Занятие 1.01

Логическая структура:

Физическая структура:

1. Режим Файл-сервера:

ИБ (данные и конфигурация) хранятся в отдельном файле (1cv8.1cd) на компьютере клиента или в сети. С ним работает платформа, располагающаяся на клиентских машинах. Обращение происходит напрямую.

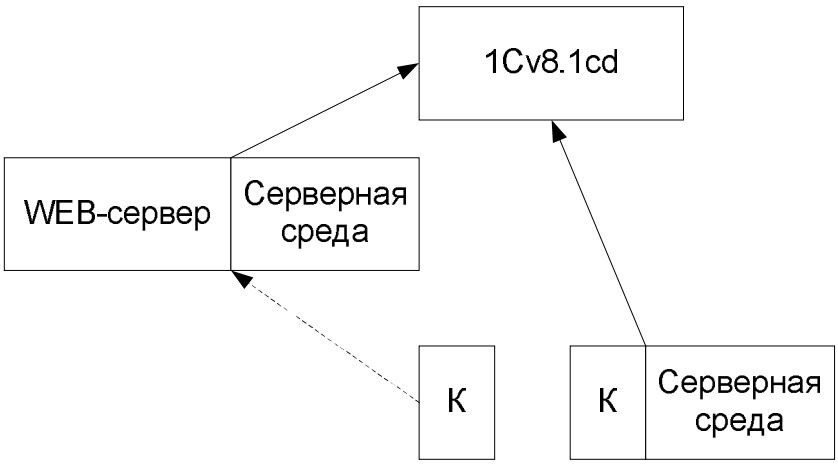
«+»

Дешево.

«-»

Если надо организовать доступ к БД с разных машин в сети, то надо дать доступ на чтение и запись к файлу ИБ от имени клиентских компьютеров.

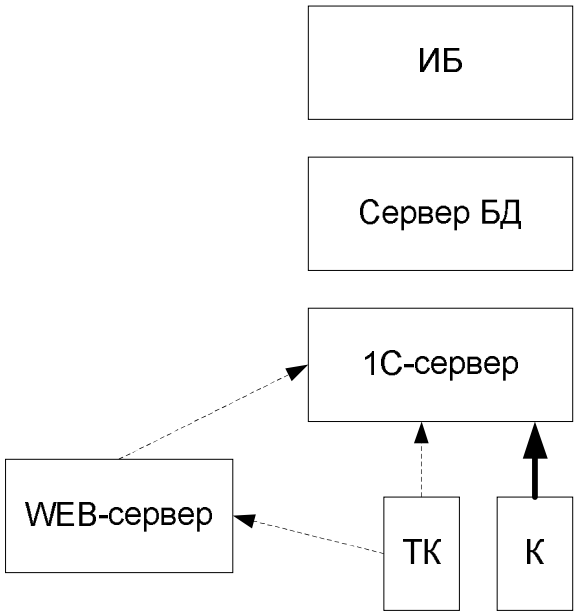
Все серверные механизмы (блокировки чтения, записи, обслуживание БД) исполняются в серверной среде. В данном режиме серверная среда может выполняться на компьютере клиента, что увеличит нагрузку на него. Но можно реализовать и удаленную работу с БД, т.е. немного защитить от проникновений извне, установив WEB-сервер (IIS или Apache), в котором будет исполняться серверная среда. Тогда WEB-сервер будет работать с файлом 1cv8.1cd, а клиент будет соединяться с WEB-сервером. При этом возможно нестабильное соединение клиента с WEB-сервером (сеанс).



2. Режим Клиент-сервера

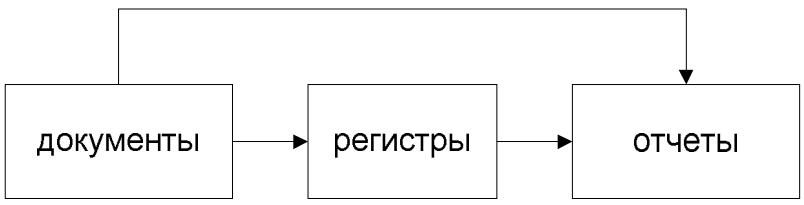
ИБ работает под управлением сервера БД (например, MS SQL Server). С сервером БД работает 1С-сервер, с которым, в свою очередь, будут работать клиентские машины либо WEB-сервер. При таком режиме вся нагрузка будет лежать на 1С-сервере.
Клиенты делают запрос на предоставление определенной информации, 1С-сервер преобразует запросы в вид, понятный серверу БД, сервер БД опрашивает БД и возвращает 1С-серверу ответ, 1С-сервер производит необходимые обработки и возвращает клиенту выборку данных или ее часть.

Клиенты разделяются на виды:
Обычный – из старых версий 1С 8, более требователен к ресурсам, выкачивает с 1С-сервера выборку данных, работает с данными напрямую на стороне клиента, может обращаться непосредственно к данным конфигурации. Поэтому соединение клиента с 1С-сервером или WEB-сервером может быть нестабильным (сеансовым).
Тонкий – показывает картинку, все данные находятся на стороне 1С-сервера и хранятся в сеансе. Возможно нестабильное соединение.



Конфигуратор можно запустить только в режиме толстого клиента.

Прикладные классы объектов системы:



На основе документов можно сформировать отчеты путем перебора документов, но это будет очень долго и сильно упадет производительность.

Допустим, есть отчет по продажам, составленный на основе документов «Продажа». Вводится новый документ «Продажа в розницу», который также надо анализировать в отчете, для этого надо будет переписывать отчет.
Чтобы избежать этого, используются «Регистры» - объекты, хранящие показатели учета. Документы записывают в них данные, а отчеты анализируют эти данные. Регистры также содержат служебные данные, накапливающие информацию для максимально быстрого ее последующего извлечения (например, промежуточные обороты на начало/конец месяца).

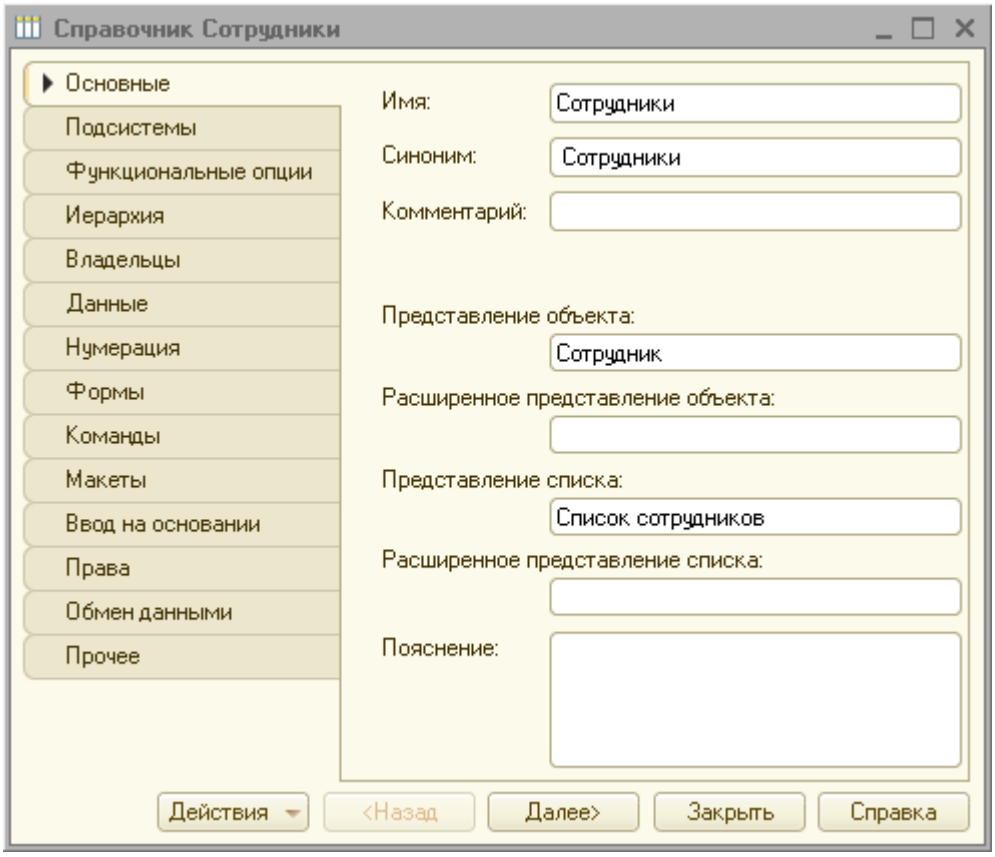
При запуске программы:
если БД создана в режиме файл-сервера, то внизу:
File = «Путь до папки БД»
если БД создана в режиме клиент-сервера, то внизу:
Srvr = «Путь к серверу»; Ref = «Имя ИБ на сервере»

Скорость соединения – указание работы клиента при взаимодействии с 1С-сервером:
- Обычная – система передает больше данных и чаще опрашивает 1С-сервер;
- Низкая – система убирает лишнюю информацию (картинки, плавный скроллинг и т.д.), сервер опрашивается реже;
- Выбирать при запуске – скорость автоматически выбирается системой.

Основной режим запуска – имеется ввиду запуск среды исполнения. В создаваемой БД выберем «Тонкий клиент».

Версия 1С: Предприятия – версия (релиз), с которой надо запускать систему.

В конфигураторе:
«Конфигурация – Открыть конфигурацию» - создается копия конфигурации ИБ (конфигурация разработчика).
«Конфигурация – Обновить конфигурацию БД» - поместить все изменения из конфигурации разработчика в конфигурацию ИБ

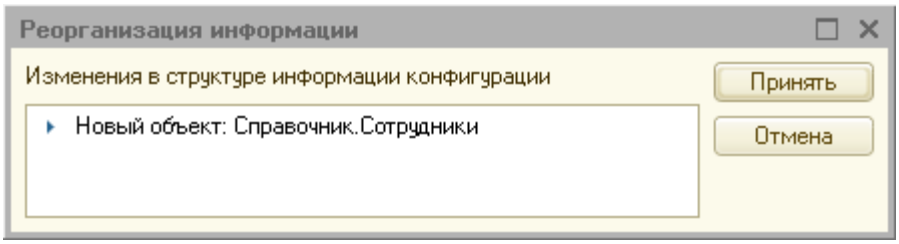


На закладке «Данные» добавим реквизит «Дата рождения»:
Имя – ДатаРождения.
Синоним – Дата рождения.
Тип – Дата.

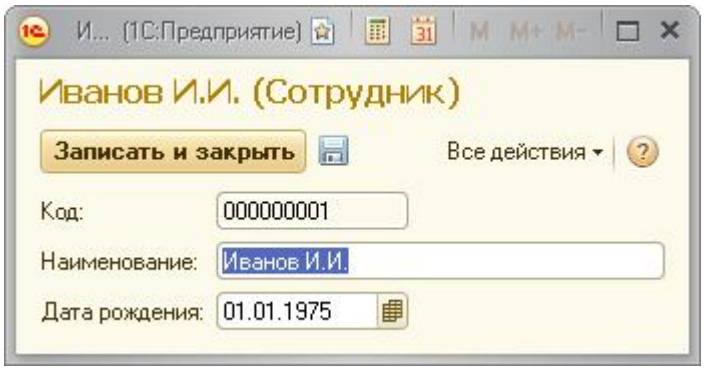
«Длина кода» и «Длина наименования» - стандартные реквизиты справочника.

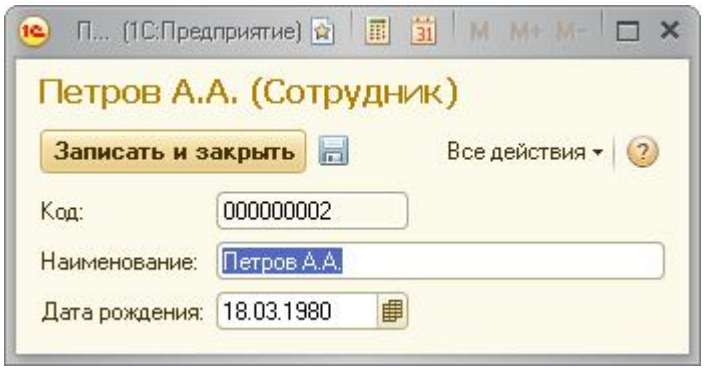
Заполнения созданного объекта данными выполняется в режиме исполнения.

«Отладка – Начать отладку» - запуск 1С в режиме исполнения с отладочным механизмом.
Если изменения задели структуру БД, то 1С выдаст предупреждение об этом:

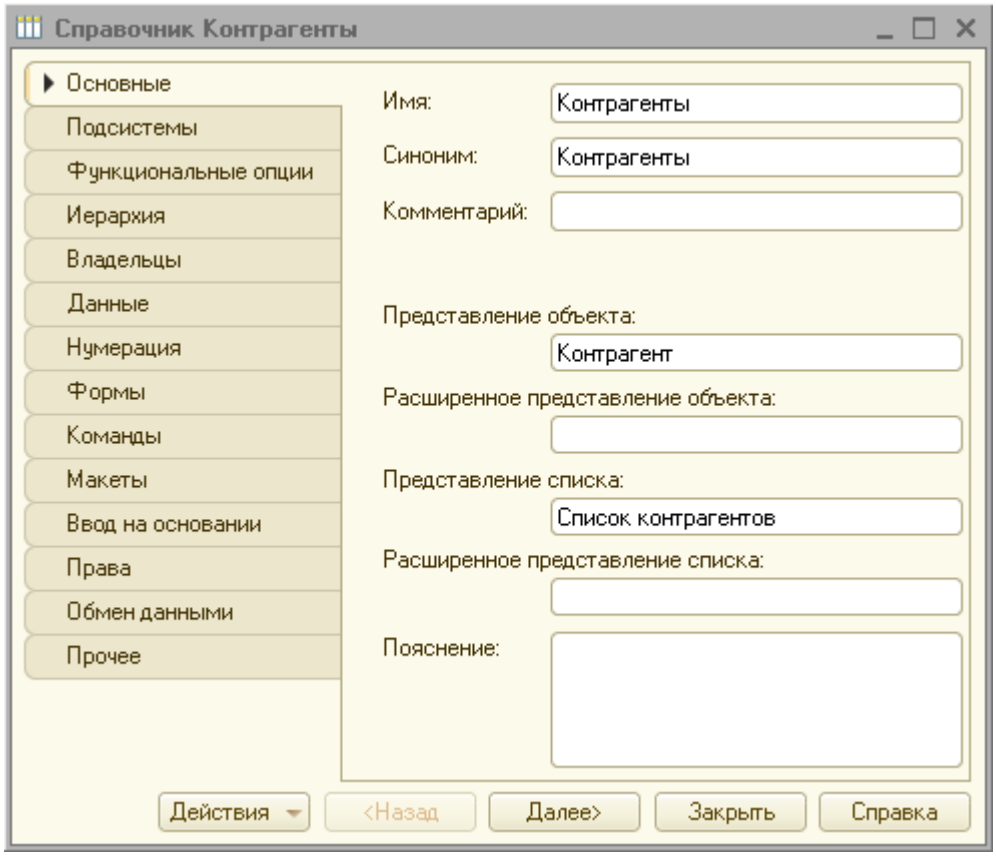


Создадим сотрудников:

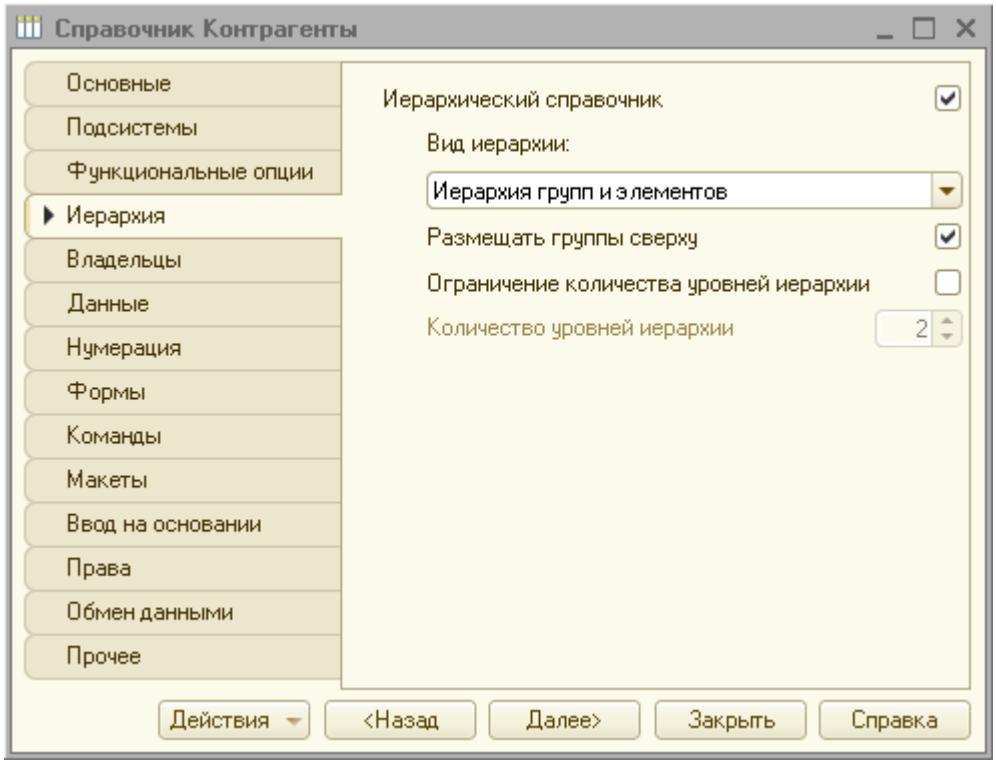




Создадим справочник «Контрагенты»:
Имя – Контрагенты.
Синоним – Контрагенты.
Представление объекта – Контрагент.
Представление списка – Список контрагентов.

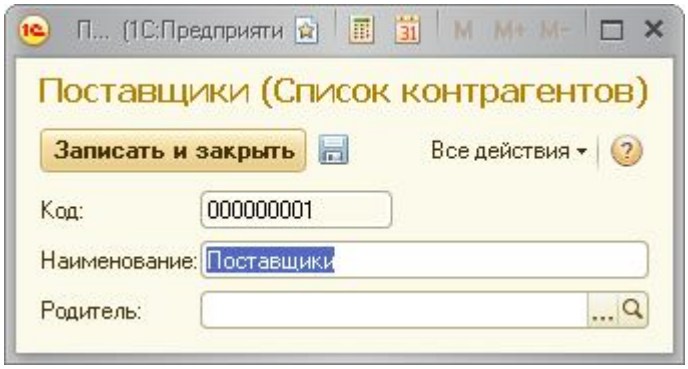


Сделаем этот справочник иерархическим:

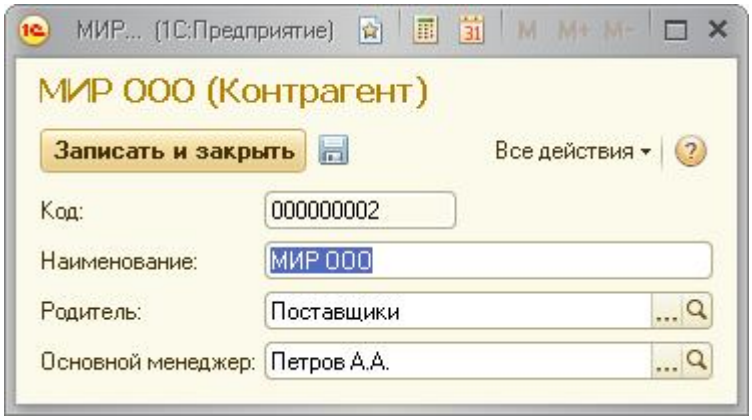


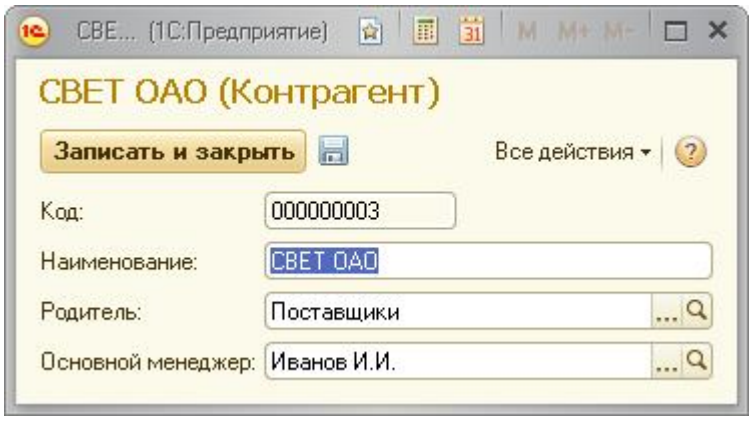
Укажем менеджера по работе с контрагентом – создадим реквизит «Основной менеджер»
Имя – ОсновнойМенеджер.
Синоним – Основной менеджер.
Тип - СправочникСсылка.Сотрудники.

В режиме исполнения откроем список контрагентов и создадим группу «Поставщики»:



а внутри нее создадим поставщика:

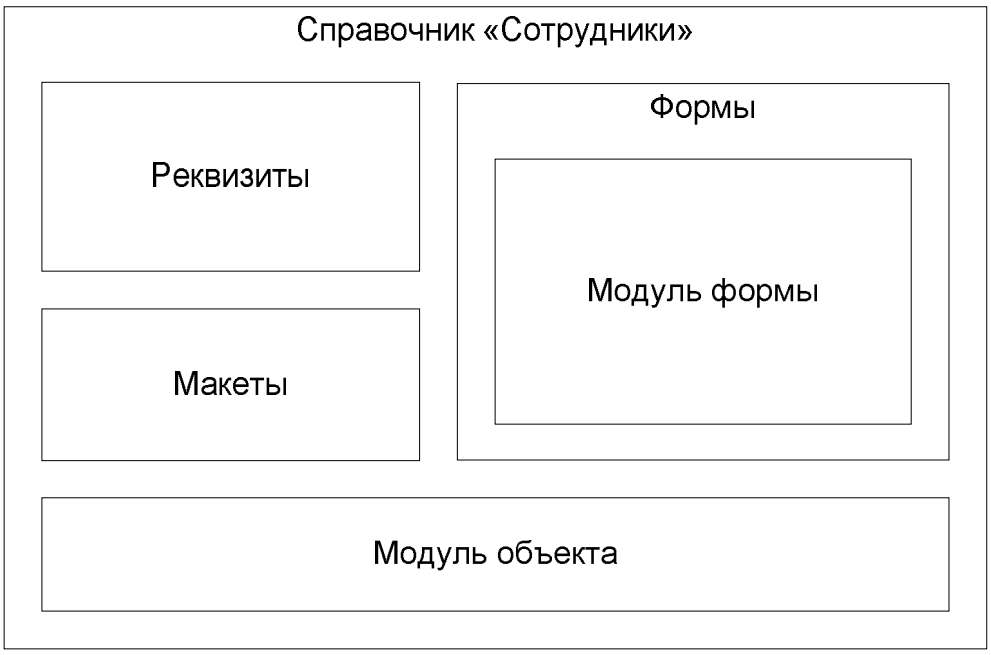




В поле «Основной менеджер» находится ссылка на элемент справочника «Сотрудники»

Занятие 1.02

Представление объекта на примере справочника «Сотрудники»:
Реквизиты – на основе настроек реквизитов система определяет их представление в формах. Влияют на структуру таблиц БД.
Формы – объект для интерактивной работы с объектами, не имеющими интерфейса. Создается вручную только тогда, когда не устраивает стандартное поведение системы при автоматическом создании форм.
Модуль формы – для описания **событий**, связанных с формой, поведения формы.
Можно наделить форму определенным поведением по умолчанию, если связать ее с прикладным объектом (справочником, документом и т.д.)
Также можно создать независимую форму, тогда все ее события придется описывать самостоятельно.
Модуль объекта – для описания событий, связанных с самим объектом (запись объекта, автоматическое копирование объекта и т.д.)
Макеты – для вывода данных.

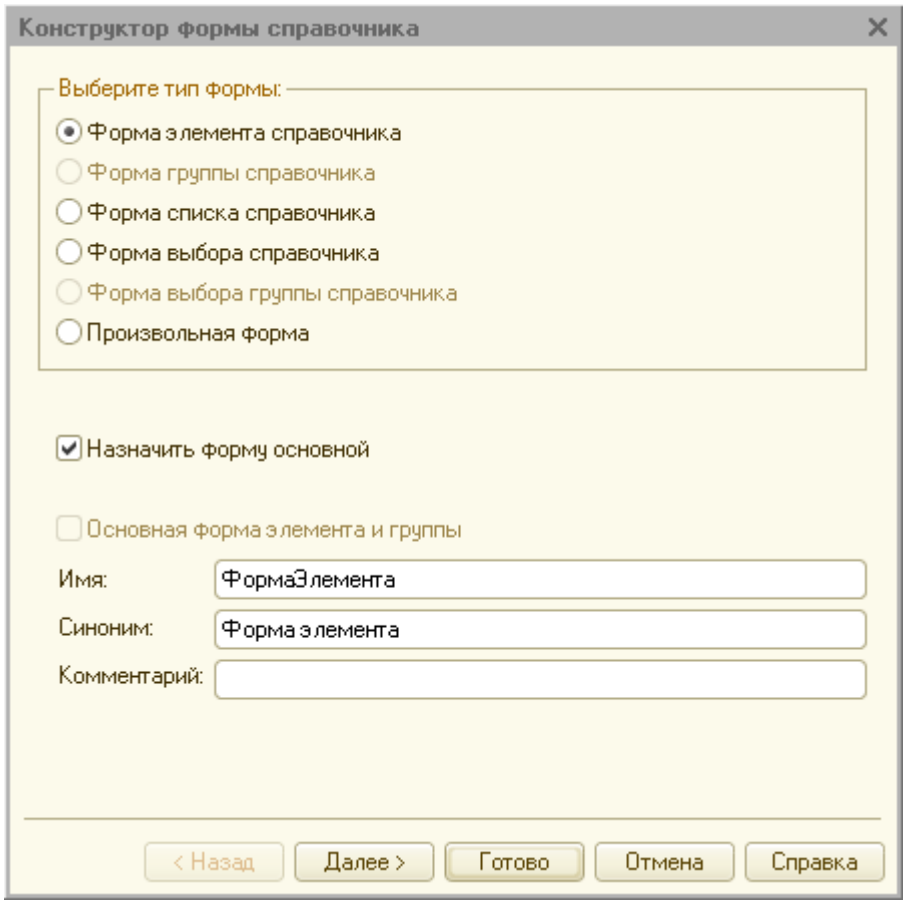


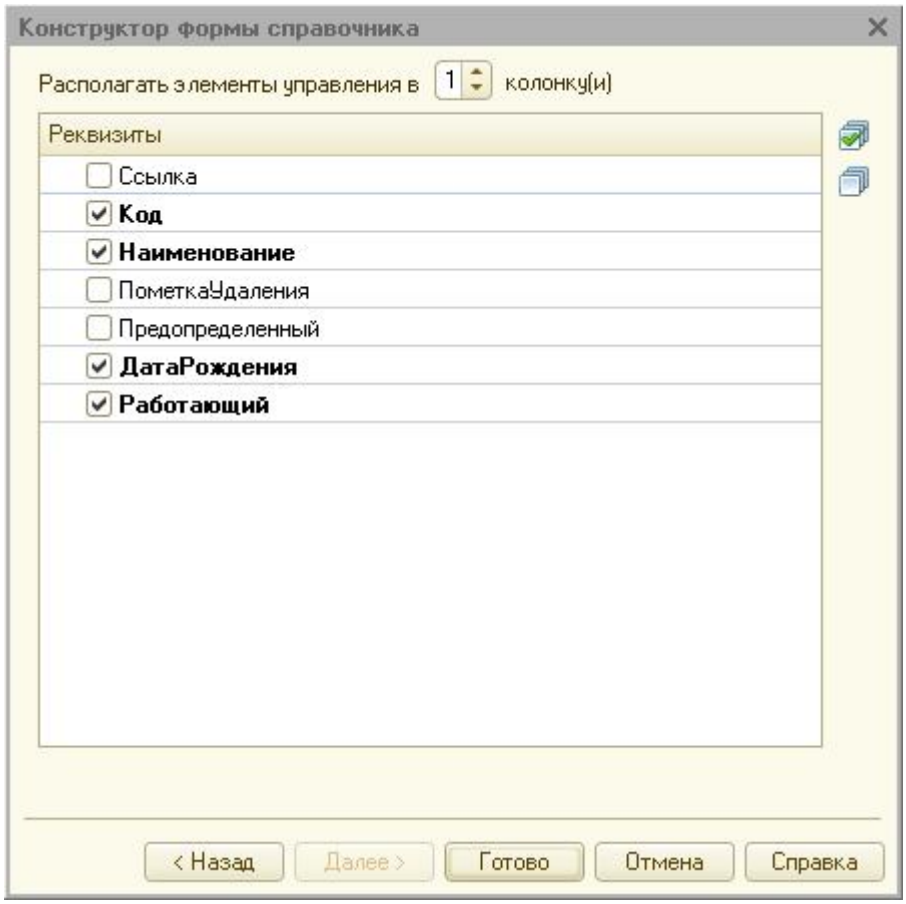
В справочник «Сотрудники» добавим признак «Работающий/Уволенный». В зависимости от настройки этого реквизита будем описывать проверку заполнения даты рождения.

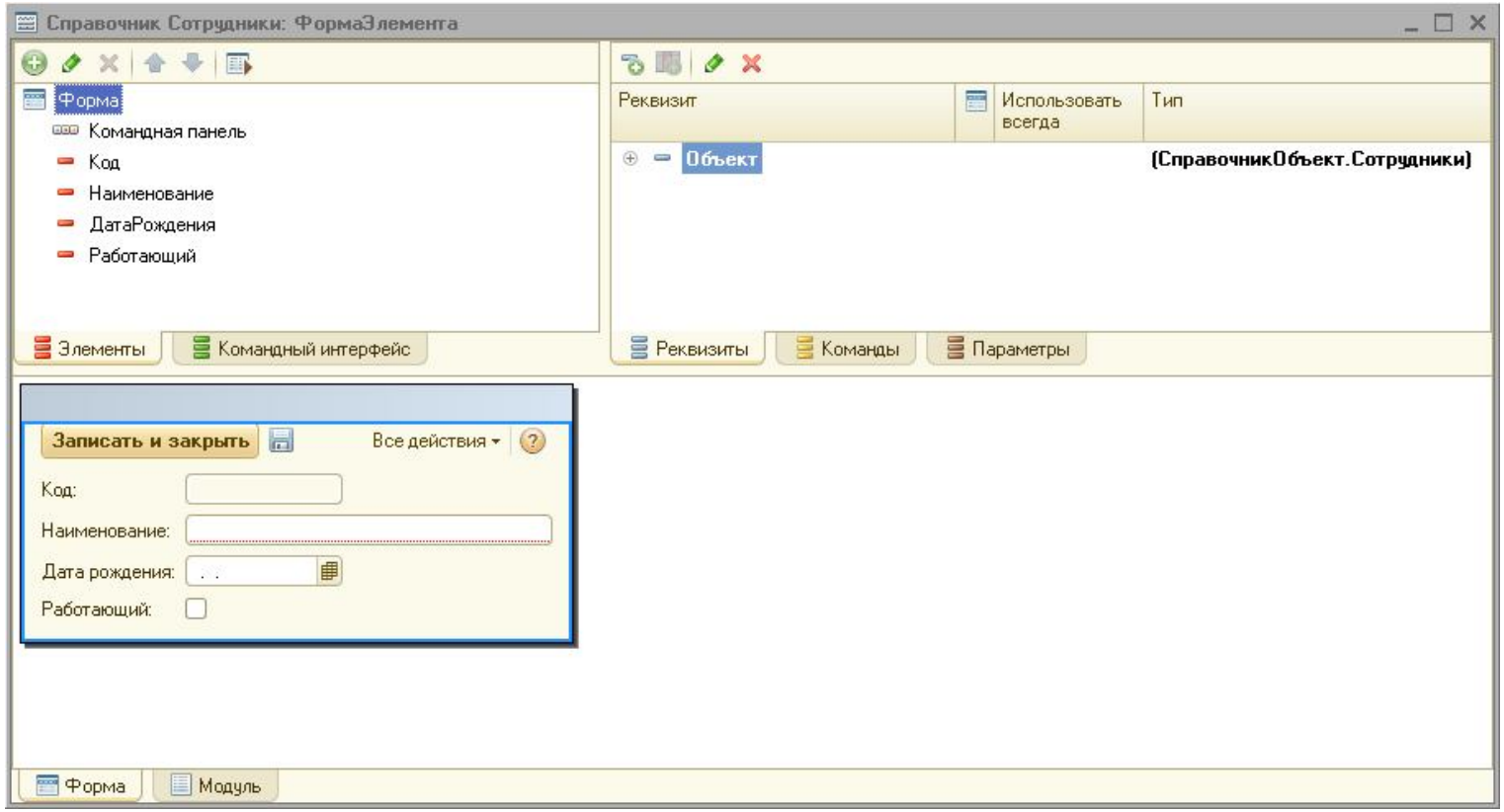
Добавим реквизит «Работающий»:
Имя – Работающий.
Синоним – Работающий.
Тип – Булево.

Чтобы проверка заполнения реквизита «Дата рождения» в зависимости от признака «Работающий» выполнялась, необходимо создать форму элемента справочника самостоятельно.

При создании форм элемента, списка, выбора и т.д., только одна из них будет являться основной.
Создадим форму элемента справочника «Сотрудники» с помощью конструктора:







В нижней части расположена не сама форма, а ее рисунок, как форма будет выглядеть у пользователя. Вверху расположены настройки формы. Вверху слева находится командная панель и реквизиты формы. Реквизиты берутся из настроек формы объекта (вверху справа). Добавить реквизиты на форму можно путем перетаскивания.

Один из реквизитов (объектов) формы может являться «Основным» (помечен жирным). Его тип определяет поведение формы по умолчанию.

Можно добавить собственные реквизиты, которые не будут записываться в БД.

Если надо описать собственные команды, т.е. создать дополнительные кнопки на форме, то на правой верхней панели надо перейти на закладку «Команды». При создании команды, в модуле формы записывается определенная процедура (функция), которая будет выполняться.

Если надо перехватить какое-либо системное событие формы, то на левой верхней панели двойным щелчком или через меню ПКМ «Формы» надо открыть свойства формы и там выбрать событие, которое надо перехватить.

В примере это будет событие «ПередЗаписью» - отрабатывает на форме со стороны клиента:

```
&НаКлиенте
Процедура ПередЗаписью(Отказ, ПараметрыЗаписи)
     // Вставить содержимое обработчика.
КонецПроцедуры
```

«&НаКлиенте» - директива процессору компиляции для обозначения места выполнения процедуры. Процедуры в форме могут исполняться либо «На клиенте», либо «На сервере».
«На клиенте» - пользователь работает только с формой, не может обратиться к структуре БД (построить запрос и т.д.). К структуре БД можно обратиться только через вызов другой процедуры, которая, в свою очередь, выполняется «На сервере»

Процедуру можно вызвать в любом месте модуля (перед или после ее описания), написав ее название.

Количество принимаемых системой параметров должно быть равно количеству параметров, передаваемых при вызове процедуры.

Синтаксис:

```
Процедура ИмяПроцедуры([Параметр1, Параметр2])

КонецПроцедуры
```

В примере система вызывает процедуру самостоятельно и передаваемые параметры нужны для управления событиями, связанными с записью:
Отказ – флаг отказа от записи в систему.

Операторы друг от друга отделяются символом «**;**».
Например, Оператор условия:

```
Если УСЛОВИЕ Тогда
     // Истина
Иначе
     // Ложь
КонецЕсли;
```

Оператор объявления переменной:

```
Перем ИмяПеременной;
```

Дата в 1С записывается в формате:

```
Дата = 'ГодМесяцЧислоЧасыМинутыСекунды'
Сегодня = '20120709033759'
```

Для повышения читабельности в дату можно добавить незначащие знаки (любые знаки, не являющиеся числом):

```
Дата = 'Год-Месяц-Число Часы:Минуты:Секунды'
Сегодня = '2012-07-09 03:37:59'
```

Если при описании даты не указывать ЧасыМинутыСекунды, то это всегда будет 00 часов, 00 минут, 00 секунд.

```
Сегодня = '2012-07-09' // 00:00:00
```

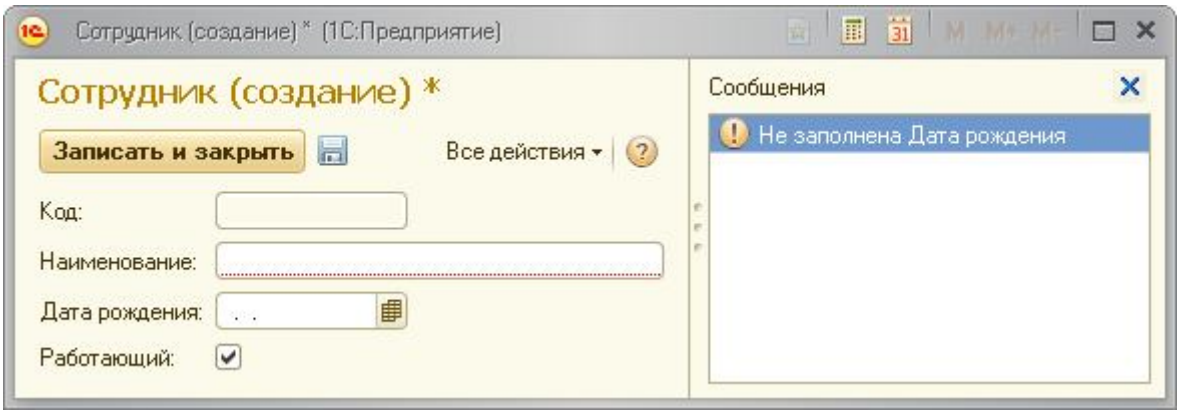
Пустая (незаполненная дата) – первое число первого месяца первого года:

```
ПустаяДата = '00010101'
```

Процедура примера:

```
&НаКлиенте
Процедура ПередЗаписью(Отказ, ПараметрыЗаписи)

    Если Объект.Работающий И Объект.ДатаРождения = '00010101' Тогда
        Сообщить("Не заполнена Дата рождения");
        Отказ = Истина;
    КонецЕсли;

КонецПроцедуры
```

Теперь, если стоит галочка «Работающий» и поле «Дата рождения» не заполнена, то будет выдаваться ошибка:



Если дату заполнить, то сотрудник запишется. Если признак «Работающий» снят и поле «Дата рождения» не заполнено, то сотрудник тоже запишется

Синтаксис:

Числа:

```
А = 10; // положительное
Б = -45; // отрицательное
В = 1.78; // дробное
```

«+» «-» «*» «/» «%» - действия над числами.
«%» - остаток от деления.

```
А = 10 % 3 //ответ 1
А = 10 % 4 //ответ 2
```

При написании кода система не воспринимает регистр, пробелы, знаки переноса строки и т.д.

Дата:

Всегда включает часы, минуты и секунды. К дате всегда можно добавить число:

```
Сегодня = ‘20120709033759’
Завтра = Сегодня + 86400 // + сутки
```

От одной даты можно отнять число. Получится другая дата, которая является результатом вычитания заданного количества секунд из вычитаемой даты.

От большей даты можно отнять меньшую дату. в результате получится разница **В СЕКУНДАХ**. Если разделить ее на 86400, то получится результат в днях.

Булево:

```
А = Истина;
Б = Ложь;
А = 10 < 3;
А = 10 = 10 ИЛИ 3 <> 5; // результат «Истина»
В = НЕ А; //результат «Ложь»
```

Строка:

```
А = «Строка»;
```

Конкатенация:

```
А = «Строка1» + «Строка2»; // результат «Строка1Строка2»
```

Многострочные строки – первая строка не заканчивается символом кавычки, а вторая и последующие будут начинаться с символа «|». Последняя строка закрывается символом кавычки и «**;**»

```
А = «Строка1
     |Строка2
     |Строка3»;
```

Если между записями поставлены пробелы, то они распознаются как символы переноса строки:

```
А = «Строка1» «Строка2» «Строка3»; // результат «Строка1
                                                  Строка2
                                                  Строка3»
```

Процедура:

```
Процедура Сумма(Перам1, Парам2)

КонецПроцедуры
```

При вызове данной процедуры используется ее имя и передается два параметра:

```
Сумма(10, 20);
```

Один параметр передавать нельзя. Это приведет к ошибке.

Если в процедуру будут передаваться переменные, то они будут передаваться «По ссылке», а не «По значению»

```
А = 10; Б = 20;
Сумма(А, Б);
```

Если внутри процедуры что-либо сделать с параметром, который является синонимом передаваемой переменной, то эти действия отразятся на переменной.

```
Процедура Сумма(Перам1, Парам2)
     Парам1 = 0;
КонецПроцедуры

А = 10; Б = 20;
Сумма(А, Б);
```

После отработки процедуры А = 0.

Если надо прервать эту связь, то надо перед принимаемым параметром написать ключевое слово «Знач»

```
Процедура Сумма(Перам1, Знач Парам2)
     Парам1 = 0;
     Парам2 = 0;
КонецПроцедуры
```

При этом «Парам2» будет являться не синонимом переменной «Б», а его копией. Действия с «Парам2» внутри процедуры никак не отразятся на переменной «Б».

Если предполагается вызов процедуры из других модулей, то процедура записывается с ключевым словом «Экспорт»:

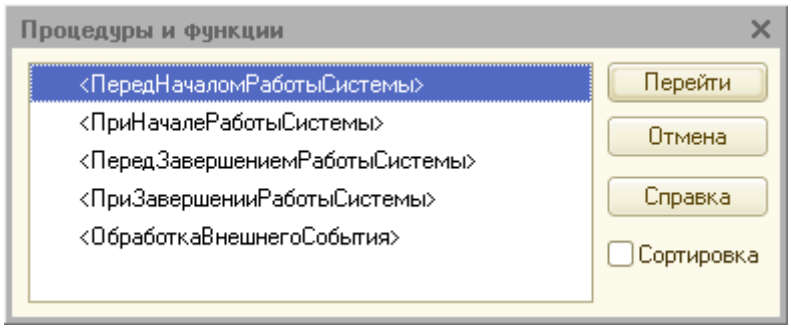
```
Процедура ИмяПроцедуры() Экспорт

КонецПроцедуры
```

Модули конфигурации:

- Модуль управляемого приложения – описание событий интерактивного старта и завершения системы в режиме управляемого приложения.
- Модуль сеанса.
- Модуль внешнего соединения.

«Дерево – Конфигурация – ПКМ – Модуль управляемого приложения»

кнопка «ПРОЦ» - список событий формы.



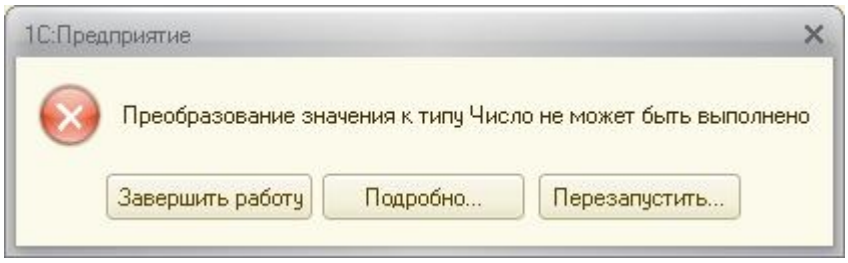
Нам нужно событие «ПриНачалеРаботыСистемы» - отрабатывает при запуске системы.

```
Процедура ПриНачалеРаботыСистемы()
    Предупреждение("Добро пожаловать!", 5);
    Сообщить("Сегодня " + ТекущаяДата());
КонецПроцедуры
```

Если

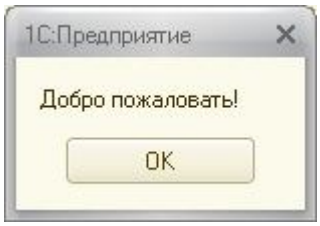
```
Сообщить(ТекущаяДата() + "Сегодня");
```

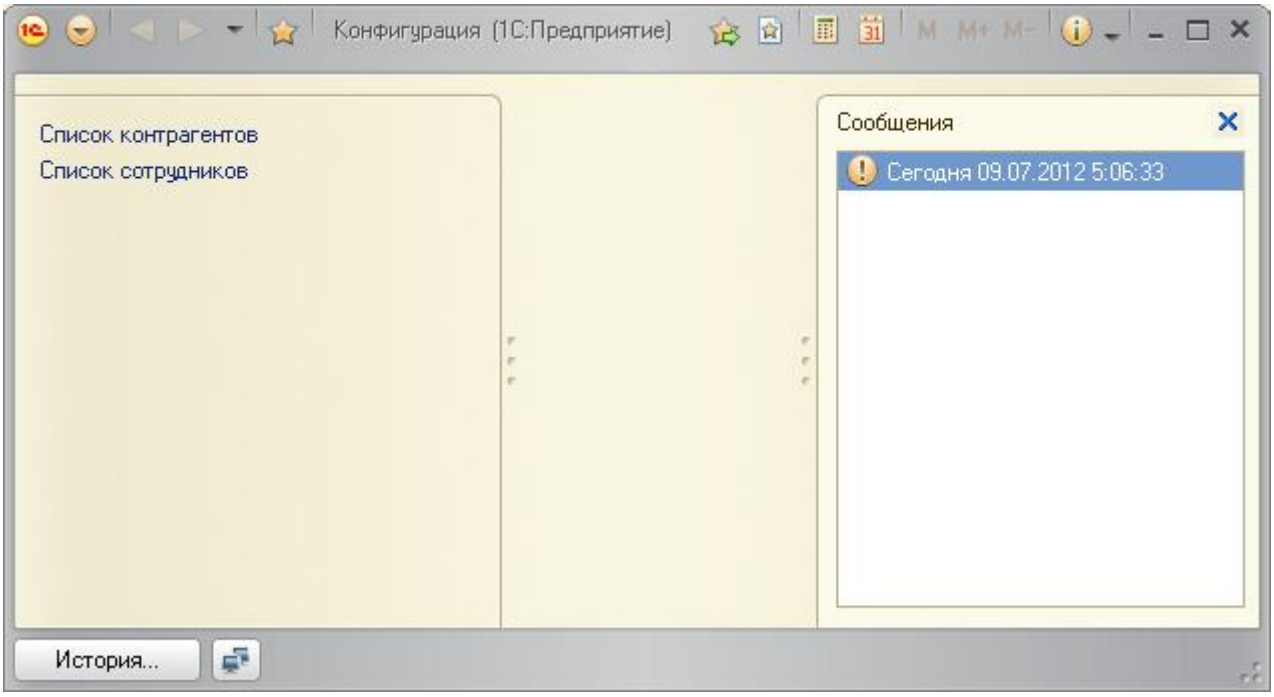
то при запуске в режиме исполнения будет ошибка:



Это вызвано тем, что «Сегодня» - это строковое выражение, и когда к нему присоединяется «ТекущаяДата()», то ее тип преобразуется в строковое значение.
«ТекущаяДата()» имеет тип «Дата», а к дате можно прибавить только число, а слово «Сегодня» числом не является, из-за этого возникает ошибка. Система пытается преобразовать значение «Сегодня» к типу число, но не может.

Результат:





Занятие 1.03

Справочники – открытые каталоги (пользователь самостоятельно может вносить записи), хранящие записи аналитики системы:
«Код» и «Наименование» - обязательные реквизиты справочника. Можно задать их длину. У реквизита «Код» можно задать тип:

- Число.
- Строка.

У справочника есть служебные реквизиты, которые скрыты от пользователя:
Установка галочки «Иерархический справочник» добавляет в справочник поле «Родитель», в которое записывается ссылка на элемент, которому принадлежит текущая запись.
В справочнике, кроме «Кода» и «Наименования» есть уникальный идентификатор (ссылка).

ID (Ссылка)	Код	Наименование	Родитель
	Число/Строка	Строка	
000AA	0001	Покупатели	
000AB	0002	Иванов ЧП	000AA

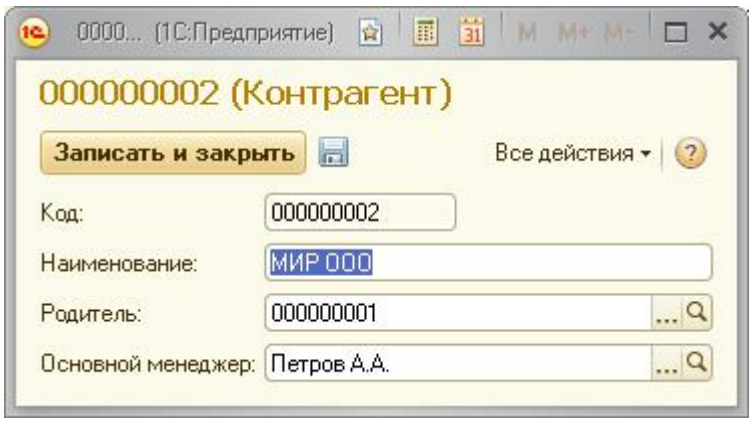
Вид иерархии:

- Иерархия групп и элементов - влияет на наличие дополнительного реквизита справочника «ЭтоГруппа».
- Иерархия элементов (например, справочник «Подразделения»).

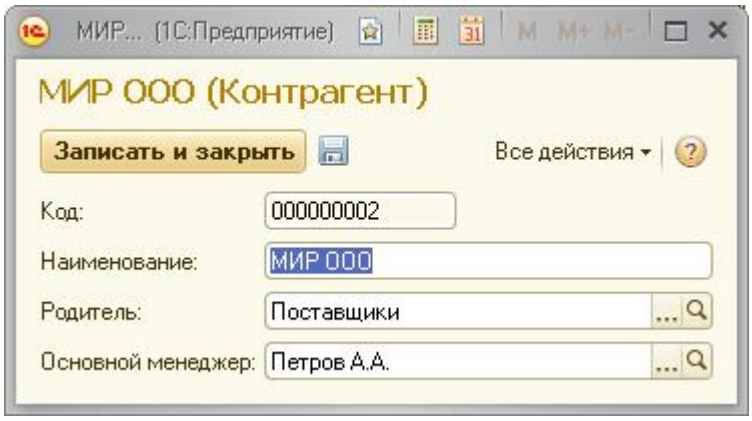
ID (Ссылка)	Код	Наименование	Родитель	ЭтоГруппа
	Число/Строка	Строка		
000AA	0001	Покупатели		Истина
000AB	0002	Иванов ЧП	000AA	Ложь

Основное представление – представление ссылки на элемент:

- В виде кода.



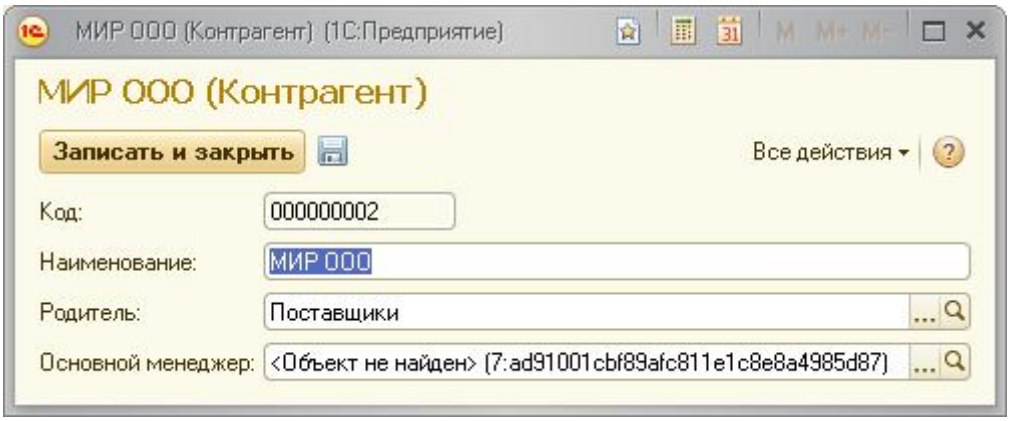
- В виде наименования.



Элементы любого справочника являются объектами аналитики, группы не являются объектами аналитики. Если для аналитики надо использовать режим подчинения, тогда надо использовать подчинение элементов элементам (иерархия элементов).
Отчеты в разрезе групп построить можно.

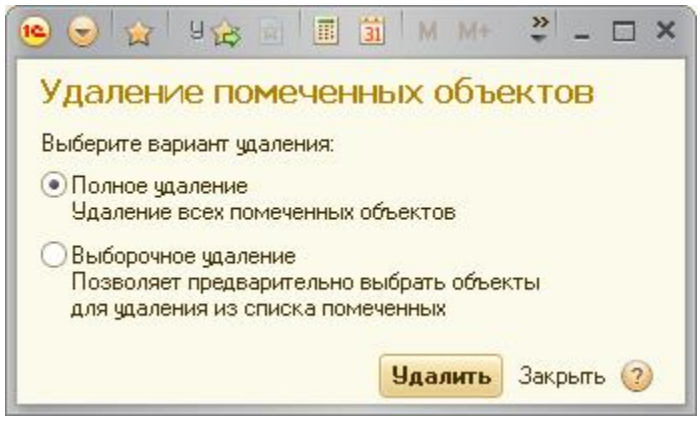
Поле «ПометкаУдаления» - хранит тип значения «Булево» - отвечает за попытку удаления записей из БД. Если элемент помечен на удаление, то данное поле содержит значение «Истина».

Удалим сотрудника Петрова А.А. (является основным менеджером «МИР ООО») непосредственно, тогда:

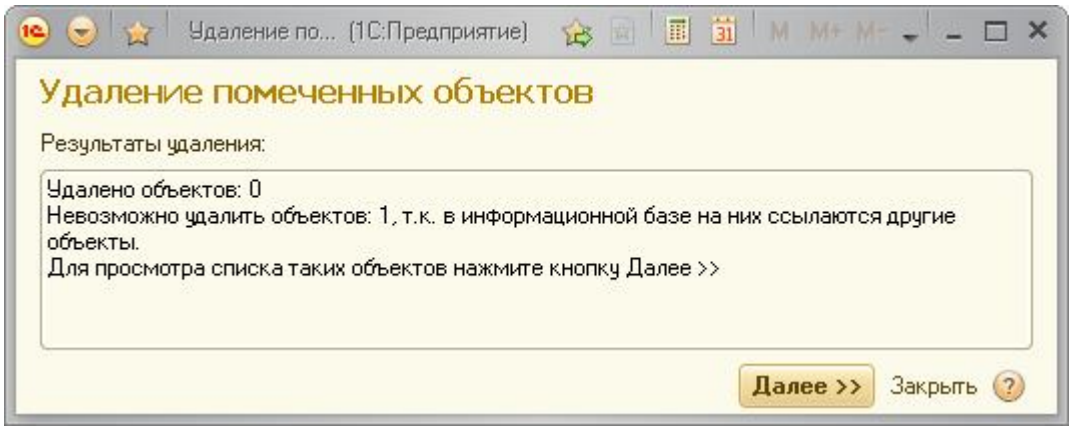


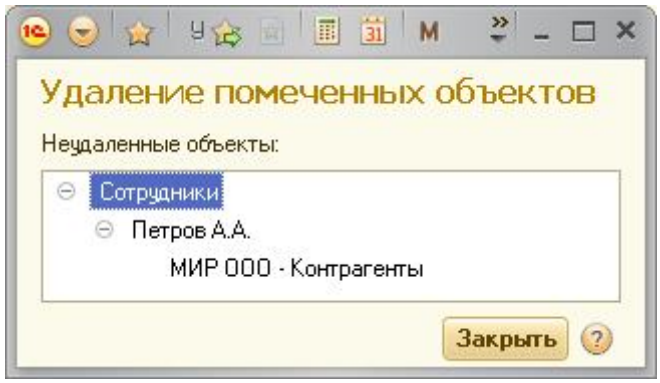
«Ctrl + Del» – непосредственное удаление объекта.
«7:ad91001cbf89afc811e1c8e8a4985d87» - это и есть ссылка на элемент справочника.

Помеченные на удаление элементы можно удалить с предварительной проверкой ссылочной целостности: «Главное меню - Все функции – Стандартные – Удаление помеченных объектов»



Если пометить на удаление сотрудника Петрова А.А. и попытаться удалить его через данную функцию, то попытка не удастся, т.к. на него ссылаются другие объекты (контрагент «МИР ООО»)



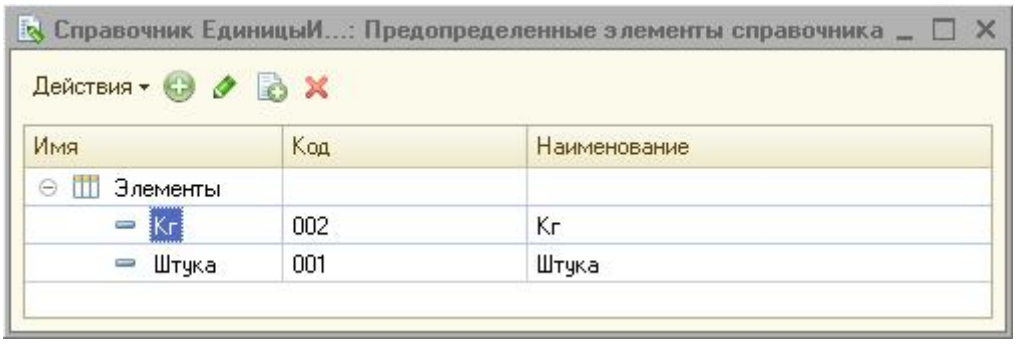


ID (Ссылка)	Код	Наименование	Родитель	ЭтоГруппа	ПометкаУдаления
	Число/Строка	Строка			
000AA	0001	Покупатели		Истина	Ложь
000AB	0002	Иванов ЧП	000AA	Ложь	Ложь

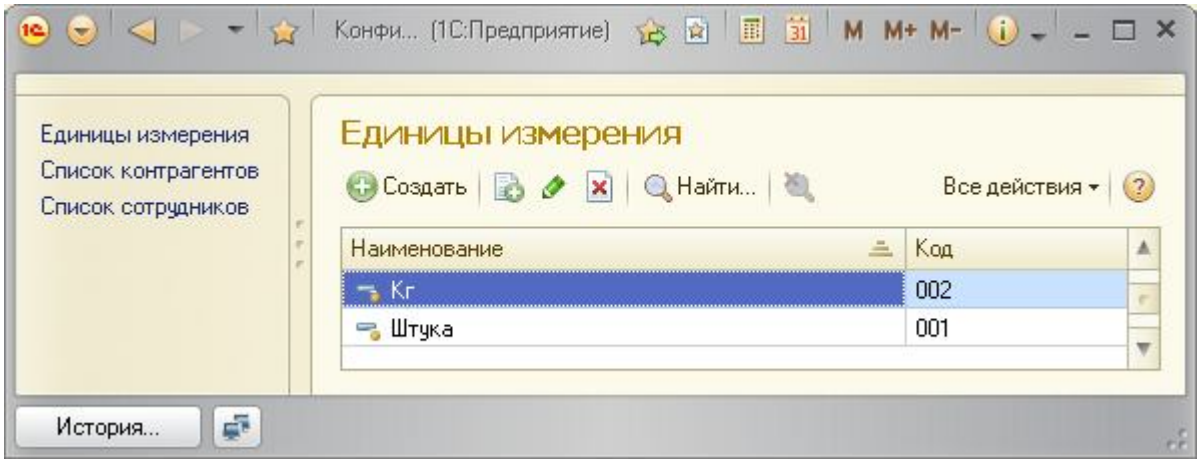
Поле «Предопределенный» - в конфигураторе можно создать записи, которые будут являться предопределенными элементами системы. Пользователь может поменять у них «Код», «Наименование», но удалить или пометить на удаление не сможет. Их можно использовать как объекты метаданных.

Создадим справочник «ЕдиницыИзмерения»:

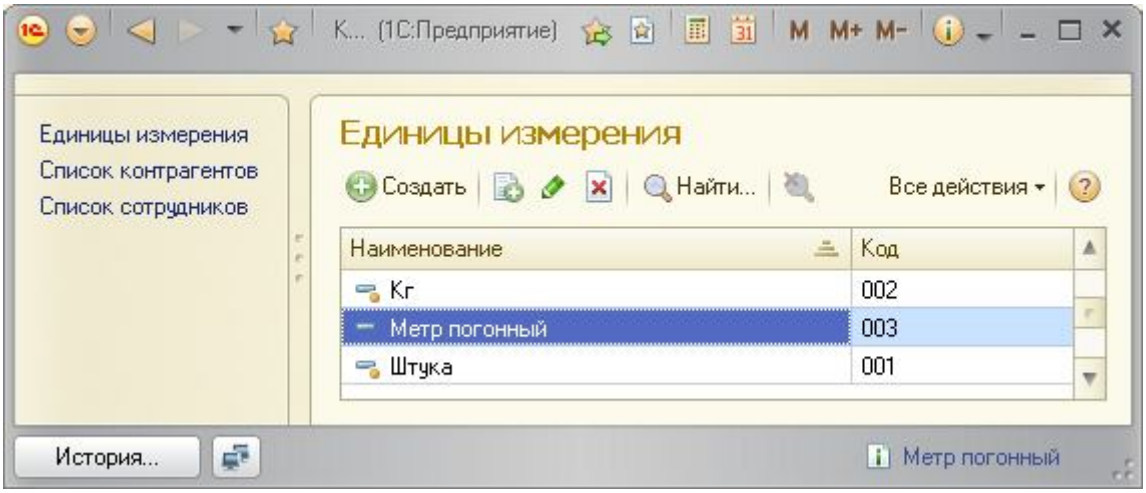
Имя – ЕдиницыИзмерения.
Синоним – Единицы измерения.
Представление объекта – Единица измерения.
Длина кода – 3.
«закл. Прочее - Предопределенные» - создадим предопределенные элементы справочника:



Предопределенные элементы в режиме исполнения:

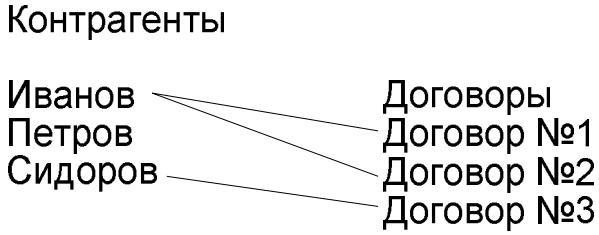


Создадим единицу измерения «Метр погонный» в режиме исполнения:



ID (Ссылка)	Код	Наименование	Родитель	ЭтоГруппа	ПометкаУдаления	Предопределенный
	Число/Строка	Строка				
000AA	0001	Покупатели		Истина	Ложь	Ложь
000AB	0002	Иванов ЧП	000AA	Ложь	Ложь	Ложь

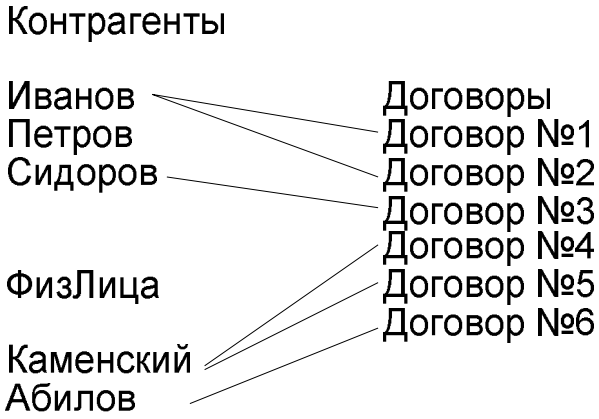
Поле «Владелец» - можно установить режим подчинения не внутри справочника, а одного справочника другому. Режим подчинения при этом можно установить как «один ко многим».



В поле «Владелец» указывается идентификатор записи справочника, которому принадлежит текущий элемент.

Родитель – это внутренний владелец.
Владелец – это внешний владелец.

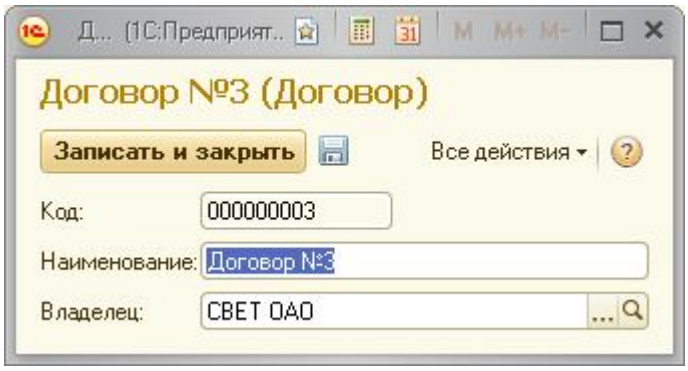
Для подчиненного справочника («Договоры») владельцами могут быть элементы разных справочников («Контрагенты», «ФизЛица»), а для элемента подчиненного справочника («Договоры») владелец существует только один (либо Контрагент, либо Физ. лицо).



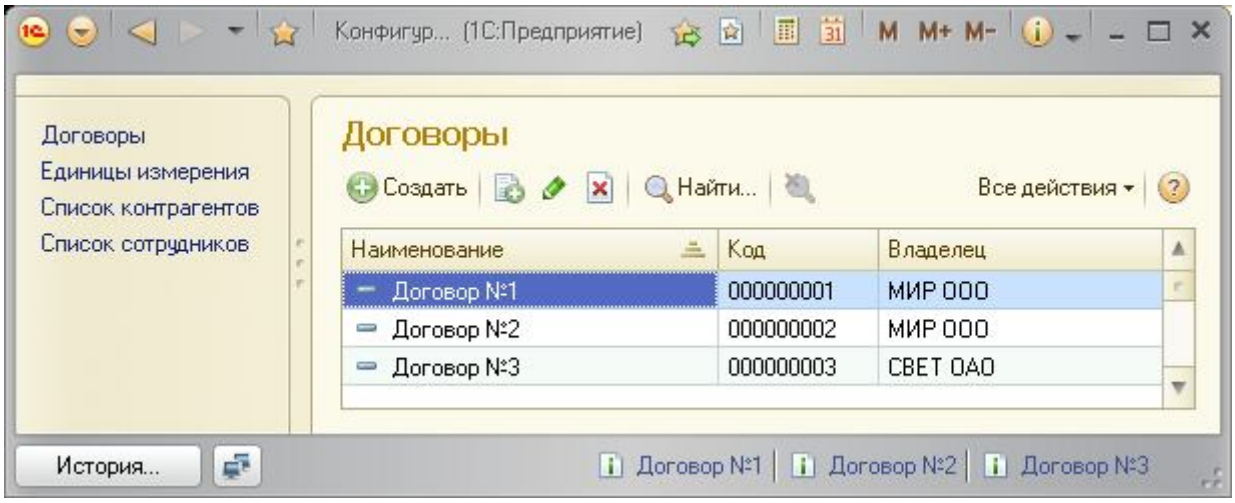
Создадим справочник «Договоры», который будет подчинен справочнику «Контрагенты»:
Имя – Договоры.
Синоним – Договоры.
Представление объекта – Договор.

На закл. «Владельцы» установим подчинение справочнику «Контрагенты».

В режиме исполнения для контрагента «МИР ООО» создадим договоры №№ 1 и 2, для контрагента «СВЕТ ОАО» создадим договор №3:



Код Договора №3 говорит о том, что записи договоров хранятся в отдельной таблице:



ID (Ссылка)	Код	Наименование	Родитель	ЭтоГруппа	ПометкаУдаления	Предопределенный	Владелец
	Число/Строка	Строка					
000AA	0001	Покупатели		Истина	Ложь	Ложь	
000AB	0002	Иванов ЧП	000AA	Ложь	Ложь	Ложь	

Рассмотрим пример справочника «Номенклатура»:

ID (Ссылка)	Код	Наименование	ПометкаУдаления	Родитель	ЭтоГруппа	Услуга	ЕдИзм	ЦенаПродажи
						Булево	Шт	

Имя – Номенклатура.
Синоним – Номенклатура.
Представление объекта - Товар/Услуга.
Закл. «Иерархия»:
Иерархический справочник – Да.
Вид иерархии – Иерархия групп и элементов.

Добавим реквизиты:

- Услуга:

Имя – Услуга.
Синоним – Услуга.
Тип – Булево.

- Единица измерения:

Имя – ЕдИзм.
Синоним – Ед изм.
Тип – СправочникСсылка.ЕдиницыИзмерения.

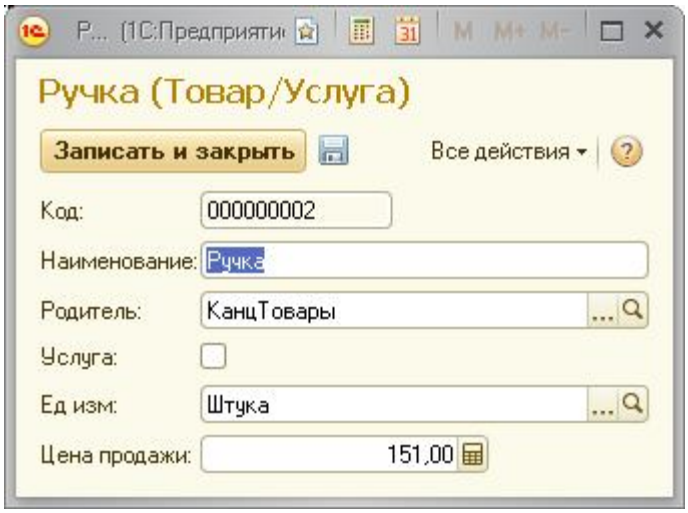
Свойства - группа «Представление» - свойство «Значение заполнения» - для новых элементов справочника в значение данного реквизита по умолчанию будет подставляться выбранное значение.

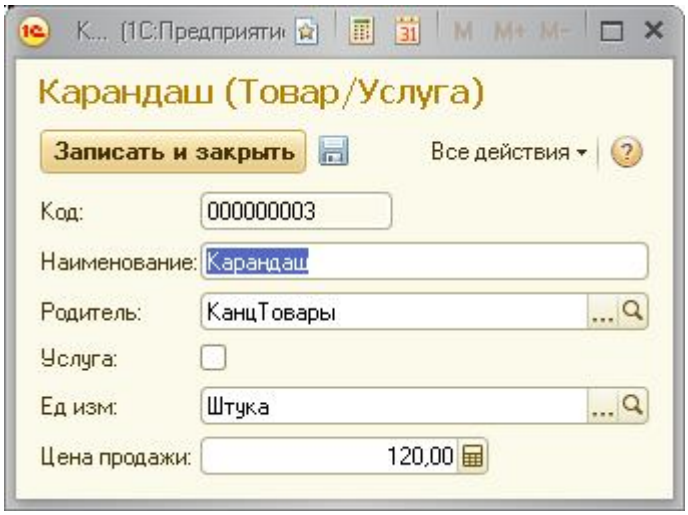
Значение заполнения – Штука.

- Цена продажи:

Имя – ЦенаПродажи.
Синоним – Цена продажи.
Тип – Число 15, 2.

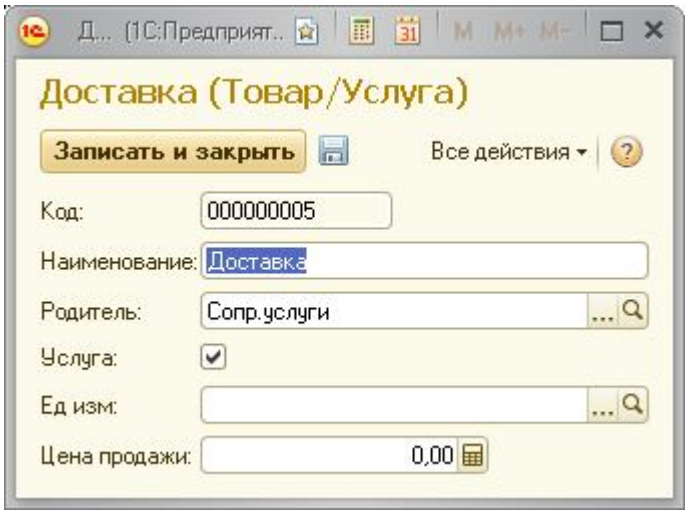
В режиме исполнения в справочнике «Номенклатура» создадим группу «КанцТовары», а в ней элементы:





Единица измерения по умолчанию (Штука) подставляется автоматически.

Создадим группу «Сопр.услуги», а в ней элемент:



Константы – для хранения условно-постоянных записей. Пользователь может их редактировать. Значение этих записей можно в любом месте, где это необходимо.

Создадим константу «Название организации»:
Имя – НазваниеОрганизации.
Синоним - Название организации.
Тип – Строка.
Неограниченная длина – Да.

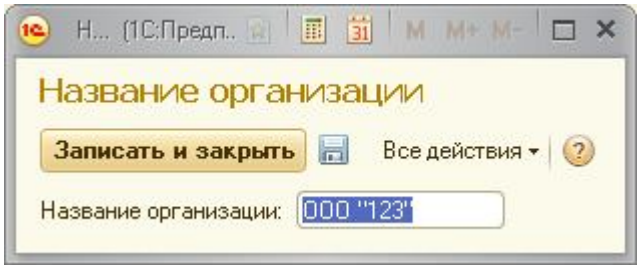
Свойство «Неограниченная длина» - определяет, где будет храниться «Название организации» в таблице констант или в отдельной таблице (содержит 3 столбца):

Ссылка на элемент, для которого сохраняется значение. В примере: ссылка на константу «Название организации»	Значение индекса записи. Нужное количество строк для хранения значения.	Само значение. Хранится в виде строки длиной 100 знаков. Если название организации 115 знаков, то для его хранения будет отведено 2 записи

Если строка неограниченной длины не содержит ни одного знака, то на размер таблицы БД она не влияет, т.е. места не занимает.

В режиме исполнения система автоматом добавляет в панель действий поле «Сервис» - для дополнительных возможностей.

Создадим название организации:



Занятие 1.04

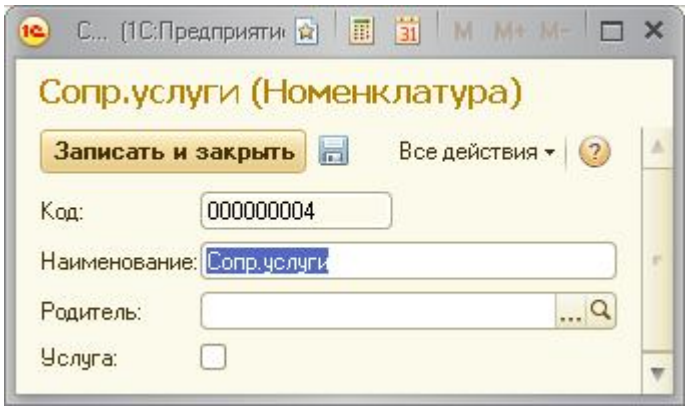
Наследование признака «Услуга»:

Сделаем, чтобы создаваемые элементы наследовали значение признака «Услуга» у группы.

Справочник «Номенклатура» - реквизит «Услуга» - свойство «Использование»:

- Для элемента.
- Для группы.
- Для группы и элемента.

Выберем значение «Для группы и элемента» - при этом в режиме исполнения галочка «Услуга» появится как у элементов, так и у групп справочника «Номенклатура».



Если выбрана группа «Сопр.услуги», то значения полей «Ед изм» и «Цена продажи» не доступны и в них хранится значение типа NULL – отсутствие значения.

Разница между значениями «Неопределено» и «NULL»

Тип значения «NULL», которое находится в полях «Ед изм» и «Цена продажи» для групп справочника обозначает, что значение здесь не может существовать в принципе.
NULL – отсутствие значения в принципе.

Тип значение «Неопределено» может возникнуть, если, например, есть описание переменной, но значение ей не присвоено – отсутствует значение.
НЕОПРЕДЕЛЕНО – неинициализированное значение.

Сделаем так, чтобы при создании нового элемента справочника «Номенклатура» значение признака «Услуга» наследовалось от родителя, т.е. группы, к которой будет принадлежать элемент.

Событие «Заполнение реквизитов нового элемента» возникает при создании нового элемента (программно или интерактивно), при копировании элемента, при вводе на основании.

Событие это выполняется в «Модуле объекта» - модуль для выполнения событий, не связанных с формой (закл. «Прочее» - «Модуль объекта»).

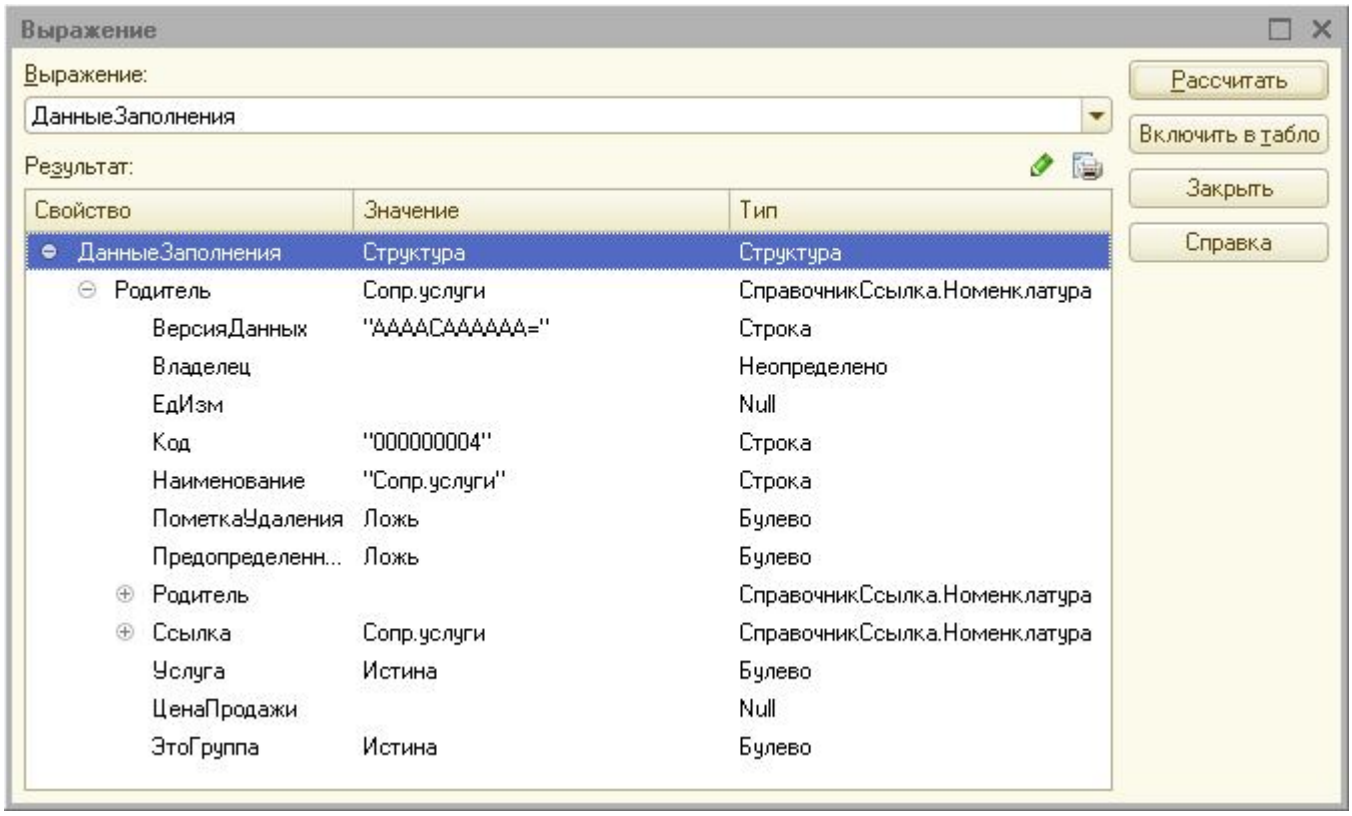
Опишем процедуру «`ОбработкаЗаполнения`». Она имеет 2 предопределенных аргумента: «ДанныеЗаполнения» - то, чем система собирается заполнить текущий элемент. Например, в форме списка справочника установлен какой-либо отбор. Если, при этом, создается новый элемент, то в процедуру «`ОбработкаЗаполнения`» система может передать информацию о том, что отбор установлен, это событие можно перехватить.

В режиме исполнения «Все действия – Настроить список» - установка различных отборов для отображения списка элементов.

Но, в нашем примере, если даже установлен отбор по признаку «Услуга», то признак не устанавливается автоматом при создании нового элемента. Чтобы признаки учитывали все отборы, надо установить настройку «Заполнять из данных заполнения» - автоматически заполнять значения по переданным данным заполнения переданных как самой системой, так и заданных программно.

Но в примере надо, чтобы вне зависимости от установки отбора «Услуга», устанавливался или не устанавливался признак «Заполнять из данных заполнения»

Поставим точку останова, в режиме исполнения в группе «Сопр.услуги» создадим новый элемент и посмотрим, что содержится в аргументе «ДанныеЗаполнения»



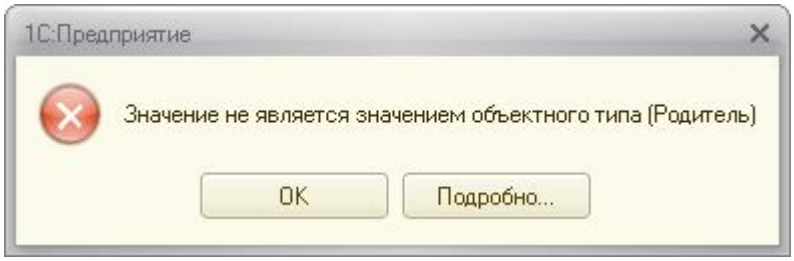
Поле «Родитель» для нового элемента справочника будет автоматически заполнено представленными данными.

Обратимся к реквизиту создаваемого элемента справочника, т.е. формы элемента еще нет, обращаемся к элементу на сервере и присваиваем ему значение:

```
Процедура ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)
     Услуга = ДанныеЗаполнения.Родитель.Услуга;
КонецПроцедуры
```

Теперь, при создании нового элемента в группе, признак «Услуга» элемента наследует значение признака «Услуга» родителя, т.е. группы.

Но если создавать элемент вне подчинения, т.е. в «ДанныеЗаполнения» нет родителя, то система выдаст ошибку:



Т.е. перед обращением к «ДанныеЗаполнения» надо проверить, есть там нужные данные или нет.

Тип значения «Структура» - это таблица, состоящая из 2-х колонок:

Ключ	Значение
Родитель	Сопр.услуги
Всегда тип «Строка»	Любой тип

Количество строк не ограничено, можно программно добавлять или удалять строки.

«`ДанныеЗаполнения = Неопределено`» тогда, когда элемент справочника создается программно и в него ничего не передается. В этом случае прервем процедуру.

```
Если ДанныеЗаполнения = Неопределено Тогда
     Возврат;
КонецЕсли;
```

Если в «ДанныеЗаполнения» что-то передается системой или программно, то надо узнать, есть ли в передаваемых данных поле «Родитель».

«Синтакс-помощник – Универсальные коллекции значений – Структура – Методы» - описание всего, что можно сделать со структурой.

Метод «Свойство» - получает значение элемента по указанному имени, а также проверяет наличие свойства.

```
Если ДанныеЗаполнения.Свойство("Родитель")
     Услуга = ДанныеЗаполнения.Родитель.Услуга;
КонецЕсли;
```

Но такая запись не очень хороша, т.к. признак «Услуга» - это реквизит справочника, а так как в нем свойство «Заполнять из данных заполнения = Истина», то если в «ДанныеЗаполнения» при отрабатывании процедуры будет найден ключ «Услуга», то система заполнит его автоматом, т.е. принудительно его устанавливать не надо.

```
Если ДанныеЗаполнения.Свойство("Родитель")
     //Услуга = ДанныеЗаполнения.Родитель.Услуга;
     ДанныеЗаполнения.Вставить("Услуга", ДанныеЗаполнения.Родитель.Услуга)
КонецЕсли;
```

«Услуга» - название реквизита, который будет заполнен.
«ДанныеЗаполнения.Родитель.Услуга» - значение, которым будет заполнен этот реквизит.

Т.е. значение реквизита не устанавливается напрямую, а стандартному механизму передается инструкция заполнить значение реквизита «Услуга» при отрабатывании процедуры «ОбработкаЗаполнения».

Параметр «СтандартнаяОбратотка» - отвечает за работу стандартного механизма заполнения реквизитов. Если «СтандартнаяОбратотка = Ложь», то ничего автоматом не заполнится, но принудительное заполнение сработает:

```
Услуга = ДанныеЗаполнения.Родитель.Услуга;
```

Итого:

```
Процедура ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)
     Если ДанныеЗаполнения = Неопределено Тогда
          Возврат;
     КонецЕсли;

     Если ДанныеЗаполнения.Свойство("Родитель")
          //Услуга = ДанныеЗаполнения.Родитель.Услуга;
          ДанныеЗаполнения.Вставить("Услуга", ДанныеЗаполнения.Родитель.
Услуга)
     КонецЕсли;

     //СтандартнаяОбработка = Ложь;
КонецПроцедуры
```

К элементу структуры можно обратиться по его ключу:

```
ДанныеЗаполнения.Вставить("Услуга", ДанныеЗаполнения.Родитель. Услуга);
```

Данные заполнения можно как прочитать:

```
А = ДанныеЗаполнения.Услуга
```

так и записать в них значение:

```
ДанныеЗаполнения.Услуга = Ложь
```

В примере у реквизита «Единица измерения» если элемент – товар, то надо проверять на заполнение, а если элемент – услуга, то заполнение не обязательно. Проверку надо проводить независимо от формы (на стороне сервера), т.е. вообще форму надо загружать по минимуму, а все проверки выполнять на стороне сервера.

Модуль объекта – процедура «ОбработкаПроверкиЗаполнения»

```
ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты)
```

Отказ – признак успешности транзакции записи объекта в БД.

```
Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты)
     Если НЕ Услуга Тогда
          Если ЕдИзм.Пустая() Тогда
               Отказ = Истина;
          КонецЕсли;
     КонецЕсли;
КонецПроцедуры
```

Но данная процедура будет работать не только для элементов, но и для групп, что неправильно. Добавим код:

```
Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты)
     Если ЭтоГруппа Тогда
          Возврат;
     КонецЕсли;

     Если НЕ Услуга Тогда
          Если ЕдИзм.Пустая() Тогда
               Отказ = Истина;
          КонецЕсли;
     КонецЕсли;
КонецПроцедуры
```

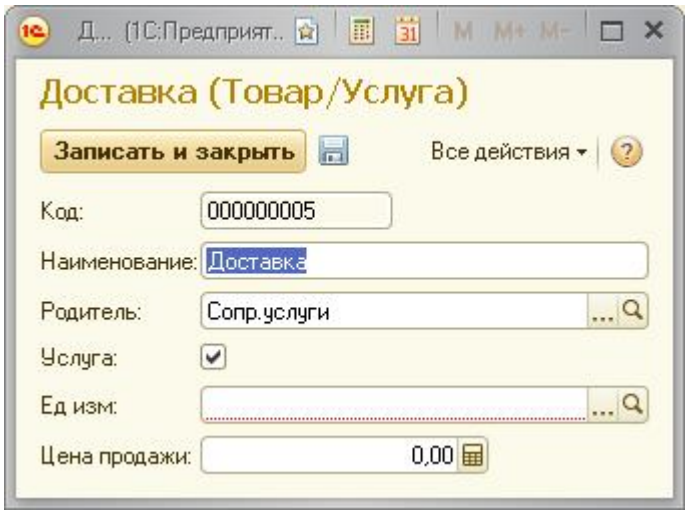
Но при этом пользователь не будет проинформирован, почему ему не удалось записать элемент. Поэтому такой способ применяется для описания каких-либо сложных алгоритмов.

