About the data

The imported data contains daily sales record of **149,116** transactions for **3** coffee sales chain stores (cafés/restaurants) owned by the same proprietor. The data covers the period from **January 2023** to **June 2023**.

The data was provided in **.xlsx** format, but was converted into **.csv** for cleaning and analysis on My**SQL** and was visualized over **POWER BI** dashboard to **communicate** insights and facilitate **decision-making**.

Detailed Power BI dashboard DAX Queries:

Steps followed in the process:

- Data Walkthrough
- Raw data file preparation
- Creating Database
- Importing File
- Cleaning Imported File
- Adding Calculated Columns
- Changing Data Types
- Firing SQL Queries for Business Requirements
- Storing Results
- Preparing SQL Documents

Important functionalities learnt:

- STR_TO_DATE
- ROUND
- CONCAT
- SUM
- COUNT
- DISTINCT
- JOINS - SELF JOIN
- UNION ALL
- ALTER TABLE
- ADD
- UPDATE
- MONTH
- DAY
- DAYOFWEEK
- MONTH
- HOUR
- ADVANCED SUBQUERIES
- CTE (Common Table Expression)
- CASE
- Window Function - ROW_NUMBER , LAG
- ALTER TABLE
- GROUP BY
- ORDER BY
- FILTER

Data Cleaning and Preparation (MySQL Queries)

Correcting data-types

```
-- updating transaction date (imported as text)

UPDATE coffee_sales
SET transaction_date = STR_TO_DATE(transaction_date,'%d-%m-%Y');

ALTER TABLE coffee_sales
MODIFY COLUMN transaction_date DATE;
```

```
-- updating transaction time (imported as text)

UPDATE coffee_sales
SET transaction_time = STR_TO_DATE(transaction_time,'%H:%i:%s');

ALTER TABLE coffee_sales
MODIFY COLUMN transaction_time TIME;
```

Checking duplicates

```
-- checiking for duplicates

WITH CTE AS
(SELECT transaction_id, ROW_NUMBER() OVER (PARTITION BY transaction_id, transaction_date, transaction_time,
                                                        transaction_qty, store_id, store_location, product_id, unit_price,
                                                        product_category, product_type, product_detail) AS rownumber
FROM coffee_sales)
SELECT transaction_id, rownumber
FROM CTE
ORDER BY rownumber DESC;

-- no duplicates found
```

Cleaning transaction_id column

```
-- cleaning transaction id name

ALTER TABLE coffee_sales
RENAME COLUMN ï»¿transaction_id TO transaction_id;
```

Calculated column: serving_size

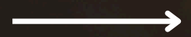
```
-- adding a serving_size column for future analysis
-- based on the column product_detail
-- creating column

ALTER TABLE coffee_sales ADD serving_size VARCHAR(100);

-- inserting values

UPDATE coffee_sales
SET serving_size =
  CASE
    WHEN RIGHT(RTRIM(product_detail), 2) = 'Lg' THEN 'Large'
    WHEN RIGHT(RTRIM(product_detail), 2) = 'Rg' THEN 'Regular'
    WHEN RIGHT(RTRIM(product_detail), 2) = 'Sm' THEN 'Small'
    ELSE 'Not Specified'
  END;
```

Key Performance Indicators

KPI Requirements

1. Total sales for each respective month:

```sql
SELECT
    MONTHNAME(transaction_date) as month,
    ROUND(SUM(unit_price * transaction_qty),2) AS total_monthly_sales
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
ORDER BY
    MONTH(transaction_date);
```



month	total_monthly_sales
January	81677.74
February	76145.19
March	98834.68
April	118941.08
May	156727.76
June	166485.88

