal': 77.43621056944382, 'Tax': 5.36, 'Total': 102.70958805103969, 'User ID': 1)
        Message to be sent: ('ID': 11, 'Created At': '2018-07-22T20:51:01.769+05:30', 'Discount': nan, 'Product ID': 76, 'Quantity': 6, 'Subtotal': 63.82421863164686, 'Tax': 3.51, 'Total': 67.08559713840213, 'User ID': 1)
        Message to be sent: ('ID': 12, 'Created At': '2018-06-20T22:21:13.273+05:30', 'Discount': 2.1773420356974987, 'Product ID': 70, 'Quantity': 7, 'Subtotal': 148.2290852552291, 'Tax': 10.19, 'Total': 158.4453828476374, 'User ID': 1)
        Message to be sent: ('ID': 13, 'Created At': '2019-04-08T01:06:43.779+05:30', 'Discount': 2.1773420356974987, 'Product ID': 70, 'Quantity': 7, 'Subtotal': 57.49908808899978, 'Tax': 3.95, 'Total': 61.4233953983595, 'User ID': 1)
        Message to be sent: ('ID': 14, 'Created At': '2017-05-25T20:58:37.137+05:30', 'Discount': nan, 'Product ID': 139, 'Quantity': 4, 'Subtotal': 51.18512212784679, 'Tax': 3.52, 'Total': 54.07137322414456, 'User ID': 1)
        Message to be sent: ('ID': 15, 'Created At': '2018-06-20T02:24:58.715+05:30', 'Discount': nan, 'Product ID': 116, 'Quantity': 5, 'Subtotal': 114.42485125407784, 'Tax': 7.87, 'Total': 122.38918146348137, 'User ID': 1)
        Message to be sent: ('ID': 16, 'Created At': '2017-12-17T11:28:51.031+05:30', 'Discount': 76.8289952153598, 'Product ID': 68, 'Quantity': 1, 'Subtotal': 76.8289952153598, 'Tax': 5.28, 'Total': 82.15945455750338, 'User ID': 1)
        Message to be sent: ('ID': 17, 'Created At': '2020-02-07T21:44:56.282+05:30', 'Discount': nan, 'Product ID': 48, 'Quantity': 1, 'Subtotal': 123.20884248534208, 'Tax': 8.47, 'Total': 131.6598789108083, 'User ID': 1)
        Message to be sent: ('ID': 18, 'Created At': '2018-04-10T22:40:36.000+05:30', 'Discount': nan, 'Product ID': 12, 'Quantity': 2, 'Subtotal': 110.44321863164686, 'Tax': 7.98, 'Total': 123.93528589902528, 'User ID': 1)
        Message to be sent: ('ID': 19, 'Created At': '2019-02-14T07:28:51:10+05:30', 'Discount': nan, 'Product ID': 136, 'Quantity': 1, 'Subtotal': 160.2042315715344, 'Tax': 7.25, 'Total': 112.53644509195752, 'User ID': 1)
        Message to be sent: ('ID': 20, 'Created At': '2018-04-21T04:18:01.863+05:30', 'Discount': 9.48087122288349, 'Product ID': 104, 'Quantity': 4, 'Subtotal': 47.59120861297272, 'Tax': 3.27, 'Total': 56.868595592608371, 'User ID': 1)
        Message to be sent: ('ID': 21, 'Created At': '2018-05-02T05:57:22.388+05:30', 'Discount': nan, 'Product ID': 94, 'Quantity': 5, 'Subtotal': 109.21864516655584, 'Tax': 7.51, 'Total': 116.6298277969602, 'User ID': 1)
        Message to be sent: ('ID': 22, 'Created At': '2019-02-12T21:32:01.553+05:30', 'Discount': 6.752658078439801, 'Product ID': 10, 'Quantity': 1, 'Subtotal': 47.679328102869, 'Tax': 1.38, 'Total': 50.45607101283766, 'User ID': 1)
        Message to be sent: ('ID': 23, 'Created At': '2019-06-07T11:35:15.096+05:30', 'Discount': nan, 'Product ID': 85, 'Quantity': 5, 'Subtotal': 54.9801734228952, 'Tax': 1.59, 'Total': 56.511988078793, 'User ID': 1)
        
```
- Logs:** Shows Kafka logs for the producer application.

