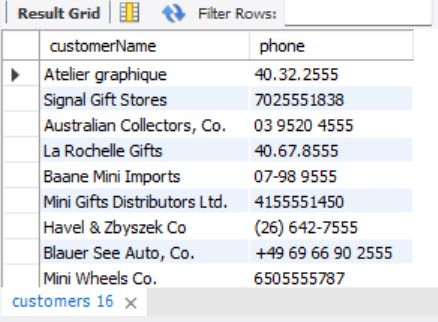
**MySQL QUERIES**

1. **Use SELECT, WHERE, ORDER BY, GROUP BY**
2. **SELECT** customer name and phone number from customers.

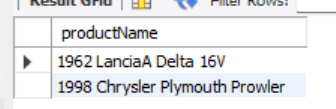
select customerName, phone from customers;



1. **WHERE**: Show all products from the products table where the buy price is greater than 100.

select productName from products

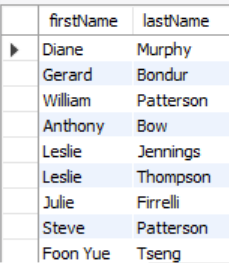
where buyPrice > 100;



1. **ORDER BY**: List all employees ordered by their job title in ascending order.

select firstName, lastName from employees

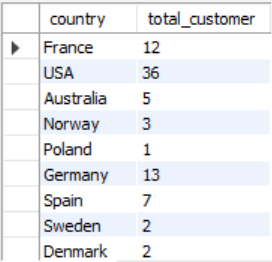
order by jobTitle asc;



1. **GROUP BY**: Show the number of customers in each country.

select country, count(customerNumber) as total\_customer from customers

group by country;

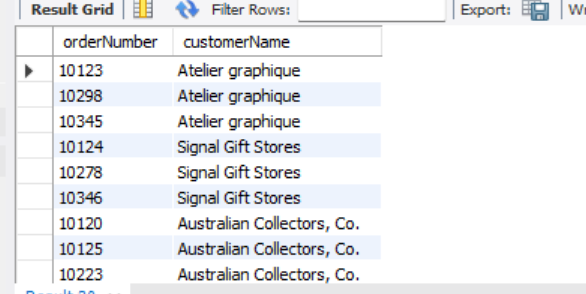


1. **Use JOINS (INNER, LEFT, RIGHT)**
2. **INNER JOIN**: List order numbers with corresponding customer names using orders and customers.

select orderNumber,customerName

from orders as o inner join customers as c

on o.customerNumber = c.customerNumber;



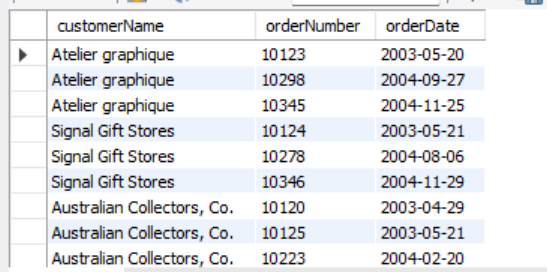
1. **LEFT JOIN**: Display all customers and their orders (if any), including those with no orders.

select customerName, orderNumber, orderDate

from customers as c left join orders as o

on c.customerNumber = o.customerNumber

order by c.customerNumber;



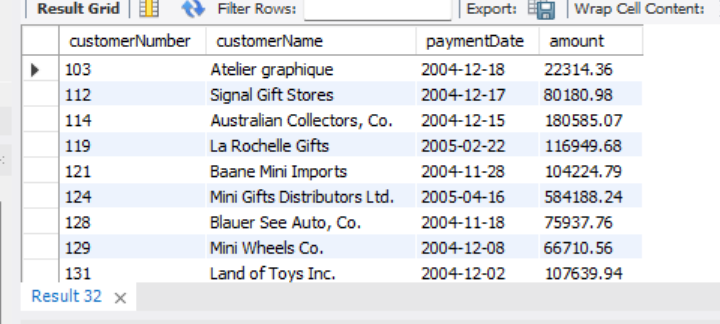
1. **RIGHT JOIN: List all payment details and include customer names if available, using payments and customers.**

select p.customerNumber, c.customerName, max(p.paymentDate) as paymentDate, sum(p.amount) as amount

from customers as c right join payments as p

on c.customerNumber = p.customerNumber

group by p.customerNumber,c.customerName;



1. **Write subqueries**
2. Show the names of customers who made a payment greater than the average payment amount.

