## Introduction

This exercise requires you to know the following aspects of SQL:

|  |  |
| --- | --- |
| CREATE TABLE | Concatenation |
| SQL Data Types | Formatting dates and numbers |
| INSERT INTO | Column aliases |
| SELECT | Simple JOIN statements |
| WHERE clause | Complex JOIN statements |
| LIKE and wildcards | Subquery |

## Exercise 1 – Northwind Queries (40 marks: 5 for each question)

* 1. Write a query that lists all Customers in either Paris or London. Include Customer ID, Company Name and all address fields.

USE

Northwind;

SELECT CustomerID, ContactName, CompanyName, Address, City, PostalCode, Country

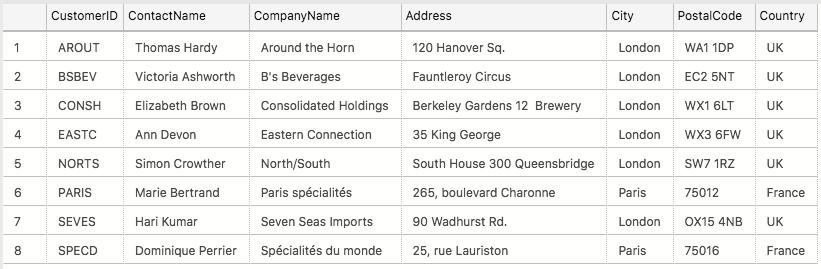
FROM

Customers

WHERE

City = 'Paris' OR City = 'London';

GROUP BY CustomerID, ContactName, CompanyName, Address, City, PostalCode, Country;



* 1. List all products stored in bottles.

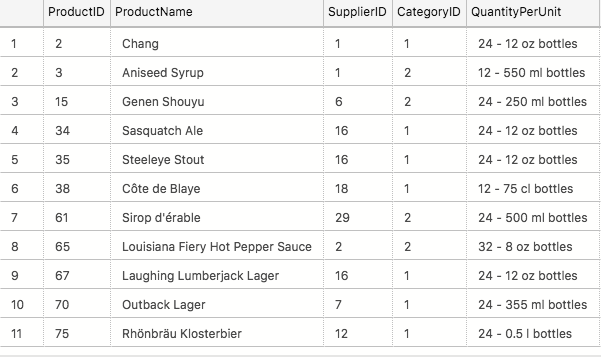
SELECT \*

FROM

Products

WHERE

QuantityPerUnit LIKE '%bottles%';



* 1. Repeat question above but add in the Supplier Name and Country.

SELECT

Suppliers.CompanyName , Suppliers.Country, Products.QuantityPerUnit

FROM

Products

INNER JOIN

Suppliers ON Products.SupplierID = Suppliers.SupplierID

WHERE QuantityPerUnit LIKE '%bottles%';



* 1. 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. (use count)

SELECT

Categories.CategoryName ,

COUNT(Products.UnitsInStock) AS 'All Products'

FROM

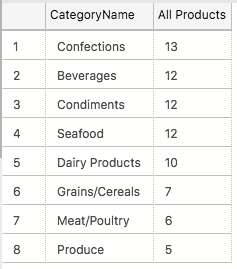
Products

INNER JOIN

Categories ON Products.CategoryID = Categories.CategoryID

GROUP BY Categories.CategoryName

ORDER BY 2 DESC;



* 1. List all UK employees using concatenation to join their title of courtesy, first name and last name together. Also include their city of residence.

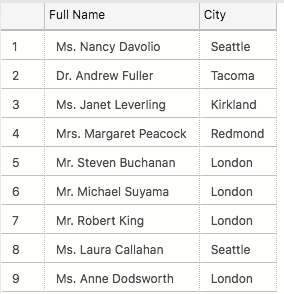
SELECT

TitleOfCourtesy + ' ' + FirstName + ' ' + LastName AS "Full Name",

City as "City"

FROM

Employees;



* 1. 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,

cast(SUM(od.UnitPrice \* od.Quantity) as DECIMAL(7,0)) as "Total Sales"

FROM

[Order Details] od

INNER JOIN

Orders o ON od.OrderID = o.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

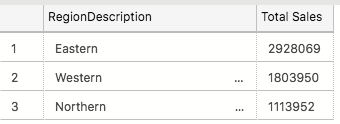
GROUP BY

r.RegionDescription

HAVING

SUM(od.UnitPrice \* od.Quantity) >1000000

ORDER BY 2 DESC;



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

SELECT

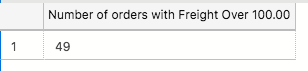
COUNT(OrderID) AS [Number of orders with Freight Over 100.00]

FROM

Orders

WHERE

Freight > 100 AND(ShipCountry = 'USA' OR ShipCountry = 'UK');



* 1. Write an SQL Statement to identify the Order Number of the Order with the highest amount of discount applied to that order.

