RevoGrocers Sales Performance Analysis

Karen Amarillys

25/07/2025
RevoU FSDA Batch Jun25

Disclaimer

1. This analysis is based on a public dataset from Kaggle.
2. RevoGrocers is a fictional company created for analytical purposes.
3. Any insights and recommendations are based on this dataset and do not reflect real-world data.

Product Category vs Revenue vs Total Units Sold

category	rev_after_disc	total_quantity
Confections	556930717.3	11078474
Meat	492888844.7	9719292
Poultry	440025565	9159847
Cereals	427393431.9	8735296
Snails	372084885.2	7199409
Beverages	366515024	7393693
Produce	362861133.5	8174673
Dairy	354358156.8	6815143
Seafood	330527987.5	6996152
Grain	323879126.7	5433152
Shell fish	299598294	6983457

Revenue vs Quantity

Most category follows the same pattern where **the bigger the revenue, the more quantity of products sold** in that category. In this case, **Confections** ranked at **the top** of the list. However, there are some like Produce, Seafood, and Shell fish that don't follow this pattern, possibly due to more discounts or low pricing.

Revenue vs Unique Customers

category	rev_after_disc	total_quantity	unique_cust
Confections	556,930,717.35	11078474	98743
Meat	492,888,844.69	9719292	98701
Poultry	440,025,564.97	9159847	98679
Cereals	427,393,431.91	8735296	98651
Snails	372,084,885.21	7199409	98376
Beverages	366,515,024.01	7393693	98424
Produce	362,861,133.52	8174673	98601
Dairy	354,358,156.77	6815143	98308
Seafood	330,527,987.47	6996152	98334
Grain	323,879,126.65	5433152	97335
Shell fish	299,598,294.03	6983457	98338

Relation on Unique Customers

Generally, **the higher the revenue, the more unique customers** that buys the item. However, this pattern is not always true. It's a **moderate positive relationship**, but not absolute.

Average Price vs Unique Customers

category	avg_price	unique_customers
Confections	51.81190322	98743
Meat	52.31211525	98701
Poultry	49.50921884	98679
Cereals	50.43802736	98651
Produce	45.78994061	98601
Beverages	51.12591016	98424
Snails	53.28068016	98376
Shell fish	44.23266616	98338
Seafood	48.67796017	98334
Dairy	53.61147548	98308
Grain	61.43254359	97335

Relation of Average Price and Number of Unique Customers

There's a **slight negative correlation** where categories with **higher average prices tend to have fewer customers**, but it's **not very strong**

Highest Contributor

category	revenue_per_category	percentage_of_total
Confections	556930717.4	12.87
Meat	492888844.7	11.39
Poultry	440025565	10.17
Cereals	427393431.9	9.88
Snails	372084885.2	8.6
Beverages	366515024	8.47
Produce	362861133.5	8.39
Dairy	354358156.8	8.19
Seafood	330527987.5	7.64
Grain	323879126.7	7.48
Shell fish	299598294	6.92

Category that Contributes the most

Percentage-wise, **Confections** contributes the most to overall revenue as it has the highest revenue among the other categories with **12.87%** contribution. Followed by meat with 11.39% and Poultry with 10.17%.

Repeat Purchase Rate

categoryname	total_customer	repeat_customer	repeat_purchase_rate
Confections	98743	98598	1
Meat	98701	98318	1
Produce	98601	97550	0.99
Cereals	98651	97867	0.99
Poultry	98679	98122	0.99
Beverages	98424	96679	0.98
Shell fish	98338	96054	0.98
Seafood	98334	96138	0.98
Snails	98376	96324	0.98
Dairy	98308	95677	0.97
Grain	97335	91184	0.94

Highest Repeat Purchase Rate

Overall, all product categories have very high purchase rates, where each of them has >90% purchase rate. However, both **Confections and Meat** scored a 100% purchase rate, making them the highest among the others.

Short Summary

Confections consistently leads across all key metrics:

1. Highest total revenue and units sold
2. Largest unique customer base
3. 100% repeat purchase rate, indicating strong loyalty

Cumulative Amount of Transaction of the Top User

customerID	salesID	trans_date	trans_amount	cumulative_amount
94800	239788	2018-04-21 18:06:07.240000 UTC	1,653.09	112,762.29
94800	4264159	2018-04-25 12:45:36.740000 UTC	1,382.43	114,144.72
94800	414118	2018-04-30 01:38:25.270000 UTC	1,452.75	115,597.47
94800	6097579	2018-04-30 08:05:01.100000 UTC	1,405.96	117,003.43
94800	1797766	2018-04-30 19:10:39.000000 UTC	849.95136	117,853.39
94800	4067701	2018-05-01 08:23:34.200000 UTC	904.2984	118,757.68
94800	27028	2018-05-01 19:57:40.370000 UTC	2,255.40	121,013.09
94800	5166766	2018-05-07 01:58:58.550000 UTC	1,416.08	122,429.17
94800	1532654	2018-05-08 07:31:30.780000 UTC	2,325.07	124,754.24
94800	5387467	2018-05-09 04:49:01.960000 UTC	45.8688	124,800.11
94800	6672193	2018-05-09 16:07:12.640000 UTC	1,785.78	126,585.89

