RETAIL SHOP SALES PROJECT

In this case study, we were tasked with conducting an end-to-end project (ETL) where we built an interactive dashboard showcasing key performance indicators (KPIs) to address a client or stakeholder business inquiries and provide data-driven insights.

For my project, I employed the ETL method by extracting the 'retail shop sales' dataset from Kaggle. I performed data transformations in MySQL to create specific datasets that address the client's business questions. Subsequently, I loaded these datasets into Power BI to develop a dashboard. This visual tool was designed to explain the KPIs and provide insights to the client, helping them make informed business decisions.

PROBLEM STATEMENT:

KPI's REQUIREMENTS BY CLIENT/STAKEHOLDER:

1. TOTAL SALES ANALYSIS
   - Calculate the total sales for each respective month and the difference in total sales (as percentage) between current and previous month
2. TOTAL ORDER ANALYSIS
   - Calculate the total order for each respective month and the difference in total orders (as percentage) between current and previous month
3. TOTAL QUANTITY SOLD ANALYSIS
   - Calculate the total quantity sold for each respective month and the difference in total quantity sold (as percentage) between current and previous month

CHARTS REQUIREMENTS:

1. CALENDAR HEAT MAP
   - A heat map that adjusts for the chosen month which is colour-coded that shows a darker shade for higher sales, lighter shade for lower sales
2. SALES ANALYSIS BY WEEKDAYS AND WEEKENDS
   - Separates sales data into weekdays and weekends to see patterns
3. SALES ANALYSIS BY AGE PER MONTH WITH AVERAGE LINE
   - Calculates the average sales by age for the chosen month which is also colour-coded, light orange for sales above average, light blue for sales below average
4. SALES BY GENDER
   - Visualises the sales for each gender for that chosen month and shows the difference in sales as percentage between current and previous month

5. SALES BY PRODUCT CATEGORY
    - Visualises the sales for each product category for that chosen month and shows the difference in sales as percentage between current and previous month
6. DAILY SALES ANALYSIS WITH AVERAGE LINE
    - Calculates the daily average sales for the chosen month which is also colour-coded, light orange for sales above average, light blue for sales below average

- Note: the dashboard should be filtered by month

SQL QUERIES:

UPDATING DATE (transaction_date) COLUMN TO PROPER DATE FORMAT AND RENAMING IT TO sale_date FOR EASY QUERYING

```
UPDATE retail_shop_sales

SET  transaction_date = STR_TO_DATE(transaction_date, '%d/%m/%Y');

ALTER TABLE retail_shop_sales

MODIFY COLUMN transaction_date DATE;

ALTER TABLE retail_shop_sales

CHANGE COLUMN transaction_date sale_date DATE;
```

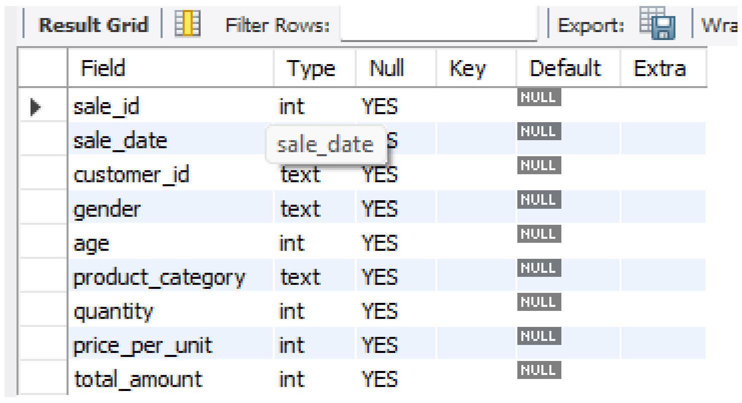
ALTERING trans_id to sale_id FOR EASY QUERYING

```
ALTER TABLE retail_shop_sales

CHANGE COLUMN trans_id sale_id INT;
```

