**p1\_a**

CREATE SCHEMA LibraryManagement;

**p1\_b**

CREATE SCHEMA LibraryManagement;

CREATE TABLE authors (

author\_id INT AUTO\_INCREMENT PRIMARY KEY,

author\_name VARCHAR(50)

);

**p1\_c**

CREATE SCHEMA LibraryManagement;

CREATE TABLE genres (

genre\_id INT AUTO\_INCREMENT PRIMARY KEY,

genre\_name VARCHAR(50)

);

**p1\_d**

CREATE SCHEMA LibraryManagement;

CREATE TABLE books (

book\_id INT AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255),

publication\_year YEAR,

author\_id INT,

genre\_id INT,

FOREIGN KEY (author\_id) REFERENCES authors(author\_id),

FOREIGN KEY (genre\_id) REFERENCES genres(genre\_id)

);

**p1\_e**

CREATE SCHEMA LibraryManagement;

CREATE TABLE users (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(255),

email VARCHAR(255)

);

**p1\_f**

CREATE SCHEMA LibraryManagement;

CREATE TABLE borrowed\_books (

borrow\_id INT AUTO\_INCREMENT PRIMARY KEY,

book\_id INT,

user\_id INT,

borrow\_date DATE,

return\_date DATE,

FOREIGN KEY (book\_id) REFERENCES books(book\_id),

FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

**p2**

USE librarymanagement;

INSERT INTO authors (author\_name) VALUES

('Jaroslav Hashek'),

('Franz Kafka');

INSERT INTO genres (genre\_name) VALUES

('detective'),

('novel');

INSERT INTO books (title, publication\_year, author\_id, genre\_id) VALUES

('The adventures of the good soldier Schweik', 1923, 1, 3),

('Description of one struggle', 1905, 2, 4);

INSERT INTO users (username, email) VALUES

('Serg', 'serg@mymail.com'),

('Elice', 'victory@in.ua');

INSERT INTO borrowed\_books (book\_id, user\_id, borrow\_date, return\_date) VALUES

(1, 1, '2023-09-28', '2024-01-10'),

(2, 2, '2023-10-25', '2024-02-05');

**p3**

USE `3`;

SELECT od.id,

od.quantity,

orders.id AS order\_id,

orders.date,

customers.id AS customer\_id,

customers.name,

customers.contact,

customers.address,

customers.city,

customers.postal\_code,

customers.country,

products.id AS product\_id,

products.name,

products.unit,

products.price,

products.supplier\_id,

products.category\_id,

categories.name,

categories.description,

employees.employee\_id,

employees.last\_name,

employees.first\_name,

employees.birthdate,

employees.photo,

employees.notes,

shippers.id AS shippers\_id,

shippers.name,

shippers.phone,

suppliers.id AS supplier\_id,

suppliers.name,

suppliers.contact,

suppliers.address,

suppliers.city,

suppliers.postal\_code,

suppliers.country,

suppliers.phone

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

**p4\_1**

USE `3`;

SELECT COUNT(\*) AS total\_rows

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

**p4\_2**

USE `3`;

SELECT COUNT(\*) AS total\_rows

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

LEFT JOIN customers ON orders.customer\_id = customers.id

RIGHT JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

LEFT JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

І отримаємо все тіж 518 строк, що говорить нам про те, що в нашому випадку, оператори INNER, LEFT та RIGHT JOIN не впливають на кількість рядків, які повертаються, оскільки кількість рядків визначається в основному наборі даних, який вибирається, і не змінюється в залежності від типу з'єднання з іншими таблицями.

**p4\_3**

USE `3`;

SELECT od.id,

od.quantity,

orders.id AS order\_id,

orders.date,

customers.id AS customer\_id,

customers.name,

customers.contact,

customers.address,

customers.city,

customers.postal\_code,

customers.country,

products.id AS product\_id,

products.name,

products.unit,

products.price,

products.supplier\_id,

products.category\_id,

categories.name,

categories.description,

employees.employee\_id,

employees.last\_name,

employees.first\_name,

employees.birthdate,

employees.photo,

employees.notes,

shippers.id AS shippers\_id,

shippers.name,

shippers.phone,

suppliers.id AS supplier\_id,

suppliers.name,

suppliers.contact,

suppliers.address,

suppliers.city,

suppliers.postal\_code,

suppliers.country,

suppliers.phone

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

WHERE employees.employee\_id > 3 AND employees.employee\_id <= 10

**p4\_4**

USE `3`;

SELECT categories.name AS category\_name,

COUNT(\*) AS total\_rows,

AVG(od.quantity) AS average\_quantity

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

GROUP BY category\_name

**p4\_5**

USE `3`;

SELECT categories.name AS category\_name,

COUNT(\*) AS total\_rows,

AVG(od.quantity) AS average\_quantity

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

GROUP BY category\_name

HAVING average\_quantity > 21

**p4\_6**

USE `3`;

SELECT categories.name AS category\_name,

COUNT(\*) AS total\_rows,

AVG(od.quantity) AS average\_quantity

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

GROUP BY category\_name

HAVING average\_quantity > 21

ORDER BY total\_rows DESC

**p4\_7**

USE `3`;

SELECT categories.name AS category\_name,

COUNT(\*) AS total\_rows,

AVG(od.quantity) AS average\_quantity

FROM order\_details AS od

INNER JOIN orders ON od.order\_id = orders.id

INNER JOIN customers ON orders.customer\_id = customers.id

INNER JOIN products ON od.product\_id = products.id

INNER JOIN categories ON products.category\_id = categories.id

INNER JOIN employees ON orders.employee\_id = employees.employee\_id

INNER JOIN shippers ON orders.shipper\_id = shippers.id

INNER JOIN suppliers ON products.supplier\_id = suppliers.id

GROUP BY category\_name

HAVING average\_quantity > 21

ORDER BY total\_rows DESC

LIMIT 4

OFFSET 1