Реализуем проверку при помощи стандартной обработки. Обратимся к коллекции «ПроверяемыеРеквизиты» - попытаемся записать услугу «Доставка» с незаполненной единицей измерения. При этом у реквизита «ЕдИзм» свойство:
«Проверка заполнения – Выдавать ошибку».

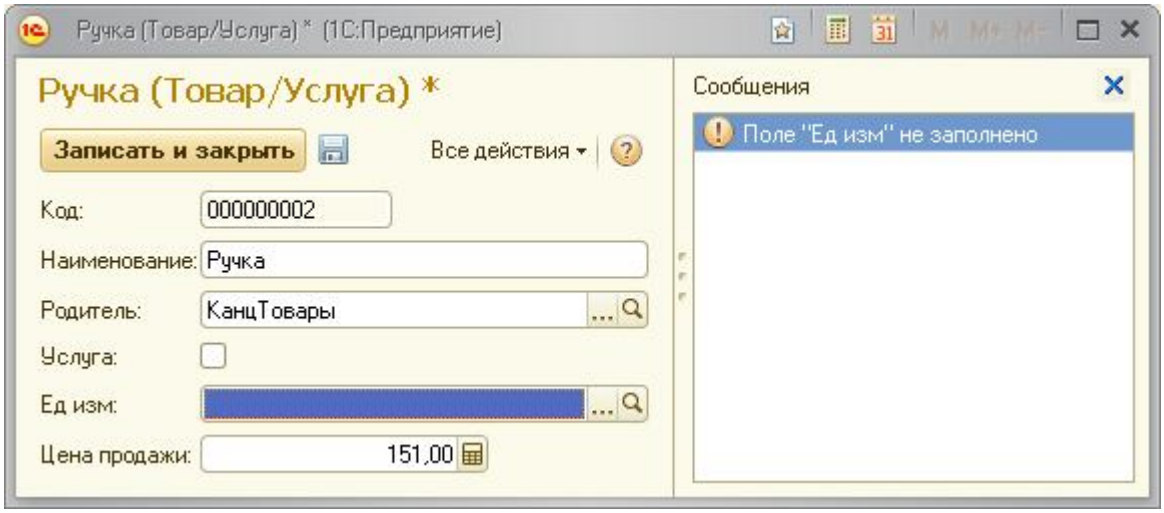
Здесь содержится список реквизитов, значение которых проверяется на заполнение.
В процедуре если элемент справочника не группа и услуга, то найдем нужный элемент массива и удалим его из проверяемых реквизитов.

```
Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты)

    Если НЕ ЭтоГруппа И Услуга Тогда
        ПроверяемыеРеквизиты.Удалить(ПроверяемыеРеквизиты.Найти("ЕдИзм"));
    КонецЕсли;

КонецПроцедуры
```

В режиме исполнения:
Перезапишем услугу «Доставка» без значения единицы измерения:



Перезапишем товар «Ручка» без значения единицы измерения:



Стандартная обработка сама формирует СОО, более правильно выбирает момент отмены транзакции.

«Синтакс-помощник – Прикладные объекты - Справочники» - объекты встроенного языка, которые появляются при создании любого справочника.

- СправочникиМенеджер – обращение ко всем справочникам. Например:

```
Справочники.ЕдиницыИзмерения.Кг
```

часть «`Справочники`» и является контекстом обращения ко всем справочникам системы.

- СправочникМенеджер – объект, позволяющий обращаться к одному конкретному справочнику.

```
Справочники.ЕдиницыИзмерения
```

В качестве свойства справочника можно получить предопределенный элемент справочника обратившись к нему по имени.

```
А = Справочники.ЕдиницыИзмерения.Кг
```

Обращаясь к методам данного объекта, можно получить остальные типы значения справочника.

- СправочникСсылка – предназначен для передачи ссылки на элемент справочника. Не предназначен для постоянного чтения, т.к. каждый раз при чтении будет производиться запрос к БД и возвращаться полная копия объекта.

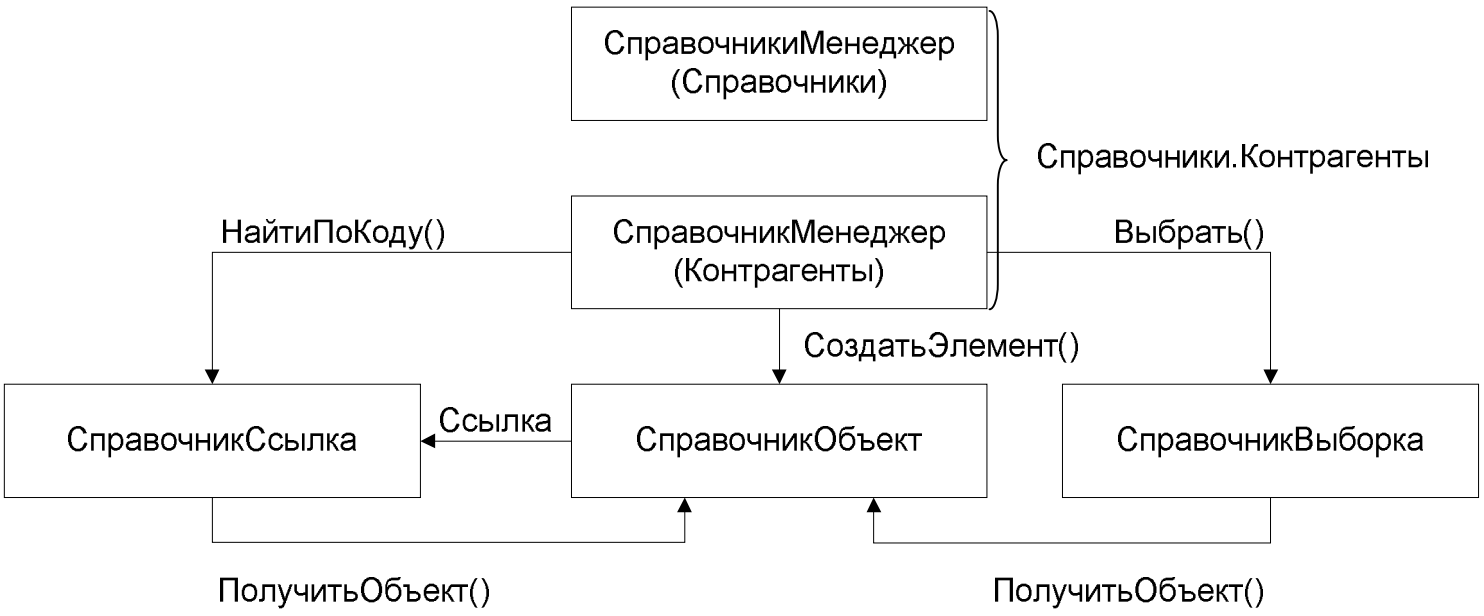
- СправочникОбъект – позволяет прочитать и изменить данные. Работает с транзакциями, т.е. блокирует записи, освобождает и следит за конфликтом версий данных.

- СправочникВыборка – порционно читает данные из БД (так, чтобы не напрягать сервер), получает ссылку и все значения реквизитов

Как найти справочник по коду, а потом в нем что-либо поменять:
СправочникОбъект можно получить методом «`ПолучитьОбъект()`» из типа значения «СправочникСсылка» или «СправочникВыборка».

Например, в цикле перебираем элементы справочника с помощью объекта «СправочникВыборка», чтобы поменять номер, надо получить объект, тогда объект блокируется, номер меняется и объект записывается обратно в БД.

Получили «СправочникОбъект», изменили в нем что-либо и передаем ссылку на него – из «СправочникОбъект» с помощью свойства «Ссылка»



Пример: Найти всех сотрудников с заданной датой рождения.

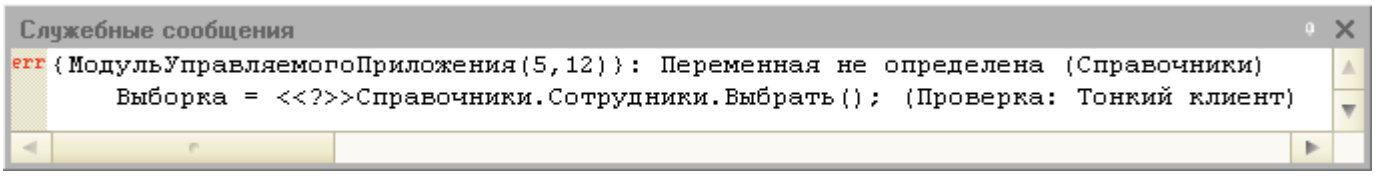
Метод «НайтиПоРеквизиту» не подойдет, т.к. по переданной дате мы найдем только новорожденных, а нам надо, чтоб совпадали только день и месяц рождения. Еще данный метод возвращает ссылку на первый найденный элемент справочника и все, а именинников может быть больше. Будем работать с типом значения «СправочникВыборка».

Процедуру будем выполнять при начале работы системы.

```
Выборка = Справочники.Сотрудники.Выбрать()
```

так можно получить тип значения «СправочникВыборка».

Но в «Модуле управляемого приложения» нельзя обратиться к БД, нельзя сделать ничего, что не связано с интерфейсом. «Модуль управляемого приложения» запускается на компьютере клиента. Поэтому появится СОО:



Но можно передать управление серверу, выполнить там все необходимые действия, вернуть результат обратно на клиента и отобразить его.

Создадим общий модуль «Общие механизмы», который будет выполняться на сервере, при этом с клиентского модуля будет разрешен вызов сервера:
Имя – ОбщиеМеханизмы.
Синоним – Общие механизмы.
Сервер – Да.
Вызов сервера – Да.

В общем модуле создадим функцию:

```
Функция ПолучитьСписокИменинников() Экспорт

КонецФункции
```

«Экспорт» - для того, чтобы можно было вызвать эту функцию из других модулей системы.

В «Модуле управляемого приложения» обратимся к этой функции:

```
Процедура ПриНачалеРаботыСистемы()
     СписокИменинников = ОбщиеМеханизмы.ПолучитьСписокИменинников();
КонецПроцедуры
```

Метод «Следующий» типа значения «СправочникВыборка» - для получения следующего элемента из выборки.

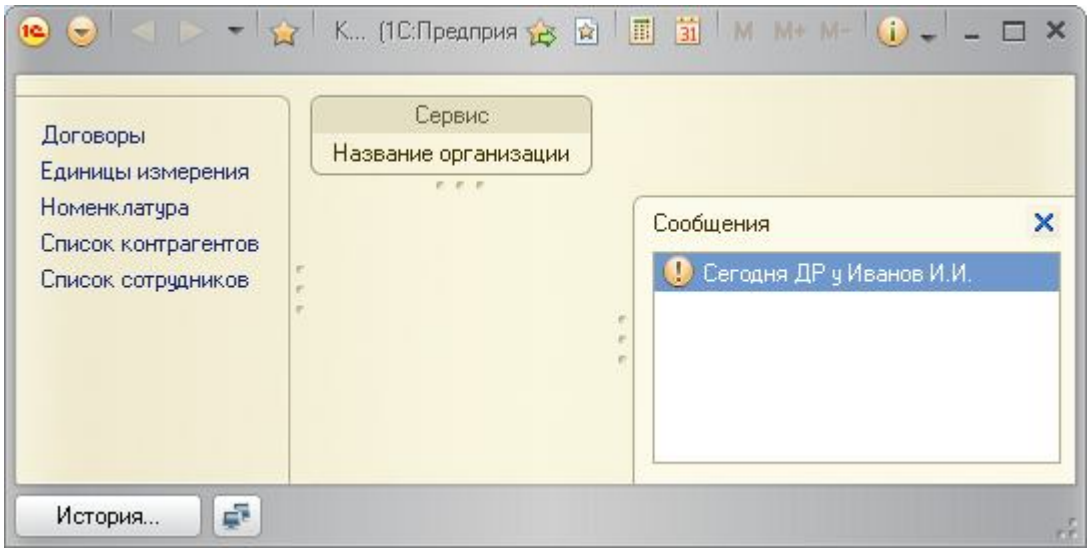
```
Функция ПолучитьСписокИменинников() Экспорт

    //получим порядковый номер дня и месяца текущей даты
    ТД_День  = День(ТекущаяДата());
    ТД_Месяц = Месяц(ТекущаяДата());

    //создадим пустой массив
    Массив = Новый Массив

    //получим выборку из справочника "Сотрудники"
    Выборка = Справочники.Сотрудники.Выбрать();

    //переберем элементы выборки
    Пока Выборка.Следующий() Цикл

        //получим порядковый номер дня и месяца даты рождения сотрудника
        ДР_День  = День(Выборка.ДатаРождения);
        ДР_Месяц = Месяц(Выборка.ДатаРождения);

        //сравним день и месяц даты рождения сотрудника с текущими
        Если ДР_День = ТД_День И ДР_Месяц = ТД_Месяц Тогда

            //добавим новый элемент в массив
            Массив.Добавить(Выборка.Наименование);

        КонецЕсли;
    КонецЦикла;

    //вернем массив выбранных именинников в точку вызова
    Возврат Массив;

КонецФункции
```

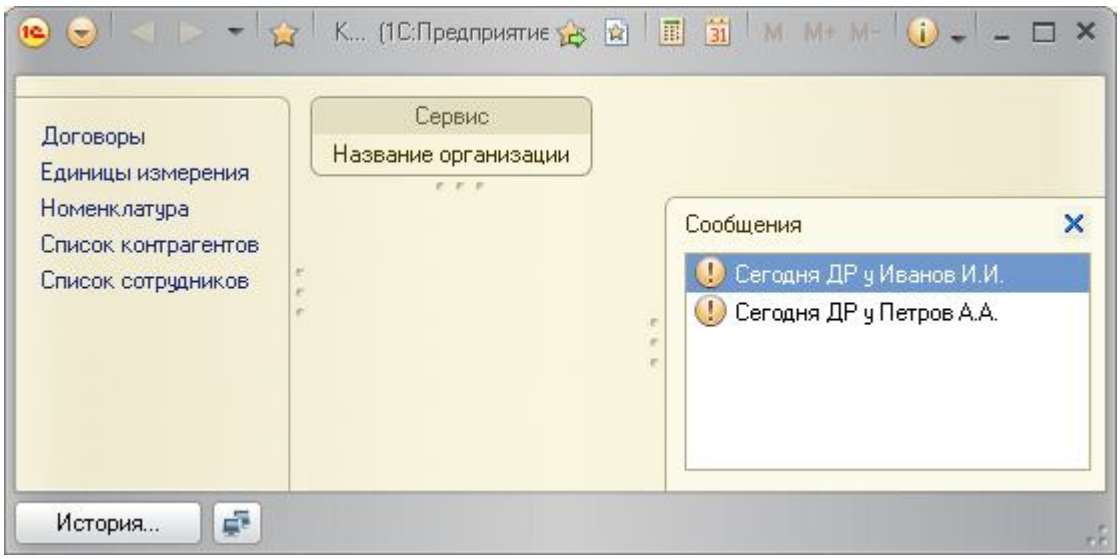
В точке вызова:

Массив «СписокИменинников» является коллекцией, которая перебирается с помощью цикла перебора коллекции.

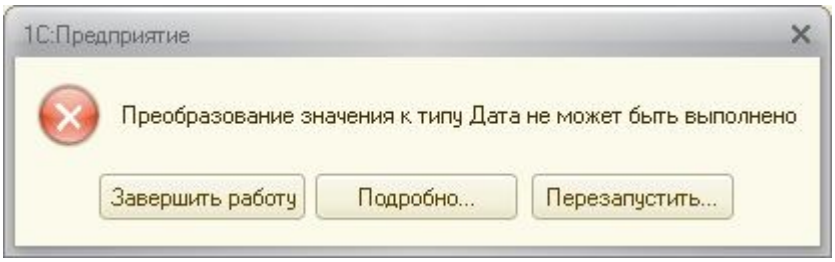
```
Процедура ПриНачалеРаботыСистемы()
    СписокИменинников = ОбщиеМеханизмы.ПолучитьСписокИменинников();

    Для Каждого ЭлементМассива Из СписокИменинников Цикл
        Сообщить("Сегодня ДР у " + ЭлементМассива);
    КонецЦикла;

КонецПроцедуры
```

Итого: сегодня 16.07, поставим у Иванова такую же дату рождения и получим:





Создадим в справочнике «Сотрудники» группу «Работающие» и поместим туда Иванова и Каменского. Теперь при запуске программы будет вылетать ошибка:



Ошибка возникла потому, что у реквизита «ДатаРождения» свойство «Использование» - «Для элемента». Т.е. когда выборка натыкается на группу, то возвращается значение:

```
Выборка.ДатаРождения = NULL
```

А значение «NULL» к типу «Дата» не преобразовывается.

Исправим ошибку:

```
Функция ПолучитьСписокИменинников() Экспорт

    //получим порядковый номер дня и месяца текущей даты
    ТД_День  = День(ТекущаяДата());
    ТД_Месяц = Месяц(ТекущаяДата());

    //создадим пустой массив
    Массив = Новый Массив;

    //получим выборку из справочника "Сотрудники"
    Выборка = Справочники.Сотрудники.Выбрать();

    //переберем элементы выборки
    Пока Выборка.Следующий() Цикл

        Если Выборка.ЭтоГруппа Тогда
            Продолжить;
        КонецЕсли;

        //получим порядковый номер дня и месяца даты рождения сотрудника
        ДР_День  = День(Выборка.ДатаРождения);
        ДР_Месяц = Месяц(Выборка.ДатаРождения);

        //сравним день и месяц даты рождения сотрудника с текущими
        Если ДР_День = ТД_День И ДР_Месяц = ТД_Месяц Тогда

            //добавим новый элемент в массив
            Массив.Добавить(Выборка.Наименование);

        КонецЕсли;
    КонецЦикла;

    //вернем массив выбранных именинников в точку вызова
    Возврат Массив;

КонецФункции
```

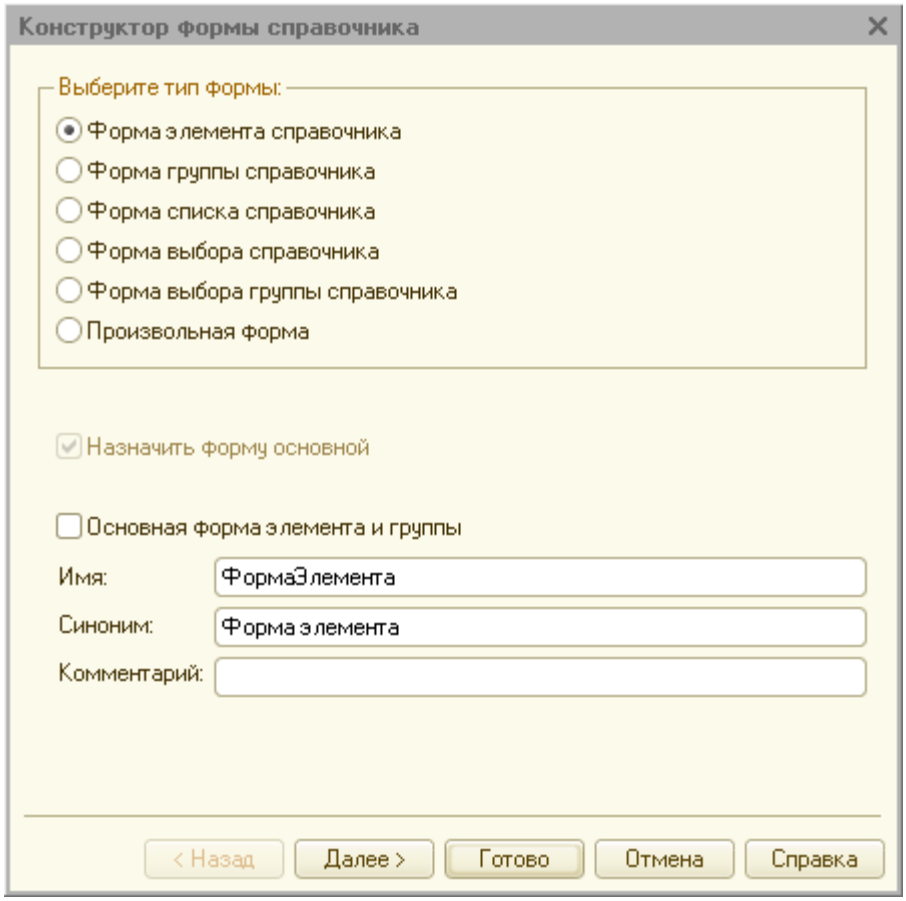
«Продолжить» - используется для перехода на следующий шаг цикла без отработки кода, который находится в цикле ниже.

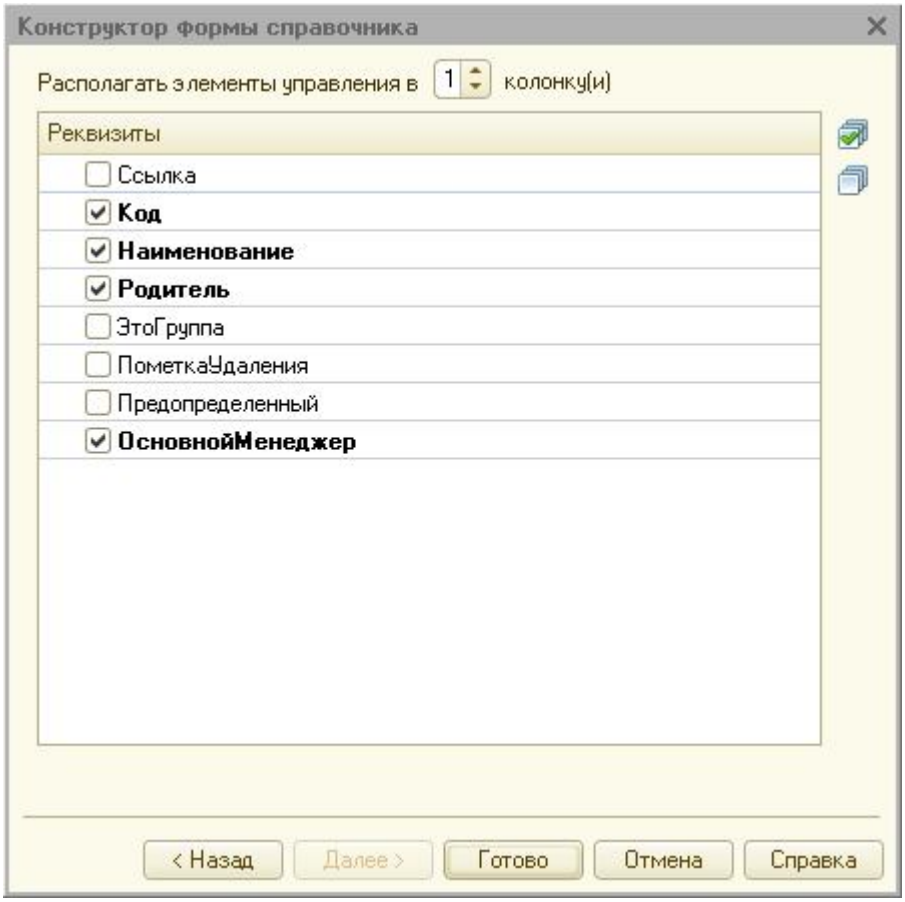
Занятие 1.05

Переименуем группу «Работающие» справочника «Сотрудники» в «Администрация».
У справочника «Контрагенты» есть реквизит «Основной менеджер» и сейчас мы можем заполнить его любым сотрудником, независимо от значения признака «Работающий» элемента справочника «Сотрудники».
Сделаем, чтобы при выборе неработающего сотрудника в качестве основного менеджера контрагента, система выдавала предупреждение.

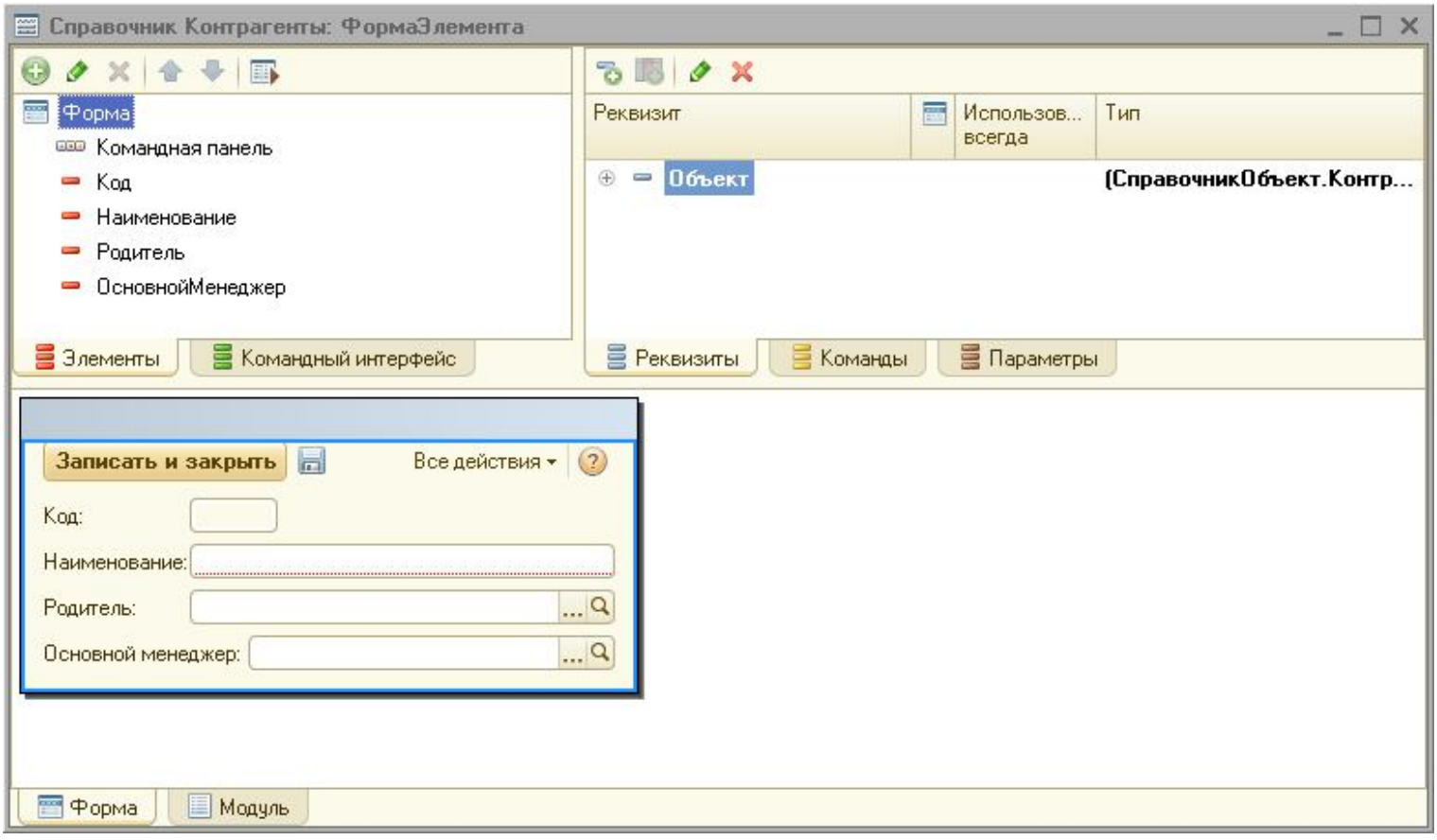
Для этого надо переопределить поведение формы элемента справочника «Контрагенты», т.е. форма справочника теперь не будет формироваться автоматом.

Форм у объекта может быть много, но основной будет только одна.





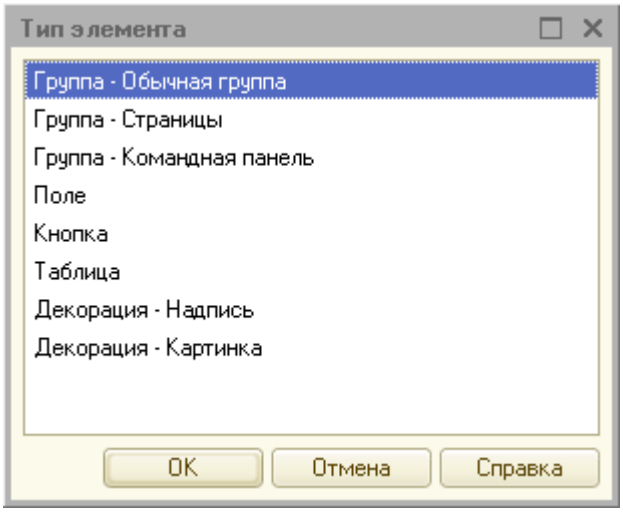
При нажатии на кнопку «Готово» мы попадем в редактор управляемых форм:



Внизу отображен образец отображения формы в режиме исполнения. Как будет выглядеть форма в режиме исполнения, зависит от прав пользователя, функциональных опций и настроек пользователя.

Элементы формы можно добавлять/удалять в левом верхнем окне. Реквизиты объекта, которые надо отобразить на форме, перетаскиваются из верхнего правого окна.

Реквизиты в форме можно объединять в группы. Например, чтобы «Код» и «Наименование» отображались в одной строке, надо добавить в дерево элементов (окно слева) группу:



И задать свойство группы:
«Группировка – Горизонтальная»

«Отображать заголовок – Ложь» - убираем заголовок группы.
«Отображение - Нет» - убираем вообще признак присутствия группы.

Можно сделать вложенные группы - таким образом можно по-разному располагать элементы формы. Процесс похож на верстку HTML-документов.

Отличие такого рисование формы в том, то клиенту передается описание взаимного расположения объектов формы (в xml-формате), а на клиенте уже происходит отрисовка формы. В результате снижается трафик между сервером и клиентом.

Кнопки на форме задаются в правом верхнем окне на закладке «Команды»:
- «Стандартные команды» - не надо описывать действие.
- «Глобальные команды» - создаются на ветке «Общие – Группы команд». Такие команды можно присвоить нескольким формам.
- «Команды формы» - описание команды текущей формы. Процедура описывается в модуле формы.

Команды на форме можно отображать как «Кнопка» или «Гиперссылка» - свойство «Вид» команды на форме. Гиперссылками рекомендуется отображать редко используемые команды.

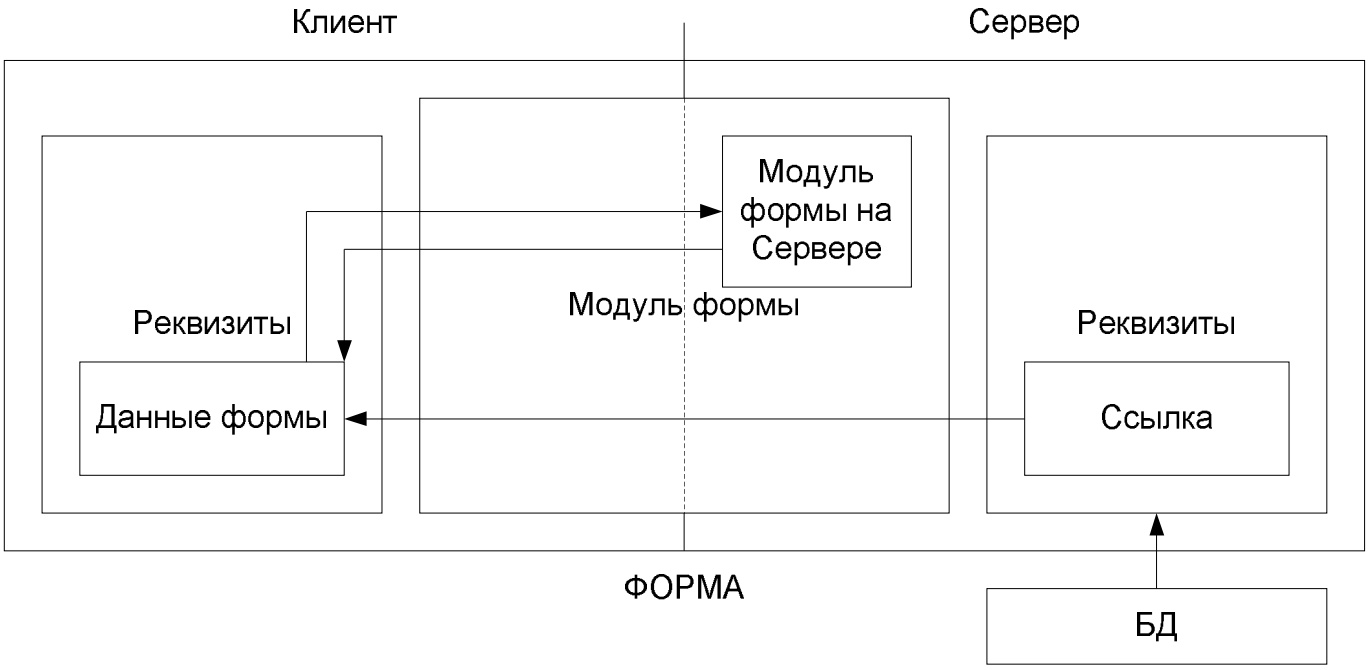
Закл. «Параметры» - используются в основном для открытия формы. Для управления поведением формы при ее открытии. Переменные формы, которые можно заполнить программно при ее открытии.

Форма создается на сервере, а клиенту передается ее проекция.

Форма делится на две части: Клиент и Сервер. Модуль формы одновременно существует на Клиенте и на Сервере. Реквизиты формы есть на Сервере, на Клиенте есть их представление.

Например, при открытии формы справочника:
Есть база данных, из которой система читает значения, в том числе и значения реквизитов. Это происходит на Сервере. Далее начинает конструироваться форма, которая открывается на клиенте.
На Клиент не передаются все данные реквизитов, а только ссылка и представление. Реквизит, который на Сервере содержит Ссылку, по которой можно обратиться к БД, переезжает на Клиента и представляется там в виде типа значений «Данные формы».
«Данные формы» - это таблица вида «Ключ - Значение»
Ссылку на клиенте можно только куда-либо передать, проверить по ней данные из БД невозможно.
Например, если надо что-то проверить в данных формы, то управление передается модулю формы на Сервере, там производятся необходимые действия и результат передается обратно Клиенту.

Итог: форма на стороне Клиента является прототипом HTML-страницы. Нельзя же в браузере найти данные БД, чтобы сделать это, надо обратиться к Серверу. Т.е. на стороне Клиента находятся только Представления данных, чтобы что0либо сделать с самими данными, надо вызывать Сервер.

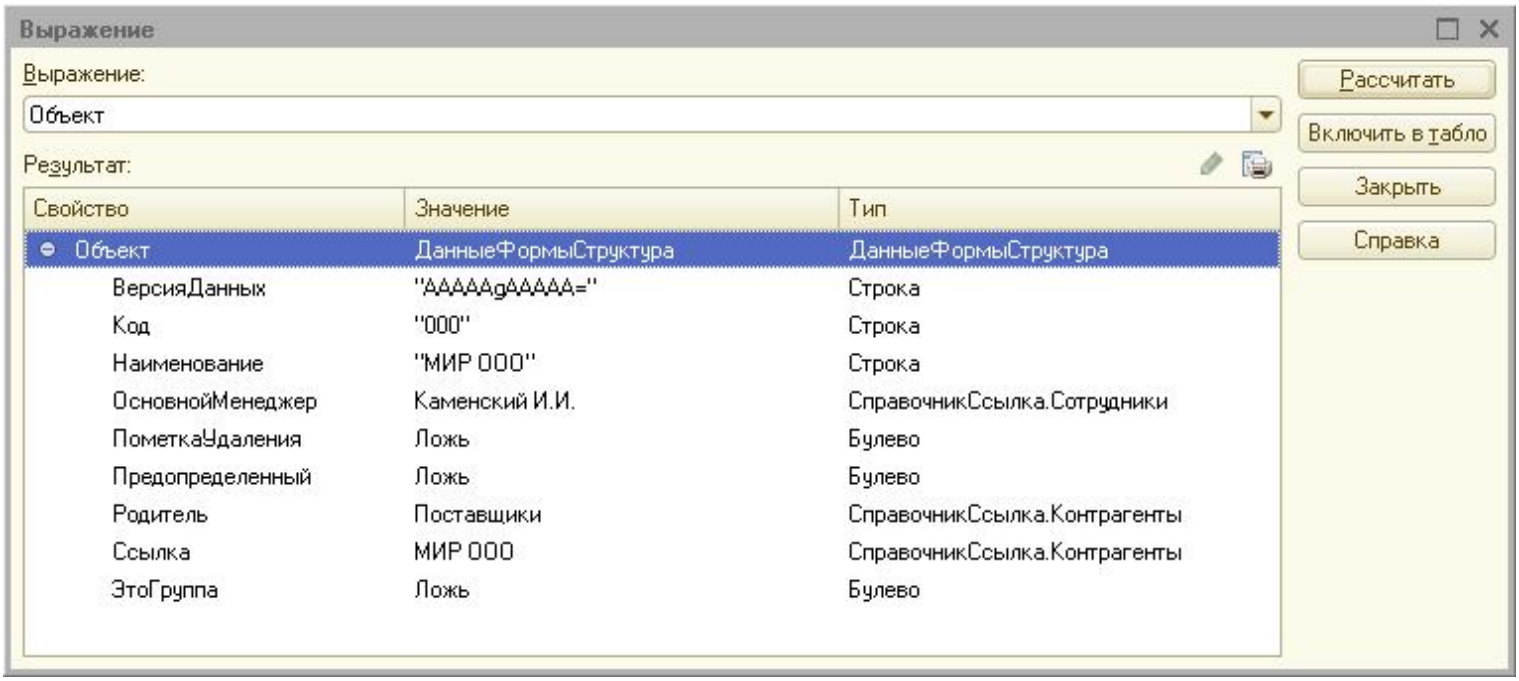


Для проверки свойства «Работающий» сотрудника, выбираемого основным менеджером, обратимся к событию «ПриИзменении» реквизита «ОсновнойМенеджер».

```
&НаКлиенте
Процедура ОсновнойМенеджерПриИзменении(Элемент)
     // Вставить содержимое обработчика.
КонецПроцедуры
```

Посмотрим, к каким реквизитам можно сейчас обратиться:
Поставим точку останова. В режиме исполнения у контрагента «МИР ООО» изменим основного менеджера:
«Элемент» - это поле формы. Нарисованный реквизит, в котором производится редактирование, с данными он ни как не связан.

Как узнать, что пользователь выбрал в элементе:
Надо обращаться к редактируемым данным:
«Объект.ОсновнойМенеджер»



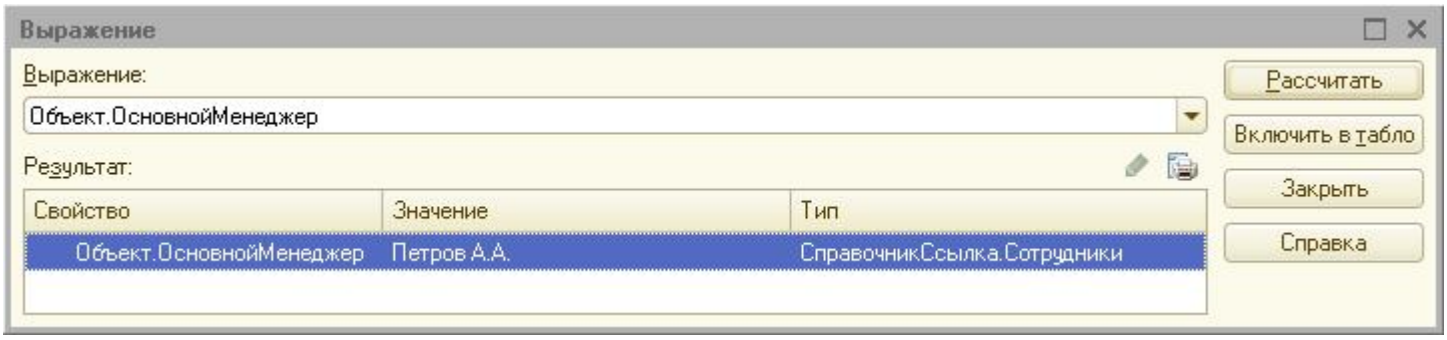
«Объект» - это не в чистом виде элемент справочника «Контрагенты», а его проекция.

Чтобы узнать значение признака «Работающий» у сотрудника напишем функцию:
В нее будем передавать самого менеджера. Поставим точку останова и вызовем эту функцию из процедуры «ПриИзменении»:

```
&НаКлиенте
Процедура ОсновнойМенеджерПриИзменении(Элемент)
     Результат = ПроверитьРаботает(Объект.ОсновнойМенеджер);
КонецПроцедуры

&НаСервере
```

```
Функция ПроверитьРаботает(Менеджер)
	А = 0;
КонецФункции
```

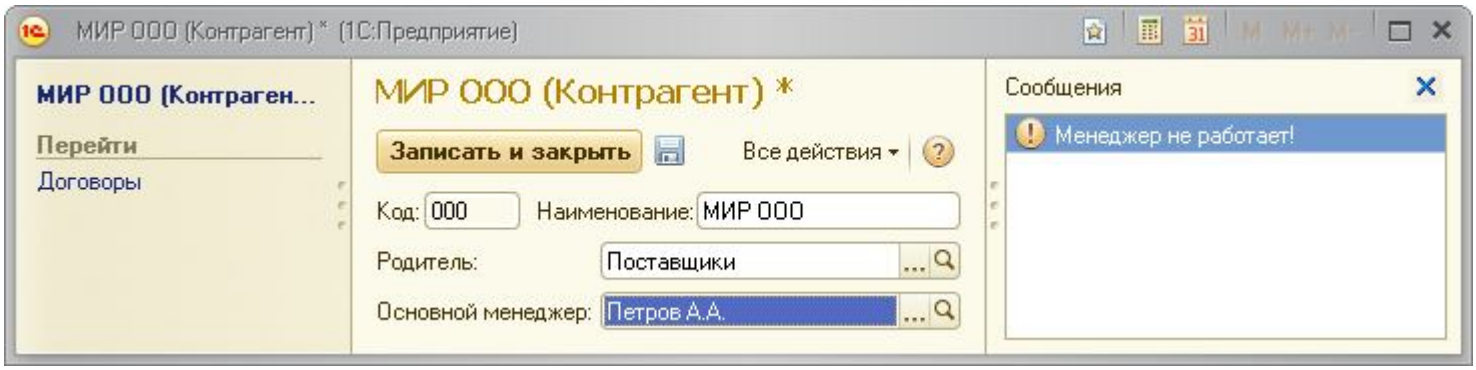
«Объект.ОсновнойМенеджер» - недоступно ничего, кроме самой ссылки:



Передаем управление Серверу и здесь у Менеджера уже доступны все свойства, т.е. можно обратиться к БД и проверить признак «Работающий». Результат возвращается на Клиента, на котором в зависимости от результата выведется сообщение:

```
&НаКлиенте
Процедура ОсновнойМенеджерПриИзменении(Элемент)
	Результат = ПроверитьРаботает(Объект.ОсновнойМенеджер);
	Если НЕ Результат Тогда
		Сообщить("Менеджер не работает!");
	КонецЕсли;
КонецПроцедуры

&НаСервере
Функция ПроверитьРаботает(Менеджер)
	Возврат Менеджер.Работающий;
КонецФункции
```

Итог: если выберем неработающего менеджера, то:



Но данное описание работы на Сервере не совсем верное. Директивы исполнения на Сервере: «&НаСервере» - все данные формы конвертируются в серверные. После выполнения необходимых действий данные опять конвертируются в данные формы и возвращаются на Клиента. Зачем:

Данные на Сервере могут быть не актуальные, пользователь их мог уже поменять в форме, поэтому и производятся конвертации. Это медленно.

«&НаСервереБезКонтекста» - система ничего не конвертирует, но при этом нельзя обратиться ни к каким свойствам объекта.
Т.е. в нашем примере в функцию на Сервере можно было не передавать Менеджера, т.к. доступен «Объект»:

```
Объект.ОсновнойМенеджер.Работающий;
```

А здесь такая информация уже недоступна, т.к. актуальные данные не сконвертированы и не получены. Доступно только то, что передается Серверу.

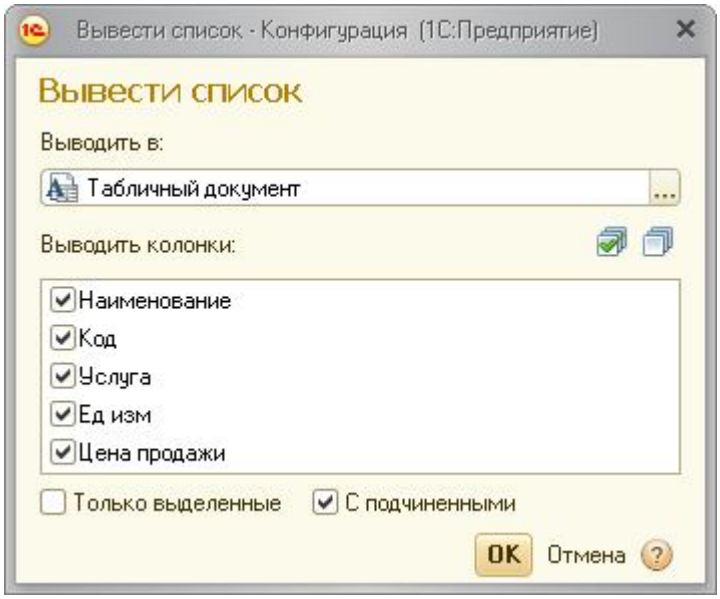
Поэтому маленькие проверки лучше описывать как «НаСервереБезКонтекста»

```
&НаСервереБезКонтекста
Функция ПроверитьРаботает(Менеджер)
    Возврат Менеджер.Работающий;
КонецФункции
```

Разница по-простому (на примере ЦУП (Сервер) и космонавтов (Клиент)):
НаСервереБезКонтекста – космонавт спросил у ЦУП «Сколько будет 2*2?», ЦУП ему ответил «4»

НаСервере – космонавту надо померить температуру. Этого нельзя сделать, не отвезя космонавта на ЦУП.

Создадим прайс-лист номенклатуры. В режиме исполнения можно вывести подобие прайс-листа и средствами системы:
«Все действия – Вывести список»
и настроить выводимый список.

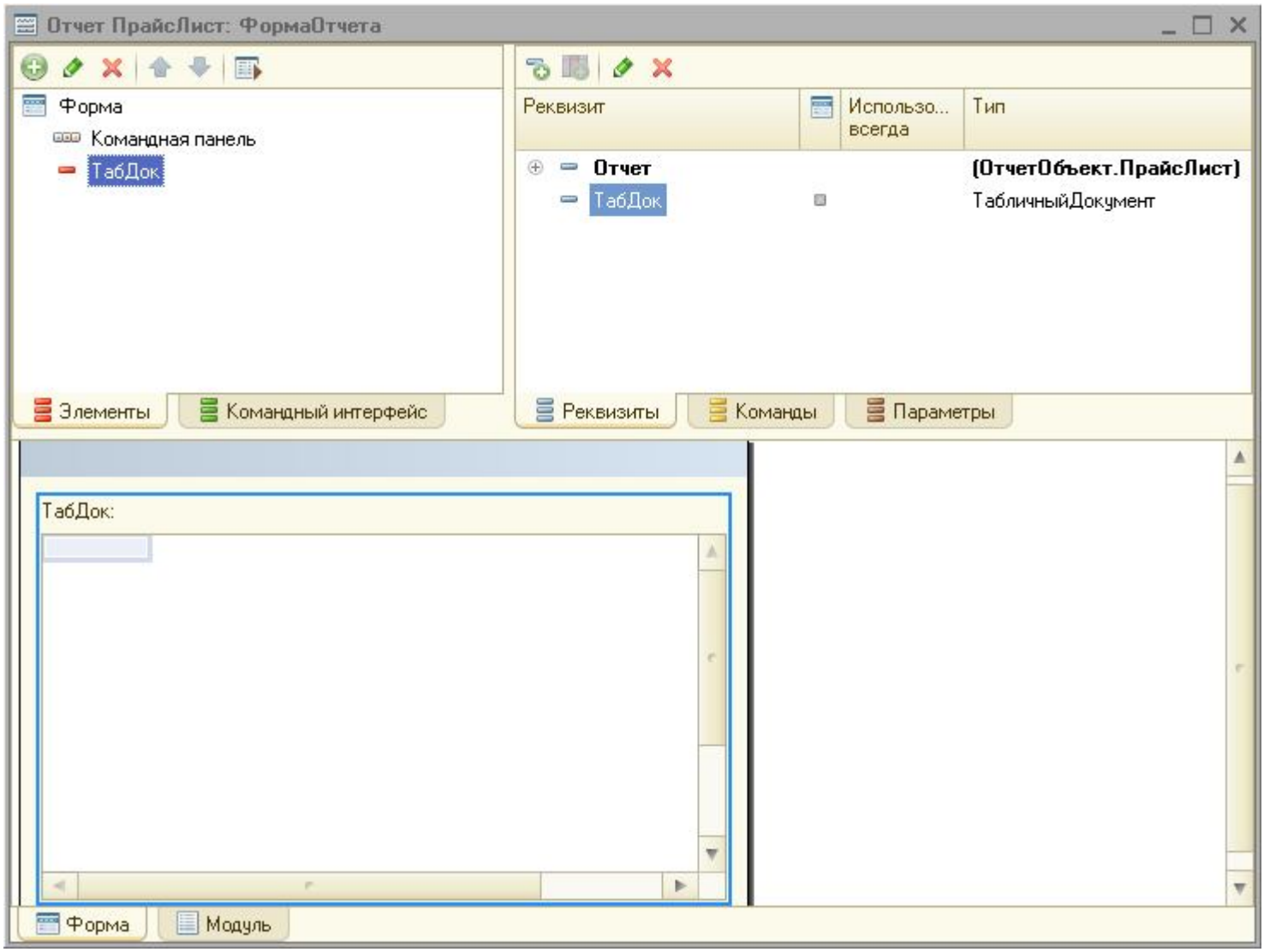


Создадим отчет «Прайс-лист»:
Имя – «ПрайсЛист»
Синоним – «Прайс-лист»

Нарисуем форму отчета:
Есть данные, которые надо показать пользователю. Нужен результирующий документ, куда эти данные попадут (могут быть разные объекты, в примере будем использовать Табличный документ). Этот табличный документ можно держать в памяти и не показывать пользователю, пока он полностью не будет сформирован, а можно сразу нарисовать его в форме:

Добавим реквизит:
Имя – «ТабДок».
Тип – «ТабличныйДокумент»

и перенесем его в форму (окно слева вверху).



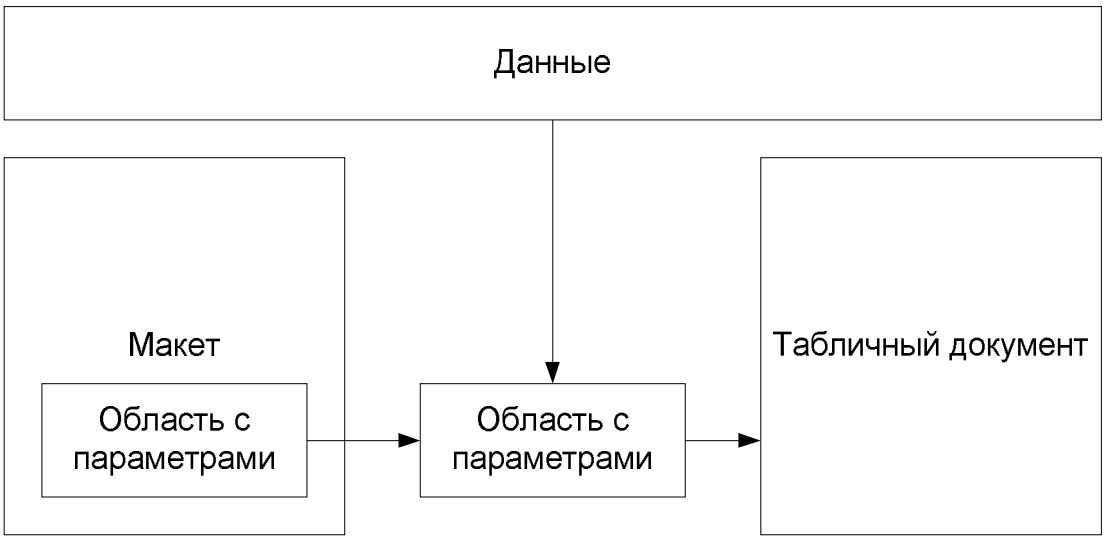
Чтобы убрать заголовок – в свойствах реквизита «ТабДок»:
ПоложениеЗаголовка – «Нет».

Теперь нужна процедура, которая будет помещать данные в табличный документ

Нарисуем кнопку – создадим команду «Сформировать прайс»
Имя – «СформироватьПрайс»

```
&НаКлиенте
Процедура СформироватьПрайс(Команда)
     // Вставить содержимое обработчика.
КонецПроцедуры
```

Создадим серверную процедуру, которая будет заполнять Табличный документ данными:
в процедуру будем передавать сам Табличный документ.
Можно напрямую заполнять Табличный документ данными, но тогда придется вручную описывать оформление, что очень долго. Выход – сделать еще один табличный документ (Макет), в котором настроить все табличное оформление. В Макете хранятся области с параметрами.
Алгоритм: получаем из Макета область с параметрами, заполняем параметры значениями и выводим ее в табличный документ.

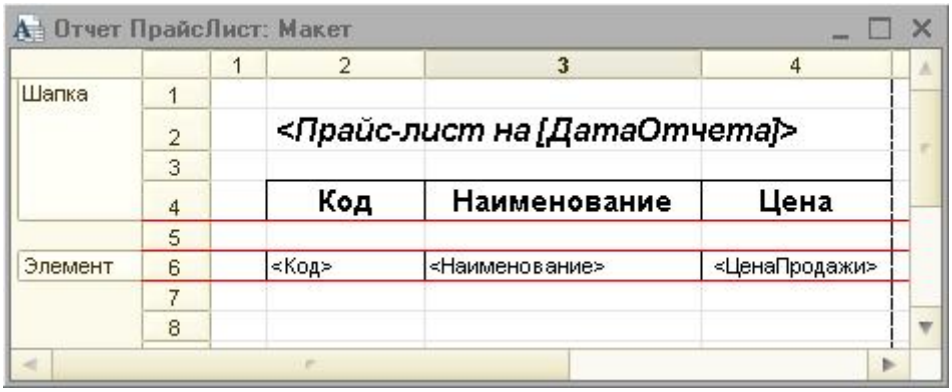


В отчете создадим новый макет с типом «Табличный документ».

«Прайс-лист на [ДатаОтчета]»:
Заполнение – «Шаблон».
в [] заключено имя параметра.

«Код», «Наименование», «ЦенаПродажи».
Заполнение – «Параметр».
т.е. эти значения будут заменены параметрами.

Область можно получить по координатам или указав ее имя:
«Таблица – Имена – Назначить имя» - создание поименованных областей в Макете.



Получаем Макет на Сервере:

```
Макет = Отчеты.ПрайсЛист.ПолучитьМакет("Макет");
```

Из Макета получаем области:

```
ОблШапка = Макет.ПолучитьОбласть("Шапка");
ОблЭлемент = Макет.ПолучитьОбласть("Элемент");
```

Заполняем параметр «ДатаОтчета» Шапки конкретным значением и выводим в Табличный документ:

```
ОблШапка.Параметры.ДатаОтчета = ТекущаяДата();
ТабДок.Вывести(ОблШапка);
```

В цикле заполним и выведем в табличный документ параметры области «Элементы». Т.к. в выборке и области «Элемент» параметры называются одинаково, то можно воспользоваться методом «`Заполнить()`», в котором указать источник заполнения:

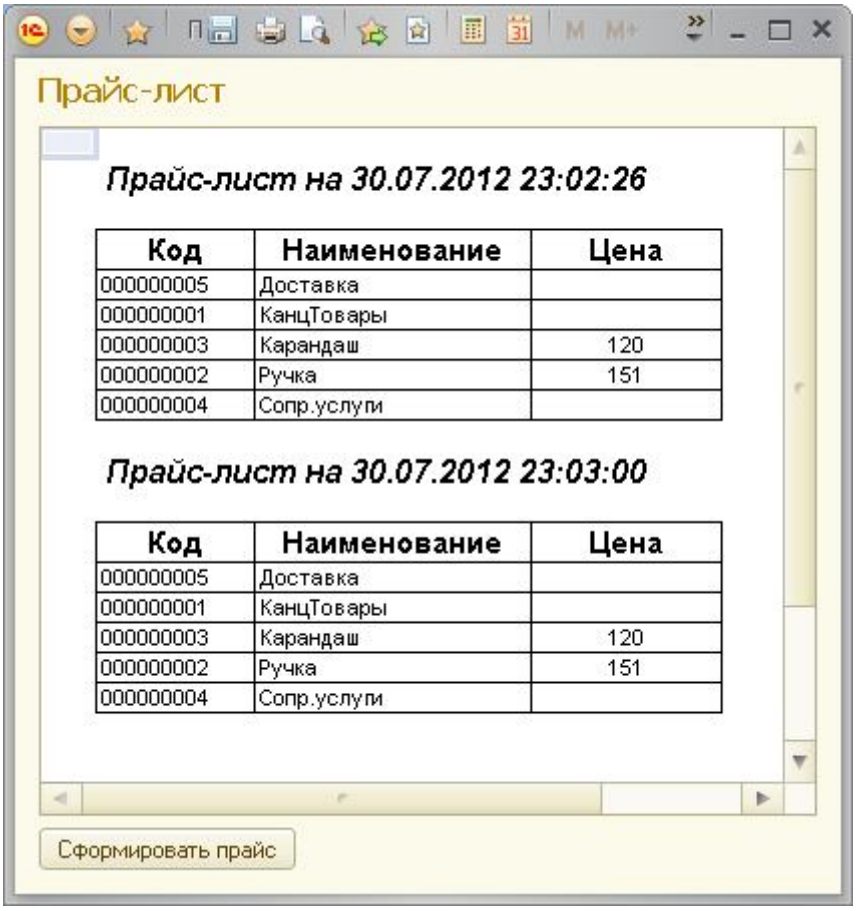
```
Пока Выборка.Следующий() Цикл
      ОблЭлемент.Параметры.Заполнить(Выборка);
      ТабДок.Вывести(ОблЭлемент);
КонецЦикла;
```

Итого:

```
&НаКлиенте
Процедура СформироватьПрайс(Команда)
      ЗаполнитьТабДок(ТабДок);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура ЗаполнитьТабДок(ТабДок)

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

      ОблШапка = Макет.ПолучитьОбласть("Шапка");
```

```
    ОблЭлемент = Макет.ПолучитьОбласть("Элемент");

    ОблШапка.Параметры.ДатаОтчета = ТекущаяДата();
    ТабДок.Вывести(ОблШапка);

    Выборка = Справочники.Номенклатура.Выбрать();
    Пока Выборка.Следующий() Цикл
        ОблЭлемент.Параметры.Заполнить(Выборка);
        ТабДок.Вывести(ОблЭлемент);
    КонецЦикла;
```

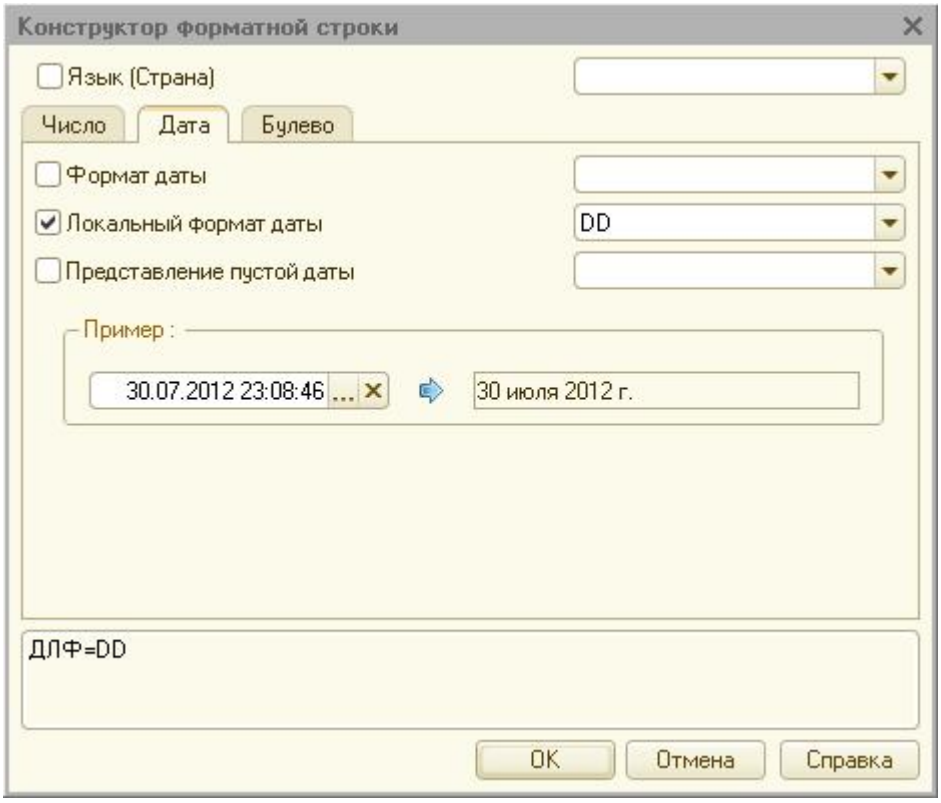
КонецПроцедуры



Недостатки:
1. При повторном нажатии на кнопку прайс дублируется. Перед повторным заполнением будем очищать Табличный документ:

```
ТабДок.Очистить();
```

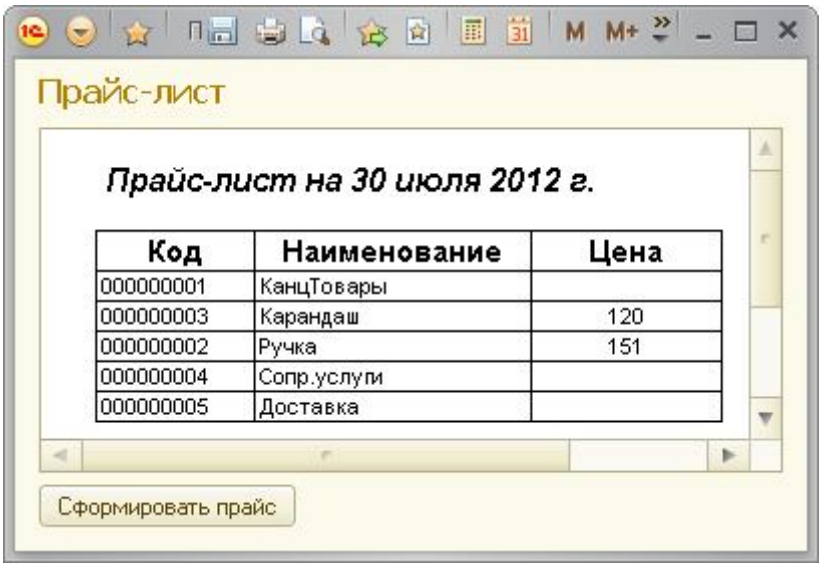
2. Дата выводится с минутами и секундами. Воспользуемся функцией «Формат». Форматную строку зададим конструктором: «ПКМ – Конструктор форматной строки»




```
ОблШапка.Параметры.ДатаОтчета = Формат(ТекущаяДата(), "ДЛФ=DD");
```

3. Данные не отсортированы по группам. Вместо метода «`Выбрать()`» воспользуемся методом «`ВыбратьИерархически()`»

```
Выборка = Справочники.Номенклатура.ВыбратьИерархически();
```



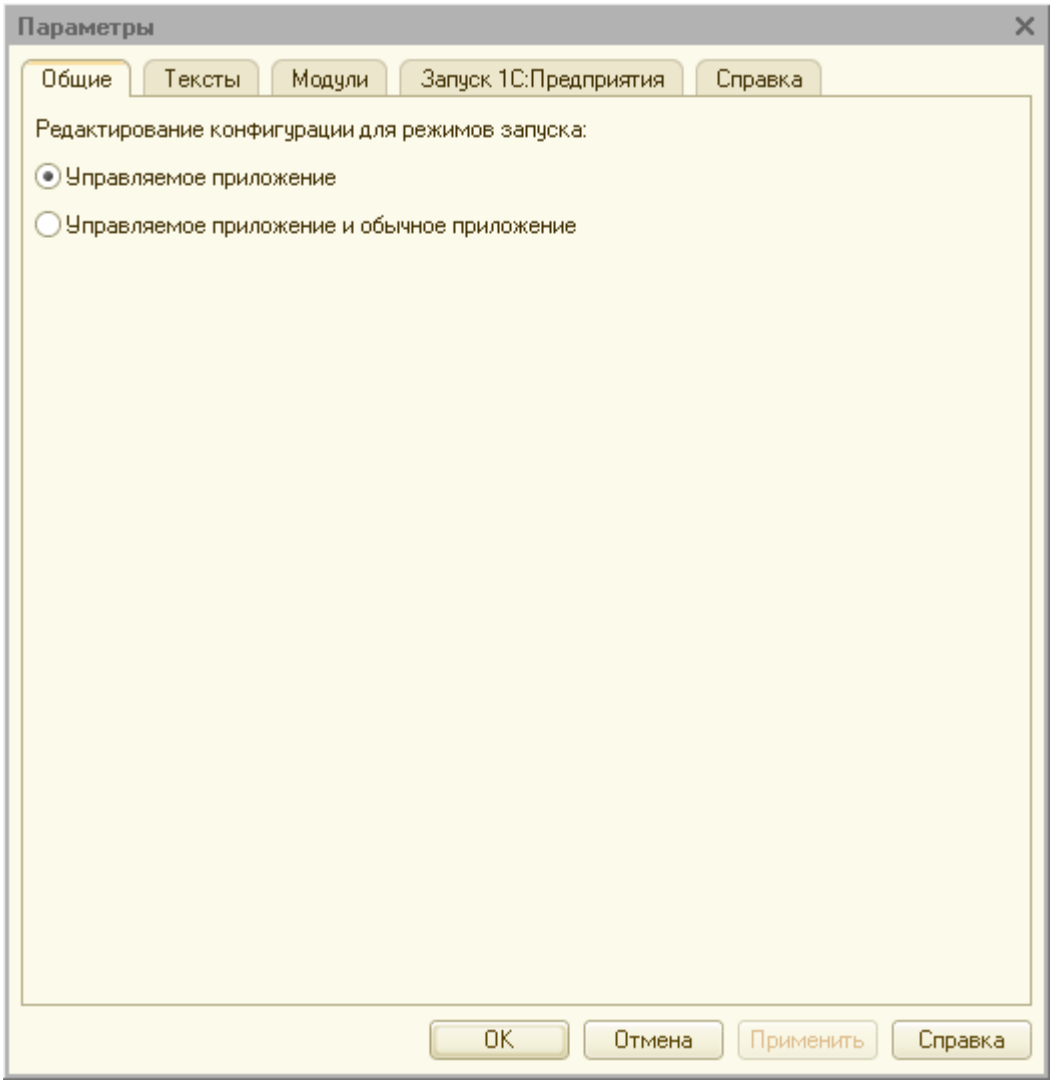

```
&НаКлиенте
Процедура СформироватьПрайс(Команда)
    ЗаполнитьТабДок(ТабДок);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура ЗаполнитьТабДок(ТабДок)
```

```
    ТабДок.Очистить();

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

    ОблШапка = Макет.ПолучитьОбласть("Шапка");
    ОблЭлемент = Макет.ПолучитьОбласть("Элемент");

    ОблШапка.Параметры.ДатаОтчета = Формат(ТекущаяДата(), "ДЛФ=DD");
    ТабДок.Вывести(ОблШапка);

    Выборка = Справочники.Номенклатура.ВыбратьИерархически();
    Пока Выборка.Следующий() Цикл
         ОблЭлемент.Параметры.Заполнить(Выборка);
         ТабДок.Вывести(ОблЭлемент);
    КонецЦикла;
```

КонецПроцедуры

Занятие 1.06

Если конфигурация создается под 8.2, то в ней уже отключены свойства, связанные с формами Толстого клиента. Но если надо будет конвертировать базу с версии 8.1 или потребуется режим Толстого клиента, обычные формы:
«Сервис – Параметры - Общие»



«Управляемое приложение и обычное приложение» - тогда у конфигурации появятся свойства:
«Использовать управляемые формы в обычном приложении»
«Использовать обычные формы в управляемом приложении»

Позволяют открывать обычные формы в толстом клиенте в режиме эмуляции тонкого клиента и наоборот.

Сделаем, чтобы группы выводились жирным курсивом и отбор вывода только товаров без услуг.

В форме отчета добавим реквизит:
Имя – «ПечататьТолькоТовары».
Заголовок – «Печатать только товары».

Тип – «Булево».

Реквизит разместим в группе. Свойства группы:
Отображение – «Нет».
ОтображатьЗаголовок – «Ложь».

Способ 1:
«Отбор» - один из параметров метода «ВыбратьИерархически()». Отбор можно установить на реквизиты «Код», «Наименование» и те, у которых свойство:
Индексировать – «Индексировать с доп. упорядочиванием» - система будет производить индексирование с символьным упорядочиванием значения, т.е. не только по значению, а еще и по символьному представлению значения.

Установим это свойство реквизиту «Услуга» и проведем отбор по нему:

```
Товары = Новый Структура("Услуга");
Товары.Вставить("Услуга", НЕ ПечататьТолькоТовары);

Если ПечататьТолькоТовары Тогда
     Выборка = Справочники.Номенклатура.ВыбратьИерархически( , , Товары);
Иначе
     Выборка = Справочники.Номенклатура.ВыбратьИерархически();
КонецЕсли;

Пока Выборка.Следующий() Цикл
     ...
КонецЦикла;
```

Способ 2:
Если элемент выборки является услугой и стоит галочка «Печатать только товары», то не выводим элемент, а идем на следующий круг цикла.

```
Пока Выборка.Следующий() Цикл
     Если Выборка.Услуга И ПечататьТолькоТовары Тогда
          Продолжить;
     КонецЕсли;
КонецЦикла;
```

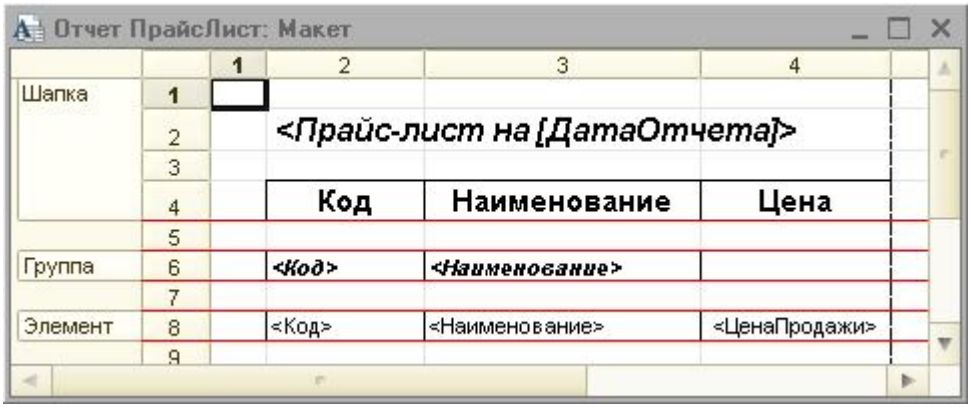
т.к. процедура без контекста, то Сервер не понимает, что нарисовано в форме, поэтому в качестве второго параметра передадим значение реквизита формы «ПечататьТолькоТовары»:

```
&НаКлиенте
Процедура СформироватьПрайс(Команда)
     ЗаполнитьТабДок(ТабДок, ПечататьТолькоТовары);
КонецПроцедуры

&НаСервереБезКонтекста
```

```
Процедура ЗаполнитьТабДок(ТабДок, ПечататьТолькоТовары)
     ...
КонецПроцедуры
```

В нашем примере первый способ не совсем корректен, т.к. в нем, если группа не является услугой, то система проигнорирует и группу и все подчиненные ей элементы. А в группе с признаком «Услуга – Ложь» все-таки могут располагаться элементы с признаком «Услуга - Истина».

Чтобы отобразить группу жирным, можно нарисовать строку для групп в макете и заполнять строку группы или элемента в зависимости от типа текущего элемента выборки в цикле:




```
Пока Выборка.Следующий() Цикл
     ...
     Если Выборка.ЭтоГруппа Тогда
          Обл = ОблГруппа;
     Иначе
          Обл = ОблЭлемент;
     КонецЕсли;
     Обл.Параметры.Заполнить(Выборка);
     ТабДок.Вывести(Обл);
КонецЦикла;
```

А можно сделать это программно, не добавляя новой строки в макет:
«Шрифт» - это переменная, которая присваивается области макета.

```
Пока Выборка.Следующий() Цикл
     ...
     ОблЭлемент.Параметры.Заполнить(Выборка);
     ЖирныйШрифт = Новый Шрифт(,, Истина, Истина);
     ОбычныйШрифт = Новый Шрифт(,, Ложь, Ложь);

     ОблЭлементТекст = ОблЭлемент.Область("R1C2:R1C4");

     Если Выборка.ЭтоГруппа Тогда
          ОблЭлементТекст.Шрифт = ЖирныйШрифт;
     Иначе
          ОблЭлементТекст.Шрифт = ОбычныйШрифт;
```

```
КонецЕсли;
    ТабДок.Вывести(ОблЭлемент);
```

Лучше создать для групп дополнительную область в макете, т.к. это менее трудоемко, чем задавать области в коде.

Итого:

```
Процедура ЗаполнитьТабДок(ТабДок, ПечататьТолькоТовары)

    ТабДок.Очистить();

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

    ОблШапка = Макет.ПолучитьОбласть("Шапка");
    ОблЭлемент = Макет.ПолучитьОбласть("Элемент");
    ОблГруппа = Макет.ПолучитьОбласть("Группа");

    ОблШапка.Параметры.ДатаОтчета = Формат(ТекущаяДата(), "ДЛФ=DD");
    ТабДок.Вывести(ОблШапка);

    Пока Выборка.Следующий() Цикл

        Если Выборка.Услуга И ПечататьТолькоТовары Тогда
            Продолжить;
        КонецЕсли;

        ОблЭлемент.Параметры.Заполнить(Выборка);

        Если Выборка.ЭтоГруппа Тогда
            Обл = ОблГруппа;
        Иначе
            Обл = ОблЭлемент;
        КонецЕсли;
        Обл.Параметры.Заполнить(Выборка);
        ТабДок.Вывести(Обл);
    КонецЦикла;

КонецПроцедуры
```

Задача: сохранять справочные значения в виде истории. В примере – информацию о курсах валют.
Саму валюту будем хранить в справочнике:
Имя – Валюты.
Синоним – Валюты.
Представление объекта – Валюта.

Для хранения истории нельзя использовать реквизиты справочника, т.к. в них сохраняется только последнее значение.

Регистры сведений – хранят показатели учета:
Состояния.
Истории – может быть не только числовое значение, а, например, перечисление (например, метод списания – FIFO или LIFO, настроение – хорошее, плохое и т.д.). Если показатель характеризуется не числом, то это признак того, что его надо хранить в регистре сведений.

Какая таблица в системе должна быть создана для хранения курсов валют:

Ресурс – то, что храним в регистре.
Измерения – показатели, в разрезе которых хранится ресурс.
Периодичность регистра - если в разрезе измерения значение ресурса может с течением времени меняться.

Пример 1.Покупаем пластиковые игрушки разных цветов. Надо знать сколько игрушек определенного цвета есть:

Ресурс – цвета.
Измерение – игрушки.
А так как с течением времени цвет игрушки не меняется, то регистр будет «Непериодический».

Пример 2. Уровни доверия к контрагенту. Доверие с течением времени может изменяться например, от «не проверенный» до «надежный». Интересно сохранять историю данного показателя, значит регистр будет периодическим и у него будет поле «Дата».

Ресурс – надежность.
Измерение – контрагент.

Дата	**Валюта**	**Курс**
Периодичность регистра	Разрез измерения	Ресурс

Создадим регистр сведений:
Имя – КурсыВалют.
Синоним – Курсы валют.
Периодичность – В пределах дня.

Режим записи:
- Независимый – можно вручную изменять значение ресурса. Есть только у регистра сведений. У остальных только значение «Подчинение регистратору».
- Подчинение регистратору – формирование движений по регистру вручную невозможно.

В примере:
Режим записи – Независимый.

Основной отбор по периоду – будет ли в основной индекс включено поле «Период». Позволяет быстро искать значения в истории. Если чаще надо получать значения на настоящее время (последнее значение регистра), то галочку можно не ставить.

В примере:
Основной отбор по периоду – Истина.

Закладка «Данные»:
Измерения:
Валюта

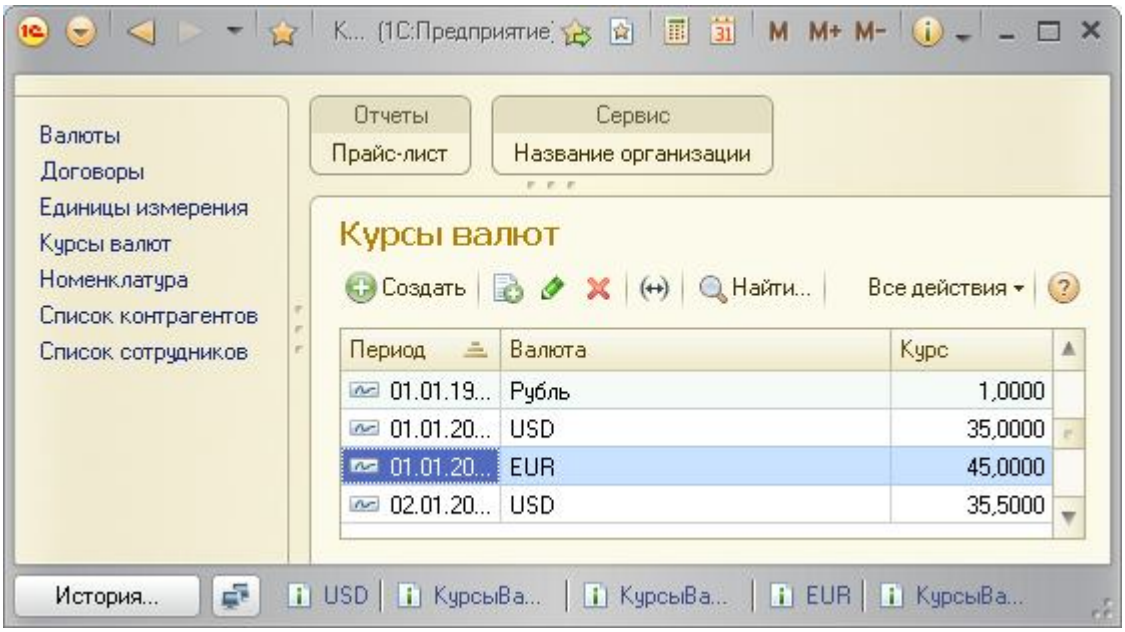
Свойство	Значение
Имя	Валюта
Синоним	Валюта
Тип	СправочникСсылка.Валюты
Ведущее	Истина
Основной отбор	Истина
Запрет незаполненных значений	Истина

- Запрет незаполненных значений – запрет записи пустых значений в регистр.
- Основной отбор – поле «Валюта» попадет в основной отбор. Если «Основной отбор по периоду - Истина» и «Основной отбор - Истина», то в истории можно быстро получать значения в сочетании «Дата - Валюта».
- Ведущее – сами значения, указанные в этом измерении владеют записями этого регистра. Допустим, при попытке удалить значение «Доллар» из справочника «Валюты»:
  - ✓ Если «Ведущее - Истина», то доллар удалится из справочника, а в регистре очистится история по нему.
  - ✓ Если «Ведущее - Ложь», то при система не даст удалить доллар из справочника, т.к. на него ссылаются записи регистра сведений, т.е. придется сначала очищать регистр, а потом уже удалять элемент справочника.

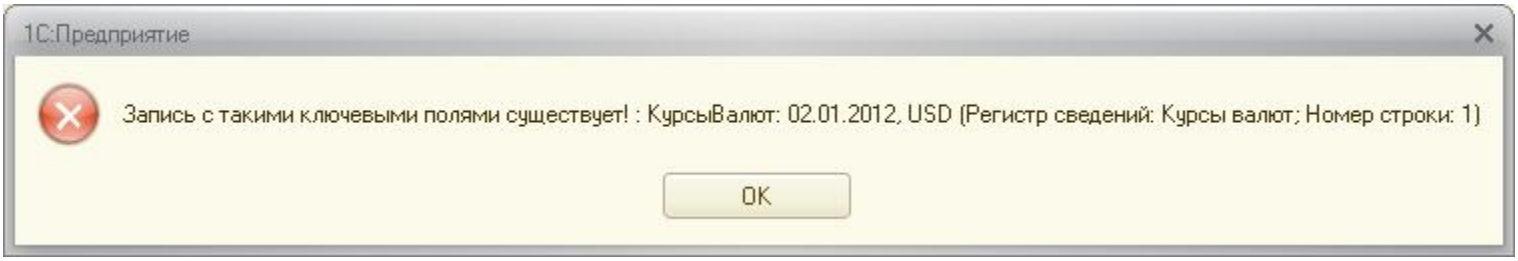
Ресурсы:
Курс

Свойство	Значение
Имя	Курс
Синоним	Курс
Тип	Число 10,4

В режиме исполнения создадим несколько записей с курсами валют:

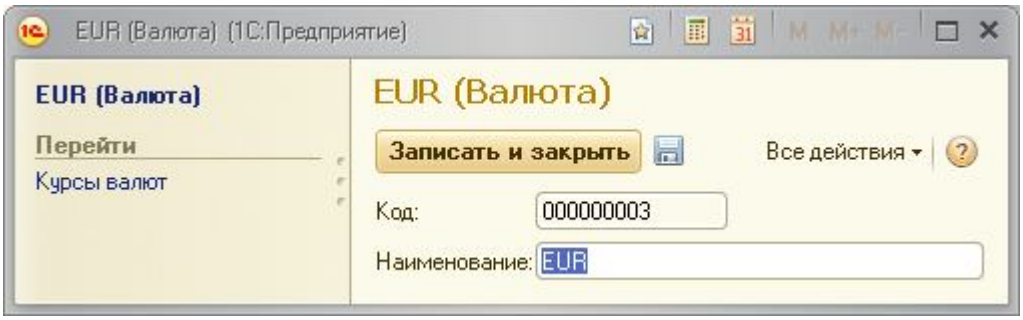


При попытке создать еще одну запись (с другим курсом) валюты USD на 02.01.12, система выдаст ошибку:

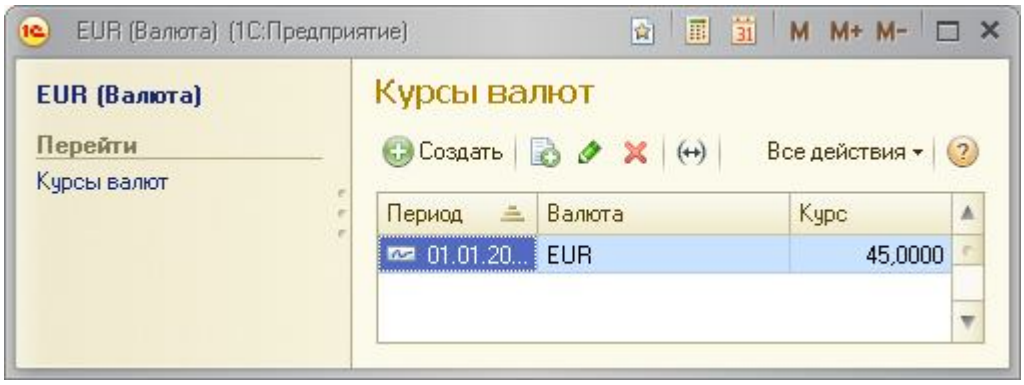


т.к. невозможно в регистр сведений записать значение с одинаковыми ключевыми полями (Период и Валюта)

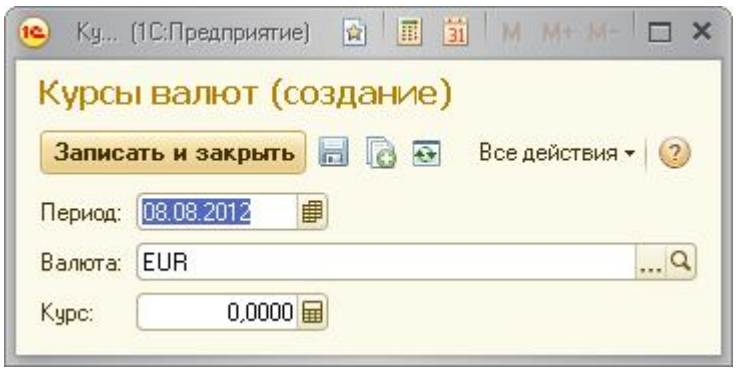
В справочнике «Валюты» есть пункт «Перейти»:



При переходе по ссылке, откроется таблица регистра с установленным предопределенным отбором по валюте:



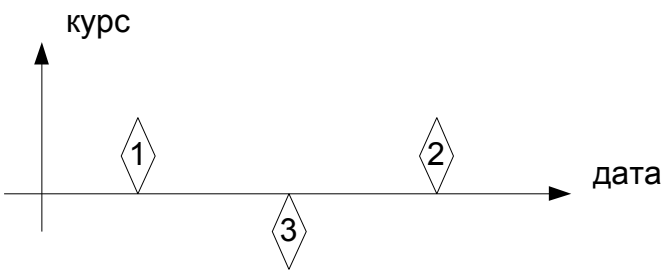
При создании новых элементов EUR будет автоматом браться из данных заполнения, т.к. отбор является данными заполнения:



Неотрицательное – лучше оставлять «Ложь», т.к. если будет введено отрицательное значение, то система не выдаст ошибки, а просто заменит его нулем, и на этапе отладки найти это значение будет тяжело. А когда уже все отлажено, то можно поставить «Истина».

