**SQL queries**

Coffee shop sales project

Basic conversions as per the data.

create database coffee\_shop\_sales;

After creating the database, the csv file is imported using table data import wizard.

use coffee\_shop\_sales;

select \* from coffee\_shop\_sales;

describe coffee\_shop\_sales;

we observe the transaction\_date and transaction\_time are in text format which should be converted to date format.

Changing the data type and conversions of the table:

update coffee\_shop\_sales set transaction\_date = str\_to\_date(transaction\_date, '%d/%m/%Y');

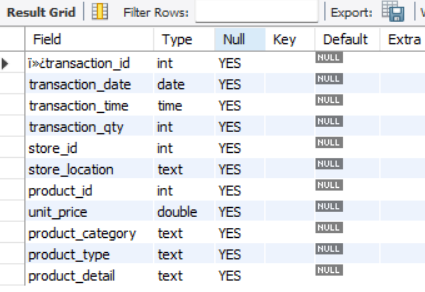
alter table coffee\_shop\_sales modify column transaction\_date date;

The code first updates the transaction\_date entries in the coffee\_shop\_sales table from a string format to a DATE format using the STR\_TO\_DATE function. Then, it modifies the transaction\_date column to have the DATE data type, ensuring that future entries will be stored in the correct format.

update coffee\_shop\_sales set transaction\_time = str\_to\_date(transaction\_time, '%H:%i:%s');

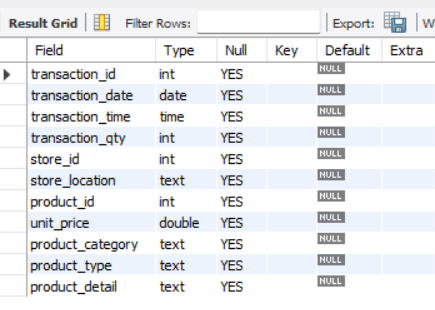
alter table coffee\_shop\_sales modify column transaction\_time time;

The code first updates the transaction\_time entries in the coffee\_shop\_sales table from a string format to a TIME format using the STR\_TO\_DATE function. Then, it modifies the transaction\_time column to have the TIME data type, ensuring that future entries will be stored in the correct format.



alter table coffee\_shop\_sales change column ï»¿transaction\_id transaction\_id int;

*There are some symbols at the beginning of transaction\_id name, we alter that to transaction\_id using the above statement.*

**

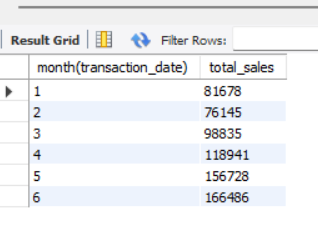
**Sales analysis:**

**calculate the total sales for each respective month:**

select month(transaction\_date), round(sum(transaction\_qty \* unit\_price),0) as total\_sales

from coffee\_shop\_sales group by month(transaction\_date)

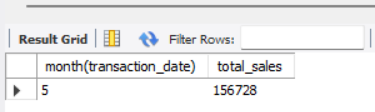
order by month(transaction\_date);



**calculate the total sales for any particular given month:**

select month(transaction\_date), round(sum(transaction\_qty \* unit\_price),0) as total\_sales

from coffee\_shop\_sales where month(transaction\_date) = 5 group by month(transaction\_date);



**Determine the month-on-month sales increase or decrease in sales:**

**calculate the difference in sales between the selected month and the previous month:**

select month(transaction\_date) as month,

sum(transaction\_qty \* unit\_price) as total\_sales,

lag(sum(transaction\_qty \* unit\_price)) over (order by month(transaction\_date)) as previous\_sales,

sum(transaction\_qty \* unit\_price) - lag(sum(transaction\_qty \* unit\_price)) over (order by month(transaction\_date)) as difference\_in\_sales,

case

when sum(transaction\_qty \* unit\_price) > lag(sum(transaction\_qty \* unit\_price)) over(order by month(transaction\_date)) then ' increase in sales'

when sum(transaction\_qty \* unit\_price) < lag(sum(transaction\_qty \* unit\_price)) over(order by month(transaction\_date)) then 'decrease in sales'

