2008 год 2 тур

Требования

Исходная ситуация

Организация оптовой торговли «Луч» имеет головной офис в г. Москве. У организации имеется порядка 150 удаленных складов, расположенных в различных городах московской области (в г. Москве организация склада не имеет). Организация ведет учет в программе «1С:Бухгалтерия». Учет товаров ведется на счете 41 в натуральном и стоимостном выражениях, в разрезе двух видов субконто: «Товары» и «Склады».

Поступление товаров от поставщиков на склады предприятия отражается проводкой в дебет счета 41 (по субконто «склад» и «товар») и кредит счета 60. До реализации товара имеют место операции внутреннего перемещения товаров со склада на склад. Перемещение товара со склада на склад отражается в информационной базе бухгалтерского учета проводкой в дебет счета 41 (по субконто «склад-получатель») и кредит счета 41 (по субконто «склад-отправитель») по определенному товару на некоторую сумму и количество.

Ревизору ставится задача провести выездную проверку удаленных складов на предмет правильности совершения ими операций внутреннего перемещения товаров. Проверка производится по следующим правилам.

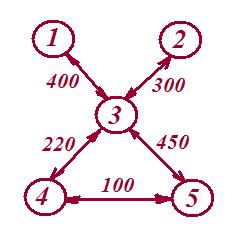
Изначально ревизор находится в головном офисе в городе Москве, где нет складов

Для того, чтобы проверить операции перемещения товаров между складом Х и складом Y ревизору требуется посетить склад X и склад Y в любой последовательности. При этом объем выполненной проверки складывается из общей стоимости перемещенных товаров между складом X и складом Y (любых наименований и в любом направлении)

Количество переездов ревизора жестко задано руководством организации

Ревизор может посещать города в любом порядке (1,3,5 или 3,1,5 или 5,1,3 и т.п.)

Ревизор заканчивает серию поездок в любом городе, где имеется удаленный склад



Например, если в ситуации, изображенной на рисунке, когда имеется 5 складов в пяти городах, а ревизору дано право на три переезда, то возможны варианты:

1, 2, 3 = 400 +300 = 700  
1, 3, 5 = 400 + 450 = 850  
3, 4, 5 = 220 + 450 + 100 = 770  
2, 3, 4 = 300 + 220 = 520   
И т.д.

Как видим, все они отличаются суммарной стоимостью проверенных операций.

Требуется

Необходимо разработать внешний отчет, который выдает список складов, которые должен посетить ревизор за заданное количество переездов, чтобы общий объем выполненной проверки в стоимостном выражении был бы максимальным.

Входные данные

Информационная база в каркасной конфигурации с проводками, отражающими операции внутреннего перемещения товаров между складами\*. А также дополнительная информация, вводимая в диалоге отчета: период, за который анализируются операции внутреннего перемещения ( по умолчанию – 1 квартал 2008 г.) и количество переездов ревизора (по умолчанию – 4).

Выходная информация

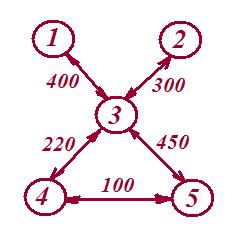
Отчет, представленный в следующей форме.

|  |
| --- |
| Оптимальный план поездок: |
| Склад01 (Химки) |
| Склад03 (Мытищи) |
| Склад05 (Дмитров) |
| Итоговая сумма = 850.00 |

* Если в базе встретятся проводки внутреннего перемещения со склада-отправителя на склад-получатель, в которых по дебету и кредиту указаны разные товары, то такие проводки считаются недействительными и при решении задачи должны игнорироваться. Если встретятся проводки внутреннего перемещения со склада-отправителя на склад-получатель, в которых по дебету и кредиту указаны одинаковые склады, то такие проводки считаются недействительными и при решении задачи должны игнорироваться.
* Список, включает наименования складов

