Northwind Solutions exercises

Series 1: Select

1. Return all fields from customers

```
SELECT * FROM Customers;
```

2. Return all fields from product

```
SELECT * FROM Products;
```

3. Return first company and country from customers

```
SELECT CompanyName, Country FROM Customers;
```

4. Return all shippers sorted ascending by the shipper name

```
SELECT * FROM Shippers ORDER BY CompanyName asc
```

5. Return name, country and city from customers. Order on country than city.

```
SELECT CompanyName, Country, City FROM Customers ORDER BY Country,
City
```

6. Return all fields from the first 3 records of employees

```
SELECT * FROM employees LIMIT 3;
```

7. Return all the fields from the table suppliers. All countries can only be shown 1 time

```
SELECT DISTINCT Country FROM Suppliers ORDER BY Country;
```

8. Return all countries and contact names from table suppliers. Every combination should be unique.

```
SELECT DISTINCT ContactName, Country FROM Suppliers ORDER BY
Country;
```

9. Return all the orders where there is a date assigned. A specific date can only be shown 1 time.

```
SELECT DISTINCT OrderDate FROM Orders;
```

10. Sort the table Customers based on the country (ascending) and on city (descending).
Then show the first record its name.

```
SELECT CompanyName FROM Customers ORDER BY Country asc, City desc
LIMIT 1;
```

Series 2: Select where

11. Show me the information from the Mexican employees.

```
SELECT * FROM Customers WHERE Country LIKE "Mexico";
```

12. Show all information from the orders that are place after 10th July 1996.

```
SELECT * FROM Orders WHERE OrderDate > "1996-07-10" ORDER BY
OrderDate;
```

13. Same as exercise 12 but ordered on column ID

```
SELECT * FROM Orders WHERE OrderDate > "1996-07-10" ORDER BY
OrderId;
```

14. Which customers are based in Paris? Create an alphabetic list with the company names from these customers.

```
SELECT * FROM Customers WHERE City LIKE "Paris";
```

15. Which employees are hired after 1th of January 1994?

Create a list from the employees with the family name, function and date when they are hired.

```
SELECT * FROM Employees WHERE HireDate > "1994-01-01";
```

16. A customer called about an order that is placed at 2 October 1996. The financial department wants to know every information about this order. Because the information is based in 2 tables(Orders, Orders details) we will need 2 queries. Later we will do this one query.

```
SELECT * FROM Orders WHERE OrderDate = "1996-10-02"; SELECT * FROM
`Order Details` WHERE OrderID = 10319
```

17. For which employees (family name and first name) we don't have the region?

```
SELECT LastName, FirstName FROM Employees WHERE Region IS NULL;
```

18. Give the address information from all the customers that live in postcode which starts with a 'W'

```
SELECT * FROM Customers WHERE PostalCode LIKE "W%";
```

19. Give the address from all the customers where the contactname second character is A or X. Sort first on country, then postalcode.

```
SELECT * FROM Customers WHERE ContactName LIKE "_A%" ORDER BY
Country, PostalCode;
```

20. Give the fax and name of the (leveranciers) where we don't know the fax. Sort the name ascending.

```
SELECT * FROM Suppliers WHERE Fax IS NULL ORDER BY CompanyName asc;
```

21. Same as question 20 where we do know the fax

```
SELECT * FROM Suppliers WHERE Fax IS NOT NULL ORDER BY CompanyName
asc;
```

Series 2: Select logical

22.Show the data from 'sales' customers (function start with sales) that come from Germany and France

```
SELECT * FROM Customers WHERE Country IN ('France', 'Germany') AND
ContactTitle LIKE "%Sales%";
```

23.Show the data from all the other customers then exercise 22

```
SELECT * FROM Customers WHERE Country NOT IN ('France', 'Germany')
AND ContactTitle NOT LIKE "%Sales%";
```

24.Show all the female 'Sales representatives' and the male 'sales manager' (employees)

```
SELECT * FROM Employees WHERE (TitleOfCourtesy = "Ms." AND Title =
"Sales Representative") OR (TitleOfCourtesy IN ("Dr.", "Mr.") AND
Title = "Sales Manager");
```

25.Show all the customers with name and contactname where the contactname starts or ends with the letter A

```
SELECT CompanyName, ContactName FROM Customers WHERE CompanyName
like "A%" OR CompanyName like "%A";
```

