**Решение олимпиадных задач**

**Автор: Михайлов Алексей**

[2013 год 2 тур 5](#_Toc381208819)

[Требования 5](#_Toc381208820)

[Решение 6](#_Toc381208821)

[Реализация 7](#_Toc381208822)

[Сформированный отчет 14](#_Toc381208823)

[2013 год 1 тур 16](#_Toc381208824)

[Требования 16](#_Toc381208825)

[Решение 17](#_Toc381208826)

[Реализация 18](#_Toc381208827)

[Сформированный отчет 29](#_Toc381208828)

[2012 год 2 тур 31](#_Toc381208829)

[Требования 31](#_Toc381208830)

[Решение 33](#_Toc381208831)

[Реализация 35](#_Toc381208832)

[Сформированный отчет 56](#_Toc381208833)

[2012 год 1 тур 57](#_Toc381208834)

[Требования 57](#_Toc381208835)

[Решение 58](#_Toc381208836)

[Реализация 61](#_Toc381208837)

[Сформированный отчет 73](#_Toc381208838)

[2011 год 2 тур 75](#_Toc381208839)

[Требования 75](#_Toc381208840)

[Решение 79](#_Toc381208841)

[Реализация 81](#_Toc381208842)

[Сформированный отчет 95](#_Toc381208843)

[2011 год 1 тур 97](#_Toc381208844)

[Требования 97](#_Toc381208845)

[Решение 98](#_Toc381208846)

[Реализация 99](#_Toc381208847)

[Сформированный отчет 102](#_Toc381208848)

[2010 год 2 тур 104](#_Toc381208849)

[Требования 104](#_Toc381208850)

[Решение 106](#_Toc381208851)

[Реализация 108](#_Toc381208852)

[Сформированный отчет 114](#_Toc381208853)

[2010 год 1 тур 116](#_Toc381208854)

[Требования 116](#_Toc381208855)

[Решение 117](#_Toc381208856)

[Реализация 118](#_Toc381208857)

[Сформированный отчет 124](#_Toc381208858)

[2009 год 2 тур 126](#_Toc381208859)

[Требования 126](#_Toc381208860)

[Решение 127](#_Toc381208861)

[Реализация 127](#_Toc381208862)

[Сформированный отчет 131](#_Toc381208863)

[2009 год 1 тур 132](#_Toc381208864)

[Требования 132](#_Toc381208865)

[Решение 133](#_Toc381208866)

[Реализация 136](#_Toc381208867)

[Сформированный отчет 140](#_Toc381208868)

[2008 год 2 тур 142](#_Toc381208869)

[Требования 142](#_Toc381208870)

[Решение 144](#_Toc381208871)

[Реализация 146](#_Toc381208872)

[Сформированный отчет 154](#_Toc381208873)

[2008 год 1 тур 155](#_Toc381208874)

[Требование 155](#_Toc381208875)

[Решение 157](#_Toc381208876)

[Реализация 158](#_Toc381208877)

[Сформированный отчет 174](#_Toc381208878)

[2007 год 2 тур 175](#_Toc381208879)

[Требование 175](#_Toc381208880)

[Решение 177](#_Toc381208881)

[Реализация 178](#_Toc381208882)

[Сформированный отчет 193](#_Toc381208883)

[2004 год 2 тур 194](#_Toc381208884)

[Требование 194](#_Toc381208885)

[Решение 196](#_Toc381208886)

[Реализация 198](#_Toc381208887)

[Сформированный отчет 205](#_Toc381208888)

[2002 год 2 тур 206](#_Toc381208889)

[Требование 206](#_Toc381208890)

[Решение 208](#_Toc381208891)

[Реализация 209](#_Toc381208892)

[Сформированный отчет 217](#_Toc381208893)

2013 год 2 тур

Требования

Пятая задача Кипукамайока

Исходная ситуация

В информационной базе «1С:Бухгалтерии» имеются проводки за некоторый период ведения учета. Каждая проводка содержит корреспонденцию двух счетов и сумму. Счета могут иметь субсчета первого порядка в виде: <код синтетического счета>[.<код субсчета>] . Проводки с отрицательными суммами («сторно») отсутствуют.

Требуется

Разработать внешний отчет, в котором предлагается вариант разбиения всех счетов, представленных в проводках, на минимальное количество групп, каждая из которых содержит синтетические счета, которые не корреспондируют друг с другом ни в одной из проводок, имеющихся в информационной  базе. При этом проводки, в которых корреспондируют одинаковые синтетические счета, а также проводки с нулевыми суммами учитывать не надо. Результат представить в виде отчета, форма которого приведена в примере.

Пример

Исходные данные (проводки в ИБ)

|  |  |  |  |
| --- | --- | --- | --- |
| Дата | Счет дебета | Счет кредита | Сумма |
| 03.03.13 | 62.1 | 90.1 | 31 000.00 |
| 10.03.13 | 76.9 | 51 | 17 000.00 |
| 12.03.13 | 26 | 51 | 150.00 |
| 14.03.13 | 60.2 | 51 | 22 000.00 |
| 17.03.13 | 26 | 60.1 | 1 200.00 |
| 19.03.13 | 76.5 | 90.1 | 41 000.00 |
| 19.03.13 | 62.1 | 76.5 | 41 000.00 |
| 23.03.13 | 51 | 62.1 | 72 000.00 |
| 24.03.13 | 60.1 | 60.2 | 22 000.00 |
| 24.03.13 | 20 | 60.1 | 22 000.00 |
| 31.03.13 | 20 | 26 | 1 350.00 |
| 31.03.13 | 90.2 | 20 | 23 350.00 |

Результат (отчет)

|  |  |
| --- | --- |
| Номер группы | Счета |
| 1 | 51, 20 |
| 2 | 60, 76 |
| 3 | 26, 62 |
| 4 | 90 |

 Исходные данные

Исходные данные представлены в ИБ каркасной конфигурации.

В каркасную конфигурацию включены:

* План счетов бухгалтерского учета, содержащий несколько счетов.
* Регистр бухгалтерии, с проводками, соответствующими примеру.
* Документ для ввода проводок.

Решение

Для разбиения счетов на группы можно использовать следующий алгоритм: обходим все синтетические счета, участвующие в корреспонденции, для каждого, ещё не сгруппированного, собирается группа других счетов (так же не сгруппированных), проверяя на каждой итерации есть ли корреспонденция между каждым из счетов. При добавлении счета в группу (при условии, что он не корреспондирует ни с одним из группы) он помечается сгруппированным и обход продолжается, пока не будут сгруппированы все счета. Для наглядности рассмотрим пример:

Проводки:

|  |  |
| --- | --- |
| Счет дебет | Счет кредит |
| 10 | 11 |
| 20 | 11 |
| 10 | 21 |
| 21 | 20 |
| 11 | 30 |

Процесс работы алгоритма:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Итерация | Начальные данные | | Составление группы | |
| Счет | Сгруппирован? | Итерация | Группа |
| 1 | 10 | Нет | 1 | 10 |
| 11 | Нет | 2 | 10,20 |
| 20 | Нет | 3 | 10,20,30 |
| 21 | Нет |  | |
| 30 | Нет |
| 2 | 10 | Да | 1 | 11 |
| 11 | Нет | 2 | 11,21 |
| 20 | Да |  | |
| 21 | Нет |
| 30 | Да |
| 3 | 10 | Да | Разбиение закончено | |
| 11 | Да |
| 20 | Да |
| 21 | Да |
| 30 | Да |

Для реализации алгоритма требуются следующие данные: список счетов участвующих в корреспонденции и списки корреспондирующих между собой счетов.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

табДок = Новый ТабличныйДокумент;

СформироватьОтчет(табДок);

табДок.Показать("Результат");

КонецПроцедуры

&НаСервере

Процедура СформироватьОтчет(табДок)

// отладка

ВремяНачала = ТекущаяУниверсальнаяДата();

Макет = ОтчетОбъект.ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьДанные = Макет.ПолучитьОбласть("Данные");

табДок.Вывести(ОбластьЗаголовок);

// массив счетов, участвующих в проводках

СчетаПроводок = Новый Массив();

// массив соответствий корреспонденции

ПроводкиСоответствие = Новый Массив();

// массив "сгруппирован или нет"

СчетСгруппирован = Новый Массив();

// получаем счета имеющиеся в проводках

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ

| ПланСчетов.Код КАК Код,

| ПланСчетов.Родитель.Код

|ИЗ

| ПланСчетов.ПланСчетов КАК ПланСчетов,

| (ВЫБРАТЬ

| РегистрБухгалтерии.СчетДт.Код КАК СчетДтКод,

| РегистрБухгалтерии.СчетКт.Код КАК СчетКтКод

| ИЗ

| РегистрБухгалтерии.РегистрБухгалтерии КАК РегистрБухгалтерии) КАК ВложенныйЗапрос

|ГДЕ

| ПланСчетов.Код В (ВложенныйЗапрос.СчетДтКод, ВложенныйЗапрос.СчетКтКод)

|

|УПОРЯДОЧИТЬ ПО

| Код";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

// получаем код синтетического счета

СинтетическийСчет = Выборка.Код;

Если Выборка.РодительКод <> null Тогда

СинтетическийСчет = Выборка.РодительКод;

КонецЕсли;

КодСинтетическогоСчета = Число(СтрЗаменить(СинтетическийСчет,".",""));

// проверяем не добавлен ли уже счет в массив

Если СчетаПроводок.Найти(КодСинтетическогоСчета) = Неопределено Тогда

СчетаПроводок.Добавить(КодСинтетическогоСчета);

// создаем массив для корреспонденции

МассивКорреспонденции = Новый Массив();

ПроводкиСоответствие.Добавить(МассивКорреспонденции);

// отладка

//Сообщить("Добавлен счет: " + КодСинтетическогоСчета);

СчетСгруппирован.Добавить(Ложь);

КонецЕсли;

КонецЦикла;

// считываем корреспонденцию

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| РегистрБухгалтерии.СчетДт.Код,

| РегистрБухгалтерии.СчетДт.Родитель.Код,

| РегистрБухгалтерии.СчетКт.Код,

| РегистрБухгалтерии.СчетКт.Родитель.Код,

| РегистрБухгалтерии.Сумма

|ИЗ

| РегистрБухгалтерии.РегистрБухгалтерии КАК РегистрБухгалтерии

|ГДЕ

| РегистрБухгалтерии.Сумма <> 0";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

// заполняем массив счетов, учавствующих в проводках

Пока Выборка.Следующий() Цикл

// СчетДТ

СинтетическийСчетДТ = Выборка.СчетДтКод;

Если Выборка.СчетДтРодительКод <> null Тогда

СинтетическийСчетДТ = Выборка.СчетДтРодительКод;

КонецЕсли;

КодСинтетическогоСчетаДТ = Число(СтрЗаменить(СинтетическийСчетДТ,".",""));

// СчетКТ

СинтетическийСчетКТ = Выборка.СчетКтКод;

Если Выборка.СчетКтРодительКод <> null Тогда

СинтетическийСчетКТ = Выборка.СчетКтРодительКод;

КонецЕсли;

КодСинтетическогоСчетаКТ = Число(СтрЗаменить(СинтетическийСчетКТ,".",""));

// отладка

//Сообщить("синтетические счета: " + КодСинтетическогоСчетаДТ + "," + КодСинтетическогоСчетаКТ);

// проверяем разные ли счета

Если КодСинтетическогоСчетаДТ <> КодСинтетическогоСчетаКТ Тогда

// ищем счета

СчетДТ = СчетаПроводок.Найти(КодСинтетическогоСчетаДТ);

СчетКТ = СчетаПроводок.Найти(КодСинтетическогоСчетаКТ);

// получаем массивы корреспонденции

МассивКорреспонденцииДТ = ПроводкиСоответствие.Получить(СчетДТ);

МассивКорреспонденцииКТ = ПроводкиСоответствие.Получить(СчетКТ);

// проверяем не было ли между ними корреспонденции

Если МассивКорреспонденцииДТ.Найти(КодСинтетическогоСчетаКТ) = Неопределено Тогда

// добавляем в массивы данные

МассивКорреспонденцииДТ.Добавить(КодСинтетическогоСчетаКТ);

МассивКорреспонденцииКТ.Добавить(КодСинтетическогоСчетаДТ);

//Сообщить("Добавлена корреспонденция: " + КодСинтетическогоСчетаДТ + "," + КодСинтетическогоСчетаКТ);

КонецЕсли;

КонецЕсли;

КонецЦикла;

НомерГруппы = 1;

КолСчетов = СчетаПроводок.Количество();

Сгруппированно = 0;

// Обходим получившуюся корреспонденцию, отмечая сгруппированные счета

Для Индекс = 0 по СчетаПроводок.ВГраница() Цикл

// проверяем не сгруппирован ли уже счет

Если СчетСгруппирован[Индекс] = Истина Тогда

Продолжить;

КонецЕсли;

// получаем данные

Счет = СчетаПроводок.Получить(Индекс);

// устанавливаем данные перед обходом

СчетаВГруппе = Новый Массив();

СчетаВГруппе.Добавить(Индекс);

СчетСгруппирован[Индекс] = Истина;

Сгруппированно = Сгруппированно + 1;

// отладка

//Сообщить("Начало группы: " + Счет);

// обходим счета, в поиске отсутствующих в корреспонденции

Для Индекс2 = 0 по СчетаПроводок.ВГраница() Цикл

// проверяем не сгруппирован ли уже счет

Если СчетСгруппирован[Индекс2] = Истина Тогда

Продолжить;

КонецЕсли;

// получаем данные

СчетКор = СчетаПроводок.Получить(Индекс2);

// отладка

//Сообщить("Проверяем корреспонденцию группы с " + СчетКор);

// обходим группу

НеКорреспондирует = 0;

Для Каждого СчетГруппы Из СчетаВГруппе Цикл

// получаем его массив корреспонденции

МассивКорреспонденции = ПроводкиСоответствие.Получить(СчетГруппы);

// проверяем наличие корреспонденции

Если МассивКорреспонденции.Найти(СчетКор) = Неопределено Тогда

// запоминаем что не корреспондирует

НеКорреспондирует = НеКорреспондирует + 1;

КонецЕсли;

КонецЦикла;

// проверяем, со всеми ли не корреспондирует

Если НеКорреспондирует = СчетаВГруппе.Количество() Тогда

// отладка

//Сообщить("Кореспонденция не найдена");

// добавляем в группу

СчетаВГруппе.Добавить(Индекс2);

// помечаем сгруппированным

СчетСгруппирован[Индекс2] = Истина;

Сгруппированно = Сгруппированно + 1;

КонецЕсли;

КонецЦикла;

ГруппыСтр = Строка(Счет);

Для Индекс3 = 1 по СчетаВГруппе.ВГраница() Цикл

ИндексСчета = СчетаВГруппе[Индекс3];

ГруппыСтр = ГруппыСтр + ", " + СчетаПроводок.Получить(ИндексСчета);

КонецЦикла;

// выводим в табличный документ

ОбластьДанные.Параметры.НомерГруппы = НомерГруппы;

ОбластьДанные.Параметры.Счета = ГруппыСтр;

табДок.Вывести(ОбластьДанные);

// следующая группа

НомерГруппы = НомерГруппы + 1;

// проверяем все ли сгруппированы, чтобы не продолжать обход

Если Сгруппированно = КолСчетов Тогда

Прервать;

КонецЕсли;

КонецЦикла;

// отладка

ВремяВыполнения = ТекущаяУниверсальнаяДата() - ВремяНачала;

Сообщить("Время выполнения: " + ВремяВыполнения + " мс");

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

Процедура КнопкаСформироватьНажатие(Кнопка)

табДок = Новый ТабличныйДокумент;

СформироватьОтчет(табДок);

табДок.Показать("Результат");

КонецПроцедуры

Данная процедура отвечает за обработку нажатия кнопки Сформировать. В ней создается новый объект – табличный документ, в который формируется отчет (на стороне сервера) и после этого документ показывается пользователю.

Макет = ОтчетОбъект.ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьДанные = Макет.ПолучитьОбласть("Данные");

табДок.Вывести(ОбластьЗаголовок);

В данном блоке происходит получение макета табличного документа, его областей для отображения и вывод заголовка.

// массив счетов, участвующих в проводках

СчетаПроводок = Новый Массив();

// массив соответствий корреспонденции

ПроводкиСоответствие = Новый Массив();

// массив "сгруппирован или нет"

СчетСгруппирован = Новый Массив();

В данном блоке создаются массивы для хранения данных. СчетаПроводок – список все счетов, участвующих в проводках и которые должны быть распределены. ПроводкиСоответствие – массив содержащий массивы счетов, которые корреспондируют с соответствующим счетом. СчетСгруппирован – массив отвечающий за хранение логической переменной, отвечающей сгруппирован ли уже счет или нет.

Пример данных:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СчетаПроводок | | ПроводкиСоответствие | | | СчетСгруппирован | |
| Индекс | Счет | Индекс | Счета | | Индекс | Значение |
| 1 | 10 | 1 | 1 | 11 | 1 | Ложь |
| 2 | 21 |
| 2 | 11 | 2 | 1 | 10 | 2 | Ложь |
| 2 | 20 |
| 3 | 30 |
| 3 | 20 | 3 | 1 | 11 | 3 | Ложь |
| 2 | 21 |
| 4 | 21 | 4 | 1 | 10 | 4 | Ложь |
| 2 | 20 |
| 5 | 30 | 5 | 1 | 11 | 5 | Ложь |

// получаем счета имеющиеся в проводках

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ

| ПланСчетов.Код КАК Код,

| ПланСчетов.Родитель.Код

|ИЗ

| ПланСчетов.ПланСчетов КАК ПланСчетов,

| (ВЫБРАТЬ

| РегистрБухгалтерии.СчетДт.Код КАК СчетДтКод,

| РегистрБухгалтерии.СчетКт.Код КАК СчетКтКод

| ИЗ

| РегистрБухгалтерии.РегистрБухгалтерии КАК РегистрБухгалтерии) КАК ВложенныйЗапрос

|ГДЕ

| ПланСчетов.Код В (ВложенныйЗапрос.СчетДтКод, ВложенныйЗапрос.СчетКтКод)

|

|УПОРЯДОЧИТЬ ПО

| Код";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

// получаем код синтетического счета

СинтетическийСчет = Выборка.Код;

Если Выборка.РодительКод <> null Тогда

СинтетическийСчет = Выборка.РодительКод;

КонецЕсли;

КодСинтетическогоСчета = Число(СтрЗаменить(СинтетическийСчет,".",""));

// проверяем не добавлен ли уже счет в массив

Если СчетаПроводок.Найти(КодСинтетическогоСчета) = Неопределено Тогда

СчетаПроводок.Добавить(КодСинтетическогоСчета);

// создаем массив для корреспонденции

МассивКорреспонденции = Новый Массив();

ПроводкиСоответствие.Добавить(МассивКорреспонденции);

// отладка

//Сообщить("Добавлен счет: " + КодСинтетическогоСчета);

СчетСгруппирован.Добавить(Ложь);

КонецЕсли;

КонецЦикла;

В данном блоке идет получение счетов из базы данных и запись их в массив СчетаПроводок. Выбираются только счета участвующие в проводках, с сортировкой по Коду. Получаем Код счета и Код родителя счета, чтобы отбросить субсчета и взять только синтетические. После получения данных идет обход результата, если имеется родитель – берем его код, далее преобразуем строковый код к числу и добавляем в массив СчетаПроводок (предварительно проверив нет ли уже в массиве данного счета). При добавлении так же создаем массив для корреспонденции и добавляем Ложь в СчетСгруппирован (чтобы один и тот же индекс отвечал во всех трех массивах за данные об одном счете).

// считываем корреспонденцию

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| РегистрБухгалтерии.СчетДт.Код,

| РегистрБухгалтерии.СчетДт.Родитель.Код,

| РегистрБухгалтерии.СчетКт.Код,

| РегистрБухгалтерии.СчетКт.Родитель.Код,

| РегистрБухгалтерии.Сумма

|ИЗ

| РегистрБухгалтерии.РегистрБухгалтерии КАК РегистрБухгалтерии

|ГДЕ

| РегистрБухгалтерии.Сумма <> 0";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

// заполняем массив счетов, учавствующих в проводках

Пока Выборка.Следующий() Цикл

// СчетДТ

СинтетическийСчетДТ = Выборка.СчетДтКод;

Если Выборка.СчетДтРодительКод <> null Тогда

СинтетическийСчетДТ = Выборка.СчетДтРодительКод;

КонецЕсли;

КодСинтетическогоСчетаДТ = Число(СтрЗаменить(СинтетическийСчетДТ,".",""));

// СчетКТ

СинтетическийСчетКТ = Выборка.СчетКтКод;

Если Выборка.СчетКтРодительКод <> null Тогда

СинтетическийСчетКТ = Выборка.СчетКтРодительКод;

КонецЕсли;

КодСинтетическогоСчетаКТ = Число(СтрЗаменить(СинтетическийСчетКТ,".",""));

// отладка

//Сообщить("синтетические счета: " + КодСинтетическогоСчетаДТ + "," + КодСинтетическогоСчетаКТ);

// проверяем разные ли счета

Если КодСинтетическогоСчетаДТ <> КодСинтетическогоСчетаКТ Тогда

// ищем счета

СчетДТ = СчетаПроводок.Найти(КодСинтетическогоСчетаДТ);

СчетКТ = СчетаПроводок.Найти(КодСинтетическогоСчетаКТ);

// получаем массивы корреспонденции

МассивКорреспонденцииДТ = ПроводкиСоответствие.Получить(СчетДТ);

МассивКорреспонденцииКТ = ПроводкиСоответствие.Получить(СчетКТ);

// проверяем не было ли между ними корреспонденции

Если МассивКорреспонденцииДТ.Найти(КодСинтетическогоСчетаКТ) = Неопределено Тогда

// добавляем в массивы данные

МассивКорреспонденцииДТ.Добавить(КодСинтетическогоСчетаКТ);

МассивКорреспонденцииКТ.Добавить(КодСинтетическогоСчетаДТ);

//Сообщить("Добавлена корреспонденция: " + КодСинтетическогоСчетаДТ + "," + КодСинтетическогоСчетаКТ);

КонецЕсли;

КонецЕсли;

КонецЦикла;

В данном блоке происходит получение данных по проводкам из базы данных и заполняется массив корреспонденции. При получении данных берутся все проводки с ненулевыми суммами (одно из условий задачи). Далее при обходе результата получаются численные коды синтетических счетов дебета и кредита, и в случае если они разные (также одно из условий задачи), определяется индекс в массиве СчетаПроводок отвечающий за оба счета, берется массив корреспонденции каждого счета (расположенный по найденному индексу) и если в корреспонденции ещё нет записи о том, что эти два счета корреспондируют друг с другом – добавляются эти записи.

НомерГруппы = 1;

КолСчетов = СчетаПроводок.Количество();

Сгруппированно = 0;

// Обходим получившуюся корреспонденцию, отмечая сгруппированные счета

Для Индекс = 0 по СчетаПроводок.ВГраница() Цикл

// проверяем не сгруппирован ли уже счет

Если СчетСгруппирован[Индекс] = Истина Тогда

Продолжить;

КонецЕсли;

// получаем данные

Счет = СчетаПроводок.Получить(Индекс);

// устанавливаем данные перед обходом

СчетаВГруппе = Новый Массив();

СчетаВГруппе.Добавить(Индекс);

СчетСгруппирован[Индекс] = Истина;

Сгруппированно = Сгруппированно + 1;

// отладка

//Сообщить("Начало группы: " + Счет);

// обходим счета, в поиске отсутствующих в корреспонденции

Для Индекс2 = 0 по СчетаПроводок.ВГраница() Цикл

// проверяем не сгруппирован ли уже счет

Если СчетСгруппирован[Индекс2] = Истина Тогда

Продолжить;

КонецЕсли;

// получаем данные

СчетКор = СчетаПроводок.Получить(Индекс2);

// отладка

//Сообщить("Проверяем корреспонденцию группы с " + СчетКор);

// обходим группу

НеКорреспондирует = 0;

Для Каждого СчетГруппы Из СчетаВГруппе Цикл

// получаем его массив корреспонденции

МассивКорреспонденции = ПроводкиСоответствие.Получить(СчетГруппы);

// проверяем наличие корреспонденции

Если МассивКорреспонденции.Найти(СчетКор) = Неопределено Тогда

// запоминаем что не корреспондирует

НеКорреспондирует = НеКорреспондирует + 1;

КонецЕсли;

КонецЦикла;

// проверяем, со всеми ли не корреспондирует

Если НеКорреспондирует = СчетаВГруппе.Количество() Тогда

// отладка

//Сообщить("Кореспонденция не найдена");

// добавляем в группу

СчетаВГруппе.Добавить(Индекс2);

// помечаем сгруппированным

СчетСгруппирован[Индекс2] = Истина;

Сгруппированно = Сгруппированно + 1;

КонецЕсли;

КонецЦикла;

ГруппыСтр = Строка(Счет);

Для Индекс3 = 1 по СчетаВГруппе.ВГраница() Цикл

ИндексСчета = СчетаВГруппе[Индекс3];

ГруппыСтр = ГруппыСтр + ", " + СчетаПроводок.Получить(ИндексСчета);

КонецЦикла;

// выводим в табличный документ

ОбластьДанные.Параметры.НомерГруппы = НомерГруппы;

ОбластьДанные.Параметры.Счета = ГруппыСтр;

табДок.Вывести(ОбластьДанные);

// следующая группа

НомерГруппы = НомерГруппы + 1;

// проверяем все ли сгруппированы, чтобы не продолжать обход

Если Сгруппированно = КолСчетов Тогда

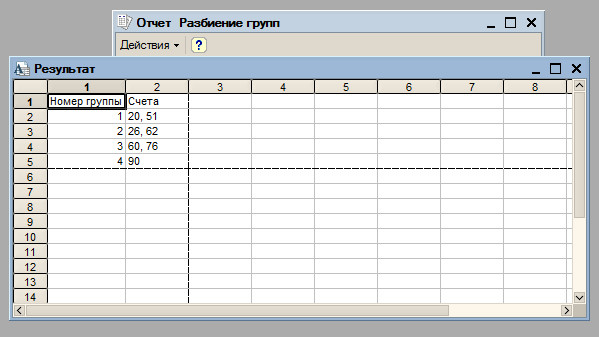
Прервать;

КонецЕсли;

КонецЦикла;

Главный блок – реализация алгоритма распределения по группам, вместе с выводом в табличный документ. Переменная НомерГруппы – отвечает за номер группы распределения, который выводится в табличный документ. КолСчетов – общее количество счетов для группировки. Сгруппировано – количество уже сгруппированных счетов. В начале итерации происходит проверка – сгруппирован ли уже текущий счет, или нет. В случае, если он уже сгруппирован – счет пропускается. Далее инициализируется массив счетов содержащихся в группе, в начале он состоит только из текущего счета. И текущий счет сразу помечается сгруппированным. Далее происходит обход всех счетов, так же пропуская уже сгруппированные и проверяется – корреспондирует ли счет с любым из счетов в массиве СчетаВГруппе. Если счет не корреспондирует ни с одним счетом из группы (переменная НеКорреспондирует = количеству счетов в группе), то счет добавляется в группу и помечается сгруппированным. При этом повышается значение счетчика Сгруппированно. После завершения составления группы она выводится в табличный документ (предварительно из массива с номерами счетов группы составляется строка), повышается счетчик НомерГруппы. В конце проверяется сгруппированы ли уже все счета, для прекращения обхода.

Сформированный отчет



2013 год 1 тур

Требования

В каркасной конфигурации разработать программу, которая анализирует текущие остатки спецодежды на складах и формирует отчет, содержащий две таблицы.

В первой таблице отображается текущее наличие запасов спецодежды на складах и определяется средняя цена для каждого наименования спецодежды.

Запасы спецодежды на складах предприятия на 01.04.2012

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Спецодежда | Цена |  | 1 склад | | 2 склад | | 3 склад | |
|  | Кол-во | Стоимость | Кол-во | Стоимость | Кол-во | Стоимость |
| СО-01 | 450 |  | 30 | 13500 | 10 | 4200 | 20 | 9300 |
| СО-02 | 280 |  | 18 | 5040 | 50 | 14000 | 15 | 4200 |
| СО-03 | 320 |  | 85 | 27200 | 60 | 19200 | 18 | 5760 |
| СО-04 | 610 |  | 5 | 3050 | 20 | 12200 | 62 | 37820 |
| СО-05 | 550 |  | 64 | 35200 | 12 | 6600 | 5 | 2750 |
| ИТОГО: |  |  |  | 83990 |  | 56200 |  | 59830 |
| ИТОГО по предприятию: | |  | 200020 | | | | | |

Во второй таблице должен быть представлен вариант размещения всех имеющихся запасов спецодежды на двух сохраняемых складах с соблюдением вышеуказанного условия – обеспечение максимальной близости (или равенства) стоимостной оценки запасов на этих складах, рассчитанной по средневзвешенной цене.

Предлагаемый вариант размещения спецодежды на двух складах

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Спецодежда | Цена | 1 склад | | 2 склад | |
| Кол-во | Стоимость | Кол-во | Стоимость |
| СО-01 | 450 | 46 | 20700 | 14 | 6300 |
| СО-02 | 280 | 28 | 7840 | 55 | 154000 |
| СО-03 | 320 | 100 | 32000 | 63 | 20160 |
| СО-04 | 610 | 7 | 4270 | 80 | 48800 |
| СО-05 | 550 | 64 | 35200 | 17 | 9350 |
| ИТОГО: |  |  | 100010 |  | 100010 |

Решение

Задача разделяется на 2 части: первая – составить сводную таблицу текущих данных, для создания таблицы требуются следующие данные: спецодежда (материалы), цена за единицу материала, количество и стоимость материала, находящегося на каждом складе. У нас имеются все данные, кроме цены за единицу материала, как именно находить цену материала сказано в задаче – «Оценка материальных запасов производится по средневзвешенной цене, которая определяется отдельно для каждого материала по предприятию в целом». Следовательно, для получения цены каждого материала, мы можем использовать следующую формулу: Цена = (Общая сумма / Общее количество). Используем следующий запрос, для получения цен материалов:

ВЫБРАТЬ

ЖурналПроводокОстатки.Субконто1,

СУММА(ЖурналПроводокОстатки.СуммаОстаток) / СУММА(ЖурналПроводокОстатки.КоличествоОстаток) КАК Цена

ИЗ

РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

ГДЕ

ЖурналПроводокОстатки.Счет.Код = "10"

СГРУППИРОВАТЬ ПО

ЖурналПроводокОстатки.Субконто1

И так, теперь у нас имеются все данные, которые требуется отобразить в первой части задания.

Для выполнения второй части задачи – составить предлагаемый вариант размещения материалов, требуется больше чем запрос. Для размещения материалов используем следующий алгоритм: отсортируем материалы от самого дорогого к самому дешевому, далее будем отправлять по одной единице материала на склад, на каждой итерации меняя склад на тот, где сумма товаров будет минимальной. Получается, товары будут равномерно распределены по оставшимся складам.

Пример:

Материал 1, цена 500, количество 3

Материал 2, цена 300, количество 2

Материал 3, цена 100, количество 5

Материалы будут распределены на 2 склада следующим образом:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Итерация | Склад 1 | Сумма Склад 1 | Склад 2 | Сумма Склад 2 |
| 1 | Материал 1 (500) | 500 |  | 0 |
| 2 |  | 500 | Материал 1 (500) | 500 |
| 3 | Материал 1 (500) | 1000 |  | 500 |
| 4 |  | 1000 | Материал 2 (300) | 800 |
| 5 |  | 1000 | Материал 2 (300) | 1100 |
| 6 | Материал 3 (100) | 1100 |  | 1100 |
| 7 | Материал 3 (100) | 1200 |  | 1100 |
| 8 |  | 1200 | Материал 3 (100) | 1200 |
| 9 | Материал 3 (100) | 1300 |  | 1200 |
| 10 |  | 1300 | Материал 3 (100) | 1300 |

В результате мы получаем равную сумму материалов на обоих складах. Для реализации алгоритма нам необходимо получить: список материалов, отсортированный по ценам от большего к меньшему, содержащий следующую информацию: материал, цена, общее количество на предприятии. Используя эту информацию можно распределить материалы по складам приведенным выше алгоритмом. Псевдокод алгоритма:

Для каждого материал Цикл

Для каждой единицы материала Цикл

Склад = склад с наименьшей суммой материалов

Отправить единицу материала на Склад

Конец цикла

Конец цикла

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабДок;

Макет = Отчеты.ЛиквидацияСклада.ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьЗаголовокОсновное = Макет.ПолучитьОбласть("ЗаголовокОсновное");

ОбластьЗаголовокСклад = Макет.ПолучитьОбласть("ЗаголовокСклад");

ОбластьЗаголовокИтого = Макет.ПолучитьОбласть("ЗаголовокИтого");

ОбластьДанныеОсновное = Макет.ПолучитьОбласть("ДанныеОсновное");

ОбластьДанныеСклад = Макет.ПолучитьОбласть("ДанныеСклад");

ОбластьДанныеИтого = Макет.ПолучитьОбласть("ДанныеИтого");

ОбластьДанныеПоПредприятию = Макет.ПолучитьОбласть("ДанныеПоПредприятию");

ТабДок.Очистить();

Склады = Новый Массив;

Материалы = Новый Массив;

Остатки = Новый Массив;

ИтогСумма = 0;

ИтогСкладыСумма = Новый Соответствие;

// Получаем склады

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Склады.Ссылка

|ИЗ

| Справочник.Склады КАК Склады";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДанныеСклада = Новый Структура;

ДанныеСклада.Вставить("Ссылка",Выборка.Ссылка);

Склады.Добавить(ДанныеСклада);

КонецЦикла;

// Получаем средние цены на материалы

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1.Ссылка КАК Ссылка,

| СУММА(ЖурналПроводокОстатки.СуммаОстаток) / СУММА(ЖурналПроводокОстатки.КоличествоОстаток) КАК Цена

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет.Код = ""10""

|

|СГРУППИРОВАТЬ ПО

| ЖурналПроводокОстатки.Субконто1,

| ЖурналПроводокОстатки.Субконто1.Ссылка

|

|УПОРЯДОЧИТЬ ПО

| Цена УБЫВ";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДанныеМатериала = Новый Структура;

ДанныеМатериала.Вставить("Ссылка",Выборка.Ссылка);

ДанныеМатериала.Вставить("Цена",Выборка.Цена);

Материалы.Добавить(ДанныеМатериала);

КонецЦикла;

// Получаем текущие запасы

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Материал,

| ЖурналПроводокОстатки.Субконто2 КАК Склад,

| ЖурналПроводокОстатки.СуммаОстаток,

| ЖурналПроводокОстатки.КоличествоОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&КонецПериода, , , ) КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет.Код = ""10""

|

|УПОРЯДОЧИТЬ ПО

| ЖурналПроводокОстатки.Субконто1.Наименование,

| ЖурналПроводокОстатки.Субконто2.Код";

Запрос.УстановитьПараметр("КонецПериода",ОтчетОбъект.Дата);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Предыдущий = Неопределено;

ТекущиеДанные = Неопределено;

Пока Выборка.Следующий() Цикл

Если Выборка.Материал <> Предыдущий Тогда

Предыдущий = Выборка.Материал;

Если ТекущиеДанные <> Неопределено Тогда

Остатки.Добавить(ТекущиеДанные);

КонецЕсли;

ТекущиеДанные = Новый Структура;

ТекущиеДанные.Вставить("Материал",Выборка.Материал);

Для Каждого Материал из Материалы Цикл

Если Материал.Ссылка = Выборка.Материал.Ссылка Тогда

ТекущиеДанные.Вставить("Цена",Материал.Цена);

Прервать;

КонецЕсли;

КонецЦикла;

КонецЕсли;

Для Каждого Материал из Материалы Цикл

Если Материал.Ссылка = Выборка.Материал.Ссылка Тогда

Количество = Неопределено;

Если Не Материал.Свойство("Количество",Количество) Тогда

Количество = 0;

КонецЕсли;

Материал.Вставить("Количество",Количество + Выборка.КоличествоОстаток);

Прервать;

КонецЕсли;

КонецЦикла;

МатериалНаСкладе = Новый Структура;

МатериалНаСкладе.Вставить("Склад",Выборка.Склад);

МатериалНаСкладе.Вставить("Сумма",Выборка.СуммаОстаток);

МатериалНаСкладе.Вставить("Количество",Выборка.КоличествоОстаток);

МатериалСклады = Неопределено;

Если Не ТекущиеДанные.Свойство("Склады",МатериалСклады) Тогда

МатериалСклады = Новый Массив;

КонецЕсли;

МатериалСклады.Добавить(МатериалНаСкладе);

ТекущиеДанные.Вставить("Склады",МатериалСклады);

СкладСумма = ИтогСкладыСумма.Получить(Выборка.Склад.Ссылка);

Если СкладСумма = Неопределено Тогда

ИтогСкладыСумма.Вставить(Выборка.Склад.Ссылка,Выборка.СуммаОстаток);

Иначе

ИтогСкладыСумма.Вставить(Выборка.Склад.Ссылка,СкладСумма + Выборка.СуммаОстаток);

КонецЕсли;

ИтогСумма = ИтогСумма + Выборка.СуммаОстаток;

КонецЦикла;

Если ТекущиеДанные <> Неопределено Тогда

Остатки.Добавить(ТекущиеДанные);

КонецЕсли;

// Строим Документ

ОбластьЗаголовок.Параметры[0] = "Запасы спецодежды на складах предприятия на " + Дата;

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьЗаголовокОсновное);

Для Каждого Склад Из Склады Цикл

ОбластьЗаголовокСклад.Параметры[0] = Склад.Ссылка.Наименование;

ТабДок.Присоединить(ОбластьЗаголовокСклад);

КонецЦикла;

Для Каждого Остаток Из Остатки Цикл

ОбластьДанныеОсновное.Параметры[0] = Остаток.Материал;

ОбластьДанныеОсновное.Параметры[1] = Остаток.Цена;

ТабДок.Вывести(ОбластьДанныеОсновное);

МатериалСклады = Остаток.Склады;

Для Каждого МатериалСклад Из МатериалСклады Цикл

ОбластьДанныеСклад.Параметры[0] = МатериалСклад.Количество;

ОбластьДанныеСклад.Параметры[1] = МатериалСклад.Сумма;

ТабДок.Присоединить(ОбластьДанныеСклад);

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьЗаголовокИтого);

Для Каждого Склад Из Склады Цикл

ОбластьДанныеИтого.Параметры[0] = ИтогСкладыСумма.Получить(Склад.Ссылка);

ТабДок.Присоединить(ОбластьДанныеИтого);

КонецЦикла;

ОбластьДанныеПоПредприятию.Параметры[0] = ИтогСумма;

ТабДок.Вывести(ОбластьДанныеПоПредприятию);

// Инициализируем данные результата

МатериалыРезультат = Новый Массив();

ИтогСкладыСумма = Новый Соответствие;

Для Каждого Материал из Материалы Цикл

Данные = Новый Структура;

Данные.Вставить("Материал",Материал.Ссылка);

Данные.Вставить("Цена",Материал.Цена);

СкладыРезультат = Новый Массив();

Для Каждого Склад Из Склады Цикл

Если Склад.Ссылка <> ОтчетОбъект.ЛиквидируемыйСклад.Ссылка Тогда

ДанныеСклад = Новый Структура;

ДанныеСклад.Вставить("Склад",Склад.Ссылка);

ДанныеСклад.Вставить("Сумма",0);

СкладыРезультат.Добавить(ДанныеСклад);

КонецЕсли;

КонецЦикла;

ТекущийСклад = СкладыРезультат[0];

Пока Материал.Количество > 0 Цикл

Количество = 0;

Если Не ТекущийСклад.Свойство("Количество",Количество) Тогда

ТекущийСклад.Вставить("Количество",1);

Иначе

ТекущийСклад.Вставить("Количество",Количество + 1);

КонецЕсли;

Сумма = 0;

Если Не ТекущийСклад.Свойство("Сумма",Сумма) Тогда

ТекущийСклад.Вставить("Сумма",1);

Иначе

ТекущийСклад.Вставить("Сумма",Сумма + Материал.Цена);

КонецЕсли;

СкладСумма = ИтогСкладыСумма.Получить(ТекущийСклад.Склад.Ссылка);

Если СкладСумма = Неопределено Тогда

ИтогСкладыСумма.Вставить(ТекущийСклад.Склад.Ссылка,Материал.Цена);

Иначе

ИтогСкладыСумма.Вставить(ТекущийСклад.Склад.Ссылка,СкладСумма + Материал.Цена);

КонецЕсли;

Для Каждого Склад Из СкладыРезультат Цикл

Если Склад.Сумма < ТекущийСклад.Сумма Тогда

ТекущийСклад = Склад;

КонецЕсли;

КонецЦикла;

Материал.Количество = Материал.Количество - 1;

КонецЦикла;

Данные.Вставить("Склады",СкладыРезультат);

МатериалыРезультат.Добавить(Данные);

КонецЦикла;

// Строим Документ

ОбластьЗаголовок.Параметры[0] = "Предлагаемый вариант размещения спецодежды";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьЗаголовокОсновное);

Для Каждого Склад Из Склады Цикл

Если Склад.Ссылка <> ОтчетОбъект.ЛиквидируемыйСклад.Ссылка Тогда

ОбластьЗаголовокСклад.Параметры[0] = Склад.Ссылка.Наименование;

ТабДок.Присоединить(ОбластьЗаголовокСклад);

КонецЕсли;

КонецЦикла;

Для Каждого Материал Из МатериалыРезультат Цикл

ОбластьДанныеОсновное.Параметры[0] = Материал.Материал;

ОбластьДанныеОсновное.Параметры[1] = Материал.Цена;

ТабДок.Вывести(ОбластьДанныеОсновное);

МатериалСклады = Материал.Склады;

Для Каждого МатериалСклад Из МатериалСклады Цикл

ОбластьДанныеСклад.Параметры[0] = МатериалСклад.Количество;

ОбластьДанныеСклад.Параметры[1] = МатериалСклад.Сумма;

ТабДок.Присоединить(ОбластьДанныеСклад);

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьЗаголовокИтого);

Для Каждого Склад Из Склады Цикл

Если Склад.Ссылка <> ОтчетОбъект.ЛиквидируемыйСклад.Ссылка Тогда

ОбластьДанныеИтого.Параметры[0] = ИтогСкладыСумма.Получить(Склад.Ссылка);

ТабДок.Присоединить(ОбластьДанныеИтого);

КонецЕсли;

КонецЦикла;

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

ТабДок = ЭлементыФормы.ПолеТабДок;

Макет = Отчеты.ЛиквидацияСклада.ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьЗаголовокОсновное = Макет.ПолучитьОбласть("ЗаголовокОсновное");

ОбластьЗаголовокСклад = Макет.ПолучитьОбласть("ЗаголовокСклад");

ОбластьЗаголовокИтого = Макет.ПолучитьОбласть("ЗаголовокИтого");

ОбластьДанныеОсновное = Макет.ПолучитьОбласть("ДанныеОсновное");

ОбластьДанныеСклад = Макет.ПолучитьОбласть("ДанныеСклад");

ОбластьДанныеИтого = Макет.ПолучитьОбласть("ДанныеИтого");

ОбластьДанныеПоПредприятию = Макет.ПолучитьОбласть("ДанныеПоПредприятию");

ТабДок.Очистить();

В данном блоке получаем элемент формы в который будем выводить результат (Поле табличного документа), макет представления данных и все требуемые области макета, которые будут использованы для построения документа. Последним действием отчищаем элемент формы, чтобы удалить старые данные (при повторном запуске формировки отчета).

Склады = Новый Массив;

Материалы = Новый Массив;

Остатки = Новый Массив;

ИтогСумма = 0;

ИтогСкладыСумма = Новый Соответствие;

В данном блоке проводится инициализация мест хранения данных, склады – массив, содержащий структуры с полями (Ссылка), Материалы – массив, содержащий структуры с полями (Ссылка, Цена), Остатки – массив, содержащий структуры с полями (Материал, Цена, Склады), ИтогСумма – сумма всех материалов по предприятию, ИтогСкладыСумма – соответствие (Склад – сумма материалов на складе). Следует уточнить, что поле Склады, структуры Остатков тоже является массивом, который содержит структуры с полями (Склад,Количество,Сумма). Приведем данные в более понятной форме:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Элемент массива… | Данные | | | | | |
| Склады | Ссылка | | | | | |
| Склад 1 | | | | | |
| Материалы | Ссылка | | Цена | | | |
| СО-01 | | 500 | | | |
| Остатки | Материал | Цена | | Склады | | |
| СО-01 | 500 | | Склад 1 | 10 | 5000 |
| Склад 2 | 5 | 2500 |

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Склады.Ссылка

|ИЗ

| Справочник.Склады КАК Склады";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДанныеСклада = Новый Структура;

ДанныеСклада.Вставить("Ссылка",Выборка.Ссылка);

Склады.Добавить(ДанныеСклада);

КонецЦикла;

В данном блоке идет обращение к базе данных, для получения всех складов. После получения данных, результат записывается в массив Склады.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1.Ссылка КАК Ссылка,

| СУММА(ЖурналПроводокОстатки.СуммаОстаток) / СУММА(ЖурналПроводокОстатки.КоличествоОстаток) КАК Цена

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет.Код = ""10""

|

|СГРУППИРОВАТЬ ПО

| ЖурналПроводокОстатки.Субконто1,

| ЖурналПроводокОстатки.Субконто1.Ссылка

|

|УПОРЯДОЧИТЬ ПО

| Цена УБЫВ";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДанныеМатериала = Новый Структура;

ДанныеМатериала.Вставить("Ссылка",Выборка.Ссылка);

ДанныеМатериала.Вставить("Цена",Выборка.Цена);

Материалы.Добавить(ДанныеМатериала);

КонецЦикла;

В данном блоке происходит обращение к базе данных для получения списка материалов, с получением средневзвешенной цены, по которой происходит сортировка от большего к меньшему. Для получения данных учитывается только счет с кодом 10. После получения данных они записываются в массив Материалы.

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Материал,

| ЖурналПроводокОстатки.Субконто2 КАК Склад,

| ЖурналПроводокОстатки.СуммаОстаток,

| ЖурналПроводокОстатки.КоличествоОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&КонецПериода, , , ) КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет.Код = ""10""

|

|УПОРЯДОЧИТЬ ПО

| ЖурналПроводокОстатки.Субконто1.Наименование,

| ЖурналПроводокОстатки.Субконто2.Код";

Запрос.УстановитьПараметр("КонецПериода",ОтчетОбъект.Дата);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Предыдущий = Неопределено;

ТекущиеДанные = Неопределено;

Пока Выборка.Следующий() Цикл

Если Выборка.Материал <> Предыдущий Тогда

Предыдущий = Выборка.Материал;

Если ТекущиеДанные <> Неопределено Тогда

Остатки.Добавить(ТекущиеДанные);

КонецЕсли;

ТекущиеДанные = Новый Структура;

ТекущиеДанные.Вставить("Материал",Выборка.Материал);

Для Каждого Материал из Материалы Цикл

Если Материал.Ссылка = Выборка.Материал.Ссылка Тогда

ТекущиеДанные.Вставить("Цена",Материал.Цена);

Прервать;

КонецЕсли;

КонецЦикла;

КонецЕсли;

Для Каждого Материал из Материалы Цикл

Если Материал.Ссылка = Выборка.Материал.Ссылка Тогда

Количество = Неопределено;

Если Не Материал.Свойство("Количество",Количество) Тогда

Количество = 0;

КонецЕсли;

Материал.Вставить("Количество",Количество + Выборка.КоличествоОстаток);

Прервать;

КонецЕсли;

КонецЦикла;

МатериалНаСкладе = Новый Структура;

МатериалНаСкладе.Вставить("Склад",Выборка.Склад);

МатериалНаСкладе.Вставить("Сумма",Выборка.СуммаОстаток);

МатериалНаСкладе.Вставить("Количество",Выборка.КоличествоОстаток);

МатериалСклады = Неопределено;

Если Не ТекущиеДанные.Свойство("Склады",МатериалСклады) Тогда

МатериалСклады = Новый Массив;

КонецЕсли;

МатериалСклады.Добавить(МатериалНаСкладе);

ТекущиеДанные.Вставить("Склады",МатериалСклады);

СкладСумма = ИтогСкладыСумма.Получить(Выборка.Склад.Ссылка);

Если СкладСумма = Неопределено Тогда

ИтогСкладыСумма.Вставить(Выборка.Склад.Ссылка,Выборка.СуммаОстаток);

Иначе

ИтогСкладыСумма.Вставить(Выборка.Склад.Ссылка,СкладСумма + Выборка.СуммаОстаток);

КонецЕсли;

ИтогСумма = ИтогСумма + Выборка.СуммаОстаток;

КонецЦикла;

Если ТекущиеДанные <> Неопределено Тогда

Остатки.Добавить(ТекущиеДанные);

КонецЕсли;

В данном блоке происходит получение остатков на складах из базы данных. Остатки получаются до указанной даты (параметр КонецПериода), работа проходит так же только со счетом 10. Данные упорядочиваются по материалу и коду склада. Сортировка по материалу необходима, для того чтобы несколько записей относящихся к одному материалу, но разным складам, можно было рассматривать вместе. После получения данных они записываются в массив Остатки. Так как несколько строк выборки относятся к одной записи массива, используется дополнительная переменная Предыдущий (для определения момента перехода к следующему материалу) и переменная ТекущиеДанные, для заполнения структуры текущего материала. За определение перехода к следующему материалу отвечает следующий код:

Если Выборка.Материал <> Предыдущий Тогда

Предыдущий = Выборка.Материал;

Если ТекущиеДанные <> Неопределено Тогда

Остатки.Добавить(ТекущиеДанные);

КонецЕсли;

ТекущиеДанные = Новый Структура;

ТекущиеДанные.Вставить("Материал",Выборка.Материал);

Для Каждого Материал из Материалы Цикл

Если Материал.Ссылка = Выборка.Материал.Ссылка Тогда

ТекущиеДанные.Вставить("Цена",Материал.Цена);

Прервать;

КонецЕсли;

КонецЦикла;

КонецЕсли;

Если текущий материал выборки не соответствует предыдущему, то предыдущим считается текущий материал выборки, а текущие данные (заполненная структура предыдущего материала) добавляются в массив Остатки. После этого создается новая структура ТекущиеДанные, в которой заполняется поле Материал из выборки, и Цена. Для Цены материал ищется в массиве Материалы, содержащим цены на материалы.

Для Каждого Материал из Материалы Цикл

Если Материал.Ссылка = Выборка.Материал.Ссылка Тогда

Количество = Неопределено;

Если Не Материал.Свойство("Количество",Количество) Тогда

Количество = 0;

КонецЕсли;

Материал.Вставить("Количество",Количество + Выборка.КоличествоОстаток);

Прервать;

КонецЕсли;

КонецЦикла;

В данной части происходит подсчет количества материалов (общее количество материалов на предприятии), для дальнейшего распределения по складам. Сначала находится элемент массива Материалы соответствующий материалу выборки, далее берется текущее количество материала со структуры (с учетом того, что поле «Количество» может отсутствовать в структуре) и выставляется новое количество материала (Старое + Количество из выборки).

МатериалНаСкладе = Новый Структура;

МатериалНаСкладе.Вставить("Склад",Выборка.Склад);

МатериалНаСкладе.Вставить("Сумма",Выборка.СуммаОстаток);

МатериалНаСкладе.Вставить("Количество",Выборка.КоличествоОстаток);

МатериалСклады = Неопределено;

Если Не ТекущиеДанные.Свойство("Склады",МатериалСклады) Тогда

МатериалСклады = Новый Массив;

КонецЕсли;

МатериалСклады.Добавить(МатериалНаСкладе);

ТекущиеДанные.Вставить("Склады",МатериалСклады);

В данной части кода составляется структура для добавления в поле Склады элемента массива Остатки. После заполнения структуры, она добавляется в массив складов (с учетом того, что поле «Склады» может отсутствовать в структуре). После изменения массива, поле Склады заменяется этим массивом.

СкладСумма = ИтогСкладыСумма.Получить(Выборка.Склад.Ссылка);

Если СкладСумма = Неопределено Тогда

ИтогСкладыСумма.Вставить(Выборка.Склад.Ссылка,Выборка.СуммаОстаток);

Иначе

ИтогСкладыСумма.Вставить(Выборка.Склад.Ссылка,СкладСумма + Выборка.СуммаОстаток);

КонецЕсли;

ИтогСумма = ИтогСумма + Выборка.СуммаОстаток;

Эта часть кода отвечает за подсчет суммы цен материалов на складах и общей суммы на всем предприятии.