2. Month-on-month increase/ decrease in sales and the month-on-month percentage of it

```sql
SELECT
    MONTHNAME(transaction_date) as month,
    ROUND((SUM(unit_price * transaction_qty)),2) AS monthly_sales,
    ROUND((SUM(unit_price * transaction_qty) - LAG(SUM(unit_price * transaction_qty)) OVER (ORDER BY MONTH(transaction_date))),2) AS mom_sales_increase,
                                                    -- (current month total sales - previous month total sales)
    CONCAT(ROUND(((
    (
    SUM(unit_price * transaction_qty) - LAG(SUM(unit_price * transaction_qty)) OVER (ORDER BY MONTH(transaction_date))
    )                                                                   -- (current month total sales - previous month total sales)
    / LAG(SUM(unit_price * transaction_qty)) OVER (ORDER BY MONTH(transaction_date)))   -- divided by previous month total sales
    *100),2),"%") AS mom_percentage_increase_sales
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
ORDER BY
    MONTH(transaction_date);
```

month	monthly_sales	mom_sales_increase	mom_percentage_increase_sales
January	81677.74	NULL	NULL
February	76145.19	-5532.55	-6.77%
March	98834.68	22689.49	29.8%
April	118941.08	20106.4	20.34%
May	156727.76	37786.68	31.77%
June	166485.88	9758.12	6.23%

3. Total no. of orders per month

```sql
SELECT
    MONTHNAME(transaction_date) as month,
    COUNT(transaction_id) AS total_monthly_orders
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
ORDER BY
    MONTH(transaction_date);
```

month	total_monthly_orders
January	17314
February	16359
March	21229
April	25335
May	33527
June	35352

4. Month-on-month increase/ decrease in number of orders and the month-on-month percentage of it

```sql
SELECT
    MONTHNAME(transaction_date) as month,
    COUNT(transaction_id) AS total_monthly_orders,
    (COUNT(transaction_id) - LAG(COUNT(transaction_id)) OVER (ORDER BY MONTH(transaction_date))) AS mom_orders_increase,
                                                    -- (current month total orders - previous month total orders)
    CONCAT(ROUND(((
    (
    COUNT(transaction_id) - LAG(COUNT(transaction_id)) OVER (ORDER BY MONTH(transaction_date))
    )                                                   -- (current month total orders - previous month total orders)
    / LAG(COUNT(transaction_id)) OVER (ORDER BY MONTH(transaction_date))) -- divided by previous month total orders
    *100),2),"%") AS mom_percentage_increase_order_numbers
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
ORDER BY
    MONTH(transaction_date);
```

month	total_monthly_orders	mom_orders_increase	mom_percentage_increase_order_numbers
January	17314	NULL	NULL
February	16359	-955	-5.52%
March	21229	4870	29.77%
April	25335	4106	19.34%
May	33527	8192	32.33%
June	35352	1825	5.44%

5. Total number of quantities sold for each respective month

```sql
SELECT
    MONTHNAME(transaction_date) as month,
    SUM(transaction_qty) AS total_monthly_order_qty
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
ORDER BY
    MONTH(transaction_date);
```

month	total_monthly_order_qty
January	24870
February	23550
March	30406
April	36469
May	48233
June	50942

6. Month-on-month increase/ decrease in quantities sold and the month-on-month percentage of it

```sql
SELECT
    MONTHNAME(transaction_date) as month,
    SUM(transaction_qty) AS total_monthly_order_qty,
    ROUND((SUM(transaction_qty) - LAG(SUM(transaction_qty)) OVER (ORDER BY MONTH(transaction_date))),2) AS mom_qty_increase,
                                                                    -- (current month total quantity size - previous month total quantity size)
    CONCAT(ROUND(((
    (
    SUM(transaction_qty) - LAG(SUM(transaction_qty)) OVER (ORDER BY MONTH(transaction_date))
    )                                                                                   -- (current month total quantity size - previous month total quantity size)
    / LAG(SUM(transaction_qty)) OVER (ORDER BY MONTH(transaction_date)))    -- divided by previous month total quantity size
    *100),2),"%") AS mom_percentage_increase_qty
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
ORDER BY
    MONTH(transaction_date);
```

month	total_monthly_order_qty	mom_qty_increase	mom_percentage_increase_qty
January	24870	NULL	NULL
February	23550	-1320	-5.31%
March	30406	6856	29.11%
April	36469	6063	19.94%
May	48233	11764	32.26%
June	50942	2709	5.62%

