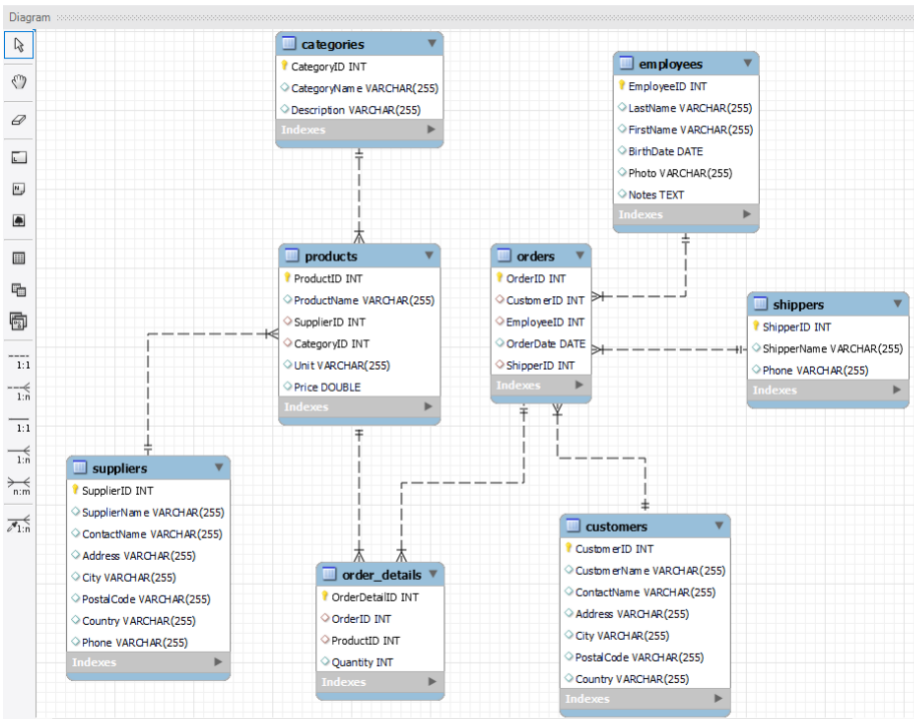
Diagram
categories
CategoryID INT
CategoryName VARCHAR(255)
Description VARCHAR(255)
Indexes
employees
EmployeeID INT
LastName VARCHAR(255)
FirstName VARCHAR(255)
BirthDate DATE
Photo VARCHAR(255)
Notes TEXT
Indexes
products
ProductID INT
ProductName VARCHAR(255)
SupplierID INT
CategoryID INT
Unit VARCHAR(255)
Price DOUBLE
Indexes
orders
OrderID INT
CustomerID INT
EmployeeID INT
OrderDate DATE
ShipperID INT
Indexes
shippers
ShipperID INT
ShipperName VARCHAR(255)
Phone VARCHAR(255)
Indexes
suppliers
SupplierID INT
SupplierName VARCHAR(255)
ContactName VARCHAR(255)
Address VARCHAR(255)
City VARCHAR(255)
PostalCode VARCHAR(255)
Country VARCHAR(255)
Phone VARCHAR(255)
Indexes
order_details
OrderDetailID INT
OrderID INT
ProductID INT
Quantity INT
Indexes
customers
CustomerID INT
CustomerName VARCHAR(255)
ContactName VARCHAR(255)
Address VARCHAR(255)
City VARCHAR(255)
PostalCode VARCHAR(255)
Country VARCHAR(255)
Indexes
1:1
1:n
1:1
1:n
n:m
1:n

Individual Assignment 3

Schema Background.

All of the questions in Assignment 3 are based on the w3schools schema. This schema models a product order management system.

There are many different products. The information of these products is listed in the table PRODUCTS.

Each of the products is provided by a supplier. The supplier information is in the table SUPPLIERS.

Each of the products belongs to a category. The category information is in the table CATEGORIES.

There are many customers. The customer information is in the table CUSTOMERS.

Customers place orders to buy products. An order is processed by an employee, and shipped by a shipper. The order information is in the table ORDERS. The employee information is in the table EMPLOYEES. The shipper information is in the table SHIPPERS.

A customer can buy a set of products in a single order. The different product IDs and quantities are recorded in the ORDER_DETAILS table.

