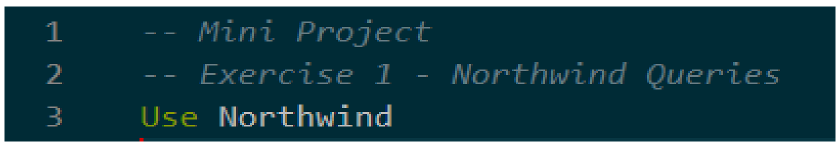
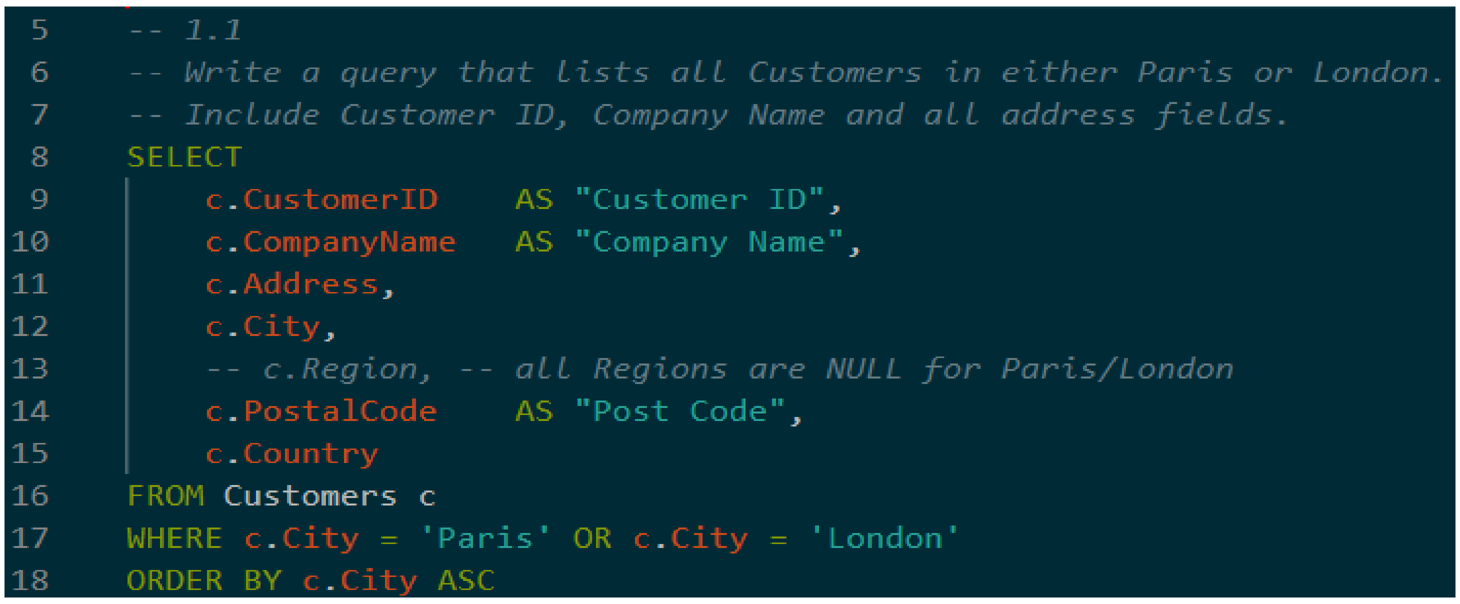
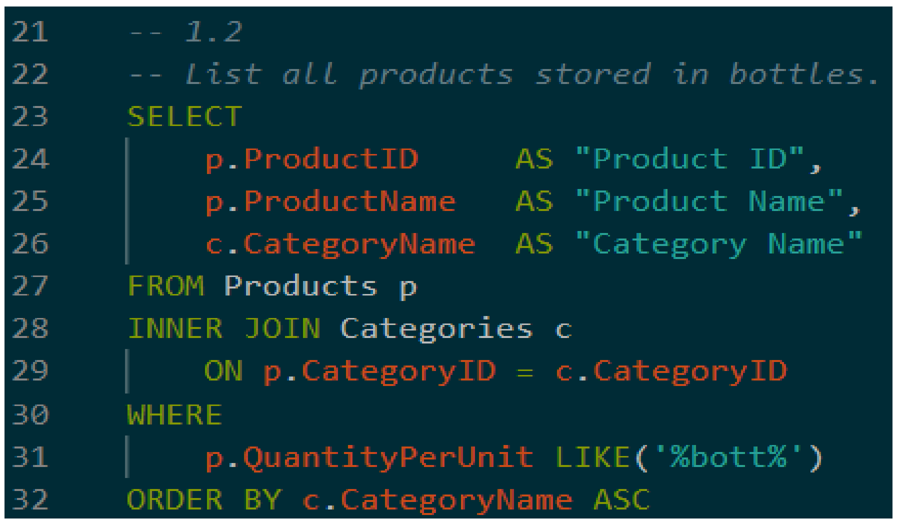


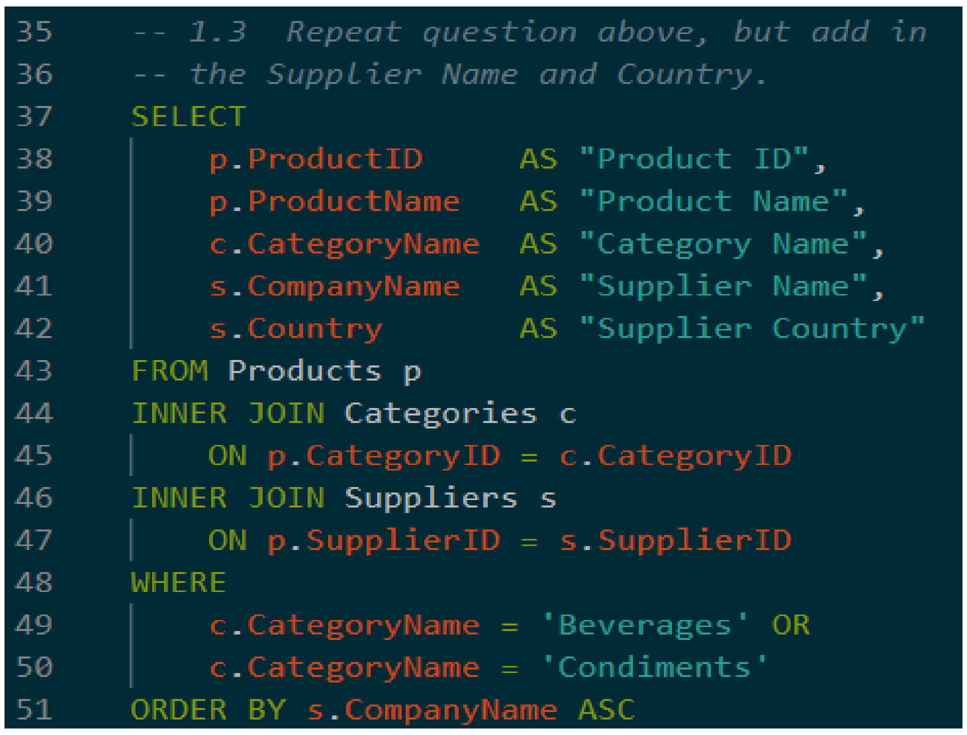
```sql
-- Mini Project
-- Exercise 1 - Northwind Queries
Use Northwind
```



