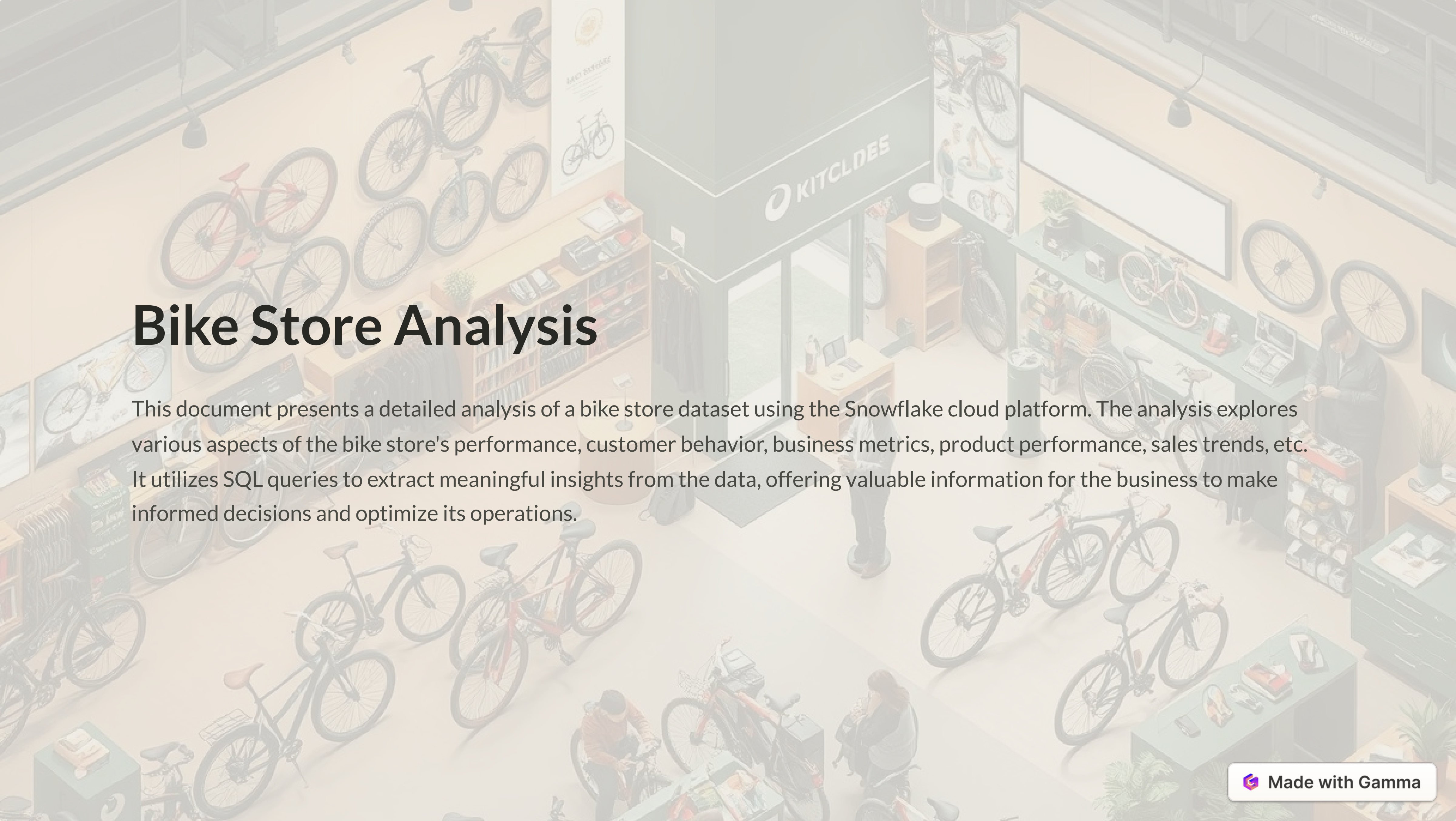

Bike Store Analysis

This document presents a detailed analysis of a bike store dataset using the Snowflake cloud platform. The analysis explores various aspects of the bike store's performance, customer behavior, business metrics, product performance, sales trends, etc. It utilizes SQL queries to extract meaningful insights from the data, offering valuable information for the business to make informed decisions and optimize its operations.