Если ТекущиеДанные <> Неопределено Тогда

Остатки.Добавить(ТекущиеДанные);

КонецЕсли;

Данная часть, расположенная сразу после цикла, необходима для того, чтобы последний материал выборки был добавлен в массив (добавление остальных происходит при изменении материала в выборке).

ОбластьЗаголовок.Параметры[0] = "Запасы спецодежды на складах предприятия на " + Дата;

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьЗаголовокОсновное);

Для Каждого Склад Из Склады Цикл

ОбластьЗаголовокСклад.Параметры[0] = Склад.Ссылка.Наименование;

ТабДок.Присоединить(ОбластьЗаголовокСклад);

КонецЦикла;

Для Каждого Остаток Из Остатки Цикл

ОбластьДанныеОсновное.Параметры[0] = Остаток.Материал;

ОбластьДанныеОсновное.Параметры[1] = Остаток.Цена;

ТабДок.Вывести(ОбластьДанныеОсновное);

МатериалСклады = Остаток.Склады;

Для Каждого МатериалСклад Из МатериалСклады Цикл

ОбластьДанныеСклад.Параметры[0] = МатериалСклад.Количество;

ОбластьДанныеСклад.Параметры[1] = МатериалСклад.Сумма;

ТабДок.Присоединить(ОбластьДанныеСклад);

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьЗаголовокИтого);

Для Каждого Склад Из Склады Цикл

ОбластьДанныеИтого.Параметры[0] = ИтогСкладыСумма.Получить(Склад.Ссылка);

ТабДок.Присоединить(ОбластьДанныеИтого);

КонецЦикла;

ОбластьДанныеПоПредприятию.Параметры[0] = ИтогСумма;

ТабДок.Вывести(ОбластьДанныеПоПредприятию);

Данный блок отвечает за построение табличного документа. Сначала производится вывод заголовков и названий складов, далее выводится все остатки. В завершение выводятся итоги по складам и общий итог.

МатериалыРезультат = Новый Массив();

ИтогСкладыСумма = Новый Соответствие;

Для Каждого Материал из Материалы Цикл

Данные = Новый Структура;

Данные.Вставить("Материал",Материал.Ссылка);

Данные.Вставить("Цена",Материал.Цена);

СкладыРезультат = Новый Массив();

Для Каждого Склад Из Склады Цикл

Если Склад.Ссылка <> ОтчетОбъект.ЛиквидируемыйСклад.Ссылка Тогда

ДанныеСклад = Новый Структура;

ДанныеСклад.Вставить("Склад",Склад.Ссылка);

ДанныеСклад.Вставить("Сумма",0);

СкладыРезультат.Добавить(ДанныеСклад);

КонецЕсли;

КонецЦикла;

ТекущийСклад = СкладыРезультат[0];

Пока Материал.Количество > 0 Цикл

Количество = 0;

Если Не ТекущийСклад.Свойство("Количество",Количество) Тогда

ТекущийСклад.Вставить("Количество",1);

Иначе

ТекущийСклад.Вставить("Количество",Количество + 1);

КонецЕсли;

Сумма = 0;

Если Не ТекущийСклад.Свойство("Сумма",Сумма) Тогда

ТекущийСклад.Вставить("Сумма",1);

Иначе

ТекущийСклад.Вставить("Сумма",Сумма + Материал.Цена);

КонецЕсли;

СкладСумма = ИтогСкладыСумма.Получить(ТекущийСклад.Склад.Ссылка);

Если СкладСумма = Неопределено Тогда

ИтогСкладыСумма.Вставить(ТекущийСклад.Склад.Ссылка,Материал.Цена);

Иначе

ИтогСкладыСумма.Вставить(ТекущийСклад.Склад.Ссылка,СкладСумма + Материал.Цена);

КонецЕсли;

Для Каждого Склад Из СкладыРезультат Цикл

Если Склад.Сумма < ТекущийСклад.Сумма Тогда

ТекущийСклад = Склад;

КонецЕсли;

КонецЦикла;

Материал.Количество = Материал.Количество - 1;

КонецЦикла;

Данные.Вставить("Склады",СкладыРезультат);

МатериалыРезультат.Добавить(Данные);

КонецЦикла;

Данный блок – главный, тут реализован алгоритм распределения материалов по оставшимся складам. Так как массив Материалы отсортирован от большего к меньшему по цене, мы обходим все элементы данного массива. На каждой итерации заполняется структура Данные для добавления в массив МатериалыРезультат, в структуру данные заполняются Материал,Цена и формируются массив Склады. Для распределения по складам составляется массив остающихся складов – СкладыРезультат (все склады кроме ликвидируемого). Далее материалы по единице распределяются по складам, выбирая склад с наименьшей суммой.

ОбластьЗаголовок.Параметры[0] = "Предлагаемый вариант размещения спецодежды";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьЗаголовокОсновное);

Для Каждого Склад Из Склады Цикл

Если Склад.Ссылка <> ОтчетОбъект.ЛиквидируемыйСклад.Ссылка Тогда

ОбластьЗаголовокСклад.Параметры[0] = Склад.Ссылка.Наименование;

ТабДок.Присоединить(ОбластьЗаголовокСклад);

КонецЕсли;

КонецЦикла;

Для Каждого Материал Из МатериалыРезультат Цикл

ОбластьДанныеОсновное.Параметры[0] = Материал.Материал;

ОбластьДанныеОсновное.Параметры[1] = Материал.Цена;

ТабДок.Вывести(ОбластьДанныеОсновное);

МатериалСклады = Материал.Склады;

Для Каждого МатериалСклад Из МатериалСклады Цикл

ОбластьДанныеСклад.Параметры[0] = МатериалСклад.Количество;

ОбластьДанныеСклад.Параметры[1] = МатериалСклад.Сумма;

ТабДок.Присоединить(ОбластьДанныеСклад);

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьЗаголовокИтого);

Для Каждого Склад Из Склады Цикл

Если Склад.Ссылка <> ОтчетОбъект.ЛиквидируемыйСклад.Ссылка Тогда

ОбластьДанныеИтого.Параметры[0] = ИтогСкладыСумма.Получить(Склад.Ссылка);

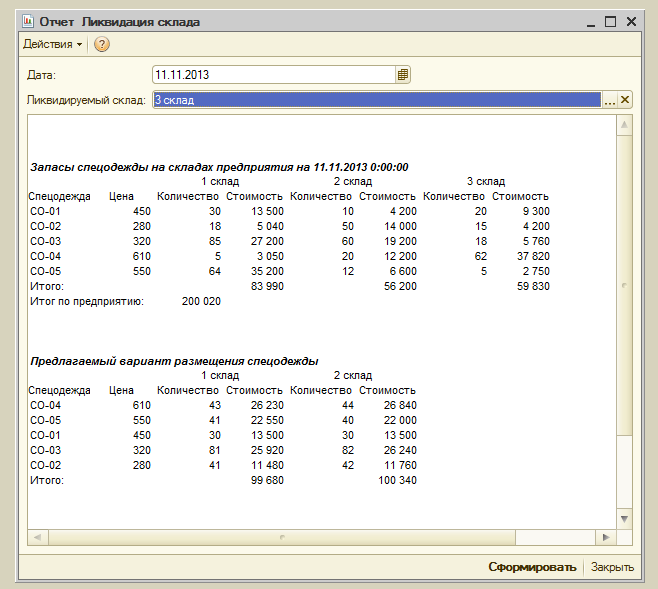
ТабДок.Присоединить(ОбластьДанныеИтого);

КонецЕсли;

КонецЦикла;

Последняя часть процедуры – добавление в табличный документ данных распределения по складам. Вывод идет аналогично выводу остатков, но данные берутся из МатериалыРезультат, и в складах не учитывается ликвидируемый склад.

Сформированный отчет



2012 год 2 тур

Требования

Задание финального тура

Исходная ситуация

Для хранения спецодежды на предприятии используется три склада.

Предприятие ведет бухгалтерский учет в программе "1С:Бухгалтерия 8". Учет спецодежды ведется в составе материальных запасов на счете 10 в разрезе субконто "Материалы" и "Склады" в стоимостном и натуральном выражении. Оценка материальных запасов производится по средневзвешенной цене, которая определяется отдельно для каждого материала по предприятию в целом.

В информационной базе предприятия зафиксированы остатки спецодежды на 1 апреля 2012 г. по каждому наименованию спецодежды и по каждому складу.

Предприятие решает ликвидировать один из складов для хранения спецодежды, а остатки спецодежды с этого склада переместить на два оставшихся склада таким образом, чтобы стоимостная оценка запасов спецодежды на этих двух складах по возможности составляла бы одинаковую величину или, если это невозможно, различалась бы минимально. При этом следует исходить из того, что перемещать исходные запасы спецодежды между неликвидируемыми складами запрещено.

Требуется

В каркасной конфигурации разработать программу, которая анализирует текущие остатки спецодежды на складах и формирует отчет, содержащий две таблицы.

В первой таблице отображается текущее наличие запасов спецодежды на складах и определяется средняя цена для каждого наименования спецодежды.

Запасы спецодежды на складах предприятия на 01.04.2012

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Спецодежда | Цена | 1 склад | | 2 склад | | 3 склад | |
| Кол-во | Стоимость | Кол-во | Стоимость | Кол-во | Стоимость |
| СО-01 | 450 | 30 | 13500 | 10 | 4200 | 20 | 9300 |
| СО-02 | 280 | 18 | 5040 | 50 | 14000 | 15 | 4200 |
| СО-03 | 320 | 85 | 27200 | 60 | 19200 | 18 | 5760 |
| СО-04 | 610 | 5 | 3050 | 20 | 12200 | 62 | 37820 |
| СО-05 | 550 | 64 | 35200 | 12 | 6600 | 5 | 2750 |
| ИТОГО: |  |  | 83990 |  | 56200 |  | 59830 |
| ИТОГО по предприятию: | | 200020 | | | | | |

Во второй таблице должен быть представлен вариант размещения всех имеющихся запасов спецодежды на двух сохраняемых складах с соблюдением вышеуказанного условия – обеспечение максимальной близости (или равенства) стоимостной оценки запасов на этих складах, рассчитанной по средневзвешенной цене.

Предлагаемый вариант размещения спецодежды на двух складах

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Спецодежда | Цена | 1 склад | | 2 склад | |
| Кол-во | Стоимость | Кол-во | Стоимость |
| СО-01 | 450 | 46 | 20700 | 14 | 6300 |
| СО-02 | 280 | 28 | 7840 | 55 | 154000 |
| СО-03 | 320 | 100 | 32000 | 63 | 20160 |
| СО-04 | 610 | 7 | 4270 | 80 | 48800 |
| СО-05 | 550 | 64 | 35200 | 17 | 9350 |
| ИТОГО: |  |  | 100010 |  | 100010 |

Исходные данные

Исходные данные представлены в ИБ каркасной конфигурации.

В каркасную конфигурацию включены:

План счетов бухгалтерского учета, содержащий счета 10 "Материалы" и   
00 "Вспомогательный"

Справочники "Материалы" и "Склады", используемые, соответственно, для ведения аналитического учета на счете 10 в разрезе материалов и складов

Документ для ввода данных о запасах материалов на складах, при помощи которого введено несколько проводок в дебет счета 10 в корреспонденции со вспомогательным счетом 00 для отражения запасов спецодежды на складах на начало дня 01.04.2012

Результаты и их представление

В результате выполнения задания должен быть разработан отчет "1С:Предприятия 8", формирующий две вышеуказанные таблицы. Необходимо предусмотреть форму диалога, в которой предоставить пользователю возможность определить дату, на которую производится формирование отчета, а также возможность выбора ликвидируемого склада. По умолчанию должна устанавливаться дата ‑ 01.04.2012, а в качестве ликвидируемого склада ‑ склад с максимальным значением кода.

Для отладки модуля отчета допускается добавлять в информационную базу каркасной конфигурации дополнительные элементы справочников и проводки. Жюри будет проводить тестирование на собственных наборах данных, которые могут отличаться от данных, содержащихся в исходной информационной базе каркасной конфигурации.

Разработанный отчет должен быть сохранен как внешний отчет "1С:Предприятия 8" и передан на проверку жюри.

Решение

Для получения средневзвешанных цен товаров требуется получить сумму товаров и сумму количества товаров по всему предприятию и после этого разделить сумму на количество. Для поиска оптимального варианта размещения спецодежды на оставшихся складах можно использовать следующий алгоритм: сначала оставшиеся склады сортируются по убыванию по сумме, которой не хватает до средней сумме по предприятию. Затем товары перемещаются на склады (сначала на склад у которого отклонение от средней суммы максимально, потом с меньшим), сначала дорогие, потом дешевые, при чем так, чтобы отклонение от средней суммы было минимальным. Рассмотрим пример:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Товар | Склад 1 | | Склад 2 | | Склад 3 (ликвид.) | |
| Количество | Сумма | Количество | Сумма | Количество | Сумма |
| Товар 1 | 3 | 21 | 2 | 14 | 4 | 28 |
| Товар 2 | 2 | 20 | 3 | 30 | 1 | 10 |

Сперва находятся средневзвешанные цены:

|  |  |  |  |
| --- | --- | --- | --- |
| Товар | Сумма по предприятию | Количество по предприятию | Средневзвешанная цена |
| Товар 1 | 63 | 9 | 7 |
| Товар 2 | 60 | 6 | 10 |

Далее расчитывается общая сумма товаров по предприятию и средняя сумма:

|  |  |
| --- | --- |
| Сумма цен по предприятию | 123 |
| Средняя сумма цен по предприятию | 61.5 |

Далее высчитывается нехватка суммы на остающихся складах до средней суммы:

|  |  |
| --- | --- |
| Склад 1 | Склад 2 |
| 61.5 – (21 + 20) = 20.5 | 61.5 – (14 + 30) = 17.5 |

Теперь склады сортируются по убыванию:

|  |  |
| --- | --- |
| Склад | Сумма |
| Склад 1 | 20.5 |
| Склад 2 | 17.5 |

Далее на склады, отсортированные по убыванию, размещаются товары (так же отсортированные по убыванию) перенося по одной единице товара с ликвидируемого склада пока сумма на складе не выше средней по предприятию.

Сначала обрабатывается Склад 1:

|  |  |  |  |
| --- | --- | --- | --- |
| Итерация | Добавленный товар | Остаток добавленного товара на ликвидируемом складе | Сумма товаров на складе |
| 1 | Товар 2 | 0 | 10.5 |
| 2 | Товар 1 | 3 | 3.5 |
| 3 | При добавлении Товара 1 на склад сумма товаров будет больше средней по предприятию, обрываем добавление на склад 1 | | |

Далее обрабатывается Склад 2:

|  |  |  |  |
| --- | --- | --- | --- |
| Итерация | Добавленный товар | Остаток добавленного товара на ликвидируемом складе | Сумма товаров на складе |
| 1 | Товар 1 | 2 | 10.5 |
| 2 | Товар 1 | 1 | 3.5 |
| 3 | При добавлении Товара 1 на склад сумма товаров будет больше средней по предприятию, обрываем добавление на склад 2 | | |

Остается один не распределенный товар, он отправляется на склад, у которого больше всего отклонение от средневзвешенной цены, но так как в примере отклонение после распределения у обоих складов будет 3.5 товар можно отправить на любой склад (к примеру на склад 1).

В результате получается следующее распределение товаров:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Товар | Склад 1 | | Склад 2 | |
| Количество | Сумма | Количество | Сумма |
| Товар 1 | 5 | 35 | 4 | 28 |
| Товар 2 | 1 | 30 | 3 | 30 |

Итог: на складе 1 сумма товаров – 65, на складе 2 сумма товаров - 58

Реализация

//

// Модуль объекта

//

// функция получения остатков товаров, учитывая/не учитывая разрез складов и товаров

Функция ОстаткиТоваров(Дата, Склад = Неопределено, СО = Неопределено, ЗнакСравненияСклада = "=") Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК СО,

| ЖурналПроводокОстатки.Субконто2 КАК Склад,

| ЖурналПроводокОстатки.СуммаОстаток КАК Сумма,

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет, , " +

?(СО <> Неопределено, "Субконто1 = &СО", "") +

?(Склад <> Неопределено и СО <> Неопределено, " И ", "") +

?(Склад <> Неопределено, "Субконто2 " + ЗнакСравненияСклада + " &Склад", "") + ") КАК ЖурналПроводокОстатки

|

|УПОРЯДОЧИТЬ ПО

| ЖурналПроводокОстатки.Субконто1.Наименование,

| ЖурналПроводокОстатки.Субконто2.Наименование";

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("СО", СО);

Запрос.УстановитьПараметр("Склад", Склад);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Возврат Запрос.Выполнить().Выгрузить();

КонецФункции

// функция посчета средневзвешанной цены исходя из остатков на указанную дату

Функция СредневзвешаннаяЦена(Дата, СО) Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| СУММА(ЖурналПроводокОстатки.СуммаОстаток) КАК Сумма,

| СУММА(ЖурналПроводокОстатки.КоличествоОстаток) КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет, , Субконто1 = &СО) КАК ЖурналПроводокОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("СО", СО);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Результат = Запрос.Выполнить().Выгрузить();

Если Результат.Количество() = 0 Тогда

Возврат 0;

ИначеЕсли Результат[0].Сумма = 0 или Результат[0].Количество = 0 Тогда

Возврат 0;

Иначе

Возврат Результат[0].Сумма / Результат[0].Количество;

КонецЕсли;

КонецФункции

// функция возвращает остатки по складам, с разрезом по количеству и сумме (по складам)

Функция ОстаткиНаСкладах(Дата, УказыватьСО = Ложь, ВычислятьСумму = Истина, ВычислятьКоличество = Ложь, ВычислятьИтого = Истина) Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| " + ?(УказыватьСО, "ЖурналПроводокОстатки.Субконто1 КАК СО,", "") + "

| ЖурналПроводокОстатки.Субконто2 КАК Склад,

| " + ?(ВычислятьСумму, "СУММА(ЖурналПроводокОстатки.СуммаОстаток) КАК Сумма"

+ ?(ВычислятьКоличество,",",""), "") + "

| " + ?(ВычислятьКоличество, "ЖурналПроводокОстатки.КоличествоОстаток КАК Количество", "") + "

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет, , ) КАК ЖурналПроводокОстатки

|

|СГРУППИРОВАТЬ ПО

| " + ?(УказыватьСО, "ЖурналПроводокОстатки.Субконто1,", "") + "

| ЖурналПроводокОстатки.Субконто2,

| ЖурналПроводокОстатки.КоличествоОстаток" +

?(ВычислятьИтого и (ВычислятьКоличество или ВычислятьСумму), "

|ИТОГИ " +

?(ВычислятьСумму, "

| СУММА(Сумма)", "") +

?(ВычислятьКоличество, "

| СУММА(Количество)", "") + "

|ПО

| ОБЩИЕ", "");

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Возврат Запрос.Выполнить();

КонецФункции

// функция возвращает количественные остатки на дату в разрезе складов и товаров

Функция КоличествоНаСкладе(Дата, Склад, СО) Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(

| &Дата,

| Счет = &Счет,

| ,

| Субконто1 = &СО

| И Субконто2 = &Склад) КАК ЖурналПроводокОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Запрос.УстановитьПараметр("СО", СО);

Запрос.УстановитьПараметр("Склад", Склад);

Результат = Запрос.Выполнить().Выгрузить();

Если Результат.Количество() = 0 Тогда

Возврат 0;

Иначе

Возврат Результат[0].Количество;

КонецЕсли;

КонецФункции

//

// Модуль формы

//

Процедура КнопкаСформироватьНажатие(Кнопка)

// таблица средневзвешанных цен

СрЦ = Новый ТаблицаЗначений;

СрЦ.Колонки.Добавить("СО");

СрЦ.Колонки.Добавить("Цена");

СрЦ.Колонки.Добавить("Количество");

ТабДок = ЭлементыФормы.ПолеВывода;

ТабДок.Очистить();

// получение областей макета

Макет = ПолучитьМакет("ТекущиеОстатки");

ОбластьЗаголовка = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаОбщая = Макет.ПолучитьОбласть("Шапка|Общее");

ОбластьШапкаСклад = Макет.ПолучитьОбласть("Шапка|Склад");

ОбластьСтрокаОбщая = Макет.ПолучитьОбласть("Строка|Общее");

ОбластьСтрокаСклад = Макет.ПолучитьОбласть("Строка|Склад");

ОбластьИтогоОбщая = Макет.ПолучитьОбласть("Итого|Общее");

ОбластьИтогоСклад = Макет.ПолучитьОбласть("Итого|Склад");

ОбластьОбщееИтого = Макет.ПолучитьОбласть("ИтогоОбщее");

ОбластьРазрыв = Макет.ПолучитьОбласть("Разрыв");

// вывод заголовка первой таблицы

ОбластьЗаголовка.Параметры.Заголовок = "Запасы спецодежды на складах предприятия на " + Формат(Дата, "ДФ=dd.MM.yyyy");

ТабДок.Вывести(ОбластьЗаголовка);

// вывод всех складов

ТабДок.Вывести(ОбластьШапкаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

ОбластьШапкаСклад.Параметры.Склад = ВыборкаСкладов.Наименование;

ТабДок.Присоединить(ОбластьШапкаСклад);

КонецЦикла;

ВыборкаСО = Справочники.Материалы.Выбрать();

Пока ВыборкаСО.Следующий() Цикл

ОбластьСтрокаОбщая.Параметры.Спецодежда = ВыборкаСО.Ссылка;

// заполнения таблицы средневзвешанных цен

НовСтр = СрЦ.Добавить();

НовСтр.СО = ВыборкаСО.Ссылка;

НовСтр.Цена = СредневзвешаннаяЦена(Дата, ВыборкаСО.Ссылка);

ОбластьСтрокаОбщая.Параметры.Цена = НовСтр.Цена;

ТабДок.Вывести(ОбластьСтрокаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

// для каждой спецодежды каждого склада вычисляем остатки

Остатки = ОстаткиТоваров(Дата, ВыборкаСкладов.Ссылка, ВыборкаСО.Ссылка);

Если Остатки.Количество() = 0 Тогда

ОбластьСтрокаСклад.Параметры.Количество = 0;

ОбластьСтрокаСклад.Параметры.Стоимость = 0;

Иначе

ОбластьСтрокаСклад.Параметры.Количество = Остатки[0].Количество;

ОбластьСтрокаСклад.Параметры.Стоимость = Остатки[0].Сумма;

КонецЕсли;

// и выводим их в отчет

ТабДок.Присоединить(ОбластьСтрокаСклад);

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьИтогоОбщая);

// общая стоимостная оценка спецодежды на складах

СОСклада = Новый ТаблицаЗначений;

СОСклада.Колонки.Добавить("Склад");

СОСклада.Колонки.Добавить("Сумма");

// получаем остатки, не учитывая разрез по материалам и складам

ОстаткиСклады = ОстаткиНаСкладах(Дата).Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

ОстаткиСклады.Следующий();

// остатки по всем складам и всей СО (сумма остатков по всем складам)

ВсегоНаПредприятии = ОстаткиСклады.Сумма;

ОбластьОбщееИтого.Параметры.Стоимость = ОстаткиСклады.Сумма;

ОстаткиСкладов = ОстаткиСклады.Выбрать();

Пока ОстаткиСкладов.Следующий() Цикл

// заполняем таблицу общей стоиомостной оценки складов

НовСтр = СОСклада.Добавить();

НовСтр.Склад = ОстаткиСкладов.Склад;

НовСтр.Сумма = ОстаткиСкладов.Сумма;

// выводим общую сумму в отчет

ОбластьИтогоСклад.Параметры.Стоимость = ОстаткиСкладов.Сумма;

ТабДок.Присоединить(ОбластьИтогоСклад);

КонецЦикла;

// вывод "итого по предприятию"

ТабДок.Вывести(ОбластьОбщееИтого);

// сортируем таблицу средневзвешанных цен по обыванию цен

СрЦ.Сортировать("Цена Убыв");

// получаем количество, которое необходимо распределить на оставшиеся склады

Для Каждого Стр Из СрЦ Цикл

Стр.Количество = КоличествоНаСкладе(Дата, ЛиквидируемыйСклад, Стр.СО);

КонецЦикла;

// остатки на складах по количеству, без учета ликвидированного склада

ОстаткиСкл = ОстаткиТоваров(Дата, ЛиквидируемыйСклад, , "<>");

// массив для хранения сумм спецодежды после переоценки

МассивСО = Новый ТаблицаЗначений;

МассивСО.Колонки.Добавить("Склад");

МассивСО.Колонки.Добавить("Сумма");

МассивСО.Колонки.Добавить("ДляРаспределения");

// переоценка по средневзвешанной цене

Для Каждого Стр Из ОстаткиСкл Цикл

// добавление склада в массив, если его еще там нет

Если МассивСО.Найти(Стр.Склад) = Неопределено Тогда

НовСтрСкл = МассивСО.Добавить();

НовСтрСкл.Склад = Стр.Склад;

НовСтрСкл.Сумма = 0;

НовСтрСкл.ДляРаспределения = 0;

КонецЕсли;

Для Каждого СтрСО Из СрЦ Цикл

Если Стр.СО = СтрСО.СО Тогда

Стр.Сумма = Стр.Количество \* СтрСО.Цена;

// получаем общую стоимостную оценку по складам после переоценки (с учетом средневзвешанных цен)

МассивСО.Найти(Стр.Склад, "Склад").Сумма = МассивСО.Найти(Стр.Склад, "Склад").Сумма + Стр.Сумма;

Прервать;

КонецЕсли;

КонецЦикла;

КонецЦикла;

// вычисляем сумму, которой не хватает до средней стоимостной оценки всех товаров на каждом складе после ликвидации

Для Каждого Стр Из МассивСО Цикл

Стр.ДляРаспределения = ВсегоНаПредприятии / МассивСО.Количество() - Стр.Сумма;

КонецЦикла;

МассивСО.Сортировать("ДляРаспределения Убыв");

Распределение = Новый ТаблицаЗначений;

Распределение.Колонки.Добавить("Склад");

Распределение.Колонки.Добавить("СО");

Распределение.Колонки.Добавить("Количество");

Распределение.Колонки.Добавить("Сумма");

//поиск оптимального перемещения товаров с ликвидируемого склада на оставшиеся два

//////////////////////////////////////////////////////////////////////////////////

//Для поиска оптимального варианта размещения спецодежды ликвидируемого склада и

//оставшихся двух складов использую следующий алогритм: сначала сортируем склады

//по убыванию по сумме, которой не хватает до средней стоимостной оценки (для этого

//общую сумму по предприятию делим на два и вычитаем из нее стоимостную оценку после

//переоценки находящуюся на каждом складе). Затем, чтобы отклонение от стоимостной

//оценки, которую необходимо набрать было минимальным, начинаем перемещение с самых

//дорогих (по переоценочным ценам) товарам, что дает возможность заполнить и маленькое

//отклонение, перемещая товары от дорогих к дешевым. При этом заполняем таблицу

//для посчета суммы и количества перемещенных товаров на каждый склад.

/////////////////////////////////////////////////////////////////////////////////

Для Каждого Стр Из МассивСО Цикл

Для Каждого СтрСО Из СрЦ Цикл

Пока СтрСО.Количество > 0 Цикл

Если Стр.ДляРаспределения - СтрСО.Цена < 0 Тогда

Прервать;

КонецЕсли;

СтрСО.Количество = СтрСО.Количество - 1;

Стр.ДляРаспределения = Стр.ДляРаспределения - СтрСО.Цена;

Стр.Сумма = Стр.Сумма + СтрСО.Цена;

НовСтр = Распределение.Добавить();

НовСтр.Склад = Стр.Склад;

НовСтр.СО = СтрСО.СО;

НовСтр.Количество = 1;

НовСтр.Сумма = СтрСО.Цена;

КонецЦикла;

КонецЦикла;

КонецЦикла;

// если не все до конца распределилось, то ищем куда лучше отнести последний товар, если он есть

ОстатокРаспределения = МассивСО.Итог("ДляРаспределения");

СрЦ.Сортировать("Количество Убыв, Цена Убыв");

МассивСО.Сортировать("Сумма Убыв");

Для Каждого Стр Из СрЦ Цикл

// выбираем товар, который остался (если остался)

Если Стр.Количество > 0 Тогда

Стр.Количество = Стр.Количество - 1;

МассивСО[1].ДляРаспределения = МассивСО[1].ДляРаспределения - Стр.Цена;

МассивСО[1].Сумма = МассивСО[1].Сумма + Стр.Цена;

// и выбираем для перемещения тот склад, добавив в который отклонение будет минимальным

НовСтр = Распределение.Добавить();

НовСтр.Склад = МассивСО[1].Склад;

НовСтр.СО = Стр.СО;

НовСтр.Количество = 1;

НовСтр.Сумма = Стр.Цена;

КонецЕсли;

КонецЦикла;

// сворачиваем таблицу с распределенными товарами, для посчета суммы и количества

Распределение.Свернуть("Склад, СО", "Количество, Сумма");

Распределение.Сортировать("Склад Возр, СО Возр");

// вывод разрыва между отчетами

ТабДок.Вывести(ОбластьРазрыв);

// вывод заголовка второй таблицы

ОбластьЗаголовка.Параметры.Заголовок = "Предлагаемый вариант размещения спецодежды на двух складах";

ТабДок.Вывести(ОбластьЗаголовка);

// выводим шапку таблицы для каждого склада

ТабДок.Вывести(ОбластьШапкаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

Если ВыборкаСкладов.Ссылка <> ЛиквидируемыйСклад Тогда

ОбластьШапкаСклад.Параметры.Склад = ВыборкаСкладов.Наименование;

ТабДок.Присоединить(ОбластьШапкаСклад);

КонецЕсли;

КонецЦикла;

// массив общей стоимостной оценки по складам после ликвидации вабранного склада

ОбщиеИтоговыеСуммы = Новый Массив;

ОбщиеИтоговыеСуммы.Добавить(0);

ОбщиеИтоговыеСуммы.Добавить(0);

// вывод спецодежды по складам (по количеству и сумме)

ВыборкаСО = Справочники.Материалы.Выбрать();

Пока ВыборкаСО.Следующий() Цикл

ОбластьСтрокаОбщая.Параметры.Спецодежда = ВыборкаСО.Ссылка;

ОбластьСтрокаОбщая.Параметры.Цена = СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

ТабДок.Вывести(ОбластьСтрокаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

н = 0;

Пока ВыборкаСкладов.Следующий() Цикл

Если ВыборкаСкладов.Ссылка <> ЛиквидируемыйСклад Тогда

Остатки = ОстаткиТоваров(Дата, ВыборкаСкладов.Ссылка, ВыборкаСО.Ссылка);

Отбор = Новый Структура;

Отбор.Вставить("СО", ВыборкаСО.Ссылка);

Отбор.Вставить("Склад", ВыборкаСкладов.Ссылка);

РаспределеннаяСО = Распределение.НайтиСтроки(Отбор);

// ищем остатки на тех складах, которые не ликвидировали

// количество считаем как остатки + распределенные товары на данный склад

Если Остатки.Количество() = 0 Тогда

ОбщееКоличество = ?(РаспределеннаяСО.Количество() = 0, 0, РаспределеннаяСО[0].Количество);

Иначе

ОбщееКоличество = Остатки[0].Количество + ?(РаспределеннаяСО.Количество() = 0, 0, РаспределеннаяСО[0].Количество);

КонецЕсли;

ОбластьСтрокаСклад.Параметры.Количество = ОбщееКоличество;

ОбщиеИтоговыеСуммы[н] = ОбщиеИтоговыеСуммы[н] + ОбщееКоличество \* СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

// сумму считаем, как количество (посчитанное выше) \* на цену (после переоценки по средневзвешанной цене)

ОбластьСтрокаСклад.Параметры.Стоимость = ОбщееКоличество \* СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

ТабДок.Присоединить(ОбластьСтрокаСклад);

н = н + 1;

КонецЕсли;

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьИтогоОбщая);

// вывод общих итоговых сумм по складам

Для н = 0 по ОбщиеИтоговыеСуммы.Количество() - 1 Цикл

// выводим общую сумму в отчет

ОбластьИтогоСклад.Параметры.Стоимость = ОбщиеИтоговыеСуммы[н];

ТабДок.Присоединить(ОбластьИтогоСклад);

КонецЦикла;

КонецПроцедуры

Процедура ПриОткрытии()

// установка даты по умолчанию (1 апреля 2012 года)

Дата = '20120401';

// установка склада по умолчанию (склад, с максимальным значением кода)

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 1

| Склады.Ссылка КАК Склад

|ИЗ

| Справочник.Склады КАК Склады

|

|УПОРЯДОЧИТЬ ПО

| Склады.Код УБЫВ";

Результат = Запрос.Выполнить().Выгрузить();

Если Результат.Количество() <> 0 Тогда

ЛиквидируемыйСклад = Результат[0].Склад;

КонецЕсли;

КонецПроцедуры

Автор: Завладаев Алексей

Пояснения к коду:

// функция получения остатков товаров, учитывая/не учитывая разрез складов и товаров

Функция ОстаткиТоваров(Дата, Склад = Неопределено, СО = Неопределено, ЗнакСравненияСклада = "=") Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК СО,

| ЖурналПроводокОстатки.Субконто2 КАК Склад,

| ЖурналПроводокОстатки.СуммаОстаток КАК Сумма,

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет, , " +

?(СО <> Неопределено, "Субконто1 = &СО", "") +

?(Склад <> Неопределено и СО <> Неопределено, " И ", "") +

?(Склад <> Неопределено, "Субконто2 " + ЗнакСравненияСклада + " &Склад", "") + ") КАК ЖурналПроводокОстатки

|

|УПОРЯДОЧИТЬ ПО

| ЖурналПроводокОстатки.Субконто1.Наименование,

| ЖурналПроводокОстатки.Субконто2.Наименование";

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("СО", СО);

Запрос.УстановитьПараметр("Склад", Склад);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Возврат Запрос.Выполнить().Выгрузить();

КонецФункции

Данная функция расположена в модуле объекта, она требуется для получения остатков из счета Материалы всей спецодежды, либо определенной, по всем складам, либо по определенному, с возможностью выбора складов по разному условию (либо один из всех, либо все кроме одного "="/"<>"). Функция возвращает выгрузку результата запроса. При составлении текста запроса используются тернарные условные операции – конструкции вида (условие?при выполнении:при не выполнении). Они используются для уменьшения дублирующего кода и объема кода в целом. Разберем одно из них:

?(СО <> Неопределено, "Субконто1 = &СО", "")

В данной конструкции "СО <> Неопределено" – условие, если условие выполняется, т.е. СО не равно Неопределено, то конструкция вернет "Субконто1 = &СО", если условие не выполнится то будет возвращено "".

// функция посчета средневзвешанной цены исходя из остатков на указанную дату

Функция СредневзвешаннаяЦена(Дата, СО) Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| СУММА(ЖурналПроводокОстатки.СуммаОстаток) КАК Сумма,

| СУММА(ЖурналПроводокОстатки.КоличествоОстаток) КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет, , Субконто1 = &СО) КАК ЖурналПроводокОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("СО", СО);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Результат = Запрос.Выполнить().Выгрузить();

Если Результат.Количество() = 0 Тогда

Возврат 0;

ИначеЕсли Результат[0].Сумма = 0 или Результат[0].Количество = 0 Тогда

Возврат 0;

Иначе

Возврат Результат[0].Сумма / Результат[0].Количество;

КонецЕсли;

КонецФункции

Данная функция расположена в модуле объекта, она требуется для получения средневзвешанной цены спецодежды на определенную дату, для этого получается сумма остатков и количество остатков указанной спецодежды, на указанную дату из счета Материалы. Далее возвращается средневзвешанная цена (Сумма / Количество), с проверкой на Количество = 0 (чтобы предотвратить деление на ноль).

// функция возвращает остатки по складам, с разрезом по количеству и сумме (по складам)

Функция ОстаткиНаСкладах(Дата, УказыватьСО = Ложь, ВычислятьСумму = Истина, ВычислятьКоличество = Ложь, ВычислятьИтого = Истина) Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| " + ?(УказыватьСО, "ЖурналПроводокОстатки.Субконто1 КАК СО,", "") + "

| ЖурналПроводокОстатки.Субконто2 КАК Склад,

| " + ?(ВычислятьСумму, "СУММА(ЖурналПроводокОстатки.СуммаОстаток) КАК Сумма"

+ ?(ВычислятьКоличество,",",""), "") + "

| " + ?(ВычислятьКоличество, "ЖурналПроводокОстатки.КоличествоОстаток КАК Количество", "") + "

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет, , ) КАК ЖурналПроводокОстатки

|

|СГРУППИРОВАТЬ ПО

| " + ?(УказыватьСО, "ЖурналПроводокОстатки.Субконто1,", "") + "

| ЖурналПроводокОстатки.Субконто2,

| ЖурналПроводокОстатки.КоличествоОстаток" +

?(ВычислятьИтого и (ВычислятьКоличество или ВычислятьСумму), "

|ИТОГИ " +

?(ВычислятьСумму, "

| СУММА(Сумма)", "") +

?(ВычислятьКоличество, "

| СУММА(Количество)", "") + "

|ПО

| ОБЩИЕ", "");

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Возврат Запрос.Выполнить();

КонецФункции

Данная функция расположена в модуле объекта, она требуется для получения остатков на складах, на указанную дату, с настройкой параметров – включать в выборку спецодежду или нет, вычислять сумму по складам или нет, вычислять количество или нет, вычислять итоги или нет. По умолчанию спецодежда не указывается, сумма вычисляется, количество нет, итоги вычисляются. Для составления текста запроса используются тернарные условные операции. Возвращает функция результат запроса, готовый для обхода.

// функция возвращает количественные остатки на дату в разрезе складов и товаров

Функция КоличествоНаСкладе(Дата, Склад, СО) Экспорт

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(

| &Дата,

| Счет = &Счет,

| ,

| Субконто1 = &СО

| И Субконто2 = &Склад) КАК ЖурналПроводокОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Запрос.УстановитьПараметр("СО", СО);

Запрос.УстановитьПараметр("Склад", Склад);

Результат = Запрос.Выполнить().Выгрузить();

Если Результат.Количество() = 0 Тогда

Возврат 0;

Иначе

Возврат Результат[0].Количество;

КонецЕсли;

КонецФункции

Данная функция расположена в модуле объекта, она требуется для получения количества спецодежды на определенном складе, на определенную дату.

Процедура КнопкаСформироватьНажатие(Кнопка)

// таблица средневзвешанных цен

СрЦ = Новый ТаблицаЗначений;

СрЦ.Колонки.Добавить("СО");

СрЦ.Колонки.Добавить("Цена");

СрЦ.Колонки.Добавить("Количество");

ТабДок = ЭлементыФормы.ПолеВывода;

ТабДок.Очистить();

// получение областей макета

Макет = ПолучитьМакет("ТекущиеОстатки");

ОбластьЗаголовка = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаОбщая = Макет.ПолучитьОбласть("Шапка|Общее");

ОбластьШапкаСклад = Макет.ПолучитьОбласть("Шапка|Склад");

ОбластьСтрокаОбщая = Макет.ПолучитьОбласть("Строка|Общее");

ОбластьСтрокаСклад = Макет.ПолучитьОбласть("Строка|Склад");

ОбластьИтогоОбщая = Макет.ПолучитьОбласть("Итого|Общее");

ОбластьИтогоСклад = Макет.ПолучитьОбласть("Итого|Склад");

ОбластьОбщееИтого = Макет.ПолучитьОбласть("ИтогоОбщее");

ОбластьРазрыв = Макет.ПолучитьОбласть("Разрыв");

// вывод заголовка первой таблицы

ОбластьЗаголовка.Параметры.Заголовок = "Запасы спецодежды на складах предприятия на " + Формат(Дата, "ДФ=dd.MM.yyyy");

ТабДок.Вывести(ОбластьЗаголовка);

// вывод всех складов

ТабДок.Вывести(ОбластьШапкаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

ОбластьШапкаСклад.Параметры.Склад = ВыборкаСкладов.Наименование;

ТабДок.Присоединить(ОбластьШапкаСклад);

КонецЦикла;

ВыборкаСО = Справочники.Материалы.Выбрать();

Пока ВыборкаСО.Следующий() Цикл

ОбластьСтрокаОбщая.Параметры.Спецодежда = ВыборкаСО.Ссылка;

// заполнения таблицы средневзвешанных цен

НовСтр = СрЦ.Добавить();

НовСтр.СО = ВыборкаСО.Ссылка;

НовСтр.Цена = СредневзвешаннаяЦена(Дата, ВыборкаСО.Ссылка);

ОбластьСтрокаОбщая.Параметры.Цена = НовСтр.Цена;

ТабДок.Вывести(ОбластьСтрокаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

// для каждой спецодежды каждого склада вычисляем остатки

Остатки = ОстаткиТоваров(Дата, ВыборкаСкладов.Ссылка, ВыборкаСО.Ссылка);

Если Остатки.Количество() = 0 Тогда

ОбластьСтрокаСклад.Параметры.Количество = 0;

ОбластьСтрокаСклад.Параметры.Стоимость = 0;

Иначе

ОбластьСтрокаСклад.Параметры.Количество = Остатки[0].Количество;

ОбластьСтрокаСклад.Параметры.Стоимость = Остатки[0].Сумма;

КонецЕсли;

// и выводим их в отчет

ТабДок.Присоединить(ОбластьСтрокаСклад);

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьИтогоОбщая);

// общая стоимостная оценка спецодежды на складах

СОСклада = Новый ТаблицаЗначений;

СОСклада.Колонки.Добавить("Склад");

СОСклада.Колонки.Добавить("Сумма");

// получаем остатки, не учитывая разрез по материалам и складам

ОстаткиСклады = ОстаткиНаСкладах(Дата).Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

ОстаткиСклады.Следующий();

// остатки по всем складам и всей СО (сумма остатков по всем складам)

ВсегоНаПредприятии = ОстаткиСклады.Сумма;

ОбластьОбщееИтого.Параметры.Стоимость = ОстаткиСклады.Сумма;

ОстаткиСкладов = ОстаткиСклады.Выбрать();

Пока ОстаткиСкладов.Следующий() Цикл

// заполняем таблицу общей стоиомостной оценки складов

НовСтр = СОСклада.Добавить();

НовСтр.Склад = ОстаткиСкладов.Склад;

НовСтр.Сумма = ОстаткиСкладов.Сумма;

// выводим общую сумму в отчет

ОбластьИтогоСклад.Параметры.Стоимость = ОстаткиСкладов.Сумма;

ТабДок.Присоединить(ОбластьИтогоСклад);

КонецЦикла;

// вывод "итого по предприятию"

ТабДок.Вывести(ОбластьОбщееИтого);

// сортируем таблицу средневзвешанных цен по обыванию цен

СрЦ.Сортировать("Цена Убыв");

// получаем количество, которое необходимо распределить на оставшиеся склады

Для Каждого Стр Из СрЦ Цикл

Стр.Количество = КоличествоНаСкладе(Дата, ЛиквидируемыйСклад, Стр.СО);

КонецЦикла;

// остатки на складах по количеству, без учета ликвидированного склада

ОстаткиСкл = ОстаткиТоваров(Дата, ЛиквидируемыйСклад, , "<>");

// массив для хранения сумм спецодежды после переоценки

МассивСО = Новый ТаблицаЗначений;

МассивСО.Колонки.Добавить("Склад");

МассивСО.Колонки.Добавить("Сумма");

МассивСО.Колонки.Добавить("ДляРаспределения");

// переоценка по средневзвешанной цене

Для Каждого Стр Из ОстаткиСкл Цикл

// добавление склада в массив, если его еще там нет

Если МассивСО.Найти(Стр.Склад) = Неопределено Тогда

НовСтрСкл = МассивСО.Добавить();

НовСтрСкл.Склад = Стр.Склад;

НовСтрСкл.Сумма = 0;

НовСтрСкл.ДляРаспределения = 0;

КонецЕсли;

Для Каждого СтрСО Из СрЦ Цикл

Если Стр.СО = СтрСО.СО Тогда

Стр.Сумма = Стр.Количество \* СтрСО.Цена;

// получаем общую стоимостную оценку по складам после переоценки (с учетом средневзвешанных цен)

МассивСО.Найти(Стр.Склад, "Склад").Сумма = МассивСО.Найти(Стр.Склад, "Склад").Сумма + Стр.Сумма;

Прервать;

КонецЕсли;

КонецЦикла;

КонецЦикла;

// вычисляем сумму, которой не хватает до средней стоимостной оценки всех товаров на каждом складе после ликвидации

Для Каждого Стр Из МассивСО Цикл

Стр.ДляРаспределения = ВсегоНаПредприятии / МассивСО.Количество() - Стр.Сумма;

КонецЦикла;

МассивСО.Сортировать("ДляРаспределения Убыв");

Распределение = Новый ТаблицаЗначений;

Распределение.Колонки.Добавить("Склад");

Распределение.Колонки.Добавить("СО");

Распределение.Колонки.Добавить("Количество");

Распределение.Колонки.Добавить("Сумма");

//поиск оптимального перемещения товаров с ликвидируемого склада на оставшиеся два

//////////////////////////////////////////////////////////////////////////////////

//Для поиска оптимального варианта размещения спецодежды ликвидируемого склада и

//оставшихся двух складов использую следующий алогритм: сначала сортируем склады

//по убыванию по сумме, которой не хватает до средней стоимостной оценки (для этого

//общую сумму по предприятию делим на два и вычитаем из нее стоимостную оценку после

//переоценки находящуюся на каждом складе). Затем, чтобы отклонение от стоимостной

//оценки, которую необходимо набрать было минимальным, начинаем перемещение с самых

//дорогих (по переоценочным ценам) товарам, что дает возможность заполнить и маленькое

//отклонение, перемещая товары от дорогих к дешевым. При этом заполняем таблицу

//для посчета суммы и количества перемещенных товаров на каждый склад.

/////////////////////////////////////////////////////////////////////////////////

Для Каждого Стр Из МассивСО Цикл

Для Каждого СтрСО Из СрЦ Цикл

Пока СтрСО.Количество > 0 Цикл

Если Стр.ДляРаспределения - СтрСО.Цена < 0 Тогда

Прервать;

КонецЕсли;

СтрСО.Количество = СтрСО.Количество - 1;

Стр.ДляРаспределения = Стр.ДляРаспределения - СтрСО.Цена;

Стр.Сумма = Стр.Сумма + СтрСО.Цена;

НовСтр = Распределение.Добавить();

НовСтр.Склад = Стр.Склад;

НовСтр.СО = СтрСО.СО;

НовСтр.Количество = 1;

НовСтр.Сумма = СтрСО.Цена;

КонецЦикла;

КонецЦикла;

КонецЦикла;

// если не все до конца распределилось, то ищем куда лучше отнести последний товар, если он есть

ОстатокРаспределения = МассивСО.Итог("ДляРаспределения");

СрЦ.Сортировать("Количество Убыв, Цена Убыв");

МассивСО.Сортировать("Сумма Убыв");

Для Каждого Стр Из СрЦ Цикл

// выбираем товар, который остался (если остался)

Если Стр.Количество > 0 Тогда

Стр.Количество = Стр.Количество - 1;

МассивСО[1].ДляРаспределения = МассивСО[1].ДляРаспределения - Стр.Цена;

МассивСО[1].Сумма = МассивСО[1].Сумма + Стр.Цена;

// и выбираем для перемещения тот склад, добавив в который отклонение будет минимальным

НовСтр = Распределение.Добавить();

НовСтр.Склад = МассивСО[1].Склад;

НовСтр.СО = Стр.СО;

НовСтр.Количество = 1;

НовСтр.Сумма = Стр.Цена;

КонецЕсли;

КонецЦикла;

// сворачиваем таблицу с распределенными товарами, для посчета суммы и количества

Распределение.Свернуть("Склад, СО", "Количество, Сумма");

Распределение.Сортировать("Склад Возр, СО Возр");

// вывод разрыва между отчетами

ТабДок.Вывести(ОбластьРазрыв);

// вывод заголовка второй таблицы

ОбластьЗаголовка.Параметры.Заголовок = "Предлагаемый вариант размещения спецодежды на двух складах";

ТабДок.Вывести(ОбластьЗаголовка);

// выводим шапку таблицы для каждого склада

ТабДок.Вывести(ОбластьШапкаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

Если ВыборкаСкладов.Ссылка <> ЛиквидируемыйСклад Тогда

ОбластьШапкаСклад.Параметры.Склад = ВыборкаСкладов.Наименование;

ТабДок.Присоединить(ОбластьШапкаСклад);

КонецЕсли;

КонецЦикла;

// массив общей стоимостной оценки по складам после ликвидации вабранного склада

ОбщиеИтоговыеСуммы = Новый Массив;

ОбщиеИтоговыеСуммы.Добавить(0);

ОбщиеИтоговыеСуммы.Добавить(0);

// вывод спецодежды по складам (по количеству и сумме)

ВыборкаСО = Справочники.Материалы.Выбрать();

Пока ВыборкаСО.Следующий() Цикл

ОбластьСтрокаОбщая.Параметры.Спецодежда = ВыборкаСО.Ссылка;

ОбластьСтрокаОбщая.Параметры.Цена = СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

ТабДок.Вывести(ОбластьСтрокаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

н = 0;

Пока ВыборкаСкладов.Следующий() Цикл

Если ВыборкаСкладов.Ссылка <> ЛиквидируемыйСклад Тогда

Остатки = ОстаткиТоваров(Дата, ВыборкаСкладов.Ссылка, ВыборкаСО.Ссылка);

Отбор = Новый Структура;

Отбор.Вставить("СО", ВыборкаСО.Ссылка);

Отбор.Вставить("Склад", ВыборкаСкладов.Ссылка);

РаспределеннаяСО = Распределение.НайтиСтроки(Отбор);

// ищем остатки на тех складах, которые не ликвидировали

// количество считаем как остатки + распределенные товары на данный склад

Если Остатки.Количество() = 0 Тогда

ОбщееКоличество = ?(РаспределеннаяСО.Количество() = 0, 0, РаспределеннаяСО[0].Количество);

Иначе

ОбщееКоличество = Остатки[0].Количество + ?(РаспределеннаяСО.Количество() = 0, 0, РаспределеннаяСО[0].Количество);

КонецЕсли;

ОбластьСтрокаСклад.Параметры.Количество = ОбщееКоличество;

ОбщиеИтоговыеСуммы[н] = ОбщиеИтоговыеСуммы[н] + ОбщееКоличество \* СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

// сумму считаем, как количество (посчитанное выше) \* на цену (после переоценки по средневзвешанной цене)

ОбластьСтрокаСклад.Параметры.Стоимость = ОбщееКоличество \* СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

ТабДок.Присоединить(ОбластьСтрокаСклад);

н = н + 1;

КонецЕсли;

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьИтогоОбщая);

// вывод общих итоговых сумм по складам

Для н = 0 по ОбщиеИтоговыеСуммы.Количество() - 1 Цикл

// выводим общую сумму в отчет

ОбластьИтогоСклад.Параметры.Стоимость = ОбщиеИтоговыеСуммы[н];

ТабДок.Присоединить(ОбластьИтогоСклад);

КонецЦикла;

КонецПроцедуры

Основная процедура, вызываемая при нажатии на кнопку «Сформировать отчет», расположена в модуле формы. В ней реализовано формирование документа-результата и распределение товаров с ликвидируемого склада на остальные.

// таблица средневзвешанных цен

СрЦ = Новый ТаблицаЗначений;

СрЦ.Колонки.Добавить("СО");

СрЦ.Колонки.Добавить("Цена");

СрЦ.Колонки.Добавить("Количество");

ТабДок = ЭлементыФормы.ПолеВывода;

ТабДок.Очистить();

В данном блоке создается таблица значений для хранения средневзвешанных цен, с колонками СО (Спецодежда), Цена и Количество. Далее поле вывода для отображения результата записывается в переменную ТабДок и очищается.

