RETAIL SALES ANALYSIS USING SQL

Presented By

RAJ SINGH



INTRODUCTION

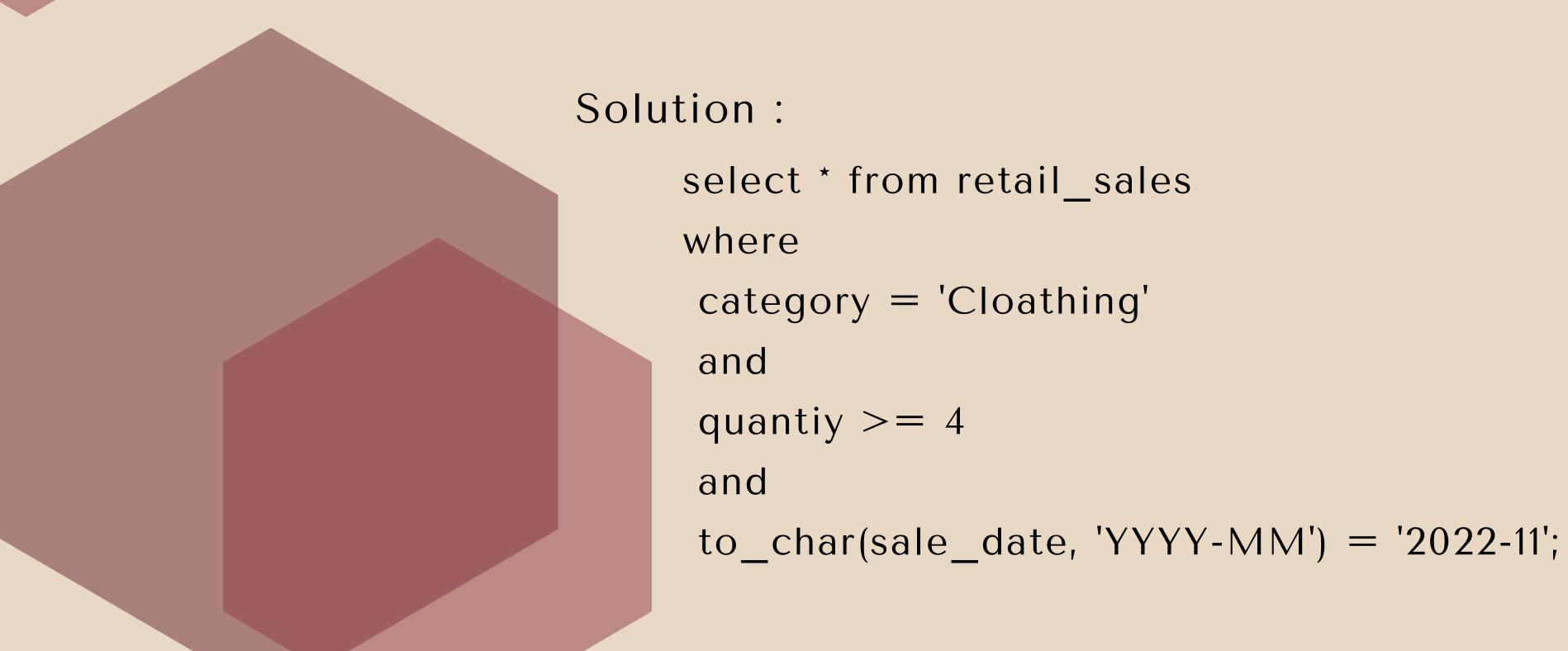
In the competitive world of retail, data-driven decisions are key to staying ahead. This project explores retail sales data using SQL to uncover trends, enhance customer satisfaction, and drive profitability. By analyzing transaction records and customer interactions, this project aims to deliver actionable insights that improve business operations and inform strategic planning.

The project focuses on analyzing key metrics such as sales performance, customer behavior, and inventory management. Using SQL's robust querying capabilities, we transform raw data into meaningful insights to optimize retail operations.

