**ST1501 CA2 Group Task**

**Class: DAAA/FT/2A/04**

**Group No: 1**

**Group Members:**

|  |  |
| --- | --- |
| **AdmNo** | **Name** |
| **P2112790** | **Jayden Yap** |
| **P2112604** | **Yao Hui** |
| **P2100746** | **Chen ZhuoFan** |
| **P2112646** | **Justin Wong** |

**Group Task**

Data Warehouse Schema

|  |
| --- |
| STAR SCHEMA  C:\Users\p2112790\OneDrive - Singapore Polytechnic\Desktop\DENG\CA2\screenshots\GAME.png |

Queries explanation

|  |  |  |
| --- | --- | --- |
| Query | Explanation | SQL script |
| 1 | What discount rate is most profitable?  We wanted to calculate the net sales of different discount rates to analyse the relationship  between the amount of discount available and net sales per unit.  We can see that the discount percentage that yields the greatest net sales per unit is actually  a 7% discount. This may be due to the fact that customers may see many 5% discounts around but  rarely a 7% discount, and are more inclined to puchase items with a higher discount. Discounts of  10-20% may only be available for higher-priced products, and therefore the net sales may not be as  high.  However, the difference in average list price per unit is about 50 dollars. This means that discount rate has no effect on number of sales.  Therefore, Upper Management should look into experimenting different list prices with different discount values, and prioritise discounting items at 7% to maximise net sales. | SELECT      discount\*100 AS 'Discount (%)',      sum(price\*(discount)\*quantity) AS Loss,      sum(price\*quantity) AS Revenue,      sum(quantity) AS Sales,      sum(price\*quantity) -sum(price\*(discount)\*quantity)  AS 'Net Sales',      (sum(price\*quantity) -sum(price\*(discount)\*quantity) )/sum(quantity) AS 'Net Sales Per Unit',      avg(price) 'Average List Price'  FROM [dbo].Sales\_Fact  JOIN [Time] t ON sales\_Fact.ORDERDate=t.TimeKey  WHERE ORDER\_status=4  GROUP BY discount  ORDER BY 'Net Sales Per Unit' DESC |
| 2 | We wanted to find out who was the best performing employee over the span of the 4 years, and per season.  This is due to the fact that if we take seasonality into account, different employees may perform differently  over different quarters of the year due to the individual strengths and weaknesses. E.g. Winter may require  employees to come earlier, Summer may require the employees to stay later as the demand for bikes are higher  due to summer actitivities.  For the first year, we can see that the best performing employee was Boyer Marcelene, with a revenue of  $149,569.03 earned and 260 sales.  For the second year, we can see that the best performing employee was Boyer Marcelene, with a revenue of  $289,539.56 earned and 337 sales.  For the third year, we can see that the best performing employee was Boyer Marcelene, with a revenue of  $231,384.49 earned and 263 sales.  Therefore, we should promote and/or increase the salary of Boyer Marcelene to further encourage them to sell more bikes | SELECT      top 1 with ties  [Year],Quarter,sum(quantity) AS Sales,      sum(price\*(1-discount)) AS Revenue,      last\_name as Last\_name,      first\_name as First\_name,      Sales\_Fact.storekey as StoreKey,      store\_name as Store\_name  FROM Sales\_fact  LEFT JOIN Store ON store.StoreKey=Sales\_Fact.StoreKey  INNER JOIN staff ON staff.StaffKey=Sales\_Fact.StaffKey  LEFT JOIN [Time] ON [Time].TimeKey=Sales\_Fact.ORDERDate  GROUP BY Sales\_fact.StaffKey,[year],quarter,last\_name,first\_name,Sales\_Fact.StoreKey,store\_name  ORDER by (row\_number() over (partition by quarter,[year] ORDER by sum(quantity) desc)) |
| 3 | Building on query 2, we wanted to analyse the impact of seasonality on the revenue gained.  For Spring, the revenue earned was the most stable, at a steadily increasing 52.8% from the first year into  the second year, and another 43.5% increase from the second year into the third.  For Summer, we can see that the revenue remained stable for the first 2 years, increasing by 38.5%. However,  this was accompanied by a stunning 97.5% loss in the third year, suggesting that there was either a major  natural disaster that occured, a burst in the market bubble, or lack of products available.  For Fall, this was also the exact same case, the revenue remained stable for the first 2 years, but  suddenly plunged by a whopping 97%.  For Winter, we can observe a trend where the revenue is volatile, increasing by 53% in the second year but  dropping 30% in the third year.  