Galen Healthcare Solutions SQL Assessment lvl 1

1. Test Taker Name

Jordi Ictech

2. In your own words, please explain the difference between a LEFT and INNER JOIN and when you would use one over the other.

the first table is the managers with employee IDs and the second table is employee IDs and department, you could LEFT JOIN the department onto the managers and get the department they manage.

An INNER JOIN returns the values that match on both tables. This can be useful to verify

3. From the Employees table we'd like to see a query that returns the FirstName, LastName, and HireDate for employees that both have the title of Sales Representative, and also are in the United States. (NOTE: Please put SQL query in answer field)

```
SELECT firstname, lastname, hiredate FROM Employees
WHERE country = 'USA' AND title = 'Sales Representative';
```

4. In the products table, we'd like to see the ProductID and ProductName for those products where the ProductName includes the string "queso". (NOTE: Please put SQL query in answer field)

```
SELECT * FROM Products
WHERE productname LIKE '%queso%';
```

5. In the Suppliers table, show the SupplierID, ContactName, and ContactTitle for those Suppliers whose ContactTitle is not Marketing Manager, and also live in Germany or Canada. (NOTE: Please put SQL query in answer field)

```
SELECT supplierid, contactname, contacttitle FROM Suppliers
WHERE country = "Germany" or country = 'Canada' AND NOT contacttitle = 'Marketing
Manager';
```

6. Looking at the Orders table, there’s a field called ShipCountry, we want to show all the orders (OrderID, CustomerID, ShipCountry) from any Latin American country. But we don’t have a list of Latin American countries in a table in the Northwind database. So, we’re going to just use this list of Latin American countries that happen to be in the Orders table: Brazil Mexico Argentina Venezuela (NOTE: Please put SQL query in answer field)

```
SELECT OrderID, CustomerID, ShipCountry FROM Orders
WHERE shipcountry IN ('Brazil', 'Mexico', 'Argentina', 'Venezuela');
```

7. For all the employees in the Employees table, show the FirstName, LastName, Title, and BirthDate. Order the results by BirthDate, so we have the oldest employees first. We see the time of the BirthDate field, which we don’t want. Show only the date portion of the BirthDate field. Put the DOB in mm/dd/yyyy format. (NOTE: Please put SQL query in answer field)

```
SELECT FirstName, LastName, Title, STRFTIME('%m/%d/%Y', birthdate) AS BirthDay
FROM Employees
ORDER BY birthdate asc;
```

8. Show the FirstName and LastName columns from the Employees table, and return a new column called FullName, showing FirstName and LastName joined together in one column, with a space in-between. (NOTE: Please put SQL query in answer field)

```
SELECT FirstName, LastName, (firstname || ' '||
lastname) as FullName from Employees;
```

9. We’d like to show a list of the Orders that were made, including the Shipper that was used. Show the OrderID, OrderDate (date only), and CompanyName of the Shipper, and sort by OrderID. In order to not show all the orders (there’s more than 800), show only those rows with an OrderID of less than 10300. (NOTE: Please put SQL query in answer field)

```
SELECT t1.OrderID, STRFTIME('%d/%m/%Y', t1.orderdate) AS OrderDate,
t2.companyname FROM Orders t1
JOIN Shippers t2 on shipvia = shipperid
WHERE orderid < 10300 ORDER BY orderid ASC;
```

10. For this problem, we’d like to see the total number of products in each category. Sort the results by the total number of products, in descending order. (NOTE: Please put SQL query in answer field)

```
SELECT COUNT(t1.ProductID) AS TotalNumber, t2.CategoryName
FROM Products AS t1
JOIN Categories AS t2 ON t1.CategoryID = t2.CategoryID
GROUP BY t2.CategoryName
ORDER BY COUNT(t1.ProductID) DESC;
```

11. In the Customers table, show the total number of customers per Country and City. (NOTE: Please put SQL query in answer field)

```
SELECT Country, City, COUNT(CustomerID) AS TotalCustomers FROM Customers
GROUP BY City ORDER BY country ASC, COUNT(CustomerID) DESC;
```

12. We're doing inventory, and need to show the following fields: EmployeeID, LastName, OrderID, ProductName, and Quantity. Sort by OrderID and Product ID from the following tables (Employees, Orders, OrderDetails, Products) (NOTE: Please put SQL query in answer field)

```
SELECT t1.EmployeeID, t1.LastName, t2.OrderID, t4.ProductName, t3.quantity FROM
Employees t1
JOIN Orders t2 ON t1.employeeid = t2.employeeid
JOIN OrderDetails t3 ON t2.orderid = t3.orderid
JOIN Products t4 ON t3.productid = t4.productid
Order BY t2.OrderID ASC, t3.ProductID ASC;
```

13. There are some customers who have never actually placed an order. Create a query that shows the Customers in the database that do not have any Orders by returning their CustomerID. (NOTE: Please put SQL query in answer field)

```
SELECT CustomerID FROM Customers
WHERE CustomerID NOT IN (SELECT CustomerID FROM Orders);
```

