Querying from One Table

1. List Order Number, Product Code, Quantity Order, and the Price for each Product. Sort the results by Order Number in ascending order and Quantity Ordered in descending order.

SELECT orderNumber,productCode,quantityOrdered, priceEach

FROM orderdetails

ORDER BY orderNumber, quantityOrdered DESC;



1. Report those payments greater than $100,000 or with a payment date after 08/20/2004.

SELECT \*

FROM payments

WHERE amount>100000 OR paymentDate > '2004-08-20';

Table

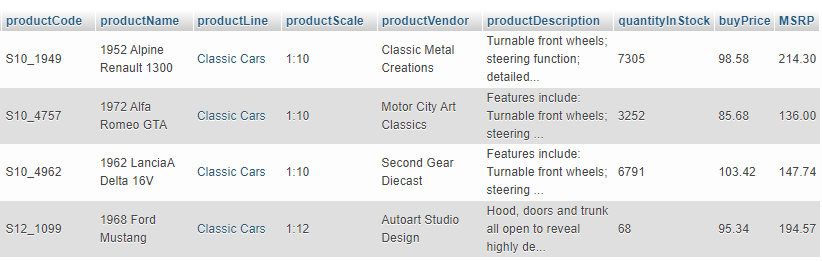
Description automatically generated

1. Display the products that belong to the Classic Car product line and either with a buy price higher than 80 or quantity in stock lower than 7000.

SELECT \*

FROM products

WHERE productLine = 'Classic Cars' AND (buyPrice>80 OR quantityInStock<7000);



1. List the 15 products with the lowest quantity in stock.

SELECT \* FROM products

ORDER BY quantityInStock

LIMIT 15;

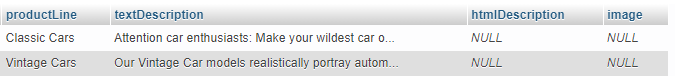
A screenshot of a computer

Description automatically generated with medium confidence

1. List the product lines that contain 'Cars'.

SELECT \* FROM productlines

WHERE productLine LIKE '%Cars%';



1. Label all the employees who are sales managers as 1 and others as 0.

SELECT \*, (jobTitle LIKE '%Manager%') as manager

FROM employees;

Graphical user interface, text, application

Description automatically generated

1. Calculate the total price for each product in each order.

SELECT \*,quantityOrdered\*priceEach AS totalPrice

FROM orderdetails;

Table

Description automatically generated

1. Report the number of products ordered in each order (Multiple quantities of the same product should be counted as one product).

SELECT orderNumber, COUNT(DISTINCT(productCode)), MAX(orderLineNumber)

FROM orderdetails

GROUP BY orderNumber;

Table

Description automatically generated

I aggregated based on orderNumber then, reported both the number of distinct products and the number of lines in the order (which is a native attribute). Order details does not appear to have multiple rows of the same product number. This was verify using the following, which returns zero rows:

SELECT orderNumber, COUNT(DISTINCT(productCode)) count1, MAX(orderLineNumber) count2

FROM orderdetails

GROUP BY orderNumber

HAVING count1 <> count2;

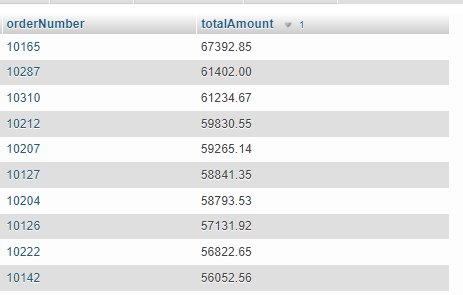
1. Report the total amount spent in each order (I want to total for each order not the total for each product in the order). Sort the results in descending order.