For example, customer 1 bought three product 1 and five product 2 in the first order, and then bought eight product 1 and ten product 2 in the second order. So in this case, the data records in the ORDERS table would be

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
1	1	1	2021-01-01	1
2	1	1	2021-02-01	1

The data records in the ORDER_DETAILS table would be

OrderDetailID	OrderID	ProductID	Quantity
1	1	1	3
2	1	2	5
3	2	1	8
4	2	2	10

For each of the following questions, use **ONE** SQL statement to query the database and get the answer. Your answer needs to have both of the SQL command and a screenshot of the running result (No need to show all rows; just the first few rows would suffice).

Hint: “One SQL statement” means that, for some questions, you might need to use subqueries.

Grading policy:

50% - the correctness of the results;

50% - SQL syntax (partial credit will be given)

Question 1 (10 points). Select and show the product name, unit, and price of any product which has a price greater than or equal to 25. Order your result by product price, in descending order.

Input:

```
1 •  USE w3schools;
2
3 •  SELECT p.ProductName, p.Unit, p.Price
4    FROM products p
5    WHERE Price >= 25
6    ORDER BY Price desc;
```

Output:

ProductName	Unit	Price
Côte de Blaye	12 - 75 cl bottles	263.5
Thüringer Rostbratwurst	50 bags x 30 sausgs.	123.79
Mishi Kobe Niku	18 - 500 g pkgs.	97
Sir Rodney's Marmalade	30 gift boxes	81
Carnarvon Tigers	16 kg pkg.	62.5
Raclette Courdavault	5 kg pkg.	55
Manjimup Dried Apples	50 - 300 g pkgs.	53
Tarte au sucre	48 pies	49.3
Ipoh Coffee	16 - 500 g tins	46
Rössle Sauerkraut	25 - 825 g cans	45.6
Schoggi Schokolade	100 - 100 g pieces	43.9
Vegie-spread	15 - 625 g jars	43.9
Northwoods Cranberry…	12 - 12 oz jars	40
Alice Mutton	20 - 1 kg tins	39
Queso Manchego La P…	10 - 500 g pkgs.	38
Gnocchi di nonna Alice	24 - 250 g pkgs.	38
Gudbrandsdalsost	10 kg pkg.	36
Mozzarella di Giovanni	24 - 200 g pkgs.	34.8
Camembert Pierrot	15 - 300 g rounds	34
Wimmers gute Semmel…	20 bags x 4 pieces	33.25
Perth Pasties	48 pieces	32.8
Mascarpone Fabioli	24 - 200 g pkgs.	32
Gumbär Gummibärchen	100 - 250 g bags	31.23
Ikura	12 - 200 ml jars	31
Uncle Bob's Organic Dr…	12 - 1 lb pkgs.	30
Sirop d'érable	24 - 500 ml bottles	28.5

Question 2 (10 points). Display the order ID, customer name, shipper name, and employee's first and last name of all the orders. Make sure that

Input:

a. The columns to display employee's first and last names are EmployeeFirstName and EmployeeLastName.

b. Order the results by order ID, in ascending order.

Input:

```
SELECT o.OrderID, c.CustomerName, s.ShipperName, e.FirstName AS EmployeeFirstName, e.LastName AS EmployeeLastName
FROM orders o JOIN customers c JOIN shippers s JOIN employees e
ON o.CustomerID = c.CustomerID
AND o.ShipperID = s.ShipperID
AND o.EmployeeID = e.EmployeeID
ORDER BY o.OrderID asc;
```

Output:

OrderID	CustomerName	ShipperName	EmployeeFirstName	EmployeeLastName
10248	Wilman Kala	Federal Shipping	Steven	Buchanan
10249	Tradição Hipermercados	Speedy Express	Michael	Suyama
10250	Hanari Carnes	United Package	Margaret	Peacock
10251	Victuailles en stock	Speedy Express	Janet	Leverling
10252	Suprêmes délices	United Package	Margaret	Peacock
10253	Hanari Carnes	United Package	Janet	Leverling
10254	Chop-suey Chinese	United Package	Steven	Buchanan
10255	Richter Supermarkt	Federal Shipping	Anne	Dodsworth
10256	Wellington Importadora	United Package	Janet	Leverling
10257	HILARIÓN-Abastos	Federal Shipping	Margaret	Peacock
10258	Ernst Handel	Speedy Express	Nancy	Davolio
10259	Centro comercial Moct…	Federal Shipping	Margaret	Peacock
10260	Old World Delicatessen	Speedy Express	Margaret	Peacock
10261	Que Delícia	United Package	Margaret	Peacock
10262	Rattlesnake Canyon Gr…	Federal Shipping	Laura	Callahan
10263	Ernst Handel	Federal Shipping	Anne	Dodsworth
10264	Folk och fä HB	Federal Shipping	Michael	Suyama
10265	Blondel père et fils	Speedy Express	Andrew	Fuller

Question 3 (10 points). Following the previous question, display the order ID, customer name, shipper name, and employee's first and last name of orders which are placed by the U.S. customers (the value of the country column is "USA") and the corresponding shipper is "Speedy Express."