Основное назначение регистра «Курсы валют» - получить значение курса определенной валюты на определенную дату.

У регистров сведений есть различные методы.
Допустим, установим курсы валюты на одну дату (1) и на другую дату (2), а надо найти курс на еще одну дату (3).
«Получить» - получим значение только на указанную дату (3) (если оно есть).
«ПолучитьПервое» - получим значение либо на указанную дату (3), либо на первое, которое встретится в будущем (2).
«ПолучитьПоследнее» - получим значение либо на указанную дату (3), либо на ближайшее значение в прошлом (1).



Т.к. курс валют используется в разных местах, то опишем универсальную процедуру (в общем модуле «ОбщиеМеханизмы»):

Валюта – валюта, курс которой получаем.
Дата – дата, на которую получаем курс.
«`Дата = Неопределено`» - дифференцированное значение – примет параметр, даже если он не будет передан.

Система получает параметр «Дата» по ссылке, опишем его как значение – «`Знач Дата`», чтобы разорвать связь с точкой вызова.

«`ПолучитьПоследнее(<КонецПериода>, <Отбор>)`» - у параметра «Отбор» тип – «Структура»:
Ключ – «Валюта».
Значение – «Валюта».

```
Функция ПолучитьКурсВалюты(Валюта, Знач Дата = Неопределено) Экспорт
    Дата = ?(Дата = Неопределено, ТекущаяДата(), Дата);

    Структура = Новый Структура;
    Структура.Вставить("Валюта", Валюта);

    Данные = РегистрыСведений.КурсыВалют.ПолучитьПоследнее(Дата, Структура);

    Возврат Данные.Курс;

КонецФункции
```

«`Возврат` **`Данные.Курс`**» - знаем, что в данных есть курс т.к. метод «ПолучитьПоследнее» возвращает значение ресурса регистра сведений.

Если бы в регистре было два измерения, то надо было бы написать две строки структуры для отбора.

Пример:
В справочник «Контрагенты» добавим реквизит «Валюта взаиморасчетов» и при редактировании элемента справочника будем показывать текущий курс.

Реквизит «Валюта взаиморасчетов»:

Свойство	Значение
Имя	ВалютаВзаиморасчетов
Синоним	Валюта взаиморасчетов
Тип	СправочникСсылка.Валюты

В форме опишем поведение программы при выборе валюты взаиморасчетов:

Перетащим реквизит в форму. Добавим реквизит, не связанный с объектом:
Реквизит «Курс»

Свойство	Значение
Имя	Курс
Синоним	Курс
Тип	Число 10, 4

Перетащим его в форму и назначим свойство:

Свойство	Значение
Вид	Поле надписи

чтобы пользователь не смог его поменять.

Моменты, когда курс должен отображаться:
1. Открытие формы существующего элемента, где валюта уже указана.
Процедура формы «ПриСозданииНаСервере()» - вызывается при создании формы на сервере, т.е. до отправки формы на Клиента.

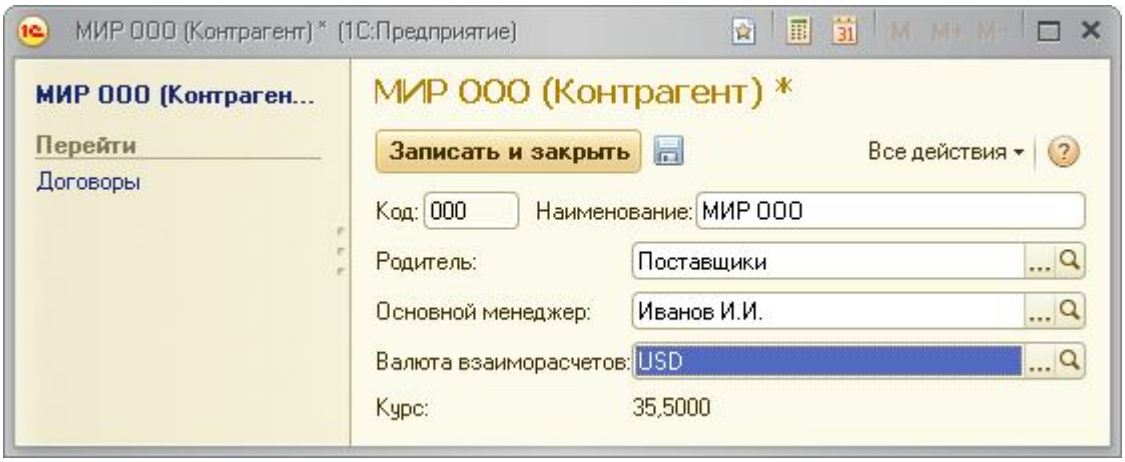
2. При смене валюты взаиморасчетов.
Процедура реквизита «ВалютаВзаиморасчетов» «ПриИзменении()».

Одна процедура выполняется на Клиенте, а другая на Сервере, тогда процедура с логикой должна выполняться одновременно на Сервере и на Клиенте, т.е. вызвать ее можно будет и с сервера и с клиента:

```
&НаКлиентеНаСервереБезКонтекста
Процедура ОбновитьКурс(Валюта, Курс)
     Если Валюта.Пустая() Тогда
          Курс = 0;
     Иначе
          Курс = ОбщиеМеханизмы.ПолучитьКурсВалюты(Валюта);
     КонецЕсли;
КонецПроцедуры
```

Будем вызывать эту процедуру из обеих процедур, передавая при этом как параметры значения Валюты и Курса, т.к. процедура выполняется без контекста.

```
&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
     ОбновитьКурс(Объект.ВалютаВзаиморасчетов, Курс);
КонецПроцедуры

&НаКлиенте
Процедура ВалютаВзаиморасчетовПриИзменении(Элемент)
     ОбновитьКурс(Объект.ВалютаВзаиморасчетов, Курс);
КонецПроцедуры
```


МИР ООО (Контрагент) * (1С:Предприятие)
МИР ООО (Контраген...
Перейти
Договоры
МИР ООО (Контрагент) *
Записать и закрыть
Все действия
Код: 000
Наименование: МИР ООО
Родитель: Поставщики
Основной менеджер: Иванов И.И.
Валюта взаиморасчетов: USD
Курс: 35,5000

Занятие 1.07

Задача: накапливать сведения о номенклатуре (например, свойства). Конфигурация должна быть универсальной, т.е. подходить для разных сфер деятельности.

Номенклатура – это справочник.
Виды свойств – это виды значений аналитики для номенклатуры. Справочник.
Свойства – это справочник, подчиненный справочнику «Виды свойств».

Мы создали справочники для хранения отдельных видов информации. Соединение данной информации может храниться:

1 Табличная часть справочника «Номенклатура»:

Добавить табличную часть «Свойства» и реквизиты:

- ВидСвойства.
- Значение.

Но в табличной части не контролируется уникальность реквизита «ВидСвойства», т.е. пользователь может ввести для одного элемента номенклатуры одновременно:

Цвет – Красный.
Цвет – Белый.

А это в нашем примере не совсем правильно.

2 Регистр сведений:

Плюс в том, что измерения будут содержать ключ уникальности, т.е. если сделать «Номенклатура» и «Вид» измерениями, то пользователь не сможет ввести два разных Значения по ключу измерений «Номенклатура» и «ВидСвойства».

«Регистр сведений» удобно использовать, когда надо хранить Виды свойств не только элементов одного справочника, а нескольких. Для этого тип Измерения «Объект аналитики» делается составным.

Допустим, создадим еще один Вид свойств «Аналог» (аналог колбасы - фрикадельки), но и колбаса и фрикадельки будут Номенклатурой. Получается, что в справочник «Свойства» надо будет вводить дублирующуюся информацию (фрикадельки будут и в Номенклатуре и в Свойствах) и элемент «Фрикадельки» справочника «Свойства», не будет иметь ничего общего с элементом «Фрикадельки» справочника «Номенклатура».

Решение: надо сделать тип колонки «Значение» Регистра сведений составным, чтобы можно было брать информацию из различных справочников.

Чтобы автоматизировать ввод Значения в зависимости от выбранного Вида свойств:

- если Цвет, то Красный, Белый...
- если Размер, то XXL, M...

Данную информацию лучше хранить в справочнике «Виды свойств»:

Виды свойств	
Наименование	**ТипЗначения**
Цвет	Спр. Свойства
Размер	Спр. Свойства
Страна происхождения	Спр. Свойства
Основной поставщик	Спр. Контрагенты
Аналог	Спр. Номенклатура

И пользователь сможет ввести в данную таблицу новые виды свойств (свои), указать, из какого справочника их брать, или это будет Число, Строка и т.д.

Т.е. тип значений в колонке «Значение» регистра сведений будет составной. Выбор значения будет зависеть от Вида свойства. В Виде свойства описано, каким типом значения свойство характеризуется и программно можно установить, что, например, при выборе Вида свойства «Цвет», значения надо выбирать из «Красный» или «Белый».

Надо иметь возможность выбирать свойства не только в регистре сведений. Например, надо создать отчет по всем красным товарам. Тогда в отчете обязательно должны быть две колонки:

- Вид свойства.
- Значение.

И тип поля Значение должен быть тоже составной и состоять он будет из указанных справочников.

Если при поступлении товаров надо будет дать пользователю возможность указывать свойства товаров, то там тоже нужно будет создать поле Значение составного типа, с указанием тех же справочников.

Такая реализация не ошибочна, но при этом надо постоянно помнить где и какие справочники выбраны.

Чтобы этого избежать, предназначен «План видов характеристик» - это аналог справочника с предопределенным полем «ТипЗначения». Хранит названия видов характеристик с описанием типов их значений. И все его типы значений образуют новый тип значения – «Характеристика» - является составным типом, который ссылается на все справочники, которые указаны в свойстве «Тип значения характеристик» плана видов характеристик (коллекция типов значений).

Во всех полях, где будет возможность выбора значения (Красный, Белый, США, Китай...) будем указывать тип значения – определенная характеристика. При этом не надо каждый раз выбирать нужные составляющие типы значения.

Реализация:

Создадим план видов характеристик «Виды свойств номенклатуры»:

Свойство	Значение
Имя	ВидыСвойствНоменклатуры
Синоним	Виды свойств номенклатуры
Тип значения характеристик	СправочникСсылка.Договоры, СправочникСсылка.Сотрудники, СправочникСсылка.ЕдиницыИзмерения, Булево, СправочникСсылка.Контрагенты, СправочникСсылка.ДопСвойстваНоменклатуры, Строка, СправочникСсылка.Валюты, Дата, Число, СправочникСсылка.Номенклатура
Дополнительные значения характеристик	ДопСвойстваНоменклатуры

Т.е. виды свойств номенклатуры пользователь может описывать всеми типами значений конфигурации.

Количество выбранных типов значения характеристик не влияет на скорость работы системы и размер БД.

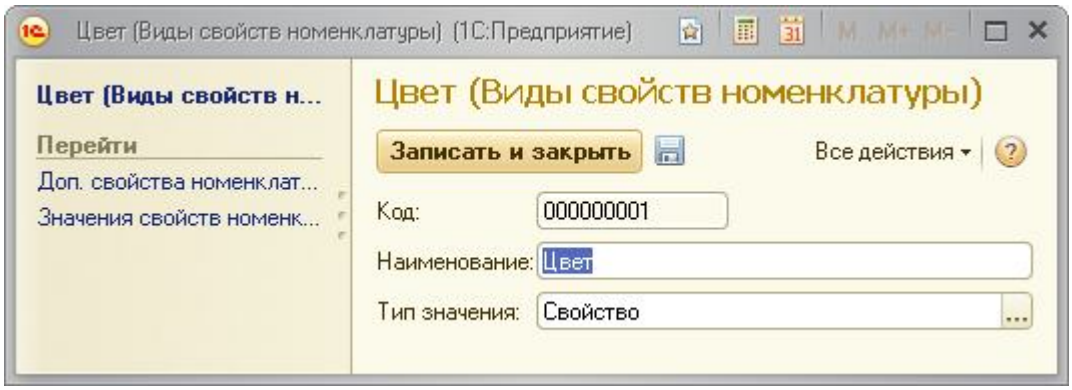
Создадим справочник «Доп. свойства номенклатуры» - будет хранить информацию о дополнительных свойствах, которых нет в существующих справочниках. Будет подчинен «Плану видов характеристик»:

Свойство	Значение
Имя	ДопСвойстваНоменклатуры
Синоним	Доп. свойства номенклатуры
Представление объекта	Свойство
Закладка Владельцы. Список владельцев справочника	ПланВидовХарактеристик.ВидыСвойствНоменклатуры

В Плане видов характеристик в Типы значений характеристик добавим данный справочник – т.е. характеристика может выбираться из значений элементов данного справочника.

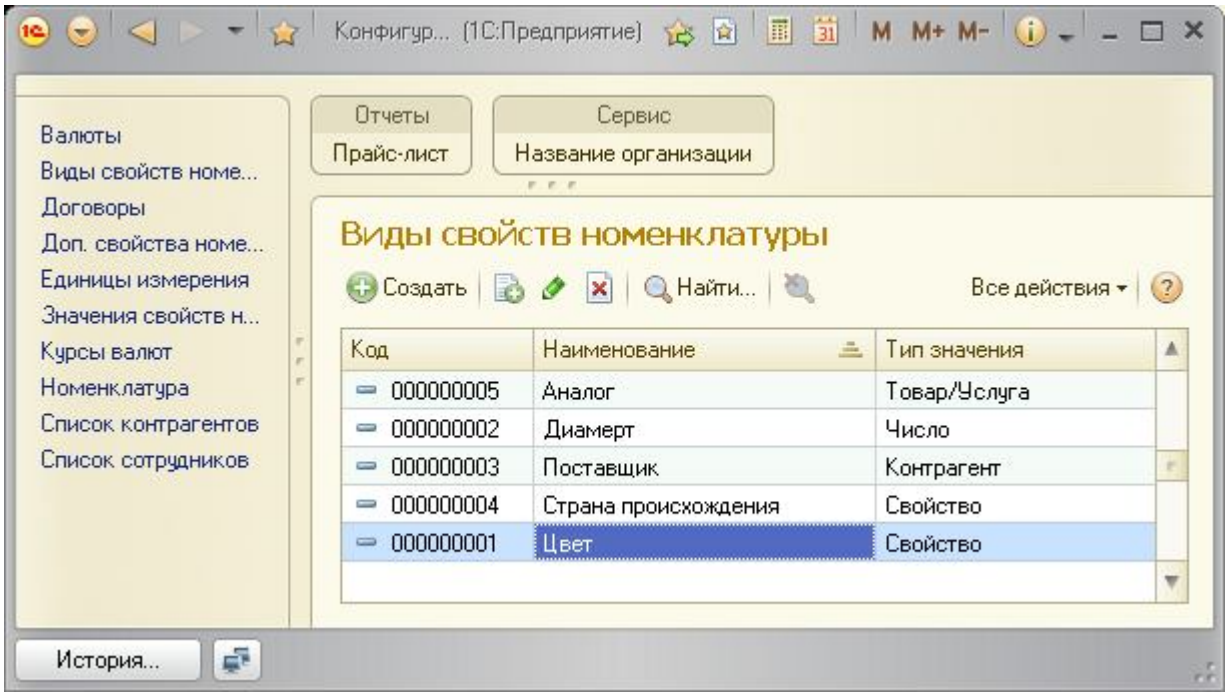
В режиме исполнения:

Введем несколько видов свойств номенклатуры:

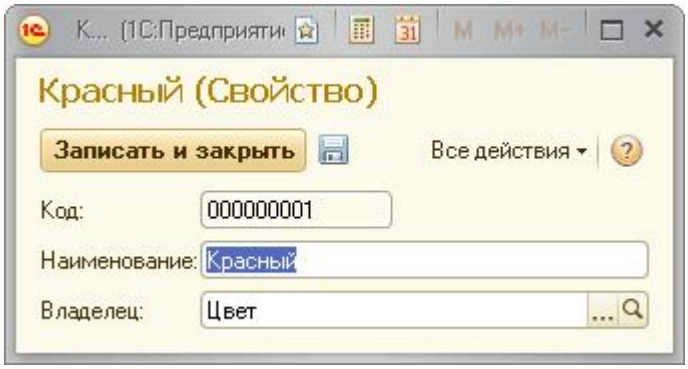


Чтобы «Тип значения» по умолчанию ставился «Свойство», надо в конфигураторе задать свойство Плана видов характеристик:
«Дополнительные значения характеристик – ДопСвойстваНоменклатуры (справочник)»

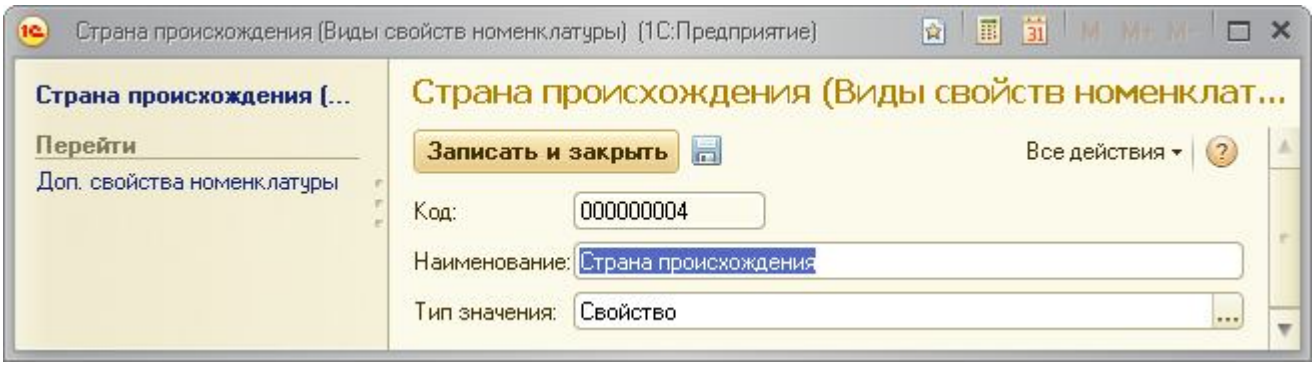
и т.д.

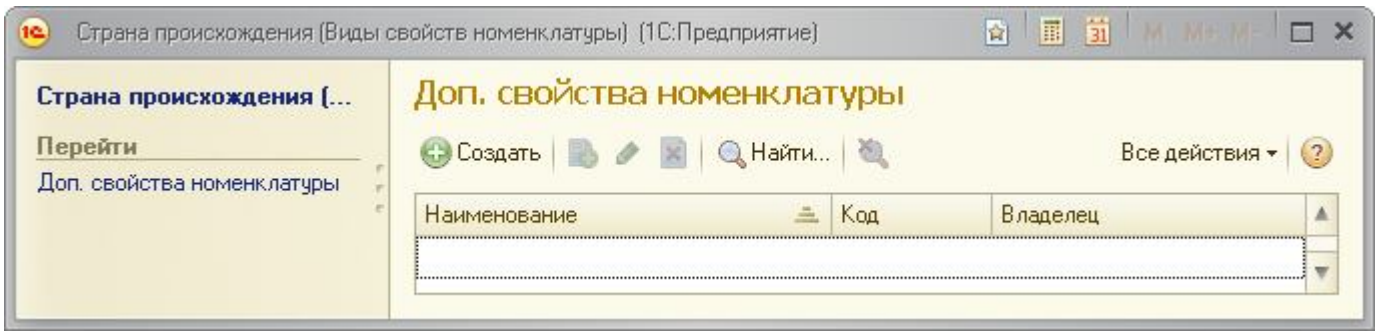


Зададим несколько значений свойств (Доп. свойства номенклатуры):

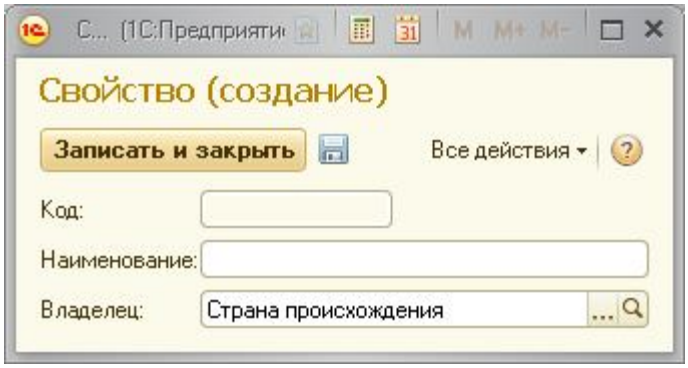


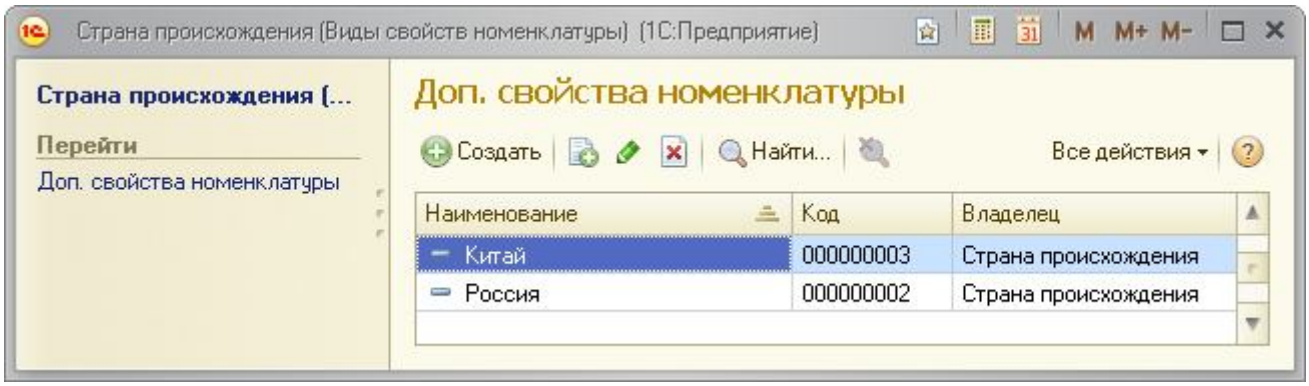
Также значения свойств можно задавать, переходя из Плана видов характеристик «Виды свойств номенклатуры» («Перейти»)





Здесь, при создании элемента, значение владельца устанавливается по умолчанию:





Создадим Регистр сведений «Значения свойств номенклатуры»

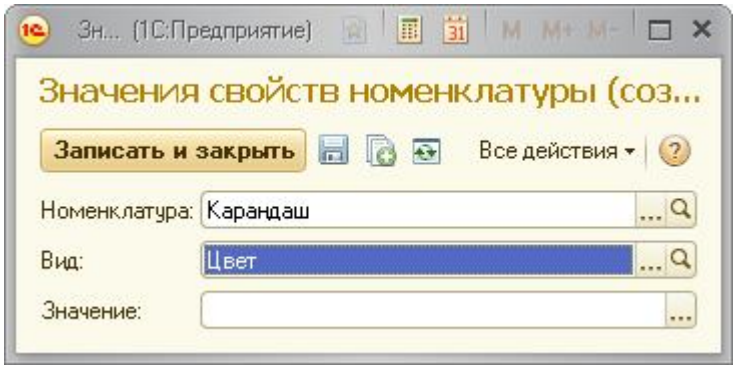
Свойство	**Значение**
Имя	ЗначенияСвойствНоменклатуры
Синоним	Значения свойств номенклатуры
Закладка «Данные»	
Измерения	
Номенклатура	
Имя	Номенклатура

Свойство	Значение
Закладка «Данные»	
Измерения	
Номенклатура	
Синоним	Номенклатура
Тип	СправочникСсылка.Номенклатура
Ведущее	Истина
Основной отбор	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Вид	
Имя	Вид
Синоним	Вид
Тип	ПланВидовХарактеристикСсылка.ВидыСвойствНоменклатуры
Ведущее	Истина
Основной отбор	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Ресурсы	
Значение	
Имя	Значение
Синоним	Значение
Тип	Характеристика.ВидыСвойствНоменклатуры

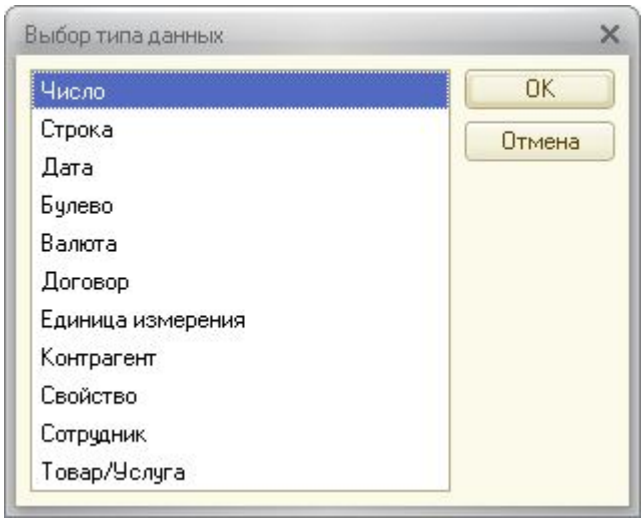
«Тип - Характеристика.ВидыСвойствНоменклатуры» - аналог составного типа из Плана видов характеристик. Отличие от составного типа – если в Плане видов характеристик увеличить количество возможных типов значений, то оно автоматически наследуется. Только для этого и используется объект «План видов характеристик»

Проверку заполнения включать не надо, т.к. если тип характеристики будет «Число» и ее значение будет «0», то система воспримет это как незаполненное поле.

В режиме исполнения:



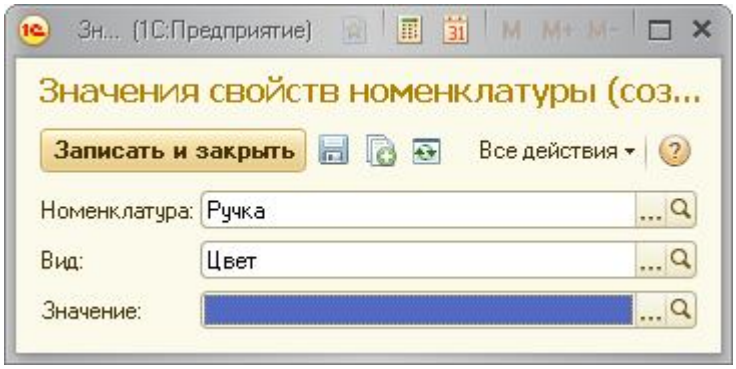
При выборе «Значения» система запросит выбор типа данных

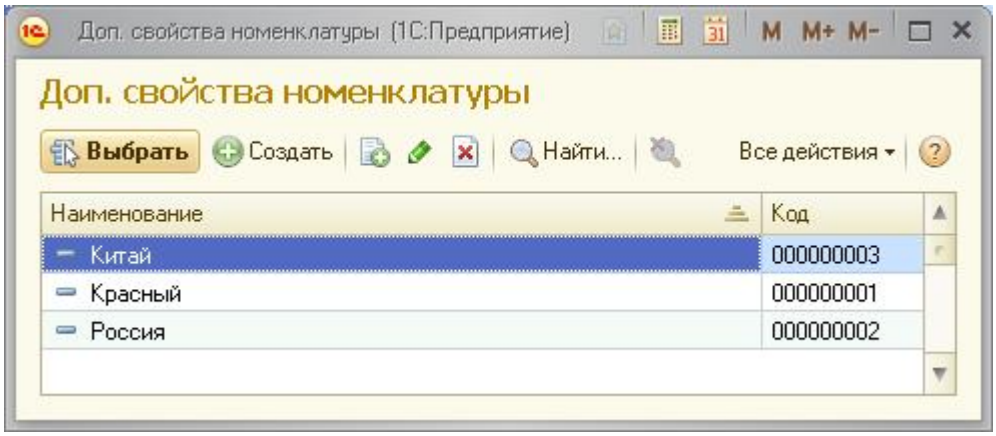


В данном случае значение надо выбирать из «Свойство», но не каждый пользователь догадается, поэтому надо организовать связь реквизитов «Вид» и «Значение»:
Свойство Ресурса «Значение»:

Свойство	Значение
Связь по типу	Вид

т.е. Тип данных «Значения» определяется по «Виду» выбранного свойства. Теперь при выборе Значения не будет появляться окно со Списком выбора типа данных.

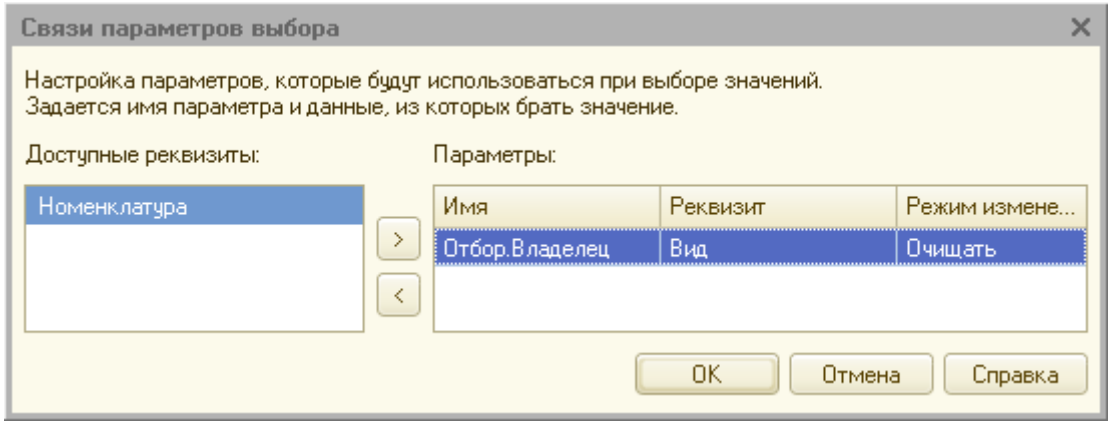




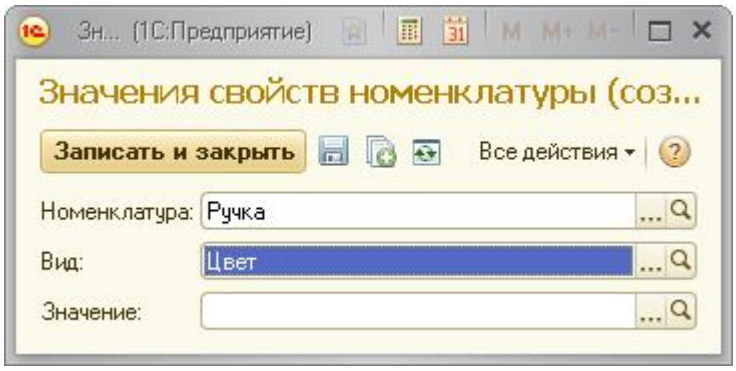
Система для выбора Значения сразу открывает справочник «ДопСвойстваНоменклатуры». Но здесь кроме цвета есть записи другого характера, т.е. отбор по типу значения реализован, а по Владельцу нет:

Свойство Ресурса «Значение»:

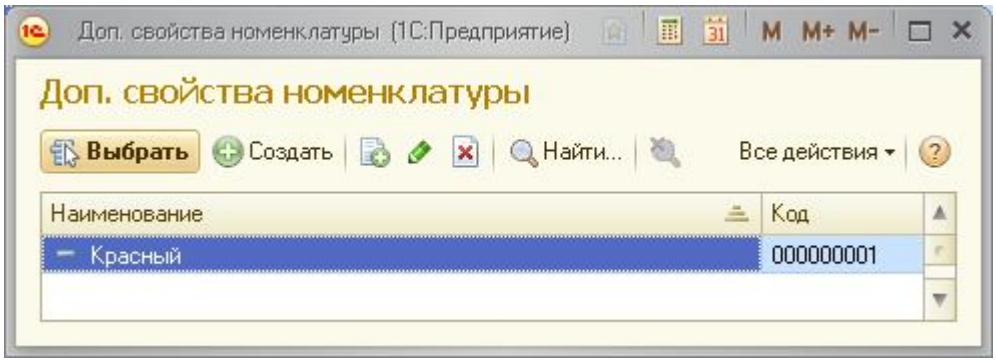
Свойство	Значение
Связи параметров выбора	Отбор.Владелец(Вид)



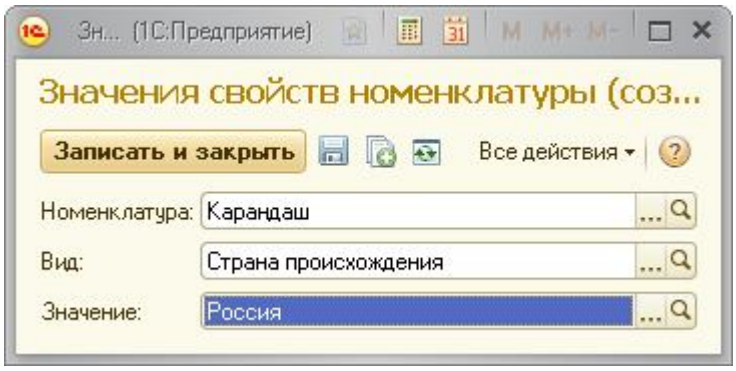
Указали, что Вид свойства является владельцем при выборе данных:

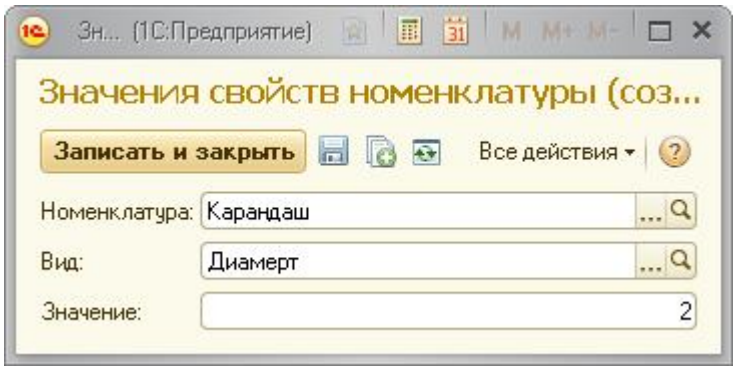


и при выборе Значения оно будет выбираться только из цветов:



Введем еще данные:


Зн... (1С:Предприятие)
Значения свойств номенклатуры (соз...
Записать и закрыть
Все действия
Номенклатура:
Карандаш
Вид:
Страна происхождения
Значение:
Россия


Зн... (1С:Предприятие)
Значения свойств номенклатуры (соз...
Записать и закрыть
Все действия
Номенклатура:
Карандаш
Вид:
Диамерт
Значение:
2

Занятие 1.08

Документы – в основном предназначены для ввода первичной информации, совпадают с реальным документом.

Основные реквизиты документа:
«Проведен» - позволяет реализовать два состояния документа:

- Записан – когда надо отразить сам факт наличия документа.
- Проведен - когда документ влияет на состояние учета.

если они необходимы.

ID (Ссылка)	Номер	Дата	ПометкаУдаления	Проведен

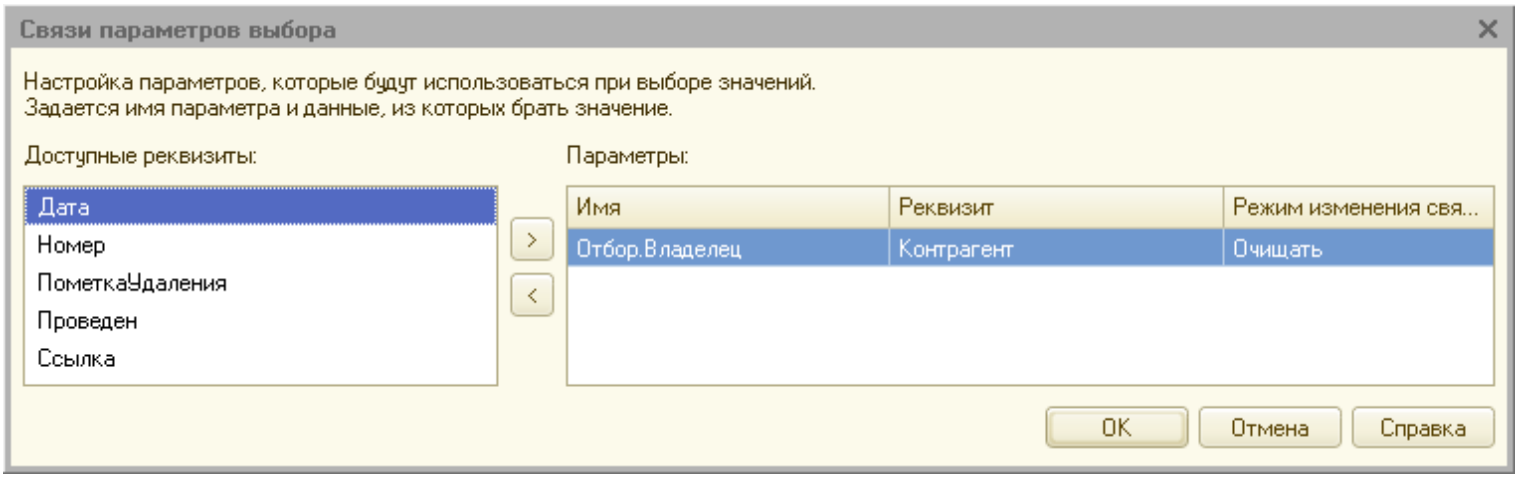
Создадим документ «Приходная»

Свойство	**Значение**
Имя	Приходная
Синоним	Приходная
Представление объекта	Поступление товаров
Представление списка	Приходные документы
Закладка «Данные»	
Реквизиты	
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Проверка заполнения	Выдавать ошибку
Договор	
Имя	Договор
Синоним	Договор
Тип	СправочникСсылка.Договоры
Проверка заполнения	Выдавать ошибку
Связи параметров выбора	Отбор.Владелец(Контрагент)
Табличные части	
Товары	
Имя	Товары
Синоним	Товары
Проверка заполнения	Выдавать ошибку
Реквизиты табличной части	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	Номенклатура
Проверка заполнения	Выдавать ошибку
Цена	
Имя	Цена
Синоним	Цена

Свойство	Значение
Табличные части	
Товары	
Реквизиты табличной части	
Цена	
Тип	Число 15, 2
Количество	
Имя	Количество
Синоним	Количество
Тип	Число 12, 3 (3 – чтобы можно было вводить граммы)
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2
Проверка заполнения	Выдавать ошибку

Т.к. важно, чтобы в документе были заполнены реквизиты «Контрагент» и «Договор», то установим им Проверку заполнения. Также установим Проверку заполнения у табличной части.

Т.к. справочник «Договоры» подчинен справочнику «Контрагенты», то, чтобы при выборе договора, открывался только список выбранного контрагента, надо в реквизите «Договор» заполнить свойство «Связи параметров выбора»:



При выборе Договора надо установить Владельца в значении реквизита Контрагент.
«Отбор.Владелец» - это свойство открываемой формы справочника «Договоры», которое установит предопределенный отбор.

Свойство	Значение
Закладка «Нумерация»	
Периодичность	В пределах года

Нумерация для одного вида документов настраивается в свойствах

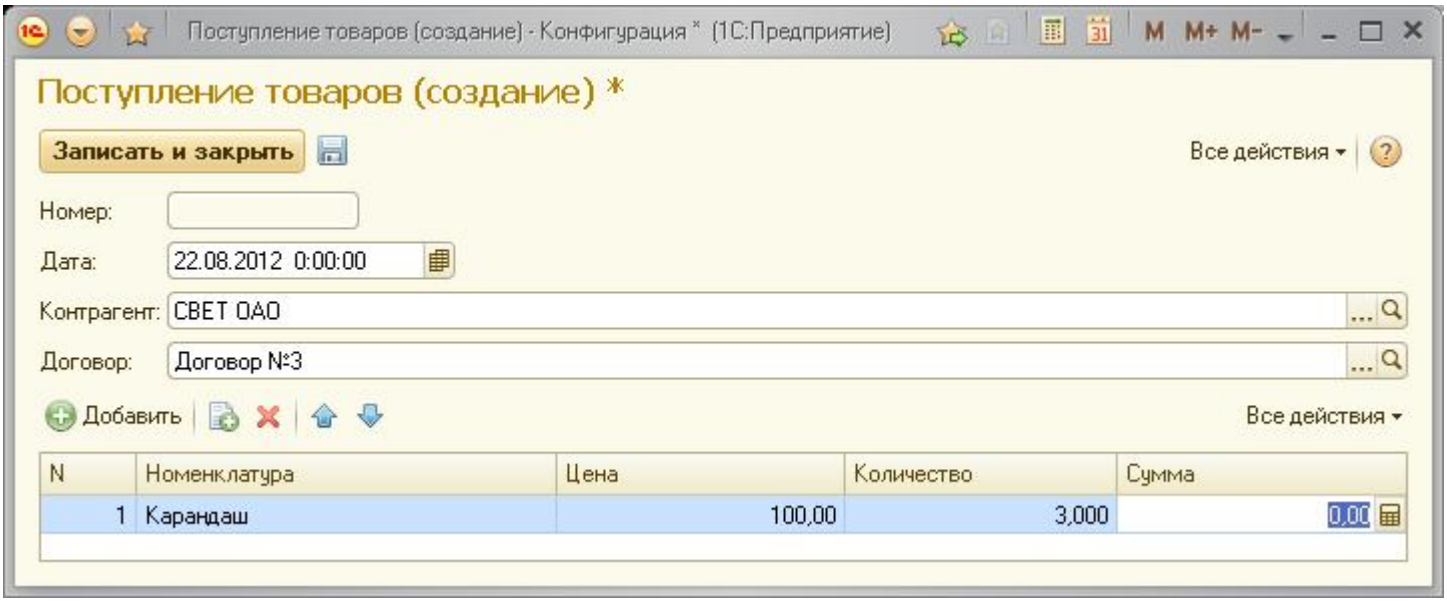
Нумератор (ветка «Документы») - если нужна нумерация между документами разных видов. Нумератор присваивают разным видам документов в свойстве «Нумератор».

Свойство	Значение
Закладка «Движения»	
Определяется, будет документ проводиться или нет	
Проведение	Запретить

Пока в конфигурации нет регистров, то проведение документов в принципе не нужно. При выключенном проведении, в режиме исполнения документы все равно будут отражаться с галочкой (как проведенные), хотя реквизит «Проведен - Ложь».

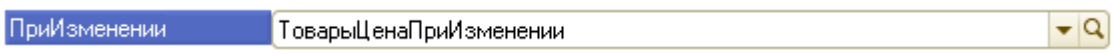
Закладка «Журналы» - список Журналов документов, в которых будет регистрироваться данный вид документа.
Журналы документов – используются для объединения документов разных видов в одну таблицу. Цель – общая индексация документов разных видов для ускорения их поиска.
Журналы документов – это дополнительная нагрузка на систему и увеличение размера БД

Закладка «Формы»
Перед созданием своих форм проверим, подходят ли нам автоматически создаваемые формы.



При вводе Цены и Количества, Сумма не посчиталась автоматически. Чтобы Сумма считалась автоматом, надо рисовать свою форму.
Форму документа создаем конструктором.
Сумма должна меняться при изменении Цены или Количества.

Опишем событие «ПриИзменении» поля «ТоварыЦена» в модуле формы:
Можно написать код здесь и связать с событием «ТоварыКоличествоПриИзменении»

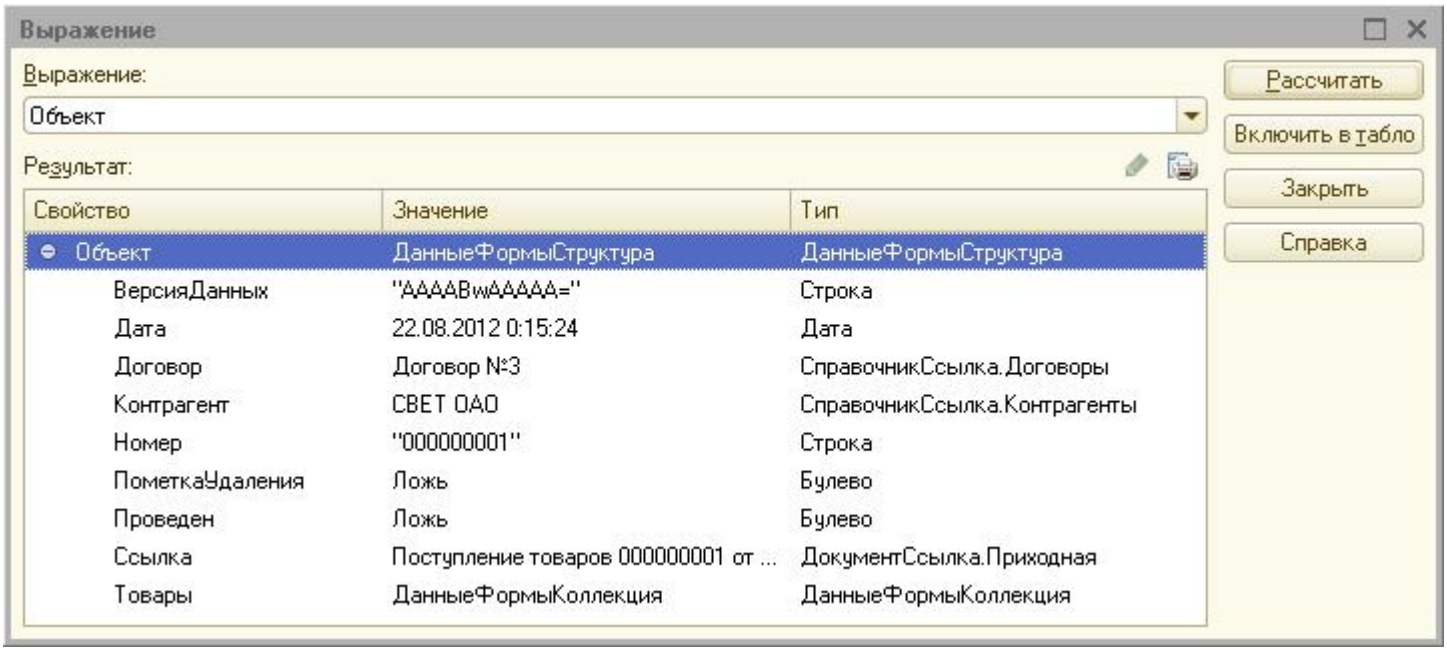


но это будет не совсем читабельно, можно запутаться, т.к. по названию процедуры будет не понятно, откуда она вызывается.

Т.к. Цена и Количество – это примитивные типы, пользователь вводит их на Клиенте, то и процедура подсчета будет выполняться на Клиенте.
Чтобы рассчитать Сумму, надо добраться до строки, с которой работает пользователь.

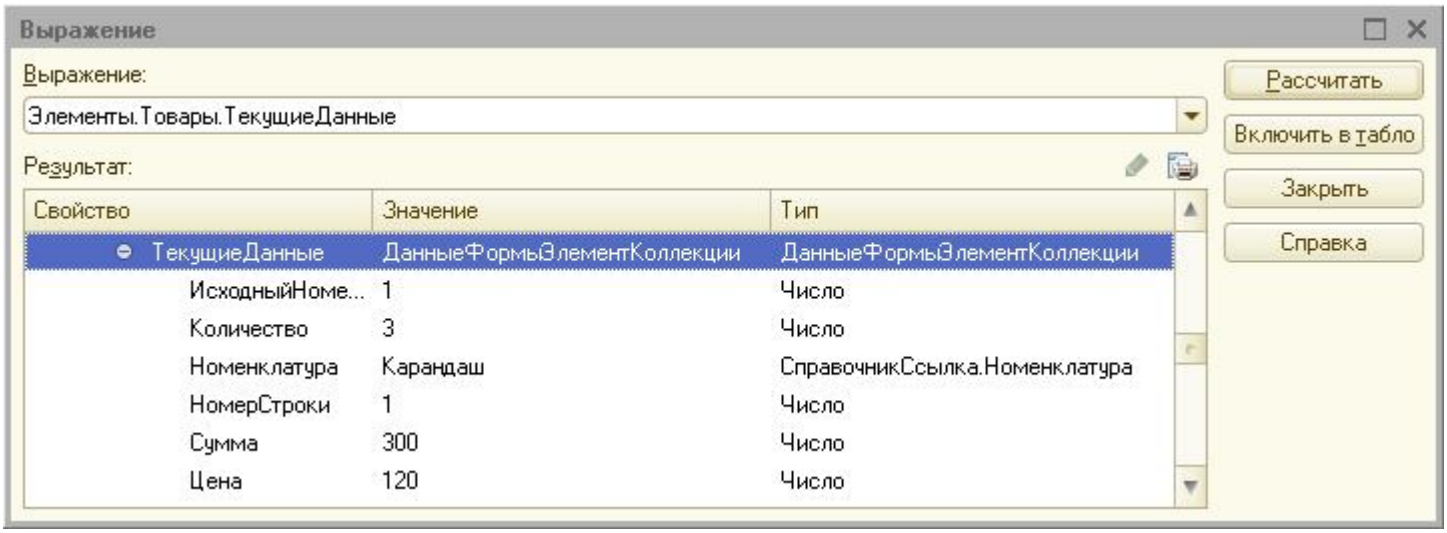
Посмотрим процедуру в режиме отладки:

Можно обратиться к «Объекту», с которым работает пользователь:



Это как бы документ, но он содержит в себе сконвертированные данные реальной БД, проекцию. Узнать, с какой строкой работает пользователь через этот объект невозможно

С чем работает пользователь можно узнать, обратившись к «Элементам» формы:



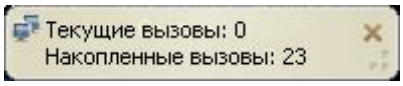

```
&НаКлиенте
Процедура ТоварыЦенаПриИзменении(Элемент)
```

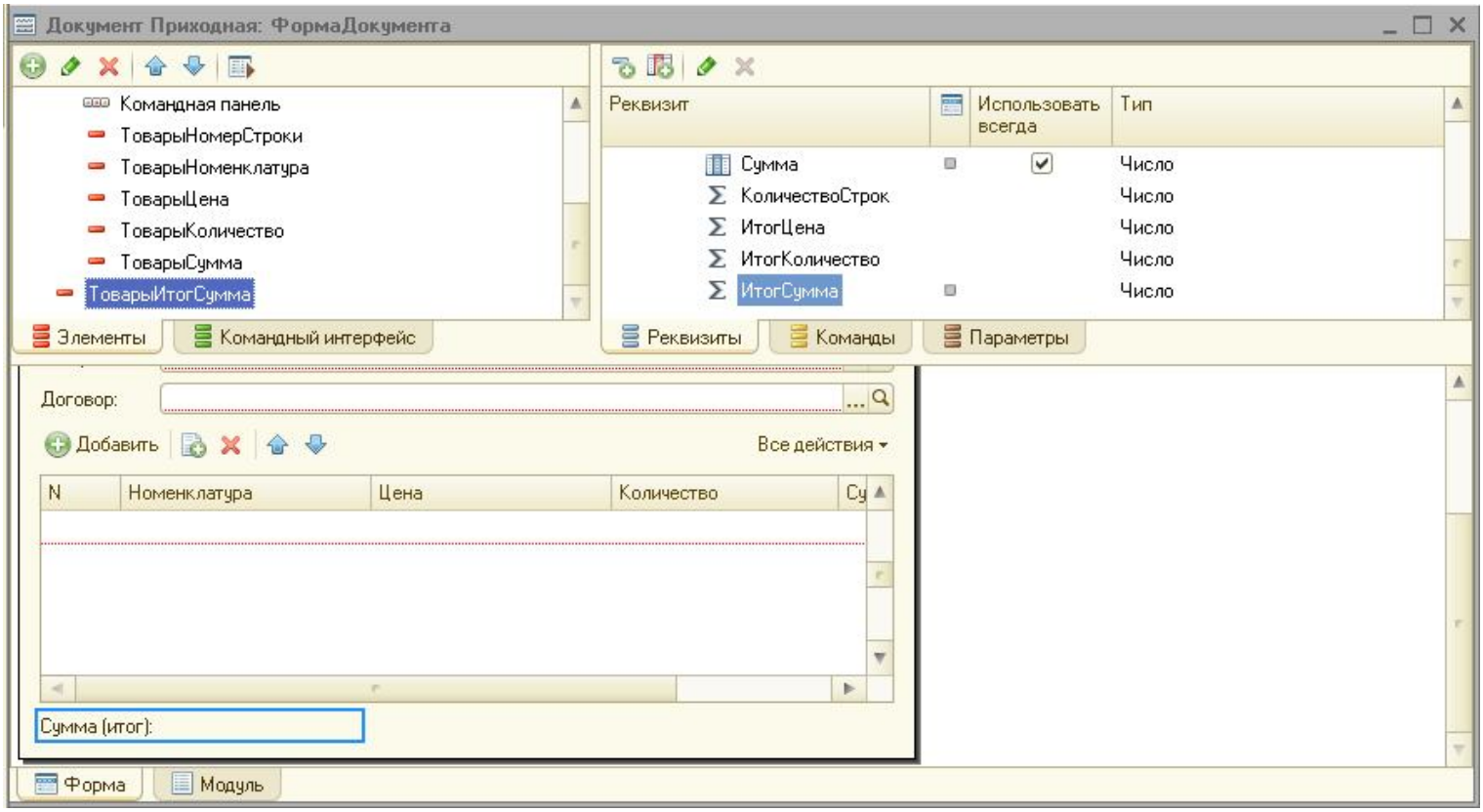
```
    РассчитатьСумму();
КонецПроцедуры

&НаКлиенте
Процедура ТоварыКоличествоПриИзменении(Элемент)
    РассчитатьСумму();
КонецПроцедуры

&НаКлиенте
Процедура РассчитатьСумму()
    Стр = Элементы.Товары.ТекущиеДанные;
    Стр.Сумма = Стр.Цена * Стр.Количество;
КонецПроцедуры
```

В режиме исполнения:
Сумму можно дополнительно запретить редактировать.

Если пользователю принесли Приходную накладную, он ввел все позиции, то для проверки правильности ввода можно рассчитать итог по всем позициям:

При настройке формы, у табличной части «Товары» доступны итоговые значения, надо только поместить их на форме. При их подсчете не производится вызов Сервера, т.е. нет дополнительной нагрузки. Проверить, вызывался ли Сервер, можно с помощью всплывающего окна:





В режиме исполнения:

Это будет визуальный итог, для пользователя.

При поступлении товара надо записывать закупочную цену в структуре:

Номенклатура	Поставщик	Цена

Создадим регистр сведений и опишем по нему проведение документа.

Создадим регистр сведений «Закупочные цены»:

Цена одного товара от одного поставщика с течением времени может меняться, но в пределах одного документа цена будет одна. Реализовать это можно с помощью свойства «Режим записи». Движения в регистре будут появляться при проведении документа, т.е. записи в регистре всегда будут подчинены документу-регистратору.

Хранить цены в самом документе не правильно, т.к. документы предназначены для ввода первичной информации, а регистры – для хранения показателей. Закупочная цена – это показатель, который анализируется с помощью истории. Служебные таблицы Регистра сведений заточены на то, чтобы максимально быстро возвращать значения на указанную дату по указанному отбору по измерениям. В документе для этого придется обращаться запросом к таблице документа, данные которой не проиндексированы, что увеличит время работы.

Свойство	Значение
Имя	ЗакупочныеЦены
Синоним	Закупочные цены
Режим записи	Подчинение регистратору
Периодичность	По позиции регистратора

Сочетание Измерений и поле «Период» будут образовывать ключ уникальности.

Свойство	Значение
Закладка «Данные»	
Измерения	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	СправочникСсылка.Номенклатура
Ведущее	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Ведущее	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Ресурсы	
Цена	
Имя	Цена
Синоним	Цена
Тип	Число 15, 2
Неотрицательное	Истина
Проверка заполнения	Выдавать ошибку

Закладка «Регистраторы» - указание документов, владеющих записями регистра.
В примере: укажем документ «Приходная».
Опишем в документе «Приходная», как он будет формировать движения в регистре сведений «Закупочные цены».

Закладка «Движение»:

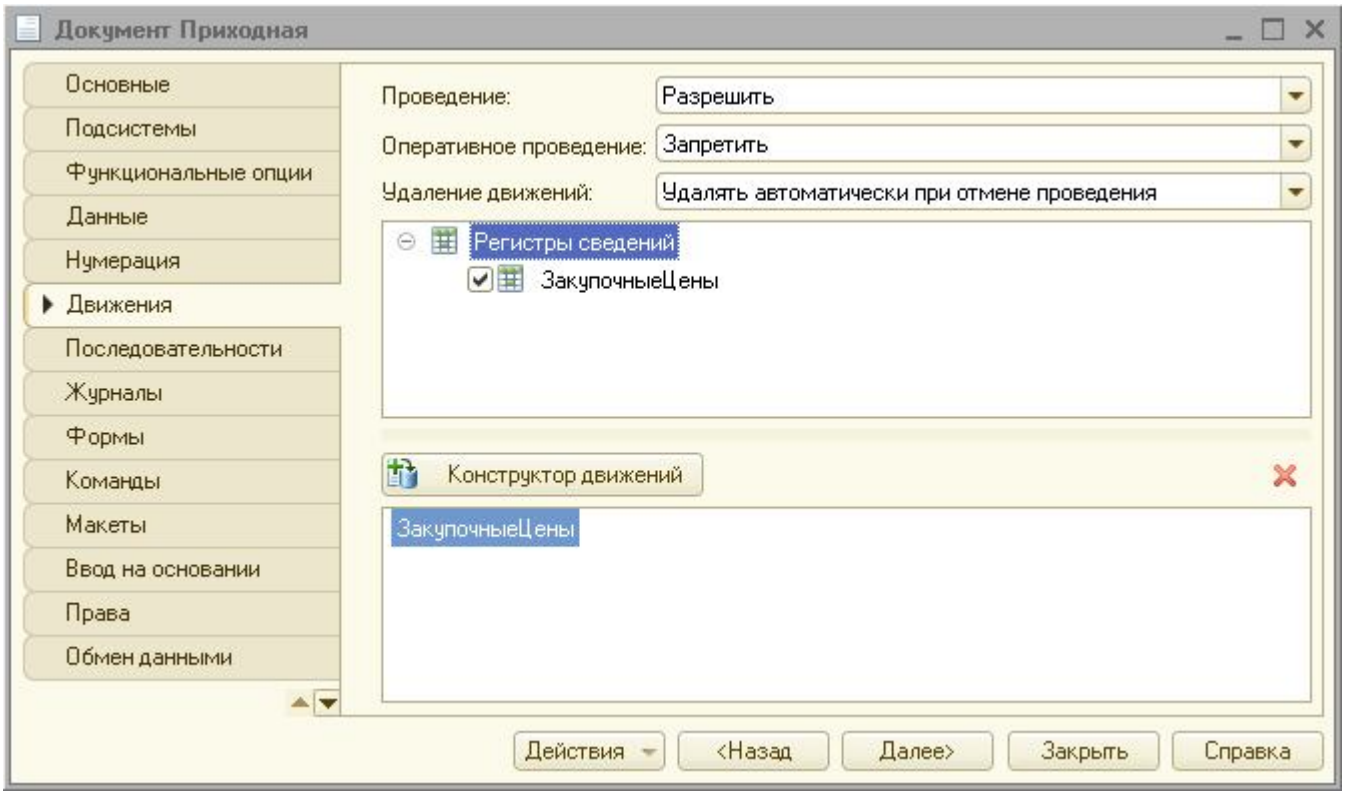
«Оперативное проведение» – необходимо для документов, у которых важен режим проведения (в режиме реального времени или нет).
В примере – не важно.

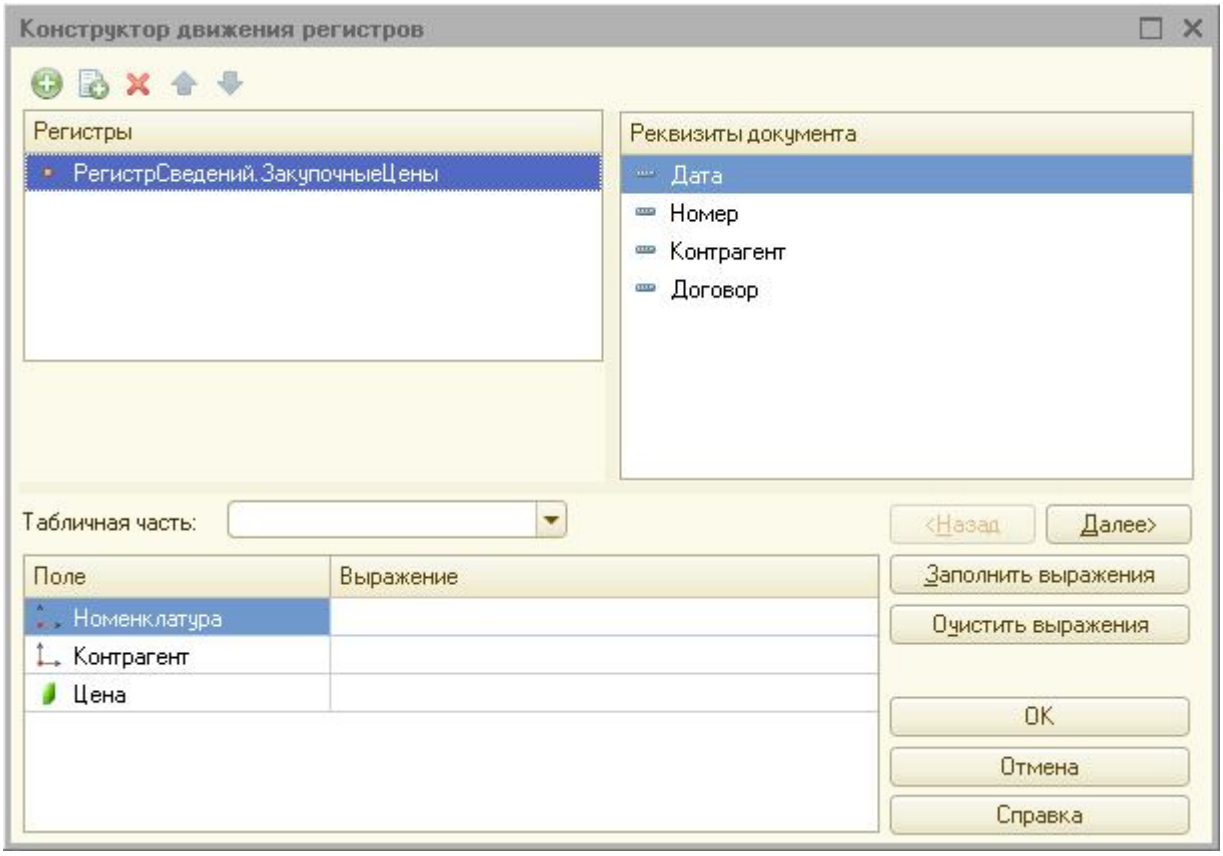
«Удаление движений»

- Удалять автоматически при отмене проведения – старые движения при отмене проведения удаляются
- Удалять автоматически – в любом случае (при отмене проведения, перепроведении) система будет удалять старые движения.
- Не удалять автоматически – движения надо будет удалять вручную.

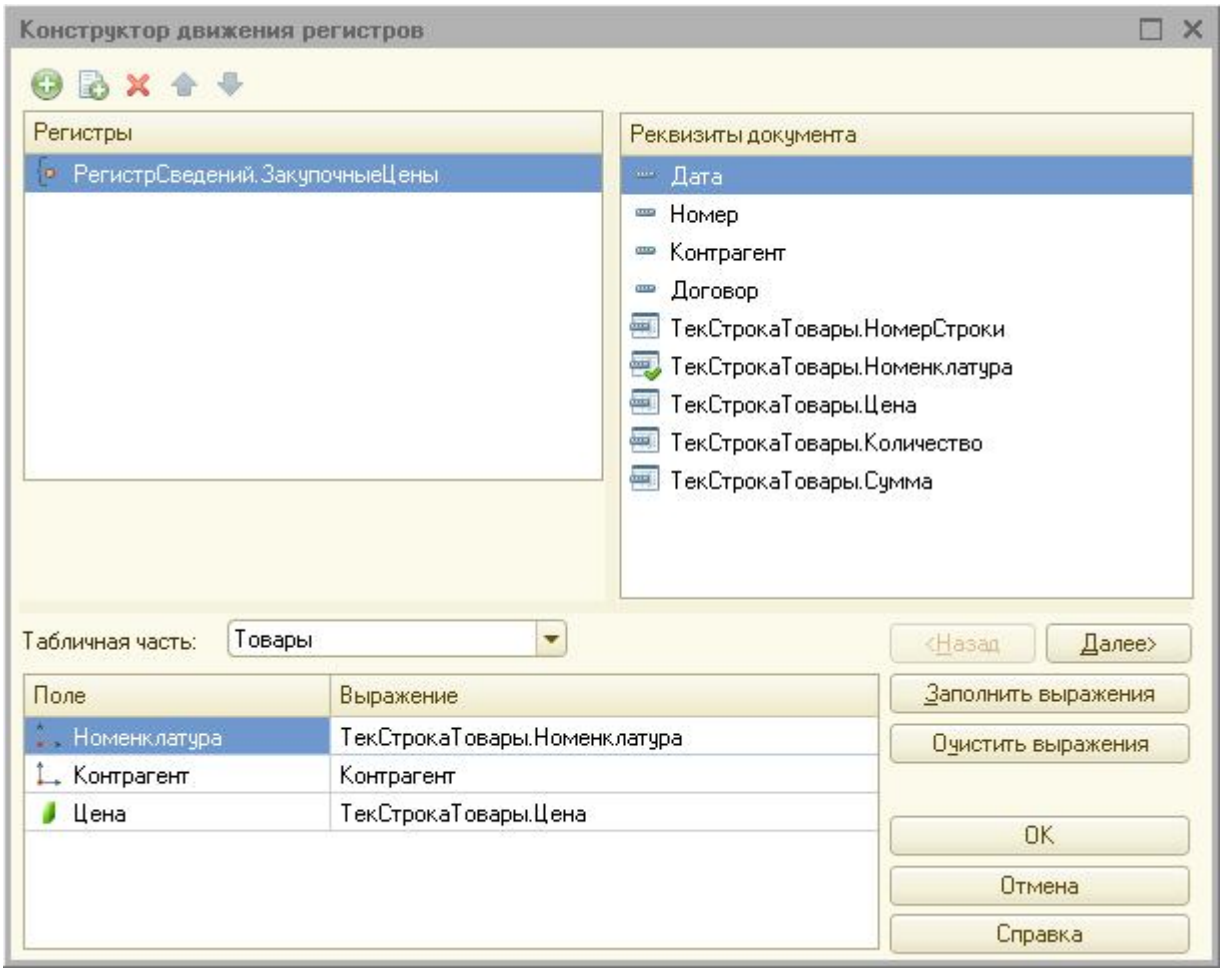
Свойство	Значение
Проведение	Разрешить
Оперативное проведение	Запретить

Отметим галочкой регистр сведений «ЗакупочныеЦены» и запустим Конструктор движений – система в модуле документа сформирует процедуру, формирующую записи в регистре «ЗакупочныеЦены».





Справа вверху – список регистров, по которым будут формироваться движения.
Укажем «Табличную часть» - Товары как табличную часть, по которой будут формироваться движения. Кнопкой «Заполнить выражения» автоматом проставим соответствия реквизитов регистра реквизитам документа. Они как правило совпадают, но могут не совпадать, если не совпадают типы значения регистра и документа или есть опечатка.



В итоге:

```
Процедура ОбработкаПроведения(Отказ, Режим)

    // регистр ЗакупочныеЦены
    Движения.ЗакупочныеЦены.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        Движение = Движения.ЗакупочныеЦены.Добавить();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Контрагент = Контрагент;
        Движение.Цена = ТекСтрокаТовары.Цена;
    КонецЦикла;

КонецПроцедуры
```

«Движения» - это свойство документа, которое позволяет обратиться ко всем движениям по всем регистрам, которые ему подчинены.

«Записывать» - при завершении транзакции проведения документа, движения в регистре будут записаны.
Если «Записывать = Ложь», то движения будут формироваться, но не запишутся.

Есть другой способ формирования и записи движений:
Свойство документа «Запись движений при проведении»:

- Записывать выбранные – запись в регистр будет произведена по наборам записей, у которых указано «Записывать = Истина»
- Записывать модифицированные – строка «Записывать = Истина» не нужна. Система, при завершении транзакции проведения, сравнит данные регистра с тем, что было сформировано внутри обработки проведения. По результату сравнения система примет решение о перезаписи данных.

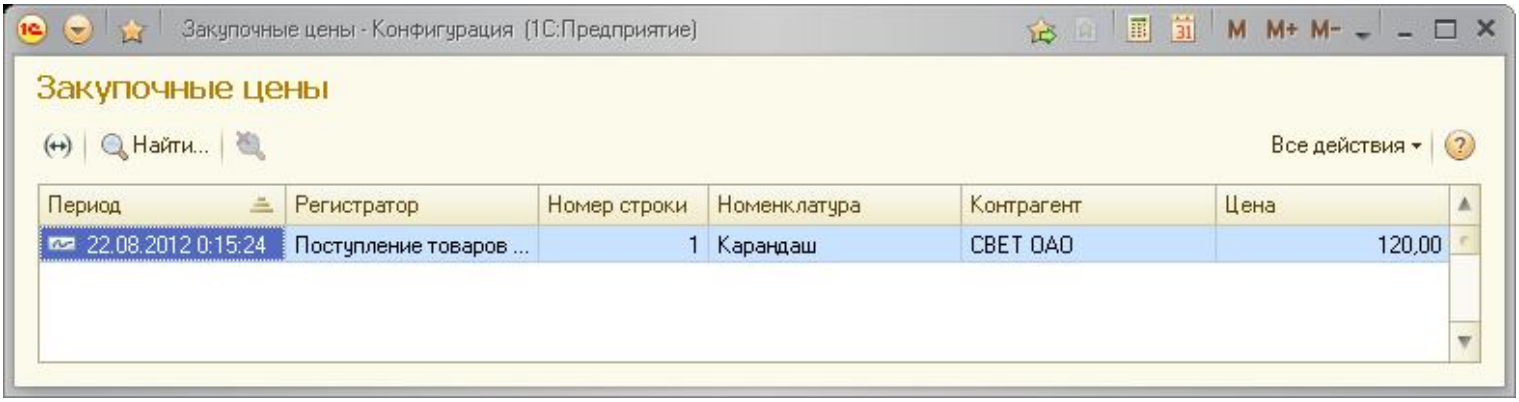
На анализ модифицированности уходит больше времени, поэтому, если есть возможность указать набор перезаписываемых данных, то лучше сделать это вручную.

Далее описан цикл обхода коллекции табличной части «Товары» документа. Мы находимся на Сервере, поэтому идет прямое обращение к табличной части формируемого документа.

«ТекСтрокаТовары» - это переменная, которая получит очередной элемент коллекции табличной части «Товары».

Далее формируется движение по регистру.
Метод «Добавить()» - добавляет в регистр новую строку
«Движение» - переменная, которой передается ссылка на новую строку регистра.

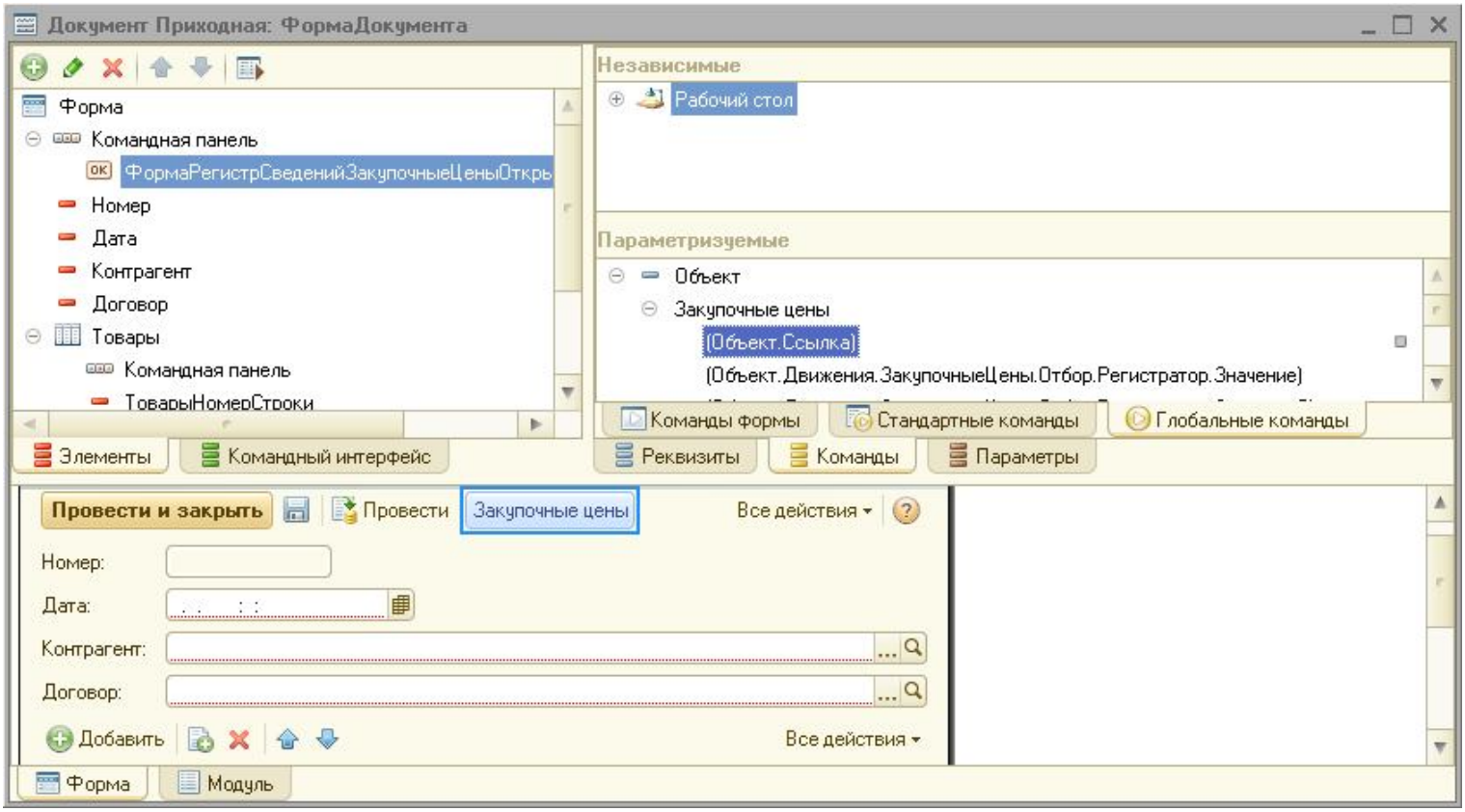
В режиме исполнения:
Проведем документ «Приходная».
Посмотрим записи по регистру сведений «Закупочные цены» - через меню «Все функции»



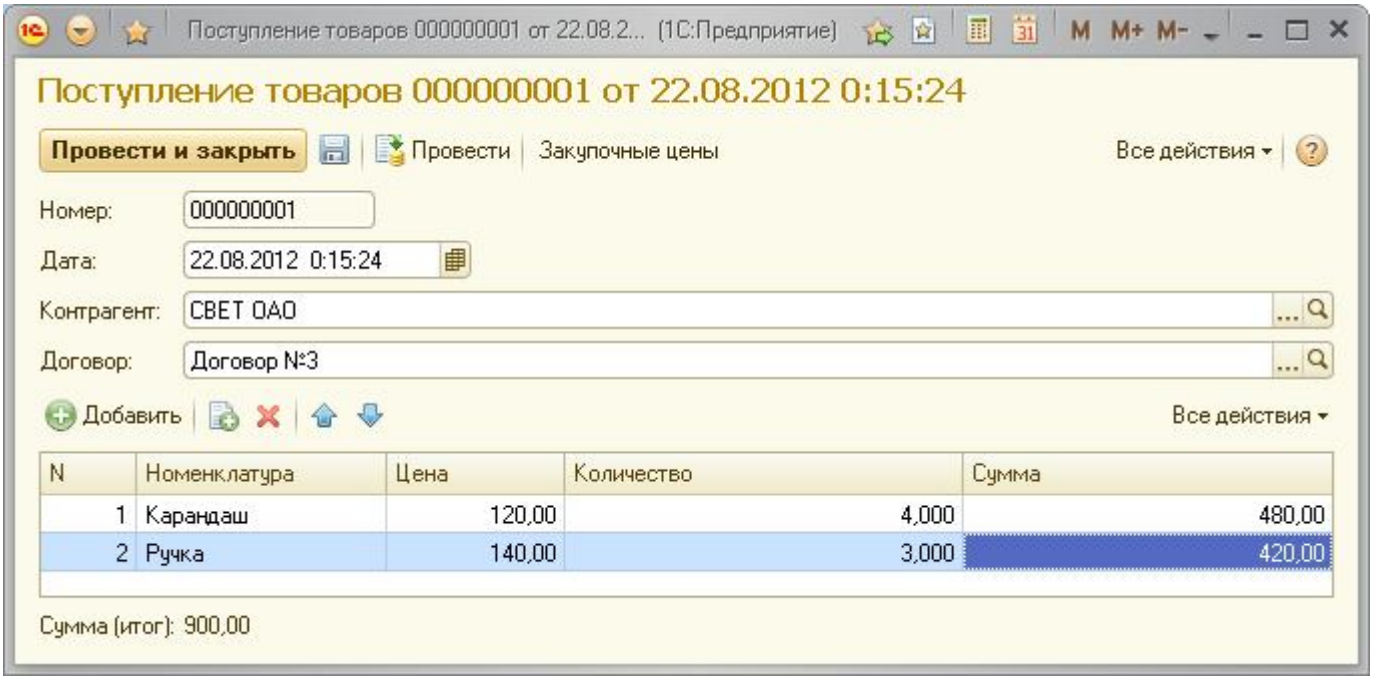
В процедуре проведения:
Параметр «Отказ» - флаг успешной транзакции. Если внутри обработки проведения поставить «Отказ = Истина», то транзакция не завершится и документ не проведется

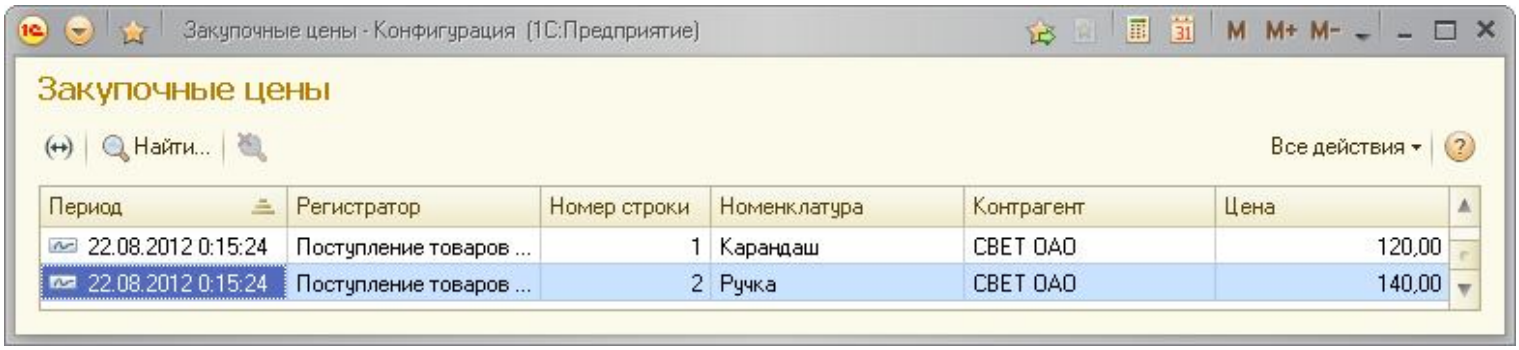
Параметр «Режим» - установка оперативного проведения.

Чтобы организовать на форме документа быстрый переход в регистр:

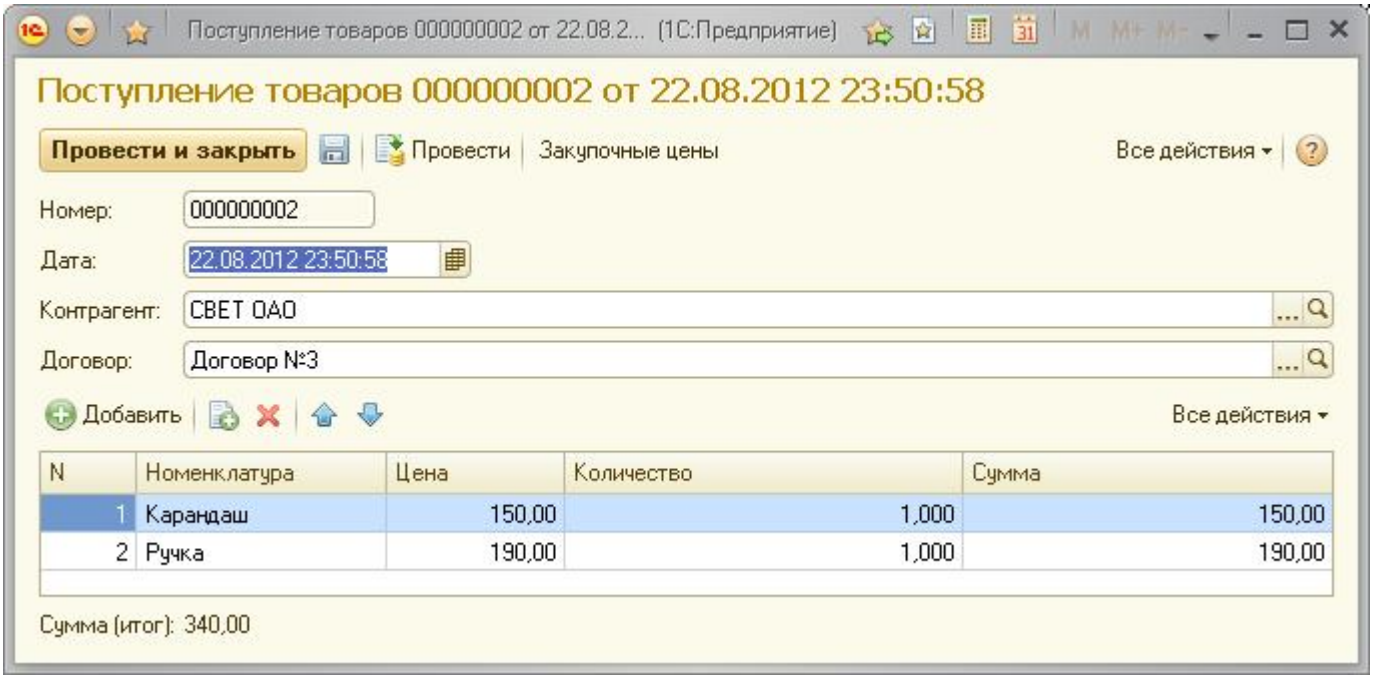


Добавим еще один товар - «Ручка», проведем документ и посмотрим на регистр сведений:

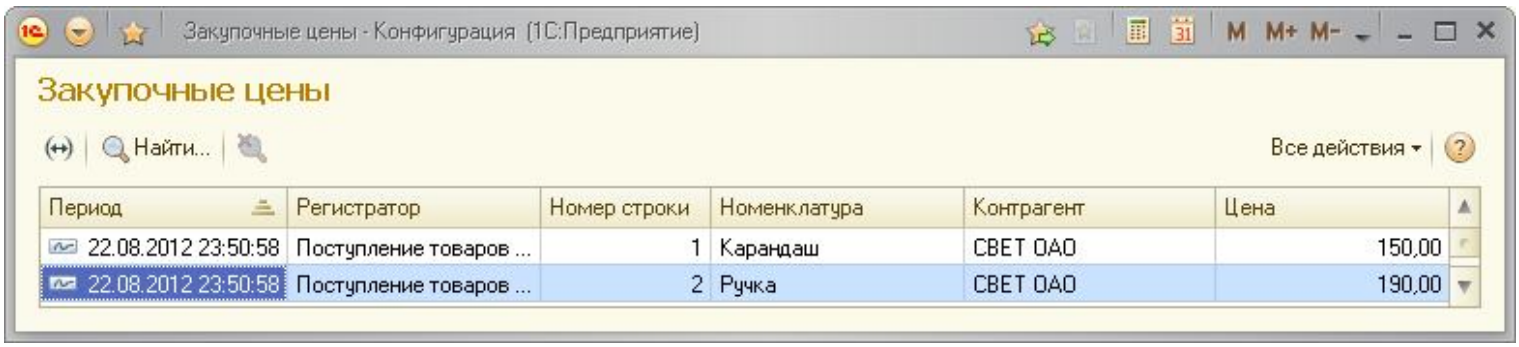




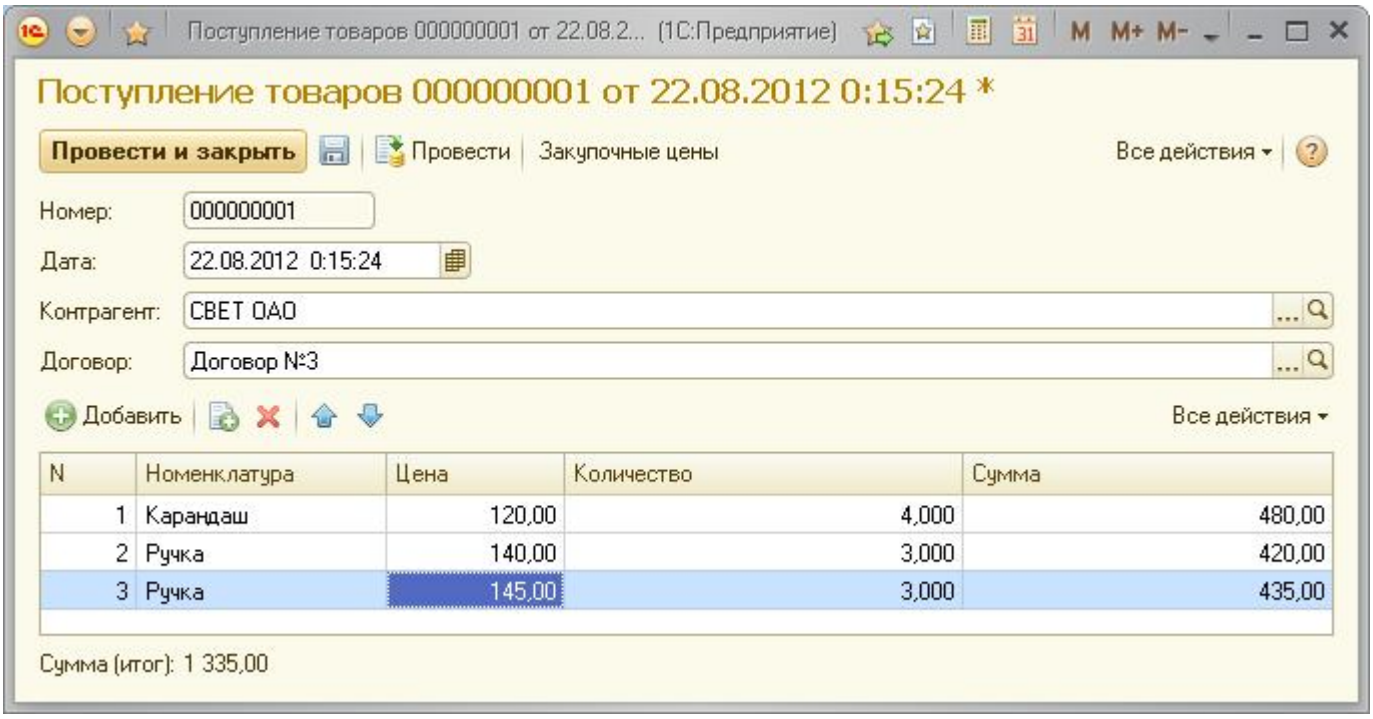
Создадим еще один документ «Приходная», проведем его и перейдем в регистр:

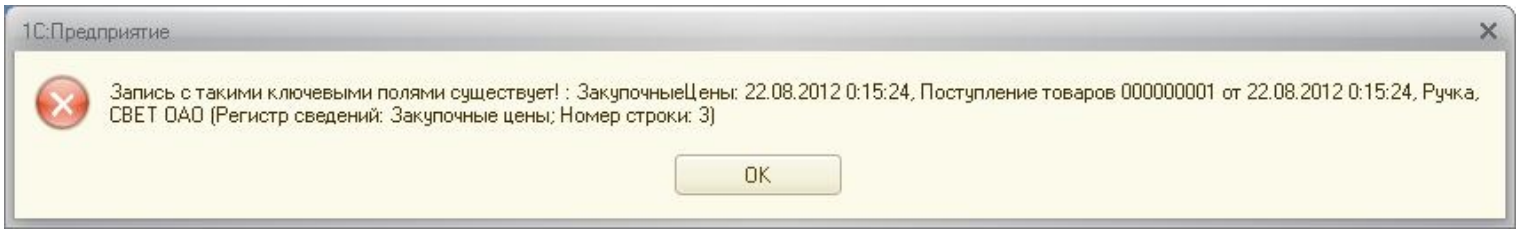


в регистре уже отобразятся цены по второму документу, т.к. установлен автоматический отбор.