Fig 6: Streaming of Data in Kafka Producer

The screenshot shows a Tmux session with the following details:

- Session:** hadoop@tufflx-vn:~/kafka_2.12-2.3.0\$
- Log Output:**

```

hadoop@tufflx-vn:~/kafka_2.12-2.3.0$ bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic orderstopic
[...]
[{"ID": 240, "Created At": "2018-01-24T03:27:21.346+05:30", "Discount": null, "Product ID": 62, "Quantity": 2, "Subtotal": 133.5202262591817, "Tax": 8.35, "Total": 140.47423188453183, "User ID": 40}, {"ID": 241, "Created At": "2017-10-05T03:52:29.698+05:30", "Discount": null, "Product ID": 175, "Quantity": 7, "Subtotal": 78.2163390223106, "Tax": 4.89, "Total": 82.2221602267029, "User ID": 40}, {"ID": 242, "Created At": "2018-04-10T01:34:47.309+05:30", "Discount": null, "Product ID": 105, "Quantity": 2, "Subtotal": 52.72352144261952, "Tax": 2.9, "Total": 55.2208881961842, "User ID": 40}, {"ID": 243, "Created At": "2019-10-16T10:09:52.916+05:30", "Discount": null, "Product ID": 11, "Quantity": 4, "Subtotal": 55.024851254077866, "Tax": 3.44, "Total": 58.67795884847735, "User ID": 40}, {"ID": 244, "Created At": "2018-08-30T06:58:11.546+05:30", "Discount": null, "Product ID": 152, "Quantity": 7, "Subtotal": 48.0598729908663, "Tax": 3.06, "Total": 51.9723946317932, "User ID": 40}, {"ID": 245, "Created At": "2019-10-21T10:56:04.968+05:30", "Discount": null, "Product ID": 138, "Quantity": 7, "Subtotal": 75.0257369311537, "Tax": 4.69, "Total": 80.08323868731879, "User ID": 40}, {"ID": 246, "Created At": "2017-12-13T19:28:29.409+05:30", "Discount": null, "Product ID": 75, "Quantity": 3, "Subtotal": 83.875175821301558, "Tax": 5.24, "Total": 88.37700246121238, "User ID": 40}, {"ID": 247, "Created At": "2019-08-04T14:14:32.044+05:30", "Discount": null, "Product ID": 88, "Quantity": 9, "Subtotal": 105.41292031262557, "Tax": 6.59, "Total": 112.96680124862464, "User ID": 40}, {"ID": 248, "Created At": "2018-11-05T05:47:19.047+05:30", "Discount": null, "Product ID": 18, "Quantity": 4, "Subtotal": 81.96307121097293, "Tax": 5.12, "Total": 86.8045940773892, "User ID": 40}, {"ID": 