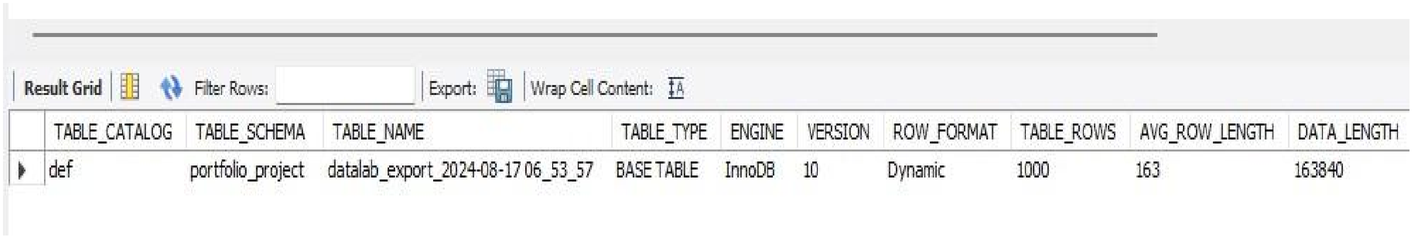
SQL Query Documentation: Motorcycle Part Sales Analysis

1. Checking Information Schema for New Imported File

(Checking information schema for the new imported file)

```
SELECT *
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_NAME = 'datalab_export_2024-08-17 06_53_57';
```

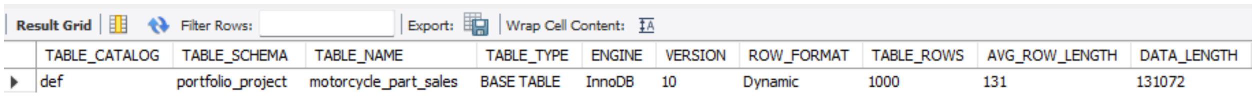


	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE	ENGINE	VERSION	ROW_FORMAT	TABLE_ROWS	AVG_ROW_LENGTH	DATA_LENGTH
▶	def	portfolio_project	datalab_export_2024-08-17 06_53_57	BASE TABLE	InnoDB	10	Dynamic	1000	163	163840

2. Renaming Table Name

(Renaming the imported table)

```
RENAME TABLE portfolio_project.`datalab_export_2024-08-17 06_53_57`
TO portfolio_project.motorcycle_part_sales;
```

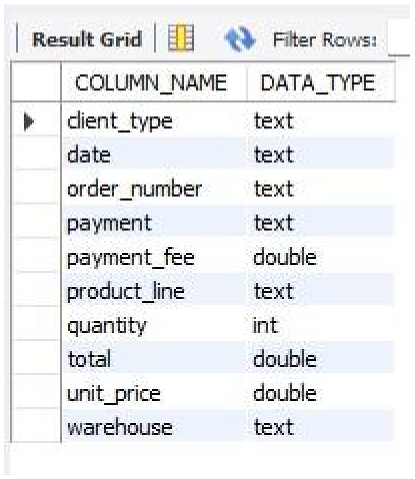


	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE	ENGINE	VERSION	ROW_FORMAT	TABLE_ROWS	AVG_ROW_LENGTH	DATA_LENGTH
▶	def	portfolio_project	motorcycle_part_sales	BASE TABLE	InnoDB	10	Dynamic	1000	131	131072

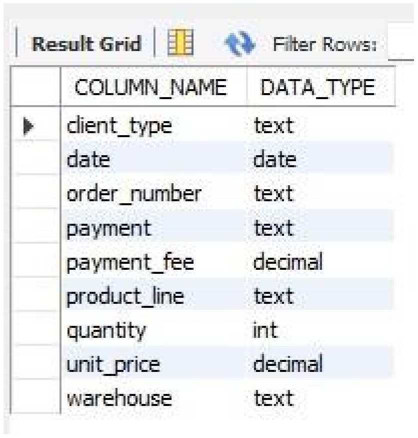
3. Changing Data Types of Renamed Table

(Checking the columns and their data types)

```
SELECT
    COLUMN_NAME,
    DATA_TYPE
FROM
    information_schema.columns
WHERE
    TABLE_NAME = 'motorcycle_part_sales'
    AND TABLE_SCHEMA = 'portfolio_project';

-- Dropping 'total' column
ALTER TABLE motorcycle_part_sales DROP COLUMN total;

-- Modifying 'date' to date data type
ALTER TABLE motorcycle_part_sales MODIFY date date;

-- Modifying 'unit_price' to decimal data type
ALTER TABLE motorcycle_part_sales MODIFY unit_price decimal(10, 2);

-- Modifying 'payment_fee' to decimal data type
ALTER TABLE motorcycle_part_sales MODIFY payment_fee decimal(10, 2);
```



