# Exercise M2.1

## Chapter 3

8. Write a SELECT statement that returns three columns from the Vendors table: vendor_name, vendor_contact_last_name, and vendor_contact_first_name. Then, run this statement to make sure it works correctly.

   Add an ORDER BY clause to this statement that sorts the result set by last name and then first name, both in ascending sequence. Then, run this statement again to make sure it works correctly. This is a good way to build and test a statement, one clause at a time.

-8
```
SELECT vendor_name,vendor_contact_last_name,vendor_contact_first_name
FROM Vendors
ORDER BY vendor_contact_last_name,vendor_contact_first_name;
```

9. Write a SELECT statement that returns one column from the Vendors table named full_name that joins the vendor_contact_last_name and vendor_contact_first_name columns.

   Format this column with the last name, a comma, a space, and the first name like this:

   ```
   Doe, John
   ```

   Sort the result set by last name and then first name in ascending sequence.

   Return only the contacts whose last name begins with the letter A, B, C, or E. This should retrieve 41 rows.

-9
```
SELECT CONCAT(vendor_contact_last_name,', ',vendor_contact_first_name)
FROM Vendors
WHERE (vendor_contact_last_name < 'D') or (vendor_contact_last_name >= 'E' and
vendor_contact_last_name <‘F’)
ORDER BY vendor_contact_last_name,vendor_contact_first_name;
```

10. Write a SELECT statement that returns these column names and data from the Invoices table:

| | |
|---|---|
| Due Date | The invoice_due_date column |
| Invoice Total | The invoice_total column |
| 10% | 10% of the value of invoice_total |
| Plus 10% | The value of invoice_total plus 10% |

Return only the rows with an invoice total that's greater than or equal to 500 and less than or equal to 1000. This should retrieve 12 rows.

Sort the result set in descending sequence by invoice_due_date.

```
-10
SELECT invoice_due_date AS "Due Date",
       invoice_total AS "Invoice Total",
       0.1*invoice_total AS '10%',
       1.1*invoice_total AS 'Plus 10%'
FROM invoices
WHERE invoice_total>=500 and invoice_total<=1000
ORDER BY invoice_due_date DESC;
```

11. Write a SELECT statement that returns these columns from the Invoices table:

| | |
|---|---|
| invoice_number | The invoice_number column |
| invoice_total | The invoice_total column |
| payment_credit_total | Sum of the payment_total and credit_total columns |
| balance_due | The invoice_total column minus the payment_total and credit_total columns |

Return only invoices that have a balance due that's greater than $50.

Sort the result set by balance due in descending sequence.

Use the LIMIT clause so the result set contains only the rows with the 5 largest balances.

```
-11
SELECT invoice_number,
       invoice_total,
       payment_total+credit_total AS payment_credit_total,
       invoice_total-payment_total-credit_total AS balance_due
FROM Invoices
WHERE invoice_total-payment_total-credit_total > 50
ORDER BY invoice_total-payment_total-credit_total DESC
LIMIT 5
```

- 12

```
SELECT invoice_number,invoice_date,invoice_total-payment_total-credit_total AS
balance_due,payment_date
FROM Invoices
WHERE payment_date IS NULL;
```

13. Write a SELECT statement without a FROM clause that uses the CURRENT_DATE function to return the current date in its default format.

    Use the DATE_FORMAT function to format the current date in this format:

    ```
    mm-dd-yyyy
    ```

    This displays the month, day, and four-digit year of the current date.

    Give this column an alias of current_date. To do that, you must enclose the alias in quotes since that name is already used by the CURRENT_DATE function.

-- 13

```
SELECT DATE_FORMAT(CURRENT_DATE, '%m-%d-%Y') AS “current_date”;
```

14. Write a SELECT statement without a FROM clause that creates a row with these columns:

| | |
|---|---|
| starting_principal | Starting principal of $50,000 |
| interest | 6.5% of the principal |
| principal_plus_interest | The principal plus the interest |

    To calculate the third column, add the expressions you used for the first two columns.

- 14

```
SELECT 50000 AS starting_principal, 0.065* 50000 AS interest, 1.065*50000 AS
principal_plus_interest
```