SQL Project with E-Commerce Dataset from Kaggle

Sales analysis uncovers valuable insights, such as identifying top-selling and underperforming products, spotting market trends, and recognizing opportunities for sales growth. See the summary below for how I conducted sales analysis on a dataset from an Indian e-commerce website. This dataset was sourced from Kaggle.

Here are the hypothetical scenarios that I created to analyze the data, along with my solutions and results.

1. Combined Orders View Creation

```sql
--Combine the list_of_orders with the order_details table to create convinent view

create view combined_orders as
    select
          d. order_id
        , d.amount
        , d.profit
        , d.quantity
        , d.category
        , d.sub_category
        , l.order_date
        , l.customer_name
        , l.state
        , l.city
    from
        order_details d
inner join list_of_orders l
    on d.order_id = l.order_id
;
```

I started by importing the data into the three created tables: list_of_orders, order_details, and sales_target. Then, a view called combined_orders was created by joining the order_details and list_of_orders tables. It merged relevant columns to make future queries more efficient by offering a combined view of all orders, including details like order ID, customer information, and product details. The inner join matched orders from both tables based on order_id, providing a detailed view that can be used in future queries

2. Key Metrics: Total Orders, Customers, Cities, and States

```sql
--Find the number of orders, customers, cities, and states

select
      count(distinct order_id) as num_of_orders
    , count(distinct customer_name) as num_of_customers
    , count(distinct city) as num_of_cities
    , count (distinct state) as num_of_states
from
    combined_orders
;
```

This query calculated key business metrics, such as the total number of orders, total number of unique customers, total number of cities, and total number of states. This information provided a high-level overview of the business's reach and activity.

	num_of_orders bigint	num_of_customers bigint	num_of_cities bigint	num_of_states bigint
1	500	332	24	19

From the data we can see that there are 500 orders and 332 customers from 24 different cities and 19 states.

3. Identify the Top 10 Most Profitable States and Cities

```
--Find the top 10 profitable states & cities so the company can expand its business
--Determine number of products sold & number of customers in these top 10 states & cities

select
      c.state
    , c.city
    , count(distinct customer_name) as num_of_customers
    , sum(profit) as total_profit
    , sum(quantity) as total_quantity
from
    combined_orders c
group by
      c.state
    , c.city
order by
    total_profit desc
limit
    10
;
```

We identified the top 10 most profitable states and cities, helping the business to make decisions on where to focus its growth strategy. We also calculated total profit, quantity of items sold in each location, and the number of customers in these top 10 locations.

	state character varying (255)	city character varying (255)	num_of_customers bigint	total_profit numeric	total_quantity bigint
1	Maharashtra	Pune	16	4539.00	329
2	Madhya Pradesh	Indore	63	4159.00	1084
3	Uttar Pradesh	Allahabad	9	3081.00	138
4	Delhi	Delhi	21	2987.00	277
5	West Bengal	Kolkata	16	2500.00	216
6	Rajasthan	Udaipur	12	2010.00	115
7	Kerala	Thiruvananthapuram	11	1871.00	157
8	Maharashtra	Mumbai	61	1637.00	727
9	Gujarat	Surat	10	1345.00	93
10	Haryana	Chandigarh	10	1325.00	111

We can see the most profitable city is Pune followed by Indore, and Allahabad. With this information, business leaders and sellers can determine how to allocate resources to match customer demand in these areas.

4. Evaluate the Monthly Profitability & Quantities Sold

```sql
--Evaluate monthly profitability & monthly quantity sold to determine if there are patterns in the data

select
      extract(year from l.order_date) as year
    , extract(month from l.order_date) as month
    , sum(d.profit) as total_profit
    , sum(d.quantity) as total_quantity_sold
from
    order_details d
join
    list_of_orders l on d.order_id = l.order_id
group by
      extract(year from l.order_date)
    , extract(month from l.order_date)
order by
      year
    , month
;
```

This query evaluates monthly profitability, and the quantity of products sold over time, providing a view of seasonality or trends that may exist in the sales activity. The extract(year/month from l.order_date) isolates the year and month from each order date.

	year numeric	month numeric	total_profit numeric	total_quantity_sold bigint
1	2018	1	-3296.00	203
2	2018	2	685.00	58
3	2018	3	669.00	144
4	2018	4	-1043.00	337
5	2018	5	-891.00	306
6	2018	6	-3759.00	353
7	2018	7	-2065.00	239
8	2018	8	-1059.00	601
9	2018	9	-3509.00	310
10	2018	10	5979.00	414
11	2018	11	4955.00	433
12	2018	12	1535.00	209
13	2019	1	8655.00	640
14	2019	2	2291.00	253
15	2019	3	6633.00	485
16	2019	4	1295.00	106
17	2019	5	943.00	63
18	2019	6	700.00	52
19	2019	7	975.00	67
20	2019	8	594.00	83

The business experienced losses from April 2018 to September 2018 but recovered with profitable months from October 2018 to August 2019. We can also observe that high quantities of units sold do not always mean profitable months as an August 2018 loss occurred even though the business sold over 600 units of merchandise.

