**Task 1: Join Tables to Retrieve Detailed Book Information**

**Problem**: Retrieve a list of books along with the branch details where the book is available, including the manager's name and branch contact information.

-- Query to join books with branch and employee details

SELECT

b.book\_title,

b.category,

br.branch\_address,

br.contact\_no,

e.emp\_name AS manager\_name

FROM

books b

JOIN

branch br ON br.branch\_id = b.branch\_id

JOIN

employees e ON e.emp\_id = br.manager\_id

ORDER BY

b.book\_title;

**Task 2: Calculate the Average Salary per Branch**

**Problem**: Calculate the average salary of employees for each branch, sorted by the branch with the highest average salary.

SELECT

br.branch\_id,

br.branch\_address,

AVG(e.salary) AS avg\_salary

FROM

employees e

JOIN

branch br ON e.branch\_id = br.branch\_id

GROUP BY

br.branch\_id, br.branch\_address

ORDER BY

avg\_salary DESC;

**Task 3: Identify Top-Paid Employees per Branch**

**Problem**: Retrieve the top 2 highest-paid employees from each branch.

WITH RankedSalaries AS (

SELECT

e.emp\_name,

e.branch\_id,

e.salary,

ROW\_NUMBER() OVER (PARTITION BY e.branch\_id ORDER BY e.salary DESC) AS rank

FROM

employees e

)

SELECT

emp\_name,

branch\_id,

salary

FROM

RankedSalaries

WHERE

rank <= 2;

**Task 4: Analyze Book Rental Trends by Category**

**Problem**: Calculate the total rental price for each book category, then rank the categories by their total revenue.

SELECT

category,

SUM(rental\_price) AS total\_revenue,

RANK() OVER (ORDER BY SUM(rental\_price) DESC) AS revenue\_rank

FROM

books

GROUP BY

category

ORDER BY

revenue\_rank;

**Task 5: List All Books Not Rented Yet**

**Problem**: Identify all books that have not been rented yet.

SELECT

book\_title,

category,

author

FROM

books

WHERE

status = 'no';

**Task 6: Calculate the Running Total of Book Rentals by Category**

**Problem**: Generate a running total of book rentals for each category over time.

sql

Copy code

-- Query to calculate the running total of book rentals by category

SELECT

category,

book\_title,

rental\_price,

SUM(rental\_price) OVER (PARTITION BY category ORDER BY book\_title) AS running\_total

FROM

books

ORDER BY

category, book\_title;

**Task 7: Retrieve Books with Above-Average Rental Price**

**Problem**: List books that have a rental price above the average rental price for their category.

SELECT

b.book\_title,

b.category,

b.rental\_price

FROM

books b

JOIN

(SELECT

category,

AVG(rental\_price) AS avg\_rental\_price

FROM

books

GROUP BY

category) c

ON

b.category = c.category

WHERE

b.rental\_price > c.avg\_rental\_price;

**Task 8: Identify Branches with No Employees**

**Problem**: Find all branches that currently have no employees assigned to them.

sql

Copy code

-- Query to find branches with no employees

SELECT

br.branch\_id,

br.branch\_address

FROM

branch br

LEFT JOIN

employees e ON br.branch\_id = e.branch\_id

WHERE

e.emp\_id IS NULL;

**Task 9: Calculate Total Rental Revenue by Author**

**Problem**: Calculate the total rental revenue generated by each author.

SELECT

author,

SUM(rental\_price) AS total\_revenue

FROM

books

GROUP BY

author

ORDER BY

total\_revenue DESC;

**Task 10: Rank Employees by Salary Within Their Position**

**Problem**: Rank employees based on their salary within their specific job position.

SELECT

emp\_name,

position,

salary,

RANK() OVER (PARTITION BY position ORDER BY salary DESC) AS salary\_rank

FROM

employees;

**Task 1: Join Tables to Retrieve Detailed Book Information**

**Problem**: Retrieve a list of books along with the branch details where the book is available, including the manager's name and branch contact information.