WRITE A SQL QUERY TO RETRIVE ALL COLUMNS FOR SALES MADE ON '2022-11-05'.

```
select *
    from retail_sales
where
sale_date = '2022-11-05';
```

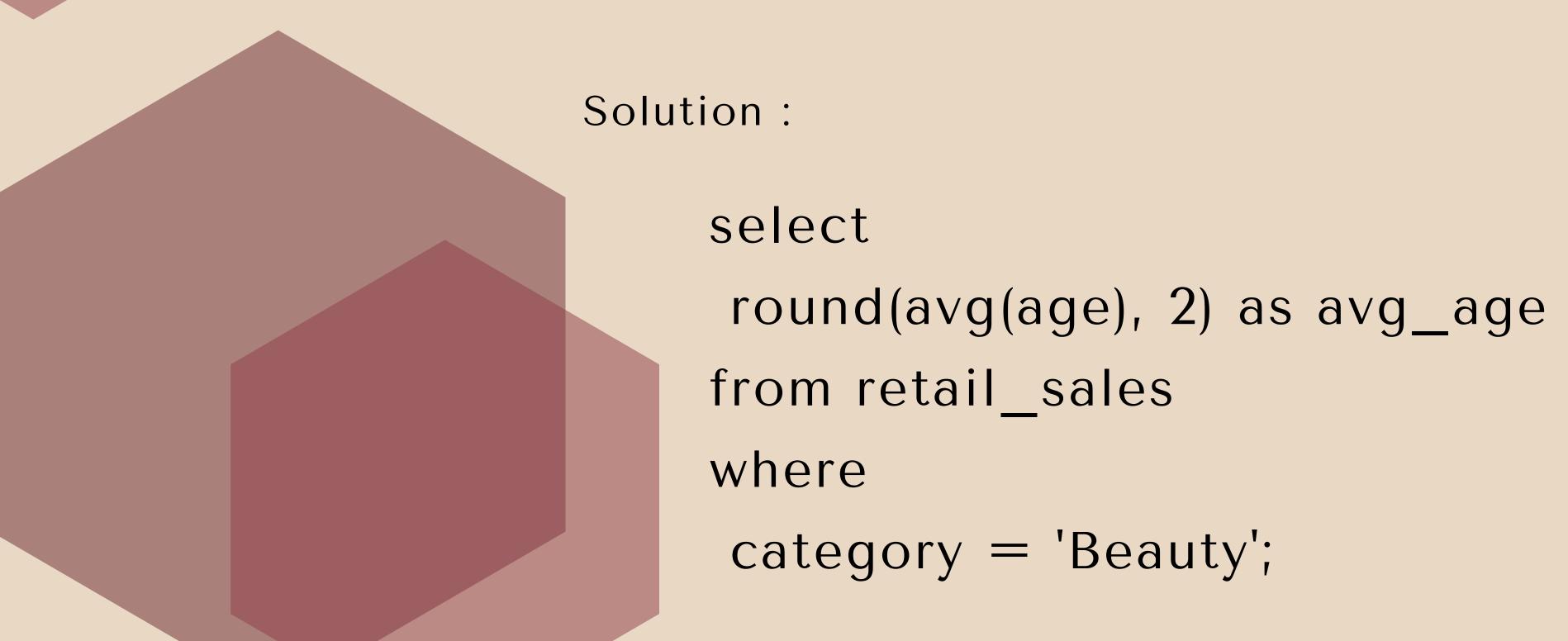
WRITE A SQL QUERY TO RETRIVE ALL TRANSACTION WHERE THE CATEGORY IS 'CLOTHING' AND THE QUANTITY SOLD IS MORE THAN 4 IN THE MONTH OF NOV-2022.



WRITE A SQL QUERY TO CALCULATE THE TOTAL SALES (TOTAL_SALE) FOR EACH CATEGORY.

```
select
 category,
 sum(total_sale) as net_sale,
 count(*) as total_orders
from retail sales
group by 1;
```

WRITE A SQL QUERY TO FIND THE AVERAGE AGE OF CUSTOMERS WHO PURCHASED ITEMS FROM THE 'BEAUTY' CATEGORY.