249, "Created At": "2019-10-10T20:34:21.626+05:30", "Discount": null, "Product ID": 97, "Quantity": 5, "Subtotal": 50.094887884945365, "Tax": 3.13, "Total": 53.33409197449717, "User ID": 40}, {"ID": 250, "Created At": "2018-08-01T14:41:21.062+05:30", "Discount": null, "Product ID": 121, "Quantity": 4, "Subtotal": 80.443282420917, "Tax": 5.01, "Total": 85.45336446792765, "User ID": 40}, {"ID": 251, "Created At": "2018-07-25T06:15:19.704+05:30", "Discount": null, "Product ID": 121, "Quantity": 4, "Subtotal": 40.485282420917, "Tax": 2.53, "Total": 43.04404242063459, "User ID": 40}, {"ID": 252, "Created At": "2016-11-01T15:43:42.664+05:30", "Discount": null, "Product ID": 21, "Quantity": 3, "Subtotal": 40.38334400304544, "Tax": 2.52, "Total": 42.05846557076474, "User ID": 40}, {"ID": 253, "Created At": "2018-01-23T08:10:09.206+05:30", "Discount": null, "Product ID": 197, "Quantity": 2, "Subtotal": 70.14610686710069, "Tax": 4.38, "Total": 75.56411182861285, "User ID": 40}, {"ID": 254, "Created At": "2019-08-10T14:41:56.799+05:30", "Discount": null, "Product ID": 129, "Quantity": 5, "Subtotal": 56.13638839064484, "Tax": 3.2, "Total": 61.53220331179552, "User ID": 40}, {"ID": 255, "Created At": "2018-08-01T14:41:56.000+05:30", "Discount": null, "Product ID": 96, "Quantity": 6, "Subtotal": 56.13638839064484, "Tax": 3.2, "Total": 61.53220331179552, "User ID": 40}, {"ID": 256, "Created At": "2017-08-01T04:50:43.603+05:30", "Discount": null, "Product ID": 77, "Quantity": 5, "Subtotal": 67.34461152277315, "Tax": 4.21, "Total": 70.97951199345664, "User ID": 40}, {"ID": 257, "Created At": "2018-02-06T08:34:07.323+05:30", "Discount": null, "Product ID": 80, "Quantity": 2, "Subtotal": 54.91325681036414, "Tax": 3.43, "Total": 58.3499049654743, "User ID": 40}, {"ID": 258, "Created At": "2017-09-08T17:20:55.086+05:30", "Discount": null, "Product ID": 191, "Quantity": 3, "Subtotal": 85.722790131719552, "Tax": 5.36, "Total": 90.852033118120, "User ID": 40}, {"ID": 259, "Created At": "2017-04-27T16:10:11.002+05:30", "Discount": null, "Product ID": 71, "Quantity": 