Chart Requirements

1. Total Sales, Orders and Quantity over different dates (daily metrics)

```sql
SELECT
    transaction_date,
    CONCAT(ROUND((SUM(transaction_qty * unit_price)/1000),2),"K") as sales_per_day,
    COUNT(transaction_id) as orders_per_day,
    SUM(transaction_qty) as quantities_sold_per_day
FROM
    coffee_sales
GROUP BY
    transaction_date
ORDER BY
    transaction_date;
```

transaction_date	sales_per_day	orders_per_day	quantities_sold_per_day
2023-01-01	2.51K	550	802
2023-01-02	2.4K	566	790
2023-01-03	2.57K	582	823
2023-01-04	2.22K	497	726
2023-01-05	2.42K	547	778
2023-01-06	2.27K	509	736
2023-01-07	2.62K	562	799
2023-01-08	2.64K	562	806
2023-01-09	2.68K	551	742
2023-01-10	2.69K	602	855
2023-01-11	2.56K	561	782
2023-01-12	2.33K	534	759
2023-01-13	3.03K	625	950
2023-01-14	2.68K	577	771
2023-01-15	3.17K	663	927
2023-01-16	2.83K	627	855
2023-01-17	3.29K	532	762
2023-01-18	2.74K	588	827
2023-01-19	2.91K	616	858
2023-01-20	2.6K	578	767
2023-01-21	3.08K	597	878
2023-01-22	2.37K	491	768
2023-01-23	2.85K	589	885
2023-01-24	2.87K	588	883
2023-01-25	2.85K	583	906

transaction_date	sales_per_day	orders_per_day	quantities_sold_per_day
2023-01-26	2.86K	586	887
2023-01-27	2.74K	579	886
2023-01-28	2.04K	470	672
2023-01-29	2.06K	459	652
2023-01-30	2.48K	481	693
2023-01-31	2.33K	462	645
2023-02-01	2.47K	546	796
2023-02-02	2.51K	569	814
2023-02-03	2.59K	586	834
2023-02-04	2.55K	584	834
2023-02-05	2.3K	533	748
2023-02-06	2.2K	489	712
2023-02-07	2.43K	546	769
2023-02-08	2.76K	599	844
2023-02-09	2.61K	562	761
2023-02-10	2.9K	635	918
2023-02-11	2.53K	567	784
2023-02-12	2.89K	607	872
2023-02-13	2.85K	593	904
2023-02-14	2.67K	587	796
2023-02-15	2.93K	624	878
2023-02-16	3.02K	639	872
2023-02-17	2.3K	534	736
2023-02-18	2.87K	594	827
2023-02-19	3.22K	692	948

(The first 50 rows from results are displayed)

2. Sales on basis of weekdays (mon - fri) vs. weekends (sat-sun) - overall

```sql
SELECT
    CASE
        WHEN DAYOFWEEK(transaction_date) IN (1,7) -- sunday == 1, saturday == 7
        THEN 'Weekends'
        ELSE 'Weekdays'
    END AS day_type,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),"K") AS total_sales
FROM
    coffee_sales
GROUP BY
    CASE
        WHEN DAYOFWEEK(transaction_date) IN (1,7) -- sunday == 1, saturday == 7
        THEN "Weekends"
        ELSE "Weekdays"
    END;
```

day_type	total_sales
Weekends	195.22K
Weekdays	503.59K

3. Sales on basis of weekdays (mon - fri) vs. weekends (sat-sun) for specific months

```sql
SELECT
    MONTHNAME(transaction_date) AS month,
    CASE
        WHEN DAYOFWEEK(transaction_date) IN (1,7) -- sunday == 1, saturday == 7
        THEN 'Weekends'
        ELSE 'Weekdays'
    END AS day_type,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),"K") AS total_sales
FROM
    coffee_sales
GROUP BY
    MONTHNAME(transaction_date),
    MONTH(transaction_date),
    CASE
        WHEN DAYOFWEEK(transaction_date) IN (1,7) -- sunday == 1, saturday == 7
        THEN "Weekends"
        ELSE "Weekdays"
    END
ORDER BY
    MONTH(transaction_date);
```

month	day_type	total_sales
January	Weekends	23.16K
January	Weekdays	58.51K
February	Weekdays	54K
February	Weekends	22.14K
March	Weekdays	73.37K
March	Weekends	25.47K
April	Weekends	39.35K
April	Weekdays	79.59K
May	Weekdays	116.63K
May	Weekends	40.1K
June	Weekdays	121.48K
June	Weekends	45K