COLUMN_NAME	DATA_TYPE
client_type	text
date	text
order_number	text
payment	text
payment_fee	double
product_line	text
quantity	int
total	double
unit_price	double
warehouse	text

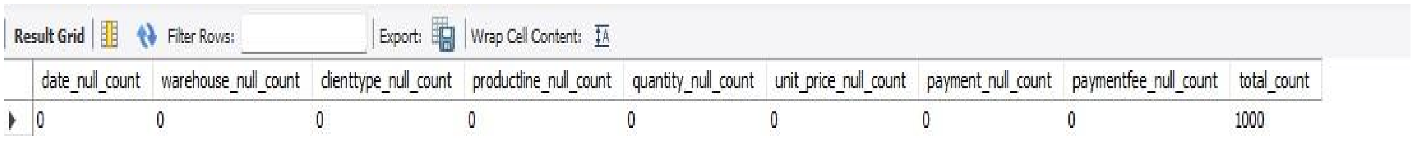


COLUMN_NAME	DATA_TYPE
client_type	text
date	date
order_number	text
payment	text
payment_fee	decimal
product_line	text
quantity	int
unit_price	decimal
warehouse	text

4. Checking for Null Values

```
(Checking for Null values in various columns)
```

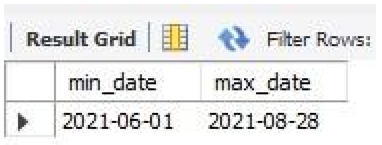
```
SELECT

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE date IS NULL) AS
date_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE warehouse IS
NULL) AS warehouse_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE client_type IS
NULL) AS clienttype_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE product_line IS
NULL) AS productline_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE quantity IS
NULL) AS quantity_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE unit_price IS
NULL) AS unit_price_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE payment IS NULL)
AS payment_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales WHERE payment_fee IS
NULL) AS paymentfee_null_count,

    (SELECT COUNT(*) FROM motorcycle_part_sales) AS total_count;
```



date_null_count	warehouse_null_count	clienttype_null_count	productline_null_count	quantity_null_count	unit_price_null_count	payment_null_count	paymentfee_null_count	total_count
0	0	0	0	0	0	0	0	1000

5. Understanding Data by Date Range

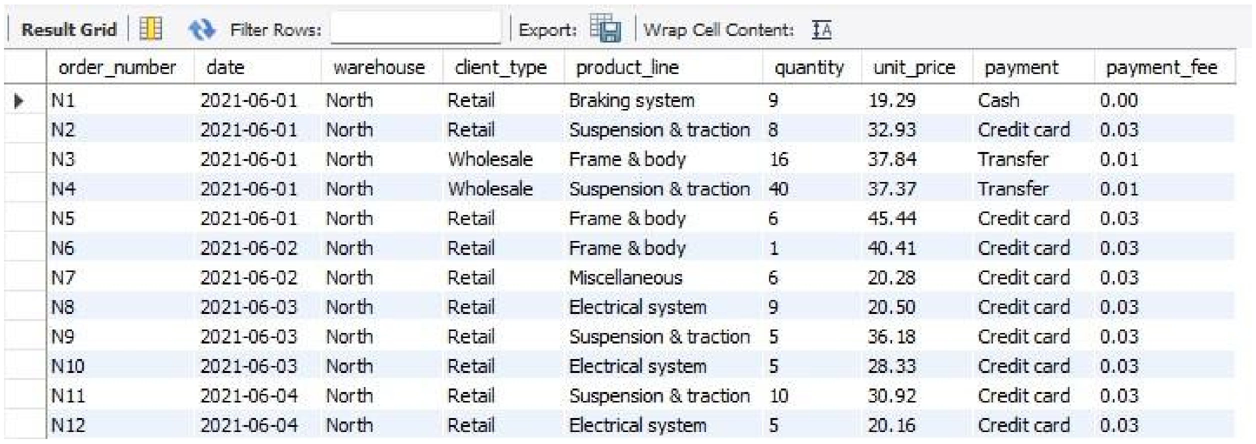
```
(Getting the date range of the data)

SELECT

    MIN(date) AS min_date,

    MAX(date) AS max_date

FROM

    Motorcycle_part_sales;
```