// получение областей макета

Макет = ПолучитьМакет("ТекущиеОстатки");

ОбластьЗаголовка = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаОбщая = Макет.ПолучитьОбласть("Шапка|Общее");

ОбластьШапкаСклад = Макет.ПолучитьОбласть("Шапка|Склад");

ОбластьСтрокаОбщая = Макет.ПолучитьОбласть("Строка|Общее");

ОбластьСтрокаСклад = Макет.ПолучитьОбласть("Строка|Склад");

ОбластьИтогоОбщая = Макет.ПолучитьОбласть("Итого|Общее");

ОбластьИтогоСклад = Макет.ПолучитьОбласть("Итого|Склад");

ОбластьОбщееИтого = Макет.ПолучитьОбласть("ИтогоОбщее");

ОбластьРазрыв = Макет.ПолучитьОбласть("Разрыв");

В данном блоке проводится получение областей макета, для вывода результата.

// вывод заголовка первой таблицы

ОбластьЗаголовка.Параметры.Заголовок = "Запасы спецодежды на складах предприятия на " + Формат(Дата, "ДФ=dd.MM.yyyy");

ТабДок.Вывести(ОбластьЗаголовка);

// вывод всех складов

ТабДок.Вывести(ОбластьШапкаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

ОбластьШапкаСклад.Параметры.Склад = ВыборкаСкладов.Наименование;

ТабДок.Присоединить(ОбластьШапкаСклад);

КонецЦикла;

В данном блоке в табличный документ результата выводится заголовок и шапка таблицы – "Спецодежда","Цена","Склад 1","Склад 2","Склад 3". При выводе складов просто обходим выборку из складов, получаемую через Справочники.Склады.Выбрать().

ВыборкаСО = Справочники.Материалы.Выбрать();

Пока ВыборкаСО.Следующий() Цикл

ОбластьСтрокаОбщая.Параметры.Спецодежда = ВыборкаСО.Ссылка;

// заполнения таблицы средневзвешанных цен

НовСтр = СрЦ.Добавить();

НовСтр.СО = ВыборкаСО.Ссылка;

НовСтр.Цена = СредневзвешаннаяЦена(Дата, ВыборкаСО.Ссылка);

ОбластьСтрокаОбщая.Параметры.Цена = НовСтр.Цена;

ТабДок.Вывести(ОбластьСтрокаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

// для каждой спецодежды каждого склада вычисляем остатки

Остатки = ОстаткиТоваров(Дата, ВыборкаСкладов.Ссылка, ВыборкаСО.Ссылка);

Если Остатки.Количество() = 0 Тогда

ОбластьСтрокаСклад.Параметры.Количество = 0;

ОбластьСтрокаСклад.Параметры.Стоимость = 0;

Иначе

ОбластьСтрокаСклад.Параметры.Количество = Остатки[0].Количество;

ОбластьСтрокаСклад.Параметры.Стоимость = Остатки[0].Сумма;

КонецЕсли;

// и выводим их в отчет

ТабДок.Присоединить(ОбластьСтрокаСклад);

КонецЦикла;

КонецЦикла;

В данном блоке заполняется таблица значений средневзвешанных цен. Для этого берется выборка всей спецодежды (через Справочники.Материалы.Выбрать). Происходит обход выборки, с выводом в табличный документ.

// заполнения таблицы средневзвешанных цен

НовСтр = СрЦ.Добавить();

НовСтр.СО = ВыборкаСО.Ссылка;

НовСтр.Цена = СредневзвешаннаяЦена(Дата, ВыборкаСО.Ссылка);

Данный блок используется для заполнения таблицы значений. СрЦ.Добавить – создает новую строку в таблице значений и возвращает её для заполнения. Далее данные строки заполняются: СО – ссылкой на материал, а цена – значением возвращенным функцией СредневзвешаннаяЦена на указанную дату (указанная в форме) и для текущего материала.

После добавления новой записи в таблицу значений делается вывод общей части таблицы (Спецодежда и Цена). После этого берется выборка складов (через Справочники.Склады.Выбрать) и обходится, с высчитыванием остатков для каждого склада, используя функцию ОстаткиТоваров на указанную дату, для текущего склада и текущего материала. После получения остатков Количество и Сумма выводятся в табличный документ результата.

ТабДок.Вывести(ОбластьИтогоОбщая);

// общая стоимостная оценка спецодежды на складах

СОСклада = Новый ТаблицаЗначений;

СОСклада.Колонки.Добавить("Склад");

СОСклада.Колонки.Добавить("Сумма");

// получаем остатки, не учитывая разрез по материалам и складам

ОстаткиСклады = ОстаткиНаСкладах(Дата).Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

ОстаткиСклады.Следующий();

// остатки по всем складам и всей СО (сумма остатков по всем складам)

ВсегоНаПредприятии = ОстаткиСклады.Сумма;

ОбластьОбщееИтого.Параметры.Стоимость = ОстаткиСклады.Сумма;

ОстаткиСкладов = ОстаткиСклады.Выбрать();

Пока ОстаткиСкладов.Следующий() Цикл

// заполняем таблицу общей стоиомостной оценки складов

НовСтр = СОСклада.Добавить();

НовСтр.Склад = ОстаткиСкладов.Склад;

НовСтр.Сумма = ОстаткиСкладов.Сумма;

// выводим общую сумму в отчет

ОбластьИтогоСклад.Параметры.Стоимость = ОстаткиСкладов.Сумма;

ТабДок.Присоединить(ОбластьИтогоСклад);

КонецЦикла;

// вывод "итого по предприятию"

ТабДок.Вывести(ОбластьОбщееИтого);

В данном блоке выводятся итоги по каждому складу и общий итог по предприятию. Для хранения информации по остаткам на складах создается таблица значений СОСклада с колонками Склад и Сумма. Для подсчета итогов по каждому складу используется функция ОстаткиНаСкладах, с указанием даты указанной в форме, остальные параметры функции берутся по умолчанию (при вызове не указываются). Так как функция возвращает результат запроса, для обхода данных требуется получить выборку, используя Выбрать, и для получения результата по всему предприятию и каждому складу используется ОбходРезультатаЗапроса.ПоГруппировкам. Тогда первым элементом выборки будет общая сумма на предприятии, которая записывается в переменную ВсегоНаПредприятии. А далее получаются остатки по каждому складу, для этого берется выборка (через Выбрать) без указания обхода и при обходе выборки заполняется таблица значений СОСклада и выводится в табличный документ. По завершению в табличный документ выводится общий итог по предприятию. На этом завершается первая часть задачи – вывести текущие остатки спецодежды на предприятии.

// сортируем таблицу средневзвешанных цен по обыванию цен

СрЦ.Сортировать("Цена Убыв");

// получаем количество, которое необходимо распределить на оставшиеся склады

Для Каждого Стр Из СрЦ Цикл

Стр.Количество = КоличествоНаСкладе(Дата, ЛиквидируемыйСклад, Стр.СО);

КонецЦикла;

В данном блоке таблица значений средневзвешанных цен сортируется по убыванию цены, чтобы самая дорогая спецодежда шла в начале, а та что дешевле в конце (для распределения на склады). После сортировки заполняется последняя колонка таблицы значений – Количество. В данное поле будет записано количество товара на ликвидируемом складе, которое предстоит распределить на другие склады. Для получения этого количества используется функция КоличествоНаСкладе с указанием даты, ликвидируемого склада (и дата и склад указываются в форме) и спецодежды для которой считается количество.

// остатки на складах по количеству, без учета ликвидированного склада

ОстаткиСкл = ОстаткиТоваров(Дата, ЛиквидируемыйСклад, , "<>");

// массив для хранения сумм спецодежды после переоценки

МассивСО = Новый ТаблицаЗначений;

МассивСО.Колонки.Добавить("Склад");

МассивСО.Колонки.Добавить("Сумма");

МассивСО.Колонки.Добавить("ДляРаспределения");

// переоценка по средневзвешанной цене

Для Каждого Стр Из ОстаткиСкл Цикл

// добавление склада в массив, если его еще там нет

Если МассивСО.Найти(Стр.Склад) = Неопределено Тогда

НовСтрСкл = МассивСО.Добавить();

НовСтрСкл.Склад = Стр.Склад;

НовСтрСкл.Сумма = 0;

НовСтрСкл.ДляРаспределения = 0;

КонецЕсли;

Для Каждого СтрСО Из СрЦ Цикл

Если Стр.СО = СтрСО.СО Тогда

Стр.Сумма = Стр.Количество \* СтрСО.Цена;

// получаем общую стоимостную оценку по складам после переоценки (с учетом средневзвешанных цен)

МассивСО.Найти(Стр.Склад, "Склад").Сумма = МассивСО.Найти(Стр.Склад, "Склад").Сумма + Стр.Сумма;

Прервать;

КонецЕсли;

КонецЦикла;

КонецЦикла;

// вычисляем сумму, которой не хватает до средней стоимостной оценки всех товаров на каждом складе после ликвидации

Для Каждого Стр Из МассивСО Цикл

Стр.ДляРаспределения = ВсегоНаПредприятии / МассивСО.Количество() - Стр.Сумма;

КонецЦикла;

МассивСО.Сортировать("ДляРаспределения Убыв");

В данном блоке происходит переоценка сумм остатков на остающихся складах. Для хранения данных создается таблица значений МассивСО с колонками Склад, Сумма, ДляРаспределения. В колонке ДляРаспределения будет храниться сумма, которой не хватает до средней стоимостной оценки после ликвидации. Для получения остатков используется функция ОстаткиТоваров с указанием даты и ликвидируемого склада (указанные в форме), и с условием "<>" – что означает, что остатки берутся для всех складов кроме ликвидируемого. Полученные остатки обходятся, если склад на котором хранится текущий материал не найден в таблице значений МассивСО – то в таблицу значений добавляется новая строка с этим складом. Далее в таблице значений СрЦ (средневзвешанные цены) ищется строка хранящая данные о текущей спецодежде, для получения средневзвешанной цены. Когда запись найдена – вычисляется сумма остатков на складе (количество на складе \* средневзвешанную цену материала), добавлется сумма этого материала к общей сумме по складу в таблице значений МассивСО и цикл прерывается, т.к. запись уже найдена. После обхода остатков для всех записей таблицы значений МассивСО высчитывается сумма ДляРаспределения по формуле ВсегоНаПредприятии / количество складов - сумма на складе, после чего таблица значений сортируется по убыванию колонки ДляРаспределения (сначала будет заполнен склад, который требует большую сумму до средней цены, а дальше требующий меньшую).

Распределение = Новый ТаблицаЗначений;

Распределение.Колонки.Добавить("Склад");

Распределение.Колонки.Добавить("СО");

Распределение.Колонки.Добавить("Количество");

Распределение.Колонки.Добавить("Сумма");

В данном блоке создается таблица значений для хранения результатов распределения материалов по складам с колонками Склад, СО, Количество, Сумма.

Для Каждого Стр Из МассивСО Цикл

Для Каждого СтрСО Из СрЦ Цикл

Пока СтрСО.Количество > 0 Цикл

Если Стр.ДляРаспределения - СтрСО.Цена < 0 Тогда

Прервать;

КонецЕсли;

СтрСО.Количество = СтрСО.Количество - 1;

Стр.ДляРаспределения = Стр.ДляРаспределения - СтрСО.Цена;

Стр.Сумма = Стр.Сумма + СтрСО.Цена;

НовСтр = Распределение.Добавить();

НовСтр.Склад = Стр.Склад;

НовСтр.СО = СтрСО.СО;

НовСтр.Количество = 1;

НовСтр.Сумма = СтрСО.Цена;

КонецЦикла;

КонецЦикла;

КонецЦикла;

// если не все до конца распределилось, то ищем куда лучше отнести последний товар, если он есть

ОстатокРаспределения = МассивСО.Итог("ДляРаспределения");

СрЦ.Сортировать("Количество Убыв, Цена Убыв");

МассивСО.Сортировать("Сумма Убыв");

Для Каждого Стр Из СрЦ Цикл

// выбираем товар, который остался (если остался)

Если Стр.Количество > 0 Тогда

Стр.Количество = Стр.Количество - 1;

МассивСО[1].ДляРаспределения = МассивСО[1].ДляРаспределения - Стр.Цена;

МассивСО[1].Сумма = МассивСО[1].Сумма + Стр.Цена;

// и выбираем для перемещения тот склад, добавив в который отклонение будет минимальным

НовСтр = Распределение.Добавить();

НовСтр.Склад = МассивСО[1].Склад;

НовСтр.СО = Стр.СО;

НовСтр.Количество = 1;

НовСтр.Сумма = Стр.Цена;

КонецЕсли;

КонецЦикла;

Данный блок – главный, в нем происходит распределение товаров по складам. Сначала обходятся все склады в таблице значений МассивСО, для каждого склада идет обход всех материалов (таблица значений СрЦ) у которых количество (количество хранящееся на ликвидируемом складе) больше нуля, если при переносе материала на текущий склад сумма товаров не преувеличит среднюю по предприятию (ДляРаспределения – Цена < 0), то один материал переносится на текущий склад, если же при переносе будет преувеличение средней суммы, то цикл по Количеству элементов (в котором проходит перенос по одному материалу) прерывается, для скорейшего перехода к другому, более дешевому материалу. После обхода всех складов получается сумма нераспределенных товаров (через МассивСО.Итог), если сумма нераспределенных не равна нулю – значит требуется найти куда распределить оставшийся товар (Прим. Ред. – в коде пропущена данная проверка и ОстатокРаспределения не используется, хотя комментарий поставлен с описанием верного действия, т.к. реализация принадлежит победителю олимпиады 2 тура 2012 года исправление не внесено. Данное условие не необходимо, но позволяет пропустить ненужное действие с обходом всех товаров для поиска нераспределенных). Если не все товары распределены, то таблица значений средневзвешанных цен сортируется по убыванию по колонкам Количество и Цена, а таблице значений МассивСО по убыванию колонки Сумма (чтобы элемент 0 имел данные с наиболее заполненным складом, а элемент 1 с менее заполненным), далее для каждого товара, у которого нераспределенное количество больше 0 переносится по 1 элементу на склад отвечающий элементу 1 (менее заполненный).

// сворачиваем таблицу с распределенными товарами, для посчета суммы и количества

Распределение.Свернуть("Склад, СО", "Количество, Сумма");

Распределение.Сортировать("Склад Возр, СО Возр");

В данном блоке таблица значений с распределенными элементами сворачивается и сортируется по возрастанию Склада и спецодежды.

// вывод разрыва между отчетами

ТабДок.Вывести(ОбластьРазрыв);

// вывод заголовка второй таблицы

ОбластьЗаголовка.Параметры.Заголовок = "Предлагаемый вариант размещения спецодежды на двух складах";

ТабДок.Вывести(ОбластьЗаголовка);

// выводим шапку таблицы для каждого склада

ТабДок.Вывести(ОбластьШапкаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

Пока ВыборкаСкладов.Следующий() Цикл

Если ВыборкаСкладов.Ссылка <> ЛиквидируемыйСклад Тогда

ОбластьШапкаСклад.Параметры.Склад = ВыборкаСкладов.Наименование;

ТабДок.Присоединить(ОбластьШапкаСклад);

КонецЕсли;

КонецЦикла;

// массив общей стоимостной оценки по складам после ликвидации вабранного склада

ОбщиеИтоговыеСуммы = Новый Массив;

ОбщиеИтоговыеСуммы.Добавить(0);

ОбщиеИтоговыеСуммы.Добавить(0);

// вывод спецодежды по складам (по количеству и сумме)

ВыборкаСО = Справочники.Материалы.Выбрать();

Пока ВыборкаСО.Следующий() Цикл

ОбластьСтрокаОбщая.Параметры.Спецодежда = ВыборкаСО.Ссылка;

ОбластьСтрокаОбщая.Параметры.Цена = СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

ТабДок.Вывести(ОбластьСтрокаОбщая);

ВыборкаСкладов = Справочники.Склады.Выбрать();

н = 0;

Пока ВыборкаСкладов.Следующий() Цикл

Если ВыборкаСкладов.Ссылка <> ЛиквидируемыйСклад Тогда

Остатки = ОстаткиТоваров(Дата, ВыборкаСкладов.Ссылка, ВыборкаСО.Ссылка);

Отбор = Новый Структура;

Отбор.Вставить("СО", ВыборкаСО.Ссылка);

Отбор.Вставить("Склад", ВыборкаСкладов.Ссылка);

РаспределеннаяСО = Распределение.НайтиСтроки(Отбор);

// ищем остатки на тех складах, которые не ликвидировали

// количество считаем как остатки + распределенные товары на данный склад

Если Остатки.Количество() = 0 Тогда

ОбщееКоличество = ?(РаспределеннаяСО.Количество() = 0, 0, РаспределеннаяСО[0].Количество);

Иначе

ОбщееКоличество = Остатки[0].Количество + ?(РаспределеннаяСО.Количество() = 0, 0, РаспределеннаяСО[0].Количество);

КонецЕсли;

ОбластьСтрокаСклад.Параметры.Количество = ОбщееКоличество;

ОбщиеИтоговыеСуммы[н] = ОбщиеИтоговыеСуммы[н] + ОбщееКоличество \* СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

// сумму считаем, как количество (посчитанное выше) \* на цену (после переоценки по средневзвешанной цене)

ОбластьСтрокаСклад.Параметры.Стоимость = ОбщееКоличество \* СрЦ.Найти(ВыборкаСО.Ссылка, "СО").Цена;

ТабДок.Присоединить(ОбластьСтрокаСклад);

н = н + 1;

КонецЕсли;

КонецЦикла;

КонецЦикла;

ТабДок.Вывести(ОбластьИтогоОбщая);

// вывод общих итоговых сумм по складам

Для н = 0 по ОбщиеИтоговыеСуммы.Количество() - 1 Цикл

// выводим общую сумму в отчет

ОбластьИтогоСклад.Параметры.Стоимость = ОбщиеИтоговыеСуммы[н];

ТабДок.Присоединить(ОбластьИтогоСклад);

КонецЦикла;

В данном блоке данные распределения выводятся в табличный документ. Сначала выводится разрыв (чтобы две таблицы не слились друг с другом), далее шапка таблицы, без ликвидируемого склада. После создаётся массив для хранения общей суммы на обоих остающихся складах. Далее берется выборка всех материалов и идет обход всех материалов, выводится спецодежда, средняя цена (ищется в таблице значений СрЦ, после обходятся все склады, кроме ликвидируемого, для них получаются остатки на указанную дату, текущего склада и текущей спецодежды, далее находятся строки из таблицы значений Распределение, которые соответствуют текущей спецодежде и складу. Далее выводится общее количество материала на складе (как сумма остатков + распределенных с ликвидируемого склада), после подсчитываются итоговые суммы текущего склада и стоимость товара на текущем складе. После вывода таблицы, выводятся итоги (итоговая стоимость по каждому складу).

Процедура ПриОткрытии()

// установка даты по умолчанию (1 апреля 2012 года)

Дата = '20120401';

// установка склада по умолчанию (склад, с максимальным значением кода)

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 1

| Склады.Ссылка КАК Склад

|ИЗ

| Справочник.Склады КАК Склады

|

|УПОРЯДОЧИТЬ ПО

| Склады.Код УБЫВ";

Результат = Запрос.Выполнить().Выгрузить();

Если Результат.Количество() <> 0 Тогда

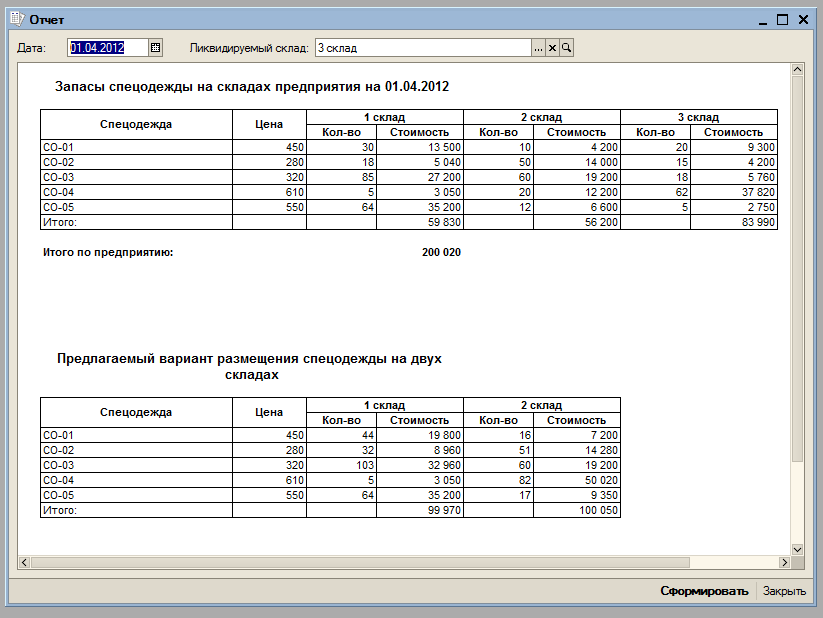
ЛиквидируемыйСклад = Результат[0].Склад;

КонецЕсли;

КонецПроцедуры

Данная процедура расположена в модуле формы и отвечает за установку даты и склада по умолчанию (дата – 1 апреля 2012, а склад – с максимальным значением кода).

Сформированный отчет



2012 год 1 тур

Требования

Крупная финансовая компания, представляющая собой группу взаимосвязанных организаций (порядка 150), ведет бухгалтерский учет в единой информационной базе в программе «1С:Бухгалтерия 8». Многофирменный учет в программе организован с использованием измерения регистра бухгалтерии «Организации».

Задолженность организаций при внутрикорпоративных расчетах отражается (условно) только на счете 76 по субконто «Организации». Значения измерения «Организации» также как и субконто вида «Организации» определяются в едином справочнике «Организации».

В процессе проведения финансовых операций возникают задолженности между организациями, входящими в состав компании. Если организация Х является должником организации Y, то это отражается в учете у организации Х как кредитовое сальдо счета 76 по субконто «Организация Y» на сумму долга, одновременно у организации Y на счете 76 по субконто «Организация Х» отражается дебетовое сальдо на эту же сумму.

Необходимо по данным бухгалтерского учета:

1. Определить внутрикорпоративные долги для организаций и общую сумму долгов, представить их в виде таблицы следующего вида.

Долги

|  |  |  |
| --- | --- | --- |
| Должник | Кредитор | Сумма долга |
| Орг-01 | Орг-02 | 500,00 |
| Орг-02 | Орг-03 | 200,00 |
| Орг-02 | Орг-04 | 150,00 |
| Орг-03 | Орг-01 | 250,00 |
| Орг-04 | Орг-01 | 100,00 |
| Итог |  | 1200,00 |

2. Для каждой организации определить сальдо взаиморасчетов как разницу между общей суммой долгов, причитающихся к получению и общей суммы долгов, причитающихся к возврату, в следующем виде.

Сольдо взаиморасчетов

|  |  |
| --- | --- |
| Организация | Сальдо  взаиморасчетов |
| Орг-01 | -150,00 |
| Орг-02 | 150,00 |
| Орг-03 | -50,00 |
| Орг-04 | 50,00 |

3. Предложить вариант зачета взаимных требований, такой, чтобы общая сумма долгов всех организаций была бы минимальной, при этом для каждой организации сальдо взаиморасчетов оставалось бы неизменным. Результат представить в виде таблицы, аналогичной исходной.

Долги после взаимозачетов

|  |  |  |
| --- | --- | --- |
| Должник | Кредитор | Сумма долга |
| Орг-01 | Орг-02 | 100,00 |
| Орг-01 | Орг-04 | 50,00 |
| Орг-03 | Орг-02 | 50,00 |
| Итог |  | 200,00 |

Каркасная конфигурация

Задание необходимо выполнить в каркасной конфигурации, в которой имеется:

* план счетов, содержащий счета 76 "Расчеты с прочими дебиторами и кредиторами" и 0 "Вспомогательный";
* справочник «Организации»;
* документ для ввода данных о задолженности организаций, при помощи которого введено несколько проводок в дебет и кредит счета 76 в корреспонденции со вспомогательным счетом 0.

ВНИМАНИЕ! Для отладки модуля отчета Вы можете добавлять в информационную базу каркасной конфигурации дополнительные организации и проводки. Жюри проводит тестирование на собственных наборах данных, которые будут отличаться от данных, содержащихся в исходной информационной базе каркасной конфигурации.

Решение

Причина больших сумм долгов внутри организации кроется в циклах и цепочках задолжностей. Как пример цикла – организация 1 должна организации 2, та в свою очередь должна организации 3, а та должна организации 1. Пример цепочки долгов (цепочки видны при отсутствии циклов) – организация 1 должна организации 2, а та должна организации 4. Для уменьшения общей суммы долгов достаточно устранить циклы и цепочки долгов, чтобы все долги сводились к тому, что один занял другому ни у кого не занимая сам. Для устранения циклов достаточно избавиться от одного из составляющих его звеньев. Для устранения цепи (первый – второй – третий) достаточно, чтобы первый отдавал напрямую третьему (за второго) и второму (если остается ещё должен).

Рассмотрим ход решения на представленных в требованиях данных.

Шаг 1 – устранение циклов.

Ищем циклы, первый цикл – (1 – 2 – 3 – 1). Для устранения цикла нужно избавиться от одного из звеньев. Для этого найдем минимальный долг:

|  |  |  |
| --- | --- | --- |
| Кто | Кому | Сколько |
| Орг 1 | Орг 2 | 500 |
| Орг 2 | Орг 3 | 200 |
| Орг 3 | Орг 1 | 250 |

Из таблицы видно, что минимальный долг у организации 2 для организации 3. Теперь можно уничтожить это звено (считая что все три организации прощают друг другу 200, никто ничего не теряет). Получается следующее состояние по организации:

|  |  |  |
| --- | --- | --- |
| Кто | Кому | Сколько |
| Орг 1 | Орг 2 | 300 |
| Орг 2 | Орг 4 | 150 |
| Орг 3 | Орг 1 | 50 |
| Орг 4 | Орг 1 | 100 |

Проверяем на наличие циклов, тут он присутствует, это цикл – (1 – 2 – 4 – 1). Находим минимальный долг:

|  |  |  |
| --- | --- | --- |
| Кто | Кому | Сколько |
| Орг 1 | Орг 2 | 300 |
| Орг 2 | Орг 4 | 150 |
| Орг 4 | Орг 1 | 100 |

Минимальный долгу у организации 4 для организации 1. После уничтожения звена получается состояние:

|  |  |  |
| --- | --- | --- |
| Кто | Кому | Сколько |
| Орг 1 | Орг 2 | 200 |
| Орг 2 | Орг 4 | 50 |
| Орг 3 | Орг 1 | 50 |

Проверяем состояние на наличие циклов, и видим что они отсутствуют.

Шаг 2 – устранение цепочек.

Ищем цепочки долгов (достаточно найти и избавиться от всех цепей длиной 3, не рассматривая более длинные). Первая найденная цепочка – (1 – 2 – 4). Разбиваем её на два независимых долга. Организация 1 отдаст за организацию 2 50 организации 4, так как организации 2 все равно придется потратить 50 из полученных 200 от организации 1 на уплату долга. Так убираем посредника и снижаем общую сумму долга. После разбития цепочки получаем состояние:

|  |  |  |
| --- | --- | --- |
| Кто | Кому | Сколько |
| Орг 1 | Орг 2 | 150 |
| Орг 1 | Орг 4 | 50 |
| Орг 3 | Орг 1 | 50 |

Проверяем состояние на наличие цепочек, находим – (3 – 1 – 2). Разбиваем цепочку. Организация 3 уплатит организации 2 50, а организация 1 останется должна только 100.

|  |  |  |
| --- | --- | --- |
| Кто | Кому | Сколько |
| Орг 1 | Орг 2 | 100 |
| Орг 1 | Орг 4 | 50 |
| Орг 3 | Орг 2 | 50 |

Проверяем состояние на наличие цепочек – их в состоянии не найдено, следовательно решение найдено, и долги организаций разбиты так, что организация даёт в займы не беря ни у кого в долг сама (организации 2 и 4).

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Долги = Новый ТаблицаЗначений;

Долги.Колонки.Добавить("Кто");

Долги.Колонки.Добавить("Кому");

Долги.Колонки.Добавить("Сумма");

Сальдо = Новый ТаблицаЗначений;

Сальдо.Колонки.Добавить("Кто");

Сальдо.Колонки.Добавить("Сумма");

Взаиморасчеты = Новый ТаблицаЗначений;

Взаиморасчеты.Колонки.Добавить("Кто");

Взаиморасчеты.Колонки.Добавить("Кому");

Взаиморасчеты.Колонки.Добавить("Сумма");

// получаем долги

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1,

| ЖурналОперацийДвиженияССубконто.Сумма

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетКт.Код = ""76""";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДолгиЗапись = Долги.Добавить();

ВзаиморасчетыЗапись = Взаиморасчеты.Добавить();

ДолгиЗапись.Кто = Выборка.Организации;

ДолгиЗапись.Кому = Выборка.СубконтоКт1;

ДолгиЗапись.Сумма = Выборка.Сумма;

ВзаиморасчетыЗапись.Кто = ДолгиЗапись.Кто;

ВзаиморасчетыЗапись.Кому = ДолгиЗапись.Кому;

ВзаиморасчетыЗапись.Сумма = ДолгиЗапись.Сумма;

КонецЦикла;

// получаем сальдо

Для Каждого Долг Из Долги Цикл

СальдоЗаписьКто = Сальдо.Найти(Долг.Кто,"Кто");

СальдоЗаписьКому = Сальдо.Найти(Долг.Кому,"Кто");

Если СальдоЗаписьКто = Неопределено Тогда

СальдоЗаписьКто = Сальдо.Добавить();

СальдоЗаписьКто.Кто = Долг.Кто;

СальдоЗаписьКто.Сумма = 0;

КонецЕсли;

Если СальдоЗаписьКому = Неопределено Тогда

СальдоЗаписьКому = Сальдо.Добавить();

СальдоЗаписьКому.Кто = Долг.Кому;

СальдоЗаписьКому.Сумма = 0;

КонецЕсли;

СальдоЗаписьКто.Сумма = СальдоЗаписьКто.Сумма - Долг.Сумма;

СальдоЗаписьКому.Сумма = СальдоЗаписьКому.Сумма + Долг.Сумма;

КонецЦикла;

// расчитываем взаиморасчеты

// ищем циклы

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

Пока ЦиклРезультат <> Неопределено Цикл

Минимальный = ЦиклРезультат[0];

Для Каждого Звено Из ЦиклРезультат Цикл

Если Звено.Сумма < Минимальный.Сумма Тогда

Минимальный = Звено;

КонецЕсли;

КонецЦикла;

// уничтожаем цикл

Для Каждого Звено Из ЦиклРезультат Цикл

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Если ВзаиморасчетыЗапись.Кто <> Звено.Кто Тогда

Продолжить;

КонецЕсли;

Если ВзаиморасчетыЗапись.Кому <> Звено.Кому Тогда

Продолжить;

КонецЕсли;

ВзаиморасчетыЗапись.Сумма = ВзаиморасчетыЗапись.Сумма - Минимальный.Сумма;

Если ВзаиморасчетыЗапись.Сумма = 0 Тогда

Взаиморасчеты.Удалить(ВзаиморасчетыЗапись);

КонецЕсли;

КонецЦикла;

КонецЦикла;

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

КонецЦикла;

// ищем цепи

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

Пока ЦепьРезультат <> Неопределено Цикл

Запись = Новый Массив(2);

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Для i = 0 по 1 Цикл

Если ВзаиморасчетыЗапись.Кто = ЦепьРезультат[i].Кто И ВзаиморасчетыЗапись.Кому = ЦепьРезультат[i].Кому Тогда

Запись[i] = ВзаиморасчетыЗапись;

КонецЕсли;

КонецЦикла;

КонецЦикла;

Если Запись[0].Сумма > Запись[1].Сумма Тогда

Второму = Запись[0].Сумма - Запись[1].Сумма;

Запись[0].Сумма = Второму;

Запись[1].Кто = Запись[0].Кто;

ИначеЕсли Запись[0].Сумма <= Запись[1].Сумма Тогда

Третьему = Запись[1].Сумма - Запись[0].Сумма;

Запись[0].Кому = Запись[1].Кому;

Запись[1].Сумма = Третьему;

КонецЕсли;

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

КонецЦикла;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаВзаиморасчеты = Макет.ПолучитьОбласть("ШапкаВзаиморасчеты");

ОбластьСтрокаВзаиморасчеты = Макет.ПолучитьОбласть("СтрокаВзаиморасчеты");

ОбластьШапкаСальдо = Макет.ПолучитьОбласть("ШапкаСальдо");

ОбластьСтрокаСальдо = Макет.ПолучитьОбласть("СтрокаСальдо");

ТабДок.Очистить();

ОбластьЗаголовок.Параметры[0] = "Долги";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаВзаиморасчеты);

Для Каждого Запись Из Долги Цикл

ОбластьСтрокаВзаиморасчеты.Параметры[0] = Запись.Кто;

ОбластьСтрокаВзаиморасчеты.Параметры[1] = Запись.Кому;

ОбластьСтрокаВзаиморасчеты.Параметры[2] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаВзаиморасчеты);

КонецЦикла;

ОбластьЗаголовок.Параметры[0] = "Сальдо";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаСальдо);

Для Каждого Запись Из Сальдо Цикл

ОбластьСтрокаСальдо.Параметры[0] = Запись.Кто;

ОбластьСтрокаСальдо.Параметры[1] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаСальдо);

КонецЦикла;

ОбластьЗаголовок.Параметры[0] = "Взаиморасчеты";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаВзаиморасчеты);

Для Каждого Запись Из Взаиморасчеты Цикл

ОбластьСтрокаВзаиморасчеты.Параметры[0] = Запись.Кто;

ОбластьСтрокаВзаиморасчеты.Параметры[1] = Запись.Кому;

ОбластьСтрокаВзаиморасчеты.Параметры[2] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаВзаиморасчеты);

КонецЦикла;

КонецПроцедуры

Функция НайтиЦикл(Взаиморасчеты,Результат)

РезультатВременный = Неопределено;

Если Результат = Неопределено Тогда

РезультатВременный = Новый ТаблицаЗначений;

РезультатВременный.Колонки.Добавить("Кто");

РезультатВременный.Колонки.Добавить("Кому");

РезультатВременный.Колонки.Добавить("Сумма");

Иначе

РезультатВременный = Результат.Скопировать();

КонецЕсли;

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Количество = РезультатВременный.Количество();

Если Количество > 0 Тогда

Если РезультатВременный[Количество - 1].Кому <> ВзаиморасчетыЗапись.Кто Тогда

Продолжить;

КонецЕсли;

КонецЕсли;

РезультатЗапись = РезультатВременный.Добавить();

РезультатЗапись.Кто = ВзаиморасчетыЗапись.Кто;

РезультатЗапись.Кому = ВзаиморасчетыЗапись.Кому;

РезультатЗапись.Сумма = ВзаиморасчетыЗапись.Сумма;

Если РезультатВременный.Количество() > 1 Тогда

Если РезультатВременный[0].Кто = РезультатЗапись.Кому Тогда

// Цикл найден

Возврат РезультатВременный;

КонецЕсли;

КонецЕсли;

ЦиклР = НайтиЦикл(Взаиморасчеты,РезультатВременный);

Если ЦиклР <> Неопределено Тогда

Возврат ЦиклР;

Иначе

РезультатВременный.Удалить(РезультатЗапись);

КонецЕсли;

КонецЦикла;

Возврат Неопределено;

КонецФункции

Функция НайтиЦепь(Взаиморасчеты,Результат)

РезультатВременный = Неопределено;

Если Результат = Неопределено Тогда

РезультатВременный = Новый ТаблицаЗначений;

РезультатВременный.Колонки.Добавить("Кто");

РезультатВременный.Колонки.Добавить("Кому");

РезультатВременный.Колонки.Добавить("Сумма");

Иначе

РезультатВременный = Результат.Скопировать();

КонецЕсли;

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Количество = РезультатВременный.Количество();

Если Количество > 0 Тогда

Если РезультатВременный[Количество - 1].Кому <> ВзаиморасчетыЗапись.Кто Тогда

Продолжить;

КонецЕсли;

КонецЕсли;

РезультатЗапись = РезультатВременный.Добавить();

РезультатЗапись.Кто = ВзаиморасчетыЗапись.Кто;

РезультатЗапись.Кому = ВзаиморасчетыЗапись.Кому;

РезультатЗапись.Сумма = ВзаиморасчетыЗапись.Сумма;

Если РезультатВременный.Количество() > 1 Тогда

// цепь из больше чем двух элементов найдена

Возврат РезультатВременный;

КонецЕсли;

Цепь = НайтиЦепь(Взаиморасчеты,РезультатВременный);

Если Цепь <> Неопределено Тогда

Возврат Цепь;

Иначе

РезультатВременный.Удалить(РезультатЗапись);

КонецЕсли;

КонецЦикла;

Возврат Неопределено;

КонецФункции

Автор: Михайлов Алексей

Пояснения к коду:

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Долги = Новый ТаблицаЗначений;

Долги.Колонки.Добавить("Кто");

Долги.Колонки.Добавить("Кому");

Долги.Колонки.Добавить("Сумма");

Сальдо = Новый ТаблицаЗначений;

Сальдо.Колонки.Добавить("Кто");

Сальдо.Колонки.Добавить("Сумма");

Взаиморасчеты = Новый ТаблицаЗначений;

Взаиморасчеты.Колонки.Добавить("Кто");

Взаиморасчеты.Колонки.Добавить("Кому");

Взаиморасчеты.Колонки.Добавить("Сумма");

// получаем долги

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1,

| ЖурналОперацийДвиженияССубконто.Сумма

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетКт.Код = ""76""";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДолгиЗапись = Долги.Добавить();

ВзаиморасчетыЗапись = Взаиморасчеты.Добавить();

ДолгиЗапись.Кто = Выборка.Организации;

ДолгиЗапись.Кому = Выборка.СубконтоКт1;

ДолгиЗапись.Сумма = Выборка.Сумма;

ВзаиморасчетыЗапись.Кто = ДолгиЗапись.Кто;

ВзаиморасчетыЗапись.Кому = ДолгиЗапись.Кому;

ВзаиморасчетыЗапись.Сумма = ДолгиЗапись.Сумма;

КонецЦикла;

// получаем сальдо

Для Каждого Долг Из Долги Цикл

СальдоЗаписьКто = Сальдо.Найти(Долг.Кто,"Кто");

СальдоЗаписьКому = Сальдо.Найти(Долг.Кому,"Кто");

Если СальдоЗаписьКто = Неопределено Тогда

СальдоЗаписьКто = Сальдо.Добавить();

СальдоЗаписьКто.Кто = Долг.Кто;

СальдоЗаписьКто.Сумма = 0;

КонецЕсли;

Если СальдоЗаписьКому = Неопределено Тогда

СальдоЗаписьКому = Сальдо.Добавить();

СальдоЗаписьКому.Кто = Долг.Кому;

СальдоЗаписьКому.Сумма = 0;

КонецЕсли;

СальдоЗаписьКто.Сумма = СальдоЗаписьКто.Сумма - Долг.Сумма;

СальдоЗаписьКому.Сумма = СальдоЗаписьКому.Сумма + Долг.Сумма;

КонецЦикла;

// расчитываем взаиморасчеты

// ищем циклы

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

Пока ЦиклРезультат <> Неопределено Цикл

Минимальный = ЦиклРезультат[0];

Для Каждого Звено Из ЦиклРезультат Цикл

Если Звено.Сумма < Минимальный.Сумма Тогда

Минимальный = Звено;

КонецЕсли;

КонецЦикла;

// уничтожаем цикл

Для Каждого Звено Из ЦиклРезультат Цикл

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Если ВзаиморасчетыЗапись.Кто <> Звено.Кто Тогда

Продолжить;

КонецЕсли;

Если ВзаиморасчетыЗапись.Кому <> Звено.Кому Тогда

Продолжить;

КонецЕсли;

ВзаиморасчетыЗапись.Сумма = ВзаиморасчетыЗапись.Сумма - Минимальный.Сумма;

Если ВзаиморасчетыЗапись.Сумма = 0 Тогда

Взаиморасчеты.Удалить(ВзаиморасчетыЗапись);

КонецЕсли;

КонецЦикла;

КонецЦикла;

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

КонецЦикла;

// ищем цепи

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

Пока ЦепьРезультат <> Неопределено Цикл

Запись = Новый Массив(2);

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Для i = 0 по 1 Цикл

Если ВзаиморасчетыЗапись.Кто = ЦепьРезультат[i].Кто И ВзаиморасчетыЗапись.Кому = ЦепьРезультат[i].Кому Тогда

Запись[i] = ВзаиморасчетыЗапись;

КонецЕсли;

КонецЦикла;

КонецЦикла;

Если Запись[0].Сумма > Запись[1].Сумма Тогда

Второму = Запись[0].Сумма - Запись[1].Сумма;

Запись[0].Сумма = Второму;

Запись[1].Кто = Запись[0].Кто;

ИначеЕсли Запись[0].Сумма <= Запись[1].Сумма Тогда

Третьему = Запись[1].Сумма - Запись[0].Сумма;

Запись[0].Кому = Запись[1].Кому;

Запись[1].Сумма = Третьему;

КонецЕсли;

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

КонецЦикла;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаВзаиморасчеты = Макет.ПолучитьОбласть("ШапкаВзаиморасчеты");

ОбластьСтрокаВзаиморасчеты = Макет.ПолучитьОбласть("СтрокаВзаиморасчеты");

ОбластьШапкаСальдо = Макет.ПолучитьОбласть("ШапкаСальдо");

ОбластьСтрокаСальдо = Макет.ПолучитьОбласть("СтрокаСальдо");

ТабДок.Очистить();

ОбластьЗаголовок.Параметры[0] = "Долги";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаВзаиморасчеты);

Для Каждого Запись Из Долги Цикл

ОбластьСтрокаВзаиморасчеты.Параметры[0] = Запись.Кто;

ОбластьСтрокаВзаиморасчеты.Параметры[1] = Запись.Кому;

ОбластьСтрокаВзаиморасчеты.Параметры[2] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаВзаиморасчеты);

КонецЦикла;

ОбластьЗаголовок.Параметры[0] = "Сальдо";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаСальдо);

Для Каждого Запись Из Сальдо Цикл

ОбластьСтрокаСальдо.Параметры[0] = Запись.Кто;

ОбластьСтрокаСальдо.Параметры[1] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаСальдо);

КонецЦикла;

ОбластьЗаголовок.Параметры[0] = "Взаиморасчеты";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаВзаиморасчеты);

Для Каждого Запись Из Взаиморасчеты Цикл

ОбластьСтрокаВзаиморасчеты.Параметры[0] = Запись.Кто;

ОбластьСтрокаВзаиморасчеты.Параметры[1] = Запись.Кому;

ОбластьСтрокаВзаиморасчеты.Параметры[2] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаВзаиморасчеты);

КонецЦикла;

КонецПроцедуры

Основная процедура, в ней происходит получение долгов, определение сальдо и взаиморасчета, вывод всего результата в поле табличного документа.

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Долги = Новый ТаблицаЗначений;

Долги.Колонки.Добавить("Кто");

Долги.Колонки.Добавить("Кому");

Долги.Колонки.Добавить("Сумма");

Сальдо = Новый ТаблицаЗначений;

Сальдо.Колонки.Добавить("Кто");

Сальдо.Колонки.Добавить("Сумма");

Взаиморасчеты = Новый ТаблицаЗначений;

Взаиморасчеты.Колонки.Добавить("Кто");

Взаиморасчеты.Колонки.Добавить("Кому");

Взаиморасчеты.Колонки.Добавить("Сумма");

В данном блоке создаются переменные для хранения данных. ТабДок – поле табличного документа формы, для вывода результата. Долги – таблица значений с колонками Кто,Кому,Сумма для хранения долгов организаций. Сальдо – таблица значений с колонками Кто,Сумма для хранения сальдо взаиморасчетов. Взаиморасчеты – таблица значений с колонками Кто,Кому,Сумма для хранения взаиморасчетов организаций.

// получаем долги

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1,

| ЖурналОперацийДвиженияССубконто.Сумма

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетКт.Код = ""76""";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДолгиЗапись = Долги.Добавить();

ВзаиморасчетыЗапись = Взаиморасчеты.Добавить();

ДолгиЗапись.Кто = Выборка.Организации;

ДолгиЗапись.Кому = Выборка.СубконтоКт1;

ДолгиЗапись.Сумма = Выборка.Сумма;

ВзаиморасчетыЗапись.Кто = ДолгиЗапись.Кто;

ВзаиморасчетыЗапись.Кому = ДолгиЗапись.Кому;

ВзаиморасчетыЗапись.Сумма = ДолгиЗапись.Сумма;

КонецЦикла;

В данном блоке происходит получение долгов организаций из базы данных. Так как долги записываются в журнал операций дважды (Как дебет одной организации в одной проводке и как кредит в другой) нужно выбрать один вариант, в данной реализации выбрано получение организации и её кредитора (СубконтоКт1), а чтобы не было пустых кредиторов (в проводках с дебетом) стоит условие на код счета кредита «76». После получения данных они обходятся и записываются в таблицы долги (для отображения в табличный документ) и взаиморасчеты (для дальнейшей обработки).

// получаем сальдо

Для Каждого Долг Из Долги Цикл

СальдоЗаписьКто = Сальдо.Найти(Долг.Кто,"Кто");

СальдоЗаписьКому = Сальдо.Найти(Долг.Кому,"Кто");

Если СальдоЗаписьКто = Неопределено Тогда

СальдоЗаписьКто = Сальдо.Добавить();

СальдоЗаписьКто.Кто = Долг.Кто;

СальдоЗаписьКто.Сумма = 0;

КонецЕсли;

Если СальдоЗаписьКому = Неопределено Тогда

СальдоЗаписьКому = Сальдо.Добавить();

СальдоЗаписьКому.Кто = Долг.Кому;

СальдоЗаписьКому.Сумма = 0;

КонецЕсли;

СальдоЗаписьКто.Сумма = СальдоЗаписьКто.Сумма - Долг.Сумма;

СальдоЗаписьКому.Сумма = СальдоЗаписьКому.Сумма + Долг.Сумма;

КонецЦикла;

В данном блоке определяется сальдо всех организаций. Для этого из каждой записи с информацией о долге берется кредитор и заемщик, они оба ищутся в таблице Сальдо (если они там есть, сумма будет увеличиваться/уменьшаться), если запись не найдена, то она добавляется. Для существующей записи заемщика сумма долга отнимается от его сальдо, для кредитора сумма долга прибавляется.

// расчитываем взаиморасчеты

// ищем циклы

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

Пока ЦиклРезультат <> Неопределено Цикл

Минимальный = ЦиклРезультат[0];

Для Каждого Звено Из ЦиклРезультат Цикл

Если Звено.Сумма < Минимальный.Сумма Тогда

Минимальный = Звено;

КонецЕсли;

КонецЦикла;

// уничтожаем цикл

Для Каждого Звено Из ЦиклРезультат Цикл

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Если ВзаиморасчетыЗапись.Кто <> Звено.Кто Тогда

Продолжить;

КонецЕсли;

Если ВзаиморасчетыЗапись.Кому <> Звено.Кому Тогда

Продолжить;

КонецЕсли;

ВзаиморасчетыЗапись.Сумма = ВзаиморасчетыЗапись.Сумма - Минимальный.Сумма;

Если ВзаиморасчетыЗапись.Сумма = 0 Тогда

Взаиморасчеты.Удалить(ВзаиморасчетыЗапись);

КонецЕсли;

КонецЦикла;

КонецЦикла;

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

КонецЦикла;

// ищем цепи

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

Пока ЦепьРезультат <> Неопределено Цикл

Запись = Новый Массив(2);

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Для i = 0 по 1 Цикл

Если ВзаиморасчетыЗапись.Кто = ЦепьРезультат[i].Кто И ВзаиморасчетыЗапись.Кому = ЦепьРезультат[i].Кому Тогда

Запись[i] = ВзаиморасчетыЗапись;

КонецЕсли;

КонецЦикла;

КонецЦикла;

Если Запись[0].Сумма > Запись[1].Сумма Тогда

Второму = Запись[0].Сумма - Запись[1].Сумма;

Запись[0].Сумма = Второму;

Запись[1].Кто = Запись[0].Кто;

ИначеЕсли Запись[0].Сумма <= Запись[1].Сумма Тогда

Третьему = Запись[1].Сумма - Запись[0].Сумма;

Запись[0].Кому = Запись[1].Кому;

Запись[1].Сумма = Третьему;

КонецЕсли;

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

КонецЦикла;

Основная часть процедуры, здесь происходит определение взаиморасчетов при которых сумма долга по предприятию минимальна. Для этого уничтожаются циклы, а после них уничтожаются цепи.

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

Пока ЦиклРезультат <> Неопределено Цикл

Минимальный = ЦиклРезультат[0];

Для Каждого Звено Из ЦиклРезультат Цикл

Если Звено.Сумма < Минимальный.Сумма Тогда

Минимальный = Звено;

КонецЕсли;

КонецЦикла;

// уничтожаем цикл

Для Каждого Звено Из ЦиклРезультат Цикл

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Если ВзаиморасчетыЗапись.Кто <> Звено.Кто Тогда

Продолжить;

КонецЕсли;

Если ВзаиморасчетыЗапись.Кому <> Звено.Кому Тогда

Продолжить;

КонецЕсли;

ВзаиморасчетыЗапись.Сумма = ВзаиморасчетыЗапись.Сумма - Минимальный.Сумма;

Если ВзаиморасчетыЗапись.Сумма = 0 Тогда

Взаиморасчеты.Удалить(ВзаиморасчетыЗапись);

КонецЕсли;

КонецЦикла;

КонецЦикла;

ЦиклРезультат = НайтиЦикл(Взаиморасчеты,Неопределено);

КонецЦикла;

Данный блок отвечает за уничтожение циклов, с помощью функции НайтиЦикл, с параметром Взаиморасчеты (текущие взаиморасчеты) и вторым параметром Неопределено (этот параметр необходим самой функции, так как она рекурсивна, и его значение появится при поиске цикла). Для каждого найденного цикла находится минимальное звено (долг с минимальной суммой), и после этого цикл разбивается – каждое звено цикла находится в таблице взаиморасчетов и уменьшается на сумму минимального долга, если сумма = 0 (а она будет равна нулю, у найденного звена) то звено удаляется из взаиморасчетов. После этого повторно вызывается функция НайтиЦикл, для следующей итерации цикла (возможно, что не все циклы уничтожены во взаиморасчетах).

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

Пока ЦепьРезультат <> Неопределено Цикл

Запись = Новый Массив(2);

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Для i = 0 по 1 Цикл

Если ВзаиморасчетыЗапись.Кто = ЦепьРезультат[i].Кто И ВзаиморасчетыЗапись.Кому = ЦепьРезультат[i].Кому Тогда

Запись[i] = ВзаиморасчетыЗапись;

КонецЕсли;

КонецЦикла;

КонецЦикла;

Если Запись[0].Сумма > Запись[1].Сумма Тогда

Второму = Запись[0].Сумма - Запись[1].Сумма;

Запись[0].Сумма = Второму;

Запись[1].Кто = Запись[0].Кто;

ИначеЕсли Запись[0].Сумма <= Запись[1].Сумма Тогда

Третьему = Запись[1].Сумма - Запись[0].Сумма;

Запись[0].Кому = Запись[1].Кому;

Запись[1].Сумма = Третьему;

КонецЕсли;

ЦепьРезультат = НайтиЦепь(Взаиморасчеты,Неопределено);

КонецЦикла;

В данном блоке уничтожаются цепи. Для поиска цепей используется функция НайтиЦепь, с параметром Взаиморасчеты (текущие взаиморасчеты) и вторым параметром Неопределено (этот параметр необходим самой функции, так как она рекурсивна, и его значение появится при поиске цепи). Если цепь найдена, то её звенья находятся в таблице Взаиморасчеты и записываются в массив Запись. Далее определяется как именно разбить цепь. Если первый должен второму больше, чем второй должен третьему, то первый должен второму часть, и отдает долг третьему за второго. Если же первый должен второму меньше либо столько же, сколько второй третьему, то первый отдает долг третьему и ничего больше не должен второму, а второй отдает свой остаток третьему. После уничтожения цепи повторно вызывается функция НайтиЦепь для следующей итерации.

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаВзаиморасчеты = Макет.ПолучитьОбласть("ШапкаВзаиморасчеты");

ОбластьСтрокаВзаиморасчеты = Макет.ПолучитьОбласть("СтрокаВзаиморасчеты");

ОбластьШапкаСальдо = Макет.ПолучитьОбласть("ШапкаСальдо");

ОбластьСтрокаСальдо = Макет.ПолучитьОбласть("СтрокаСальдо");

В данном блоке получается макет для вывода табличного документа, и его области.