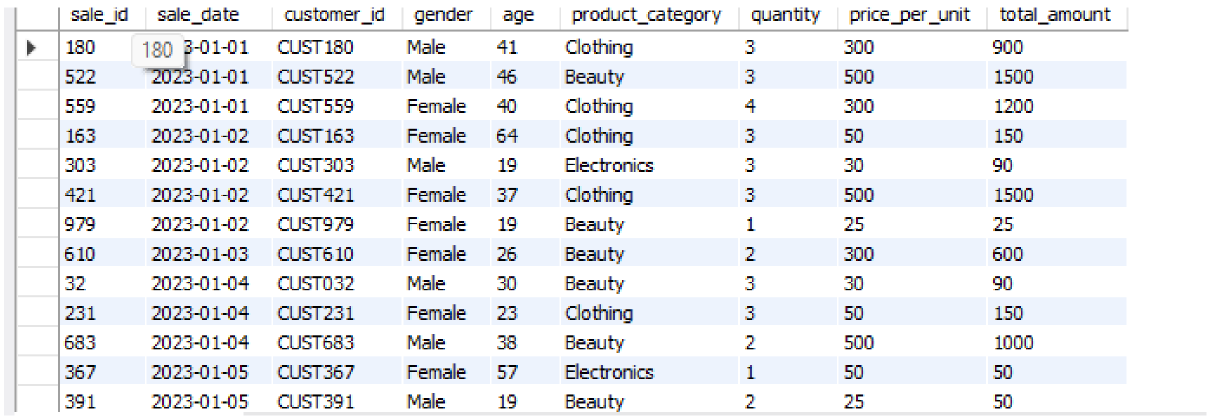
CHECKING IF THE QUERIES ARE SUCCESFUL

```
DESCRIBE retail_shop_sales;
```


Result Grid | Filter Rows: | Export: | Wra

Field	Type	Null	Key	Default	Extra
sale_id	int	YES		NULL	
sale_date	sale_date	S		NULL	
customer_id	text	YES		NULL	
gender	text	YES		NULL	
age	int	YES		NULL	
product_category	text	YES		NULL	
quantity	int	YES		NULL	
price_per_unit	int	YES		NULL	
total_amount	int	YES		NULL	

SELECT * FROM retail_shop_sales;



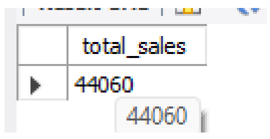
sale_id	sale_date	customer_id	gender	age	product_category	quantity	price_per_unit	total_amount
180	180 3-01-01	CUST180	Male	41	Clothing	3	300	900
522	2023-01-01	CUST522	Male	46	Beauty	3	500	1500
559	2023-01-01	CUST559	Female	40	Clothing	4	300	1200
163	2023-01-02	CUST163	Female	64	Clothing	3	50	150
303	2023-01-02	CUST303	Male	19	Electronics	3	30	90
421	2023-01-02	CUST421	Female	37	Clothing	3	500	1500
979	2023-01-02	CUST979	Female	19	Beauty	1	25	25
610	2023-01-03	CUST610	Female	26	Beauty	2	300	600
32	2023-01-04	CUST032	Male	30	Beauty	3	30	90
231	2023-01-04	CUST231	Female	23	Clothing	3	50	150
683	2023-01-04	CUST683	Male	38	Beauty	2	500	1000
367	2023-01-05	CUST367	Female	57	Electronics	1	50	50
391	2023-01-05	CUST391	Male	19	Beauty	2	25	50

-- TOTAL SALES FOR MONTH SELECTED

SELECT ROUND(SUM(total_amount)) AS total_sales

FROM retail_shop_sales

WHERE MONTH(sale_date) = 2; -- Februrary



total_sales
44060

44060

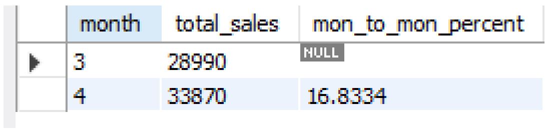
-- THE DIFFERENCE OF TOTAL SALES FROM CURRENT MONTH TO PREVIOUS MONTH SHOWN AS A PERCENTAGE WITH THE USE OF LAG()

-- note that january 2023 will return null as it has no previous month to be comapred to

SELECT

    MONTH(sale_date) AS month,

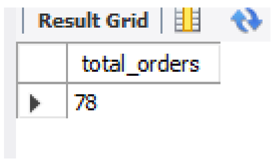
```
        ROUND(SUM(total_amount)) AS total_sales,
        (SUM(total_amount) - LAG(SUM(total_amount),1)
        OVER (ORDER BY MONTH(sale_date))) / LAG(SUM(total_amount),1)
        OVER (ORDER BY MONTH(sale_date)) * 100 AS mon_to_mon_percent
FROM retail_shop_sales
WHERE MONTH(sale_date) IN (3,4) – March and April
GROUP BY MONTH(sale_date)
ORDER BY MONTH(sale_date);
```



```
month  total_sales  mon_to_mon_percent
3      28990        NULL
4      33870        16.8334
```

-- TOTAL ORDERS FOR MONTH SELECTED

```
SELECT COUNT(sale_id) AS total_orders
FROM retail_shop_sales
WHERE MONTH(sale_date) = 1;
```