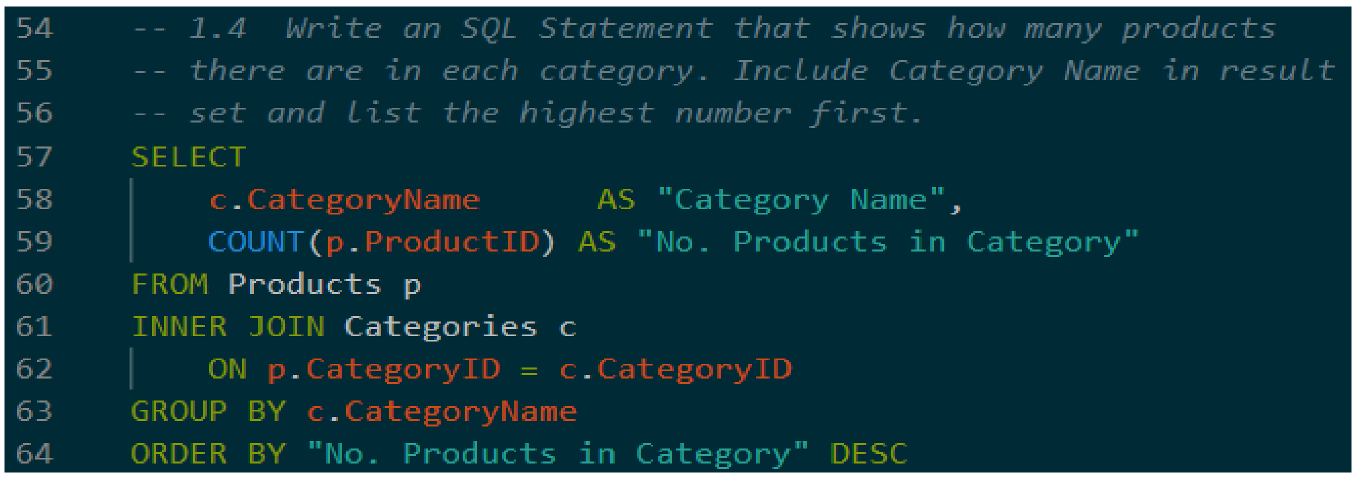
```sql
-- 1.1
-- Write a query that lists all Customers in either Paris or London.
-- Include Customer ID, Company Name and all address fields.
SELECT
    c.CustomerID    AS "Customer ID",
    c.CompanyName   AS "Company Name",
    c.Address,
    c.City,
    -- c.Region, -- all Regions are NULL for Paris/London
    c.PostalCode    AS "Post Code",
    c.Country
FROM Customers c
WHERE c.City = 'Paris' OR c.City = 'London'
ORDER BY c.City ASC
```



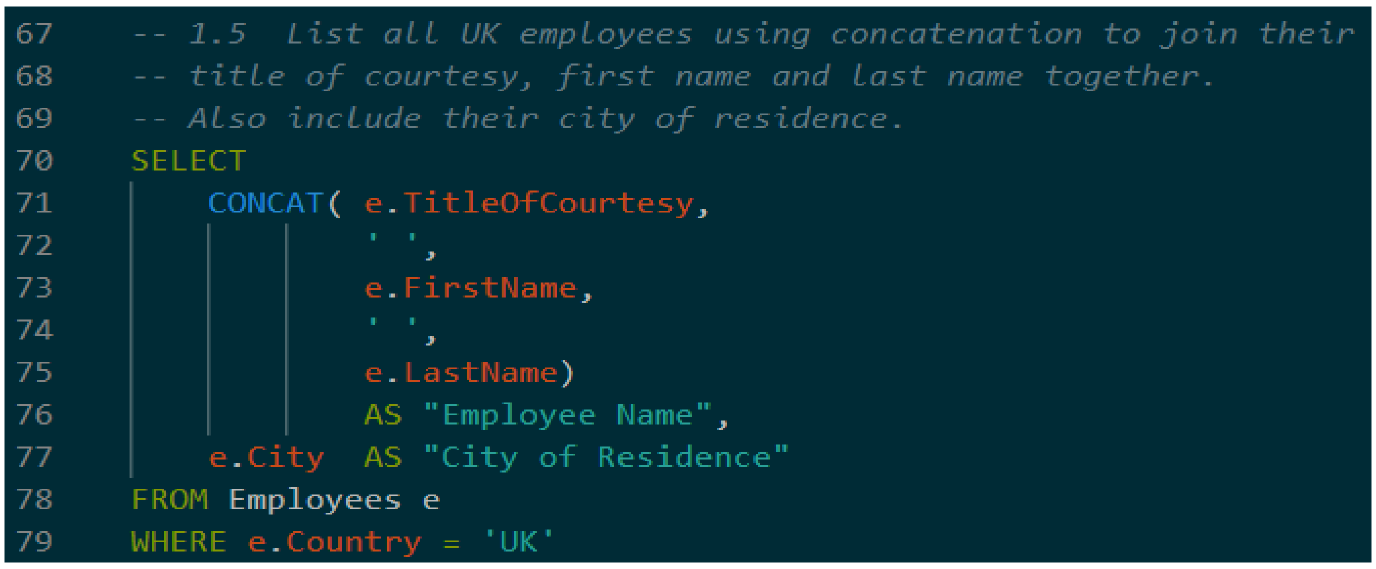
```sql
-- 1.2
-- List all products stored in bottles.
SELECT
    p.ProductID     AS "Product ID",
    p.ProductName   AS "Product Name",
    c.CategoryName  AS "Category Name"
FROM Products p
INNER JOIN Categories c
    ON p.CategoryID = c.CategoryID
WHERE
    p.QuantityPerUnit LIKE('%bott%')
ORDER BY c.CategoryName ASC
```