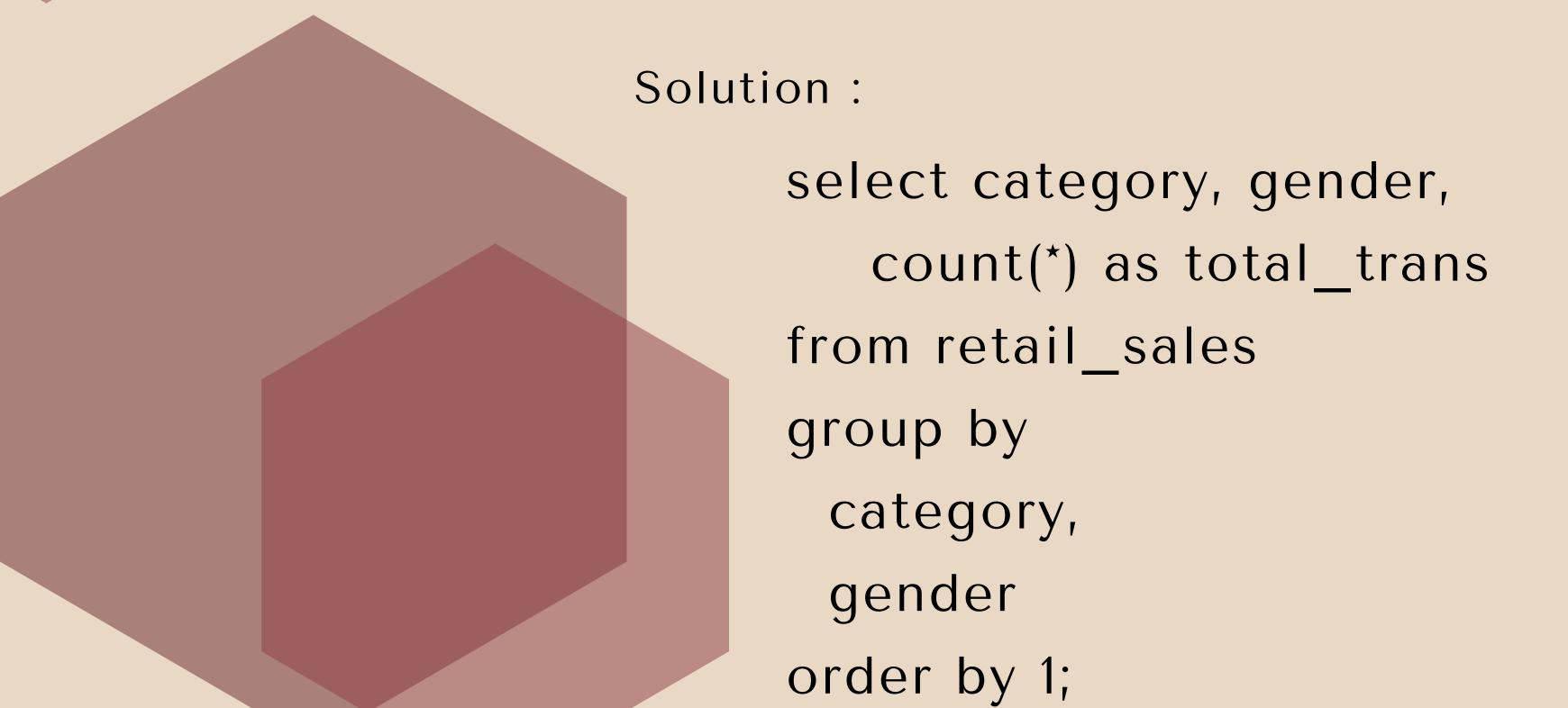


WRITE A SQL QUERY TO FIND ALL TRANSACTIONS WHERE THE TOTAL_SALE IS GREATER THAN 1000.

Solution:

select *
from retail_sales
where
total_sale > 1000;