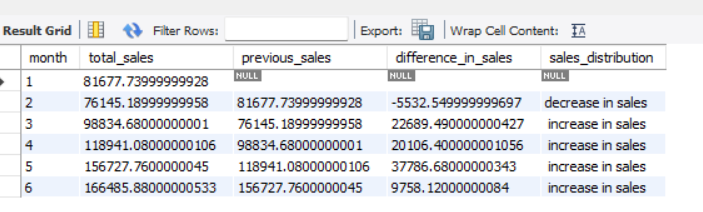
when sum(transaction\_qty \* unit\_price) = lag(sum(transaction\_qty \* unit\_price)) over(order by month(transaction\_date)) then 'no profit no loss'

end sales\_distribution

from coffee\_shop\_sales where month(transaction\_date)

group by month(transaction\_date)

order by month(transaction\_date);

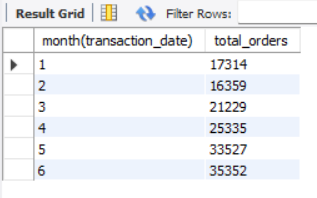


**Order analysis:**

**calculate the total number of orders for each respective month:**

select month(transaction\_date), count(transaction\_id) as total\_orders from coffee\_shop\_sales

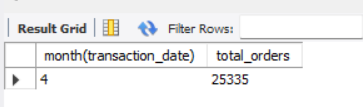
group by month(transaction\_date) order by month(transaction\_date);



**calculate the total orders for any particular given month:**

select month(transaction\_date), count(transaction\_id) as total\_orders from coffee\_shop\_sales where month(transaction\_date) = 4

group by month(transaction\_date);



**Determine the month-on-month sales increase or decrease in the number of orders:**

**calculate the difference in the number of orders between the selected month and the previous month:**

select month(transaction\_date) as month\_no, count(transaction\_id) as current\_month\_orders,

lag(count(transaction\_id)) over(order by month(transaction\_date)) as previous\_month\_orders,

count(transaction\_id) - lag(count(transaction\_id)) over(order by month(transaction\_date)) as difference\_in\_orders,

case

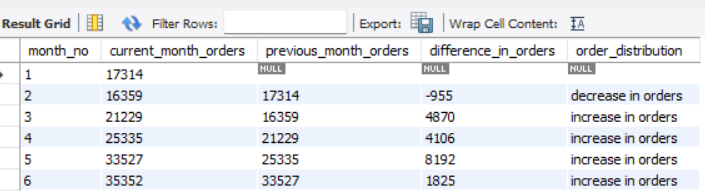
when count(transaction\_id) > lag(count(transaction\_id)) over(order by month(transaction\_date)) then 'increase in orders'

when count(transaction\_id) < lag(count(transaction\_id)) over(order by month(transaction\_date)) then 'decrease in orders'

when count(transaction\_id)=lag(count(transaction\_id)) over(order by month(transaction\_date)) then 'same orders'

end order\_distribution

from coffee\_shop\_sales group by month(transaction\_date) order by month(transaction\_date);

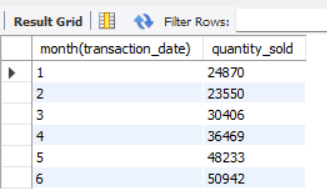


**Total quantity sold analysis:**

**calculate the total quantity sold for each respective month:**

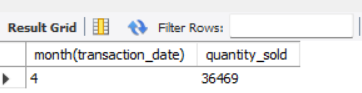
select month(transaction\_date), sum(transaction\_qty) as quantity\_sold from coffee\_shop\_sales

group by month(transaction\_date) order by month(transaction\_date);



**calculate the total quantity sold for any particular month**

select month(transaction\_date), sum(transaction\_qty) as quantity\_sold from coffee\_shop\_sales where month(transaction\_date) = 4 group by month(transaction\_date);