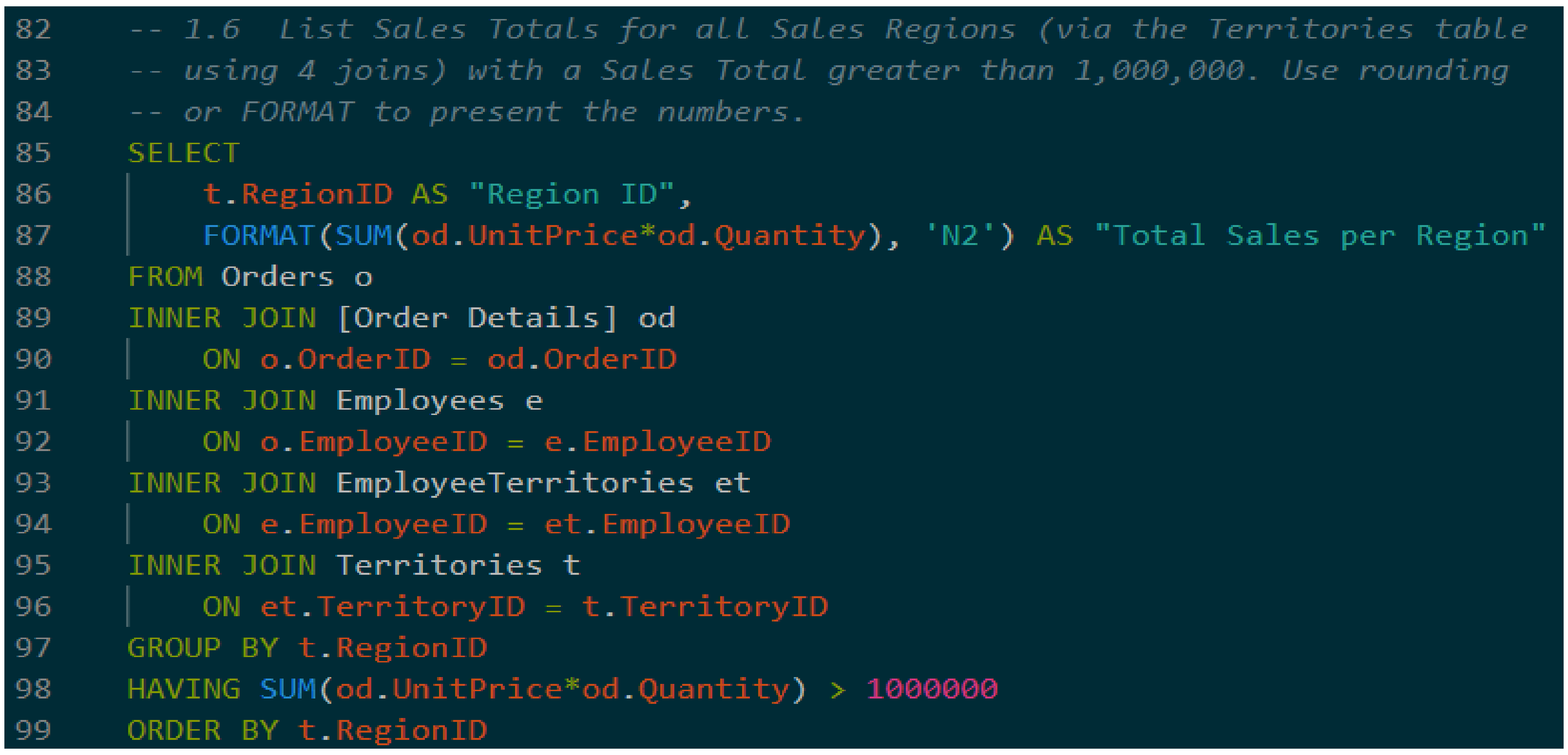
```sql
-- 1.3  Repeat question above, but add in
-- the Supplier Name and Country.
SELECT
    p.ProductID     AS "Product ID",
    p.ProductName   AS "Product Name",
    c.CategoryName  AS "Category Name",
    s.CompanyName   AS "Supplier Name",
    s.Country       AS "Supplier Country"
FROM Products p
INNER JOIN Categories c
    ON p.CategoryID = c.CategoryID
INNER JOIN Suppliers s
    ON p.SupplierID = s.SupplierID
WHERE
    c.CategoryName = 'Beverages' OR
    c.CategoryName = 'Condiments'
ORDER BY s.CompanyName ASC
```



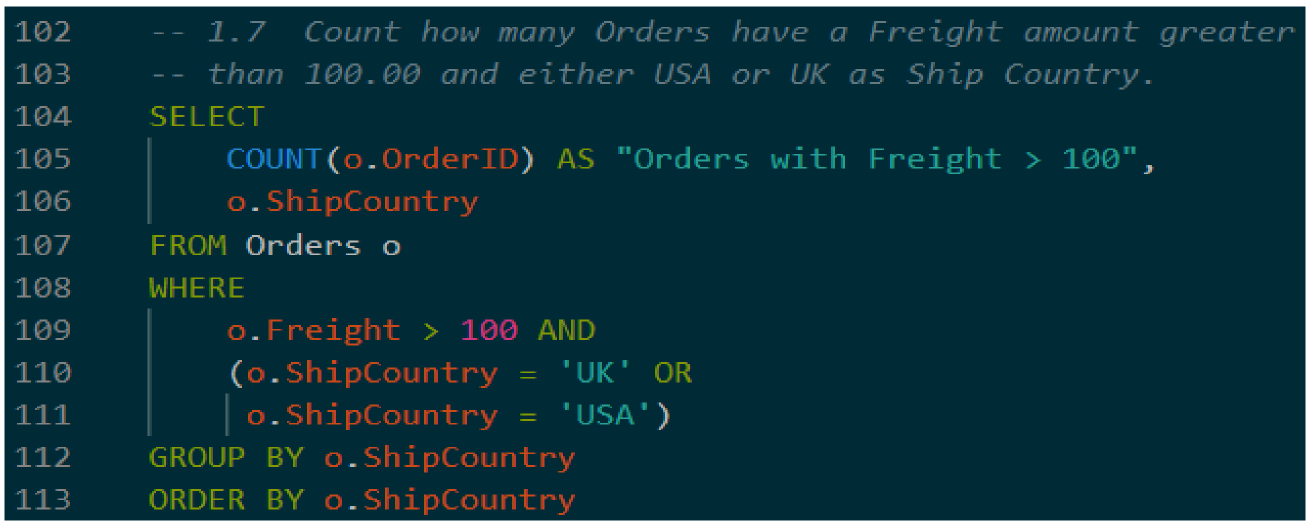
```sql
-- 1.4  Write an SQL Statement that shows how many products
-- there are in each category. Include Category Name in result
-- set and list the highest number first.
SELECT
    c.CategoryName      AS "Category Name",
    COUNT(p.ProductID) AS "No. Products in Category"
FROM Products p
INNER JOIN Categories c
    ON p.CategoryID = c.CategoryID
GROUP BY c.CategoryName
ORDER BY "No. Products in Category" DESC
```



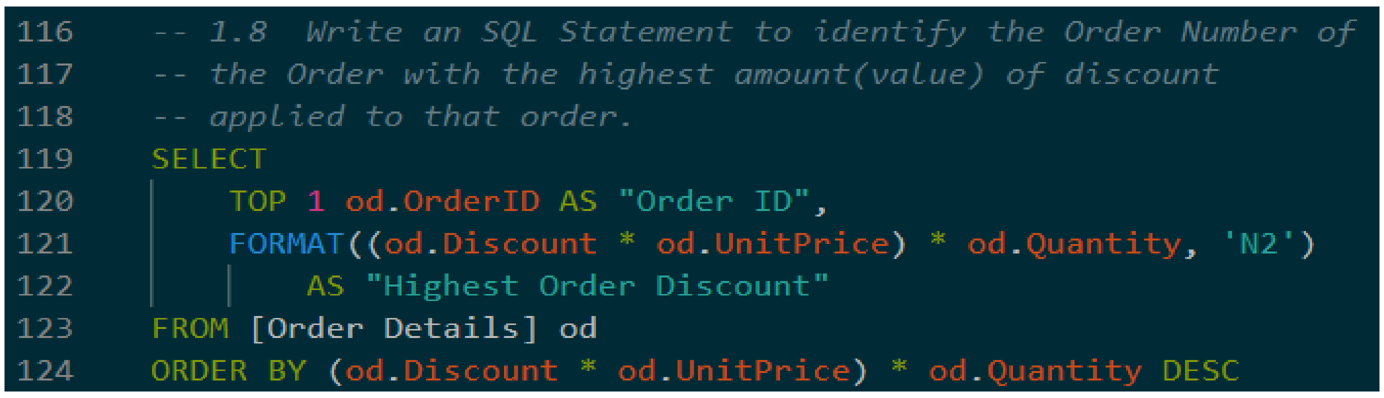
```sql
-- 1.5  List all UK employees using concatenation to join their
-- title of courtesy, first name and last name together.
-- Also include their city of residence.
SELECT
    CONCAT( e.TitleOfCourtesy,
            ' ',
            e.FirstName,
            ' ',
            e.LastName)
            AS "Employee Name",
    e.City  AS "City of Residence"
FROM Employees e
WHERE e.Country = 'UK'
```