Input:

```
SELECT o.OrderID, c.CustomerName, s.ShipperName, e.FirstName AS EmployeeFirstName, e.LastName AS EmployeeLastName
FROM orders o JOIN customers c JOIN shippers s JOIN employees e
ON o.CustomerID = c.CustomerID
AND o.ShipperID = s.ShipperID
AND o.EmployeeID = e.EmployeeID
WHERE c.Country = 'USA' AND s.ShipperName = 'Speedy Express'
ORDER BY o.OrderID ASC;
```

Output:

OrderID	CustomerName	ShipperName	EmployeeFirstName	EmployeeLastName
10260	Old World Delicatessen	Speedy Express	Margaret	Peacock
10269	White Clover Markets	Speedy Express	Steven	Buchanan
10317	Lonesome Pine Restaurant	Speedy Express	Michael	Suyama
10324	Save-a-lot Markets	Speedy Express	Anne	Dodsworth
10349	Split Rail Beer & Ale	Speedy Express	Robert	King
10401	Rattlesnake Canyon Grocery	Speedy Express	Nancy	Davolio
10415	Hungry Coyote Import Store	Speedy Express	Janet	Leverling

Question 4 (10 points). For each product category, display its category name, the average product price, and the number of unique products that belong to this category. Sort the result in descending order by the count, and then in ascending order by the average price.

Input:

```
SELECT c.CategoryName, AVG(p.Price) AS AveragePrice, COUNT(DISTINCT p.ProductID) as ProductNumber
FROM Categories c JOIN products p
ON c.CategoryID = p.CategoryID
GROUP BY c.CategoryName
ORDER BY ProductNumber DESC, AveragePrice ASC;
```

Output:

CategoryName	AveragePrice	ProductNumb...
Confections	25.16	13
Seafood	20.6825	12
Condiments	23.0625	12
Beverages	37.979166666666664	12
Dairy Products	28.73	10
Grains/Cereals	20.25	7
Meat/Poultry	54.00666666666667	6
Produce	32.37	5

Question 5 (10 points). Display the information of customers who have never placed an order.

Input:

```
SELECT c.CustomerID, c.CustomerName, c.ContactName, c.Address, c.City, c.PostalCode, c.Country
FROM customers c LEFT JOIN orders o
ON c.CustomerID = o.CustomerID
WHERE o.OrderID IS NULL;
```

Output:

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
12	Cactus Comidas para llevar	Patricio Simpson	Cerrito 333	Buenos Aires	1010	Argentina
22	FISSA Fabrica Inter. Salchichas S.A.	Diego Roel	C/ Moralzarzal, 86	Madrid	28034	Spain
26	France restauration	Carine Schmitt	54, rue Royale	Nantes	44000	France
32	Great Lakes Food Market	Howard Snyder	2732 Baker Blvd.	Eugene	97403	USA
40	La corne d'abondance	Daniel Tonini	67, avenue de l'Europe	Versailles	78000	France
42	Laughing Bacchus Wine Cellars	Yoshi Tannamuri	1900 Oak St.	Vancouver	V3F 2K1	Canada
43	Lazy K Kountry Store	John Steel	12 Orchestra Terrace	Walla Walla	99362	USA
45	Let's Stop N Shop	Jaime Yorres	87 Polk St. Suite 5	San Francisco	94117	USA
50	Maison Dewey	Catherine Dewey	Rue Joseph-Bens 532	Bruxelles	B-1180	Belgium
53	North/South	Simon Crowther	South House 300 Qu...	London	SW7 1RZ	UK
57	Paris spécialités	Marie Bertrand	265, boulevard Charo...	Paris	75012	France
64	Rancho grande	Sergio Gutiérrez	Av. del Libertador 900	Buenos Aires	1010	Argentina
74	Spécialités du monde	Dominique Perr...	25, rue Lauriston	Paris	75016	France
78	The Cracker Box	Liu Wong	55 Grizzly Peak Rd.	Butte	59801	USA
82	Trail's Head Gourmet Provisioners	Helvetius Nagy	722 DaVinci Blvd.	Kirkland	98034	USA

Question 6 (10 points). Show the customer name and the total spent of customers who have placed at least one order. Sort your result by the amount of the total spent, in descending order.

