1) List all purchases made by customers. Your query should return the Cus\_Code, Inv\_Number, Inv\_Date, P\_Descript, Line\_Units and Line\_Price. Sort by Customer code, invoice number, and product description in that order.

SELECT

CUSTOMER.CUS\_CODE,

INVOICE.INV\_NUMBER,

INVOICE.INV\_DATE,

PRODUCT.P\_DESCRIPT,

LINE.LINE\_UNITS,

LINE.LINE\_PRICE

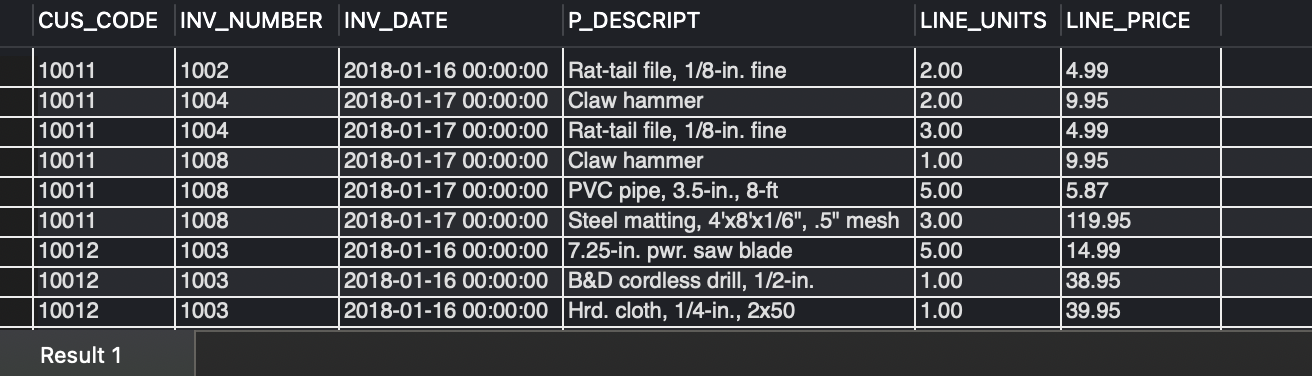
FROM CUSTOMER

JOIN INVOICE ON CUSTOMER.CUS\_CODE = INVOICE.CUS\_CODE

JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER

JOIN PRODUCT ON LINE.P\_CODE = PRODUCT.P\_CODE

ORDER BY CUSTOMER.CUS\_CODE, INVOICE.INV\_NUMBER, PRODUCT.P\_DESCRIPT;

Query 2: Generate a list of customer purchases, including subtotals for the invoice line numbers. Subtotals are a derived attribute, calculated by multipying Line\_Units by Line\_Price. Columns displayed should be Cus\_Code, Inv\_Number, P\_Description, Units Bought (this is an alias), Unit Price (alias) and Subtotal (alias). It is up to you to determine which columns you will use to create the aliases.

SELECT

CUSTOMER.CUS\_CODE, INVOICE.INV\_NUMBER, PRODUCT.P\_DESCRIPT,

LINE.LINE\_UNITS AS `Units Bought`,

LINE.LINE\_PRICE AS `Unit Price`,

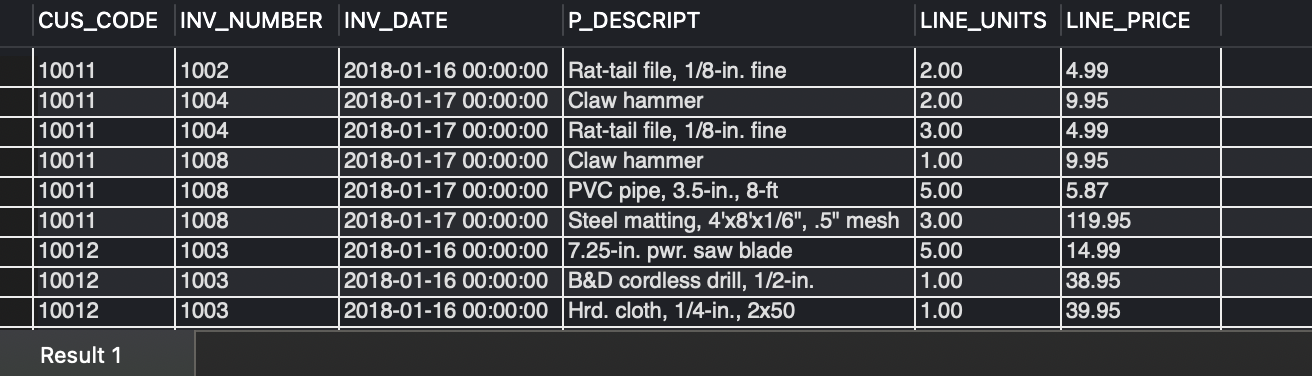
(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Subtotal`