Introduction to the Data Set

The bike store dataset comprises a collection of tables that provide comprehensive information about the business's operations. These tables include details on categories, brands, products, stocks, customers, stores, staffs, orders, and order_items. The data spans a significant period, allowing for the analysis of historical trends and patterns. By analyzing this dataset, we can gain valuable insights into customer demographics, product popularity, sales performance, and other key business metrics.

Questions and Queries

This section outlines a series of questions that we seek to answer through our data analysis. Each question is accompanied by the corresponding SQL query used to retrieve the necessary data. The queries are designed to extract specific information related to customer behavior, sales trends, product performance, and other key aspects of the bike store's business. By examining the results of these queries, we can gain a deeper understanding of the data and identify valuable insights.

Question 1

What are the total sales for each product category?

```
SELECT
   pro.category_id,
   round(sum(ite.quantity * ite.list_price * (1 - ite.discount)),2) as
Total_sales
FROM
   sales.order_items ite
JOIN
   production.products pro
ON
   ite.product_id = pro.product_id
GROUP BY
   pro.category_id
ORDER BY
   total_sales desc;
```

Question 2

Which product has the highest sales revenue?

```
SELECT
    pro.product_name,
    round(sum(ite.quantity * ite.list_price * (1-ite.discount)), 2) Revenue
FROM
    sales.order_items ite
JOIN
    production.products pro
ON
    ite.product_id = pro.product_id
GROUP BY
    product_name
ORDER BY
    Revenue
Desc limit 1
;
```

Question 3

How many unique products do we have per category and brand?

```
SELECT
    cat.category_name,
    cat.category_id,
    br.brand_id,
    COUNT(distinct pro.product_name) as total_uni_pro
FROM
    production.products pro
JOIN
    production.categories cat
ON
    pro.category_id = cat.category_id
JOIN
    production.brands br
ON
    pro.brand_id = br.brand_id
GROUP BY
    cat.category_id,
    cat.category_name,
    br.brand_id
ORDER BY
    total_uni_pro
Desc ;
```

Question 4

Which store has the highest total sales revenue?

```
SELECT
   sto.store_id,
   sto.store_name,
   SUM(ite.quantity * ite.list_price * (1- ite.Discount)) as Total_sales_revenue
FROM
   sales.order_items ite
JOIN
   sales.orders ord
ON
   ite.order_id = ord.order_id
JOIN
   sales.stores sto
ON
   ord.store_id = sto.store_id
GROUP BY
   sto.STORE_ID,
   sto.store_name
ORDER BY
   sto.store_id
DESC LIMIT 1;
```

Question 5

What is the total number of orders per store?

```
SELECT
   Sto.store_id,
   COUNT(ord.order_id) as total_orders
FROM
   sales.orders ord
JOIN
   sales.stores sto
ON
   ord.store_id = sto.store_id
GROUP BY
   sto.store_id;
```

Question 6

How many unique customers have purchased in the last year and have more than a 15% discount?

```
SELECT
    count(distinct(o.customer_id)) as customer_without_discount
FROM
    sales.orders o
JOIN
    sales.order_items oi
ON
    o.order_id = oi.order_id
WHERE
    YEAR(o.order_date) = 2017 and oi.discount > 0.15
order by
    o.customer_id;
```

Question 7

What is the distribution of customers across different stores?

```
SELECT
    store_id,
    count(distinct customer_id) customers_across_each_store
from
    sales.orders
GROUP BY
    store_id;
```

Question 8

What are the total sales per month for the last 12 months?

```
SELECT
   DATE_TRUNC(Month ,o.order_date) as sales_month,
   SUM(round(oi.list_price * oi.quantity * (1- oi.discount),2)) Total_sales
FROM
   sales.orders o
JOIN
   sales.order_items oi
ON
   o.order_id = oi.order_id
WHERE
   o.order_date between '2018-01-01' and '2018-12-28'
GROUP BY
   sales_month
ORDER BY
   Total_sales
Desc;
```

Question 9

Which month had the highest sales across each year?

```
with cte as
(SELECT
    DATE_TRUNC(Month ,o.order_date) as sales_month,
    year(o.order_date) as year_of_order,
    SUM(round(oi.list_price * oi.quantity * (1- oi.discount),2)) Total_sales,
    rank() over(partition by year(o.order_date) order by total_sales desc) as ranked_sales
FROM
    sales.orders o
JOIN
    sales.order_items oi
ON
    o.order_id = oi.order_id
WHERE
    o.order_date between '2016-01-01' and '2018-12-28'
GROUP BY
    sales_month,
    year_of_order
    )
    select
        sales_month,
        total_sales
    from
        cte
    where
        ranked_sales = 1
    order by
        total_sales
    desc;
```

