P1_a

```
CREATE SCHEMA LibraryManagement;
```

P1_b-f

```
USE LibraryManagement;

CREATE TABLE authors (
    author_id INT AUTO_INCREMENT PRIMARY KEY,
    author_name VARCHAR(45)
);

CREATE TABLE genres (
    genre_id INT AUTO_INCREMENT PRIMARY KEY,
    genre_name VARCHAR(45)
);

CREATE TABLE books (
    book_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(45),
    publication_year YEAR,
    author_id INT,
    FOREIGN KEY (author_id)
        REFERENCES authors (author_id),
    genre_id INT,
    FOREIGN KEY (genre_id)
        REFERENCES genres (genre_id)
);

CREATE TABLE users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(45),
    email VARCHAR(225)
);

CREATE TABLE borrowed_books (
    borrow_id INT AUTO_INCREMENT PRIMARY KEY,
    borrow_date DATE,
```

```
    return_date DATE,
    book_id INT,
    FOREIGN KEY (book_id)
        REFERENCES books (book_id),
    user_id INT,
    FOREIGN KEY (user_id)
        REFERENCES users (user_id)
);
```

P2

```
USE LibraryManagement;

INSERT INTO authors (author_name)
VALUES ("Олександр Довженко"), ("Василь Симоненко");
SELECT * FROM authors;

INSERT INTO genres (genre_name)
VALUES ("Повісті"), ("Вірші");
SELECT * FROM genres;

INSERT INTO books (title, publication_year, author_id, genre_id)
VALUES ("Зачарована Десна", 1956, 1, 1), ("Тиша і грім", 1962, 2, 2);
SELECT * FROM books;

INSERT INTO users (username, email)
VALUES ("Юлія", "juliastop@gmail.com"), ("Олександр",
"alexde@gmail.com");
SELECT * FROM users;

INSERT INTO borrowed_books (borrow_date, return_date, book_id, user_id)
VALUES ("2024-04-24", "2023-07-16", 1, 1), ("2022-12-01", "2024-03-11", 2,
2);
SELECT * FROM borrowed_books;
```

P3

```
USE mydb;

SELECT *
FROM orders
INNER JOIN customers ON orders.customer_id = customers.id
INNER JOIN employees ON orders.employee_id  = employees.employee_id
INNER JOIN shippers ON orders.shipper_id  = shippers.id
INNER JOIN order_details ON orders.id = order_details.order_id
INNER JOIN products ON order_details.product_id = products.id
INNER JOIN suppliers ON products.supplier_id = suppliers.id
INNER JOIN categories ON products.category_id = categories.id;
```

P4

P4_a

```
SELECT
   COUNT(*) AS total_row
FROM
   order_details
      INNER JOIN
   orders ON order_details.order_id = orders.id
      INNER JOIN
   products ON order_details.product_id = products.id
      INNER JOIN
   customers ON orders.customer_id = customers.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
    INNER JOIN
  categories ON products.category_id = categories.id
```

P4_b

```
SELECT
  COUNT(*) AS row_quantity
FROM
  order_details
    LEFT JOIN
  orders ON order_details.order_id = orders.id
    LEFT JOIN
  products ON order_details.product_id = products.id
    LEFT JOIN
  customers ON orders.customer_id = customers.id
    LEFT JOIN
  employees ON orders.employee_id = employees.employee_id
    LEFT JOIN
  shippers ON orders.shipper_id = shippers.id
    LEFT JOIN
  suppliers ON products.supplier_id = suppliers.id
    LEFT JOIN
  categories ON products.category_id = categories.id
```

P4_c

```
SELECT
  COUNT(*) AS row_quantity
FROM
  order_details
    LEFT JOIN
  orders ON order_details.order_id = orders.id
    LEFT JOIN
  products ON order_details.product_id = products.id
```

```
    LEFT JOIN
  customers ON orders.customer_id = customers.id
    LEFT JOIN
  employees ON orders.employee_id = employees.employee_id
    LEFT JOIN
  shippers ON orders.shipper_id = shippers.id
    LEFT JOIN
  suppliers ON products.supplier_id = suppliers.id
    LEFT JOIN
  categories ON products.category_id = categories.id
WHERE
  employees.employee_id > 3
    AND employees.employee_id <= 10
```