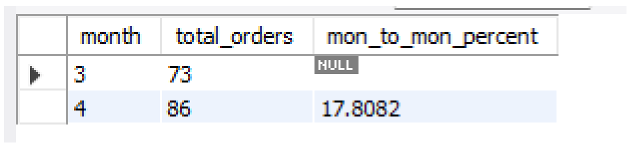
```
total_orders
78
```

-- THE DIFFERENCE OF TOTAL ORDERS FROM CURRENT MONTH TO PREVIOUS MONTH SHOWN AS A PERCENTAGE WITH THE USE OF LAG()

```
SELECT
        MONTH(sale_date) AS month,
        ROUND(COUNT(sale_id)) as total_orders,
        (COUNT(sale_id) - lag(COUNT(sale_id),1)
         OVER (ORDER BY MONTH(sale_date))) / LAG(COUNT(sale_id),1)
        OVER (ORDER BY MONTH(sale_date)) * 100 as mon_to_mon_percent
FROM retail_shop_sales
WHERE MONTH(sale_date) IN (3,4)
```

GROUP BY MONTH(sale_date)

ORDER BY MONTH(sale_date);



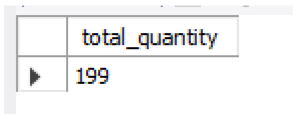
month	total_orders	mon_to_mon_percent
3	73	NULL
4	86	17.8082

-- TOTAL QUANTITY FOR MONTH SELECTED

SELECT SUM(quantity) as total_quantity

FROM retail_shop_sales

WHERE MONTH(sale_date) = 1;



total_quantity
199

-- THE DIFFERENCE OF TOTAL QUANTITY FROM CURRENT MONTH TO PREVIOUS MONTH SHOWN AS A PERCENTAGE WITH THE USE OF LAG()

SELECT

MONTH(sale_date) AS month,

ROUND(SUM(quantity)) AS total_quantity_sold,

(SUM(quantity) - LAG(SUM(quantity), 1)

OVER (ORDER BY MONTH(sale_date))) / LAG(SUM(quantity), 1)

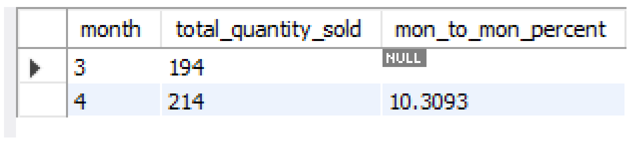
OVER (ORDER BY MONTH(sale_date)) * 100 AS mon_to_mon_percent

FROM retail_shop_sales

WHERE MONTH(sale_date) IN (3,4)

GROUP BY MONTH(sale_date)

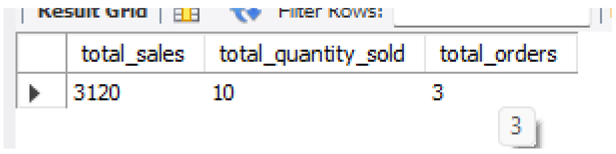
ORDER BY MONTH(sale_date);



month	total_quantity_sold	mon_to_mon_percent
3	194	NULL
4	214	10.3093

-- TOTAL SALES, TOTAL QUANTITY SOLD and TOTAL ORDERS FOR A SPECIFIC DAY

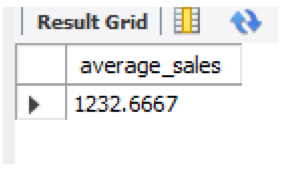
```
SELECT
   SUM(total_amount) AS total_sales,
   SUM(quantity) AS total_quantity_sold,
   COUNT(sale_id) AS total_orders
FROM retail_shop_sales
WHERE sale_date = '2023-01-23';
```



```
total_sales   total_quantity_sold   total_orders
3120          10                    3
```

-- **SALES TRENDLINE FOR MONTH SELECTED**

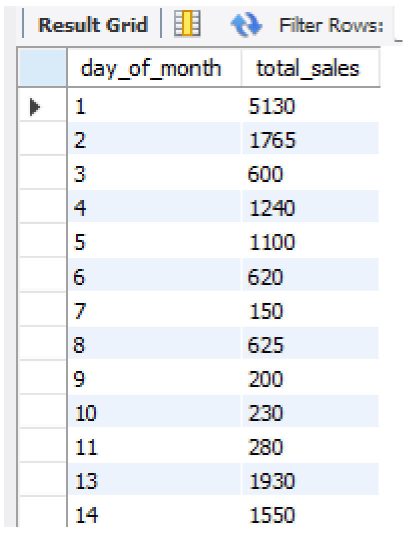
```
SELECT AVG(total_sales) AS average_sales
FROM
            ( SELECT SUM(total_amount) AS total_sales
            FROM retail_shop_sales
            WHERE MONTH(sale_date) = 1
             GROUP BY sale_date
            ) AS internal_query;
```



```
average_sales
1232.6667
```