**Determine the month on month increase or decrease in the total quantity sold:**

**Calculate the difference in the total quantity sold between the selected month and the previous month:**

select month(transaction\_date), sum(transaction\_qty) as quantity\_sold\_current\_month,

lag(sum(transaction\_qty)) over(order by month(transaction\_date)) as previous\_month\_quantity,

sum(transaction\_qty) - lag(sum(transaction\_qty)) over(order by month(transaction\_date)) as difference\_in\_qty\_sold,

case

when sum(transaction\_qty) > lag(sum(transaction\_qty)) over(order by month(transaction\_date)) then 'increse in qty sold'

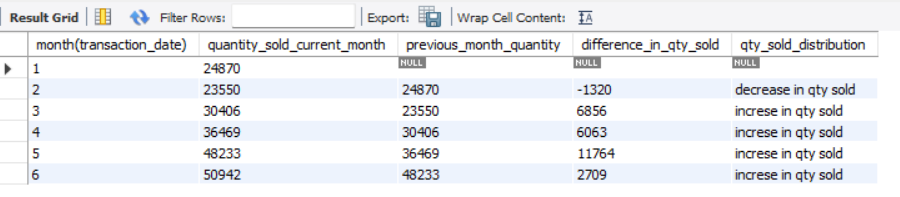
when sum(transaction\_qty) < lag(sum(transaction\_qty)) over(order by month(transaction\_date)) then 'decrease in qty sold'

when sum(transaction\_qty) = lag(sum(transaction\_qty)) over(order by month(transaction\_date)) then 'no change in qty sold'

end as qty\_sold\_distribution

from coffee\_shop\_sales

group by month(transaction\_date) order by month(transaction\_date);

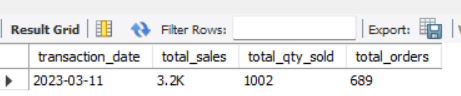
****

**daily sales, quantity and total order analysis by any given date**

select transaction\_date, concat(round(sum(transaction\_qty \* unit\_price)/1000,1),'K') as total\_sales,

sum(transaction\_qty) as total\_qty\_sold,

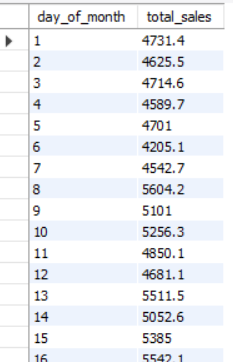
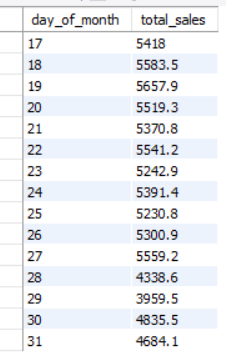
count(transaction\_id) as total\_orders from coffee\_shop\_sales where transaction\_date = '2023-03-11';



**Day by day SALES FOR a given month**

select day(transaction\_date) as day\_of\_month, round(sum(unit\_price \* transaction\_qty),1) as total\_sales

from coffee\_shop\_sales where month(transaction\_date) = 5 group by day(transaction\_date) order by day(transaction\_date);

**SALES BY WEEKDAY / WEEKEND:**

select month(transaction\_date) as monthNo,

case when dayofweek(transaction\_date) in (1,7) then 'weekend'

else 'weekdays'

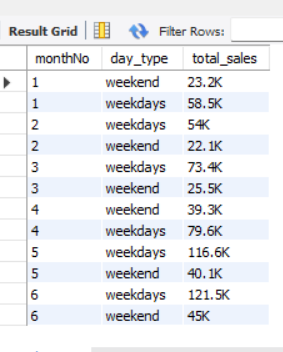
end as day\_type,

concat(round(sum(unit\_price \* transaction\_qty)/1000,1),'K') as total\_sales from coffee\_shop\_sales

group by month(transaction\_date), case when dayofweek(transaction\_date) in (1,7) then 'weekend'

else 'weekdays'

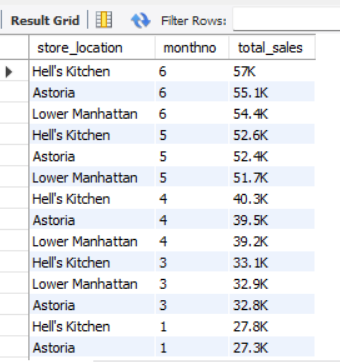
end;