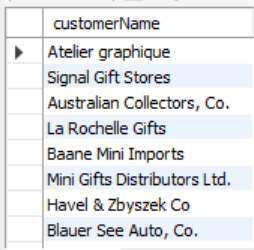
select customerName

from customers join payments

where amount > (select avg(amount)

from payments)

group by customerName;

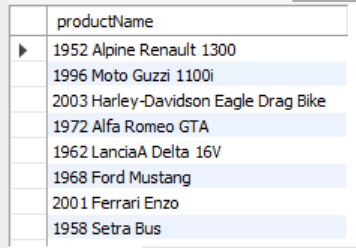


1. List all products with a price higher than the average price of all products.

select productName from products

where buyPrice > (select avg(buyPrice)

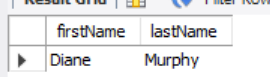
from products);



1. Find the employees who have the same office code as employee number 1002.

select firstName, lastName from employees

where employeeNumber = 1002;

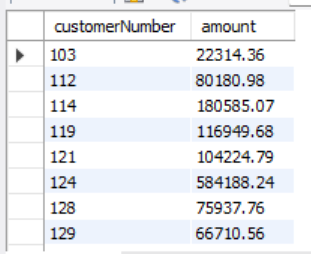


1. **Use aggregate functions (SUM, AVG)**
2. Show the total payments received from each customer.

select customerNumber, sum(amount) as amount

from payments

group by customerNumber;

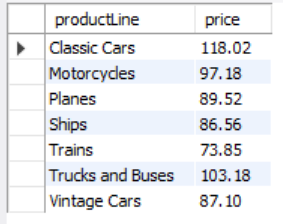


1. Display the average product price for each product line.

select productLine, round(avg(MSRP),2) as price

from products

group by productLine;

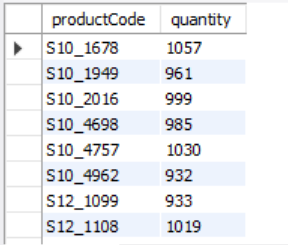


1. Find the total quantity ordered for each product from the orderdetails table.

select productCode, sum(quantityOrdered) as quantity

from orderdetails

group by productCode;



1. **Create views for analysis**
2. Create a view named CustomerOrdersView that shows customer name, order number, and order date.

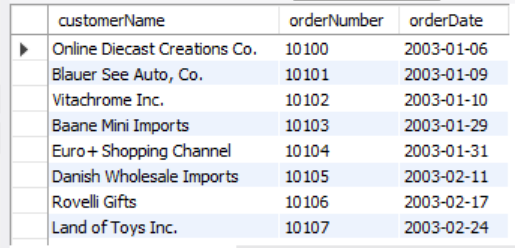
CREATE VIEW CustomerOrdersView

AS SELECT customerName, orderNumber, orderDate

from orders as o left join customers as c

on o.customerNumber = c.customerNumber;

select \* from CustomerOrdersView;



1. Create a view ProductLineStats to display product line, average price, and total products in each line.

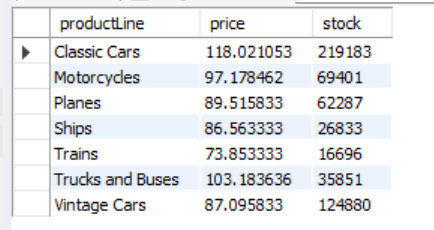
create view ProductLineStats

as select productLine, avg(MSRP) as price, sum(quantityInStock) as stock

from products

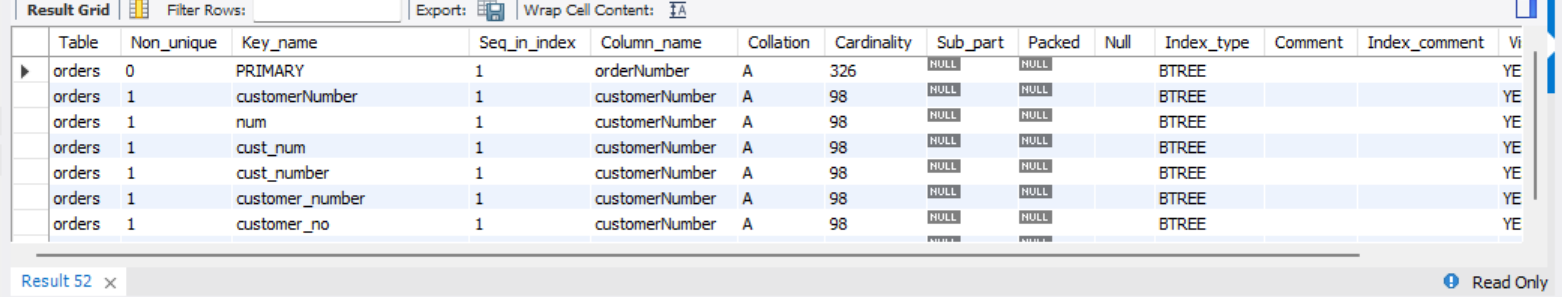
group by productLine;

select \* from ProductLineStats;



1. **Optimize queries with indexes**
2. Create an index on the customerNumber column in the orders table.

SHOW INDEXES FROM orders;



1. Create a composite index on orderNumber and productCode in orderDetails table.

CREATE INDEX idx\_order\_product

ON orderdetails(orderNumber, productCode);

SHOW INDEXES FROM orderdetails;