14. The Northwind mobile app developers are testing an app that customers will use to show orders. In order to make sure that even the largest orders will show up correctly on the app, they'd like some samples of orders that have lots of individual line items. Line items are not 'Quantity', but rather the distinct Products in an Order. Show the 10 orders with the most line items, in order of total line items. (NOTE: Please put SQL query in answer field)

```
--Could also use LIMIT 10, but considering many orders tie, the rank() function was used.
WITH
CTE1 AS (Select orderid, COUNT(DISTINCT(productid)) AS Lines,
RANK() OVER (ORDER BY COUNT(DISTINCT(productid)) DESC) Ranked
FROM OrderDetails GROUP BY orderid)
SELECT * FROM CTE1 WHERE Ranked <= 10;
```

15. Janet Leverling, one of the salespeople, has come to you with a request. She thinks that she accidentally double-entered a line item on an order, with a different ProductID, but the same quantity. She remembers that the quantity was 60 or more. Show all the OrderIDs with order details that match this, in order of OrderID. (NOTE: Please put SQL query in answer field)

```
SELECT DISTINCT productid, OrderID, quantity
FROM OrderDetails
WHERE quantity > 60
GROUP BY OrderID, quantity
HAVING COUNT(*) > 1;
```

16. Your manager needs a list of all the products included in each order. They will need the company name, order date (mm/dd/yyyy), and a comma-delimited list of the products for that order. You will need to return a single row for each order. (NOTE: Please put SQL query in answer field)

```
SELECT t2.orderid, t1.companyname, STRFTIME('%m/%d/%Y', t2.orderdate) AS
OrderDate, GROUP_CONCAT(t4.ProductName) Products
FROM customers t1
JOIN Orders t2 ON t1.customerid = t2.customerid
JOIN OrderDetails t3 on t2.orderid = t3.orderid
JOIN Products t4 on t3.productid = t4.productid
GROUP BY t2.orderid;
```

17. Some customers have been complaining about their orders arriving late, which could be a problem with a specific employees. However, if the employee has more orders, they’re more likely to have late orders. Write a query that returns the EmployeeID, the employee name (Lastname, Firstname), total number of orders submitted by the employee, and the total number of orders shipped late. An order is considered late if the ship date is 5 days after the required date. (NOTE: Please put SQL query in answer field)

```
SELECT t1.EmployeeID, t2.LastName, t2.firstname, count(OrderID) AS TotalOrders,
count(case when  ShippedDate >= DATE(RequiredDate, '+5 days') then 1 else null end) AS
LateOrders
FROM Orders t1
JOIN Employees t2 on t1.EmployeeID = t2.EmployeeID
GROUP BY t1.EmployeeID,  t2.LastName
ORDER BY TotalOrders DESC;
```

18. We want to compile a concise list of some important information for customers in the USA. Write an insert script that will insert the necessary data from the customers table into the USACustomers table. There should be no blank or null values in the new table. Hint: Take a close look at the USACustomers table, some data may need to be transformed in your Select statement when inserting. (NOTE: Please put SQL query in answer field)

```
INSERT INTO USACustomers (CustomerID, customername, customercompany, address,
stateabbreviation, phone)
SELECT customerid, contactname, companyname, address,
(case
 when region = 'Oregon' then 'OR'
 when region = 'Washington' then 'WA'
 when region = 'California' then 'CA'
 when region = 'Alaska' then 'AK'
 when region = 'New Mexico' then 'NM'
 when region = 'Montana' then 'MT'
 when region = 'Wyoming' then 'WY'
 when region = 'Idaho' then 'ID'
 end), CAST(phone AS int)
FROM Customers WHERE country = 'USA';
```

19. The Reviews table has become cluttered with unwanted duplicate data. Your boss has asked you to migrate the data over to the new CustomerReviews table. Insert the necessary data to the CustomerReviews table. There should be no blank or null values in the new table. (NOTE: Please put SQL query in answer field)

```
INSERT INTO CustomerReviews (reviewid, companyname, rating, dateofreview)
SELECT DISTINCT t1.reviewid, t2.companyname, t1.rating, t1.dateofreview FROM Reviews
t1
JOIN customers t2 ON t1.customerid = t2. customerid;
```

20. You have received a request from your boss to find total freight shipped per month. He has provided you with a table called FreightPerMonth that you need to insert data into. Insert the Month and value of total freight from the Orders table into your new table. There should only be 12 rows in your new table. Hint: Value does not need to be exact. (NOTE: Please put SQL query in answer field)

```
INSERT INTO FreightPerMonth (Month, TotalFreight)
SELECT STRFTIME('%m', orderdate), CAST(SUM(freight) AS INT) FROM orders
GROUP BY STRFTIME('%m', orderdate);
```

Submit

0%