26.Show all the customers where fax or phone is empty

```
SELECT * FROM Customers WHERE Fax IS NULL OR Phone IS NULL;
```

27.Show following products (CHAI, KONBU, TOFU) sorted on name

```
SELECT * FROM PRODUCTS WHERE ProductName IN ('CHAI', 'KONBU',
'TOFU');
```

28.Show all the order where the shipping date is between 8th September 1996 and 9h October 1996

```
SELECT * FROM Orders WHERE ShippedDate between "1996-09-08" and
"1996-10-09";
```

29.Give the products with a unit price between 10 and 12 and where there are more than 10 units in stock

```
SELECT * FROM Products WHERE UnitsInStock > 10 AND (QuantityPerUnit
LIKE "10%" OR QuantityPerUnit LIKE "12%");
```

Series 3: Select functions

30. How many customer are there? How much customers live in Germany?

```
SELECT count(*) FROM Customers; SELECT COUNT(*) FROM Customers
WHERE Country = "Germany";
```

31. How many items are sold in total?

```
SELECT SUM(Quantity) FROM `Order Details`;
```

32. What is the average unit price from the products that are not in an order?

```
SELECT AVG(UnitPrice) FROM Products WHERE UnitsOnOrder > 0
```

33. Give the maximum assigned discount from an product that is ordered by a customer

```
SELECT MAX(discount) FROM `Order Details`;
```

34. Give the total value of the customer order products, included with discount. Round the number to an integer.

```

```

35. Give a list of the orderid, unitprice, quantity and total from order details.Total is a column that is calculated by multiplying unitprice with quantity.Add a correct column name as well.

```
SELECT OrderID, ProductID, UnitPrice, Quantity, UnitPrice *
Quantity as "TotalPrice" FROM `Order Details`
```

36. Give a list with the names from the employees with their starting date together with the age they were at that time.

```
SELECT LastName, FirstName, YEAR(HireDate) - YEAR(BirthDate) as
'Age when hired' FROM Employees;
```

37. Give a list with product names, unit price, new price from the products. New price is calculated by increasing the unit price with 2%.

```
SELECT ProductName, UnitPrice, UnitPrice * 1.02 AS 'NewPrice' FROM
Products;
```

38. Give the same list but rounded with 0 decimals

```
SELECT ProductName, UnitPrice, ROUND(UnitPrice * 1.02, 0) AS
'NewPrice' FROM Products;
```

39. From how many employees we know the region?

```
SELECT count(*) AS "Amount of employees" FROM Employees WHERE
Region IS NOT NULL;
```

40. Show the product names where the new price is greater then 50

```
SELECT ProductName, UnitPrice, ROUND(UnitPrice * 1.02, 0) AS
'NewPrice' FROM Products WHERE ROUND(UnitPrice * 1.02, 0) > 50;
```

41. Show a new identification code from the employees. Every identification code consist out (2 first letters of the family name and 1 letter from the first name) Everything should be in capital.

```
SELECT UPPER(CONCAT(LEFT(Lastname,2),LEFT(Firstname, 1))) as
'indentifcation code' FROM Employees;
```

Series 4: Select group by functions

42. Show the most recent hiring date per function.

```
SELECT Title, MAX(HireDate) AS "Most recent hiringdate" FROM
`Employees` GROUP BY Title
```

43. What is the average unit price and the average quantity from the sold products?

```
SELECT ProductId, AVG(UnitPrice) AS "Average unit price",
AVG(Quantity) AS "Average qty" FROM `Order Details` GROUP BY
ProductId
```

44. How many suppliers do you have for Japan and USA?
Sort by the country with the most suppliers

```
SELECT Country, COUNT(*) FROM Suppliers WHERE Country IN ("USA",
"JAPAN") GROUP BY Country ORDER BY COUNT(*) DESC;
```

45. Count the employees per city but only show the city that have more than 1 employee

```
SELECT City, COUNT(*) AS "Amount of employees" FROM Employees GROUP
BY City HAVING COUNT(*) > 1
```

46. What is the maximum and minimum unitPrice per categoryID?

```
SELECT CategoryID, MIN(UnitPrice) as 'Minimum price',
Max(UnitPrice) as 'Maximum price' FROM Products GROUP BY
CategoryID;
```

