**Q1 -**

**All Products**

**Task**

Write a query to output the Products table.

**Table name:** Products

| **product\_id** | **product\_name** | **price** | **category** |
| --- | --- | --- | --- |
| 1 | Laptop | 999.99 | Electronics |
| 2 | Desk Chair | 149.99 | Furniture |
| 3 | Smartphone | 599.99 | Electronics |
| 4 | Notebook | 2.99 | Stationery |
| 5 | Headphones | 89.99 | Electronics |
| 6 | Coffee Maker | 49.99 | Appliances |
| 7 | Standing Desk | 249.99 | Furniture |
| 8 | Tablet | 399.99 | Electronics |
| 9 | Mouse | 19.99 | Electronics |
| 10 | Water Bottle | 12.99 | Stationery |

--your code goes here

select \* from products;

**Q2 -**

**High Price of Products**

**Task**

Write a query to find all product\_name and category that have a price greater than 100.00 from the **Products** table.

Table name: **Products**

| **product\_id** | **product\_name** | **price** | **category** |
| --- | --- | --- | --- |
| 1 | Laptop | 999.99 | Electronics |
| 2 | Desk Chair | 149.99 | Furniture |
| 3 | Smartphone | 599.99 | Electronics |
| 4 | Notebook | 2.99 | Stationery |
| 5 | Headphones | 89.99 | Electronics |
| 6 | Coffee Maker | 49.99 | Appliances |
| 7 | Standing Desk | 249.99 | Furniture |
| 8 | Tablet | 399.99 | Electronics |
| 9 | Mouse | 19.99 | Electronics |
| 10 | Water Bottle | 12.99 | Stationery |

-- your code goes here

select product\_name, category from products

where price > 100.00

**Q3 -**

**Average Salary**

**Task**

* Write a query to calculate the average salary across all companies combined. Rename the column as avg\_salary.

**Table name:** Works

| **employee\_id** | **company\_name** | **salary** |
| --- | --- | --- |
| 1 | TechCorp | 75000.00 |
| 2 | InnovateLtd | 62000.50 |
| 3 | HealthPlus | 54000.75 |
| 4 | EduWorks | 48000.00 |
| 5 | GreenTech | 68000.00 |
| 6 | TechCorp | 80000.00 |
| 7 | InnovateLtd | 66000.20 |
| 8 | HealthPlus | 50000.10 |
| 9 | EduWorks | 51000.00 |
| 10 | GreenTech | 72000.00 |

-- your code goes here

select avg(salary) as avg\_salary

from works ;

**Q4 -**

**Locate People**

**Task**

* Write a query to retrieve the department\_name and location of people who live in location that starts with 'S'.

Table name: **departments**

| **department\_id** | **department\_name** | **location** |
| --- | --- | --- |
| 1 | Human Resources | New York |
| 2 | Research and Development | San Francisco |
| 3 | Sales | Los Angeles |
| 4 | Marketing | New York |
| 5 | Customer Support | Boston |
| 6 | Finance | Austin |
| 7 | IT Support | Seattle |
| 8 | Product Management | San Francisco |
| 9 | Quality Assurance | Los Angeles |
| 10 | Legal | Boston |

-- your code goes here

select department\_name, location from departments

where location like 'S%';

**Q5 -**

**Distinct Companies**

**Task**

Write a query to select all the distinct companies (company\_name) in the Works table.  
Table name: **Works**

| **employee\_id** | **company\_name** | **salary** |
| --- | --- | --- |
| 1 | TechCorp | 75000.00 |
| 2 | InnovateLtd | 62000.50 |
| 3 | HealthPlus | 54000.75 |
| 4 | EduWorks | 48000.00 |
| 5 | GreenTech | 68000.00 |
| 6 | TechCorp | 80000.00 |
| 7 | InnovateLtd | 66000.20 |
| 8 | HealthPlus | 50000.10 |
| 9 | EduWorks | 51000.00 |
| 10 | GreenTech | 72000.00 |

-- your code goes here

select distinct company\_name

from works

**Q6 -**

**Fiction Collection Size**

**Task**

Write a query to find the total count of books whose genre is **Fiction**.  
**Note**: Output column name should be **fiction\_count**.

