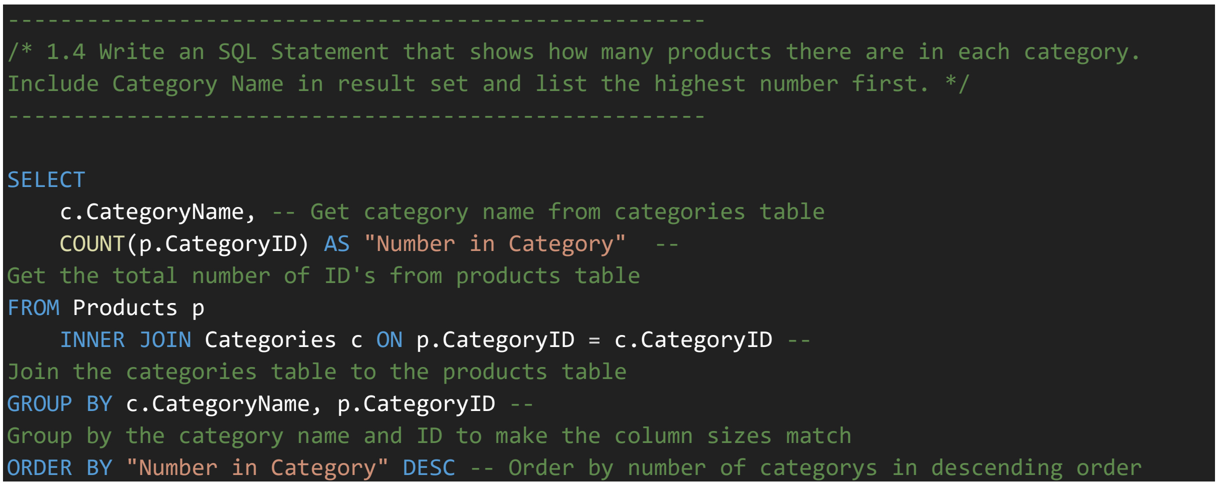
```
USE Northwind
```

```
/* Exercise 1 – Northwind Queries (40 marks: 5 for each question) */
```

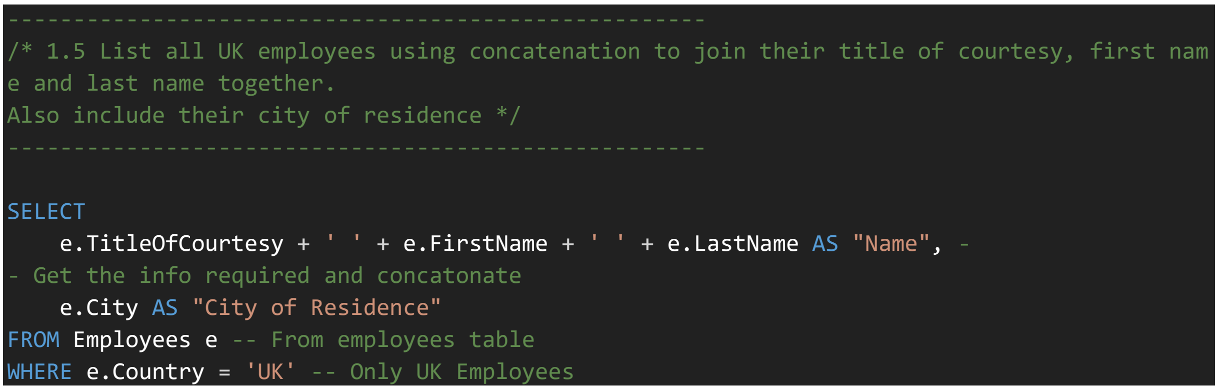
```
/* 1.1 Write a query that lists all Customers in either Paris or London. Include Customer I
D, Company Name and all address fields. */

SELECT
    c.CustomerID, --Select Customer ID
    c.CompanyName, -- Select Company Name
    c.Address, -- Select Address
    c.City, -- SELECT City
    c.Country -- SELECT Country
FROM Customers c -- From the customers table
WHERE City = 'Paris' OR City = 'London' -- Limits the list to customers in paris/london
```

```
-----------------------------------------------------
/* 1.2 List all products stored in bottles. */
-----------------------------------------------------

SELECT
    p.ProductName --Select the product name
FROM Products p  -- From the products table
