**Optimizing Online Sports Retail Revenue**

Sports clothing and athleisure attire is a huge industry, worth approximately $193 billion in 2021 with a strong growth forecast over the next decade!

For this project, we play the role of a data analyst for an online sports clothing company. The company is specifically interested in how it can improve revenue. We will dive into product data such as pricing, reviews, descriptions, and ratings, as well as revenue and website traffic, to produce recommendations for its marketing and sales teams.

The database provided to us, sports, contains five tables, with product\_id being the primary key for all of them:

**retail\_info**

|  |  |  |
| --- | --- | --- |
| column | data type | description |
| product\_name | varchar | Name of the product |
| product\_id | varchar | Unique ID for product |
| description | varchar | Description of the product |

**retail\_finance**

|  |  |  |
| --- | --- | --- |
| column | data type | description |
| product\_id | varchar | Unique ID for product |
| listing\_price | float | Listing price for product |
| sale\_price | float | Price of the product when on sale |
| discount | float | Discount, as a decimal, applied to the sale price |
| revenue | float | Amount of revenue generated by each product, in US dollars |

**retail\_reviews**

|  |  |  |
| --- | --- | --- |
| column | data type | description |
| product\_name | varchar | Name of the product |
| product\_id | varchar | Unique ID for product |
| rating | float | Product rating, scored from 1.0 to 5.0 |
| reviews | float | Number of reviews for the product |

**retail\_traffic**

|  |  |  |
| --- | --- | --- |
| column | data type | description |
| product\_id | varchar | Unique ID for product |
| last\_visited | timestamp | Date and time the product was last viewed on the website |

**retail\_brands**

|  |  |  |
| --- | --- | --- |
| column | data type | description |
| product\_id | varchar | Unique ID for product |
| brand | varchar | Brand of the product |

|  |
| --- |
|  |

**1. Counting missing values**

We will be dealing with missing data as well as numeric, string, and timestamp data types to draw insights about the products in the online store. Let's start by finding out how complete the data is.

|  |
| --- |
| -- Count all columns as total\_rows  -- Count the number of non-missing entries for description, listing\_price, and last\_visited  -- Join info, finance, and traffic  SELECT COUNT(\*) AS "total\_rows", COUNT(description) AS "count\_description", COUNT(listing\_price) AS "count\_listing\_price", COUNT(last\_visited) AS "count\_last\_visited"  FROM retail\_info  JOIN retail\_finance ON retail\_info.product\_id= retail\_finance.product\_id  JOIN retail\_traffic ON retail\_info.product\_id= retail\_traffic.product\_id; |
| |  |  |  |  | | --- | --- | --- | --- | | total\_rows | count\_description | count\_listing\_price | count\_last\_visited | | 3179 | 3117 | 3120 | 2928 | |

**2. Nike vs Adidas pricing**

We can see the database contains 3,179 products in total. Of the columns we previewed, only one last\_visited ; is missing more than five percent of its values. Now let's turn our attention to pricing.

How do the price points of Nike and Adidas products differ? Answering this question can help us build a picture of the company's stock range and customer market. We will run a query to produce a distribution of the listing\_price and the count for each price, grouped by brand.

|  |
| --- |
| -- Select the brand, listing\_price as an integer, and a count of all products in finance  -- Join brands to finance on product\_id  -- Aggregate results by brand and listing\_price, and sort the results by listing\_price in descending order  -- Filter for products with a listing\_price more than zero  SELECT brand, ROUND(listing\_price) AS 'listing\_price', COUNT(\*) AS 'count'  FROM retail\_brands  INNER JOIN retail\_finance ON retail\_brands.product\_id= retail\_finance.product\_id  GROUP BY brand, listing\_price  HAVING listing\_price > 0  ORDER BY listing\_price DESC; |
| |  |  |  | | --- | --- | --- | | brand | listing\_price | count | | Adidas | 300 | 2 | | Adidas | 280 | 4 | | Adidas | 240 | 5 | | Adidas | 230 | 8 | | Adidas | 220 | 11 | | Nike  … | 200 | 1 | |

**3. Labeling price ranges**

It turns out there are 77 unique prices for the products in our database, which makes the output of our last query quite difficult to analyze.

