Coffee Shop Sales Analysis:

Description of data set: This is a sales data of 3 store locations across New York City, which consists of 149116 rows. The rows and its data types are shown in the table below:

transaction_id	**int**
transaction_date	**text**
transaction_time	**text**
transaction_qty	**int**
store_id	**int**
store_location	**text**
product_id	**int**
unit_price	**double**
product_category	**text**

Database Creation and importing data: Created a database named **coffee_shop_sales_db** imported the .csv file of dataset inside this database.

Query:

```
DROP DATABASE IF EXISTS `coffee_shop_sales_db`;

CREATE DATABASE `coffee_shop_sales_db`;

USE `coffee_shop_sales_db`;

    # After importing the data

SELECT * FROM coffee_shop_sales;

DESCRIBE coffee_shop_sales;
```

Data cleaning:

The data type of **transaction_date** and **transaction_time** are in **text** format which is not desirable so the data type of these columns should be changed to **date** and **time** format.

Query:

```
UPDATE coffee_shop_sales

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

ALTER TABLE coffee_shop_sales

MODIFY COLUMN transaction_date DATE;


UPDATE coffee_shop_sales

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

ALTER TABLE coffee_shop_sales
```

MODIFY COLUMN transaction_time TIME;

Now if I want to describe the data set and check the data types of the columns:

Query: ***DESCRIBE coffee_shop_sales;***

Result:

Field	Type	Null	Key	Default	Extra
transaction_id	int	YES		NULL	
transaction_date	date	YES		NULL	
transaction_time	time	YES		NULL	
transaction_qty	int	YES		NULL	
store_id	int	YES		NULL	
store_location	text	YES		NULL	
product_id	int	YES		NULL	
unit_price	double	YES		NULL	
product_category	text	YES		NULL	
product_type	text	YES		NULL	
product_detail	text	YES		NULL	

Business Analysis:

1. What is total sales amount combining all the dates?
   Query: **SELECT ROUND (SUM (unit_price*transaction_qty),1) AS Total_Sales FROM coffee_shop_sales;**
   Result:

Total_Sales
698812.3

2. What will be the Month-wise Sales?
   Query: **SELECT MONTH (transaction_date) AS Month_,**
   ROUND (SUM (unit_price*transaction_qty),1) AS Total_Sales
   FROM coffee_shop_sales
   GROUP BY 1
   Result:

Month_	Total_Sales
1	81677.7
2	76145.2
3	98834.7
4	118941.1
5	156727.8
6	166485.9

3. Find out the Month-on-Month % change in sales.

Query: First method:

```
SELECT
        Month(transaction_date) AS Month,
        Round(SUM(unit_price*transaction_qty),1) AS Total_Sales,
   (SUM(unit_price*transaction_qty)- lag(SUM(unit_price*transaction_qty),1)
   over (ORDER BY Month(transaction_date)) /
lag(SUM(unit_price*transaction_qty),1)
   over (ORDER BY Month(transaction_date)) *100) AS
MoM_increASe_percentage
FROM
        coffee_shop_sales
where month(transaction_date) in (4,5)
GROUP BY
   month(transaction_date)
ORDER BY month(transaction_date) ;
```

Result:

Month	Total_Sales	MoM_increase_percentage
4	118941.1	NULL
5	156727.8	156627.7600000045

Query: Second method:

```
WITH monthly_sales AS (
   SELECT
      month(transaction_date) AS MONTH,
      SUM (unit_price * transaction_qty) AS total_sales
   FROM coffee_shop_sales
   GROUP BY MONTH
),
sales_with_lag AS (
   SELECT
      MONTH,
      total_sales,
      LAG (total_sales) OVER (ORDER BY MONTH) AS previous_month_sales
   FROM monthly_sales
)
SELECT
   MONTH,
   Round(total_sales,2),
   Round((total_sales-previous_month_sales),2) AS Sales_Diff,
   ROUND (
      CASE
         WHEN previous_month_sales = 0 THEN NULL
         ELSE ((total_sales - previous_month_sales) / previous_month_sales) * 100
```

```
        END, 2
 ) AS mom_percentage_increASe
FROM sales_with_lag
ORDER BY MONTH;
```

Result:

MONTH	Round(total_sales,2)	Sales_Diff	mom_percentage_increase
1	81677.74	NULL	NULL
2	76145.19	-5532.55	-6.77
3	98834.68	22689.49	29.8
4	118941.08	20106.4	20.34
5	156727.76	37786.68	31.77
6	166485.88	9758.12	6.23

4. Find the total number of month-wise orders.

Query:

```
with Monthly_Order AS
    (SELECT MONTH (transaction_date) AS MONTH,
                COUNT (transaction_id) AS Number_of_orders
     FROM coffee_shop_sales
   GROUP BY 1
   )
   SELECT * FROM Monthly_Order;
```

Result:

