**CS411, Winter 2023, Homework 3– ch4, 5, 6, and 7.**

1. **(5) Write a SELECT statement that returns these four columns where the balance due is less than 200 and greater than 0.**

**vendor\_name The vendor\_name column from the Vendors table**

**invoice\_number The invoice\_number column from the Invoices table**

**invoice\_date The invoice\_date column from the Invoices table**

**balance\_due The invoice\_total column minus the payment\_total and credit\_total columns from the Invoices table**

**Sort the result by balance\_due.**

USE ap;

SELECT vendor\_name, invoice\_number, invoice\_date, invoice\_total-payment\_total - credit\_total

AS balance\_due

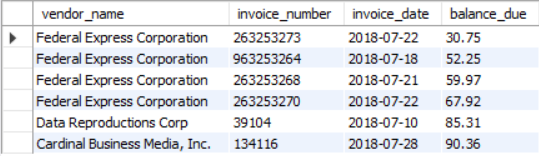
FROM vendors INNER JOIN invoices

ON vendors.vendor\_id = invoices.vendor\_id

WHERE invoice\_total - payment\_total - credit\_total < 200

AND invoice\_total - payment\_total - credit\_total > 0

ORDER BY invoice\_total - payment\_total - credit\_total

****

****

1. **(5) Write a SELECT statement that returns three columns:**

**vendor\_id The vendor\_id column from the Vendors table**

**vendor\_name The vendor\_name column from the Vendors table**

**contact\_name A concatenation of the vendor\_contact\_first\_name and vendor\_contact\_last\_name columns with a space between**

**Return one row for each vendor whose contact has the same last name as another vendor’s contact. Sort the result set by vendor\_contact\_last\_name.**

USE ap;

SELECT v1.vendor\_id, v2.vendor\_name,

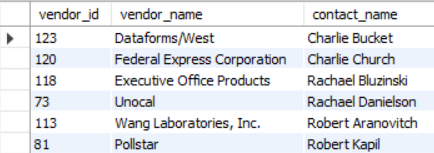
CONCAT(v1.vendor\_contact\_first\_name, "",v1.vendor\_contact\_last\_name) AS contact\_name

FROM vendors AS v1 JOIN vendors AS v2

ON v1.vendor\_id != v2.vendor\_id

AND v1.vendor\_contact\_first\_name = v2.vendor\_contact\_first\_name

ORDER BY v1.vendor\_contact\_first\_name

****

****

1. **(5) Show the vendor name, line\_item\_description, and account number for each line item. Show vendors which use just one account number.**

USE ap;

SELECT distinct v.vendor\_name, k.line\_item\_description, k.account\_number

FROM vendors v

JOIN invoices i ON v.vendor\_id = i.vendor\_id

JOIN invoice\_line\_items k ON k.invoice\_id = i.invoice\_id

WHERE vendor\_name IN (

SELECT v.vendor\_name

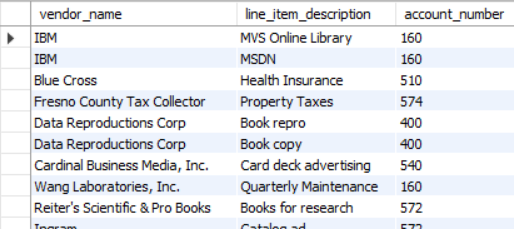
FROM vendors v

JOIN invoices i ON v.vendor\_id = i.vendor\_id

JOIN invoice\_line\_items k ON k.invoice\_id = i.invoice\_id

GROUP BY vendor\_name

HAVING COUNT(DISTINCT k.account\_number) = 1)





1. **(5) Show two columns from the Vendors table: vendor\_name and vendor\_phone. If the vendor has a phone number, the vendor\_phone value should be its phone number. Otherwise, the vendor\_phone value should be “No Phone.” Just show the vendors whose initial starts with ‘A’ through ‘K’ and sort the result set by vendor\_name.**

****

USE ap;

SELECT vendor\_name, vendor\_phone

FROM vendors

WHERE vendor\_phone is not null

AND vendor\_name >= 'A'

AND vendor\_name < 'L'

UNION

SELECT vendor\_name, vendor\_phone

FROM vendors

WHERE vendor\_phone is null

AND vendor\_name >= 'A'

AND vendor\_name < 'L';

****UPDATE vendors

SET vendor\_phone = 'No Phone'

WHERE vendor\_phone is null

ORDER BY vendor\_name

1. **(40) Consider the following ER diagram.**
2. **(10) Show the account\_description and the number of invoices for each account\_description. Show the top five results in terms of the number of invoices.**

USE ap;

