Diamond Beckom

10/27/24

Lab 4: DML

**7-3a (successful)**

SELECT P\_CODE, P\_DESCRIPT AS DESCRIPTION, P\_PRICE AS "Unit Price",

P\_QOH QTY

FROM PRODUCT

**7-3b (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, P\_QOH \* P\_PRICE

FROM PRODUCT;

**7-3b (written w/ alias) (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, P\_QOH \* P\_PRICE AS TOTVALUE

FROM PRODUCT;

**7-3e (successful)**

SELECT DISTINCT V\_CODE

FROM PRODUCT;

**7-4a (successful)**

SELECT P\_CODE, P\_DESCRIPT, P\_INDATE, P\_QOH, P\_MIN, P\_PRICE, P\_DISCOUNT, V\_CODE FROM PRODUCT;

**7-5b (successful)**

SELECT P\_CODE, P\_DESCRIPT, P\_QOH, P\_PRICE

FROM PRODUCT

ORDER BY P\_PRICE;

**7-5c (successful)**

SELECT P\_CODE, P\_DESCRIPT, P\_QOH, P\_PRICE

FROM PRODUCT

ORDER BY P\_PRICE DESC;

**7-5d (successful)**

SELECT EMP\_LNAME, EMP\_FNAME, EMP\_INITIAL, EMP\_AREACODE, EMP\_PHONE

FROM EMPLOYEE

ORDER BY EMP\_LNAME, EMP\_FNAME, EMP\_INITIAL

**7-5e (successful)**

SELECT P\_CODE, P\_DESCRIPT, V\_CODE, P\_PRICE \* P\_QOH AS TOTAL

FROM PRODUCT

ORDER BY V\_CODE, TOTAL DESC;

**7-6ab (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, V\_CODE

FROM PRODUCT

WHERE V\_CODE = 21344;

**7-6c (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, V\_CODE

FROM PRODUCT

WHERE V\_CODE <> 21344;

**7-6ad (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_MIN, P\_PRICE

FROM PRODUCT

WHERE P\_PRICE <= 10;

**7-6ba (successful)**

SELECT P\_CODE, P\_DESCRIPT, P\_QOH, P\_MIN, P\_PRICE

FROM PRODUCT

WHERE P\_CODE < '1558-QW1';

**7-6ca (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_MIN, P\_PRICE, P\_INDATE

FROM PRODUCT

WHERE P\_INDATE >= '2022-01-20';

**7-6da (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, V\_CODE

FROM PRODUCT

WHERE V\_CODE = 21344 OR V\_CODE = 24288;

**7-6db (successful)**

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, V\_CODE

FROM PRODUCT

WHERE P\_PRICE > 100

AND P\_QOH < 20;

**7-6dc (successful)**

SELECT P\_DESCRIPT, P\_PRICE, V\_CODE

FROM PRODUCT

WHERE V\_CODE = 25595 OR V\_CODE = 24288 AND P\_PRICE > 100;

**7-6dd (successful)**

SELECT P\_DESCRIPT, P\_PRICE, V\_CODE

FROM PRODUCT

WHERE (V\_CODE = 25595 OR V\_CODE = 24288) AND P\_PRICE > 100;

**7-6de (successful)**

SELECT \*

FROM PRODUCT

WHERE NOT (V\_CODE = 21344);

**7-6ea (successful)**

SELECT \*

FROM PRODUCT

WHERE P\_PRICE BETWEEN 50.00 AND 100.00;

**7-6eb (unsuccessful)**

SELECT \*

FROM PRODUCT

WHERE P\_PRICE => 50.00 AND P\_PRICE <= 100.00;

**\*COMMENTS\*: > Is not valid at this position; it’s expecting** **FOR, SELECT, WITH, LOCK, TABLE, VALUES…**

**7-6ec (successful)**

SELECT \*

FROM PRODUCT

WHERE V\_CODE = 21344 OR V\_CODE = 24288; / WHERE V\_CODE IN ('21344', '24288');