WHERE p.QuantityPerUnit LIKE '%bottle%' -
- The quantity includes the word 'bottle' (Will also include bottles)
```

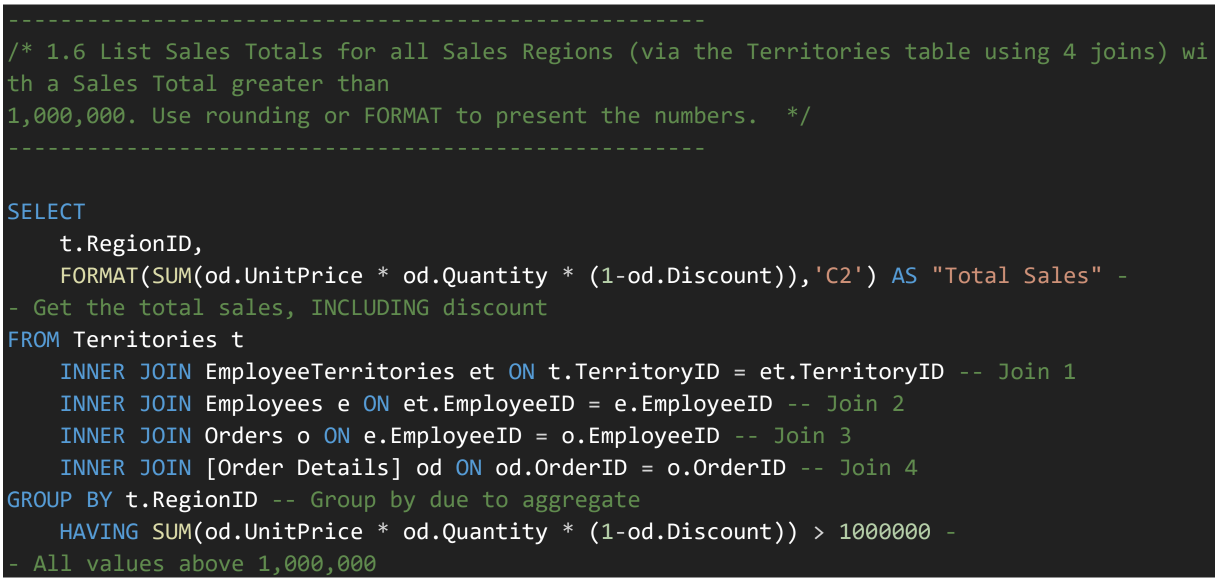
```
-----------------------------------------------------
/* 1.3 Repeat question above, but add in the Supplier Name and Country. */
-----------------------------------------------------

SELECT
    p.ProductName,

    s.CompanyName AS "Company Name", --
Get company name from supplier table, give colum title
    s.Country -- Get country from supplier table
FROM Products p
    INNER JOIN Suppliers s -- Join suppliers table
        ON s.SupplierID = p.SupplierID -- where the supplier ID's match
        WHERE QuantityPerUnit LIKE '%bottle%'
```



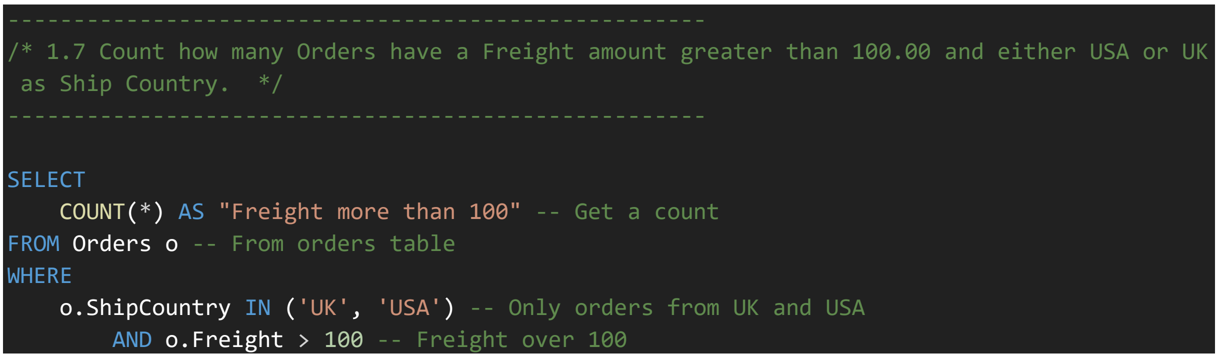
```
------------------------------------------------------
/* 1.4 Write an SQL Statement that shows how many products there are in each category.
