|  |  |
| --- | --- |
|  | Database Assignment 1: Part Two  SQL Queries |
|  | CSC2001F |
|  | Zuhayr Halday (HLDZUH001)  4/16/24 |

# Question 2.1:

## SQL Code

*-- Question 2.1*

*-- Find an available employee number and update the record*

UPDATE

    employees

SET

    employeeNumber = (SELECT MAX(employeeNumber) + 1 FROM employees)

WHERE

    employeeNumber = 1313;

# Question 2.2:

## SQL Code

*-- Question 2.2*

*-- This SQL query provides valuable information for understanding the sales performance of different product types in the database.*

*-- It shows which types of product sells the most in terms of amount of money made, as many important business conclusions can be drawn from this data.*

SELECT

    pl.productLines, *-- Selects the product line*

    CONCAT('R', FORMAT(SUM(od.quantityOrdered \* od.priceEach), 2)) AS totalSalesAmount

*-- Calculates the total sales amount for each product line and formats it as currency*

FROM

    products p

JOIN

    orderdetails od ON p.productCode = od.productCode *-- Joins products and orderdetails tables based on product code*

JOIN

    productlines pl ON p.productLine = pl.productLines *-- Joins productlines table to get the product line name*

WHERE

    p.productCode IN (

        SELECT

            productCode

        FROM

            orderdetails

        GROUP BY

            productCode

        HAVING

            SUM(quantityOrdered) > 50

    )

*-- Filters the products to only those with total quantity ordered exceeding 50*

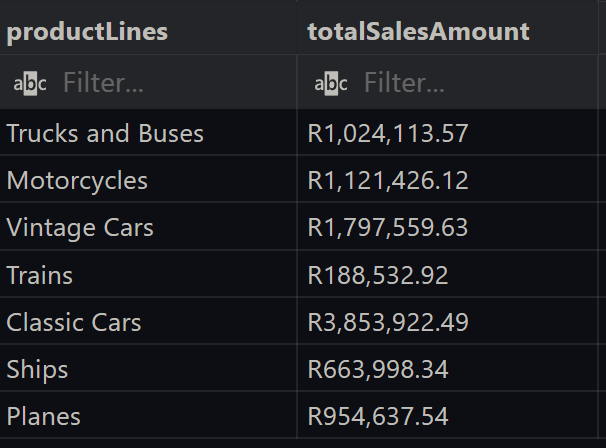
GROUP BY

    pl.productLines *-- Groups the results by product line*

ORDER BY

    totalSalesAmount ASC; *-- Orders the result set by total sales amount in ascending order*

## Results of SQL Query



# Question 2.3:

## SQL Code

As shown in the above graphs, both the Insert and Search algorithms of the AVL tree have a worst case and average case time complexity similar to O(log n), as the log curves in each graph can easily be seen. While the best case time complexity for the AVL Insert does in fact correspond to O(1), the same cannot be said for that of the AVL Search.

While O(1) is theoretically the actual best case time complexity for an AVL Search, in practice there are many occasions where a randomized dataset will not have a search term as the first node of an AVL tree, especially for extremely large datasets. This can be seen in the graphs above, as the experiments show the best case of an AVL Search to also be roughly O(log n).

Despite this minor difference, this experiment can be considered a success. All fair-testing practices were maintained, and the practical results of these calculated time complexities almost all agree with the theoretical Big O time complexities for each algorithm.

# Implementation of Creativity

While conducting the experiment by manually changing the dataset for each test was always an option, creativity was used when writing a completely new Java program to assist in automating the entire process in order to create a fair and efficient experiment. Using ArrayLists in order to calculate the minimum, maximum and average number of comparisons greatly improved the accuracy of results and writing these results neatly to an *experimentation.txt* file was also going above and beyond what was required. The implementation of these creative additions not only added more content to the assignment, but also served a functional role in obtaining extremely accurate results from the experiment.