ТабДок.Очистить();

ОбластьЗаголовок.Параметры[0] = "Долги";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаВзаиморасчеты);

Для Каждого Запись Из Долги Цикл

ОбластьСтрокаВзаиморасчеты.Параметры[0] = Запись.Кто;

ОбластьСтрокаВзаиморасчеты.Параметры[1] = Запись.Кому;

ОбластьСтрокаВзаиморасчеты.Параметры[2] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаВзаиморасчеты);

КонецЦикла;

ОбластьЗаголовок.Параметры[0] = "Сальдо";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаСальдо);

Для Каждого Запись Из Сальдо Цикл

ОбластьСтрокаСальдо.Параметры[0] = Запись.Кто;

ОбластьСтрокаСальдо.Параметры[1] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаСальдо);

КонецЦикла;

ОбластьЗаголовок.Параметры[0] = "Взаиморасчеты";

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаВзаиморасчеты);

Для Каждого Запись Из Взаиморасчеты Цикл

ОбластьСтрокаВзаиморасчеты.Параметры[0] = Запись.Кто;

ОбластьСтрокаВзаиморасчеты.Параметры[1] = Запись.Кому;

ОбластьСтрокаВзаиморасчеты.Параметры[2] = Запись.Сумма;

ТабДок.Вывести(ОбластьСтрокаВзаиморасчеты);

КонецЦикла;

В данном блоке происходит очистка табличного документа от старых данных и вывод новых.

Функция НайтиЦикл(Взаиморасчеты,Результат)

РезультатВременный = Неопределено;

Если Результат = Неопределено Тогда

РезультатВременный = Новый ТаблицаЗначений;

РезультатВременный.Колонки.Добавить("Кто");

РезультатВременный.Колонки.Добавить("Кому");

РезультатВременный.Колонки.Добавить("Сумма");

Иначе

РезультатВременный = Результат.Скопировать();

КонецЕсли;

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Количество = РезультатВременный.Количество();

Если Количество > 0 Тогда

Если РезультатВременный[Количество - 1].Кому <> ВзаиморасчетыЗапись.Кто Тогда

Продолжить;

КонецЕсли;

КонецЕсли;

РезультатЗапись = РезультатВременный.Добавить();

РезультатЗапись.Кто = ВзаиморасчетыЗапись.Кто;

РезультатЗапись.Кому = ВзаиморасчетыЗапись.Кому;

РезультатЗапись.Сумма = ВзаиморасчетыЗапись.Сумма;

Если РезультатВременный.Количество() > 1 Тогда

Если РезультатВременный[0].Кто = РезультатЗапись.Кому Тогда

// Цикл найден

Возврат РезультатВременный;

КонецЕсли;

КонецЕсли;

ЦиклР = НайтиЦикл(Взаиморасчеты,РезультатВременный);

Если ЦиклР <> Неопределено Тогда

Возврат ЦиклР;

Иначе

РезультатВременный.Удалить(РезультатЗапись);

КонецЕсли;

КонецЦикла;

Возврат Неопределено;

КонецФункции

Данная функция отвечает за поиск циклов в таблице Взаиморасчеты. Функция рекурсивна (вызывает сама себя). Второй параметр используется для хранения текущего пути обхода графа взаиморасчетов. Если он Неопределен, то таблица для хранения пути создается, и будет заполнена при обходе первым попавшимся долгом. Далее, если в таблице результата уже больше одной записи, то проверяется не совпадает ли первый должник, с тем, кому должен последний (условие цикличности долгов). Если совпадает, значит цикл найден и таблица результата возвращается. Если количество не больше 1, либо ещё не определен цикл, то вызывается НайтиЦикл, с указанием текущего пройденного пути (для дальнейшего поиска), если цикл будет найден – он будет возвращен как результат, если нет – данная ветвь не приводит к решению и удаляется из таблицы результата. Если же были проверены все ветви и нигде не были найдены циклы значит цикла нет и возвращается Неопределено.

Функция НайтиЦепь(Взаиморасчеты,Результат)

РезультатВременный = Неопределено;

Если Результат = Неопределено Тогда

РезультатВременный = Новый ТаблицаЗначений;

РезультатВременный.Колонки.Добавить("Кто");

РезультатВременный.Колонки.Добавить("Кому");

РезультатВременный.Колонки.Добавить("Сумма");

Иначе

РезультатВременный = Результат.Скопировать();

КонецЕсли;

Для Каждого ВзаиморасчетыЗапись Из Взаиморасчеты Цикл

Количество = РезультатВременный.Количество();

Если Количество > 0 Тогда

Если РезультатВременный[Количество - 1].Кому <> ВзаиморасчетыЗапись.Кто Тогда

Продолжить;

КонецЕсли;

КонецЕсли;

РезультатЗапись = РезультатВременный.Добавить();

РезультатЗапись.Кто = ВзаиморасчетыЗапись.Кто;

РезультатЗапись.Кому = ВзаиморасчетыЗапись.Кому;

РезультатЗапись.Сумма = ВзаиморасчетыЗапись.Сумма;

Если РезультатВременный.Количество() > 1 Тогда

// цепь из больше чем двух элементов найдена

Возврат РезультатВременный;

КонецЕсли;

Цепь = НайтиЦепь(Взаиморасчеты,РезультатВременный);

Если Цепь <> Неопределено Тогда

Возврат Цепь;

Иначе

РезультатВременный.Удалить(РезультатЗапись);

КонецЕсли;

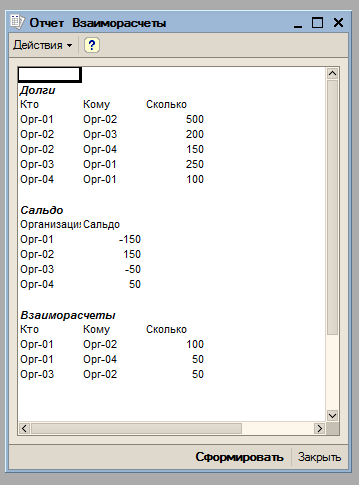
КонецЦикла;

Возврат Неопределено;

КонецФункции

Данная функция используется для поиска цепочек долгов, она реализована так же как и поиск циклов, с той лишь разницей, что тут не должно быть замкнутого круга и достаточно чтобы одна запись долга продолжалась другой (первый должен второму, а второй третьему – 2 записи о долге).

Сформированный отчет



2011 год 2 тур

Требования

Исходная ситуация

Торговая организация заключает договора с поставщиками и покупателями. В соответствии с этими договорами организация получает платежи по договорам с покупателями и производит платежи по договорам с поставщиками.

Задолженность перед поставщиками отражается информационной базе (ИБ) по кредиту счета 60 «Поставщики». Задолженность покупателей отражается в ИБ по дебету счета 62 «Покупатели». Аналитический учет на этих счетах ведется в разрезе субконто двух видов – «Контрагенты» и «Договоры». Видам субконто «Контрагенты» и «Договоры» соответствуют одноименные справочники.

Справочник «Контрагенты» содержит код и наименование контрагента, ему подчинен справочник «Договоры».

Справочник «Договоры» помимо реквизитов «Код» и «Наименование» содержит реквизиты – «Дата оплаты» и «Процент штрафа». Для договоров с покупателями реквизит «Дата оплаты» – это дата планируемого поступления денежных средств по договору. Для договоров с поставщиками реквизит «Дата оплаты» – это дата наступления штрафных санкций (день, начиная с которого ежедневно начисляются суммы штрафов). Реквизит «Процент штрафа» *(р)* указывает процент штрафа, начисляемого за каждый день просрочки платежа.

Начисление суммы штрафа при задержке оплаты поставщику

Сумма штрафа начисляется ежедневно, начиная со дня начала действия штрафных санкций, до дня окончательного погашения долга (включительно), и рассчитывается от суммы фактического остатка долга на начало текущего дня (*Sтек*) по формуле: *Sтек\*р/100.* Начисленная сумма штрафа увеличивает сумму долга, но она не участвует при исчислении суммы штрафа в последующие дни.

Погашение долга

Долг может погашаться частично или полностью. В любом случае при погашении долга сначала полностью погашается накопленная на этот момент сумма начисленного штрафа. Оставшаяся сумма (если осталась) идет на погашение фактического остатка долга, зафиксированного на начало текущего дня. Если сумма долга погашена не полностью, тогда на оставшуюся сумму долга в последующие дни также начисляются штрафы.

Требуется

На основании анализа данных ИБ по предполагаемому поступлению и расходованию денежных средств необходимо сформировать отчет, содержащий оптимальный график предстоящих платежей.

Для этого на момент составления отчета необходимо определить:

- фактическое наличие денежных средств на расчетном счете организации на дату составления отчета (сальдо счета 51);

- суммы задолженностей покупателей по заключенным с ними договорам, как текущие остатки по договорам, учтенные на счете 62 на дату составления отчета. Договора с просроченной оплатой при составлении графика учитываться не должны;

- сумму задолженности перед поставщиками, как текущий остаток на начало дня по договору на счете 60, увеличенный на сумму штрафа.

На основании этих данных в отчете необходимо определить последовательность и суммы предполагаемых поступлений денежных средств и планируемых платежей. При недостаточном наличии денежных средств на расчетном счете допускаются частичные погашения долгов, не приводящие к овердрафту на расчетном счете. График должен быть составлен таким образом, чтобы общая сумма уплачиваемых штрафов была бы минимальной. Форма представления отчета приведена в разделе «*Пример*».

Если общая сумма задолженности перед поставщиками превышает текущую сумму на расчетном счете и общую сумму планируемых поступлений денежных средств, то в таком случае составлять график не требуется, а следует выдать соответствующее сообщение.

Исходные данные

Задание необходимо выполнить в каркасной конфигурации, в которой имеется План счетов, содержащий счета 41, 51, 60, 62, 90, справочники «Контрагенты», «Договоры», «Товары», документы для ввода проводок, отражающих суммы задолженностей по договорам с покупателями и поставщиками, а также поступление денежных средств на расчетный счет.

При помощи документов введено несколько проводок по приходу товаров – в дебет счета 41 и кредит счета 60 и по продаже товаров – в дебет счета 62 и кредит счета 90.

Введено поступление денежных средств на расчетный счет для формирования начального остатка денежных средств. Проводки бухгалтерского учета, не относящиеся к сути задания на программирование, опущены.

Для отладки модуля отчета разрешается добавлять в информационную базу каркасной конфигурации дополнительные справочные данные и проводки.

Жюри проводит тестирование на собственных данных, которые будут отличаться от данных, содержащихся в исходной информационной базе каркасной конфигурации.

Пример

Пусть на начало дня 15 января 2011 года у предприятия на расчетном счете имеется 10 000 руб., при этом в учете зафиксирована следующая картина взаимных обязательств по договорам купли-продажи.

Договоры с покупателями

|  |  |  |  |
| --- | --- | --- | --- |
| Покупатель | Договор | сумма | Дата планируемого поступления денежных средств |
| Пок010 | Д10 | 30 000,00 | 16.01.2011 |
| Пок020 | Д20 | 35 000,00 | 18.01.2011 |
| Пок030 | Д30 | 45 000,00 | 21.01.2011 |
| Пок040 | Д40 | 60 000,00 | 25.01.2011 |
| ИТОГО |  | 170 000,00 |  |

Договоры с поставщиками

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поставщик | Договор | Cумма | Дата оплаты | % штрафа |
| Пост001 | Д1 | 20 000,00 | 15.01.2011 | 2,00 |
| Пост002 | Д2 | 35 000,00 | 15.01.2011 | 3,00 |
| Пост003 | Д3 | 25 000,00 | 17.01.2011 | 2,00 |
| Пост004 | Д4 | 60 000,00 | 19.01.2011 | 5,00 |
| ИТОГО |  | 140 000,00 |  |  |

 Необходимо предложить отчет, содержащий планируемый график платежей. Искомый график будет иметь следующий вид.

Планируемый график платежей по состоянию на 15.01.2011

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Дата | сальдо  счета 51 | Поступление по договорам  с покупателями (сч. 62) | | Выплаты по договорам  с поставщиками (сч. 60) | |
| договор | сумма | договор | Сумма |
| 15.01.2011 | 10 000,00 |  |  | Д2 | 10 000,00 |
| 16.01.2011 | 0,00 | Д10 | 30 000,00 |  |  |
| 17.01.2011 | 30 000,00 |  |  | Д2 | 27 613,00 |
|  | 2 387,00 |  |  | Д4 | 2 387,00 |
|  | 0,00 |  |  |  |  |
| 18.01.2011 | 0,00 | Д20 | 35 000,00 |  |  |
| 19.01.2011 | 35 000,00 |  |  | Д4 | 35 000,00 |
| 21.01.2011 | 0,00 | Д30 | 45 000,00 |  |  |
| 22.01.2011 | 45 000,00 |  |  | Д3 | 15 682,30 |
|  | 29 317,70 |  |  | Д4 | 29 317,70 |
| 25.01.2011 | 0,00 | Д40 | 60 000,00 |  |  |
| 26.01.2011 | 60 000,00 |  |  | Д1 | 24 800,00 |
|  | 35 200,00 |  |  | Д3 | 13 303,12 |
|  | 21 896,88 |  |  |  |  |
| ИТОГО: |  |  | 170 000,00 |  | 158 103,12 |

Сохранение выполненного задания

В результате выполнения задания должен быть разработан отчет, работающий в среде каркасной конфигурации. Данный отчет необходимо сохранить на как внешний отчет.

Решение

Суть данной задачи – найти в графе путь минимальной длинны. Для этого требуется пройти по всем возможным вариантам и выбрать тот, где сумма долгов будет минимальной. Вершины графа – состояния долгов, содержащие – дату, сольдо, ожидаемые получения от покупателей, требуемые выплаты поставщикам. Вес связи двух вершин – сумма затраченная на переход от состояния первой вершины, к состоянию второй (оплачиваемая сумма долга). Рассмотрим пример переходов по состояниям, приводящий к решению как в примере требований.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Состояние | Дата | Сальдо | Покупатели | | | | Поставщики | | | | | Оплачено |
| Договор | Сумма | | Дата | Договор | Сумма | Дата | Процент | Долг |
| S0 | 15.01.11 | 10000 | Д10 | 30000 | | 16.01.11 | Д1 | 20000 | 15.01.11 | 2 | 400 | 0 |
| Д20 | 35000 | | 18.01.11 | Д2 | 35000 | 15.01.11 | 3 | 1050 |
| Д30 | 45000 | | 21.01.11 | Д3 | 25000 | 17.01.11 | 2 | 0 |
| Д40 | 60000 | | 25.01.11 | Д4 | 60000 | 19.01.11 | 5 | 0 |
| S02 | 15.01.11 | 0 | Д10 | 30000 | | 16.01.11 | Д1 | 20000 | 15.01.11 | 2 | 400 | 10000 |
| Д20 | 35000 | | 18.01.11 | Д2 | 26050 | 15.01.11 | 3 | 0 |
| Д30 | 45000 | | 21.01.11 | Д3 | 25000 | 17.01.11 | 2 | 0 |
| Д40 | 60000 | | 25.01.11 | Д4 | 60000 | 19.01.11 | 5 | 0 |
| S022 | 17.01.11 | 2387 | Д20 | 35000 | | 18.01.11 | Д1 | 20000 | 15.01.11 | 2 | 1200 | 37613 |
| Д30 | 45000 | | 21.01.11 | Д3 | 25000 | 17.01.11 | 2 | 500 |
| Д40 | 60000 | | 25.01.11 | Д4 | 60000 | 19.01.11 | 5 | 0 |
| S0224 | 17.01.11 | 0 | Д20 | 35000 | | 18.01.11 | Д1 | 20000 | 15.01.11 | 2 | 1200 | 40000 |
| Д30 | 45000 | | 21.01.11 | Д3 | 25000 | 17.01.11 | 2 | 500 |
| Д40 | 60000 | | 25.01.11 | Д4 | 57613 | 19.01.11 | 5 | 0 |
| S02244 | 19.01.11 | 0 | Д30 | 45000 | | 21.01.11 | Д1 | 20000 | 15.01.11 | 2 | 2000 | 75000 |
| Д40 | 60000 | | 25.01.11 | Д3 | 25000 | 17.01.11 | 2 | 1500 |
| Д4 | 25493,65 | 19.01.11 | 5 | 0 |
| S022444 | 22.01.11 | 15682,30 | Д40 | | 60000 | 25.01.11 | Д1 | 20000 | 15.01.11 | 2 | 2800 | 105592,38 |
| Д3 | 25000 | 17.01.11 | 2 | 3000 |
| S0224443 | 22.01.11 | 0 | Д40 | | 60000 | 25.01.11 | Д1 | 20000 | 15.01.11 | 2 | 2800 | 120000 |
| Д3 | 12317,7 | 17.01.11 | 2 | 0 |
| S02244431 | 26.01.11 | 35200 |  | | | | Д3 | 12317,7 | 17.01.11 | 2 | 985,42 | 144800 |
| S022444313 | 26.01.11 | 21896,88 |  | | | |  | | | | | 158103,12 |

Это один из путей, начиная с состояния S0 существует несколько вариантов развития событий: оплата по договору Д1, либо по договору Д2, либо по Д3, либо по Д4. После оплаты по договору снова появляется несколько вариантов для оплаты. Для верного решения задачи требуется рассмотреть все варианты последовательности оплаты, и выбрать минимальный по сумме выплат. Для этого подойдет рекурсивная реализация обхода графа, рассмотрим алгоритм:

Функция ОбходГрафа(Граф)

Если вершин идущих из корня нет

Вернуть конец графа

Конец если

Для всех вершин (A) идущих из корня

Результат = ОбходГрафа(Граф начинающийся в вершине A)

Если Результат > Минимального результата Тогда

Минимальный результат = Результат

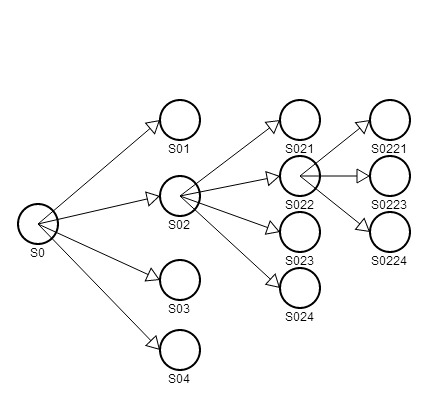
Конец если

Конец цикла

Вернуть минимальный результат

Конец функции

Функция вызывает сама себя, чтобы обойти подграфы. Рассмотрим граф:



S0 – корень графа, из него можно перейти в S01,S02,S03,S04. Функция вызыванная для графа с корнем S0, вызовет саму себя, для графов с корнями S01,S02,S03,S04, внутри опять вызовет саму себя для каждого из графов, с другими графами, и так далее. Так функция обойдет весь граф и, дойдя до вершины не имеющей дочерних вершин, остановится и соберет результат выбирая на наименьший из всех получившихся. Так будет найден путь, приводящий к требуемому результату.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

ОбластьИтог = Макет.ПолучитьОбласть("Итог");

// Получаем данные из базы и формируем начальные структуры

СтартДанные = Новый Структура;

СтартДанные.Вставить("Сумма",0);

// Дата

СтартДата = Дата;

// Сальдо

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.СуммаОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""51"", , ) КАК ЖурналОперацийОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

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

Результат.Следующий();

СтартСальдо = Результат.СуммаОстаток;

// Поставщики

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.Субконто1,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто1),

| ЖурналОперацийОстатки.Субконто2 КАК Субконто2,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто2),

| ЖурналОперацийОстатки.СуммаОстатокКт,

| Договоры.ДатаОплаты КАК ДатаОплаты,

| Договоры.Процент

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""60"", , ) КАК ЖурналОперацийОстатки

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Договоры КАК Договоры

| ПО ЖурналОперацийОстатки.Субконто2 = Договоры.Ссылка

|

|УПОРЯДОЧИТЬ ПО

| ДатаОплаты,

| Субконто2";

Запрос.УстановитьПараметр("Дата", Дата);

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

Поставщики = Новый Массив;

Пока Результат.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Результат.ДатаОплаты);

Запись.Вставить("Договор",Результат.Субконто2);

Запись.Вставить("Сумма",Результат.СуммаОстатокКт);

Запись.Вставить("Штраф",Результат.Процент);

Поставщики.Добавить(Запись);

КонецЦикла;

СтартДанные.Вставить("Поставщики",Поставщики);

// Покупатели

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.Субконто1,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто1),

| ЖурналОперацийОстатки.Субконто2 КАК Субконто2,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто2),

| ЖурналОперацийОстатки.СуммаОстатокДт КАК СуммаОстатокДт,

| Договоры.ДатаОплаты КАК ДатаОплаты

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""62"", , ) КАК ЖурналОперацийОстатки

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Договоры КАК Договоры

| ПО ЖурналОперацийОстатки.Субконто2 = Договоры.Ссылка

|ГДЕ

| Договоры.ДатаОплаты >= &Дата

|

|УПОРЯДОЧИТЬ ПО

| ДатаОплаты,

| Субконто2";

Запрос.УстановитьПараметр("Дата", Дата);

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

Покупатели = Новый Массив;

Пока Результат.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Результат.ДатаОплаты);

Запись.Вставить("Договор",Результат.Субконто2);

Запись.Вставить("Сумма",Результат.СуммаОстатокДт);

Покупатели.Добавить(Запись);

КонецЦикла;

СтартДанные.Вставить("Покупатели",Покупатели);

// Планируем расчеты

Результат = СпланироватьРасчеты(СтартДата,СтартСальдо,СтартДанные);

ТабДок.Очистить();

Если Результат = Неопределено Тогда

Сообщить("Не хватает средств для оплаты долгов");

Иначе

ОбластьЗаголовок.Параметры[0] = "Планируемый график платежей по состоянию на " + Дата;

ТабДок.Вывести(ОбластьЗаголовок);

ПОЛУЧЕНИЕ\_ДЕНЕГ = 0;

ВЫПЛАТА\_ДЕНЕГ = 1;

ПокупателиИтог = 0;

ПоставщикиИтог = 0;

Для Каждого Вершина Из Результат.Путь Цикл

ОбластьЗапись.Параметры[0] = Вершина.Дата;

ОбластьЗапись.Параметры[1] = Вершина.Сальдо;

Если Вершина.Действие = ПОЛУЧЕНИЕ\_ДЕНЕГ Тогда

ОбластьЗапись.Параметры[2] = Вершина.Покупатель.Договор;

ОбластьЗапись.Параметры[3] = Вершина.Покупатель.Сумма;

ОбластьЗапись.Параметры[4] = "";

ОбластьЗапись.Параметры[5] = "";

ПокупателиИтог = ПокупателиИтог + Вершина.Покупатель.Сумма;

Иначе

ОбластьЗапись.Параметры[2] = "";

ОбластьЗапись.Параметры[3] = "";

ОбластьЗапись.Параметры[4] = Вершина.Поставщик.Договор;

ОбластьЗапись.Параметры[5] = Вершина.Поставщик.Оплачено;

ПоставщикиИтог = ПоставщикиИтог + Вершина.Поставщик.Оплачено;

КонецЕсли;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

ОбластьИтог.Параметры[0] = ПокупателиИтог;

ОбластьИтог.Параметры[1] = ПоставщикиИтог;

ТабДок.Вывести(ОбластьИтог);

КонецЕсли;

КонецПроцедуры

Функция СпланироватьРасчеты(Дата,Сальдо,Данные)

ПОЛУЧЕНИЕ\_ДЕНЕГ = 0;

ВЫПЛАТА\_ДЕНЕГ = 1;

Результат = Новый Структура;

Путь = Новый Массив;

Результат.Вставить("Сумма",0);

Результат.Вставить("Путь",Путь);

// проверяем не завершены ли получения и выплаты

Если Данные.Покупатели.Количество() = 0 И Данные.Поставщики.Количество() = 0 Тогда

Возврат Результат;

КонецЕсли;

РДата = Дата;

РСальдо = Сальдо;

// проверяем есть ли деньги на счете для оплаты поставщикам

Если Сальдо = 0 Тогда

Если Данные.Покупатели.Количество() = 0 Тогда

Возврат Неопределено;

КонецЕсли;

// денег нет, выполняем ближайшую получку денег от покупателя и планируем дальше, начиная со следующего после получения денег дня

Покупатель = Данные.Покупатели[0];

РДата = НачалоДня(Покупатель.Дата) + (60\*60\*24);

РСальдо = Покупатель.Сумма;

// убираем оформленную получку денег

Данные.Покупатели.Удалить(0);

// добавляем в путь состояние с получением оплаты

НовоеСостояние = Новый Структура;

НовоеСостояние.Вставить("Дата",Покупатель.Дата);

НовоеСостояние.Вставить("Сальдо",Сальдо);

НовоеСостояние.Вставить("Действие",ПОЛУЧЕНИЕ\_ДЕНЕГ);

НовоеСостояние.Вставить("Покупатель",Покупатель);

Результат.Путь.Добавить(НовоеСостояние);

КонецЕсли;

ЛучшийРезультат = Неопределено;

ЛучшийНачало = Неопределено;

// пробуем разные выплаты поставщикам (обходим их по очереди, выбираем результат с минимальной суммой выплат)

Для Каждого Поставщик Из Данные.Поставщики Цикл

НовоеСальдо = 0;

Начало = Неопределено;

НовыеДанные = Новый Структура;

НовыеДанные.Вставить("Поставщики",Новый Массив);

НовыеДанные.Вставить("Покупатели",Новый Массив);

Для Каждого Запись Из Данные.Покупатели Цикл

НоваяЗапись = Новый Структура;

НоваяЗапись.Вставить("Дата",Запись.Дата);

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма);

НовыеДанные.Покупатели.Добавить(НоваяЗапись);

КонецЦикла;

Для Каждого Запись Из Данные.Поставщики Цикл

НоваяЗапись = Неопределено;

Если Поставщик = Запись Тогда

Штраф = 0;

Дни = 0;

Если РДата >= Запись.Дата Тогда

Дни = ((КонецДня(РДата)+1)-НачалоДня(Запись.Дата))/(24\*60\*60);

Штраф = (Запись.Сумма \* (Запись.Штраф/100)) \* Дни;

КонецЕсли;

Если (Поставщик.Сумма + Штраф) > РСальдо Тогда

НоваяЗапись = Новый Структура;

Если Штраф = 0 Тогда

НоваяЗапись.Вставить("Дата",Запись.Дата);

Иначе

НоваяЗапись.Вставить("Дата",(КонецДня(РДата)+1));

КонецЕсли;

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма + Штраф - РСальдо);

НоваяЗапись.Вставить("Штраф",Запись.Штраф);

НовоеСальдо = 0;

Начало = Новый Структура;

Начало.Вставить("Дата",Запись.Дата);

Начало.Вставить("Договор",Запись.Договор);

Начало.Вставить("Оплачено",РСальдо);

Иначе

НовоеСальдо = РСальдо - (Запись.Сумма + Штраф);

Начало = Новый Структура;

Начало.Вставить("Дата",Запись.Дата);

Начало.Вставить("Договор",Запись.Договор);

Начало.Вставить("Оплачено",Запись.Сумма + Штраф);

КонецЕсли;

Иначе

НоваяЗапись = Новый Структура;

НоваяЗапись.Вставить("Дата",Запись.Дата);

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма);

НоваяЗапись.Вставить("Штраф",Запись.Штраф);

КонецЕсли;

Если НоваяЗапись <> Неопределено Тогда

НовыеДанные.Поставщики.Добавить(НоваяЗапись);

КонецЕсли;

КонецЦикла;

РезультатВетки = СпланироватьРасчеты(РДата,НовоеСальдо,НовыеДанные);

Если ЛучшийРезультат <> Неопределено И РезультатВетки <> Неопределено Тогда

Если (ЛучшийНачало.Оплачено + ЛучшийРезультат.Сумма) > (Начало.Оплачено + РезультатВетки.Сумма) Тогда

ЛучшийРезультат = РезультатВетки;

ЛучшийНачало = Начало;

КонецЕсли;

Иначе

ЛучшийРезультат = РезультатВетки;

ЛучшийНачало = Начало;

КонецЕсли;

КонецЦикла;

Если ЛучшийРезультат = Неопределено Тогда

Возврат Неопределено;

КонецЕсли;

// найден лучший результат, составляем итоговый результат

Результат.Сумма = ЛучшийНачало.Оплачено + ЛучшийРезультат.Сумма;

НоваяВершина = Новый Структура;

НоваяВершина.Вставить("Дата",РДата);

НоваяВершина.Вставить("Сальдо",РСальдо);

НоваяВершина.Вставить("Действие",ВЫПЛАТА\_ДЕНЕГ);

НоваяВершина.Вставить("Поставщик",ЛучшийНачало);

Результат.Путь.Добавить(НоваяВершина);

Для Каждого Вершина Из ЛучшийРезультат.Путь Цикл

Результат.Путь.Добавить(Вершина);

КонецЦикла;

Возврат Результат;

КонецФункции

Процедура ПриОткрытии()

Дата = Дата(2011,01,15);

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

ОбластьИтог = Макет.ПолучитьОбласть("Итог");

// Получаем данные из базы и формируем начальные структуры

СтартДанные = Новый Структура;

СтартДанные.Вставить("Сумма",0);

// Дата

СтартДата = Дата;

// Сальдо

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.СуммаОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""51"", , ) КАК ЖурналОперацийОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

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

Результат.Следующий();

СтартСальдо = Результат.СуммаОстаток;

// Поставщики

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.Субконто1,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто1),

| ЖурналОперацийОстатки.Субконто2 КАК Субконто2,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто2),

| ЖурналОперацийОстатки.СуммаОстатокКт,

| Договоры.ДатаОплаты КАК ДатаОплаты,

| Договоры.Процент

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""60"", , ) КАК ЖурналОперацийОстатки

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Договоры КАК Договоры

| ПО ЖурналОперацийОстатки.Субконто2 = Договоры.Ссылка

|

|УПОРЯДОЧИТЬ ПО

| ДатаОплаты,

| Субконто2";

Запрос.УстановитьПараметр("Дата", Дата);

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

Поставщики = Новый Массив;

Пока Результат.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Результат.ДатаОплаты);

Запись.Вставить("Договор",Результат.Субконто2);

Запись.Вставить("Сумма",Результат.СуммаОстатокКт);

Запись.Вставить("Штраф",Результат.Процент);

Поставщики.Добавить(Запись);

КонецЦикла;

СтартДанные.Вставить("Поставщики",Поставщики);

// Покупатели

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.Субконто1,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто1),

| ЖурналОперацийОстатки.Субконто2 КАК Субконто2,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто2),

| ЖурналОперацийОстатки.СуммаОстатокДт КАК СуммаОстатокДт,

| Договоры.ДатаОплаты КАК ДатаОплаты

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""62"", , ) КАК ЖурналОперацийОстатки

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Договоры КАК Договоры

| ПО ЖурналОперацийОстатки.Субконто2 = Договоры.Ссылка

|ГДЕ

| Договоры.ДатаОплаты >= &Дата

|

|УПОРЯДОЧИТЬ ПО

| ДатаОплаты,

| Субконто2";

Запрос.УстановитьПараметр("Дата", Дата);

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

Покупатели = Новый Массив;

Пока Результат.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Результат.ДатаОплаты);

Запись.Вставить("Договор",Результат.Субконто2);

Запись.Вставить("Сумма",Результат.СуммаОстатокДт);

Покупатели.Добавить(Запись);

КонецЦикла;

СтартДанные.Вставить("Покупатели",Покупатели);

// Планируем расчеты

Результат = СпланироватьРасчеты(СтартДата,СтартСальдо,СтартДанные);

ТабДок.Очистить();

Если Результат = Неопределено Тогда

Сообщить("Не хватает средств для оплаты долгов");

Иначе

ОбластьЗаголовок.Параметры[0] = "Планируемый график платежей по состоянию на " + Дата;

ТабДок.Вывести(ОбластьЗаголовок);

ПОЛУЧЕНИЕ\_ДЕНЕГ = 0;

ВЫПЛАТА\_ДЕНЕГ = 1;

ПокупателиИтог = 0;

ПоставщикиИтог = 0;

Для Каждого Вершина Из Результат.Путь Цикл

ОбластьЗапись.Параметры[0] = Вершина.Дата;

ОбластьЗапись.Параметры[1] = Вершина.Сальдо;

Если Вершина.Действие = ПОЛУЧЕНИЕ\_ДЕНЕГ Тогда

ОбластьЗапись.Параметры[2] = Вершина.Покупатель.Договор;

ОбластьЗапись.Параметры[3] = Вершина.Покупатель.Сумма;

ОбластьЗапись.Параметры[4] = "";

ОбластьЗапись.Параметры[5] = "";

ПокупателиИтог = ПокупателиИтог + Вершина.Покупатель.Сумма;

Иначе

ОбластьЗапись.Параметры[2] = "";

ОбластьЗапись.Параметры[3] = "";

ОбластьЗапись.Параметры[4] = Вершина.Поставщик.Договор;

ОбластьЗапись.Параметры[5] = Вершина.Поставщик.Оплачено;

ПоставщикиИтог = ПоставщикиИтог + Вершина.Поставщик.Оплачено;

КонецЕсли;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

ОбластьИтог.Параметры[0] = ПокупателиИтог;

ОбластьИтог.Параметры[1] = ПоставщикиИтог;

ТабДок.Вывести(ОбластьИтог);

КонецЕсли;

КонецПроцедуры

Данная процедура выполняется при нажатии на кнопку сформировать отчет и выполняет загрузку данных из базы данных, заполнение графа, вызов функции обхода графа и вывод результата в поле табличного документа.

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

ОбластьИтог = Макет.ПолучитьОбласть("Итог");

В данном блоке идет получение элемента формы ТабличныйДокумент и получение областей макета, по которым будет строиться результат.

// Получаем данные из базы и формируем начальные структуры

СтартДанные = Новый Структура;

СтартДанные.Вставить("Сумма",0);

// Дата

СтартДата = Дата;

В данном блоке создается структура начального состояния СтартДанные и переменная даты начала СтартДата.

// Сальдо

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.СуммаОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""51"", , ) КАК ЖурналОперацийОстатки";

Запрос.УстановитьПараметр("Дата", Дата);

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

Результат.Следующий();

СтартСальдо = Результат.СуммаОстаток;

В данном блоке получается Сальдо на дату начала и записывается в переменную СтартСальдо.

// Поставщики

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.Субконто1,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто1),

| ЖурналОперацийОстатки.Субконто2 КАК Субконто2,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто2),

| ЖурналОперацийОстатки.СуммаОстатокКт,

| Договоры.ДатаОплаты КАК ДатаОплаты,

| Договоры.Процент

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""60"", , ) КАК ЖурналОперацийОстатки

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Договоры КАК Договоры

| ПО ЖурналОперацийОстатки.Субконто2 = Договоры.Ссылка

|

|УПОРЯДОЧИТЬ ПО

| ДатаОплаты,

| Субконто2";

Запрос.УстановитьПараметр("Дата", Дата);

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

Поставщики = Новый Массив;

Пока Результат.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Результат.ДатаОплаты);

Запись.Вставить("Договор",Результат.Субконто2);

Запись.Вставить("Сумма",Результат.СуммаОстатокКт);

Запись.Вставить("Штраф",Результат.Процент);

Поставщики.Добавить(Запись);

КонецЦикла;

СтартДанные.Вставить("Поставщики",Поставщики);

В данном блоке получается список поставщиков, на дату начала. Полученные из базы данных записи добавляются в массив Поставщики, и массив сохраняется в начальном состоянии в поле Поставщики.

// Покупатели

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналОперацийОстатки.Субконто1,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто1),

| ЖурналОперацийОстатки.Субконто2 КАК Субконто2,

| ПРЕДСТАВЛЕНИЕ(ЖурналОперацийОстатки.Субконто2),

| ЖурналОперацийОстатки.СуммаОстатокДт КАК СуммаОстатокДт,

| Договоры.ДатаОплаты КАК ДатаОплаты

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&Дата, Счет.Код = ""62"", , ) КАК ЖурналОперацийОстатки

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Договоры КАК Договоры

| ПО ЖурналОперацийОстатки.Субконто2 = Договоры.Ссылка

|ГДЕ

| Договоры.ДатаОплаты >= &Дата

|

|УПОРЯДОЧИТЬ ПО

| ДатаОплаты,

| Субконто2";

Запрос.УстановитьПараметр("Дата", Дата);

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

Покупатели = Новый Массив;

Пока Результат.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Результат.ДатаОплаты);

Запись.Вставить("Договор",Результат.Субконто2);

Запись.Вставить("Сумма",Результат.СуммаОстатокДт);

Покупатели.Добавить(Запись);

КонецЦикла;

СтартДанные.Вставить("Покупатели",Покупатели);

В данном блоке получаются покупатели на дату начала. Данные записываются в массив, а массив добавляется в начальное состояние в поле Покупатели.

// Планируем расчеты

Результат = СпланироватьРасчеты(СтартДата,СтартСальдо,СтартДанные);

ТабДок.Очистить();

В данном блоке происходит вызов рекурсивной функции СпланироватьРасчеты, которая определит лучший путь. Как параметры указаны дата начала (СтартДата), начальное сальдо (СтартСальдо) и начальное состояние (СтартДанные), на основе которого будут строиться дальнейшие состояния. После завершения планирования идет отчистка табличного документа от старых данных.

Если Результат = Неопределено Тогда

Сообщить("Не хватает средств для оплаты долгов");

Иначе

ОбластьЗаголовок.Параметры[0] = "Планируемый график платежей по состоянию на " + Дата;

ТабДок.Вывести(ОбластьЗаголовок);

ПОЛУЧЕНИЕ\_ДЕНЕГ = 0;

ВЫПЛАТА\_ДЕНЕГ = 1;

ПокупателиИтог = 0;

ПоставщикиИтог = 0;

Для Каждого Вершина Из Результат.Путь Цикл

ОбластьЗапись.Параметры[0] = Вершина.Дата;

ОбластьЗапись.Параметры[1] = Вершина.Сальдо;

Если Вершина.Действие = ПОЛУЧЕНИЕ\_ДЕНЕГ Тогда

ОбластьЗапись.Параметры[2] = Вершина.Покупатель.Договор;

ОбластьЗапись.Параметры[3] = Вершина.Покупатель.Сумма;

ОбластьЗапись.Параметры[4] = "";

ОбластьЗапись.Параметры[5] = "";

ПокупателиИтог = ПокупателиИтог + Вершина.Покупатель.Сумма;

Иначе

ОбластьЗапись.Параметры[2] = "";

ОбластьЗапись.Параметры[3] = "";

ОбластьЗапись.Параметры[4] = Вершина.Поставщик.Договор;

ОбластьЗапись.Параметры[5] = Вершина.Поставщик.Оплачено;

ПоставщикиИтог = ПоставщикиИтог + Вершина.Поставщик.Оплачено;

КонецЕсли;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

ОбластьИтог.Параметры[0] = ПокупателиИтог;

ОбластьИтог.Параметры[1] = ПоставщикиИтог;

ТабДок.Вывести(ОбластьИтог);

КонецЕсли;

В данном блоке выводится найденный результат. Если результат неопределен, то выводится сообщение о нехватке средств для решения. Если результат найден, то выводится заголовок, и вершины (состояния) пути. В вершинах присутствует поле Действие, которое помогает определить, что именно делается на данном шаге (получение денег от покупателя, либо выплата денег поставщику). В зависимости от действия заполняются нужные поля области записи. После вывода всех вершин, выводится итог по покупателям и по поставщикам.

Функция СпланироватьРасчеты(Дата,Сальдо,Данные)

ПОЛУЧЕНИЕ\_ДЕНЕГ = 0;

ВЫПЛАТА\_ДЕНЕГ = 1;

Результат = Новый Структура;

Путь = Новый Массив;

Результат.Вставить("Сумма",0);

Результат.Вставить("Путь",Путь);

// проверяем не завершены ли получения и выплаты

Если Данные.Покупатели.Количество() = 0 И Данные.Поставщики.Количество() = 0 Тогда

Возврат Результат;

КонецЕсли;

РДата = Дата;

РСальдо = Сальдо;

// проверяем есть ли деньги на счете для оплаты поставщикам

Если Сальдо = 0 Тогда

Если Данные.Покупатели.Количество() = 0 Тогда

Возврат Неопределено;

КонецЕсли;

// денег нет, выполняем ближайшую получку денег от покупателя и планируем дальше, начиная со следующего после получения денег дня

Покупатель = Данные.Покупатели[0];

РДата = НачалоДня(Покупатель.Дата) + (60\*60\*24);

РСальдо = Покупатель.Сумма;

// убираем оформленную получку денег

Данные.Покупатели.Удалить(0);

// добавляем в путь состояние с получением оплаты

НовоеСостояние = Новый Структура;

НовоеСостояние.Вставить("Дата",Покупатель.Дата);

НовоеСостояние.Вставить("Сальдо",Сальдо);

НовоеСостояние.Вставить("Действие",ПОЛУЧЕНИЕ\_ДЕНЕГ);

НовоеСостояние.Вставить("Покупатель",Покупатель);

Результат.Путь.Добавить(НовоеСостояние);

КонецЕсли;

ЛучшийРезультат = Неопределено;

ЛучшийНачало = Неопределено;

// пробуем разные выплаты поставщикам (обходим их по очереди, выбираем результат с минимальной суммой выплат)

Для Каждого Поставщик Из Данные.Поставщики Цикл

НовоеСальдо = 0;

Начало = Неопределено;

НовыеДанные = Новый Структура;

НовыеДанные.Вставить("Поставщики",Новый Массив);

НовыеДанные.Вставить("Покупатели",Новый Массив);

Для Каждого Запись Из Данные.Покупатели Цикл

НоваяЗапись = Новый Структура;

НоваяЗапись.Вставить("Дата",Запись.Дата);

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма);

НовыеДанные.Покупатели.Добавить(НоваяЗапись);

КонецЦикла;

Для Каждого Запись Из Данные.Поставщики Цикл

НоваяЗапись = Неопределено;

Если Поставщик = Запись Тогда

Штраф = 0;

Дни = 0;

Если РДата >= Запись.Дата Тогда

Дни = ((КонецДня(РДата)+1)-НачалоДня(Запись.Дата))/(24\*60\*60);

Штраф = (Запись.Сумма \* (Запись.Штраф/100)) \* Дни;

КонецЕсли;

Если (Поставщик.Сумма + Штраф) > РСальдо Тогда

НоваяЗапись = Новый Структура;

Если Штраф = 0 Тогда

НоваяЗапись.Вставить("Дата",Запись.Дата);

Иначе

НоваяЗапись.Вставить("Дата",(КонецДня(РДата)+1));

КонецЕсли;

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма + Штраф - РСальдо);

НоваяЗапись.Вставить("Штраф",Запись.Штраф);

НовоеСальдо = 0;

Начало = Новый Структура;

Начало.Вставить("Дата",Запись.Дата);

Начало.Вставить("Договор",Запись.Договор);

Начало.Вставить("Оплачено",РСальдо);

Иначе

НовоеСальдо = РСальдо - (Запись.Сумма + Штраф);

Начало = Новый Структура;

Начало.Вставить("Дата",Запись.Дата);

Начало.Вставить("Договор",Запись.Договор);

Начало.Вставить("Оплачено",Запись.Сумма + Штраф);

КонецЕсли;

Иначе

НоваяЗапись = Новый Структура;

НоваяЗапись.Вставить("Дата",Запись.Дата);

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма);

НоваяЗапись.Вставить("Штраф",Запись.Штраф);

КонецЕсли;

Если НоваяЗапись <> Неопределено Тогда

НовыеДанные.Поставщики.Добавить(НоваяЗапись);

КонецЕсли;

КонецЦикла;

РезультатВетки = СпланироватьРасчеты(РДата,НовоеСальдо,НовыеДанные);

Если ЛучшийРезультат <> Неопределено И РезультатВетки <> Неопределено Тогда

Если (ЛучшийНачало.Оплачено + ЛучшийРезультат.Сумма) > (Начало.Оплачено + РезультатВетки.Сумма) Тогда

ЛучшийРезультат = РезультатВетки;

ЛучшийНачало = Начало;

КонецЕсли;

Иначе

ЛучшийРезультат = РезультатВетки;

ЛучшийНачало = Начало;

КонецЕсли;

КонецЦикла;

Если ЛучшийРезультат = Неопределено Тогда

Возврат Неопределено;

КонецЕсли;

// найден лучший результат, составляем итоговый результат

Результат.Сумма = ЛучшийНачало.Оплачено + ЛучшийРезультат.Сумма;

НоваяВершина = Новый Структура;

НоваяВершина.Вставить("Дата",РДата);

НоваяВершина.Вставить("Сальдо",РСальдо);

НоваяВершина.Вставить("Действие",ВЫПЛАТА\_ДЕНЕГ);

НоваяВершина.Вставить("Поставщик",ЛучшийНачало);

Результат.Путь.Добавить(НоваяВершина);

Для Каждого Вершина Из ЛучшийРезультат.Путь Цикл

Результат.Путь.Добавить(Вершина);

КонецЦикла;

Возврат Результат;

КонецФункции

Главная функция, производит поиск минимального пути в графе.

ПОЛУЧЕНИЕ\_ДЕНЕГ = 0;

ВЫПЛАТА\_ДЕНЕГ = 1;

В данном блоке определяются константы, для добавления вершин в результат (чтобы различать состояния с получением денег и состояния с выплатой денег).

Результат = Новый Структура;

Путь = Новый Массив;

Результат.Вставить("Сумма",0);

Результат.Вставить("Путь",Путь);

В данном блоке создаются переменные для результата выполнения функции, и хранения пути для результата.

// проверяем не завершены ли получения и выплаты

Если Данные.Покупатели.Количество() = 0 И Данные.Поставщики.Количество() = 0 Тогда

Возврат Результат;

КонецЕсли;

В данном блоке проверяется не завершен ли обход графа (закончились поступления и выплаты).

РДата = Дата;

РСальдо = Сальдо;

// проверяем есть ли деньги на счете для оплаты поставщикам

Если Сальдо = 0 Тогда

Если Данные.Покупатели.Количество() = 0 Тогда

Возврат Неопределено;

КонецЕсли;

// денег нет, выполняем ближайшую получку денег от покупателя и планируем дальше, начиная со следующего после получения денег дня

Покупатель = Данные.Покупатели[0];

РДата = НачалоДня(Покупатель.Дата) + (60\*60\*24);

РСальдо = Покупатель.Сумма;

// убираем оформленную получку денег

Данные.Покупатели.Удалить(0);

// добавляем в путь состояние с получением оплаты

НовоеСостояние = Новый Структура;

НовоеСостояние.Вставить("Дата",Покупатель.Дата);

НовоеСостояние.Вставить("Сальдо",Сальдо);

НовоеСостояние.Вставить("Действие",ПОЛУЧЕНИЕ\_ДЕНЕГ);

НовоеСостояние.Вставить("Покупатель",Покупатель);

Результат.Путь.Добавить(НовоеСостояние);

КонецЕсли;

В данном блоке определяются Дата и Сальдо с которым обходить исходящие из корня вершины. Если Сальдо равно нулю, то требуется получить деньги от покупателя. Если покупателей не осталось, то такой путь считается тупиковым и возвращается Неопределено. Иначе – запись покупателя убирается из данных, дата берется – следующий день после даты получения денег, а сальдо – количество полученых денег. Так же добавляется состояние отвечающее за получение денег от покупателя.

ЛучшийРезультат = Неопределено;

ЛучшийНачало = Неопределено;

// пробуем разные выплаты поставщикам (обходим их по очереди, выбираем результат с минимальной суммой выплат)

Для Каждого Поставщик Из Данные.Поставщики Цикл

НовоеСальдо = 0;

Начало = Неопределено;

НовыеДанные = Новый Структура;

НовыеДанные.Вставить("Поставщики",Новый Массив);

НовыеДанные.Вставить("Покупатели",Новый Массив);

Для Каждого Запись Из Данные.Покупатели Цикл

НоваяЗапись = Новый Структура;

НоваяЗапись.Вставить("Дата",Запись.Дата);

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма);

НовыеДанные.Покупатели.Добавить(НоваяЗапись);

КонецЦикла;

Для Каждого Запись Из Данные.Поставщики Цикл

НоваяЗапись = Неопределено;

Если Поставщик = Запись Тогда

Штраф = 0;

Дни = 0;

Если РДата >= Запись.Дата Тогда

Дни = ((КонецДня(РДата)+1)-НачалоДня(Запись.Дата))/(24\*60\*60);

Штраф = (Запись.Сумма \* (Запись.Штраф/100)) \* Дни;

КонецЕсли;

Если (Поставщик.Сумма + Штраф) > РСальдо Тогда

НоваяЗапись = Новый Структура;

Если Штраф = 0 Тогда

НоваяЗапись.Вставить("Дата",Запись.Дата);

Иначе

НоваяЗапись.Вставить("Дата",(КонецДня(РДата)+1));

КонецЕсли;

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма + Штраф - РСальдо);

НоваяЗапись.Вставить("Штраф",Запись.Штраф);

НовоеСальдо = 0;

Начало = Новый Структура;

Начало.Вставить("Дата",Запись.Дата);

Начало.Вставить("Договор",Запись.Договор);

Начало.Вставить("Оплачено",РСальдо);

Иначе

НовоеСальдо = РСальдо - (Запись.Сумма + Штраф);

Начало = Новый Структура;

Начало.Вставить("Дата",Запись.Дата);

Начало.Вставить("Договор",Запись.Договор);

Начало.Вставить("Оплачено",Запись.Сумма + Штраф);

КонецЕсли;

Иначе

НоваяЗапись = Новый Структура;

НоваяЗапись.Вставить("Дата",Запись.Дата);

НоваяЗапись.Вставить("Договор",Запись.Договор);

НоваяЗапись.Вставить("Сумма",Запись.Сумма);

НоваяЗапись.Вставить("Штраф",Запись.Штраф);

КонецЕсли;

Если НоваяЗапись <> Неопределено Тогда

НовыеДанные.Поставщики.Добавить(НоваяЗапись);

КонецЕсли;

КонецЦикла;

РезультатВетки = СпланироватьРасчеты(РДата,НовоеСальдо,НовыеДанные);

Если ЛучшийРезультат <> Неопределено И РезультатВетки <> Неопределено Тогда

Если (ЛучшийНачало.Оплачено + ЛучшийРезультат.Сумма) > (Начало.Оплачено + РезультатВетки.Сумма) Тогда

ЛучшийРезультат = РезультатВетки;

ЛучшийНачало = Начало;

КонецЕсли;

Иначе

ЛучшийРезультат = РезультатВетки;

ЛучшийНачало = Начало;

КонецЕсли;

КонецЦикла;

В данном блоке обходятся все возможные варианты оплаты из текущего состояния. Начальные данные копируются в НовыеДанные, чтобы все вытекающие состояния работали с начальными данными, а не испорченными предыдущим.

Далее составляем новое состояние, которое будет корнем графа, который нужно будет обойти рекурсивным вызовом. Все записи просто копируются, а та, по которой проводится оплата либо удаляется (если сумма полностью оплачена), либо сумма меняется на новую (сумма + штраф – сумма оплаты) и дата начала для расчета штрафа меняется на день после оплаты.

Далее вычисляется результат обхода графа, с новым корнем, если результат лучше сохраненного – сохраняем как лучший его.

Если ЛучшийРезультат = Неопределено Тогда

Возврат Неопределено;

КонецЕсли;

// найден лучший результат, составляем итоговый результат

Результат.Сумма = ЛучшийНачало.Оплачено + ЛучшийРезультат.Сумма;

НоваяВершина = Новый Структура;

НоваяВершина.Вставить("Дата",РДата);

НоваяВершина.Вставить("Сальдо",РСальдо);

НоваяВершина.Вставить("Действие",ВЫПЛАТА\_ДЕНЕГ);

НоваяВершина.Вставить("Поставщик",ЛучшийНачало);

Результат.Путь.Добавить(НоваяВершина);

Для Каждого Вершина Из ЛучшийРезультат.Путь Цикл

Результат.Путь.Добавить(Вершина);

КонецЦикла;

Возврат Результат;

Если лучший результат не найден – считаем ветку тупиковой и возвращаем Неопределено. Иначе – добавляем в путь результата вершину, которая приводит к лучшему результату, и все вершины пути лучшего результата.

