SQL SALES

SELECT

YEAR(orderDate) AS order\_year,

MONTHNAME(orderDate) AS order\_month,

productlines.productLine AS category,

SUM(quantityOrdered) AS total\_quantity,

SUM(quantityOrdered) - LAG(SUM(quantityOrdered), 12) OVER (PARTITION BY productlines.textDescription ORDER BY YEAR(orderDate), MONTHNAME(orderDate)) AS variation,

ROUND((SUM(quantityOrdered) - LAG(SUM(quantityOrdered), 12) OVER (PARTITION BY productlines.textDescription ORDER BY YEAR(orderDate), MONTHNAME(orderDate))) / LAG(SUM(quantityOrdered), 12) OVER (PARTITION BY productlines.textDescription ORDER BY YEAR(orderDate), MONTHNAME(orderDate)), 2) AS rate\_of\_change

FROM

orderdetails

JOIN products ON orderdetails.productCode = products.productCode

JOIN productlines ON products.productLine = productlines.productLine

JOIN orders ON orderdetails.orderNumber = orders.orderNumber

GROUP BY

order\_year, order\_month, category

ORDER BY

order\_year DESC, order\_month DESC, category ASC

SELECT

\*,

total\_quantity-previous\_year as variation,

round(((total\_quantity - previous\_year)/previous\_year)\*100) as taux\_de\_variation

FROM (SELECT

\*,

LAG(total\_quantity, 1, 0) OVER(PARTITION BY category, order\_month ORDER BY category, order\_month, order\_year) as previous\_year

FROM (SELECT

YEAR(orderDate) AS order\_year,

MONTH(orderDate) AS order\_month,

products. productline AS category,

sum(quantityOrdered) AS total\_quantity

FROM orderdetails

JOIN orders ON orderdetails.orderNumber = orders.orderNumber

JOIN products ON orderdetails.productCode = products.productCode

GROUP BY category, order\_month, order\_year ) as sum\_quantity\_order) AS sum\_quantity\_order\_previous\_year

Sql logistique

SELECT products.productName, products.quantityInStock, sum(orderdetails.quantityOrdered) as Ordered\_quantity

FROM products

JOIN orderdetails ON products.productCode = orderdetails.productCode

GROUP BY products.productName, products.quantityInStock

ORDER BY sum(orderdetails.quantityOrdered) DESC

LIMIT 5;

SELECT products.productName, products.quantityInStock, sum(orderdetails.quantityOrdered)

FROM products

JOIN orderdetails ON products.productCode = orderdetails.productCode

GROUP BY products.productName, products.quantityInStock

ORDER BY quantityInStock

LIMIT 5;

Sql RH

SELECT \*

FROM (

SELECT

YEAR(orders.orderdate) AS order\_year,

MONTH(orders.orderDate) AS order\_month,

employees.firstname,

employees.lastname,

SUM(orderdetails.quantityOrdered \* orderdetails.priceEach) AS turnover,

RANK() OVER (PARTITION BY YEAR(orders.orderdate), MONTH(orders.orderdate) ORDER BY SUM(orderdetails.quantityOrdered \* orderdetails.priceEach) DESC) AS sales\_rank

FROM

orders

JOIN orderdetails ON orders.orderNumber = orderdetails.orderNumber

JOIN customers ON orders.customerNumber = customers.customerNumber

JOIN employees ON customers.salesRepEmployeeNumber = employees.employeeNumber

GROUP BY

order\_year, order\_month, employees.firstname, employees.lastname

) ranked

WHERE

sales\_rank < 3

ORDER BY

order\_year DESC, order\_month DESC, turnover DESC, lastname DESC, firstname DESC

SQL FINANCE

With debt

SELECT

f.customerNumber,

f.customerName,

Total\_commande,

Total\_reglement,

ROUND((Total\_commande-Total\_reglement),3) AS Debt

FROM

(SELECT

o.customerNumber,

customerName,

SUM(quantityOrdered\*priceEach) AS Total\_commande

FROM orderdetails AS od

INNER JOIN orders AS o ON od.orderNumber=o.orderNumber

INNER JOIN customers AS c ON c.customerNumber=o.customerNumber

GROUP BY o.customerNumber) AS f

INNER JOIN

(SELECT

payments.customerNumber,

SUM(amount) AS Total\_reglement

FROM payments

GROUP BY payments.customerNumber) AS d

ON f.customerNumber=d.customerNumber

ORDER BY Debt DESC

Lag 2 month

SELECT

payments.paymentDate,

customers.customerName,

customers.country,

SUM(payments.amount) as turnover

FROM customers

JOIN orders ON customers.customerNumber = orders.customerNumber

JOIN payments ON customers.customerNumber = payments.customerNumber

WHERE

payments.paymentDate >= DATE\_FORMAT(DATE\_SUB(DATE\_ADD(LAST\_DAY(NOW() - INTERVAL 2 MONTH), INTERVAL 1 DAY), INTERVAL 1 MONTH), '%Y-%m-%d')

AND payments.paymentDate <= DATE\_FORMAT(DATE\_SUB(DATE\_ADD(LAST\_DAY(NOW()), INTERVAL 1 DAY), INTERVAL 1 MONTH), '%Y-%m-%d')

GROUP BY payments.paymentDate, customers.customerName, customers.country

ORDER BY paymentDate DESC;

SELECT

customers.country,

SUM(payments.amount) as turnover

FROM customers

JOIN orders ON customers.customerNumber = orders.customerNumber

JOIN payments ON customers.customerNumber = payments.customerNumber

WHERE

payments.paymentDate >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH) and now()

GROUP BY customers.country

ORDER BY turnover DESC;