FROM CUSTOMER

JOIN INVOICE ON CUSTOMER.CUS\_CODE = INVOICE.CUS\_CODE

JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER

JOIN PRODUCT ON LINE.P\_CODE = PRODUCT.P\_CODE;



Query 3: Improve the query in question 2 by displaying Cus\_LName, Cus\_FName instead of Cus\_Code. Use proper, modern JOIN syntax to accomplish this. Add meaningful aliases to all columns.

SELECT

CONCAT(CUSTOMER.CUS\_FNAME, ' ', CUSTOMER.CUS\_LNAME) AS `Customer Name`,

INVOICE.INV\_NUMBER AS `Invoice Number`,

PRODUCT.P\_DESCRIPT AS `Product Description`,

LINE.LINE\_UNITS AS `Units Bought`,

LINE.LINE\_PRICE AS `Unit Price`,

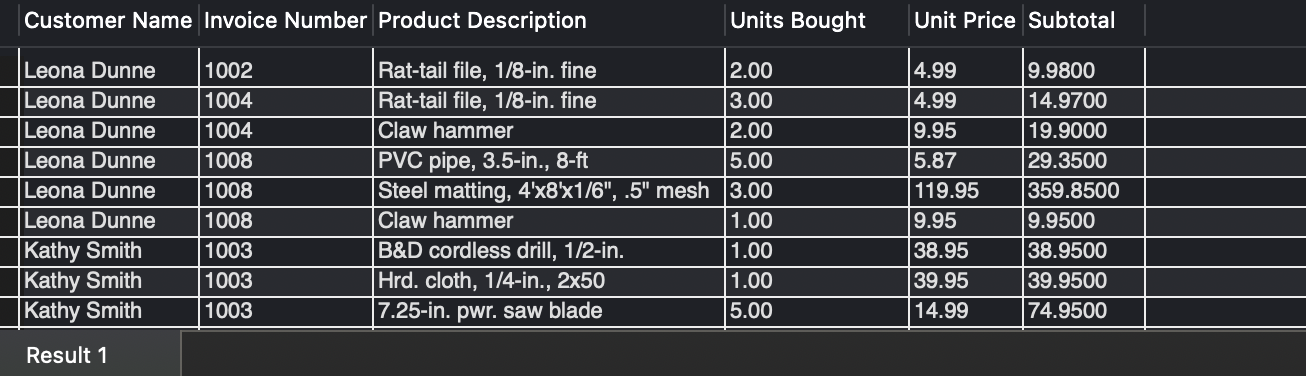
(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Subtotal`

FROM CUSTOMER

JOIN INVOICE ON CUSTOMER.CUS\_CODE = INVOICE.CUS\_CODE

JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER

JOIN PRODUCT ON LINE.P\_CODE = PRODUCT.P\_CODE;



Query 4: Write a query to display the Customer's Name (same format as above), the Cus\_Balance and Total purchases for

each customer. Total purchases is calculated by summing subtotals from problem 3.

SELECT

CONCAT(CUSTOMER.CUS\_FNAME, ' ', CUSTOMER.CUS\_LNAME) AS `Customer Name`,

CUSTOMER.CUS\_BALANCE AS `Customer Balance`,

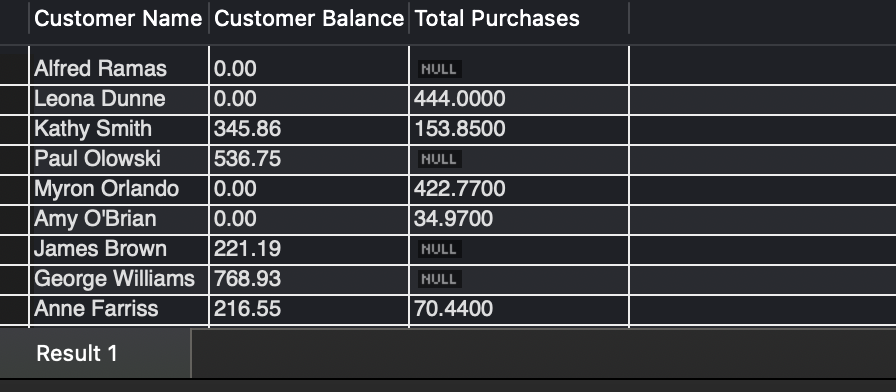
SUM(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Total Purchases`

