Jago digital academy

Jago | DKATALIS

TECHNICAL TEST

Amelia Ananda Setiawan_Data Analyst Intern

Hello,

I’m Amelia Ananda Setiawan

Data Enthusiast

Final-year Digital Business student with with hands-on experience in data analysis. Skilled in **SQL, Python, Google Sheets**, and data visualization tools such as **Looker Studio** and **Tableau**. Experienced in data cleaning, analysis, and creating actionable insights to comprehensive report. Passionate about supporting **data driven business** decisions through impactful analysis.

Identify Repeat Customers with High Spending

1
2
3
4
5
6
7
8
9

Query

```sql
SELECT
  customer_id,
  SUM(list_price) as total_spending,
  COUNT(DISTINCT(transaction_id)) as number_of_transactions
FROM `jago_intern.transactions`
WHERE order_status ="Approved"
GROUP BY customer_id
HAVING number_of_transactions > 5 AND total_spending > 10000
ORDER BY number_of_transactions DESC, total_spending DESC
```

Results

Row	customer_id	total_spending	number_of_transacti
1	2183	19071.32	14
2	2476	14578.69	14
3	1068	14254.55	14
4	1129	18349.27	13
5	1302	17035.83	13
6	1140	16199.24	13

Explanation

SELECT : this query is used to retrieve the colomns customer_id, list_price, and transaction_id
SUM(list_price) as total spending : an agregat function used to sum the list price as total spending
COUNT(DISTINCT(transaction_id)) : COUNT is used as an agregat to calculate transaction_id and DISTINCT ensures that there are no duplicates .
WHERE order_status = approved : this condition filters to include only successful transactions
GROUP BY customer_id : ensure that the data is grouped by customer_id
HAVING number_of_transactions > 5 and total_spending > 10000 : Filters the data to include only customers who have made more than 5 transactions and have total spending over 10,000
ORDER BY number_of_transactions DESC, total_spending DESC : sort the results in descending order

Monthly Sales Trend for Each Brand

1
2
3
4
5
6
7
8
9

Query

```
SELECT
  brand,
  EXTRACT(MONTH FROM t.transaction_date) as month,
  SUM(list_price) as total_sales
FROM `jago_intern.transactions` as t
WHERE brand is not null and order_status ="Approved"
GROUP BY brand, month
ORDER BY brand ASC, month ASC
```

Results

brand	month	total_sales
Giant Bicycles	1	331479.4600000...
Giant Bicycles	2	330016.7399999...
Giant Bicycles	3	290067.4999999...
Giant Bicycles	4	332630.9699999...

Explanation

SELECT : this query retrieves the columns brand, transaction_date, and list_price
EXTRACT(MONTH FROM t.transaction_date) as month : extracts the month from transaction_date
SUM(list_price) as total_sales : an aggregate function that sums the list_price as total_sales
WHERE brand is not null : condition filter to ensure that brand column is not null
and order_status = "Approved" : filters the data to include only successful transaction
GROUP BY brand, month : group the data by brand and month
ORDER BY brand asc, month asc : sort the result in ascending order

Top 3 States by Number of High-Value Customers

1
2
3
4
5
6
7
8
9

Query

```
SELECT
  state,
  COUNT(DISTINCT(customer_id)) as number_of_high_value_customers
FROM `jago_intern.customer_address`
WHERE property_valuation > 10
GROUP BY state
ORDER BY number_of_high_value_customers DESC
LIMIT 3
```

Results

state	number_of_high_valu
NSW	365
VIC	74
New South Wales	22

Explanation

SELECT : this query retrieves the columns state and customer_id
COUNT(DISTINCT(customer_id)) as number_of_high_value_customers : count is used to calculate the number of customer_id and distinct ensures that there are no duplicates
WHERE property_valuation >10 : since the values in the file are in 100k units, the condition must be greater than 1,000,000 (1,000,000 / 100k = 10). The multiplication factor of 100k ensures the condition aligns with the criteria
ORDER BY number_of_high_value_customers DESC : sort the results in descending order
GROUP BY state : groups the data by state
LIMIT 3 : restricts the output to only the top 3 states

