Exercise 1

```sql
USE Northwind;                                   --Run at start to access databases

SELECT CustomerID, CompanyName, Address          --1.1
FROM Customers
WHERE City IN ('London', 'Paris')

SELECT ProductName                               --1.2
FROM Products
WHERE QuantityPerUnit LIKE '%bottle%'

SELECT p.ProductName, s.CompanyName, s.Country   --1.3
FROM Products p
INNER JOIN Suppliers s ON s.SupplierID = p.SupplierID
WHERE QuantityPerUnit LIKE '%bottle%'

SELECT c.CategoryName, COUNT(p.ProductID) AS 'Category count'   --1.4
FROM Categories c
INNER JOIN Products p ON c.CategoryID = p.CategoryID
GROUP BY c.CategoryName
ORDER BY 'Category count' DESC

SELECT CONCAT(TitleOfCourtesy, FirstName, ' ', LastName)        --1.5
    AS 'Employee Name', City
FROM Employees
WHERE Country IN ('UK')

SELECT r.RegionDescription AS 'Region',          --1.6
    ROUND(SUM((od.UnitPrice * od.Quantity) * (1 - od.Discount)), 0) AS 'Total Order'
FROM [Order Details] od
INNER JOIN Orders o ON o.OrderID = od.OrderID
INNER JOIN Employees e ON e.EmployeeID = o.EmployeeID
INNER JOIN EmployeeTerritories et ON e.EmployeeID = et.EmployeeID
INNER JOIN Territories t ON t.TerritoryID = et.TerritoryID
INNER JOIN Region r ON r.RegionID = t.RegionID
GROUP BY r.RegionDescription
HAVING SUM((od.UnitPrice * od.Quantity) * (1 - od.Discount)) > 1000000

SELECT COUNT(OrderID)                            --1.7
    AS 'Orders with freight over 100 and ship country UK or USA'
FROM Orders
WHERE Freight > 100 AND ShipCountry IN ('UK', 'USA')

SELECT TOP 1 OrderID,                            --1.8
    SUM(ROUND((UnitPrice * Quantity) * Discount, 0)) AS "Highest discount"
FROM [Order Details]
GROUP BY OrderID
ORDER BY 'Highest Discount' DESC
```

Exercise 2

```sql
CREATE DATABASE jordan_db;                          --Creates empty database

USE jordan_db;                                      --Selects created database to use

CREATE TABLE spartan_table                          --Creates table in database
(
    spartan_id INT IDENTITY(1,1) PRIMARY KEY,
    chosen_title VARCHAR(4),
    first_name VARCHAR(8) NOT NULL,
    last_name VARCHAR(12) NOT NULL,
    university VARCHAR(18),
    course_studied VARCHAR(20),
    grade_achieved CHAR(3),
    sparta_stream VARCHAR(6)
)

ALTER TABLE spartan_table                           --Add a default for sparta_stream
ADD CONSTRAINT Default_Stream DEFAULT 'DevOps' for sparta_stream

ALTER TABLE spartan_table                           --Add default for course_studied
ADD CONSTRAINT Default_Course DEFAULT 'Computer Science' for course_studied

INSERT INTO                                         --Insert values for specified fields
spartan_table(first_name, last_name, university, course_studied, grade_achieved)
VALUES
('Jordan','Clarke','Salford','Physics','2:1'),
('Benjamin','Ranson','Essex','Computer Science','1st'),
('Andrew','Asare','London Met','Computer Science','2:1'),
('Dunni','Adebusuyi','Goldsmiths','Computer Science','2:2'),
('William','King','Swansea','Computer Science','1st'),
('Ayaz','Yar','Exeter','PPE','2:1'),
('Arun','Panesar','De Montford','Software Engineering','1st'),
('Jose','Torres','Madrid','Computer Science','2:1')

INSERT INTO spartan_table(                          --Update table to add further entry
first_name, last_name, university, course_studied, grade_achieved
)
VALUES ('Georgia','Bradshaw','Salford','Physics','1st')

SELECT * FROM spartan_table                         --Displays table with all fields
ORDER BY last_name

DROP TABLE spartan_table                            --Drops the table, erasing all fields
```