FROM CUSTOMER

LEFT JOIN INVOICE ON CUSTOMER.CUS\_CODE = INVOICE.CUS\_CODE

LEFT JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER

GROUP BY `Customer Name`, `Customer Balance`;



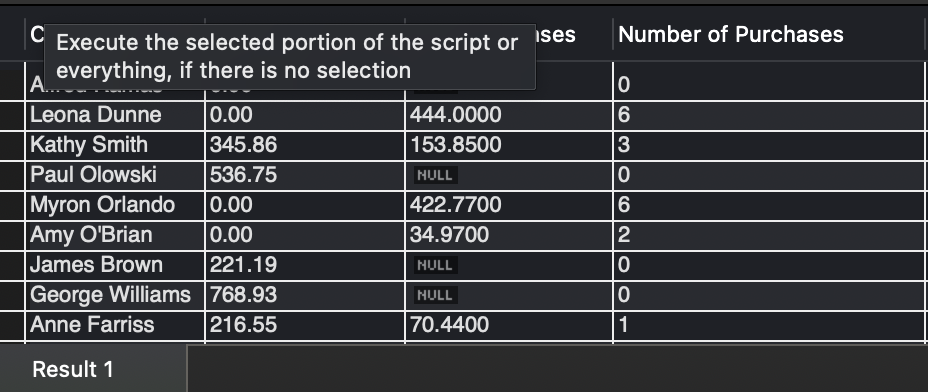
Query 5: Add the number of total purchases to query 4.

SELECT

CONCAT(CUSTOMER.CUS\_FNAME, ' ', CUSTOMER.CUS\_LNAME) AS `Customer Name`,

CUSTOMER.CUS\_BALANCE AS `Customer Balance`,

SUM(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Total Purchases`,

COUNT(INVOICE.INV\_NUMBER) AS `Number of Purchases`

FROM CUSTOMER

LEFT JOIN INVOICE ON CUSTOMER.CUS\_CODE = INVOICE.CUS\_CODE

LEFT JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER

GROUP BY `Customer Name`, `Customer Balance`;

Query 6: Calculate the average purchase amount for each customer to query 5. Your column headers should be Customer Last Name, Customer First Name, Total Purchases, Number of Purchases, and Average Purchase Amount. The average purchase amount is

calculated for each customer.

SELECT

CUSTOMER.CUS\_LNAME AS `Customer Last Name`,

CUSTOMER.CUS\_FNAME AS `Customer First Name`,

CUSTOMER.CUS\_BALANCE AS `Customer Balance`,

SUM(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Total Purchases`,

COUNT(INVOICE.INV\_NUMBER) AS `Number of Purchases`,

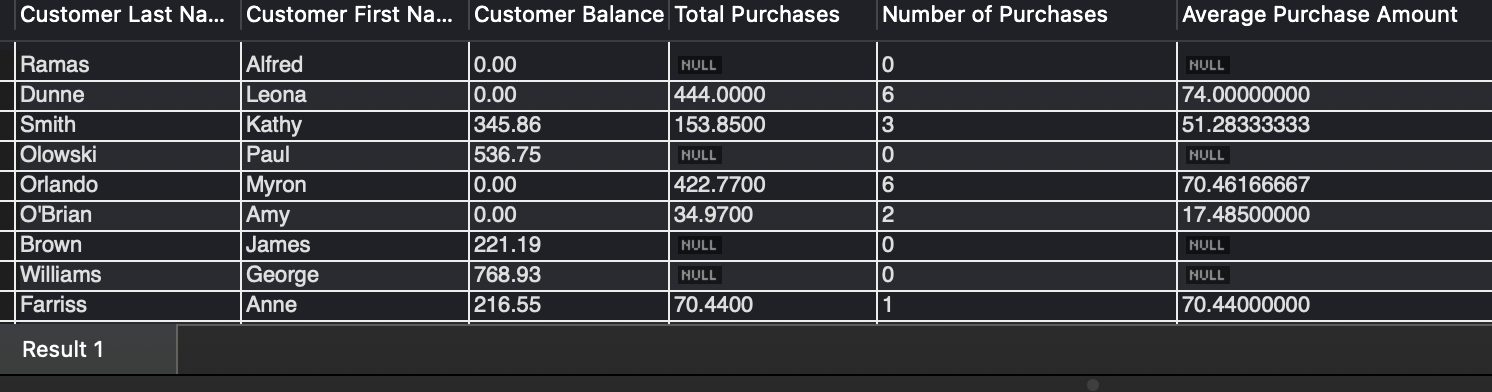
SUM(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) / COUNT(INVOICE.INV\_NUMBER) AS `Average Purchase Amount`