47. What is the average stock unit per category for product that are more expensive than 10. Show only the information when the average of stock is greater than 40.

```
SELECT ProductId, ProductName, AVG(UnitsInStock) AS "Average unit
in stock" FROM Products WHERE UnitPrice > 10 GROUP BY ProductId,
ProductName HAVING AVG(UnitsInStock) > 40 ORDER BY 3 DESC;
```

48. Give us an overview from the amount a customer has placed(ID is enough). When they have no

more than 10 orders, the customer should not be included in the list.

```
SELECT CustomerID, COUNT(*) AS "Amount of orders" FROM Orders GROUP
BY CustomerID HAVING COUNT(*) <= 10 ORDER BY 2;
```

49. When we take a second look, maybe it is better to show a top 10 list with the customers who has the most orders Change the previous query

```
SELECT CustomerID, COUNT(*) AS "Amount of orders" FROM Orders GROUP
BY CustomerID ORDER BY 2 DESC LIMIT 10;
```

50. Every we year the department "Purchases" wants to have a logical overview of the top selling days in 1996 with a total of more than 1 order a day

```
SELECT OrderDate, COUNT(*) AS "Amount of orders" FROM Orders WHERE
YEAR(OrderDate) = 1996 GROUP BY OrderDate HAVING COUNT(*) > 1;
```

51. Previous query is not very satisfying. More interesting would be if we would know what the best month was. Show the months completely.

```
SELECT DATE_FORMAT(OrderDate, "%M"), COUNT(*) AS "Amount of orders"
FROM Orders WHERE YEAR(OrderDate) = 1996 GROUP BY 1 HAVING COUNT(*)
> 1;
```

Series 5: Data manipulation

52. Copy the entire table Products in CopyProducts

```
CREATE TABLE CopyProducts AS SELECT * FROM Products;
```

53. Create a copy from Orders with name CopyOrders. We will not work anymore with belgium. Remove all the records from CopyOrders with ship country to Belgium

```
CREATE TABLE OrdersCopy AS SELECT * FROM Orders WHERE ShipCountry
NOT LIKE 'Belgium';
```

```
CREATE TABLE OrdersCopy AS SELECT * FROM Orders; DELETE FROM
OrdersCopy WHERE ShipCountry LIKE 'Belgium';
```

54. Create a new table ProductsCopy with the name and the price of all the products that have category seafood (8) or category Beverages (1)

```
CREATE TABLE ProductsCopy SELECT ProductName, UnitPrice FROM
Products WHERE CategoryID = 8 OR CategoryID = 1;
```

55. Add all the products with categoryId2 in table ProductsCopy

```
INSERT INTO ProductsCopy (ProductName, UnitPrice) SELECT
ProductName, UnitPrice FROM Products WHERE CategoryID = 2;
```

56. Update the orderdate from all the orders that has a shipped date in Augustus 1997(CopyOrders)

```
UPDATE OrdersCopy SET OrderDate = date_add(OrderDate, INTERVAL 10
day) WHERE MONTH(OrderDate) = 8 AND YEAR(OrderDate) = 1997
```

57. Show the shippedDate from Orders and from CopyOrders. Show only the orderdate from august 1997. Order by OrderDate

```
SELECT * FROM OrdersCopy WHERE MONTH(OrderDate) = 8 AND
YEAR(OrderDate) = 1997 ORDER BY OrderDate; SELECT * FROM Orders
WHERE MONTH(OrderDate) = 8 AND YEAR(OrderDate) = 1997 ORDER BY
OrderDate;
```

58. Copy the table employees to EmployeesCopy. Add new colomn NewSalary.

```
CREATE TABLE EmployeesCopy AS SELECT * FROM Employees; ALTER TABLE
EmployeesCopy ADD NewSalary INT;
```

59. Fill in the newSalary for all the employees with 60000

```
UPDATE EmployeesCopy SET NewSalary = 60000;
```

60. All Sales Representatives should have 20% extra wage.

```
UPDATE EmployeesCopy SET NewSalary = NewSalary * 1.2 WHERE Title
LIKE 'Sales Representative';
```

61. Anne Dodsworth has an internship contract, so lower her wage with 10%

```
UPDATE EmployeesCopy SET NewSalary = NewSalary * 0.9 WHERE
Firstname LIKE 'Anne' AND LastName LIKE 'Dodsworth';
```

62. Add following product to ProductsCopy: Meatballs (unitprice 12)

```
INSERT INTO ProductsCopy(ProductName, UnitPrice) VALUES
('Meatballs', 12)
```