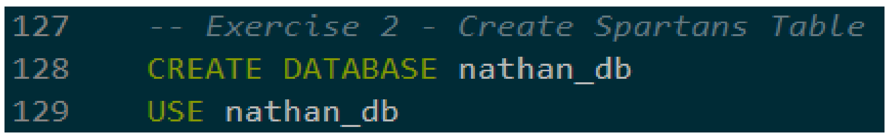
```sql
-- 1.6  List Sales Totals for all Sales Regions (via the Territories table
-- using 4 joins) with a Sales Total greater than 1,000,000. Use rounding
-- or FORMAT to present the numbers.
SELECT
    t.RegionID AS "Region ID",
    FORMAT(SUM(od.UnitPrice*od.Quantity), 'N2') AS "Total Sales per Region"
FROM Orders o
INNER JOIN [Order Details] od
    ON o.OrderID = od.OrderID
INNER JOIN Employees e
    ON o.EmployeeID = e.EmployeeID
INNER JOIN EmployeeTerritories et
    ON e.EmployeeID = et.EmployeeID
INNER JOIN Territories t
    ON et.TerritoryID = t.TerritoryID
GROUP BY t.RegionID
HAVING SUM(od.UnitPrice*od.Quantity) > 1000000
ORDER BY t.RegionID
```



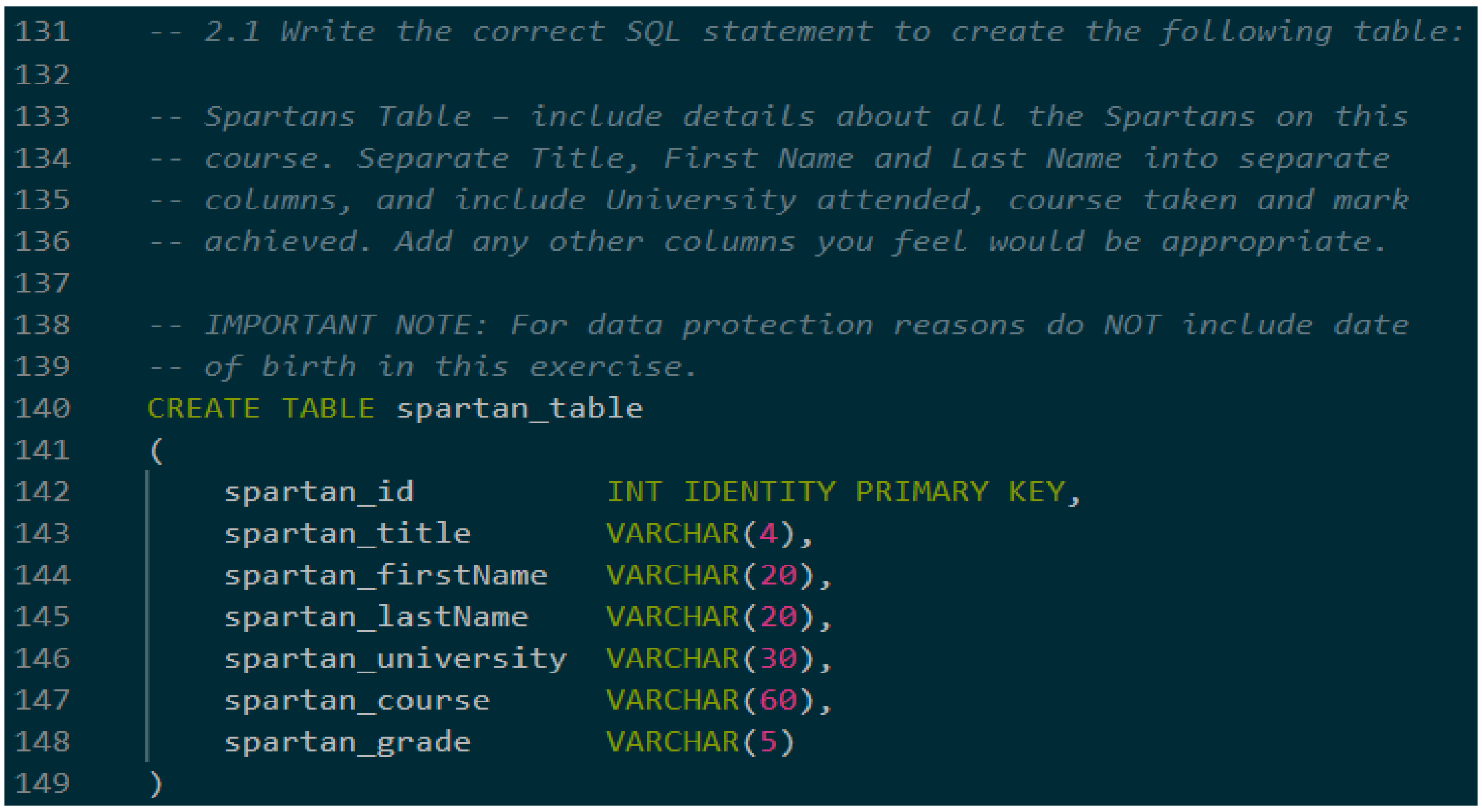
```sql
-- 1.7  Count how many Orders have a Freight amount greater
-- than 100.00 and either USA or UK as Ship Country.
SELECT
    COUNT(o.OrderID) AS "Orders with Freight > 100",
    o.ShipCountry
FROM Orders o
WHERE
    o.Freight > 100 AND
    (o.ShipCountry = 'UK' OR
     o.ShipCountry = 'USA')
GROUP BY o.ShipCountry
ORDER BY o.ShipCountry
```