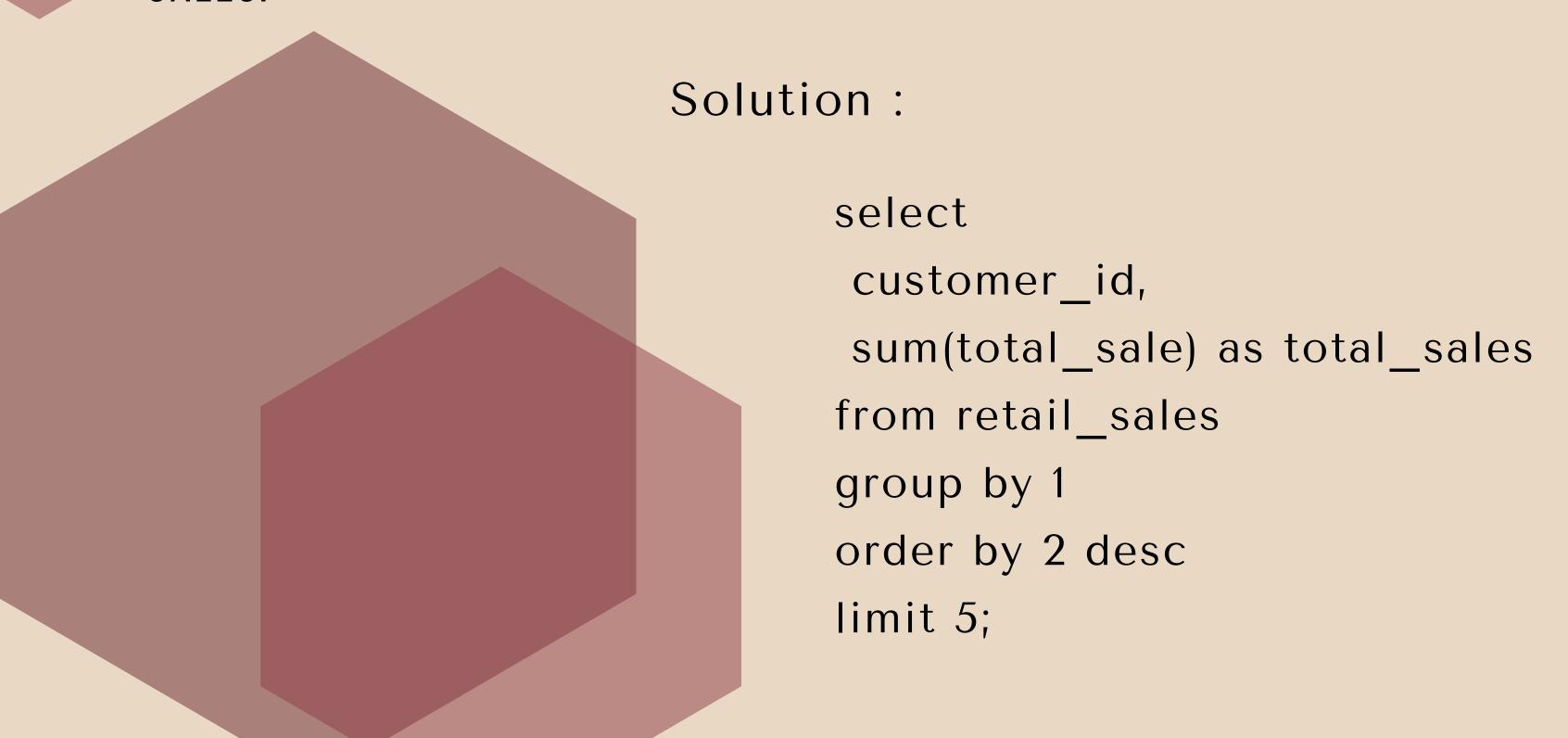
WRITE A SQL QUERY TO FIND THE TOTAL NUMBER AS TRANSACTIONS (TRANSACTION_ID) MADE BY EACH GENDER IN EACH CATEGORY.





```
select year, month, avg_sale from
(select
  extract(year from sale_date) as year,
  extract(month from sale_date) as month,
  avg(total_sale) as avg_sale
from retail_sales
group by 1, 2) as t1
where rank = 1;
```

WRITE A SQL QUERY TO FIND THE TOP 5 CUSTOMERS BASED ON THE HIGHEST TOTAL SALES.



WRITE A SQL QUERY TO FIND THE NUMBER OF UNIQUE CUSTOMERS WHO PURCHASED ITEMS FROM EACH CATEGORY.

```
select
  category,
  count(distinct customer_id) as
  cnt_unique_cs
  from retail_sales
  group by category;
```

WRITE A SQL QUERY TO CREATE EACH SHIFT AND NUMBER OF ORDERS (EXAMPLE MORNING <= 12, AFTERNOON BETWEEN !2 & 17, EVENING > 17)

Solution: with hourly_sale as select *, case when extract(hour from sale_time) < 12 then 'Morning' when extract(hour from sale_time) between 12 and 17 then 'Afternon' else 'Evening' end as shift from retail_sales select shift, count(*) as total_orders from hourly_sale group by shift;

THANK YOU

25 Jan, 2025