Решение

Решение задачи сводится к нахождению самого дорогостоящего пути по графу. Рассмотрим возможные варианты обхода графа (с ограничением в 3 шага), данного в примере:



|  |  |  |
| --- | --- | --- |
| Путь | Расчет стоимости | Стоимость |
| 1,3,2 | 400+300 | 700 |
| 1,3,4 | 400+220 | 620 |
| 1,3,5 | 400+450 | 850 |
| 2,3,1 | 300+400 | 700 |
| 2,3,4 | 300+220 | 520 |
| 2,3,5 | 300+450 | 750 |
| 3,1 | 400 | 400 |
| 3,2 | 300 | 300 |
| 3,4,5 | 220+100 | 320 |
| 3,5,4 | 450+100 | 550 |
| 4,3,1 | 220+400 | 620 |
| 4,3,2 | 220+300 | 520 |
| 4,3,5 | 220+450 | 670 |
| 4,5,3 | 100+450 | 550 |
| 5,3,1 | 450+400 | 850 |
| 5,3,2 | 450+300 | 750 |
| 5,3,4 | 450+220 | 670 |
| 5,4,3 | 100+220 | 320 |

По результатам, данным в таблице, видно, что путь 1,3,5 (5,3,1) – самый дорогостоящий, а значит это и есть искомый путь.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьДанные = Макет.ПолучитьОбласть("Данные");

ОбластьИтоги = Макет.ПолучитьОбласть("Итоги");

Склады = Новый Соответствие;

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийОбороты.Субконто2,

| ЖурналОперацийОбороты.КорСубконто2,

| СУММА(ЖурналОперацийОбороты.СуммаОборотДт) КАК СуммаОборот

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, , Счет = &Счет, , , , ) КАК ЖурналОперацийОбороты

|ГДЕ

| ЖурналОперацийОбороты.Субконто1 = ЖурналОперацийОбороты.КорСубконто1

| И ЖурналОперацийОбороты.Субконто2 <> ЖурналОперацийОбороты.КорСубконто2

|

|СГРУППИРОВАТЬ ПО

| ЖурналОперацийОбороты.Субконто2,

| ЖурналОперацийОбороты.КорСубконто2";

Запрос.УстановитьПараметр("ДатаНач",ДатаНачала);

Запрос.УстановитьПараметр("ДатаКон",ДатаКонца);

Запрос.УстановитьПараметр("Счет",ПланыСчетов.Счета.Товары);

Выборка = Запрос.Выполнить().Выбрать();

Пока Выборка.Следующий() Цикл

Склад1 = Выборка.Субконто2;

Склад2 = Выборка.КорСубконто2;

Связи1 = Склады.Получить(Склад1);

Связи2 = Склады.Получить(Склад2);

Если Связи1 = Неопределено Тогда

Связи1 = Новый Структура;

Связи1.Вставить("Посещено",Ложь);

Связи1.Вставить("Связи",Новый Массив);

Склады.Вставить(Склад1,Связи1);

КонецЕсли;

Если Связи2 = Неопределено Тогда

Связи2 = Новый Структура;

Связи2.Вставить("Посещено",Ложь);

Связи2.Вставить("Связи",Новый Массив);

Склады.Вставить(Склад2,Связи2);

КонецЕсли;

Связь1 = Неопределено;

Связь2 = Неопределено;

Для Каждого Связь Из Связи1.Связи Цикл

Если Связь.Склад = Склад2 Тогда

Связь1 = Связь;

Прервать;

КонецЕсли;

КонецЦикла;

Для Каждого Связь Из Связи2.Связи Цикл

Если Связь.Склад = Склад1 Тогда

Связь2 = Связь;

Прервать;

КонецЕсли;

КонецЦикла;

Если Связь1 = Неопределено Тогда

Связь1 = Новый Структура;

Связь1.Вставить("Склад",Склад2);

Связь1.Вставить("Сумма",0);

Связи1.Связи.Добавить(Связь1);

КонецЕсли;

Если Связь2 = Неопределено Тогда

Связь2 = Новый Структура;

Связь2.Вставить("Склад",Склад1);

Связь2.Вставить("Сумма",0);

Связи2.Связи.Добавить(Связь2);

КонецЕсли;

Связь1.Сумма = Связь1.Сумма + Выборка.СуммаОборот;

