SQL Mini Project

```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 CustomerID, CompanyName, Address, City, Region, PostalCode, Country
FROM Customers WHERE City IN ('Paris', 'London')
```

```sql
-- 1.2 List all products stored in bottles.
SELECT * FROM Products WHERE QuantityPerUnit LIKE '%bottle%'
```

```sql
-- 1.3 Repeat question above, but add in the Supplier Name and Country.
SELECT Products.*,
    Suppliers.CompanyName as "Supplier Company",
    Suppliers.Country as "Supplier Country"
FROM Products
INNER JOIN Suppliers ON Products.SupplierID = Suppliers.SupplierID
WHERE QuantityPerUnit LIKE '%bottle%'
```

```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 p.CategoryID, COUNT(*) AS "Products in Category", c.CategoryName
FROM Products p
INNER JOIN Categories c ON p.CategoryID = c.CategoryID
GROUP BY p.CategoryID, c.CategoryName
ORDER BY "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(TitleOfCourtesy, ' ', FirstName, ' ', LastName) as "Employee", City
FROM Employees WHERE 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 r.RegionDescription,
    ROUND(SUM(od.UnitPrice * od.Quantity * (1 - od.Discount)), 2) AS "Sales Total"
FROM [Order Details] od
INNER JOIN Orders o ON od.OrderID = o.OrderID
INNER JOIN EmployeeTerritories et ON o.EmployeeID = et.EmployeeID
INNER JOIN Territories t ON et.TerritoryID = t.TerritoryID
INNER JOIN Region r ON t.RegionID = r.RegionID
GROUP BY r.RegionDescription
HAVING ROUND(SUM(od.UnitPrice * od.Quantity * (1 - od.Discount)), 2) > 1000000
```

```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(*) AS
"Orders have a Freight amount greater than 100.00 and either USA or UK as Ship Country"
FROM Orders
WHERE Freight > 100.00 AND ShipCountry IN ('USA', 'UK')
```

```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 OrderID, SUM(UnitPrice * Quantity * Discount) AS "Discount Applied"
FROM [Order Details]
GROUP BY OrderID
ORDER BY "Discount Applied" DESC
```

SQL Mini Project

```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.  */


USE georgeRepole_db;

-- IDENTITY is autoincremented operator (1,1) means starts at 1 and increments by 1
CREATE TABLE spartan_table(
    person_id INT IDENTITY(1,1),
    PRIMARY KEY(person_id),
    title VARCHAR(5),
    first_name VARCHAR(20),
    last_name VARCHAR(20),
    university VARCHAR(50),
    course VARCHAR(50),
    mark_achieved VARCHAR(5)
);
```

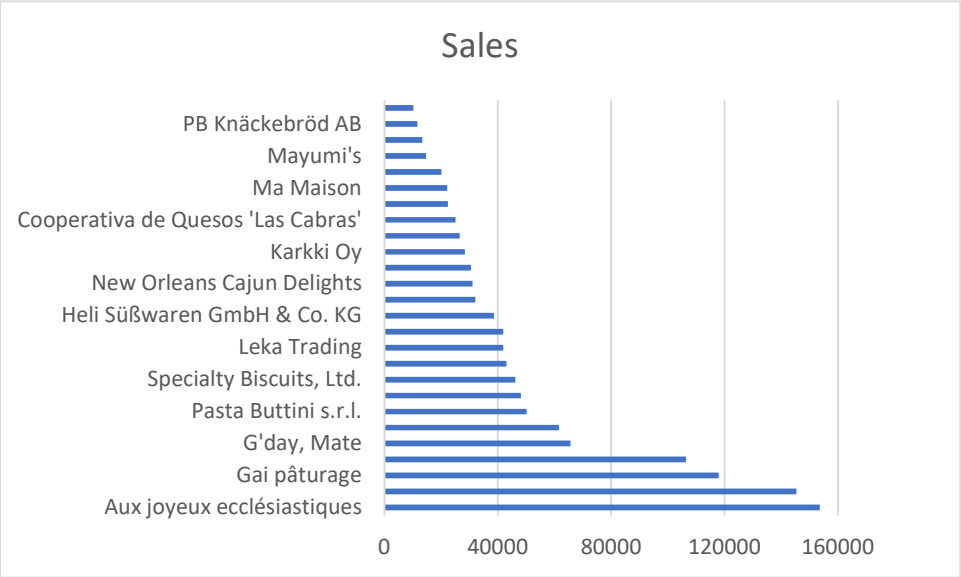
```sql
-- 2.2 Write SQL statements to add the details of the Spartans
-- in your course to the table you have created.

INSERT INTO spartan_table VALUES
    ('Mr.', 'Alexander', 'Legon', 'Uni', 'Course', '1st'),
    ('Mr.', 'Jian', 'Cruz', 'Uni', 'Course', '1st'),
    ('Mr.', 'Alex', 'Barber-Lynch', 'Uni', 'Course', '1st'),
    ('Mr.', 'Alex', 'Chang', 'Uni', 'Course', '1st'),
    ('Mr.', 'Sotiris', 'Loizou', 'Uni', 'Course', '1st'),
    ('Mr.', 'Adrian', 'Wong', 'Uni', 'Course', '1st'),
    ('Mr.', 'Thomas', 'Canfield', 'Uni', 'Course', '1st'),
    ('Mr.', 'Thomas', 'Kirkwood', 'Uni', 'Course', '1st'),
    ('Mr.', 'Karim', 'Wohler', 'Uni', 'Course', '1st');
```

SQL Mini Project

```sql
-- 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 CONCAT(e.FirstName, ' ', e.LastName) AS "Employee Name",
    CONCAT(em.FirstName, ' ', em.LastName) AS "ReportTo Name"
FROM Employees e
LEFT JOIN Employees em ON e.ReportsTo = em.EmployeeID
```

```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 as below: (5 Marks)
SELECT s.CompanyName,
    ROUND(SUM(od.UnitPrice * od.Quantity * (1 - od.Discount)), 2) AS 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 ROUND(SUM(od.UnitPrice * od.Quantity * (1 - od.Discount)), 2) > 10000
ORDER BY Sales DESC
```




```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. (10 Marks)
SELECT TOP 10 c.CustomerID, c.CompanyName,
ROUND(SUM(od.UnitPrice * od.Quantity * (1 - od.Discount)), 2) AS "Sales YTD"
FROM Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
INNER JOIN [Order Details] od on o.OrderID = od.OrderID
WHERE YEAR(o.ShippedDate) = (SELECT MAX(YEAR(ShippedDate)) From Orders)
    AND o.ShippedDate IS NOT NULL
GROUP BY c.CustomerID, c.CompanyName
ORDER BY ROUND(SUM(od.UnitPrice * od.Quantity * (1 - od.Discount)), 2) DESC
```

SQL Mini Project

```sql
-- 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 AVG(CAST(DATEDIFF(d, OrderDate, ShippedDate) AS DECIMAL (10, 2)))
    AS "Average Ship Time Per Month",
    FORMAT(OrderDate, 'MMM-yy') AS "Month-Year",  MONTH(OrderDate), YEAR(OrderDate)
FROM Orders
WHERE ShippedDate IS NOT NULL
GROUP BY FORMAT(OrderDate, 'MMM-yy'), MONTH(OrderDate), YEAR(OrderDate)
ORDER BY YEAR(OrderDate), MONTH(OrderDate) ASC
```