SELECT account\_description, COUNT(invoice\_id) AS number\_of\_invoices

FROM general\_ledger\_accounts g

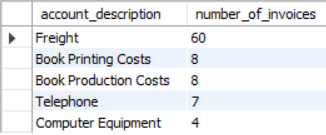
JOIN vendors v ON g.account\_number = v.default\_account\_number

JOIN invoices i ON v.vendor\_id = i.vendor\_id

GROUP BY account\_description

ORDER BY COUNT(invoice\_id) DESC

LIMIT 5



1. **(10) What is the total invoice for the account from the above question? Show the query that finds the answer. The result should show the account\_number, account\_description, the number of invoices, and the total invoice for each account\_number.**

USE ap;

SELECT account\_number, account\_description, COUNT(invoice\_id) AS number\_of\_invoices, SUM(invoice\_id) AS total\_invoice

FROM general\_ledger\_accounts g

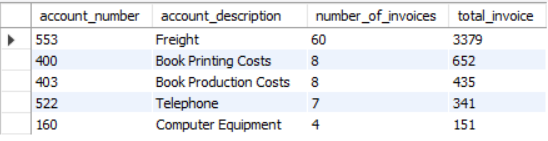
JOIN vendors v ON g.account\_number = v.default\_account\_number

JOIN invoices i ON v.vendor\_id = i.vendor\_id

GROUP BY account\_description, account\_number

ORDER BY COUNT(invoice\_id) DESC

LIMIT 5

****

1. **(10) Find the vendor which has the biggest amount of total invoices. Show the query that shows the vendor name and the total invoice amount.**

USE ap;

SELECT v.vendor\_name, SUM(invoice\_total) AS total\_invoices

FROM vendors v

JOIN invoices i ON v.vendor\_id = i.vendor\_id

GROUP BY v.vendor\_name

ORDER BY total\_invoices DESC

LIMIT 1

****

1. **(10) Find the vendor which has the single most expensive order for an item. Show the query that shows the vendor name, the item description, and the amount.**

USE ap;

SELECT vendor\_name, line\_item\_description, line\_item\_amount

FROM vendors v

JOIN invoices i ON v.vendor\_id = i.vendor\_id

JOIN invoice\_line\_items k ON k.invoice\_id = i.invoice\_id

ORDER BY line\_item\_amount DESC

LIMIT 1

****

1. **5) Create a table called “new\_terms” from the terms table. Then, write an INSERT statement that adds this row to the new\_terms table:**

**terms\_id: 6terms\_description: Net due 120 days**

**terms\_due\_days: 120**

**Then, show the result of the following query: SELECT \* FROM new\_terms;**

USE ap;

CREATE TABLE new\_terms AS

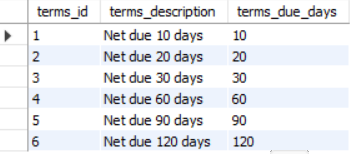
SELECT \*

FROM terms;

INSERT INTO new\_terms

(terms\_id, terms\_description, terms\_due\_days)

VALUES (6, 'Net due 120 days', 120)



1. **(5) Then, write an UPDATE statement that modifies the row you just added to the Terms table. This statement should change the terms\_description column to “Net due 100 days”, and it should change the terms\_due\_days column to 100.**

**Then, show the result of the following query: SELECT \* FROM new\_terms;**

USE ap;

SELECT terms\_description, terms\_due\_days

FROM new\_terms

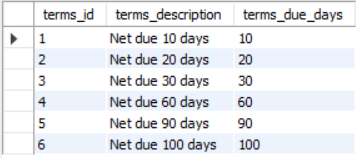
WHERE terms\_id = 6;

UPDATE new\_terms

SET terms\_description = 'Net due 100 days',

terms\_due\_days = 100

WHERE terms\_id = 6

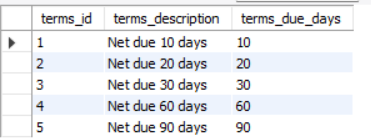
****

1. **(5) Then, write a DELETE statement that deletes the row you added to the new Terms table (that is with the terms\_id 6).Then, show the result of the following query: SELECT \* FROM new\_terms;**

USE ap;

DELETE FROM new\_terms

WHERE terms\_id = 6;

****

1. **(5) Write a query that displays the vendor name, the number of invoices, and the sum of invoice totals for each vendor, sorted by the sum of invoice totals. Show only those vendors whose invoice totals are less than the average invoice totals.**

USE ap;