SELECT orderNumber,sum(quantityOrdered\*priceEach) as totalAmount

FROM orderdetails

GROUP BY orderNumber

ORDER BY totalAmount DESC;



1. Report the highest and lowest amount each customer has paid.

SELECT customerNumber,MIN(amount),MAX(amount)

FROM payments

GROUP BY customerNumber;

Table

Description automatically generated

1. Find the average, variance, and standard deviation for the price of the product for each order.

SELECT orderNumber,

AVG(priceEach),VAR\_POP(priceEach),STDDEV\_POP(priceEach)

FROM orderdetails

GROUP BY orderNumber;

Table

Description automatically generated

1. Find the total quantity ordered and average price for each order and only display those with an average price >$100.

SELECT orderNumber,SUM(quantityOrdered),AVG(priceEach) AS AVG\_PRICE

FROM orderdetails

GROUP BY orderNumber

HAVING AVG\_PRICE>100;

Table

Description automatically generated

1. Display employees' full name as first name followed by a space, and by last name in one column. Display their names as upper case.

SELECT UPPER(CONCAT(firstName,’ ‘, lastName)) AS fullName

FROM employees;

Text

Description automatically generated with low confidence

1. Report how many products are in the motorcycles product line.

SELECT COUNT(\*)

FROM products

WHERE productLine LIKE '%Motorcycles%';



The thirteen are the following:

SELECT productName

FROM products

WHERE productLine LIKE '%Motorcycles%';

Table

Description automatically generated

1. The productCode in the products table has two parts. The part before the underscore shows the supplier code. Display the supplier code for each product.

SELECT SUBSTRING(productCode FROM 1 FOR 3)

FROM products;

Table

Description automatically generated

1. Display the initial of each employee in one column.

SELECT CONCAT(SUBSTRING(firstName FROM 1 FOR 1),SUBSTRING(lastname FROM 1 FOR 1))

AS initials

FROM employees;

Graphical user interface, application

Description automatically generated

1. Display the office phone number without the + sign at the beginning.

SELECT TRIM(LEADING '+' FROM phone)

FROM offices;

Table

Description automatically generated

1. Display "Rep" as "Director" for employees' job title.

SELECT jobTitle,REPLACE(jobTitle,'Rep','Director')

FROM employees;

Table

Description automatically generated

1. Display the month, day, and year of the payment date in separate columns.

SELECT MONTH(paymentDate) AS month, DAY(paymentDate) AS day, YEAR(paymentDate) AS year

FROM payments;

Table

Description automatically generated

1. Calculate the average, maximum, and minimum number of days it takes for an order to be shipped.

SELECT AVG(shippedDate-orderDate) AS average,

MAX(shippedDate-orderDate) AS maximum,

MIN(shippedDate-orderDate) AS minimum

FROM orders;



1. Mark those orders which shipped more than 3 days after the order date as 0 and those which shipped within 3 days as 1.

SELECT ((shippedDate-orderDate)<=3) AS marked

FROM orders;

Graphical user interface, application

Description automatically generated

1. Display the price difference between the MSRP price and buy price for each product. Round the results to integer.

SELECT ROUND(MSRP-buyPrice,0) as margin

FROM products;

A picture containing table

Description automatically generated

1. Change and display the credit limit of each customer as integer.

SELECT CAST(creditLimit AS UNSIGNED) AS creditLimitINT

FROM customers;

Table

Description automatically generated with medium confidence

1. Mark those customers who have placed more than 3 orders as Frequent and those who have placed 3 of fewer orders as Target.

SELECT customerNumber,COUNT(orderNumber),

CASE

WHEN COUNT(orderNumber)>3 THEN 'Frequent'

ELSE 'Target'

END AS marked

FROM orders

GROUP BY customerNumber;

Table

Description automatically generated

1. For those customers who do not have a sales rep, display Need Rep. Otherwise, display their sales rep's number.

SELECT customerNumber,salesRepEmployeeNumber,

CASE

WHEN salesRepEmployeeNumber IS NULL THEN 'Need Rep'

ELSE salesRepEmployeeNumber

END rep

FROM customers

Table

Description automatically generated

1. Mark those orders which took more than 5 days to shipped as Too Slow, and those which took between 3 and 5 days as OK, and the rest as Fast.

SELECT orderNumber,

CASE

WHEN (shippedDate-orderDate)>5 THEN 'Too Slow'

WHEN 5>=(shippedDate-orderDate) AND (shippedDate-orderDate)>3 THEN 'OK'

ELSE 'Fast'

END ship

FROM orders;

Table

Description automatically generated

Querying from Multiple Tables

1. Display each employee's employee number, last name, first name, email, job title, their office city, office phone, office state, office country. office postal code.