4. Sales on basis of different store locations

```sql
SELECT
    MONTHNAME(transaction_date) AS month,
    store_id,
    store_location,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),"K") AS total_sales
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date),
    store_id,
    store_location
ORDER BY
    store_id,
    MONTH(transaction_date);
```

month	store_id	store_location	total_sales
January	3	Astoria	27.31K
February	3	Astoria	25.11K
March	3	Astoria	32.84K
April	3	Astoria	39.48K
May	3	Astoria	52.43K
June	3	Astoria	55.08K
January	5	Lower Manhattan	26.54K
February	5	Lower Manhattan	25.32K
March	5	Lower Manhattan	32.89K
April	5	Lower Manhattan	39.16K
May	5	Lower Manhattan	51.7K
June	5	Lower Manhattan	54.45K
January	8	Hell's Kitchen	27.82K
February	8	Hell's Kitchen	25.72K
March	8	Hell's Kitchen	33.11K
April	8	Hell's Kitchen	40.3K
May	8	Hell's Kitchen	52.6K
June	8	Hell's Kitchen	56.96K

5. Month-on-month increase/ decrease in sales per store location along with percentage of it

```sql
SELECT
    MONTHNAME(transaction_date) AS month,
    store_id,
    store_location,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') AS curr_month_sales,    -- current month sales
    CONCAT(ROUND(LAG(SUM(transaction_qty * unit_price)) OVER(PARTITION BY store_id ORDER BY MONTH(transaction_date))/1000,2),'K') AS prev_month_sales, -- previous month sales

    ROUND((SUM(transaction_qty * unit_price) - LAG(SUM(transaction_qty * unit_price)) OVER(PARTITION BY store_id ORDER BY MONTH(transaction_date))),2)
    AS mom_sales_increase,                                                    -- current month sales - pervious month sales

    CONCAT(ROUND((((SUM(transaction_qty * unit_price) - LAG(SUM(transaction_qty * unit_price)) OVER(PARTITION BY store_id ORDER BY MONTH(transaction_date)))
    / LAG(SUM(transaction_qty * unit_price)) OVER(PARTITION BY store_id ORDER BY MONTH(transaction_date)))*100),2),'%')
    AS mom_sales_increase_in_percentage -- mom increase = ((curr. month - prtev. month)/ prev. month) *100 %
FROM
    coffee_sales
GROUP BY
    MONTH(transaction_date),
    MONTHNAME(transaction_date),
    store_id,
    store_location
ORDER BY
    store_id,
    MONTH(transaction_date);
```

month	store_id	store_location	curr_month_sales	prev_month_sales	mom_sales_increase	mom_sales_increase_in_percentage
January	3	Astoria	27.31K	NULL	NULL	NULL
February	3	Astoria	25.11K	27.31K	-2208.32	-8.09%
March	3	Astoria	32.84K	25.11K	7730.09	30.79%
April	3	Astoria	39.48K	32.84K	6642.18	20.23%
May	3	Astoria	52.43K	39.48K	12951.15	32.81%
June	3	Astoria	55.08K	52.43K	2654.35	5.06%
January	5	Lower Manhattan	26.54K	NULL	NULL	NULL
February	5	Lower Manhattan	25.32K	26.54K	-1223.38	-4.61%
March	5	Lower Manhattan	32.89K	25.32K	7568.63	29.89%
April	5	Lower Manhattan	39.16K	32.89K	6270.65	19.07%
May	5	Lower Manhattan	51.7K	39.16K	12540.74	32.02%
June	5	Lower Manhattan	54.45K	51.7K	2745.62	5.31%
January	8	Hell's Kitchen	27.82K	NULL	NULL	NULL
February	8	Hell's Kitchen	25.72K	27.82K	-2100.85	-7.55%
March	8	Hell's Kitchen	33.11K	25.72K	7390.77	28.74%
April	8	Hell's Kitchen	40.3K	33.11K	7193.57	21.73%
May	8	Hell's Kitchen	52.6K	40.3K	12294.79	30.51%
June	8	Hell's Kitchen	56.96K	52.6K	4358.15	8.29%