Exercise 3

```sql
USE Northwind;                                             --Select database

SELECT CONCAT(e.FirstName, ' ', e.LastName) AS 'Employee name',   --3.1
    CONCAT(r.FirstName, ' ', r.LastName) AS 'Reports to'
FROM Employees r
RIGHT JOIN Employees e ON r.EmployeeID = e.ReportsTo

SELECT s.CompanyName,                                      --3.2
    ROUND(SUM((od.UnitPrice * od.Quantity)*(1 - od.Discount)), 0) AS 'Total Sales'
FROM Suppliers s
INNER JOIN Products p ON s.SupplierID = p.SupplierID
INNER JOIN [Order Details] od ON p.ProductID = od.ProductID
GROUP BY s.CompanyName
HAVING SUM((od.UnitPrice * od.Quantity)*(1 - od.Discount)) > 10000
ORDER BY 'Total Sales' DESC

SELECT TOP 10 c.CompanyName,                               --3.3
    YEAR(o.OrderDate) AS 'Year'
FROM Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
INNER JOIN [Order Details] od ON od.OrderID = o.OrderID
WHERE
    (YEAR(o.OrderDate) =
    (SELECT TOP 1 YEAR(OrderDate)
    FROM Orders
    ORDER BY YEAR(OrderDate) DESC))
GROUP BY c.CompanyName, YEAR(o.OrderDate)
ORDER BY SUM((od.UnitPrice * od.Quantity)*(1 - od.Discount))

SELECT FORMAT(OrderDate, 'yy/MM') AS 'Month',              --3.4
    AVG(DATEDIFF(dd, OrderDate, ShippedDate))AS 'Days to ship'
FROM Orders
GROUP BY FORMAT(OrderDate, 'yy/MM')
```
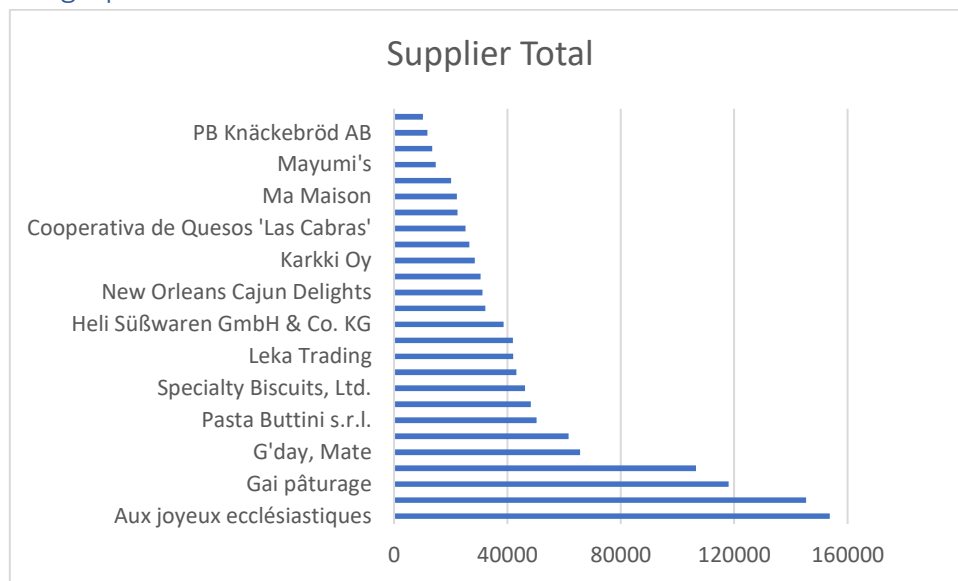
Exercise 3.2 graph

Supplier Total

Supplier	
PB Knäckebröd AB	
Mayumi's	
Ma Maison	
Cooperativa de Quesos 'Las Cabras'	
Karkki Oy	
New Orleans Cajun Delights	
Heli Süßwaren GmbH & Co. KG	
Leka Trading	
Specialty Biscuits, Ltd.	
Pasta Buttini s.r.l.	
G'day, Mate	
Gai pâturage	
Aux joyeux ecclésiastiques	

Horizontal axis: 0, 40000, 80000, 120000, 160000

Exercise 3.4 graph

Average Ship Time by Month

Vertical axis: 0, 2, 4, 6, 8, 10, 12

Horizontal axis: 96/07, 96/08, 96/09, 96/10, 96/11, 96/12, 97/01, 97/02, 97/03, 97/04, 97/05, 97/06, 97/07, 97/08, 97/09, 97/10, 97/11, 97/12, 98/01, 98/02, 98/03, 98/04, 98/05