Question 10

How do sales vary by day of the week across all stores?

```
SELECT
    o.store_id,
    dayofweek(o.order_date) day_of_week,
    CASE
        WHEN DAYOFWEEK(o.order_date) = 0 THEN 'Sunday'
        WHEN DAYOFWEEK(o.order_date) = 1 THEN 'Monday'
        WHEN DAYOFWEEK(o.order_date) = 2 THEN 'Tuesday'
        WHEN DAYOFWEEK(o.order_date) = 3 THEN 'Wednesday'
        WHEN DAYOFWEEK(o.order_date) = 4 THEN 'Thursday'
        WHEN DAYOFWEEK(o.order_date) = 5 THEN 'Friday'
        WHEN DAYOFWEEK(o.order_date) = 6 THEN 'Saturday'
        END as Day_name,
    Round(SUM(oi.list_price * oi.quantity * (1 - oi.discount)),2) weekly_sales_variation
FROM
    sales.orders o
JOIN
    sales.order_items oi
ON
    o.order_id = oi.order_id
GROUP BY
    o.store_id,
    day_of_week
ORDER BY
    o.store_id,
    day_of_week,
    weekly_sales_variation;
```

Question 11

What is the average order size (in terms of quantity and value) per store?

```
with cte as
(SELECT
    o.store_id,
    o.order_id,
    SUM(oi.quantity) as total_quantity,
    SUM(oi.list_price * oi.quantity * (1 - oi.discount)) total_value
FROM
    sales.orders o
JOIN
    sales.order_items oi
ON
    o.order_id = oi.order_id
GROUP BY
    o.order_id,
    o.store_id
    )
SELECT
    s.store_name,
    ROUND(AVG(cte.total_quantity), 2) as avg_order_quantity,
    ROUND(AVG(cte.total_value), 2) as avg_order_value
FROM
    CTE
JOIN
    sales.stores s
ON
    cte.store_id = s.store_id
GROUP BY
    s.store_name
ORDER BY
    avg_order_quantity,
    avg_order_value;
```

Question 12

Which product categories generate the most revenue?

```
SELECT
   c.category_name,
   round(SUM(oi.list_price * oi.quantity * (1-oi.discount)),2) as revenue
FROM
   sales.order_items oi
JOIN
   production.products p
ON
   oi.product_id = p.product_id
JOIN
   production.categories c
ON
   p.category_id = c.category_id
GROUP BY
   c.category_name
ORDER BY
   revenue desc limit 1;
```

Question 13

Are there any brands that consistently underperform in sales?

```
With brand_sales as(
SELECT
    b.brand_name,
    round(sum(oi.list_price * oi.quantity * (1 - oi.discount))) total_sales
FROM
    sales.order_items oi
JOIN
    production.products p
ON
    oi.product_id = p.product_id
JOIN
    production.brands b
ON
    p.brand_id = b.brand_id
GROUP BY
    b.brand_name
    ),
average_sales as (
SELECT
    AVG(total_sales) as  avg_sales
FROM
    brand_sales
)
SELECT
    bs.brand_name,
    bs.total_sales,
    round(av.avg_sales, 0) aver_sales
FROM
    brand_sales bs
CROSS JOIN
    average_sales av
WHERE
    bs.total_sales < av.avg_sales
ORDER BY
    bs.total_sales;
```

Question 14

How do discounts affect the quantity sold for different product categories?

```
WITH category_sales AS (
    SELECT
        c.category_name,
        CASE
            WHEN oi.discount BETWEEN 0 AND 0.1 THEN '0%-10%'
            WHEN oi.discount BETWEEN 0.1 AND 0.2 THEN '10%-20%'
            WHEN oi.discount BETWEEN 0.2 AND 0.3 THEN '20%-30%'
            ELSE '30%+'
        END AS discount_range,
        SUM(oi.quantity) AS total_quantity,
        AVG(oi.quantity) AS avg_quantity
    FROM
        sales.order_items oi
    JOIN
        production.products p
    ON
        oi.product_id = p.product_id
    JOIN
        production.categories c
    ON
        p.category_id = c.category_id
    GROUP BY
        c.category_name,
        discount_range
)
SELECT
    category_name,
    discount_range,
    total_quantity,
    avg_quantity
FROM
    category_sales
ORDER BY
    total_quantity desc;
```