min_date	max_date
2021-06-01	2021-08-28

6. Reviewing the Data

```
(Reviewing the full dataset)

SELECT *

FROM motorcycle_part_sales;
```



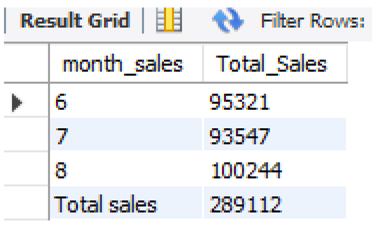
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

```
order_number  date        warehouse  client_type  product_line          quantity  unit_price  payment      payment_fee
N1            2021-06-01  North      Retail       Braking system        9         19.29       Cash         0.00
N2            2021-06-01  North      Retail       Suspension & traction 8         32.93       Credit card  0.03
N3            2021-06-01  North      Wholesale    Frame & body          16        37.84       Transfer     0.01
N4            2021-06-01  North      Wholesale    Suspension & traction 40        37.37       Transfer     0.01
N5            2021-06-01  North      Retail       Frame & body          6         45.44       Credit card  0.03
N6            2021-06-02  North      Retail       Frame & body          1         40.41       Credit card  0.03
N7            2021-06-02  North      Retail       Miscellaneous         6         20.28       Credit card  0.03
N8            2021-06-03  North      Retail       Electrical system     9         20.50       Credit card  0.03
N9            2021-06-03  North      Retail       Suspension & traction 5         36.18       Credit card  0.03
N10           2021-06-03  North      Retail       Electrical system     5         28.33       Credit card  0.03
N11           2021-06-04  North      Retail       Suspension & traction 10        30.92       Credit card  0.03
N12           2021-06-04  North      Retail       Electrical system     5         20.16       Credit card  0.03
```

7. Total Sales Calculation

```
(Calculating total sales for each month and overall total sales)


SELECT

    MONTH(date) AS month_sales,

    ROUND(SUM(unit_price * quantity)) AS Total_Sales

FROM motorcycle_part_sales

GROUP BY MONTH(date)

UNION ALL

SELECT

    'Total sales' AS month_sales,

    ROUND(SUM(unit_price * quantity)) AS Total_Sales

FROM motorcycle_part_sales
```

```
WHERE MONTH(date) IN (6, 7, 8);
```

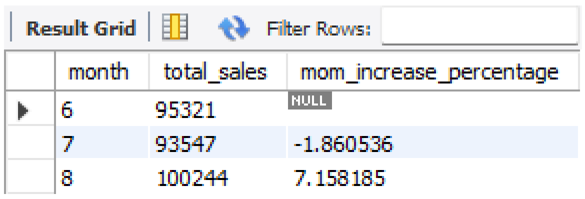


month_sales	Total_Sales
6	95321
7	93547
8	100244
Total sales	289112

8. Total Sales KPI - MOM Difference and MOM Growth

(Calculating Month-over-Month (MOM) difference and growth for total sales)

```
SELECT
    MONTH(date) AS month,
    ROUND(SUM(unit_price * quantity)) AS total_sales,
    (SUM(unit_price * quantity) - LAG(SUM(unit_price * quantity), 1)
    OVER (ORDER BY MONTH(date))) / LAG(SUM(unit_price * quantity), 1)
    OVER (ORDER BY MONTH(date)) * 100 AS mom_increase_percentage
FROM
    motorcycle_part_sales
WHERE MONTH(date) IN (6, 7, 8)
GROUP BY MONTH(date)
ORDER BY MONTH(date);
```