63. Add a new order to OrdersCopy. Customer ATRA helped by employee 3 with an orderDate of 30/10/2009 and requiredDate of 30/11/2009

```
INSERT INTO OrdersCopy(CustomerID, EmployeeID,
OrderDate,RequiredDate) VALUES ('ATRA', 3, '2009-10-30','2009-11-
30')
```

Series 6: JOINS

64. Give for every order the name of the employee that is responsible.

```
SELECT O.OrderId, O.CustomerId, O.OrderDate, E.FirstName,
E.LastName FROM Orders AS O INNER JOIN Employees AS E ON
O.EmployeeID = E.EmployeeID;
```

65. Show the amount of orders per employee, give the name and lastname of the employee

```
SELECT E.Firstname, E.LastName, COUNT(*) AS AmountOfOrders FROM
Orders AS O INNER JOIN Employees AS E ON O.EmployeeID =
E.EmployeeID GROUP BY E.Firstname, E.LastName;
```

66. Give for every product the name, the name of the category and the name of the supplier.

```
SELECT * FROM Products AS P LEFT JOIN Categories AS C ON
P.CategoryID = C.CategoryID LEFT JOIN Suppliers AS S ON
P.SupplierID = S.SupplierID;
```

67. Show per order the companyname, firstname and lastname on employees.

```
SELECT O.OrderID, C.CompanyName, E.FirstName, E.LastName FROM
Orders AS O INNER JOIN Customers AS C ON O.CustomerID =
C.CustomerID INNER JOIN Employees AS E ON O.EmployeeId =
E.EmployeeId
```

68. Show the names of the customers and their orders. All customers need be to showed even when they don’t got orders.

```
SELECT C.CompanyName, O.OrderID FROM Customers AS C LEFT JOIN
Orders AS O ON C.CustomerID = O.CustomerID
```

69. Show from every customer the total amount of orders. Sort the amount from small to big

```
SELECT C.CompanyName, COUNT(O.OrderID) AS 'Amount Of orders' FROM
Customers AS C LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID
GROUP BY C.CompanyName ORDER BY 2 ASC
```

70. Show the lastname and firstname of the employees and the territoria that they are active at. Used tabled : Employees, EmployeeTerritories, Territories.

```
SELECT E.LastName, E.FirstName, T.TerritoryDescription FROM
Employees AS E INNER JOIN EmployeeTerritories AS ET ON E.EmployeeID
= ET.EmployeeID INNER JOIN Territories AS T ON ET.TerritoryID =
T.TerritoryID;
```

71. Show the amount of territioriums per employee

```
SELECT E.LastName, E.FirstName, COUNT(T.TerritoryDescription) AS
AmountEmployees FROM Employees AS E INNER JOIN EmployeeTerritories
AS ET ON E.EmployeeID = ET.EmployeeID INNER JOIN Territories AS T
ON ET.TerritoryID = T.TerritoryID GROUP BY E.LastName, E.FirstName;
```

72. Show the names of all the territoria with the firstname and lastname of the responsible Employee. Show also the territoria where there is no responsible.

```
SELECT TerritoryDescription, E.FirstName, E.LastName FROM
Territories AS T LEFT JOIN EmployeeTerritories AS ET ON
T.TerritoryID = ET.TerritoryID LEFT JOIN Employees AS E ON
ET.EmployeeID = E.EmployeeID;
```

73. Show per employees the amount of unique customers

```
SELECT E.EmployeeID, COUNT(DISTINCT O.CustomerId) AS "Amount Of
Customers" FROM Orders AS O INNER JOIN Employees AS E ON
O.EmployeeID = E.EmployeeID INNER JOIN Customers AS C ON
O.CustomerID = C.CustomerID GROUP BY E.EmployeeID;
```

74. Show the name of the all the customer that bought chocolat

```
SELECT DISTINCT C.CompanyName FROM `Order Details` AS OD INNER JOIN
Orders AS O ON OD.OrderID = O.OrderID INNER JOIN Customers AS C ON
O.CustomerID = C.CustomerID INNER JOIN Products AS P ON
OD.ProductID = P.ProductID WHERE P.ProductName LIKE '%CHOCO%' OR
P.ProductName LIKE '%CHOKO%' ORDER BY 1;
```

75. Show customers with less then 5 orders, also the customer that not have any order. Sort by amount.

```
SELECT C.CustomerId, COUNT(O.OrderID) AS 'Amount of orders' FROM
Orders AS O RIGHT JOIN Customers AS C ON O.CustomerID =
C.CustomerID GROUP BY CustomerID HAVING COUNT(*) < 5 ORDER BY 2
```

76. The accountable of the commercial department thinks that lately we don't deliver on time anymore. That is why he/she is asking for a summary per year and per customer of the difference between require date and shipping date. He only wants to see them when the difference is lower then 0 (because then the deliver is to late).