**7-6ed (successful)**

SELECT V\_NAME, V\_CONTACT, V\_AREACODE, V\_PHONE

FROM VENDOR

WHERE V\_CONTACT LIKE 'Smith%';

**7-6ee (successful)**

SELECT V\_NAME, V\_CONTACT, V\_AREACODE, V\_PHONE

FROM VENDOR

WHERE UPPER(V\_CONTACT) LIKE 'SMITH%';

**7-6ef (successful)**

SELECT V\_NAME, V\_CONTACT, V\_AREACODE, V\_PHONE

FROM VENDOR

WHERE V\_CONTACT NOT LIKE 'Smith%';

**7-6eg (unsuccessful)**

SELECT \*

FROM VENDOR

WHERE V\_CONTACT LIKE 'Johns\_n';

**\*COMMENTS\*: NULL was returned instead of matches when only approximate spellings are known.**

**7-6eh (successful)**

SELECT P\_CODE, P\_DESCRIPT, V\_CODE

FROM PRODUCT

WHERE V\_CODE IS NULL;

**7-7aa (successful)**

SELECT CUS\_CODE, CUS\_LNAME, INV\_NUMBER, INV\_DATE FROM CUSTOMER NATURAL JOIN INVOICE;

**7-6ab (successful)**

SELECT INV\_NUMBER, P\_CODE, P\_DESCRIPT, LINE\_UNITS, LINE\_PRICE FROM INVOICE NATURAL JOIN LINE NATURAL JOIN PRODUCT;

**7-7ba (successful)**

SELECT P\_CODE, P\_DESCRIPT, V\_CODE, V\_NAME, V\_AREACODE, V\_PHONE FROM PRODUCT JOIN VENDOR USING (V\_CODE);

**7-7ca (successful)**

SELECT INVOICE.INV\_NUMBER, PRODUCT.P\_CODE, P\_DESCRIPT,

LINE\_UNITS, LINE\_PRICE

FROM INVOICE JOIN LINE ON INVOICE.INV\_NUMBER =

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

**7-7da (successful)**

SELECT P\_CODE, VENDOR.V\_CODE, V\_NAME

FROM PRODUCT JOIN VENDOR ON PRODUCT.V\_CODE = VENDOR.V\_CODE;

**7-7ea (successful)**

SELECT P\_CODE, VENDOR.V\_CODE, V\_NAME

FROM PRODUCT, VENDOR

WHERE PRODUCT.V\_CODE = VENDOR.V\_CODE;

**7-7eb (unsuccessful)**

SELECT CUS\_FNAME, CUS\_LNAME, V\_NAME

FROM CUSTOMER JOIN INVOICE ON CUSTOMER.CUS\_CODE =

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

LINE.INV\_NUMBER JOIN PRODUCT JOIN VENDOR ON

PRODUCT.V\_CODE 5 VENDOR.V\_CODE

WHERE V\_STATE 5 'TN';

**\*COMMENTS\*: The DBMS detects a missing join condition.**

**7-7ec (successful)**

SELECT CUS\_FNAME, CUS\_LNAME, V\_NAME

FROM CUSTOMER, INVOICE, LINE, PRODUCT, VENDOR

WHERE V\_STATE = 'TN' AND CUSTOMER.CUS\_CODE =

INVOICE.CUS\_CODE AND INVOICE.INV\_NUMBER =

LINE.INV\_NUMBER AND PRODUCT.V\_CODE = VENDOR.V\_CODE;

**7-7fa (unsuccessful)**

SELECT column-list

FROM table1 LEFT [OUTER] JOIN table2 ON join-condition

**\*COMMENTS\*:** **Column is not valid at this position, expects a FOR, SELECT, WITH, LOCK, TABLE, VALUES…**

**7-7fb (successful)**

SELECT P\_CODE, VENDOR.V\_CODE, V\_NAME

FROM VENDOR LEFT JOIN PRODUCT ON VENDOR. V\_CODE =

PRODUCT.V\_CODE;