Include Category Name in result set and list the highest number first. */
------------------------------------------------------

SELECT
    c.CategoryName, -- Get category name from categories table
    COUNT(p.CategoryID) AS "Number in Category"  --
Get the total number of ID's from products table
FROM Products p
    INNER JOIN Categories c ON p.CategoryID = c.CategoryID --
Join the categories table to the products table
GROUP BY c.CategoryName, p.CategoryID --
Group by the category name and ID to make the column sizes match
ORDER BY "Number in Category" DESC -- Order by number of categorys in descending order
```



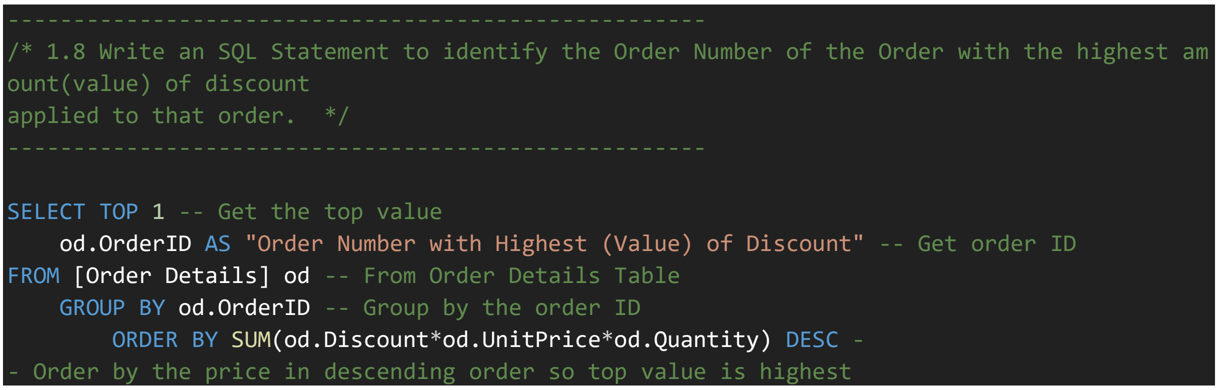
```
------------------------------------------------------
/* 1.5 List all UK employees using concatenation to join their title of courtesy, first nam
e and last name together.
Also include their city of residence */
------------------------------------------------------

SELECT
    e.TitleOfCourtesy + ' ' + e.FirstName + ' ' + e.LastName AS "Name", -
- Get the info required and concatonate
    e.City AS "City of Residence"
FROM Employees e -- From employees table
WHERE e.Country = 'UK' -- Only UK Employees
```



```
------------------------------------------------------