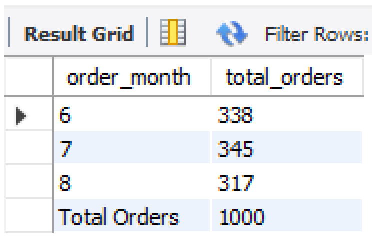
month	total_sales	mom_increase_percentage
6	95321	NULL
7	93547	-1.860536
8	100244	7.158185

9. Total Orders Calculation

(Calculating total orders for each month and overall total orders)

```
SELECT
```

```
    MONTH(date) AS order_month,

    COUNT(order_number) AS total_orders

FROM motorcycle_part_sales

WHERE MONTH(date) IN (6, 7, 8)

GROUP BY MONTH(date)

UNION ALL

SELECT

    'Total Orders' AS order_month,

    COUNT(order_number) AS total_orders

FROM motorcycle_part_sales

WHERE MONTH(date) IN (6, 7, 8);
```



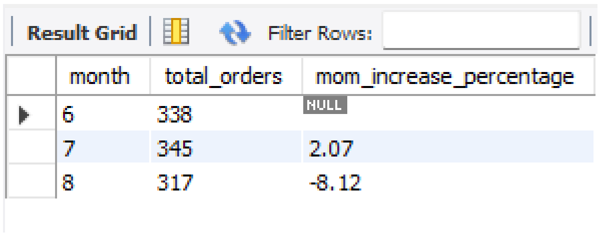
order_month	total_orders
6	338
7	345
8	317
Total Orders	1000

10. Total Orders KPI - MOM Difference and MOM Growth

(Calculating Month-over-Month (MOM) difference and growth for total orders)

```
SELECT

    MONTH(date) AS month,

    ROUND(COUNT(order_number)) AS total_orders,

    ROUND(

        (

            (COUNT(order_number) - LAG(COUNT(order_number), 1) OVER
(ORDER BY MONTH(date)))

            / NULLIF(LAG(COUNT(order_number), 1) OVER (ORDER BY
MONTH(date)), 0)
```

```
            * 100
        ), 2
    ) AS mom_increase_percentage
FROM
    motorcycle_part_sales
WHERE
    MONTH(date) IN (6, 7, 8)
GROUP BY
    MONTH(date)
ORDER BY
    MONTH(date);
```



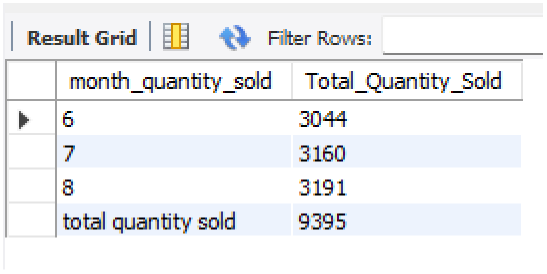
```
month  total_orders  mom_increase_percentage
6      338           NULL
7      345           2.07
8      317           -8.12
```

11. Total Quantity Sold

(Calculating total quantity sold for each month and overall total quantity sold)

```
SELECT
    MONTH(date) AS month_quantity_sold,
    SUM(quantity) AS Total_Quantity_Sold
FROM motorcycle_part_sales
GROUP BY MONTH(date)
UNION ALL
SELECT
    'Total Quantity Sold' AS month_quantity_sold,
    SUM(quantity) AS total_quantity_sold
FROM motorcycle_part_sales
```

```
WHERE MONTH(date) IN (6, 7, 8);
```



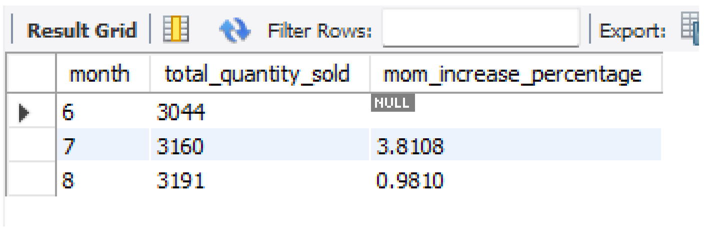
Result Grid | Filter Rows:

	month_quantity_sold	Total_Quantity_Sold
▶	6	3044
	7	3160
	8	3191
	total quantity sold	9395