Свойство Регистра сведений «Периодичность – По позиции регистратора» - регистратор с измерениями образует ключ уникальности. Если попытаться записать в одном документе две одинаковых позиции (без разницы, по одной цене или по разным), то система выдаст ошибку. Система не даст записать документ ни интерактивно, ни программно.



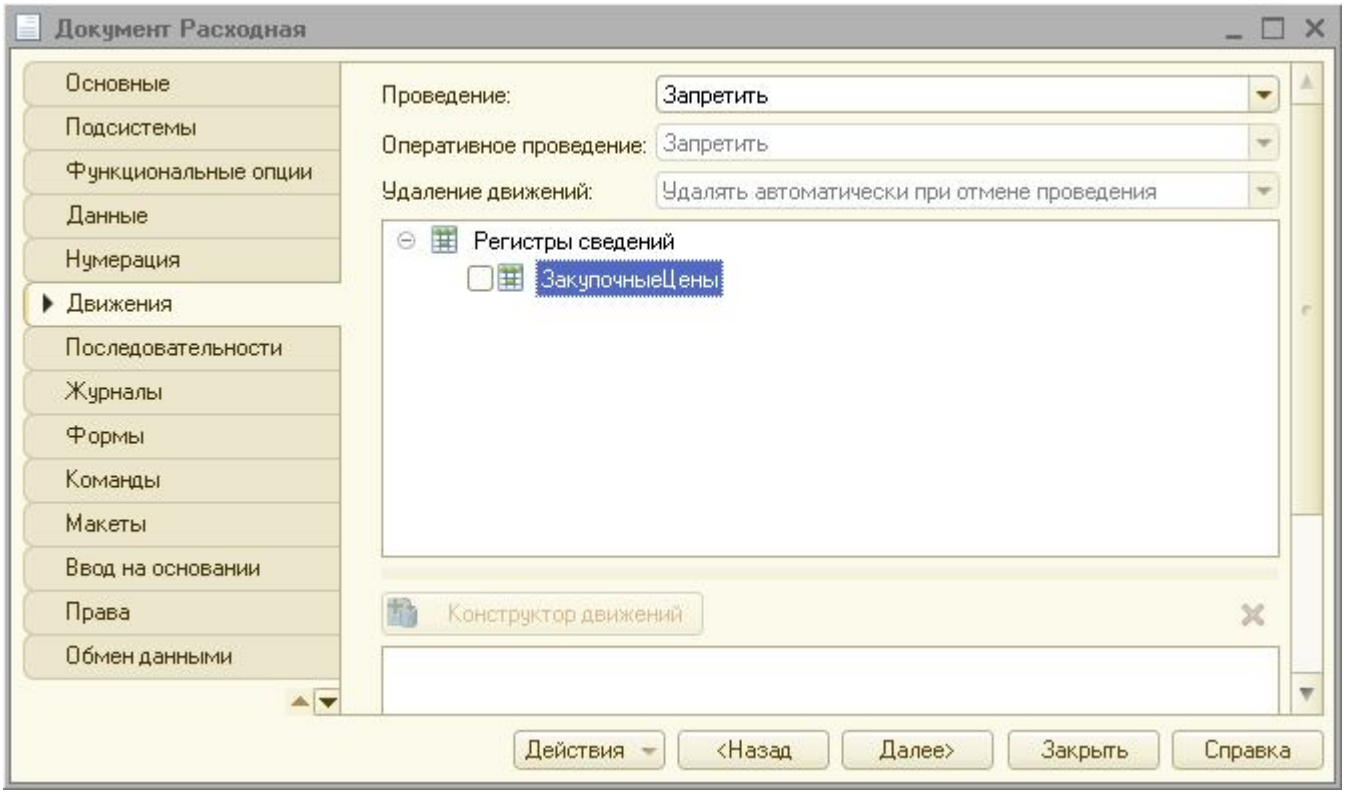


Сделаем документ «Расходная» копированием документа «Приходная»

Свойство	**Значение**
Имя	Расходная
Синоним	Расходная
Представление объекта	Продажа товаров
Представление списка	Расходные документы
Закладка «Данные»	
Реквизиты	
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Проверка заполнения	Выдавать ошибку
Договор	
Имя	Договор
Синоним	Договор
Тип	СправочникСсылка.Договоры
Проверка заполнения	Выдавать ошибку
Связи параметров выбора	Отбор.Владелец(Контрагент)
Табличные части	
Товары	
Имя	Товары
Синоним	Товары
Проверка заполнения	Выдавать ошибку

Свойство	Значение
Табличные части	
Товары	
Реквизиты табличной части	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	Номенклатура
Проверка заполнения	Выдавать ошибку
Цена	
Имя	Цена
Синоним	Цена
Тип	Число 15, 2
Количество	
Имя	Количество
Синоним	Количество
Тип	Число 12, 3 (3 – чтобы можно было вводить граммы)
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2
Проверка заполнения	Выдавать ошибку

Закладка «Движения»:
Движения пока не нужны, тем более по регистру «ЗакупочныеЦены»

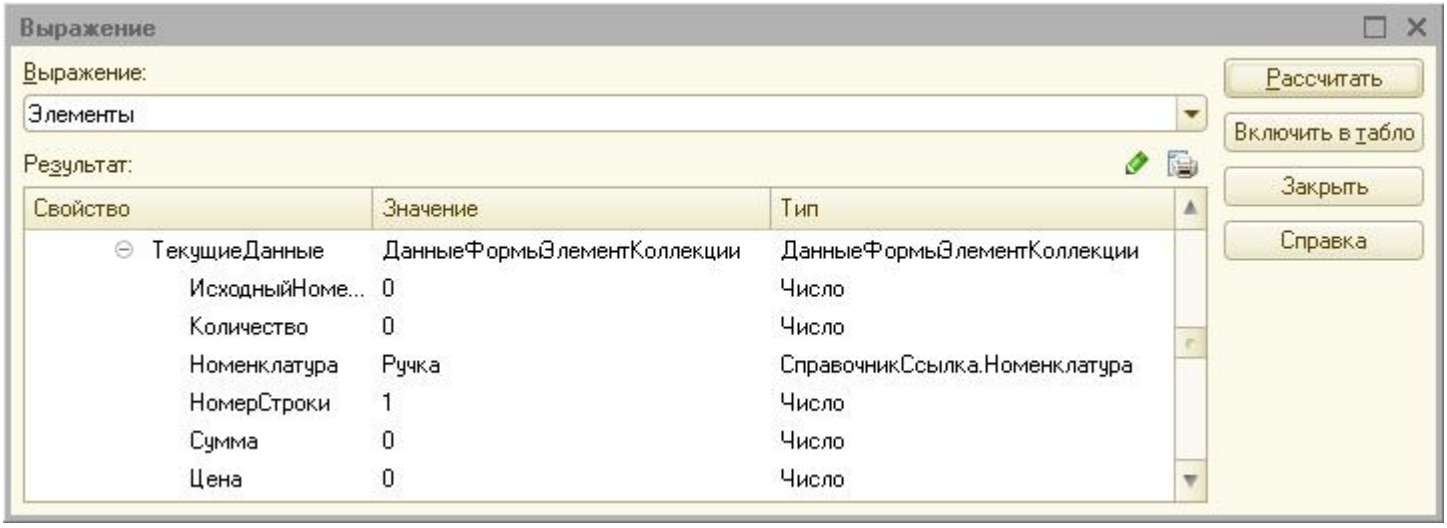


Закладка «Формы»:
В свойствах формы уберем команду перехода в регистр «ЗакупочныеЦены»

Удалим обработку проведения в Модуле объекта.

Пользователь в справочнике «Номенклатура» вводит цены продажи. Будет удобно, если в документе «Расходная», в табличной части, при выборе номенклатуры будет автоматически подставляться Цена продажи.

Опишем событие «ПриИзменении» реквизита «Номенклатура» табличной части «Товары» формы:
Поставим точку останова и обратимся к элементам формы:



но, хотя у реквизита «Номенклатура» тип «СправочникСсылка.Номенклатура», на стороне Клиента невозможно обратиться к свойствам номенклатуры (получить «Цену продажи»).

Надо перейти на сторону Сервера и создать там функцию, которая вернет цену. В функцию передадим Номенклатуру.
В процедуре «ТоварыНоменклатураПриИзменении» получим строку и передадим ссылку на товар в создаваемую функцию.

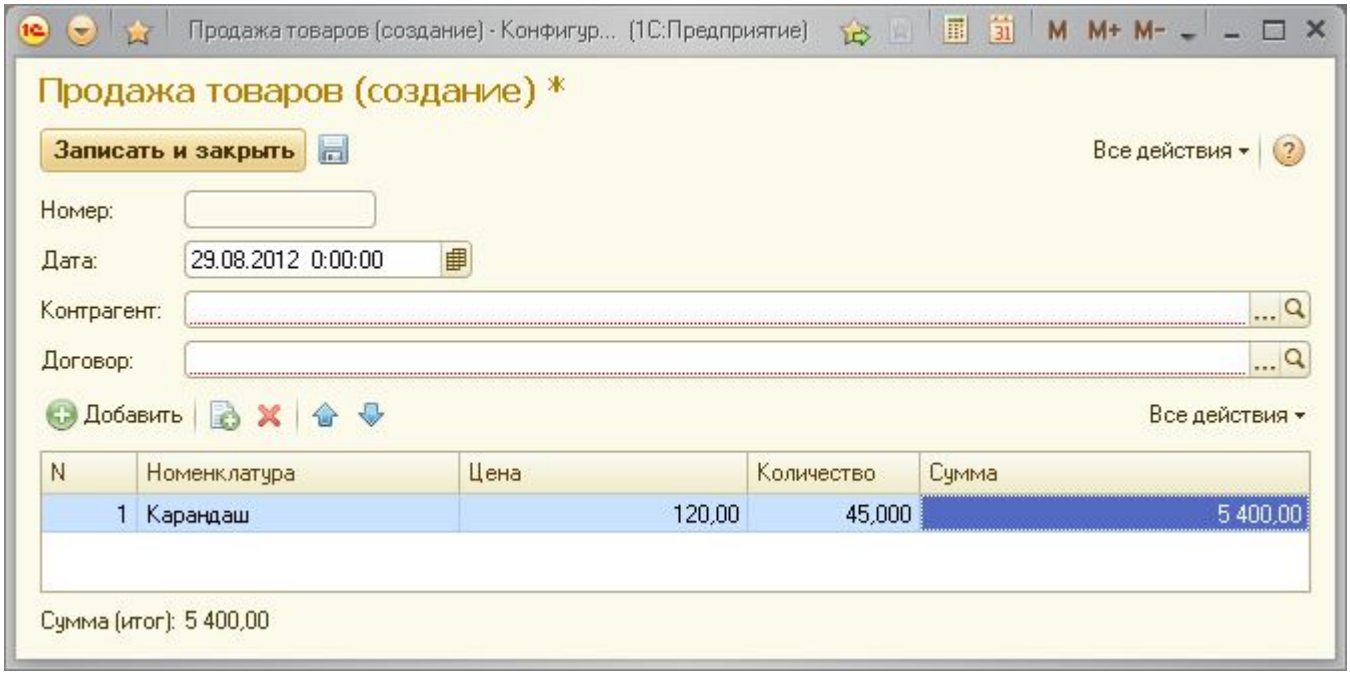
В функции «ПолучитьЦену» Номенклатура передана. Функция выполняется на Сервере, т.е. можно использовать полноценные запросы к БД.

При изменении Номенклатуры, ее Цены надо, чтобы автоматом пересчитывалась и сумма.

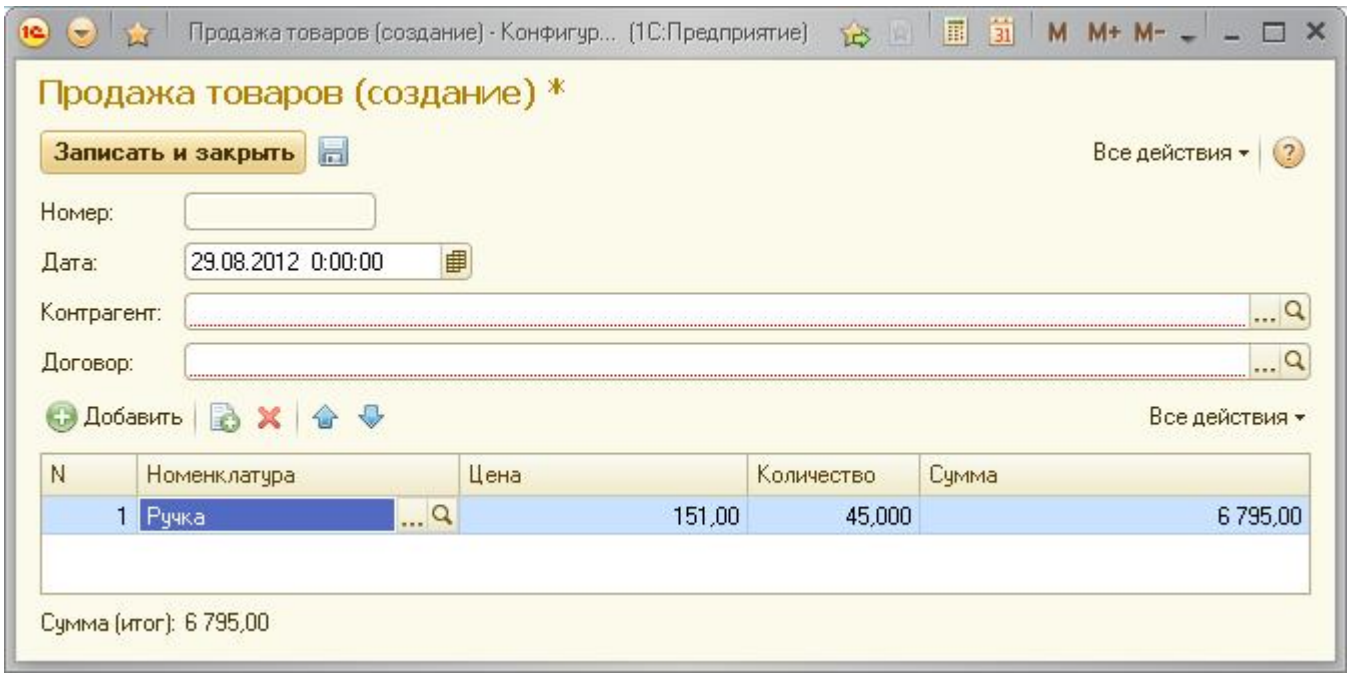
```
&НаКлиенте
Процедура ТоварыНоменклатураПриИзменении(Элемент)
      Стр = Элементы.Товары.ТекущиеДанные;
      Стр.Цена = ПолучитьЦену(Стр.Номенклатура);
      РассчитатьСумму();
КонецПроцедуры

&НаСервереБезКонтекста
Функция ПолучитьЦену(Номенклатура)
      Возврат Номенклатура.ЦенаПродажи;
КонецФункции
```

В режиме исполнения:

В «Расходной» выбираем Номенклатуру – автоматом появляется ее Цена продажи. Ставим Количество – автоматом считается Сумма.



Меняем Номенклатуру – меняется Цена продажи и Сумма.



Итого:

```
&НаКлиенте
Процедура ТоварыЦенаПриИзменении(Элемент)
    РассчитатьСумму();
КонецПроцедуры
```

```
&НаКлиенте
Процедура ТоварыКоличествоПриИзменении(Элемент)
    РассчитатьСумму();
КонецПроцедуры

&НаКлиенте
Процедура РассчитатьСумму()
    Стр = Элементы.Товары.ТекущиеДанные;
    Стр.Сумма = Стр.Цена * Стр.Количество;
КонецПроцедуры

&НаКлиенте
Процедура ТоварыНоменклатураПриИзменении(Элемент)
    Стр = Элементы.Товары.ТекущиеДанные;

    Стр.Цена = ПолучитьЦену(Стр.Номенклатура);
    РассчитатьСумму();
КонецПроцедуры

&НаСервереБезКонтекста
Функция ПолучитьЦену(Номенклатура)
    Возврат Номенклатура.ЦенаПродажи;
КонецФункции
```

Занятие 1.09

Регистр накопления – поддерживает сохранение двух показателей учета:

- Остатки.
- Обороты.

Аналогия действия регистра накопления:

Бабушка продает картошку и записывает, сколько каждый клиент купил у нее ведер, а в конце дня она может подсчитать количество проданных ведер, но считать она будет долго. Быстро узнать, сколько ведер она продала, в любой момент времени, поможет ведение накопительного итога.

1	1
1	2
2	4
1	5
3	8
1	9
1	10
1	11
1	12

Таблица движений	Таблица остатков и оборотов
	Служебная таблица, хранящаяся в БД, доступ к которой напрямую запрещен. Используется для расчета различных показателей. Использование: Регистры заточены на хранение остатков (итогов) на начало и конец месяца (стандартные периоды отчетности). Если надо рассчитать итоги на середину месяца: Берутся остатки/обороты на начало месяца и по реальной «Таблице движений» накопительным расчетом получается результат.

Надо хранить остатки товаров. Показателем в регистре будет «Остаток». Есть измерения остатка:

- Натуральное (штуки, литры и т.д.)
- Стоимостное (различные денежные единицы)

В примере регистр будет простой – храниться будет только «Количество». Остаток будет храниться пока только в разрезе «Номенклатуры» (можно добавить другие разрезы, например, «Склад»).

Остатки товаров	
Номенклатура	Количество

Добавим регистр накопления «Остатки товаров»

Свойство	Значение
Имя	ОстаткиТоваров
Синоним	Остатки товаров
Вид регистра	Остатки
Закладка «Данные»	
Ресурсы	
Количество	
Имя	Количество
Синоним	Количество
Тип	Число 12, 3
Измерения	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	СправочникСсылка.Номенклатура
Запрет незаполненных значений	Истина
Закладка «Регистраторы»	
Регистраторы	Приходная
	Расходная

Вид регистра – определяет, какие показатели можно считать из виртуальных таблиц регистра. Вид регистра «Остатки» сохраняет как «Остатки», так и «Обороты».

Ресурсы – это что хранится.

«Проверка заполнения – Не проверять» т.к. регистр будет записываться только при проведении документов. Хотя, если пользователь введет в документе «Ноль», то можно выдать СОО.

Неотрицательное (ИМЕННО в РЕГИСТРЕ НАКОПЛЕНИЯ) – это не контроль отрицательных остатков! Это свойство не позволит сформировать движение со знаком «минус» (поступило «-5» коробок).

Минусы в регистрах – это «Сторно». Например, купили ведро картошки, а потом вернули. В программе нельзя просто удалить документ о покупке. Формируется сторно-запись с такими же движениями как при продаже, но со знаком «минус» (вернет остатки и уничтожит обороты).

Запрет незаполненных значений – не позволит сформировать движение по регистру с пустой номенклатурной позицией (ни программно ни интерактивно).

Закладка «Регистраторы» - указание регистраторов, которые будут формировать движения в регистре.

Сформируем движения в регистраторах, т.е. в документах.

Документ «Приходная»:

«Модуль объекта», процедура «ОбработкаПроведения» - в цикл добавим формирование движений по регистру «Остатки товаров».

```
Процедура ОбработкаПроведения(Отказ, Режим)

    // регистр ЗакупочныеЦены
    Движения.ЗакупочныеЦены.Записывать = Истина;
    Движения.ОстаткиТоваров.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        Движение = Движения.ЗакупочныеЦены.Добавить();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Контрагент = Контрагент;
        Движение.Цена = ТекСтрокаТовары.Цена;

        Движение = Движения.ОстаткиТоваров.ДобавитьПриход();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Количество = ТекСтрокаТовары.Количество;

    КонецЦикла;

КонецПроцедуры
```

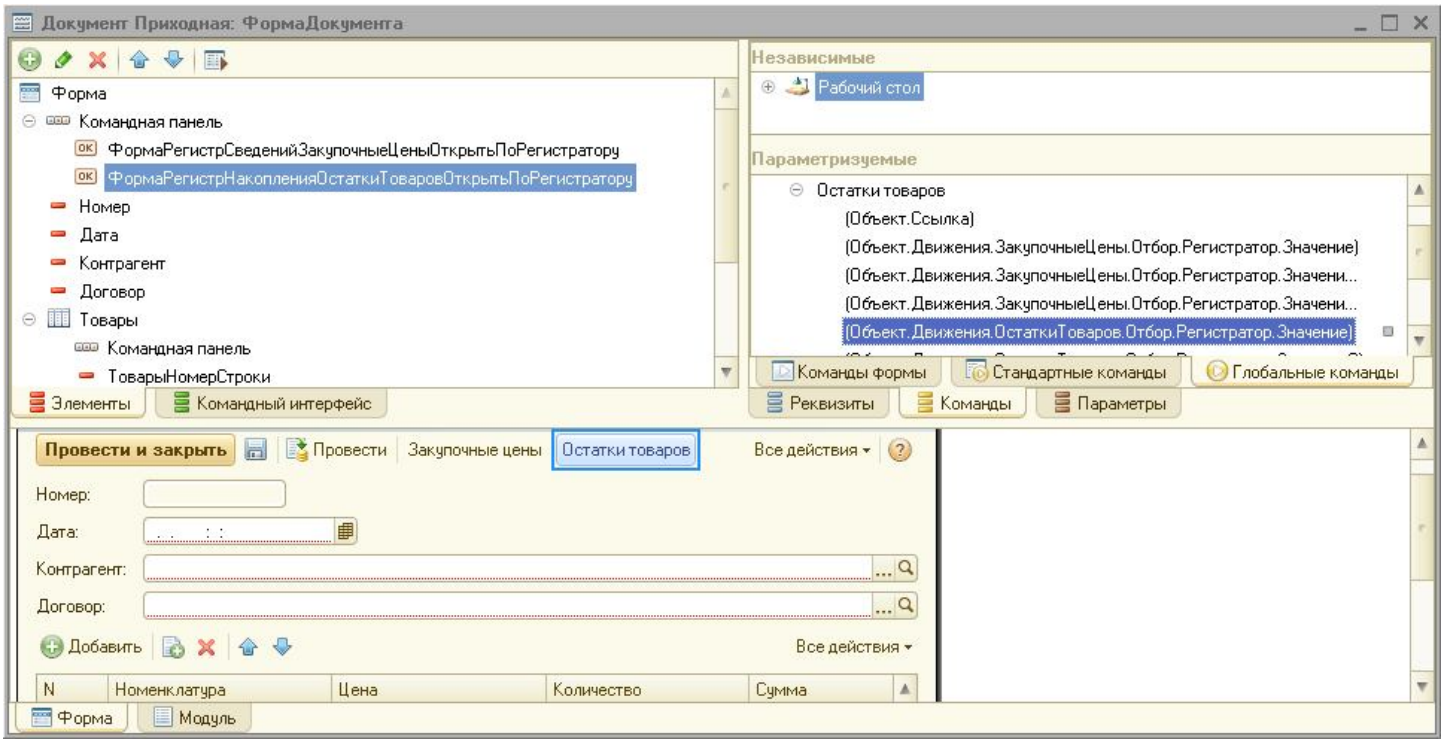
Движения можно создавать не только в обработке проведения, но и вручную (кнопка с отдельной процедурой в форме, прямой доступ к регистру накопления и т.д.). Но здесь движения формируются через свойство документа «Движения», организуется транзакция проведения документа. Если транзакция успешна, то свойство «Движения» документа физически записывается в БД. При завершении транзакции система записывает движения в том порядке, в котором они представлены на дереве метаданных – это гарантия уменьшения количества ошибок SQL-сервера вида DeadLock (взаимная блокировка таблиц).
При записи в регистр вручную, порядок определяется произвольно - риск возникновения ошибки «DeadLock» возрастает.

Документ «Расходная»:
В «Модуль объекта» добавим процедуру:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)
    Движения.ОстаткиТоваров.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
```

```
        Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Количество = ТекСтрокаТовары.Количество;

    КонецЦикла;
КонецПроцедуры
```

В форму документа «Приходная» добавим команду перехода к движениям по регистру накопления «Остатки товаров»:



При перепроведении движения можно очищать и через код. Т.е. на закладке «Движения» свойство будет:

Свойство	Значение
Удаление движений	Удалять автоматически при отмене проведения

А в коде надо прописать:

```
Движения.ЗакупочныеЦены.Очистить();
```

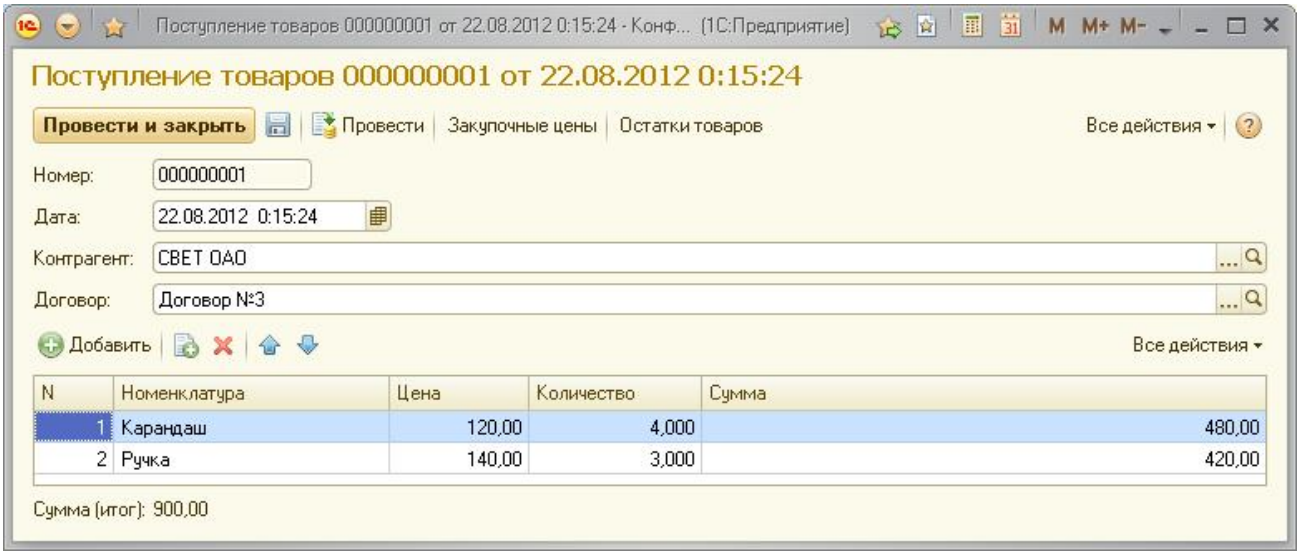
Аналогичную команду добавим на форму документа «Расходная».
На закладке «Движения» изменим свойства:

Свойство	Значение
Проведение	Разрешить
Удаление движений	Удалять автоматически

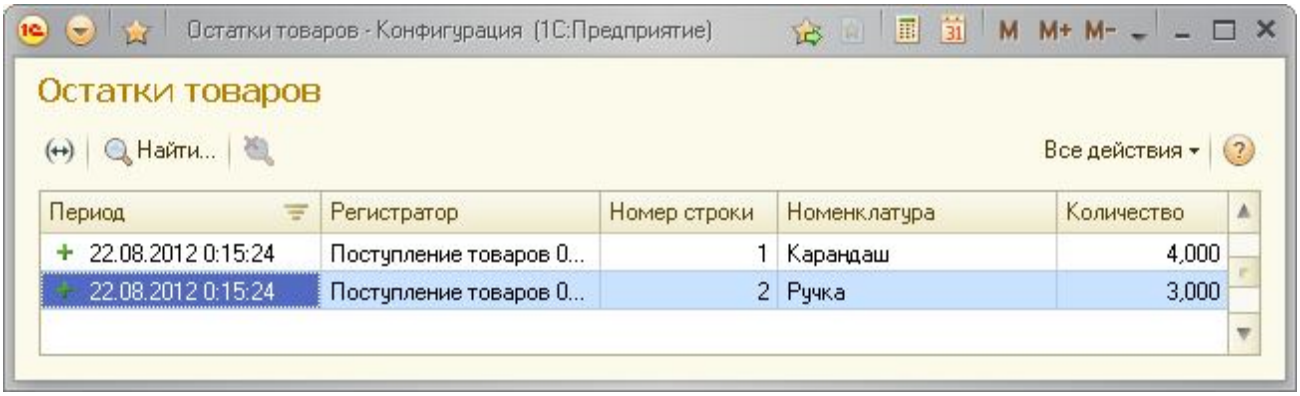
Т.е. движения будут удаляться при отмене проведения и при перепроведении.

В режиме исполнения:

Перепроведем документ «Приходная»

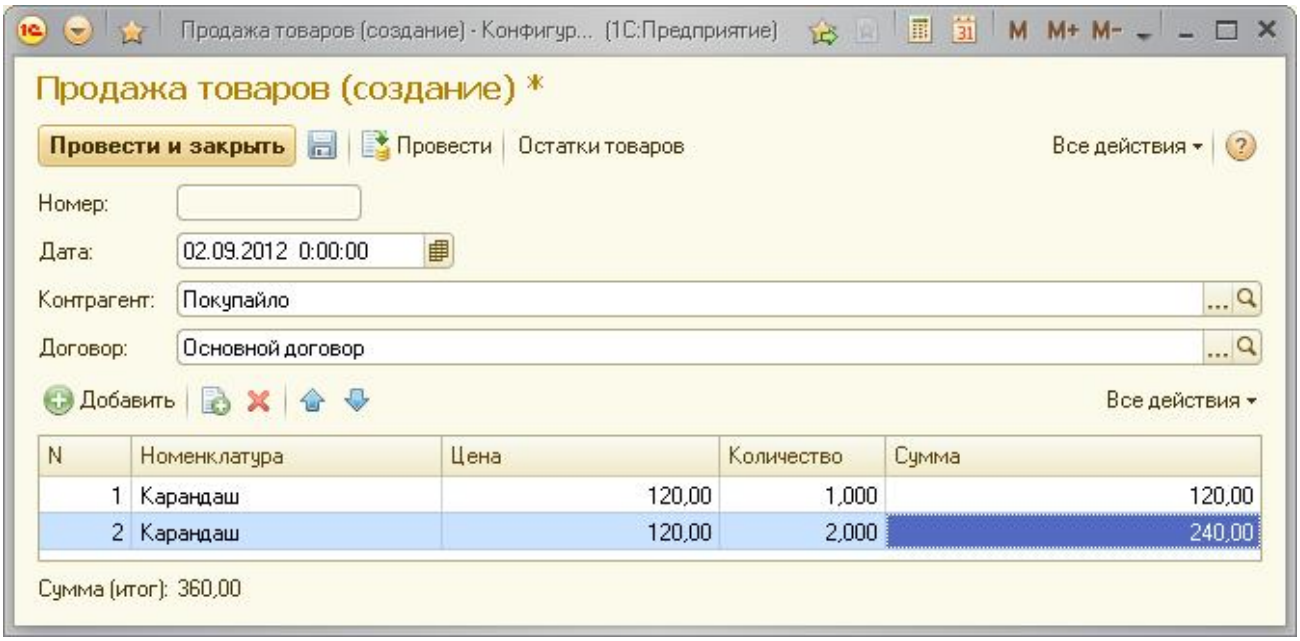


И проверим Остатки товаров:

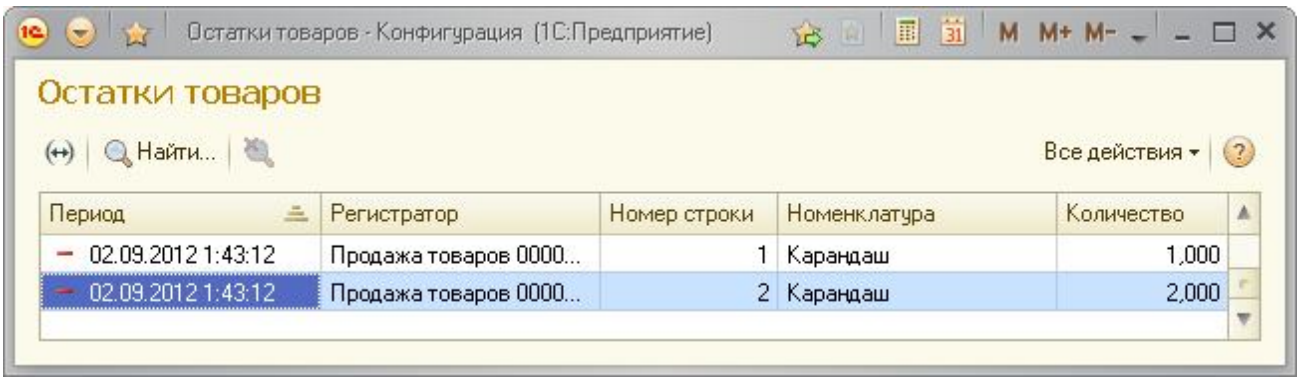


Создадим документ «Расходная»:

В справочнике «Контрагенты» создадим группу «Покупатели», а в ней создадим покупателя и выберем его в качестве контрагента. Создадим договор контрагента и также выберем его.



Проведем документ «Расходная» и проверим Остатки:



Теория Регистра накопления:

Таблица «Движения» - можно увидеть в режиме исполнения.

ВидДвижения – хранится системное перечисление «ВидДвиженияНакопления», которое может принимать значения: «Приход» или «Расход».

Период – дата, начиная с которой движение повлияет на остатки и обороты.

Регистратор – ссылка на документ, которому принадлежат записи в регистре. Но это не документ, который сформировал движения, т.к. можно одним документом сформировать движения, принадлежащие другому документу.

НомерСтроки – номер строки в наборе записей регистра (не номер строки в таблице документа).

Активность – признак (булево). Показывает, будет ли данное движение влиять на остатки и обороты. Например, покупатель заказал товар, можно заранее сформировать документ заказа и провести его, но записи в регистре сделать неактивными. А после оплаты заказчиком това-

ра, в документе нажать кнопку (сделанную), которая активизирует записи движений в регистре.

Движения						
ВидДвижения	Период	Регистратор	НомерСтроки	Активность	Номенклатура	Количество
Приход	12.04.2012	Приходная №1	1	Истина	Ложка	10
Приход	12.04.2012	Приходная №1	2	Истина	Вилка	5
Приход	13.04.2012	Приходная №2	1	Истина	Ложка	10
Приход	13.04.2012	Приходная №2	2	Истина	Нож	1

Движения, созданные по двум документам

Таблица «Остатки» (тип регистра – «Остатки»):

Остатки				
Период	Измерение	Ресурс		
Период	Номенклатура	КоличествоОстаток	КоличествоПриход	КоличествоРасход

После формирования движений по первому документу, в таблице остатков появится запись:

Остатки				
Период	Измерение	Ресурс		
Период	Номенклатура	КоличествоОстаток	КоличествоПриход	КоличествоРасход
30.04.2012	Ложка	10	10	0
30.04.2012	Вилка	5	5	0

После формирования движений по второму документу, система сделает анализ записей. Т.к. Ложка в остатках уже есть, то запись по ней будет обновлена. Ножа в измерениях нет, поэтому будет сформирована новая запись:

Остатки				
Период	**Измерение**	**Ресурс**		
Период	**Номенклатура**	**КоличествоОстаток**	**КоличествоПриход**	**КоличествоРасход**
30.04.2012	Ложка	20	10	0
30.04.2012	Вилка	5	5	0
30.04.2012	Нож	1	1	0

Аналогично формируется и расход:

Движения						
ВидДвижения	**Период**	**Регистратор**	**НомерСтроки**	**Активность**	**Номенклатура**	**Количество**
Приход	12.04.2012	Приходная №1	1	Истина	Ложка	10
Приход	12.04.2012	Приходная №1	2	Истина	Вилка	5
Приход	13.04.2012	Приходная №2	1	Истина	Ложка	10
Приход	13.04.2012	Приходная №2	2	Истина	Нож	1
Расход	15.04.2012	Расходная №1	1	Истина	Нож	1
Расход	15.04.2012	Расходная №1	2	Истина	Поварежка	3

Остатки				
Период	Измерение	Ресурс		
Период	Номенклатура	КоличествоОстаток	КоличествоПриход	КоличествоРасход
30.04.2012	Ложка	20	10	0
30.04.2012	Вилка	5	5	0
30.04.2012	Нож	0	1	1
30.04.2012	Поварежка	-3	0	3

Пройдет месяц и система начнет накапливать итоги на новый период:

Движения						
ВидДвижения	Период	Регистратор	НомерСтроки	Активность	Номенклатура	Количество
Приход	12.04.2012	Приходная №1	1	Истина	Ложка	10
Приход	12.04.2012	Приходная №1	2	Истина	Вилка	5
Приход	13.04.2012	Приходная №2	1	Истина	Ложка	10
Приход	13.04.2012	Приходная №2	2	Истина	Нож	1
Расход	15.04.2012	Расходная №1	1	Истина	Нож	1
Расход	15.04.2012	Расходная №1	2	Истина	Поварежка	3
Приход	01.05.2012	Приходная №3	1	Истина	Кочерга	5

Система хранит остатки на начало и конец месяца. Пройдет май, и система скопирует все остатки за апрель, причем копируется информация только о ненулевых значениях и без информации об оборотах:

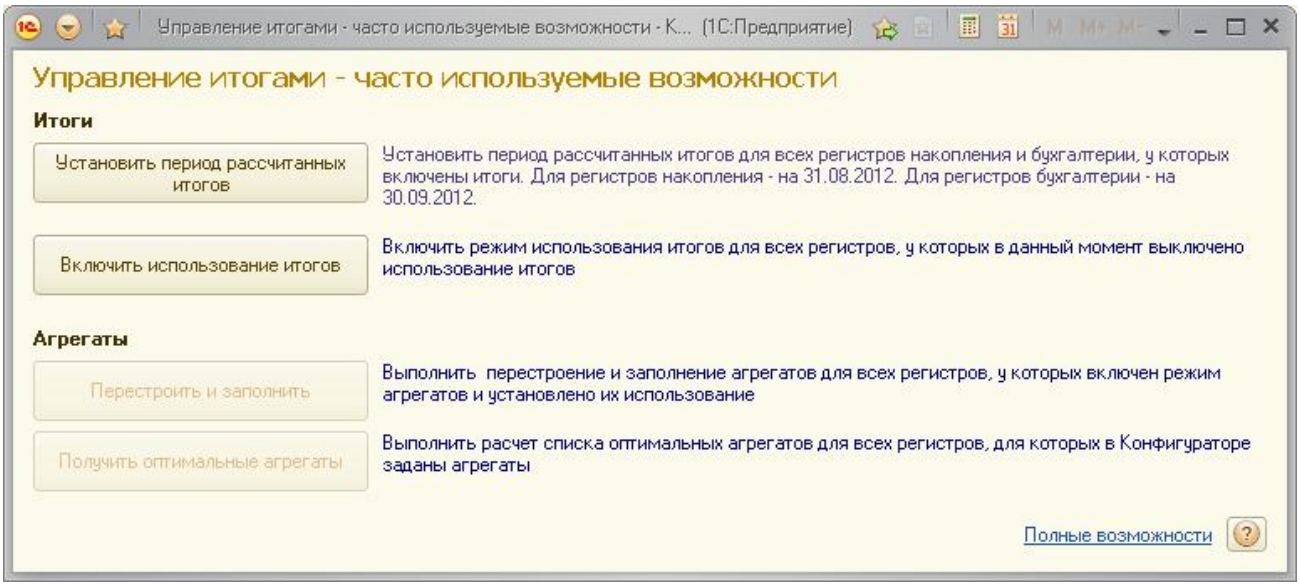
Остатки				
Период	Измерение	Ресурс		
Период	Номенклатура	КоличествоОстаток	КоличествоПриход	КоличествоРасход
30.04.2012	Ложка	20	10	0
30.04.2012	Вилка	5	5	0
30.04.2012	Нож	0	1	1
30.04.2012	Поварежка	-3	0	3
31.05.2012	Ложка	20		
31.05.2012	Вилка	5		
31.05.2012	Поварежка	-3		

После этого система еще сформирует новую запись (за май):

Остатки				
Период	Измерение	Ресурс		
Период	Номенклатура	КоличествоОстаток	КоличествоПриход	КоличествоРасход
30.04.2012	Ложка	20	10	0
30.04.2012	Вилка	5	5	0
30.04.2012	Нож	0	1	1
30.04.2012	Поварежка	-3	0	3
31.05.2012	Ложка	20		
31.05.2012	Вилка	5		
31.05.2012	Поварежка	-3		
31.05.2012	Кочерга	5	5	0

Копируется это все при управлении итогами. Если его не настроить, то можно скопировать вручную (открыть новые таблицы с итогами):

Режим исполнения - - Все функции – Стандартные – Управление итогами:

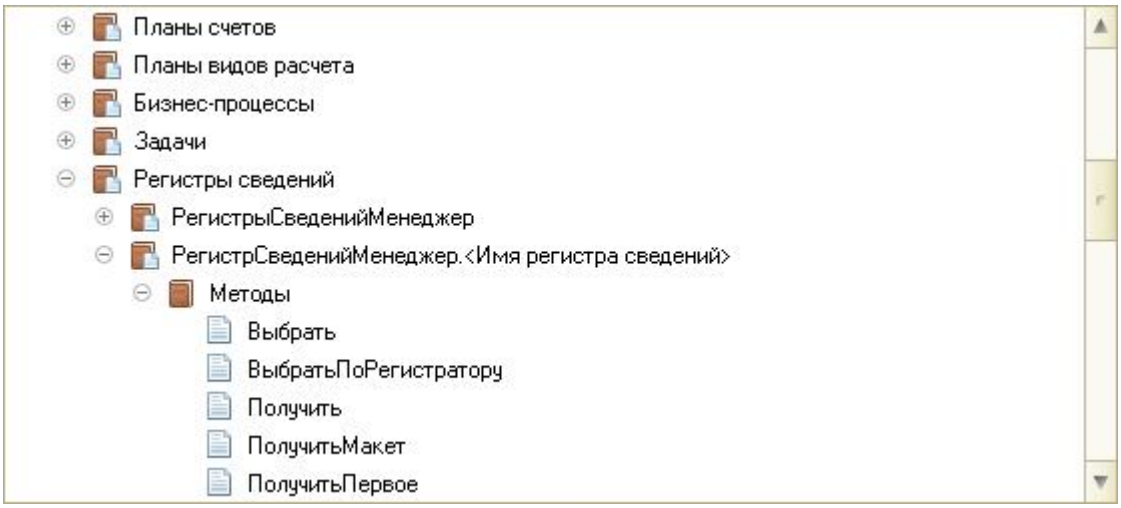


Для регистров накопления – рассчитываются итоги на конец прошлого оперативного периода. Если идет апрель (еще нет таблицы за май) и в середине апреля опять добавляются записи, то система обновит все записи, которых коснулись изменения.
А если записи будут уже за апрель и май и в апреле добавятся записи, то системе надо будет обновить таблицу и за апрель и за май.
Для регистров накопления итоги на текущий оперативный период всегда рассчитаны. Это так называемые «Grand Total (основные итоги)», и они рассчитаны на 30.11.3999. На эту дату система всегда хранит остатки и обороты по всем значениям регистра (оперативные, реальные итоги на текущий момент времени).
Поэтому в регистрах накопления нет смысла рассчитывать текущий оперативный период, т.к. его таблица всегда будет совпадать с таблицей «Основных итогов».

Если не включать расчет итогов, то они не будут накапливаться в системе и она будет обращаться к «Основным итогам», а на начало и конец каждого месяца их не будет. Как следствие - будут очень медленно формироваться отчеты по оборотам, по остаткам на произвольный момент времени (не на реальный).

Для регистров бухгалтерии – рассчитываются итоги на конец текущего оперативного периода.

Механизмы извлечения данных из регистров:
Объектная модель – с помощью Методов регистра.



Но система на основе методов неявным образом запрос к БД, которым мы управлять не можем. Еще есть ограничения в передаче фильтров в запрос, т.е. запрос будет не оптимален (лишние данные).

Модель запросов – альтернативный способ извлечения данных из БД, работающий по модели таблиц, а не объектов.

Отличие модели таблиц от модели объектов:
Объект – это набор таблиц. Например, объект – документ «Расходная» состоит из двух таблиц (шапка и табличная часть). Оперируя объектом, мы оперируем всеми его таблицами. Именно поэтому с помощью объектной модели можно сформировать движения, создать новый объект, удалить объект. Нужные таблицы будут блокироваться автоматически.
С помощью модели запросов можно только читать данные, чтобы, изменив одну таблицу и забыв изменить при этом другую (связанную) таблицу объекта, мы его не разрушили.
Но т.к. при этом можно обращаться к самой таблице, ее конкретным полям, то запрос – это самый быстрый способ извлечения данных из БД.

Работа запроса:
В любом месте модуля, который исполняется на Сервере, можно создать объект «Запрос»:

```
Запрос = Новый Запрос;
```

Далее можно описать текст запроса (самая важная часть):

```
Запрос.Текст = «»;
```

Если надо, установить параметры запроса – чтобы передавать в запрос значения извне:

```
Запрос.УстановитьПераметр(«ИмяПараметра», Значение);
```

Получить результат запроса:

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

Что можно делать с результатом:

Перебрать в цикле:

```
Выборка = Результат.Выбрать();
Пока Выборка.Следующий() Цикл

КонецЦикла;
```

Выгрузить в табличную часть:

```
Таблица = Результат.Выгрузить();
```

и дальше выполнять действия уже с самой таблицей.

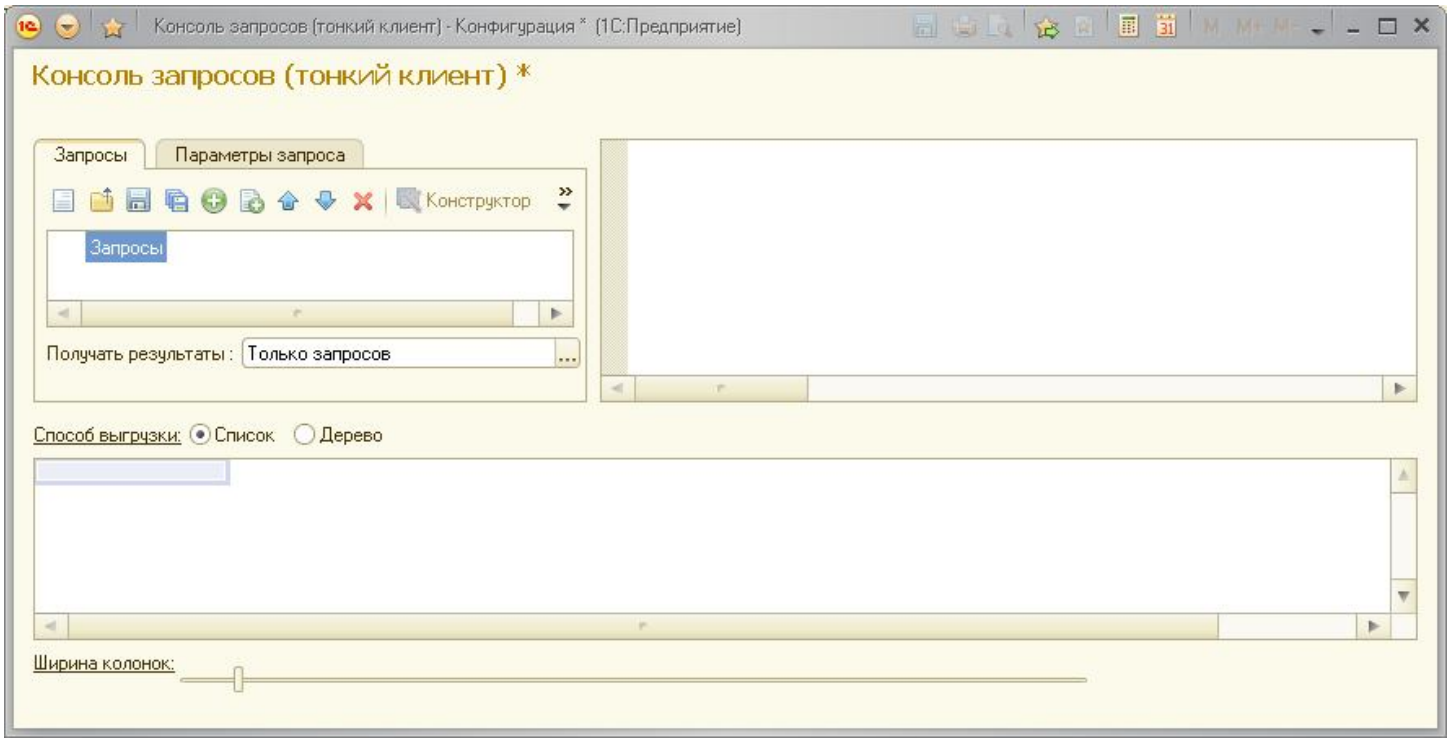
Можно передать в диаграмму. Диаграмму можно даже сформировать автоматом на основе результата запроса.

Можно передать как источник данных для Построителя отчетов, Системы Компоновки Данных (СКД).

И т.д.

Почему лучше делать выборку, а не выгрузку запроса и обработку Таблицы значений (ТЗ)?
Результат запроса кэшируется на сервере. Выборка получает его порционно, а в ТЗ результат запроса выгружается полностью и ТЗ на сервере передается в контекст сеанса работы.
ТЗ занимает память.
Обход результата запроса. Например, если остатки товаров выбраны в разрезе товаров и складов, то можно выбрать сначала товары, а потом склады или наоборот, обратившись к результату запроса, описав разные параметры выборки. В ТЗ надо будет перебирать записи еще раз..
Результат запроса проиндексирован по значениям итогов, а ТЗ надо будет индексировать дополнительно.

Консоль запросов – обработка для описания текста запроса и просмотра результата.

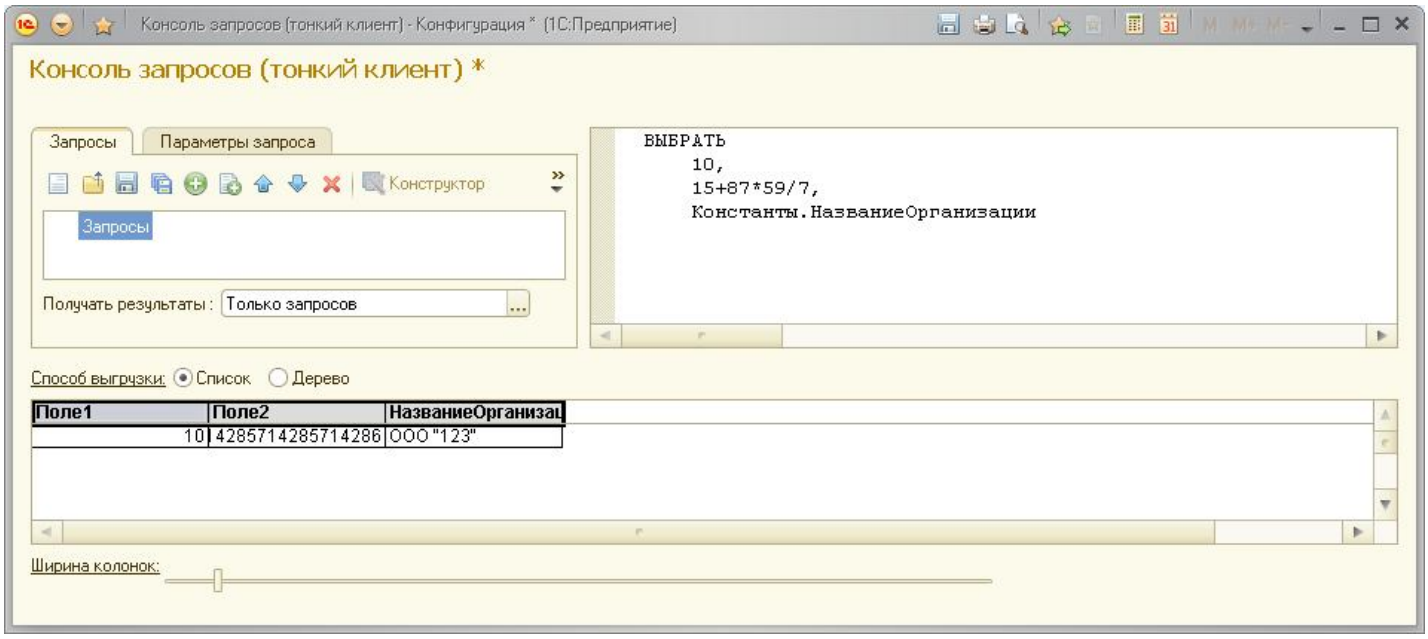


Предложения запроса:

«ВЫБРАТЬ» - единственное обязательное предложение запроса. После этого ключевого слова описываются поля запроса:

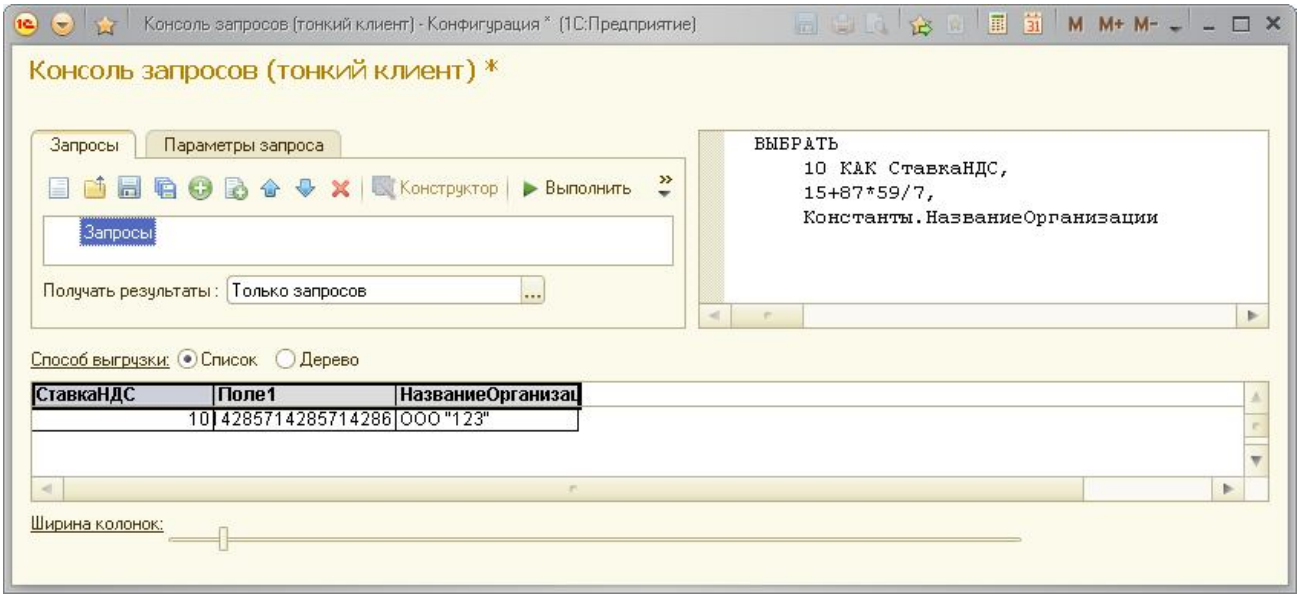
- Числа,
- Математические действия,
- Обращения к полям,
- Обращения к системным функциям,
- и т.д.

Выполним запрос:



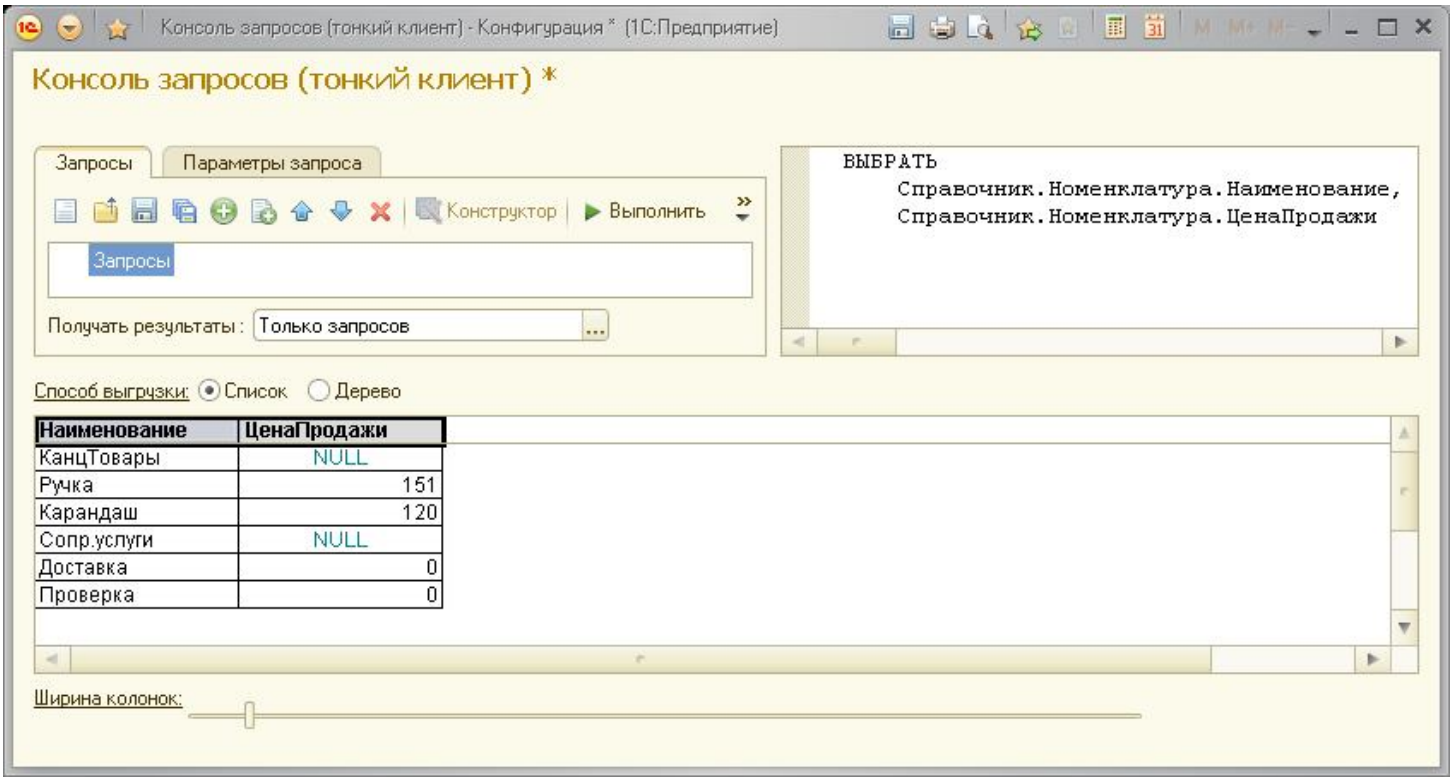
Свои названия полей можно описать с помощью синонимов – ключевое слово «КАК»:

```
Поле КАК Синоним
```



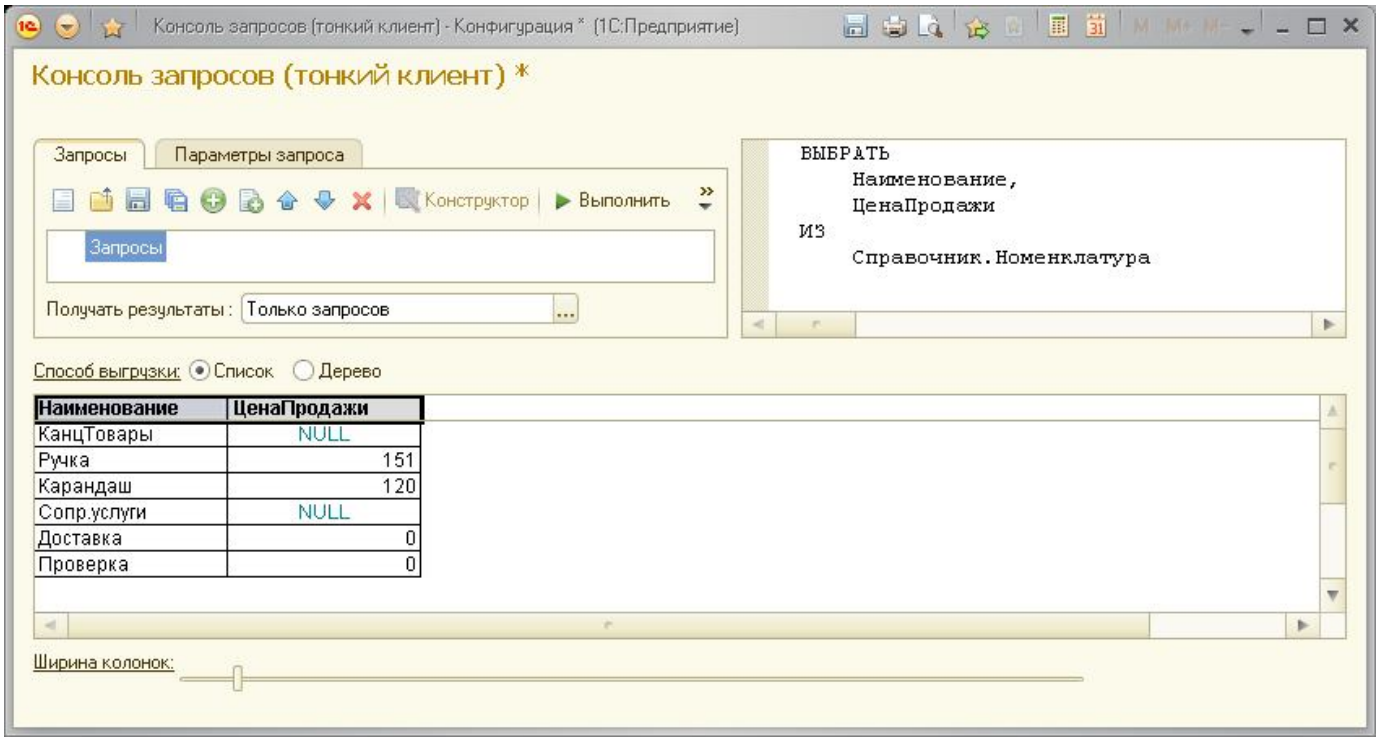
Можно использовать различные функции («ДобавитьКДате», «РазностьДат» и т.д.).

Можно обращаться целиком к таблицам:



Можно использовать обращение с помощью предложения «ИЗ»

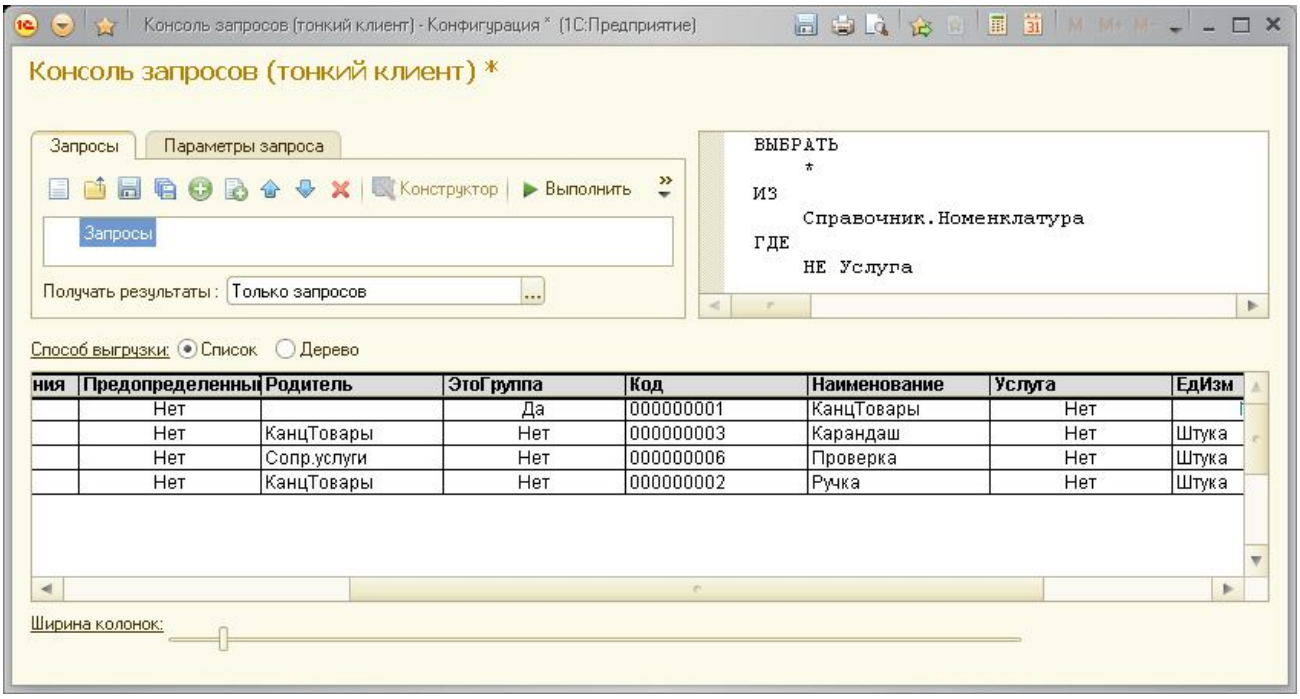
```
ВЫБРАТЬ
Поля
ИЗ
      Источники полей
```



«*» - если надо выбрать все поля из указанного источника.

Предложение «ГДЕ» - если надо ограничить выборку данных

```
ГДЕ Условие
```



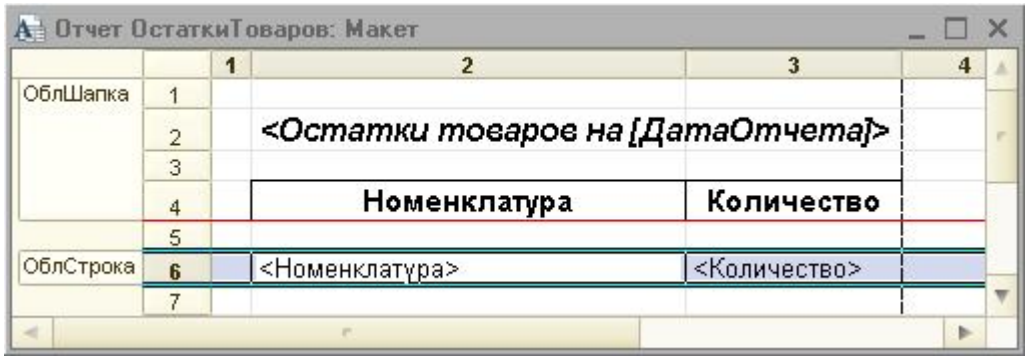
также допустимы составные условия.

Создадим отчет с остатками товаров «Остатки товаров»:

Свойство	Значение
Имя	ОстаткиТоваров
Синоним	Остатки товаров

Создадим макет отчета:

Закладка «Макеты»	
Свойство	**Значение**
Имя	Макет
Синоним	Макет
Тип макета	Табличный документ



Создадим форму отчета:

Добавим реквизиты:

Свойство	**Значение**
Дата отчета	
Имя	ДатаОтчета
Заголовок	Дата отчета
Тип	Дата
Табличный документ	
Имя	ТабДок
Заголовок	Таб док
Тип	ТабличныйДокумент

Добавим «Команду формы»:

Свойство	**Значение**
Сформировать	
Имя	Сформировать
Заголовок	Сформировать

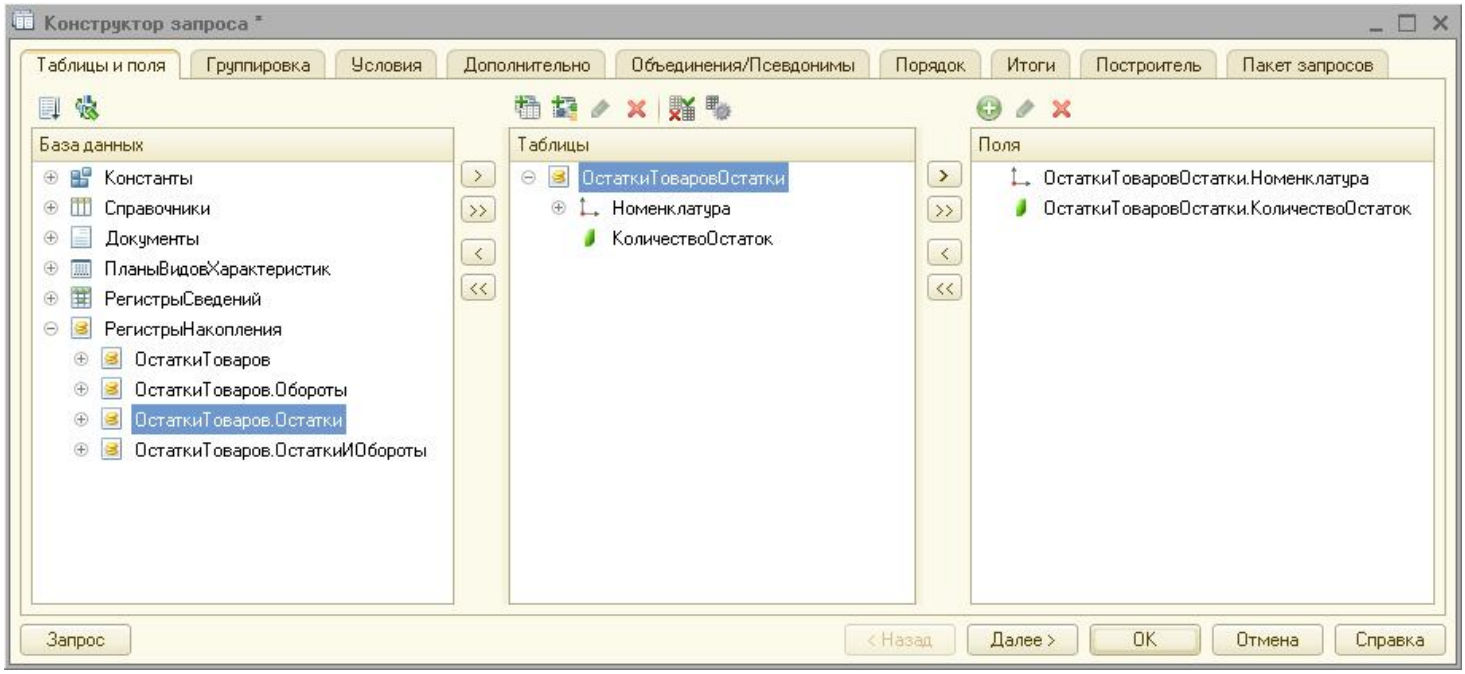
И опишем ее действие:

```
&НаКлиенте
Процедура Сформировать(Команда)
     СформироватьОтчет(ТабДок, ДатаОтчета);
КонецПроцедуры

&НаСервереБезКонтекста
```

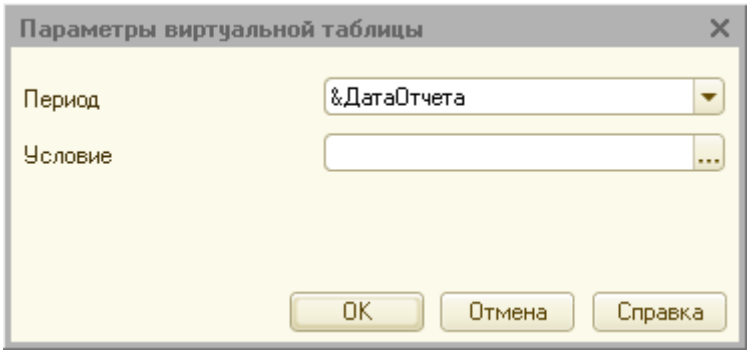
```
Процедура СформироватьОтчет(ТабДок, ДатаОтчета)

    Макет = Отчеты.ОстаткиТоваров.ПолучитьМакет("Макет");
    ОблШапка = Макет.ПолучитьОбласть("ОблШапка");
    ОблСтрока = Макет.ПолучитьОбласть("ОблСтрока");

    ТабДок.Очистить();

    ОблШапка.Параметры.ДатаОтчета = ДатаОтчета;

    Запрос = Новый Запрос;
    Запрос.Текст =
    "";

КонецПроцедуры
```

Создадим сам текст запроса с помощью «Конструктора запроса»

Обращаемся к виртуальной таблице «Остатки» регистра накопления.
Виртуальная таблица – это механизм, анализирующий реальные таблицы остатков и оборотов и «Таблицу движений» и возвращает данные в оптимизированном для чтения виде.



«КоличествоОстаток» - это рассчитываемое поле, которое, если дата отчета указана не будет, то выберет данные из «Основных итогов». Если будет указана дата, и она будет являться началом/концом месяца – выберет из временных таблиц итогов. Если дата произвольная, то система временно рассчитает остаток на нее.
«Дата расчета» указывается в параметрах виртуальной таблицы. Т.к. виртуальная таблица является функцией, которая возвращает значение, то параметры таблицы – это параметры функции:



Итого запрос:

```
ВЫБРАТЬ
	ОстаткиТоваровОстатки.Номенклатура,
	ОстаткиТоваровОстатки.КоличествоОстаток
ИЗ
	РегистрНакопления.ОстаткиТоваров.Остатки(&ДатаОтчета, ) КАК ОстаткиТова-
ровОстатки
```

Итого:

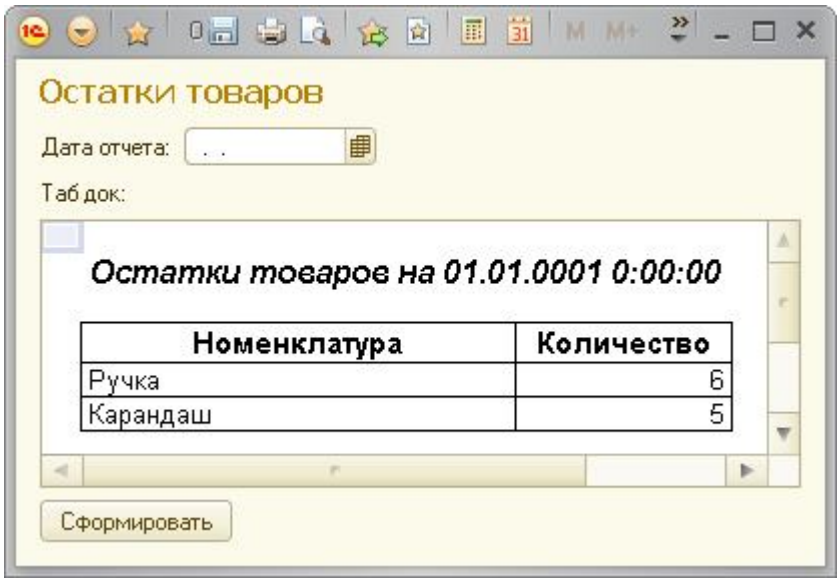
```
&НаКлиенте
Процедура Сформировать(Команда)
	СформироватьОтчет(ТабДок, ДатаОтчета);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура СформироватьОтчет(ТабДок, ДатаОтчета)

	Макет = Отчеты.ОстаткиТоваров.ПолучитьМакет("Макет");
	ОблШапка = Макет.ПолучитьОбласть("ОблШапка");
	ОблСтрока = Макет.ПолучитьОбласть("ОблСтрока");

	ТабДок.Очистить();

	ОблШапка.Параметры.ДатаОтчета = ДатаОтчета;
	ТабДок.Вывести(ОблШапка);

	Запрос = Новый Запрос;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	Номенклатура,
	|	КоличествоОстаток
	|ИЗ
	|	РегистрНакопления.ОстаткиТоваров.Остатки(&ДатаОтчета, )";

	Запрос.УстановитьПараметр("ДатаОтчета", ДатаОтчета);

	Результат = Запрос.Выполнить();

	Выборка = Результат.Выбрать();
```

```
    Пока Выборка.Следующий() Цикл
        ОблСтрока.Параметры.Номенклатура = Выборка.Номенклатура;
        ОблСтрока.Параметры.Количество = Выборка.КоличествоОстаток;

        ТабДок.Вывести(ОблСтрока);
    КонецЦикла;
```

КонецПроцедуры

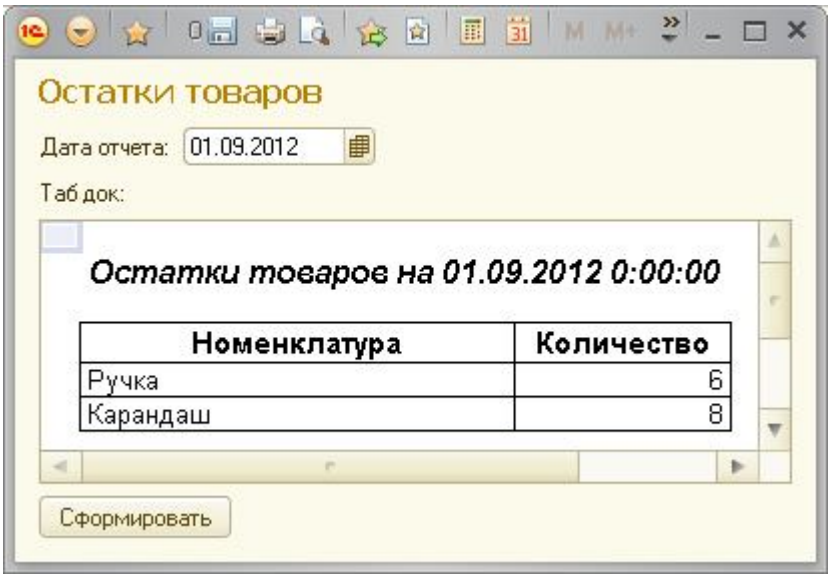
В режиме исполнения:

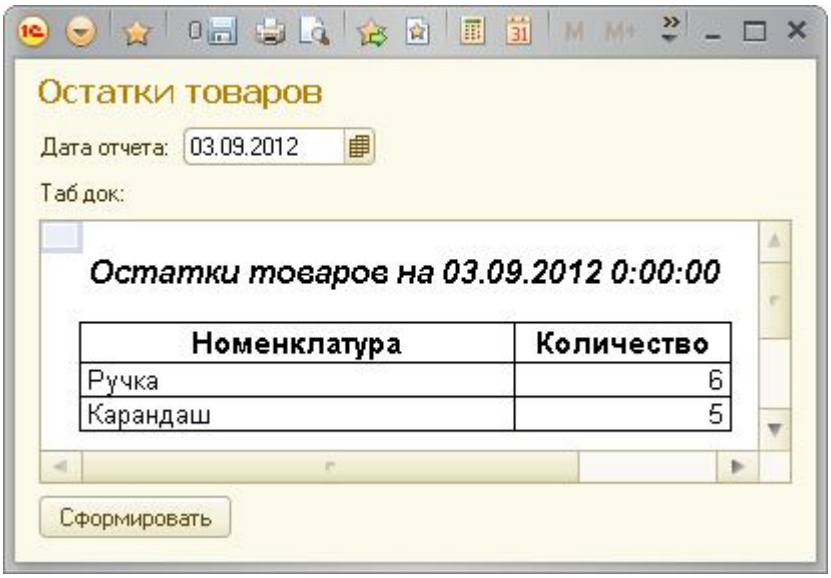
Не укажем дату:



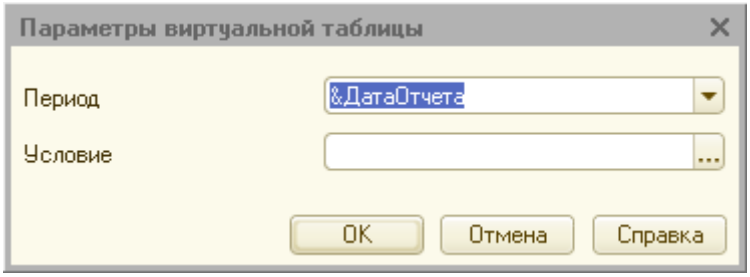
Отчет сформирован на 30.11.3999.

Остатки на указанную дату:




Остатки товаров
Дата отчета: 03.09.2012
Таб док:
Остатки товаров на 03.09.2012 0:00:00
Номенклатура
Количество
Ручка
6
Карандаш
5
Сформировать

```
ВЫБРАТЬ
      Номенклатура,
      КоличествоОстаток
ИЗ
      РегистрНакопления.ОстаткиТоваров.Остатки(&ДатаОтчета, )";
```



Т.е. в виртуальной таблице первый параметр – это «Период», а второй – «Условие». Например:

```
РегистрНакопления.ОстаткиТоваров.Остатки(&ДатаОтчета,
Номенклатура = &Номенклатура)";
```

Также условие можно задать в предложении «ГДЕ»

```
ВЫБРАТЬ
      Номенклатура,
      КоличествоОстаток
ИЗ
      РегистрНакопления.ОстаткиТоваров.Остатки(&ДатаОтчета, )";
ГДЕ
      Номенклатура = &Номенклатура
```

Если параметр передавать как параметр виртуальной таблицы, то система не рассчитает данные, которые не соответствуют этому параметру, а если использовать условие после «ГДЕ», то система рассчитает все данные, а потом отбросит те, которые не соответствуют условию.

Создадим отчет с информацией по купленным товарам:

Закупки			
Контрагент	**Товар**	**Количество**	**Сумма**

```
ВЫБРАТЬ *
ИЗ
Документ.Приходная.Товары
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Поступление това	1	Карандаш	120	4	480
Поступление това	2	Ручка	140	3	420
Поступление това	2	Ручка	190	1	190
Поступление това	1	Карандаш	150	1	150

Это данные табличных частей всех документов «Приходная».

```
ВЫБРАТЬ *
ИЗ
Документ.Приходная
```

Ссылка	ВерсияДанных	ПометкаУдалени	Номер	Дата	Проведен	Контрагент	Договор	Товары
Поступление това	AAAAFQAAAAA=	Нет	000000001	22.08.2012 0:15:2	Да	СВЕТ ОАО	Договор №3	ЕЗУЛЬТАТ ЗАПРО(
Поступление това	AAAAEgAAAAA=	Нет	000000002	22.08.2012 23:50:5	Да	СВЕТ ОАО	Договор №3	ЕЗУЛЬТАТ ЗАПРО(

Это данные «Шапок» всех документов «Приходная».

Из табличной части «Товары» документа «Приходная» можно вытащить «Контрагента» - через «Ссылку» на документ, которому принадлежит табличная часть:

```
ВЫБРАТЬ Ссылка.Контрагент, *
ИЗ
Документ.Приходная.Товары
```

Контрагент	Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
СВЕТ ОАО	Поступление товаров 00	1	Карандаш	120	4	480
СВЕТ ОАО	Поступление товаров 00	2	Ручка	140	3	420
СВЕТ ОАО	Поступление товаров 00	2	Ручка	190	1	190
СВЕТ ОАО	Поступление товаров 00	1	Карандаш	150	1	150

Но «Ссылка» - это уже не табличная часть документа «Приходная». Т.е. здесь используется уже две таблицы.
Есть ограничение на количество используемых таблиц в запросе – не более 256.