-- DAILY SALES FOR MONTH SELECTED

```
SELECT
            DAY(sale_date) AS day_of_month,
            ROUND(SUM(total_amount),1) AS total_sales
FROM  retail_shop_sales
WHERE MONTH(sale_date) = 1
GROUP BY DAY(sale_date)
ORDER BY DAY(sale_date);
```

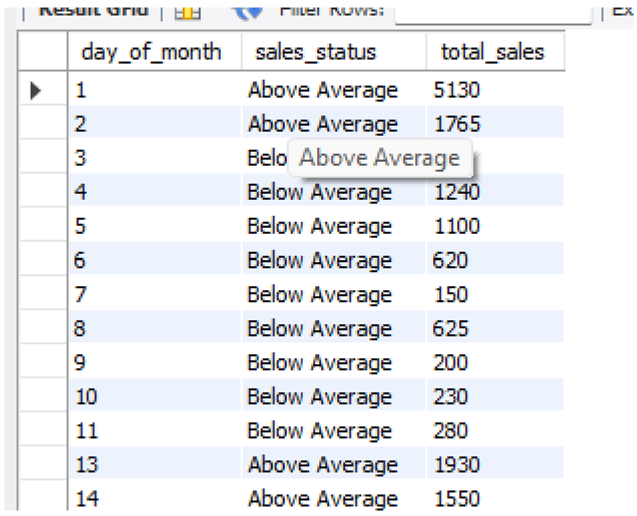

Result Grid | Filter Rows:

day_of_month	total_sales
1	5130
2	1765
3	600
4	1240
5	1100
6	620
7	150
8	625
9	200
10	230
11	280
13	1930
14	1550

-- COMPARING DAILY SALES WITH AVERAGE SALES – IF GREATER THAN "ABOVE AVERAGE" and LESSER THAN "BELOW AVERAGE"

```
SELECT
  day_of_month,
  CASE
    WHEN total_sales > average_sales THEN 'Above Average'
    WHEN total_sales < average_sales THEN 'Below Average'
    ELSE 'Average'
  END AS sales_status,
  total_sales
FROM (
  SELECT
    DAY(sale_date) AS day_of_month,
    SUM(total_amount) AS total_sales,
    AVG(SUM(total_amount)) OVER () AS average_sales
  FROM
    retail_shop_sales
  WHERE
    MONTH(sale_date) = 1
  GROUP BY
```

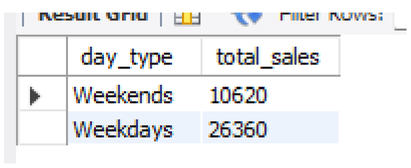
```
    DAY(sale_date)
) AS sales_data
ORDER BY
  day_of_month;
```



	day_of_month	sales_status	total_sales
▶	1	Above Average	5130
	2	Above Average	1765
	3	Belo Above Average	[illegible]
	4	Below Average	1240
	5	Below Average	1100
	6	Below Average	620
	7	Below Average	150
	8	Below Average	625
	9	Below Average	200
	10	Below Average	230
	11	Below Average	280
	13	Above Average	1930
	14	Above Average	1550

-- SALES BY WEEKDAY / WEEKEND FOR MONTH SELECTED

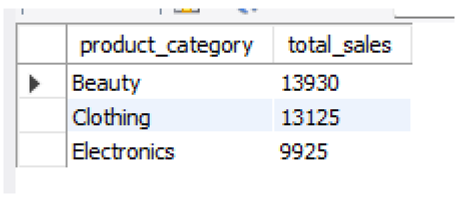
```
SELECT
  CASE
    WHEN DAYOFWEEK(sale_date) IN (1, 7) THEN 'Weekends'
    ELSE 'Weekdays'
  END AS day_type,
  ROUND(SUM(total_amount),2) AS total_sales
FROM
  retail_shop_sales
WHERE
  MONTH(sale_date) = 1
GROUP BY
  CASE
    WHEN DAYOFWEEK(sale_date) IN (1, 7) THEN 'Weekends'
    ELSE 'Weekdays'
  END;
```



