**DBMS Assignment 2**

-- 1. Retrieve all employees

SELECT \* FROM Employees;

-- 2. Retrieve specific columns from the Products table

SELECT ProductID, ProductName, Price FROM Products;

-- 3. Retrieve all orders made on a specific date

SELECT \* FROM Orders WHERE OrderDate = '2024-03-01';

-- 4. Retrieve employees who earn more than 75,000

SELECT \* FROM Employees WHERE Salary > 75000;

-- 5. Retrieve all products with stock quantity equal to zero

SELECT \* FROM Products WHERE StockQuantity = 0;

-- 6. Retrieve all unique departments from Employees table

SELECT DISTINCT Department FROM Employees;

-- 7. Retrieve all unique product categories from the Products table

SELECT DISTINCT Category FROM Products;

-- 8. Retrieve all unique order amounts from the Orders table

SELECT DISTINCT TotalAmount FROM Orders;

-- 9. Retrieve all employees hired after January 1, 2021, and return only unique salaries

SELECT DISTINCT Salary FROM Employees WHERE HireDate > '2021-01-01';

-- 10. Retrieve all distinct years from OrderDate in Orders

SELECT DISTINCT YEAR(OrderDate) FROM Orders;

-- 11. Retrieve employees in the IT department

SELECT \* FROM Employees WHERE Department = 'IT';

-- 12. Retrieve products with a price greater than $500

SELECT \* FROM Products WHERE Price > 500;

-- 13. Retrieve orders placed on or after March 1, 2024

SELECT \* FROM Orders WHERE OrderDate >= '2024-03-01';

-- 14. Retrieve employees who are NOT in the HR department

SELECT \* FROM Employees WHERE Department <> 'HR';

-- 15. Retrieve employees with salaries between 70,000 and 85,000

SELECT \* FROM Employees WHERE Salary BETWEEN 70000 AND 85000;

-- 16. Retrieve orders placed between February 1, 2024, and March 31, 2024

SELECT \* FROM Orders WHERE OrderDate BETWEEN '2024-02-01' AND '2024-03-31';

-- 17. Retrieve products priced between 100 and 300

SELECT \* FROM Products WHERE Price BETWEEN 100 AND 300;

-- 18. Retrieve employees working in IT, Finance, or Marketing

SELECT \* FROM Employees WHERE Department IN ('IT', 'Finance', 'Marketing');

-- 19. Retrieve products in the Electronics or Accessories category

SELECT \* FROM Products WHERE Category IN ('Electronics', 'Accessories');

-- 20. Retrieve orders placed by customers with specific names

SELECT \* FROM Orders WHERE CustomerName IN ('Alice Johnson', 'Bob Smith');

-- 21. Retrieve employees whose first names start with 'J'

SELECT \* FROM Employees WHERE FirstName LIKE 'J%';

-- 22. Retrieve products that contain the word 'Smart' in their name

SELECT \* FROM Products WHERE ProductName LIKE '%Smart%';

-- 23. Retrieve customers whose names end with 'son'

SELECT \* FROM Orders WHERE CustomerName LIKE '%son';

-- 24. Retrieve employees whose first names contain 'a' as the second letter

SELECT \* FROM Employees WHERE FirstName LIKE '\_a%';

-- 25. Retrieve orders placed by customers whose names have exactly 5 characters

SELECT \* FROM Orders WHERE LENGTH(CustomerName) = 5;

-- 26. Count the total number of employees

SELECT COUNT(\*) FROM Employees;

-- 27. Count the number of products in the 'Electronics' category

SELECT COUNT(\*) FROM Products WHERE Category = 'Electronics';

-- 28. Count the number of unique departments in the Employees table

SELECT COUNT(DISTINCT Department) FROM Employees;

-- 29. Count the number of orders placed in March 2024

SELECT COUNT(\*) FROM Orders WHERE OrderDate BETWEEN '2024-03-01' AND '2024-03-31';

-- 30. Calculate the total salary paid to all employees

SELECT SUM(Salary) FROM Employees;

-- 31. Calculate the total stock quantity of all products

SELECT SUM(StockQuantity) FROM Products;

-- 32. Calculate the total revenue from all orders

SELECT SUM(TotalAmount) FROM Orders;

-- 33. Calculate the total revenue from orders placed after February 1, 2024

SELECT SUM(TotalAmount) FROM Orders WHERE OrderDate > '2024-02-01';

-- 34. Find the average employee salary

SELECT AVG(Salary) FROM Employees;

-- 35. Find the average product price

SELECT AVG(Price) FROM Products;

-- 36. Find the average order total

SELECT AVG(TotalAmount) FROM Orders;

-- 37. Find the lowest employee salary

SELECT MIN(Salary) FROM Employees;

-- 38. Find the cheapest product in the inventory

SELECT MIN(Price) FROM Products;

-- 39. Find the smallest order total

SELECT MIN(TotalAmount) FROM Orders;

-- 40. Find the highest salary among employees

SELECT MAX(Salary) FROM Employees;

-- 41. Find the most expensive product in stock

SELECT MAX(Price) FROM Products;

-- 42. Find the largest order total in the database

SELECT MAX(TotalAmount) FROM Orders;

-- 43. Retrieve all employees sorted by salary in ascending order

SELECT \* FROM Employees ORDER BY Salary ASC;

-- 44. Retrieve all employees sorted by salary in descending order

SELECT \* FROM Employees ORDER BY Salary DESC;

-- 45. Retrieve products sorted by price in descending order

SELECT \* FROM Products ORDER BY Price DESC;

-- 46. Retrieve orders sorted by OrderDate in ascending order

SELECT \* FROM Orders ORDER BY OrderDate ASC;

-- 47. Retrieve orders sorted by OrderDate in descending order

SELECT \* FROM Orders ORDER BY OrderDate DESC;

-- 48. Retrieve employees sorted by department alphabetically, then salary descending

SELECT \* FROM Employees ORDER BY Department ASC, Salary DESC;

-- 49. Retrieve products sorted by stock quantity in ascending order

SELECT \* FROM Products ORDER BY StockQuantity ASC;

-- 50. Retrieve the top 5 highest-paid employees

SELECT \* FROM Employees ORDER BY Salary DESC LIMIT 5;

-- 51. Retrieve the top 3 most expensive products

SELECT \* FROM Products ORDER BY Price DESC LIMIT 3;

-- 52. Retrieve the second highest-paid employee

SELECT \* FROM Employees ORDER BY Salary DESC LIMIT 1 OFFSET 1;

-- 53. Retrieve the third and fourth most expensive products

SELECT \* FROM Products ORDER BY Price DESC LIMIT 2 OFFSET 2;

-- 54. Retrieve the top 10 most recent orders

SELECT \* FROM Orders ORDER BY OrderDate DESC LIMIT 10;

-- 55. Retrieve the 5 cheapest products, ensuring correct tie-breaking

SELECT \* FROM Products ORDER BY Price ASC LIMIT 5;

-- 56. Retrieve the 10 employees with the lowest salaries, but skip the first 5

SELECT \* FROM Employees ORDER BY Salary ASC LIMIT 10 OFFSET 5;

-- 57. Retrieve the single most expensive product, ensuring correct tie-breaking

SELECT \* FROM Products ORDER BY Price DESC LIMIT 1;