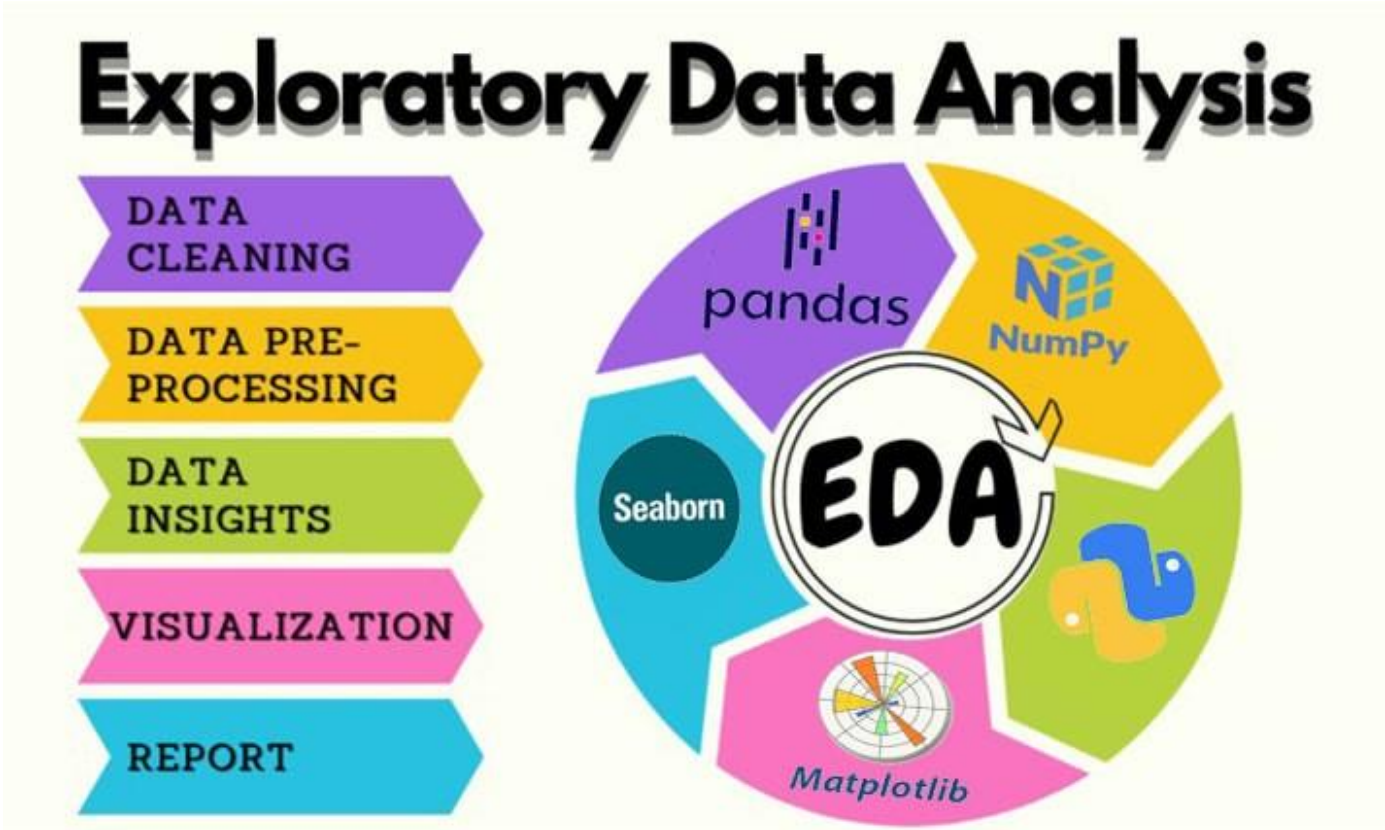
Marketing Analytics Business Case

Harsh Wardhan Singh



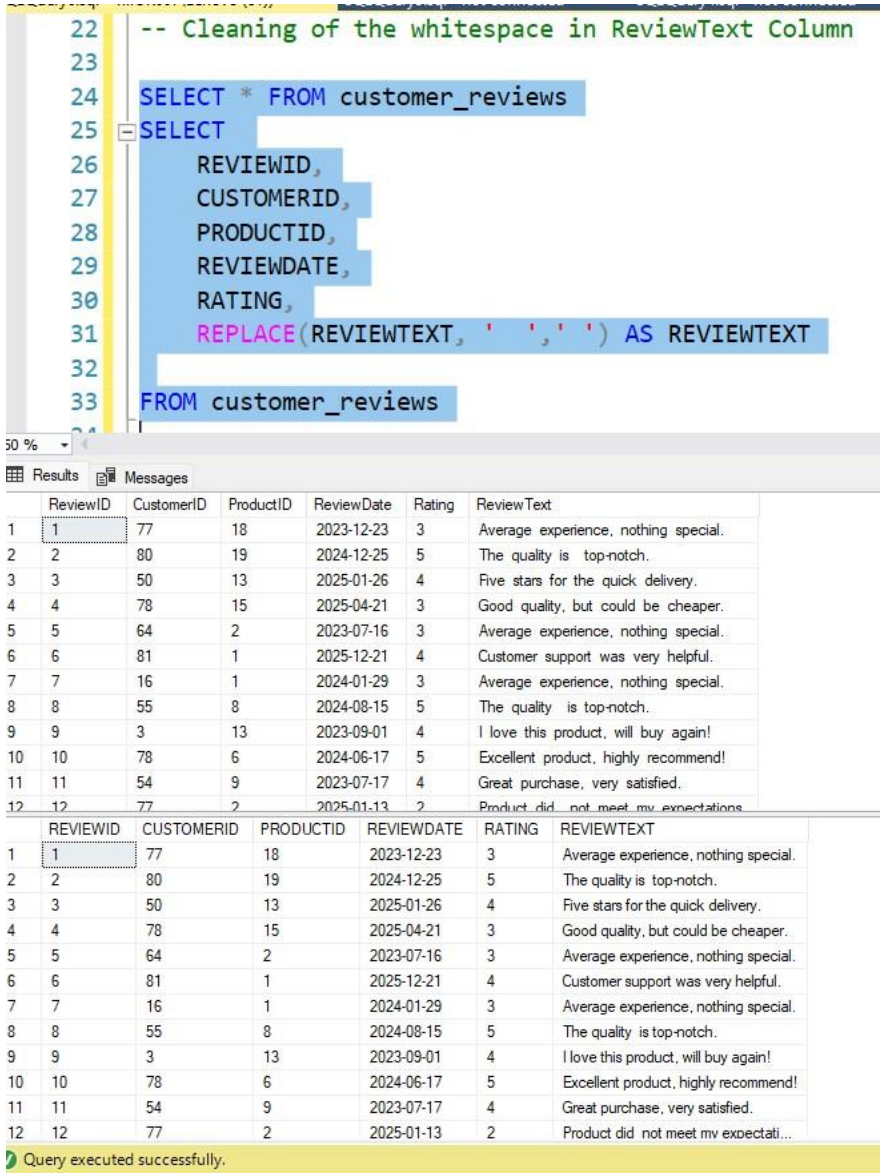
Day - 2

Data Cleaning

1. Cleaning the white space .

We also used the REPLACE() function to clean the review text by removing unwanted double spaces, which helps improve the accuracy of text-based analysis.

This clean and structured data was used in further processing and sentiment scoring..



Data Cleaning

1. Standardizing the ContentType field (fixing spelling from "Socailmedia" to "Social Media").
2. Splitting combined views and clicks into separate VIEWS and CLICKS columns.
3. Formatting the EngagementDate into a readable dd.MM.yyyy format.
4. Filtering out entries where ContentType is "Newsletter" to focus on relevant channels.

```sql
-- Cleaning of the engagement_data table

SELECT * FROM engagement_data

SELECT
    ENGAGEMENTID,
    CONTENTID,
    UPPER(REPLACE(CONTENTTYPE,'Socailmedia','Social Media')) as ContentType,
    LIKES,