Input:

```
SELECT c.CustomerName, SUM(od.Quantity * p.Price) AS Total_Spent
FROM customers c JOIN orders o JOIN order_details od JOIN products p
ON c.CustomerID = o.CustomerID
AND o.OrderID = od.OrderID
AND od.ProductID = p.ProductID
GROUP BY c.CustomerName
HAVING SUM(od.Quantity * p.Price) > 0
ORDER BY Total_Spent DESC;
```

Output:

CustomerName	Total_Spent
Ernst Handel	35631.21000000001
Mère Paillarde	23362.600000000002
Save-a-lot Markets	22500.06
Rattlesnake Canyon Grocery	18421.42
QUICK-Stop	18178.8
Queen Cozinha	17880.6
Piccolo und mehr	16040.75
Hungry Owl All-Night Grocers	15391.02
Blondel père et fils	15253.750000000002
Simons bistro	14619
Frankenversand	13384.32
Split Rail Beer & Ale	11114.02
Old World Delicatessen	10744.5
Suprêmes délices	8051.3
Bottom-Dollar Marketse	7963.75
Seven Seas Imports	7438.7
LILA-Supermercado	7358.68
Richter Supermarkt	5875.75

Question 7 (20 points). Which products are delivered by the most commonly used shipping service?

Display the product IDs and names. Only show unique IDs and names.alter

Input:

```
SELECT DISTINCT p.ProductID, p.ProductName
FROM products p JOIN orders o JOIN order_details od
ON p.ProductID = od.ProductID
AND od.OrderID = o.OrderID
WHERE o.shipperID = (SELECT s.ShipperID
FROM orders o JOIN shippers s
ON  s.ShipperID = o.ShipperID
GROUP BY ShipperID
ORDER BY COUNT(o.OrderID) DESC
LIMIT 1);
```

Output:

ProductID	ProductName
41	Jack's New England Clam Chowder
51	Manjimup Dried Apples
65	Louisiana Fiery Hot Pepper Sauce
20	Sir Rodney's Marmalade
33	Geitost
60	Camembert Pierrot
31	Gorgonzola Telino
39	Chartreuse verte
49	Maxilaku
24	Guaraná Fantástica
55	Pâté chinois
74	Longlife Tofu
53	Perth Pasties
77	Original Frankfurter grüne Soße
21	Sir Rodney's Scones
35	Steeleye Stout
72	Mozzarella di Giovanni
44	Gula Malacca

Question 8 (20 points). Write a query to identify orders whose order value is greater than the average of the overall order values. Display order IDs and their order values. Sort the results by order value in descending order.

Input:

```
SELECT o.OrderID, (SELECT SUM(od.Quantity * p.Price)
                   FROM order_details od
                   JOIN products p ON od.ProductID = p.ProductID
                   WHERE od.OrderID = o.OrderID) AS OrderValue # calculates the total value for each order
FROM
    orders o
WHERE (SELECT SUM(od.Quantity * p.Price) #calculates the average order value for all orders
       FROM order_details od
       JOIN products p
       ON od.ProductID = p.ProductID
       WHERE od.OrderID = o.OrderID) > (SELECT AVG(OrderTotal) FROM (SELECT SUM(od.Quantity * p.Price) AS OrderTotal
                                                                     FROM order_details od
                                                                     JOIN products p ON od.ProductID = p.ProductID
                                                                     GROUP BY od.OrderID) AS SubQuery)
                                       #this subquery filters out orders with a total value greater than the average of the overall order values
ORDER BY OrderValue DESC;
```

Output:

OrderID	OrderValue
10372	15353.6
10424	14366.5
10417	14104
10353	13427
10360	9244.250000000002
10324	7698.45
10440	7246.01
10430	7245
10351	7103.599999999999
10329	6025.12
10305	5197.25
10267	5040
10401	4837.02
10252	4662.5
10359	4572.2
10339	4330.4
10393	4135.6
10298	3909.5
10400	3829.59