    The list should be completed that we can see the average difference between required date and delivery date. Only the average smaller then 0 should be shown.

```
SELECT YEAR(RequiredDate), C.CompanyName,
AVG(DATEDIFF(RequiredDate, ShippedDate)) FROM Orders AS O INNER
JOIN Customers AS C ON O.CustomerID = C.CustomerID WHERE
DATEDIFF(RequiredDate, ShippedDate) < 0 GROUP BY
YEAR(RequiredDate), C.CompanyName HAVING AVG(DATEDIFF(RequiredDate,
ShippedDate)) < 0 ORDER BY 2, 1;
```

Series 7: Subqueries

77. Which products were delivered by the supplier Pavlova, ltd

```
SELECT productid, productname FROM products WHERE supplierid in (
SELECT supplierid FROM suppliers WHERE companyname = 'Pavlova,
ltd.' );
```

78. How many Tofu's are sold in total?

```
SELECT SUM(Quantity) FROM `Order Details` WHERE ProductID IN (
SELECT ProductId FROM Products WHERE ProductName like '%tofu%' )
```

```
SELECT SUM(OD.Quantity) FROM `Order Details` AS OD INNER JOIN
Products AS P ON OD.ProductID = P.ProductID WHERE P.ProductName
LIKE '%tofu%';
```

79. How many orders were created by Robert King in May 1997?

```
SELECT COUNT(OrderId) FROM Orders AS O WHERE EmployeeID IN ( SELECT
EmployeeID FROM EMPLOYEES WHERE LastName LIKE "%king%" AND
FirstName LIKE "%robert%" ) AND MONTH(OrderDate) = 5 AND
YEAR(OrderDate) = 1997
```

```
SELECT COUNT(OrderId) FROM Orders AS O INNER JOIN Employees AS E ON
O.EmployeeID = E.EmployeeID WHERE E.LastName LIKE "%king%" AND
E.FirstName LIKE "%robert%" AND MONTH(O.OrderDate) = 5 AND
YEAR(O.OrderDate) = 1997
```

80. Which products that belong to the category Confections are ordered once? (use unitsonorder)

```
SELECT productID, productname, unitsonorder FROM Products WHERE
categoryID IN ( SELECT categoryID FROM Categories WHERE
categoryname = 'Confections' ) AND unitsonorder > 0;
```

81. Give the name of all product that have the same unitprice as chang

```
SELECT Productname FROM Products WHERE Unitprice IN ( SELECT
Unitprice FROM Products WHERE Productname = 'Chang' ) And
productname <> 'Chang'
```

82. Give a list of all the products that are more expensive than tofu

```
SELECT Productname FROM Products WHERE Unitprice > ( SELECT
Unitprice FROM Products WHERE productname = 'Tofu' )
```

83. Give a list of all the customers that have more than 15 orders

```
SELECT companyname FROM customers WHERE customerid IN ( SELECT
customerid FROM orders GROUP BY customerid HAVING count(orderid) >
15 )
```

84. Who is hired most recently?

```
SELECT lastname FROM employees WHERE hiredate IN ( SELECT
MAX(hiredate) FROM employees )
```

85. Which products are never sold yet?

```
SELECT Productname FROM Products WHERE NOT EXISTS ( SELECT * FROM
`Order Details` WHERE productid = Products.productid );
```

86. Which city has the most customers? Show a table of the customers from this country

```
// ERROR LIMIT
```

87. Show information about the product with the heighest unitprice.

```
SELECT ProductName FROM Products WHERE UnitPrice = ( SELECT
MAX(UnitPrice) FROM Products );
```

88. Show the products that are sold more then 50 times. Give a list of the productnames

```
SELECT ProductName FROM Products WHERE productid IN ( SELECT
productid FROM `Order Details` GROUP BY productid HAVING COUNT(*)
>50 )
```

