Таблица departments — отделы

CREATE TABLE departments (

department\_id INT PRIMARY KEY,

name VARCHAR(100)

);

INSERT INTO departments (department\_id, name) VALUES(1, 'Development'),

(2, 'Marketing'),

(3, 'Finance'),

(4, 'HR');

Таблица employees — сотрудники

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

name VARCHAR(100),

department\_id INT,

salary DECIMAL(10, 2),

manager\_id INT);

INSERT INTO employees (employee\_id, name, department\_id, salary, manager\_id) VALUES(101, 'Alice', 1, 60000, NULL),

(102, 'Bob', 1, 55000, 101),

(103, 'Charlie', 2, 50000, 101),

(104, 'Diana', NULL, 40000, NULL),

(105, 'Eve', 3, 45000, 102),

(106, 'Frank', 4, 47000, NULL);

Таблица projects — проекты

CREATE TABLE projects (

project\_id INT PRIMARY KEY,

name VARCHAR(100),

employee\_id INT);

INSERT INTO projects (project\_id, name, employee\_id) VALUES(201, 'CRM System', 101),

(202, 'Marketing Campaign', 103),

(203, 'ERP System', 105),

(204, 'HR Platform', NULL);

Задание 1: INNER JOIN

Выведите имена сотрудников и названия отделов, в которых они работают (только для тех, кто прикреплён к отделу).

Ожидается использование: INNER JOIN

Подсказка: соединить employees и departments по department\_id

**SELECT**

*e*.name **AS** *employees\_name*,

*d*.name **AS** department*\_name*

**FROM** employees *e*

**INNER** **JOIN** departments **AS** *d*

**ON** *e*.department\_id= *d*.department\_id;

Задание 2: LEFT JOIN

Выведите список всех сотрудников и, если есть, названия отделов, в которых они работают. Даже если сотрудник не прикреплён к отделу, он должен быть в списке.

Ожидается использование: LEFT JOIN

**SELECT**

*e*.name **AS** *employees\_name*,

*d*.name **AS** department *\_name*

**FROM** employees *e*

**LEFT** **JOIN** departments **AS** *d*

**ON** *e*.department\_id = *d*.department\_id;

Задание 3: RIGHT JOIN

Выведите список всех отделов и, если есть, сотрудников, которые в них работают. Даже если в отделе никого нет, он должен быть в выводе.

Ожидается использование: RIGHT JOIN

**SELECT**

*e*.name **AS** *employees\_name*,

*d*.name **AS** *department\_name*

**FROM** employees *e*

**RIGHT** **JOIN** departments **AS** *d*

**ON** *e*.department\_id = *d*.department\_id;

Задание 4: FULL JOIN

Постройте таблицу, в которой будут отображены все проекты и сотрудники, к которым они привязаны. Покажите также проекты без сотрудников и сотрудников без проектов.

Ожидается использование: FULL JOIN

**SELECT**

*p*.name **AS** *projects\_name*,

*e*.name **AS** *employees\_name*

**FROM** employees *e*

**FULL** **JOIN** projects **as** *p*

**ON** *e*.employee\_id = *p*.employee\_id;

Задание 5: SELF JOIN (иерархия сотрудников)

Сформируйте таблицу, в которой будет указано имя сотрудника и имя его менеджера. Если у сотрудника нет менеджера, укажите NULL.

Ожидается использование: SELF JOIN

**SELECT**

*e*.name **AS** *employees\_name*,

*m*.name **AS** *manager\_name*

**FROM** employees *e*

**LEFT** **JOIN** employees **as** *m*

**ON** *e*.department\_id = *m*.department\_id;

Задание 6: Дополнительное — фильтрация по NULL (ANTI JOIN)

Выведите список всех сотрудников, у которых нет отдела.

Ожидается использование: LEFT JOIN + WHERE department.name IS NULL

**SELECT**

*e*.name **AS** *employees\_name*,

*d*.name **AS** *department\_name*

**FROM** employees *e*

**LEFT** **JOIN** departments **AS** *d*

**ON** *e*.department\_id = *d*.department\_id

**WHERE** *e*.department\_id **IS** **NULL**;

Задание 7: Дополнительное — сотрудники без проектов

Выведите имена сотрудников, которые не участвуют ни в одном проекте.

Ожидается использование: LEFT JOIN + WHERE project.name IS NULL

**SELECT**

*e*.name **AS** *employees\_name*,

*p*.name **AS** *projects\_name*

**FROM** employees *e*

**LEFT** **JOIN** projects **as** *p*

**ON** *e*.employee\_id = *p*.employee\_id

**WHERE** *p*.employee\_id **IS** **NULL**;