12. Total Quantity Sold KPI - MOM Difference and MOM Growth

(Calculating Month-over-Month (MOM) difference and growth for total quantity sold)

```
SELECT

    MONTH(date) AS month,

    ROUND(SUM(quantity)) AS total_quantity_sold,

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

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

    OVER (ORDER BY MONTH(date)) * 100 AS mom_increase_percentage

FROM

    motorcycle_part_sales

WHERE

    MONTH(date) IN (6, 7, 8)

GROUP BY

    MONTH(date)

ORDER BY

    MONTH(date);
```



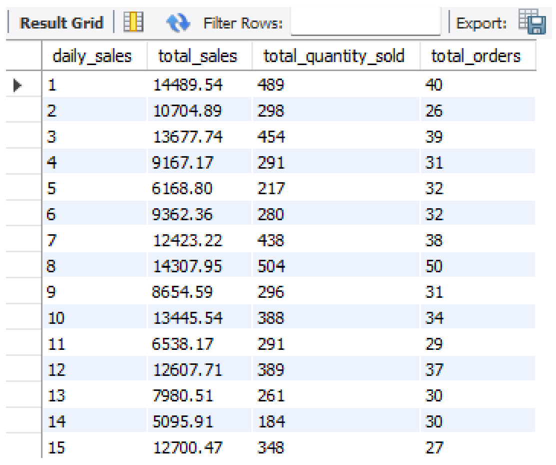
Result Grid | Filter Rows: | Export:

	month	total_quantity_sold	mom_increase_percentage
▶	6	3044	NULL
	7	3160	3.8108
	8	3191	0.9810

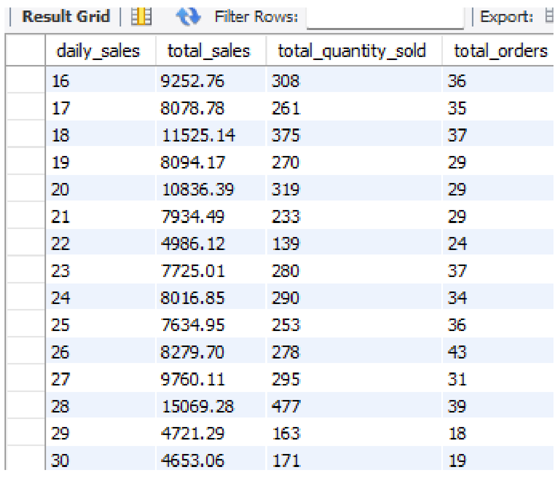
13. Calendar Table – Daily Sales, Quantity, and Total Orders

(Generating a calendar table to summarize daily sales, total quantity sold, and total orders)

```
SELECT

    DAY(date) AS daily_sales,

    SUM(unit_price * quantity) AS total_sales,

    SUM(quantity) AS total_quantity_sold,

    COUNT(order_number) AS total_orders

FROM

    motorcycle_part_sales

GROUP BY

    DAY(date)

ORDER BY

    DAY(date);
```



daily_sales	total_sales	total_quantity_sold	total_orders
1	14489.54	489	40
2	10704.89	298	26
3	13677.74	454	39
4	9167.17	291	31
5	6168.80	217	32
6	9362.36	280	32
7	12423.22	438	38
8	14307.95	504	50
9	8654.59	296	31
10	13445.54	388	34
11	6538.17	291	29
12	12607.71	389	37
13	7980.51	261	30
14	5095.91	184	30
15	12700.47	348	27

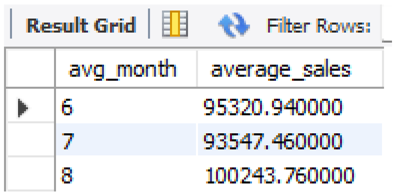


daily_sales	total_sales	total_quantity_sold	total_orders
16	9252.76	308	36
17	8078.78	261	35
18	11525.14	375	37
19	8094.17	270	29
20	10836.39	319	29
21	7934.49	233	29
22	4986.12	139	24
23	7725.01	280	37
24	8016.85	290	34
25	7634.95	253	36
26	8279.70	278	43
27	9760.11	295	31
28	15069.28	477	39
29	4721.29	163	18
30	4653.06	171	19