89. Which customers doesn't have any orders?

```
SELECT companyName FROM customers WHERE CustomerID NOT IN ( SELECT
DISTINCT CustomerID FROM Orders )
```

Series 8: Union + more subquery

90. Create a telephone book with the phone and mail from the customers and from the suppliers
(companyname, contactname, contacttitle, type_of_contact, phone)

```
SELECT CompanyName, ContactTitle, Phone, 'Customer' FROM Customers
UNION SELECT CompanyName, ContactTitle, Phone, 'Supplier' FROM
Suppliers;
```

91. Show all the countries and regions that are both in customers and suppliers. (Region can not be empty

```
SELECT City, Region FROM Suppliers WHERE City IS NOT NULL AND
Region IS NOT NULL AND CITY IN (SELECT CITY FROM CUSTOMERS) AND
COUNTRY IN (SELECT COUNTRY FROM CUSTOMERS);
```

92. Show every order with total order price from year 1997
    The last line need to be the total amount of money

```
SELECT OD.OrderId, SUM(UnitPrice * Quantity) FROM `Order Details`
AS OD INNER JOIN Orders AS O ON OD.OrderId = O.OrderID WHERE
YEAR(O.OrderDate) = 1997 GROUP BY OD.OrderID UNION SELECT 'TOTAL',
SUM(UnitPrice * Quantity) FROM `Order Details` AS OD INNER JOIN
Orders AS O ON OD.OrderId = O.OrderID WHERE YEAR(O.OrderDate) =
1997
```

93. Show regions from suppliers that are not in customers.

```
SELECT Region FROM Suppliers WHERE Region NOT IN ( SELECT Region
FROM Customers WHERE Region IS NOT NULL) AND Region IS NOT NULL;
```

94. Show the name of the employees who ordered the product Chang and that have customers in Charleroi

```
SELECT DISTINCT E.FirstName, E.LastName FROM `Order Details` AS OD
INNER JOIN Products AS P ON OD.ProductID = P.ProductID INNER JOIN
Orders AS O ON OD.OrderId = O.OrderId INNER JOIN Employees AS E ON
O.EmployeeID = E.EmployeeID WHERE P.ProductName LIKE '%chang%' AND
E.EmployeeID IN ( SELECT E.EmployeeID FROM Orders AS O INNER JOIN
Employees AS E ON O.EmployeeID = E.EmployeeID INNER JOIN Customers
AS C ON O.CustomerID = C.CustomerID WHERE C.City LIKE '%charleroi%'
)
```

95. Show in 1 table: the amount of customers, the amount of products and the amount of employees

```
SELECT 'Amount Of Employees' AS Description, COUNT(*) AS Amount
FROM Employees UNION SELECT 'Amount Of Products', COUNT(*) FROM
Products UNION SELECT 'Amount Of Customers', COUNT(*) FROM
Customers;
```

Series 9: Stored procedures

96. Create an SP with the name getOrdersByProduct where you can give the productname and it will display all the orders that have sells this product

```
CREATE PROCEDURE `getOrdersByProduct` (IN inputProductName
VARCHAR(255)) BEGIN SELECT DISTINCT O.OrderID FROM Orders AS O
INNER JOIN `Order Details` AS OD ON O.OrderID = OD.OrderID INNER
JOIN Products AS P ON OD.ProductID = P.ProductID WHERE
P.ProductName LIKE inputProductName; END
```

97. Create an SP with the name createTableGetOrdersByproduct where you output the result of assignment 96 to a new table

```
CREATE PROCEDURE `createTableGetOrdersByproductname` (IN
inputProductName TEXT) BEGIN DROP TABLE IF EXISTS
tempProductOrders; CREATE TABLE tempProductOrders AS SELECT
DISTINCT O.OrderID FROM Orders AS O INNER JOIN `Order Details` AS
OD ON O.OrderID = OD.OrderID INNER JOIN Products AS P ON
OD.ProductID = P.ProductID WHERE P.ProductName LIKE
inputProductName; END
```

98. Create an SP with the name getOrdersBetweenDates where you can give 2 dates and it will return all the orders between this dates

```
CREATE PROCEDURE `getOrdersBetweenDates` (IN startDate DATE, IN
endDate DATE) BEGIN SELECT * FROM Orders WHERE OrderDate BETWEEN
startDate AND endDate; END
```