6. Average daily sales per month

```sql
WITH sales AS
(
SELECT
    transaction_date,
    MONTHNAME(transaction_date) AS month,
    SUM(transaction_qty * unit_price) AS daily_sales
FROM
    coffee_sales
GROUP BY
    transaction_date,
    MONTH(transaction_date),
    MONTHNAME(transaction_date)
)
SELECT
    month,
    CONCAT(ROUND((SUM(daily_sales)/1000),2),"K") AS monthly_sales,
    CONCAT(ROUND((AVG(daily_sales)/1000),1),"K") AS avg_daily_sales_per_month
FROM
    sales
GROUP BY
    month,
    MONTH(transaction_date)
ORDER BY
    MONTH(transaction_date);
```

month	monthly_sales	avg_daily_sales_per_month
January	81.68K	2.6K
February	76.15K	2.7K
March	98.83K	3.2K
April	118.94K	4K
May	156.73K	5.1K
June	166.49K	5.5K

7. daily_average | avg_monthly_sales comparison

```sql
-- part a
WITH day_sales AS (
    SELECT
        transaction_date,
        MONTHNAME(transaction_date) AS month,
        SUM(transaction_qty * unit_price) AS daily_sales
    FROM
        coffee_sales
    GROUP BY
        transaction_date,
        MONTHNAME(transaction_date)
)
SELECT
    d.month,
    d.transaction_date,
    CONCAT(ROUND((d.daily_sales/1000),2),'K') as avg_daily_sales,
    CONCAT(ROUND((m.avg_sales/1000),2),'K') as avg_daily_sales_per_month,
    CASE
        WHEN d.daily_sales > m.avg_sales THEN "Above Average"
        WHEN d.daily_sales < m.avg_sales THEN "Below Average"
        ELSE "Equal to Average"
    END AS sales_status
FROM
    day_sales d
```

```sql
-- part b
    JOIN
    (SELECT
        month,
        AVG(daily_sales) as avg_sales
    FROM
        day_sales
    GROUP BY
        month
    ) AS m
    ON
        d.month = m.month
GROUP BY
    MONTH(transaction_date),
    transaction_date,
    d.daily_sales,
    m.avg_sales
ORDER BY
    MONTH(transaction_date),
    transaction_date;
```

month	transaction_date	avg_daily_sales	avg_daily_sales_per_month	sales_status
January	2023-01-01	2.51K	2.63K	Below Average
January	2023-01-02	2.4K	2.63K	Below Average
January	2023-01-03	2.57K	2.63K	Below Average
January	2023-01-04	2.22K	2.63K	Below Average
January	2023-01-05	2.42K	2.63K	Below Average
January	2023-01-06	2.27K	2.63K	Below Average
January	2023-01-07	2.62K	2.63K	Below Average
January	2023-01-08	2.64K	2.63K	Above Average
January	2023-01-09	2.68K	2.63K	Above Average
January	2023-01-10	2.69K	2.63K	Above Average

(The results are displayed for first 10 results)

8. top 10 products

```sql
SELECT
    product_type,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') AS product_sales
FROM
    coffee_sales
GROUP BY
    product_type
ORDER BY
    SUM(transaction_qty * unit_price) DESC
    LIMIT 10;
```

product_type	product_sales
Barista Espresso	91.41K
Brewed Chai tea	77.08K
Hot chocolate	72.42K
Gourmet brewed coffee	70.03K
Brewed Black tea	47.93K
Brewed herbal tea	47.54K
Premium brewed coffee	38.78K
Organic brewed coffee	37.75K
Scone	36.87K
Drip coffee	31.98K