    CAMPAIGNID,
    PRODUCTID,
    LEFT(VIEWSCLICKSCOMBINED, CHARINDEX('-', VIEWSCLICKSCOMBINED)-1) AS VIEWS,
    RIGHT(ViewsClicksCombined, LEN(ViewsClicksCombined) - CHARINDEX('-', ViewsClicksCombined)) AS Cli
    FORMAT(CONVERT(DATE,ENGAGEMENTDATE),'dd.MM.yyyy') as EngagementDate

from engagement_data

Where ContentType!='Newsletter'
```

Messages

GEMENTID	CONTENTID	ContentType	LIKES	CAMPAIGNID	PRODUCTID	VIEWS	Clicks	EngagementDate
	39	BLOG	190	1	9	1883	671	30.08.2023
	48	BLOG	114	18	20	5280	532	28.03.2023
	16	VIDEO	32	7	14	1905	204	08.12.2023
	43	VIDEO	17	19	20	2766	257	21.01.2025
	32	SOCIALMEDIA	648	18	19	8237	1641	18.06.2023
	33	SOCIALMEDIA	1	12	2	750	34	01.10.2025
	47	BLOG	1	17	6	891	35	31.03.2025
	48	BLOG	123	13	16	5571	1527	19.03.2024
	4	BLOG	25	15	15	4279	297	03.12.2023
	38	SOCIALMEDIA	29	19	6	4297	234	26.05.2024
	29	SOCIALMEDIA	396	3	10	7877	2037	15.06.2023
	14	VIDEO	12	12	18	1796	136	04.02.2025
	2	SOCIALMEDIA	317	16	2	6837	2251	26.08.2023
	16	SOCIALMEDIA	0	6	19	36	1	20.10.2025