/* 1.6 List Sales Totals for all Sales Regions (via the Territories table using 4 joins) wi
th a Sales Total greater than
1,000,000. Use rounding or FORMAT to present the numbers.  */
------------------------------------------------------

SELECT
    t.RegionID,
    FORMAT(SUM(od.UnitPrice * od.Quantity * (1-od.Discount)),'C2') AS "Total Sales" -
- Get the total sales, INCLUDING discount
FROM Territories t
    INNER JOIN EmployeeTerritories et ON t.TerritoryID = et.TerritoryID -- Join 1
    INNER JOIN Employees e ON et.EmployeeID = e.EmployeeID -- Join 2
    INNER JOIN Orders o ON e.EmployeeID = o.EmployeeID -- Join 3
    INNER JOIN [Order Details] od ON od.OrderID = o.OrderID -- Join 4
GROUP BY t.RegionID -- Group by due to aggregate
    HAVING SUM(od.UnitPrice * od.Quantity * (1-od.Discount)) > 1000000 -
- All values above 1,000,000
```



```
-----------------------------------------------------
/* 1.7 Count how many Orders have a Freight amount greater than 100.00 and either USA or UK
 as Ship Country.  */
-----------------------------------------------------

SELECT
    COUNT(*) AS "Freight more than 100" -- Get a count
FROM Orders o -- From orders table
WHERE
    o.ShipCountry IN ('UK', 'USA') -- Only orders from UK and USA