Table name: **Books**

| **id** | **title** | **author** | **genre** | **price** | **published\_year** |
| --- | --- | --- | --- | --- | --- |
| 1 | The Great Gatsby | F. Scott Fitzgerald | Fiction | 10.99 | 1925 |
| 2 | 1984 | George Orwell | Dystopian | 15.99 | 1949 |
| 3 | To Kill a Mockingbird | Harper Lee | Fiction | 12.99 | 1960 |
| 4 | The Catcher in the Rye | J.D. Salinger | Fiction | 14.99 | 1951 |
| 5 | Brave New World | Aldous Huxley | Dystopian | 13.99 | 1932 |
| 6 | The Hobbit | J.R.R. Tolkien | Fantasy | 9.99 | 1937 |
| 7 | Moby Dick | Herman Melville | Fiction | 18.50 | 1851 |
| 8 | War and Peace | Leo Tolstoy | Historical | 20.00 | 1869 |
| 9 | The Picture of Dorian Gray | Oscar Wilde | Fiction | 11.50 | 1890 |
| 10 | The Alchemist | Paulo Coelho | Fiction | 16.00 | 1988 |
| 11 | Fahrenheit 451 | Ray Bradbury | Dystopian | 12.50 | 1953 |
| 12 | The Chronicles of Narnia | C.S. Lewis | Fantasy | 14.00 | 1950 |
| 13 | The Handmaid's Tale | Margaret Atwood | Dystopian | 15.50 | 1985 |
| 14 | A Tale of Two Cities | Charles Dickens | Historical | 13.00 | 1859 |
| 15 | Little Women | Louisa May Alcott | Fiction | 9.50 | 1868 |

-- your code goes here

select count(\*) as fiction\_count

from books

where genre = 'Fiction' ;

**Q7 -**

**List of Movies with Ratings**

**Task**

Write a query to select only the movie names where the ratings are greater than 7 but less than 9.

Table: **Cinema**

| **Movie\_id** | **Movie\_name** | **Description** | **Rating** |
| --- | --- | --- | --- |
| 1 | War | great 3D | 8.9 |
| 2 | Science | fiction | 8.5 |
| 3 | Irish | boring | 6.2 |
| 4 | Ice Song | Fantasy | 8.6 |
| 5 | House Card | Interesting | 9.1 |
| 6 | The Escape | Thriller | 7.8 |
| 7 | Solar Flare | Sci-Fi | 8.3 |
| 8 | The Joker | Drama | 9.0 |
| 9 | Lost Dreams | Mystery | 7.5 |
| 10 | Galaxy War | Action | 8.7 |

-- your code goes here

select movie\_name

from cinema

where rating > 7 and rating < 9;

**Q8 -**

**Handling NULL Values**

**Task**

Write a query to retrieve book\_id, title, author and published\_year of the books which have **NULL** rating for their books.

**Table name**: Library

| **book\_id** | **title** | **author** | **published\_year** | **rating** |
| --- | --- | --- | --- | --- |
| 1 | The Great Gatsby | F. Scott Fitzgerald | 1925 | 4.2 |
| 2 | To Kill a Mockingbird | Harper Lee | 1960 | NULL |
| 3 | 1984 | George Orwell | 1949 | 4.8 |
| 4 | The Catcher in the Rye | J.D. Salinger | 1951 | NULL |
| 5 | Brave New World | Aldous Huxley | 1932 | 4.3 |

-- your code goes here

select book\_id, title, author, published\_year

from library

where rating is null;

**Q9 -**

**Salary of Employees**

**Task**

Create a query to retrieve the employee\_name, company, and salary for employees in the full-time category, ordered by salary in **descending** order

**Table name**: Employees

| **employee\_id** | **employee\_name** | **company** | **category** | **department** | **salary** |
| --- | --- | --- | --- | --- | --- |
| 1 | John Smith | TechCorp | Full-Time | Engineering | 80000 |
| 2 | Alice Johnson | TechCorp | Part-Time | HR | 30000 |
| 3 | Bob Brown | FinServ | Full-Time | Finance | 90000 |
| 4 | Carol White | HealthPlus | Contract | IT | 75000 |
| 5 | David Green | TechCorp | Full-Time | Engineering | 85000 |
| 6 | Emma Blue | FinServ | Part-Time | Finance | 32000 |
| 7 | Frank Black | HealthPlus | Full-Time | HR | 60000 |
| 8 | Grace Grey | TechCorp | Full-Time | Marketing | 70000 |
| 9 | Henry Red | FinServ | Contract | IT | 95000 |
| 10 | Ivy Yellow | HealthPlus | Part-Time | Marketing | 28000 |

--your code goes here

select employee\_name, company, salary

from employees

where category = 'Full-Time'

order by salary desc;

**Q10 -**

**Department of Each Employee**

**Task**

Write a query to group the employees by their department and display the total number of employees (as total\_employees) in each department.

**Table name**: Employees

| **employee\_id** | **employee\_name** | **company** | **category** | **department** | **salary** |
| --- | --- | --- | --- | --- | --- |
| 1 | John Smith | TechCorp | Full-Time | Engineering | 80000 |
| 2 | Alice Johnson | TechCorp | Part-Time | HR | 30000 |
| 3 | Bob Brown | FinServ | Full-Time | Finance | 90000 |
| 4 | Carol White | HealthPlus | Contract | IT | 75000 |
| 5 | David Green | TechCorp | Full-Time | Engineering | 85000 |
| 6 | Emma Blue | FinServ | Part-Time | Finance | 32000 |
| 7 | Frank Black | HealthPlus | Full-Time | HR | 60000 |
| 8 | Grace Grey | TechCorp | Full-Time | Marketing | 70000 |
| 9 | Henry Red | FinServ | Contract | Sales | 95000 |
| 10 | Ivy Yellow | HealthPlus | Part-Time | Marketing | 28000 |

-- your code goes here

select department, count(\*) as total\_employees

from employees

group by department

**Q11 -**

**Article views**

**Task**

Write a query to retrieve the author\_id, author\_name, and publication\_name for authors whose articles got zero views. The result should be sorted by author\_id in ascending order.

