**Задача 1 (БД):**

***(решение*** *прислать на адрес**nixlab@nix.ru с заголовком «ПБП: БДЗ3 - задача 1», желательно сделать до 09.03.25 для подготовки к контрольной, но дедлайн на 16.03.25****)***

**Решать нужно одну из задач: А, Б или В. Если есть желание и возможность, можно сделать более одной.**

**A. Альтернативная задача №1**

Получение состава изделия для графа произвольной глубины (заранее неизвестной), у которого есть источник (вершина, в которую ничего не входит) и стоки (вершины, из которых ничего не выходит). Примером такого графа является, например, «деревья» на рис.1 и рис.2. (Комментарий: понятно, что это совсем не дерево, но можно «расщепить» вершины таким образом, чтобы визуально (и только визуально) получить дерево).

Неформальная постановка: в БД задан граф состава изделия в виде: вершина1 = начало ребра, вершина2 = конец ребра, вес ребра = Колво вершины 2 в вершине 1 (например, в системном блоке (вершина1) есть две (вес = 2) одинаковых видеокарты (вершина2)). Необходимо вычислить состав изделия (корневой вершины) в элементарных компонентах (листьях дерева), то есть сколько нужно взять листьев, чтобы можно было получить корень.

Например, для рис. 1 мы должны получить вывод в виде:

|  |  |  |
| --- | --- | --- |
| Корень | Лист | Колво |
| 1 | 4 | 21 |
| 1 | 5 | 21 |
| 1 | 6 | 1 |

А для рис. 2:

|  |  |  |
| --- | --- | --- |
| Корень | Лист | Колво |
| 1 | 4 | 36 |
| 1 | 6 | 40 |
| 1 | 9 | 8 |
| 1 | 7 | 4 |

Реализация построения должна быть выполнена либо циклом WHILE в T-SQL (или при помощи CTE, что даже лучше). Алгоритм обоснования построения иерархии (почему все будет правильно) и подробное описание, что и как запускать, обязательно указать в письме или в файле со скриптом.

Для начала рекомендуется осознать, как решается задача для обычных деревьев (разбирали на занятии или в МДЗ7), и подумать, что нужно изменить для случая произвольного графа.

Если не будет получаться реализация с дублирующимися вершинами – обязательно сделать и прислать с недублирующимися (МДЗ7).

**Комментарий:** в этой задаче не требуется строить именно полный граф, достаточно построить нужное для задачи подмножество полного графа.

**Примеры на которых Ваш алгоритм должен уметь работать:**

**Пример 1:** Компьютерный класс (вершина 1) состоит из:

а) вычислительного кластера (вершина 2) из: 1-ого управляющего узла (вершина 6) + 16-ти вычислительных центров (вершина 3)

б) пяти рабочих компьютеров (вершина 3) (совпадающими по конфигурации с вычислительными центрами, так как вершина 3 состоит из одинакового количества вершин 4 и 5).

Сколько листовых компонент 4, 5 и 6 Вам нужно взять, чтобы Вы могли собрать такой компьютерный класс?

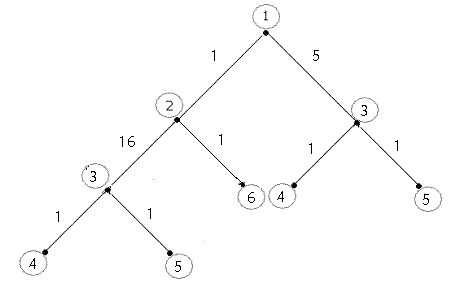


Рис. 1. Пример 1 графа (построение полной иерархии).

**Пример 2:** Для выполнения, например, ремонтных работ над 5-тью участками дороги Вам необходимо заказать материалы. Вы посчитали, что эти участки можно разбить на 2 группы по количеству и типу выполняемых работ. На трех участках нужно на каждом починить по 4 км дороги. При этом на каждом километре нужно потратить 3 условных единицы материала типа 4 для того, чтобы полностью переложить асфальт (работа на вершине 4) и 2 условных единицы материала типа 6, чтобы нанести новую разметку (работа на вершине 6 в потомке вершины 3). Другие 2 участка следует починить следующим образом: чтобы исправить по 2 фонаря освещения на каждом участке (работа на вершине 7) требуется 2 условные единицы материала типа 7. Каждый из этих 2-х участков также имеет длину 4 км. Для проведения ремонта нужно: на каждый километр потратить 1 условную единицу материала типа 9 для проведения «ямочного ремонта» (работа в вершине 9) и 2 условные единицы материала типа 6 для нанесения новой разметки (работа в вершине 6, потомке вершины 8).

Сколько условных единиц материала для каждого типа работы нужно заказать, чтобы можно было полностью выполнить работу?