Let's build on our previous query by assigning labels to different price ranges, grouping by brand and label. We will also include the total revenue for each price range and brand.

|  |
| --- |
| -- Select the brand, a count of all products in the finance table, and total revenue  -- Create four labels for products based on their price range, aliasing as price\_category  -- Join brands to finance on product\_id  -- Group results by brand and price\_category, sort by total\_revenue and filter out products missing a value for brand  SELECT brand, COUNT(\*), SUM(revenue) as total\_revenue,  CASE WHEN listing\_price < 42 THEN 'Budget'  WHEN listing\_price >= 42 AND listing\_price < 74 THEN 'Average'  WHEN listing\_price >= 74 AND listing\_price < 129 THEN 'Expensive'  ELSE 'Elite' END AS price\_category  FROM retail\_finance  INNER JOIN retail\_brands  ON retail\_finance.product\_id = retail\_brands.product\_id  GROUP BY brand, price\_category  HAVING brand IS NOT NULL  ORDER BY total\_revenue DESC; |
| brand count total\_revenue price\_category  Adidas 849 4626980.069999999 Expensive  Adidas 1060 3233661.060000001 Average  Adidas 307 3014316.8299999987 Elite  Adidas 359 651661.1200000002 Budget  Nike 357 595341.0199999992 Budget  Nike 82 128475.59000000003 Elite  Nike 90 71843.15000000004 Expensive  Nike 16 6623.5 Average |

**4. Average discount by brand**

Interestingly, grouping products by brand and price range allows us to see that Adidas items generate more total revenue regardless of price category! Specifically, ‘Elite’ Adidas products priced \$129 or more typically generate the highest revenue, so the company can potentially increase revenue by shifting their stock to have a larger proportion of these products!

Note we have been looking at listing\_price so far. The listing\_price may not be the price that the product is ultimately sold for. To understand revenue better, let's take a look at the discount, which is the percent reduction in the listing\_price when the product is actually sold. We would like to know whether there is a difference in the amount of discount offered between brands, as this could be influencing revenue.

|  |
| --- |
| -- Select brand and average\_discount as a percentage  -- Join brands to finance on product\_id  -- Aggregate by brand  -- Filter for products without missing values for brand  SELECT brand, AVG(discount) \*100 AS "average\_discount"  FROM retail\_brands  JOIN retail\_finance ON retail\_brands.product\_id= retail\_finance.product\_id  GROUP BY brand  HAVING brand IS NOT NULL; |
| |  |  | | --- | --- | | brand | average\_discount | | Adidas | 33.452427184465606 | | Nike | 0 | |

**5. Correlation between revenue and reviews**

Strangely, no discount is offered on Nike products! In comparison, not only do Adidas products generate the most revenue, but these products are also heavily discounted!

To improve revenue further, the company could try to reduce the amount of discount offered on Adidas products, and monitor sales volume to see if it remains stable. Alternatively, it could try offering a small discount on Nike products. This would reduce average revenue for these products, but may increase revenue overall if there is an increase in the volume of Nike products sold.

Now explore whether relationships exist between the columns in our database. We will check the strength and direction of a correlation between revenue and reviews.

|  |
| --- |
| -- Calculate the correlation between reviews and revenue as corr  SELECT @firstValue := AVG(reviews),  @secondValue := AVG(revenue),  @division := (stddev\_samp(reviews) \* stddev\_samp(revenue))  FROM retail\_reviews  JOIN retail\_finance ON retail\_reviews.product\_id= retail\_finance.product\_id;  SELECT SUM((reviews-@firstValue)\*(revenue-@secondValue))/((COUNT(reviews)-1)\*@division) AS review\_revenue\_corr  FROM retail\_reviews  JOIN retail\_finance ON retail\_reviews.product\_id= retail\_finance.product\_id; |
| review\_revenue\_corr  0.6518512283481315 |

**6. Ratings and reviews by product description length**

Interestingly, there is a strong positive correlation between revenue and reviews. This means, potentially, if we can get more reviews on the company's website, it may increase sales of those items with a larger number of reviews.

Perhaps the length of a product's description might influence a product's rating and reviews - if so, the company can produce content guidelines for listing products on their website and test if this influences revenue. Let's check this out!

|  |
| --- |
| -- Calculate description\_length  -- Convert rating to an integer and calculate average\_rating  -- Join info to reviews on product\_id and group the results by description\_length  -- Filter for products without missing values for description, and sort results by description\_length  SELECT TRUNCATE(LENGTH(description), -2) AS description\_length,  ROUND(AVG(rating),2) AS average\_rating  FROM retail\_info  JOIN retail\_reviews ON retail\_info.product\_id = retail\_reviews.product\_id  GROUP BY description\_length  HAVING description\_length IS NOT NULL  ORDER BY description\_length; |
| description\_length average\_rating  0 1.87  100 3.21  200 3.27  300 3.29  400 3.32  500 3.12  600 3.65 |

**7. Reviews by month and brand**

Unfortunately, there doesn't appear to be a clear pattern between the length of a product's description and its rating.