Связь2.Сумма = Связь2.Сумма + Выборка.СуммаОборот;

КонецЦикла;

Если КоличествоГородов > Склады.Количество() Тогда

КоличествоГородов = Склады.Количество();

КонецЕсли;

Результат = НайтиПуть(Склады,Неопределено,КоличествоГородов);

ТабДок.Очистить();

ТабДок.Вывести(ОбластьЗаголовок);

Для Каждого Склад Из Результат.Склады Цикл

ОбластьДанные.Параметры[0] = Склад.Наименование;

ТабДок.Вывести(ОбластьДанные);

КонецЦикла;

ОбластьИтоги.Параметры[0] = Результат.Сумма;

ТабДок.Вывести(ОбластьИтоги);

КонецПроцедуры

Функция НайтиПуть(Склады,Корень,ОсталосьШагов)

ЛучшийПуть = Неопределено;

ЛучшийВес = 0;

ЛучшийШаг = Неопределено;

Если ОсталосьШагов = 0 Тогда

Возврат Неопределено;

КонецЕсли;

Если Корень = Неопределено Тогда

Для Каждого Склад Из Склады Цикл

Склад.Значение.Посещено = Истина;

Путь = НайтиПуть(Склады,Склад.Ключ,ОсталосьШагов);

Склад.Значение.Посещено = Ложь;

Если ЛучшийПуть = Неопределено Тогда

ЛучшийПуть = Путь;

ЛучшийПуть.Склады.Вставить(0,Склад.Ключ);

ИначеЕсли Путь <> Неопределено И ЛучшийПуть.Сумма < Путь.Сумма Тогда

ЛучшийПуть = Путь;

ЛучшийПуть.Склады.Вставить(0,Склад.Ключ);

КонецЕсли;

КонецЦикла;

Возврат ЛучшийПуть;

Иначе

СкладСвязи = Склады.Получить(Корень);

Для Каждого Связь из СкладСвязи.Связи Цикл

Склад = Склады.Получить(Связь.Склад);

Если Склад.Посещено Тогда

Продолжить;

КонецЕсли;

Склад.Посещено = Истина;

Путь = НайтиПуть(Склады,Связь.Склад,ОсталосьШагов - 1);

Склад.Посещено = Ложь;

Если ЛучшийПуть = Неопределено Тогда

ЛучшийПуть = Путь;

ЛучшийВес = Связь.Сумма;

ЛучшийШаг = Связь.Склад;

ИначеЕсли Путь <> Неопределено И ЛучшийПуть.Сумма < Путь.Сумма Тогда

ЛучшийПуть = Путь;

ЛучшийВес = Связь.Сумма;

ЛучшийШаг = Связь.Склад;

КонецЕсли;

КонецЦикла;

КонецЕсли;

СписокСкладов = Новый Массив;

СуммаОпераций = 0;

Если ЛучшийПуть <> Неопределено Тогда

СписокСкладов.Добавить(ЛучшийШаг);

СуммаОпераций = ЛучшийВес;

Для Каждого Шаг Из ЛучшийПуть.Склады Цикл

СписокСкладов.Добавить(Шаг);

КонецЦикла;

СуммаОпераций = СуммаОпераций + ЛучшийПуть.Сумма;

КонецЕсли;

Путь = Новый Структура;

Путь.Вставить("Склады",СписокСкладов);

Путь.Вставить("Сумма",СуммаОпераций);

Возврат Путь;

КонецФункции

Процедура ПриОткрытии()

КоличествоГородов = 4;

ДатаНачала = '20080101';

ДатаКонца = '20080331';

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьДанные = Макет.ПолучитьОбласть("Данные");

ОбластьИтоги = Макет.ПолучитьОбласть("Итоги");

Склады = Новый Соответствие;

