```
CREATE TABLE Employees (
EmployeeID SERIAL PRIMARY KEY, -- SERIAL for auto-incrementing integer IDs in
PostgreSQL
FirstName VARCHAR(50) NOT NULL,
LastName VARCHAR(50) NOT NULL,
Department VARCHAR(50),
Salary DECIMAL(10, 2)
);
CREATE TABLE Projects (
ProjectID SERIAL PRIMARY KEY, -- SERIAL for auto-incrementing integer IDs
ProjectName VARCHAR(100) NOT NULL,
Budget DECIMAL(12, 2),
StartDate DATE,
EndDate DATE
);
CREATE TABLE EmployeeProjects (
EmployeeID INT,
ProjectID INT,
HoursWorked INT,
PRIMARY KEY (EmployeeID, ProjectID),
FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID),
FOREIGN KEY (ProjectID) REFERENCES Projects(ProjectID)
);

INSERT INTO Employees (FirstName, LastName, Department, Salary) VALUES
('Alice', 'Smith', 'HR', 60000.00),
('Bob', 'Johnson', 'IT', 75000.00),
('Charlie', 'Brown', 'Finance', 62000.00),
('Diana', 'Prince', 'IT', 80000.00),
('Eve', 'Davis', 'HR', 58000.00);
INSERT INTO Projects (ProjectName, Budget, StartDate, EndDate) VALUES
('Website Redesign', 150000.00, '2023-01-15', '2023-06-30'),
('Mobile App Development', 200000.00, '2023-03-01', '2023-10-31'),
('Internal Tools Upgrade', 80000.00, '2023-05-10', '2023-09-15');
INSERT INTO EmployeeProjects (EmployeeID, ProjectID, HoursWorked) VALUES
(2, 1, 160),  -- Bob Johnson (ID 2) on Website Redesign (ID 1)
(4, 1, 120),  -- Diana Prince (ID 4) on Website Redesign (ID 1)
(2, 2, 200),  -- Bob Johnson (ID 2) on Mobile App Development (ID 2)
(1, 3, 80),   -- Alice Smith (ID 1) on Internal Tools Upgrade (ID 3)
(3, 3, 100)   -- Charlie Brown (ID 3) on Internal Tools Upgrade (ID 3)
ON CONFLICT (EmployeeID, ProjectID) DO NOTHING;


-- Задание 1: DML

--Вставить двух новых сотрудников в таблицу Employees
INSERT INTO Employees (FirstName, LastName, Department, Salary) VALUES
('Petr', 'Miller', 'Marketing', 45000.00),
('Ivan', 'Wilson', 'IT', 65000.00);

--Выбрать всех сотрудников из таблицы Employees.
SELECT *
FROM Employees;
```

```
--Выбрать только FirstName и LastName сотрудников из отдела 'IT'.
SELECT FirstName, LastName, Department
FROM Employees
WHERE Department = 'IT';

--Обновить Salary 'Alice Smith' до 65000.00.
UPDATE Employees
SET Salary = 65000.00
WHERE FirstName = 'Alice' AND LastName = 'Smith';

--Удалить сотрудника, чья LastName — 'Prince'.
ALTER TABLE EmployeeProjects DROP CONSTRAINT employeeprojects_employeeid_fkey;

ALTER TABLE EmployeeProjects
        ADD CONSTRAINT employeeprojects_employeeid_fkey
        FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID) ON DELETE
CASCADE;


DELETE FROM Employees WHERE LastName = 'Prince';


--Задание 2: DDL

--Создать новую таблицу с именем Departments
CREATE TABLE Departments (
    DepartmentID SERIAL PRIMARY KEY,
    DepartmentName VARCHAR(50) UNIQUE NOT NULL,
    Location VARCHAR(50)
);

SELECT *
FROM Departments;

--Изменить таблицу Employees, добавив новый столбец с именем Email
(VARCHAR(100))
ALTER TABLE Employees
ADD COLUMN Email VARCHAR(100);

--Добавить ограничение UNIQUE к столбцу Email в таблице Employees,
предварительно заполнив любыми значениями
UPDATE Employees
SET Email = LOWER(FirstName || '.' || LastName || '@example.com')
WHERE Email IS NULL;

ALTER TABLE Employees
ADD CONSTRAINT unique_email UNIQUE (Email);

--Переименовать столбец Location в таблице Departments в OfficeLocation
ALTER TABLE Departments
RENAME COLUMN Location TO OfficeLocation;

--Задание 3: DCL
```

```sql
--Создать нового пользователя PostgreSQL (роль) с именем hr_user и простым
паролем.