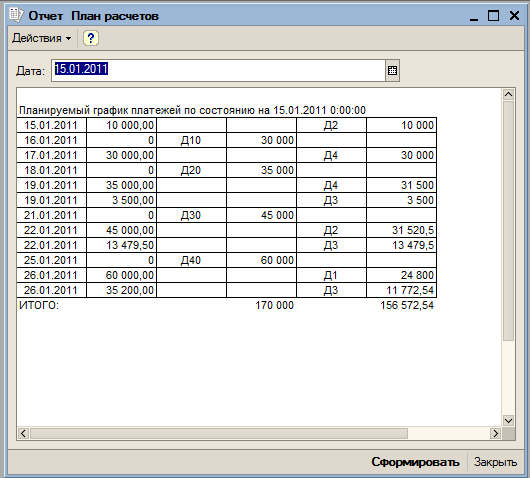
Процедура ПриОткрытии()

Дата = Дата(2011,01,15);

КонецПроцедуры

Данный блок необходим для задания даты по умолчанию – 15 января 2011.

Сформированный отчет



2011 год 1 тур

Требования

Крупная аптека ведет бухгалтерский учет в системе «1С:Предприятие 8». Лекарственные средства учитываются на счете 41 «Товары» в стоимостном и натуральном выражении. Аналитический учет на счете 41 ведется по товарам (лекарственным средствам). Сведения о товарной номенклатуре хранятся в справочнике «Товары». Для каждого товара в справочнике указан уникальный код лекарственного средства и наименование. Код имеет следующую структуру : xxx-yyy-zzz и представляет собой три трехзначных числа, отделяемых знаком «–». Лекарственные средства являются аналогами, если в их кодах совпадают средние три цифры. Например, лекарственные препараты с кодами 232-145-876 и 816-145-567 являются аналогами, поскольку у них совпадает средняя часть кода (145).

На основании данных информационной базы требуется сформировать отчет, в который необходимо включить пары лекарственных препаратов-аналогов, наиболее близких по цене. Если в базе имеются несколько пар аналогов с одинаковым разбросом цен, то в отчет надо включить более дешевые лекарственные средства, если среди них имеются несколько аналогов с одинаковыми ценами, то в отчет надо включить только один любой аналог.

Каркасная конфигурация

Задание необходимо выполнить в каркасной конфигурации, в которой имеется План счетов, содержащий счет 41, справочник «Товары», документ для ввода остатков товаров, при помощи которого введено несколько проводок по приходу товаров в дебет счета 41 и кредит условного счета 00.

ВНИМАНИЕ! Для отладки модуля отчета Вы можете добавлять в информационную базу каркасной конфигурации дополнительные товары и проводки. Жюри проводит тестирование на собственных данных, которые будут отличаться от данных, содержащихся в исходной информационной базе каркасной конфигурации.

Пример

Исходные данные

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код товара | Наименование | Остаток сумма | Остаток кол-во | Цена (расчет) |
| 111-222-333 | Изд-01 | 5000 | 5 | 1000.00 |
| 222-333-444 | Изд-02 | 15000 | 15 | 1000.00 |
| 111-555-222 | Изд-03 | 15000 | 5 | 3000.00 |
| 345-222-111 | Изд-04 | 25250 | 25 | 1010.00 |
| 321-555-003 | Изд-05 | 16500 | 5 | 3300.00 |
| 112-333-987 | Изд-06 | 5000 | 5 | 1000.00 |
| 345-222-765 | Изд-07 | 7105 | 7 | 1015.00 |
| 990-555-654 | Изд-08 | 5200 | 2 | 2600.00 |
| 990-555-658 | Изд-09 | 14000 | 5 | 2800.00 |

Итоговый отчет

Лекарственные средства-аналоги

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Лекарственное средство | | | Аналог | | |
| Код товара | Наименование | Цена | Код товара | Наименование | Цена |
| 345-222-111 | Изд-04 | 1010.00 | 345-222-765 | Изд-07 | 1015.00 |
| 222-333-444 | Изд-02 | 1000.00 | 112-333-987 | Изд-06 | 1000.00 |
| 990-555-654 | Изд-08 | 2600.00 | 990-555-658 | Изд-09 | 2800.00 |

Решение

Для получения пар аналогов с минимальной разницей в цене достаточно отсортировать все аналоги в порядке возрастанию цены (чтобы при одинаковой разнице в цене была взята первая пара – с меньшей ценой), далее обойти каждую группу аналогов, выбирая пару с минимальной разницей в цене. Рассмотрим пример (за исходные данные возьмем пример из требований):

Берем группы аналогов и сортируем их по возрастанию цены:

|  |  |  |
| --- | --- | --- |
| Группа | Товар | Цена |
| 555 | 990-555-654 | 2600 |
| 990-555-658 | 2800 |
| 111-555-222 | 3000 |
| 321-555-003 | 3300 |
| 333 | 222-333-444 | 1000 |
| 112-333-987 | 1000 |
| 222 | 111-222-333 | 1000 |
| 345-222-111 | 1010 |
| 345-222-765 | 1015 |

Обходим группы и определяем разницу в цене:

|  |  |  |  |
| --- | --- | --- | --- |
| Итерация | Товар 1 | Товар 2 | Разница |
| 1 | 990-555-658 | 990-555-654 | 200 |
| 2 | 111-555-222 | 990-555-658 | 200 |
| 3 | 321-555-003 | 111-555-222 | 300 |
| 4 | 222-333-444 | 112-333-987 | 0 |
| 5 | 345-222-111 | 111-222-333 | 10 |
| 6 | 345-222-765 | 345-222-111 | 5 |

После обхода сразу видны пары товаров с минимальной разницей в цене (если разница равна – берем ту пару, которая выше в пределах группы). Это пары:

|  |  |  |
| --- | --- | --- |
| Товар 1 | Товар 2 | Разница |
| 990-555-658 | 990-555-654 | 200 |
| 222-333-444 | 112-333-987 | 0 |
| 345-222-765 | 345-222-111 | 5 |

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок=ЭлементыФормы.ТабДок;

ТабДок.Очистить();

Макет=ПолучитьМакет("Макет");

Область=Макет.ПолучитьОбласть("Шапка");

ТабДок.Вывести(Область);

Запрос=Новый Запрос;

Запрос.Текст=

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Номенклатура,

| ЖурналПроводокОстатки.Субконто1.Код КАК Код,

| ЖурналПроводокОстатки.СуммаОстатокДт / ЖурналПроводокОстатки.КоличествоОстатокДт КАК Цена,

| ПОДСТРОКА(ЖурналПроводокОстатки.Субконто1.Код, 5, 3) КАК Артикул,

| 1 КАК КоличествоАртикулов

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет41, , ) КАК ЖурналПроводокОстатки

|

|УПОРЯДОЧИТЬ ПО

| ЖурналПроводокОстатки.Субконто1.Наименование";

Запрос.УстановитьПараметр("Дата",КонецДня(Дата));

Запрос.УстановитьПараметр("Счет41",ПланыСчетов.ПланСчетов.Товары);

ТабЗапроса=Запрос.Выполнить().Выгрузить();

ТабЗапроса.Колонки.Добавить("РазницаСоСледСтрокой");

ТабЗапроса.Сортировать("Цена");

ТабВрем=ТабЗапроса.Скопировать();

ТабВрем.Свернуть("Артикул","КоличествоАртикулов");

ТабВрем.Сортировать("Артикул");

Для Каждого ТекСтркоа из ТабВрем Цикл

Если ТекСтркоа.КоличествоАртикулов>1 Тогда

Отбор=Новый Структура;

Отбор.Вставить("Артикул", ТекСтркоа.Артикул);

товар1="";

товар2="";

СтрокиПоиска=ТабЗапроса.НайтиСтроки(Отбор);

МинРазница=строкиПоиска.Получить(0);

МинРазница=МинРазница.цена;

Для строк=0 По СтрокиПоиска.Количество()-2 цикл

строка = СтрокиПоиска.Получить(строк);

строка2= СтрокиПоиска.Получить(строк+1);

разница= строка.цена - строка2.цена;

строка.РазницаСоСледСтрокой = макс(разница,-разница);

если строка.РазницаСоСледСтрокой < МинРазница Тогда

МинРазница=строка.РазницаСоСледСтрокой;

НомТовар1=строк;

НомТовар2=строк+1;

конецЕсли;

КонецЦикла;

Товар1=СтрокиПоиска.Получить(НомТовар1);

Товар2=СтрокиПоиска.Получить(НомТовар2);

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.Код = Товар1.Код;

Область.Параметры.Номенклатура = Товар1.Номенклатура;

Область.Параметры.Цена = Товар1.Цена;

Область.Параметры.Код2 = Товар2.Код;

Область.Параметры.Номенклатура2 = Товар2.Номенклатура;

Область.Параметры.Цена2 = Товар2.Цена;

ТабДок.Вывести(Область);

//сообщить (сокрлп(товар1)+" "+сокрлп(товар2));

КонецЕсли;

КонецЦикла;

ТабДок.ТолькоПросмотр=Истина;

КонецПроцедуры

Процедура ПередОткрытием(Отказ, СтандартнаяОбработка)

Дата=ТекущаяДата();

КонецПроцедуры

Решение с сайта <http://олимпиада1с.рф>

Пояснения к коду:

ТабДок=ЭлементыФормы.ТабДок;

ТабДок.Очистить();

Макет=ПолучитьМакет("Макет");

Область=Макет.ПолучитьОбласть("Шапка");

ТабДок.Вывести(Область);

В данном блоке получается элемент формы ТабличныйДокумент, для вывода результата. Идет отчистка табличного документа и получение макета, по которому будет строиться таблица результата. После этого – ввыводится шапка в табличный документ.

Запрос=Новый Запрос;

Запрос.Текст=

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Номенклатура,

| ЖурналПроводокОстатки.Субконто1.Код КАК Код,

| ЖурналПроводокОстатки.СуммаОстатокДт / ЖурналПроводокОстатки.КоличествоОстатокДт КАК Цена,

| ПОДСТРОКА(ЖурналПроводокОстатки.Субконто1.Код, 5, 3) КАК Артикул,

| 1 КАК КоличествоАртикулов

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки(&Дата, Счет = &Счет41, , ) КАК ЖурналПроводокОстатки

|

|УПОРЯДОЧИТЬ ПО

| ЖурналПроводокОстатки.Субконто1.Наименование";

Запрос.УстановитьПараметр("Дата",КонецДня(Дата));

Запрос.УстановитьПараметр("Счет41",ПланыСчетов.ПланСчетов.Товары);

ТабЗапроса=Запрос.Выполнить().Выгрузить();

ТабЗапроса.Колонки.Добавить("РазницаСоСледСтрокой");

ТабЗапроса.Сортировать("Цена");

В данном блоке генерируется запрос к базе данных. В запросе выбираются поля Номенклатура, Код номенклатуры, Цена (как средневзвешенная по остатками на указанную дату), Артикул (3 цифры посреди кода) и поле КоличествоАртиклов (изначально у всех равно 1), остатки берутся на указанную дату, из счета 41 и идет сортировка по наименованию номенклатуры. После указания текста запроса указываются параметры (Дата, как конец дня указанного в форме в поле Дата, Счет41 – план счетов Товары). Далее запрос выполняется и выгружается в таблицу значений ТабЗапроса. В неё добавляется колонка РазницаСоСледСтрокой, которая будет использоваться при расчетах. И сортируется по Цене.

ТабВрем=ТабЗапроса.Скопировать();

ТабВрем.Свернуть("Артикул","КоличествоАртикулов");

ТабВрем.Сортировать("Артикул");

В данном блоке таблица значений копируется в ТабВрем и группируется по полю Артикул, с указанием количества элементов в поле КоличествоАртикулов. После этого идет сортировка по полю Артикул.

Для Каждого ТекСтркоа из ТабВрем Цикл

Если ТекСтркоа.КоличествоАртикулов>1 Тогда

Отбор=Новый Структура;

Отбор.Вставить("Артикул", ТекСтркоа.Артикул);

товар1="";

товар2="";

СтрокиПоиска=ТабЗапроса.НайтиСтроки(Отбор);

МинРазница=строкиПоиска.Получить(0);

МинРазница=МинРазница.цена;

Для строк=0 По СтрокиПоиска.Количество()-2 цикл

строка = СтрокиПоиска.Получить(строк);

строка2= СтрокиПоиска.Получить(строк+1);

разница= строка.цена - строка2.цена;

строка.РазницаСоСледСтрокой = макс(разница,-разница);

если строка.РазницаСоСледСтрокой < МинРазница Тогда

МинРазница=строка.РазницаСоСледСтрокой;

НомТовар1=строк;

НомТовар2=строк+1;

конецЕсли;

КонецЦикла;

Товар1=СтрокиПоиска.Получить(НомТовар1);

Товар2=СтрокиПоиска.Получить(НомТовар2);

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.Код = Товар1.Код;

Область.Параметры.Номенклатура = Товар1.Номенклатура;

Область.Параметры.Цена = Товар1.Цена;

Область.Параметры.Код2 = Товар2.Код;

Область.Параметры.Номенклатура2 = Товар2.Номенклатура;

Область.Параметры.Цена2 = Товар2.Цена;

ТабДок.Вывести(Область);

//сообщить (сокрлп(товар1)+" "+сокрлп(товар2));

КонецЕсли;

КонецЦикла;

В данном блоке идет обход групп артикулов. Если в группе больше одной записи (а значит есть 2 аналога), то из таблицы значений ТабЗапроса отбираются записи, соответствующие текущей группе и ищутся два товара с минимальной разницей в цене. Для этого обходятся все отобранные записи, сравниваются со следующей строкой и разницы сохраняется в поле РазницаСоСледСтрокой. Если разница меньше чем сохраненная разница записей, то данная пара запоминается, и её разница. После обхода отобранных записей найденная пара выводится в табличный документ результата.

ТабДок.ТолькоПросмотр=Истина;

Данная строка запрещает редактирование табличного документа с результатом.

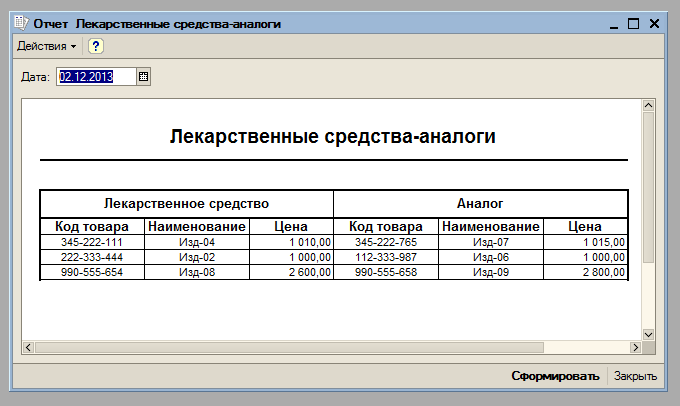
Процедура ПередОткрытием(Отказ, СтандартнаяОбработка)

Дата=ТекущаяДата();

КонецПроцедуры

Данный блок используется для установки значения по умолчанию поля Дата на форме (выставляется текущая дата).

Сформированный отчет



2010 год 2 тур

Требования

С 1 апреля 2010 г. организация переходит с ручного на автоматизированное ведение бухгалтерского учета. При учете основных средств организация применяет метод начисления амортизации способом «по сумме чисел лет срока полезного использования».

В компьютерной программе на счетах 01 «Основные средства» и 02 «Амортизация ОС» организован пообъектный аналитический учет в разрезе элементов справочника «Основные средства». В учетную систему по каждому объекту ОС на дату начала ведения учета введены следующие данные:

* Первоначальная стоимость ОС - дебетовое сальдо счета 01
* Накопленная сумма амортизации ОС на первое число месяца начала ведения автоматизированного учета - кредитовое сальдо счета 02
* Оставшийся срок полезного использования в месяцах (включая первый месяц начала ведения учета) - реквизит справочника «Основные средства».

Введя указанные данные учетные работники выяснили, что программа не сможет по этим данным производить начисление амортизации по объектам основных средств в течение оставшегося срока их полезного использования. Для последующего начисления амортизации необходимо ввести в программу:

* Год и месяц первого периода начисления амортизации,
* Полный срок полезного использования (в целых годах).

Необходимо смоделировать описанную ситуацию в программе «1С:Предприятие 8» и по имеющимся данным определить недостающие реквизиты. Результаты следует представить в виде отчета следующего вида.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Основное средство | Первоначальная стоимость ОС | Износ на дату начала ведения учета | Оставшийся срок полезного использования (мес) | Первый период начисления амортизации | | Полный срок полезного использования (лет) |
| месяц | год |
| ОС-0001 | 1620000,00 | 1593750,00 | 7 | 11 | 2002 | 8 |
| ОС-0002 | 4925907,00 | 3401221,50 | 37 | 5 | 2007 | 6 |
| ОС-0003 | 120000,00 | 65250,00 | 30 | 10 | 2008 | 4 |
| ОС-0004 | 1440000,00 | 1114000,00 | 48 | 4 | 2005 | 9 |
| ОС-0005 | 600000,00 | 300000,00 | 37 | н/о | н/о | н/о |

Если для некоторого объекта ОС указаны исходные данные, для которых не существует корректной схемы начисления амортизации методом «по сумме чисел лет...», то для этого объекта в последних трех графах отчета вместо результатов следует указать «н/о» (не определено).

Для справки: Методика ежемесячного начисления амортизации способом «по сумме чисел лет»

При начислении амортизации способом «по сумме чисел лет» согласно ПБУ 6/01 "...годовая сумма амортизационных отчислений определяется исходя из первоначальной стоимости объекта ОС и соотношения, в числителе которого - число лет, остающихся до конца срока полезного использования объекта, а в знаменателе - сумма чисел лет срока полезного использования объекта". Согласно требованию ПБУ 6/01 "...в течение отчетного года ежемесячная сумма амортизации определяется в размере 1/12 годовой суммы независимо от применяемого способа начисления амортизации". Таким образом, если первый период начисления амортизации ОС не совпадает с началом отчетного (календарного года), то до конца первого отчетного (календарного) года ежемесячно (в течение n оставшихся месяцев) списывается 1/12 годовой суммы первого года эксплуатации ОС, второй отчетный (календарный) год ежемесячно списывается 1/12 от суммы, приходящейся на оставшиеся месяцы первого года эксплуатации, и первые n месяцев второго года эксплуатации и т.д.

Исходные данные

В каркасной конфигурации имеется план счетов, содержащий счета 01 «Основные средства» и 02 «Амортизация ОС», для которых организован аналитический учет по субконто вида «Основные средства», которому соответствует одноименный справочник. Начальные остатки на счетах 01 и 02 ведены проводками в корреспонденции со счетом 00, в справочнике «Основные средства» заполнен реквизит СПИ – оставшийся срок полезного использования объекта ОС в месяцах. Всего в организации предполагается наличие до 10000 ОС. Максимальный срок эксплуатации ОС до 100 лет.

В отчете предусмотреть форму диалога, в которой указать дату - первое число месяца начала ведения учета, по умолчанию – 1 апреля 2010 г.

Решение

Рассмотрим начисление амортизации по сумме чисел лет для первой записи из примера, данного в требованиях.

Полный срок использования = 8

Начало использования = 11 2002

Стоимость ОС (S) = 1 620 000

Сумма чисел лет = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36

Так как начало срока амортизации не совпадает с началом календарного года, расчеты вычисляются так: сумма амортизации за год = сумма амортизации за n первых месяцев по коэффициенту текущего года + сумма амортизации за (12-n) месяцев по коэффициенту прошлого года.

|  |  |  |
| --- | --- | --- |
| Год | Коэф | Формула суммы |
| Результат |
| 2002 | 8/36 | (2/12)\*(8/36\*S) |
| 60000 |
| 2003 | 7/36 | (10/12)\*(8/36\*S)+(2/12)\*(7/36\*S) |
| 300000+52500=352500 |
| 2004 | 6/36 | (10/12)\*(7/36\*S)+(2/12)\*(6/36\*S) |
| 262500+45000=307500 |
| 2005 | 5/36 | (10/12)\*(6/36\*S)+(2/12)\*(5/36\*S) |
| 225000+37500=262500 |
| 2006 | 4/36 | (10/12)\*(5/36\*S)+(2/12)\*(4/36\*S) |
| 187500+30000=217500 |
| 2007 | 3/36 | (10/12)\*(4/36\*S)+(2/12)\*(3/36\*S) |
| 150000+22500=172500 |
| 2008 | 2/36 | (10/12)\*(3/36\*S)+(2/12)\*(2/36\*S) |
| 112500+15000=127500 |
| 2009 | 1/36 | (10/12)\*(2/36\*S)+(2/12)\*(1/36\*S) |
| 75000+7500=82500 |
| 2010 |  | (10/12)\*(1/36\*S) |
| 37500 |

Теперь попробуем вычислить число лет, зная только стоимость, количество оставшихся месяцев и какая сумма уже уплачена:

Осталось месяцев = 7

Уплачено = 1593750

Видно, что оставшиеся 7 месяцев приходятся на последний год начисления амортизации. Число лет можно найти из суммы чисел лет (по формуле суммы арифметической прогрессии – , в нашем случае, если взять x – число лет, формула будет – ), для нахождения суммы чисел лет можно использовать формулу, по которой рассчитывается сумма уплаты на год. Для начала, найдем сколько уплачивается за месяц (Стоимость-Уплачено)/(Осталось месяцев) = 3750 (А). Значит, А=(1/12)\*(1/B)\*S, где В – сумма чисел лет. Значит В = (S/А)/12, т.е. (1620000/3750)/12, что равно 36. Теперь из суммы чисел лет можно найти число лет. Преобразуем формулу суммы арифметической прогрессии к квадратному уравнению - . Решая уравнение получаем два корня – 8 и -9, так как число лет не может быть отрицательным, берем 8, то есть число лет полезного использования = 8, что соответствует данным в таблице.

Теперь рассмотрим вторую запись:

Осталось месяцев = 37

Стоимость = 4925907

Уплачено = 3401221,50

Конец уплаты = 01.05.13

Для определения суммы чисел лет, используем следующую формулу:

Где A – сумма, которую осталось уплатить за оставшиеся месяцы. b – сумма чисел лет. S – стоимость основного средства. Так как оплата будет идти ещё несколько лет, для расчета суммы чисел лет нужно взять уплаты за всё это время, не забывая, что начало оплаты получается на 5 месяц, а не на начало календарного года, а значит идет смещение по оплате (сумма за 5 месяцев предыдущего амортизационного года и 7 месяцев текущего). И не забывая, что часть за текущий календарный год уже уплачена (4 месяца уплачены), а значит нужно взять лишь 8/12 от суммы за текущий календарный год. Выводим b из данного уравнения, и получаем:

Данную формулу можно использовать для расчета количества лет в программе. Более подробно о реализации разобрано в разделе Реализация, пояснения к коду.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Макет = ПолучитьМакет("Макет");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ТабДок.Очистить();

ОсновныеСредства = Новый Массив();

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПроводкиДвиженияССубконто.СчетДт,

| ПроводкиДвиженияССубконто.СубконтоДт1,

| ПроводкиДвиженияССубконто.СчетКт,

| ПроводкиДвиженияССубконто.СубконтоКт1,

| ПроводкиДвиженияССубконто.Сумма,

| ОсновныеСредства.СПИ

|ИЗ

| РегистрБухгалтерии.Проводки.ДвиженияССубконто(, &Дата, , , ) КАК ПроводкиДвиженияССубконто

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ОсновныеСредства КАК ОсновныеСредства

| ПО ПроводкиДвиженияССубконто.СубконтоКт1 = ОсновныеСредства.Ссылка";

Запрос.УстановитьПараметр("Дата",Дата);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

// проверяем есть ли уже запись с субконто

Субконто = Неопределено;

Стоимость = 0;

Выплачено = 0;

Осталось = 0;

Если Выборка.СчетДт.Код = "01" Тогда

Субконто = Выборка.СубконтоДт1;

Стоимость = Выборка.Сумма;

ИначеЕсли Выборка.СчетКт.Код = "02" Тогда

Субконто = Выборка.СубконтоКт1;

Выплачено = Выборка.Сумма;

Осталось = Выборка.СПИ;

КонецЕсли;

Запись = Неопределено;

Для Каждого ОС Из ОсновныеСредства Цикл

Если ОС.Субконто = Субконто Тогда

Запись = ОС;

КонецЕсли;

КонецЦикла;

Если Запись = Неопределено Тогда

Запись = Новый Структура;

Запись.Вставить("Субконто",Субконто);

Запись.Вставить("Стоимость",Стоимость);

Запись.Вставить("Выплачено",Выплачено);

Запись.Вставить("Осталось",Осталось);

ОсновныеСредства.Добавить(Запись);

Иначе

Если Стоимость <> 0 Тогда

Запись.Стоимость = Стоимость;

КонецЕсли;

Если Выплачено <> 0 Тогда

Запись.Выплачено = Выплачено;

КонецЕсли;

Если Осталось <> 0 Тогда

Запись.Осталось = Осталось;

КонецЕсли;

КонецЕсли;

КонецЦикла;

Для Каждого ОС Из ОсновныеСредства Цикл

ОсталосьВыплатить = ОС.Стоимость - ОС.Выплачено;

СуммаЛет = 0;

Осталось = ОС.Осталось;

ГодКонца = Год(Дата);

МесяцКонца = Месяц(Дата);

Пока Осталось > 12 Цикл

Осталось = Осталось - 12;

ГодКонца = ГодКонца + 1;

КонецЦикла;

Если Осталось > 0 Тогда

МесяцКонца = МесяцКонца + Осталось;

Если МесяцКонца > 12 Тогда

МесяцКонца = МесяцКонца - 12;

ГодКонца = ГодКонца + 1;

КонецЕсли;

КонецЕсли;

Осталось = ОС.Осталось;

Если Осталось > МесяцКонца Тогда

СуммаЛет = (МесяцКонца/12); // начисления амортизации за последний календарный год

Осталось = Осталось - МесяцКонца;

Иначе

СуммаЛет = Осталось/12; // оставшаяся амортизация вся приходится на текущий календарный год

Осталось = 0;

КонецЕсли;

ТекГодКоэф = 1;

Пока Осталось > 0 Цикл

// сумма за этот год

Сумма = ((12-МесяцКонца)\*ТекГодКоэф + МесяцКонца\*(ТекГодКоэф+1))/12;

// если осталось больше года - берем всю сумму, иначе - часть суммы (в зависимости от оставшихся месяцев)

Если Осталось > 12 Тогда

СуммаЛет = СуммаЛет + Сумма;

Осталось = Осталось - 12;

Иначе

СуммаЛет = СуммаЛет + Сумма\*(Осталось/12);

Осталось = 0;

КонецЕсли;

ТекГодКоэф = ТекГодКоэф + 1;

КонецЦикла;

СуммаЧисел = Окр((ОС.Стоимость\*СуммаЛет)/ОсталосьВыплатить,0);

Дескр = 1 + 4\*СуммаЧисел\*2;

Если Sqrt(Дескр) <> Цел(Sqrt(Дескр)) Тогда

ОС.Вставить("Годы","н/о");

ОС.Вставить("МесяцНачала","н/о");

ОС.Вставить("ГодНачала","н/о");

Иначе

Годы1 = (-1 + Sqrt(Дескр))/2;

Годы2 = (-1 - Sqrt(Дескр))/2;

Годы = 0;

Если Годы1 > 0 Тогда

Годы = Годы1;

ИначеЕсли Годы2 > 0 Тогда

Годы = Годы2;

КонецЕсли;

ОС.Вставить("Годы",Годы);

ОС.Вставить("МесяцНачала",МесяцКонца);

ОС.Вставить("ГодНачала",ГодКонца - Годы);

КонецЕсли;

КонецЦикла;

ТабДок.Вывести(ОбластьЗаголовок);

Для Каждого ОС Из ОсновныеСредства Цикл

ОбластьЗапись.Параметры[0] = ОС.Субконто;

ОбластьЗапись.Параметры[1] = ОС.Стоимость;

ОбластьЗапись.Параметры[2] = ОС.Выплачено;

ОбластьЗапись.Параметры[3] = ОС.Осталось;

ОбластьЗапись.Параметры[4] = ОС.МесяцНачала;

ОбластьЗапись.Параметры[5] = ОС.ГодНачала;

ОбластьЗапись.Параметры[6] = ОС.Годы;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

КонецПроцедуры

Процедура ПриОткрытии()

Дата = Дата(2010,4,1);

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента;

Макет = ПолучитьМакет("Макет");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ТабДок.Очистить();

В данном блоке идет получение элемента формы ТабличныйДокумент для дальнейшего вывода результата. Далее получаются области макета, по которому будет строиться результат. После этого табличный документ очищается от старых данных.

ОсновныеСредства = Новый Массив();

Эта строка отвечает за создание массива ОсновныеСредства, в котором будут содержаться структуры с полями Субконто (основное средство), Стоимость (цена ос), Выплачено (сколько выплачено на дату начала учета), Осталось (сколько месяцев), МесяцНачала (месяц начала амортизации), ГодНачала (год начала амортизации), Годы (полный срок амортизации).

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПроводкиДвиженияССубконто.СчетДт,

| ПроводкиДвиженияССубконто.СубконтоДт1,

| ПроводкиДвиженияССубконто.СчетКт,

| ПроводкиДвиженияССубконто.СубконтоКт1,

| ПроводкиДвиженияССубконто.Сумма,

| ОсновныеСредства.СПИ

|ИЗ

| РегистрБухгалтерии.Проводки.ДвиженияССубконто(, &Дата, , , ) КАК ПроводкиДвиженияССубконто

| ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ОсновныеСредства КАК ОсновныеСредства

| ПО ПроводкиДвиженияССубконто.СубконтоКт1 = ОсновныеСредства.Ссылка";

Запрос.УстановитьПараметр("Дата",Дата);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

В данном блоке идет получение данных из базы данных посредством запроса. Выбираются данные проводок и сколько месяцев амортизации осталось (из справочника ОсновныеСредства). Дата берется указанная в форме.

Пока Выборка.Следующий() Цикл

// проверяем есть ли уже запись с субконто

Субконто = Неопределено;

Стоимость = 0;

Выплачено = 0;

Осталось = 0;

Если Выборка.СчетДт.Код = "01" Тогда

Субконто = Выборка.СубконтоДт1;

Стоимость = Выборка.Сумма;

ИначеЕсли Выборка.СчетКт.Код = "02" Тогда

Субконто = Выборка.СубконтоКт1;

Выплачено = Выборка.Сумма;

Осталось = Выборка.СПИ;

КонецЕсли;

Запись = Неопределено;

Для Каждого ОС Из ОсновныеСредства Цикл

Если ОС.Субконто = Субконто Тогда

Запись = ОС;

КонецЕсли;

КонецЦикла;

Если Запись = Неопределено Тогда

Запись = Новый Структура;

Запись.Вставить("Субконто",Субконто);

Запись.Вставить("Стоимость",Стоимость);

Запись.Вставить("Выплачено",Выплачено);

Запись.Вставить("Осталось",Осталось);

ОсновныеСредства.Добавить(Запись);

Иначе

Если Стоимость <> 0 Тогда

Запись.Стоимость = Стоимость;

КонецЕсли;

Если Выплачено <> 0 Тогда

Запись.Выплачено = Выплачено;

КонецЕсли;

Если Осталось <> 0 Тогда

Запись.Осталось = Осталось;

КонецЕсли;

КонецЕсли;

КонецЦикла;

В данном блоке данные полученные из запроса переносятся в массив ОсновныеСредства. Т.к. для заполнения одной записи массива требуется две записи из запроса – вначале определяется какая именно запись запроса сейчас имеется (запись со стоимостью покупки – счет 01 либо запись с выплаченной суммой и оставшимися месяцами – счет 02). После определения содержимого запроса, идет поиск записи в массиве ОсновныеСредства соответствующей Субконто запроса. Если запись с текущим Субконто есть в массиве – она заполняется недостающими данными, если же записи нет – она создается и добавляется в массив.

Для Каждого ОС Из ОсновныеСредства Цикл

ОсталосьВыплатить = ОС.Стоимость - ОС.Выплачено;

СуммаЛет = 0;

Осталось = ОС.Осталось;

ГодКонца = Год(Дата);

МесяцКонца = Месяц(Дата);

Пока Осталось > 12 Цикл

Осталось = Осталось - 12;

ГодКонца = ГодКонца + 1;

КонецЦикла;

Если Осталось > 0 Тогда

МесяцКонца = МесяцКонца + Осталось;

Если МесяцКонца > 12 Тогда

МесяцКонца = МесяцКонца - 12;

ГодКонца = ГодКонца + 1;

КонецЕсли;

КонецЕсли;

Осталось = ОС.Осталось;

Если Осталось > МесяцКонца Тогда

СуммаЛет = (МесяцКонца/12); // начисления амортизации за последний календарный год

Осталось = Осталось - МесяцКонца;

Иначе

СуммаЛет = Осталось/12; // оставшаяся амортизация вся приходится на текущий календарный год

Осталось = 0;

КонецЕсли;

ТекГодКоэф = 1;

Пока Осталось > 0 Цикл

// сумма за этот год

Сумма = ((12-МесяцКонца)\*ТекГодКоэф + МесяцКонца\*(ТекГодКоэф+1))/12;

// если осталось больше года - берем всю сумму, иначе - часть суммы (в зависимости от оставшихся месяцев)

Если Осталось > 12 Тогда

СуммаЛет = СуммаЛет + Сумма;

Осталось = Осталось - 12;

Иначе

СуммаЛет = СуммаЛет + Сумма\*(Осталось/12);

Осталось = 0;

КонецЕсли;

ТекГодКоэф = ТекГодКоэф + 1;

КонецЦикла;

СуммаЧисел = Окр((ОС.Стоимость\*СуммаЛет)/ОсталосьВыплатить,0);

Дескр = 1 + 4\*СуммаЧисел\*2;

Если Sqrt(Дескр) <> Цел(Sqrt(Дескр)) Тогда

ОС.Вставить("Годы","н/о");

ОС.Вставить("МесяцНачала","н/о");

ОС.Вставить("ГодНачала","н/о");

Иначе

Годы1 = (-1 + Sqrt(Дескр))/2;

Годы2 = (-1 - Sqrt(Дескр))/2;

Годы = 0;

Если Годы1 > 0 Тогда

Годы = Годы1;

ИначеЕсли Годы2 > 0 Тогда

Годы = Годы2;

КонецЕсли;

ОС.Вставить("Годы",Годы);

ОС.Вставить("МесяцНачала",МесяцКонца);

ОС.Вставить("ГодНачала",ГодКонца - Годы);

КонецЕсли;

КонецЦикла;

Главный блок – здесь определяются годы амортизации и даты начала. Сначала определяется сумма амортизации, которую предстоит выплатить за оставшиеся месяцы (Выплачено – Стоимость), Далее определяется Год и Месяц окончания амортизации. После этого начинается определение коэффициентов лет оставшейся амортизации. Сначала определяется число за последний год (когда коэффициент = 1/(сумма чисел лет)). Далее в цикле, пока оставшиеся месяцы есть, вычисляется полная сумма за год (часть месяцев с прошлого года + оставшиеся месяцы за этот год). После этого, если осталось больше года – прибавляется полученная полная сумма. Если же осталось меньше года – прибавляется полученная сумма умноженная на 1\12 за каждый оставшийся месяц. После этого находится сумма чисел лет (Стоимость \* сумма коэффициентов лет)/(осталось уплатить). Далее находится дискриминант квадратного уравнения (х^2 + х - (осталось уплатить) = 0). Если корень дискриминанта не целое число, значит найти количество лет амортизации нельзя (т.к. сумма чисел лет хоть как целое число, то корень дискриминанта должен быть целым числом), если же корень целый – то находятся корни квадратного уравнения и берется положительный (количество лет амортизации не может быть отрицательным).

ТабДок.Вывести(ОбластьЗаголовок);

Для Каждого ОС Из ОсновныеСредства Цикл

ОбластьЗапись.Параметры[0] = ОС.Субконто;

ОбластьЗапись.Параметры[1] = ОС.Стоимость;

ОбластьЗапись.Параметры[2] = ОС.Выплачено;

ОбластьЗапись.Параметры[3] = ОС.Осталось;

ОбластьЗапись.Параметры[4] = ОС.МесяцНачала;

ОбластьЗапись.Параметры[5] = ОС.ГодНачала;

ОбластьЗапись.Параметры[6] = ОС.Годы;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

В данном блоке идет вывод заголовка и записей таблицы результата в табличный документ.

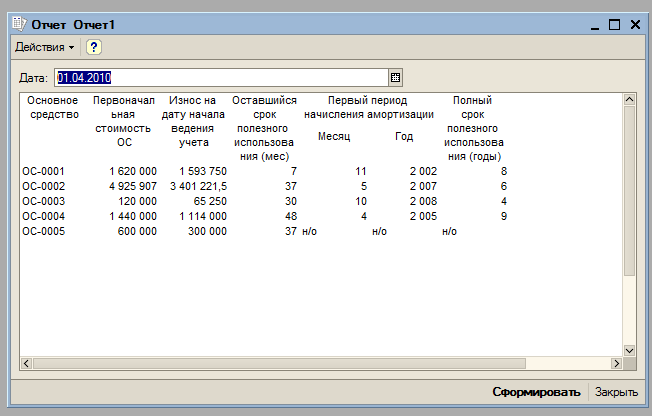
Процедура ПриОткрытии()

Дата = Дата(2010,4,1);

КонецПроцедуры

Данная процедура требуется для установки даты по умолчанию – 1 апреля 2010 года.

Сформированный отчет



2010 год 1 тур

Требования

Исходная ситуация

Организация учитывает товары на счете 41 в стоимостном (рублевом) и натуральном выражениях в аналитическом разрезе товаров и складов. Валютный учет на счете 41 не ведется.

Поступление товаров отражается проводками в дебет счета 41 (соответствующий субсчет), списание (расходование) товаров отражается проводками по кредиту счета 41. Отрицательные остатки товаров не допускаются.

Необходимо написать программу, которая выполнит анализ данных информационной базы бухгалтерского учета за указанный пользователем период и для каждого товара (независимо от склада) определит временной интервал от Даты-1 до Даты-2, в течение которого оценка запаса данного товара в валютном (долларовом) исчислении была максимальной. Дата-1 входит в искомый интервал, если на начало этого дня был впервые зафиксирован максимальный (в долларовом исчислении) запас. Дата-2 входит в искомый интервал, если это последний день, на конец которого сохранялся максимальный (в долларовом исчислении) запас.

Если максимальный запас был обнаружен в течение анализируемого периода не один раз, то в ответе указать наиболее продолжительный из всех периодов существования максимального запаса.

При решении задачи следует исходить из того, что курс доллара отражается в соответствующем регистре сведений ежедневно.

Исходные данные

В диалоге предусмотреть два поля ввода для ввода двух дат – даты начала анализируемого периода (включительно) и дату окончания анализируемого периода (включительно). Предусмотреть, чтобы по умолчанию устанавливался период за весь 2009 год: 01.01.09 - 31.12.09

Вывод результата

Результат должен быть представлен в виде отчета табличной формы, как это показано ниже.

Максимальные запасы товаров в USD за период 01.01.09 - 31.12.09

|  |  |  |  |
| --- | --- | --- | --- |
| Товары | Дата 1 | Дата 2 | Стоимость  (USD) |
| Товар 1 | 02.04.09 | 03.04.09 | 1500,00 |
| Товар 2 | 31.03.09 | 07.04.09 | 8512,21 |
| Товар 3 | 24.09.09 | 02.11.09 | 2987,64 |

Каркасная конфигурация

Для выполнения задания Вам предлагается каркасная конфигурация, в которой имеется План счетов, содержащий счет 41. Два документа – для отражения операций поступления и списания товаров. Также введено несколько проводок по приходу товаров в дебет счета 41 и кредит условного счета 00, а также по расходу – по кредиту счета 41 в корреспонденции счетом 00. В конфигурации имеется Регистр сведений «Курс доллара» в котором введены курсы доллара за несколько дат.

ВНИМАНИЕ! Для отладки модуля отчета Вы можете добавлять в информационную базу каркасной конфигурации дополнительные товары, склады, проводки, курсы. Жюри проводит тестирование на собственных данных, которые будут отличаться от данных, содержащихся в исходной информационной базе каркасной конфигурации.

Решение

Для решения задачи требуется определить стоимость товаров в валюте на каждый день из заданного диапазона и выбрать среди них самый большой период с максимальной стоимостью. Рассмотрим пример:

Валюта:

|  |  |
| --- | --- |
| Дата | Курс |
| 01.04.09 | 30 |
| 02.04.09 | 30 |
| 03.04.09 | 30 |
| 04.04.09 | 31 |
| 05.04.09 | 31 |
| 06.04.09 | 30 |
| 07.04.09 | 31 |
| 08.04.09 | 29 |
| 09.04.09 | 29 |

Движения товаров:

|  |  |  |
| --- | --- | --- |
| Товар | Действие | Стоимость |
| Товар-1 | приход | 50000 |
| Товар-1 | расход | 5000 |
| Товар-1 | расход | 10000 |

Тогда, составим стоимость товара на каждый день (в рублях и в валюте)

|  |  |  |
| --- | --- | --- |
| Товар | Стоимость (RUB) | Стоимость (USD) |
| Товар-1 | 50000 | 1666.66 |
| Товар-1 | 45000 | 1500 |
| Товар-1 | 45000 | 1500 |
| Товар-1 | 45000 | 1451.61 |
| Товар-1 | 45000 | 1451.61 |
| Товар-1 | 45000 | 1500 |
| Товар-1 | 45000 | 1451.61 |
| Товар-1 | 35000 | 1206.89 |
| Товар-1 | 35000 | 1206.89 |

Из составленной таблицы видно, что максимальная стоимость в валюте для товара «Товар-1» составляет 1666.66 и держится в течении 1 дня - 01.04.09, т.е. период будет 01.04.09 – 01.04.09.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Валюта = Новый Массив;

Товары = Новый Массив;

Результаты = Новый Массив;

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| КурсДоллара.Период,

| КурсДоллара.Курс

|ИЗ

| РегистрСведений.КурсДоллара КАК КурсДоллара

|ГДЕ

| КурсДоллара.Период МЕЖДУ &ДатаНач И &ДатаКон";

Запрос.УстановитьПараметр("ДатаНач",ДатаНач);

Запрос.УстановитьПараметр("ДатаКон",ДатаКон);

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

Пока Выборка.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Выборка.Период);

Запись.Вставить("Курс",Выборка.Курс);

Валюта.Добавить(Запись);

КонецЦикла;

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Период,

| ЖурналОперацийДвиженияССубконто.СчетДт.Код КАК СчетДтКод,

| ЖурналОперацийДвиженияССубконто.СубконтоДт1,

| ЖурналОперацийДвиженияССубконто.СчетКт.Код КАК СчетКтКод,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1,

| ЖурналОперацийДвиженияССубконто.Сумма

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| (ЖурналОперацийДвиженияССубконто.СчетДт = &Счет

| ИЛИ ЖурналОперацийДвиженияССубконто.СчетКт = &Счет)

| И ЖурналОперацийДвиженияССубконто.Период МЕЖДУ &ДатаНач И &ДатаКон";

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

Запрос.УстановитьПараметр("ДатаНач",ДатаНач);

Запрос.УстановитьПараметр("ДатаКон",ДатаКон);

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

Пока Выборка.Следующий() Цикл

Субконто = Неопределено;

Сумма = 0;

Если Выборка.СчетДтКод = ПланыСчетов.Счета.Товары.Код Тогда

Субконто = Выборка.СубконтоДт1;

Сумма = Выборка.Сумма;

ИначеЕсли Выборка.СчетКтКод = ПланыСчетов.Счета.Товары.Код Тогда

Субконто = Выборка.СубконтоКт1;

Сумма = -Выборка.Сумма;

КонецЕсли;

Запись = Неопределено;

Для Каждого Товар Из Товары Цикл

Если Товар.Ссылка = Субконто Тогда

Запись = Товар;

Прервать;

КонецЕсли;

КонецЦикла;

Если Запись = Неопределено Тогда

Запись = Новый Структура;

Запись.Вставить("Ссылка",Субконто);

Цены = Новый Массив;

Для День = 1 по Валюта.Количество() Цикл

Цены.Добавить(Сумма);

КонецЦикла;

Запись.Вставить("Цены",Цены);

Товары.Добавить(Запись);

РезЗапись = Новый Структура;

РезЗапись.Вставить("Ссылка",Субконто);

РезЗапись.Вставить("Дата1",ДатаНач);

РезЗапись.Вставить("Дата2",ДатаКон);

РезЗапись.Вставить("Сумма",0);

Результаты.Добавить(РезЗапись);

Иначе

ДеньНач = Окр(((Выборка.Период - ДатаНач) / (60\*60\*24)),0,0);

Для День = ДеньНач по Валюта.Количество() - 1 Цикл

Запись.Цены[День] = Запись.Цены[День] + Сумма;

КонецЦикла;

КонецЕсли;

КонецЦикла;

Для Ид = 0 По Результаты.ВГраница() Цикл

ВалСуммаМакс = 0;

Для День = 0 По Валюта.ВГраница() Цикл

ЦенаНовая = Товары[Ид].Цены[День] / Валюта[День].Курс;

Товары[Ид].Цены[День] = ЦенаНовая;

Если ЦенаНовая > ВалСуммаМакс Тогда

ВалСуммаМакс = ЦенаНовая;

КонецЕсли;

КонецЦикла;

Начало = Неопределено;

Конец = Неопределено;

НачалоМакс = 0;

КонецМакс = 0;

Для День = 0 По Валюта.ВГраница() Цикл

Если Товары[Ид].Цены[День] = ВалСуммаМакс Тогда

Если Начало = Неопределено Тогда

Начало = День;

Конец = День + 1;

КонецЕсли;

Иначе

Если Начало <> Неопределено Тогда

Конец = День;

Если (Конец - Начало) > (КонецМакс - НачалоМакс) Тогда

НачалоМакс = Начало;

КонецМакс = Конец;

КонецЕсли;

Начало = Неопределено;

Конец = Неопределено;

КонецЕсли;

КонецЕсли;

КонецЦикла;

Результаты[Ид].Дата1 = ДатаНач + (НачалоМакс \* 60\*60\*24);

Результаты[Ид].Дата2 = ДатаНач + (КонецМакс \* 60\*60\*24) - 1;

Результаты[Ид].Сумма = ВалСуммаМакс;

КонецЦикла;

Макет = ПолучитьМакет("Макет");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

ТабДок.Очистить();

Для Каждого Результат Из Результаты Цикл

ОбластьЗапись.Параметры[0] = Результат.Ссылка;

ОбластьЗапись.Параметры[1] = Результат.Дата1;

ОбластьЗапись.Параметры[2] = Результат.Дата2;

ОбластьЗапись.Параметры[3] = Результат.Сумма;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

КонецПроцедуры

Процедура ПриОткрытии()

ДатаНач = Дата(2009,04,01);

ДатаКон = Дата(2009,04,30);

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Валюта = Новый Массив;

Товары = Новый Массив;

Результаты = Новый Массив;

В данном блоке инициализируются массивы для хранения данных и получается табличный документ, для вывода результата. В массиве Валюта будут храниться структуры с полями Дата, Курс. В массиве Товары будут храниться структуры с полями Ссылка, Цены (тоже массив, индекс – день по порядку, значение – стоимость в этот день). В массиве Результаты будут храниться структуры с полями Ссылка, Дата 1, Дата 2, Сумма.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| КурсДоллара.Период,

| КурсДоллара.Курс

|ИЗ

| РегистрСведений.КурсДоллара КАК КурсДоллара

|ГДЕ

| КурсДоллара.Период МЕЖДУ &ДатаНач И &ДатаКон";

Запрос.УстановитьПараметр("ДатаНач",ДатаНач);

Запрос.УстановитьПараметр("ДатаКон",ДатаКон);

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

Пока Выборка.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Дата",Выборка.Период);

Запись.Вставить("Курс",Выборка.Курс);

Валюта.Добавить(Запись);

КонецЦикла;

В данном блоке идет получение данных из базы данных, получаются значения курса валюты за указанный период, и полученные значения записываются в массив Валюта.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Период,

| ЖурналОперацийДвиженияССубконто.СчетДт.Код КАК СчетДтКод,

| ЖурналОперацийДвиженияССубконто.СубконтоДт1,

| ЖурналОперацийДвиженияССубконто.СчетКт.Код КАК СчетКтКод,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1,

| ЖурналОперацийДвиженияССубконто.Сумма

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| (ЖурналОперацийДвиженияССубконто.СчетДт = &Счет

| ИЛИ ЖурналОперацийДвиженияССубконто.СчетКт = &Счет)

| И ЖурналОперацийДвиженияССубконто.Период МЕЖДУ &ДатаНач И &ДатаКон";

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

Запрос.УстановитьПараметр("ДатаНач",ДатаНач);

Запрос.УстановитьПараметр("ДатаКон",ДатаКон);

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

Пока Выборка.Следующий() Цикл

Субконто = Неопределено;

Сумма = 0;

Если Выборка.СчетДтКод = ПланыСчетов.Счета.Товары.Код Тогда

Субконто = Выборка.СубконтоДт1;

Сумма = Выборка.Сумма;

ИначеЕсли Выборка.СчетКтКод = ПланыСчетов.Счета.Товары.Код Тогда

Субконто = Выборка.СубконтоКт1;

Сумма = -Выборка.Сумма;

КонецЕсли;

Запись = Неопределено;

Для Каждого Товар Из Товары Цикл

Если Товар.Ссылка = Субконто Тогда

Запись = Товар;

Прервать;

КонецЕсли;

КонецЦикла;

Если Запись = Неопределено Тогда

Запись = Новый Структура;

Запись.Вставить("Ссылка",Субконто);

Цены = Новый Массив;

Для День = 1 по Валюта.Количество() Цикл

Цены.Добавить(Сумма);

КонецЦикла;

Запись.Вставить("Цены",Цены);

Товары.Добавить(Запись);

РезЗапись = Новый Структура;

РезЗапись.Вставить("Ссылка",Субконто);

РезЗапись.Вставить("Дата1",ДатаНач);

РезЗапись.Вставить("Дата2",ДатаКон);

РезЗапись.Вставить("Сумма",0);

Результаты.Добавить(РезЗапись);

Иначе

ДеньНач = Окр(((Выборка.Период - ДатаНач) / (60\*60\*24)),0,0);

Для День = ДеньНач по Валюта.Количество() - 1 Цикл

Запись.Цены[День] = Запись.Цены[День] + Сумма;

КонецЦикла;

КонецЕсли;

КонецЦикла;

В данном блоке идет получение товаров из базы данных, данные берутся из журнала движений. При записи данных в массив, сначала определяется действие записанное в журнале (приход товара либо расход), если приход – сумма записывается положительным значением, если расход – сумма будет отрицательной. Далее идет поиск записи относящейся к текущему товару в массиве Товары, если запись не найдена – она создается, при чем поле Цены это массив размером равным количеству дней в периоде (количество элементов массива Валюта). Так же при добавлении в массив Товары, создается новая запись в массиве Результаты, для хранения результата расчетов по этому товару. Если же запись в массиве Товары найдена, то для всех записей Цен, начиная с дня проводки до конечного дня прибавляем сумму (если проводка была приход – сумма увеличится, если же расход – сумма уменьшится).

Для Ид = 0 По Результаты.ВГраница() Цикл

ВалСуммаМакс = 0;

Для День = 0 По Валюта.ВГраница() Цикл

ЦенаНовая = Товары[Ид].Цены[День] / Валюта[День].Курс;

Товары[Ид].Цены[День] = ЦенаНовая;

Если ЦенаНовая > ВалСуммаМакс Тогда

ВалСуммаМакс = ЦенаНовая;

КонецЕсли;

КонецЦикла;

Начало = Неопределено;

Конец = Неопределено;

НачалоМакс = 0;

КонецМакс = 0;

Для День = 0 По Валюта.ВГраница() Цикл

Если Товары[Ид].Цены[День] = ВалСуммаМакс Тогда

Если Начало = Неопределено Тогда

Начало = День;

Конец = День + 1;

КонецЕсли;

Иначе

Если Начало <> Неопределено Тогда

Конец = День;

Если (Конец - Начало) > (КонецМакс - НачалоМакс) Тогда

НачалоМакс = Начало;

КонецМакс = Конец;

КонецЕсли;

Начало = Неопределено;

Конец = Неопределено;

КонецЕсли;

КонецЕсли;

КонецЦикла;

Результаты[Ид].Дата1 = ДатаНач + (НачалоМакс \* 60\*60\*24);

Результаты[Ид].Дата2 = ДатаНач + (КонецМакс \* 60\*60\*24) - 1;

Результаты[Ид].Сумма = ВалСуммаМакс;

КонецЦикла;

В данном блоке для всех товаров, хранящихся в массиве Результаты, расчитываются их периоды в течении которых стоимость в валюте была максимальна. Сначала находится максимальная стоимость в валюте. При чем, стоимость в рублях сразу же заменяется на стоимость в валюте, чтобы не повторять конвертацию при следующем обходе. После нахождения максимальной стоимости, ищется самый долгий период с данной стоимостью. Для этого обходятся цены за все дни, и если стоимость равна максимальной, то этот день один из нужных, а значит, если Начало не задано (ещё не было найдено начало периода с данной стоимостью), то задается Начало и Конец (как следующий день). Далее если стоимость не максимальна, то значит день не входит в искомые, и, если период уже начат (Начало определено), то запоминается период с максимальной стоимостью (если длина нового периода больше чем запомненного максимального, то запоминаем этот период как максимальный). Далее обнуляем Начало и Конец для нахождения следующего периода с максимальной стоимостью. После обхода Результаты поиска сохраняются. Начало и Конец переводятся в даты, относительно даты начала заданного периода.