Return the result table sorted by id in **ascending order**.

**Table name**: Views

| **article\_id** | **author\_id** | **author\_name** | **viewer\_id** | **view\_count** | **publication\_name** |
| --- | --- | --- | --- | --- | --- |
| 101 | 1 | John Doe | 1 | 3 | The Daily Times |
| 102 | 2 | Jane Smith | 2 | 1 | Global News |
| 103 | 3 | Emily Clark | 4 | 0 | Tech Monthly |
| 104 | 4 | Mark Lewis | 4 | 2 | Health Digest |
| 105 | 5 | Sara White | 3 | 0 | The Business Journal |
| 106 | 1 | John Doe | 2 | 0 | The Daily Times |
| 107 | 2 | Jane Smith | 2 | 2 | Global News |

-- your code goes here

select author\_id, author\_name, publication\_name

from views

where view\_count = 0

order by author\_id asc;

**Q12 -**

**Player Performance Insights**

**Task**

Write a query to find the names of the **top 3 distinct players** by highest score who have **won** matches, including their **scores**.

**Expected Output Columns:**

| **player\_name** | **score** |
| --- | --- |

Table 1: **Players**

| **player\_id** | **player\_name** | **score** | **rank** |
| --- | --- | --- | --- |
| 1 | Alice | 1200 | 5 |
| 2 | Bob | 1500 | 2 |
| 3 | Charlie | 1300 | 4 |
| 4 | David | 1600 | 1 |
| 5 | Eve | 1100 | 6 |

Table 2: **Matches**

| **match\_id** | **player1** | **player2** | **winner** | **match\_date** |
| --- | --- | --- | --- | --- |
| 101 | Alice | Bob | Bob | 2024-01-15 |
| 102 | Charlie | David | David | 2024-01-16 |
| 103 | Eve | Bob | Bob | 2024-01-17 |
| 104 | Alice | David | David | 2024-01-18 |
| 105 | Charlie | Eve | Charlie | 2024-01-19 |

-- your code goes here

select distinct player\_name, score

from players

order by score desc

limit 3;

**Q13 -**

**Player Details**

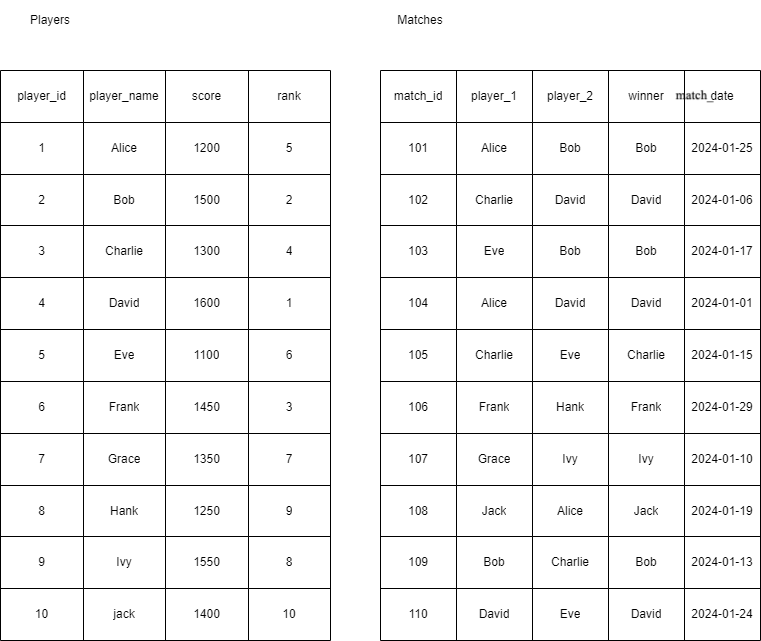
**Task**

Write a query to retrieve the details of the **last five matches** played, including the match ID, the names of the players who participated, the name of the winning player, and the final score of the winner.

**Expected Output Columns:**

| **match\_id** | **player\_1** | **player\_2** | **winner** | **match\_date** | **score** |
| --- | --- | --- | --- | --- | --- |

There are two tables named **Players** and **Matches**.



-- your code goes here

select m.match\_id, m.player\_1, m.player\_2, m.winner, m.match\_date, p.score

from matches m

inner join players p on p.player\_name = m.winner

order by match\_date desc

limit 5;