SELECT

b.book\_title,

b.category,

br.branch\_address,

br.contact\_no,

e.emp\_name AS manager\_name

FROM

books b

JOIN

branch br ON br.branch\_id = b.branch\_id

JOIN

employees e ON e.emp\_id = br.manager\_id

ORDER BY

b.book\_title;

**Task 2: Calculate the Average Salary per Branch**

**Problem**: Calculate the average salary of employees for each branch, sorted by the branch with the highest average salary.

SELECT

br.branch\_id,

br.branch\_address,

AVG(e.salary) AS avg\_salary

FROM

employees e

JOIN

branch br ON e.branch\_id = br.branch\_id

GROUP BY

br.branch\_id, br.branch\_address

ORDER BY

avg\_salary DESC;

*Explanation*: This query calculates the average salary of employees per branch, using GROUP BY and sorts the branches by the highest average salary.

**Task 3: Identify Top-Paid Employees per Branch**

**Problem**: Retrieve the top 2 highest-paid employees from each branch.

WITH RankedSalaries AS (

SELECT

e.emp\_name,

e.branch\_id,

e.salary,

ROW\_NUMBER() OVER (PARTITION BY e.branch\_id ORDER BY e.salary DESC) AS rank

FROM

employees e

)

SELECT

emp\_name,

branch\_id,

salary

FROM

RankedSalaries

WHERE

rank <= 2;

**Task 4: Analyze Book Rental Trends by Category**

**Problem**: Calculate the total rental price for each book category, then rank the categories by their total revenue.

SELECT

category,

SUM(rental\_price) AS total\_revenue,

RANK() OVER (ORDER BY SUM(rental\_price) DESC) AS revenue\_rank

FROM

books

GROUP BY

category

ORDER BY

revenue\_rank;

*Explanation*: This query sums the rental prices per category and ranks them using the RANK() function.

**Task 5: List All Books Not Rented Yet**

**Problem**: Identify all books that have not been rented yet.

SELECT

book\_title,

category,

author

FROM

books

WHERE

status = 'no';

**Task 6: Calculate the Running Total of Book Rentals by Category**

**Problem**: Generate a running total of book rentals for each category over time.

SELECT

category,

book\_title,

rental\_price,

SUM(rental\_price) OVER (PARTITION BY category ORDER BY book\_title) AS running\_total

FROM

books

ORDER BY

category, book\_title;

*Explanation*: The query uses the SUM() window function to calculate the running total of rentals per category.

**Task 7: Retrieve Books with Above-Average Rental Price**

**Problem**: List books that have a rental price above the average rental price for their category.

SELECT

b.book\_title,

b.category,

b.rental\_price

FROM

books b

JOIN

(SELECT

category,

AVG(rental\_price) AS avg\_rental\_price

FROM

books

GROUP BY

category) c

ON

b.category = c.category

WHERE

b.rental\_price > c.avg\_rental\_price;

**Task 8: Identify Branches with No Employees**

**Problem**: Find all branches that currently have no employees assigned to them.

SELECT

br.branch\_id,

br.branch\_address

FROM

branch br

LEFT JOIN

employees e ON br.branch\_id = e.branch\_id

WHERE

e.emp\_id IS NULL;

**Task 9: Calculate Total Rental Revenue by Author**

**Problem**: Calculate the total rental revenue generated by each author.

SELECT

author,

SUM(rental\_price) AS total\_revenue

FROM

books

GROUP BY

author

ORDER BY

total\_revenue DESC;

.

**Task 10: Rank Employees by Salary Within Their Position**

**Problem**: Rank employees based on their salary within their specific job position.

SELECT

emp\_name,

position,

salary,

RANK() OVER (PARTITION BY position ORDER BY salary DESC) AS salary\_rank

FROM

employees;