9. top 10 products for a specific month

```sql
SELECT
    product_type,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') AS product_sales
FROM
    coffee_sales
WHERE
    MONTH(transaction_date) = 5 -- January = 1, May = 5
GROUP BY
    product_type
ORDER BY
    SUM(transaction_qty * unit_price) DESC
LIMIT 10;
```

product_type	product_sales
Barista Espresso	20.42K
Brewed Chai tea	17.43K
Hot chocolate	16.32K
Gourmet brewed coffee	15.56K
Brewed herbal tea	10.93K
Brewed Black tea	10.78K
Premium brewed coffee	8.74K
Organic brewed coffee	8.35K
Scone	8.31K
Drip coffee	7.29K

10. top 10 products for a specific month in a specific category

```sql
SELECT
    product_type,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') AS product_sales
FROM
    coffee_sales
WHERE
    MONTH(transaction_date) = 5 AND product_category = 'Coffee' -- January = 1, May = 5
GROUP BY
    product_type
ORDER BY
    SUM(transaction_qty * unit_price) DESC
LIMIT 10;
```

product_type	product_sales
Barista Espresso	20.42K
Gourmet brewed coffee	15.56K
Premium brewed coffee	8.74K
Organic brewed coffee	8.35K
Drip coffee	7.29K

11. Sales performance across different categories

```sql
SELECT
    product_category,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') AS product_sales
FROM
    coffee_sales
GROUP BY
    product_category
ORDER BY
    SUM(transaction_qty * unit_price) DESC;
```

product_category	product_sales
Coffee	269.95K
Tea	196.41K
Bakery	82.32K
Drinking Chocolate	72.42K
Coffee beans	40.09K
Branded	13.61K
Loose Tea	11.21K
Flavours	8.41K
Packaged Chocolate	4.41K

12. Sales performance across different categories for a specific month

```sql
SELECT
    product_category,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') AS product_sales
FROM
    coffee_sales
WHERE
    MONTH(transaction_date) = 5 -- January = 1, May = 5
GROUP BY
    product_category
ORDER BY
    SUM(transaction_qty * unit_price) DESC;
```

product_category	product_sales
Coffee	60.36K
Tea	44.54K
Bakery	18.57K
Drinking Chocolate	16.32K
Coffee beans	8.77K
Branded	2.89K
Loose Tea	2.4K
Flavours	1.91K
Packaged Chocolate	0.98K

13. Total sales, orders, quantity for a specific day-hour

```sql
SELECT
    ROUND(SUM(transaction_qty * unit_price),2) AS total_sales,
    COUNT(transaction_id) AS total_orders,
    SUM(transaction_qty) AS total_quantities_sold
FROM
    coffee_sales
WHERE
    HOUR(transaction_time) = 8 -- hour of the day is 0800 hrs (24-hour clock 8am)
    AND DAYOFWEEK(transaction_date) = 2 -- Day is Monday, Sunday is 1
    AND MONTH(transaction_date) = 5; -- Month is May
```

total_sales	total_orders	total_quantities_sold
2697.03	572	819

14. Peak hours of sales in a month

```sql
SELECT
    CASE
        WHEN DAYOFWEEK(transaction_date) = 2 THEN "Monday"
        WHEN DAYOFWEEK(transaction_date) = 3 THEN "Tuesday"
        WHEN DAYOFWEEK(transaction_date) = 4 THEN "Wednesday"
        WHEN DAYOFWEEK(transaction_date) = 5 THEN "Thursday"
        WHEN DAYOFWEEK(transaction_date) = 6 THEN "Friday"
        WHEN DAYOFWEEK(transaction_date) = 7 THEN "Saturday"
        ELSE "Sunday"
    END AS "day_of_week",
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') as total_sales
FROM
    coffee_sales
WHERE
    MONTH(transaction_date) = 5 -- month = May
GROUP BY
    CASE
        WHEN DAYOFWEEK(transaction_date) = 2 THEN "Monday"
        WHEN DAYOFWEEK(transaction_date) = 3 THEN "Tuesday"
        WHEN DAYOFWEEK(transaction_date) = 4 THEN "Wednesday"
        WHEN DAYOFWEEK(transaction_date) = 5 THEN "Thursday"
        WHEN DAYOFWEEK(transaction_date) = 6 THEN "Friday"
        WHEN DAYOFWEEK(transaction_date) = 7 THEN "Saturday"
        ELSE "Sunday"
    END;
```

day_of_week	total_sales
Monday	25.22K
Tuesday	25.35K
Wednesday	25.46K
Thursday	20.25K
Friday	20.34K
Saturday	20.8K
Sunday	19.3K