14. Sales Trend Over Period

(Analyzing the sales trend over the period)

```
SELECT

    month AS avg_month,
    AVG(total_sales) AS average_sales
FROM (
    SELECT
        MONTH(date) AS month,
        SUM(unit_price * quantity) AS total_sales
    FROM
        motorcycle_part_sales
    WHERE
        MONTH(date) IN (6, 7, 8)
    GROUP BY
        MONTH(date)
) AS internal_query
GROUP BY
    Month;
```



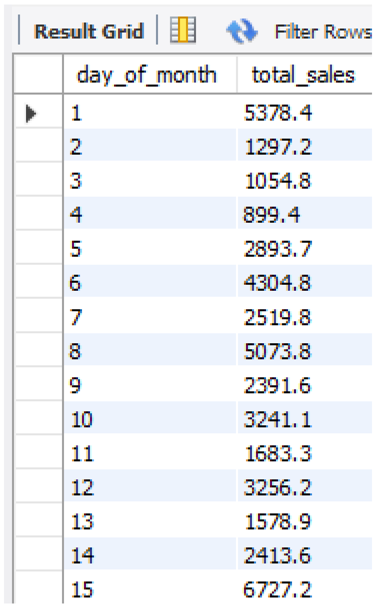
```
avg_month  average_sales
6          95320.940000
7          93547.460000
8          100243.760000
```

15. Daily Sales for Selected Month

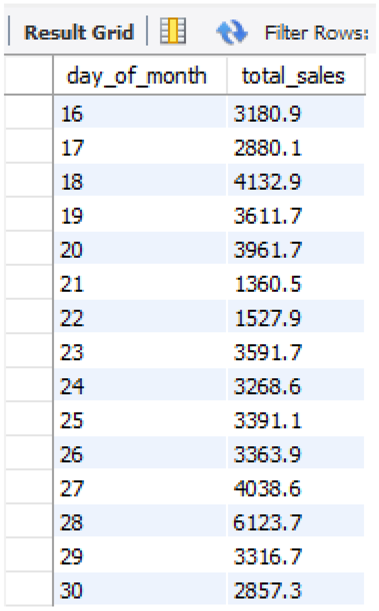
(Analyzing daily sales for the selected month)

```
SELECT
    DAY(date) AS day_of_month,
    ROUND(SUM(unit_price * quantity),1) AS total_sales
FROM
    motorcycle_part_sales
WHERE
    MONTH(date) = 6  -- Filter for the rest
```

```
GROUP BY
    DAY(date)
ORDER BY
    DAY(date);
```



day_of_month	total_sales
1	5378.4
2	1297.2
3	1054.8
4	899.4
5	2893.7
6	4304.8
7	2519.8
8	5073.8
9	2391.6
10	3241.1
11	1683.3
12	3256.2
13	1578.9
14	2413.6
15	6727.2



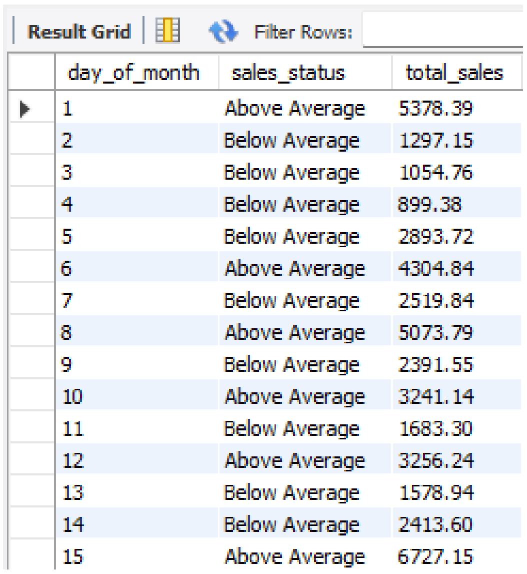
day_of_month	total_sales
16	3180.9
17	2880.1
18	4132.9
19	3611.7
20	3961.7
21	1360.5
22	1527.9
23	3591.7
24	3268.6
25	3391.1
26	3363.9
27	4038.6
28	6123.7
29	3316.7
30	2857.3