В данном блоке идет получение табличного документа, в который будет производиться вывод результата и получение макета, с областями, по которым будет строиться таблица результата. Создается соотвествие Склады, в котором будут храниться соответствия Склад – Структура(Посещен,Связи), где поле Посещен – требуется для обхода графа (отмечает, что на данном складе уже были), а поле Связи – массив с записями Структура(Склад,Сумма) – которые обозначают склады с которыми связан текущий склад и сумму связи.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийОбороты.Субконто2,

| ЖурналОперацийОбороты.КорСубконто2,

| СУММА(ЖурналОперацийОбороты.СуммаОборотДт) КАК СуммаОборот

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, , Счет = &Счет, , , , ) КАК ЖурналОперацийОбороты

|ГДЕ

| ЖурналОперацийОбороты.Субконто1 = ЖурналОперацийОбороты.КорСубконто1

| И ЖурналОперацийОбороты.Субконто2 <> ЖурналОперацийОбороты.КорСубконто2

|

|СГРУППИРОВАТЬ ПО

| ЖурналОперацийОбороты.Субконто2,

| ЖурналОперацийОбороты.КорСубконто2";

Запрос.УстановитьПараметр("ДатаНач",ДатаНачала);

Запрос.УстановитьПараметр("ДатаКон",ДатаКонца);

Запрос.УстановитьПараметр("Счет",ПланыСчетов.Счета.Товары);

В данном блоке создается запрос к базе данных, который получит пары Склад1 – Склад2 с суммой связи, на указанный период времени, причем товар связи обязательно один и тот же, а склады обязательно разные.

Выборка = Запрос.Выполнить().Выбрать();

Пока Выборка.Следующий() Цикл

Склад1 = Выборка.Субконто2;

Склад2 = Выборка.КорСубконто2;

Связи1 = Склады.Получить(Склад1);

Связи2 = Склады.Получить(Склад2);

Если Связи1 = Неопределено Тогда

Связи1 = Новый Структура;

Связи1.Вставить("Посещено",Ложь);

Связи1.Вставить("Связи",Новый Массив);

Склады.Вставить(Склад1,Связи1);

КонецЕсли;

Если Связи2 = Неопределено Тогда

Связи2 = Новый Структура;

Связи2.Вставить("Посещено",Ложь);

Связи2.Вставить("Связи",Новый Массив);

Склады.Вставить(Склад2,Связи2);

КонецЕсли;

Связь1 = Неопределено;

Связь2 = Неопределено;

Для Каждого Связь Из Связи1.Связи Цикл

Если Связь.Склад = Склад2 Тогда

Связь1 = Связь;

Прервать;

КонецЕсли;

КонецЦикла;

Для Каждого Связь Из Связи2.Связи Цикл

Если Связь.Склад = Склад1 Тогда

Связь2 = Связь;

Прервать;

КонецЕсли;

КонецЦикла;

Если Связь1 = Неопределено Тогда

Связь1 = Новый Структура;

Связь1.Вставить("Склад",Склад2);

Связь1.Вставить("Сумма",0);

Связи1.Связи.Добавить(Связь1);

КонецЕсли;

Если Связь2 = Неопределено Тогда

Связь2 = Новый Структура;

Связь2.Вставить("Склад",Склад1);

Связь2.Вставить("Сумма",0);

Связи2.Связи.Добавить(Связь2);

КонецЕсли;

Связь1.Сумма = Связь1.Сумма + Выборка.СуммаОборот;

Связь2.Сумма = Связь2.Сумма + Выборка.СуммаОборот;

КонецЦикла;

В данном блоке идет заполнение соответствия Склады результатами запроса. Из выборки получается пара складов, каждый склад ищется в соответствии Склады, если склад не найден, то он добавляется в соответствие с начальными данными (Посещено – Ложь, Связи – пустой массив). Далее в связях обоих складов ищутся записи о связи этих складов, если связь не найдена, то она добавляется с начальными данными (Склад – связуемый склад, Сумма – 0), после нахождения (создания) связи – её сумма увеличивается на сумму полученную в выборке.

Если КоличествоГородов > Склады.Количество() Тогда

КоличествоГородов = Склады.Количество();

КонецЕсли;

Результат = НайтиПуть(Склады,Неопределено,КоличествоГородов);

В данном блоке проверяется, чтобы количество городов для поиска пути не было больше количества складов. Далее ищется путь, через вызов рекурсивной функции НайтиПуть.