1, "Subtotal": 76.936184284695953, "Tax": 3.23, "Total": 80.1672933305083, "User ID": 40}, {"ID": 260, "Created At": "2017-09-04T11:13:31.013+05:30", "Discount": null, "Product ID": 65, "Quantity": 1, "Subtotal": 65.90981855960731, "Tax": 3.09, "Total": 68.884033991761, "User ID": 40}, {"ID": 261, "Created At": "2019-08-26T21:10:14.407+05:30", "Discount": null, "Product ID": 115, "Quantity": 3, "Subtotal": 83.03740654322647, "Tax": 4.87, "Total": 83.03740654322647, "User ID": 40}, {"ID": 262, "Created At": "2019-03-06T01:08:17.786+05:30", "Discount": null, "Product ID": 62, "Quantity": 1, "Subtotal": 133.5202262591817, "Tax": 8.35, "Total": 143.857021316444028, "User ID": 40}, {"ID": 263, "Created At": "2018-08-10T17:09:02.659+05:30", "Discount": null, "Product ID": 104, "Quantity": 1, "Subtotal": 51.1672933305083, "Tax": 3.11, "Total": 54.2777821028671, "User ID": 40}, {"ID": 264, "Created At": "2017-01-07T09:08:02.659+05:30", "Discount": null, "Product ID": 164, "Quantity": 1, "Subtotal": 61.9785918725397, "Tax": 2.48, "Total": 65.2575918725397, "User ID": 40}, {"ID": 265, "Created At": "2017-08-16T03:33:03.727+05:30", "Discount": null, "Product ID": 146, "Quantity": 7, "Subtotal": 84.8315141444699, "Tax": 3.36, "Total": 86.45472386595857, "User ID": 40}, {"ID": 266, "Created At": "2019-07-17T15:11:09.590+05:30", "Discount": null, "Product ID": 122, "Quantity": 2, "Subtotal": 32.13677946634949, "Tax": 1.29, "Total": 33.278920529573234, "User ID": 40}, {"ID": 267, "Created At": "2018-10-26T12:46:48.799+05:30", "Discount": null, "Product ID": 103, "Quantity": 4, "Subtotal": 79.93608046792765, "Tax": 3.3, "Total": 83.07479106553187, "User ID": 40}, {"ID": 268, "Created At": "2018-08-10T14:41:56.799+05:30", "Discount": null, "Product ID": 100, "Quantity": 1, "Subtotal": 71.6297239950959, "Tax": 3.19, "Total": 74.823184284695953, "User ID": 40}, {"ID": 269, "Created At": "2019-04-11T19:13:31.013+05:30", "Discount": null, "Product ID": 159, "Quantity": 2, "Subtotal": 35.53017445737361, "Tax": 1.42, "Total": 36.90981855960731, "User ID": 40}, {"ID": 