Full data here

Thank you

Karen Amarillys
Team 6

Problem 1

```sql
SELECT
  categoryname as category,
  sum(quantity*price*(1-discount)) as rev_after_disc
from fsda-sql-01.grocery_dataset.categories categories
join fsda-sql-01.grocery_dataset.products products
  on categories.categoryid = products.categoryid
join fsda-sql-01.grocery_dataset.sales sales
  on sales.productid = products.productid
group by category
order by rev_after_disc DESC
limit 1
```

Problem 2

```
SELECT
  categoryname as category,
  sum(quantity*price*(1-discount)) as rev_after_disc,
  sum(quantity) as total_quantity
from fsda-sql-01.grocery_dataset.categories categories
join fsda-sql-01.grocery_dataset.products products
  on categories.categoryid = products.categoryid
join fsda-sql-01.grocery_dataset.sales sales
  on sales.productid = products.productid
group by category
order by rev_after_disc DESC
```

Problem 3

```
SELECT
  categoryname as category,
  sum(quantity*price*(1-discount)) as rev_after_disc,
  sum(quantity) as total_quantity,
  count(distinct customerID) as unique_cust
from fsda-sql-01.grocery_dataset.categories categories
join fsda-sql-01.grocery_dataset.products products
  on categories.categoryid = products.categoryid
join fsda-sql-01.grocery_dataset.sales sales
  on sales.productid = products.productid
group by category
order by rev_after_disc DESC
```

Problem 4

```
SELECT
  categoryname as category,
  avg (products.price)
from fsda-sql-01.grocery_dataset.categories categories
join fsda-sql-01.grocery_dataset.products products
  on categories.categoryid = products.categoryid
group by 1
```

Problem 5

```sql
SELECT
  categoryname as category,
  avg (products.price) as avg_price,
  count(distinct CustomerID) as unique_customers
from fsda-sql-01.grocery_dataset.categories categories
join fsda-sql-01.grocery_dataset.products products
  on categories.categoryid = products.categoryid
join fsda-sql-01.grocery_dataset.sales sales
  on products.productID = sales.productID
group by 1
order by 3 DESC
```

Problem 6

```
with category_revenue as (
  SELECT
    categoryname as category,
    sum(quantity*price*(1-discount)) as category_rev,
  from fsda-sql-01.grocery_dataset.categories categories
  join fsda-sql-01.grocery_dataset.products products
    on categories.categoryid = products.categoryid
  join fsda-sql-01.grocery_dataset.sales sales
    on sales.productid = products.productid
    group by 1),

revenue_total as (
  SELECT
    sum(quantity*price*(1-discount)) as total_rev,
  from fsda-sql-01.grocery_dataset.products products
  join fsda-sql-01.grocery_dataset.sales sales
    on sales.productid = products.productid)
```

```
SELECT
  category,
  round(category_rev, 2) as revenue_per_category,
  round(category_rev/total_rev*100, 2) as
percentage_of_total,
from category_revenue
cross join revenue_total
order by percentage_of_total desc
```

Problem 7

```sql
-- count each customer's purchase in each category
with customer_category_count as (
  SELECT
    categoryname,
    customerID,
    count(*) as purchase_count
  from fsda-sql-01.grocery_dataset.categories categories
  join fsda-sql-01.grocery_dataset.products products
    on categories.categoryid = products.categoryid
  join fsda-sql-01.grocery_dataset.sales sales
    on sales.productid = products.productid
  group by 1,2),

-- how many made a repeat purchase in each category
repeat_flags as (
  SELECT
    categoryname,
    customerID,
    purchase_count,
    case when purchase_count >1 then 1 else 0 end is_repeat
  from customer_category_count),

-- calculating repeat purchase rate for each category
repeat_rate as (
  Select
    categoryname,
    count(distinct customerID) as total_customer,
    sum(is_repeat) as repeat_customer,
    round(sum(is_repeat)/count(distinct
customerID),2) as repeat_purchase_rate
  from repeat_flags
  group by categoryname)

Select *
from repeat_rate
order by 4 desc
```

Problem 9

```
--finding the top spender
with total_spending_per_user as (
  select
    customerID,
    sum(quantity*price*(1-discount)) as total_spent
  from fsda-sql-01.grocery_dataset.products products
  join fsda-sql-01.grocery_dataset.sales sales
    on sales.productid = products.productid
  group by customerID
  order by 2 desc
  limit 1),

--listing the top user's transaction
user_transaction as (
  select
    sales.customerID,
    salesID,
    salesdate,
    sum(quantity*price*(1-discount)) as trans_amount
  from fsda-sql-01.grocery_dataset.products products
  join fsda-sql-01.grocery_dataset.sales sales
    on sales.productid = products.productid
  join total_spending_per_user top_spender
    on sales.customerID = top_spender.customerID
    group by sales.customerID, salesID, salesdate)

select
  customerID,
  salesID,
  salesdate as trans_date,
  trans_amount,
  sum(trans_amount) over (order by salesdate) as
cumulative_amount
from user_transaction
where salesdate is not null
order by salesdate
```