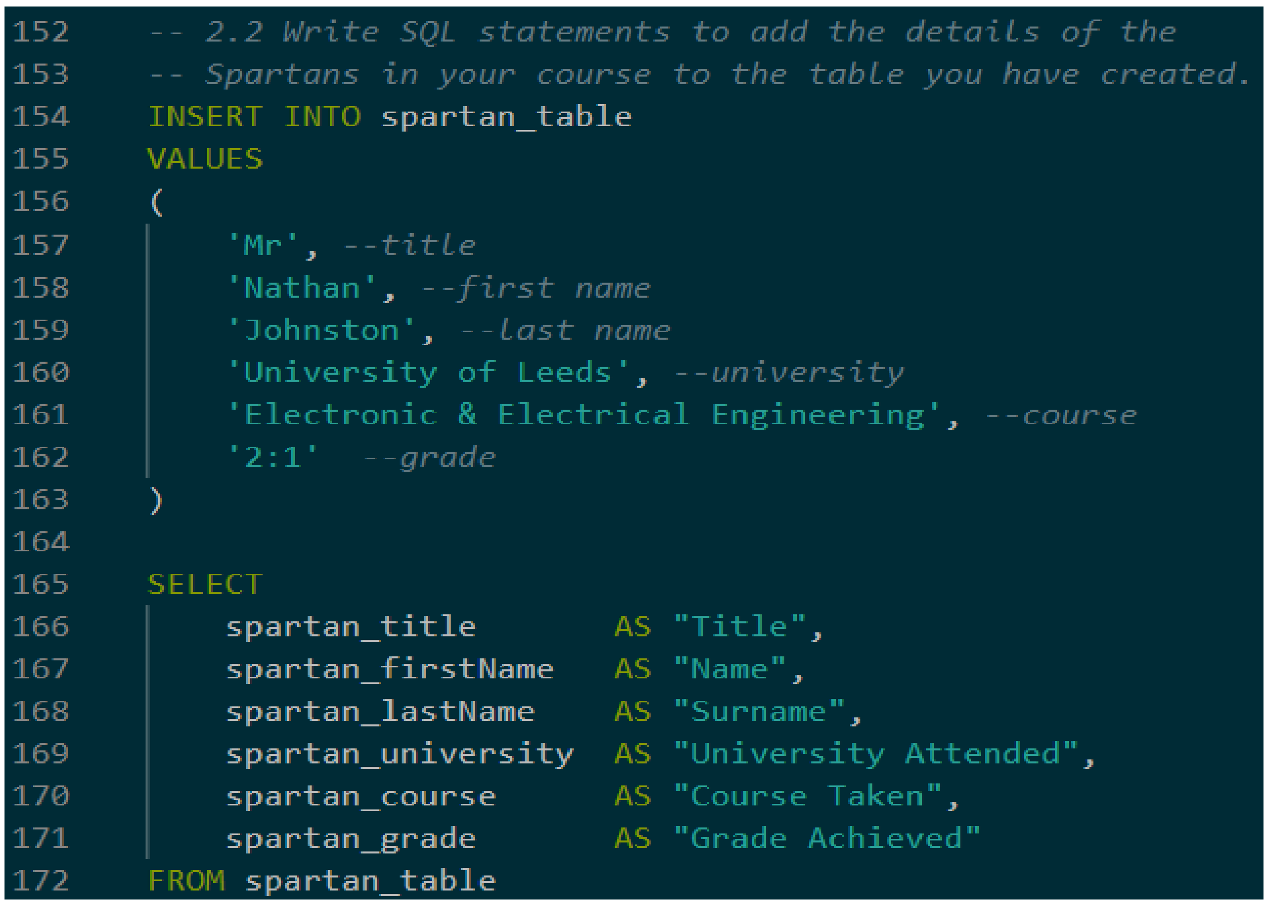
```sql
-- 1.8  Write an SQL Statement to identify the Order Number of
-- the Order with the highest amount(value) of discount
-- applied to that order.
SELECT
    TOP 1 od.OrderID AS "Order ID",
    FORMAT((od.Discount * od.UnitPrice) * od.Quantity, 'N2')
        AS "Highest Order Discount"
FROM [Order Details] od
ORDER BY (od.Discount * od.UnitPrice) * od.Quantity DESC
```



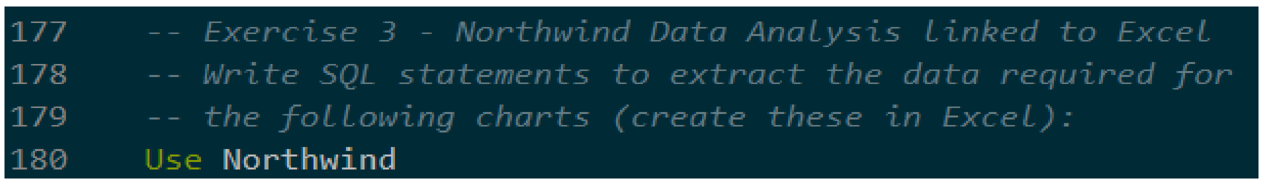
```sql
-- Exercise 2 - Create Spartans Table
CREATE DATABASE nathan_db
USE nathan_db
```



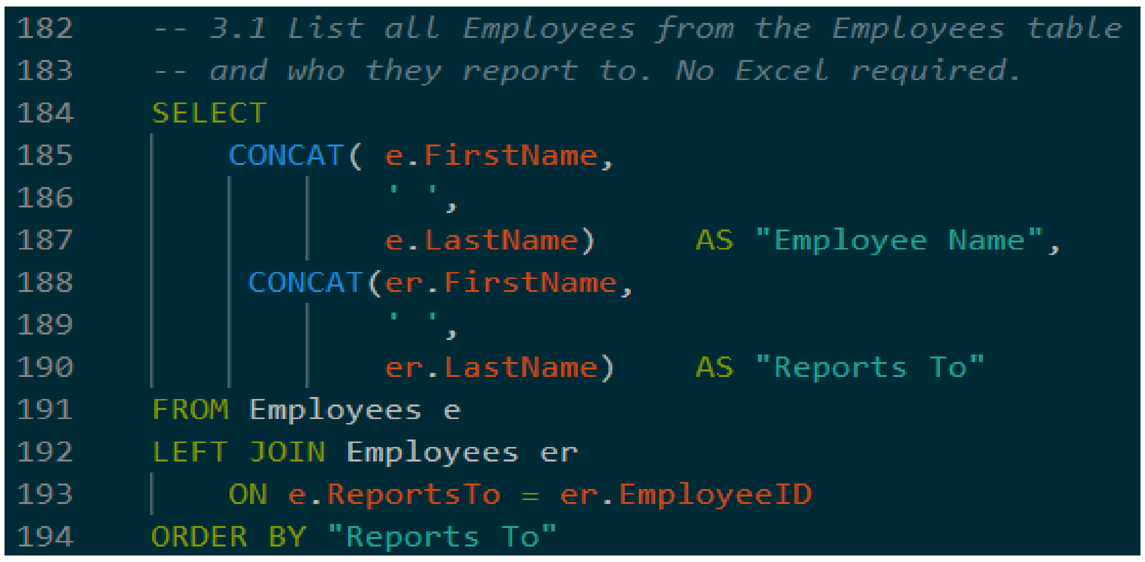
```sql
-- 2.1 Write the correct SQL statement to create the following table:

-- Spartans Table – include details about all the Spartans on this
-- course. Separate Title, First Name and Last Name into separate
-- columns, and include University attended, course taken and mark
-- achieved. Add any other columns you feel would be appropriate.

-- IMPORTANT NOTE: For data protection reasons do NOT include date
-- of birth in this exercise.
CREATE TABLE spartan_table
(
    spartan_id          INT IDENTITY PRIMARY KEY,
    spartan_title       VARCHAR(4),
    spartan_firstName   VARCHAR(20),
    spartan_lastName    VARCHAR(20),
    spartan_university  VARCHAR(30),
    spartan_course      VARCHAR(60),
    spartan_grade       VARCHAR(5)
)
```