270, "Created At": "2019-02-26T20:42:08.386+05:30", "Discount": null, "Product ID": 119, "Quantity": 1, "Subtotal": 41.43814529652384, "Tax": 1.74, "Total": 44.7936118699993, "User ID": 40}, {"ID": 271, "Created At": "2019-12-15T22:10:26.427+05:30", "Discount": null, "Product ID": 38, "Quantity": 2, "Subtotal": 64.17448218671084, "Tax": 2.57, "Total": 65.97681854234172, "User ID": 40}, {"ID": 272, "Created At": "2018-06-15T16:10:39.973+05:30", "Discount": null, "Product ID": 101, "Quantity": 7, "Subtotal": 148.1672932165971, "Tax": 5.93, "Total": 154.1322003537157, "User ID": 40}, {"ID": 273, "Created At": "2018-08-01T14:41:56.799+05:30", "Discount": null, "Product ID": 132, "Quantity": 1, "Subtotal": 71.6297239950959, "Tax": 3.19, "Total": 74.823184284695953, "User ID": 40}, {"ID": 274, "Created At": "2019-05-31T00:40:37.002+05:30", "Discount": null, "Product ID": 149, "Quantity": 6, "Subtotal": 69.15415037577924, "Tax": 2.77, "Total": 71.3472363103508, "User ID": 40}, {"ID": 275, "Created At": "2018-01-30T17:39:43.668+05:30", "Discount": null, "Product ID": 17, "Quantity": 2, "Subtotal": 79.93608046792765, "Tax": 3.2, "Total": 83.4196335859973, "User ID": 40}, {"ID": 276, "Created At": "2018-04-26T23:13:59.515+05:30", "Discount": null, "Product ID": 38, "Quantity": 3, "Subtotal": 66.60937238339370, "Tax": 2.64, "Total": 67.87614540095575, "User ID": 40}, {"ID": 277, "Created At": "2018-03-23T18:23:24.986+05:30", "Discount": null, "Product ID": 79, "Quantity": 1, "Subtotal": 41.616917284150765, "Tax": 1.66, "Total": 43.3348537337784, "User ID": 40}, {"ID": 278, "Created At": "2019-09-14T19:35:23.076+05:30", "Discount": null, "Product ID": 121, "Quantity": 4, "Subtotal": 26.09352219205405, "Tax": 1.08, "Total": 28.4848918518004533, "User ID": 40}, {"ID": 279, "Created At": "2017-08-10T08:29:32.145+05:30", "Discount": null, "Product ID": 60, "Quantity": 7, "Subtotal": 19.86809834527266, "Tax": 1.19, "Total": 20.897073621065454, "User ID": 40}, {"ID": 280, "Created At": "2018-10-17T19:07:02.234+05:30", "Discount": null, "Product ID": 8, "Subtotal": 