Обратимся еще и к полю справочника «Номенклатура»:

```
ВЫБРАТЬ Ссылка.Контрагент, Номенклатура.ЕдИзм, *
ИЗ
Документ.Приходная.Товары
```

Контрагент	НоменклатураЕдИзм	Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
СВЕТ ОАО	Штука	Поступление товаро	1	Карандаш	120	4	480
СВЕТ ОАО	Штука	Поступление товаро	2	Ручка	140	3	420
СВЕТ ОАО	Штука	Поступление товаро	2	Ручка	190	1	190
СВЕТ ОАО	Штука	Поступление товаро	1	Карандаш	150	1	150

Т.е. в запросе используются уже три таблицы:

- Табличная часть «Товары» документа «Приходная»:

```
Документ.Приходная.Товары
```

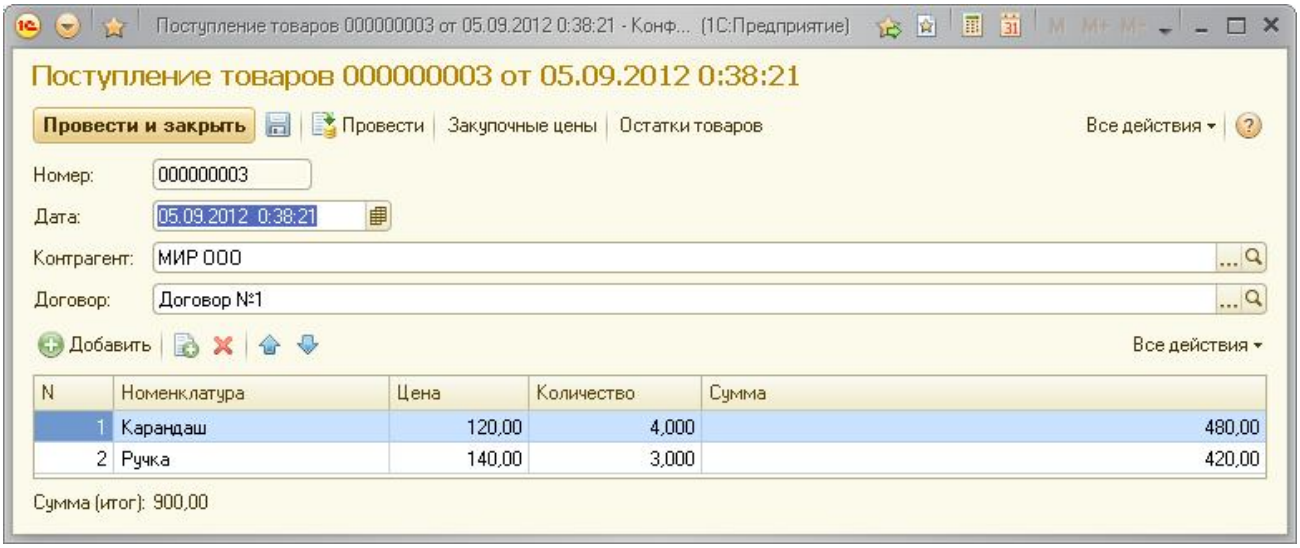
- «Шапка» документа «Приходная»:

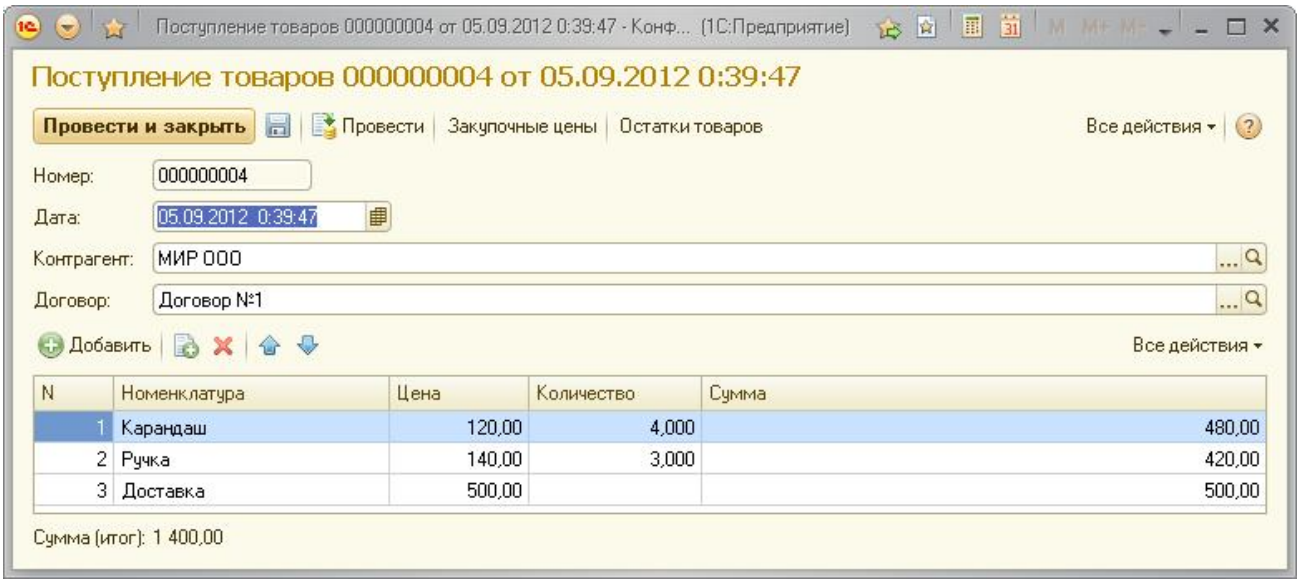
```
Документ.Приходная
```

- Таблица справочника «Номенклатура»:

```
Номенклатура.ЕдИзм
```

Добавим документы «Приходная»






```
ВЫБРАТЬ Ссылка.Контрагент,
Номенклатура,
Количество,
Сумма
ИЗ
Документ.Приходная.Товары
```

Контрагент	Номенклатура	Количество	Сумма
СВЕТ ОАО	Карандаш	4	480
СВЕТ ОАО	Ручка	3	420
СВЕТ ОАО	Ручка	1	190
СВЕТ ОАО	Карандаш	1	150
МИР ООО	Карандаш	4	480
МИР ООО	Ручка	3	420
МИР ООО	Карандаш	4	480
МИР ООО	Ручка	3	420
МИР ООО	Доставка	0	500

Минус – дублирующиеся поля. Чтобы сгруппировать отчет, его поля надо разделить на два типа:
Значение группировки – поля, в которых будет происходить поиск одинаковых значений. В примере это «Контрагент» и «Номенклатура».
Ресурсы группировки – по ним надо указать, что надо сделать (просуммировать, найти максимальное и т.д.).

```
ВЫБРАТЬ Ссылка.Контрагент,
      Номенклатура,
      СУММА(Количество),
      СУММА(Сумма)
ИЗ
      Документ.Приходная.Товары
СГРУППИРОВАТЬ ПО
      Ссылка.Контрагент, Номенклатура
```

Контрагент	Номенклатура	Количество	Сумма
МИР ООО	Ручка	6	840
МИР ООО	Карандаш	8	960
МИР ООО	Доставка	0	500
СВЕТ ОАО	Ручка	4	610
СВЕТ ОАО	Карандаш	5	630

Подведем итоги по купленным товарам:

Если поля, по которым подводятся итоги, выбираются через точку или с ними производятся какие-либо действия, то для них надо вывести Псевдоним.

```
ВЫБРАТЬ Ссылка.Контрагент КАК Контрагент,
      Номенклатура,
      СУММА(Количество) КАК Количество,
      СУММА(Сумма) КАК Сумма
ИЗ
      Документ.Приходная.Товары
СГРУППИРОВАТЬ ПО
      Ссылка.Контрагент, Номенклатура
ИТОГИ
      СУММА(Сумма) ПО ОБЩИЕ
```

Контрагент	Номенклатура	Количество	Сумма
NULL	NULL	NULL	3 540
МИР ООО	Ручка	6	840
МИР ООО	Карандаш	8	960
МИР ООО	Доставка	0	500
СВЕТ ОАО	Ручка	4	610
СВЕТ ОАО	Карандаш	5	630

Можно подвести итоги по «Контрагентам»:

```
ВЫБРАТЬ Ссылка.Контрагент КАК Контрагент,
      Номенклатура,
      СУММА(Количество) КАК Количество,
      СУММА(Сумма) КАК Сумма
ИЗ
      Документ.Приходная.Товары
СГРУППИРОВАТЬ ПО
      Ссылка.Контрагент, Номенклатура
ИТОГИ
      СУММА(Сумма) ПО Контрагент
```

Можно одновременно подвести «Общие» итоги и по «Контрагенту»:

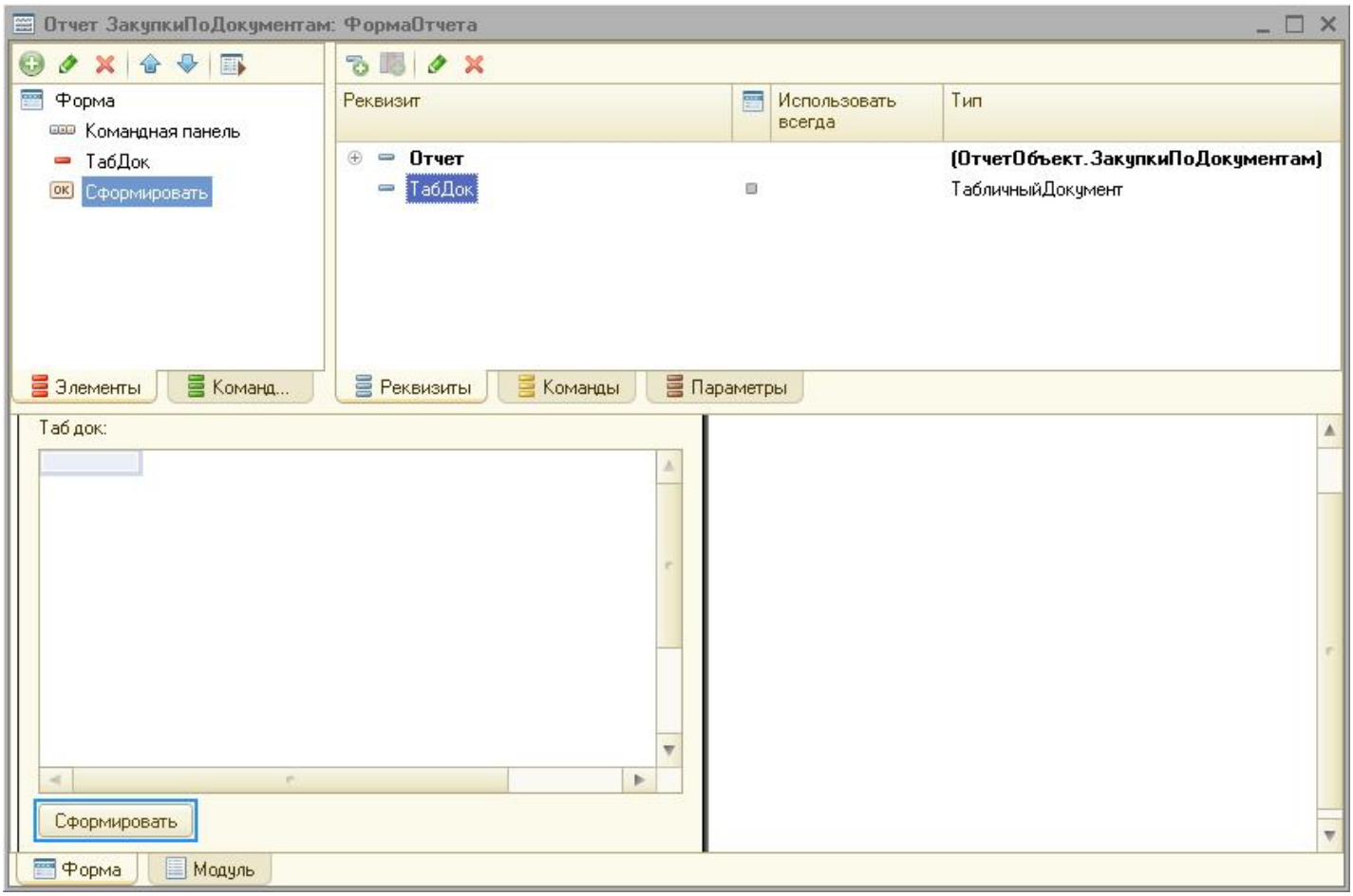
```
ВЫБРАТЬ Ссылка.Контрагент КАК Контрагент,
      Номенклатура,
      СУММА(Количество) КАК Количество,
      СУММА(Сумма) КАК Сумма
ИЗ
      Документ.Приходная.Товары
СГРУППИРОВАТЬ ПО
      Ссылка.Контрагент, Номенклатура
ИТОГИ
      СУММА(Сумма) ПО ОБЩИЕ, Контрагент
```

Контрагент	Номенклатура	Количество	Сумма
NULL	NULL	NULL	3 540
МИР ООО	NULL	NULL	2 300
МИР ООО	Ручка	6	840
МИР ООО	Карандаш	8	960
МИР ООО	Доставка	0	500
СВЕТ ОАО	NULL	NULL	1 240
СВЕТ ОАО	Ручка	4	610
СВЕТ ОАО	Карандаш	5	630

Создадим отчет «Закупки по документам»

Свойство	Значение
Имя	ЗакупкиПоДокументам
Синоним	Закупки по документам

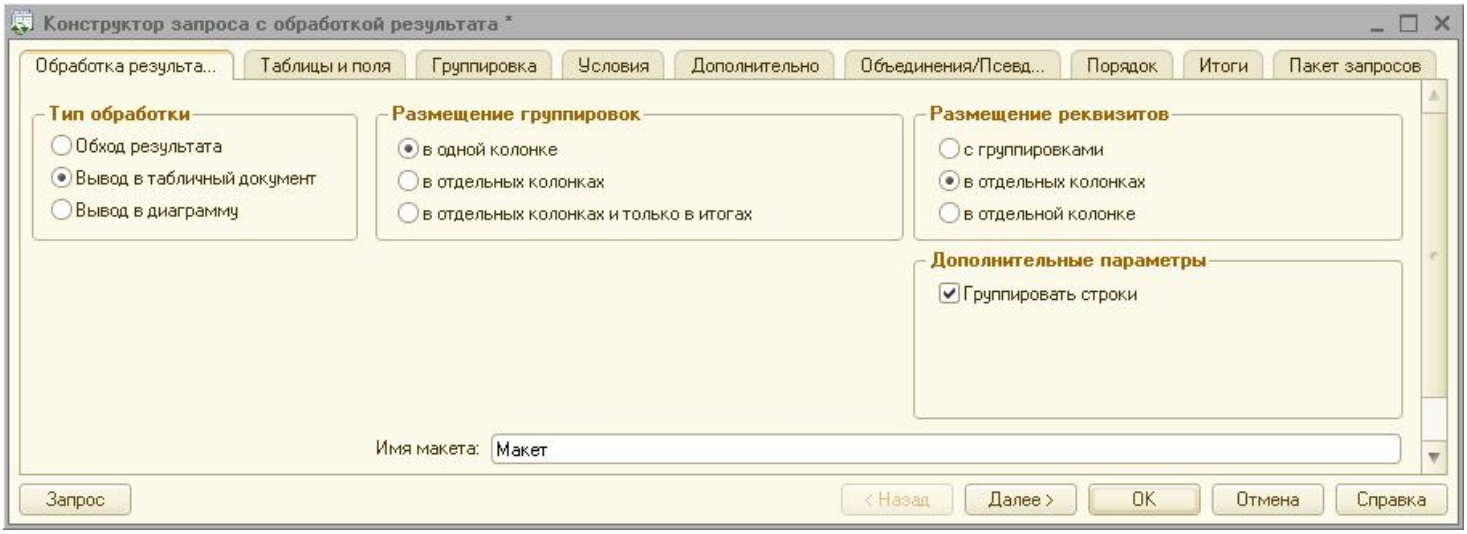
Создадим форму отчета:
Создадим реквизит «ТабДок» и команду «Сформировать», поместим их в форму и опишем действие команды «Сформировать»:



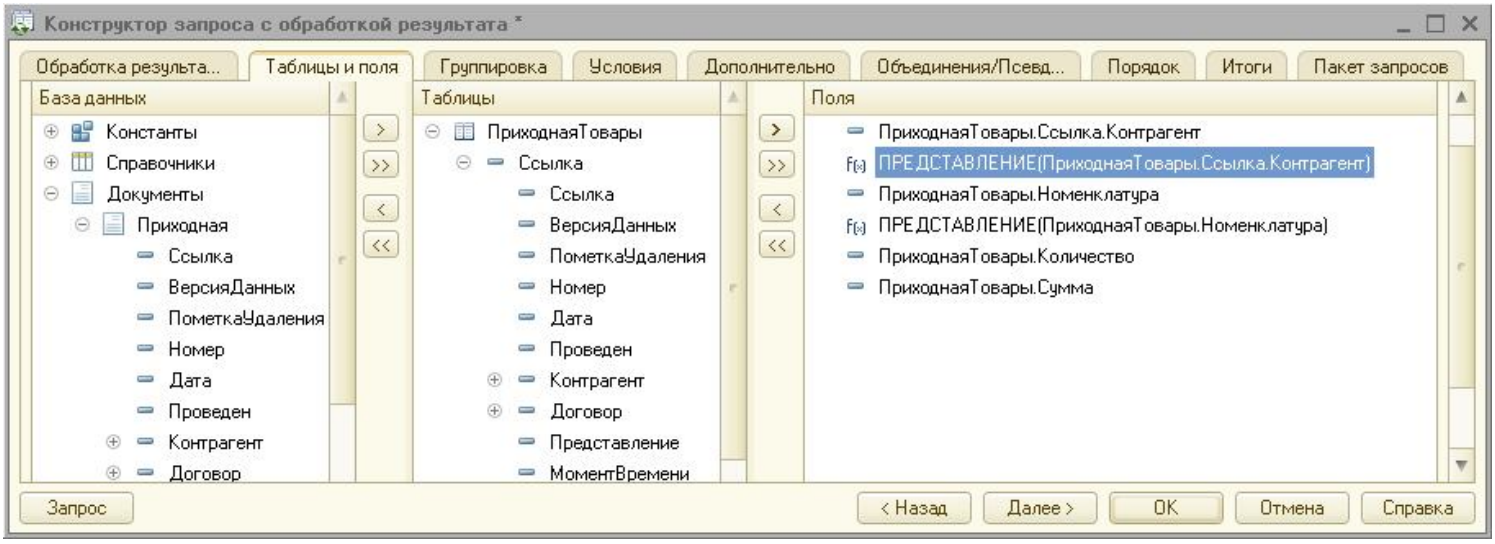
Запрос будем выполнять на Сервере:

```
&НаКлиенте
Процедура Сформировать(Команда)
    СформироватьОтчет(ТабДок);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура СформироватьОтчет(ТабДок)

КонецПроцедуры
```

В процедуре сформировать: «Текст – Конструктор запроса с обработкой результата»

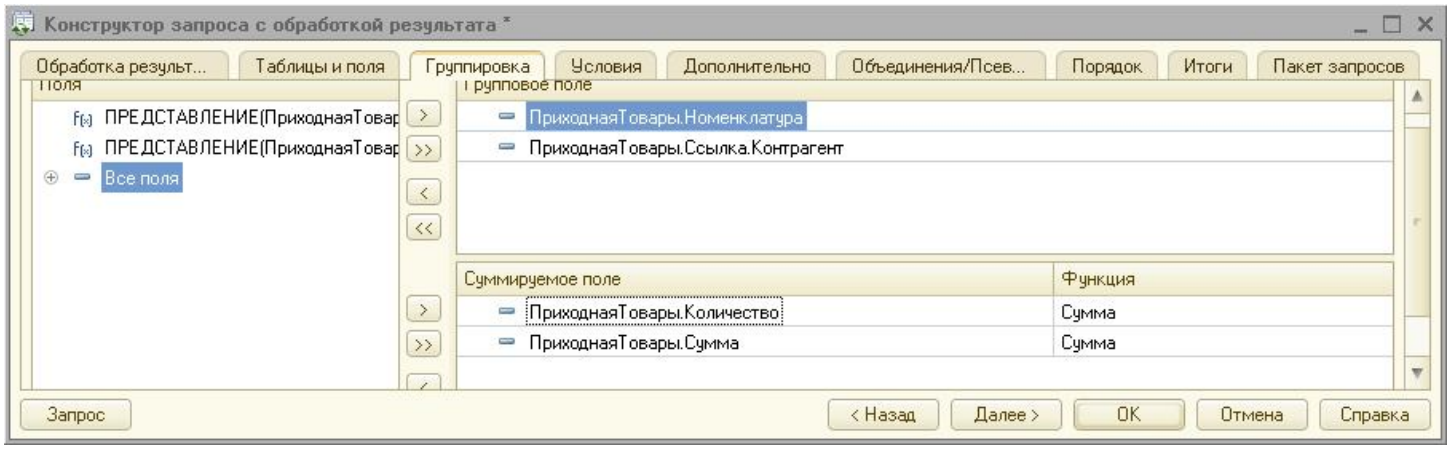


На закладке «Таблицы и поля» указываются поля запроса:

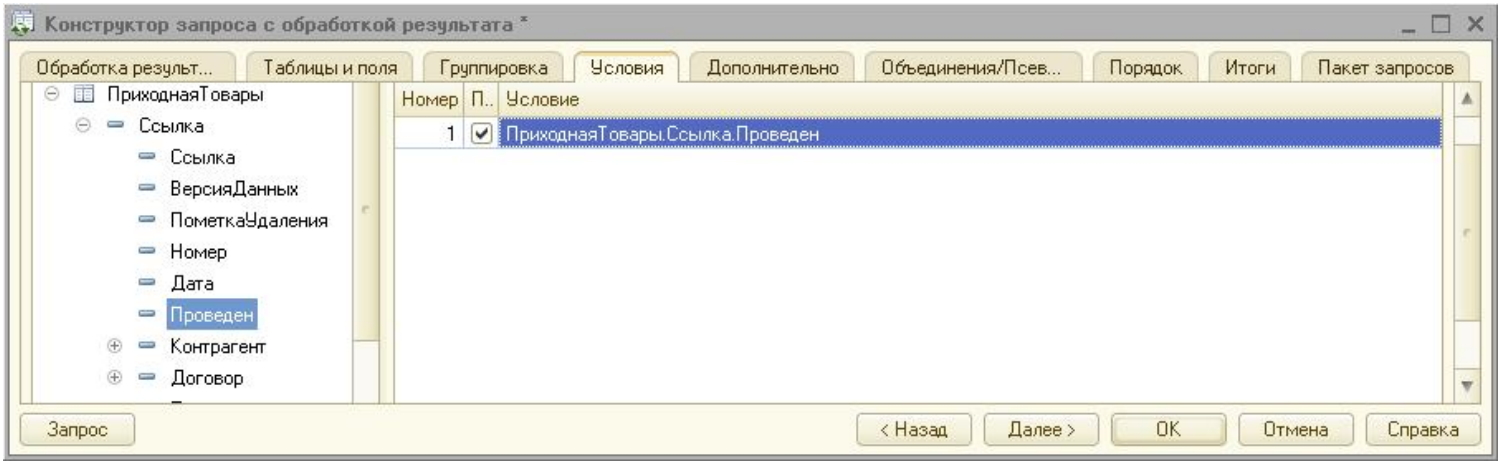


«ПРЕДСТАВЛЕНИЕ(Поле)» - это символьное представление значений. Например, тип значения «Контрагент» - Ссылка – идентификатор, который пользователю не понятен. Если запросом получать только Ссылку, то система обратится к БД и по этой Ссылке получит Представление (понятное пользователю), т.е. это опять запрос.
В примере запросом сразу будет получена и Ссылка и Представление, т.е. обращается к БД только один раз.

Закладка «Группировка» - сгруппируем записи по Контрагенту и Номенклатуре.

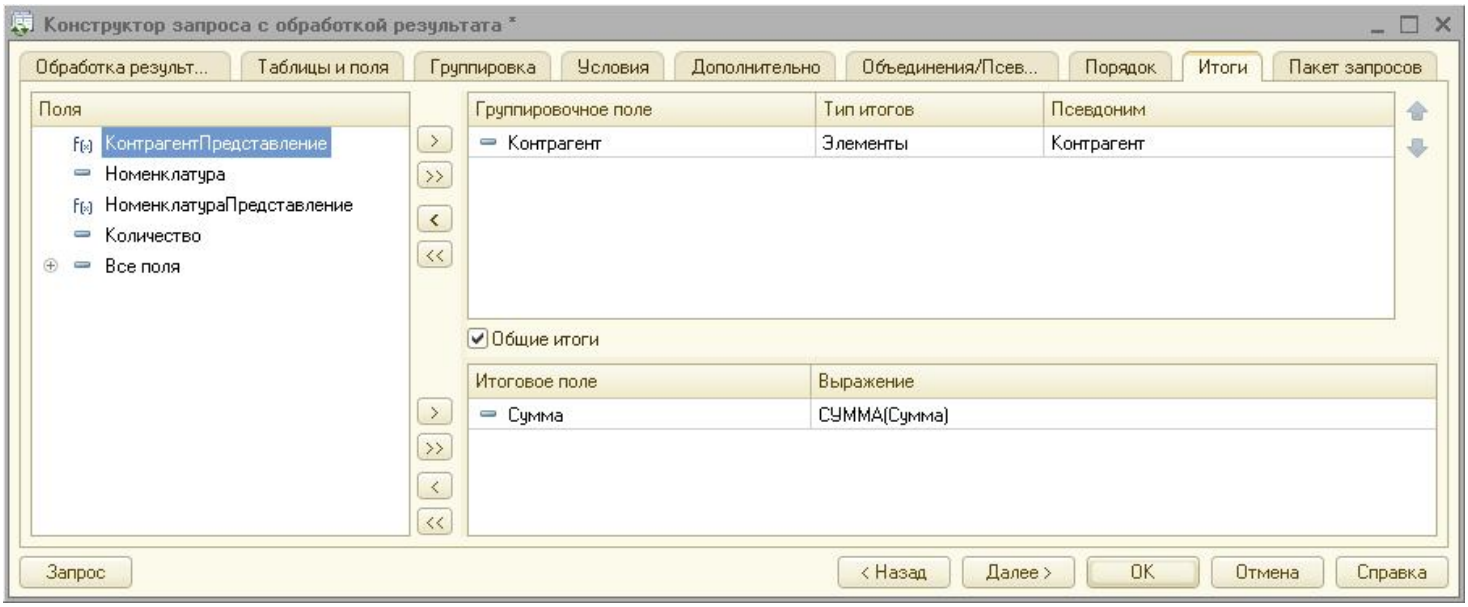


Закладка «Условия»
Будем рассматривать только проведенные документы:

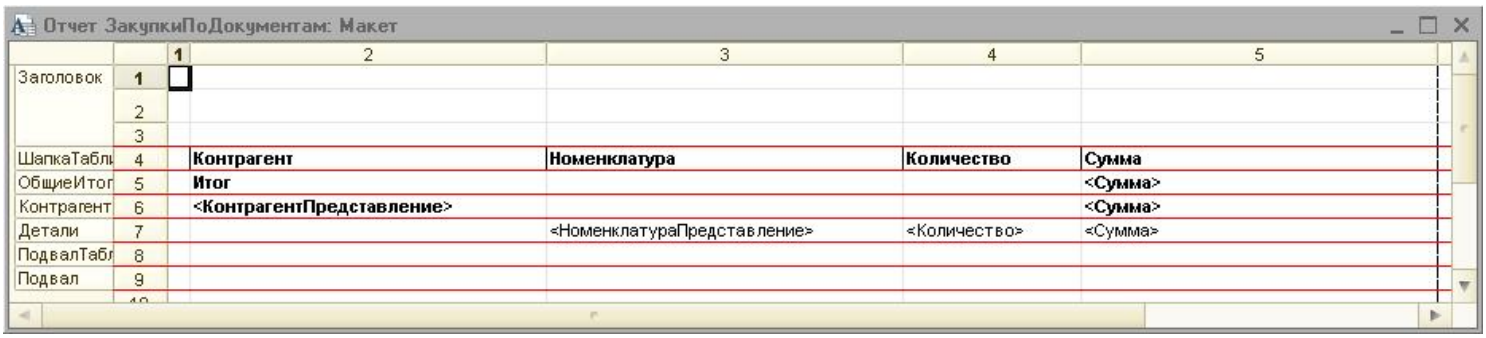


Закладка «Объединения/Псевдонимы» - для задания синонимов полям.

Закладка «Итоги» - посчитаем сумму по Контрагенту и Общие итоги:



Итого получится:
Макет:



Полностью работу отчета с табличным документом:

```
Макет = Отчеты.ЗакупкиПоДокументам.ПолучитьМакет("Макет");
Запрос = Новый Запрос;
Запрос.Текст =
     "ВЫБРАТЬ
     |    ПриходнаяТовары.Ссылка.Контрагент КАК Контрагент,
     |    ПРЕДСТАВЛЕНИЕ(ПриходнаяТовары.Ссылка.Контрагент),
     |    ПриходнаяТовары.Номенклатура,
     |    ПРЕДСТАВЛЕНИЕ(ПриходнаяТовары.Номенклатура),
     |    СУММА(ПриходнаяТовары.Количество) КАК Количество,
     |    СУММА(ПриходнаяТовары.Сумма) КАК Сумма
     |ИЗ
     |    Документ.Приходная.Товары КАК ПриходнаяТовары
     |ГДЕ
     |    ПриходнаяТовары.Ссылка.Проведен
     |
     |СГРУППИРОВАТЬ ПО
     |    ПриходнаяТовары.Ссылка.Контрагент,
     |    ПриходнаяТовары.Номенклатура
     |ИТОГИ
     |    СУММА(Сумма)
     |ПО
     |    ОБЩИЕ,
     |    Контрагент";

Запрос.УстановитьПараметр("Проведен", Проведен);

Результат = Запрос.Выполнить();

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
ОбластьПодвал = Макет.ПолучитьОбласть("Подвал");
ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");
ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");
ОбластьОбщийИтог = Макет.ПолучитьОбласть("ОбщиеИтоги");
ОбластьКонтрагент = Макет.ПолучитьОбласть("Контрагент");
ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();
ТабДок.Вывести(ОбластьЗаголовок);
ТабДок.Вывести(ОбластьШапкаТаблицы);
ТабДок.НачатьАвтогруппировкуСтрок();

ВыборкаОбщийИтог = Результат.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

ВыборкаОбщийИтог.Следующий(); // Общий итог
ОбластьОбщийИтог.Параметры.Заполнить(ВыборкаОбщийИтог);
ТабДок.Вывести(ОбластьОбщийИтог, ВыборкаОбщийИтог.Уровень());

ВыборкаКонтрагент = ВыборкаОбщийИтог.Выбрать(ОбходРезультатаЗапроса.
ПоГруппировкам);
```

```
Пока ВыборкаКонтрагент.Следующий() Цикл
      ОбластьКонтрагент.Параметры.Заполнить(ВыборкаКонтрагент);
      ТабДок.Вывести(ОбластьКонтрагент, ВыборкаКонтрагент.Уровень());

      ВыборкаДетальныеЗаписи = ВыборкаКонтрагент.Выбрать();

      Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
            ОбластьДетальныхЗаписей.Параметры.Заполнить(ВыборкаДетальныеЗаписи);
            ТабДок.Вывести(ОбластьДетальныхЗаписей,        ВыборкаДетальныеЗапи-
си.Уровень());
      КонецЦикла;
КонецЦикла;

ТабДок.ЗакончитьАвтогруппировкуСтрок();
ТабДок.Вывести(ОбластьПодвалТаблицы);
ТабДок.Вывести(ОбластьПодвал);
```

В предложении «ИЗ» запрос использует синоним «ПриходнаяТовары», но здесь его можно не использовать (соответственно, убрать и в полях), т.к. мы работаем с одной таблицей.

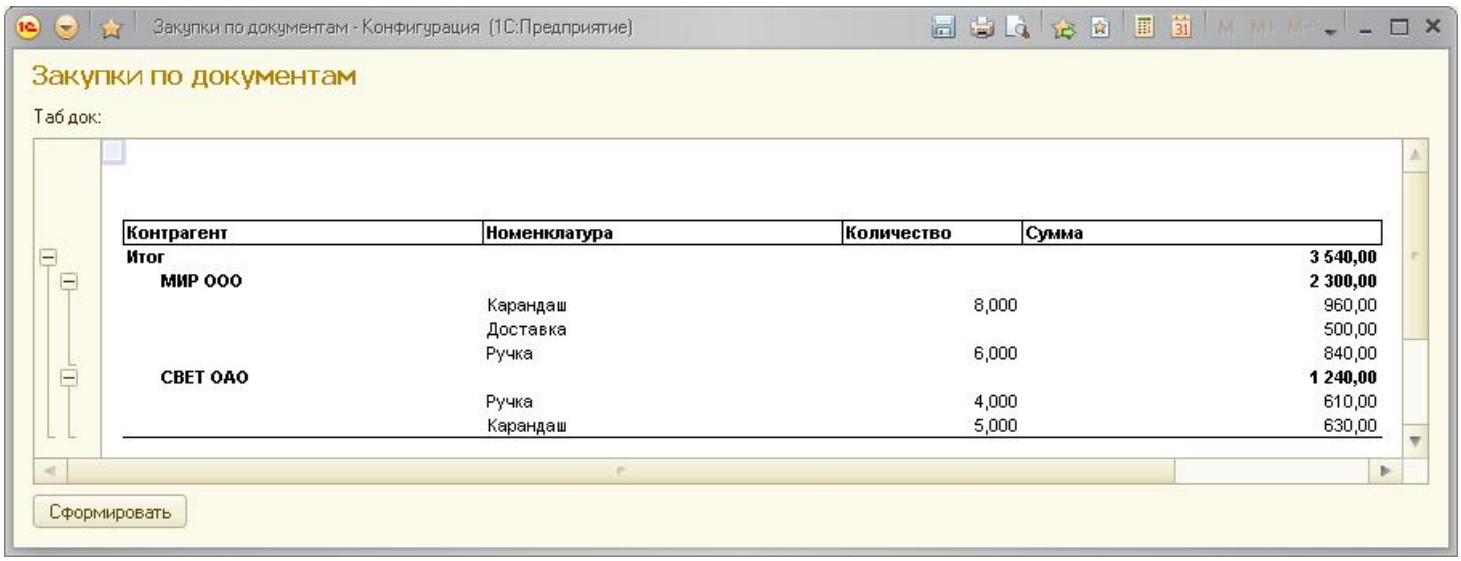
Запрос автоматом устанавливает параметр «Проведен»:

```
Запрос.УстановитьПараметр("Проведен", Проведен);
```

но он в данном случае не нужен, т.к. в запросе уже есть условие:

```
ГДЕ
      ПриходнаяТовары.Ссылка.Проведен
```

В режиме исполнения:



Контрагент	Номенклатура	Количество	Сумма
Итог			**3 540,00**
МИР ООО			**2 300,00**
	Карандаш	8,000	960,00
	Доставка		500,00
	Ручка	6,000	840,00
СВЕТ ОАО			**1 240,00**
	Ручка	4,000	610,00
	Карандаш	5,000	630,00

Создадим отчет «Закупки-Продажи»
Отчет должен формироваться за определенный, заданный период.

Закупки-Продажи		
Номенклатура	**Закупки**	**Продажи**

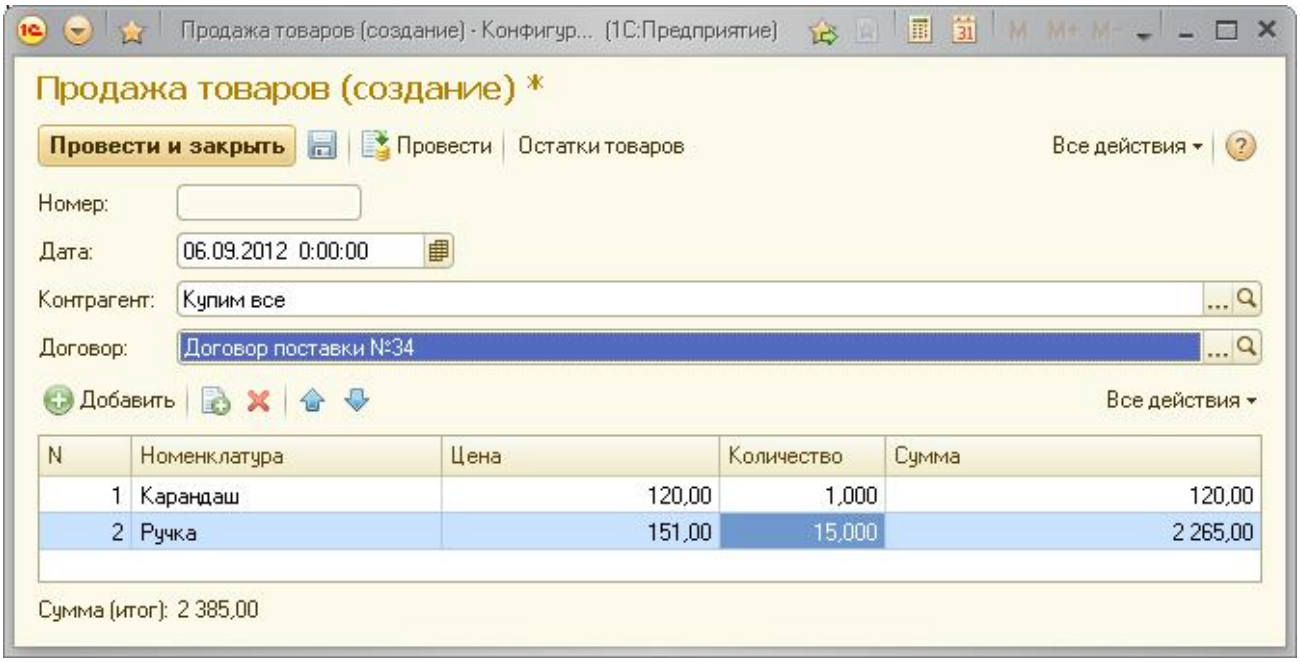
Данные будем брать из документов «Приходная» и «Расходная».

Есть две технологии:
1. Выборка одного запроса накладывается на выборку другого – сначала выберем необходимые поля из табличной части документов «Приходная», затем напишем точно такой же запрос, но для документов «Расходная», а между ними напишем ключевое слово:
«ОБЪЕДЕНИТЬ ВСЕ».

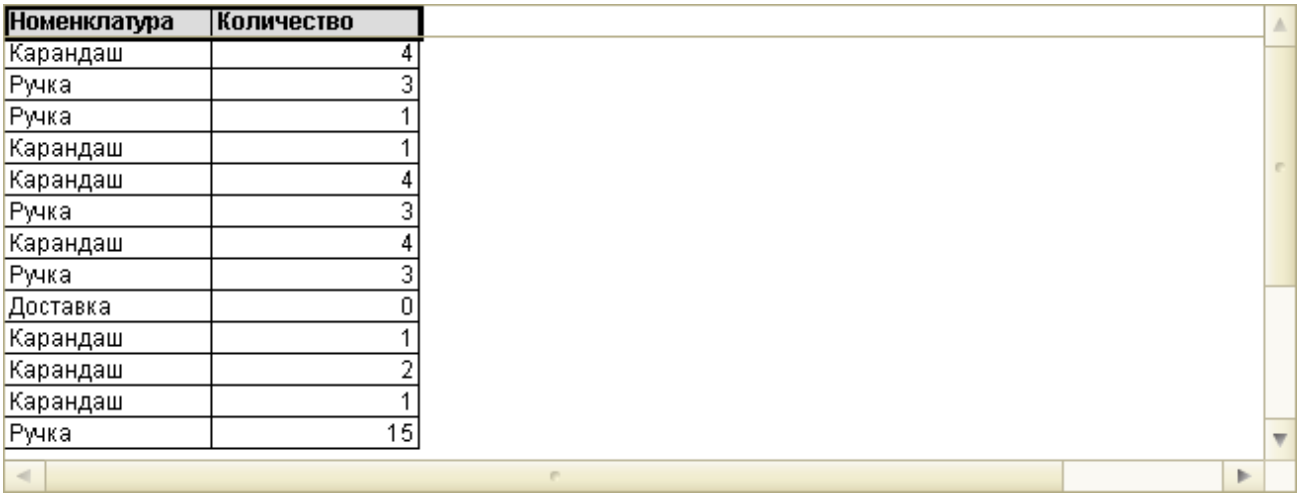
```
ВЫБРАТЬ
      Номенклатура,
      Количество
ИЗ
      Документ.Приходная.Товары

ОБЪЕДИНИТЬ ВСЕ

ВЫБРАТЬ
      Номенклатура,
      Количество
ИЗ
      Документ.Расходная.Товары
```

Номенклатура	Количество
Карандаш	4
Ручка	3
Ручка	1
Карандаш	1
Карандаш	4
Ручка	3
Карандаш	4
Ручка	3
Доставка	0
Карандаш	1
Карандаш	2

Создадим еще один документ «Расходная». К нему создадим Контрагента и его Договор:



И теперь результат отчета (ниже «Доставки» идут продажи):



Номенклатура	Количество
Карандаш	4
Ручка	3
Ручка	1
Карандаш	1
Карандаш	4
Ручка	3
Карандаш	4
Ручка	3
Доставка	0
Карандаш	1
Карандаш	2
Карандаш	1
Ручка	15

Надо разделить Закупки и Продажи:
В первом запросе зададим синонимы для поля Количество, причем эти синонимы распространяются и на поля второго запроса.

```
ВЫБРАТЬ
      Номенклатура,
      Количество КАК Закупки,
      0 КАК Продажи
ИЗ
      Документ.Приходная.Товары

ОБЪЕДИНИТЬ ВСЕ

ВЫБРАТЬ
      Номенклатура,
      0,
      Количество
ИЗ
      Документ.Расходная.Товары
```

Номенклатура	Закупки	Продажи
Карандаш	4	0
Ручка	3	0
Ручка	1	0
Карандаш	1	0
Карандаш	4	0
Ручка	3	0
Карандаш	4	0
Ручка	3	0
Доставка	0	0
Карандаш	0	1
Карандаш	0	2
Карандаш	0	1
Ручка	0	15

Чтобы сгруппировать поля, надо результат того, что получилось, использовать как источник (Вложенный запрос):

```
ВЫБРАТЬ
      Номенклатура,
      СУММА(Закупки),
      СУММА(Продажи)
ИЗ
(ВЫБРАТЬ
      Номенклатура,
      Количество КАК Закупки,
      0 КАК Продажи
ИЗ
      Документ.Приходная.Товары

ОБЪЕДИНИТЬ ВСЕ

ВЫБРАТЬ
      Номенклатура,
      0,
      Количество
ИЗ
      Документ.Расходная.Товары) КАК ВложенныйЗапрос
      СГРУППИРОВАТЬ ПО Номенклатура
```

Номенклатура	Закупки	Продажи
Ручка	10	15
Карандаш	13	4
Доставка	0	0

При объединении, количество полей у объединяемых запросов должно быть одинаковым.

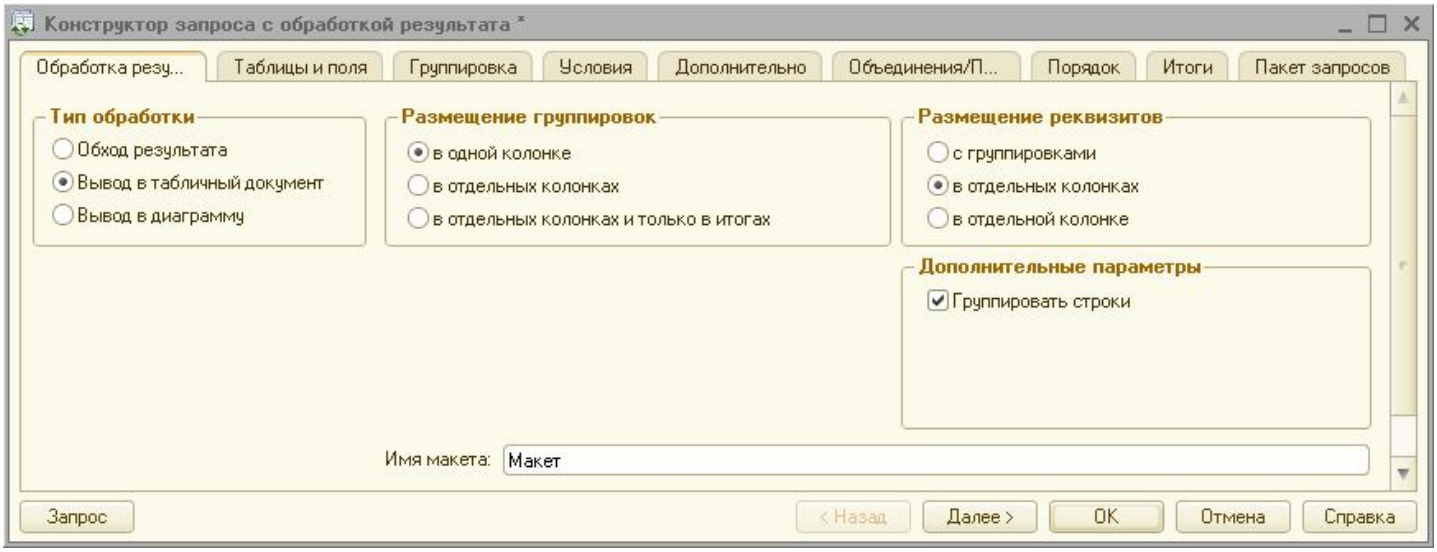
Создадим отчет «Закупки-Продажи объединение» по данному запросу:

Свойство	Значение
Имя	ЗакупкиПродажи_Объединение
Синоним	Закупки-продажи объединение

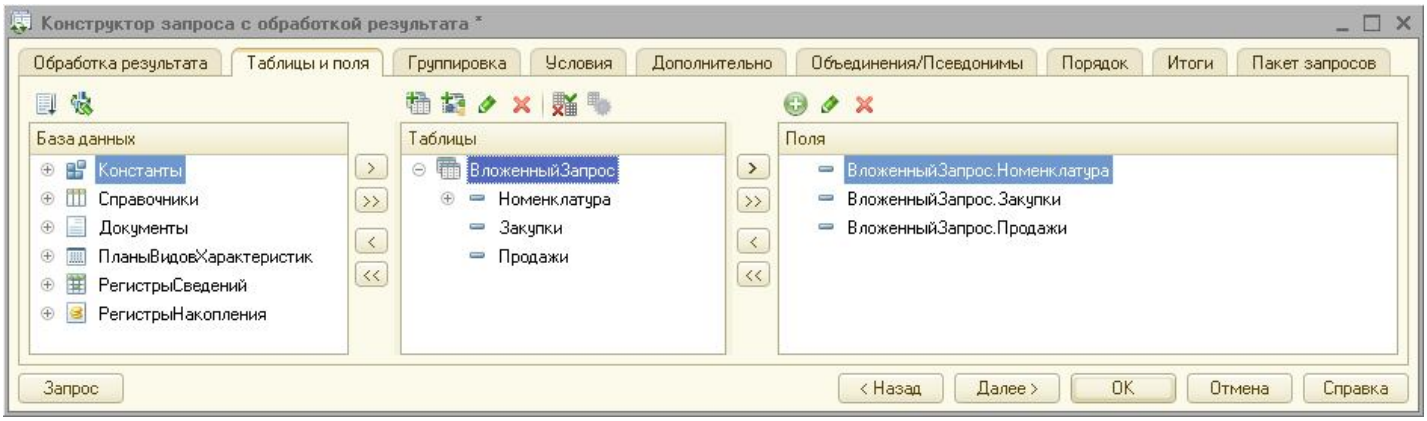
Создадим форму отчета:
Создадим команду «Сформировать», поместим ее в форму и опишем действие:
На Сервере создадим процедуру «СформироватьОтчет» и в ней запустим «Конструктор запроса с обработкой результата»

```
&НаКлиенте
Процедура Сформировать(Команда)

КонецПроцедуры

&НаСервереБезКонтекста
Процедура СформироватьОтчет(ТабДок)

КонецПроцедуры
```

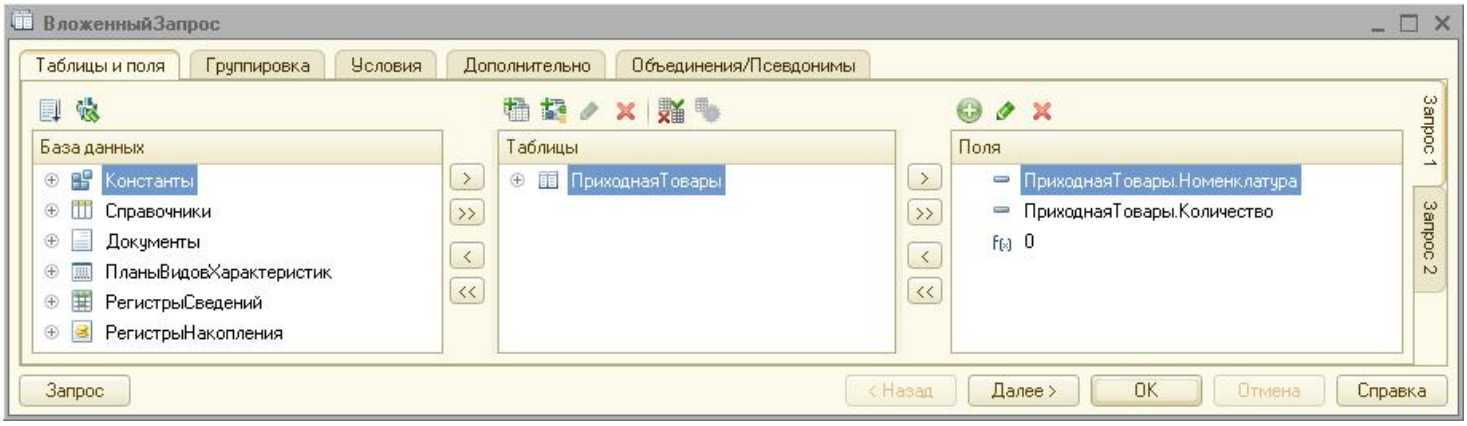
Закладка «Обработка результата»



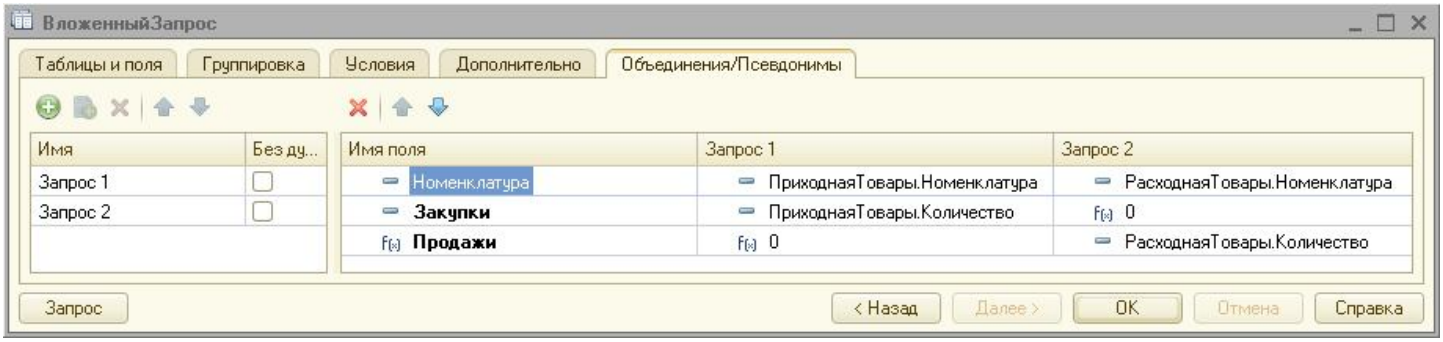
Закладка «Таблицы и поля» - вставим запрос из «Консоли запросов».
Вложенный запрос:



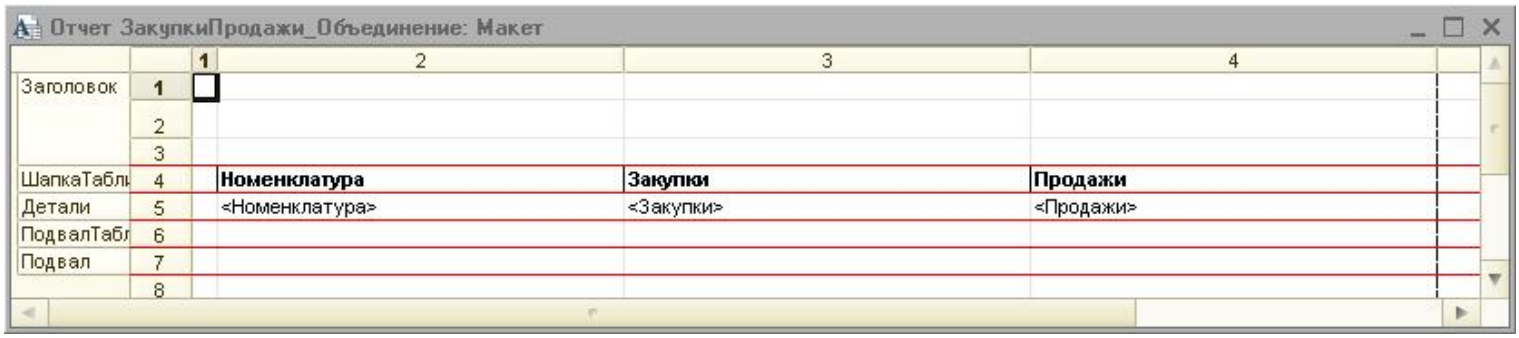
При нажатии на кнопку «Изменить» над «Таблицы» откроется вложенный запрос:



Он содержит два подзапроса, которые объединены:



Итого получилось:
Макет:



Вывод данных в табличный документ:

```
Макет = Отчеты.ЗакупкиПродажи_Объединение.ПолучитьМакет("Макет");
Запрос = Новый Запрос;
Запрос.Текст =
    "ВЫБРАТЬ
    |    ВложенныйЗапрос.Номенклатура,
    |    СУММА(ВложенныйЗапрос.Закупки) КАК Закупки,
    |    СУММА(ВложенныйЗапрос.Продажи) КАК Продажи
    |ИЗ
    |    (ВЫБРАТЬ
    |        ПриходнаяТовары.Номенклатура КАК Номенклатура,
    |        ПриходнаяТовары.Количество КАК Закупки,
    |        0 КАК Продажи
```

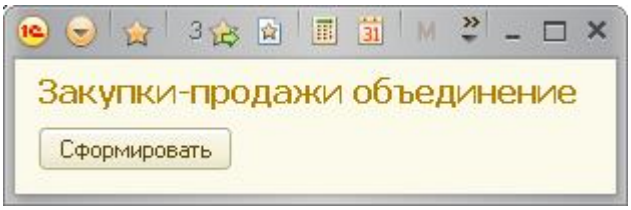
```
      |    ИЗ
      |         Документ.Приходная.Товары КАК ПриходнаяТовары
      |
      |    ОБЪЕДИНИТЬ ВСЕ
      |
      |    ВЫБРАТЬ
      |         РасходнаяТовары.Номенклатура,
      |         0,
      |         РасходнаяТовары.Количество
      |    ИЗ
      |         Документ.Расходная.Товары КАК РасходнаяТовары)
КАК ВложенныйЗапрос
      |
      |СГРУППИРОВАТЬ ПО
      |    ВложенныйЗапрос.Номенклатура";

Результат = Запрос.Выполнить();

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
ОбластьПодвал = Макет.ПолучитьОбласть("Подвал");
ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");
ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");
ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();
ТабДок.Вывести(ОбластьЗаголовок);
ТабДок.Вывести(ОбластьШапкаТаблицы);
ТабДок.НачатьАвтогруппировкуСтрок();

ВыборкаДетальныеЗаписи = Результат.Выбрать();

Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
     ОбластьДетальныхЗаписей.Параметры.Заполнить(ВыборкаДетальныеЗаписи);
     ТабДок.Вывести(ОбластьДетальныхЗаписей, ВыборкаДетальныеЗаписи.
Уровень());
КонецЦикла;

ТабДок.ЗакончитьАвтогруппировкуСтрок();
ТабДок.Вывести(ОбластьПодвалТаблицы);
ТабДок.Вывести(ОбластьПодвал);
```

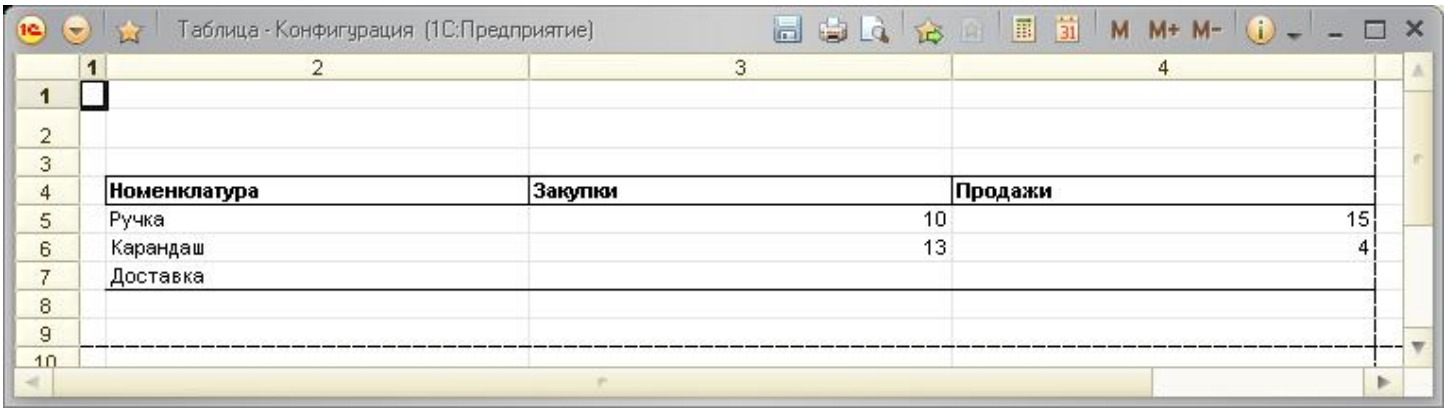
Процедура «Сформировать»:

```
&НаКлиенте
Процедура Сформировать(Команда)
     ТабДок = Новый ТабличныйДокумент;
     СформироватьОтчет(ТабДок);
     ТабДок.Показать();
КонецПроцедуры
```

Откроет в новом окне отчет.

В режиме исполнения:

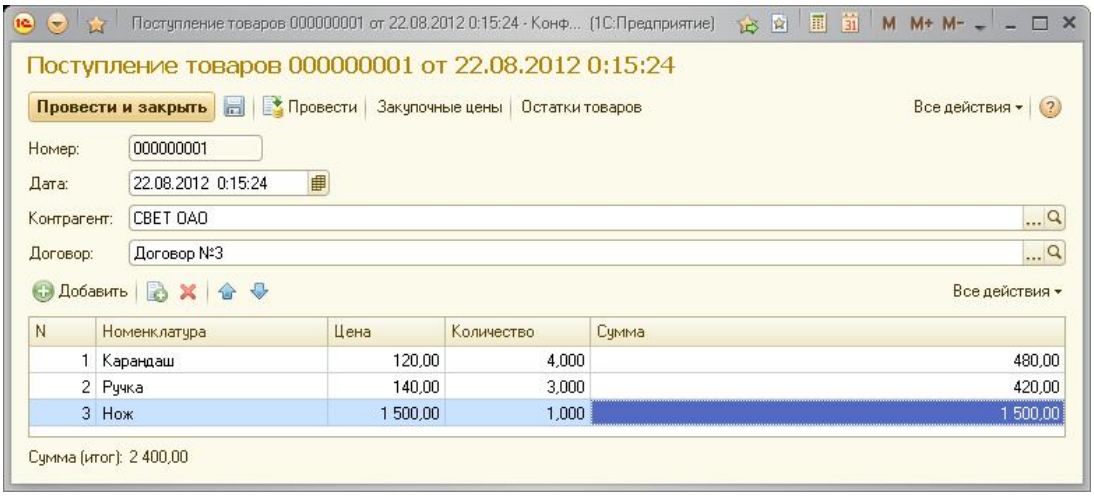


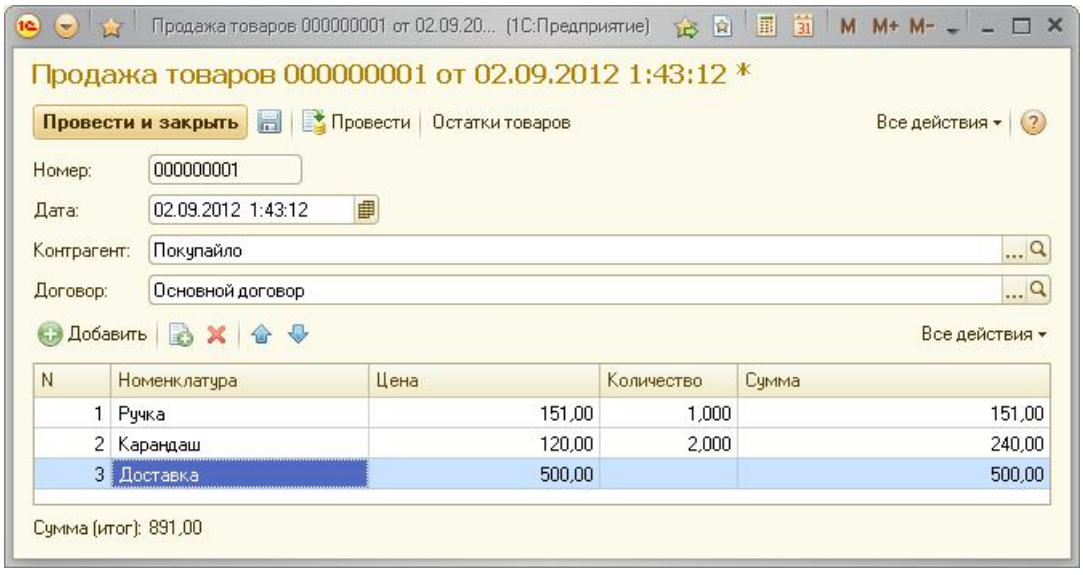


Номенклатура	Закупки	Продажи
Ручка	10	15
Карандаш	13	4
Доставка		

2. С помощью различных соединений

Подготовка – удалим все «Приходные» и «Расходные», кроме первых. Изменим еще и список номенклатуры в обоих видах документов.





Надо горизонтально присоединить к полям одной таблицы поля другой, найдя точку пересечения.

Сначала обратимся к таблице «Товары» документа «Приходная». К ней присоединим таблицу «Товары» документа «Расходная» с помощью предложения указания типа соединения. Присоединение производим по определенному условию.

```
ВЫБРАТЬ
      *
ИЗ
      Документ.Приходная.Товары КАК Приходная
ПОЛНОЕ СОЕДИНЕНИЕ //тип соединения
      Документ.Расходная.Товары КАК Расходная
ПО //условие соединения
      Приходная.Номенклатура = Расходная.Номенклатура
```

Ссылка	НомерСтрока	Номенклатура	Цена	Количество	Сумма	Ссылка1	НомерСтрока1	Номенклатура1	Цена1	Количество1	Сумма1
Поступление	3	Нож	1 500	1	1 500	NULL	NULL	NULL	NULL	NULL	NULL
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240
NULL	NULL	NULL	NULL	NULL	NULL	Продажа тов	3	Доставка	500	0	500

Виды соединений:

ПОЛНОЕ СОЕДИНЕНИЕ – в результат попадают все данные из обеих таблиц.

Ссылка	НомерСтрока	Номенклатура	Цена	Количество	Сумма	Ссылка1	НомерСтрока1	Номенклатура1	Цена1	Количество1	Сумма1
Поступление	3	Нож	1 500	1	1 500	NULL	NULL	NULL	NULL	NULL	NULL
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240
NULL	NULL	NULL	NULL	NULL	NULL	Продажа тов	3	Доставка	500	0	500

СОЕДИНЕНИЕ – внутреннее соединение. В результат попадет только то, что есть в обеих таблицах.

Ссылка	НомерСтрока	Номенклатура	Цена	Количество	Сумма	Ссылка1	НомерСтрока1	Номенклатура1	Цена1	Количество1	Сумма1
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240

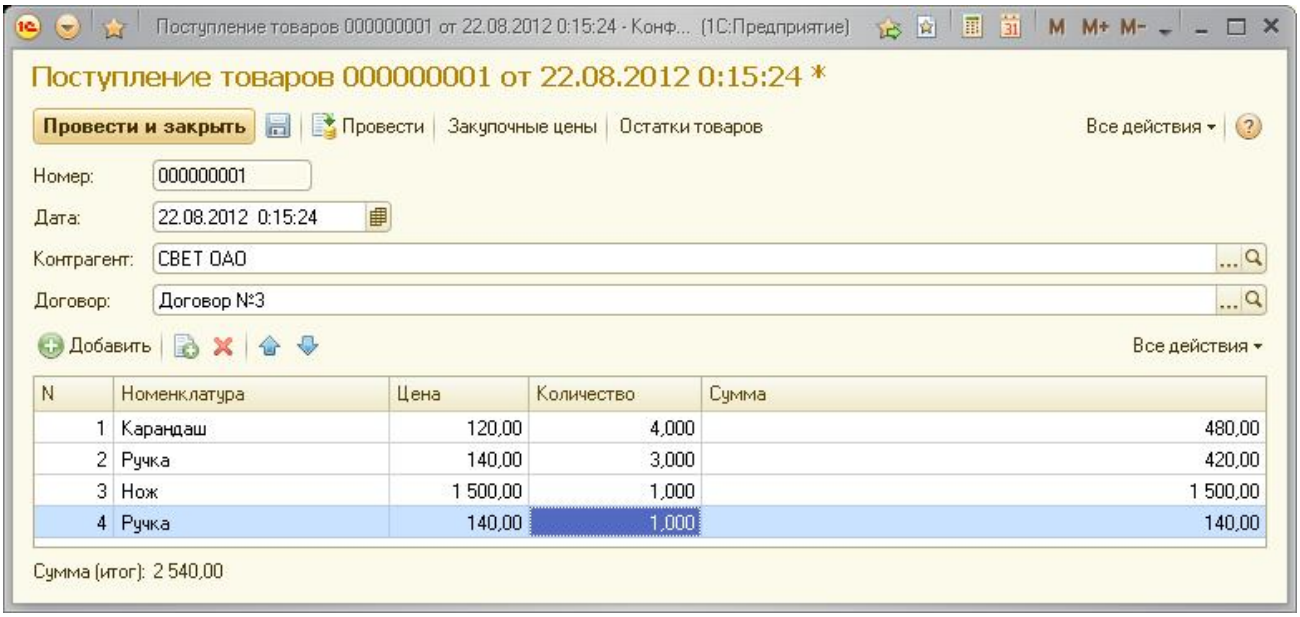
ЛЕВОЕ СОЕДИНЕНИЕ – в результат попадут все данные из левой таблицы (основной), а из правой только те поля, которые совпадают с полями левой (основной).

Ссылка	НомерСтрока	Номенклатура	Цена	Количество	Сумма	Ссылка1	НомерСтрока1	Номенклатура1	Цена1	Количество1	Сумма1
Поступление	3	Нож	1 500	1	1 500	NULL	NULL	NULL	NULL	NULL	NULL
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240

ПРАВОЕ СОЕДИНЕНИЕ - в результат попадут все данные из правой таблицы (основной), а из левой только те поля, которые совпадают с полями правой (основной).

Ссылка	НомерСтрок	Номенклатур	Цена	Количество	Сумма	Ссылка1	НомерСтрок	Номенклатур	Цена1	Количество1	Сумма1
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
NULL	NULL	NULL	NULL	NULL	NULL	Продажа тов	3	Доставка	500	0	500

Почему удалили «Приходные» и «Расходные»:
Добавим номенклатуру в «Приходной» и запишем документ:

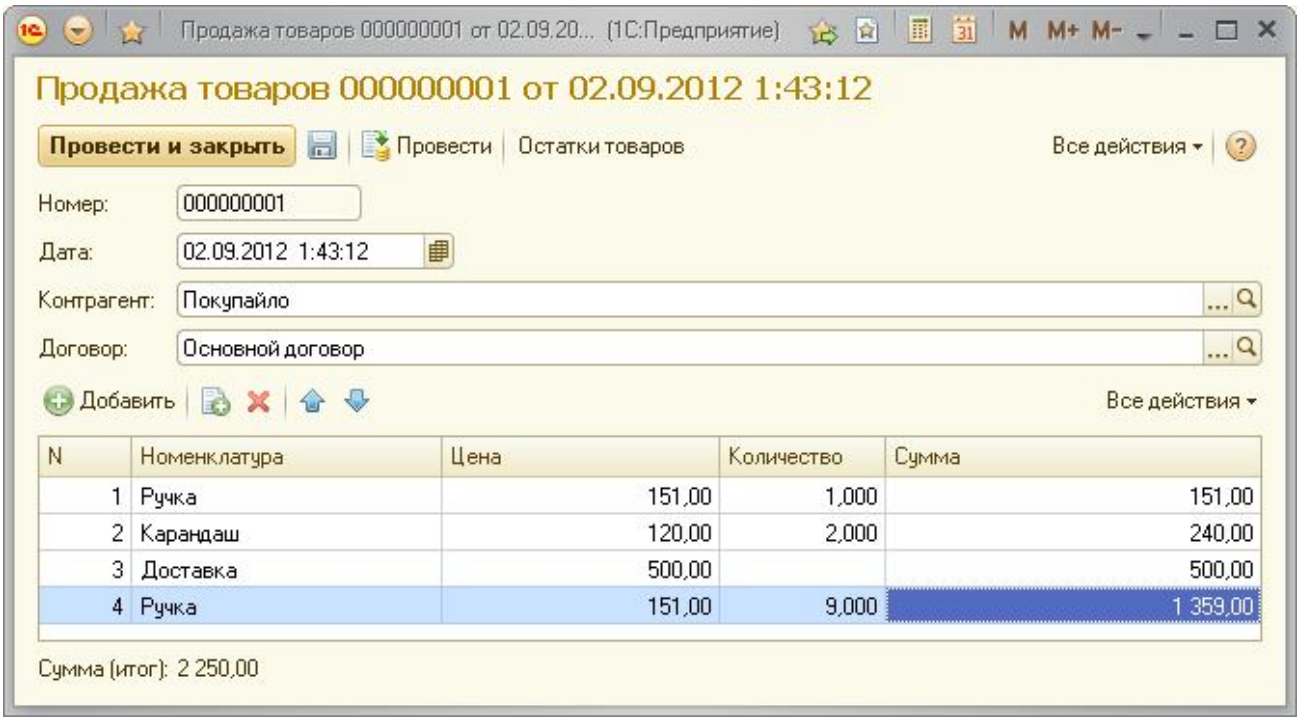


В запросе получится:

Ссылка	НомерСтрок	Номенклатур	Цена	Количество	Сумма	Ссылка1	НомерСтрок	Номенклатур	Цена1	Количество1	Сумма1
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240
Поступление	3	Нож	1 500	1	1 500	NULL	NULL	NULL	NULL	NULL	NULL
Поступление	4	Ручка	140	1	140	Продажа тов	1	Ручка	151	1	151
NULL	NULL	NULL	NULL	NULL	NULL	Продажа тов	3	Доставка	500	0	500

Ручка встретилась в документе «Приходная» два раза. По условию соединения ручка из документа «Расходная» была присоединена два раза, но это одна и та же ручка.

Добавим номенклатуру в «Расходной» и запишем документ:



В запросе получится:

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма	Ссылка1	НомерСтроки1	Номенклатура1	Цена1	Количество1	Сумма1
Поступление	2	Ручка	140	3	420	Продажа тов	1	Ручка	151	1	151
Поступление	2	Ручка	140	3	420	Продажа тов	4	Ручка	151	9	1 359
Поступление	1	Карандаш	120	4	480	Продажа тов	2	Карандаш	120	2	240
Поступление	3	Нож	1 500	1	1 500	NULL	NULL	NULL	NULL	NULL	NULL
Поступление	4	Ручка	140	1	140	Продажа тов	1	Ручка	151	1	151
Поступление	4	Ручка	140	1	140	Продажа тов	4	Ручка	151	9	1 359
NULL	NULL	NULL	NULL	NULL	NULL	Продажа тов	3	Доставка	500	0	500

Появилось еще больше дублей.

ВЫВОД: Перед соединением источники должны быть максимально сгруппированы.

Удалим дублирующиеся элементы номенклатуры из документов «Приходная» и «Расходная».

```
ВЫБРАТЬ
      Приходная.Номенклатура,
      Расходная.Номенклатура,
      Приходная.Количество КАК Купили,
      Расходная.Количество КАК Продали
ИЗ
      Документ.Приходная.Товары КАК Приходная
ПОЛНОЕ СОЕДИНЕНИЕ
      Документ.Расходная.Товары КАК Расходная
ПО
      Приходная.Номенклатура = Расходная.Номенклатура
```

Номенклатура	Номенклатура1	Купили	Продали
Карандаш	Карандаш	4	2
Ручка	Ручка	3	1
Нож	NULL	1	NULL
NULL	Доставка	NULL	0

Если в дальнейшем надо будет обрабатывать результат запроса (сравнивать поля, производить с ними математические действия), то значения «NULL» будут помехой.
В таких случаях значение «NULL» заменяется нулем с помощью функции «ЕСТЬNULL»

```
ВЫБРАТЬ
      Приходная.Номенклатура,
      Расходная.Номенклатура,
      ЕСТЬNULL(Приходная.Количество, 0) КАК Купили,
      ЕСТЬNULL(Расходная.Количество, 0) КАК Продали
ИЗ
      Документ.Приходная.Товары КАК Приходная
ПОЛНОЕ СОЕДИНЕНИЕ
      Документ.Расходная.Товары КАК Расходная
ПО
      Приходная.Номенклатура = Расходная.Номенклатура
```

Номенклатура	Номенклатура1	Купили	Продали
Карандаш	Карандаш	4	2
Ручка	Ручка	3	1
Нож	NULL	1	0
NULL	Доставка	0	0

Аналогично поступим и с номенклатурой:

```
ВЫБРАТЬ
      ЕСТЬNULL(Приходная.Номенклатура, Расходная.Номенклатура) КАК Номенклату-
ра,
      ЕСТЬNULL(Приходная.Количество, 0) КАК Купили,
      ЕСТЬNULL(Расходная.Количество, 0) КАК Продали
ИЗ
      Документ.Приходная.Товары КАК Приходная
ПОЛНОЕ СОЕДИНЕНИЕ
      Документ.Расходная.Товары КАК Расходная
ПО
      Приходная.Номенклатура = Расходная.Номенклатура
```

Номенклатура	Купили	Продали
Карандаш	4	2
Ручка	3	1
Нож	1	0
Доставка	0	0

По-хорошему данные в источниках должны быть предварительно сгруппированы.

```
ИЗ
      Документ.Приходная.Товары КАК Приходная //должен быть вложенный запрос
ПОЛНОЕ СОЕДИНЕНИЕ
      Документ.Расходная.Товары КАК Расходная //должен быть вложенный запрос
```

Скопируем отчет «Закупки-Продажи объединение», изменим реквизиты

Свойство	Значение
Имя	ЗакупкиПродажи_Соединение
Синоним	Закупки-продажи соединение

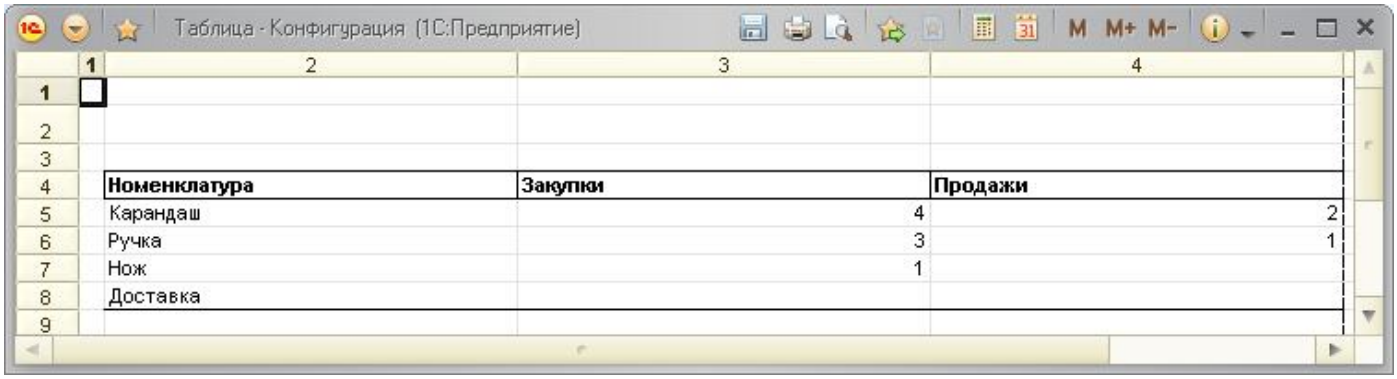
В коде заменим текст запрос на запрос из «Консоли отчетов»:

```
&НаКлиенте
Процедура Сформировать(Команда)
	ТабДок = Новый ТабличныйДокумент;
	СформироватьОтчет(ТабДок);
	ТабДок.Показать();
КонецПроцедуры

&НаСервереБезКонтекста
Процедура СформироватьОтчет(ТабДок)

	Макет = Отчеты.ЗакупкиПродажи_Объединение.ПолучитьМакет("Макет");
	Запрос = Новый Запрос;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	ЕСТЬNULL(Приходная.Номенклатура, Расходная.Номенклатура)
КАК Номенклатура,
	|	ЕСТЬNULL(Приходная.Количество, 0) КАК Закупки,
	|	ЕСТЬNULL(Расходная.Количество, 0) КАК Продажи
	|ИЗ
	|	Документ.Приходная.Товары КАК Приходная
	|		ПОЛНОЕ СОЕДИНЕНИЕ Документ.Расходная.Товары КАК Расходная
	|		ПО Приходная.Номенклатура = Расходная.Номенклатура";

	Результат = Запрос.Выполнить();

	ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
	ОбластьПодвал = Макет.ПолучитьОбласть("Подвал");
	ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");
	ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");
	ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

	ТабДок.Очистить();
	ТабДок.Вывести(ОбластьЗаголовок);
	ТабДок.Вывести(ОбластьШапкаТаблицы);
	ТабДок.НачатьАвтогруппировкуСтрок();

	ВыборкаДетальныеЗаписи = Результат.Выбрать();

	Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
		ОбластьДетальныхЗаписей.Параметры.Заполнить(ВыборкаДетальныеЗаписи);
		ТабДок.Вывести(ОбластьДетальныхЗаписей, ВыборкаДетальныеЗаписи.
```

```
Уровень());
      КонецЦикла;

      ТабДок.ЗакончитьАвтогруппировкуСтрок();
      ТабДок.Вывести(ОбластьПодвалТаблицы);
      ТабДок.Вывести(ОбластьПодвал);
```

КонецПроцедуры

В режиме исполнения:



Для построения отчетов рекомендуют использовать механизм – «Система компоновки данных (СКД)».

Создадим отчет «Ведомость по товарам»:
Пользователь будет выбирать период.

Ведомость по товарам				
Номенклатура	**Начальный остаток**	**Приход**	**Расход**	**Конечный остаток**

Отчет будем делать на основе регистра накопления «ОстаткиТоваров».

Свойство	Значение
Имя	ВедомостьПоТоварам
Синоним	Ведомость по товарам

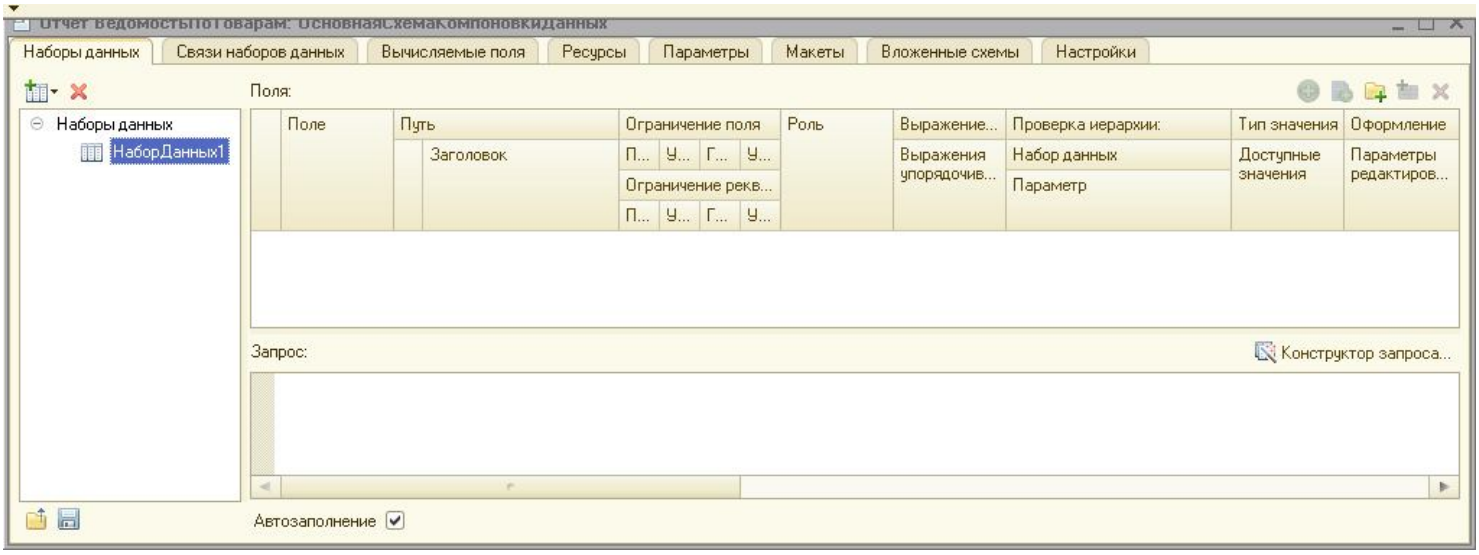
Схема компоновки данных – часть СКД. Описывает источник данных и его представление.

Созданная схема будет храниться как макет в виде XML-документа.

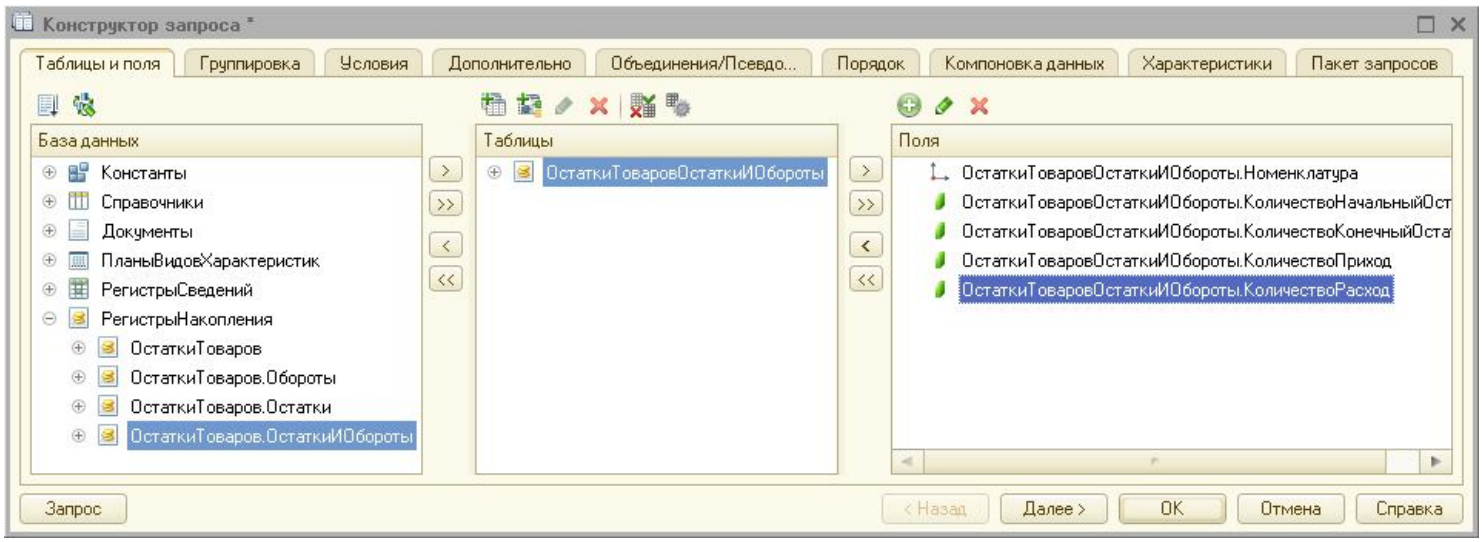
Закладка «Наборы данных» - указываем, с какими источниками данных будем работать:

- Запрос.
- Объект (например, табличная часть документа).
- Объединение (объединение запросов).

Добавим запрос. На его основе система автоматом сформирует список полей Схемы КД – то, что будет отображаться в отчете.

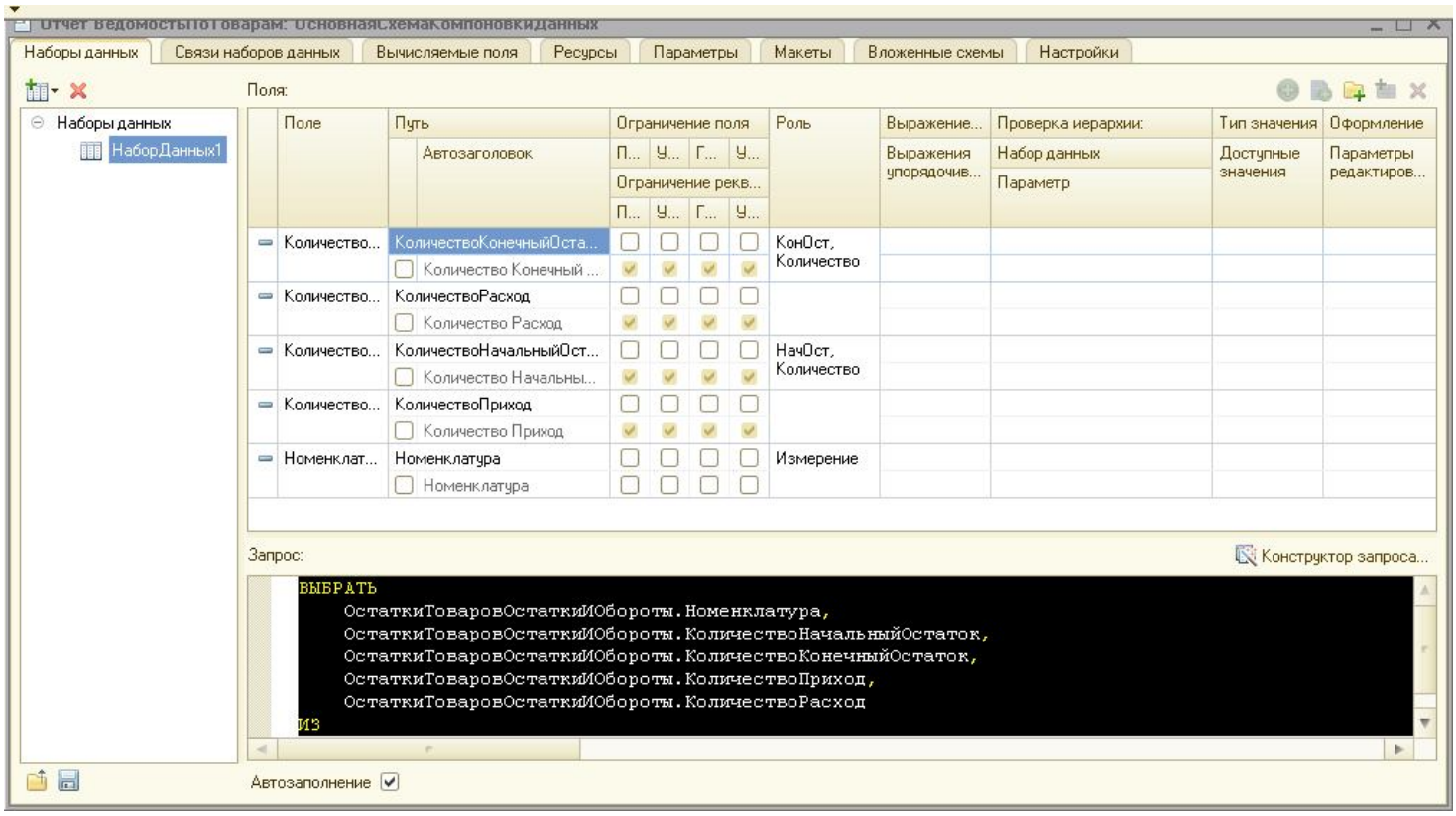


Воспользуемся «Конструктором запроса»

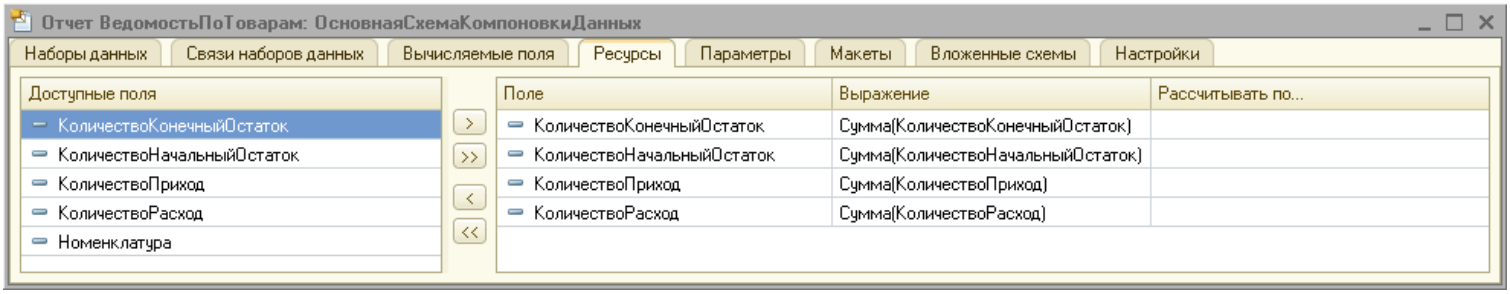


Представления получать не надо, СКД их получит самостоятельно.