Макет = ПолучитьМакет("Макет");

ОбластьЗапись = Макет.ПолучитьОбласть("Запись");

В данном блоке получается макет и его область, для вывода результата.

ТабДок.Очистить();

Для Каждого Результат Из Результаты Цикл

ОбластьЗапись.Параметры[0] = Результат.Ссылка;

ОбластьЗапись.Параметры[1] = Результат.Дата1;

ОбластьЗапись.Параметры[2] = Результат.Дата2;

ОбластьЗапись.Параметры[3] = Результат.Сумма;

ТабДок.Вывести(ОбластьЗапись);

КонецЦикла;

В данном блоке табличный документ очищается от старых данных и выводятся результаты.

Процедура ПриОткрытии()

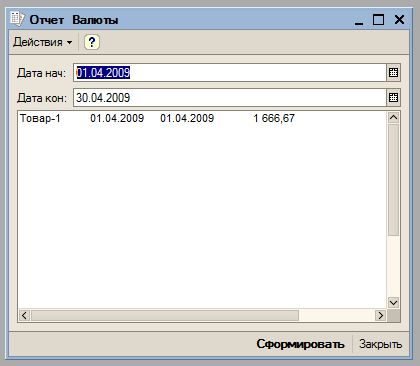
ДатаНач = Дата(2009,04,01);

ДатаКон = Дата(2009,04,30);

КонецПроцедуры

В данном блоке устанавливаются даты по умолчанию.

Сформированный отчет



2009 год 2 тур

Требования

Хозяйственная ситуация

Крупная торговая фирма «Ювелирторг» занимается розничной продажей ювелирных изделий. Товарная номенклатура исчисляется тысячами ювелирных изделий.

Менеджеры фирмы обратили внимание, на то, что среди их клиентов очень популярно приобретать товар комплектами из двух разных изделий. (Например, кольцо и серьги, кулон и цепочка, серьги и брошка и т.п.). Руководство решило для оформления витрин использовать популярные среди клиентов комплекты из двух изделий со следующим ограничением – все популярные пары изделий выложить на витрины (при наличии соответствующих изделий в запасе по данным натурального учета), чтобы при этом ни одно изделие на витринах не повторялось.

Исходные данные

Вся информация о товарах и продажах хранится в базе «1С:Предприятия». Первоначальные остатки по товарам вводятся одной операцией, содержащей проводки в дебет счета 41 с кредита счета 00.

Продажи отражаются в ИБ проводками в д-т счета 90 и кр-т счета 41 на сумму и количество отпущенного товара. На счете 41 ведется аналитика по товарам (ювелирным изделиям), на счете 90 аналитический учет (условно) не ведется. Считается, что операция отражает продажу комплекта из n изделий, если она содержит проводки по продаже n различных изделий вне зависимости от количества каждого из изделий, указанных в проводках. Если операция содержит несколько проводок по продаже только одного изделия, то эта операция не считается операцией продажи комплекта. Если в проводке по кредиту счета 41 указано нулевое количество, то такая проводка не считается проводкой по продаже изделия.

Анализируемый период задается в диалоге с пользователем. По умолчанию устанавливается период от начала текущего года до момента формирования отчета.

Задача

Составить программу, которая отбирала бы из ИБ операции по продаже комплектов из двух изделий, сформировала список из максимально возможного количества комплектов (пар), которые можно выложить на витрины, соблюдая названные выше условия, – наличие изделий в запасе и уникальность (неповторяемость) изделий, выкладываемых на витрины.

Выходная информация

Результатную информацию представить в виде отчета

|  |  |  |
| --- | --- | --- |
| № | Комплекты для витрины | |
| п/п | Изделие 1 | Изделие 2 |
| 1 | Кольцо рубировое | Серьги изумрудные |
| 2 | Браслет платиновый | Цепочка золотая |
| 3 | Серьги жемчужные | Перстень с бриллиантом |

Решение

Для решения задачи требуется получить проводки, содержащие продажу пар товаров и вывести те пары, товары из которых ещё есть и не были использованы ранее. Задача проста и не требует особых дополнений к пояснению кода.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Макет = ПолучитьМакет("Макет");

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ

| ЖурналПроводокДвиженияССубконто.СубконтоКт1 КАК Товар,

| ВложенныйЗапрос.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК ЖурналПроводокДвиженияССубконто

| ВНУТРЕННЕЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| ЖурналПроводокОстатки.КоличествоОстаток КАК КоличествоОстаток,

| ЖурналПроводокОстатки.Субконто1 КАК Субконто1

| ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

| ГДЕ

| ЖурналПроводокОстатки.Счет.Код = ""41"") КАК ВложенныйЗапрос

| ПО ЖурналПроводокДвиженияССубконто.СубконтоКт1 = ВложенныйЗапрос.Субконто1

|ГДЕ

| ВложенныйЗапрос.КоличествоОстаток > 0";

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

Товары = Новый Массив();

Пока Выборка.Следующий() Цикл

Если Выборка.Количество > 0 Тогда

Запись = Новый Структура;

Запись.Вставить("Ссылка",Выборка.Товар.Ссылка);

Запись.Вставить("НаВитрине",ложь);

Товары.Добавить(Запись);

КонецЕсли;

КонецЦикла;

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Движения1.СубконтоКт1 КАК Товар1,

| Движения2.СубконтоКт1 КАК Товар2

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК Движения1

| ВНУТРЕННЕЕ СОЕДИНЕНИЕ РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК Движения2

| ПО Движения1.Регистратор = Движения2.Регистратор

| И Движения1.СубконтоКт1 <> Движения2.СубконтоКт1

| ВНУТРЕННЕЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| Движения.Регистратор КАК Регистратор,

| КОЛИЧЕСТВО(Движения.НомерСтроки) КАК КолСтрок

| ИЗ

| РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК Движения

|

| СГРУППИРОВАТЬ ПО

| Движения.Регистратор) КАК ВложенныйЗапрос

| ПО Движения1.Регистратор = ВложенныйЗапрос.Регистратор

|ГДЕ

| ВложенныйЗапрос.КолСтрок = 2

| И Движения1.НомерСтроки = 1

| И Движения2.НомерСтроки = 2";

Результат = Запрос.Выполнить();

ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();

ТабДок.Вывести(ОбластьШапкаТаблицы);

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

Товар1 = Неопределено;

Товар2 = Неопределено;

Для Каждого Товар Из Товары Цикл

Если Товар.Ссылка = Выборка.Товар1.Ссылка Тогда

Товар1 = Товар;

КонецЕсли;

Если Товар.Ссылка = Выборка.Товар2.Ссылка Тогда

Товар2 = Товар;

КонецЕсли;

КонецЦикла;

Если Товар1 = Неопределено ИЛИ Товар2 = Неопределено Тогда

Продолжить;

КонецЕсли;

Если Товар1.НаВитрине ИЛИ Товар2.НаВитрине Тогда

Продолжить;

КонецЕсли;

Товар1.НаВитрине = Истина;

Товар2.НаВитрине = Истина;

ОбластьДетальныхЗаписей.Параметры.Заполнить(Выборка);

ТабДок.Вывести(ОбластьДетальныхЗаписей);

КонецЦикла;

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Макет = ПолучитьМакет("Макет");

В данном блоке идет получение табличного документа (для дальнейшего вывода результата) и получение макета, по которому будет строиться таблица результата.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ

| ЖурналПроводокОстатки.Субконто1 КАК Товар

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет.Код = ""41""

| И ЖурналПроводокОстатки.КоличествоОстаток > 0";

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

Товары = Новый Массив();

Пока Выборка.Следующий() Цикл

Запись = Новый Структура;

Запись.Вставить("Ссылка",Выборка.Товар.Ссылка);

Запись.Вставить("НаВитрине",ложь);

Товары.Добавить(Запись);

КонецЦикла;

В данном блоке идет получение товаров и их количества, с дальнейшим добавлением товара в массив Товары. Из базы данных берутся товары, из остатков по счету 41, количество которых больше нуля. В массив добавляются структуры с полями Ссылка (ссылка на товар) и НаВитрине (флаг, чтобы отметить, что товар выложен на витрину).

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Движения1.СубконтоКт1 КАК Товар1,

| Движения2.СубконтоКт1 КАК Товар2

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК Движения1

| ВНУТРЕННЕЕ СОЕДИНЕНИЕ РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК Движения2

| ПО Движения1.Регистратор = Движения2.Регистратор

| И Движения1.СубконтоКт1 <> Движения2.СубконтоКт1

| ВНУТРЕННЕЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| Движения.Регистратор КАК Регистратор,

| КОЛИЧЕСТВО(Движения.НомерСтроки) КАК КолСтрок

| ИЗ

| РегистрБухгалтерии.ЖурналПроводок.ДвиженияССубконто КАК Движения

|

| СГРУППИРОВАТЬ ПО

| Движения.Регистратор) КАК ВложенныйЗапрос

| ПО Движения1.Регистратор = ВложенныйЗапрос.Регистратор

|ГДЕ

| ВложенныйЗапрос.КолСтрок = 2

| И Движения1.НомерСтроки = 1

| И Движения2.НомерСтроки = 2";

Результат = Запрос.Выполнить();

В данном блоке выполняется запрос, получающий из базы данных пары товаров, содержащиеся в проводках, которые фиксируют продажу двух товаров, чтобы избежать дубликации данных типа «Товар1 – Товар2», «Товар2 – Товар1» фиксируется, что первый товар берется из Движения1, а второй товар из Движения2 (объединение таблицы ЖурналПроводок.ДвиженияССубконто самой с собой).

ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();

ТабДок.Вывести(ОбластьШапкаТаблицы);

В данном блоке идет получение областей макета, очистка табличного документа от старых данных и вывод шапки таблицы.

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

Товар1 = Неопределено;

Товар2 = Неопределено;

Для Каждого Товар Из Товары Цикл

Если Товар.Ссылка = Выборка.Товар1.Ссылка Тогда

Товар1 = Товар;

КонецЕсли;

Если Товар.Ссылка = Выборка.Товар2.Ссылка Тогда

Товар2 = Товар;

КонецЕсли;

КонецЦикла;

Если Товар1 = Неопределено ИЛИ Товар2 = Неопределено Тогда

Продолжить;

КонецЕсли;

Если Товар1.НаВитрине ИЛИ Товар2.НаВитрине Тогда

Продолжить;

КонецЕсли;

Товар1.НаВитрине = Истина;

Товар2.НаВитрине = Истина;

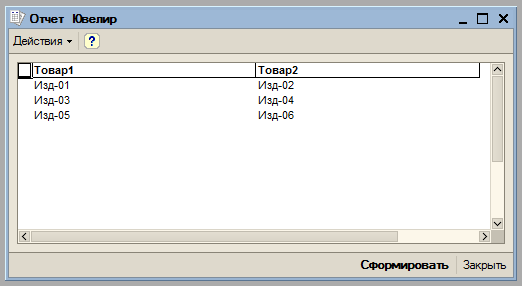
ОбластьДетальныхЗаписей.Параметры.Заполнить(Выборка);

ТабДок.Вывести(ОбластьДетальныхЗаписей);

КонецЦикла;

В данном блоке берется выборка из запроса с парами, для каждой пары оба товара ищутся в массиве Товары. Если какой то из товаров не найден в таблице Товары – значит этот товар уже отсутствует на складе и данная пара пропускается. Если какой то товар уже выложен на витрине – пропускаем данную пару. Далее отмечаем, что товары выложены на витрину и пара выводится в табличный документ.

Сформированный отчет



2009 год 1 тур

Требования

Крупная компания, представляющая собой группу взаимосвязанных организаций (порядка 100), ведет бухгалтерский учет в единой информационной базе в программе «1С:Бухгалтерия 8». Многофирменный учет организован с использованием измерения регистра бухгалтерии «Организации». (В версии 7.7 многофирменный учет организован при помощи разделителя учета "Организации").

Среди операций, проводимых между организациями, имеют место взаимные перечисления денежных средств – внутренние платежи.

Операция перечисления денежных средств у организации-плательщика  отражается проводкой д-т 79 «Внутренние расчеты» - кр-т 51 «Расчетный счет», а у организации-получателя  -  проводкой д-т 51 – кр-т 79 на такую же сумму.

На счете 79 ведется аналитический учет  по субконто «Организации».

Значения измерения «Организации», также как и субконто «Организации» определяются в едином справочнике «Организации».

Руководство компании принимает решение послать в организации компании ревизоров для проведения тотальной выездной проверки всех внутренних платежей, совершенных за некоторый период. Для проверки  всех внутренних платежей, совершенных между двумя организациями, ревизору  достаточно  посетить только одну из этих организаций.

Необходимо  на основании анализа ИБ бухгалтерского учета за заданный период определить, в какие организации следует направить ревизоров, чтобы обеспечить одновременную проверку всех внутренних платежей за данный период минимальным количеством ревизоров. При этом важен только лишь факт наличия операций, а количество операций, проведенных организациями, и суммы перечислений, роли не играют.

Например, если организация X получала (перечисляла) деньги от организации Y и получала (перечисляла) деньги от организации  Z, то для проверки всех внутренних платежей между этими организациями достаточно послать одного ревизора только в одну организацию Х.

Исходные данные представлены в информационной базе каркасной конфигурации, которая содержит

* План счетов бухгалтерского учета, с введенными в него счетами для отражения внутренних платежей на счетах бухгалтерского учета в регистре -бухгалтерии «Журнал операций».
* Справочник «Организации», содержащий наименование организаций, входящих в Группу.
* Документ «Ввод данных» отражающий операции перечисления денежных средств одновременно в системе бухгалтерского учета двух организаций – плательщика и получателя.

Период, за который анализируются операции, задается в форме диалога отчета в виде: «Дата начала -  Дата окончания». По умолчанию должен устанавливаться период, соответствующий 2009 году в целом.

Результат работы программы должен представлять список наименований организаций, которые должны быть посещены ревизорами.  Результат выдать в виде отчета следующего вида.

|  |  |  |
| --- | --- | --- |
| № | Код организации | Наименование организации |
| 1 | 103 | Пронто |
| 2 | 93 | Пунто |
| 3 | 13 | Паронто |

Решение

Рассмотрим пример (так как важен лишь факт наличия операции между двумя организациями, начальные данные представим в виде связей платежей и получений):

|  |  |
| --- | --- |
| Платежи | Получения |
| 02-01 | 01-02 |
| 03-02 | 02-03 |
| 05-02 | 02-05 |
| 06-05 | 05-06 |
| 04-05 | 05-04 |
| 03-05 | 05-03 |

Теперь представим данные в виде взаиморасчетов организаций (проведем соответствие каждой организации, с теми, с которыми идут расчеты):

|  |  |
| --- | --- |
| Организация | Связи |
| 01 | 02 |
| 02 | 01 |
| 03 |
| 05 |
| 03 | 02 |
| 05 |
| 04 | 05 |
| 05 | 02 |
| 03 |
| 04 |
| 06 |
| 06 | 05 |

Теперь следуем следующему алгоритму: пока есть не посещенные организации, посещаем ту, которая имеет больше всех связей и отмечаем те организации, с которыми данная организация связывается, если у организации связи кроме посещенной нет больше организаций – считаем организацию посещенной. Рассмотрим таблицу (выделенные жирным организации – посещенные, выделенные курсивом связи – уже проверенные, подчеркнутые организации – те организации, проводки которых проверены при посещении другой организации):

|  |  |  |
| --- | --- | --- |
| Итерация | Организация | Связи |
| 1 | 01 | 02 |
| 02 | 01 |
| 03 |
| 05 |
| 03 | 02 |
| 05 |
| 04 | 05 |
| 05 | 02 |
| 03 |
| 04 |
| 06 |
| 06 | 05 |
| У организации 05 максимальное количество связей – посещаем её, и проверяем организации-связи, требуется ли их посещать, по причине наличия связей с другими организациями | | |
| 2 | 01 | 02 |
| 02 | 01 |
| 03 |
| *05* |
| 03 | 02 |
| *05* |
| 04 | *05* |
| **05** | *02* |
| *03* |
| *04* |
| *06* |
| 06 | *05* |
| После посещения организации 05 отпала необходимость в посещении организаций 04 и 06, выбираем новую организацию для посещения (по количеству связей) – это организация 02 (осталось 2 не проверенные связи – 01 и 03), посещаем её | | |
| 3 | 01 | *02* |
| **02** | *01* |
| *03* |
| *05* |
| 03 | *02* |
| *05* |
| 04 | *05* |
| **05** | *02* |
| *03* |
| *04* |
| *06* |
| 06 | *05* |
| После посещения организации 02 проводки организаций 01 и 03 тоже считаются проверенными | | |

В результате решение найдено – требуется посетить организации 05 и 02, чтобы проверить все взаиморасчеты.

Реализация

Процедура КнопкаСформироватьНажатие(Кнопка)

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьДанные = Макет.ПолучитьОбласть("Данные");

Взаимосвязи = Новый Соответствие;

Обход = Новый Массив;

// организации должники

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации КАК Организация1,

| ЖурналОперацийДвиженияССубконто.СубконтоДт1 КАК Организация2

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетДт.Код = ""79""

|

|УПОРЯДОЧИТЬ ПО

| ЖурналОперацийДвиженияССубконто.Организации";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДобавитьВзаимосвязь(Взаимосвязи,Выборка);

КонецЦикла;

// организации кредиторы

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации КАК Организация1,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1 КАК Организация2

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетДт.Код = ""51""

|

|УПОРЯДОЧИТЬ ПО

| ЖурналОперацийДвиженияССубконто.Организации";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДобавитьВзаимосвязь(Взаимосвязи,Выборка);

КонецЦикла;

Пока Взаимосвязи.Количество() > 0 Цикл

Максимальный = Неопределено;

Для Каждого Запись Из Взаимосвязи Цикл

Если Максимальный = Неопределено Тогда

Максимальный = Запись;

ИначеЕсли Максимальный.Значение.Количество() < Запись.Значение.Количество() Тогда

Максимальный = Запись;

КонецЕсли;

КонецЦикла;

Обход.Добавить(Максимальный.Ключ);

Для Каждого Связь Из Максимальный.Значение Цикл

Взаимосвязь = Взаимосвязи.Получить(Связь);

Индекс = Взаимосвязь.Найти(Максимальный.Ключ);

Взаимосвязь.Удалить(Индекс);

Если Взаимосвязь.Количество() = 0 Тогда

Взаимосвязи.Удалить(Связь);

КонецЕсли;

КонецЦикла;

Взаимосвязи.Удалить(Максимальный.Ключ);

КонецЦикла;

ТабДок.Очистить();

ТабДок.Вывести(ОбластьЗаголовок);

Номер = 1;

Для Каждого Организация Из Обход Цикл

ОбластьДанные.Параметры[0] = Номер;

ОбластьДанные.Параметры[1] = Организация.Код;

ОбластьДанные.Параметры[2] = Организация.Наименование;

ТабДок.Вывести(ОбластьДанные);

Номер = Номер + 1;

КонецЦикла;

КонецПроцедуры

Процедура ДобавитьВзаимосвязь(Взаимосвязи,Выборка)

Взаимосвязи1 = Взаимосвязи.Получить(Выборка.Организация1);

Взаимосвязи2 = Взаимосвязи.Получить(Выборка.Организация2);

Если Взаимосвязи1 = Неопределено Тогда

Взаимосвязи1 = Новый Массив;

Взаимосвязи.Вставить(Выборка.Организация1,Взаимосвязи1);

КонецЕсли;

Если Взаимосвязи2 = Неопределено Тогда

Взаимосвязи2 = Новый Массив;

Взаимосвязи.Вставить(Выборка.Организация2,Взаимосвязи2);

КонецЕсли;

Если Взаимосвязи1.Найти(Выборка.Организация2) = Неопределено Тогда

Взаимосвязи1.Добавить(Выборка.Организация2);

КонецЕсли;

Если Взаимосвязи2.Найти(Выборка.Организация1) = Неопределено Тогда

Взаимосвязи2.Добавить(Выборка.Организация1);

КонецЕсли;

КонецПроцедуры

Автор: Михайлов Алексей

Пояснения к коду:

ТабДок = ЭлементыФормы.ПолеТабличногоДокумента1;

Макет = ПолучитьМакет("Макет");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьДанные = Макет.ПолучитьОбласть("Данные");

Взаимосвязи = Новый Соответствие;

Обход = Новый Массив;

В данном блоке идет получение поля табличного документа, для вывода результата, получение макета, для построения таблицы результата, областей макета и создание массива с результатом – Обход, который будет содержать организации, которые требуется посетить для проверок всех взаиморасчетов, и создание соответствия Взаимосвязи, для хранения организаций с их связями.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации КАК Организация1,

| ЖурналОперацийДвиженияССубконто.СубконтоДт1 КАК Организация2

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетДт.Код = ""79""

|

|УПОРЯДОЧИТЬ ПО

| ЖурналОперацийДвиженияССубконто.Организации";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДобавитьВзаимосвязь(Взаимосвязи,Выборка);

КонецЦикла;

В данном блоке идет получение платежей из базы данных, и результаты выполнения запроса добавляются в соответствие Взаимосвязи.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийДвиженияССубконто.Организации КАК Организация1,

| ЖурналОперацийДвиженияССубконто.СубконтоКт1 КАК Организация2

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.ДвиженияССубконто КАК ЖурналОперацийДвиженияССубконто

|ГДЕ

| ЖурналОперацийДвиженияССубконто.СчетДт.Код = ""51""

|

|УПОРЯДОЧИТЬ ПО

| ЖурналОперацийДвиженияССубконто.Организации";

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Пока Выборка.Следующий() Цикл

ДобавитьВзаимосвязь(Взаимосвязи,Выборка);

КонецЦикла;

В данном блоке так же идет получение платежей и добавление в соответствие Взаимосвязи, но получение идет со счета 51.

Пока Взаимосвязи.Количество() > 0 Цикл

Максимальный = Неопределено;

Для Каждого Запись Из Взаимосвязи Цикл

Если Максимальный = Неопределено Тогда

Максимальный = Запись;

ИначеЕсли Максимальный.Значение.Количество() < Запись.Значение.Количество() Тогда

Максимальный = Запись;

КонецЕсли;

КонецЦикла;

Обход.Добавить(Максимальный.Ключ);

Для Каждого Связь Из Максимальный.Значение Цикл

Взаимосвязь = Взаимосвязи.Получить(Связь);

Индекс = Взаимосвязь.Найти(Максимальный.Ключ);

Взаимосвязь.Удалить(Индекс);

Если Взаимосвязь.Количество() = 0 Тогда

Взаимосвязи.Удалить(Связь);

КонецЕсли;

КонецЦикла;

Взаимосвязи.Удалить(Максимальный.Ключ);

КонецЦикла;

В данном блоке реализован алгоритм, описанный в разделе решение. Пока есть не посещенные организации ищется организация с максимальным количеством связей, добавляется в список посещений и для каждой из связей убирается данная организация из списка организации-связи, а если связей у той организации больше не остается – она не требует посещения и удаляется из соответствия Взаимосвязи.

ТабДок.Очистить();

ТабДок.Вывести(ОбластьЗаголовок);

Номер = 1;

Для Каждого Организация Из Обход Цикл

ОбластьДанные.Параметры[0] = Номер;

ОбластьДанные.Параметры[1] = Организация.Код;

ОбластьДанные.Параметры[2] = Организация.Наименование;

ТабДок.Вывести(ОбластьДанные);

Номер = Номер + 1;

КонецЦикла;

В данном блоке идет очистка табличного документа от старых данных и вывод полученного результата.

Процедура ДобавитьВзаимосвязь(Взаимосвязи,Выборка)

Взаимосвязи1 = Взаимосвязи.Получить(Выборка.Организация1);

Взаимосвязи2 = Взаимосвязи.Получить(Выборка.Организация2);

Если Взаимосвязи1 = Неопределено Тогда

Взаимосвязи1 = Новый Массив;

Взаимосвязи.Вставить(Выборка.Организация1,Взаимосвязи1);

КонецЕсли;

Если Взаимосвязи2 = Неопределено Тогда

Взаимосвязи2 = Новый Массив;

Взаимосвязи.Вставить(Выборка.Организация2,Взаимосвязи2);

КонецЕсли;

Если Взаимосвязи1.Найти(Выборка.Организация2) = Неопределено Тогда

Взаимосвязи1.Добавить(Выборка.Организация2);

КонецЕсли;

Если Взаимосвязи2.Найти(Выборка.Организация1) = Неопределено Тогда

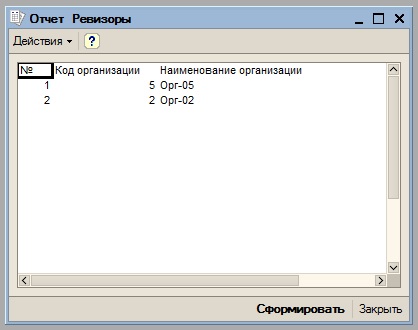
Взаимосвязи2.Добавить(Выборка.Организация1);

КонецЕсли;

КонецПроцедуры

Данная процедура добавляет пару организаций в соответствие Взаимосвязи. При условиях: если нет первой или второй организации в соответствии, то для её связей создается массив и добавляется в соответствие. Если в списке связей организации не найдена вторая – то идет добавление в список связей.

Сформированный отчет



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.

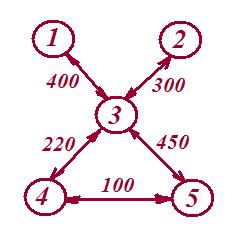
Изначально ревизор находится в головном офисе в городе Москве, где нет складов

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

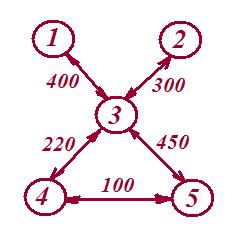
Отчет, представленный в следующей форме.

|  |
| --- |
| 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;

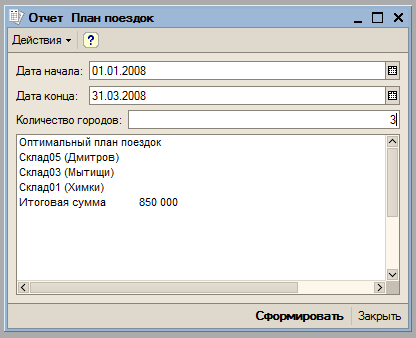
ДатаНачала = '20080101';

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



2008 год 1 тур

Требование

Исходная база

Предприятие ведет бухгалтерский учет в программе «1С:Бухгалтерия 8 (7.7)».

Учет материалов ведется на счете 10 в разрезе субконто «Материалы» в стоимостном и натуральном выражении. Поступление материалов отражается в учете проводками д-т 10, кр-т 60. На счете 60 аналитический учет (условно) не ведется.

В информационной базе предприятия имеются проводки за 1 квартал 2008 г. по приходу материалов. Стоимость списания для разных групп материалов может определяться методами: «по средней скользящей» или ЛИФО.

Входные данные

Предусмотреть форму диалога, в которой пользователь выбирает наименование материала из справочника «Материалы», вводит для него величину расхода в натуральном выражении и методику списания: «Средняя скользящая» или «ЛИФО» путем выбора значения перечисления «МетодСписанияМЦ».  Введенная величина расхода материала не должна превышать величину остатка этого материала в натуральном выражении на конец периода (1 квартал 2008 г.).

Требуется

В каркасной конфигурации разработать отчет, в котором для выбранного материала и заданного количества расхода предлагается такой состав и порядок следования операций расхода (относительно операций прихода), при котором обеспечивается максимизация стоимости списанных материалов (по выбранной методике списания) минимальным количеством расходных операций. При разработке отчета следует исходить из следующего:

* в один день не может быть более одной операции прихода материала одного вида;
* поступивший материал может расходоваться в этот же день;
* если в один день имеются операции прихода и расхода материалов, то считается, что операции расхода в этот день всегда следуют за операцией прихода;
* количество материалов может быть выражено только целыми числами, количественный остаток материала на любую дату не может быть отрицательным.

Выходные данные

Результат должен быть представлен в виде отчета, содержащего операции прихода выбранного материала и операции расхода, обеспечивающие выполнение условий постановки задачи. Отчет должен быть представлен в следующем виде.

Списание 15 единиц материала М-01 по методу ЛИФО

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ДАТА | Количество  приход | Сумма приход | Количество  расход | Сумма  расход |
| 05.01.2008 | 8 | 8000,00 | 8 | 8 000,00 |
| 16.01.2008 | 12 | 11400.00 |  |  |
| 11.02.2008 | 5 | 5500,00 | 7 | 7 400,00 |
| 27.03.2008 | 10 | 9000,00 |  |  |
| ИТОГО: | 35 |  | 15 | 15 400,00 |

Решение

Для решения нужно понимать методики списания, используемые в задаче. Методика "скользящая средняя" – при использовании данной методики цена товара будет равна средней арифметической цене, за период (сумма цен за период деленная на сумму количества). Методика "ЛИФО" – означает "последний пришел, первый ушел" – означает, что списываться будут первыми те товары, которые поступили последними. То есть, если сначала поступило 3 товара с ценой 30 и потом 2 товара с ценой 50, то при списании 3 товаров, будет взято 2 товара по цене 50 и 1 товар с ценой 30.

Реализация

Модуль объекта:

// Производит основной расчет данных для отчета

//

Процедура РассчитатьОтчет() Экспорт

Если МетодикаСписания = Перечисления.МетодСписанияМЦ.СредняяСкользящая Тогда // средняя

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПриходыМатериалов.Период КАК Дата,

| ПриходыМатериалов.КоличествоОборотДт КАК КоличествоПриход,

| ПриходыМатериалов.СуммаОборотДт КАК СуммаПриход

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, День, Счет.Код = ""10"", , Субконто1 = &Материал, , ) КАК ПриходыМатериалов

|ГДЕ

| (ПриходыМатериалов.СуммаОборотДт > 0

| ИЛИ ПриходыМатериалов.КоличествоОборотДт > 0)

|

|УПОРЯДОЧИТЬ ПО

| Дата";

Запрос.УстановитьПараметр("ДатаНач" , НачалоКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("Материал", Материал);

Результат = Запрос.Выполнить();

РезультатРасчетов.Загрузить(Результат.Выгрузить());

Если КоличествоРасхода = РезультатРасчетов.Итог("КоличествоПриход") Тогда // списываем все (обработка специфического случая)

ПоследняяСтрока = РезультатРасчетов.Получить(РезультатРасчетов.Количество() - 1);

ПоследняяСтрока.КоличествоРасход = КоличествоРасхода;

ПоследняяСтрока.СуммаРасход = РезультатРасчетов.Итог("СуммаПриход");

Возврат;

КонецЕсли;

ВременныйРезультат.Загрузить(Результат.Выгрузить());

ОсталосьСписать = КоличествоРасхода;

Пока ОсталосьСписать > 0 Цикл

ВременнаяТаблица = Новый Запрос;

МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

ВременнаяТаблица.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

ВременнаяТаблица.Текст = "ВЫБРАТЬ \*

|ПОМЕСТИТЬ врТаблицаВременныхДанных

|ИЗ &ТаблицаВременныхДанных КАК ТД";

ВременнаяТаблица.УстановитьПараметр("ТаблицаВременныхДанных", ВременныйРезультат);

ВременнаяТаблица.Выполнить();

Запрос = Новый Запрос;

Запрос.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

Запрос.Текст = "ВЫБРАТЬ

| ВложенныйЗапрос.Дата КАК Дата,

| ВложенныйЗапрос.КоличествоПриход КАК КоличествоПриход,

| ВложенныйЗапрос.СуммаПриход,

| ВложенныйЗапрос.ОстатокКоличество,

| ВложенныйЗапрос.ОстатокСумма,

| ВложенныйЗапрос.ОстатокСумма / ВложенныйЗапрос.ОстатокКоличество КАК СредняяЦена

|ИЗ

| (ВЫБРАТЬ

| Основной.Дата КАК Дата,

| СРЕДНЕЕ(Основной.КоличествоПриход) КАК КоличествоПриход,

| СРЕДНЕЕ(Основной.СуммаПриход) КАК СуммаПриход,

| СУММА(Остатки.КоличествоПриход) КАК ОстатокКоличество,

| СУММА(Остатки.СуммаПриход) КАК ОстатокСумма

| ИЗ

| (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Основной

| ЛЕВОЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Остатки

| ПО Основной.Дата >= Остатки.Дата

|

| СГРУППИРОВАТЬ ПО

| Основной.Дата) КАК ВложенныйЗапрос

|

|УПОРЯДОЧИТЬ ПО

| СредняяЦена УБЫВ,

| ОстатокКоличество УБЫВ,

| Дата";

Результат = Запрос.Выполнить();

ВременныйРезультат.Загрузить(Результат.Выгрузить());

//Предупреждение("ОсталосьСписать = " + ОсталосьСписать);

Если ВременныйРезультат.Количество() > 0 Тогда

ОстатокКоличество = ВременныйРезультат[0].ОстатокКоличество;

ОстатокСумма = ВременныйРезультат[0].ОстатокСумма;

СредняяЦенаЕдиницы = ВременныйРезультат[0].СредняяЦена;

ДатаПрихода = ВременныйРезультат[0].Дата;

Если ОстатокКоличество >= ОсталосьСписать Тогда // весь остаток спишем (часть текущей партии)

ОстатокКоличество = ОсталосьСписать;

ОстатокСумма = ОсталосьСписать \* СредняяЦенаЕдиницы;

Иначе // часть остатка спишем (всю текущую партию и все предыдущие)

ВременныйРезультат.Сортировать("Дата");

Пока ВременныйРезультат.Количество() > 0 Цикл

ДатаСтроки = ВременныйРезультат[0].Дата;

ВременныйРезультат.Удалить(0);

Если ДатаСтроки = ДатаПрихода Тогда

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

КонецЕсли;

СтрокаРезультата = РезультатРасчетов.Найти(ДатаПрихода, "Дата");

Если СтрокаРезультата <> Неопределено Тогда

СтрокаРезультата.КоличествоРасход = ОстатокКоличество;

СтрокаРезультата.СуммаРасход = ОстатокСумма;

Иначе

Сообщить(" Не могу найти строку по дате!", СтатусСообщения.Важное);

КонецЕсли;

ОсталосьСписать = ОсталосьСписать - ОстатокКоличество;

Иначе

Сообщить(" Во временном результате количество " + ВременныйРезультат.Количество() + ", что не больше 0!", СтатусСообщения.Важное);

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

ИначеЕсли МетодикаСписания = Перечисления.МетодСписанияМЦ.ЛИФО Тогда // ЛИФО

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПриходыМатериалов.Период КАК Дата,

| ПриходыМатериалов.КоличествоОборотДт КАК КоличествоПриход,

| ПриходыМатериалов.СуммаОборотДт КАК СуммаПриход,

| ПриходыМатериалов.СуммаОборотДт / ПриходыМатериалов.КоличествоОборотДт КАК ЦенаЕдиницы

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, День, Счет.Код = ""10"", , Субконто1 = &Материал, , ) КАК ПриходыМатериалов

|ГДЕ

| (ПриходыМатериалов.СуммаОборотДт > 0

| ИЛИ ПриходыМатериалов.КоличествоОборотДт > 0)

|

|УПОРЯДОЧИТЬ ПО

| ЦенаЕдиницы УБЫВ,

| Дата";

Запрос.УстановитьПараметр("ДатаНач" , НачалоКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("Материал", Материал);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

РезультатРасчетов.Очистить();

ОсталосьСписать = КоличествоРасхода;

Пока Выборка.Следующий() Цикл

НоваяСтрока = РезультатРасчетов.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Выборка);

Если ОсталосьСписать <> 0 Тогда

Если НоваяСтрока.КоличествоПриход >= ОсталосьСписать Тогда // весь остаток спишем

НоваяСтрока.КоличествоРасход = ОсталосьСписать;

НоваяСтрока.СуммаРасход = ОсталосьСписать \* Выборка.ЦенаЕдиницы;

ОсталосьСписать = 0;

Иначе // часть остатка спишем

НоваяСтрока.КоличествоРасход = Выборка.КоличествоПриход;

НоваяСтрока.СуммаРасход = Выборка.СуммаПриход;

ОсталосьСписать = ОсталосьСписать - Выборка.КоличествоПриход;

КонецЕсли;

КонецЕсли;

КонецЦикла;

РезультатРасчетов.Сортировать("Дата");

Итератор = РезультатРасчетов.Количество() - 1;

Пока Итератор >= 0 Цикл

Строка = РезультатРасчетов.Получить(Итератор);

//Сообщить("Дата = " + Строка.Дата);

Если Строка.КоличествоРасход <> 0 Или Строка.СуммаРасход <> 0 Тогда // обрабатываем только строки с заполненным расходом

СобранноеКоличество = Строка.КоличествоРасход;

СобраннаяСумма = Строка.СуммаРасход;

ОбратныйСчетчик = Итератор - 1;

Пока ОбратныйСчетчик >= 0 Цикл

Подстрока = РезультатРасчетов.Получить(ОбратныйСчетчик);

//Сообщить(" Подстрока дата = " + Подстрока.Дата);

СобранноеКоличество = СобранноеКоличество + Подстрока.КоличествоРасход;

СобраннаяСумма = СобраннаяСумма + Подстрока.СуммаРасход;

КоличествоПодстроки = Подстрока.КоличествоРасход;

Подстрока.КоличествоРасход = 0;

Подстрока.СуммаРасход = 0;

Если КоличествоПодстроки <> Подстрока.КоличествоПриход Тогда // дошли до той партии, которая списана только частично

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

ОбратныйСчетчик = ОбратныйСчетчик - 1;

КонецЦикла;

Строка.КоличествоРасход = СобранноеКоличество;

Строка.СуммаРасход = СобраннаяСумма;

КонецЕсли;

ОбработкаПрерыванияПользователя();

Итератор = Итератор - 1;

КонецЦикла;

КонецЕсли;

КонецПроцедуры // РассчитатьОтчет()

// Производит вывод отчета в переданный параметром табличный документ

//

Процедура ВывестиОтчет(ТабДок) Экспорт

Макет = ЭтотОбъект.ПолучитьМакет("МакетОтчета");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");

ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();

ЗаполнитьЗначенияСвойств(ОбластьЗаголовок.Параметры, ЭтотОбъект);

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаТаблицы);

ТабДок.ФиксацияСверху = 4;

Для Каждого Строка Из РезультатРасчетов Цикл

ОбластьДетальныхЗаписей.Параметры.Заполнить(Строка);

ТабДок.Вывести(ОбластьДетальныхЗаписей);

КонецЦикла;

ОбластьПодвалТаблицы.Параметры.ИтогоКоличествоПриход = РезультатРасчетов.Итог("КоличествоПриход");

ОбластьПодвалТаблицы.Параметры.ИтогоСуммаПриход = РезультатРасчетов.Итог("СуммаПриход");

ОбластьПодвалТаблицы.Параметры.ИтогоКоличествоРасход = РезультатРасчетов.Итог("КоличествоРасход");

ОбластьПодвалТаблицы.Параметры.ИтогоСуммаРасход = РезультатРасчетов.Итог("СуммаРасход");

ТабДок.Вывести(ОбластьПодвалТаблицы);

КонецПроцедуры // ВывестиОтчет()

Модуль формы:

// Проводит предварительные проверки входной информации введенной пользователем, запускает основной код расчета и формирования отчета

//

Процедура ДействияФормыСформировать(Кнопка)

// сначала всякие проверки на корректный ввод исходных данных

Если Не ЗначениеЗаполнено(Материал) Тогда

Предупреждение("Пожалуйста укажите материал по которому необходимо рассчитать и сформировать отчет!",,);

Возврат;

КонецЕсли;

Если Не ЗначениеЗаполнено(КоличествоРасхода) Тогда

Предупреждение("Пожалуйста укажите величину расхода в натуральном выражении (поле ""количество расхода"") по которой нужно рассчитать отчет!",,);

Возврат;

КонецЕсли;

Если Не ЗначениеЗаполнено(МетодикаСписания) Тогда

Предупреждение("Пожалуйста укажите методику списания по которой необходимо рассчитать и сформировать отчет!",,);

Возврат;

КонецЕсли;

// проверка остатка величины расхода материала введенной пользователем

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ОстаткиМатериала.КоличествоОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&ДатаКон, Счет.Код = ""10"", , Субконто1 = &Материал) КАК ОстаткиМатериала

|ГДЕ

| ОстаткиМатериала.КоличествоОстаток < &КоличествоРасхода";

Запрос.УстановитьПараметр("Материал" , Материал);

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("КоличествоРасхода", КоличествоРасхода);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Если Выборка.Следующий() Тогда

Сообщить("Величина расхода материала """ + Материал + """ в натуральном выражении (" + КоличествоРасхода + " единиц) не может быть больше остатка (" + Выборка.КоличествоОстаток + " единиц) по данным на конец I квартала 2008 г.!", СтатусСообщения.Важное);

Предупреждение("Расчет отчета не выполнен!",,);

Возврат;

КонецЕсли;

// выполним расчеты

РассчитатьОтчет();

// выведем расчеты

ВывестиОтчет(ЭлементыФормы.ТабДокумент);

КонецПроцедуры // ДействияФормыСформировать()

Автор: Андрей Александрович

Пояснения к коду:

Модуль объекта:

Процедура РассчитатьОтчет() Экспорт

Если МетодикаСписания = Перечисления.МетодСписанияМЦ.СредняяСкользящая Тогда // средняя

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПриходыМатериалов.Период КАК Дата,

| ПриходыМатериалов.КоличествоОборотДт КАК КоличествоПриход,

| ПриходыМатериалов.СуммаОборотДт КАК СуммаПриход

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, День, Счет.Код = ""10"", , Субконто1 = &Материал, , ) КАК ПриходыМатериалов

|ГДЕ

| (ПриходыМатериалов.СуммаОборотДт > 0

| ИЛИ ПриходыМатериалов.КоличествоОборотДт > 0)

|

|УПОРЯДОЧИТЬ ПО

| Дата";

Запрос.УстановитьПараметр("ДатаНач" , НачалоКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("Материал", Материал);

Результат = Запрос.Выполнить();

РезультатРасчетов.Загрузить(Результат.Выгрузить());

Если КоличествоРасхода = РезультатРасчетов.Итог("КоличествоПриход") Тогда // списываем все (обработка специфического случая)

ПоследняяСтрока = РезультатРасчетов.Получить(РезультатРасчетов.Количество() - 1);

ПоследняяСтрока.КоличествоРасход = КоличествоРасхода;

ПоследняяСтрока.СуммаРасход = РезультатРасчетов.Итог("СуммаПриход");

Возврат;

КонецЕсли;

ВременныйРезультат.Загрузить(Результат.Выгрузить());

ОсталосьСписать = КоличествоРасхода;

Пока ОсталосьСписать > 0 Цикл

ВременнаяТаблица = Новый Запрос;

МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

ВременнаяТаблица.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

ВременнаяТаблица.Текст = "ВЫБРАТЬ \*

|ПОМЕСТИТЬ врТаблицаВременныхДанных

|ИЗ &ТаблицаВременныхДанных КАК ТД";

ВременнаяТаблица.УстановитьПараметр("ТаблицаВременныхДанных", ВременныйРезультат);

ВременнаяТаблица.Выполнить();

Запрос = Новый Запрос;

Запрос.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

Запрос.Текст = "ВЫБРАТЬ

| ВложенныйЗапрос.Дата КАК Дата,

| ВложенныйЗапрос.КоличествоПриход КАК КоличествоПриход,

| ВложенныйЗапрос.СуммаПриход,

| ВложенныйЗапрос.ОстатокКоличество,

| ВложенныйЗапрос.ОстатокСумма,

| ВложенныйЗапрос.ОстатокСумма / ВложенныйЗапрос.ОстатокКоличество КАК СредняяЦена

|ИЗ

| (ВЫБРАТЬ

| Основной.Дата КАК Дата,

| СРЕДНЕЕ(Основной.КоличествоПриход) КАК КоличествоПриход,

| СРЕДНЕЕ(Основной.СуммаПриход) КАК СуммаПриход,

| СУММА(Остатки.КоличествоПриход) КАК ОстатокКоличество,

| СУММА(Остатки.СуммаПриход) КАК ОстатокСумма

| ИЗ

| (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Основной

| ЛЕВОЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Остатки

| ПО Основной.Дата >= Остатки.Дата

|

| СГРУППИРОВАТЬ ПО

| Основной.Дата) КАК ВложенныйЗапрос

|

|УПОРЯДОЧИТЬ ПО

| СредняяЦена УБЫВ,

| ОстатокКоличество УБЫВ,

| Дата";

Результат = Запрос.Выполнить();

ВременныйРезультат.Загрузить(Результат.Выгрузить());

//Предупреждение("ОсталосьСписать = " + ОсталосьСписать);

Если ВременныйРезультат.Количество() > 0 Тогда

ОстатокКоличество = ВременныйРезультат[0].ОстатокКоличество;

ОстатокСумма = ВременныйРезультат[0].ОстатокСумма;

СредняяЦенаЕдиницы = ВременныйРезультат[0].СредняяЦена;

ДатаПрихода = ВременныйРезультат[0].Дата;

Если ОстатокКоличество >= ОсталосьСписать Тогда // весь остаток спишем (часть текущей партии)

ОстатокКоличество = ОсталосьСписать;

ОстатокСумма = ОсталосьСписать \* СредняяЦенаЕдиницы;

Иначе // часть остатка спишем (всю текущую партию и все предыдущие)

ВременныйРезультат.Сортировать("Дата");

Пока ВременныйРезультат.Количество() > 0 Цикл

ДатаСтроки = ВременныйРезультат[0].Дата;

ВременныйРезультат.Удалить(0);

Если ДатаСтроки = ДатаПрихода Тогда

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

КонецЕсли;

СтрокаРезультата = РезультатРасчетов.Найти(ДатаПрихода, "Дата");

Если СтрокаРезультата <> Неопределено Тогда

СтрокаРезультата.КоличествоРасход = ОстатокКоличество;

СтрокаРезультата.СуммаРасход = ОстатокСумма;

Иначе

Сообщить(" Не могу найти строку по дате!", СтатусСообщения.Важное);

КонецЕсли;

ОсталосьСписать = ОсталосьСписать - ОстатокКоличество;

Иначе

Сообщить(" Во временном результате количество " + ВременныйРезультат.Количество() + ", что не больше 0!", СтатусСообщения.Важное);

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

ИначеЕсли МетодикаСписания = Перечисления.МетодСписанияМЦ.ЛИФО Тогда // ЛИФО

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПриходыМатериалов.Период КАК Дата,

| ПриходыМатериалов.КоличествоОборотДт КАК КоличествоПриход,

| ПриходыМатериалов.СуммаОборотДт КАК СуммаПриход,

| ПриходыМатериалов.СуммаОборотДт / ПриходыМатериалов.КоличествоОборотДт КАК ЦенаЕдиницы

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, День, Счет.Код = ""10"", , Субконто1 = &Материал, , ) КАК ПриходыМатериалов

|ГДЕ

| (ПриходыМатериалов.СуммаОборотДт > 0

| ИЛИ ПриходыМатериалов.КоличествоОборотДт > 0)

|

|УПОРЯДОЧИТЬ ПО

| ЦенаЕдиницы УБЫВ,

| Дата";

Запрос.УстановитьПараметр("ДатаНач" , НачалоКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("Материал", Материал);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

РезультатРасчетов.Очистить();

ОсталосьСписать = КоличествоРасхода;

Пока Выборка.Следующий() Цикл

НоваяСтрока = РезультатРасчетов.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Выборка);

Если ОсталосьСписать <> 0 Тогда

Если НоваяСтрока.КоличествоПриход >= ОсталосьСписать Тогда // весь остаток спишем

НоваяСтрока.КоличествоРасход = ОсталосьСписать;

НоваяСтрока.СуммаРасход = ОсталосьСписать \* Выборка.ЦенаЕдиницы;

ОсталосьСписать = 0;

Иначе // часть остатка спишем

НоваяСтрока.КоличествоРасход = Выборка.КоличествоПриход;

НоваяСтрока.СуммаРасход = Выборка.СуммаПриход;

ОсталосьСписать = ОсталосьСписать - Выборка.КоличествоПриход;

КонецЕсли;

КонецЕсли;

КонецЦикла;

РезультатРасчетов.Сортировать("Дата");

Итератор = РезультатРасчетов.Количество() - 1;

Пока Итератор >= 0 Цикл

Строка = РезультатРасчетов.Получить(Итератор);

//Сообщить("Дата = " + Строка.Дата);

Если Строка.КоличествоРасход <> 0 Или Строка.СуммаРасход <> 0 Тогда // обрабатываем только строки с заполненным расходом

СобранноеКоличество = Строка.КоличествоРасход;

СобраннаяСумма = Строка.СуммаРасход;

ОбратныйСчетчик = Итератор - 1;

Пока ОбратныйСчетчик >= 0 Цикл

Подстрока = РезультатРасчетов.Получить(ОбратныйСчетчик);

//Сообщить(" Подстрока дата = " + Подстрока.Дата);

СобранноеКоличество = СобранноеКоличество + Подстрока.КоличествоРасход;

СобраннаяСумма = СобраннаяСумма + Подстрока.СуммаРасход;

КоличествоПодстроки = Подстрока.КоличествоРасход;

Подстрока.КоличествоРасход = 0;

Подстрока.СуммаРасход = 0;

Если КоличествоПодстроки <> Подстрока.КоличествоПриход Тогда // дошли до той партии, которая списана только частично

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

ОбратныйСчетчик = ОбратныйСчетчик - 1;

КонецЦикла;

Строка.КоличествоРасход = СобранноеКоличество;

Строка.СуммаРасход = СобраннаяСумма;

КонецЕсли;

ОбработкаПрерыванияПользователя();

Итератор = Итератор - 1;

КонецЦикла;

КонецЕсли;

КонецПроцедуры // РассчитатьОтчет()

Данная процедура производит получение данных из базы и вычисление результата в зависимости от выбранной методики списания.

Методика списания «средняя скользящая»:

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПриходыМатериалов.Период КАК Дата,

| ПриходыМатериалов.КоличествоОборотДт КАК КоличествоПриход,

| ПриходыМатериалов.СуммаОборотДт КАК СуммаПриход

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, День, Счет.Код = ""10"", , Субконто1 = &Материал, , ) КАК ПриходыМатериалов

|ГДЕ

| (ПриходыМатериалов.СуммаОборотДт > 0

| ИЛИ ПриходыМатериалов.КоличествоОборотДт > 0)

|

|УПОРЯДОЧИТЬ ПО

| Дата";

Запрос.УстановитьПараметр("ДатаНач" , НачалоКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("Материал", Материал);

Результат = Запрос.Выполнить();

РезультатРасчетов.Загрузить(Результат.Выгрузить());

В данном блоке происходит получение данных о оборотах материала и выгрузка результата в переменную РезультатРасчетов.

Если КоличествоРасхода = РезультатРасчетов.Итог("КоличествоПриход") Тогда // списываем все (обработка специфического случая)

ПоследняяСтрока = РезультатРасчетов.Получить(РезультатРасчетов.Количество() - 1);

ПоследняяСтрока.КоличествоРасход = КоличествоРасхода;

ПоследняяСтрока.СуммаРасход = РезультатРасчетов.Итог("СуммаПриход");

Возврат;

КонецЕсли;

В данном блоке проверяется частный случай – количество расходуемого товара равно общему количеству прихода, в таком случае расходуется всё. И расчет завершается.

ВременныйРезультат.Загрузить(Результат.Выгрузить());

В данной строке в переменную ВременныйРезультат выгружаются данные из базы (так же как и в РезультатРасчетов).