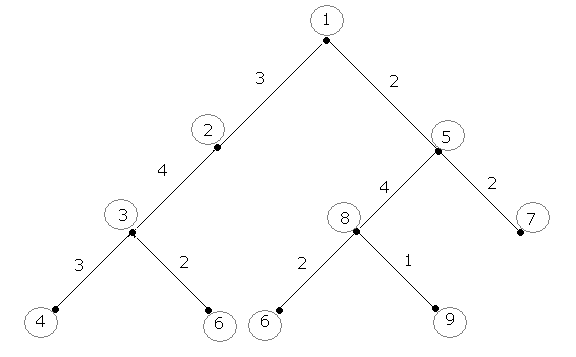


Рис. 2. Пример 2 графа (построение полной иерархии).

**Б. Альтернативная задача №2 «Задача построения критического маршрута»**

Для графа с указанием событий, работ и их длительностей научиться выводить не только длину критического пути, но и все вершины, через которые проходит граф (то есть научиться выводить критический маршрут). За пример графа можно взять граф выдачи на рознице (скорости обслуживания) или проект по написанию программного обеспечения. В случае, если критических маршрутов несколько, выводить все. Выводить критический маршрут нужно в виде столбца вершин (порядок не важен, главное вывести все вершины и отделить один маршрут от другого). Различать вершины из разных маршрутов можно любым удобным Вам способом.

Например, для задачи на рис. 3 маршрут должен вывестись в форме:

|  |
| --- |
| Вершина |
| 1 |
| 2 |
| 5 |
| 8 |
| 9 |
| 12 |
| 13 |
| 14 |
| 15 |

Данные для задачи приложены в xlsx файлах.

|  |
| --- |
|  |
| Рис.3. Диаграмма Гантта для примера разработки программного обеспечения. |

|  |
| --- |
|  |
| Рис.4. Пример сетевого графа для бизнес – процесса. Красным показан критический маршрут (обратите внимание, что в этой задаче их 2). |

**В\*. Альтернативная задача №3 «Поиск кратчайшего маршрута для карты метро».**

* Составить граф для карты метро Москвы: вершины – соседние станции, вес – длительность маршрута между соседними станциями. Подумайте, как хранить переходы между станциями/ветками. Граф можно составить, например, при помощи построения маршрутов от Яндекса (можете самостоятельно попробовать найти на сайте Яндекса все длительности переходов между соседними станциями в xml или json формате и загрузить их в базу, с xml мы научимся работать чуть позже, но можно попробовать разобраться самостоятельно уже сейчас).
* Построить граф минимальных путей для такого графа методом построения полного графа (с модификациями для выбора только минимальных путей).
* Реализовать алгоритм поиска кратчайшего маршрута между двумя заданными вершинами (взять реальный алгоритм для поиска минимального пути, например, из книги «Алгоритмы: построение и анализ» Кормен, Ривест, Лейзерсон).

Алгоритм построения полного графа кратчайших путей должен работать приемлемое время (например, не более 10-ти минут). Алгоритм поиска кратчайшего маршрута без использования полного графа должен работать достаточно быстро (точно менее 30 сек).

**Задача 2 (Аналитическая):**

*(решение прислать на адрес nixlab@nix.ru с заголовком «ПБП: БДЗ3 - задача 2»):*

Оценить, сколько в Москве должно быть магазинов типа

А) «Ашан» (крупный продуктовый гипермаркет);

Б) «Пятерочка» (маленький продуктовый магазин).

Основные предположения: магазины должны работать эффективно.

Для того, чтобы задача была принята, необходимо:

- Указать, по каким параметрам эти типы магазинов принципиально отличаются друг от друга? (чтобы ответить на этот вопрос, подумайте, как рядовой покупатель принимает решение, в какой магазин и когда он отправится).

- Указать главные показатели, при помощи которых Вы будете оценивать необходимое количество магазинов.

- Оценить, сколько магазинов каждого типа должно быть (ответом должно быть какое-то число).

\* при желании можно сравнить с реальным количеством магазинов, при этом не забыв учесть все магазины одного типа, а не только ТМ Ашан и Пятерочка.

Комментарий. По сути, это задача с ограничениями сверху и снизу: хорошо было бы построить Пятерочки (площадь и колво касс) в каждом доме, но разве это оптимально с точки зрения владельца бренда? И наоборот, может быть достаточно одной Пятерочки на весь город Долгопрудный, но подумайте, какие были бы очереди, например, сколько времени тратили бы жители на очереди и на то, чтобы добраться.

Поэтому Вам нужно определить самые главные критерии, исходя из которых, Вы будете оценивать колво магазинов каждого типа. Нам не нужна супер точность, достаточно простой оценки.

Необязательный пункт(\*): Подумайте, а как можно оценить колво компьютерных магазинов на город размером с Долгопрудный? Какие факторы Вы бы учитывали?