**sales by store location for all the months**

select store\_location, month(transaction\_date) as monthno, concat(round(sum(transaction\_qty \* unit\_price)/1000,1),'K') as total\_sales

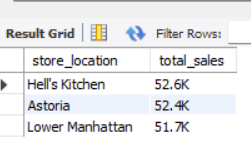
from coffee\_shop\_sales group by store\_location, month(transaction\_date) order by sum(transaction\_qty \* unit\_price) desc;



**sales by store location for a given month**

select store\_location, concat(round(sum(transaction\_qty \* unit\_price)/1000,1),'K') as total\_sales

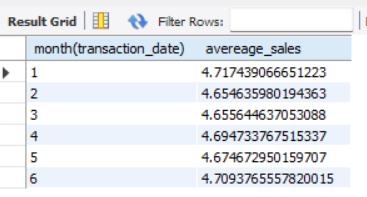
from coffee\_shop\_sales where month(transaction\_date) =5 group by store\_location order by sum(transaction\_qty \* unit\_price) desc;



**Daily sales with average line:**

select month(transaction\_date), avg(unit\_price \* transaction\_qty) as avereage\_sales

from coffee\_shop\_sales group by month(transaction\_date);



**comparing daily sales with average sales – if greater than “above average” and lesser than “below average”**

select day\_of\_month, total\_sales, avg\_sales,

case

when total\_sales > avg\_sales then "above average"

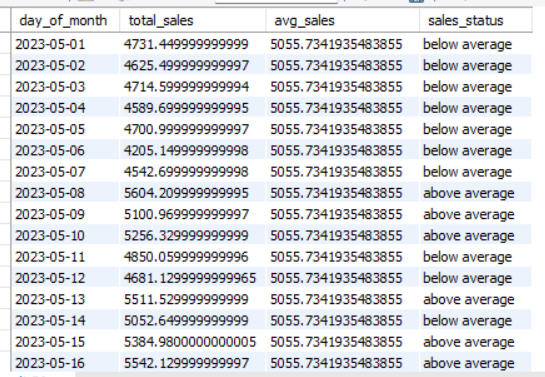
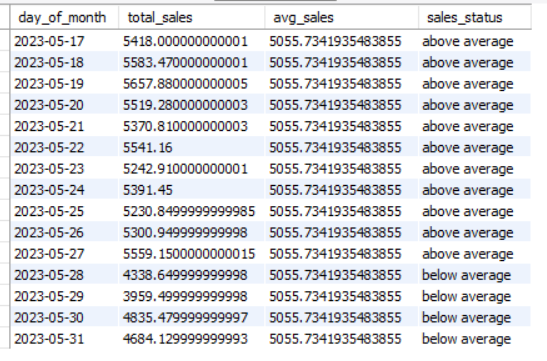
when total\_sales < avg\_sales then "below average"

else "equal to average"

end as sales\_status from (

select date(transaction\_date) as day\_of\_month, sum(transaction\_qty \* unit\_price) as total\_sales, avg(sum(transaction\_qty \* unit\_price)) over()

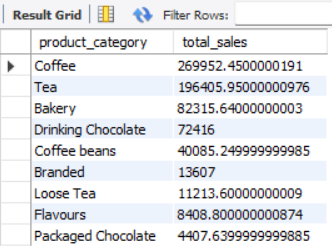
as avg\_sales from coffee\_shop\_sales where month(transaction\_date) = 5 group by date(transaction\_date)) as sales\_data order by day\_of\_month

** **

**Sales analysis by product category:**

select product\_category, sum(transaction\_qty \* unit\_price) as total\_sales

from coffee\_shop\_sales group by product\_category order by sum(transaction\_qty \* unit\_price) desc;

****

**Top 10 products by sales:**

select product\_type, sum(transaction\_qty \* unit\_price) as total\_sales from coffee\_shop\_sales

group by product\_type order by sum(transaction\_qty \* unit\_price) desc limit 10