        AND o.Freight > 100 -- Freight over 100
```

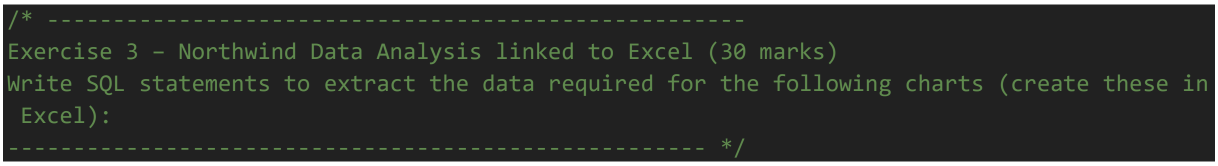


```
-----------------------------------------------------
/* 1.8 Write an SQL Statement to identify the Order Number of the Order with the highest am
ount(value) of discount
applied to that order.  */
-----------------------------------------------------

SELECT TOP 1 -- Get the top value
    od.OrderID AS "Order Number with Highest (Value) of Discount" -- Get order ID
FROM [Order Details] od -- From Order Details Table
    GROUP BY od.OrderID -- Group by the order ID
        ORDER BY SUM(od.Discount*od.UnitPrice*od.Quantity) DESC -
- Order by the price in descending order so top value is highest
```

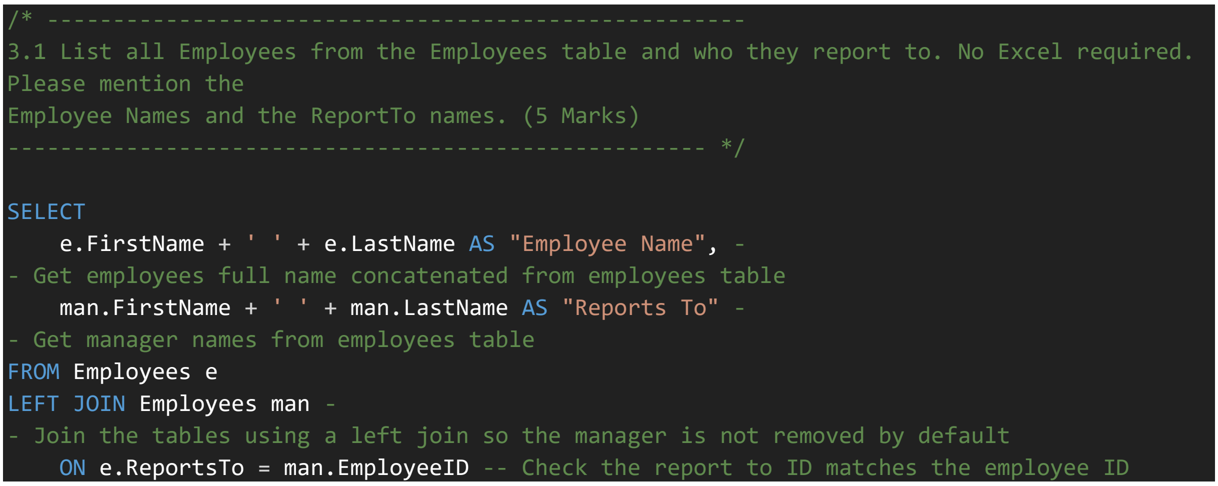
```
-- Exercise 2 – Create Spartans Table (20 marks – 10 each)
```

```
-----------------------------------------------------
/* 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, Fir
st 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
*/
-----------------------------------------------------

CREATE DATABASE mitul_db -- Create the database

USE mitul_db -- Use the database ust created

CREATE TABLE spartan_table -- Create the table
(
    spartan_id INT IDENTITY(1,1) PRIMARY KEY, -
- This will be the primary key and will auto increment