MONTH	Number_of_orders
1	17314
2	16359
3	21229
4	25335
5	33527
6	35352

5. Calculate the MoM %change in number of orders placed.

Query:

```
WITH monthly_quantity AS (
   SELECT
      MONTH (transaction_date) AS MONTH,
     SUM(transaction_qty) AS Qty_sold
   FROM coffee_shop_sales
   GROUP BY MONTH
),
Qty_with_lag AS (
   SELECT
      MONTH,
      Qty_sold,
      LAG(Qty_sold) OVER (ORDER BY MONTH) AS previous_month_qty
   FROM monthly_quantity
)
```

```
SELECT
   MONTH,
   Qty_sold,
   (Qty_sold-previous_month_qty) AS Difference_qty,
    ROUND (
     CASE
        WHEN previous_month_qty = 0 THEN NULL
        ELSE ((Qty_sold-previous_month_qty) / previous_month_qty) * 100
     END, 2
   ) AS mom_percentage_increase
FROM Qty_with_lag
ORDER BY MONTH;
```

Result:

MONTH	Qty_sold	Difference_qty	mom_percentage_increase
1	24870	NULL	NULL
2	23550	-1320	-5.31
3	30406	6856	29.11
4	36469	6063	19.94
5	48233	11764	32.26
6	50942	2709	5.62

6. Find out the highest selling products of each month.

Query:

```
WITH monthly_product_sales AS (
   SELECT
      MONTH (transaction_date) AS month,
      product_detail, product_id,
      SUM (transaction_qty) AS total_qty_sold
   FROM coffee_shop_sales
   GROUP BY month, product_detail,product_id
),
ranked_products AS (
   SELECT
      month,product_id,
      product_detail,
      total_qty_sold,
      RANK () OVER (PARTITION BY month ORDER BY total_qty_sold
DESC) AS rank_
   FROM monthly_product_sales
)
SELECT
   month,
   product_detail,product_id,
   total_qty_sold
FROM ranked_products
```

```
WHERE rank_ = 1
ORDER BY month;
```

Result:

month	product_detail	product_id	total_qty_sold
1	Brazilian Lg	27	551
2	Peppermint Rg	44	542
3	Earl Grey Rg	50	685
4	Dark chocolate Lg	59	833
5	Earl Grey Rg	50	1075
6	Earl Grey Rg	50	1109

7. Find out the top 10 best-selling products.

Query:

```
SELECT
   product_detail,
   SUM (transaction_qty) AS total_units_sold
FROM coffee_shop_sales
GROUP BY 1
ORDER BY total_units_sold DESC
LIMIT10 ;
```

Result:

product_detail	total_units_sold
Earl Grey Rg	4708
Dark chocolate Lg	4668
Morning Sunrise Chai Rg	4643
Latte	4602
Peppermint Rg	4564
Columbian Medium Roast Rg	4547
Traditional Blend Chai Rg	4512
Latte Rg	4497
Our Old Time Diner Blend Sm	4484
Serenity Green Tea Rg	4477

8. Find out the total sales of all the store locations.

Query:

```
SELECT
   store_location,
   ROUND (SUM (unit_price * transaction_qty),1) AS total_sales
FROM coffee_shop_sales
GROUP BY store_location
ORDER BY total_sales DESC;
```

Result:

store_location	total_sales
Hell's Kitchen	236511.2
Astoria	232243.9
Lower Manhattan	230057.3

9. Find out the Month-wise sales of all the 3 locations.

Query:

```
SELECT
   MONTH (transaction_date) AS month,
   store_location,
   ROUND (SUM (transaction_qty*unit_price),2) AS monthly_sales
FROM coffee_shop_sales
GROUP BY month, store_location
ORDER BY month, store_location;
```

Result:

month	store_location	monthly_sales
1	Astoria	27313.66
1	Hell's Kitchen	27820.65
1	Lower Manhattan	26543.43
2	Astoria	25105.34
2	Hell's Kitchen	25719.8
2	Lower Manhattan	25320.05
3	Astoria	32835.43
3	Hell's Kitchen	33110.57
3	Lower Manhattan	32888.68
4	Astoria	39477.61
4	Hell's Kitchen	40304.14
4	Lower Manhattan	39159.33
5	Astoria	52428.76
5	Hell's Kitchen	52598.93
5	Lower Manhattan	51700.07
6	Astoria	55083.11
6	Hell's Kitchen	56957.08
6	Lower Manhattan	54445.69

10. Find out the Daily sales.

Query:

```
SELECT
     DISTINCT (transaction_date),
     CONCAT (ROUND (SUM(transaction_qty*unit_price)/1000,2),'K') AS
total_sales,
    SUM (transaction_qty) AS Total_Qty_Sold,
   COUNT(transaction_id) AS total_orders
FROM coffee_shop_sales
GROUP BY transaction_date
```

```
ORDER BY transaction_date
LIMIT10;
```