15. Peak days of sales in a month

```sql
SELECT
    HOUR(transaction_time) as hour_of_day,
    CONCAT(ROUND(SUM(transaction_qty * unit_price)/1000,2),'K') as total_sales
FROM
    coffee_sales
WHERE
    MONTH(transaction_date) = 5 -- month = May
GROUP BY
    HOUR(transaction_time)
ORDER BY
    hour_of_day;
```

hour_of_day	total_sales
6	4.91K
7	14.35K
8	18.82K
9	19.15K
10	19.64K
11	10.31K
12	8.87K
13	9.38K
14	9.06K
15	9.53K
16	9.15K
17	8.97K
18	7.68K
19	6.26K
20	0.66K

16. Top preferred serving size of a product

```sql
SELECT
    product_type,
    COUNT(serving_size) AS total_orders,
    MAX(serving_size) AS top_selling_size
FROM
    coffee_sales
GROUP BY
    product_type
ORDER BY
    total_orders DESC;
```

product_type	total_orders	top_selling_size
Brewed Chai tea	17183	Regular
Gourmet brewed coffee	16912	Small
Barista Espresso	16403	Regular
Hot chocolate	11468	Regular
Brewed Black tea	11350	Regular
Brewed herbal tea	11245	Regular
Scone	10173	Not Specified
Organic brewed coffee	8489	Small
Drip coffee	8477	Small
Premium brewed coffee	8135	Small

(The first 10 results are displayed)

17. Most preferred products on different days of the week and total no. of times ordered in the day

```sql
SELECT
    day_of_week,
    product_type AS most_purchased_product,
    total_orders_in_the_day
FROM (
    SELECT
        CASE
            WHEN DAYOFWEEK(transaction_date) = 2 THEN 'Monday'
            WHEN DAYOFWEEK(transaction_date) = 3 THEN 'Tuesday'
            WHEN DAYOFWEEK(transaction_date) = 4 THEN 'Wednesday'
            WHEN DAYOFWEEK(transaction_date) = 5 THEN 'Thursday'
            WHEN DAYOFWEEK(transaction_date) = 6 THEN 'Friday'
            WHEN DAYOFWEEK(transaction_date) = 7 THEN 'Saturday'
            ELSE 'Sunday'
        END AS day_of_week,
        product_type,
        COUNT(*) AS total_orders_in_the_day,
        ROW_NUMBER() OVER(PARTITION BY
            CASE
                WHEN DAYOFWEEK(transaction_date) = 2 THEN 'Monday'
                WHEN DAYOFWEEK(transaction_date) = 3 THEN 'Tuesday'
                WHEN DAYOFWEEK(transaction_date) = 4 THEN 'Wednesday'
                WHEN DAYOFWEEK(transaction_date) = 5 THEN 'Thursday'
                WHEN DAYOFWEEK(transaction_date) = 6 THEN 'Friday'
                WHEN DAYOFWEEK(transaction_date) = 7 THEN 'Saturday'
                ELSE 'Sunday'
            END
            ORDER BY COUNT(*) DESC) AS row_num
    FROM
        coffee_sales
    GROUP BY
        day_of_week,
        product_type
) AS ranked
WHERE
    row_num = 1
ORDER BY
    FIELD(day_of_week, 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday');
```

day_of_week	most_purchased_product	total_orders_in_the_day
Monday	Brewed Chai tea	2514
Tuesday	Gourmet brewed coffee	2449
Wednesday	Brewed Chai tea	2441
Thursday	Brewed Chai tea	2556
Friday	Gourmet brewed coffee	2481
Saturday	Brewed Chai tea	2391
Sunday	Brewed Chai tea	2417

Thank you very much!