Customer Segmentation by Wealth Segment and Order Status

1 2 3 4 5 6 7 8 9

Query

```
SELECT
  d.wealth_segment,
  t.order_status,
  COUNT(DISTINCT(t.customer_id)) as customer_count
FROM `jago_intern.customer_demographics` as d
JOIN `jago_intern.transactions` as t
ON d.customer_id = t.customer_id
WHERE t.order_status = 'Approved'
GROUP BY d.wealth_segment, t.order_status
```

Results

wealth_segment	order_status	customer_
Mass Customer	Approved	1747
High Net Worth	Approved	895
Affluent Custom...	Approved	850

Explanation

SELECT : this query retrieves the columns wealth_segment, order_status, and customer_id
COUNT(DISTINCT(t.customer_id)) as customer_count : count is used to calculate the number of customer_id and distinct ensures that there are no duplicates
FROM customer_demographics as d join transactions as t : join 2 tables, customer_demographics and transactions
ON d.customer_id = t.customer_id : ensures that join is correctly aligned based on customer_id
WHERE t.order_status = 'Approved' : to ensures that only completed orders
GROUP BY d.wealth_segment, t.order_status : groups the data by wealth_segment and order_status

Average Product List Price by Product Line and Brand

1 2 3 4 5 6 7 8 9

Query

```
SELECT
  product_line,
  brand,
  AVG(list_price) as average_list_price
FROM `jago_intern.transactions`
WHERE product_line is not null
GROUP BY product_line, brand
ORDER BY product_line, brand ASC
```

Results

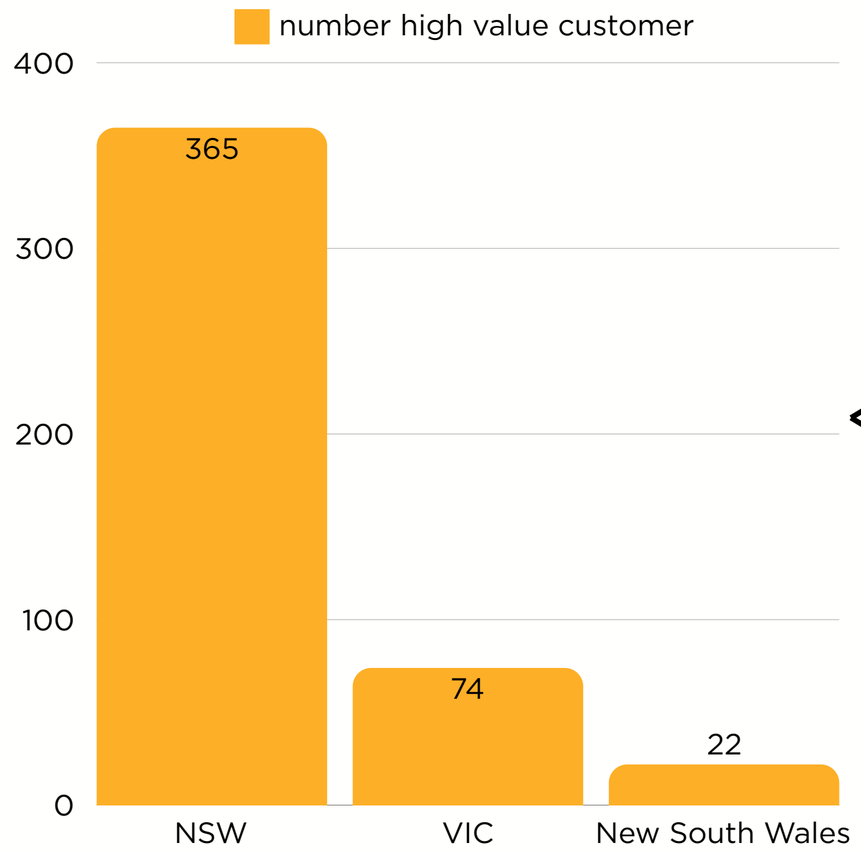
product_line	brand	average_list_price
Mountain	Norco Bicycl...	688.63
Mountain	Trek Bicycles	574.64
Road	Giant Bicycles	951.0941278065...
Road	Norco Bicycl...	817.6463380281...