SELECT employees.employeeNumber, employees.lastName,

employees.firstName, employees.email, employees.jobTitle,

offices.city, offices.phone, offices.state,

offices.country, offices.postalCode

FROM employees, offices

WHERE employees.officeCode = offices.officeCode;

A picture containing text, indoor, screenshot, decorated

Description automatically generated

1. List all the customers who have not placed any orders.

SELECT \*

FROM customers

LEFT JOIN orders

ON customers.customerNumber = orders.customerNumber

WHERE orders.customerNumber IS NULL;

A screenshot of a computer

Description automatically generated with medium confidence

1. Count and display the number of different types of products each customer has ordered. The same product, regardless of the quantity ordered counts as one type of product.

SELECT customers.customerName,products.productName, COUNT(DISTINCT(products.productName))

FROM customers,orders,orderdetails,products

WHERE customers.customerNumber = orders.customerNumber

AND orders.orderNumber = orderdetails.orderNumber

AND orderdetails.productCode = products.productCode

GROUP BY customers.customerName,products.productName

WITH ROLLUP;

Table

Description automatically generated

1. Use your answer for query 29 and only display those customers who have order more than 10 types of products.

SELECT customers.customerName, COUNT(DISTINCT(products.productName)) AS typesOdered

FROM customers,orders,orderdetails,products

WHERE customers.customerNumber = orders.customerNumber

AND orders.orderNumber = orderdetails.orderNumber

AND orderdetails.productCode = products.productCode

GROUP BY customers.customerName,products.productName

WITH ROLLUP

HAVING typesOdered > 10;

Graphical user interface, table

Description automatically generated with medium confidence

1. Create a column with all the country names for the employees as well as the customers. Do not display duplicate country names.

SELECT DISTINCT(country) FROM (

SELECT DISTINCT(offices.country) as country

FROM offices

UNION

SELECT DISTINCT(customers.country)

FROM customers)

AS tmp;

Table

Description automatically generated

1. Identify the most productive employee in terms of the highest number of customers served.

SELECT COUNT(customerNumber) AS customers\_servered,employees.employeeNumber,

employees.firstName, employees.lastName

FROM employees,customers

WHERE customers.salesRepEmployeeNumber = employees.employeeNumber

GROUP BY employeeNumber

ORDER BY customers\_servered

LIMIT 1;



1. Are there any products that have not been ordered? If there are, please list them.

SELECT productName,products.productCode

FROM products

LEFT JOIN orderdetails

ON products.productCode=orderdetails.productCode

WHERE orderdetails.productCode is NULL



1. List all employees' employee number, first name, last name, and their manager's employee number, first name, and last name.

SELECT employeeNumber,firstName,lastName,reportsTo,manager\_firstName,manager\_lastName

FROM employees

INNER JOIN (

(SELECT employeeNumber AS manager\_employeeNumber, firstName AS manager\_firstName,lastName AS manager\_lastName FROM employees)

AS managers)

ON employees.reportsTo = managers.manager\_employeeNumber;

Table

Description automatically generated

1. Which company is the most profitable customer in terms of the dollar amount spent. Display the company's name, phone number, and the total dollar amount spent.

SELECT customerName,phone,sum(amount) AS total

FROM payments,customers

WHERE payments.customerNumber = customers.customerNumber

GROUP BY customerName

ORDER BY total DESC

LIMIT 1



1. Calculate the markup (selling price-buying price) for each product bought in each order. Sort the mark up in descending order.

SELECT orderNumber,orderdetails.productCode,(priceEach-buyPrice) AS markup

FROM orderdetails

INNER JOIN products

ON products.productCode = orderdetails.productCode

GROUP BY orderNumber,orderdetails.productCode

ORDER BY markup DESC;

Table

Description automatically generated

1. For each product, display all the other products which have the same vendor as the local product.

SELECT products.productName, products.productVendor, tmp1.allOthers

FROM products

INNER JOIN (

(SELECT productName AS allOthers, productVendor

FROM products)

AS tmp1)

WHERE (

(products.productVendor = tmp1.productVendor)

AND

(products.productName <> tmp1.allOthers)