**7-7fc (unsuccessful)**

SELECT column-list

FROM table1 RIGHT [OUTER] JOIN table2 ON join-condition

**\*COMMENTS\*: Column is not valid at this position, expects a FOR, SELECT, WITH, LOCK, TABLE, VALUES…**

**7-7fd (successful)**

SELECT P\_CODE, VENDOR.V\_CODE, V\_NAME

FROM VENDOR RIGHT JOIN PRODUCT ON VENDOR. V\_CODE = PRODUCT.V\_CODE;

**7-7fe (unsuccessful)**

SELECT V\_CODE, V\_NAME, P\_CODE

FROM PRODUCT RIGHT JOIN VENDOR ON PRODUCT.V\_CODE =

VENDOR.V\_CODE

WHERE P\_CODE IS NULL;

**\*COMMENTS\*: V\_CODE in field list is ambiguous.**

**7-7ff (unsuccessful)**

SELECT P\_CODE, VENDOR.V\_CODE, V\_NAME

FROM VENDOR FULL JOIN PRODUCT ON VENDOR. V\_CODE =

PRODUCT.V\_CODE;

**\*COMMENTS\*: Unknown column is detected.**

**7-7ga (successful)**

SELECT INVOICE.INV\_NUMBER, CUS\_CODE, INV\_DATE, P\_CODE

FROM INVOICE CROSS JOIN LINE;

**7-7gb (successful)**

SELECT INVOICE.INV\_NUMBER, CUS\_CODE, INV\_DATE, P\_CODE

FROM INVOICE, LINE;

**7-7ha (successful)**

SELECT P\_DESCRIPT, P\_PRICE, V\_NAME, V\_CONTACT, V\_AREACODE,

V\_PHONE

FROM PRODUCT P JOIN VENDOR V ON P.V\_CODE = V.V\_CODE;

**7-7ia (successful)**

SELECT E.EMP\_NUM, E.EMP\_LNAME, E.EMP\_MGR, M.EMP\_LNAME

FROM EMP E JOIN EMP M ON E.EMP\_MGR = M.EMP\_NUM;

**7-8aa (successful)**

SELECT COUNT(P\_CODE)

FROM PRODUCT;

**7-8ab (successful)**

SELECT COUNT(P\_PRICE)

FROM PRODUCT

WHERE P\_PRICE < 10

**7-8ac (successful)**

SELECT COUNT(V\_CODE)

FROM PRODUCT;

**7-8ad (successful)**

SELECT COUNT(DISTINCT V\_CODE) AS "COUNT DISTINCT"

FROM PRODUCT;

**7-8ae (successful)**

SELECT MAX(P\_PRICE) AS MAXPRICE, MIN(P\_PRICE) AS MINPRICE

FROM PRODUCT;

**7-8af (successful)**

SELECT SUM(CUS\_BALANCE) AS TOTBALANCE

FROM CUSTOMER;

**7-8ag (successful)**

SELECT SUM(P\_QOH \* P\_PRICE) AS TOTVALUE

FROM PRODUCT;

**7-8ah (successful)**

SELECT AVG(P\_PRICE) AS AVGPRICE

FROM PRODUCT;

**7-8ba (successful)**

SELECT V\_CODE, AVG(P\_PRICE) AS AVGPRICE

FROM PRODUCT

GROUP BY V\_CODE;

**7-8bb (unsuccessful)**

SELECT V\_CODE, V\_NAME, COUNT(P\_CODE) AS NUMPRODS,

AVG(P\_PRICE) AS AVGPRICE

FROM PRODUCT JOIN VENDOR ON PRODUCT.V\_CODE = VENDOR.V\_CODE

GROUP BY V\_CODE, V\_NAME

ORDER BY V\_NAME;