	day_type	total_sales
▶	Weekends	10620
	Weekdays	26360

-- SALES BY PRODUCT CATEGORY FOR MONTH SELECTED

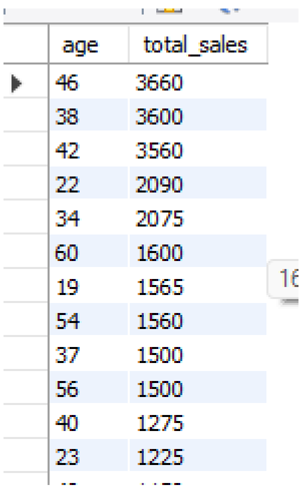
```
SELECT
        product_category,
        SUM(total_amount) as total_sales
FROM retail_shop_sales
WHERE
        MONTH(sale_date) = 1
GROUP BY product_category
ORDER BY total_sales DESC;
```



	product_category	total_sales
▶	Beauty	13930
	Clothing	13125
	Electronics	9925

-- SALES BY AGE FOR MONTH SELECTED

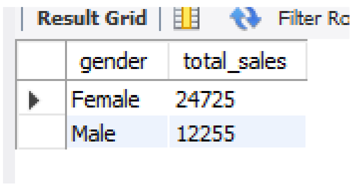
```
SELECT
        age,
        SUM(total_amount) as total_sales
FROM retail_shop_sales
WHERE
        MONTH(sale_date) = 1
GROUP BY age
ORDER BY total_sales DESC;
```



age	total_sales
46	3660
38	3600
42	3560
22	2090
34	2075
60	1600
19	1565
54	1560
37	1500
56	1500
40	1275
23	1225

-- SALES BY GENDER FOR MONTH SELECTED

```
SELECT
        gender,
        SUM(total_amount) as total_sales
FROM retail_shop_sales
WHERE
        MONTH(sale_date) = 1
GROUP BY gender
ORDER BY total_sales DESC;
```

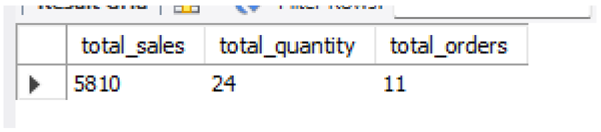


gender	total_sales
Female	24725
Male	12255

-- SALES BY SPECIFIC DAY OF MONTH

```
SELECT
  ROUND(SUM(total_amount)) AS total_sales,
  SUM(quantity) AS total_quantity,
  COUNT(*) AS total_orders
FROM
  retail_shop_sales
```

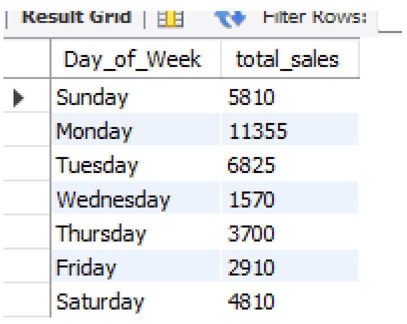
```sql
WHERE
  DAYOFWEEK(sale_date) = 1 -- Filter for Tuesday (1 is Sunday, 2 is Monday, ..., 7 is Saturday)
  AND MONTH(sale_date) = 1;
```



```
total_sales  total_quantity  total_orders
5810         24              11
```

-- TO GET SALES FROM MONDAY TO SUNDAY FOR MONTH SELECTED

```sql
SELECT
  CASE
    WHEN DAYOFWEEK(sale_date) = 2 THEN 'Monday'
    WHEN DAYOFWEEK(sale_date) = 3 THEN 'Tuesday'
    WHEN DAYOFWEEK(sale_date) = 4 THEN 'Wednesday'
    WHEN DAYOFWEEK(sale_date) = 5 THEN 'Thursday'
    WHEN DAYOFWEEK(sale_date) = 6 THEN 'Friday'
    WHEN DAYOFWEEK(sale_date) = 7 THEN 'Saturday'
    ELSE 'Sunday'
  END AS Day_of_Week,
  ROUND(SUM(total_amount)) AS total_sales
FROM
  retail_shop_sales
WHERE
  MONTH(sale_date) = 1
GROUP BY
  CASE
    WHEN DAYOFWEEK(sale_date) = 2 THEN 'Monday'
    WHEN DAYOFWEEK(sale_date) = 3 THEN 'Tuesday'
    WHEN DAYOFWEEK(sale_date) = 4 THEN 'Wednesday'
    WHEN DAYOFWEEK(sale_date) = 5 THEN 'Thursday'
    WHEN DAYOFWEEK(sale_date) = 6 THEN 'Friday'
    WHEN DAYOFWEEK(sale_date) = 7 THEN 'Saturday'
    ELSE 'Sunday'
  END;
```



Day_of_Week	total_sales
Sunday	5810
Monday	11355
Tuesday	6825
Wednesday	1570
Thursday	3700
Friday	2910
Saturday	4810