Задание 3

Необходимо составить SQL-запрос, который выводит фамилии сотрудников, которые находятся в западном филиале (включая подчиненные подразделения).

Для получения фамилий сотрудников, находящихся в западном филиале, включая подчиненные подразделения, в СУБД PostgreSQL можно использовать рекурсивный SQL запрос с использованием общей таблицы выражений и расширения Itree.

**-**

WITH RECURSIVE RecursiveUnits AS (

SELECT

e.EmployeeId, e.FullName, u.UnitId

FROM Employees e INNER JOIN Units u ON e.UnitId = u.UnitId

WHERE u.Name = 'Западный филиал'

UNION ALL

SELECT

e.EmployeeId, e.FullName, u.UnitId

FROM Employees e

INNER JOIN Units u ON e.UnitId = u.UnitId

INNER JOIN RecursiveUnits ru ON u.ParentUnitId = ru.UnitId

) SELECT DISTINCT FullName FROM RecursiveUnits;

-

В этом запросе связываем таблицу Employees с таблицей Units по полю UnitId и фильтруем записи, чтобы выбрать только сотрудников из западного филиала. Затем, в рекурсивной части CTE, идёт объединение таблицы Units с предыдущей итерацией CTE, чтобы получить сотрудников из подчиненных подразделений. После используется оператор SELECT DISTINCT для выбора уникальных фамилий сотрудников из результатов рекурсивной итерации. Отвечая на вопрос, то в представленной структуре таблиц не подходит для случаев, когда один сотрудник может принадлежать к нескольким подразделениям одновременно, поскольку каждая запись сотрудника имеет только одно значение UnitId, что ограничивает принадлежность сотрудника только к одному подразделению.

Для исправления этого можно изменить структуру таблиц используя связь "многие-ко-многим" через общую таблицу, после чего в таблице Employees нам не понадобиться столбец UnitId.

**-**

CREATE TABLE Units (

UnitId int NOT NULL,

ParentUnitId int NULL,

Name nvarchar(50) NULL,

CONSTRAINT PK\_Units PRIMARY KEY (UnitId),

CONSTRAINT FK\_Units\_Units FOREIGN KEY (ParentUnitId) REFERENCES Units(UnitId)

);

CREATE TABLE Employees (

EmployeeId int NOT NULL,

FullName nvarchar(150) NULL,

CONSTRAINT PK\_Employees PRIMARY KEY (EmployeeId)

);

CREATE TABLE EmployeeUnits (

EmployeeId int NOT NULL,

UnitId int NOT NULL,

CONSTRAINT PK\_EmployeeUnits PRIMARY KEY (EmployeeId, UnitId),

CONSTRAINT FK\_EmployeeUnits\_Employees FOREIGN KEY (EmployeeId) REFERENCES Employees(EmployeeId),

CONSTRAINT FK\_EmployeeUnits\_Units FOREIGN KEY (UnitId) REFERENCES Units(UnitId)

);

-

Таблица EmployeeUnits представляет собой таблицу связей между сотрудниками и подразделениями. В этой таблице каждая строка содержит пару значений EmployeeId и UnitId, указывающих, что сотрудник с указанным идентификатором EmployeeId принадлежит к подразделению с указанным идентификатором UnitId. Использование вспомогательной таблицы EmployeeUnits может быть полезным в случаях когда есть дополнительные атрибуты, связанные с отношениями между сотрудниками и подразделениями, позволяя управлять этими атрибутами более гибко и если в данных есть сложные связи, которые не могут быть выражены простыми иерархическими путями.