)

ORDER BY products.productName;

Table

Description automatically generated with medium confidence

1. For those products which have not been ordered, display their product line and product line description.

SELECT products.productLine,productlines.textDescription

FROM products

LEFT JOIN orderdetails

ON products.productCode=orderdetails.productCode

INNER JOIN productlines

ON productlines.productLine = products.productLine

WHERE orderdetails.productCode is NULL



1. Display each product's code, name, buy price and the average buying price for its product line.

SELECT products.productCode, products.productName,

products.buyPrice, avg\_productline\_buyPrice

FROM products

INNER JOIN ((

SELECT AVG(buyPrice) AS avg\_productline\_buyPrice,products.productLine FROM products

INNER JOIN productlines

ON products.productLine = productlines.productLine

GROUP BY products.productLine) AS tmp1)

ON products.productLine = tmp1.productLine;

Table

Description automatically generated

1. List those customers (their number, name, order total amount for each order) which have placed an order with total amount spent of more than $5000.

SELECT customers.customerNumber,customers.customerName,total

FROM customers

INNER JOIN ((

SELECT sum(quantityOrdered\*priceEach) AS total,

orders.customerNumber

FROM orderdetails

INNER JOIN orders

ON orders.orderNumber = orderdetails.orderNumber

GROUP BY orders.orderNumber) as tmp1)

ON customers.customerNumber = tmp1.customerNumber

HAVING total>5000;

Table

Description automatically generated

1. List all the offices which are in the same city and state as the customers.

SELECT DISTINCT(office\_loc)

FROM (SELECT CONCAT(offices.city, ', ',offices.state)

AS office\_loc

FROM offices) AS tmp1

INNER JOIN (SELECT CONCAT(customers.city, ', ',customers.state)

AS customer\_loc

FROM customers) AS tmp2

ON office\_loc = customer\_loc;

Graphical user interface, text, application

Description automatically generated

1. Display each manager's employee number, their first name, last name and the # of employees they supervise.

SELECT employees.employeeNumber, employees.firstName, employees.lastName, supervises

FROM employees, (

SELECT reportsTo, COUNT(reportsTo) AS supervises

FROM employees

GROUP BY reportsTo

)

AS managers

WHERE employees.employeeNumber = managers.reportsTo;

A picture containing table

Description automatically generated

1. The data scientists often use data from the products and orderdetails tables to conduct data analyses. Create a view for all the data in these two tables.

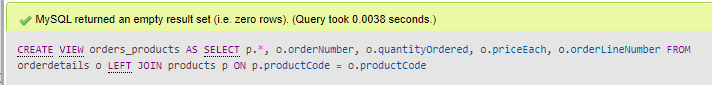
CREATE VIEW orders\_products AS

SELECT p.\*, o.orderNumber, o.quantityOrdered, o.priceEach, o.orderLineNumber

FROM orderdetails o

LEFT JOIN products p

ON p.productCode = o.productCode;



1. The data scientists often use data from the customers, orders, and orderdetails tables to conduct data analyses. Create a view for all the data in these three tables.

CREATE VIEW customers\_orders AS

SELECT c.\*,

r.orderNumber,r.orderDate,r.requiredDate,r.shippedDate,r.status,r.comments,

o.quantityOrdered,o.priceEach, o.orderLineNumber

FROM orderdetails o

RIGHT JOIN orders r ON r.orderNumber = o.orderNumber

RIGHT JOIN customers c on c.customerNumber = r.customerNumber;

Text

Description automatically generated

1. The data scientists often use data from the customers, orders, orderdetails, product tables to conduct data analyses. Create a view for all the data in these four tables.

CREATE VIEW customers\_orders\_products AS

SELECT

c.\*,

r.orderNumber,r.orderDate,r.requiredDate,r.shippedDate,r.status,r.comments,

o.quantityOrdered,o.priceEach, o.orderLineNumber,

p.\*

FROM orderdetails o

RIGHT JOIN products p ON p.productCode = o.productCode

RIGHT JOIN orders r ON r.orderNumber = o.orderNumber

RIGHT JOIN customers c on c.customerNumber = r.customerNumber;

Graphical user interface, text, application

Description automatically generated