ТабДок.Очистить();

ТабДок.Вывести(ОбластьЗаголовок);

Для Каждого Склад Из Результат.Склады Цикл

ОбластьДанные.Параметры[0] = Склад.Наименование;

ТабДок.Вывести(ОбластьДанные);

КонецЦикла;

ОбластьИтоги.Параметры[0] = Результат.Сумма;

ТабДок.Вывести(ОбластьИтоги);

В данном блоке табличный документ очищается от старых данных и выводится результат: заголовок, список наименований складов и сумма пути.

Функция НайтиПуть(Склады,Корень,ОсталосьШагов)

ЛучшийПуть = Неопределено;

ЛучшийВес = 0;

ЛучшийШаг = Неопределено;

Если ОсталосьШагов = 0 Тогда

Возврат Неопределено;

КонецЕсли;

Если Корень = Неопределено Тогда

Для Каждого Склад Из Склады Цикл

Склад.Значение.Посещено = Истина;

Путь = НайтиПуть(Склады,Склад.Ключ,ОсталосьШагов);

Склад.Значение.Посещено = Ложь;

Если ЛучшийПуть = Неопределено Тогда

ЛучшийПуть = Путь;

ЛучшийПуть.Склады.Вставить(0,Склад.Ключ);

ИначеЕсли Путь <> Неопределено И ЛучшийПуть.Сумма < Путь.Сумма Тогда

ЛучшийПуть = Путь;

ЛучшийПуть.Склады.Вставить(0,Склад.Ключ);

КонецЕсли;

КонецЦикла;

Возврат ЛучшийПуть;

Иначе

СкладСвязи = Склады.Получить(Корень);

Для Каждого Связь из СкладСвязи.Связи Цикл

Склад = Склады.Получить(Связь.Склад);

Если Склад.Посещено Тогда

Продолжить;

КонецЕсли;

Склад.Посещено = Истина;

Путь = НайтиПуть(Склады,Связь.Склад,ОсталосьШагов - 1);

Склад.Посещено = Ложь;

Если ЛучшийПуть = Неопределено Тогда

ЛучшийПуть = Путь;

ЛучшийВес = Связь.Сумма;

ЛучшийШаг = Связь.Склад;

ИначеЕсли Путь <> Неопределено И ЛучшийПуть.Сумма < Путь.Сумма Тогда

ЛучшийПуть = Путь;

ЛучшийВес = Связь.Сумма;

ЛучшийШаг = Связь.Склад;

КонецЕсли;

КонецЦикла;

КонецЕсли;

СписокСкладов = Новый Массив;

СуммаОпераций = 0;

Если ЛучшийПуть <> Неопределено Тогда

СписокСкладов.Добавить(ЛучшийШаг);

СуммаОпераций = ЛучшийВес;

Для Каждого Шаг Из ЛучшийПуть.Склады Цикл

СписокСкладов.Добавить(Шаг);

КонецЦикла;

СуммаОпераций = СуммаОпераций + ЛучшийПуть.Сумма;

КонецЕсли;

Путь = Новый Структура;

Путь.Вставить("Склады",СписокСкладов);

Путь.Вставить("Сумма",СуммаОпераций);

Возврат Путь;

КонецФункции

Данная функция производит всё решение задачи – ищет путь по графу. Она рекурсивна (вызывает сама себя). Первый параметр – граф, второй параметр – вершина, в которой требуется начать обход (если Неопределено, то пробуются все вершины графа), третий параметр – максимальное количество шагов.

ЛучшийПуть = Неопределено;

ЛучшийВес = 0;

ЛучшийШаг = Неопределено;

В данном блоке инициализируются переменные необходимые для поиска лучшего варианта прохождения пути, ЛучшийПуть – хранит результат лучшего варианта, ЛучшийВес – сумма связи, ведущей к лучшему варианту, ЛучшийШаг – вершина, ведущая к лучшему варианту.