CREATE USER hr_user WITH PASSWORD '163_password'

GRANT SELECT ON Employees TO hr_user;

SELECT * FROM Employees;


--Задание 4: DML/DCL

--Увеличить Salary всех сотрудников в отделе 'HR' на 10%.
SELECT * FROM Employees;
UPDATE Employees
SET Salary = Salary * 1.10
WHERE Department = 'HR';

--Обновить Department любого сотрудника с Salary выше 70000.00 на 'Senior IT'.
UPDATE Employees
SET Department = 'Senior IT'
WHERE Salary > 70000.00;


--Удалить всех сотрудников, которые не назначены ни на один проект в таблице
EmployeeProjects. Подсказка: Используйте подзапрос NOT EXISTS или LEFT JOIN
DELETE FROM Employees
WHERE NOT EXISTS (
    SELECT 1
    FROM EmployeeProjects
    WHERE EmployeeProjects.EmployeeID = Employees.EmployeeID
);

INSERT INTO Projects (ProjectName, StartDate, EndDate)
VALUES ('Новый Проект', '2023-10-27', '2023-12-31');

SELECT * FROM Projects;



CREATE OR REPLACE FUNCTION CalculateAnnualBonus(employee_id INT, Salary DECIMAL)
RETURNS DECIMAL AS $$
DECLARE
    bonus DECIMAL;
BEGIN
    -- Рассчитать бонус (10% от Salary)
    bonus := Salary * 0.10;

    -- Вернуть рассчитанный бонус
    RETURN bonus;
END;
$$ LANGUAGE plpgsql;

SELECT
    EmployeeID,
```

```
    FirstName,
    LastName,
    Salary,
    CalculateAnnualBonus(EmployeeID, Salary) AS PotentialBonus
FROM
    Employees;

CREATE OR REPLACE VIEW IT_Department_View AS
SELECT
    EmployeeID,
    FirstName,
    LastName,
    Salary
FROM
    Employees
WHERE
    Department = 'IT';

SELECT *
FROM IT_Department_View;


--Задание 6: DML (Optional)


SELECT P.ProjectName
FROM Projects P
JOIN EmployeeProjects EP ON P.ProjectID = EP.ProjectID
JOIN Employees E ON EP.EmployeeID = E.EmployeeID
WHERE E.FirstName = 'Bob' AND E.LastName = 'Johnson'
  AND EP.HoursWorked > 150;

--Увеличиваем бюджет на 10%
UPDATE Projects
SET Budget = Budget * 1.10
WHERE ProjectID IN (
    SELECT EP.ProjectID
    FROM EmployeeProjects EP
    JOIN Employees E ON EP.EmployeeID = E.EmployeeID
    WHERE E.Department = 'IT'
);

SELECT *
FROM Projects

UPDATE Projects
SET EndDate = StartDate + INTERVAL '1 year'
WHERE EndDate IS NULL;

--Вставка нового сотрудника и назначение его на проект
WITH NewEmployee AS (
    INSERT INTO Employees (FirstName, LastName, Department, Salary)
    VALUES ('Lina', 'Berd', 'IT', 45000.00)
    RETURNING EmployeeID
```

```
),
TargetProject AS (
    SELECT ProjectID
    FROM Projects
    WHERE ProjectName = 'Website Redesign'
    LIMIT 1
)
INSERT INTO EmployeeProjects (EmployeeID, ProjectID, HoursWorked)
SELECT (SELECT EmployeeID FROM NewEmployee),
       (SELECT ProjectID FROM TargetProject),
       80;

--Проверка
select Employees.EmployeeID, Employees.firstname e, Employees.LastName,
Employeeprojects.ProjectID, Employeeprojects.HoursWorked
from Employees
join EmployeeProjects on Employees.EmployeeID = employeeprojects.EmployeeID
where Employees.FirstName = 'Lina' and Employees.LastName = 'Berd';
```