SELECT

OrderID,

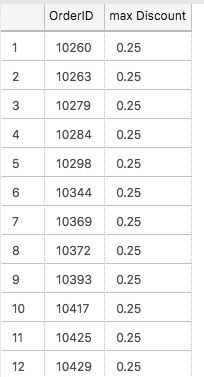
MAX(Discount) as "max Discount"

FROM

[Order Details]

GROUP BY OrderID

ORDER BY 2 DESC;



## 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, 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.

-- USED TO CREATE TABLE

CREATE TABLE Spartans\_Table (

PersonID INT NOT NULL IDENTITY PRIMARY KEY,

Title VARCHAR(5),

[First Name ]VARCHAR(20),

[Last Name] VARCHAR(30),

University VARCHAR(50),

[Univeristy Course] VARCHAR(50),

Grade VARCHAR(10),

);

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

-- USED INSERT INTO Spartans\_Table

INSERT INTO Spartans\_Table

( PersonID

,Title

,[First Name]

,[Last Name]

,University

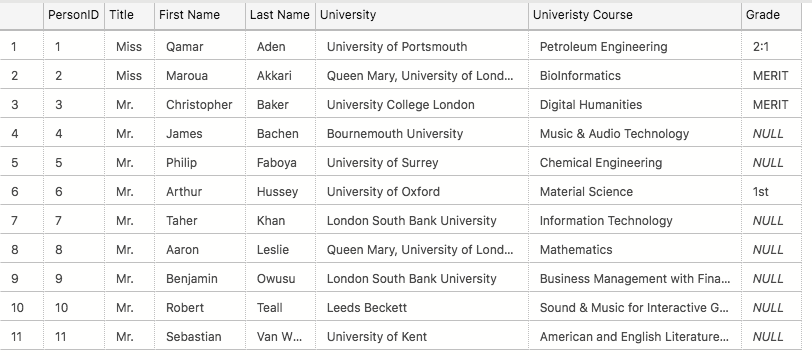
,[Univeristy Course]

,Grade)

VALUES

('','','','','','','');

-- I then put the data for each column in the quotes in the correct order



## 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. (5 Marks)

SELECT

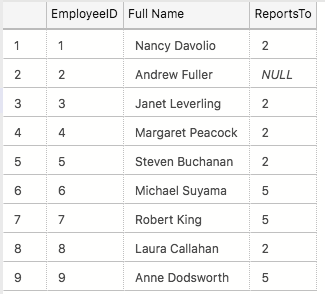
[EmployeeID],

[FirstName] + ' '+ [LastName] as "Full Name"

,[ReportsTo]

FROM

[Northwind].[dbo].[Employees];



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

Suppliers.CompanyName,

SUM([Order Details].UnitPrice\*Quantity)AS [Total sales]

FROM

[Order Details]

INNER JOIN

Products ON products.ProductID= [Order Details] .ProductID

INNER JOIN

Suppliers ON Products.SupplierID= Suppliers.SupplierID

GROUP BY Suppliers.CompanyName

HAVING SUM([Order Details].UnitPrice\*Quantity) > 10000

ORDER BY 2 DESC;

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)

Customers.ContactName,

SUM([Order Details].Quantity) AS "Total Shipped"

FROM

Customers

INNER JOIN

Orders ON Customers.CustomerID = Orders.CustomerID

INNER JOIN

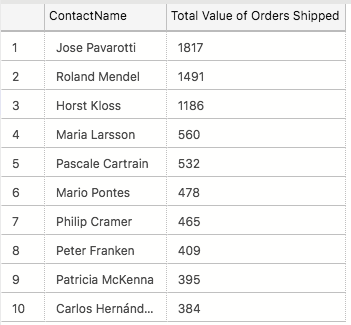
[Order Details] ON [Order Details].[OrderID]= Orders.[OrderID]

WHERE

ShippedDate >= '01/01/1998 00:00:00 AM'

GROUP BY Customers.ContactName

ORDER BY 2 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)

-- All the times were 00:00:00:00 so I used the average day instead

SELECT

AVG(DATEPART(DAY, ShippedDate)) AS DAY, DATEPART(month, ShippedDate) AS month, DATEPART(year, ShippedDate) AS year

FROM

Orders

GROUP BY DATEPART(month, ShippedDate), DATEPART(year, ShippedDate);

## Standards (10 marks)

Remember to apply all the following standards:

* Use consistent capitalisation and indentation of SQL Statements
* Use concise and consistent table alias names
* Use column aliases to ensure tidy column headings (spaces and consistent capitalisation)
* Concatenate any closely related columns e.g. First Name and Last Name or Address and City etc
* Put comments throughout