Result: *(the result is showing the first 10 daily sales AS I used LIMIT10 query)*

transaction_date	total_sales	Total_Qty_Sold	total_orders
2023-01-01	2.51K	802	550
2023-01-02	2.4K	790	566
2023-01-03	2.57K	823	582
2023-01-04	2.22K	726	497
2023-01-05	2.42K	778	547
2023-01-06	2.27K	736	509
2023-01-07	2.62K	799	562
2023-01-08	2.64K	806	562
2023-01-09	2.68K	742	551
2023-01-10	2.69K	855	602

11. Find out Total sales on Weekdays and Weekends.

Query:

```
SELECT Month(transaction_date) AS month,
        CASE when dayofweek(transaction_date) in (1,7) then 'Weekends'
             else 'weekdays'
        end AS Day_Type,
        ROUND (SUM (transaction_qty*unit_price),2) AS Total_Sales
        FROM coffee_shop_sales
    GROUP BY 1,2
    ORDER BY 3 DESC;
```

Result:

month	Day_Type	Total_Sales
6	weekdays	121484.08
5	weekdays	116627.84
4	weekdays	79592.51
3	weekdays	73367.33
1	weekdays	58513.11
2	weekdays	54002.67
6	Weekends	45001.8
5	Weekends	40099.92
4	Weekends	39348.57
3	Weekends	25467.35
1	Weekends	23164.63
2	Weekends	22142.52

12. Find out the days on which the Daily sales are greater and below than the monthly average of a particular month.

Query:

```
SELECT
day_of_month,
CASE
    when total_sales > avg_sales THEN 'Above Average'
   when total_sales < avg_sales THEN 'Below Average'
   ELSE 'Average'
```

```
END AS Sales_status,
total_sales
FROM (SELECT
            Day(transaction_date) AS day_of_month,
      ROUND (SUM(transaction_qty*unit_price),2) AS total_sales,
            AVG (SUM(transaction_qty*unit_price)) Over () AS avg_sales
      FROM
            coffee_shop_sales
      WHERE MONTH (transaction_date) = 4
    GROUP BY 1) AS sale_data
ORDER BY day_of_month;
```

Result: (Showing only the first 10 rows FROM a particular month)

day_of_month	Sales_status	total_sales
1	Below Average	3699.9
2	Below Average	3575.85
3	Below Average	3604.95
4	Below Average	3327.3
5	Below Average	3552.7
6	Below Average	3250.2
7	Below Average	3682.8
8	Above Average	4573.06
9	Above Average	4088.88
10	Above Average	4220.3

13. Find out the total sales product category- wise of a particular month.

Query:

```
SELECT
product_category,
ROUND (SUM(transaction_qty*unit_price),2) AS total_sales
FROM coffee_shop_sales
WHERE month(transaction_date) = 5
GROUP BY 1
ORDER BY 2 DESC;
```

Result:

product_category	total_sales
Coffee	60362.85
Tea	44539.85
Bakery	18565.52
Drinking Chocolate	16319.75
Coffee beans	8768.95
Branded	2889
Loose Tea	2395.15
Flavours	1905.6
Packaged Chocolate	981.09

14. Identify top 10 products bASed on sales volume inside a product category:

Query:

```
SELECT
product_type ,
ROUND (SUM(transaction_qty*unit_price),2) AS total_sales
FROM coffee_shop_sales
WHERE MONTH(transaction_date) = 5 AND product_category='coffee'
GROUP BY 1
ORDER BY 2 DESC
LIMIT10;
```

Result:

product_type	total_sales
Barista Espresso	20423.75
Gourmet brewed coffee	15559.2
Premium brewed coffee	8739.2
Organic brewed coffee	8350.2
Drip coffee	7290.5

15. Find out what is the total sale on a particular hour combining all the day of a particular month.

Query:

```
SELECT
  HOUR(transaction_time),
  SUM (unit_price*transaction_qty) AS total_sales,
   SUM (transaction_qty) AS total_num_sales
FROM
  coffee_shop_sales
WHERE MONTH(transaction_date) = 5
GROUP BY 1
ORDER BY 2 DESC;
```

Result:

Hour(transaction_time)	total_sales	total_num_sales
10	19639.13000000001	6022
9	19145.270000000022	5692
8	18822.31000000003	5670
7	14350.680000000037	4373
11	10312.160000000014	3108
15	9525.15000000002	2962
13	9379.210000000008	2913
16	9154.310000000012	2890
14	9057.660000000007	2846
17	8966.850000000013	2841
12	8869.790000000008	2803
18	7679.909999999997	2447
19	6256.469999999997	1922
6	4912.930000000001	1549
20	655.9300000000002	195

16. Find out the total sales on a particular day of a week in a particular month.

Query:

```
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_,
   ROUND (SUM(transaction_qty*unit_price),2) AS total_sales
   FROM coffee_shop_sales
   WHERE MONTH(transaction_date)=4
   GROUP BY 1
   ORDER BY 2 DESC;
```

Result:

Day_	total_sales
Sunday	20038.74
Saturday	19309.83
Wednesday	16470.64
Monday	16422.8
Tuesday	15789.23
Thursday	15716.61
Friday	15193.23