Results per page: 50 1

Explanation

SELECT: this query retrieves the columns product_line, brand, and list_price
AVG(list_price) as average_list_price: an aggregate function that calculate the average of list_price
WHERE product_line is not null: because there is product_line that null, so this query ensures that the data clean from null.
GROUP BY product_line, brand: Groups the data by product_line and brand
ORDER BY product_line, brand ASC: sorts the results in ascending order based on product_line and brand column

Strategic Focus on High-Value Customers



State with High-Value Customers

Focus Marketing Approach

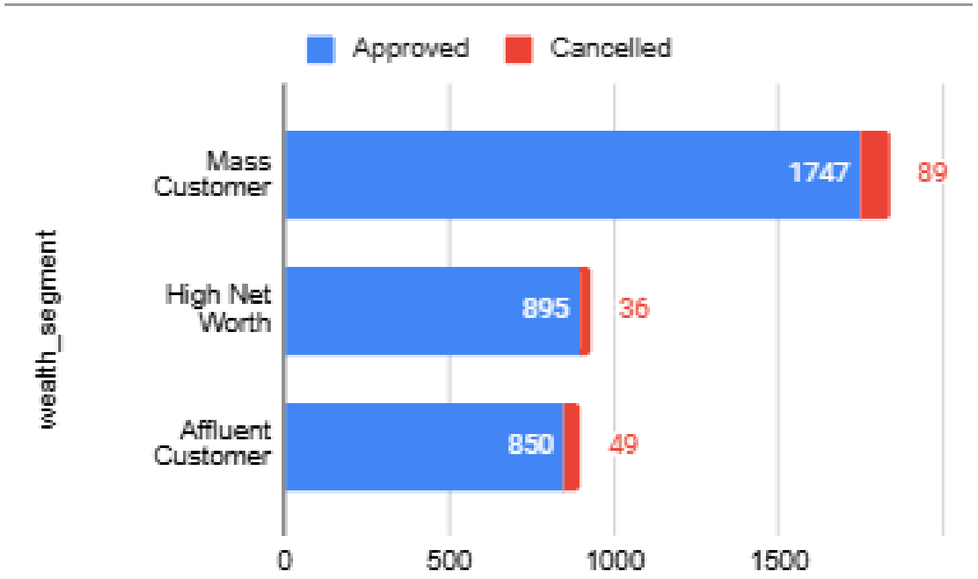
Personalized Marketing

- Use customer data to offer special promotions and premium products
- Exclusive content

Strategic Partnership

- Collaborations with influencer and luxury brands
- Create luxury event with offer priority customer experience

Wealth Segment and Order Status Insights



The **Mass Customer segment** has the highest number of approved orders, reaching **1,747**, making them the **primary driver of sales**. However, they also have the highest number of cancellations (89), indicating **potential issues**.

A similar pattern is seen in the High Net Worth and Affluent Customer segments, with both having over 800 approved orders and more than 35 cancellations. The **Affluent segment** has slightly more cancellations (49), suggesting they might be **more price-sensitive for certain products.**

Strategies for Product and Customer Services:

1. **For Mass Customers** : Maintain their loyalty with bundling or membership program, while investigating the reasons behind high cancellation rate.
2. **For High Net Worth** : Offer premium products and exclusive experience with personalized deals.
3. **For Affluent Customer** : Create loyalty reward program and conduct A/B testing on pricing strategies.

Thank you

Contact Details

Phone : *+6285871954428*

Email : *Amelanandas02@gmail.com*

LinkedIn : *https://www.linkedin.com/in/ameliaanandas/*