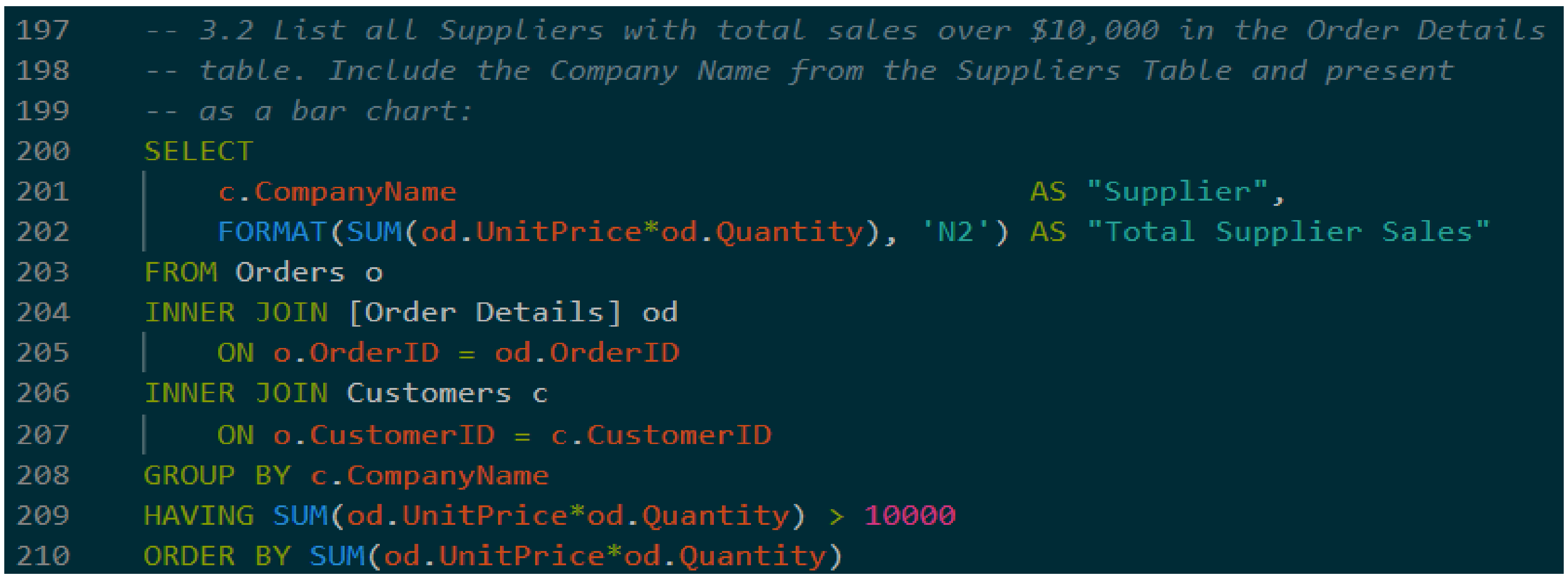
```sql
-- 2.2 Write SQL statements to add the details of the
-- Spartans in your course to the table you have created.
INSERT INTO spartan_table
VALUES
(
    'Mr', --title
    'Nathan', --first name
    'Johnston', --last name
    'University of Leeds', --university
    'Electronic & Electrical Engineering', --course
    '2:1'   --grade
)

SELECT
    spartan_title       AS "Title",
    spartan_firstName   AS "Name",
    spartan_lastName    AS "Surname",
    spartan_university  AS "University Attended",
    spartan_course      AS "Course Taken",
    spartan_grade       AS "Grade Achieved"
FROM spartan_table
```



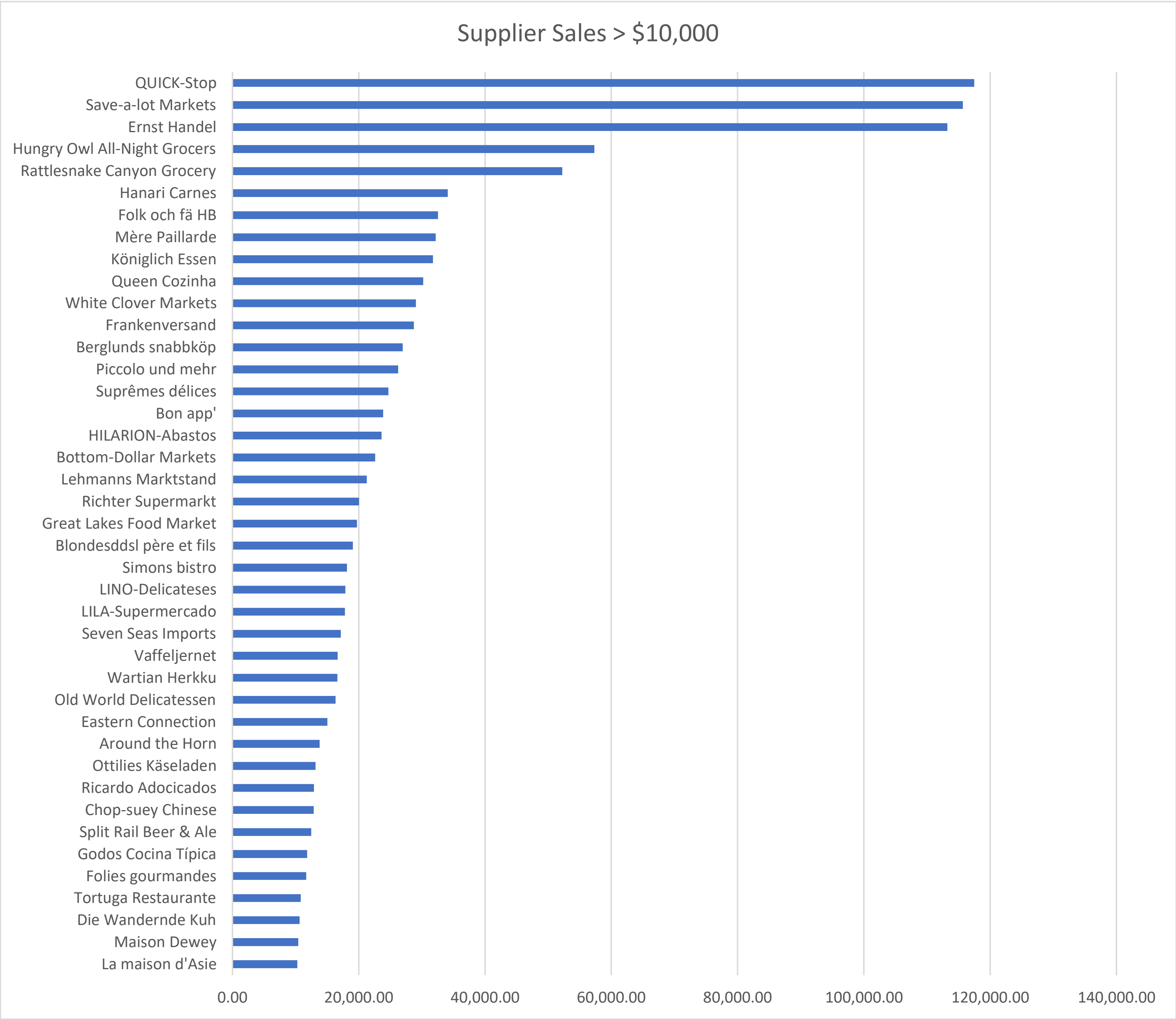
```sql
-- Exercise 3 - Northwind Data Analysis linked to Excel
-- Write SQL statements to extract the data required for
-- the following charts (create these in Excel):
Use Northwind
```