    spartan_info VARCHAR(50), -- Up to 50 characters
    title VARCHAR(5),
    first_name VARCHAR(15),
    last_name VARCHAR(15),
    university_name VARCHAR(30),
    course_taken VARCHAR(30),
    mark_achieved CHAR(3) NOT NULL -- 3 Characters ONLY, cannot be a NULL value
)
```

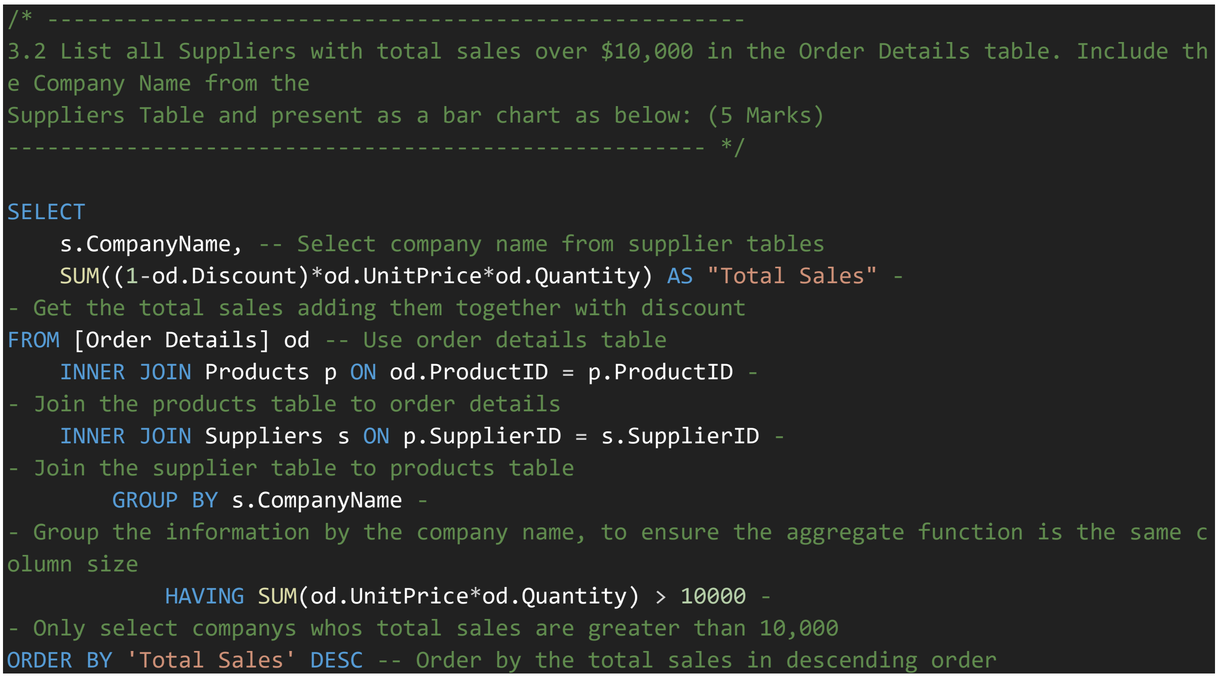
```
-----------------------------------------------------
/* 2.2 Write SQL statements to add the details of the Spartans in your course to the table
you have created.  */
-----------------------------------------------------

INSERT INTO spartan_table
(
    spartan_info,title,first_name,last_name,university_name,course_taken,mark_achieved -
- Insert into the table in THIS order
)
VALUES
(
    'Spartan Info','Mr','Mitul','Dattani','Aston','Maths with Computing','2.1' -
- Info to be entered
);

SELECT * FROM spartan_table; -- Display the table
```



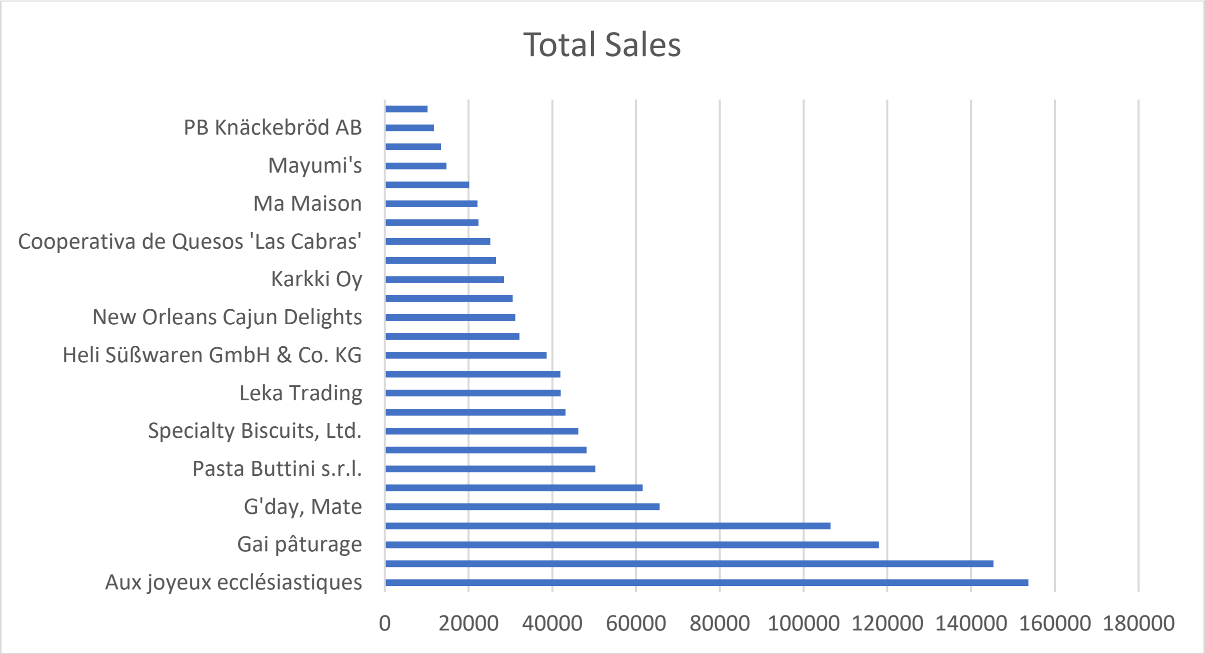
```
/* -----------------------------------------------------
Exercise 3 – Northwind Data Analysis linked to Excel (30 marks)
Write SQL statements to extract the data required for the following charts (create these in
 Excel):
----------------------------------------------------- */
```

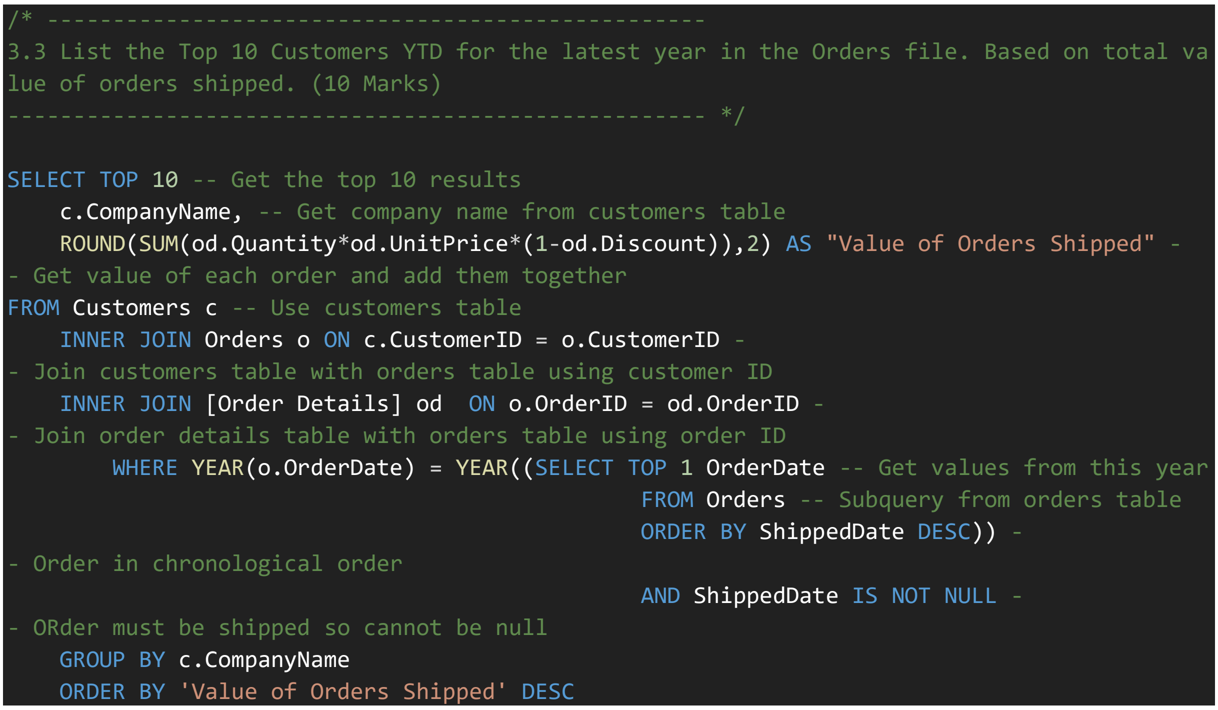