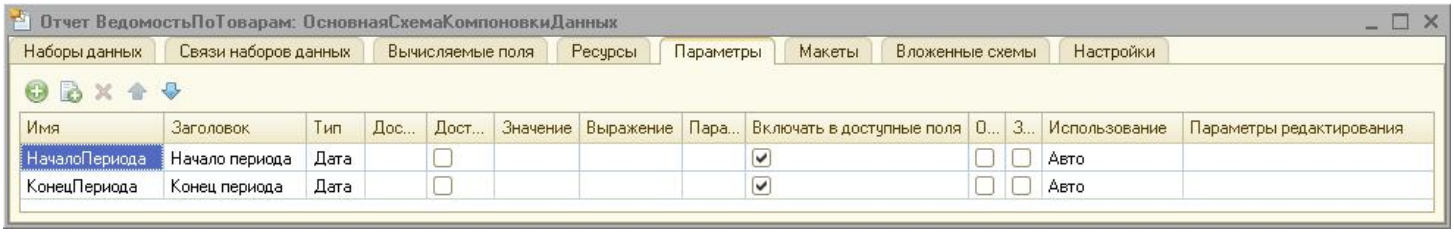
По тексту запроса система создала Представления и Роли полей:
Роли анализируются на основе метаданных, т.е. того, к чему обращаемся с помощью СКД.



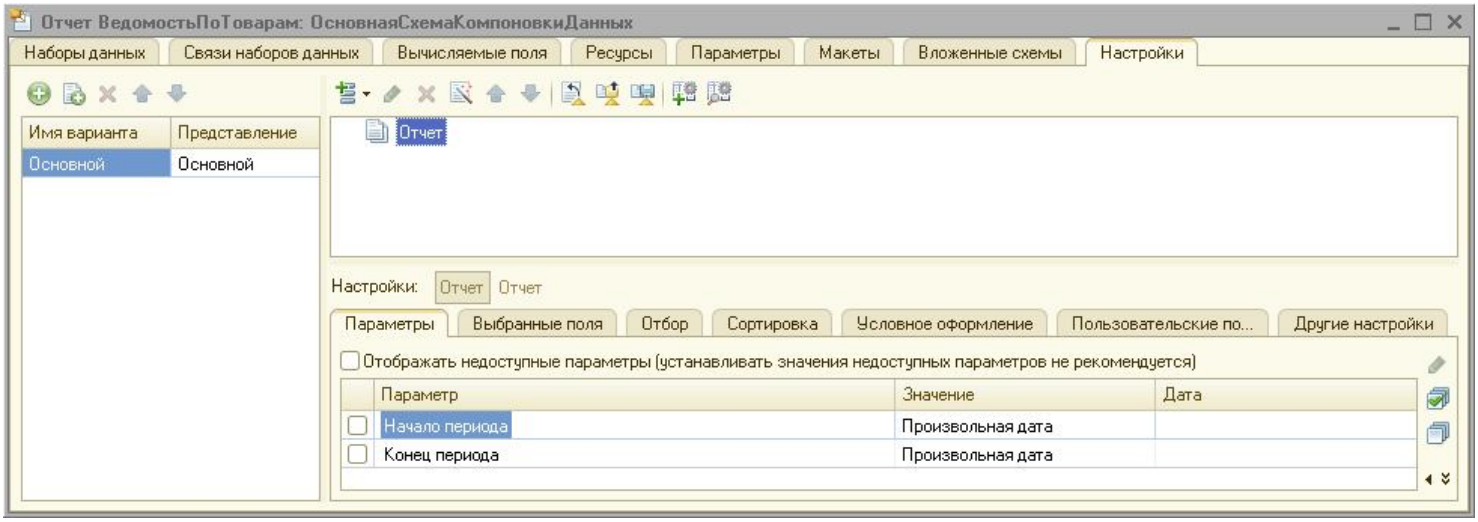
Все поля делятся на:
«Измерения»
«Ресурсы» - то, что будет складываться по группировкам.



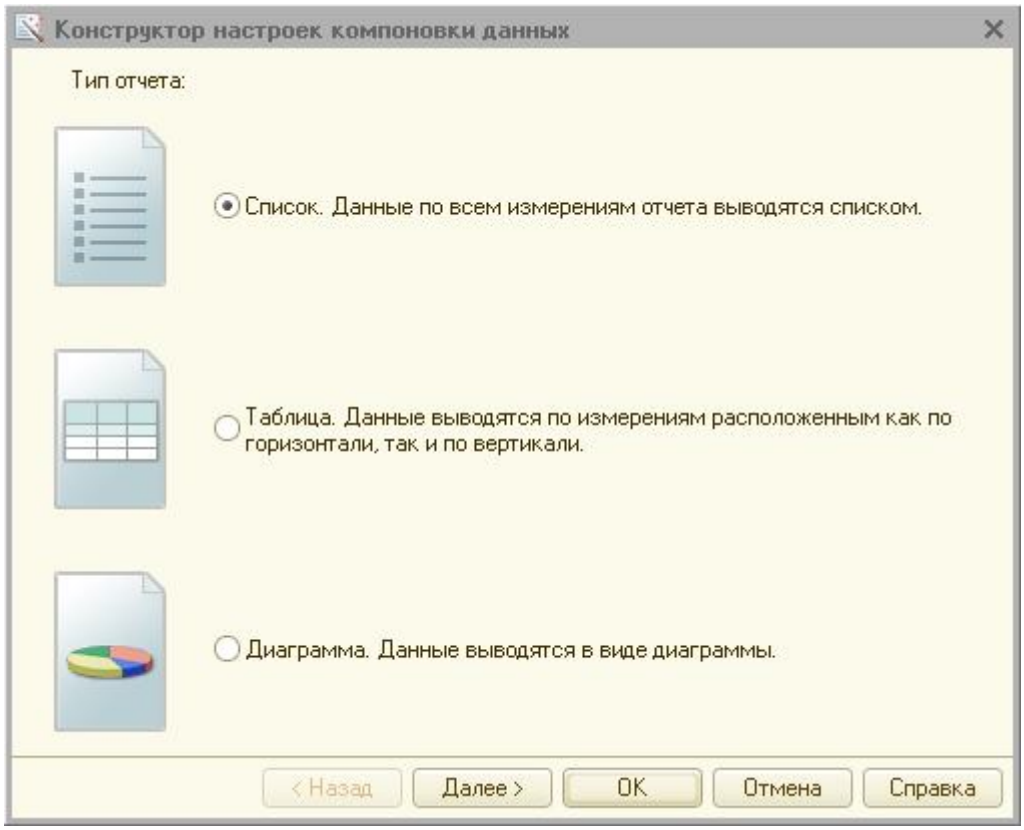
СКД проанализировала, что обращение идет к виртуальной таблице и поняла, что можно задать ограничение на Период отчета – эта настройка будет доступна пользователю:



Закладка «Настройки» - настройка формирования отчета у пользователя по умолчанию.



Воспользуемся «Конструктором настроек»:

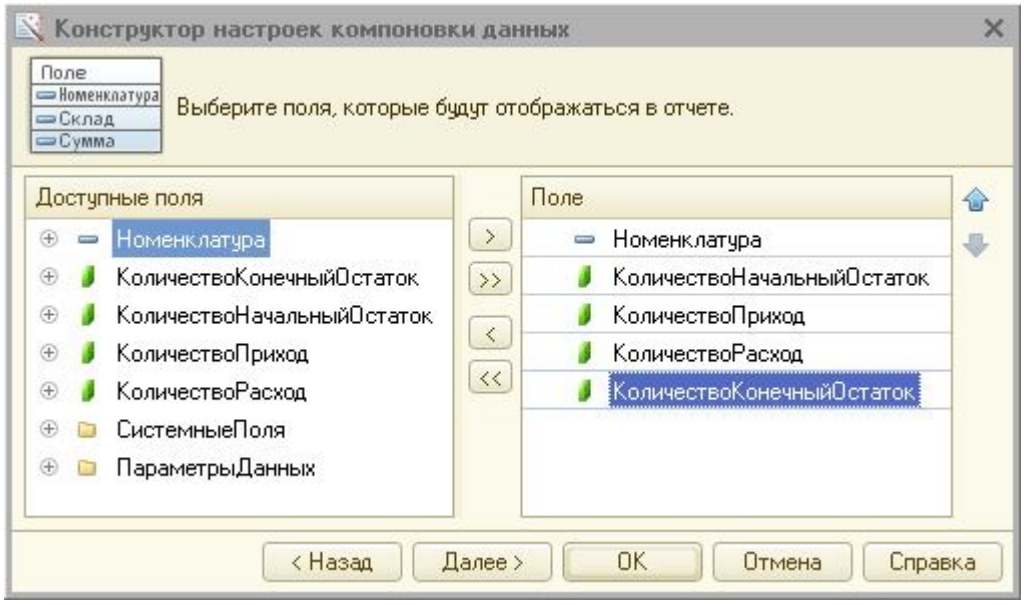


Данные можно выводить в:

- Список – например, когда надо просто вывести список товаров (с любым количеством показателей).
- Таблицу – когда надо выводить что-либо и в строках и в колонках. Например, товары по складам (склады – в колонках, товары – в строках).
- Диаграмму.

В примере данные будем выводить в список.

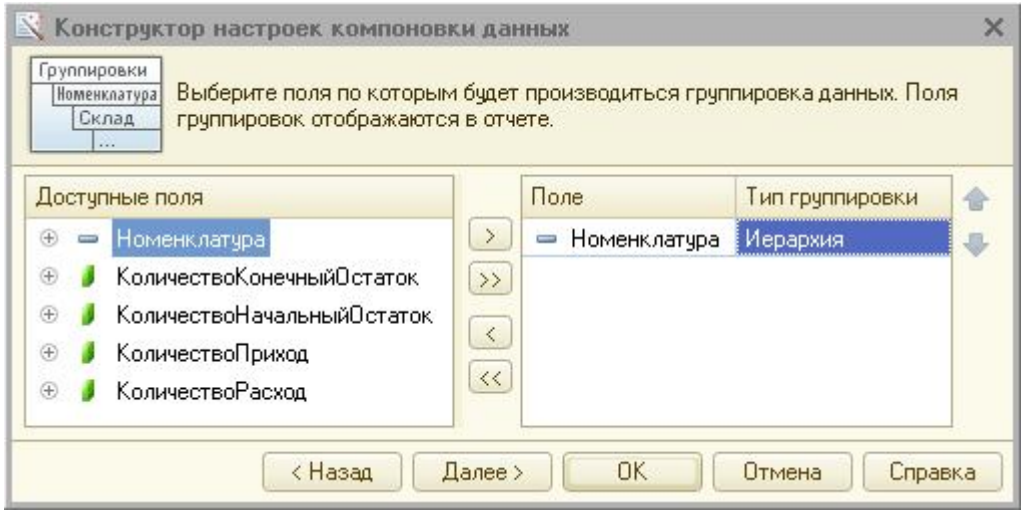
Выбор выводимых полей и порядка их вывода:



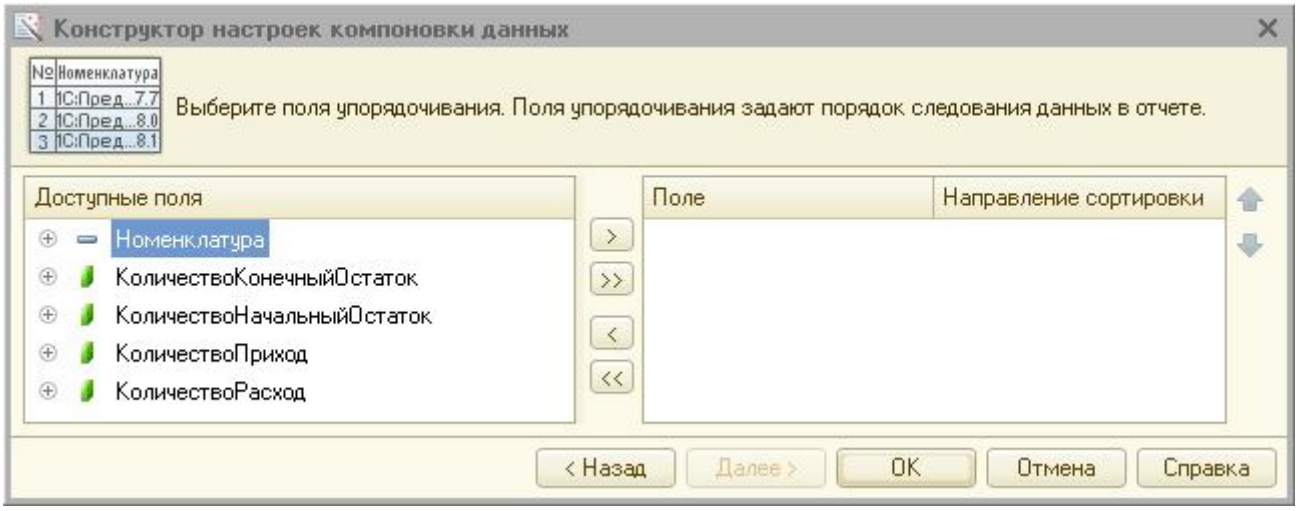
Количество будет выводиться в разрезе товаров – это определяется группировкой. Если:

Свойство	Значение
Тип группировки	Иерархия

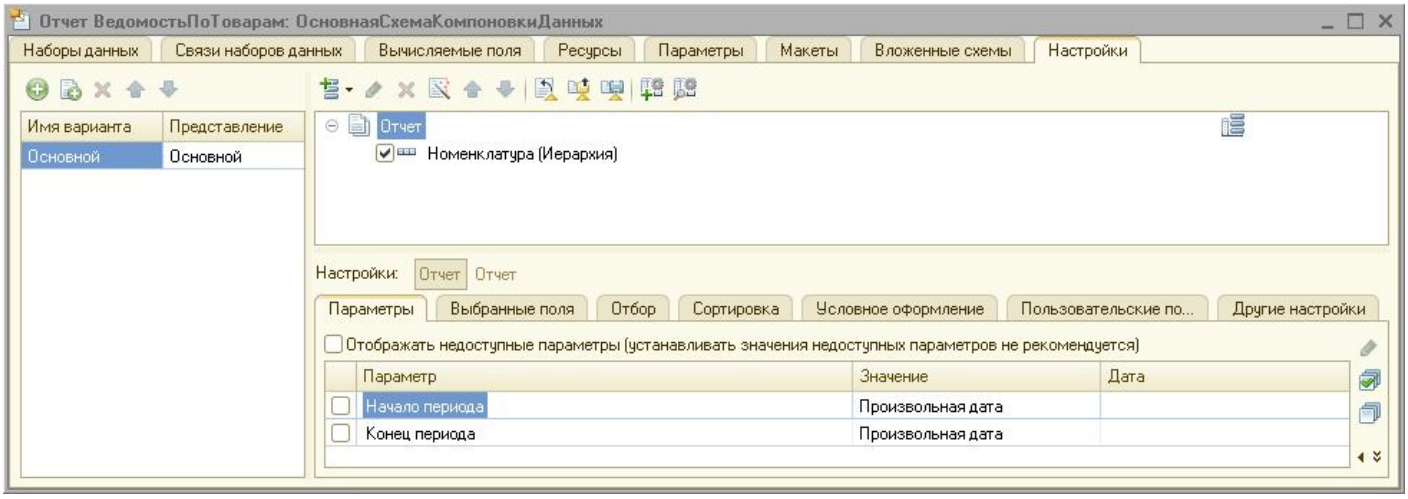
Тогда выведутся еще и группы товаров.

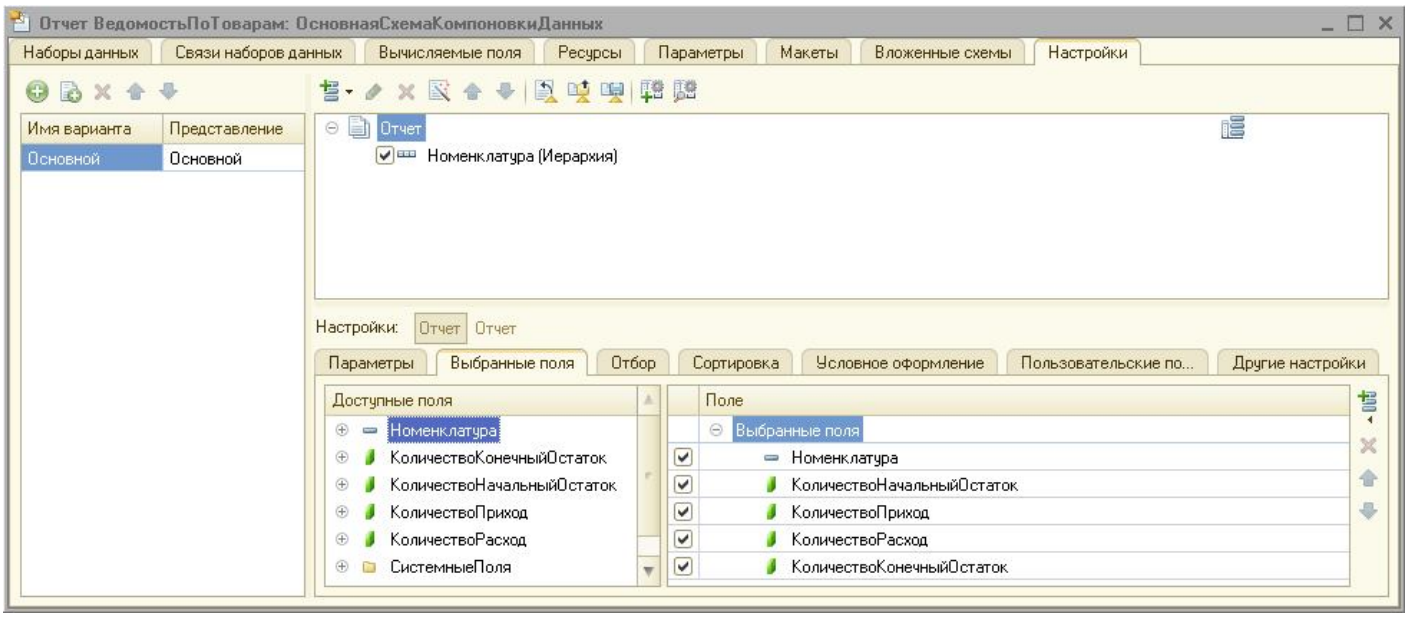


На следующем этапе настраивается сортировка. В примере не надо.



В итоге получилось:

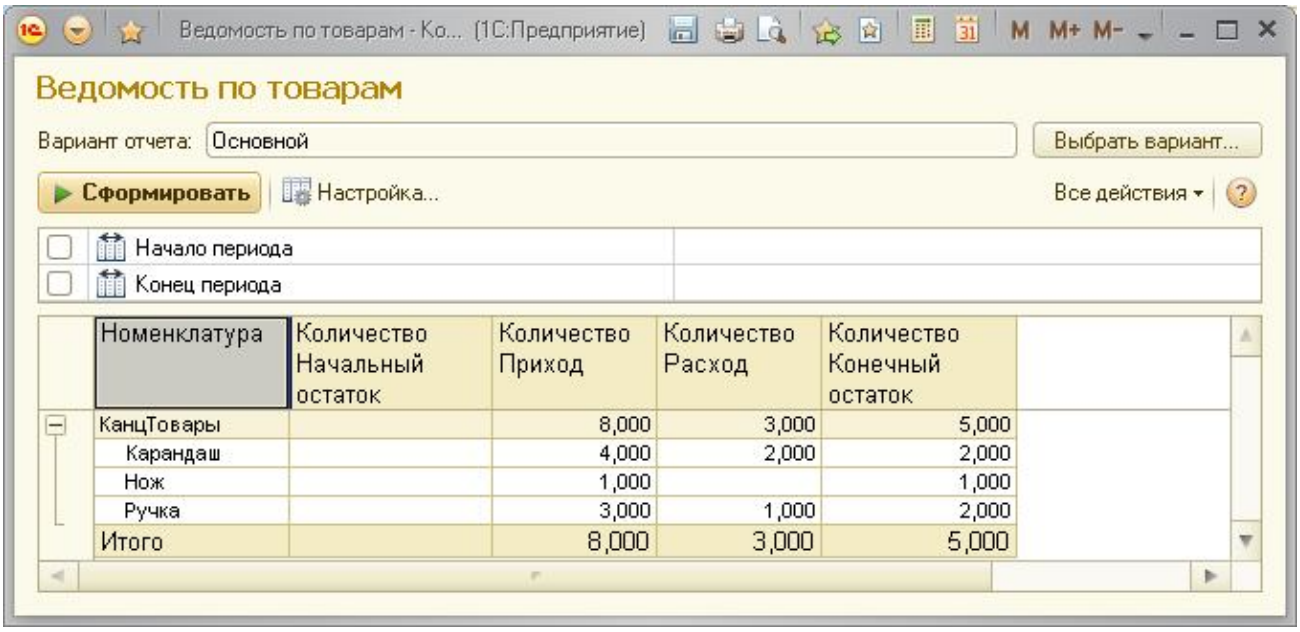




Сделаем доступными пользователю настройки параметров начала и конца периода: «Параметр – ПКМ – Свойства элемента пользовательских настроек – Включать в пользовательские настройки».

Если в запросе дополнительно указать какие-то свои параметры, настройки, то они подхватятся СКД. Сама СКД умеет выбирать только основные параметры.

В режиме исполнения:



В СКД в свойстве поля «Оформление» можно настроить формат выводимых данных. Аналогично и в «Вычисляемых полях».

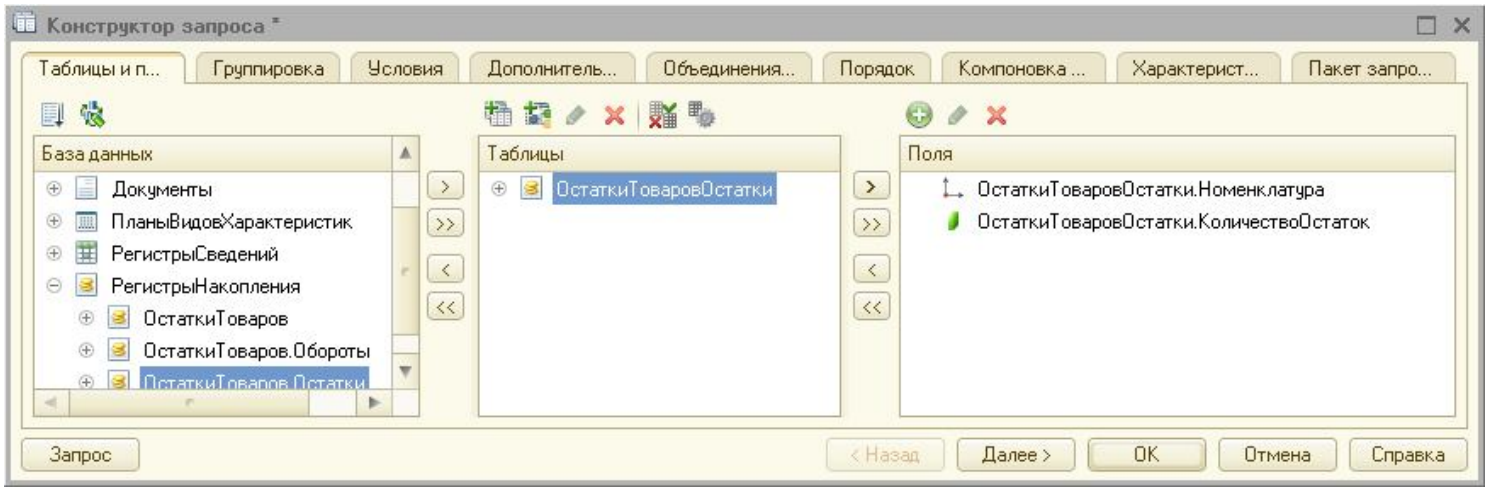
Занятие 1.11

Запрос в отчете «Остатки товаров» не совсем оптимальный, т.к. номенклатура берется по ссылке, а чтобы показать пользователю представление, система делает еще один скрытый запрос к БД.

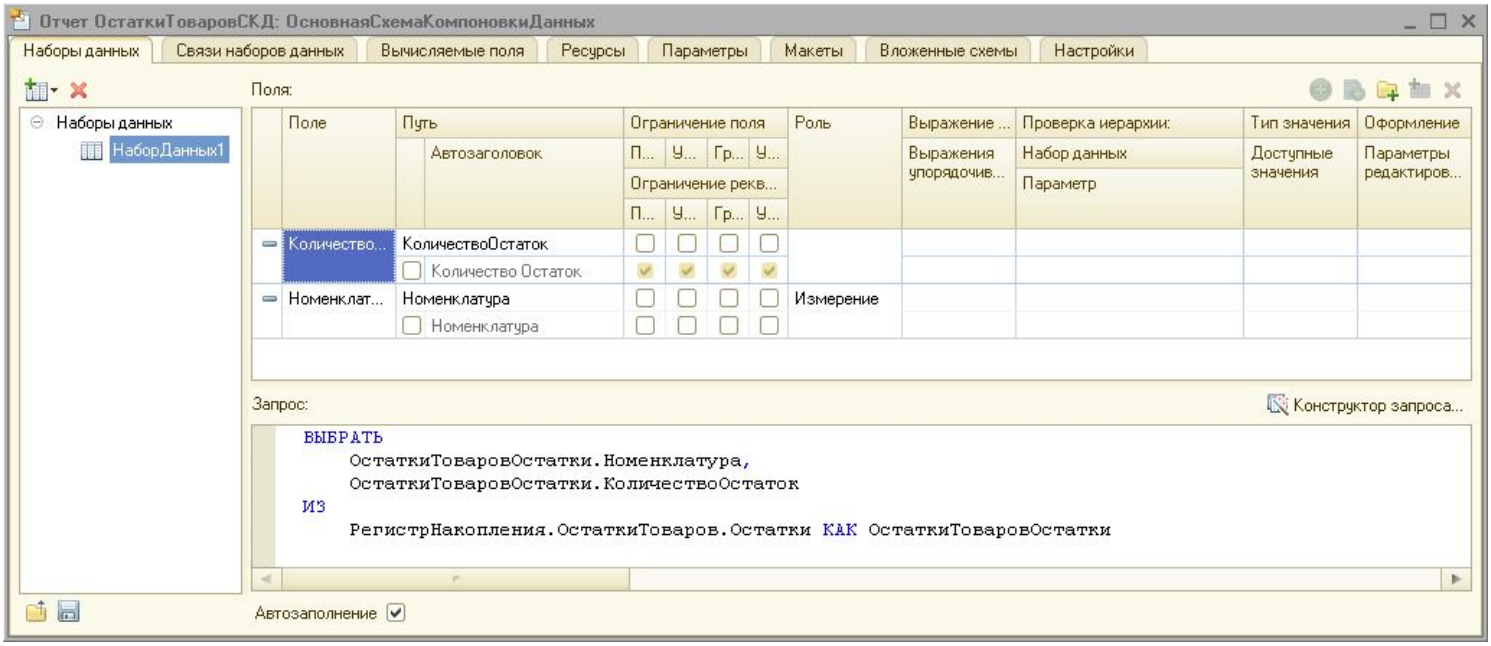
Сделаем этот отчет с помощью СКД:

Свойство	Значение
Имя	ОстаткиТоваровСКД
Синоним	Остатки товаров СКД

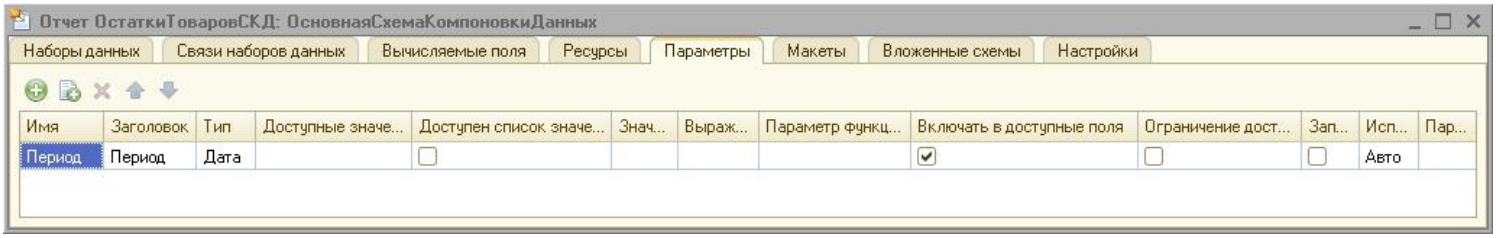
Откроем конструктор СКД:
Добавим «Набор данных–запрос» и откроем «Конструктор запроса»



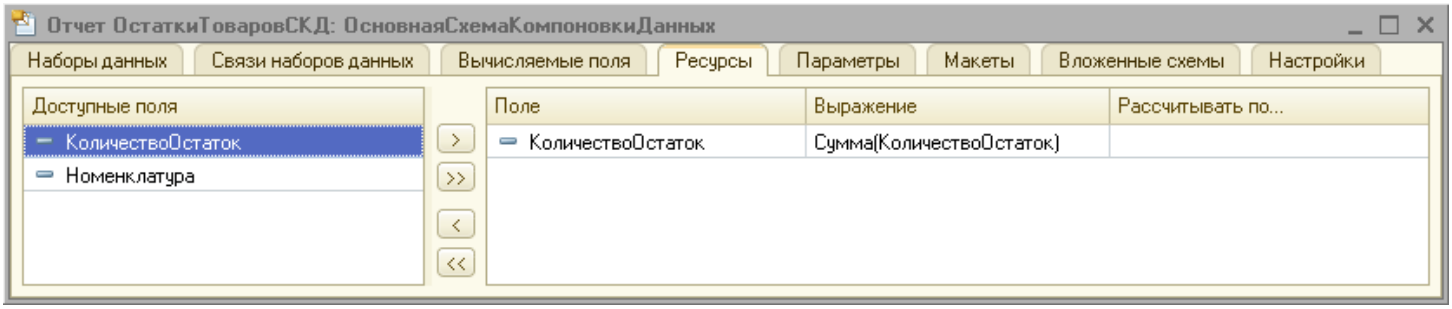
Параметры виртуальной таблицы пока задавать не будем. Основной параметр «Период» СКД задаст сама.



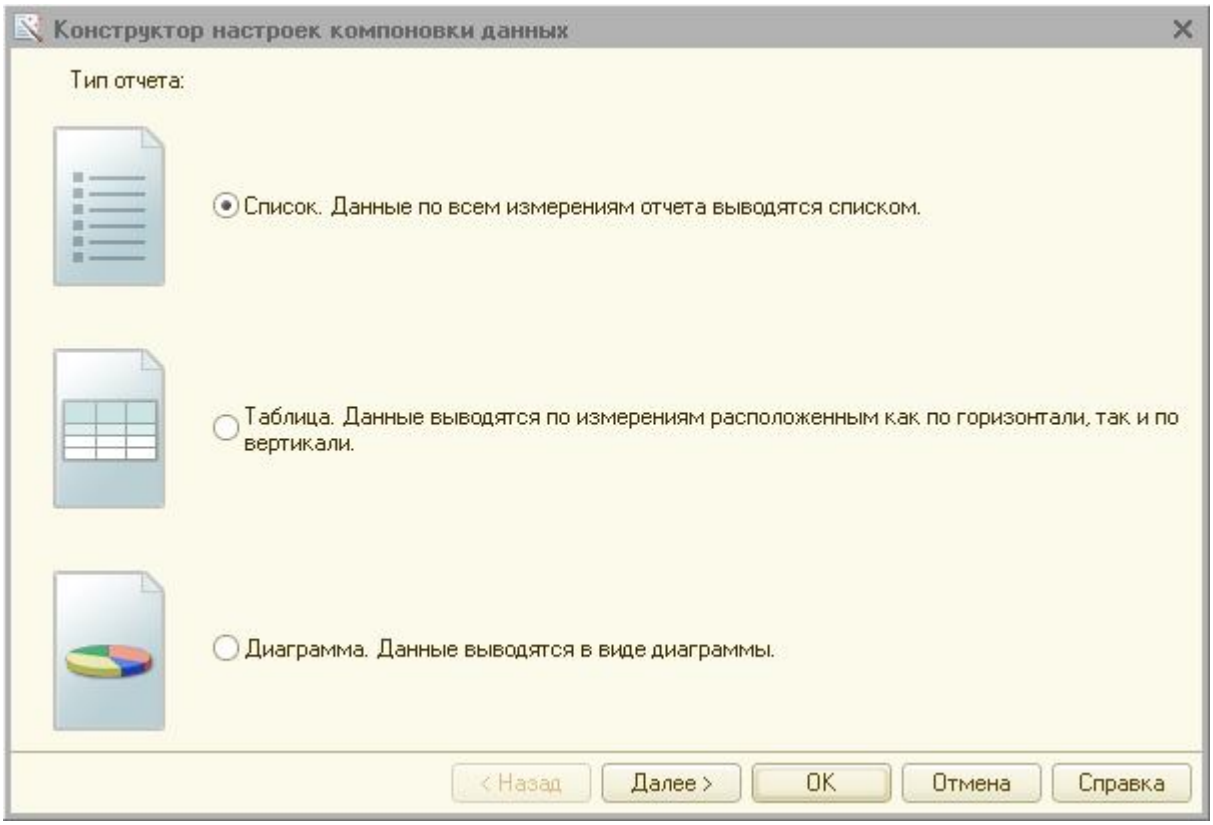
Если «Ограничение доступности = Ложь», то пользователю будет доступен выбор периода.

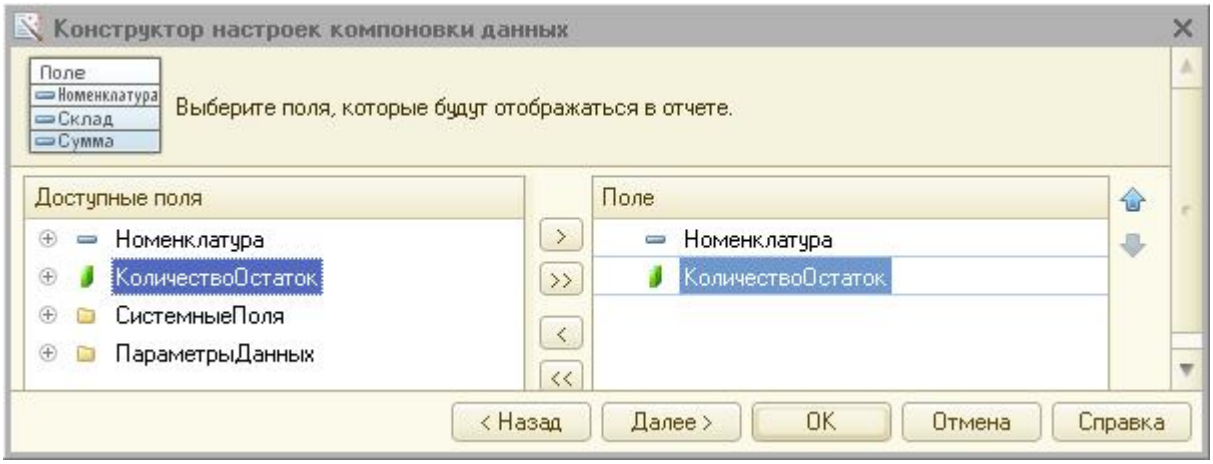


Зададим «Ресурсы» отчета. Ресурс – это всегда то, что расположено на пересечении строки и столбца:

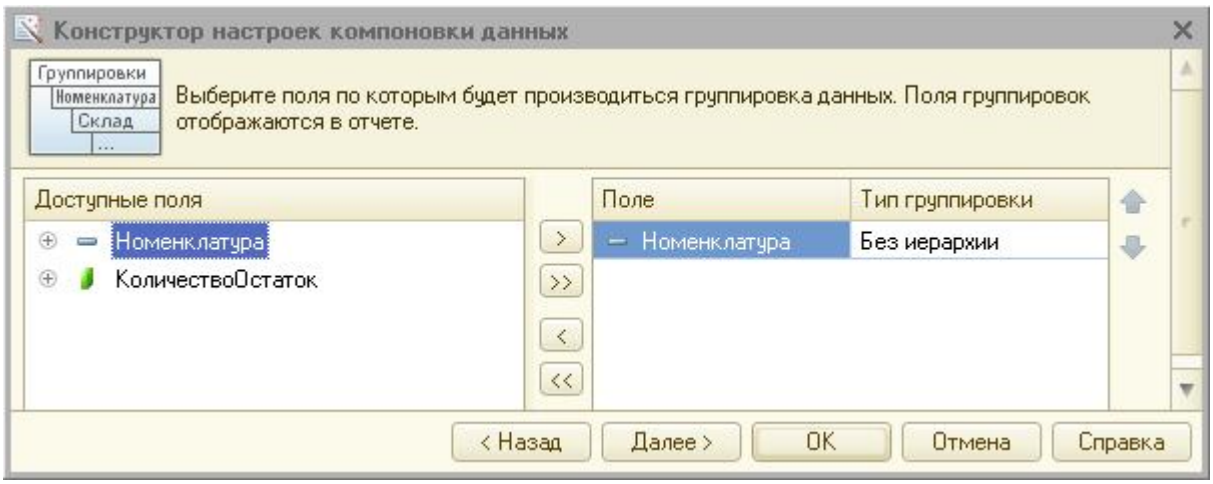


Настройки по умолчанию – закладка «Настройки». Зададим с помощью мастера:



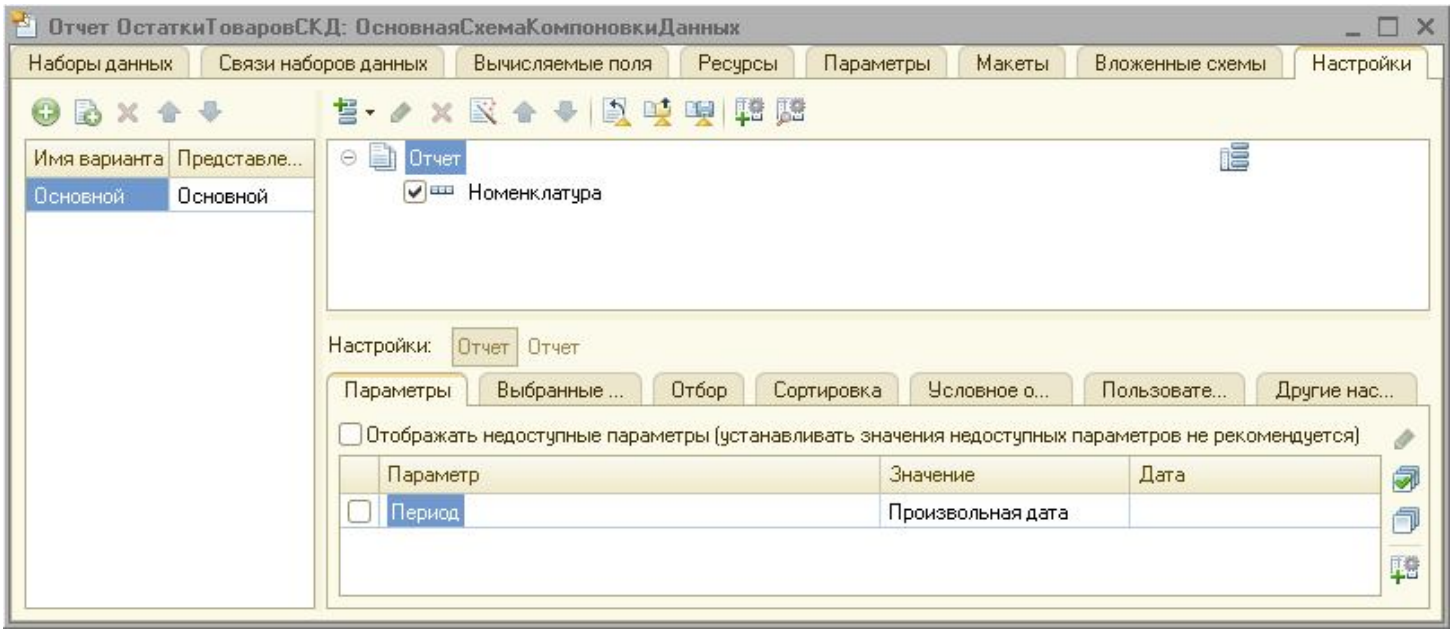


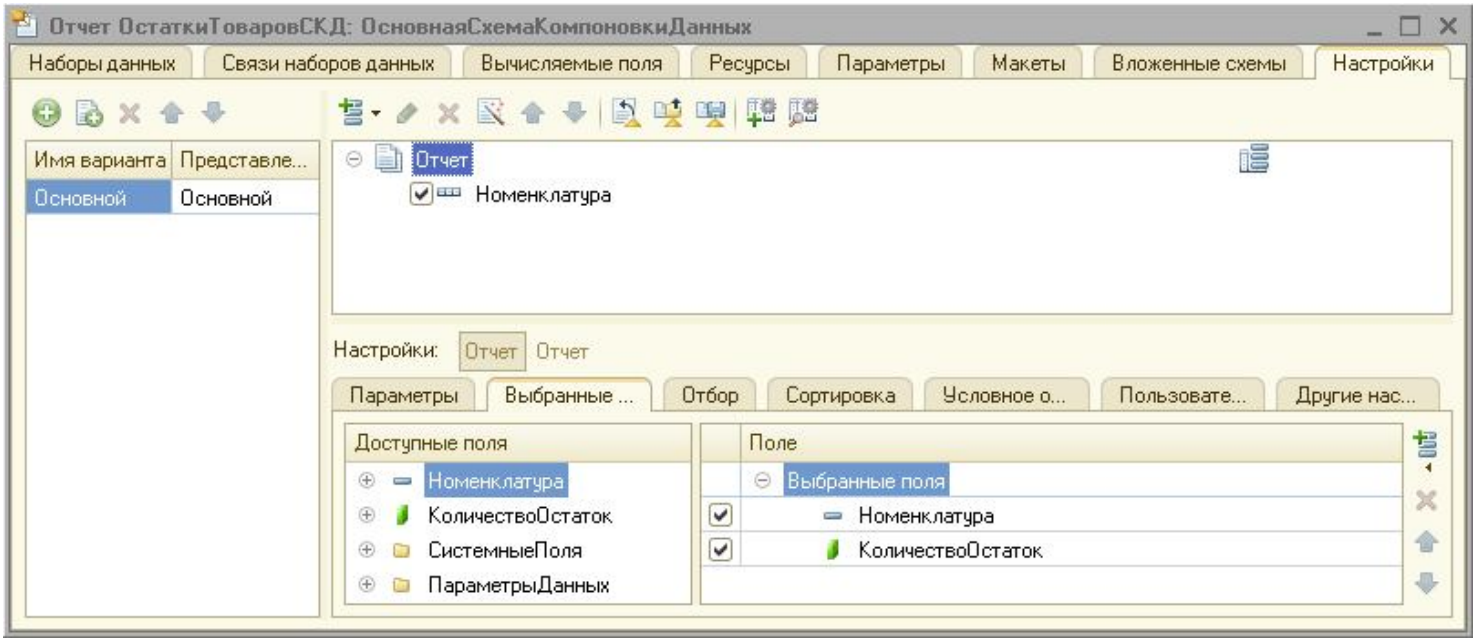
Группировка по номенклатуре:



Сортировка не нужна.

Итого получилось:

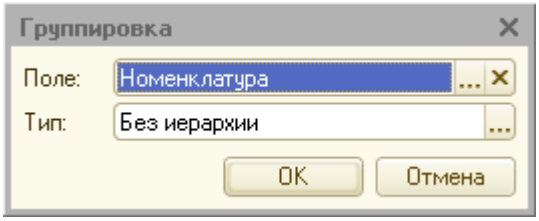




Слева есть таблица вариантов оформления отчета. Сейчас мы оформили «Основной». Добавим еще один вариант::

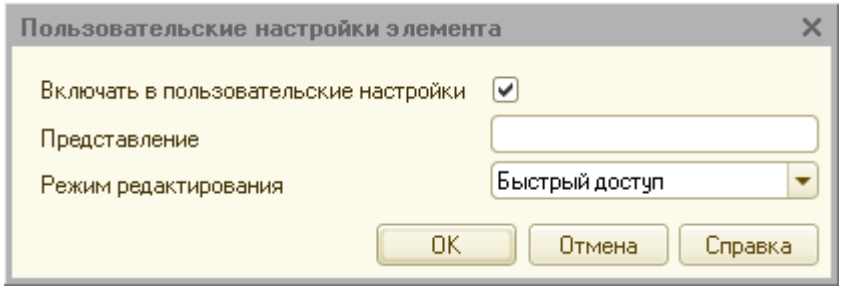
Свойство	Значение
Имя варианта настроек	Диаграмма
Представление варианта настроек	Диаграмма

Выведем «Количество» в виде Диаграммы – «Отчет – ПКМ – Новая диаграмма». Диаграмма состоит из «Точек» и «Серий». Добавим точку – «Точки – ПКМ – Новая группировка»



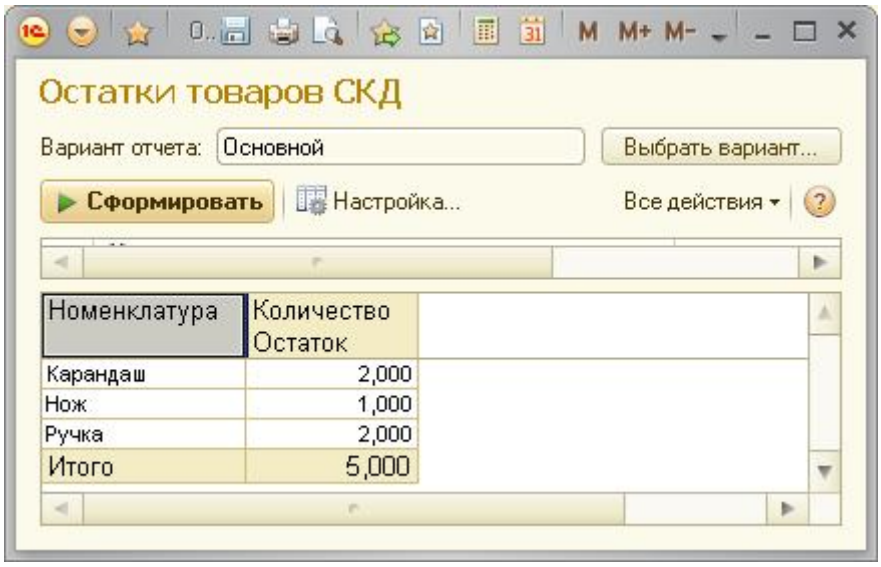
Если бы были «Склады», то можно было бы добавить «Серии».

Параметр «Период» в варианте оформления «Основной» редактируется в форме отчета: «Параметр – ПКМ – Свойства элемента пользовательских настроек»

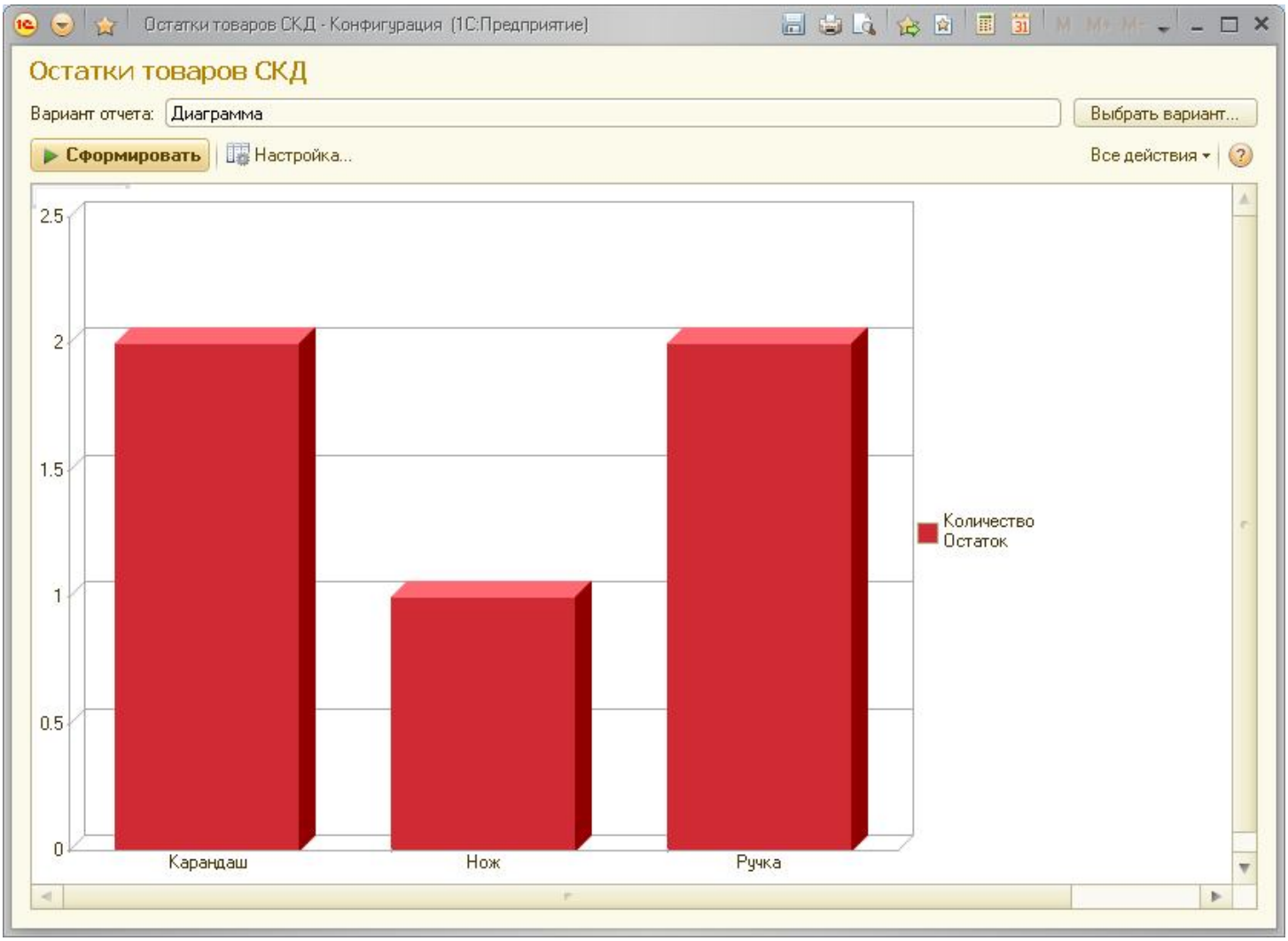


В режиме исполнения:

Вариант оформления «Основной»:

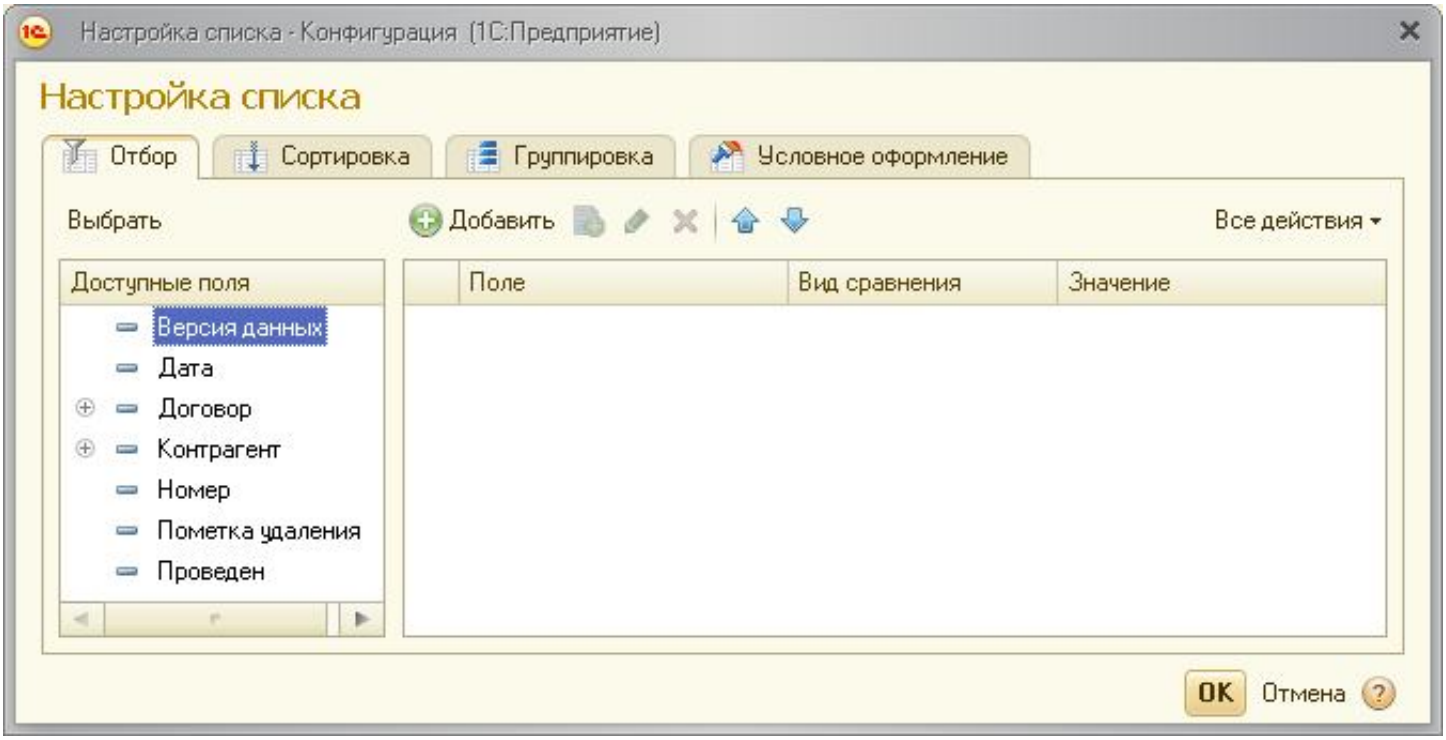


Вариант оформления «Диаграмма»:



Варианты оформления можно создавать/изменять и в режиме исполнения. Но созданные в режиме исполнения варианты оформления нельзя исправить в режиме конфигуратора.

СКД позволяет также выводить различные списочные представления, т.е. любая форма списка – результат работы СКД. Проверить это можно с помощью «Все действия – Настроить список»:



Настройка имеет такие же закладки, как на закладке «Настройки» СКД.

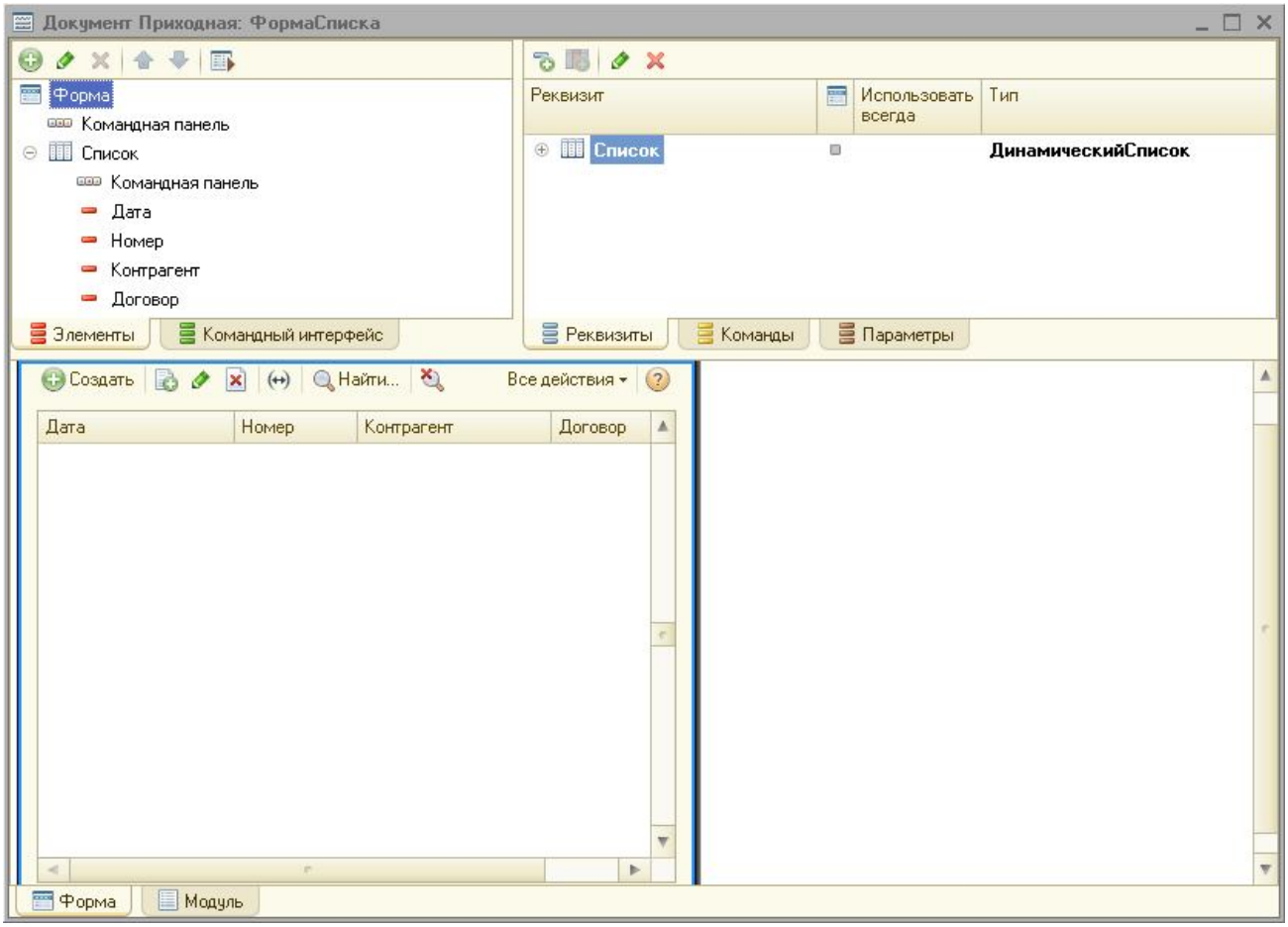
Т.е. можно переопределить вывод данных, добавив в него дополнительные возможности.

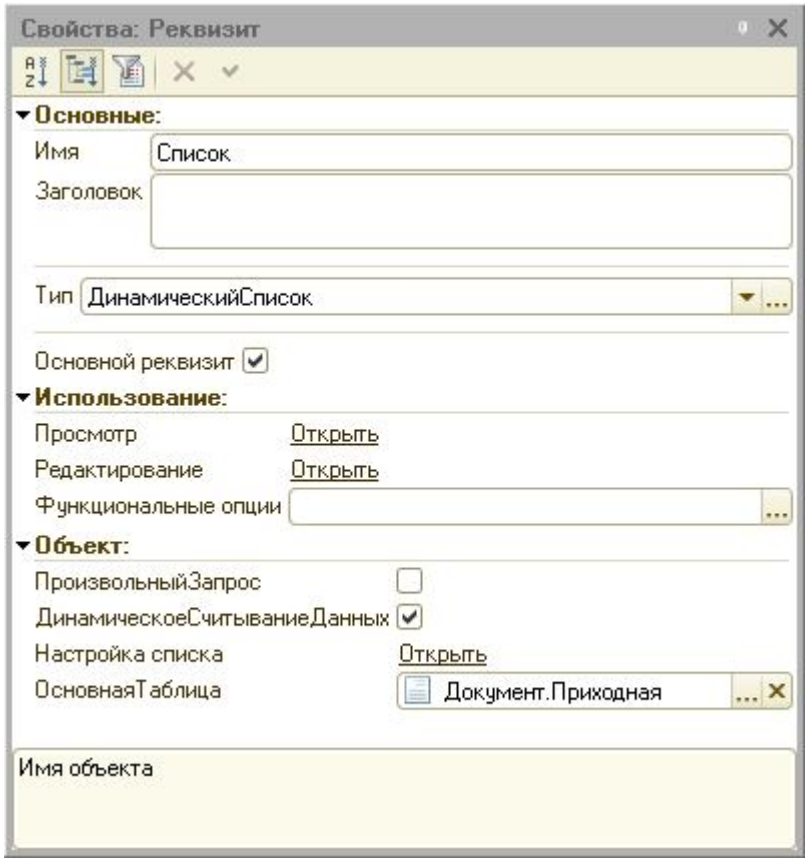
Добавим в форму списка документов «Приходная» и «Расходная» поле, в котором будет отображаться итоговая сумма документа.

Нарисуем форму списка документа «Приходная»:
В мастере для отображения добавим реквизиты «Контрагент» и «Договор»

Список выводится с помощью объекта «ДинамическийСписок»
Динамический список – это и есть обрезанная СКД, которую можно переопределить:

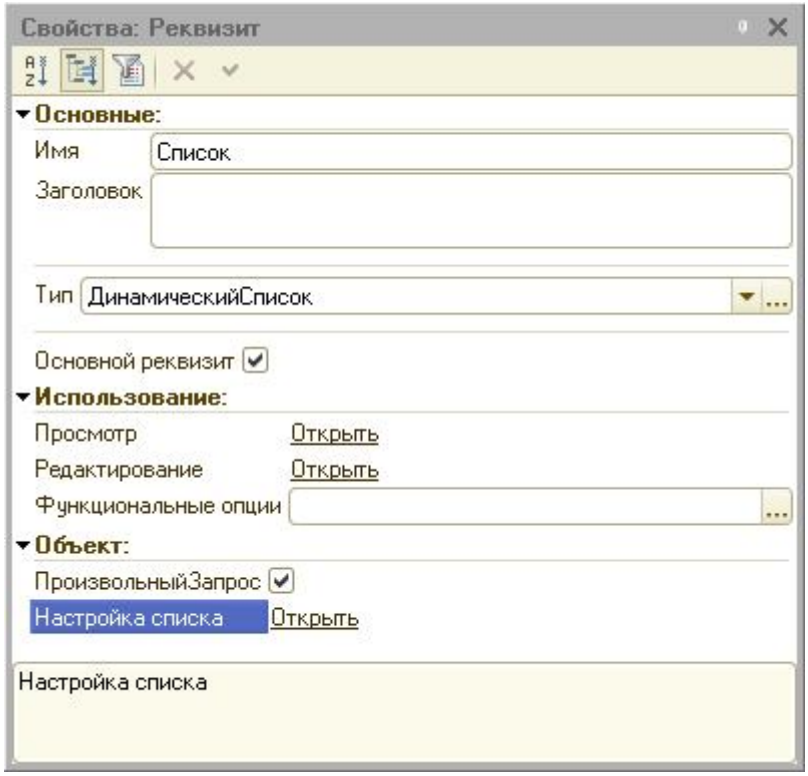
«Список – ПКМ - Свойства». Можно реализовать настройку списка.



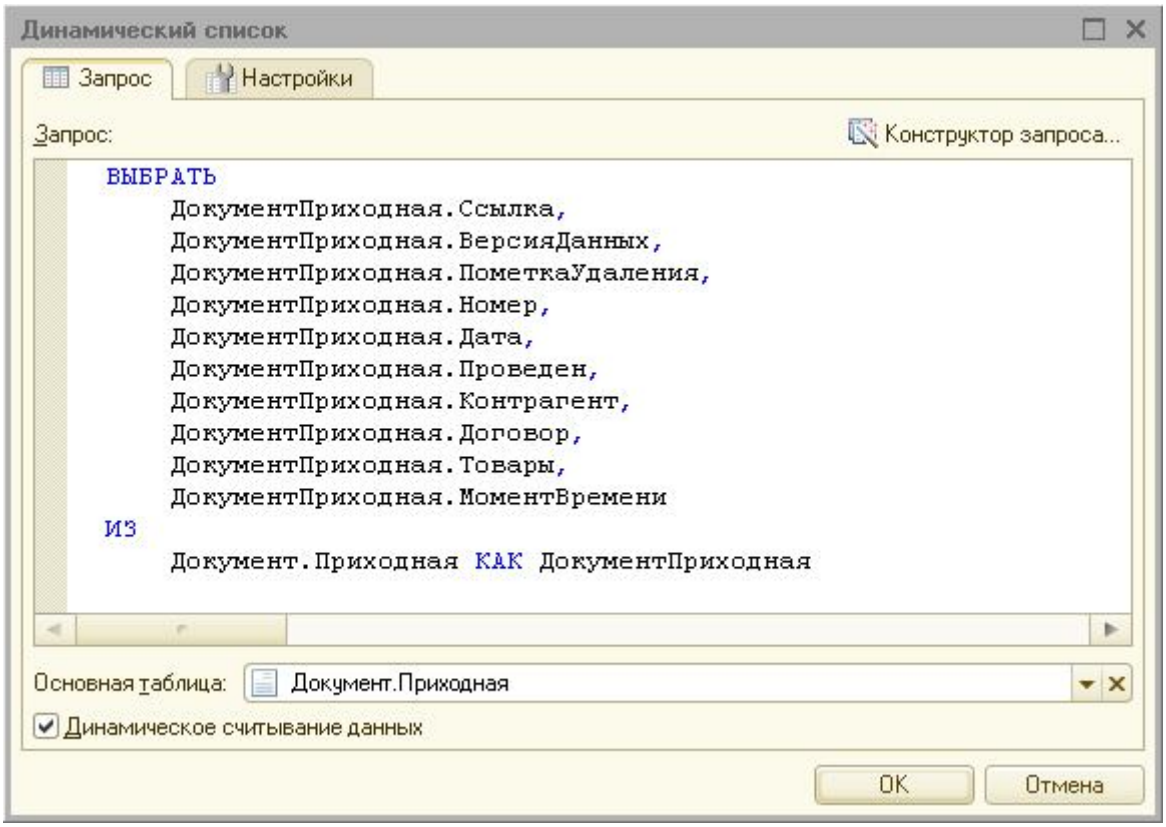


Свойство «Основная таблица – Документ.Приходная» обозначает, что система сама написала запрос к таблице документа «Приходная».

Чтобы переопределить и написать свой запрос надо:
«Произвольный запрос = Истина»

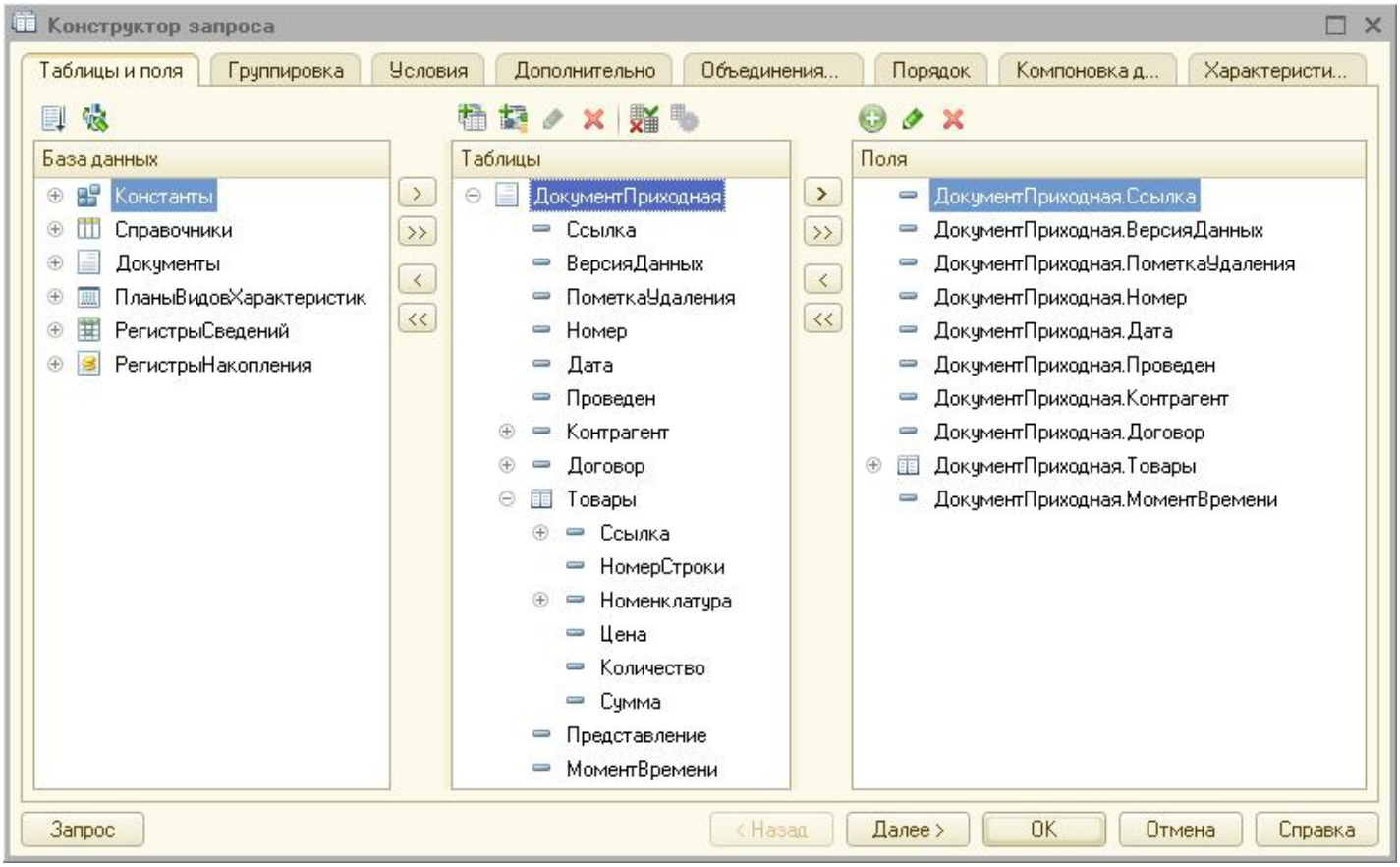


И открыть «Настройку списка».

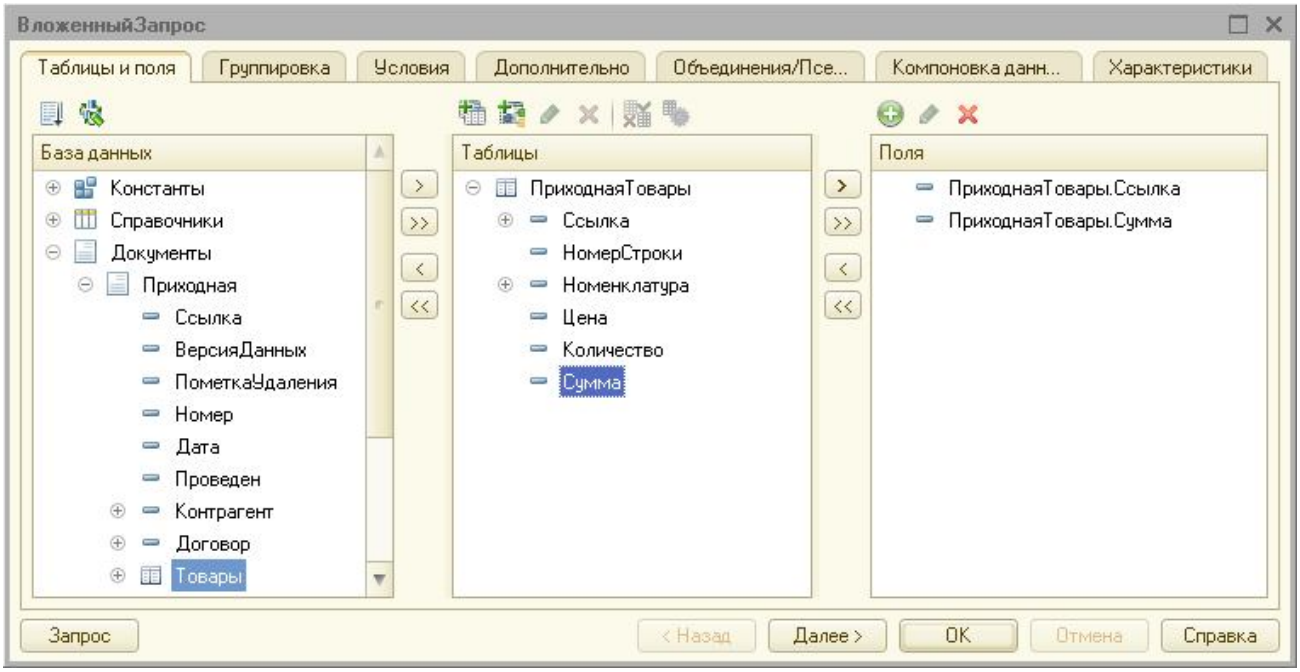


Этот запрос сформировала сама система.

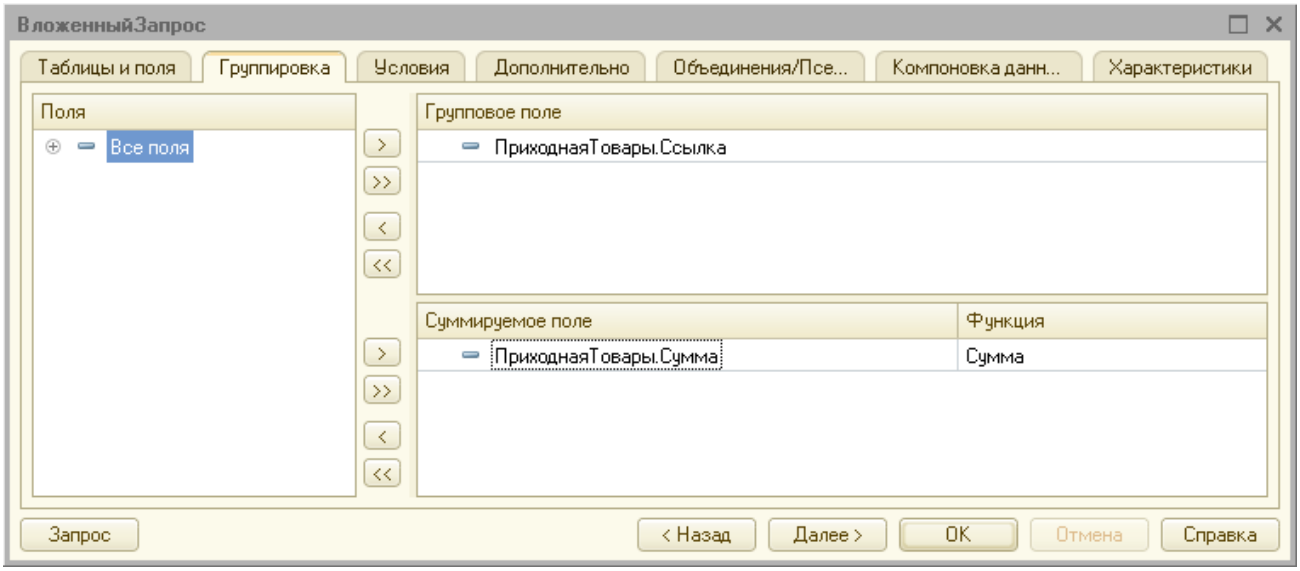
В «Конструкторе запроса» видно, что напрямую нет доступа к сумме документа. Сумма есть в табличной части «Товары»:



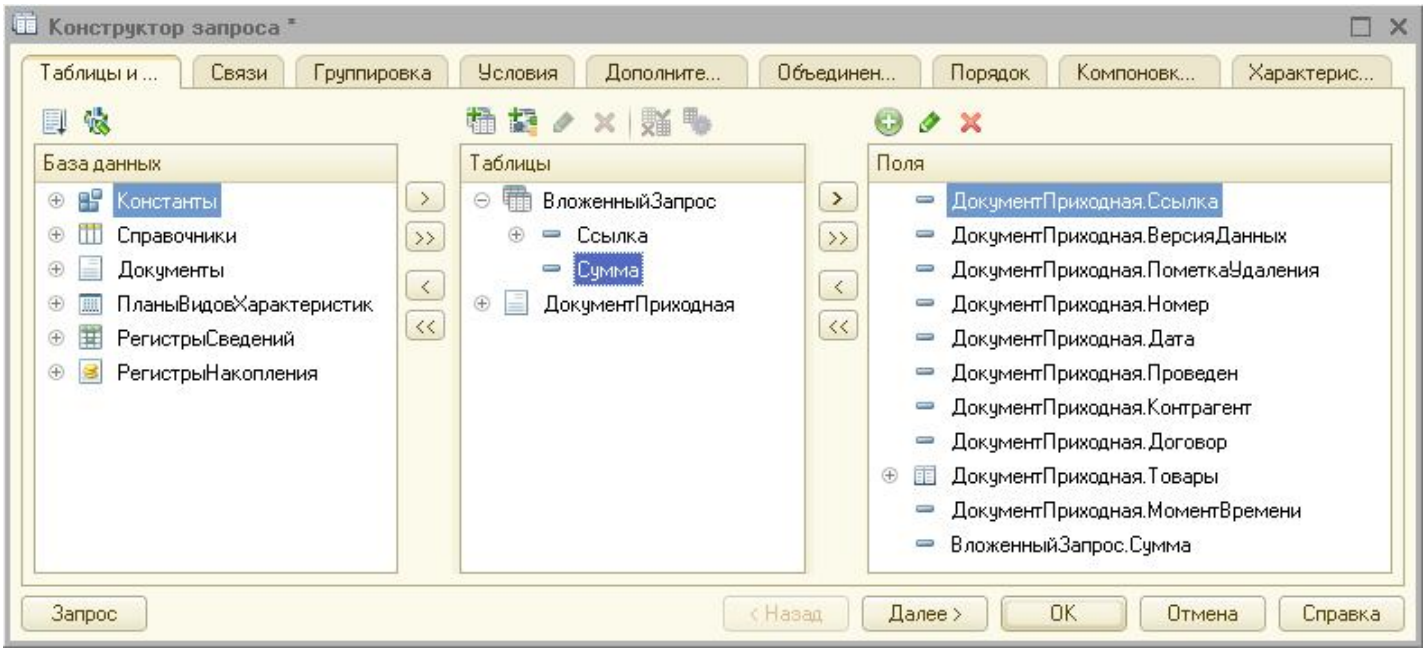
Чтобы привязать сумму документа, сделаем вложенный запрос:



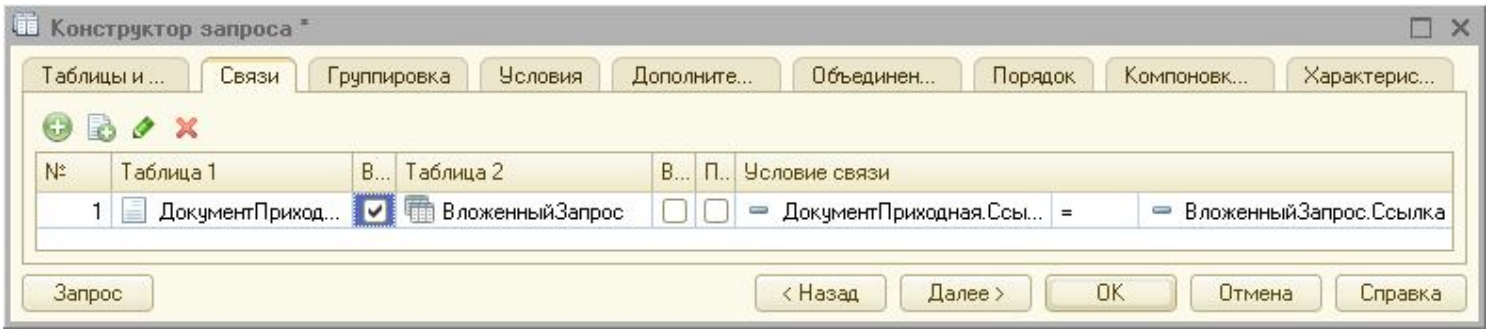
И сгруппируем в нем выбранные поля:



И из этого вложенного запроса итоговую сумму выберем в основной запрос:



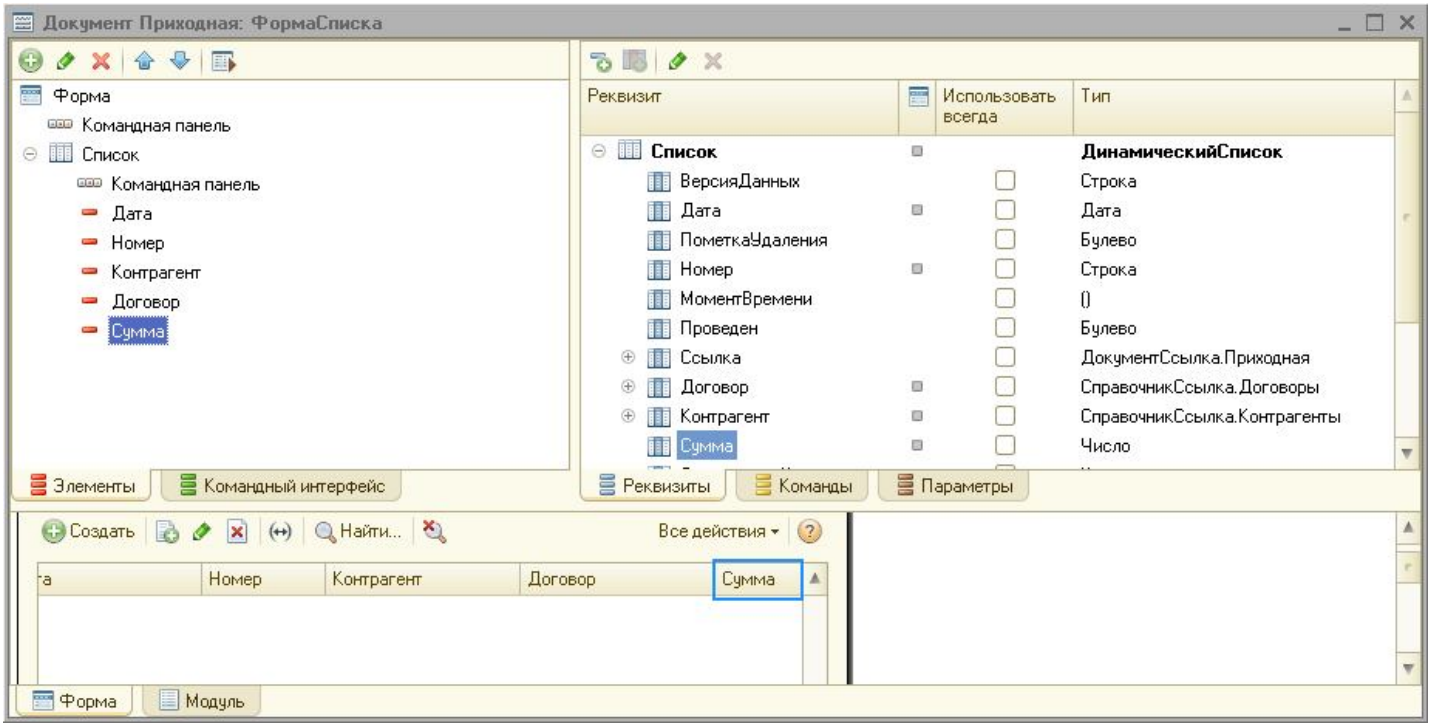
Т.к. теперь две таблицы, то их надо связать. Реквизит «Ссылка» и документе и во вложенном запросе имеет одинаковое значение – ссылка на документ «Приходная». По нему и свяжем таблицы:



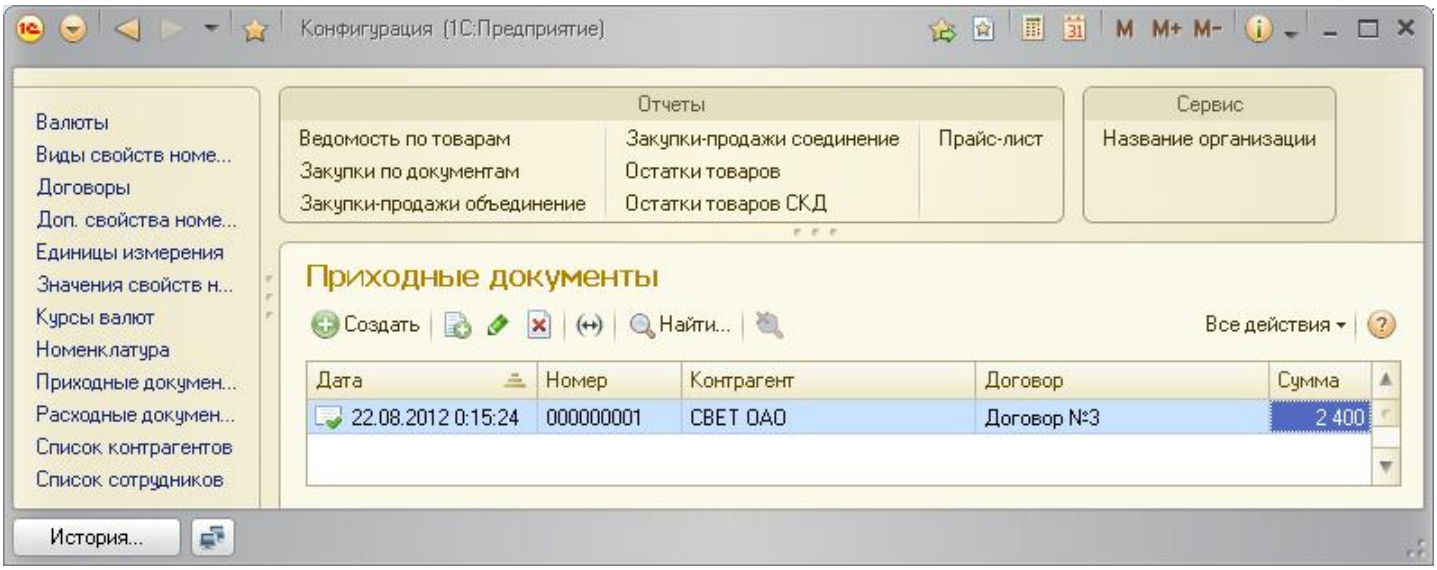
Итого запрос:

```
ВЫБРАТЬ
	ДокументПриходная.Ссылка,
	ДокументПриходная.ВерсияДанных,
	ДокументПриходная.ПометкаУдаления,
	ДокументПриходная.Номер,
	ДокументПриходная.Дата,
	ДокументПриходная.Проведен,
	ДокументПриходная.Контрагент,
	ДокументПриходная.Договор,
	ДокументПриходная.МоментВремени,
	ВложенныйЗапрос.Сумма
ИЗ
	Документ.Приходная КАК ДокументПриходная
		ЛЕВОЕ СОЕДИНЕНИЕ (ВЫБРАТЬ
			ПриходнаяТовары.Ссылка КАК Ссылка,
			СУММА(ПриходнаяТовары.Сумма) КАК Сумма
		ИЗ
			Документ.Приходная.Товары КАК ПриходнаяТовары

		СГРУППИРОВАТЬ ПО
			ПриходнаяТовары.Ссылка) КАК ВложенныйЗапрос
		ПО ДокументПриходная.Ссылка = ВложенныйЗапрос.Ссылка
```

Теперь в списке появилось поле «Сумма», его можно добавить на форму:



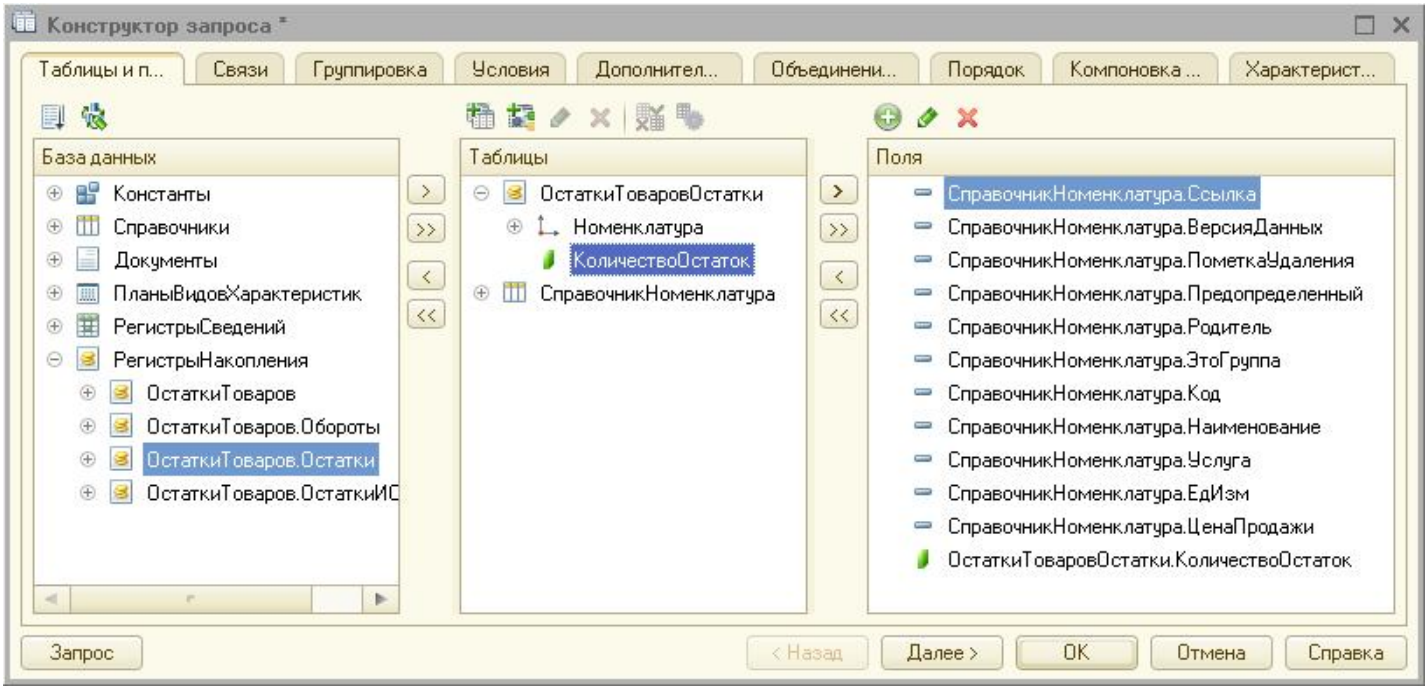
В режиме исполнения:



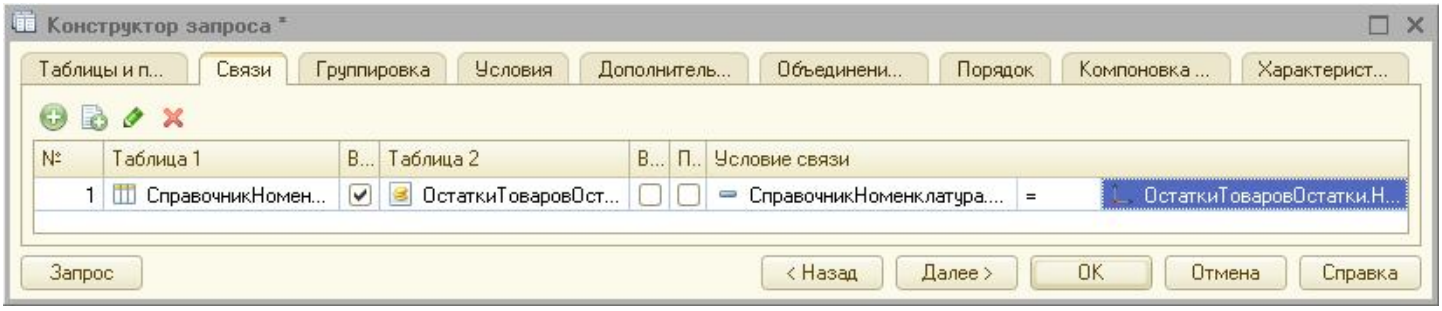
При «ПроизвольныйЗапрос = Истина» при открытии формы списка система кэширует на Сервере записи и передает клиенту первую тысячу. Если клиент пролистывает динамический список дальше тысячи, то система передает клиенту следующую выборку данных (каждая последующая выборка в два раза больше предыдущей). Аналогично проводятся обратные действия.