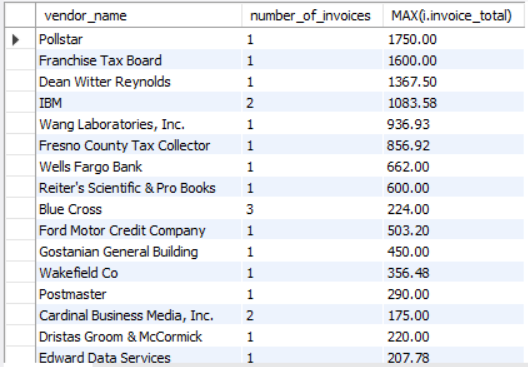
SELECT v.vendor\_name, COUNT(invoice\_id) AS number\_of\_invoices, MAX(i.invoice\_total)

FROM invoices i JOIN vendors v

ON i.vendor\_id = v.vendor\_id

GROUP BY v.vendor\_id

HAVING SUM(invoice\_total) < (SELECT AVG(invoice\_total)

 FROM invoices i

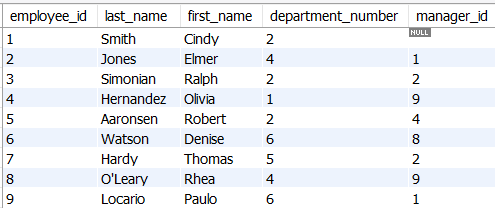
WHERE vendor\_id = i.vendor\_id

)

ORDER BY SUM(invoice\_total) DESC

****

1. **(5) Consider the following database. Show a query that displays the employee ids, employee names and their manager names, ordered by the employee\_id. Also, show the query result.  
   Note: You should make this table first.**

****

CREATE TABLE employees(

employee\_id INT PRIMARY KEY AUTO\_INCREMENT,

last\_name VARCHAR(20) NOT NULL,

first\_name VARCHAR(20) NOT NULL,

department\_number INT NOT NULL,

manager\_id INT

);

USE ap;

INSERT INTO employees

VALUES

(1, 'Smith', 'Cindy', 2, null),

(2, 'Jones', 'Elmer', 4, 1),

(3, 'Simonian', 'Ralph', 2, 2),

(4, 'Herandez', 'Olivia', 1, 9),

(5, 'Aaronsen', 'Robert', 2, 4),

(6, 'Watson', 'Denise', 6, 8),

(7, 'Hardy','Thomas',5,2),

(8, "O'Leary",'Rhea',4,9),

(9, 'Locario', 'Paulo',6,1);

USE ap;

SELECT e.employee\_id, CONCAT(e.first\_name, " ", e.last\_name) AS 'Employee Name', CONCAT(f.first\_name, " ", f.last\_name) AS 'Manager Name'

FROM employees e LEFT JOIN employees f

ON e.manager\_id = f.employee\_id

ORDER BY e.employee\_id

1. **(5) From the above question, show a query that displays employees who manage more than one employee.**

SELECT f.employee\_id, CONCAT(f.first\_name, " ", f.last\_name) AS 'Manager Name'

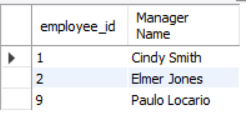
FROM employees e LEFT JOIN employees f

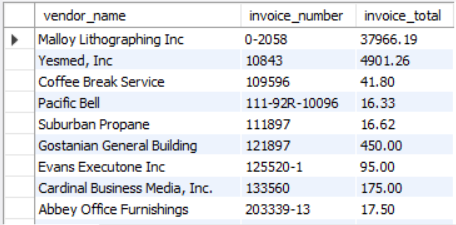
ON e.manager\_id = f.employee\_id

GROUP BY e.manager\_id

HAVING count(\*) > 1

ORDER BY employee\_id

****

1. **(5) Use a correlated subquery to return one row per vendor, representing the vendor’s oldest invoice (the one with the earliest date). Each row should include these three column: vendor\_name, invoice\_number, and invoice\_total. Sort the result by the invoice\_number column.**

USE ap;

SELECT vendor\_name, invoice\_number, invoice\_total

FROM vendors v JOIN invoices i

ON v.vendor\_id = i.vendor\_id

WHERE invoice\_date =

(SELECT MIN(invoice\_date)

FROM invoices sub

WHERE v.vendor\_id = sub.vendor\_id)

****ORDER BY invoice\_number

1. **(5) Rewrite the query for the above problem that does not use a correlated query.**

SELECT v.vendor\_name, i.invoice\_number, i.invoice\_total

FROM vendors v JOIN invoices i

ON v.vendor\_id = i.vendor\_id

WHERE (v.vendor\_name, i.invoice\_date) IN

(SELECT v.vendor\_name, MIN(i.invoice\_date)

FROM vendors v JOIN invoices i

WHERE v.vendor\_id = i.vendor\_id

GROUP BY v.vendor\_name

ORDER BY invoice\_date)

ORDER BY invoice\_number;