5. Product Category Performance vs. Sales Targets

```sql
--Evaluate how well each product category is performing against its sales target
--Which categories are consistently over- or under-performing across months
--Format sales and difference as currency and percent difference as percentage for easier reading

select
      to_char(l.order_date, 'Mon-YY') as month_year
    , extract(year from l.order_date) as year
    , extract(month from l.order_date) as month
    , d.category
    , to_char(sum(d.amount), 'FM$999,999,999.00') as actual_sales
    , to_char(t.target, 'FM$999,999,999.00') as target_sales
    , to_char(sum(d.amount) - t.target, 'FM$999,999,999.00') as difference
    , round((sum(d.amount) - t.target) / nullif(t.target, 0) * 100, 2) || '%' as percent_difference
from
    order_details d
join
    list_of_orders l
    on d.order_id = l.order_id
join
    sales_target t
        on to_char(l.order_date, 'Mon-YY') = t.month_of_order_date
        and d.category = t.category
group by
      to_char(l.order_date, 'Mon-YY')
    , extract(year from l.order_date)
    , extract(month from l.order_date)
    , d.category
    , t.target
order by
      year
    , month
    , category
;
```

I thought it would be useful to business leaders and stakeholders to show how well each product category is performing against its sales target and which categories are consistently over- or under-performing across months. I aggregated actual sales and target sales for each month and category, joined order_details and list_of_orders using order_id, and joined sales_target on both month and category to link sales data with the corresponding target. The difference between actual sales and target sales was calculated along with a percent difference to show how the sales deviate from the target as a percentage. The output was formatted to show currency amounts and percentages for easier readability.

	month_year text	year numeric	month numeric	category character varying (255)	actual_sales text	target_sales text	difference text	percent_difference text
1	Apr-18	2018	4	Clothing	$21,486.00	$12,000.00	$9,486.00	79.05%
2	Apr-18	2018	4	Electronics	$17,714.00	$9,000.00	$8,714.00	96.82%
3	Apr-18	2018	4	Furniture	$13,140.00	$10,400.00	$2,740.00	26.35%
4	May-18	2018	5	Clothing	$13,972.00	$12,000.00	$1,972.00	16.43%
5	May-18	2018	5	Electronics	$17,518.00	$9,000.00	$8,518.00	94.64%
6	May-18	2018	5	Furniture	$9,354.00	$10,500.00	$-1,146.00	-10.91%
7	Jun-18	2018	6	Clothing	$14,476.00	$12,000.00	$2,476.00	20.63%
8	Jun-18	2018	6	Electronics	$13,092.00	$9,000.00	$4,092.00	45.47%
9	Jun-18	2018	6	Furniture	$7,244.00	$10,600.00	$-3,356.00	-31.66%
10	Jul-18	2018	7	Clothing	$8,040.00	$14,000.00	$-5,960.00	-42.57%
11	Jul-18	2018	7	Electronics	$12,826.00	$9,000.00	$3,826.00	42.51%
12	Jul-18	2018	7	Furniture	$10,498.00	$10,800.00	$-302.00	-2.80%
13	Aug-18	2018	8	Clothing	$26,540.00	$14,000.00	$12,540.00	89.57%
14	Aug-18	2018	8	Electronics	$33,480.00	$9,000.00	$24,480.00	272.00%
15	Aug-18	2018	8	Furniture	$30,518.00	$10,900.00	$19,618.00	179.98%

Furniture has experienced under-performance for the business, not meeting sales targets for three consecutive months from May through July of 2018. However, this may be due to early-summer seasonality as it exceeded targets by nearly 180% in August.