FROM CUSTOMER

LEFT JOIN INVOICE ON CUSTOMER.CUS\_CODE = INVOICE.CUS\_CODE

LEFT JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER

GROUP BY `Customer Last Name`, `Customer First Name`, `Customer Balance`;



Query 7: Write a query that provides the total number of invoices, the sales total for all invoices, the smallest purchase amount, largest purchase amount, and average purchase amount. You should use aliases to make your output look reasonable for a business user.

SELECT

COUNT(INVOICE.INV\_NUMBER) AS `Total Invoices`,

SUM(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Total Sales`,

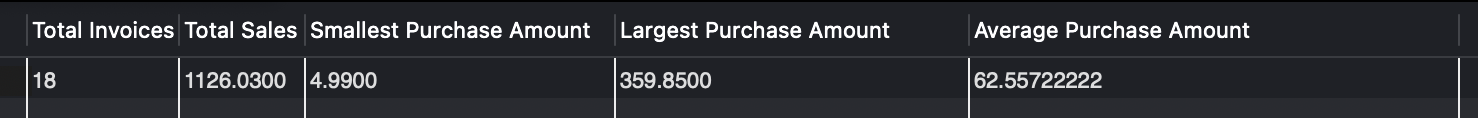
MIN(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Smallest Purchase Amount`,

MAX(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Largest Purchase Amount`,

AVG(LINE.LINE\_UNITS \* LINE.LINE\_PRICE) AS `Average Purchase Amount`

FROM INVOICE

LEFT JOIN LINE ON INVOICE.INV\_NUMBER = LINE.INV\_NUMBER;



Query 8: Advertising has requested a list of all customers who have never made a purchase, along with the customer's name and phone number. Provide them with a query with this information. Display the name in the style you believe would be most useful - briefly explain your choice in the comments.

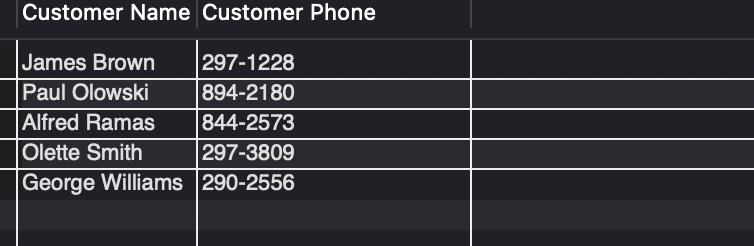
SELECT

CONCAT(CUSTOMER.CUS\_FNAME, ' ', CUSTOMER.CUS\_LNAME) AS `Customer Name`,

CUSTOMER.CUS\_PHONE AS `Customer Phone`

FROM CUSTOMER

WHERE CUSTOMER.CUS\_CODE NOT IN (SELECT DISTINCT CUS\_CODE FROM INVOICE);



Query 9: Calculate the value of products currently in inventory. The value of the products can be calculated using P\_QOH (quantity on hand) and P\_Price. Display results with a subtotal for each different product. You should display the P\_Descript, P\_QOH, P\_Price, and Subtotal for each product. Use aliases to improve the display.

SELECT

P\_DESCRIPT AS `Product Description`,

P\_QOH AS `Quantity On Hand`,

P\_PRICE AS `Unit Price`,

(P\_QOH \* P\_PRICE) AS `Subtotal`

FROM PRODUCT;



Query 10:Which Vendor does this company use the most frequently to supply products? Display the Vendor's name with the most products currently in the Product Table as well as the total number of products for that particular vendor. This query should display correctly if there is a tie between two vendors and if the data in the product table changes (e.g. we begin ordering more products from a particular vendor).

SELECT

V.V\_NAME AS `Vendor Name`,

COUNT(P.P\_CODE) AS `Total Products`

FROM VENDOR V

LEFT JOIN PRODUCT P ON V.V\_CODE = P.V\_CODE

GROUP BY `Vendor Name`

ORDER BY `Total Products` DESC

LIMIT 3;