ecuted successfully. THINKBOOK007\MSSQL

Data Cleaning

1. This query identifies duplicate entries in the customer_journey table based on a combination of key fields: CustomerID, ProductID, VisitDate, Stage, and Action.
2. It uses the **ROW_NUMBER()** function to assign a sequence number within each duplicate group.
3. Any record with **ROW_NUM > 1** is flagged as a duplicate.
4. The result helps in cleaning redundant data and improving analysis accuracy.
5. Duplicates are sorted by JourneyID for easy inspection.

```
60  -- Only Duplicates
61  SELECT *
62      FROM(
63          SELECT *,
64           ROW_NUMBER() OVER(
65              PARTITION BY CUSTOMERID, ProductID, VisitDate, Stage, Action
66              ORDER BY JOURNEYID
67
68          ) AS ROW_NUM
69
70          FROM customer_journey
71
72      ) AS SUB
73
74      WHERE ROW_NUM >1
75      ORDER BY JourneyID
```

Results | Messages

JourneyID	CustomerID	ProductID	VisitDate	Stage	Action	Duration	ROW_NUM
23	58	1	2023-07-05	Homepage	View	48	2
97	92	19	2025-05-26	Homepage	View	203	2
151	79	20	2023-11-04	Checkout	Purchase	298	2
203	44	11	2024-12-28	Homepage	View	230	2
404	30	13	2023-04-22	Homepage	View	112	2
443	98	5	2023-09-04	Checkout	Drop-off	157	2
494	29	6	2024-10-27	Checkout	Drop-off	157	2
520	42	16	2025-07-13	Checkout	Drop-off	157	2
664	41	18	2025-01-30	ProductPage	View	59	2
676	67	8	2023-01-15	ProductPage	View	91	2
677	83	8	2025-12-23	Homepage	Click	262	2
687	90	6	2023-09-24	Homepage	View	10	2
703	60	16	2023-08-30	Homepage	View	23	2
746	32	20	2024-08-25	Homepage	View	127	2
841	27	6	2025-10-18	ProductPage	View	244	2
853	69	12	2024-11-02	ProductPage	View	91	2

Query executed successfully.

Data Cleaning

1. This query updates the customer_journey table by filling in missing (NULL) values in the Duration column.
2. It calculates the average(AVG) Duration from all non-null records.
3. Then replaces all NULL durations with this average value(AVG) using an **UPDATE** statement.
4. The **ROUND()** function ensures the average is stored as a whole number.
5. This step helps maintain data completeness and improves the reliability of time-based analysis.

```
77  -- This will replace the Null values in durations with their Avg Value
78
79  UPDATE customer_journey
80  SET Duration = (
81      SELECT ROUND(AVG(Duration * 1.0), 0)
82      FROM customer_journey
83      WHERE Duration IS NOT NULL
84  )
85  WHERE Duration IS NULL;
86  SELECT * FROM customer_journey
87
88
```

Results | Messages

	JoumeyID	CustomerID	ProductID	VisitDate	Stage	Action	Duration
1	1	64	18	2024-06-10	Checkout	Drop-off	157
2	2	94	11	2025-07-09	Checkout	Drop-off	157
3	3	34	8	2024-06-14	ProductPage	View	235
4	4	33	18	2025-05-28	Checkout	Drop-off	157
5	5	91	10	2023-02-11	Homepage	Click	156
6	6	54	11	2025-12-19	Homepage	View	264
7	7	80	4	2023-08-25	Homepage	View	298
8	8	99	10	2025-07-03	ProductPage	View	287
9	9	31	4	2025-06-13	ProductPage	View	278
10	10	44	16	2025-04-23	ProductPage	View	30
11	11	23	12	2023-08-22	Checkout	Drop-off	157
12	12	97	12	2023-07-28	ProductPage	View	227
13	13	83	5	2024-06-20	Homepage	View	223
14	14	14	13	2025-11-01	ProductPage	View	13
15	15	9	1	2025-11-04	ProductPage	View	292
16	16	14	16	2024-07-17	Homepage	View	46
17	17	67	2	2025-12-11	Homepage	View	185
18	18	70	15	2025-10-13	Homepage	View	191
19	19	9	1	2024-08-15	Homepage	View	187
20	20	44	18	2024-02-19	productpage	View	154
21	21	58	20	2024-03-07	Checkout	Drop-off	157
22	22	12	1	2025-11-19	Homepage	Click	20

Query executed successfully.

Thank You!