As such, we recommend Upper Management to stock up and offer discounts such as “Back to Summer Sale” during the Spring period. This maximises and doubles down on profits as consumers will be purchasing new bicycles to ride with friends and family heading into Summer Holidays | SELECT      year AS 'Year',      season AS 'Season',      FORMAT(sum(price\*(1-discount)),'C2', 'us-US') AS 'Current\_Revenue',      FORMAT(LAG(sum(price\*(1-discount)),4) OVER (          ORDER BY year, CASE season               WHEN 'Spring' THEN 1               WHEN 'Summer' THEN 2               WHEN 'Fall' THEN 3               WHEN 'Winter' THEN 4               ELSE 5  END      ),'C2', 'us-US') 'Previous Revenue',      FORMAT(ROUND((sum(price\*(1-discount))-LAG(sum(price\*(1-discount)),4) OVER (          ORDER BY year,               CASE season                   WHEN 'Spring' THEN 1                   WHEN 'Summer' THEN 2                   WHEN 'Fall' THEN 3                   WHEN 'Winter' THEN 4                   ELSE 5               END)  )       / LAG(sum(price\*(1-discount)),4) OVER (          ORDER BY year,               CASE season                   WHEN 'Spring' THEN 1                   WHEN 'Summer' THEN 2                   WHEN 'Fall' THEN 3                   WHEN 'Winter' THEN 4                   ELSE 5              END      ), 3),'P') AS '% Increase FROM Previous Year'  FROM [Time]  JOIN Sales\_Fact SF ON Time.TimeKey = SF.ORDERDate  GROUP BY year, season  ORDER by YEAR,          CASE season               WHEN 'Spring' THEN 1               WHEN 'Summer' THEN 2               WHEN 'Fall' THEN 3               WHEN 'Winter' THEN 4               ELSE 5          END |
| 4 | Which state is the most popular and profitable for Trek Bicycle Store Inc. ?  Returns:  1. Year  2. US State (3 states)  3. Number of Customers  4. Revenue($)  The state with most customers is New York , with 327 customers. New York also makes the most revenue, in 2017, it generated more than $2 million.Throughout the 3 years of data, New York was best performing, followed by California and then Texas.  Since we have 1 store in each of the 3 states, we can conclude New York has the best exposure for customers.  We did this query by Grouping by both Year and State. Followed by calculating Customers as well as Revenue from SalesFacts.  This query allows the user to know which state to open a new store in, so as to maximise the profits , as well as customer exposure per store.  As such, we recommend Upper Management to open expand their scope and build more stores in states like New York to capture more customers and generate more revenue | SELECT      t.year AS 'Year',      C.State AS 'US State',      COUNT(C.CustomerKey) AS 'Customers',      CAST(SUM(S.Quantity\*S.Price\*(1-S.Discount)) AS DECIMAL(18,2)) AS 'Revenue($)'  FROM      sales\_fact S,      customer C,      time t  WHERE      --link FKs      S.CustomerKey=C.CustomerKey      AND S.ORDERDate=T.TimeKey      -- only get completed orders      AND S.ORDER\_status=4  GROUP BY t.year,C.state  ORDER BY t.year DESC, 'Customers' DESC,'Revenue($)' DESC |
| 5 | Query 5: What are our most popular product categories?  Returns:  1. Product Category  2. Stocks (available)  3. Sales  4. Total Stock (sold or available)  5. Percent Sold (%)  6. Revenue ($)  Mountain bikes have the most revenue with ~$2.7million. 39% of all stocks were sold with 1755 total sales.  2nd highest revenue category was Road Bikes, likely due to them being the most expensive bike type on average. Despite Crusiers having the most sales (2063), it was only 3rd highest revenue. This is likely due to the difference in average unit price ($484 compared to mountain bikes $1545)  This query was achieved with multiple join statements with subqueries so that calculations were accurate.  This query allows the user to know which product categories are most worth to purchase stocks of. It also allows for easier management of the current stocks of each category.  We recommend Upper Management to stock up on Mountain Bikes and Cyclocross Bicycles.  This is due to Mountain Bikes generating the most revenue and Cyclocross bicycles selling the most of their stock.  We should take note of Road Bikes and Electric Bikes, as these bikes only sold 20-22% of their available.  However, it is important to keep the right amount of stock available for Road Bikes as these account for the second most revenue generated by all stores. | SELECT     t.category\_name AS 'Category',     sum(t.total\_stock) AS 'Available Stock',     r.sales AS 'Sales',     sum(t.total\_stock) +r.sales AS 'Total Stock',     r.sales\*100 /(sum(t.total\_stock) +r.sales ) AS 'Percent Sold (%)' ,     pri.AveragePrice AS 'Average Price($)',     j.revenue AS 'Revenue($)'  FROM      (SELECT          DISTINCT product\_id,category\_name,          inventory.total\_stock      FROM  inventory      LEFT JOIN Sales\_Fact ON Inventory.InventoryKey=Sales\_Fact.inventorykey      ) t  LEFT JOIN Inventory ON Inventory.product\_id=t.product\_id  LEFT JOIN (              SELECT category\_name,                      sum(quantity) AS sales              FROM Sales\_Fact              LEFT JOIN Inventory ON Sales\_Fact.InventoryKey=Inventory.InventoryKey              GROUP BY category\_name            ) r  ON r.category\_name=t.category\_name  LEFT JOIN (              SELECT category\_name,                      SUM(Quantity\*Price\*(1-Discount)) AS 'Revenue'              FROM Sales\_Fact              INNER JOIN Inventory ON Inventory.InventoryKey=Sales\_Fact.InventoryKey              GROUP BY category\_name            ) j  ON r.category\_name=j.category\_name  LEFT JOIN (              SELECT category\_name,                      AVG(Price\*(1-Discount)) AS 'AveragePrice'              FROM Sales\_Fact              INNER JOIN Inventory ON Inventory.InventoryKey=Sales\_Fact.InventoryKey              GROUP BY category\_name            ) pri  ON j.category\_name=pri.category\_name  GROUP BY t.category\_name,r.sales,j.revenue,pri.AveragePrice  ORDER BY 'Revenue($)' DESC |