Question 15

Who are the top 10 customers in terms of total purchase value?

```
with cte as
(
SELECT
    c.customer_id,
    concat(c.first_name,' ',c.last_name) customer_name,
    ROUND(sum(oi.quantity * oi.list_price * (1 - oi.discount)),2) as total_purchase_by_each_cust
FROM
    sales.order_items oi
JOIN
    sales.orders o
ON
    oi.order_id = o.order_id
JOIN
    sales.customers c
ON
    o.customer_id = c.customer_id
GROUP BY
    c.customer_id, customer_name
    )
SELECT
    customer_name,
    total_purchase_by_each_cust
FROM
    cte
ORDER BY
    total_purchase_by_each_cust
    desc limit 10;
```

Question 16

What is the average frequency of purchases per customer?

```
SELECT
    round(avg(total_count),2) as order_frequency
FROM(
SELECT
    CONCAT(c.first_name,' ', c.last_name) as customer_name,
    count(o.order_id) as total_count
from
    sales.customers c
JOIN
    sales.orders o
ON
    c.customer_id = o.customer_id
GROUP BY
    customer_name
    );
```

Question 17

How do sales compare across different regions where stores are located?

```
SELECT
    st.state,
    st.city,
    ROUND(SUM(oi.quantity * oi.list_price * (1 - oi.discount)), 2) as total_sales
FROM
    sales.stores st
LEFT JOIN
    sales.orders o
ON
    st.store_id = o.store_id
JOIN
    sales.order_items oi
ON
    o.order_id = oi.order_id
GROUP BY
    st.state,
    st.city
ORDER BY
    total_sales
DESC;
```

Question 18

Are there any regions with declining sales trends?

```
with regional_sales as(
SELECT
    st.state,
    st.city,
    date_trunc(month, o.order_date) as order_month,
    ROUND(SUM(oi.quantity * oi.list_price * (1 - oi.discount)), 2) as total_sales
FROM
    sales.stores st
LEFT JOIN
    sales.orders o
ON
    st.store_id = o.store_id
JOIN
    sales.order_items oi
ON
    o.order_id = oi.order_id
GROUP BY
    st.state,
    st.city,
    order_month
),
sales_trends as (
SELECT
    state,
    city,
    order_month,
    total_sales,
    lag(total_sales) over(partition by state, city order by order_month) prev_month_sales
FROM
    regional_sales
    )
SELECT
    state,
    city,
    count(*) as month_declining
FROM
    sales_trends
WHERE
    total_sales < prev_month_sales
GROUP BY
    state,
    city
HAVING
    month_declining > 0
ORDER BY
    month_declining;
```

Question 19

What is the average shipment time for orders across all stores?

```
with avg_ship_date as(
SELECT
    st.store_id,
    o.order_id,
    o.order_date,
    o.shipped_date,
    DATEDIFF(day, o.order_date, o.shipped_date) date_difference
 FROM
    sales.orders o
JOIN
    SALES.stores st
ON
    o.store_id = st.store_id
GROUP BY
    st.store_id,
    o.order_id,
    o.order_date,
    o.shipped_date
    )
SELECT
    store_id,
    round(avg(date_difference), 2) as avg_shipment_date
FROM
    avg_ship_date
GROUP BY
    store_id;
```

Question 20

Are there any stores that consistently have delayed order processing?

```
WITH store_ship_times AS (
    SELECT
        s.store_id,
        s.store_name,
        DATEDIFF(Day, o.order_date, o.shipped_date) AS ship_time
    FROM
        sales.orders o
    JOIN
        sales.stores s
    ON
        o.store_id = s.store_id
    WHERE
        o.shipped_date IS NOT NULL -- Exclude orders with missing delivery dates
),
average_ship_times AS (
    SELECT
        store_id,
        store_name,
        ROUND(AVG(ship_time), 2) AS avg_ship_time
    FROM
        store_ship_times
    GROUP BY
        store_id, store_name
),
delayed_stores AS (
    SELECT
        store_name,
        avg_ship_time
    FROM
        average_ship_times
    WHERE
        avg_ship_time > (
            SELECT ROUND(AVG(ship_time), 2)
            FROM store_ship_times
        ) -- Compare store-specific delays to the overall average
)
SELECT
    store_name,
    avg_ship_time
FROM
    delayed_stores
ORDER BY
    avg_ship_time DESC; -- Show stores with the highest delays first
```