```
/* -----------------------------------------------------
3.1 List all Employees from the Employees table and who they report to. No Excel required.
Please mention the
Employee Names and the ReportTo names. (5 Marks)
----------------------------------------------------- */

SELECT
    e.FirstName + ' ' + e.LastName AS "Employee Name", -
- Get employees full name concatenated from employees table
    man.FirstName + ' ' + man.LastName AS "Reports To" -
- Get manager names from employees table
FROM Employees e
LEFT JOIN Employees man -
- Join the tables using a left join so the manager is not removed by default
    ON e.ReportsTo = man.EmployeeID -- Check the report to ID matches the employee ID
```

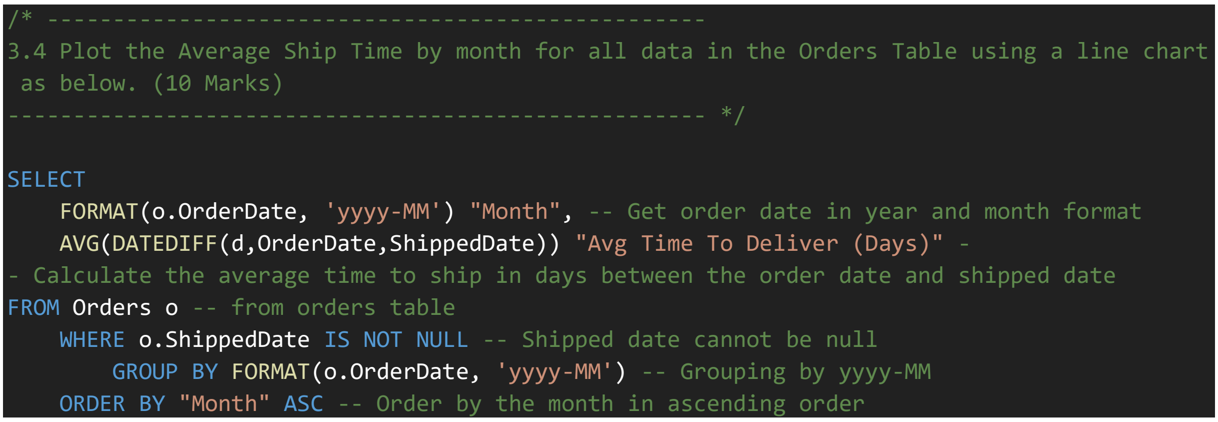


```
/* -----------------------------------------------------
3.2 List all Suppliers with total sales over $10,000 in the Order Details table. Include th
e Company Name from the
Suppliers Table and present as a bar chart as below: (5 Marks)
----------------------------------------------------- */

SELECT
    s.CompanyName, -- Select company name from supplier tables
    SUM((1-od.Discount)*od.UnitPrice*od.Quantity) AS "Total Sales" -
- Get the total sales adding them together with discount
FROM [Order Details] od -- Use order details table
    INNER JOIN Products p ON od.ProductID = p.ProductID -
- Join the products table to order details
    INNER JOIN Suppliers s ON p.SupplierID = s.SupplierID -
- Join the supplier table to products table
        GROUP BY s.CompanyName -
- Group the information by the company name, to ensure the aggregate function is the same c
olumn size
            HAVING SUM(od.UnitPrice*od.Quantity) > 10000 -
- Only select companys whos total sales are greater than 10,000
ORDER BY 'Total Sales' DESC -- Order by the total sales in descending order
```

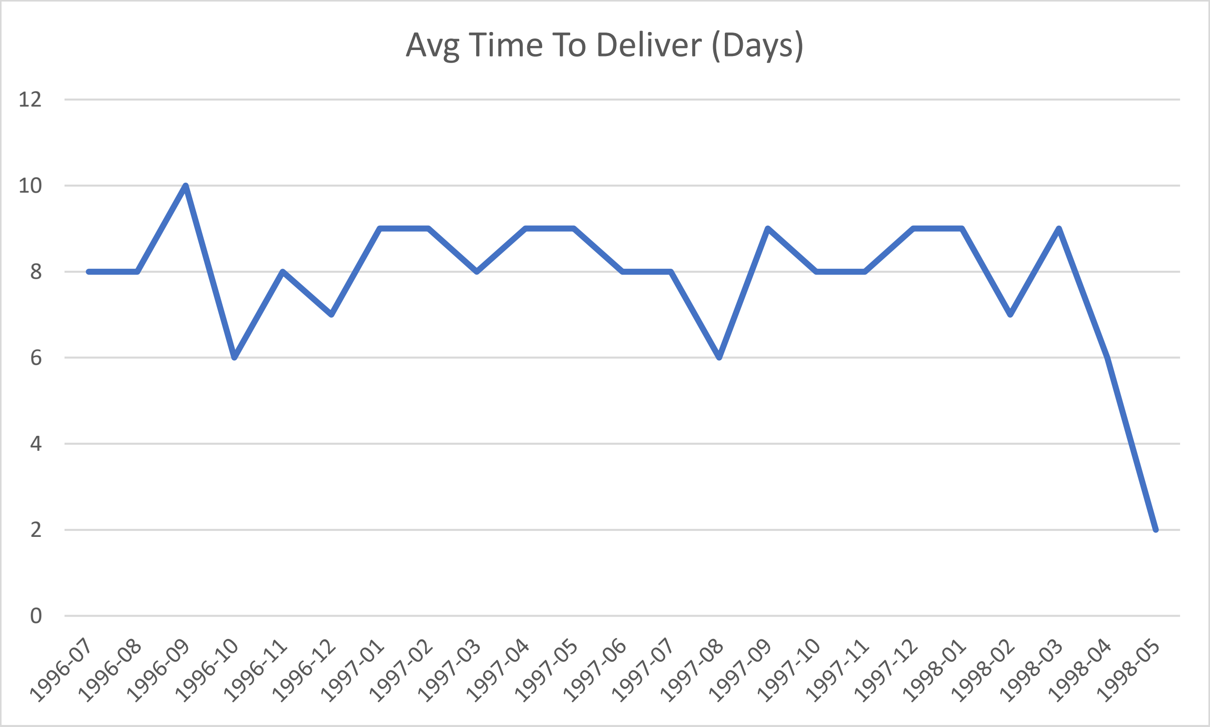




```
/* ---------------------------------------------------