As we know a correlation exists between reviews and revenue, one approach the company could take is to run experiments with different sales processes encouraging more reviews from customers about their purchases, such as by offering a small discount on future purchases.

Let's take a look at the volume of reviews by month to see if there are any trends or gaps we can look to exploit.

|  |
| --- |
| -- Select brand, month from last\_visited, and a count of all products in reviews aliased as num\_reviews  -- Join traffic with reviews and brands on product\_id  -- Group by brand and month, filtering out missing values for brand and month  -- Order the results by brand and month  SELECT brand, MONTH(last\_visited) AS 'month', COUNT(retail\_reviews.product\_id) AS 'num\_reviews'  FROM retail\_reviews  JOIN retail\_brands ON retail\_reviews.product\_id= retail\_brands.product\_id  JOIN retail\_traffic ON retail\_reviews.product\_id= retail\_traffic.product\_id  GROUP BY brand, month  HAVING brand IS NOT NULL  AND month IS NOT NULL  ORDER BY brand, month; |
| |  |  |  | | --- | --- | --- | | brand | month | num\_reviews | | Adidas | 1.0 | 253 | | Adidas | 2.0 | 272 | | Adidas | 3.0 | 269 | | Adidas | 4.0 | 180 | | … |  |  | |

**8. Footwear product performance**

Looks like product reviews are highest in the first quarter of the calendar year, so there is scope to run experiments aiming to increase the volume of reviews in the other nine months!

So far, we have been primarily analyzing Adidas vs Nike products. Now, let's switch our attention to the type of products being sold. As there are no labels for product type, we will create a Common Table Expression (CTE) that filters description for keywords, then use the results to find out how much of the company's stock consists of footwear products and the median revenue generated by these items.

|  |
| --- |
| -- Create the footwear CTE, containing description and revenue  -- Filter footwear for products with a description containing %shoe%, %trainer, or %foot%  -- Also filter for products that are not missing values for description  -- Calculate the number of products and median revenue for footwear products  WITH median\_calculation AS  (  SELECT  revenue,  @rownum:=@rownum+1 AS x,  @total\_rows:=@rownum  FROM (  SELECT  revenue,  description  FROM retail\_info  JOIN retail\_finance ON retail\_finance.product\_id = retail\_info.product\_id  WHERE description IS NOT NULL  AND revenue IS NOT NULL  AND description LIKE '%shoe%'  OR description LIKE '%trainer%'  OR description LIKE '%foot%'  ORDER BY revenue  ) footwear\_selection  , (SELECT @rownum:=0) tab  WHERE revenue is NOT NULL  ORDER BY revenue  )  SELECT AVG(revenue) AS median\_footwear, @total\_rows AS number\_footwear  FROM median\_calculation  WHERE x IN ( FLOOR((@total\_rows+1)/2), FLOOR((@total\_rows+2)/2) ) |
| median\_footwear number\_footwear  3121.02 2700 |

**9. Clothing product performance**

Recall from the first task that we found there are 3,117 products without missing values for description. Of those, 2,700 are footwear products, which accounts for around 85% of the company's stock. They also generate a median revenue of over $3000 dollars!

This is interesting, but we have no point of reference for whether footwear's median\_revenue is good or bad compared to other products. So, for our final task, let's examine how this differs to clothing products. We will re-use footwear, adding a filter afterward to count the number of products and median\_revenue of products that are not in footwear.

|  |
| --- |
| -- Copy the footwear CTE from the previous task  -- Calculate the number of products in info and median revenue from finance  -- Inner join info with finance on product\_id  -- Filter the selection for products with a description not in footwear  WITH median\_calculation AS  (  SELECT  revenue,  @rownum:=@rownum+1 AS x,  @total\_rows:=@rownum  FROM (  WITH footwear AS  (  SELECT  revenue,  description  FROM retail\_info  JOIN retail\_finance ON retail\_finance.product\_id = retail\_info.product\_id  WHERE description IS NOT NULL  AND revenue IS NOT NULL  AND description LIKE '%shoe%'  OR description LIKE '%trainer%'  OR description LIKE '%foot%'  ORDER BY revenue  )  SELECT revenue, description  FROM retail\_info  JOIN retail\_finance ON retail\_finance.product\_id = retail\_info.product\_id  WHERE description NOT IN (SELECT description FROM footwear)  ) footwear\_selection  ,(SELECT @rownum:=0) tab  WHERE revenue is NOT NULL  ORDER BY revenue  )  SELECT AVG(revenue) AS median\_footwear, @total\_rows AS number\_footwear  FROM median\_calculation  WHERE x IN ( FLOOR((@total\_rows+1)/2), FLOOR((@total\_rows+2)/2) ) |
| median\_clothing number\_clothing  503.82 417 |