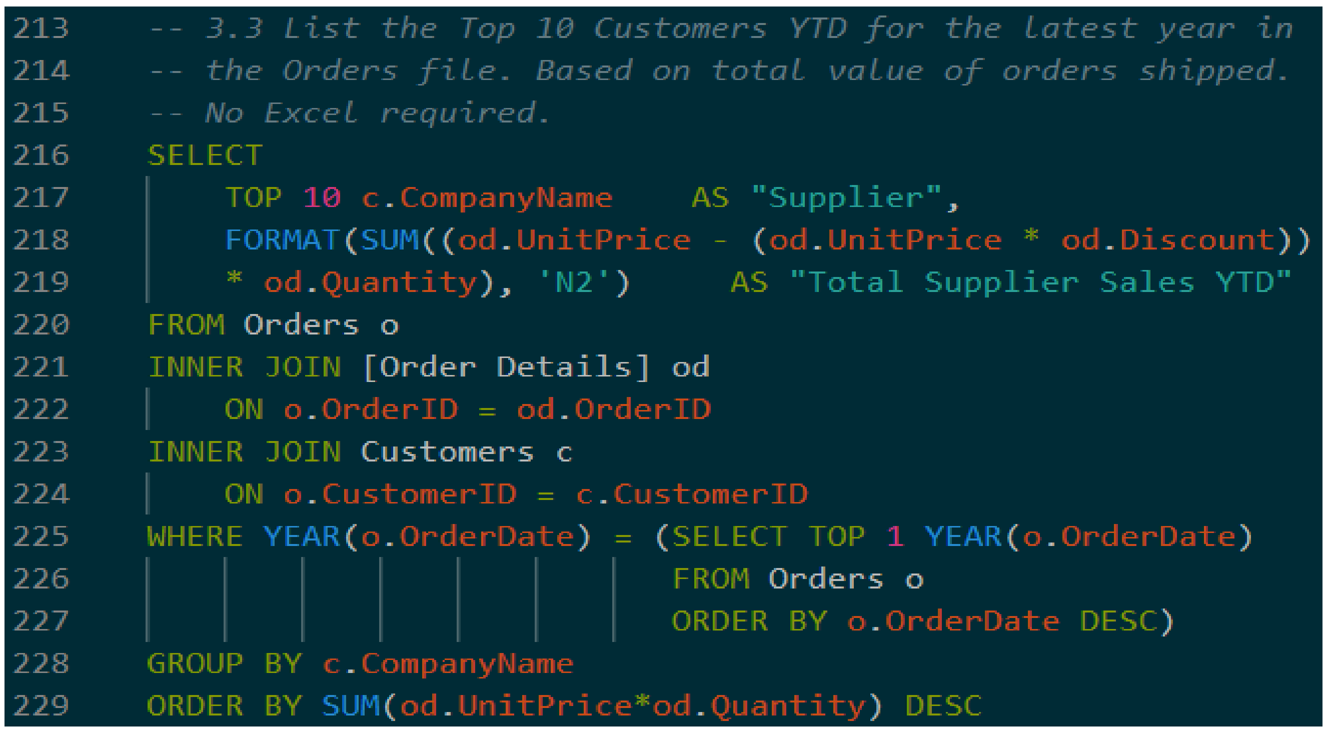


```sql
-- 3.1 List all Employees from the Employees table
-- and who they report to. No Excel required.
SELECT
    CONCAT( e.FirstName,
            ' ',
            e.LastName)     AS "Employee Name",
     CONCAT(er.FirstName,
            ' ',
            er.LastName)    AS "Reports To"
FROM Employees e
LEFT JOIN Employees er
    ON e.ReportsTo = er.EmployeeID
ORDER BY "Reports To"
```

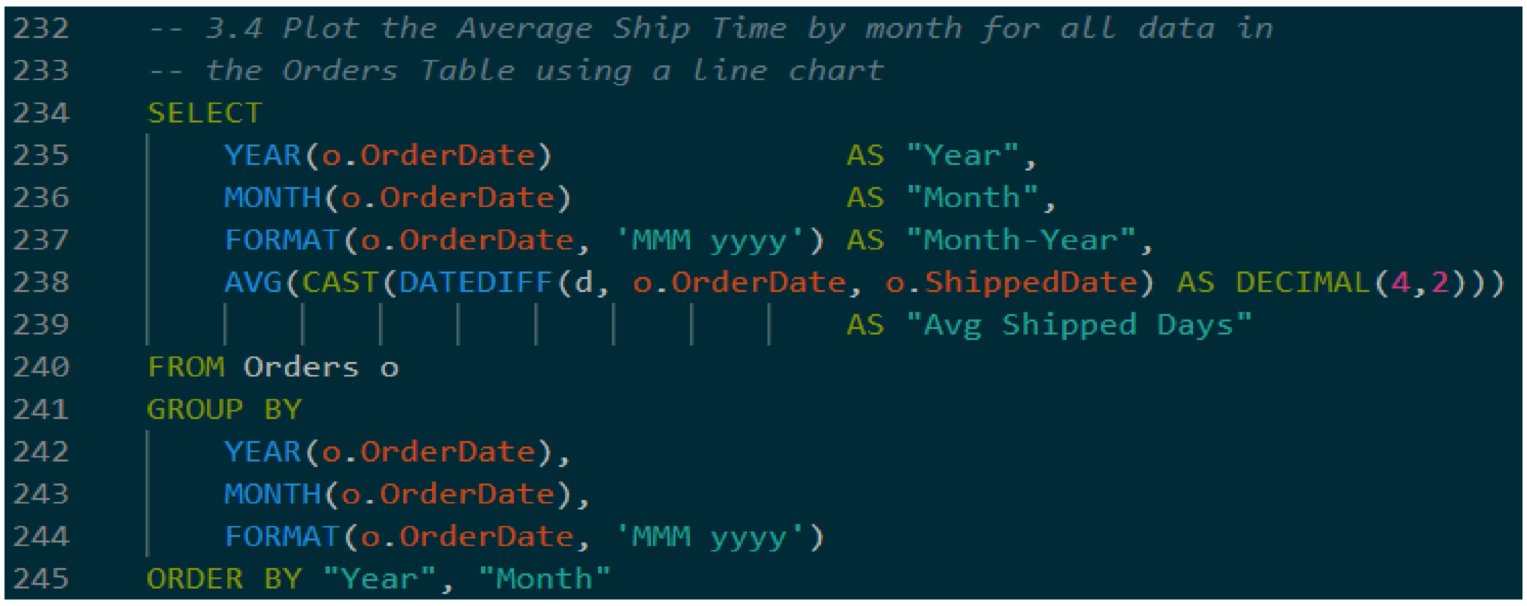


```sql
-- 3.2 List all Suppliers with total sales over $10,000 in the Order Details
-- table. Include the Company Name from the Suppliers Table and present
-- as a bar chart:
SELECT
    c.CompanyName                               AS "Supplier",
    FORMAT(SUM(od.UnitPrice*od.Quantity), 'N2') AS "Total Supplier Sales"
FROM Orders o
INNER JOIN [Order Details] od
    ON o.OrderID = od.OrderID
INNER JOIN Customers c
    ON o.CustomerID = c.CustomerID
GROUP BY c.CompanyName
HAVING SUM(od.UnitPrice*od.Quantity) > 10000
ORDER BY SUM(od.UnitPrice*od.Quantity)
```

