**SQL Project BY Saleh Sandhu**

--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, c.CompanyName, c.Address, c.City --Selecting columns

FROM Customers c --From table customer

WHERE c.City IN ('London', 'Paris') --Choosing specific cities

--1.2 List all products stored in bottles.

SELECT \* --Selecting all products

FROM Products p --from table products

WHERE CHARINDEX('bottles', p.QuantityPerUnit) >0 --Where there is a bottle string

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

SELECT p.\*, s.Country, s.CompanyName -- Selecting values

FROM Products p INNER JOIN Suppliers s --From various tables

ON p.SupplierID=s.SupplierID --Connecting foreign keys

WHERE CHARINDEX('bottles', p.QuantityPerUnit) >0 --Where there is a bottle string

--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, COUNT(p.CategoryID) AS "Total" --Select values

FROM Categories c LEFT JOIN Products p --From tables

ON c.CategoryID=p.CategoryID --Connecting foreign key

GROUP BY c.CategoryName, c.CategoryID --Grouping values

ORDER BY "Total" DESC --Ordering

--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 "Name", e.City --Selecting values

FROM Employees e --From table

WHERE e.Country = 'UK'--Where they are from UK

--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.Quantity \* od.UnitPrice \* 1-od.Discount),2) AS "Total Sales" --Selecting variable

FROM  [Order Details] od --From table

INNER JOIN Orders o ON o.OrderID = od.OrderID --From various tables

INNER JOIN EmployeeTerritories et ON et.EmployeeID = o.EmployeeID

INNER JOIN Territories t ON t.TerritoryID = et.TerritoryID

INNER JOIN Region r ON r.RegionID = t.RegionID

GROUP BY r.RegionDescription --Group the data

HAVING ROUND(SUM(od.Quantity \* od.UnitPrice \* 1-od.Discount),2) >1000000 --Having clause used because where cannot be used with aggregate functions

--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" --counting number of variables

FROM Orders o --From table

WHERE o.Freight > 100.00 AND o.ShipCountry IN ('UK', 'USA') --adding conditions

--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.MAX

SELECT TOP 1 od.OrderID, od.UnitPrice \* od.Quantity \* od.Discount /100 AS "Value" --Selecting top variables

FROM [Order Details] od --From order details table

ORDER BY "Value" DESC --Order by descending

--2.1 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.

CREATE DATABASE saleh\_db --Creating database

USE saleh\_db --Using database

CREATE TABLE spartans --Creating table

(

    spartan\_ID INT PRIMARY KEY IDENTITY,

    title VARCHAR(10), --Adding in rows

    first\_name VARCHAR(20),

    last\_name VARCHAR(20),

    university\_attended VARCHAR(50),

    course\_taken VARCHAR(50),

    mark\_achieved VARCHAR(20),

)

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

INSERT INTO spartans VALUES --inserting values to table

(

    'Mr', 'Reece', 'Louch', 'University of Warwick', 'Computer Science', '2:2'

),

(

    'Mr', 'Saleh', 'Sandhu', 'University of Westminster', 'Computer Science', '2:1'

),

(

    'Mr', 'Svilen', 'Petrov', 'London Metropolitan University', 'BSc Computing', 'First'

),

(

    'Mr', 'Toyin', 'Ajani', 'University of Bath', 'Chemical Engineering', 'First'

),

(

    'Mr', 'Ben', 'Swift', 'Nottingham Trent University', 'Computer Science', '2:1'

),

(

    'Mr', 'Abdullah', 'Muhammad', 'University of Southampton', 'Physics', 'First'

),

(

    'Mr', 'Chris', 'Cunningham', 'Loughborough', 'Computer Science', '2:1'

),

(

    'Mr', 'Dami', 'Oshidele', 'Electronic Engineering with Management BEng', 'Kings College London','2:1'

),

(

    'Ms', 'Janja', 'Kovacevic', 'University of Massachusetts Amherst', 'Computer Sience and Computational Mathematics', '3.9'

),

(

    'Mr','Shahid','Enayat','Brunel University', 'Electronic and Electrical Engineering','2:2'

),

(

    'Mr', 'Emmanuel', 'Buraimo', 'Kings College London', 'Computer Science Bsc', '2:1'

)

SELECT \* FROM spartans --viewing table

--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(em.FirstName, ' ', em.LastName) AS "Reporting to" --Seleting variables

FROM Employees e --From table employee

LEFT JOIN Employees em ON e.ReportsTo=em.EmployeeID --Connecting foreign keys

--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, SUM(od.Quantity \* od.UnitPrice \* 1-od.Discount) AS "Total" --selecting variables

FROM [Order Details] od --from table

INNER JOIN Products p ON od.ProductID=p.ProductID --connecting foreign keys

INNER JOIN Suppliers s ON p.SupplierID=s.SupplierID

GROUP BY s.CompanyName --grouping value

HAVING SUM(od.Quantity \* od.UnitPrice \* 1-od.Discount) > 10000 --having use because where cannot be used with aggregate

ORDER BY "Total" DESC --order by descending

--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, SUM(od.UnitPrice \* od.Quantity \* 1-od.Discount) AS "Value" --selecing top 10 values

FROM [Order Details] od INNER JOIN Orders o ON od.OrderID=o.OrderID --from table

INNER JOIN Customers c ON o.CustomerID=c.CustomerID --connecting foreign key

GROUP BY c.CompanyName --grouping values

HAVING YEAR(MAX(o.ShippedDate)) = '1998' --having function used because where cannot be used with aggregate keywords

ORDER By "Value" DESC --order by descending

--3.4 Plot the Average Ship Time by month for all data in the Orders Table using a line chart as below

SELECT YEAR(o.OrderDate) AS "Year", MONTH(o.OrderDate) AS "Month", AVG(DATEDIFF(d, o.OrderDate, o.ShippedDate)) AS "Days to Ship" --electing varaibles

FROM Orders o --from table

GROUP BY MONTH(o.OrderDate), YEAR(o.OrderDate) --grouping values

ORDER BY "Year" ASC, "Month" ASC --order by descending