ОсталосьСписать = КоличествоРасхода;

Пока ОсталосьСписать > 0 Цикл

ВременнаяТаблица = Новый Запрос;

МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

ВременнаяТаблица.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

ВременнаяТаблица.Текст = "ВЫБРАТЬ \*

|ПОМЕСТИТЬ врТаблицаВременныхДанных

|ИЗ &ТаблицаВременныхДанных КАК ТД";

ВременнаяТаблица.УстановитьПараметр("ТаблицаВременныхДанных", ВременныйРезультат);

ВременнаяТаблица.Выполнить();

Запрос = Новый Запрос;

Запрос.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

Запрос.Текст = "ВЫБРАТЬ

| ВложенныйЗапрос.Дата КАК Дата,

| ВложенныйЗапрос.КоличествоПриход КАК КоличествоПриход,

| ВложенныйЗапрос.СуммаПриход,

| ВложенныйЗапрос.ОстатокКоличество,

| ВложенныйЗапрос.ОстатокСумма,

| ВложенныйЗапрос.ОстатокСумма / ВложенныйЗапрос.ОстатокКоличество КАК СредняяЦена

|ИЗ

| (ВЫБРАТЬ

| Основной.Дата КАК Дата,

| СРЕДНЕЕ(Основной.КоличествоПриход) КАК КоличествоПриход,

| СРЕДНЕЕ(Основной.СуммаПриход) КАК СуммаПриход,

| СУММА(Остатки.КоличествоПриход) КАК ОстатокКоличество,

| СУММА(Остатки.СуммаПриход) КАК ОстатокСумма

| ИЗ

| (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Основной

| ЛЕВОЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Остатки

| ПО Основной.Дата >= Остатки.Дата

|

| СГРУППИРОВАТЬ ПО

| Основной.Дата) КАК ВложенныйЗапрос

|

|УПОРЯДОЧИТЬ ПО

| СредняяЦена УБЫВ,

| ОстатокКоличество УБЫВ,

| Дата";

Результат = Запрос.Выполнить();

ВременныйРезультат.Загрузить(Результат.Выгрузить());

//Предупреждение("ОсталосьСписать = " + ОсталосьСписать);

Если ВременныйРезультат.Количество() > 0 Тогда

ОстатокКоличество = ВременныйРезультат[0].ОстатокКоличество;

ОстатокСумма = ВременныйРезультат[0].ОстатокСумма;

СредняяЦенаЕдиницы = ВременныйРезультат[0].СредняяЦена;

ДатаПрихода = ВременныйРезультат[0].Дата;

Если ОстатокКоличество >= ОсталосьСписать Тогда // весь остаток спишем (часть текущей партии)

ОстатокКоличество = ОсталосьСписать;

ОстатокСумма = ОсталосьСписать \* СредняяЦенаЕдиницы;

Иначе // часть остатка спишем (всю текущую партию и все предыдущие)

ВременныйРезультат.Сортировать("Дата");

Пока ВременныйРезультат.Количество() > 0 Цикл

ДатаСтроки = ВременныйРезультат[0].Дата;

ВременныйРезультат.Удалить(0);

Если ДатаСтроки = ДатаПрихода Тогда

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

КонецЕсли;

СтрокаРезультата = РезультатРасчетов.Найти(ДатаПрихода, "Дата");

Если СтрокаРезультата <> Неопределено Тогда

СтрокаРезультата.КоличествоРасход = ОстатокКоличество;

СтрокаРезультата.СуммаРасход = ОстатокСумма;

Иначе

Сообщить(" Не могу найти строку по дате!", СтатусСообщения.Важное);

КонецЕсли;

ОсталосьСписать = ОсталосьСписать - ОстатокКоличество;

Иначе

Сообщить(" Во временном результате количество " + ВременныйРезультат.Количество() + ", что не больше 0!", СтатусСообщения.Важное);

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

В данном блоке идет получение количества, которое осталось списать и пока остается, что списывать будет проводиться списание.

ВременнаяТаблица = Новый Запрос;

МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

ВременнаяТаблица.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

ВременнаяТаблица.Текст = "ВЫБРАТЬ \*

|ПОМЕСТИТЬ врТаблицаВременныхДанных

|ИЗ &ТаблицаВременныхДанных КАК ТД";

ВременнаяТаблица.УстановитьПараметр("ТаблицаВременныхДанных", ВременныйРезультат);

ВременнаяТаблица.Выполнить();

Запрос = Новый Запрос;

Запрос.МенеджерВременныхТаблиц = МенеджерВременныхТаблиц;

Запрос.Текст = "ВЫБРАТЬ

| ВложенныйЗапрос.Дата КАК Дата,

| ВложенныйЗапрос.КоличествоПриход КАК КоличествоПриход,

| ВложенныйЗапрос.СуммаПриход,

| ВложенныйЗапрос.ОстатокКоличество,

| ВложенныйЗапрос.ОстатокСумма,

| ВложенныйЗапрос.ОстатокСумма / ВложенныйЗапрос.ОстатокКоличество КАК СредняяЦена

|ИЗ

| (ВЫБРАТЬ

| Основной.Дата КАК Дата,

| СРЕДНЕЕ(Основной.КоличествоПриход) КАК КоличествоПриход,

| СРЕДНЕЕ(Основной.СуммаПриход) КАК СуммаПриход,

| СУММА(Остатки.КоличествоПриход) КАК ОстатокКоличество,

| СУММА(Остатки.СуммаПриход) КАК ОстатокСумма

| ИЗ

| (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Основной

| ЛЕВОЕ СОЕДИНЕНИЕ (ВЫБРАТЬ

| врТаблицаВременныхДанных.Дата КАК Дата,

| врТаблицаВременныхДанных.КоличествоПриход КАК КоличествоПриход,

| врТаблицаВременныхДанных.СуммаПриход КАК СуммаПриход

| ИЗ

| врТаблицаВременныхДанных КАК врТаблицаВременныхДанных) КАК Остатки

| ПО Основной.Дата >= Остатки.Дата

|

| СГРУППИРОВАТЬ ПО

| Основной.Дата) КАК ВложенныйЗапрос

|

|УПОРЯДОЧИТЬ ПО

| СредняяЦена УБЫВ,

| ОстатокКоличество УБЫВ,

| Дата";

Результат = Запрос.Выполнить();

ВременныйРезультат.Загрузить(Результат.Выгрузить());

В данном блоке происходит получение данных о приходе, средней цены, количества, суммы, остатков основываясь на предыдущем временном результате. Новый результат фиксируется в переменной ВременныйРезультат.

Если ВременныйРезультат.Количество() > 0 Тогда

ОстатокКоличество = ВременныйРезультат[0].ОстатокКоличество;

ОстатокСумма = ВременныйРезультат[0].ОстатокСумма;

СредняяЦенаЕдиницы = ВременныйРезультат[0].СредняяЦена;

ДатаПрихода = ВременныйРезультат[0].Дата;

Если ОстатокКоличество >= ОсталосьСписать Тогда // весь остаток спишем (часть текущей партии)

ОстатокКоличество = ОсталосьСписать;

ОстатокСумма = ОсталосьСписать \* СредняяЦенаЕдиницы;

Иначе // часть остатка спишем (всю текущую партию и все предыдущие)

ВременныйРезультат.Сортировать("Дата");

Пока ВременныйРезультат.Количество() > 0 Цикл

ДатаСтроки = ВременныйРезультат[0].Дата;

ВременныйРезультат.Удалить(0);

Если ДатаСтроки = ДатаПрихода Тогда

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

КонецЦикла;

КонецЕсли;

СтрокаРезультата = РезультатРасчетов.Найти(ДатаПрихода, "Дата");

Если СтрокаРезультата <> Неопределено Тогда

СтрокаРезультата.КоличествоРасход = ОстатокКоличество;

СтрокаРезультата.СуммаРасход = ОстатокСумма;

Иначе

Сообщить(" Не могу найти строку по дате!", СтатусСообщения.Важное);

КонецЕсли;

ОсталосьСписать = ОсталосьСписать - ОстатокКоличество;

Иначе

Сообщить(" Во временном результате количество " + ВременныйРезультат.Количество() + ", что не больше 0!", СтатусСообщения.Важное);

КонецЕсли;

В данном блоке происходит списание остатка (если остатки есть). Если оставшееся количество больше необходимого – списываем весь остаток. Иначе спишем часть остатка – текущую партию и все предыдущие до даты прихода. Далее на дату прихода указывается количество расхода и сумма.

ОбработкаПрерыванияПользователя();

Данная стока позволяет прерывать выполнение цикла сочетанием Ctrl+Break.

Методика ЛИФО:

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПриходыМатериалов.Период КАК Дата,

| ПриходыМатериалов.КоличествоОборотДт КАК КоличествоПриход,

| ПриходыМатериалов.СуммаОборотДт КАК СуммаПриход,

| ПриходыМатериалов.СуммаОборотДт / ПриходыМатериалов.КоличествоОборотДт КАК ЦенаЕдиницы

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(&ДатаНач, &ДатаКон, День, Счет.Код = ""10"", , Субконто1 = &Материал, , ) КАК ПриходыМатериалов

|ГДЕ

| (ПриходыМатериалов.СуммаОборотДт > 0

| ИЛИ ПриходыМатериалов.КоличествоОборотДт > 0)

|

|УПОРЯДОЧИТЬ ПО

| ЦенаЕдиницы УБЫВ,

| Дата";

Запрос.УстановитьПараметр("ДатаНач" , НачалоКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("Материал", Материал);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

РезультатРасчетов.Очистить();

ОсталосьСписать = КоличествоРасхода;

Пока Выборка.Следующий() Цикл

НоваяСтрока = РезультатРасчетов.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Выборка);

Если ОсталосьСписать <> 0 Тогда

Если НоваяСтрока.КоличествоПриход >= ОсталосьСписать Тогда // весь остаток спишем

НоваяСтрока.КоличествоРасход = ОсталосьСписать;

НоваяСтрока.СуммаРасход = ОсталосьСписать \* Выборка.ЦенаЕдиницы;

ОсталосьСписать = 0;

Иначе // часть остатка спишем

НоваяСтрока.КоличествоРасход = Выборка.КоличествоПриход;

НоваяСтрока.СуммаРасход = Выборка.СуммаПриход;

ОсталосьСписать = ОсталосьСписать - Выборка.КоличествоПриход;

КонецЕсли;

КонецЕсли;

КонецЦикла;

РезультатРасчетов.Сортировать("Дата");

Итератор = РезультатРасчетов.Количество() - 1;

Пока Итератор >= 0 Цикл

Строка = РезультатРасчетов.Получить(Итератор);

//Сообщить("Дата = " + Строка.Дата);

Если Строка.КоличествоРасход <> 0 Или Строка.СуммаРасход <> 0 Тогда // обрабатываем только строки с заполненным расходом

СобранноеКоличество = Строка.КоличествоРасход;

СобраннаяСумма = Строка.СуммаРасход;

ОбратныйСчетчик = Итератор - 1;

Пока ОбратныйСчетчик >= 0 Цикл

Подстрока = РезультатРасчетов.Получить(ОбратныйСчетчик);

//Сообщить(" Подстрока дата = " + Подстрока.Дата);

СобранноеКоличество = СобранноеКоличество + Подстрока.КоличествоРасход;

СобраннаяСумма = СобраннаяСумма + Подстрока.СуммаРасход;

КоличествоПодстроки = Подстрока.КоличествоРасход;

Подстрока.КоличествоРасход = 0;

Подстрока.СуммаРасход = 0;

Если КоличествоПодстроки <> Подстрока.КоличествоПриход Тогда // дошли до той партии, которая списана только частично

Прервать;

КонецЕсли;

ОбработкаПрерыванияПользователя();

ОбратныйСчетчик = ОбратныйСчетчик - 1;

КонецЦикла;

Строка.КоличествоРасход = СобранноеКоличество;

Строка.СуммаРасход = СобраннаяСумма;

КонецЕсли;

ОбработкаПрерыванияПользователя();

Итератор = Итератор - 1;

КонецЦикла;

Сначала происходит получение данных из базы, с заполнением таблицы значений РезультатРасчетов. После заполнения таблицы, она сортируется по дате, для списания по методике (Последний пришел – первый ушел). Далее записи, с последней по первую обходятся циклом, списывая строки с заполненным расходом.

Процедура ВывестиОтчет(ТабДок) Экспорт

Макет = ЭтотОбъект.ПолучитьМакет("МакетОтчета");

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");

ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");

ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();

ЗаполнитьЗначенияСвойств(ОбластьЗаголовок.Параметры, ЭтотОбъект);

ТабДок.Вывести(ОбластьЗаголовок);

ТабДок.Вывести(ОбластьШапкаТаблицы);

ТабДок.ФиксацияСверху = 4;

Для Каждого Строка Из РезультатРасчетов Цикл

ОбластьДетальныхЗаписей.Параметры.Заполнить(Строка);

ТабДок.Вывести(ОбластьДетальныхЗаписей);

КонецЦикла;

ОбластьПодвалТаблицы.Параметры.ИтогоКоличествоПриход = РезультатРасчетов.Итог("КоличествоПриход");

ОбластьПодвалТаблицы.Параметры.ИтогоСуммаПриход = РезультатРасчетов.Итог("СуммаПриход");

ОбластьПодвалТаблицы.Параметры.ИтогоКоличествоРасход = РезультатРасчетов.Итог("КоличествоРасход");

ОбластьПодвалТаблицы.Параметры.ИтогоСуммаРасход = РезультатРасчетов.Итог("СуммаРасход");

ТабДок.Вывести(ОбластьПодвалТаблицы);

КонецПроцедуры // ВывестиОтчет()

Данная процедура выводит результат расчетов в табличный документ. Для этого сначала идет получение макета, далее получаются области макета, табличный документ очищается от старых данных, заполняется заголовок и выводится, а так же шапка. После этого выводятся все строки результата расчетов. И в конце выводятся итоги.

Модуль формы:

Процедура ДействияФормыСформировать(Кнопка)

// сначала всякие проверки на корректный ввод исходных данных

Если Не ЗначениеЗаполнено(Материал) Тогда

Предупреждение("Пожалуйста укажите материал по которому необходимо рассчитать и сформировать отчет!",,);

Возврат;

КонецЕсли;

Если Не ЗначениеЗаполнено(КоличествоРасхода) Тогда

Предупреждение("Пожалуйста укажите величину расхода в натуральном выражении (поле ""количество расхода"") по которой нужно рассчитать отчет!",,);

Возврат;

КонецЕсли;

Если Не ЗначениеЗаполнено(МетодикаСписания) Тогда

Предупреждение("Пожалуйста укажите методику списания по которой необходимо рассчитать и сформировать отчет!",,);

Возврат;

КонецЕсли;

// проверка остатка величины расхода материала введенной пользователем

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ОстаткиМатериала.КоличествоОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&ДатаКон, Счет.Код = ""10"", , Субконто1 = &Материал) КАК ОстаткиМатериала

|ГДЕ

| ОстаткиМатериала.КоличествоОстаток < &КоличествоРасхода";

Запрос.УстановитьПараметр("Материал" , Материал);

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("КоличествоРасхода", КоличествоРасхода);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Если Выборка.Следующий() Тогда

Сообщить("Величина расхода материала """ + Материал + """ в натуральном выражении (" + КоличествоРасхода + " единиц) не может быть больше остатка (" + Выборка.КоличествоОстаток + " единиц) по данным на конец I квартала 2008 г.!", СтатусСообщения.Важное);

Предупреждение("Расчет отчета не выполнен!",,);

Возврат;

КонецЕсли;

// выполним расчеты

РассчитатьОтчет();

// выведем расчеты

ВывестиОтчет(ЭлементыФормы.ТабДокумент);

КонецПроцедуры // ДействияФормыСформировать()

Данная процедура обрабатывает нажатие кнопки Сформировать. Она проверяет корректность данных, начинает расчет и выводит результат.

Если Не ЗначениеЗаполнено(Материал) Тогда

Предупреждение("Пожалуйста укажите материал по которому необходимо рассчитать и сформировать отчет!",,);

Возврат;

КонецЕсли;

Если Не ЗначениеЗаполнено(КоличествоРасхода) Тогда

Предупреждение("Пожалуйста укажите величину расхода в натуральном выражении (поле ""количество расхода"") по которой нужно рассчитать отчет!",,);

Возврат;

КонецЕсли;

Если Не ЗначениеЗаполнено(МетодикаСписания) Тогда

Предупреждение("Пожалуйста укажите методику списания по которой необходимо рассчитать и сформировать отчет!",,);

Возврат;

КонецЕсли;

В данном блоке проверяется заполнение необходимых полей.

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ОстаткиМатериала.КоличествоОстаток

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Остатки(&ДатаКон, Счет.Код = ""10"", , Субконто1 = &Материал) КАК ОстаткиМатериала

|ГДЕ

| ОстаткиМатериала.КоличествоОстаток < &КоличествоРасхода";

Запрос.УстановитьПараметр("Материал" , Материал);

Запрос.УстановитьПараметр("ДатаКон" , КонецКвартала(Дата("20080101")));

Запрос.УстановитьПараметр("КоличествоРасхода", КоличествоРасхода);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Если Выборка.Следующий() Тогда

Сообщить("Величина расхода материала """ + Материал + """ в натуральном выражении (" + КоличествоРасхода + " единиц) не может быть больше остатка (" + Выборка.КоличествоОстаток + " единиц) по данным на конец I квартала 2008 г.!", СтатусСообщения.Важное);

Предупреждение("Расчет отчета не выполнен!",,);

Возврат;

КонецЕсли;

В данном блоке идет получение данных из базы и проверяется количество материала.

// выполним расчеты

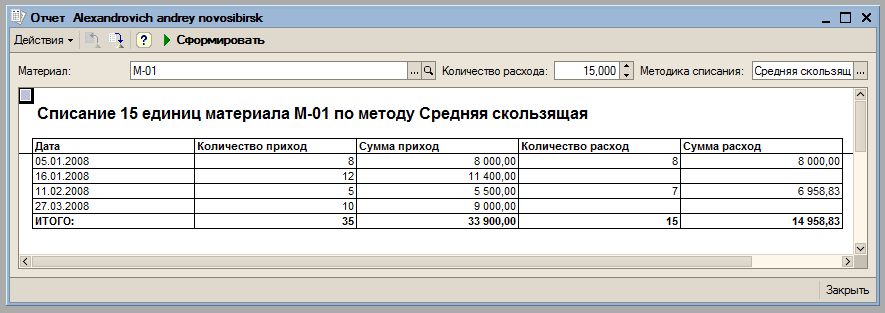
РассчитатьОтчет();

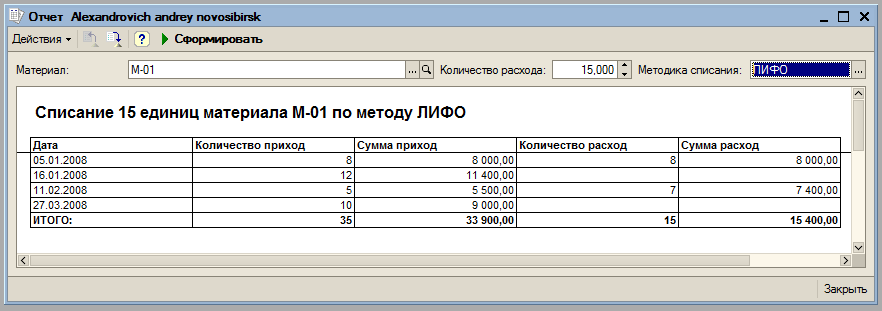
// выведем расчеты

ВывестиОтчет(ЭлементыФормы.ТабДокумент);

В данном блоке вызывается расчет отчета и вывод в табличный документ формы.

Сформированный отчет





2007 год 2 тур

Требование

Исходная ситуация

Предприятие ведет бухгалтерский учет в программе «1С:Бухгалтерия 8.0 (7.7)».

Учет материалов ведется на счете 10 в разрезе субконто «Материалы» в стоимостном и натуральном выражении. Поступление материалов от-ражается в учете проводками д-т 10, кр-т 60. Расход материалов отражается проводками д т 20, кр-т 10. На счетах 20 и 60 аналитический учет (условно) не ведется.

В информационной базе предприятия имеются проводки за 1 квартал 2007 г. по приходу и расходу материалов. Стоимость списания материалов в них определяется по «средней скользящей».

Требуется

В каркасной конфигурации разработать программу, которая анализирует все проводки информационной базы и формирует отчет, где для выбранного материала отображается такой порядок следования операций расхода (относительно операций прихода), при котором общая сумма списанных материалов (по «средней скользящей цене») будет с точностью до копейки равна заданной величине.

При этом следует исходить из следующего:

* в один день не может быть более одной операции прихода материала одного вида, поступивший материал может расходоваться в этот же день, количество операций расхода в один день не ограничено;
* даты поступления материала изменять не допускается;
* количество списываемого материала в операциях расхода изменять не допускается;
* количественный остаток материала на каждую дату не может быть отрицательным;
* при вычислениях округление выполняется по правилу: 0,5 =1.

Искомый материал и предполагаемая сумма списания вводятся в диалоге.

Если требуемого порядка следования операций расхода не существует, то необходимо выдать соответствующее сообщение.

Требования по представлению результата

Разработанный модуль отчета следует сохранить его как внешний файл. Имя файла должно включать фамилию участника, его имя (без разделителя, русскими буквами) и расширение, например, ИвановОлег.ert. Форма отчета должна соответствовать представленной в примере.

Пример

Исходные данные

Пусть ИБ содержит следующие проводки по приходу и расходу материала М-01:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Дата | Счет Дт | Субк. Дт | Счет Кт | Субк. Кт | Кол-во | Сумма | Содержание |
| 05.03.2007 | 10 | М-01 | 60 |  | 8 | 8000,00 | Приход МЦ |
| 16.03.2007 | 10 | М-01 | 60 |  | 12 | 10800,00 | Приход МЦ |
| 21.03.2007 | 10 | М-01 | 60 |  | 5 | 5500,00 | Приход МЦ |
| 27.03.2007 | 10 | М-01 | 60 |  | 10 | 9500,00 | Приход МЦ |
| 31.03.2007 | 20 |  | 10 | М-01 | 7 | 6760,00 | Расход МЦ |
| 31.03.2007 | 20 |  | 10 | М-01 | 5 | 4828,57 | Расход МЦ |
| 31.03.2007 | 20 |  | 10 | М-01 | 3 | 2897,14 | Расход МЦ |
| 31.03.2007 | 20 |  | 10 | М-01 | 8 | 7725,72 | Расход МЦ |

Необходимо

найти вариант списания материала М-01 на общую сумму 22 300,00 руб.

Результат

При заданных исходных данных искомый отчет будет иметь вид:

Вариант расхода Материала: М-01 на сумму: 22300,00

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ДАТА | Кол-во приход | Цена приход | Сумма приход | Кол-во расход | Цена расход | Сумма расход |
| 05.03.2007 | 8 | 1000,00 | 8 000,00 | 3 | 1000,00 | 3 000,00 |
|  |  |  |  | 5 | 1000,00 | 5 000,00 |
| 16.03.2007 | 12 | 900,00 | 10 800,00 | 7 | 900,00 | 6 300,00 |
| 21.03.2007 | 5 | 1100,00 | 5 500,00 | 8 | 1000,00 | 8 000,00 |
| 27.03.2007 | 10 | 950,00 | 9 500,00 |  |  |  |
| ИТОГО: | 35 |  | 33 800,00 | 23 |  | 22 300,00 |

Решение

В задаче требуется определить порядок списания материала, не изменяя количество списания. Возможно множество различных комбинаций списания, и для того, чтобы найти нужный вариант – нужно обойти дерево комбинаций, для этого можно использовать рекурсивную функцию, перебирающую варианты. Более подробно расписано в разделе Реализация.

Реализация

Перем ТипБулево;

Перем ТипЧисло;

Перем ФлагНашли;

Перем ТаблицаСписания;

Функция ПолучитьТаблицуДопустимыхОпераций(ТаблицаРасход, КоличествоОстаток)

ОбщийСписокВариантов = ТаблицаРасход.Скопировать();

ОбщийСписокВариантов.Очистить();

Для Каждого СтрокаТЧ из ТаблицаРасход Цикл

Если СтрокаТЧ.Флаг Тогда

Продолжить;

КонецЕсли;

НоваяСтрока = ОбщийСписокВариантов.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаТЧ);

КонецЦикла;

СписокДопустимыхОпераций = Новый СписокЗначений;

Для Каждого Строка1 из ОбщийСписокВариантов Цикл

БуфКоличество = КоличествоОстаток;

Если Строка1.Количество > БуфКоличество Тогда

Продолжить;

КонецЕсли;

НовыйСписокОпераций = Новый ТаблицаЗначений;

НовыйСписокОпераций.Колонки.Добавить("Регистратор");

НовыйСписокОпераций.Колонки.Добавить("Количество", ТипЧисло);

СписокДопустимыхОпераций.Добавить(НовыйСписокОпераций);

НоваяСтрока = НовыйСписокОпераций.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка1);

БуфКоличество = БуфКоличество - Строка1.Количество;

Если БуфКоличество = 0 Тогда

Прервать;

КонецЕсли;

Для Каждого Строка2 Из ОбщийСписокВариантов Цикл

Если Строка1.Регистратор = Строка2.Регистратор Тогда

Продолжить;

КонецЕсли;

Если Строка2.Количество > БуфКоличество Тогда

Продолжить;

КонецЕсли;

НоваяСтрока = НовыйСписокОпераций.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка2);

БуфКоличество = БуфКоличество - Строка2.Количество;

Если БуфКоличество = 0 Тогда

Прервать;

КонецЕсли;

КонецЦикла;

КонецЦикла;

НовыйСписокОпераций = Новый ТаблицаЗначений;

НовыйСписокОпераций.Колонки.Добавить("Регистратор");

НовыйСписокОпераций.Колонки.Добавить("Количество", ТипЧисло);

СписокДопустимыхОпераций.Добавить(НовыйСписокОпераций);

Возврат СписокДопустимыхОпераций;

КонецФункции

Процедура РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход, НомерСтрокиПриход = 0,

КоличествоСтрокПриход, Знач КоличествоОстаток = 0, Знач СуммаОстаток = 0, Знач НайденнаяСумма = 0)

Если НомерСтрокиПриход = КоличествоСтрокПриход Тогда

Возврат;

КонецЕсли;

ОбрабатываемаяСтрока = ТаблицаПриход[НомерСтрокиПриход];

КоличествоОстаток = ОбрабатываемаяСтрока.Количество + КоличествоОстаток;

СуммаОстаток = ОбрабатываемаяСтрока.Сумма + СуммаОстаток;

МассивНайденныхСтрок = Новый Массив;

СписокДопустимыхОпераций = ПолучитьТаблицуДопустимыхОпераций(ТаблицаРасход, КоличествоОстаток);

Для Каждого ЭлементСписка из СписокДопустимыхОпераций Цикл

ТаблицаДопустимыхОпераций = ЭлементСписка.Значение;

ТаблицаДопустимыхОпераций.Колонки.Добавить("Цена", ТипЧисло);

ТаблицаДопустимыхОпераций.Колонки.Добавить("Сумма", ТипЧисло);

МассивНайденныхСтрок.Очистить();

ВсегоСписать = 0;

СуммаТекущегоУровня = 0;

СуммаОстатокБуф = СуммаОстаток;

КоличествоОстатокБуф = КоличествоОстаток;

Для Каждого СтрокаТаблицы из ТаблицаДопустимыхОпераций Цикл

ВсегоСписать = ВсегоСписать + СтрокаТаблицы.Количество;

НайденнаяСтрока = ТаблицаРасход.Найти(СтрокаТаблицы.Регистратор, "Регистратор");

НайденнаяСтрока.Флаг = Истина;

МассивНайденныхСтрок.Добавить(НайденнаяСтрока);

СтрокаТаблицы.Цена = СуммаОстатокБуф / КоличествоОстатокБуф;

СтрокаТаблицы.Сумма = СтрокаТаблицы.Количество \* СтрокаТаблицы.Цена;

СуммаТекущегоУровня = СуммаТекущегоУровня + СтрокаТаблицы.Сумма;

СуммаОстатокБуф = СуммаОстатокБуф - СтрокаТаблицы.Сумма;

КоличествоОстатокБуф = КоличествоОстатокБуф - СтрокаТаблицы.Количество;

КонецЦикла;

БуфНайденнаяСумма = НайденнаяСумма + СуммаТекущегоУровня;

Если Окр(БуфНайденнаяСумма, 2) = СуммаСписания И ТаблицаРасход.Найти(Ложь, "Флаг") = Неопределено Тогда

ФлагНашли = Истина;

ИначеЕсли БуфНайденнаяСумма < СуммаСписания Тогда

РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход, НомерСтрокиПриход + 1,

КоличествоСтрокПриход, КоличествоОстатокБуф,

СуммаОстатокБуф, БуфНайденнаяСумма);

КонецЕсли;

Если ФлагНашли Тогда

НоваяСтрока = ТаблицаСписания.Добавить();

НоваяСтрока.Период = ОбрабатываемаяСтрока.Период;

НоваяСтрока.Материалы = ТаблицаДопустимыхОпераций;

Возврат;

КонецЕсли;

Для Каждого СтрокаТаблицы из МассивНайденныхСтрок Цикл

СтрокаТаблицы.Флаг = Ложь;

КонецЦикла;

ОбработкаПрерыванияПользователя();

КонецЦикла;

КонецПроцедуры

Процедура ОсновнаяПанельСформировать(Кнопка)

Если Материал = Справочники.Материалы.ПустаяСсылка() Тогда

Предупреждение("Укажите материал!");

Возврат;

КонецЕсли;

Запрос = Новый Запрос("ВЫБРАТЬ

| ЖурналОперацийОбороты.Период КАК Период,

| ЖурналОперацийОбороты.КоличествоОборотДт КАК Количество,

| ЖурналОперацийОбороты.СуммаОборотДт КАК Сумма,

| ЖурналОперацийОбороты.СуммаОборотДт / ЖурналОперацийОбороты.КоличествоОборотДт КАК Цена

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(, , День, Счет = &Счет, , Субконто1 = &Материал, , ) КАК ЖурналОперацийОбороты

|

|УПОРЯДОЧИТЬ ПО

| Период");

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Запрос.УстановитьПараметр("Материал", Материал);

ТаблицаПриход = Запрос.Выполнить().Выгрузить();

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийОбороты.Регистратор,

| ЖурналОперацийОбороты.КоличествоОборотКт КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(, , Регистратор, Счет = &Счет, , Субконто1 = &Материал, , ) КАК ЖурналОперацийОбороты";

ТаблицаРасход = Запрос.Выполнить().Выгрузить();

ТаблицаРасход.Колонки.Добавить("Флаг", ТипБулево);

Если ТаблицаПриход.Итог("Количество") < ТаблицаРасход.Итог("Количество") Тогда

Предупреждение("Поступившее количество меньше списанного количества. Списание не возможно.");

Возврат;

КонецЕсли;

ТаблицаСписания = Новый ТаблицаЗначений;

ТаблицаСписания.Колонки.Добавить("Период", Новый ОписаниеТипов("Дата"));

ТаблицаСписания.Колонки.Добавить("Материалы", Новый ОписаниеТипов("ТаблицаЗначений"));

ФлагНашли = Ложь;

НайденнаяСумма = 0;

РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход,,ТаблицаПриход.Количество());

ТаблицаСписания.Сортировать("Период Возр");

Макет = ПолучитьМакет("Отчет");

ДокументРезультат = ЭлементыФормы.ДокументРезультат;

ДокументРезультат.Очистить();

Если Не ФлагНашли Тогда

СтрокаБезВариантов = Макет.ПолучитьОбласть("СтрокаБезВариантов");

ДокументРезультат.Вывести(СтрокаБезВариантов);

Возврат;

КонецЕсли;

ШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

СтрокаТаблицы = Макет.ПолучитьОбласть("СтрокаТаблицы");

СтрокаПодвал = Макет.ПолучитьОбласть("СтрокаПодвал");

ШапкаТаблицы.Параметры.Материал = Материал;

ШапкаТаблицы.Параметры.СуммаСписания = Формат(СуммаСписания, "ЧДЦ=2");

ДокументРезультат.Вывести(ШапкаТаблицы);

СуммаРасход = 0;

КоличествоРасход = 0;

Для Каждого СтрокаПриход Из ТаблицаПриход Цикл

Для Сч = 0 По СтрокаТаблицы.Параметры.Количество() - 1 Цикл

СтрокаТаблицы.Параметры[Сч] = "";

КонецЦикла;

СтрокаТаблицы.Параметры.Период = СтрокаПриход.Период;

СтрокаТаблицы.Параметры.КоличествоПриход = СтрокаПриход.Количество;

СтрокаТаблицы.Параметры.ЦенаПриход = СтрокаПриход.Цена;

СтрокаТаблицы.Параметры.СуммаПриход = СтрокаПриход.Сумма;

НайденнаяСтрока = ТаблицаСписания.Найти(СтрокаПриход.Период, "Период");

Если НайденнаяСтрока <> Неопределено И НайденнаяСтрока.Материалы.Количество() > 0 Тогда

Для Каждого СтрокаРасход из НайденнаяСтрока.Материалы Цикл

СтрокаТаблицы.Параметры.КоличествоРасход = СтрокаРасход.Количество;

СтрокаТаблицы.Параметры.ЦенаРасход = СтрокаРасход.Цена;

СтрокаТаблицы.Параметры.СуммаРасход = СтрокаРасход.Сумма;

ДокументРезультат.Вывести(СтрокаТаблицы);

Для Сч = 0 По СтрокаТаблицы.Параметры.Количество() - 1 Цикл

СтрокаТаблицы.Параметры[Сч] = "";

КонецЦикла;

КоличествоРасход = КоличествоРасход + СтрокаРасход.Количество;

СуммаРасход = СуммаРасход + СтрокаРасход.Сумма;

КонецЦикла;

Иначе

ДокументРезультат.Вывести(СтрокаТаблицы);

КонецЕсли;

КонецЦикла;

СтрокаПодвал.Параметры.ИтогоСуммаПриход = ТаблицаПриход.Итог("Сумма");

СтрокаПодвал.Параметры.ИтогоКоличествоПриход = ТаблицаПриход.Итог("Количество");

СтрокаПодвал.Параметры.ИтогоСуммаРасход = СуммаРасход;

СтрокаПодвал.Параметры.ИтогоКоличествоРасход = КоличествоРасход;

ДокументРезультат.Вывести(СтрокаПодвал);

КонецПроцедуры

ТипБулево = Новый ОписаниеТипов("Булево");

ТипЧисло = Новый ОписаниеТипов("Число");

Автор: Наумов Сергей

Пояснения к коду:

Перем ТипБулево;

Перем ТипЧисло;

Перем ФлагНашли;

Перем ТаблицаСписания;

В данном блоке объявляются глобальные переменные, ТипБулево и ТипЧисло – для упрощения работы с описаниями типов. ФлагНашли – для индикации нахождения решения. ТаблицаСписания – для сохранения решения в виде таблицы значений.

Функция ПолучитьТаблицуДопустимыхОпераций(ТаблицаРасход, КоличествоОстаток)

ОбщийСписокВариантов = ТаблицаРасход.Скопировать();

ОбщийСписокВариантов.Очистить();

Для Каждого СтрокаТЧ из ТаблицаРасход Цикл

Если СтрокаТЧ.Флаг Тогда

Продолжить;

КонецЕсли;

НоваяСтрока = ОбщийСписокВариантов.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаТЧ);

КонецЦикла;

СписокДопустимыхОпераций = Новый СписокЗначений;

Для Каждого Строка1 из ОбщийСписокВариантов Цикл

БуфКоличество = КоличествоОстаток;

Если Строка1.Количество > БуфКоличество Тогда

Продолжить;

КонецЕсли;

НовыйСписокОпераций = Новый ТаблицаЗначений;

НовыйСписокОпераций.Колонки.Добавить("Регистратор");

НовыйСписокОпераций.Колонки.Добавить("Количество", ТипЧисло);

СписокДопустимыхОпераций.Добавить(НовыйСписокОпераций);

НоваяСтрока = НовыйСписокОпераций.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка1);

БуфКоличество = БуфКоличество - Строка1.Количество;

Если БуфКоличество = 0 Тогда

Прервать;

КонецЕсли;

Для Каждого Строка2 Из ОбщийСписокВариантов Цикл

Если Строка1.Регистратор = Строка2.Регистратор Тогда

Продолжить;

КонецЕсли;

Если Строка2.Количество > БуфКоличество Тогда

Продолжить;

КонецЕсли;

НоваяСтрока = НовыйСписокОпераций.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка2);

БуфКоличество = БуфКоличество - Строка2.Количество;

Если БуфКоличество = 0 Тогда

Прервать;

КонецЕсли;

КонецЦикла;

КонецЦикла;

НовыйСписокОпераций = Новый ТаблицаЗначений;

НовыйСписокОпераций.Колонки.Добавить("Регистратор");

НовыйСписокОпераций.Колонки.Добавить("Количество", ТипЧисло);

СписокДопустимыхОпераций.Добавить(НовыйСписокОпераций);

Возврат СписокДопустимыхОпераций;

КонецФункции

Данная функция получает список допустимых операций. Результат представлен как СписокЗначений, в котором каждая запись это ТаблицаЗначений, с колонками Регистратор и Количество. В таблицу значений попадают операции с одинаковым Регистратором и допустимым количеством, то есть сумма по колонке Количество не превышает КоличествоОстаток.

ОбщийСписокВариантов = ТаблицаРасход.Скопировать();

ОбщийСписокВариантов.Очистить();

Для Каждого СтрокаТЧ из ТаблицаРасход Цикл

Если СтрокаТЧ.Флаг Тогда

Продолжить;

КонецЕсли;

НоваяСтрока = ОбщийСписокВариантов.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаТЧ);

КонецЦикла;

В данном блоке создается копия таблицы расхода, очищается (сделано для того, чтобы взять от таблицы расхода только колонки) и заполняется теми строками таблицы расхода, у которых Флаг не равен Истина.

СписокДопустимыхОпераций = Новый СписокЗначений;

В данной строке создается СписокЗначений для хранения результата выполнения функции.

Для Каждого Строка1 из ОбщийСписокВариантов Цикл

БуфКоличество = КоличествоОстаток;

Если Строка1.Количество > БуфКоличество Тогда

Продолжить;

КонецЕсли;

НовыйСписокОпераций = Новый ТаблицаЗначений;

НовыйСписокОпераций.Колонки.Добавить("Регистратор");

НовыйСписокОпераций.Колонки.Добавить("Количество", ТипЧисло);

СписокДопустимыхОпераций.Добавить(НовыйСписокОпераций);

НоваяСтрока = НовыйСписокОпераций.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка1);

БуфКоличество = БуфКоличество - Строка1.Количество;

Если БуфКоличество = 0 Тогда

Прервать;

КонецЕсли;

Для Каждого Строка2 Из ОбщийСписокВариантов Цикл

Если Строка1.Регистратор = Строка2.Регистратор Тогда

Продолжить;

КонецЕсли;

Если Строка2.Количество > БуфКоличество Тогда

Продолжить;

КонецЕсли;

НоваяСтрока = НовыйСписокОпераций.Добавить();

ЗаполнитьЗначенияСвойств(НоваяСтрока, Строка2);

БуфКоличество = БуфКоличество - Строка2.Количество;

Если БуфКоличество = 0 Тогда

Прервать;

КонецЕсли;

КонецЦикла;

КонецЦикла;

В данном блоке идет обход всего списка вариантов (те строки из таблицы расхода, у которых флаг = ложь), с выбором тех, которые при добавлении не переполняют остаток количества. При нахождении подходящего варианта, создается новая таблица значений, которая добавляется в список результата и в данную таблицу значений добавляются все строки относящиеся к одному и тому же регистратору и не вызывающие переполнение остатка количества. Когда остаток количества становится равен нулю – составление списка допустимых операций прекращается.

НовыйСписокОпераций = Новый ТаблицаЗначений;

НовыйСписокОпераций.Колонки.Добавить("Регистратор");

НовыйСписокОпераций.Колонки.Добавить("Количество", ТипЧисло);

СписокДопустимыхОпераций.Добавить(НовыйСписокОпераций);

Возврат СписокДопустимыхОпераций;

В данном блоке создается и добавляется в список пустая таблица значений, для предотвращения ошибок, в случае отсутствия возможных операций. После этого выполняется возврат сгенерированного списка.

Процедура РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход, НомерСтрокиПриход = 0,

КоличествоСтрокПриход, Знач КоличествоОстаток = 0, Знач СуммаОстаток = 0, Знач НайденнаяСумма = 0)

Если НомерСтрокиПриход = КоличествоСтрокПриход Тогда

Возврат;

КонецЕсли;

ОбрабатываемаяСтрока = ТаблицаПриход[НомерСтрокиПриход];

КоличествоОстаток = ОбрабатываемаяСтрока.Количество + КоличествоОстаток;

СуммаОстаток = ОбрабатываемаяСтрока.Сумма + СуммаОстаток;

МассивНайденныхСтрок = Новый Массив;

СписокДопустимыхОпераций = ПолучитьТаблицуДопустимыхОпераций(ТаблицаРасход, КоличествоОстаток);

Для Каждого ЭлементСписка из СписокДопустимыхОпераций Цикл

ТаблицаДопустимыхОпераций = ЭлементСписка.Значение;

ТаблицаДопустимыхОпераций.Колонки.Добавить("Цена", ТипЧисло);

ТаблицаДопустимыхОпераций.Колонки.Добавить("Сумма", ТипЧисло);

МассивНайденныхСтрок.Очистить();

ВсегоСписать = 0;

СуммаТекущегоУровня = 0;

СуммаОстатокБуф = СуммаОстаток;

КоличествоОстатокБуф = КоличествоОстаток;

Для Каждого СтрокаТаблицы из ТаблицаДопустимыхОпераций Цикл

ВсегоСписать = ВсегоСписать + СтрокаТаблицы.Количество;

НайденнаяСтрока = ТаблицаРасход.Найти(СтрокаТаблицы.Регистратор, "Регистратор");

НайденнаяСтрока.Флаг = Истина;

МассивНайденныхСтрок.Добавить(НайденнаяСтрока);

СтрокаТаблицы.Цена = СуммаОстатокБуф / КоличествоОстатокБуф;

СтрокаТаблицы.Сумма = СтрокаТаблицы.Количество \* СтрокаТаблицы.Цена;

СуммаТекущегоУровня = СуммаТекущегоУровня + СтрокаТаблицы.Сумма;

СуммаОстатокБуф = СуммаОстатокБуф - СтрокаТаблицы.Сумма;

КоличествоОстатокБуф = КоличествоОстатокБуф - СтрокаТаблицы.Количество;

КонецЦикла;

БуфНайденнаяСумма = НайденнаяСумма + СуммаТекущегоУровня;

Если Окр(БуфНайденнаяСумма, 2) = СуммаСписания И ТаблицаРасход.Найти(Ложь, "Флаг") = Неопределено Тогда

ФлагНашли = Истина;

ИначеЕсли БуфНайденнаяСумма < СуммаСписания Тогда

РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход, НомерСтрокиПриход + 1,

КоличествоСтрокПриход, КоличествоОстатокБуф,

СуммаОстатокБуф, БуфНайденнаяСумма);

КонецЕсли;

Если ФлагНашли Тогда

НоваяСтрока = ТаблицаСписания.Добавить();

НоваяСтрока.Период = ОбрабатываемаяСтрока.Период;

НоваяСтрока.Материалы = ТаблицаДопустимыхОпераций;

Возврат;

КонецЕсли;

Для Каждого СтрокаТаблицы из МассивНайденныхСтрок Цикл

СтрокаТаблицы.Флаг = Ложь;

КонецЦикла;

ОбработкаПрерыванияПользователя();

КонецЦикла;

КонецПроцедуры

Данная процедура рекурсивно обходит дерево решений, по завершению работы этой процедуры, в глобальной переменной ТаблицаСписания будет находиться результат выполнения.

Если НомерСтрокиПриход = КоличествоСтрокПриход Тогда

Возврат;

КонецЕсли;

В данном блоке проверяется, был ли достигнут конец таблицы прихода (все записи прихода проверены).

ОбрабатываемаяСтрока = ТаблицаПриход[НомерСтрокиПриход];

КоличествоОстаток = ОбрабатываемаяСтрока.Количество + КоличествоОстаток;

СуммаОстаток = ОбрабатываемаяСтрока.Сумма + СуммаОстаток;

В данном блоке получаются остатки количества и суммы для текущей строки из таблицы прихода.

МассивНайденныхСтрок = Новый Массив;

СписокДопустимыхОпераций = ПолучитьТаблицуДопустимыхОпераций(ТаблицаРасход, КоличествоОстаток);

В данном блоке создается массив для хранения строк таблицы расхода, над которыми будем работать (для управления флагом). И получаем весь список допустимых операций.

Для Каждого ЭлементСписка из СписокДопустимыхОпераций Цикл

ТаблицаДопустимыхОпераций = ЭлементСписка.Значение;

ТаблицаДопустимыхОпераций.Колонки.Добавить("Цена", ТипЧисло);

ТаблицаДопустимыхОпераций.Колонки.Добавить("Сумма", ТипЧисло);

МассивНайденныхСтрок.Очистить();

ВсегоСписать = 0;

СуммаТекущегоУровня = 0;

СуммаОстатокБуф = СуммаОстаток;

КоличествоОстатокБуф = КоличествоОстаток;

Для Каждого СтрокаТаблицы из ТаблицаДопустимыхОпераций Цикл

ВсегоСписать = ВсегоСписать + СтрокаТаблицы.Количество;

НайденнаяСтрока = ТаблицаРасход.Найти(СтрокаТаблицы.Регистратор, "Регистратор");

НайденнаяСтрока.Флаг = Истина;

МассивНайденныхСтрок.Добавить(НайденнаяСтрока);

СтрокаТаблицы.Цена = СуммаОстатокБуф / КоличествоОстатокБуф;

СтрокаТаблицы.Сумма = СтрокаТаблицы.Количество \* СтрокаТаблицы.Цена;

СуммаТекущегоУровня = СуммаТекущегоУровня + СтрокаТаблицы.Сумма;

СуммаОстатокБуф = СуммаОстатокБуф - СтрокаТаблицы.Сумма;

КоличествоОстатокБуф = КоличествоОстатокБуф - СтрокаТаблицы.Количество;

КонецЦикла;

БуфНайденнаяСумма = НайденнаяСумма + СуммаТекущегоУровня;

Если Окр(БуфНайденнаяСумма, 2) = СуммаСписания И ТаблицаРасход.Найти(Ложь, "Флаг") = Неопределено Тогда

ФлагНашли = Истина;

ИначеЕсли БуфНайденнаяСумма < СуммаСписания Тогда

РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход, НомерСтрокиПриход + 1,

КоличествоСтрокПриход, КоличествоОстатокБуф,

СуммаОстатокБуф, БуфНайденнаяСумма);

КонецЕсли;

Если ФлагНашли Тогда

НоваяСтрока = ТаблицаСписания.Добавить();

НоваяСтрока.Период = ОбрабатываемаяСтрока.Период;

НоваяСтрока.Материалы = ТаблицаДопустимыхОпераций;

Возврат;

КонецЕсли;

Для Каждого СтрокаТаблицы из МассивНайденныхСтрок Цикл

СтрокаТаблицы.Флаг = Ложь;

КонецЦикла;

ОбработкаПрерыванияПользователя();

КонецЦикла;

В данном блоке происходит обход списка допустимых операций. Для начала в таблицу значений допустимых операций добавляются две колонки – Цена и Сумма, типа Число. Далее очищается массив найденных строк, для заполнения его новыми строками, которые будем использовать на данной итерации цикла. Каждая строка из таблицы допустимых операций находится по регистратору в таблице расхода, с записью результата в массив найденных строк и выставлением флага в значение истина, а также рассчитывается списанная сумма, цена, сумма и сумма по уровню обновляются остатки. Далее определяется значение найденной суммы. Если найденная сумма равна сумме списания (с округлением 0.5 = 1) и в таблице расхода все строки помечены флагом истина, то запоминаем, что решение найдено и рекурсивно возвращаясь к корню вызова функции таблица списания будет заполнена. Если найденная сумма меньше суммы списания, то функция рекурсивно вызывается, для следующей строки прихода. Далее если ФлагНашли истина, значит решение найдено и требуется добавлять обрабатываемую строку в ТаблицуСписания, с возвратом из процедуры. Если же решение не найдено, то используя массив найденных строк, таблица расхода будет восстановлена к первоначальному состоянию. Так же вызываем процедуру ОбработкаПрерыванияПользователя(), для возможности остановить работу процедуры (в случае зацикливания к примеру) клавишами Ctrl+Break.

Процедура ОсновнаяПанельСформировать(Кнопка)

Если Материал = Справочники.Материалы.ПустаяСсылка() Тогда

Предупреждение("Укажите материал!");

Возврат;

КонецЕсли;

Запрос = Новый Запрос("ВЫБРАТЬ

| ЖурналОперацийОбороты.Период КАК Период,

| ЖурналОперацийОбороты.КоличествоОборотДт КАК Количество,

| ЖурналОперацийОбороты.СуммаОборотДт КАК Сумма,

| ЖурналОперацийОбороты.СуммаОборотДт / ЖурналОперацийОбороты.КоличествоОборотДт КАК Цена

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(, , День, Счет = &Счет, , Субконто1 = &Материал, , ) КАК ЖурналОперацийОбороты

|

|УПОРЯДОЧИТЬ ПО

| Период");

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Запрос.УстановитьПараметр("Материал", Материал);

ТаблицаПриход = Запрос.Выполнить().Выгрузить();

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийОбороты.Регистратор,

| ЖурналОперацийОбороты.КоличествоОборотКт КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(, , Регистратор, Счет = &Счет, , Субконто1 = &Материал, , ) КАК ЖурналОперацийОбороты";

ТаблицаРасход = Запрос.Выполнить().Выгрузить();

ТаблицаРасход.Колонки.Добавить("Флаг", ТипБулево);

Если ТаблицаПриход.Итог("Количество") < ТаблицаРасход.Итог("Количество") Тогда

Предупреждение("Поступившее количество меньше списанного количества. Списание не возможно.");

Возврат;

КонецЕсли;

ТаблицаСписания = Новый ТаблицаЗначений;

ТаблицаСписания.Колонки.Добавить("Период", Новый ОписаниеТипов("Дата"));

ТаблицаСписания.Колонки.Добавить("Материалы", Новый ОписаниеТипов("ТаблицаЗначений"));

ФлагНашли = Ложь;

НайденнаяСумма = 0;

РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход,,ТаблицаПриход.Количество());

ТаблицаСписания.Сортировать("Период Возр");

Макет = ПолучитьМакет("Отчет");

ДокументРезультат = ЭлементыФормы.ДокументРезультат;

ДокументРезультат.Очистить();

Если Не ФлагНашли Тогда

СтрокаБезВариантов = Макет.ПолучитьОбласть("СтрокаБезВариантов");

ДокументРезультат.Вывести(СтрокаБезВариантов);

Возврат;

КонецЕсли;

ШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

СтрокаТаблицы = Макет.ПолучитьОбласть("СтрокаТаблицы");

СтрокаПодвал = Макет.ПолучитьОбласть("СтрокаПодвал");

ШапкаТаблицы.Параметры.Материал = Материал;

ШапкаТаблицы.Параметры.СуммаСписания = Формат(СуммаСписания, "ЧДЦ=2");

ДокументРезультат.Вывести(ШапкаТаблицы);

СуммаРасход = 0;

КоличествоРасход = 0;

Для Каждого СтрокаПриход Из ТаблицаПриход Цикл

Для Сч = 0 По СтрокаТаблицы.Параметры.Количество() - 1 Цикл

СтрокаТаблицы.Параметры[Сч] = "";

КонецЦикла;

СтрокаТаблицы.Параметры.Период = СтрокаПриход.Период;

СтрокаТаблицы.Параметры.КоличествоПриход = СтрокаПриход.Количество;

СтрокаТаблицы.Параметры.ЦенаПриход = СтрокаПриход.Цена;

СтрокаТаблицы.Параметры.СуммаПриход = СтрокаПриход.Сумма;

НайденнаяСтрока = ТаблицаСписания.Найти(СтрокаПриход.Период, "Период");

Если НайденнаяСтрока <> Неопределено И НайденнаяСтрока.Материалы.Количество() > 0 Тогда

Для Каждого СтрокаРасход из НайденнаяСтрока.Материалы Цикл

СтрокаТаблицы.Параметры.КоличествоРасход = СтрокаРасход.Количество;

СтрокаТаблицы.Параметры.ЦенаРасход = СтрокаРасход.Цена;

СтрокаТаблицы.Параметры.СуммаРасход = СтрокаРасход.Сумма;

ДокументРезультат.Вывести(СтрокаТаблицы);

Для Сч = 0 По СтрокаТаблицы.Параметры.Количество() - 1 Цикл

СтрокаТаблицы.Параметры[Сч] = "";

КонецЦикла;

КоличествоРасход = КоличествоРасход + СтрокаРасход.Количество;

СуммаРасход = СуммаРасход + СтрокаРасход.Сумма;

КонецЦикла;

Иначе

ДокументРезультат.Вывести(СтрокаТаблицы);

КонецЕсли;

КонецЦикла;

СтрокаПодвал.Параметры.ИтогоСуммаПриход = ТаблицаПриход.Итог("Сумма");

СтрокаПодвал.Параметры.ИтогоКоличествоПриход = ТаблицаПриход.Итог("Количество");

СтрокаПодвал.Параметры.ИтогоСуммаРасход = СуммаРасход;

СтрокаПодвал.Параметры.ИтогоКоличествоРасход = КоличествоРасход;

ДокументРезультат.Вывести(СтрокаПодвал);

КонецПроцедуры

Данная процедура вызывается при нажатии кнопки формы Сформировать. В ней происходит получение данных из базы, вызов функции поиска решения и вывод результата в табличный документ.