Question 21

What percentage of customers are repeat buyers?

```
WITH total_buyers as(
SELECT
  customer_id,
  COUNT(DISTINCT order_id) as total_orders
FROM
  SALES.orders
GROUP BY
  customer_id
ORDER BY
  customer_id
  ),
repeat_buyers as(
  SELECT
    COUNT(customer_id) as repeat_buyers
  FROM
    total_buyers
  WHERE
    total_orders > 1
),
total_customers as(
  SELECT
    count(distinct customer_id) as total_customers
  FROM
    sales.orders
)
  SELECT
    ROUND((rc.repeat_buyers / tc.total_customers)*100, 2) repeat_buyers_percentage
  FROM
    repeat_buyers rc
  CROSS JOIN
    total_customers tc;
```

Question 22

Which customer segments respond best to promotional discounts?

```
with customer_purchase as(
    SELECT
        c.customer_id,
        sum(oi.quantity) total_quantity,
        ROUND(SUM(oi.quantity * oi.list_price * (1-oi.discount)), 2) as total_spending,
        Round(AVG(oi.discount), 2) avg_discount
    FROM
        sales.order_items oi
    JOIN
        sales.orders o
    ON
        oi.order_id = o.order_id
    JOIN
        sales.customers c
    ON
        o.customer_id = c.customer_id
    GROUP BY
        c.customer_id
        ),
customer_repeatitivness as (
    SELECT
        customer_id,
        CASE
            WHEN avg_discount = 0 then 'No Discount'
            WHEN avg_discount < 0.1 then 'Low Discount'
            WHEN avg_discount between 0.1 and 0.2 THEN 'Moderate Discount'
            ELSE 'High Discount'
        END as Discount_Segmentation ,
        Sum(total_quantity) total_quantity,
        SUM(total_spending) total_spending
    FROM
        customer_purchase
    GROUP BY
    customer_id, discount_segmentation
)
SELECT
    Discount_Segmentation,
    COUNT(customer_id) num_of_customers,
    sum(total_quantity) as total_quantity,
    sum(total_spending) as total_revenue
FROM
    customer_repeatitivness
GROUP BY
    Discount_Segmentation
order by
    total_revenue desc;
```

Data Analysis and Insights

- **Trek Slash 8 27.5 – 2016** has the highest Sales revenue.
- **Cruisers Bicycles** has the highest unique category of product **64** followed by **Road Bikes** with **53** unique products.
- Store - 3, **Rowlett Bikes** has the highest total sales revenue of $**867542.24**.
- Store – 2 has the highest number of orders **1093** which is **67.89%** of other stores combined,

  because store 2 has the maximum number of customer visits.

- **375** customers made a purchase last year having a discount of more than **15%.**
- The average order value and average order quantity across each store lie between

  4614.43 - 4985.87 and **4.36 – 4.50** respectively.

- The product category **Mountain Bikes** generates the highest revenue of **$2715079.53.**
- There are a few brands like **Strider, Ritchey, Pure Cycles, Heller, Haro, and Sun Bicycles** that are consistently underperforming with sales below average.
- The product category having discounts between 0-10% is the most ordered category amongst other discounts.
- The average frequency of purchase per customer is **1.12.**
- The sales are highest in the state of New York compared to the other 2 states.
- The average shipment time for each store lies between **1.92 and 2.05.**
- **9.07%** of customers are repeat buyers.

Conclusion and Key Takeaways

Analysis reveals Trek Slash 8 27.5 (2016) as the top revenue generator, while Cruisers and Road Bikes boast the most diverse product lines. Store 3 (Rowlett Bikes) leads in total revenue, though Store 2 dominates in order volume. High-discount customers (15%+ discount) numbered 375. Mountain Bikes are the highest-revenue product category. Underperforming brands (Strider, Ritchey, etc.) require attention. The 0-10% discount bracket proved most popular. Average purchase frequency is 1.12 per customer; repeat buyers account for 9.07%. New York leads in sales, with average shipment times between 1.92 and 2.05 across stores. These insights inform strategies for targeted marketing, product development, and operational efficiency.

Thank You

Thank you for taking the time to view my project.

Feel free to reach out with any suggestions.

I hope you learned something new from this.

Connect with me on LinkedIn: **https://www.linkedin.com/in/gaurav-mishra-75b68619a/**

Email: **nilesh0307mishra@gmail.com**

Phone: **+91-9082523687**