Exercises

1. Write a SELECT statement that returns three columns from the Vendors table: VendorContactFName, VendorContactLName, and VendorName. Sort the result set by last name, then by first name.
2. Write a SELECT statement that returns four columns from the Invoices table, named Number, Total, Credits, and Balance:

Number	Column alias for the InvoiceNumber column
Total	Column alias for the InvoiceTotal column
Credits	Sum of the PaymentTotal and CreditTotal columns
Balance	InvoiceTotal minus the sum of PaymentTotal and CreditTotal

   Use the AS keyword to assign column aliases.
3. Write a SELECT statement that returns one column from the Vendors table named Full Name. Create this column from the VendorContactFName and VendorContactLName columns. Format it as follows: last name, comma, first name (for example, "Doe, John"). Sort the result set by last name, then by first name.
4. Write a SELECT statement that returns three columns:

InvoiceTotal	From the Invoices table
10%	10% of the value of InvoiceTotal
Plus 10%	The value of InvoiceTotal plus 10%

   (For example, if InvoiceTotal is 100.0000, 10% is 10.0000, and Plus 10% is 110.0000.) Only return those rows with a balance due greater than 1000. Sort the result set by InvoiceTotal, with the largest invoice first.
5. Modify the solution to exercise 2 to filter for invoices with an InvoiceTotal that's greater than or equal to $500 but less than or equal to $10,000.
6. Modify the solution to exercise 3 to filter for contacts whose last name begins with the letter A, B, C, or E.
7. Write a SELECT statement that determines whether the PaymentDate column of the Invoices table has any invalid values. To be valid, PaymentDate must be a null value if there's a balance due and a non-null value if there's no balance due. Code a compound condition in the WHERE clause that tests for these conditions.