**\*COMMENTS\*: Column V-CODE in field list is ambiguous.**

**7-8ca (successful)**

SELECT V\_CODE, COUNT(P\_CODE) AS NUMPRODS

FROM PRODUCT

GROUP BY V\_CODE

HAVING AVG(P\_PRICE) < 10

ORDER BY V\_CODE;

**7-11aa (unsuccessful)**

SELECT CUS\_LNAME, CUS\_FNAME, CUS\_INITIAL, CUS\_AREACODE,

CUS\_PHONE

FROM CUSTOMER

UNION

SELECT CUS\_LNAME, CUS\_FNAME, CUS\_INITIAL, CUS\_AREACODE,

CUS\_PHONE

FROM CUSTOMER\_2;

**\*COMMENTS\*: Error Code: 1146. Table 'saleco.customer\_2' doesn't exist**

**Problem Questions:**

9. Write a query to count the number of invoices.

SELECT COUNT(\*) FROM INVOICE;

10. Write a query to count the number of customers with a balance of more than $500.

SELECT COUNT(\*) FROM CUSTOMER WHERE Cus\_Balance > 500;

11. Generate a listing of all purchases made by the customers, using the output shown in Figure P7.11 as your guide. Sort the results by customer code, invoice number, and product description.

SELECT CUS\_CODE, INV\_NUMBER, INV\_DATE, P\_DESCRIPT, LINE\_UNITS, LINE\_PRICE

FROM Purchases

ORDER BY CUS\_CODE, INV\_NUMBER, P\_DESCRIPT;

12. Using the output shown in Figure P7.12 as your guide, generate a list of customer purchases, including the subtotals for each of the invoice line numbers. The subtotal is a derived attribute calculated by multiplying LINE\_UNITS by LINE\_PRICE. Sort the output by customer code, invoice number, and product description. Be certain to use the col umn aliases as shown in the figure.

SELECT I.CUS\_CODE, I.INV\_NUMBER, P.P\_DESCRIPT, L.LINE\_UNITS AS "Units Bought", L.LINE\_PRICE AS "Unit Price", ROUND(L.LINE\_UNITS \* L.LINE\_PRICE, 2) AS "Subtotal"

FROM CUSTOMER C, INVOICE I, LINE L, PRODUCT P WHERE C.CUS\_CODE = I.CUS\_CODE AND I.INV\_NUMBER = L.INV\_NUMBER AND P.P\_CODE = L.P\_CODE

ORDER BY I.CUS\_CODE, I.INV\_NUMBER, P.P\_DESCRIPT;

13. Write a query to display the customer code, balance, and total purchases for each customer. Total purchase is calculated by summing the line subtotals (as calculated in Problem 12) for each customer. Sort the results by customer code, and use aliases as shown in Figure P7.13.

SELECT c.customer\_code, c.balance, SUM(i.line\_subtotal) AS total\_purchases

FROM customers c

JOIN invoices i ON c.customer\_code = i.customer\_code

GROUP BY c.customer\_code, c.balance

ORDER BY c.customer\_code;

14. Modify the query in Problem 13 to include the number of individual product purchases made by each customer. (In other words, if the customer’s invoice is based on three products, one per LINE\_NUMBER, you count three product purchases. Note that in the original invoice data, customer 10011 generated three invoices, which contained a total of six lines, each representing a product purchase.) Your output values must match those shown in Figure P7.14, sorted by customer code.

SELECT c.customer\_code, c.balance, SUM(i.line\_subtotal) AS total\_purchases,

COUNT(i.line\_number) AS product\_purchases

FROM customers c

JOIN invoices i ON c.customer\_code = i.customer\_code

GROUP BY c.customer\_code, c.balance

ORDER BY c.customer\_code;

15. Use a query to compute the total of all purchases, the number of purchases, and the average purchase amount made by each customer. Your output values must match those shown in Figure P7.15. Sort the results by customer code.