Задача: когда пользователь будет выбирать номенклатуру в табличной части документа «Расходная», надо чтобы в форме выбора списка товаров кроме «Кода» и «Наименования» выводился «Остаток».

Создадим «Форму выбора» из справочника «Номенклатура» и изменим запрос формирования динамического списка:

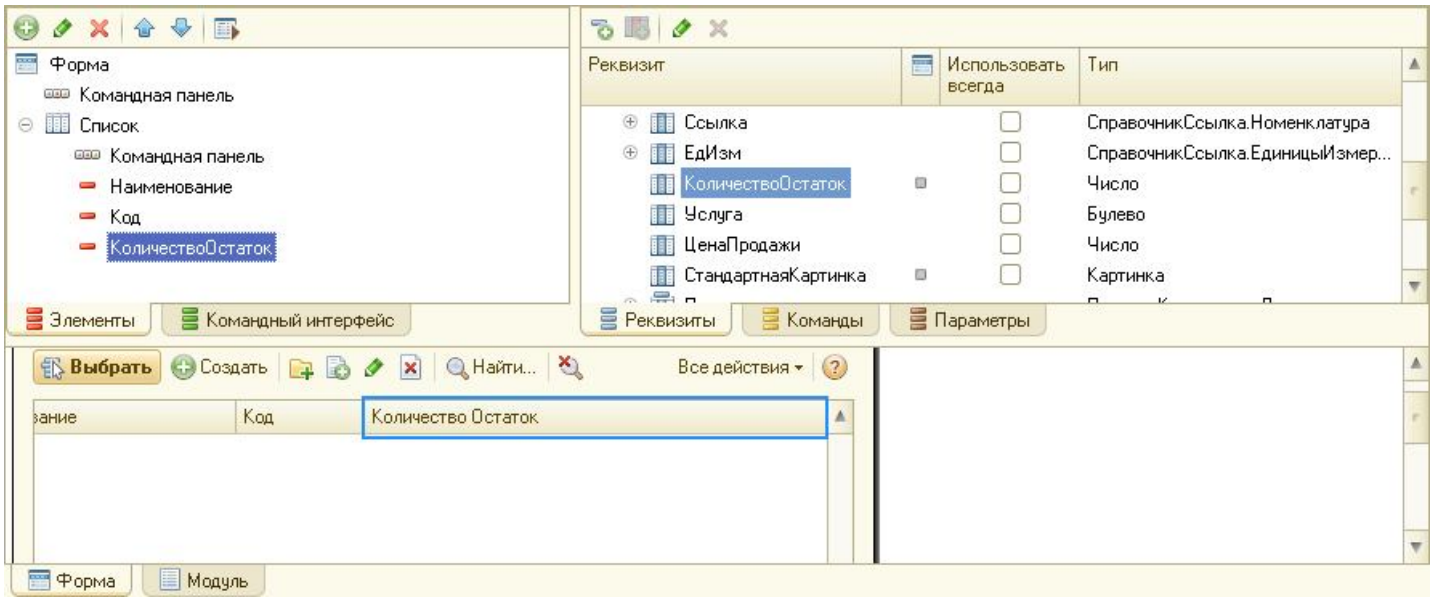


Свяжем таблицы:
Основная таблица – таблица справочника «Номенклатура»

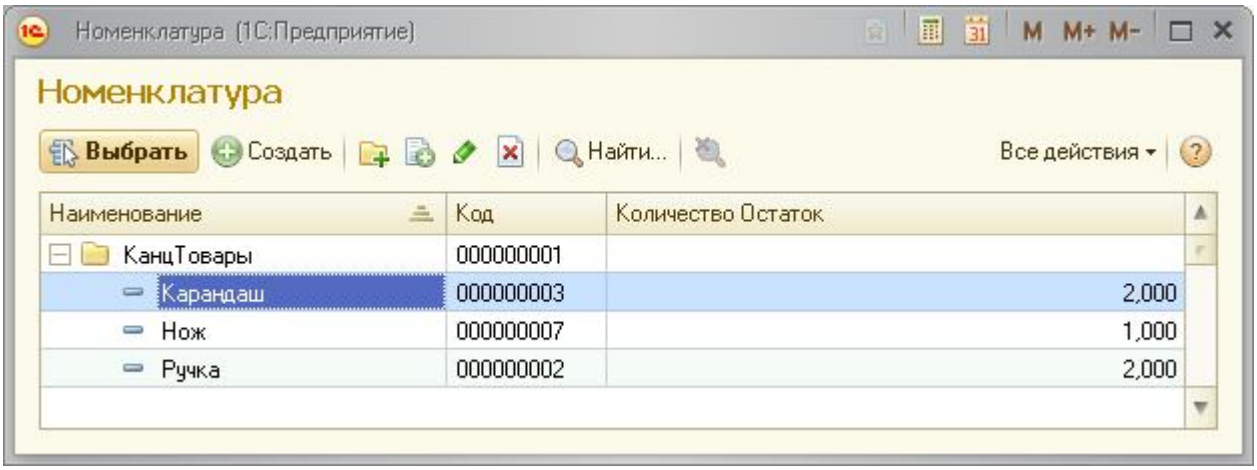



```
ВЫБРАТЬ
	СправочникНоменклатура.Ссылка,
	СправочникНоменклатура.ВерсияДанных,
	СправочникНоменклатура.ПометкаУдаления,
	СправочникНоменклатура.Предопределенный,
	СправочникНоменклатура.Родитель,
	СправочникНоменклатура.ЭтоГруппа,
	СправочникНоменклатура.Код,
	СправочникНоменклатура.Наименование,
	СправочникНоменклатура.Услуга,
	СправочникНоменклатура.ЕдИзм,
	СправочникНоменклатура.ЦенаПродажи,
	ОстаткиТоваровОстатки.КоличествоОстаток
ИЗ
	Справочник.Номенклатура КАК СправочникНоменклатура
		ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки КАК ОстаткиТоваровОстатки
		ПО СправочникНоменклатура.Ссылка = ОстаткиТоваровОстатки.Номенклатура
```

Вытащим поле с остатком на форму:



В режиме исполнения:



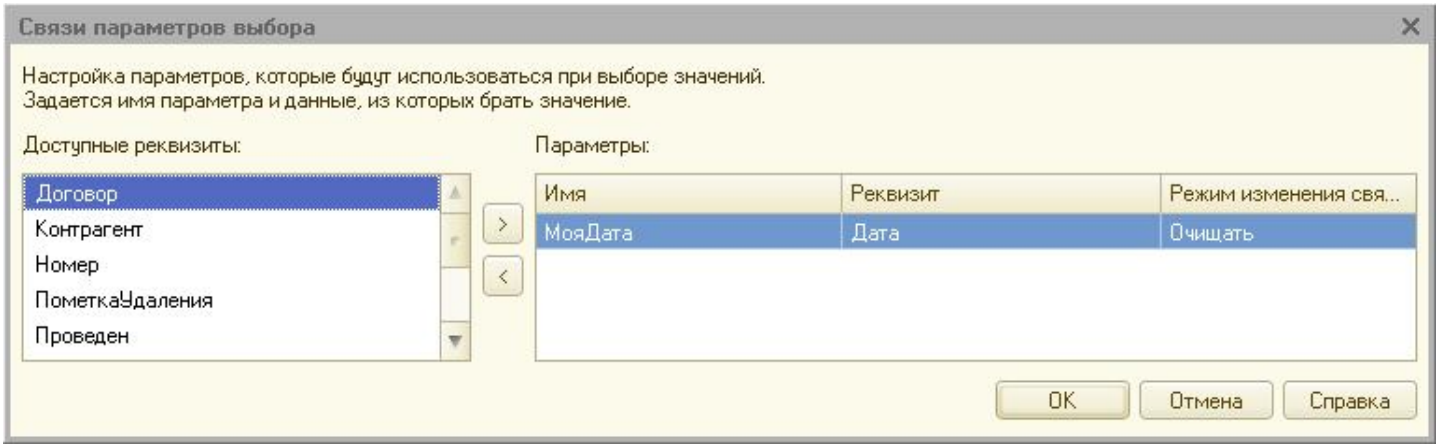
Можно смотреть остаток на любую дату, для этого надо в таблицу остатков передавать дату документа «Расходная». Но запрос к таблице остатков на произвольную дату – это трудоемкая и сравнительно медленная операция.

Сначала поймаем дату в динамическом списке, укажем ее в параметрах виртуальной таблицы остатков запроса:

```
ВЫБРАТЬ
	СправочникНоменклатура.Ссылка,
	СправочникНоменклатура.ВерсияДанных,
	СправочникНоменклатура.ПометкаУдаления,
	СправочникНоменклатура.Предопределенный,
	СправочникНоменклатура.Родитель,
	СправочникНоменклатура.ЭтоГруппа,
	СправочникНоменклатура.Код,
	СправочникНоменклатура.Наименование,
	СправочникНоменклатура.Услуга,
	СправочникНоменклатура.ЕдИзм,
	СправочникНоменклатура.ЦенаПродажи,
	ОстаткиТоваровОстатки.КоличествоОстаток
ИЗ
	Справочник.Номенклатура КАК СправочникНоменклатура
		ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки(&Дата) КАК
ОстаткиТоваровОстатки
		ПО СправочникНоменклатура.Ссылка =
ОстаткиТоваровОстатки.Номенклатура
```

Передадим дату из документа «Расходная» сначала в форму выбора справочника «Номенклатура»:

Укажем, что реквизит «Номенклатура» табличной части «Товары» документа «Расходная» использует «Связи параметров выбора» (зададим одноименное свойство):
В открываемую форму хотим получить дату, т.е. дата реквизита «Номенклатура» документа передается в форму, где она будет называться «МояДата»:



Когда будет открываться форма выбора справочника «Номенклатура», то в ней опишем событие создания формы на Сервере (до передачи Клиенту). Поймаем здесь переданную дату:

```
&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
    Если Параметры.Свойство("МояДата") = Неопределено Тогда
        Дата = '00010101';
    Иначе
        Дата = Параметры.МояДата;
    КонецЕсли;

    Список.Параметры.УстановитьЗначениеПараметра("Дата", Дата);
КонецПроцедуры
```

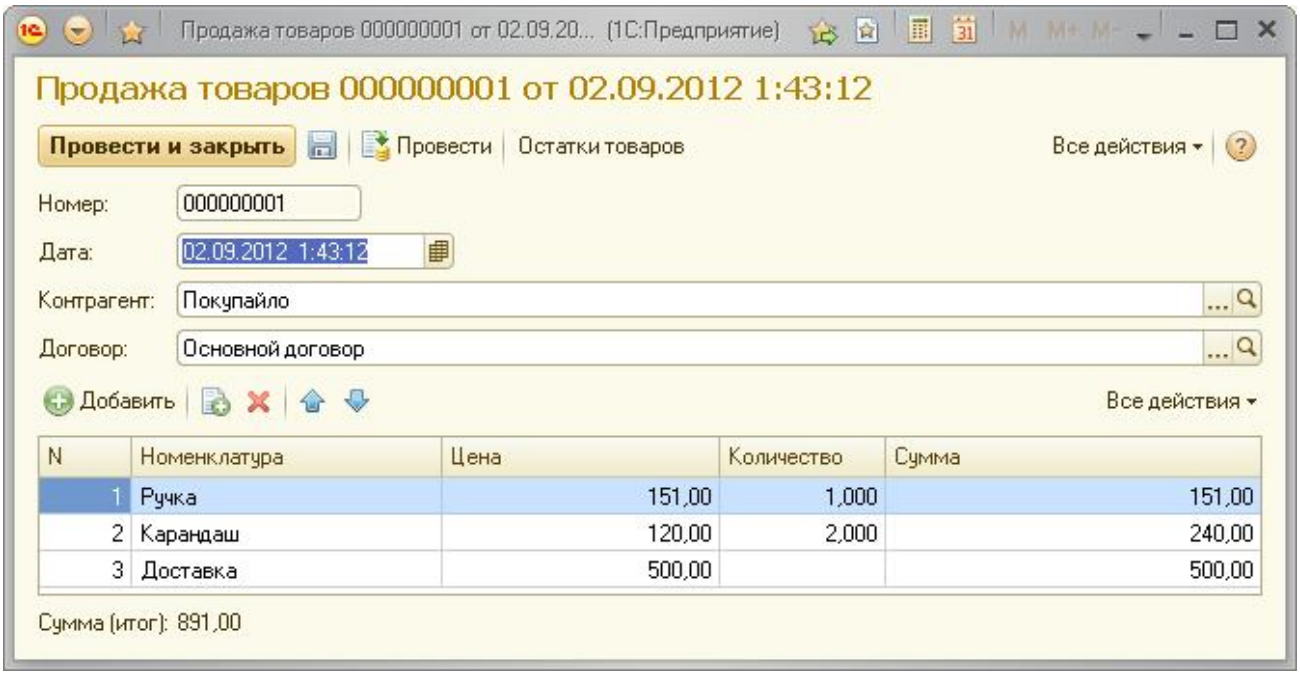
«Параметры» - предопределенное свойство формы. Это все, что передается в форму при ее вызове. Есть конкретные параметры формы (например, которые мы передали – «МояДата»), а есть стандартные.

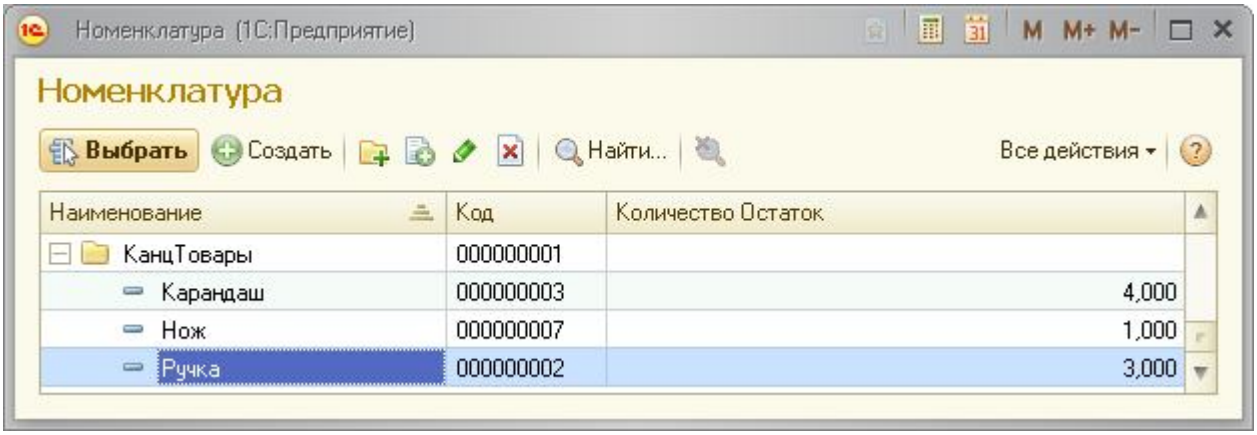
Дата в форму будет передаваться не из всех объектов, поэтому сделаем проверку: Если «МояДата» не передается, то данные получаются на дату Основных итогов.

Установим параметр в Динамическом списке.

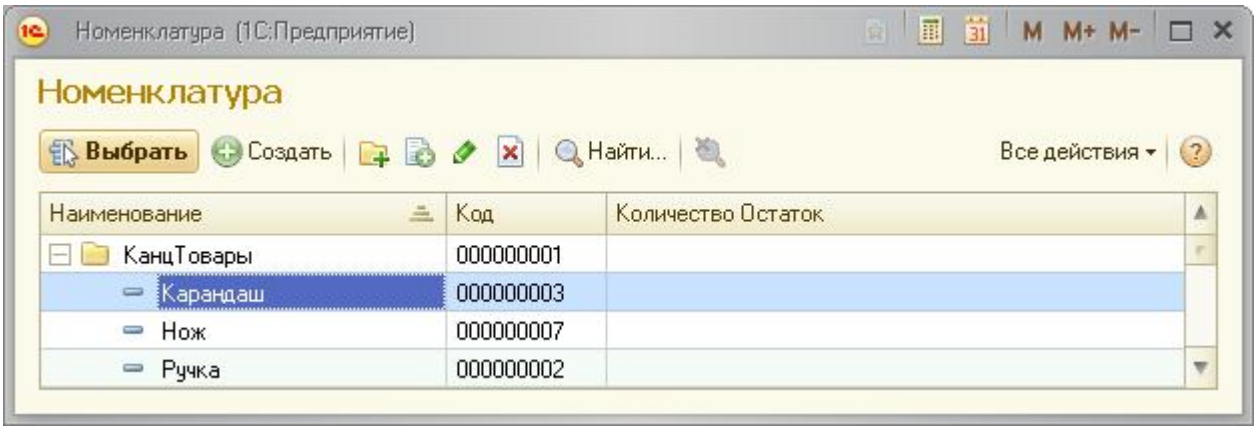
Закончил на 27 минуте

В режиме исполнения:





Поменяем дату документа на 02.08.2012. На эту дату товар еще даже не приходил, поэтому остатка нет:



В отчете «Остатки товаров СКД» надо показывать не только количественный остаток, но и суммовой.

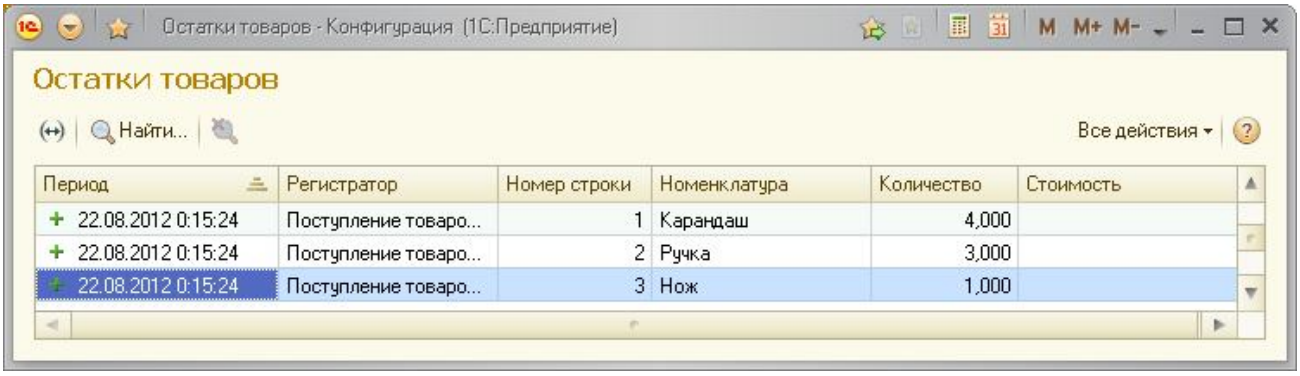
В регистр накопления «Остатки товаров» добавим ресурс:

Свойство	Значение
Стоимость	
Имя	Стоимость
Синоним	Стоимость
Тип	Число 15, 2

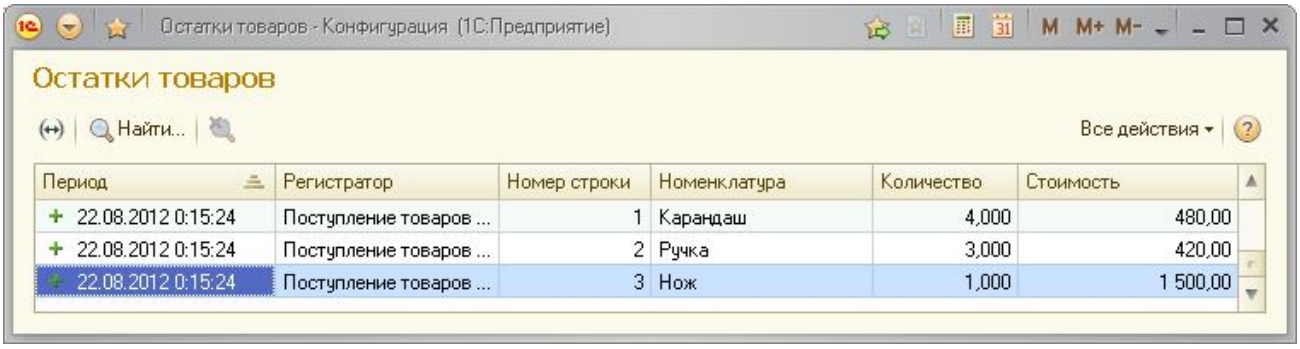
Это именно покупная стоимость товара.
При проведении документа «Приходная» будем увеличивать сумму стоимостей товара. В модуле объекта допишем движения по регистру «ОстаткиТоваров»:

```
Движение.Стоимость = ТекСтрокаТовары.Сумма;
```

В режиме исполнения:
Перейдем к регистру накопления «ОстаткиТоваров» из документа «Приходная»:



Перепроведем документ «Приходная»:



Продублируем документ «Приходная».

При проведении документа «Расходная» остаток надо уменьшать:
В документе «Расходная» мы не можем просто добавить такую же строку в движения, т.к. поле «Сумма» в документе «Расходная» - это выручка, продажная стоимость.

Стоимость товара, которую будем списывать из регистра, надо рассчитать.

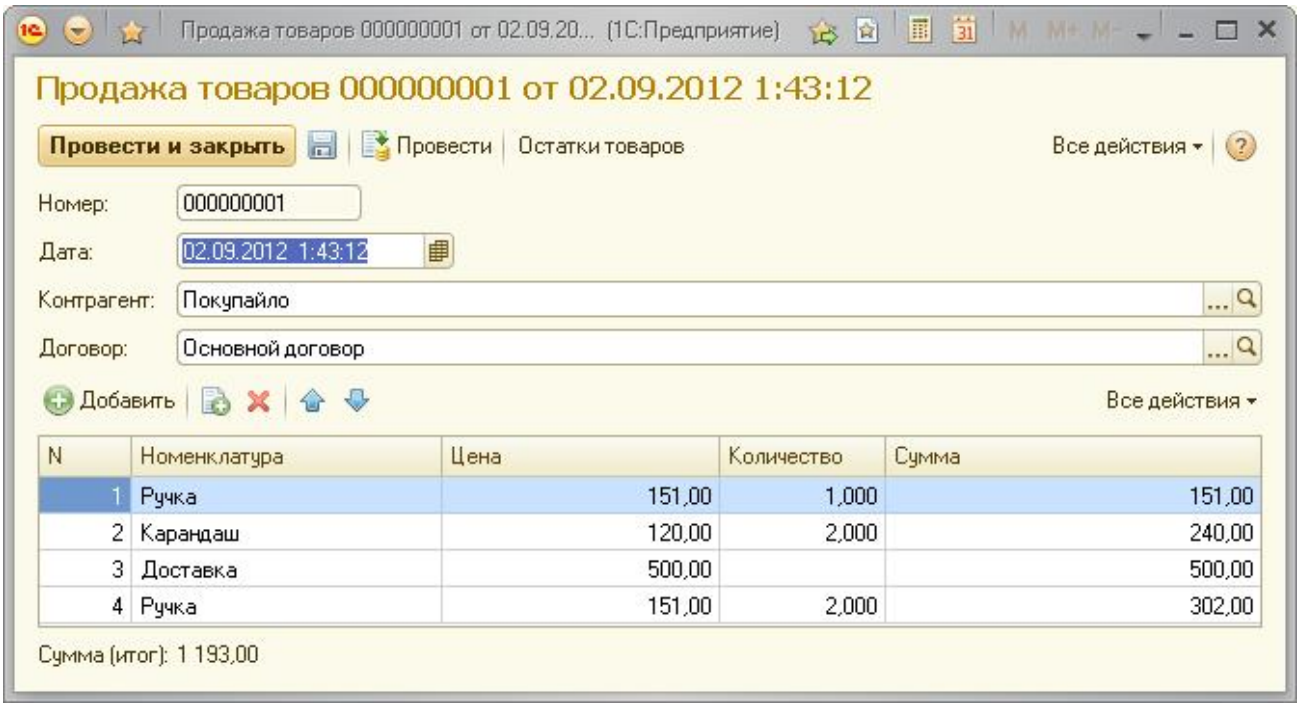
По средневзвешенной скользящей – метод расчета себестоимости товара:

Общий остаток товара - 10 штук.
Общая стоимость остатка - 100 рублей.

$$\text{себестоимость 1 штуки} = \frac{\text{общая стоимость остатка}}{\text{общее количество остатка}} * \text{продаваемое количество}$$

Продаваемое количество возьмем из документа «Расходная», а стоимость и количество остатка надо взять из регистра накопления «ОстаткиТоваров».

Подготовим тестовые данные:
Исправим документ «Расходная», предварительно отменив его проведение, и создадим второй, аналогичный и запишем их:



Продаем товары из документа №1:

Выберем табличную часть «Товары» - получим данные всех документов:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 000000001	2	Карандаш	120	2	240
Продажа товаров 000000001	4	Ручка	151	2	302
Продажа товаров 000000001	3	Доставка	500	0	500
Продажа товаров 000000001	1	Ручка	151	1	151
Продажа товаров 000000002	1	Ручка	151	1	151
Продажа товаров 000000002	2	Карандаш	120	2	240
Продажа товаров 000000002	3	Доставка	500	0	500
Продажа товаров 000000002	4	Ручка	151	2	302

Получим данные из всех документов «Расходная».

Получим данные только одного документа:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Документ
```

В качестве параметра выберем первый документ:

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 000000001	1	Ручка	151	1	151
Продажа товаров 000000001	2	Карандаш	120	2	240
Продажа товаров 000000001	3	Доставка	500	0	500
Продажа товаров 000000001	4	Ручка	151	2	302

Теперь к этой таблице привяжем таблицу «Остатки» регистра накопления «ОстаткиТоваров». Условие соединения - товары:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
	ДокТЧ.Номенклатура = Остатки.Номенклатура
ГДЕ
	Ссылка = &Документ
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма	Номенклатура1	КоличествоОста	СтоимостьОстат
Продажа товаро	1	Ручка	151	1	151	Ручка	6	840
Продажа товаро	2	Карандаш	120	2	240	Карандаш	8	960
Продажа товаро	3	Доставка	500	0	500	NULL	NULL	NULL
Продажа товаро	4	Ручка	151	2	302	Ручка	6	840

Уберем лишние поля – выберем те, которые нам нужны:

```
ВЫБРАТЬ
	ДокТЧ.Номенклатура,
	ДокТЧ.Количество,
	Остатки.КоличествоОстаток,
	Остатки.СтоимостьОстаток
ИЗ
	Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
	ДокТЧ.Номенклатура = Остатки.Номенклатура
```

```
ГДЕ
      Ссылка = &Документ
```

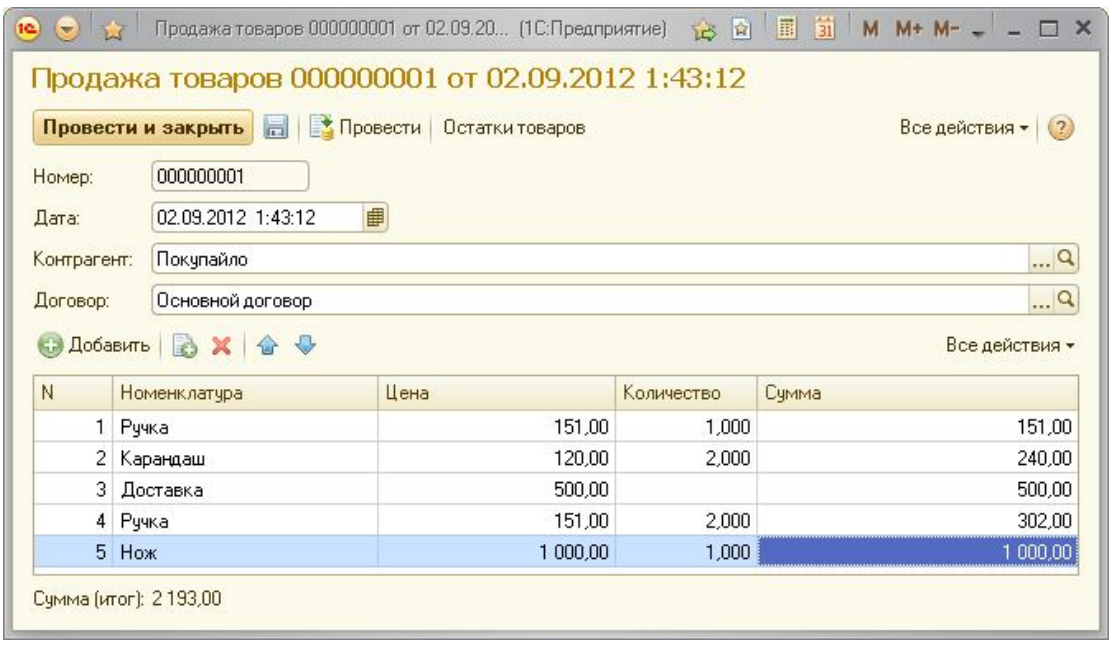
Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток
Ручка	1	6	840
Карандаш	2	8	960
Доставка	0	NULL	NULL
Ручка	2	6	840

Уберем Доставку, добавив условие:

```
ВЫБРАТЬ
      ДокТЧ.Номенклатура,
      ДокТЧ.Количество,
      Остатки.КоличествоОстаток,
      Остатки.СтоимостьОстаток
ИЗ
      Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
      РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
      ДокТЧ.Номенклатура = Остатки.Номенклатура
ГДЕ
      Ссылка = &Документ И НЕ ДокТЧ.Номенклатура.Услуга
```

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток
Ручка	1	6	840
Карандаш	2	8	960
Ручка	2	6	840

Но мы можем продавать товары, которых не существует:
Уберем элемент номенклатуры «Нож» из обоих приходных документов и добавим его в расходный документ:



Т.к. ножа нет, то из регистра получим значение «NULL»:

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток
Ручка	1	6	840
Карандаш	2	8	960
Ручка	2	6	840
Нож	1	NULL	NULL

Т.к. мы будем проверять наличие товара, то, чтобы при сравнении не выдавались ошибки о преобразовании типов, воспользуемся функцией «ЕСТЬNULL»:

```
ВЫБРАТЬ
      ДокТЧ.Номенклатура,
      ДокТЧ.Количество,
      ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
      ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток
ИЗ
      Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
      РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
      ДокТЧ.Номенклатура = Остатки.Номенклатура
ГДЕ
      Ссылка = &Документ И НЕ ДокТЧ.Номенклатура.Услуга
```

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток
Ручка	1	6	840
Карандаш	2	8	960
Ручка	2	6	840
Нож	1	0	0

В документе «Расходная» ручка дублируется. Итого мы продаем 3 ручки, т.е. поле «Количество» суммируем, а вот поле «КоличествоОстаток» дублируется, остаток ручек у нас всего 6.

Выполним группировку полей:

```
ВЫБРАТЬ
      ДокТЧ.Номенклатура,
      СУММА(ДокТЧ.Количество) КАК Количество,
      МАКСИМУМ(ЕСТЬNULL(Остатки.КоличествоОстаток, 0)) КАК КоличествоОстаток,
      МАКСИМУМ(ЕСТЬNULL(Остатки.СтоимостьОстаток, 0)) КАК СтоимостьОстаток
ИЗ
      Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
      РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
      ДокТЧ.Номенклатура = Остатки.Номенклатура
ГДЕ
```

```
    Ссылка = &Документ И НЕ ДокТЧ.Номенклатура.Услуга
СГРУППИРОВАТЬ ПО ДокТЧ.Номенклатура
```

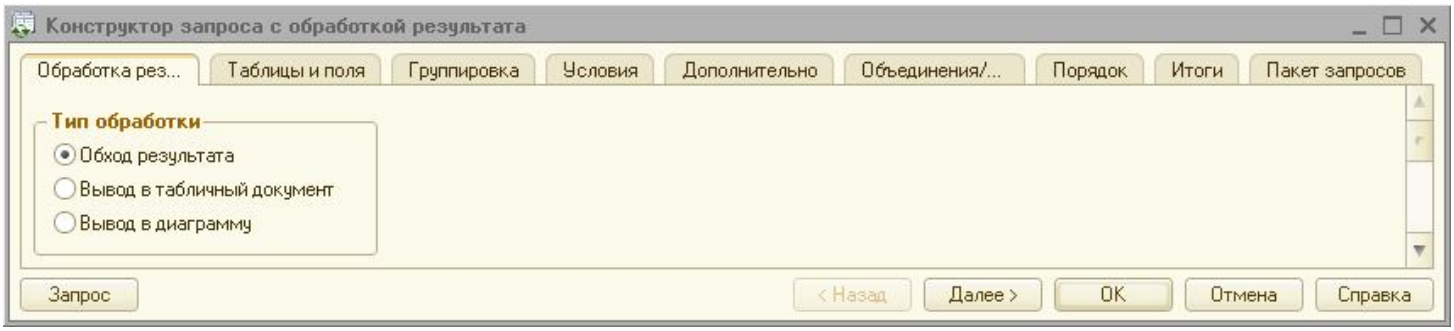
Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток
Нож	1	0	0
Ручка	3	6	840
Карандаш	2	8	960

Если в расчет остатков не передавать Дату, то остатки рассчитаются на последние движения, а иначе надо передавать Дату как параметр виртуальной таблицы регистра накопления:

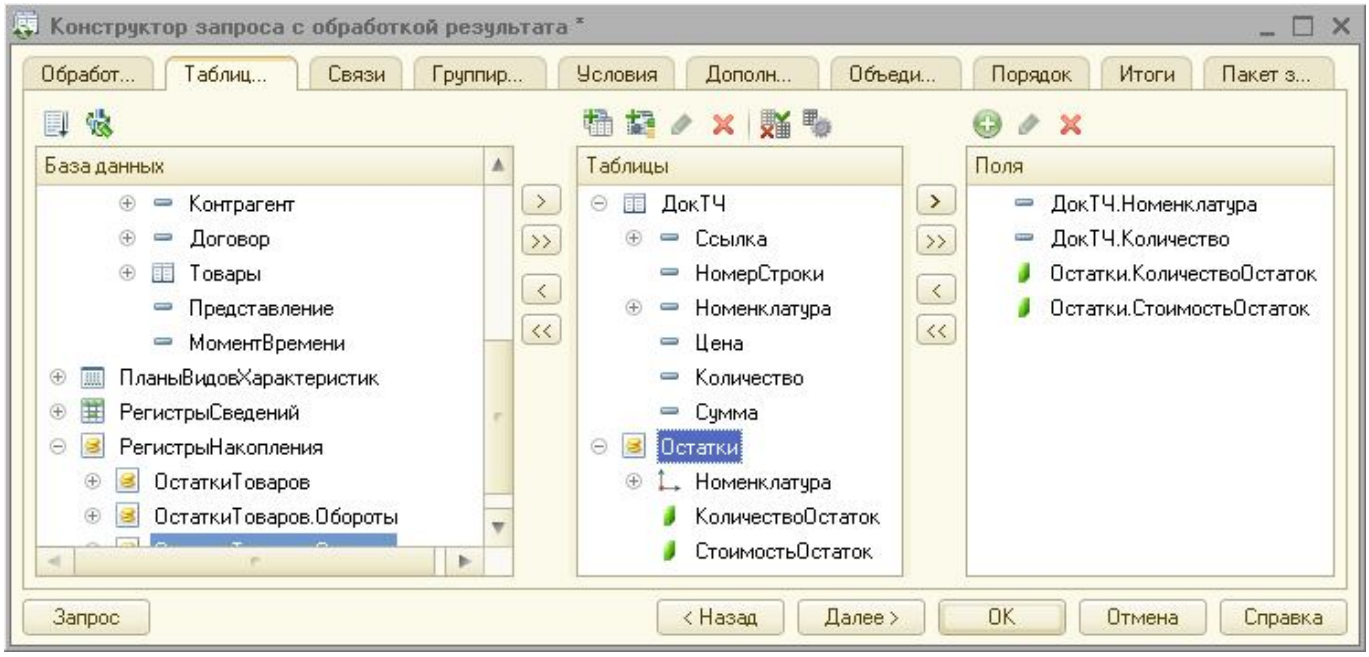
```
ВЫБРАТЬ
    ДокТЧ.Номенклатура,
    СУММА(ДокТЧ.Количество) КАК Количество,
    МАКСИМУМ(ЕСТЬNULL(Остатки.КоличествоОстаток, 0)) КАК КоличествоОстаток,
    МАКСИМУМ(ЕСТЬNULL(Остатки.СтоимостьОстаток, 0)) КАК СтоимостьОстаток
ИЗ
    Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
    РегистрНакопления.ОстаткиТоваров.Остатки(&МоментВремени) КАК Остатки
ПО
    ДокТЧ.Номенклатура = Остатки.Номенклатура
ГДЕ
    Ссылка = &Документ И НЕ ДокТЧ.Номенклатура.Услуга
СГРУППИРОВАТЬ ПО ДокТЧ.Номенклатура
```

Используем этот запрос при проведении документа «Расходная» - запустим «Конструктор запроса с обработкой результата»:

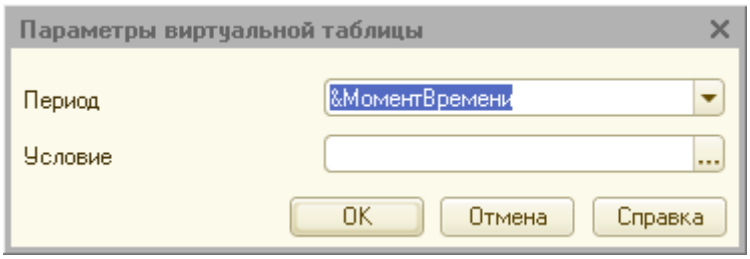
Просто обходим результат:



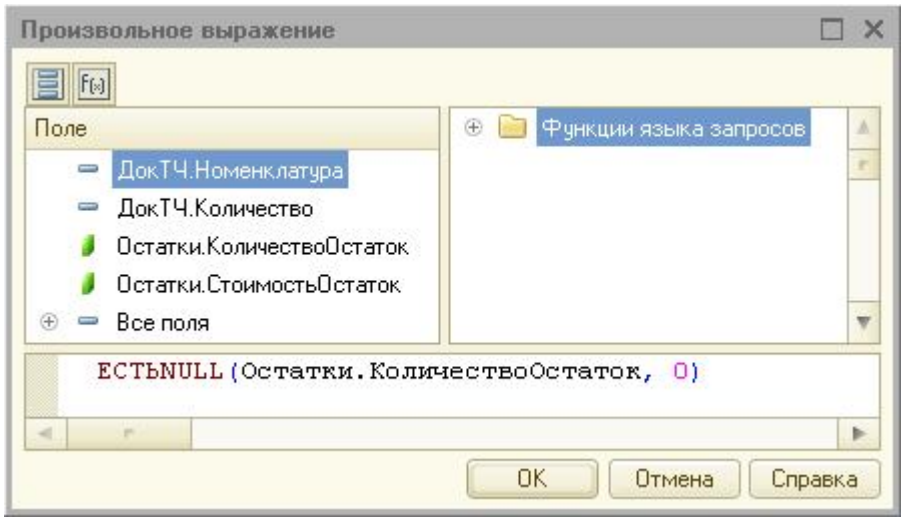
Выберем и переименуем табличную часть «Товары» документа «Расходная» и регистр накопления «ОстаткиТоваров.Остатки». Выберем из них необходимые поля:

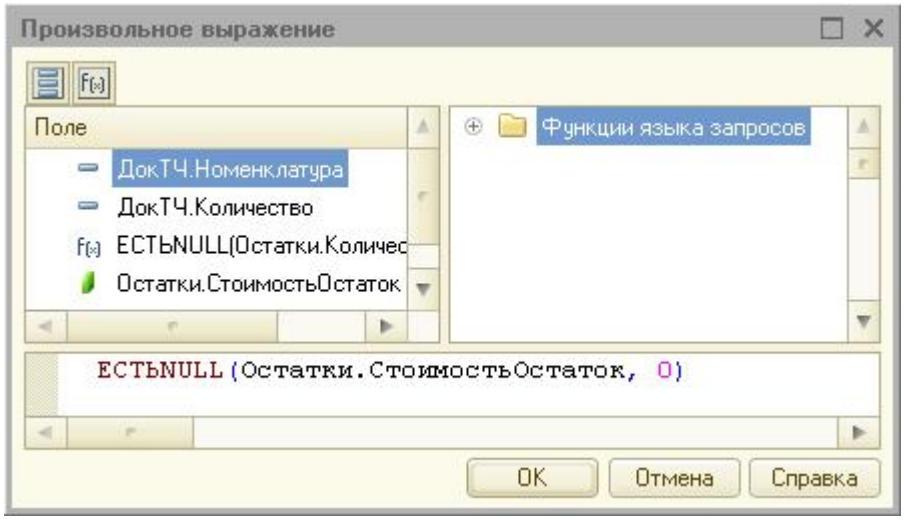


Укажем параметры виртуальной таблицы «Остатки»:

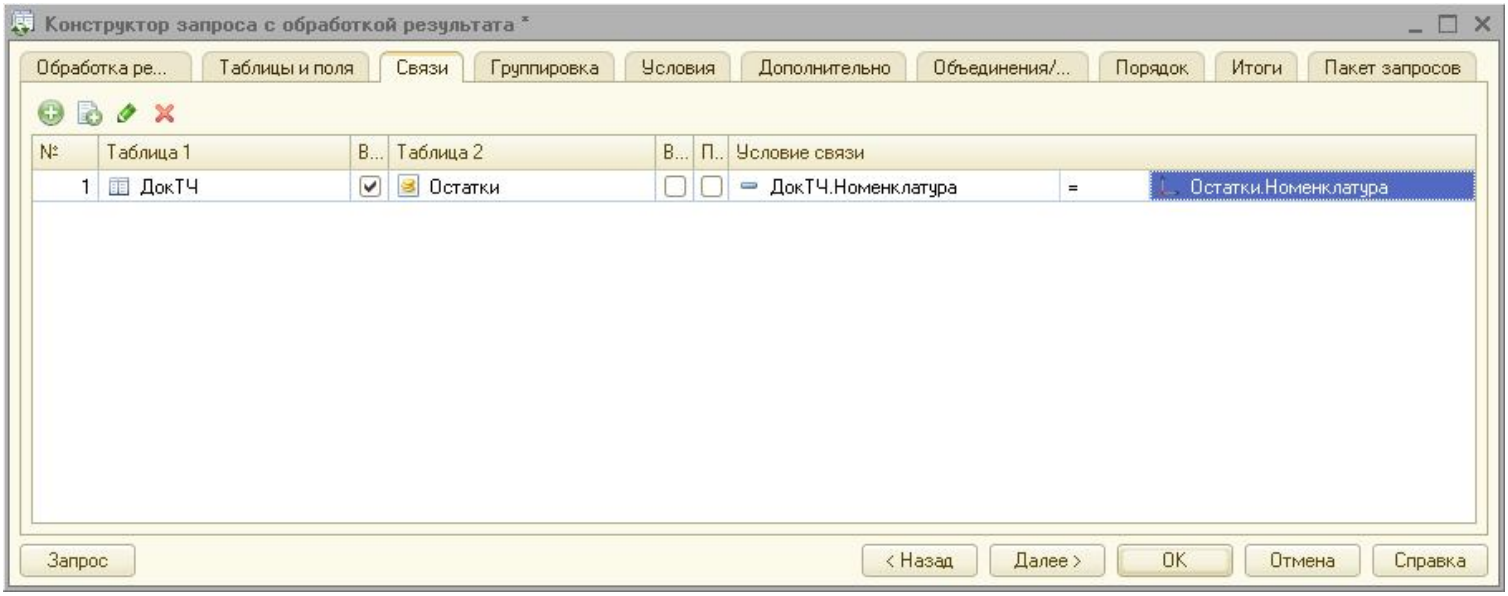


К полям «КоличествоОстаток» и «СтоимостьОстаток» применим функцию «ЕСТЬNULL» - это выполняется только вручную:

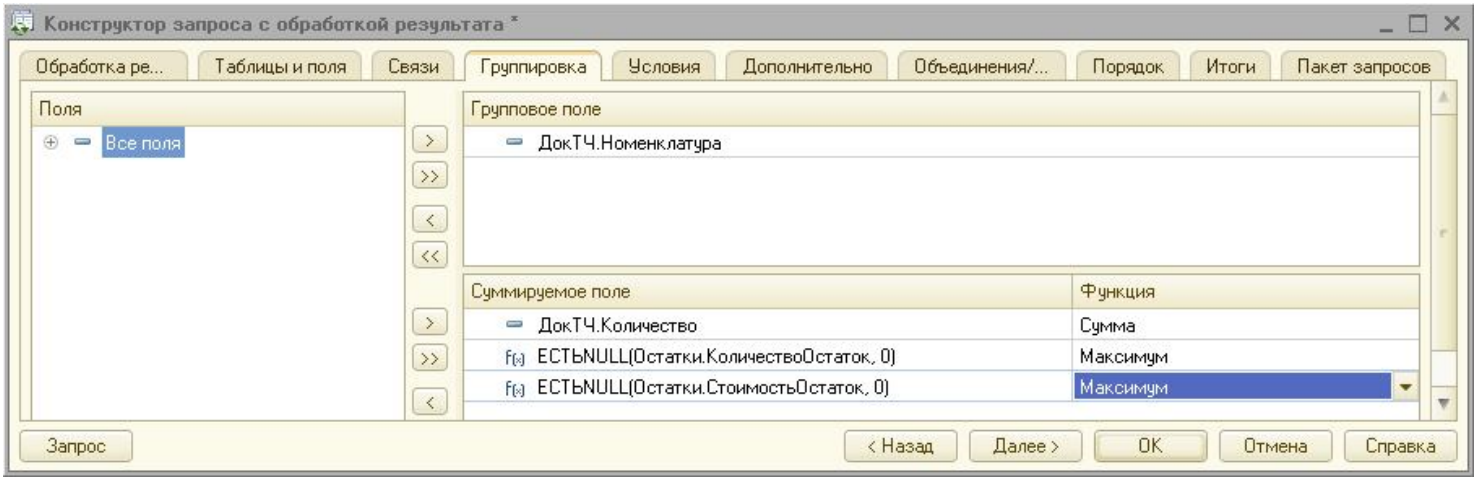




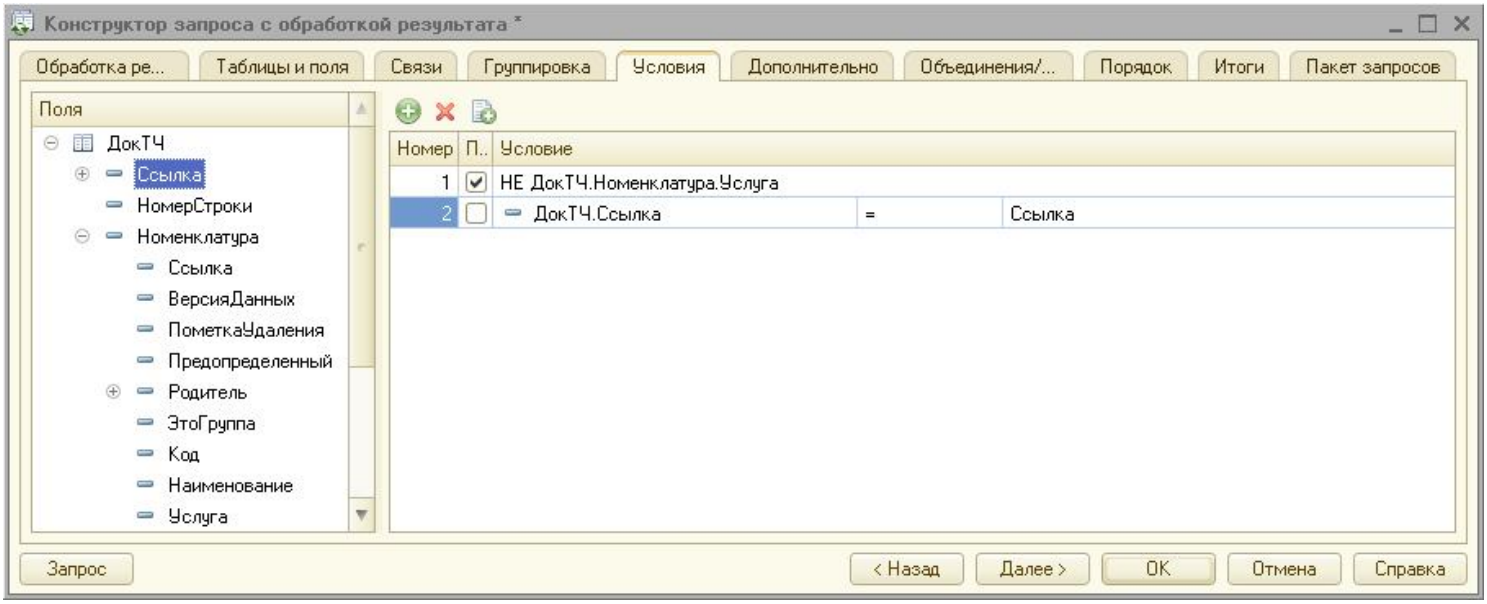
Свяжем таблицы:



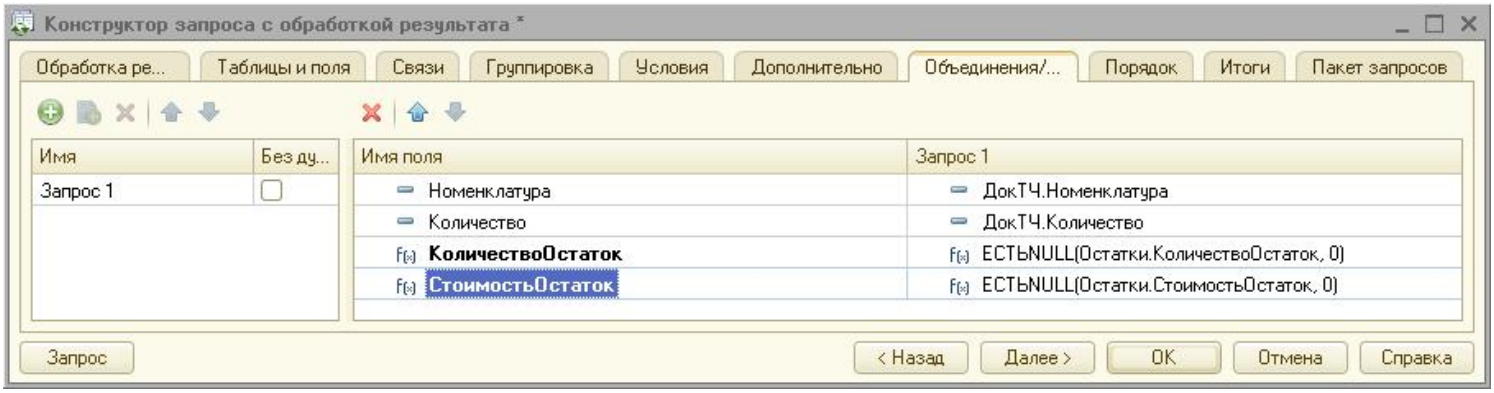
Группируем поля:



Условия:



Переименуем поля:



Система сформировала запрос и его обработку:

```
	Запрос = Новый Запрос;
	Запрос.Текст =
		"ВЫБРАТЬ
		|	ДокТЧ.Номенклатура,
		|	СУММА(ДокТЧ.Количество) КАК Количество,
		|	МАКСИМУМ(ЕСТЬNULL(Остатки.КоличествоОстаток, 0)) КАК
КоличествоОстаток,
		|	МАКСИМУМ(ЕСТЬNULL(Остатки.СтоимостьОстаток, 0)) КАК
СтоимостьОстаток
		|ИЗ
		|	Документ.Расходная.Товары КАК ДокТЧ
		|		ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки
(&МоментВремени, ) КАК Остатки
		|		ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
		|ГДЕ
		|	НЕ ДокТЧ.Номенклатура.Услуга
		|	И ДокТЧ.Ссылка = &Ссылка
		|
		|СГРУППИРОВАТЬ ПО
		|	ДокТЧ.Номенклатура";
```

```
    Запрос.УстановитьПараметр("МоментВремени", МоментВремени);
    Запрос.УстановитьПараметр("Ссылка", Ссылка);

    Результат = Запрос.Выполнить();

    ВыборкаДетальныеЗаписи = Результат.Выбрать();

    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл

    КонецЦикла;
```

В значении параметра «МоментВремени» надо поставить «()», т.е. «МоментВремени()», т.к. это метод документа.

«МоментВремени()» - точная позиция документа на оси времени. Состоит из двух значений:

- Дата.
- Ссылка на документ.

Внутри цикла обхода выборки проверим, хватает ли товара, если не хватает, то выводим предупреждение и запрещаем проведение документа:

```
Пока Выборка.Следующий() Цикл
    Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура
+ """ из необходимых " + Выборка.Количество
 + " в наличии осталось только " + Выборка.КоличествоОстаток;
        Сообщение.Сообщить();
        Отказ = Истина;
    КонецЕсли;
КонецЦикла;
```

Если же товара хватает, то формируем движения

```
Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
Движение.Период = Дата;
Движение.Номенклатура = Выборка.Номенклатура;
Движение.Количество = Выборка.Количество;
Движение.Стоимость = Выборка.Количество / Выборка.КоличествоОстаток *
Выборка.СтоимостьОстаток;
```

Формула расчета себестоимости – находит отношение списываемого количества к общему остатку. Почему так: если списывается все количество, то пропорция вернет единицу и себестоимость товара гарантированно спишется вся.

При использовании формулы, описанной выше, сначала производится расчет средней стоимости одной штуки, а потом умножение на количество продаваемого товара. Из-за проблем с округлением в регистре могут образоваться лишние копейки

С Отказом рекомендуется указать отмену записей регистра

```
Движения.ОстаткиТоваров.Записывать = Ложь;
```

А до начала цикла запись данных можно включить:

```
Движения.ОстаткиТоваров.Записывать = Истина;
```

Чтобы не формировать движения впустую, вставим заглушку:

```
Если Отказ Тогда
     Продолжить;
КонецЕсли;
```

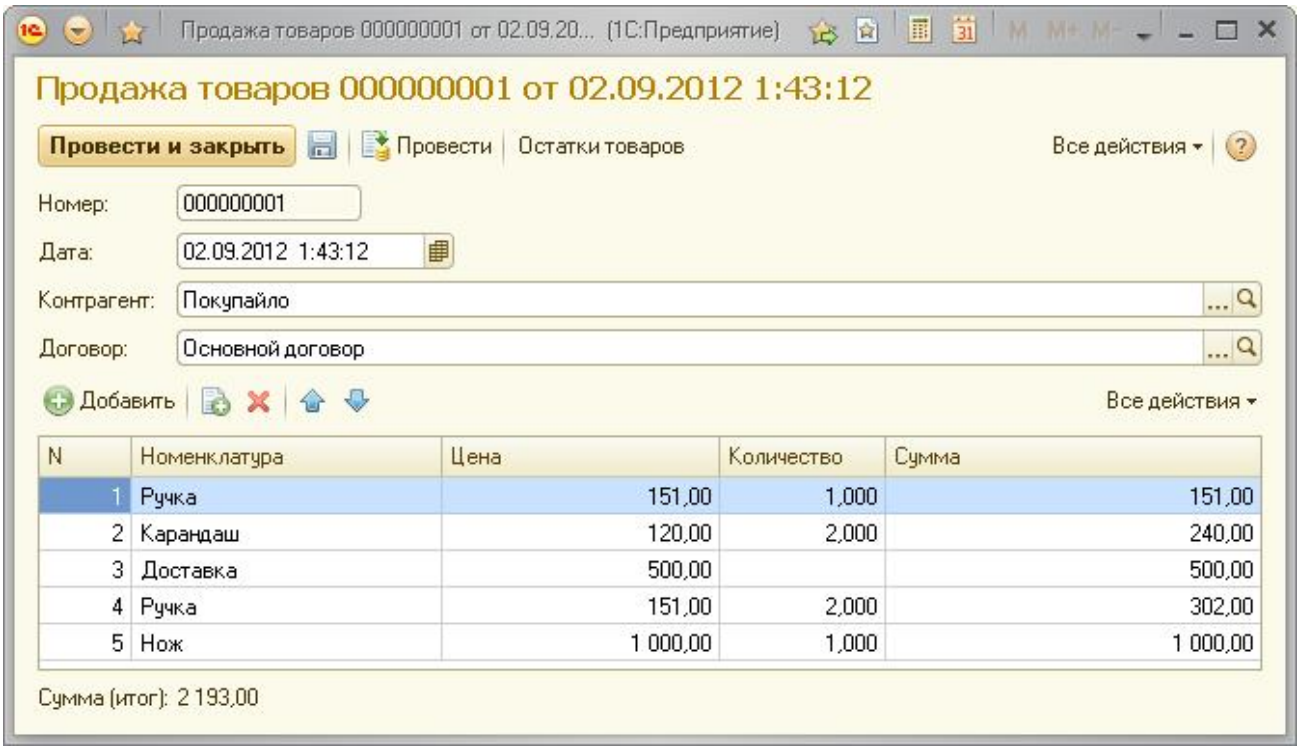
Итого процедура проведения:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

     Движения.ОстаткиТоваров.Записывать = Истина;
     Запрос = Новый Запрос;
     Запрос.Текст =
          "ВЫБРАТЬ
          |    ДокТЧ.Номенклатура,
          |    СУММА(ДокТЧ.Количество) КАК Количество,
          |    МАКСИМУМ(ЕСТЬNULL(Остатки.КоличествоОстаток, 0)) КАК
КоличествоОстаток,
          |    МАКСИМУМ(ЕСТЬNULL(Остатки.СтоимостьОстаток, 0)) КАК
СтоимостьОстаток
          |ИЗ
          |    Документ.Расходная.Товары КАК ДокТЧ
          |         ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.
Остатки(&МоментВремени, ) КАК Остатки
          |         ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
          |ГДЕ
          |    НЕ ДокТЧ.Номенклатура.Услуга
          |    И ДокТЧ.Ссылка = &Ссылка
          |
          |СГРУППИРОВАТЬ ПО
          |    ДокТЧ.Номенклатура";

     Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
     Запрос.УстановитьПараметр("Ссылка", Ссылка);

     Результат = Запрос.Выполнить();

     Выборка = Результат.Выбрать();

     Пока Выборка.Следующий() Цикл
          Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
               Сообщение = Новый СообщениеПользователю;
```

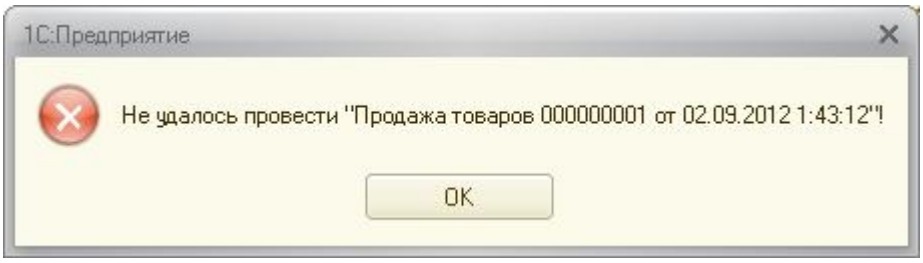
```
                Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура
+ """ из необходимых " + Выборка.Количество +
                " в наличии осталось только " + Выборка.КоличествоОстаток;
                Сообщение.Сообщить();
                Отказ = Истина;
                Движения.ОстаткиТоваров.Записывать = Ложь;
           КонецЕсли;

           Если Отказ Тогда
                Продолжить;
           КонецЕсли;

           Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
           Движение.Период = Дата;
           Движение.Номенклатура = Выборка.Номенклатура;
           Движение.Количество = Выборка.Количество;
           Движение.Стоимость = Выборка.Количество / Выборка.КоличествоОстаток
* Выборка.СтоимостьОстаток;

      КонецЦикла;
```

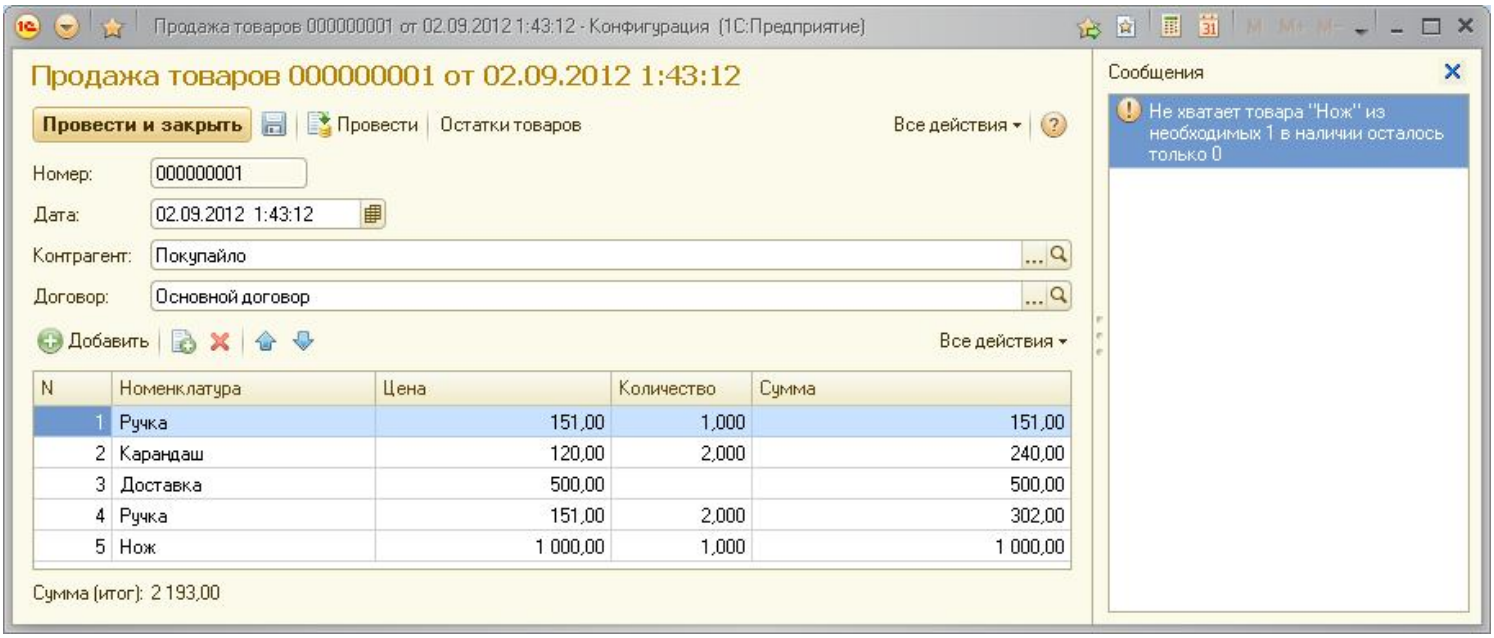
КонецПроцедуры

В режиме исполнения:

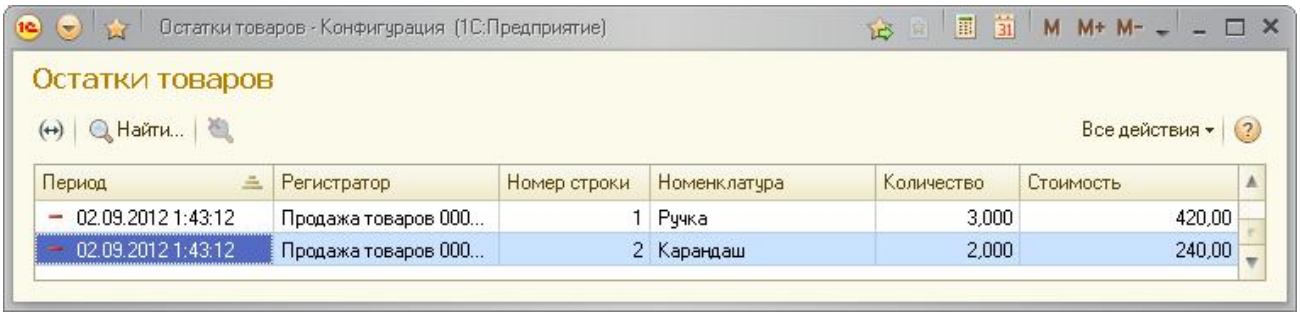


Ножа в приходе нет, поэтому документ не проведется:





Уберем нож, проведем документ и посмотрим движения по регистру «Остатки товаров»



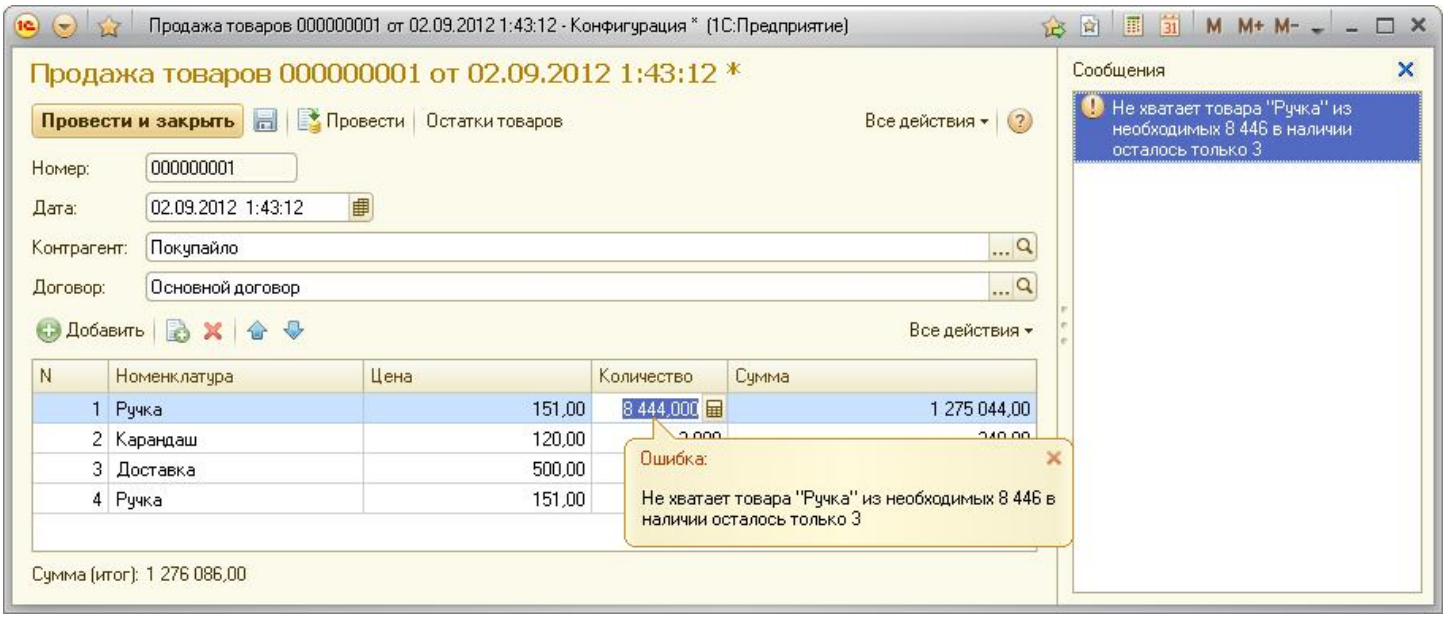
Сделаем, чтобы выводилось контекстное сообщение, которое будет выводиться у товара, которого не хватает:

Выведем пользователю сообщение, прикрепив его к данным. Чтобы прикрепить сообщение к данным, надо узнать номер строки, для которой предназначено сообщение

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

    Движения.ОстаткиТоваров.Записывать = Истина;
    Запрос = Новый Запрос;
    Запрос.Текст =
        "ВЫБРАТЬ
        |    МИНИМУМ(ДокТЧ.НомерСтроки) КАК НомерСтроки,
        |    ДокТЧ.Номенклатура,
        |    СУММА(ДокТЧ.Количество) КАК Количество,
        |    МАКСИМУМ(ЕСТЬNULL(Остатки.КоличествоОстаток, 0)) КАК
```

```
КоличествоОстаток,
          |    МАКСИМУМ(ЕСТЬNULL(Остатки.СтоимостьОстаток, 0)) КАК
СтоимостьОстаток
          |ИЗ
          |    Документ.Расходная.Товары КАК ДокТЧ
          |         ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.
Остатки(&МоментВремени, ) КАК Остатки
          |         ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
          |ГДЕ
          |    НЕ ДокТЧ.Номенклатура.Услуга
          |    И ДокТЧ.Ссылка = &Ссылка
          |
          |СГРУППИРОВАТЬ ПО
          |    ДокТЧ.Номенклатура";

     Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
     Запрос.УстановитьПараметр("Ссылка", Ссылка);

     Результат = Запрос.Выполнить();

     Выборка = Результат.Выбрать();

     Пока Выборка.Следующий() Цикл
          Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
               Сообщение = Новый СообщениеПользователю;
               Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура
+ """ из необходимых " + Выборка.Количество +
               " в наличии осталось только " + Выборка.КоличествоОстаток;
               Сообщение.Поле = "Товары[" + (Выборка.НомерСтроки - 1) + "].
Количество";
               Сообщение.УстановитьДанные(ЭтотОбъект);
               Сообщение.Сообщить();
               Отказ = Истина;
               Движения.ОстаткиТоваров.Записывать = Ложь;
          КонецЕсли;

          Если Отказ Тогда
               Продолжить;
          КонецЕсли;

          Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
          Движение.Период = Дата;
          Движение.Номенклатура = Выборка.Номенклатура;
          Движение.Количество = Выборка.Количество;
          Движение.Стоимость = Выборка.Количество / Выборка.КоличествоОстаток
* Выборка.СтоимостьОстаток;

     КонецЦикла;

КонецПроцедуры
```

В режиме исполнения:



Занятие 1.12

Можно ли добавить в регистр накопления «Остатки товаров» дополнительное измерение по складам?

Купим на склады ложки: на один склад – по 100 рублей, а на второй – по 200.

	Таблица движений			
	Номенклатура	**Склад**	**Количество**	**Цена**
+	Ложка	Парнас	1	100
+	Ложка	Шушары	1	200

Таблица остатков			
Номенклатура	**Склад**	**Количество**	**Цена**
Ложка	Парнас	1	100
Ложка	Шушары	1	200

Продадим одну ложку. Списываем по средневзвешенной скользящей:

	Таблица движений			
	Номенклатура	**Склад**	**Количество**	**Цена**
+	Ложка	Парнас	1	100
+	Ложка	Шушары	1	200
-	Ложка	Парнас	1	150

Таблица остатков			
Номенклатура	**Склад**	**Количество**	**Цена**
Ложка	Парнас	0	-50
Ложка	Шушары	1	200

То, что остаток на складе выражен только в суммовом эквиваленте – это не правильно.

Продадим ложку и со второго склада. Причем, себестоимость рассчитается внешне правильно:

	Таблица движений			
	Номенклатура	**Склад**	**Количество**	**Цена**
+	Ложка	Парнас	1	100
+	Ложка	Шушары	1	200
-	Ложка	Парнас	1	150
-	Ложка	Шушары	1	150

Таблица остатков			
Номенклатура	**Склад**	**Количество**	**Цена**
Ложка	Парнас	0	-50
Ложка	Шушары	0	50

А что, если рассчитывать среднее по складу?

Допустим, два склада стоят рядом. Но себестоимость списания нужна для оценки прибыли. А в данной ситуации получается, что мы, в зависимости от склада, получим абсолютно разный результат. Учитывать стоимость внутри склада имеет смысл для автономных складов

Т.е. хранить остатки в разрезе складов в регистре «Остатки товаров» не совсем правильно в данном примере.

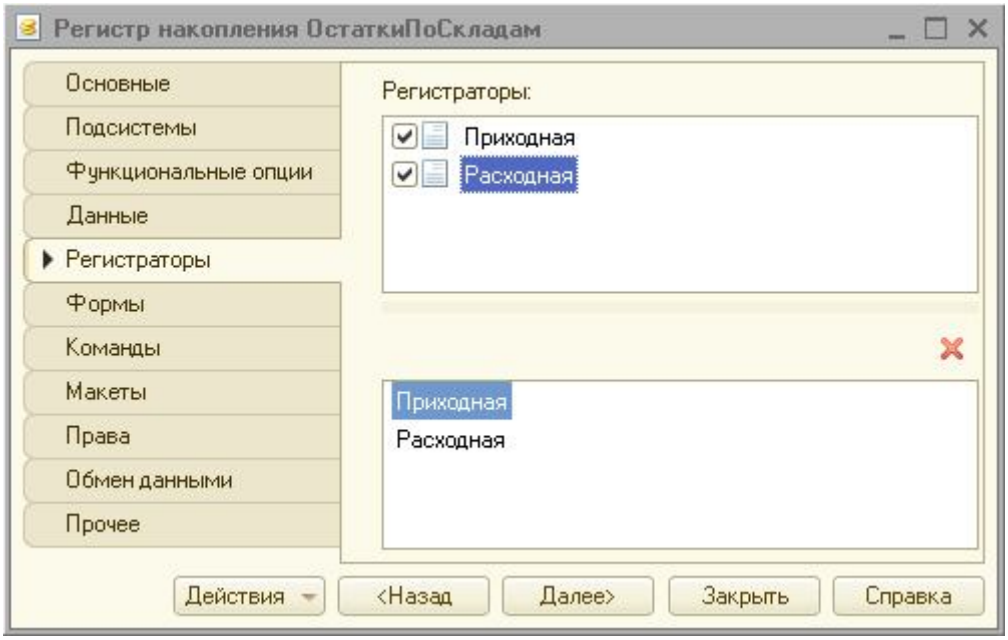
Надо создать новый регистр накопления «Остатки по складам»:

Предварительно создадим справочник «Склады»:

Свойство	**Значение**
Имя	Склады
Синоним	Склады
Представление объекта	Склад

Свойство	**Значение**
Имя	ОстаткиПоСкладам
Синоним	Остатки по складам
Закладка «Данные»	
Измерения	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	СправочникСсылка.Номенклатура
Запрет незаполненных значений	Истина
Склад	
Имя	Склад
Синоним	Склад
Тип	СправочникСсылка.Склады
Запрет незаполненных значений	Истина
Ресурсы	
Количество	
Имя	Количество
Синоним	Количество
Тип	Число 12, 3

Регистраторами будут оба документа:



Чтобы формировать движения по регистру, надо чтобы в документах были аналогичные реквизиты. Добавим в документы реквизит «Склад»:

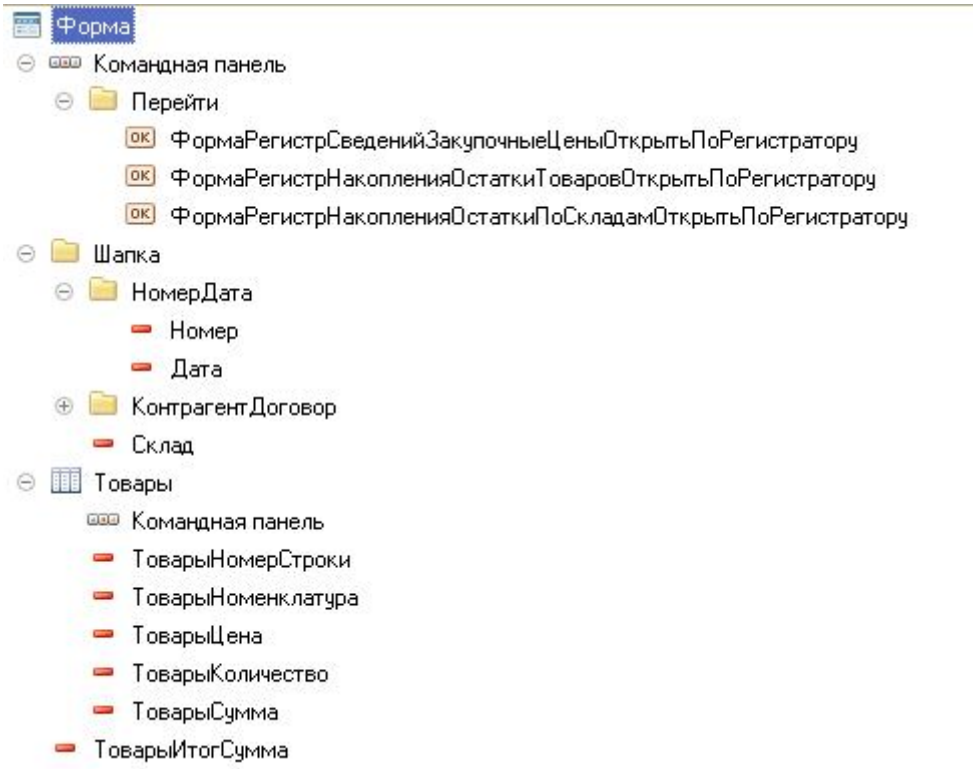
Свойство	Значение
Имя	Склад
Синоним	Склад
Тип	СправочникСсылка.Склады
Проверка заполнения	Выдавать ошибку

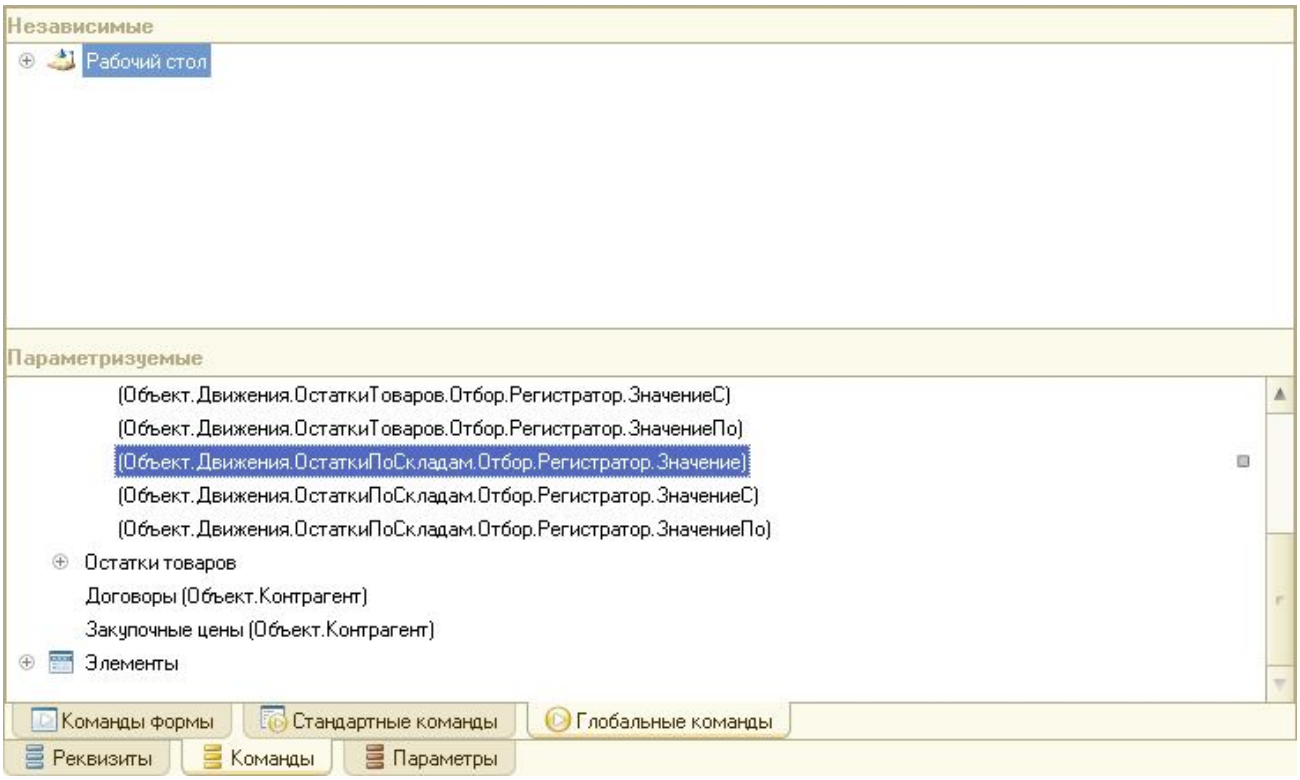
Поправим модуль объекта:

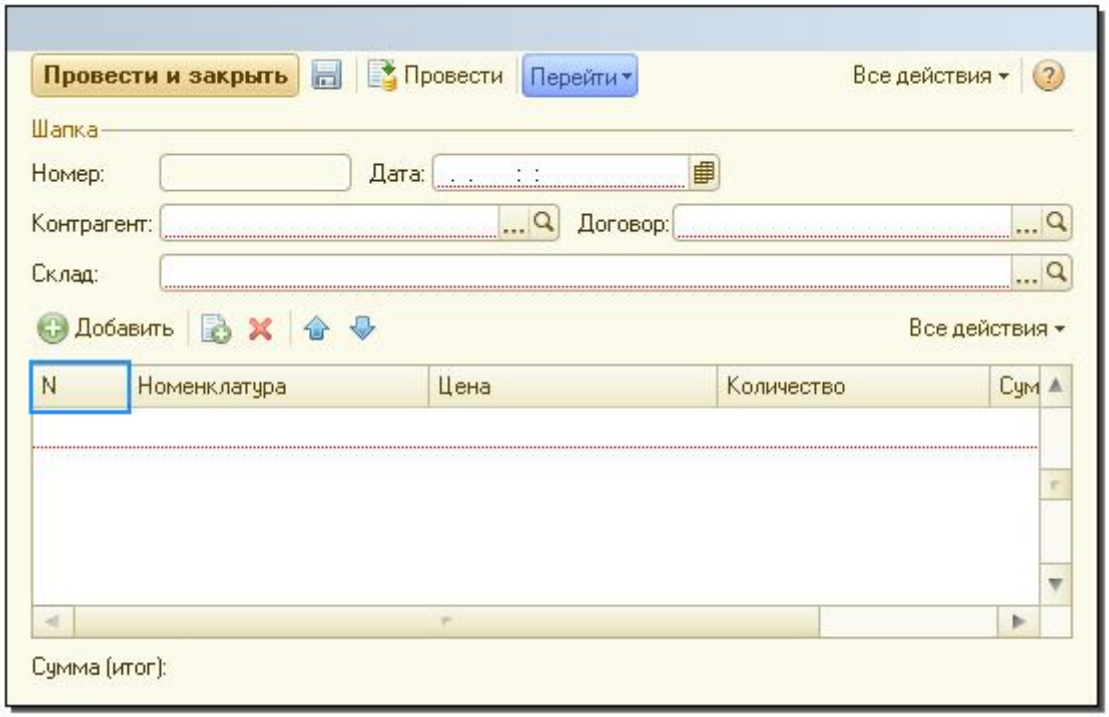
```
Процедура ОбработкаПроведения(Отказ, Режим)