Если Материал = Справочники.Материалы.ПустаяСсылка() Тогда

Предупреждение("Укажите материал!");

Возврат;

КонецЕсли;

В данном блоке проверяется указан ли материал.

Запрос = Новый Запрос("ВЫБРАТЬ

| ЖурналОперацийОбороты.Период КАК Период,

| ЖурналОперацийОбороты.КоличествоОборотДт КАК Количество,

| ЖурналОперацийОбороты.СуммаОборотДт КАК Сумма,

| ЖурналОперацийОбороты.СуммаОборотДт / ЖурналОперацийОбороты.КоличествоОборотДт КАК Цена

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(, , День, Счет = &Счет, , Субконто1 = &Материал, , ) КАК ЖурналОперацийОбороты

|

|УПОРЯДОЧИТЬ ПО

| Период");

Запрос.УстановитьПараметр("Счет", ПланыСчетов.ПланСчетов.Материалы);

Запрос.УстановитьПараметр("Материал", Материал);

ТаблицаПриход = Запрос.Выполнить().Выгрузить();

В данном блоке происходит получение таблицы приходов из базы и выгрузка в переменную.

Запрос.Текст = "ВЫБРАТЬ

| ЖурналОперацийОбороты.Регистратор,

| ЖурналОперацийОбороты.КоличествоОборотКт КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналОпераций.Обороты(, , Регистратор, Счет = &Счет, , Субконто1 = &Материал, , ) КАК ЖурналОперацийОбороты";

ТаблицаРасход = Запрос.Выполнить().Выгрузить();

ТаблицаРасход.Колонки.Добавить("Флаг", ТипБулево);

В данном блоке происходит получение таблицы расходов из базы и выгрузка в переменную, а также добавление колонки Флаг.

Если ТаблицаПриход.Итог("Количество") < ТаблицаРасход.Итог("Количество") Тогда

Предупреждение("Поступившее количество меньше списанного количества. Списание не возможно.");

Возврат;

КонецЕсли;

В данном блоке проверяется корректность данных.

ТаблицаСписания = Новый ТаблицаЗначений;

ТаблицаСписания.Колонки.Добавить("Период", Новый ОписаниеТипов("Дата"));

ТаблицаСписания.Колонки.Добавить("Материалы", Новый ОписаниеТипов("ТаблицаЗначений"));

ФлагНашли = Ложь;

НайденнаяСумма = 0;

РассчитатьДеревоРешений(ТаблицаПриход, ТаблицаРасход,,ТаблицаПриход.Количество());

ТаблицаСписания.Сортировать("Период Возр");

В данном блоке создается таблица значений для хранения результата, с колонками Период и Материалы, указываются начальные данные (флаг = ложь и найденная сумма = 0) и вызывается процедура нахождения решения. После нахождения решения таблица с результатом сортируется по колонке период.

Макет = ПолучитьМакет("Отчет");

ДокументРезультат = ЭлементыФормы.ДокументРезультат;

ДокументРезультат.Очистить();

В данном блоке получается макет для вывода результата и табличный документ, в который будет выводиться результат, с очисткой от старых данных.

Если Не ФлагНашли Тогда

СтрокаБезВариантов = Макет.ПолучитьОбласть("СтрокаБезВариантов");

ДокументРезультат.Вывести(СтрокаБезВариантов);

Возврат;

КонецЕсли;

Если поиск решений не дал результатов, то в табличный документ будет выведена соответствующая область.

ШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");

СтрокаТаблицы = Макет.ПолучитьОбласть("СтрокаТаблицы");

СтрокаПодвал = Макет.ПолучитьОбласть("СтрокаПодвал");

В данном блоке идет получение областей макета.

ШапкаТаблицы.Параметры.Материал = Материал;

ШапкаТаблицы.Параметры.СуммаСписания = Формат(СуммаСписания, "ЧДЦ=2");

ДокументРезультат.Вывести(ШапкаТаблицы);

В данном блоке выводится шапка таблицы.

СуммаРасход = 0;

КоличествоРасход = 0;

Для Каждого СтрокаПриход Из ТаблицаПриход Цикл

Для Сч = 0 По СтрокаТаблицы.Параметры.Количество() - 1 Цикл

СтрокаТаблицы.Параметры[Сч] = "";

КонецЦикла;

СтрокаТаблицы.Параметры.Период = СтрокаПриход.Период;

СтрокаТаблицы.Параметры.КоличествоПриход = СтрокаПриход.Количество;

СтрокаТаблицы.Параметры.ЦенаПриход = СтрокаПриход.Цена;

СтрокаТаблицы.Параметры.СуммаПриход = СтрокаПриход.Сумма;

НайденнаяСтрока = ТаблицаСписания.Найти(СтрокаПриход.Период, "Период");

Если НайденнаяСтрока <> Неопределено И НайденнаяСтрока.Материалы.Количество() > 0 Тогда

Для Каждого СтрокаРасход из НайденнаяСтрока.Материалы Цикл

СтрокаТаблицы.Параметры.КоличествоРасход = СтрокаРасход.Количество;

СтрокаТаблицы.Параметры.ЦенаРасход = СтрокаРасход.Цена;

СтрокаТаблицы.Параметры.СуммаРасход = СтрокаРасход.Сумма;

ДокументРезультат.Вывести(СтрокаТаблицы);

Для Сч = 0 По СтрокаТаблицы.Параметры.Количество() - 1 Цикл

СтрокаТаблицы.Параметры[Сч] = "";

КонецЦикла;

КоличествоРасход = КоличествоРасход + СтрокаРасход.Количество;

СуммаРасход = СуммаРасход + СтрокаРасход.Сумма;

КонецЦикла;

Иначе

ДокументРезультат.Вывести(СтрокаТаблицы);

КонецЕсли;

КонецЦикла;

В данном блоке выводятся операции прихода и операции списания, полученные в результате решения. А также считается сумма расхода и количество расхода.

СтрокаПодвал.Параметры.ИтогоСуммаПриход = ТаблицаПриход.Итог("Сумма");

СтрокаПодвал.Параметры.ИтогоКоличествоПриход = ТаблицаПриход.Итог("Количество");

СтрокаПодвал.Параметры.ИтогоСуммаРасход = СуммаРасход;

СтрокаПодвал.Параметры.ИтогоКоличествоРасход = КоличествоРасход;

ДокументРезультат.Вывести(СтрокаПодвал);

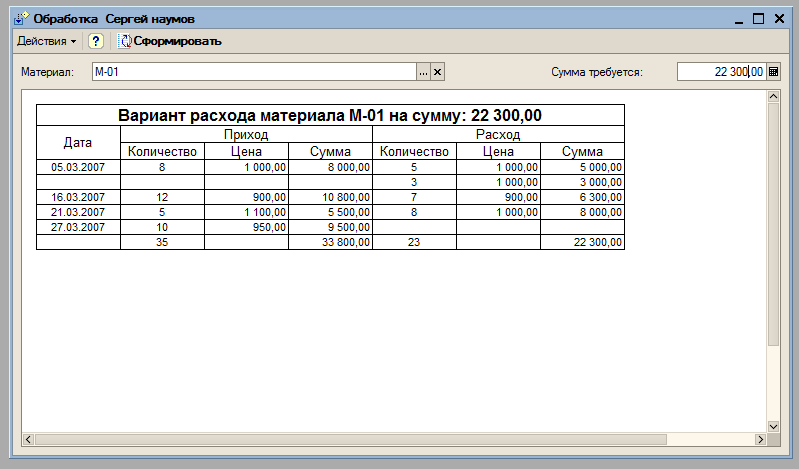
В данном блоке выводятся итоговые сумма и количество по расходу и приходу.

ТипБулево = Новый ОписаниеТипов("Булево");

ТипЧисло = Новый ОписаниеТипов("Число");

В данном блоке происходит получение описаний типов для глобальных переменных.

Сформированный отчет



2004 год 2 тур

Требование

На торговом предприятии ХХХ в информационной базе программы «1С:Бухгалтерия» хранится учетная информация о реализации товаров покупателям. Реализация товаров отражается в виде проводок в дебет счета 62.1 с кредита счета 41.1. Проводки сгруппированы по операциям так, что каждая операция содержит информацию об одной покупке, сделанной одним покупателем. Каждая покупка может состоять из одного или нескольких различных товаров в произвольном количестве. Необходимо разработать программу, позволяющую выявить наиболее часто встречающиеся сочетания товаров, приобретаемых покупателями в рамках одной покупки (то есть, выявить наиболее популярные комбинации), из 2-х и 3-х товаров.

При разработке программы следует исходить из следующих условий и допущений.

1. На счете 41.1 аналитический учет ведется по номенклатуре товаров, аналитический учет на счете 62.1 не предусмотрен.
2. В проводках одной операции каждый товар может встречаться только однократно.
3. При анализе встречаемости комбинаций количество отпущенного товара не учитывается.
4. При наличии нескольких комбинаций товаров с одинаковой частотой встречаемости необходимо включить в отчет все такие комбинации.
5. Результатная информация должна содержать для каждой комбинации перечень наименований входящих в нее товаров и частоту ее встречаемости в ИБ, например, в следующем виде:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Количество товаров в комбинации | Комбинации товаров | Частота встречаемости |
| 1 | 2 | Товар15, Товар39 | 31 |
| 2 | 2 | Товар23, Товар75 | 31 |
| 3 | 2 | Товар29, Товар88 | 31 |
| 4 | 3 | Товар15, Товар39, Товар 62 | 11 |
| 5 | 3 | Товар19, Товар41, Товар 77 | 11 |

Для отладки программы необходимо подготовить информационную базу предприятия ХХХ, содержащую операции реализации товаров за некоторый период. Поскольку количество и стоимость отпущенного товара не влияют на результаты анализа, данные показатели во всех проводках могут быть одинаковыми. Первоначальные остатки по товарам вводятся одной операцией, содержащей проводки в дебет счета 41.1 с кредита счета 00.

Решение

Перед решением требуется создать информационную базу, и заполнить данными. Требуемые объекты конфигурации указаны в задании. Для дальнейшей отладки будет использоваться следующий набор начальных данных:

Ввод остатков:

|  |  |  |
| --- | --- | --- |
| Товар | Стоимость | Количество |
| Изд-01 | 5000 | 5 |
| Изд-02 | 5000 | 5 |
| Изд-03 | 5000 | 5 |
| Изд-04 | 5000 | 5 |
| Изд-05 | 5000 | 5 |
| Изд-06 | 5000 | 5 |
| Изд-07 | 5000 | 5 |
| Изд-08 | 5000 | 5 |
| Изд-09 | 5000 | 5 |

Реализация товара:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номер | Товары | | | | | |
| 1 | Изд-01 | | | | | |
| 2 | Изд-01 | | | Изд-03 | | |
| 3 | Изд-03 | | | Изд-01 | | |
| 4 | Изд-03 | | Изд-01 | | Изд-02 | |
| 5 | Изд-02 | | Изд-01 | | Изд-03 | |
| 6 | Изд-03 | Изд-01 | | Изд-04 | | Изд-02 |
| 7 | Изд-03 | | Изд-01 | | Изд-04 | |
| 8 | Изд-03 | Изд-04 | | Изд-02 | | Изд-01 |
| 9 | Изд-03 | | Изд-04 | | Изд-01 | |
| 10 | Изд-07 | | | | | |
| 11 | Изд-07 | | Изд-04 | | Изд-02 | |
| 12 | Изд-07 | | | | | |
| 13 | Изд-01 | | | | | |
| 14 | Изд-01 | | | | | |

Для решения задачи будет использоваться объект АнализДанных. Данный объект предназначен для выполнения анализа данных (кластерный анализ, поиск ассоциативных правил, поиск последовательностей и другие виды анализа). Для данной задачи требуется «поиск ассоциативных правил». Данный тип анализа осуществляет поиск часто встречае­мых вместе групп объектов или значений характеристик, а также производит поиск правил ассоциаций. С помощью поиска ас­социаций легко можно опреде­лять приобретаемые вместе товары. Для анализа передаются следующие пары - Документ «Реализация товара» - товар (проданный по данному документу). Для требуемого результата нужно установить следующие параметры анализа: тип отсечения правил – покрытые (для отсечения правил, покрытых другими правилами) и порядок – по количеству случаев. После выполнения анализа требуется разобрать полученные результаты и отобразить в табличный документ.

Реализация

&НаКлиенте

Процедура Сформировать(Команда)

СформироватьНаСервере();

КонецПроцедуры

&НаСервере

Процедура СформироватьНаСервере()

ТабДок.Очистить();

ЭтотОбъект=РеквизитФормыВЗначение("Отчет");

Макет = ЭтотОбъект.ПолучитьМакет("Макет");

Область = Макет.ПолучитьОбласть("Шапка");

ТабДок.Вывести(Область);

Анализ = Новый АнализДанных;

Анализ.ТипАнализа = Тип("АнализДанныхПоискАссоциаций");

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПродажиКомпанииОбороты.Регистратор Как ДокументПродажи,

| ПродажиКомпанииОбороты.Субконто1

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Обороты(, , Регистратор, Счет = &Счет, , , , ) КАК ПродажиКомпанииОбороты

|ГДЕ

|ПродажиКомпанииОбороты.КорСчет = &НСчет

|УПОРЯДОЧИТЬ ПО

| ДокументПродажи";

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

Запрос.УстановитьПараметр("НСчет",ПланыСчетов.ПланСчетов.Продажи);

Анализ.ИсточникДанных = Запрос.Выполнить();

Анализ.Параметры.ТипОтсеченияПравил.Значение = ТипОтсеченияПравилАссоциации.Покрытые;

Анализ.Параметры.Порядок.Значение = ТипУпорядочиванияПравилАссоциацииАнализаДанных.ПоКоличествуСлучаев;

РезультатАнализа = Анализ.Выполнить();

АнализТовар = РезультатАнализа.Правила;

МаксимумДва = 0;

КоличествоТри = 0;

КоличествоДва = 0;

МаксимумТри = 0;

МассивТри = новый Массив();

МассивДва = новый Массив();

НомерСтроки=1;

Для каждого Правило Из АнализТовар Цикл

Сообщить(Правило);

Строка = РазборПравила(Правило);

Если Строка["Количество"]=3 Тогда

Если Строка["Процент"]>=МаксимумТри Тогда

Массив = Строка["Массив"];

Если ПроверкаПар(МассивТри,Массив) Тогда

МассивТри.Добавить((Массив));

КоличествоТри = Окр(РезультатАнализа.КоличествоОбъектов\*Строка["Процент"])/100;

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.НомерСтроки=НомерСтроки;

Область.Параметры.КолКомб=3;

Область.Параметры.Комбинации=Массив[0]+", "+Массив[1]+", "+Массив[2];

Область.Параметры.Частота=КоличествоТри;

ТабДок.Вывести(Область);

МаксимумТри = Строка["Процент"];

НомерСтроки=НомерСтроки+1;

КонецЕсли;

КонецЕсли;

КонецЕсли;

Если Строка["Количество"]=2 Тогда

Если Строка["Процент"]>=МаксимумДва Тогда

Массив = Строка["Массив"];

Если ПроверкаПар(МассивДва,Массив) Тогда

МассивДва.Добавить(Массив);

МаксимумДва = Строка["Процент"];

КоличествоДва = Окр(РезультатАнализа.КоличествоОбъектов\*Строка["Процент"])/100;

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.НомерСтроки=НомерСтроки;

Область.Параметры.КолКомб=2;

Область.Параметры.Комбинации=Массив[0]+", "+Массив[1];

Область.Параметры.Частота=КоличествоДва;

ТабДок.Вывести(Область);

НомерСтроки=НомерСтроки+1;

КонецЕсли;

КонецЕсли;

КонецЕсли;

КонецЦикла;

ТабДок.ТолькоПросмотр=Истина;

КонецПроцедуры

Функция РазборПравила(Правило)

ИндексСкобочки=0;

Бегун=СтрДлина(Правило)-1;

Пока Не ИндексСкобочки И Бегун>=0 Цикл

Буква = Сред(Правило,Бегун,1);

Если(Буква="(") Тогда

ИндексСкобочки=Бегун;

КонецЕсли;

Бегун = Бегун-1;

КонецЦикла;

ПроцентСлуч = Прав(Правило,СтрДлина(Правило) - ИндексСкобочки);

//ПроцентСлуч = Лев(ПроцентСлуч, СтрДлина(ПроцентСлуч)-1);

ПроцентСлуч = Сред(ПроцентСлуч,Найти(ПроцентСлуч,";")+2);

ПроцентСлуч = Лев(ПроцентСлуч,Найти(ПроцентСлуч,";")-1);

Товары = Лев (Правило,ИндексСкобочки - 1);

Товары = СтрЗаменить(Товары,"Субконто1 = ","");

Товары = СтрЗаменить(Товары," => ",",");

МассТов = РазложитьСтрокуВМассивПодстрок(Товары, ",");

Для бегун = 0 По МассТов.Количество()-1 Цикл

МассТов[бегун] = СокрЛП(МассТов[бегун]);

КонецЦикла;

Результат = новый Соответствие();

Результат.Вставить("Процент",Число(ПроцентСлуч));

Результат.Вставить("Количество",МассТов.Количество());

Результат.Вставить("Массив",МассТов);

Возврат Результат;

КонецФункции

Функция РазложитьСтрокуВМассивПодстрок(Знач Стр, Разделитель = ",") Экспорт

МассивСтрок = Новый Массив();

Если Разделитель = " " Тогда

Стр = СокрЛП(Стр);

Пока 1=1 Цикл

Поз = Найти(Стр,Разделитель);

Если Поз=0 Тогда

МассивСтрок.Добавить(Стр);

Возврат МассивСтрок;

КонецЕсли;

МассивСтрок.Добавить(Лев(Стр,Поз-1));

Стр = СокрЛ(Сред(Стр,Поз));

КонецЦикла;

Иначе

ДлинаРазделителя = СтрДлина(Разделитель);

Пока 1=1 Цикл

Поз = Найти(Стр,Разделитель);

Если Поз=0 Тогда

МассивСтрок.Добавить(Стр);

Возврат МассивСтрок;

КонецЕсли;

МассивСтрок.Добавить(Лев(Стр,Поз-1));

Стр = Сред(Стр,Поз+ДлинаРазделителя);

КонецЦикла;

КонецЕсли;

КонецФункции // глРазложить

Функция ПроверкаПар(Массив,Строка)

Если Массив.Количество()=0 Тогда

Возврат Истина;

КонецЕсли;

Нашлось = Истина;

МассивНовый = Строка;//РазложитьСтрокуВМассивПодстрок(Строка,",");

Для каждого эл из Массив Цикл

МассивЭл = эл;//РазложитьСтрокуВМассивПодстрок(эл,",");

Если МассивЭл.Количество()=МассивНовый.Количество() Тогда

Для каждого ололо из МассивНовый Цикл

Позиция = МассивЭл.Найти(ололо);

Нашлось = Нашлось И (Позиция<>неопределено);

КонецЦикла;

Если Нашлось Тогда

Возврат Ложь;

КонецЕсли;

КонецЕсли;

КонецЦикла;

Возврат Истина;

КонецФункции

Автор: <http://1c-olympiad.prutzkow.com/>

Пояснения к коду:

&НаСервере

Процедура СформироватьНаСервере()

ТабДок.Очистить();

ЭтотОбъект=РеквизитФормыВЗначение("Отчет");

Макет = ЭтотОбъект.ПолучитьМакет("Макет");

Область = Макет.ПолучитьОбласть("Шапка");

ТабДок.Вывести(Область);

Анализ = Новый АнализДанных;

Анализ.ТипАнализа = Тип("АнализДанныхПоискАссоциаций");

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПродажиКомпанииОбороты.Регистратор Как ДокументПродажи,

| ПродажиКомпанииОбороты.Субконто1

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Обороты(, , Регистратор, Счет = &Счет, , , , ) КАК ПродажиКомпанииОбороты

|ГДЕ

|ПродажиКомпанииОбороты.КорСчет = &НСчет

|УПОРЯДОЧИТЬ ПО

| ДокументПродажи";

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

Запрос.УстановитьПараметр("НСчет",ПланыСчетов.ПланСчетов.Продажи);

Анализ.ИсточникДанных = Запрос.Выполнить();

Анализ.Параметры.ТипОтсеченияПравил.Значение = ТипОтсеченияПравилАссоциации.Покрытые;

Анализ.Параметры.Порядок.Значение = ТипУпорядочиванияПравилАссоциацииАнализаДанных.ПоКоличествуСлучаев;

РезультатАнализа = Анализ.Выполнить();

АнализТовар = РезультатАнализа.Правила;

МаксимумДва = 0;

КоличествоТри = 0;

КоличествоДва = 0;

МаксимумТри = 0;

МассивТри = новый Массив();

МассивДва = новый Массив();

НомерСтроки=1;

Для каждого Правило Из АнализТовар Цикл

Сообщить(Правило);

Строка = РазборПравила(Правило);

Если Строка["Количество"]=3 Тогда

Если Строка["Процент"]>=МаксимумТри Тогда

Массив = Строка["Массив"];

Если ПроверкаПар(МассивТри,Массив) Тогда

МассивТри.Добавить((Массив));

КоличествоТри = Окр(РезультатАнализа.КоличествоОбъектов\*Строка["Процент"])/100;

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.НомерСтроки=НомерСтроки;

Область.Параметры.КолКомб=3;

Область.Параметры.Комбинации=Массив[0]+", "+Массив[1]+", "+Массив[2];

Область.Параметры.Частота=КоличествоТри;

ТабДок.Вывести(Область);

МаксимумТри = Строка["Процент"];

НомерСтроки=НомерСтроки+1;

КонецЕсли;

КонецЕсли;

КонецЕсли;

Если Строка["Количество"]=2 Тогда

Если Строка["Процент"]>=МаксимумДва Тогда

Массив = Строка["Массив"];

Если ПроверкаПар(МассивДва,Массив) Тогда

МассивДва.Добавить(Массив);

МаксимумДва = Строка["Процент"];

КоличествоДва = Окр(РезультатАнализа.КоличествоОбъектов\*Строка["Процент"])/100;

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.НомерСтроки=НомерСтроки;

Область.Параметры.КолКомб=2;

Область.Параметры.Комбинации=Массив[0]+", "+Массив[1];

Область.Параметры.Частота=КоличествоДва;

ТабДок.Вывести(Область);

НомерСтроки=НомерСтроки+1;

КонецЕсли;

КонецЕсли;

КонецЕсли;

КонецЦикла;

ТабДок.ТолькоПросмотр=Истина;

КонецПроцедуры

Данная процедура – обработчик нажатия кнопки Сформировать. Здесь происходит очистка табличного документа от старых данных, получение данных для анализа, настройка анализа и вывод подходящих результатов.

ТабДок.Очистить();

ЭтотОбъект=РеквизитФормыВЗначение("Отчет");

Макет = ЭтотОбъект.ПолучитьМакет("Макет");

Область = Макет.ПолучитьОбласть("Шапка");

ТабДок.Вывести(Область);

В данном блоке идет очистка от старых данных, получение макета и области для вывода результата.

Анализ = Новый АнализДанных;

Анализ.ТипАнализа = Тип("АнализДанныхПоискАссоциаций");

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ПродажиКомпанииОбороты.Регистратор Как ДокументПродажи,

| ПродажиКомпанииОбороты.Субконто1

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Обороты(, , Регистратор, Счет = &Счет, , , , ) КАК ПродажиКомпанииОбороты

|ГДЕ

|ПродажиКомпанииОбороты.КорСчет = &НСчет

|УПОРЯДОЧИТЬ ПО

| ДокументПродажи";

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

Запрос.УстановитьПараметр("НСчет",ПланыСчетов.ПланСчетов.Продажи);

Анализ.ИсточникДанных = Запрос.Выполнить();

Анализ.Параметры.ТипОтсеченияПравил.Значение = ТипОтсеченияПравилАссоциации.Покрытые;

Анализ.Параметры.Порядок.Значение = ТипУпорядочиванияПравилАссоциацииАнализаДанных.ПоКоличествуСлучаев;

РезультатАнализа = Анализ.Выполнить();

В данном блоке идет получение пар «документ – товар» используя запрос, создается объект АнализДанных и настраивается (тип отсечения, порядок, источник данных, тип анализа), после чего анализ выполняется.

АнализТовар = РезультатАнализа.Правила;

МаксимумДва = 0;

КоличествоТри = 0;

КоличествоДва = 0;

МаксимумТри = 0;

МассивТри = новый Массив();

МассивДва = новый Массив();

НомерСтроки=1;

Для каждого Правило Из АнализТовар Цикл

Строка = РазборПравила(Правило);

Если Строка["Количество"]=3 Тогда

Если Строка["Процент"]>=МаксимумТри Тогда

Массив = Строка["Массив"];

Если ПроверкаПар(МассивТри,Массив) Тогда

МассивТри.Добавить((Массив));

КоличествоТри = Окр(РезультатАнализа.КоличествоОбъектов\*Строка["Процент"])/100;

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.НомерСтроки=НомерСтроки;

Область.Параметры.КолКомб=3;

Область.Параметры.Комбинации=Массив[0]+", "+Массив[1]+", "+Массив[2];

Область.Параметры.Частота=КоличествоТри;

ТабДок.Вывести(Область);

МаксимумТри = Строка["Процент"];

НомерСтроки=НомерСтроки+1;

КонецЕсли;

КонецЕсли;

КонецЕсли;

Если Строка["Количество"]=2 Тогда

Если Строка["Процент"]>=МаксимумДва Тогда

Массив = Строка["Массив"];

Если ПроверкаПар(МассивДва,Массив) Тогда

МассивДва.Добавить(Массив);

МаксимумДва = Строка["Процент"];

КоличествоДва = Окр(РезультатАнализа.КоличествоОбъектов\*Строка["Процент"])/100;

Область=Макет.ПолучитьОбласть("Строка");

Область.Параметры.НомерСтроки=НомерСтроки;

Область.Параметры.КолКомб=2;

Область.Параметры.Комбинации=Массив[0]+", "+Массив[1];

Область.Параметры.Частота=КоличествоДва;

ТабДок.Вывести(Область);

НомерСтроки=НомерСтроки+1;

КонецЕсли;

КонецЕсли;

КонецЕсли;

КонецЦикла;

ТабДок.ТолькоПросмотр=Истина;

В данном блоке берутся результаты анализа, и выводятся в табличный документ пары и тройки товаров, без повторений и максимальной частотой.

Функция РазборПравила(Правило)

ИндексСкобочки=0;

Бегун=СтрДлина(Правило)-1;

Пока Не ИндексСкобочки И Бегун>=0 Цикл

Буква = Сред(Правило,Бегун,1);

Если(Буква="(") Тогда

ИндексСкобочки=Бегун;

КонецЕсли;

Бегун = Бегун-1;

КонецЦикла;

ПроцентСлуч = Прав(Правило,СтрДлина(Правило) - ИндексСкобочки);

//ПроцентСлуч = Лев(ПроцентСлуч, СтрДлина(ПроцентСлуч)-1);

ПроцентСлуч = Сред(ПроцентСлуч,Найти(ПроцентСлуч,";")+2);

ПроцентСлуч = Лев(ПроцентСлуч,Найти(ПроцентСлуч,";")-1);

Товары = Лев (Правило,ИндексСкобочки - 1);

Товары = СтрЗаменить(Товары,"Субконто1 = ","");

Товары = СтрЗаменить(Товары," => ",",");

МассТов = РазложитьСтрокуВМассивПодстрок(Товары, ",");

Для бегун = 0 По МассТов.Количество()-1 Цикл

МассТов[бегун] = СокрЛП(МассТов[бегун]);

КонецЦикла;

Результат = новый Соответствие();

Результат.Вставить("Процент",Число(ПроцентСлуч));

Результат.Вставить("Количество",МассТов.Количество());

Результат.Вставить("Массив",МассТов);

Возврат Результат;

КонецФункции

Данная функция преобразует строку с правилом, полученным при анализе данных, в соответствие, с полями Процент, Количество и Массив (в котором содержится список товаров).

Функция РазложитьСтрокуВМассивПодстрок(Знач Стр, Разделитель = ",") Экспорт

МассивСтрок = Новый Массив();

Если Разделитель = " " Тогда

Стр = СокрЛП(Стр);

Пока 1=1 Цикл

Поз = Найти(Стр,Разделитель);

Если Поз=0 Тогда

МассивСтрок.Добавить(Стр);

Возврат МассивСтрок;

КонецЕсли;

МассивСтрок.Добавить(Лев(Стр,Поз-1));

Стр = СокрЛ(Сред(Стр,Поз));

КонецЦикла;

Иначе

ДлинаРазделителя = СтрДлина(Разделитель);

Пока 1=1 Цикл

Поз = Найти(Стр,Разделитель);

Если Поз=0 Тогда

МассивСтрок.Добавить(Стр);

Возврат МассивСтрок;

КонецЕсли;

МассивСтрок.Добавить(Лев(Стр,Поз-1));

Стр = Сред(Стр,Поз+ДлинаРазделителя);

КонецЦикла;

КонецЕсли;

КонецФункции

Данная функция преобразует строку вида «один,два,три,четыре» в массив с элементами «один», «два», «три», «четыре».

Функция ПроверкаПар(Массив,Строка)

Если Массив.Количество()=0 Тогда

Возврат Истина;

КонецЕсли;

Нашлось = Истина;

МассивНовый = Строка;//РазложитьСтрокуВМассивПодстрок(Строка,",");

Для каждого эл из Массив Цикл

МассивЭл = эл;//РазложитьСтрокуВМассивПодстрок(эл,",");

Если МассивЭл.Количество()=МассивНовый.Количество() Тогда

Для каждого ололо из МассивНовый Цикл

Позиция = МассивЭл.Найти(ололо);

Нашлось = Нашлось И (Позиция<>неопределено);

КонецЦикла;

Если Нашлось Тогда

Возврат Ложь;

КонецЕсли;

КонецЕсли;

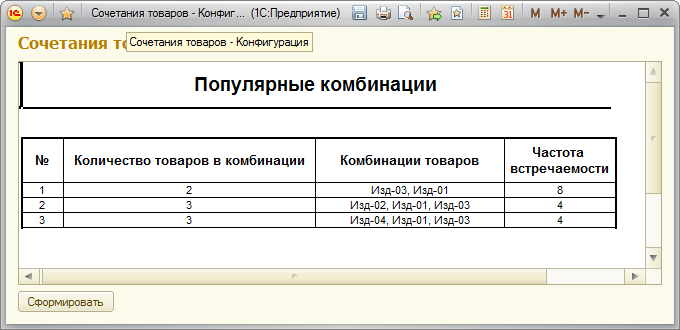
КонецЦикла;

Возврат Истина;

КонецФункции

Данная функция проверяет наличие списка товаров в уже встречающихся парах и тройках.

Сформированный отчет



2002 год 2 тур

Требование

Исходная ситуация. Предприятие "Витязь" торгует рубероидом одной марки. Рубероид поступает в рулонах различной длины. Учет товара ведется на счете 41, на котором предусмотрен аналитический и количественный учет по товарам. Каждый товар имеет следующие обязательные характеристики: Код, наименование, длина рулона, отпускная цена рулона.

Отпуск товара осуществляется по заказам покупателей только целыми рулонами. Количество отпускаемого товара запрашивается покупателем в метрах.

Требуется в среде программы «1С:Бухгалтерия» разработать настройку, обеспечивающую формирование счета покупателю, исходя из заказа покупателя и учетных остатков товаров.

В счет необходимо включить такое количество рулонов, чтобы их суммарная длина была не меньше запрошенной. При этом требуется, чтобы отклонение от заказанной длины было бы минимальным.

В случае если имеющийся запас товара недостаточен для удовлетворения запроса покупателя, то необходимо предусмотреть вывод соответствующего сообщения с указанием учетного остатка.

Пример

По данным учета текущие запасы товара "Рубероид РБ22" составляют:

|  |  |  |
| --- | --- | --- |
| Метраж рулона | Кол-во рулонов | Отпускная цена |
| 27 | 12 | 270 |
| 45 | 11 | 450 |
| 33 | 21 | 330 |
| 60 | 0 | 600 |
| 22 | 17 | 220 |
| 47 | 9 | 470 |

Покупатель ООО "Фабрика грез" затребовал 260 м. товара "Рубероид РБ22".   
На основании данного требования программа сформировала следующий счет.

Счет № от " " 2000 г.

Покупатель: ООО "Фабрика грез"

Заказано: 260 м

К оплате:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование | Метраж рулона | Кол-во рулонов | Метров | Сумма |
| Рубероид РБ22 | 27 | 8 | 216 | 2160 |
| Рубероид РБ22 | 22 | 2 | 44 | 440 |
| Итого |  |  | 260 | 2600 |

Решение

Для решения данной задачи требуется рассмотреть возможные варианты комбинирования рулонов в группы, подходящие под условие задачи. Используем рекурсивную функцию, для обхода всего графа комбинаций. Условием остановки поиска будет равенство длинны рулонов требуемой длине, если же варианта с равенством нет, то выбираем с минимальным отклонением от цели.

Реализация

&НаКлиенте

Процедура Сформировать(Команда)

СформироватьНаСервере();

КонецПроцедуры

&НаСервере

Процедура СформироватьНаСервере()

ЭтотОбъект=РеквизитФормыВЗначение("Отчет");

Макет = ЭтотОбъект.ПолучитьМакет("Макет");

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Рулон,

| ЖурналПроводокОстатки.Субконто2 КАК Длина,

| ЖурналПроводокОстатки.Субконто3 КАК Цена,

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет = &Счет

| И ЖурналПроводокОстатки.Субконто1.Ссылка = &Рулон

|

|УПОРЯДОЧИТЬ ПО

| Длина";

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

Запрос.УстановитьПараметр("Рулон", Рулон);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Остатки = Новый Массив;

Пока Выборка.Следующий() Цикл

Данные = Новый Структура;

Данные.Вставить("Длина", Выборка.Длина);

Данные.Вставить("Цена", Выборка.Цена);

Данные.Вставить("Количество", Выборка.Количество);

Остатки.Добавить(Данные);

КонецЦикла;

Результат = НайтиЛучшийПуть(Остатки, Неопределено, Длина);

ТабДок.Очистить();

Рулоны = Новый Массив;

Для Каждого Элемент Из Результат.Путь Цикл

Найден = Ложь;

Для Каждого РулонЭлемент Из Рулоны Цикл

Если РулонЭлемент.Длина = Элемент.Длина Тогда

РулонЭлемент.Количество = РулонЭлемент.Количество + 1;

РулонЭлемент.Сумма = РулонЭлемент.Сумма + Элемент.Цена;

Найден = Истина;

Прервать;

КонецЕсли;

КонецЦикла;

Если Найден = Ложь Тогда

РулонЭлемент = Новый Структура;

РулонЭлемент.Вставить("Длина", Элемент.Длина);

РулонЭлемент.Вставить("Количество", 1);

РулонЭлемент.Вставить("Сумма", Элемент.Цена);

Рулоны.Добавить(РулонЭлемент);

КонецЕсли;

КонецЦикла;

ЗаголовокОбласть = Макет.ПолучитьОбласть("Заголовок");

ДанныеОбласть = Макет.ПолучитьОбласть("Данные");

ИтогОбласть = Макет.ПолучитьОбласть("Итог");

ТабДок.Вывести(ЗаголовокОбласть);

Метров = 0;

Сумма = 0;

Для Каждого РулонЭлемент Из Рулоны Цикл

ДанныеОбласть.Параметры.Рулон = Рулон;

ДанныеОбласть.Параметры.Метраж = РулонЭлемент.Длина;

ДанныеОбласть.Параметры.Количество = РулонЭлемент.Количество;

ДанныеОбласть.Параметры.Метров = РулонЭлемент.Длина \* РулонЭлемент.Количество;

ДанныеОбласть.Параметры.Сумма = РулонЭлемент.Сумма;

ТабДок.Вывести(ДанныеОбласть);

Метров = Метров + ДанныеОбласть.Параметры.Метров;

Сумма = Сумма + ДанныеОбласть.Параметры.Сумма;

КонецЦикла;

ИтогОбласть.Параметры.Метров = Метров;

ИтогОбласть.Параметры.Сумма = Сумма;

ТабДок.Вывести(ИтогОбласть);

КонецПроцедуры

&НаСервере

Функция НайтиЛучшийПуть(Остатки, ПрошлыйРезультат, Цель)

Если ПрошлыйРезультат = Неопределено Тогда

ПрошлыйРезультат = Новый Структура;

ПрошлыйРезультат.Вставить("Длина", 0);

ПрошлыйРезультат.Вставить("Путь", Новый Массив);

КонецЕсли;

ЛучшийПуть = Неопределено;

ЛучшийЭлемент = Неопределено;

ЛучшаяДлина = 0;

Для Каждого Остаток Из Остатки Цикл

Если Остаток.Количество = 0 Тогда

Продолжить;

КонецЕсли;

НоваяДлина = ПрошлыйРезультат.Длина + Остаток.Длина;

Если НоваяДлина < Цель Тогда

Остаток.Количество = Остаток.Количество - 1;

ПрошлыйРезультат.Длина = НоваяДлина;

ПрошлыйРезультат.Путь.Добавить(Остаток);

НовыйРезультат = НайтиЛучшийПуть(Остатки, ПрошлыйРезультат, Цель);

ПрошлыйРезультат.Путь.Удалить(ПрошлыйРезультат.Путь.ВГраница());

ПрошлыйРезультат.Длина = ПрошлыйРезультат.Длина - Остаток.Длина;

Остаток.Количество = Остаток.Количество + 1;

Если НовыйРезультат = Неопределено Тогда

Продолжить;

КонецЕсли;

Если НовыйРезультат.Длина >= Цель Тогда

Если ЛучшаяДлина = 0 ИЛИ (ЛучшаяДлина - Цель) > (НовыйРезультат.Длина - Цель) Тогда

ЛучшаяДлина = НовыйРезультат.Длина;

ЛучшийПуть = НовыйРезультат.Путь;

ЛучшийЭлемент = Неопределено;

КонецЕсли;

КонецЕсли;

Иначе

Если ЛучшаяДлина = 0 ИЛИ (ЛучшаяДлина - Цель) > (НоваяДлина - Цель) Тогда

ЛучшаяДлина = НоваяДлина;

ЛучшийЭлемент = Остаток;

ЛучшийПуть = Неопределено;

КонецЕсли;

КонецЕсли;

Если ЛучшаяДлина = Цель Тогда

Прервать;

КонецЕсли;

КонецЦикла;

Если ЛучшийЭлемент <> Неопределено Тогда

ЛучшийПуть = Новый Массив;

Для Каждого Прошлый Из ПрошлыйРезультат.Путь Цикл

ЛучшийПуть.Добавить(Прошлый);

КонецЦикла;

ЛучшийПуть.Добавить(ЛучшийЭлемент);

КонецЕсли;

Если ЛучшийПуть = Неопределено Тогда

Возврат Неопределено;

КонецЕсли;

Результат = Новый Структура;

Результат.Вставить("Длина", ЛучшаяДлина);

Результат.Вставить("Путь", ЛучшийПуть);

Возврат Результат;

КонецФункции

Автор: Михайлов Алексей

Пояснения к коду:

&НаСервере

Процедура СформироватьНаСервере()

ЭтотОбъект=РеквизитФормыВЗначение("Отчет");

Макет = ЭтотОбъект.ПолучитьМакет("Макет");

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Рулон,

| ЖурналПроводокОстатки.Субконто2 КАК Длина,

| ЖурналПроводокОстатки.Субконто3 КАК Цена,

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет = &Счет

| И ЖурналПроводокОстатки.Субконто1.Ссылка = &Рулон

|

|УПОРЯДОЧИТЬ ПО

| Длина";

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

Запрос.УстановитьПараметр("Рулон", Рулон);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Остатки = Новый Массив;

Пока Выборка.Следующий() Цикл

Данные = Новый Структура;

Данные.Вставить("Длина", Выборка.Длина);

Данные.Вставить("Цена", Выборка.Цена);

Данные.Вставить("Количество", Выборка.Количество);

Остатки.Добавить(Данные);

КонецЦикла;

Результат = НайтиЛучшийПуть(Остатки, Неопределено, Длина);

ТабДок.Очистить();

Рулоны = Новый Массив;

Для Каждого Элемент Из Результат.Путь Цикл

Найден = Ложь;

Для Каждого РулонЭлемент Из Рулоны Цикл

Если РулонЭлемент.Длина = Элемент.Длина Тогда

РулонЭлемент.Количество = РулонЭлемент.Количество + 1;

РулонЭлемент.Сумма = РулонЭлемент.Сумма + Элемент.Цена;

Найден = Истина;

Прервать;

КонецЕсли;

КонецЦикла;

Если Найден = Ложь Тогда

РулонЭлемент = Новый Структура;

РулонЭлемент.Вставить("Длина", Элемент.Длина);

РулонЭлемент.Вставить("Количество", 1);

РулонЭлемент.Вставить("Сумма", Элемент.Цена);

Рулоны.Добавить(РулонЭлемент);

КонецЕсли;

КонецЦикла;

ЗаголовокОбласть = Макет.ПолучитьОбласть("Заголовок");

ДанныеОбласть = Макет.ПолучитьОбласть("Данные");

ИтогОбласть = Макет.ПолучитьОбласть("Итог");

ТабДок.Вывести(ЗаголовокОбласть);

Метров = 0;

Сумма = 0;

Для Каждого РулонЭлемент Из Рулоны Цикл

ДанныеОбласть.Параметры.Рулон = Рулон;

ДанныеОбласть.Параметры.Метраж = РулонЭлемент.Длина;

ДанныеОбласть.Параметры.Количество = РулонЭлемент.Количество;

ДанныеОбласть.Параметры.Метров = РулонЭлемент.Длина \* РулонЭлемент.Количество;

ДанныеОбласть.Параметры.Сумма = РулонЭлемент.Сумма;

ТабДок.Вывести(ДанныеОбласть);

Метров = Метров + ДанныеОбласть.Параметры.Метров;

Сумма = Сумма + ДанныеОбласть.Параметры.Сумма;

КонецЦикла;

ИтогОбласть.Параметры.Метров = Метров;

ИтогОбласть.Параметры.Сумма = Сумма;

ТабДок.Вывести(ИтогОбласть);

КонецПроцедуры

Данная процедура – обработчик нажатия кнопки Сформировать на сервере, в ней происходит получение данных об остатках рулонов в базе, составление массива остатков, вызов рекурсивной функции поиска пути, подсчет количества рулонов с группировкой по длине и вывод результата в табличный документ.

ЭтотОбъект=РеквизитФормыВЗначение("Отчет");

Макет = ЭтотОбъект.ПолучитьМакет("Макет");

В данном блоке производится получение макета, так как отчет – внешний, то чтобы получить его макет требуется перевести реквизит формы Отчет в объект и получить его макет.

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЖурналПроводокОстатки.Субконто1 КАК Рулон,

| ЖурналПроводокОстатки.Субконто2 КАК Длина,

| ЖурналПроводокОстатки.Субконто3 КАК Цена,

| ЖурналПроводокОстатки.КоличествоОстаток КАК Количество

|ИЗ

| РегистрБухгалтерии.ЖурналПроводок.Остатки КАК ЖурналПроводокОстатки

|ГДЕ

| ЖурналПроводокОстатки.Счет = &Счет

| И ЖурналПроводокОстатки.Субконто1.Ссылка = &Рулон

|

|УПОРЯДОЧИТЬ ПО

| Длина";

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

Запрос.УстановитьПараметр("Рулон", Рулон);

Результат = Запрос.Выполнить();

Выборка = Результат.Выбрать();

Остатки = Новый Массив;

Пока Выборка.Следующий() Цикл

Данные = Новый Структура;

Данные.Вставить("Длина", Выборка.Длина);

Данные.Вставить("Цена", Выборка.Цена);

Данные.Вставить("Количество", Выборка.Количество);

Остатки.Добавить(Данные);

КонецЦикла;

В данном блоке производится получение остатков указанных рулонов на счете Товары и запись результатов в массив Остатки, записи которого являются структурами с полями Длина, Цена, Количество.

Результат = НайтиЛучшийПуть(Остатки, Неопределено, Длина);

Данная строка вызывает рекурсивную функцию поиска пути, первый параметр – начальные остатки рулонов, второй параметр – предыдущие выбранные рулоны (на начало поиска их нет, поэтому Неопределено) и третий параметр – требуемая длина. Возвращает функция структуру с полями Длина и Путь (массив из структур аналогичный массиву Остатки).

Рулоны = Новый Массив;

Для Каждого Элемент Из Результат.Путь Цикл

Найден = Ложь;

Для Каждого РулонЭлемент Из Рулоны Цикл

Если РулонЭлемент.Длина = Элемент.Длина Тогда

РулонЭлемент.Количество = РулонЭлемент.Количество + 1;

РулонЭлемент.Сумма = РулонЭлемент.Сумма + Элемент.Цена;

Найден = Истина;

Прервать;

КонецЕсли;

КонецЦикла;

Если Найден = Ложь Тогда

РулонЭлемент = Новый Структура;

РулонЭлемент.Вставить("Длина", Элемент.Длина);

РулонЭлемент.Вставить("Количество", 1);

РулонЭлемент.Вставить("Сумма", Элемент.Цена);

Рулоны.Добавить(РулонЭлемент);

КонецЕсли;

КонецЦикла;

В данном блоке полученный результат разбивается на группы рулонов, по полю Длина. Для каждой записи из Результат.Путь ищется в массиве Рулоны элемент, с такой же длиной, если элемент найден то у него добавляется количество и меняется сумма, если не найден то элемент создается и добавляется в массив Рулоны.

ЗаголовокОбласть = Макет.ПолучитьОбласть("Заголовок");

ДанныеОбласть = Макет.ПолучитьОбласть("Данные");

ИтогОбласть = Макет.ПолучитьОбласть("Итог");

ТабДок.Вывести(ЗаголовокОбласть);

Метров = 0;

Сумма = 0;

Для Каждого РулонЭлемент Из Рулоны Цикл

ДанныеОбласть.Параметры.Рулон = Рулон;

ДанныеОбласть.Параметры.Метраж = РулонЭлемент.Длина;

ДанныеОбласть.Параметры.Количество = РулонЭлемент.Количество;

ДанныеОбласть.Параметры.Метров = РулонЭлемент.Длина \* РулонЭлемент.Количество;

ДанныеОбласть.Параметры.Сумма = РулонЭлемент.Сумма;

ТабДок.Вывести(ДанныеОбласть);

Метров = Метров + ДанныеОбласть.Параметры.Метров;

Сумма = Сумма + ДанныеОбласть.Параметры.Сумма;

КонецЦикла;

ИтогОбласть.Параметры.Метров = Метров;

ИтогОбласть.Параметры.Сумма = Сумма;

ТабДок.Вывести(ИтогОбласть);

В данном блоке производится получение областей макета, вывод заголовка, вывод каждого элемента массива Рулоны с суммированием метров и суммы, и вывод итогов.

&НаСервере

Функция НайтиЛучшийПуть(Остатки, ПрошлыйРезультат, Цель)

Если ПрошлыйРезультат = Неопределено Тогда

ПрошлыйРезультат = Новый Структура;

ПрошлыйРезультат.Вставить("Длина", 0);

ПрошлыйРезультат.Вставить("Путь", Новый Массив);

КонецЕсли;

ЛучшийПуть = Неопределено;

ЛучшийЭлемент = Неопределено;

ЛучшаяДлина = 0;

Для Каждого Остаток Из Остатки Цикл

Если Остаток.Количество = 0 Тогда

Продолжить;

КонецЕсли;

НоваяДлина = ПрошлыйРезультат.Длина + Остаток.Длина;

Если НоваяДлина < Цель Тогда

Остаток.Количество = Остаток.Количество - 1;

ПрошлыйРезультат.Длина = НоваяДлина;

ПрошлыйРезультат.Путь.Добавить(Остаток);

НовыйРезультат = НайтиЛучшийПуть(Остатки, ПрошлыйРезультат, Цель);

ПрошлыйРезультат.Путь.Удалить(ПрошлыйРезультат.Путь.ВГраница());

ПрошлыйРезультат.Длина = ПрошлыйРезультат.Длина - Остаток.Длина;

Остаток.Количество = Остаток.Количество + 1;

Если НовыйРезультат = Неопределено Тогда

Продолжить;

КонецЕсли;

Если НовыйРезультат.Длина >= Цель Тогда

Если ЛучшаяДлина = 0 ИЛИ (ЛучшаяДлина - Цель) > (НовыйРезультат.Длина - Цель) Тогда

ЛучшаяДлина = НовыйРезультат.Длина;

ЛучшийПуть = НовыйРезультат.Путь;

ЛучшийЭлемент = Неопределено;

КонецЕсли;

КонецЕсли;

Иначе

Если ЛучшаяДлина = 0 ИЛИ (ЛучшаяДлина - Цель) > (НоваяДлина - Цель) Тогда

ЛучшаяДлина = НоваяДлина;

ЛучшийЭлемент = Остаток;

ЛучшийПуть = Неопределено;

КонецЕсли;

КонецЕсли;

Если ЛучшаяДлина = Цель Тогда

Прервать;

КонецЕсли;

КонецЦикла;

Если ЛучшийЭлемент <> Неопределено Тогда

ЛучшийПуть = Новый Массив;

Для Каждого Прошлый Из ПрошлыйРезультат.Путь Цикл

ЛучшийПуть.Добавить(Прошлый);

КонецЦикла;

ЛучшийПуть.Добавить(ЛучшийЭлемент);

КонецЕсли;

Если ЛучшийПуть = Неопределено Тогда

Возврат Неопределено;

КонецЕсли;

Результат = Новый Структура;

Результат.Вставить("Длина", ЛучшаяДлина);

Результат.Вставить("Путь", ЛучшийПуть);

Возврат Результат;

КонецФункции

Данная функция рекурсивна, производит поиск такой комбинации остатков рулонов, которая в длине будет не меньше указанной. Для начала проверяется есть ли предыдущий результат, то есть вызвана ли функция рекурсивно. Если предыдущий результат не определен, то структура создается, с начальными данными (путь пуст и длина 0). Далее происходит проверка каждого рулона из остатков, количество которого не нулевое. Проверяется следующим образом – длина рулона прибавляется к начальной (предыдущий результат), если новая длина меньше требуемой, то рекурсивно вызываем функцию, предварительно уменьшив количество выбранных рулонов и указав нужный предыдущий результат, после получения рекурсивного результата отменяем изменения, чтобы продолжить поиск по другим веткам графа. Если длина по данной ветке больше либо равна целевой и лучше, чем предыдущий сохраненный результат или такого нет, то запоминаем данный выбор. Если же новая длина больше либо равна целевой, то опять же сверяемся с предыдущим сохраненным результатом. Далее проверяем насколько точно подходит результат, если длина точно равна целевой – завершаем поиск. После обхода всех рулонов, составляется структура результата с полями Длина и Путь.

Сформированный отчет