98.83823053099358, "Tax": 5.93, "Total": 105.64430116705226, "User ID": 40}, {"ID": 281, "Created At": "2018-08-10T14:41:56.799+05:30", "Discount": null, "Product ID": 106, "Quantity": 1, "Subtotal": 65.97547151229557, "Tax": 3.19, "Total": 67.94748151229557, "User ID": 40}, {"ID": 282, "Created At": "2017-01-08T15:11:09.444+05:30", "Discount": null, "Product ID": 6, "Subtotal": 64.95747510229557, "Tax": 3.91, "Total": 69.0927782774265655, "User ID": 40}, {"ID": 283, "Created At": "2018-10-03T09:10:25.062+05:30", "Discount": null, "Product ID": 39, "Quantity": 4, "Subtotal": 114.5815818628346, "Tax": 6.87, "Total": 122.8808053279916, "User ID": 40}, {"ID": 284, "Created At": "2017-11-09T21:39:14.743+05:30", "Discount": null, "Product ID": 200, "Quantity": 3, "Subtotal": 48.00263087861605, "Tax": 2.93, "Total": 52.1398766655059, "User ID": 40}, {"ID": 285, "Created At": "2018-05-13T13:08:31.727+05:30", "Discount": null, "Product ID": 118, "Quantity": 5, "Subtotal": 57.627613096978735, "Tax": 3.46, "Total": 66.5069661866073, "User ID": 40}, {"ID": 286, "Created At": "2018-01-06T22:36:17.244+05:30", "Discount": null, "Product ID": 115, "Quantity": 2, "Subtotal": 7.959897757965706, "Tax": 4.67, "Total": 83.02842687886262, "User ID": 40}
        
```

Fig 7: Streaming of Data in Kafka Consumer

```
Printing Schema of orders_df:  
root  
|-- key: binary (nullable = true)  
|-- value: binary (nullable = true)  
|-- topic: string (nullable = true)  
|-- partition: integer (nullable = true)  
|-- offset: long (nullable = true)  
|-- timestamp: timestamp (nullable = true)  
|-- timestampType: integer (nullable = true)
```

Fig 8: Schema of data frames generated, of DataStream of orders received from Kafka-consumer

```
root  
|-- ID: integer (nullable = true)  
|-- Name: string (nullable = true)  
|-- Address: string (nullable = true)  
|-- Birth Date: timestamp (nullable = true)  
|-- City: string (nullable = true)  
|-- Created At: timestamp (nullable = true)  
|-- Email: string (nullable = true)  
|-- Latitude: double (nullable = true)  