P4_d

```
SELECT
  categories.name AS category_name,
  COUNT(*) AS row_quantity,
  AVG(order_details.quantity) AS avg_quantity
FROM
  order_details
    LEFT JOIN
  orders ON order_details.order_id = orders.id
    LEFT JOIN
  products ON order_details.product_id = products.id
    LEFT JOIN
  customers ON orders.customer_id = customers.id
    LEFT JOIN
  employees ON orders.employee_id = employees.employee_id
    LEFT JOIN
  shippers ON orders.shipper_id = shippers.id
    LEFT JOIN
  suppliers ON products.supplier_id = suppliers.id
    LEFT JOIN
  categories ON products.category_id = categories.id
WHERE
```

```
    employees.employee_id > 3
        AND employees.employee_id <= 10
GROUP BY categories.name
```

P4_e

```
SELECT
    categories.name AS category_name,
    COUNT(*) AS row_quantity,
    AVG(order_details.quantity) AS avg_quantity
FROM
    order_details
        LEFT JOIN
    orders ON order_details.order_id = orders.id
        LEFT JOIN
    products ON order_details.product_id = products.id
        LEFT JOIN
    customers ON orders.customer_id = customers.id
        LEFT JOIN
    employees ON orders.employee_id = employees.employee_id
        LEFT JOIN
    shippers ON orders.shipper_id = shippers.id
        LEFT JOIN
    suppliers ON products.supplier_id = suppliers.id
        LEFT JOIN
    categories ON products.category_id = categories.id
WHERE
    employees.employee_id > 3
        AND employees.employee_id <= 10
GROUP BY categories.name
HAVING AVG(order_details.quantity) > 21
```

P4_f

```
SELECT
    categories.name AS category_name,
    COUNT(*) AS row_quantity,
```

```
    AVG(order_details.quantity) AS avg_quantity
FROM
    order_details
        LEFT JOIN
    orders ON order_details.order_id = orders.id
        LEFT JOIN
    products ON order_details.product_id = products.id
        LEFT JOIN
    customers ON orders.customer_id = customers.id
        LEFT JOIN
    employees ON orders.employee_id = employees.employee_id
        LEFT JOIN
    shippers ON orders.shipper_id = shippers.id
        LEFT JOIN
    suppliers ON products.supplier_id = suppliers.id
        LEFT JOIN
    categories ON products.category_id = categories.id
WHERE
    employees.employee_id > 3
        AND employees.employee_id <= 10
GROUP BY categories.name
HAVING AVG(order_details.quantity) > 21
ORDER BY row_quantity DESC
```

P4_g

```
SELECT
    categories.name AS category_name,
    COUNT(*) AS row_quantity,
    AVG(order_details.quantity) AS avg_quantity
FROM
    order_details
        LEFT JOIN
    orders ON order_details.order_id = orders.id
        LEFT JOIN
    products ON order_details.product_id = products.id
        LEFT JOIN
```

```
    customers ON orders.customer_id = customers.id
        LEFT JOIN
    employees ON orders.employee_id = employees.employee_id
        LEFT JOIN
    shippers ON orders.shipper_id = shippers.id
        LEFT JOIN
    suppliers ON products.supplier_id = suppliers.id
        LEFT JOIN
    categories ON products.category_id = categories.id
WHERE
    employees.employee_id > 3
        AND employees.employee_id <= 10
GROUP BY categories.name
HAVING AVG(order_details.quantity) > 21
ORDER BY row_quantity DESC
LIMIT 4 OFFSET 1
```

Висновок P4_b:

Обидва запити, що використовують INNER JOIN та LEFT JOIN, повернуть однакові результати, оскільки у нашому випадку всі записи мають відповідності у всіх залучених таблицях.

LEFT JOIN включає всі записи з основної таблиці (order_details) і відповідні записи з інших таблиць. Однак, оскільки у нас немає записів без відповідностей, результати будуть такими ж, як і при використанні INNER JOIN. Тому, незалежно від того, який тип з'єднання використовується, кількість рядків у кінцевому результаті запиту залишиться незмінною через відсутність пропущених співставлень та наявність всіх значень у пов'язаних таблицях.