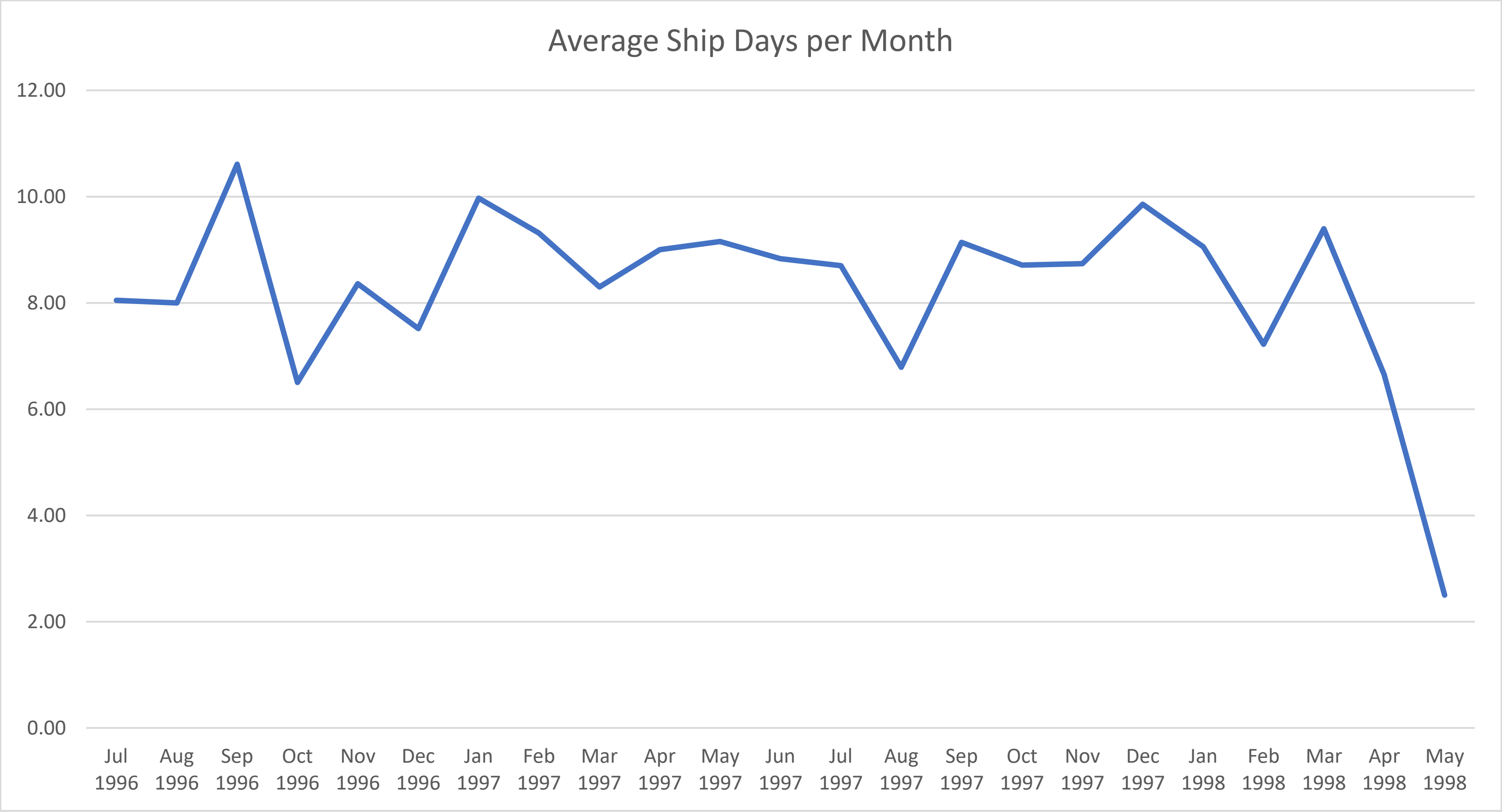




```sql
-- 3.3 List the Top 10 Customers YTD for the latest year in
-- the Orders file. Based on total value of orders shipped.
-- No Excel required.
SELECT
    TOP 10 c.CompanyName     AS "Supplier",
    FORMAT(SUM((od.UnitPrice - (od.UnitPrice * od.Discount))
    * od.Quantity), 'N2')      AS "Total Supplier Sales YTD"
FROM Orders o
INNER JOIN [Order Details] od
    ON o.OrderID = od.OrderID
INNER JOIN Customers c
    ON o.CustomerID = c.CustomerID
WHERE YEAR(o.OrderDate) = (SELECT TOP 1 YEAR(o.OrderDate)
                           FROM Orders o
                           ORDER BY o.OrderDate DESC)
GROUP BY c.CompanyName
ORDER BY SUM(od.UnitPrice*od.Quantity) DESC
```



```sql
-- 3.4 Plot the Average Ship Time by month for all data in
-- the Orders Table using a line chart
SELECT
    YEAR(o.OrderDate)               AS "Year",
    MONTH(o.OrderDate)              AS "Month",
    FORMAT(o.OrderDate, 'MMM yyyy') AS "Month-Year",
    AVG(CAST(DATEDIFF(d, o.OrderDate, o.ShippedDate) AS DECIMAL(4,2)))
                                    AS "Avg Shipped Days"
FROM Orders o
GROUP BY
    YEAR(o.OrderDate),
    MONTH(o.OrderDate),
    FORMAT(o.OrderDate, 'MMM yyyy')
ORDER BY "Year", "Month"
```


Average Ship Days per Month
Jul 1996
Aug 1996
Sep 1996
Oct 1996
Nov 1996
Dec 1996
Jan 1997
Feb 1997
Mar 1997
Apr 1997
May 1997
Jun 1997
Jul 1997
Aug 1997
Sep 1997
Oct 1997
Nov 1997
Dec 1997
Jan 1998
Feb 1998
Mar 1998
Apr 1998
May 1998