6. Identify Trends in Product Category Performance // Which Categories Show Growth or Decline

```sql
--Identify trends in product category performance
--Determine which categories are showing the most significant growth or decline

with sales_data as (
    select
          d.category
        , to_char(l.order_date, 'Mon-YY') as month_year
        , sum(d.amount) as total_sales
        , date_trunc('month', l.order_date) as month_date
    from
        list_of_orders l
    join
        order_details d on l.order_id = d.order_id
    where
        1=1
        and l.order_date >= date '2018-01-01' --start from beginning of 2018
        and l.order_date < date '2020-01-01' --end before 2020
    group by
          d.category
        , month_year
        , month_date
),
category_performance as (
    select
          category
        , month_year
        , total_sales
        , month_date
        , lag(total_sales) over (partition by category order by month_date) as previous_month_sales
    from
        sales_data
)
select
      category
    , month_year
    , to_char(total_sales, '$FM999,999,999.00') as total_sales
    , to_char(previous_month_sales, '$FM999,999,999.00') as previous_month_sales
    , case
        when previous_month_sales is null then null
        else round(((total_sales - previous_month_sales) / nullif(previous_month_sales, 0)) * 100, 2) || '%'
    end as percent_change
from
    category_performance
order by
      category
    , month_date
;
```

This analysis aims to identify trends in sales performance for different product categories. By examining both the total sales amount and the growth rate, stakeholders can make informed decisions regarding inventory, marketing focus, and potential areas for expansion or reduction. The CTE “sales_data” creates a temporary result set that pull together essential sales data grouped by category and month. It joins the list_of_orders table with the order_details table on the order_id to combine sales data with order information. The second CTE “category_performance” creates another result set that builds on sales_data with the lag function retrieving the total sales for the previous month within each category. We get a data output that shows trends in product categories over time, allowing us to track

the monthly sales for each category, compare them to the previous month's sales, and calculate the percentage change. This is useful for identifying which categories are growing or declining in performance.

	category character varying (255)	month_year text	total_sales text	previous_month_sales text	percent_change text
1	Clothing	Jan-18	$5,355.00	[null]	[null]
2	Clothing	Feb-18	$1,662.00	$5,355.00	-68.96%
3	Clothing	Mar-18	$3,521.00	$1,662.00	111.85%
4	Clothing	Apr-18	$10,743.00	$3,521.00	205.11%
5	Clothing	May-18	$6,986.00	$10,743.00	-34.97%
6	Clothing	Jun-18	$7,238.00	$6,986.00	3.61%
7	Clothing	Jul-18	$4,020.00	$7,238.00	-44.46%
8	Clothing	Aug-18	$13,270.00	$4,020.00	230.10%
9	Clothing	Sep-18	$8,414.00	$13,270.00	-36.59%
10	Clothing	Oct-18	$11,761.00	$8,414.00	39.78%
11	Clothing	Nov-18	$15,862.00	$11,761.00	34.87%
12	Clothing	Dec-18	$5,769.00	$15,862.00	-63.63%
13	Clothing	Jan-19	$11,867.00	$5,769.00	105.70%
14	Clothing	Feb-19	$4,507.00	$11,867.00	-62.02%
15	Clothing	Mar-19	$14,411.00	$4,507.00	219.75%
16	Clothing	Apr-19	$2,654.00	$14,411.00	-81.58%
17	Clothing	May-19	$1,503.00	$2,654.00	-43.37%
18	Clothing	Jun-19	$1,084.00	$1,503.00	-27.88%
19	Clothing	Jul-19	$1,195.00	$1,084.00	10.24%
20	Clothing	Aug-19	$2,149.00	$1,195.00	79.83%
21	Clothing	Sep-19	$1,371.00	$2,149.00	-36.20%
22	Clothing	Oct-19	$3,077.00	$1,371.00	124.43%
23	Clothing	Nov-19	$522.00	$3,077.00	-83.04%
24	Clothing	Dec-19	$113.00	$522.00	-78.35%
25	Electronics	Jan-18	$7,624.00	[null]	[null]
26	Electronics	Feb-18	$3,763.00	$7,624.00	-50.64%
27	Electronics	Mar-18	$3,345.00	$3,763.00	-11.11%
28	Electronics	Apr-18	$8,857.00	$3,345.00	164.78%
29	Electronics	May-18	$8,759.00	$8,857.00	-1.11%
30	Electronics	Jun-18	$6,546.00	$8,759.00	-25.27%

According to results we can see that clothing sales trended down from April 2019 to June 2019 perhaps signaling a review of marketing spend to draw new or returning customers in early-summer.

This SQL project provided a comprehensive analysis of sales data, identifying trends, growth, and performance of product categories. Thank you for taking the time to explore.