DB Assignment 3

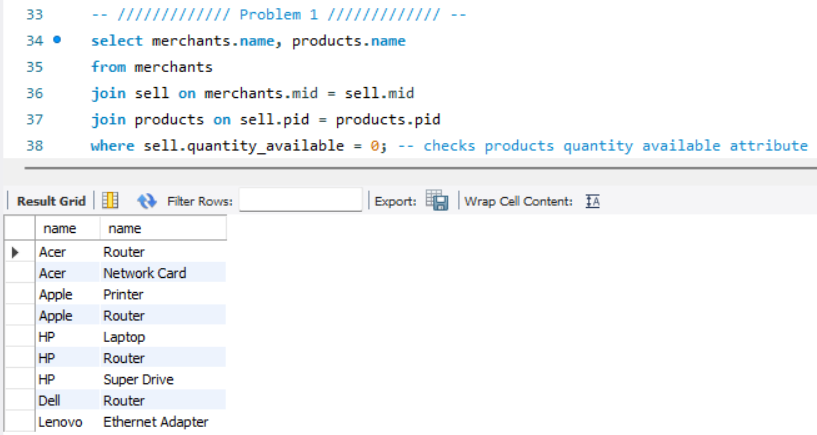
Leonidas Kesaris

10/11/2024

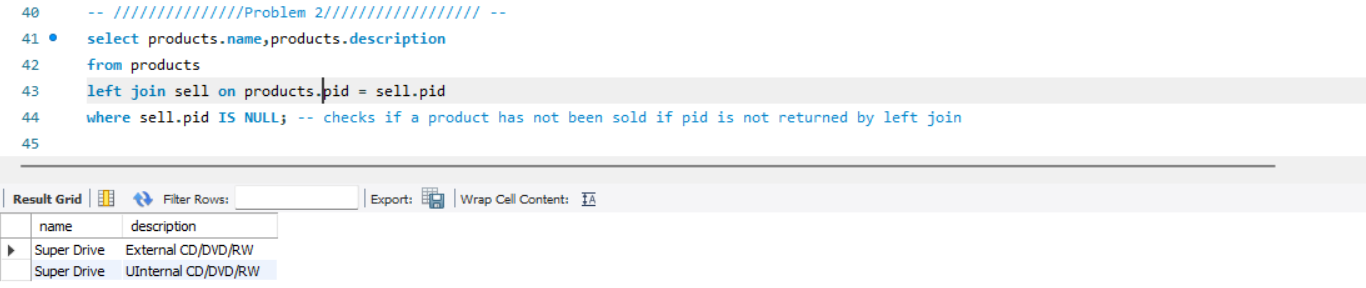
A screenshot of a computer code

Description automatically generated

The above screenshots contain the primary key, foreign key, and constraints given to the tables utilized throughout the assignment. Beginning with the sell table, sell is given a price constraint of 0 – 100,000, and a quantity available constraint of 0-1000. The place table is given a constraint which ensures that the order date is less than the current date. Products is given a constraint which ensures ‘Printer', 'Ethernet Adapter', 'Desktop', 'Hard Drive', 'Laptop', 'Router', 'Network Card', 'Super Drive', 'Monitor' are the only possible product names, and 'Peripheral', 'Networking', 'Computer' are the only categories available. Finally orders is altered so shipping cost are between 0 and 500 and the shipping method is one of UPS FedEX or USPS.



Above is the query and result for problem 1. The query displays merchant name and product name in which the product is still available via joins statements to the sell table and utilizing the sell.quantity\_available column to indicate that a product is no longer available when the value is 0.



Above is the query and result for problem 2. The query left joins the products and sell table, which allows a null value to indicate when a product has not been sold due to being absent in the sell table. By using where sell.pid IS NULL returns all values which have not been sold.

A white background with blue text

Description automatically generated

A close up of a text

Description automatically generated

The above query for problem 3 displays the number of unique customers who have ordered Super Drives and no Routers. The query counts customers Super Drives, then this is filtered to omit customers with CIDs which have purchased Routers.

A white screen with text

Description automatically generated

The above query updates the sell table to apply a 20% sale on the prices of Networking products sold by Merchants with the name HP. The query sets the sell tables price to 80% of current value, and checks that the product ID belongs to a product with the category Networking and a merchant ID of HP.

A screenshot of a computer

Description automatically generated

The above query for problem 6 returns the annual sales for each merchant per year. This is done by converting place.order\_date to a YEAR value and giving it the alias Year, as well as calculating the sum of each product sold via Annual\_Total\_Sales alias. The join statements within the query allow for calculating sales, as by joining the sell, contain, place and merchant table only products which have an order id, indicating they have been sold, will be returned. This allows the query to then group results by merchant and year aggregating the total sales to each merchant per year.

A screenshot of a computer program

Description automatically generated

The above query for problem 7 shows the merchant which had the highest total sales in a year. The result is acquired by joining merchants, sell, contain, and place to only sum values which have been sold. These results are then grouped by merchant and year, which are then ordered to have the highest value at the top and limited to one.

A screen shot of a computer

Description automatically generated

The above query displays the shipping method with the lowest average shipping cost. This is done by displaying the shipping method and the average cost, grouping the values by orders.shipping\_method to find the average of each method, and then ordering by ascending and limiting to one to display the smallest value.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

The above query and results for problem 9 indicate which category for each company has the highest value. This query utilizes several CTEs: first the CategorySales CTE calculates total sales per product category from each merchant. The CTE selects merchant name, product category and sum of price. Merchants, sell, contain and orders are joined to only calculate the sum of products which have been sold. The results are grouped via merchant and category. Next MaxCat CTE identifies the largest value for each merchant’s category sales. This is done by using the MAX function on Total\_Sales, giving this value the alias MaxSales, and grouping by Merchant to ensure that the maximum sale value is calculated per merchant. Finally, CategorySales.Merchant, CategorySales.Category and CategorySales.Total\_Sales are selected to retrieve the name, best selling category, and the sales for the best-selling category. Joining MaxCat and CategorySales on both Merchant and Total\_Sales = MaxSales only allows the query to show rows where the total sales are the maximum sales acquired from the MaxCat CTE. This allows the query to show the category with highest sales per merchant.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The above query for problem 10 calculates the total spending of each customer per merchant and indicates the highest and lowest spenders. There are several CTEs utilized to acquire the results, first aliases are given to customers.cid as custID, merchants.name as Merchant, customers.name as CNAME and the sum of sell price as TotalPrice. Next, CustomerSpending calculates total spending each customer had per merchant. This is done by joining sell, contain, order and place to return only results of sold products. Next the results are grouped by merchant and customer ID to calculate the sum of sales for each customer per merchant. Next the CTE MinMaxSpending calculates the maximum and minimum spending each merchant received. The CTE returns the MaxSpent and MinSpent by using MIN and MAX functions on the TotalSpent values provided by CustomerSpending CTE. Finally, Merchant, custID, CName TotalSpent and SpendingType are selected to be displayed in the result. A CASE Statement is used to create the condition that when a customer is a max or min spender they will be indicated as a Max Spender or Min Spender via the SpendingType column. Next, the CTE’s MinMaxSpending and CustomerSpending are joined to indicate maximum and minimum values of spending for each merchant. Finally, the results are filtered to include only customers who have the SpendingType values of Max Spender or Min Spender and displayed in descending order.