|-- Longitude: double (nullable = true)  
|-- Password: string (nullable = true)  
|-- Source: string (nullable = true)  
|-- State: string (nullable = true)  
|-- Zip: integer (nullable = true)
```

Fig 9: Schema of data frames generated, of static customer data from the dataset

```

Printing Schema of Order_df4:
root
|-- order_id: string (nullable = true)
|-- created_at: string (nullable = true)
|-- discount: string (nullable = true)
|-- product_id: string (nullable = true)
|-- quantity: string (nullable = true)
|-- subtotal: string (nullable = true)
|-- tax: string (nullable = true)
|-- total: string (nullable = true)
|-- customer_id: string (nullable = true)
|-- timestamp: timestamp (nullable = true)
|-- ID: integer (nullable = true)
|-- Name: string (nullable = true)
|-- Address: string (nullable = true)
|-- Birth Date: timestamp (nullable = true)
|-- City: string (nullable = true)
|-- Created At: timestamp (nullable = true)
|-- Email: string (nullable = true)
|-- Latitude: double (nullable = true)
|-- Longitude: double (nullable = true)
|-- Password: string (nullable = true)
|-- Source: string (nullable = true)
|-- State: string (nullable = true)
|-- Zip: integer (nullable = true)

Printing Schema of orders_df5:
root
|-- Source: string (nullable = true)
|-- State: string (nullable = true)
|-- Amount_Paid: double (nullable = true)

```

Fig 10: Schema of data frames generated, of forked data of customers and orders used for data analysis

Organic	NM	293.81937092516944	2019-10-20 14:02:45	412
Facebook	MS	3319.8868346644153	2019-10-20 14:02:45	412
Organic	MN	3978.229446433442	2019-10-20 14:02:45	412
Facebook	CA	4684.518476233977	2019-10-20 14:03:00	413
Facebook	MS	4407.1372272317	2019-10-20 14:03:00	413
Facebook	CA	5032.269939224234	2019-10-20 14:03:15	414
Google	NE	712.4077658179535	2019-10-20 14:03:15	414
Organic	ND	2905.9799464917774	2019-10-20 14:03:15	414
Twitter	IL	3348.821727410377	2019-10-20 14:03:30	415
Organic	ND	3674.91247055596	2019-10-20 14:03:30	415
Twitter	IL	3568.417520548799	2019-10-20 14:03:45	416
Twitter	VT	1037.8921538200907	2019-10-20 14:03:45	416