Если ОсталосьШагов = 0 Тогда

Возврат Неопределено;

КонецЕсли;

В данном блоке проверяется, есть ли ещё шаги. Если шагов не осталось – возвращается Неопределено (означает дальше путь не идет).

Если Корень = Неопределено Тогда

Для Каждого Склад Из Склады Цикл

Склад.Значение.Посещено = Истина;

Путь = НайтиПуть(Склады,Склад.Ключ,ОсталосьШагов);

Склад.Значение.Посещено = Ложь;

Если ЛучшийПуть = Неопределено Тогда

ЛучшийПуть = Путь;

ЛучшийПуть.Склады.Вставить(0,Склад.Ключ);

ИначеЕсли Путь <> Неопределено И ЛучшийПуть.Сумма < Путь.Сумма Тогда

ЛучшийПуть = Путь;

ЛучшийПуть.Склады.Вставить(0,Склад.Ключ);

КонецЕсли;

КонецЦикла;

Возврат ЛучшийПуть;

Иначе

В данном блоке обрабатывается вариант, если Корень не задан, тогда функция НайтиПуть вызывается для каждой вершины графа (предварительно помечая её посещенной, и после попытки поиска пути убирая отметку, чтобы граф пришел к первоначальному виду). Из полученных результатов выбирается лучший, в начало списка складов добавляется склад, принятый за начало пути. После обхода графа лучший результат возвращается, как результат выполнения функции.

Иначе

СкладСвязи = Склады.Получить(Корень);

Для Каждого Связь из СкладСвязи.Связи Цикл

Склад = Склады.Получить(Связь.Склад);

Если Склад.Посещено Тогда

Продолжить;

КонецЕсли;

Склад.Посещено = Истина;

Путь = НайтиПуть(Склады,Связь.Склад,ОсталосьШагов - 1);

Склад.Посещено = Ложь;

Если ЛучшийПуть = Неопределено Тогда

ЛучшийПуть = Путь;

ЛучшийВес = Связь.Сумма;

ЛучшийШаг = Связь.Склад;

ИначеЕсли Путь <> Неопределено И ЛучшийПуть.Сумма < Путь.Сумма Тогда

ЛучшийПуть = Путь;

ЛучшийВес = Связь.Сумма;

ЛучшийШаг = Связь.Склад;

КонецЕсли;

КонецЦикла;

КонецЕсли;

В данном блоке обрабатывается вариант, когда Корень задан. Тогда получаются все связи этого корня и, если склад связи не посещен, то он отмечается посещенным и путь ищется уже начиная с данного склада. После нахождения пути отметка о посещении убирается. Из всех вариантов запоминается лучший путь, вместе с весом и вершиной, которые привели к лучшему результату.

СписокСкладов = Новый Массив;

СуммаОпераций = 0;

Если ЛучшийПуть <> Неопределено Тогда

СписокСкладов.Добавить(ЛучшийШаг);

СуммаОпераций = ЛучшийВес;

Для Каждого Шаг Из ЛучшийПуть.Склады Цикл

СписокСкладов.Добавить(Шаг);

КонецЦикла;

СуммаОпераций = СуммаОпераций + ЛучшийПуть.Сумма;

КонецЕсли;

В данном блоке заполняется список складов для результата, в виде: вершина, приведшая к лучшему результату + все вершины, полученные из лучшего результата. А сумма пути высчитывается как вес связи, приведшей к лучшему результату, плюс сумма лучшего результата.

Путь = Новый Структура;

Путь.Вставить("Склады",СписокСкладов);

Путь.Вставить("Сумма",СуммаОпераций);

Возврат Путь;

В данном блоке создается структура результата с полями Склады – список складов лучшего пути и Сумма – сумма переходов по лучшему пути. После создания структура возвращается как результат функции.

Процедура ПриОткрытии()

КоличествоГородов = 4;

ДатаНачала = '20080101';

ДатаКонца = '20080331';

КонецПроцедуры

Данная процедура устанавливает значения по умолчанию для полей формы. Количество городов, которые может посетить ревизор – 4, дата начала периода – 01 января 2008, а дата конца периода 31 марта 2008.

Сформированный отчет