SELECT c.customer\_code, SUM(i.line\_subtotal) AS total\_purchases, COUNT(i.line\_number) AS number\_of\_purchases,

AVG(i.line\_subtotal) AS average\_purchase\_amount

FROM customers c

JOIN invoices i ON c.customer\_code = i.customer\_code

GROUP BY c.customer\_code

ORDER BY c.customer\_code;

16. Create a query to produce the total purchase per invoice, generating the results shown in Figure P7.16, sorted by invoice number. The invoice total is the sum of the product purchases in the LINE that corresponds to the INVOICE.

SELECT INV\_NUMBER, SUM(LINE) AS "Invoice Total"

FROM Invoice

GROUP BY INV\_NUMBER

ORDER BY INV\_NUMBER;

17. Use a query to show the invoices and invoice totals in Figure P7.17. Sort the results by customer code and then by invoice number.

SELECT I.CUS\_CODE AS CUS\_CODE, I.INV\_NUMBER AS INV\_NUMBER,

SUM(L.LINE\_UNITS \* L.LINE\_PRICE) AS "INVOICE TOTAL"

FROM INVOICE I

INNER JOIN LINE L ON I.INVOICE\_NUMBER = L.INVOICE\_NUMBER

GROUP BY I.CUS\_CODE, I.INV\_NUMBER

ORDER BY I.CUS\_CODE, I.INV\_NUMBER;

18. Write a query to produce the number of invoices and the total purchase amounts by customer, using the output shown in Figure P7.18 as your guide. Note the results are sorted by customer code. (Compare this summary to the results shown in Problem 17.)

SELECT Customer\_Code, COUNT(Invoice\_Number) AS Number\_of\_Invoices,

SUM(Invoice\_Amount) AS Total\_Purchase\_Amount

FROM Invoice

GROUP BY Customer\_Code

ORDER BY Customer\_Code;

20. List the balances of customers who have made purchases during the current invoice cycle—that is, for the customers who appear in the INVOICE table. The results of this query are shown in Figure P7.20, sorted by customer code.

SELECT CUSTOMER.CUS\_CODE, ROUND(CUS\_BALANCE,2) "CUS\_BALANCE"

FROM CUSTOMER INNER JOIN INVOICE

WHERE CUSTOMER.CUS\_CODE=INVOICE.CUS\_CODE

GROUP BY CUSTOMER.CUS\_CODE

ORDER BY CUSTOMER.CUS\_CODE;

21. Provide a summary of customer balance characteristics for customers who made purchases. Include the minimum balance, maximum balance, and average balance, as shown in Figure P7.21.

SELECT MIN(C.CUS\_BALANCE) AS "Minimum Balance", MAX(C.CUS\_BALANCE) AS "Maximum Balance", ROUND(AVG(C.CUS\_BALANCE),2) AS "Average Balance"

FROM INVOICE I, CUSTOMER C

WHERE C.CUS\_CODE=I.CUS\_CODE;

22. Create a query to find the balance characteristics for all customers, including the total of the outstanding balances. The results of this query are shown in Figure P7.22.

SELECT SUM(Balance) AS "Total Balances", MIN(Balance) AS "Minimum Balance", MAX(Balance) AS "Maximum Balance", AVG(Balance) AS "Average Balance"

FROM Customers;

25. Create a query that summarizes the value of products currently in inventory. Note that the value of each product is a result of multiplying the units currently in inventory by the unit price. Sort the results in descending order by subtotal, as shown in Figure P7.25.

SELECT P\_DESCRIPT, P\_QOH, P\_PRICE, P\_QOH\*P\_PRICE AS Subtotal

FROM PRODUCT

ORDER BY Subtotal DESC;

26. Find the total value of the product inventory. The results are shown in Figure P7.26.

SELECT SUM(price \* quantity) AS "Total Value of Inventory"

FROM productinventory;