Organic	MT	9385.43902457641	2019-10-20 14:04:00	417
Twitter	VT	1464.726981042852	2019-10-20 14:04:00	417
Affiliate	TX	8335.897361481708	2019-10-20 14:04:15	418
Organic	MT	9496.278196043422	2019-10-20 14:04:15	418
Twitter	CA	7506.6223139450385	2019-10-20 14:04:15	418
Google	MI	4442.819923353086	2019-10-20 14:04:43	419
Twitter	CA	7593.102021168229	2019-10-20 14:04:43	419
Affiliate	WA	1129.1928465841045	2019-10-20 14:04:59	420
Twitter	OH	1608.691926708404	2019-10-20 14:04:59	420
Organic	WI	5012.404838772943	2019-10-20 14:04:59	420
Affiliate	WA	1345.4541751066963	2019-10-20 14:05:08	421
Affiliate	MO	5828.3170450380285	2019-10-20 14:05:08	421
Affiliate	MO	6831.34902858962	2019-10-20 14:05:22	422
Google	TX	8200.504406063952	2019-10-20 14:05:22	422
Google	TX	9082.447178734588	2019-10-20 14:05:31	423
Organic	NC	2558.6695328576693	2019-10-20 14:05:52	424
Google	TX	9946.049614318941	2019-10-20 14:05:52	424

Fig 11: Processed data stored in Mysql Database

Fig 12: Raw input data stored in Cassandra Database

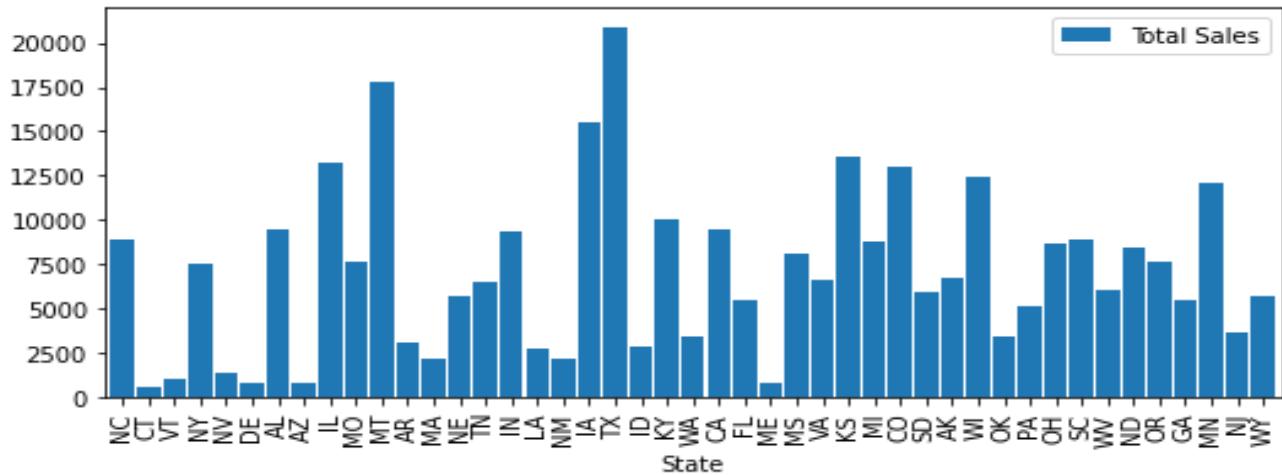


Fig 13: Total Sales in different states with Source Organic

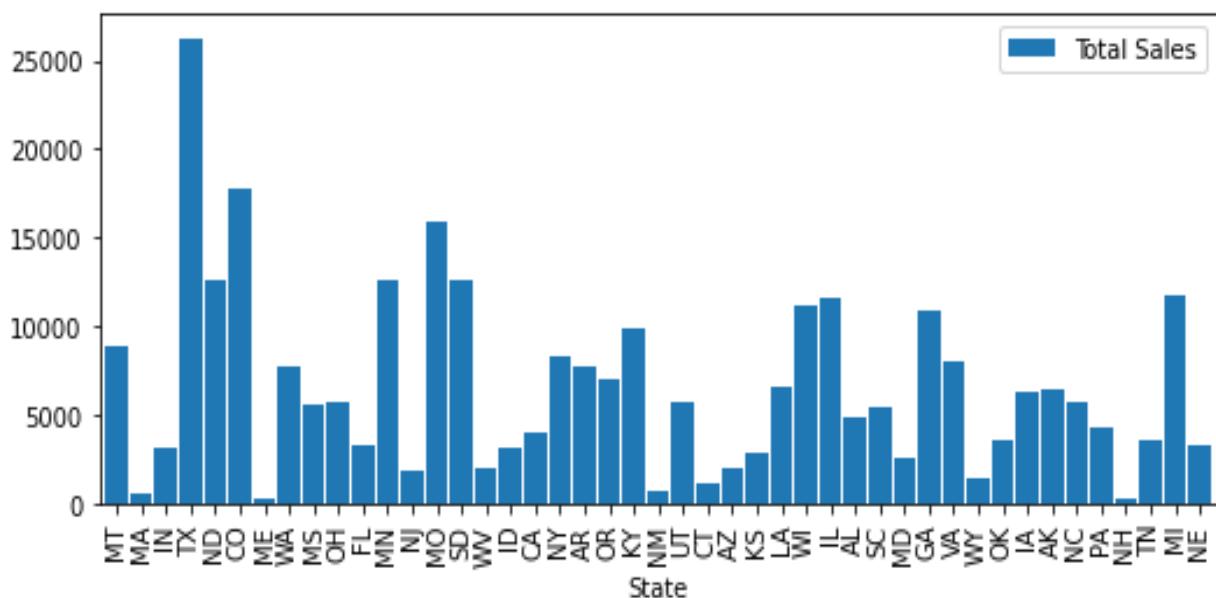


Fig 14: Total Sales in different states with Source Affiliate

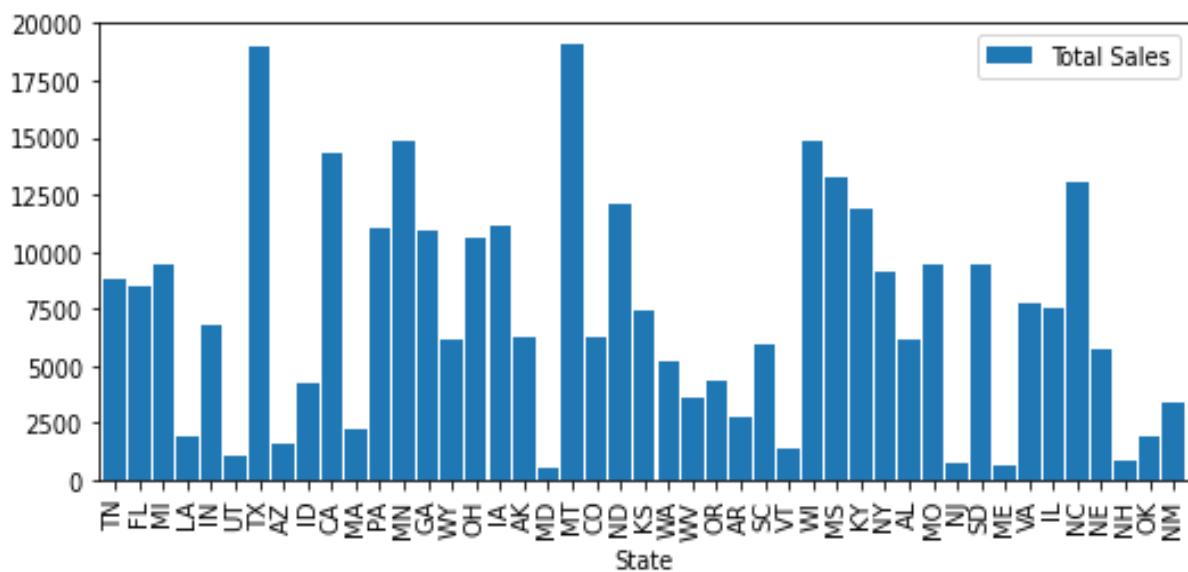


Fig 15: Total Sales in different states with Source Facebook

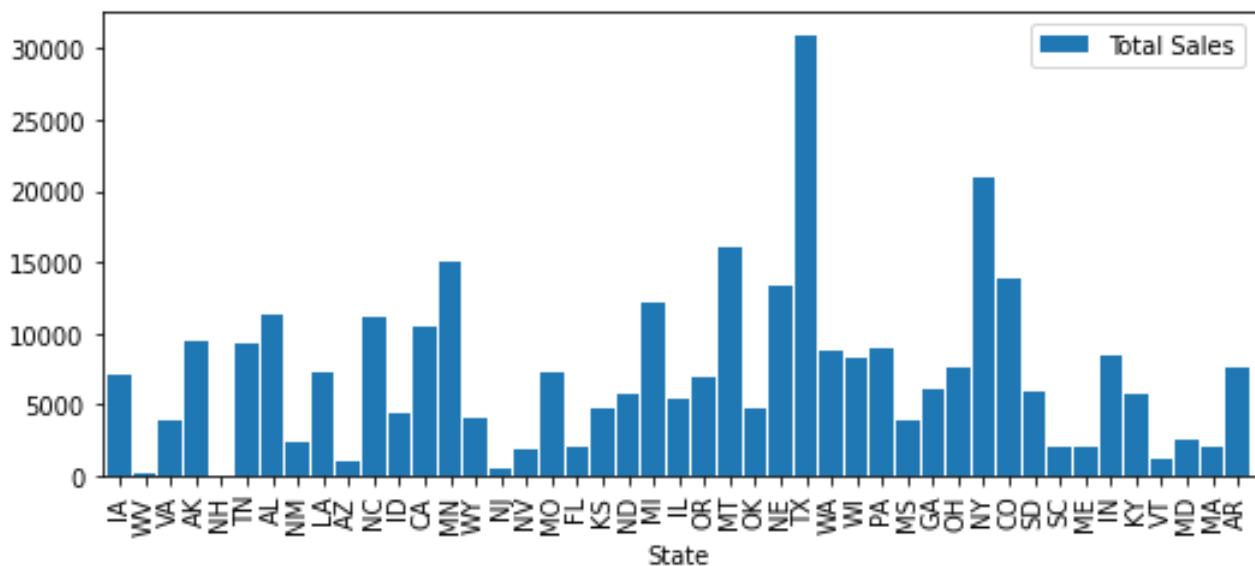


Fig 16: Total Sales in different states with Source Google

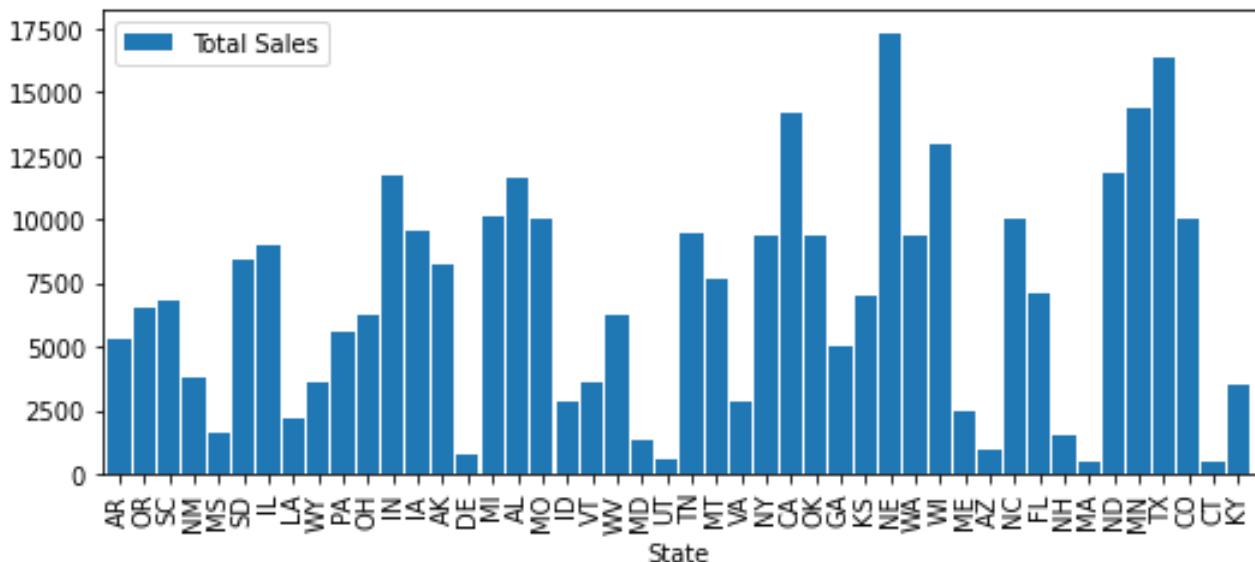


Fig 17: Total Sales in different states with Source Twitter

FUTURE SCOPE

Currently, we have used Cassandra NoSQL data store to store raw data.

As cloud storage becomes more scalable and offers more security, we will be able to employ S3 object storage to develop machine learning models and further complex analytics.

CONCLUSION

Analyzing the total sales in each state generated by each source will help the e-commerce company choose which marketing platform will maximize its product sales.

Likewise, for the marketing platform, the analysis will aid in increasing their business strategy.