3.3 List the Top 10 Customers YTD for the latest year in the Orders file. Based on total va
lue of orders shipped. (10 Marks)
----------------------------------------------------- */

SELECT TOP 10 -- Get the top 10 results
    c.CompanyName, -- Get company name from customers table
    ROUND(SUM(od.Quantity*od.UnitPrice*(1-od.Discount)),2) AS "Value of Orders Shipped" -
- Get value of each order and add them together
FROM Customers c -- Use customers table
    INNER JOIN Orders o ON c.CustomerID = o.CustomerID -
- Join customers table with orders table using customer ID
    INNER JOIN [Order Details] od  ON o.OrderID = od.OrderID -
- Join order details table with orders table using order ID
        WHERE YEAR(o.OrderDate) = YEAR((SELECT TOP 1 OrderDate -- Get values from this year
                                                FROM Orders -- Subquery from orders table
                                                ORDER BY ShippedDate DESC)) -
- Order in chronological order
                                                AND ShippedDate IS NOT NULL -
- ORder must be shipped so cannot be null
    GROUP BY c.CompanyName
    ORDER BY 'Value of Orders Shipped' DESC
```



```
/* ---------------------------------------------------
3.4 Plot the Average Ship Time by month for all data in the Orders Table using a line chart
 as below. (10 Marks)
--------------------------------------------------- */

SELECT
    FORMAT(o.OrderDate, 'yyyy-MM') "Month", -- Get order date in year and month format
    AVG(DATEDIFF(d,OrderDate,ShippedDate)) "Avg Time To Deliver (Days)" -
- Calculate the average time to ship in days between the order date and shipped date
FROM Orders o -- from orders table
    WHERE o.ShippedDate IS NOT NULL -- Shipped date cannot be null
        GROUP BY FORMAT(o.OrderDate, 'yyyy-MM') -- Grouping by yyyy-MM
    ORDER BY "Month" ASC -- Order by the month in ascending order
```


Avg Time To Deliver (Days)