16. Comparing Daily Sales with Average Sales

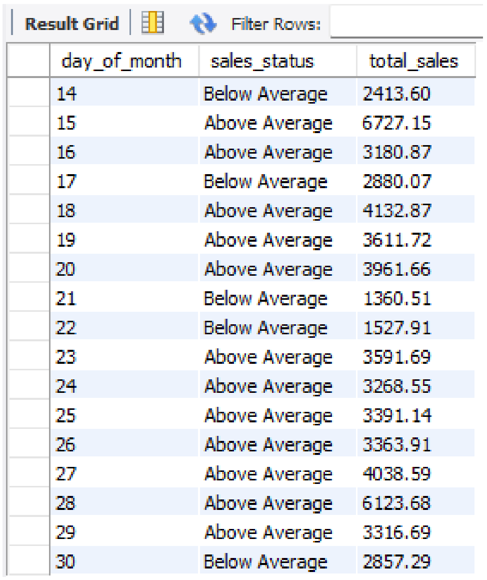
(Comparing daily sales with average sales to determine if it's above or below average)

```
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(date) AS day_of_month,
        SUM(unit_price * quantity) AS total_sales,
        AVG(SUM(unit_price * quantity)) OVER () AS avg_sales
    FROM
        motorcycle_part_sales
    WHERE
        MONTH(date) = 6  (Month of June)
    GROUP BY
        DAY(date)
) AS sales_data
ORDER BY
    Day_of_month;
```



day_of_month	sales_status	total_sales
1	Above Average	5378.39
2	Below Average	1297.15
3	Below Average	1054.76
4	Below Average	899.38
5	Below Average	2893.72
6	Above Average	4304.84
7	Below Average	2519.84
8	Above Average	5073.79
9	Below Average	2391.55
10	Above Average	3241.14
11	Below Average	1683.30
12	Above Average	3256.24
13	Below Average	1578.94
14	Below Average	2413.60
15	Above Average	6727.15

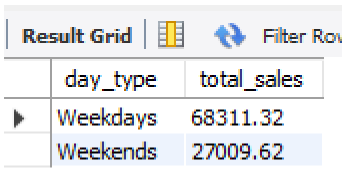


day_of_month	sales_status	total_sales
14	Below Average	2413.60
15	Above Average	6727.15
16	Above Average	3180.87
17	Below Average	2880.07
18	Above Average	4132.87
19	Above Average	3611.72
20	Above Average	3961.66
21	Below Average	1360.51
22	Below Average	1527.91
23	Above Average	3591.69
24	Above Average	3268.55
25	Above Average	3391.14
26	Above Average	3363.91
27	Above Average	4038.59
28	Above Average	6123.68
29	Above Average	3316.69
30	Below Average	2857.29

17. Sales by Weekday / Weekend

(Analyzing sales by weekdays and weekends)

```
SELECT
    CASE
        WHEN DAYOFWEEK(date) IN (1, 7) THEN 'Weekends'
        ELSE 'Weekdays'
    END AS day_type,

    ROUND(SUM(unit_price * quantity),2) AS total_sales

FROM

    motorcycle_part_sales

WHERE

    MONTH(date) = 6  (June)

GROUP BY

    CASE

        WHEN DAYOFWEEK(date) IN (1, 7) THEN 'Weekends'

        ELSE 'Weekdays'

    END;
```

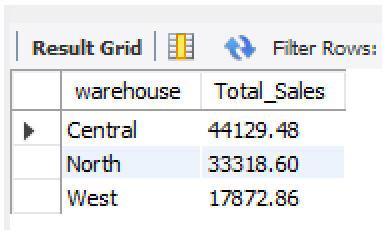


day_type	total_sales
Weekdays	68311.32
Weekends	27009.62

18. Sales by Store Location

(Analyzing sales by store location)

```
SELECT

    warehouse,

    SUM(unit_price * quantity) AS Total_Sales

FROM motorcycle_part_sales

WHERE

    MONTH;
```



warehouse	Total_Sales
Central	44129.48
North	33318.60
West	17872.86