	Движения.ЗакупочныеЦены.Записывать    = Истина;
	Движения.ОстаткиТоваров.Записывать    = Истина;
	Движения.ОстаткиПоСкладам.Записывать = Истина;

	Движения.ЗакупочныеЦены.Очистить();
	Движения.ОстаткиТоваров.Очистить();
	Движения.ОстаткиПоСкладам.Очистить();

	Для Каждого ТекСтрокаТовары Из Товары Цикл
		Движение = Движения.ЗакупочныеЦены.Добавить();
		Движение.Период = Дата;
		Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
		Движение.Контрагент = Контрагент;
		Движение.Цена = ТекСтрокаТовары.Цена;

		Движение = Движения.ОстаткиТоваров.ДобавитьПриход();
		Движение.Период = Дата;
		Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
		Движение.Количество = ТекСтрокаТовары.Количество;
		Движение.Стоимость = ТекСтрокаТовары.Сумма;
```

```
            Движение = Движения.ОстаткиПоСкладам.ДобавитьПриход();
            Движение.Период = Дата;
            Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
            Движение.Количество = ТекСтрокаТовары.Количество;
            Движение.Склад = Склад;

    КонецЦикла;

КонецПроцедуры
```

Добавим реквизит «Склад» на форму и изменим вид формы. Добавим команду на открытие сформированных движений по регистру «Остатки по складам»:







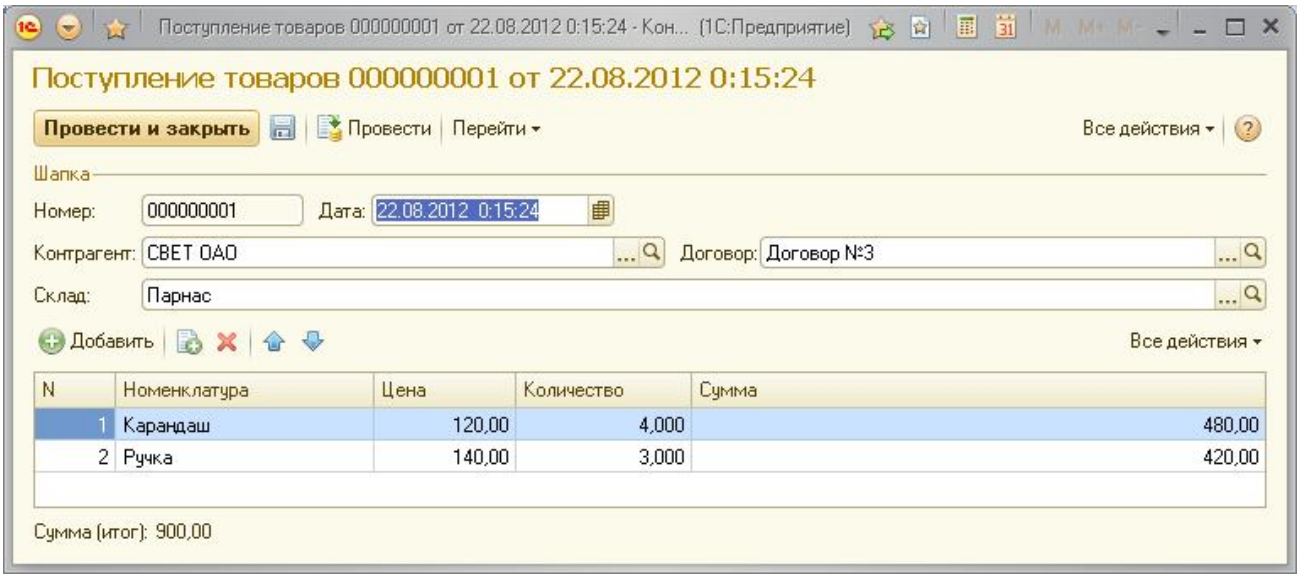
Свойства группировочных папок в форме:

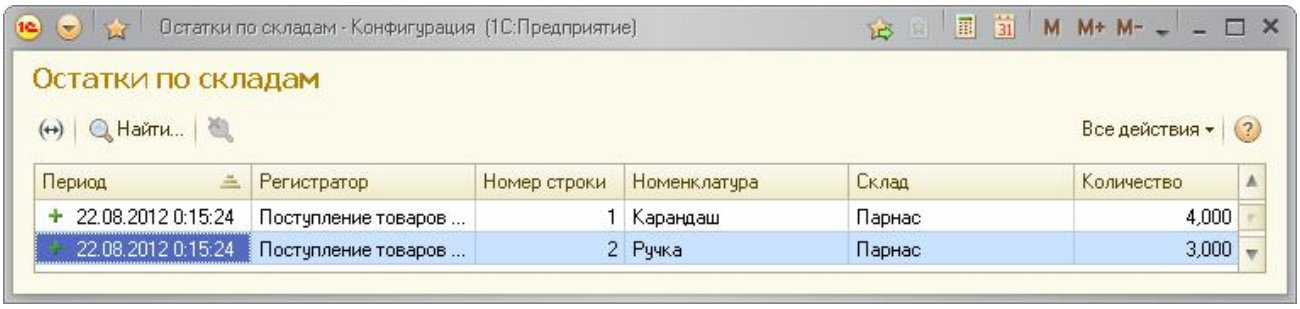
Свойство	Значение
Отображение	Нет
Группировка	Горизонтальная
ОтображатьЗаголовок	Ложь

В режиме исполнения:

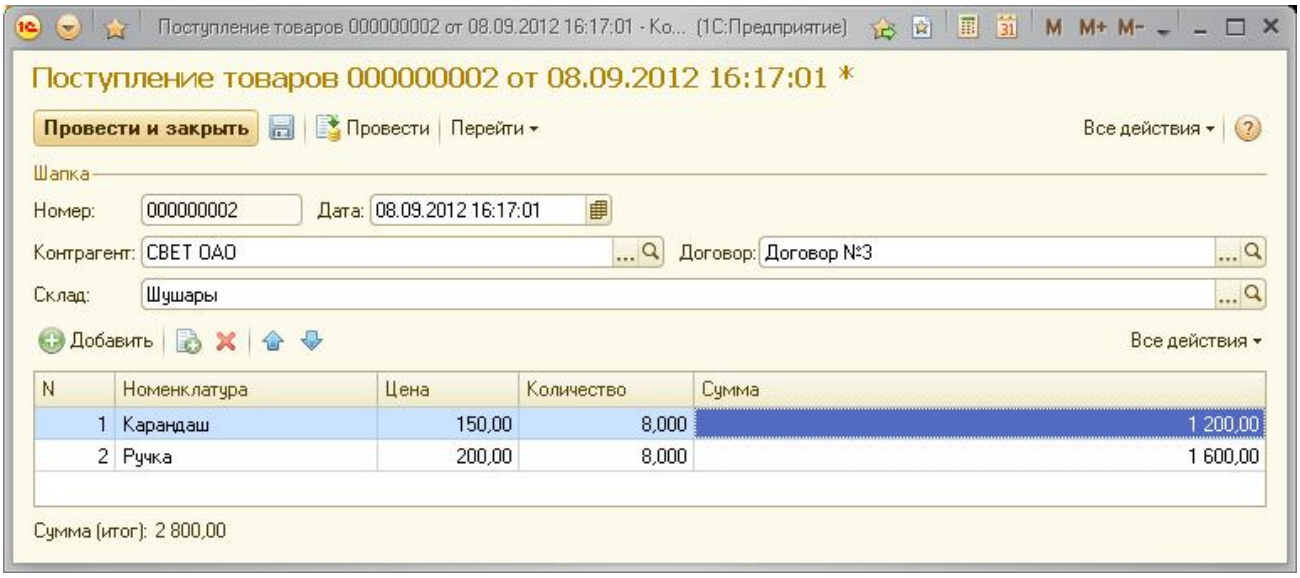
Отменим проведение у всех Расходных документов. Добавим склады «Парнас» и «Шушары», укажем их в Приходных документах и перепроведем их:

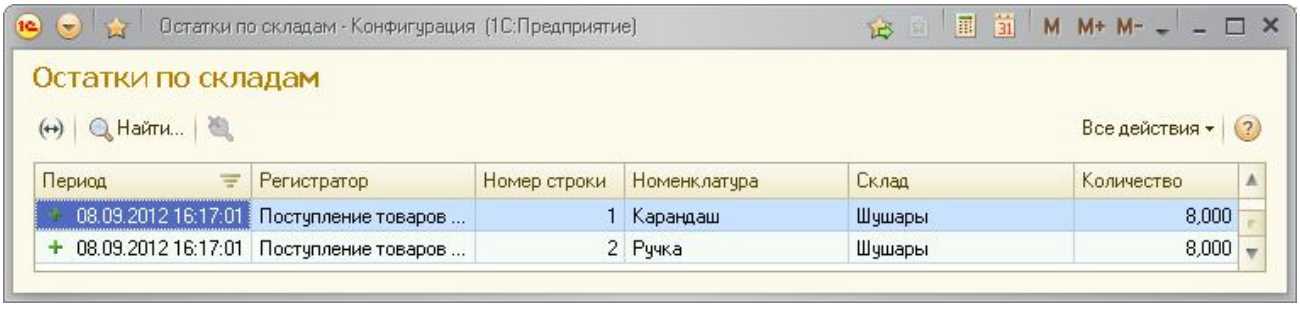
Первый документ:





Второй документ:



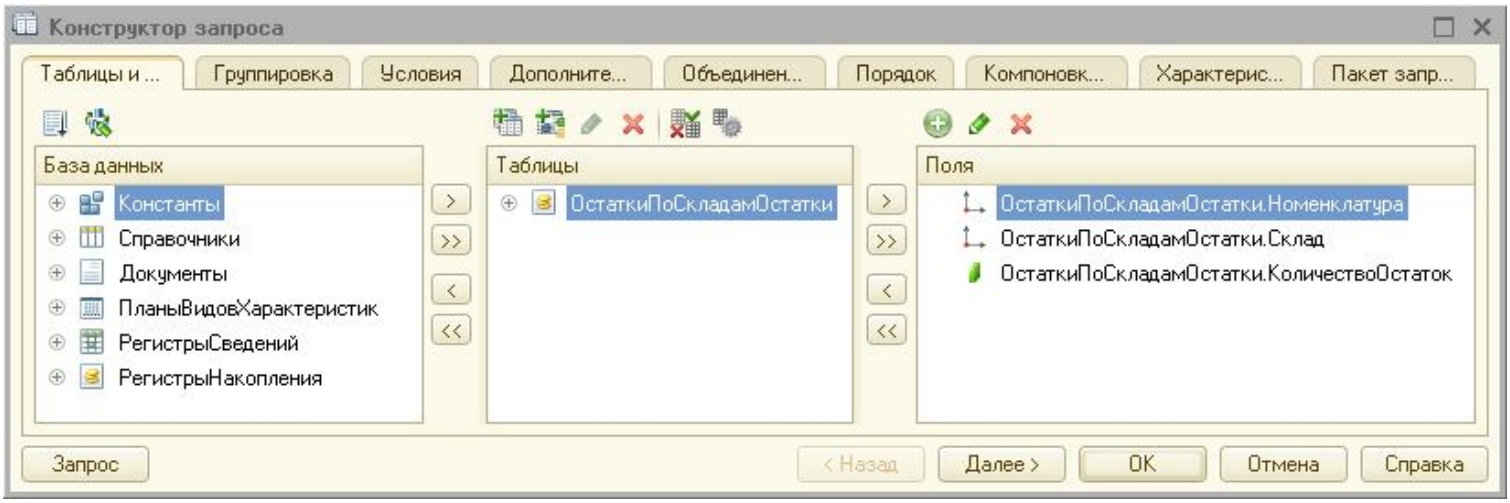


Построим отчет, наглядно отображающий где какие товары лежат:

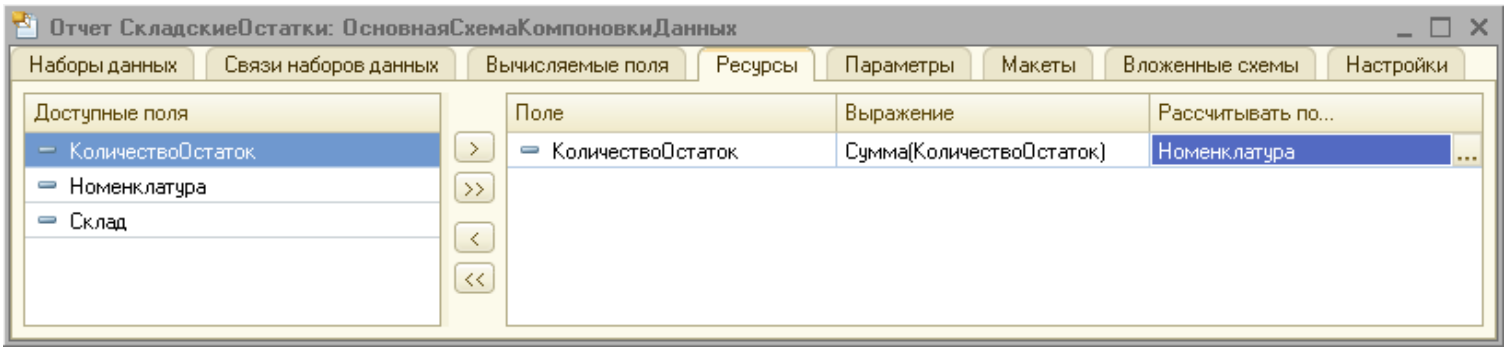
Отчет «Складские остатки»

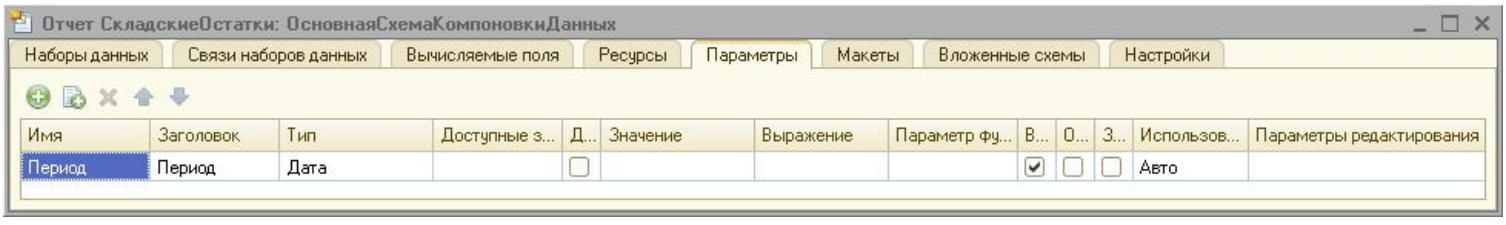
Свойство	Значение
Имя	СкладскиеОстатки
Синоним	Складские остатки

Откроем СКД:
Добавим «Набор данных-запрос» и откроем «Конструктор запроса»:

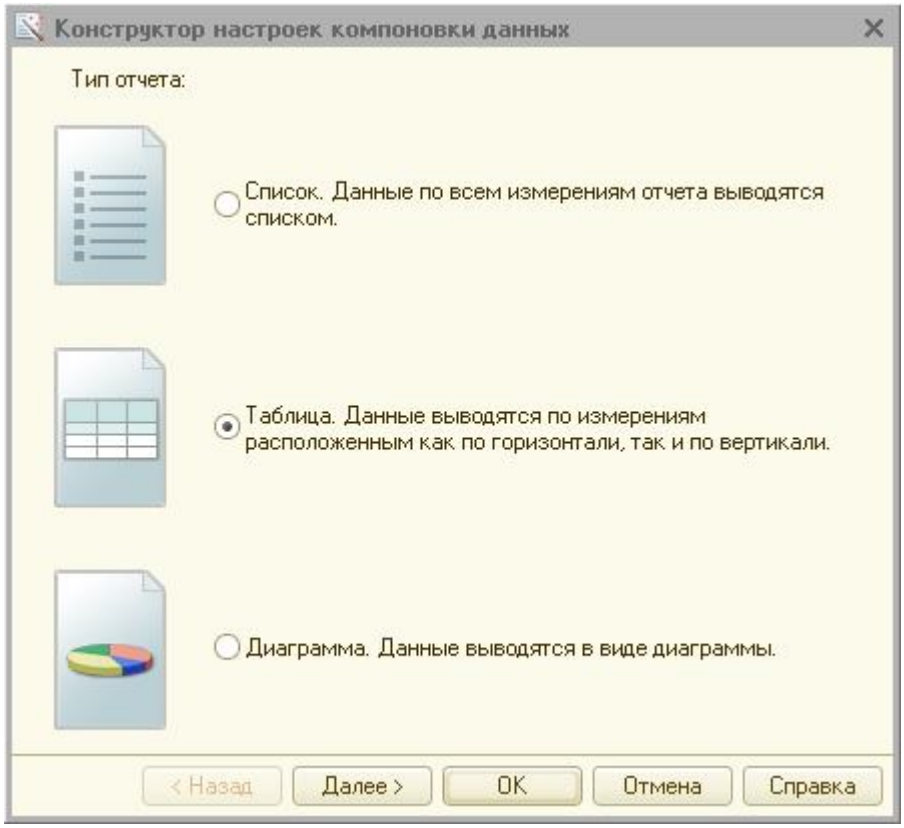


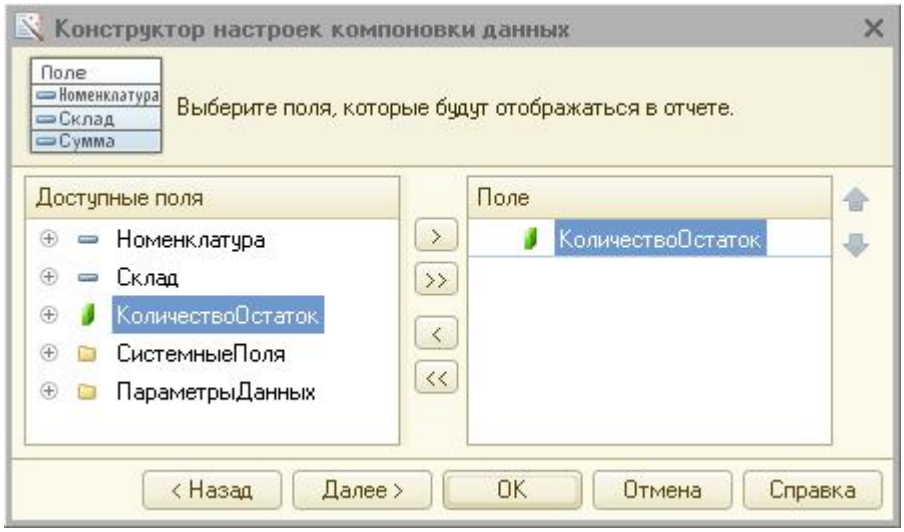
В СКД:

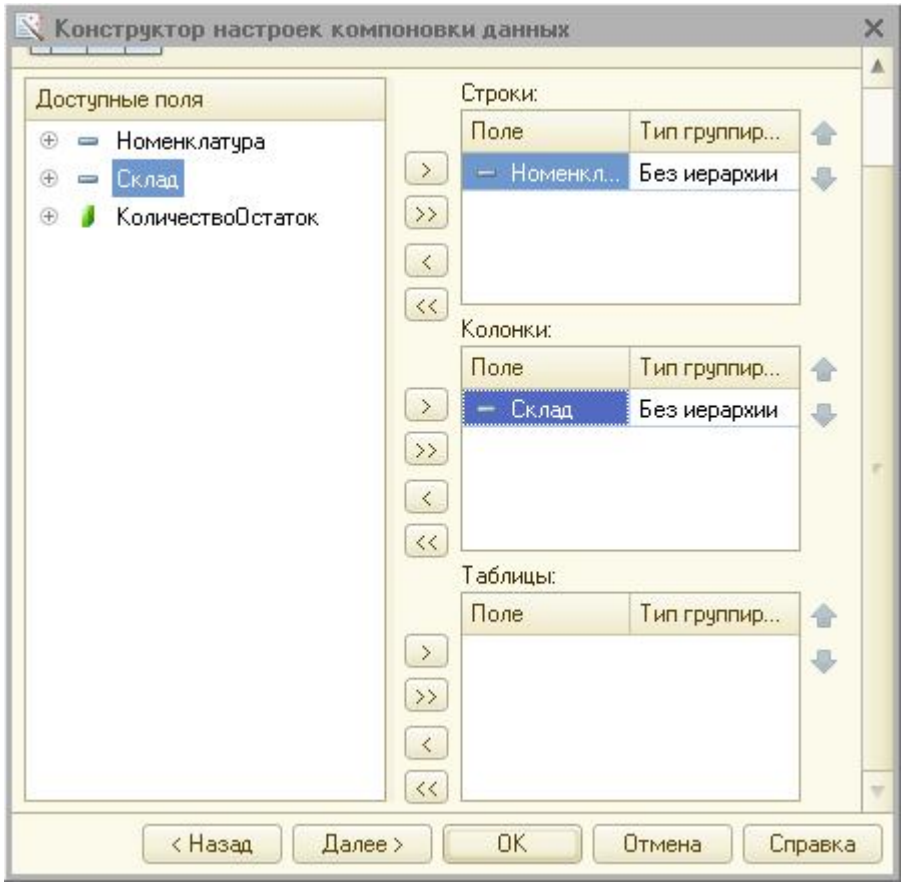




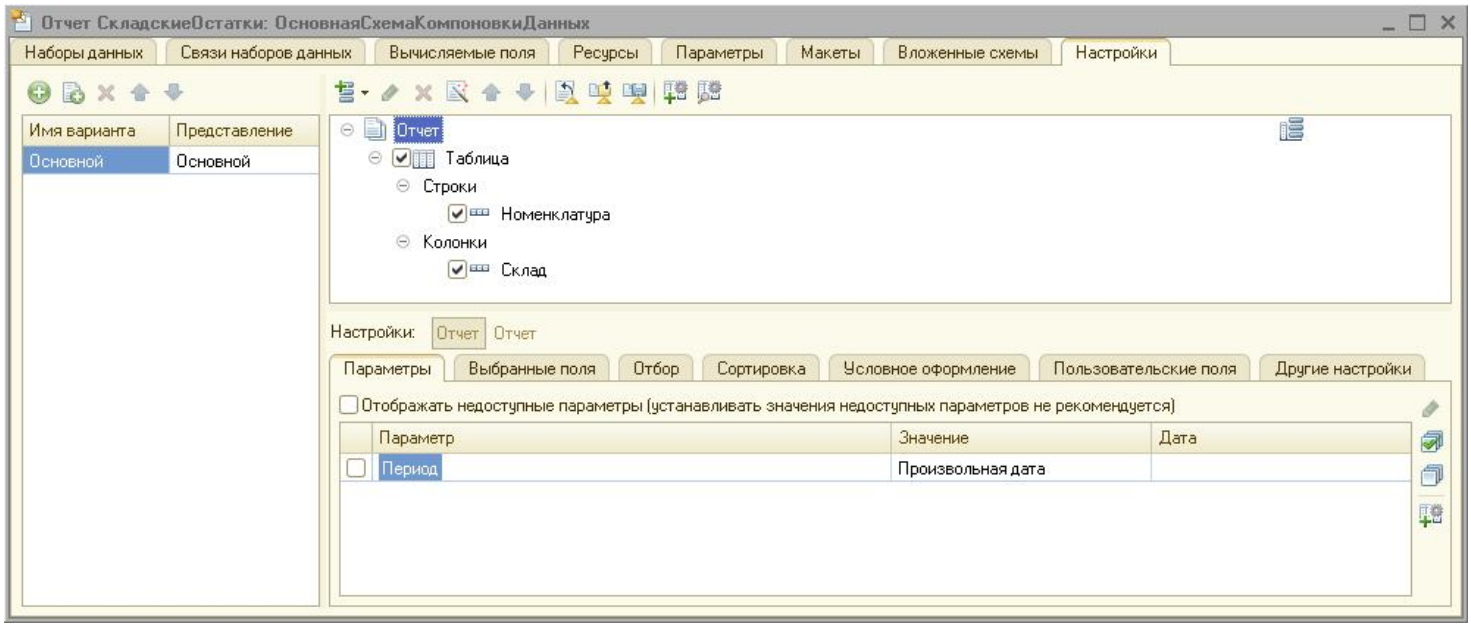
На закладке «Настройки» сформируем основной вариант отчета (конструктором):

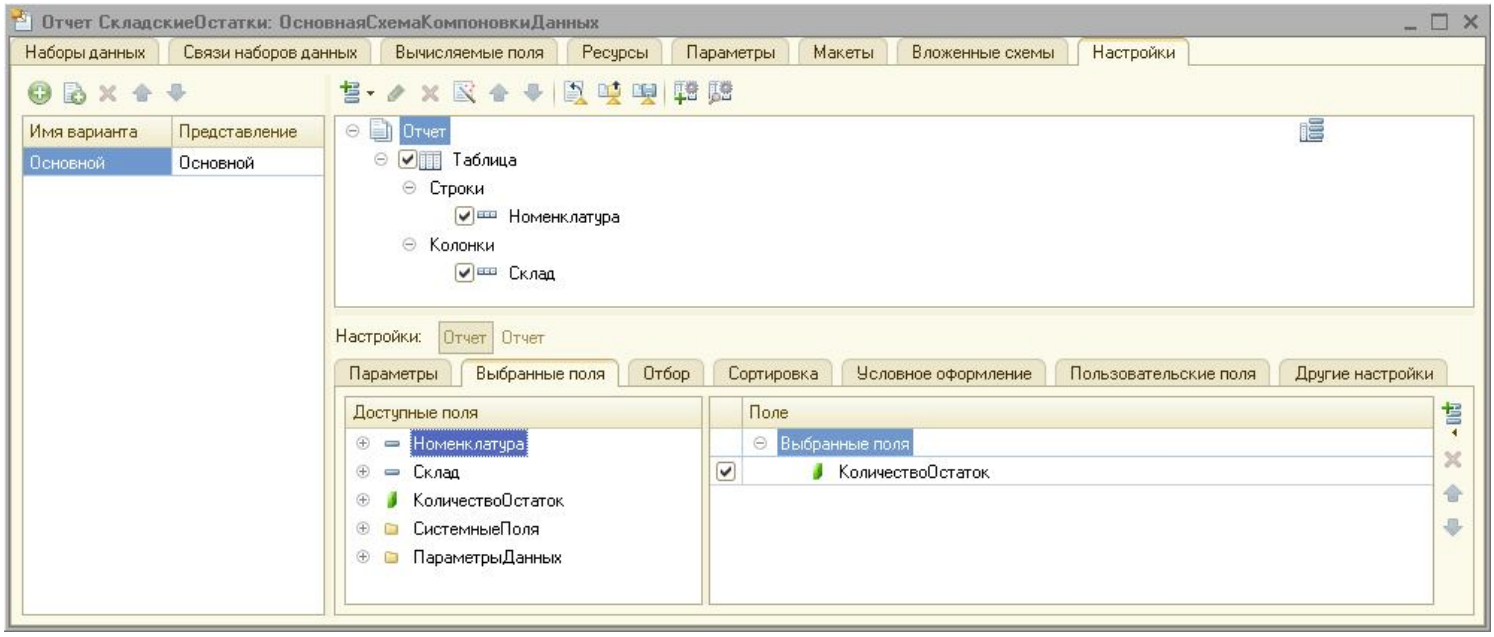

Конструктор настроек компоновки данных
Тип отчета:
Список. Данные по всем измерениям отчета выводятся списком.
Таблица. Данные выводятся по измерениям расположенным как по горизонтали, так и по вертикали.
Диаграмма. Данные выводятся в виде диаграммы.
< Назад
Далее >
OK
Отмена
Справка


Конструктор настроек компоновки данных
Поле
Номенклатура
Склад
Сумма
Выберите поля, которые будут отображаться в отчете.
Доступные поля
Номенклатура
Склад
КоличествоОстаток
СистемныеПоля
ПараметрыДанных
Поле
КоличествоОстаток
< Назад
Далее >
OK
Отмена
Справка

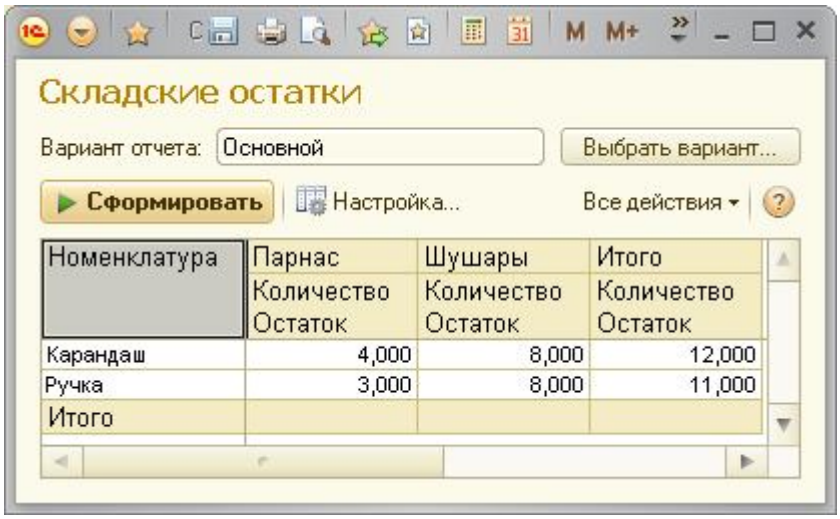


Итого настройки:





В режиме исполнения:



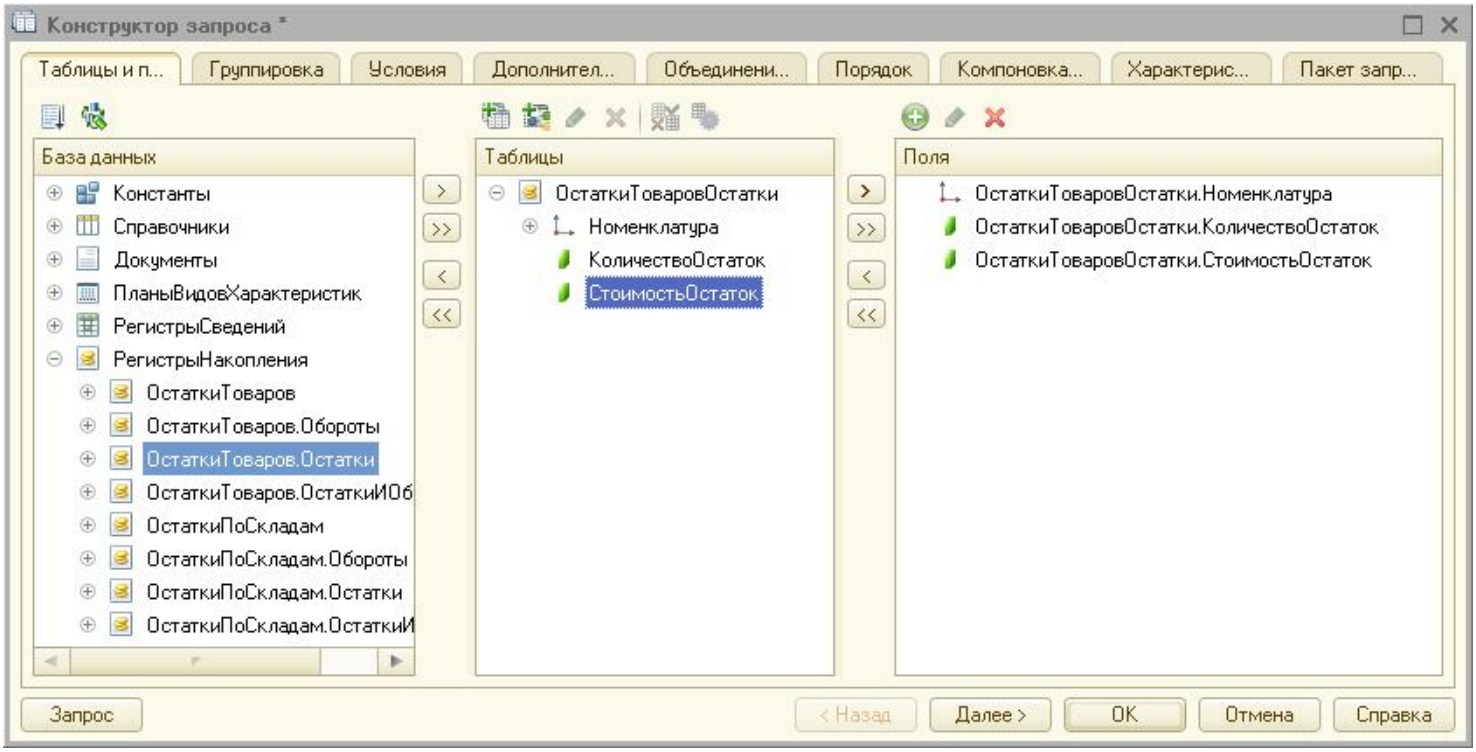
Итоги считаются только по Номенклатуре. По Складу итоги не считаются, т.к. в СКД, на закладке «Ресурсы» у ресурса «Количество»:

Свойство	Значение
Рассчитывать по	Номенклатура

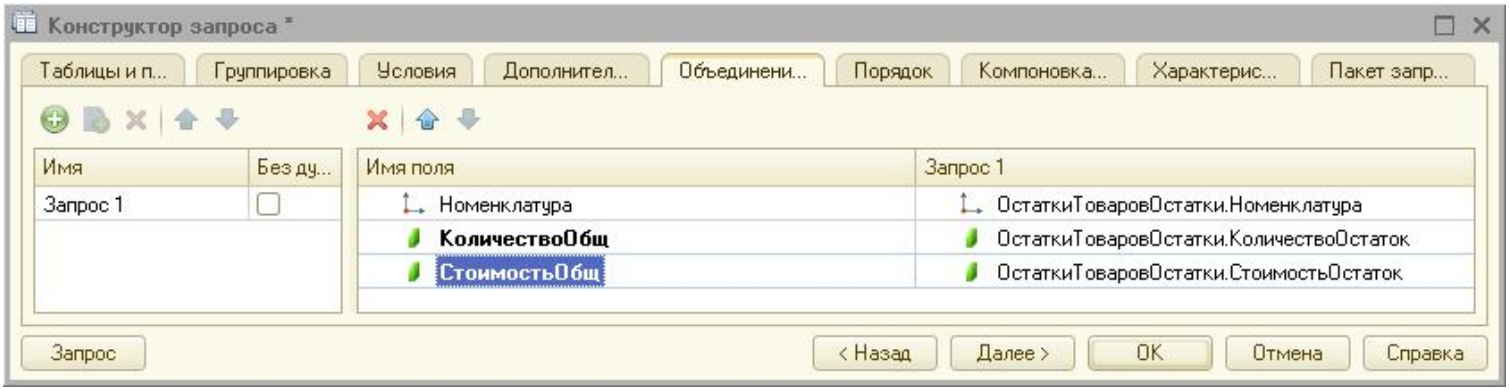
Надо выводить по складу не только количество, но и стоимость товара:

В СКД:

Добавим еще один «Набор данных-запрос» и откроем «Конструктор запроса»:



Переименуем поля:

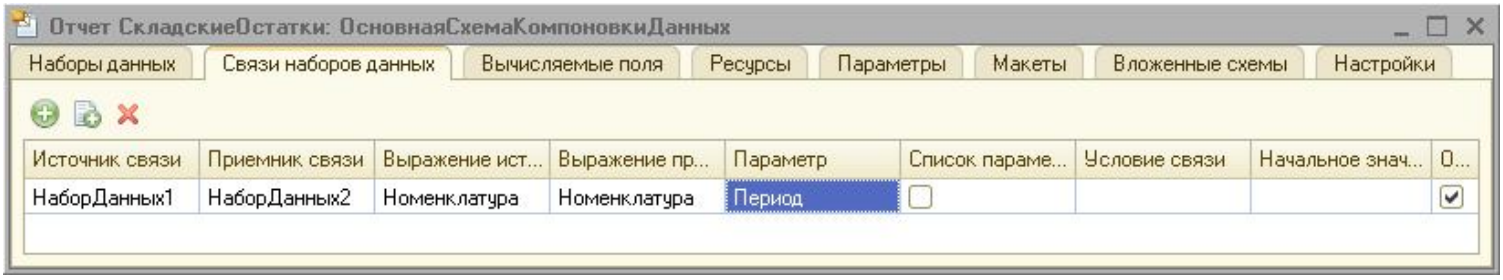


И нажмем «ОК».

Наборы данных свяжем вместе по полю «Номенклатура» - закладка «Связи наборов данных». Здесь всегда указывается только ЛЕВОЕ соединение:

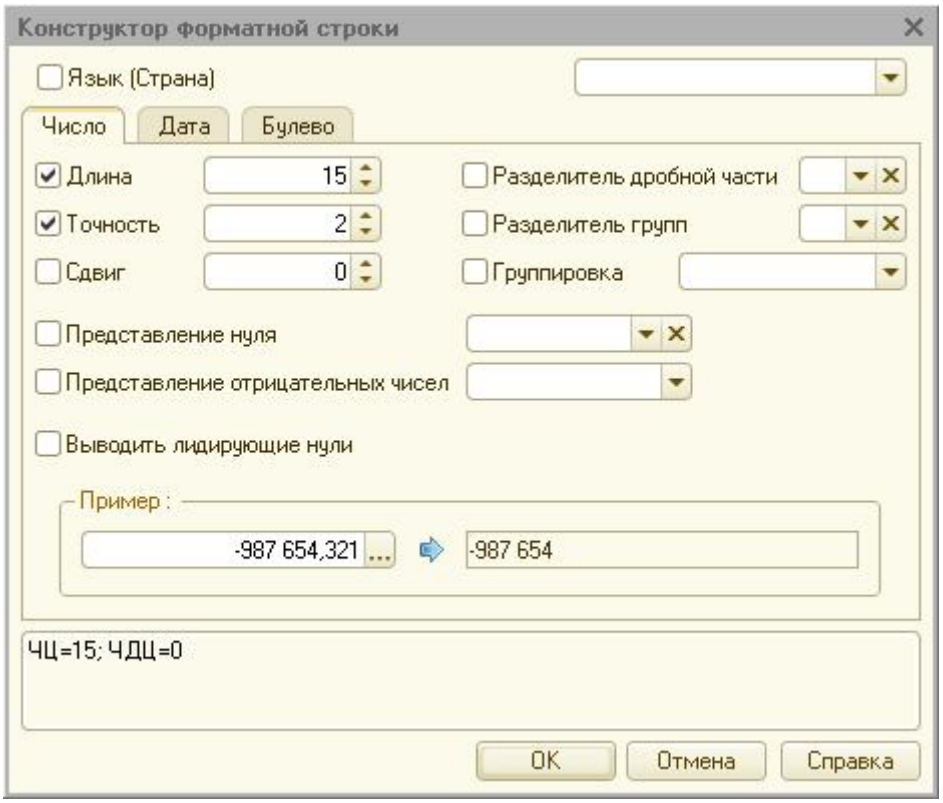
- Источник связи – основная таблица.
- Приемник связи – присоединяемая таблица.

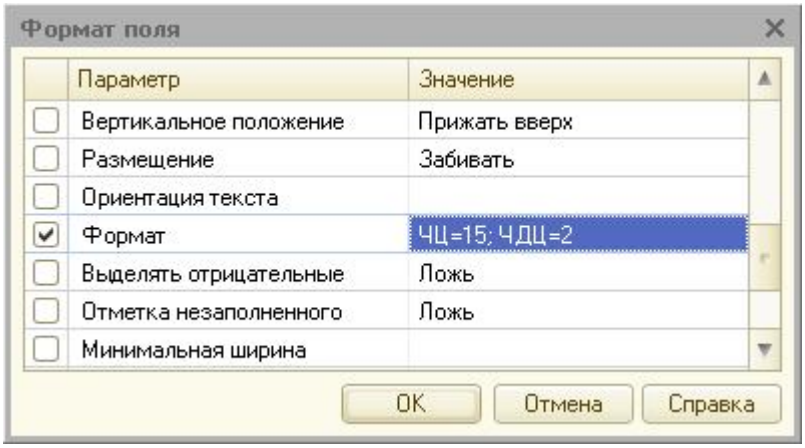
Связываем по Номенклатуре. При этом важно, чтобы остатки были рассчитаны на один и тот же период.



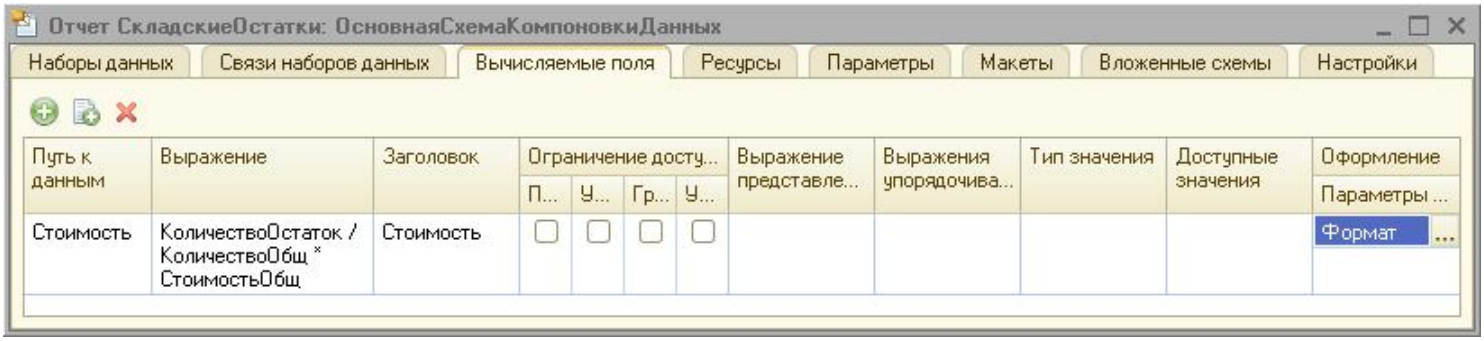
Рассчитаем стоимость склада – закладка «Вычисляемые поля»:

Настроим Оформление этого поля, а именно «Формат», т.к. значение может получиться дробным, а дробная часть в периоде:





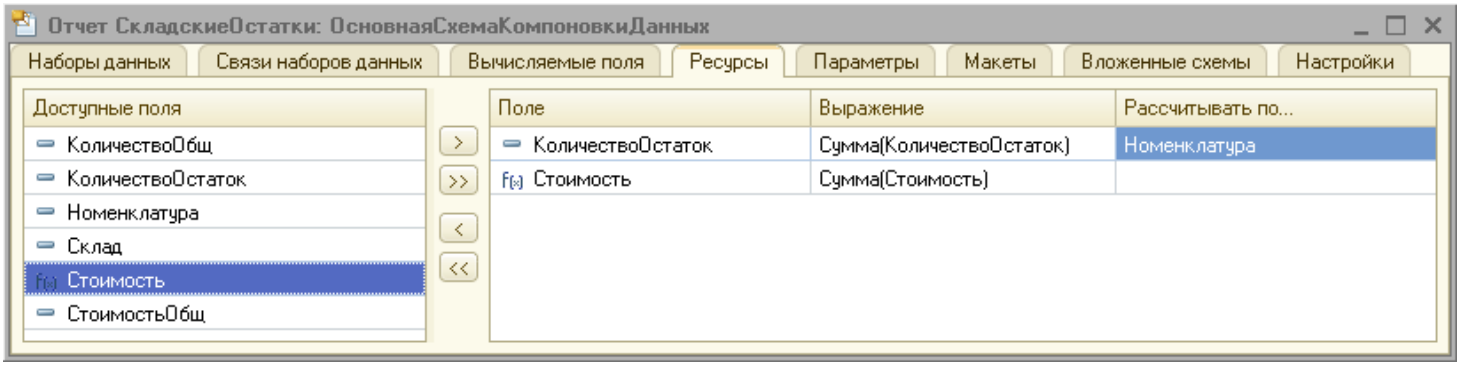
Итого поле:



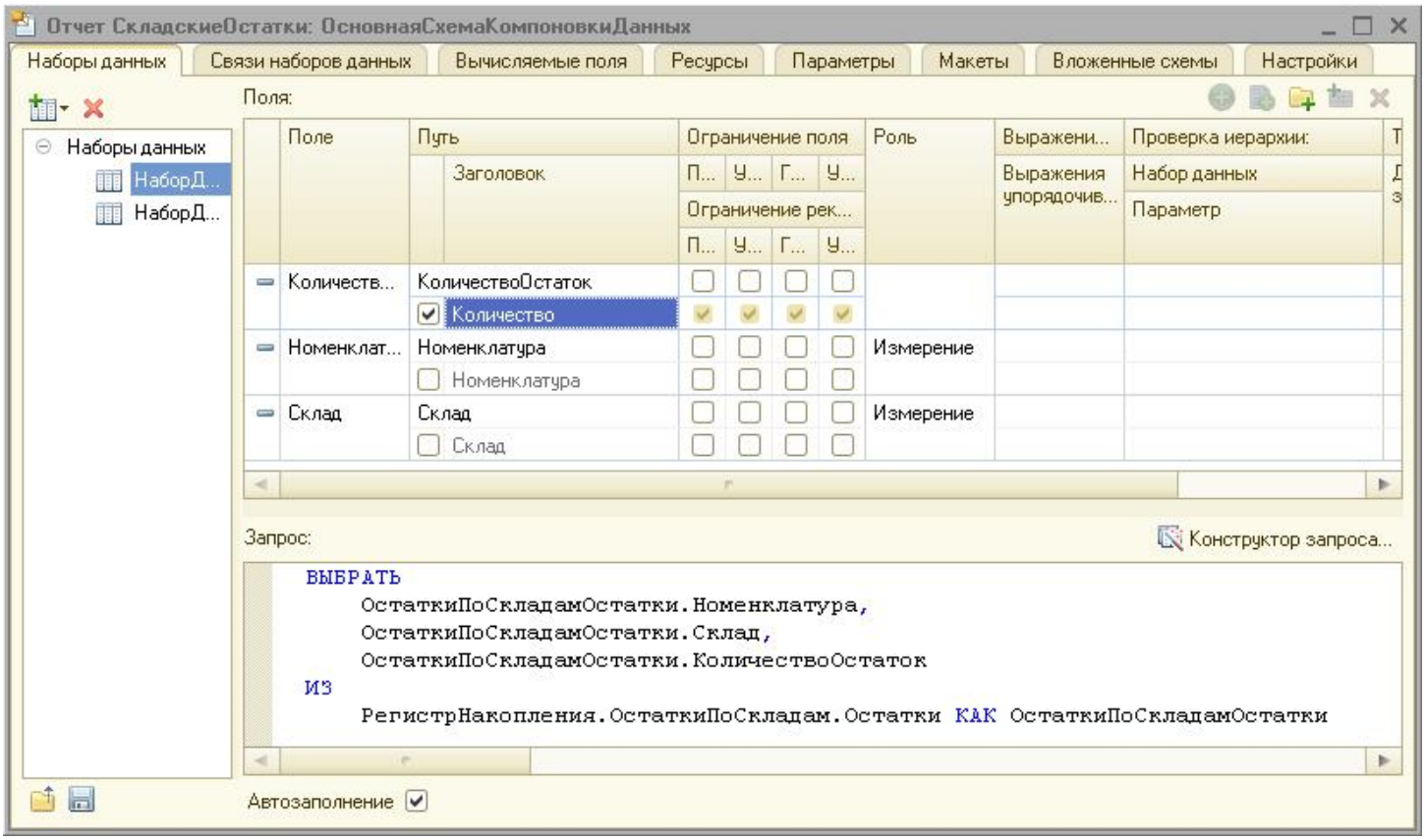
В «Выражение» можно к формуле добавить проверку на ноль:

```
ВЫБОР КОГДА КоличествоОстаток < 0 ТОГДА 0
КОГДА ЕСТЬNULL(КоличествоОбщ,0) = 0 ТОГДА 0
ИНАЧЕ КоличествоОстаток / КоличествоОбщ * СтоимостьОбщ
КОНЕЦ
```

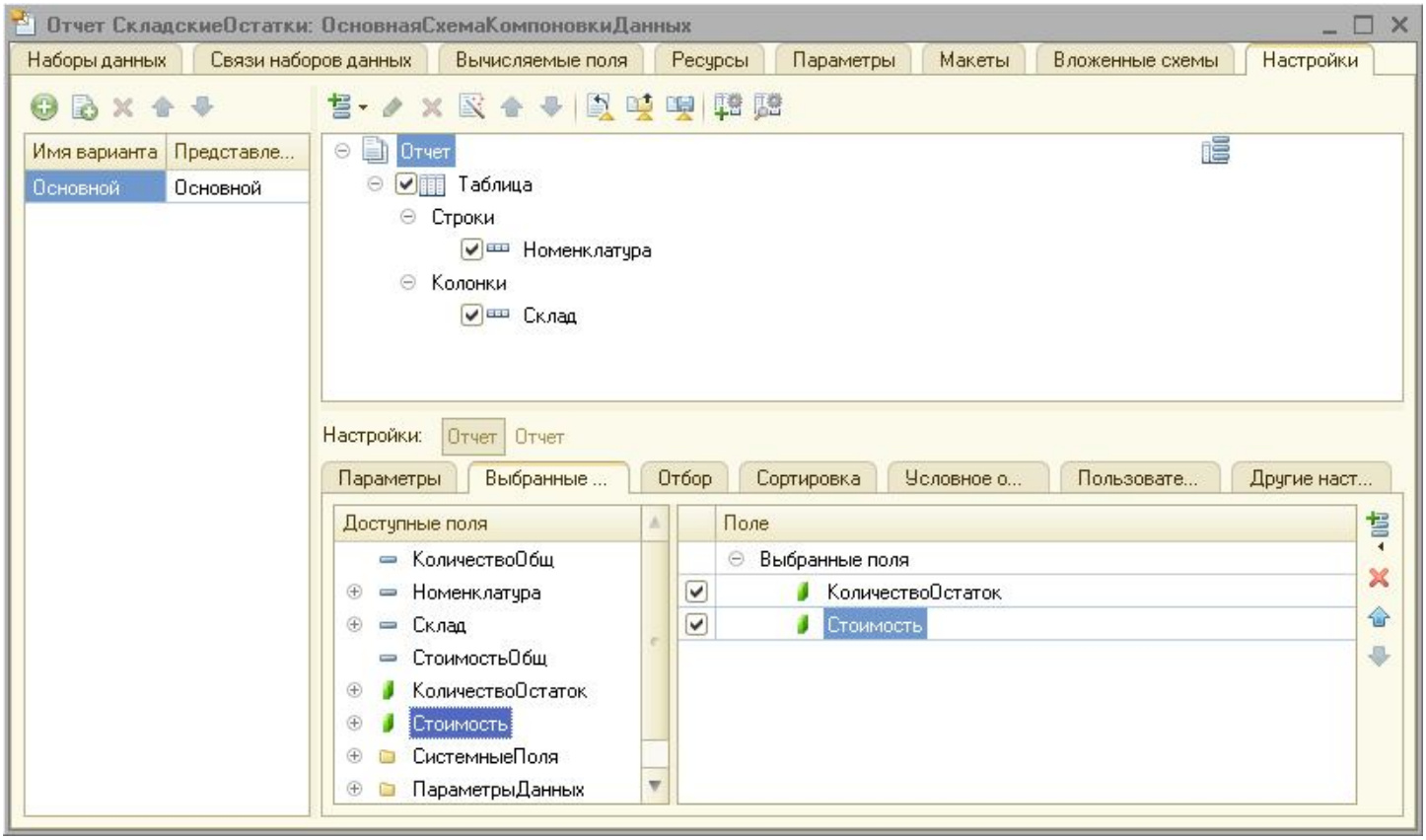
Стоимость – это тоже ресурс. Будет подсчитываться по всем измерениям (Склады и Номенклатура):



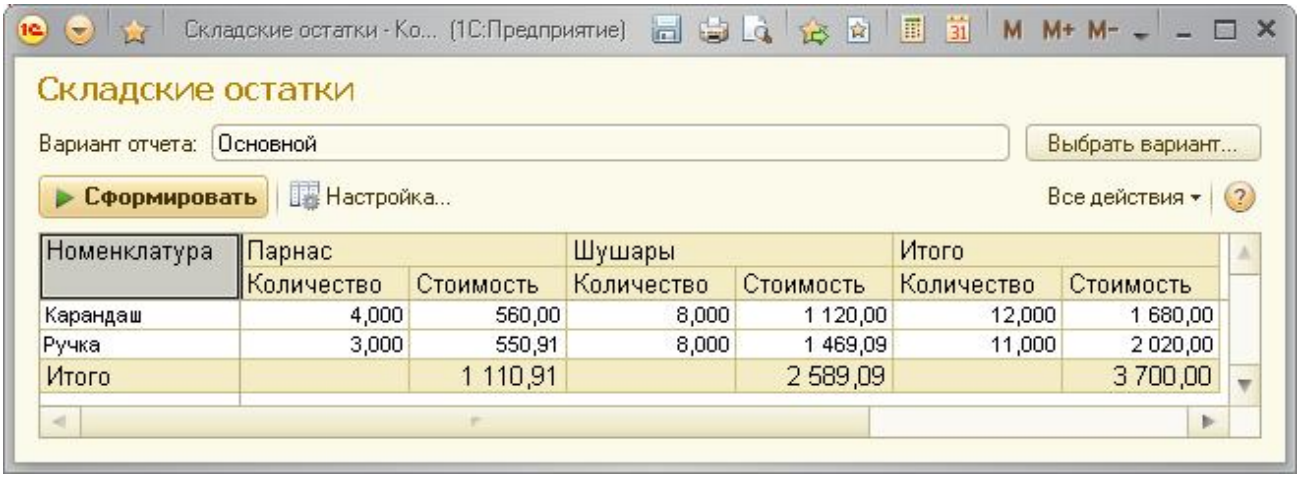
Переименуем поле «КоличествоОстаток» в «Количество»:



На закладке «Настройки» поместим «Стоимость» в «Выбранные поля», т.е. отобразим:



В режиме исполнения:



Номенклатура	Парнас		Шушары		Итого	
	Количество	Стоимость	Количество	Стоимость	Количество	Стоимость
Карандаш	4,000	560,00	8,000	1 120,00	12,000	1 680,00
Ручка	3,000	550,91	8,000	1 469,09	11,000	2 020,00
Итого		1 110,91		2 589,09		3 700,00

Реализуем продажу товара с определенного склада с контролем отрицательных остатков:

При проведении документа «Расходная» будем делать проверку, хватает ли на конкретном складе определенного товара.

Перейдем в режим исполнения и откроем «Консоль запросов». Покажем, как использовать запрос наиболее корректно:

Обратимся к документу и выберем поля табличной части:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 00	2	Карандаш	120	2	240
Продажа товаров 00	4	Ручка	151	2	302
Продажа товаров 00	3	Доставка	500	0	500
Продажа товаров 00	1	Ручка	151	1	151
Продажа товаров 00	2	Карандаш	120	2	240
Продажа товаров 00	3	Доставка	500	0	500
Продажа товаров 00	4	Ручка	151	2	302
Продажа товаров 00	1	Ручка	151	1	151

Будем продавать товары по первому документу:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 00	1	Ручка	151	1	151
Продажа товаров 00	2	Карандаш	120	2	240
Продажа товаров 00	3	Доставка	500	0	500
Продажа товаров 00	4	Ручка	151	2	302

Для расчета себестоимости товаров обратимся к регистру накопления «Остатки товаров»:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
	ДокТЧ.Номенклатура = Остатки.Номенклатура
ГДЕ
	Ссылка = &Ссылка
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма	Номенклатура1	КоличествоОстаток	СтоимостьОстаток
Продажа товаров 00	1	Ручка	151	1	151	Ручка	11	2 020
Продажа товаров 00	2	Карандаш	120	2	240	Карандаш	12	1 680
Продажа товаров 00	3	Доставка	500	0	500	NULL	NULL	NULL
Продажа товаров 00	4	Ручка	151	2	302	Ручка	11	2 020

Привяжем еще одну таблицу – с остатками товаров на складах:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
	ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиПоСкладам.Остатки КАК ОстаткиСКЛ
ПО
	ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
ГДЕ
	Ссылка = &Ссылка
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма	Номенклатура1	КоличествоОста	СтоимостьОстат	Номенклатура2	Склад	КоличествоОста
Продажа товаро	1	Ручка	151	1	151	Ручка	11	2 020	Ручка	Парнас	3
Продажа товаро	1	Ручка	151	1	151	Ручка	11	2 020	Ручка	Шушары	8
Продажа товаро	2	Карандаш	120	2	240	Карандаш	12	1 680	Карандаш	Парнас	4
Продажа товаро	2	Карандаш	120	2	240	Карандаш	12	1 680	Карандаш	Шушары	8
Продажа товаро	3	Доставка	500	0	500	NULL	NULL	NULL	NULL	NULL	NULL
Продажа товаро	4	Ручка	151	2	302	Ручка	11	2 020	Ручка	Парнас	3
Продажа товаро	4	Ручка	151	2	302	Ручка	11	2 020	Ручка	Шушары	8

Т.к. склады разные, то система задублировала данные.

Получим остатки по конкретному складу – установим условие отбора в параметрах виртуальной таблицы:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
	ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиПоСкладам.Остатки( , Склад = &Склад) КАК ОстаткиСКЛ
ПО
	ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
ГДЕ
	Ссылка = &Ссылка
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма	Номенклатура1	КоличествоОста	СтоимостьОстат	Номенклатура2	Склад	КоличествоОста
Продажа товаро	1	Ручка	151	1	151	Ручка	11	2 020	Ручка	Парнас	3
Продажа товаро	2	Карандаш	120	2	240	Карандаш	12	1 680	Карандаш	Парнас	4
Продажа товаро	3	Доставка	500	0	500	NULL	NULL	NULL	NULL	NULL	NULL
Продажа товаро	4	Ручка	151	2	302	Ручка	11	2 020	Ручка	Парнас	3

Выберем только нужные нам поля:

```
ВЫБРАТЬ
      ДокТЧ.Номенклатура,
      ДокТЧ.Количество,
      Остатки.КоличествоОстаток,
      Остатки.СтоимостьОстаток,
      ОстаткиСКЛ.КоличествоОстаток КАК ОстатокНаСкладе
ИЗ
      Документ.Расходная.Товары КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
      РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО
      ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
      РегистрНакопления.ОстаткиПоСкладам.Остатки( , Склад = &Склад) КАК ОстаткиСКЛ
ПО
      ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
ГДЕ
      Ссылка = &Ссылка
```

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
Ручка	1	11	2 020	3
Карандаш	2	12	1 680	4
Доставка	0	NULL	NULL	NULL
Ручка	2	11	2 020	3

Поля «КоличествоОстаток» и «КоличествоСтоимость» будут использоваться для расчета себестоимости.
Поле «ОстатокНаСкладе» - для проверки наличия нужного количества товара.

Но данный вариант не совсем рационален:

Данные виртуальных таблиц всегда максимально группируются. Т.к. мы присоединяем эти таблицы к несгруппированной табличной части документа, то они дублируются, и приходится повторно вручную (через запрос) их группировать.

Сгруппируем данные табличной части документа до соединения:

Используем механизм «Пакетных» запросов. В виртуальной таблице «Остатки по складам» мы установили отбор по складу. Т.е. сейчас система высчитает остатки всех товаров по этому складу, а уже при соединении таблиц отбросит лишние товары. Это не рационально.
Надо перед соединением таблиц не только сгруппировать ТЧ документа, но и использовать ее данные как фильтр.

Создадим пакетный запрос:
Первый запрос получит таблицу:

Номенклатура	Количество
Ручка	1
Карандаш	2
Доставка	0
Ручка	2

Но в сгруппированном виде.

Таблица будет являться источником для соединения полей:

КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
11	2 020	3
12	1 680	4
NULL	NULL	NULL
11	2 020	3

И колонка «Номенклатура» будет являться фильтром для расчета виртуальных таблиц.

Верный запрос:

Выберем из ТЧ первого документа элементы номенклатуры, не являющиеся услугами:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 000	1	Ручка	151	1	151
Продажа товаров 000	2	Карандаш	120	2	240
Продажа товаров 000	4	Ручка	151	2	302

Выберем только необходимые поля и сгруппируем их:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
```

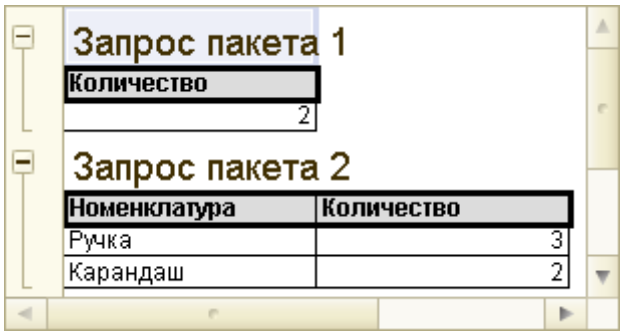
Номенклатура	Количество
Ручка	3
Карандаш	2

Эту таблицу поместим во временные, чтобы потом к ней обращаться:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
```

Через «;» опишем второй запрос, который может в качестве источника использовать таблицу «ДокТЧ»:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
	*
ИЗ
	ДокТЧ КАК ДокТЧ
```



К полученной таблице присоединяем остатки из регистров:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
	*
ИЗ
	ДокТЧ КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиПоСкладам.Остатки КАК ОстаткиСКЛ
ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```

Запрос пакета 1

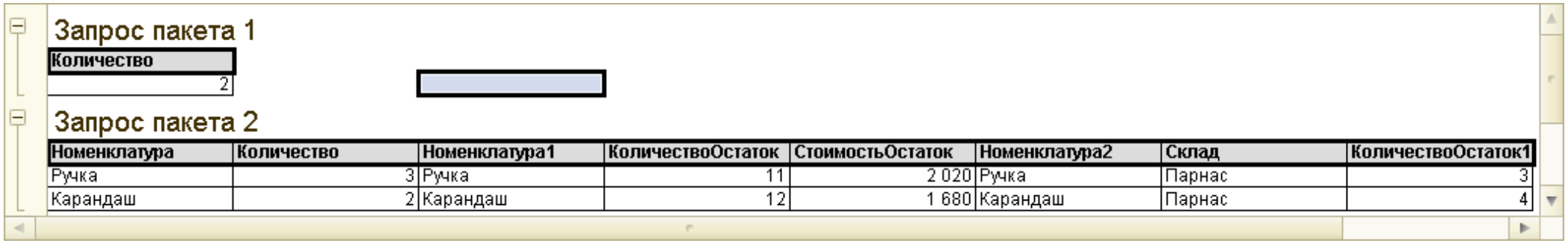
Количество
2

Запрос пакета 2

Номенклатура	Количество	Номенклатура1	КоличествоОстаток	СтоимостьОстаток	Номенклатура2	Склад	КоличествоОстаток1
Ручка	3	Ручка	11	2 020	Ручка	Парнас	3
Ручка	3	Ручка	11	2 020	Ручка	Шушары	8
Карандаш	2	Карандаш	12	1 680	Карандаш	Парнас	4
Карандаш	2	Карандаш	12	1 680	Карандаш	Шушары	8

Данные дублируются из-за того, что есть несколько складов. Поставим ограничение по складу:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
	*
ИЗ
	ДокТЧ КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиПоСкладам.Остатки( , Склад = &Склад) КАК ОстаткиСКЛ
ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```



Запрос пакета 1

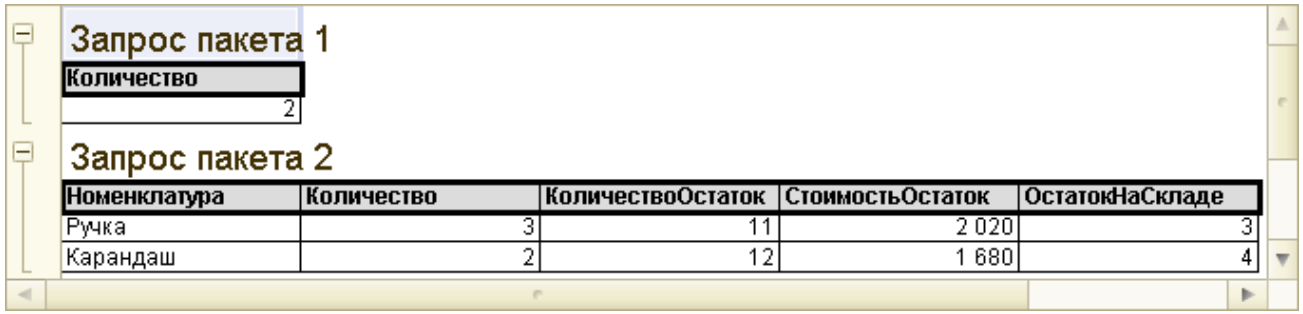
Количество
2

Запрос пакета 2

Номенклатура	Количество	Номенклатура1	КоличествоОстаток	СтоимостьОстаток	Номенклатура2	Склад	КоличествоОстаток1
Ручка	3	Ручка	11	2 020	Ручка	Парнас	3
Карандаш	2	Карандаш	12	1 680	Карандаш	Парнас	4

Выберем только необходимые поля:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
	ДокТЧ.Номенклатура,
	ДокТЧ.Количество,
	Остатки.КоличествоОстаток,
	Остатки.СтоимостьОстаток,
	ОстаткиСКЛ.КоличествоОстаток КАК ОстатокНаСкладе
ИЗ
	ДокТЧ КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиПоСкладам.Остатки( , Склад = &Склад) КАК ОстаткиСКЛ
ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```



Запрос пакета 1

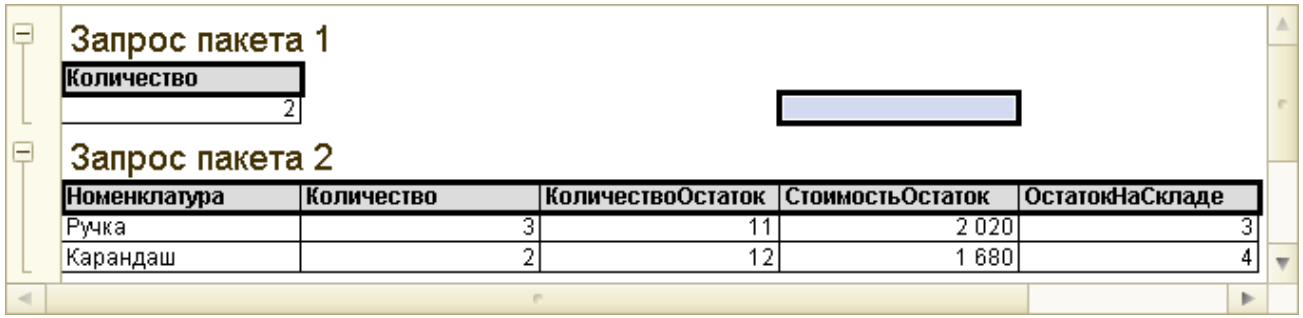
Количество
2

Запрос пакета 2

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
Ручка	3	11	2 020	3
Карандаш	2	12	1 680	4

Оптимизируем запрос. Кроме условия по складу, добавим условие по номенклатуре – отбираем только номенклатуру, которая входит в состав таблицы «ДокТЧ»:

```
ВЫБРАТЬ
     Номенклатура,
     СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары
ГДЕ
     Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
     ДокТЧ.Номенклатура,
     ДокТЧ.Количество,
     Остатки.КоличествоОстаток,
     Остатки.СтоимостьОстаток,
     ОстаткиСКЛ.КоличествоОстаток КАК ОстатокНаСкладе
ИЗ
     ДокТЧ КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
     РегистрНакопления.ОстаткиТоваров.Остатки КАК Остатки
ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
     РегистрНакопления.ОстаткиПоСкладам.Остатки( , Склад = &Склад
          И Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```



Запрос пакета 1

Количество
2

Запрос пакета 2

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
Ручка	3	11	2 020	3
Карандаш	2	12	1 680	4

Запрос формируется намного быстрее, т.к. рассчитываются остатки только по конкретным элементам номенклатуры.

Поставим такое же условие и по первой таблице:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
	ДокТЧ.Номенклатура,
	ДокТЧ.Количество,
	Остатки.КоличествоОстаток,
	Остатки.СтоимостьОстаток,
	ОстаткиСКЛ.КоличествоОстаток КАК ОстатокНаСкладе
ИЗ
	ДокТЧ КАК ДокТЧ
ЛЕВОЕ СОЕДИНЕНИЕ
	РегистрНакопления.ОстаткиТоваров.Остатки( , Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК ДокТЧ))
КАК Остатки
ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
```

```
ЛЕВОЕ СОЕДИНЕНИЕ
     РегистрНакопления.ОстаткиПоСкладам.Остатки( , Склад = &Склад
          И Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```

Запрос пакета 1

Количество
2

Запрос пакета 2

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
Ручка	3	11	2 020	3
Карандаш	2	12	1 680	4

Добавим условие на «МоментВремени» и присоединяемые поля опишем с функцией «ЕСТЬNULL», иначе они могут не присоединиться:

```
ВЫБРАТЬ
     Номенклатура,
     СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары
ГДЕ
     Ссылка = &Ссылка И НЕ Номенклатура.Услуга
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
     ДокТЧ.Номенклатура,
     ДокТЧ.Количество,
     ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
     ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
     ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
ИЗ
     ДокТЧ КАК ДокТЧ
```

```
ЛЕВОЕ СОЕДИНЕНИЕ
     РегистрНакопления.ОстаткиТоваров.Остатки(&МоментВремени, Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК
ДокТЧ)) КАК Остатки
ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
ЛЕВОЕ СОЕДИНЕНИЕ
     РегистрНакопления.ОстаткиПоСкладам.Остатки(&МоментВремени, Склад = &Склад
          И Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```

Запрос пакета 1

Количество
2

Запрос пакета 2

Номенклатура	Количество	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
Ручка	3	11	2 020	3
Карандаш	2	12	1 680	4

Теперь перенесем запрос в документ «Расходная»:

При копировании запроса не вставляются знаки «Перенос строки». Их можно добавить:
«Текст – Блок – Добавить перенос строки»

Если надо сохранять временную таблицу, то используется «Менеджер временных таблиц» - объект, хранящий временные таблицы. Данный объект присваивается запросу. После этого можно выполнять другие действия, не связанные с этим запросом. Можно передать в другой запрос, другую процедуру или функцию переменную менеджера временных таблиц. Если другому запросу присвоить менеджер, то там можно использовать таблицу предыдущего запроса.

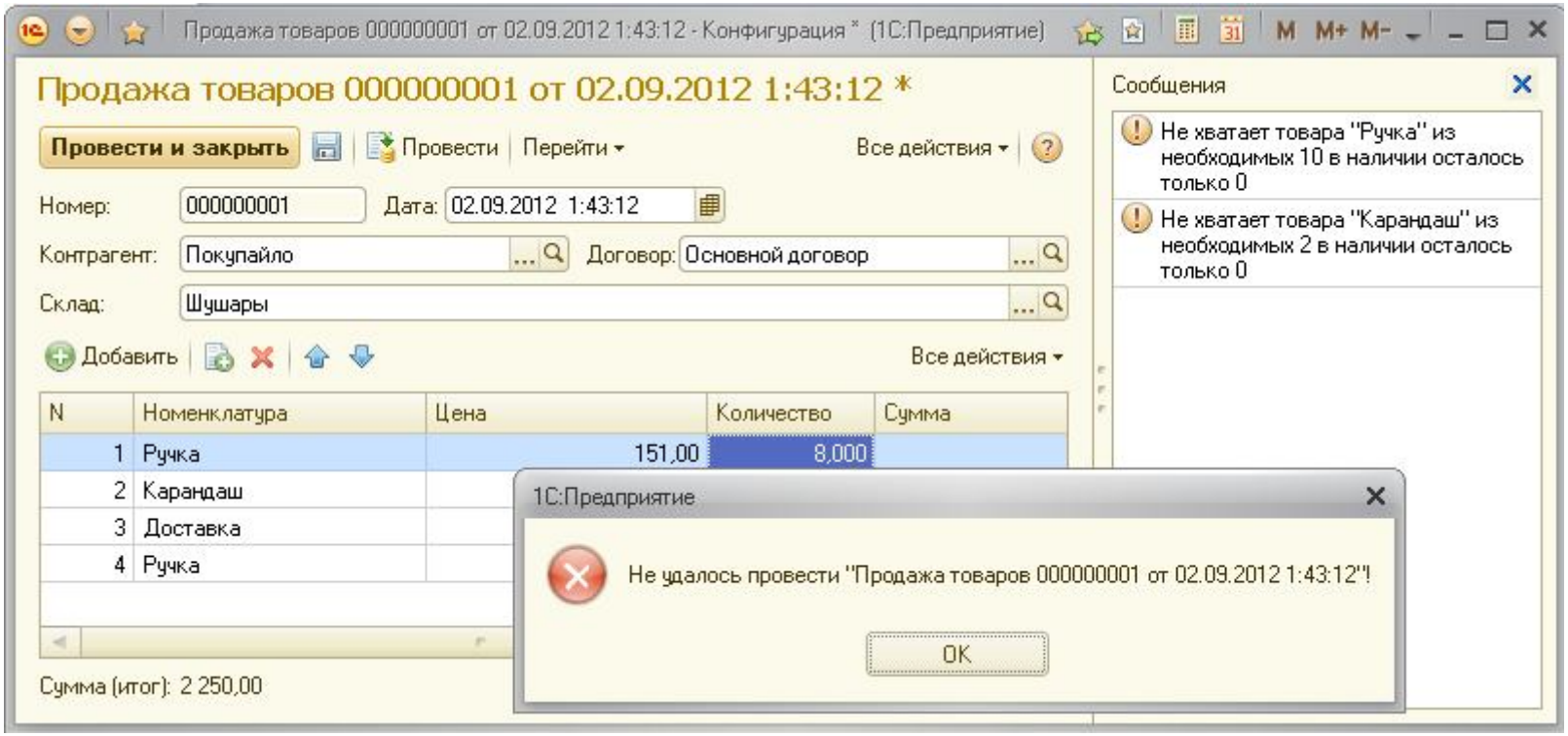
Итого код процедуры проведения:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

     Движения.ОстаткиТоваров.Записывать = Истина;
     Движения.ОстаткиПоСкладам.Записывать = Истина;
```

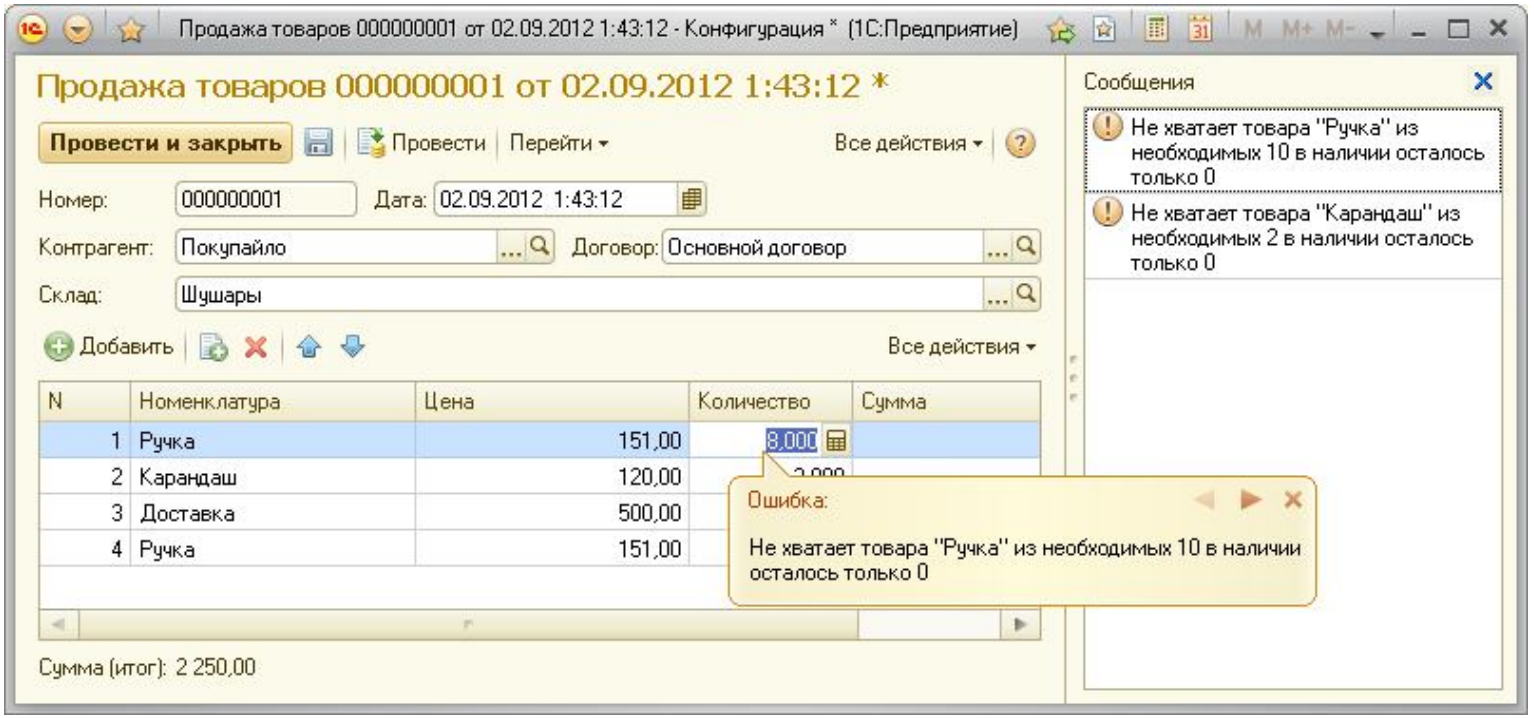
```
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    Номенклатура,
    |    СУММА(Количество) КАК Количество,
    |    МИНИМУМ(НомерСтроки) КАК НомерСтроки
    |ПОМЕСТИТЬ ДокТЧ
    |ИЗ
    |    Документ.Расходная.Товары
    |ГДЕ
    |    Ссылка = &Ссылка И НЕ Номенклатура.Услуга
    |СГРУППИРОВАТЬ ПО Номенклатура
    |;
    |ВЫБРАТЬ
    |    ДокТЧ.НомерСтроки,
    |    ДокТЧ.Номенклатура,
    |    ДокТЧ.Количество,
    |    ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
    |    ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
    |    ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
    |ИЗ
    |    ДокТЧ КАК ДокТЧ
    |ЛЕВОЕ СОЕДИНЕНИЕ
    |    РегистрНакопления.ОстаткиТоваров.Остатки(&МоментВремени, Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ
КАК ДокТЧ)) КАК Остатки
    |ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
    |ЛЕВОЕ СОЕДИНЕНИЕ
    |    РегистрНакопления.ОстаткиПоСкладам.Остатки(&МоментВремени, Склад = &Склад
    |        И Номенклатура В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
    |ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура";

    Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
    Запрос.УстановитьПараметр("Ссылка", Ссылка);
    Запрос.УстановитьПараметр("Склад", Склад);
```

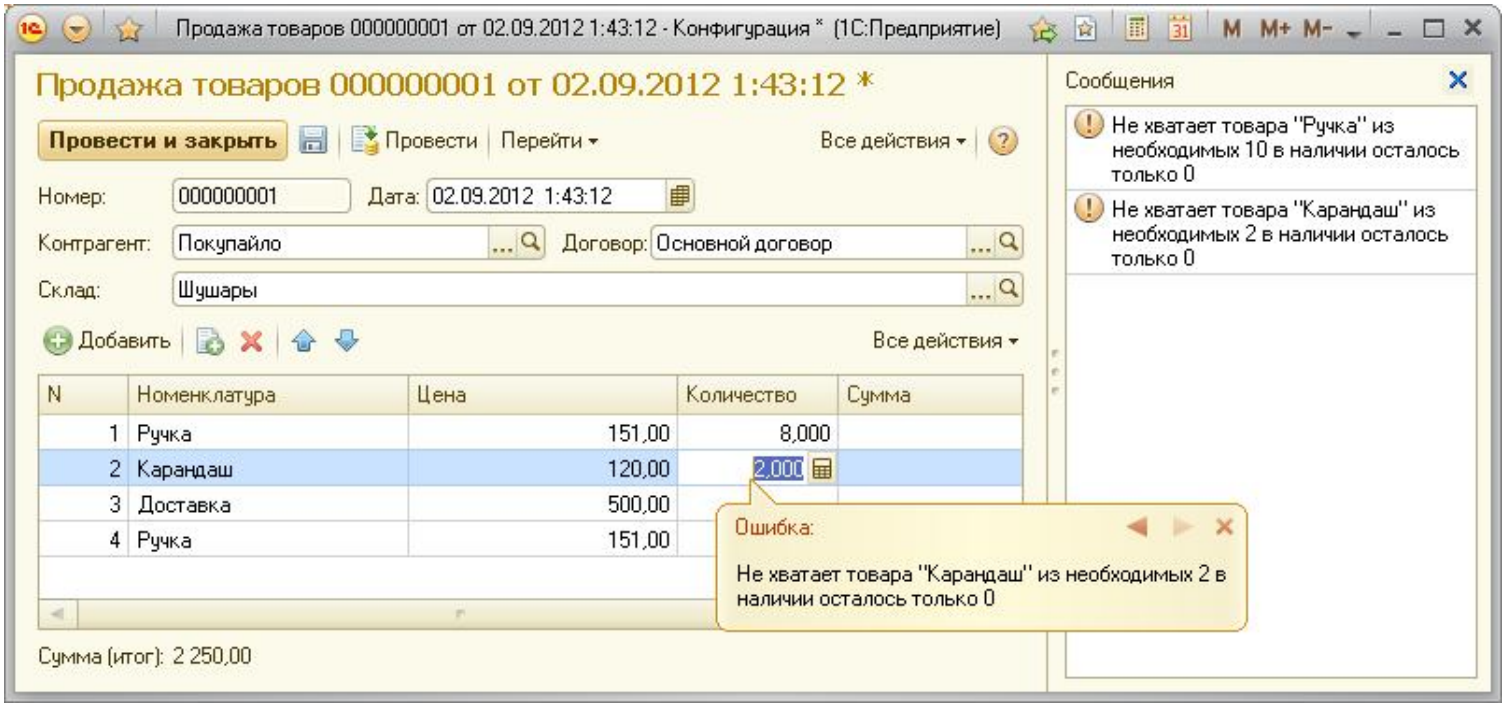
```
    Результат = Запрос.Выполнить();

    Выборка = Результат.Выбрать();

    Пока Выборка.Следующий() Цикл
        Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура + """ из необходимых "
+ Выборка.Количество +
            " в наличии осталось только " + Выборка.ОстатокНаСкладе;
            Сообщение.Поле = "Товары[" + (Выборка.НомерСтроки - 1) + "].Количество";
            Сообщение.УстановитьДанные(ЭтотОбъект);
            Сообщение.Сообщить();
            Отказ = Истина;
            Движения.ОстаткиТоваров.Записывать = Ложь;
            Движения.ОстаткиПоСкладам.Записывать = Ложь;
        КонецЕсли;

        Если Отказ Тогда
            Продолжить;
        КонецЕсли;

        Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
        Движение.Период = Дата;
        Движение.Номенклатура = Выборка.Номенклатура;
        Движение.Количество = Выборка.Количество;
        Движение.Стоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СтоимостьОстаток;

        Движение = Движения.ОстаткиПоСкладам.ДобавитьРасход();
        Движение.Период = Дата;
        Движение.Номенклатура = Выборка.Номенклатура;
        Движение.Количество = Выборка.Количество;
        Движение.Склад = Склад;

    КонецЦикла;
КонецПроцедуры
```

В режиме исполнения:

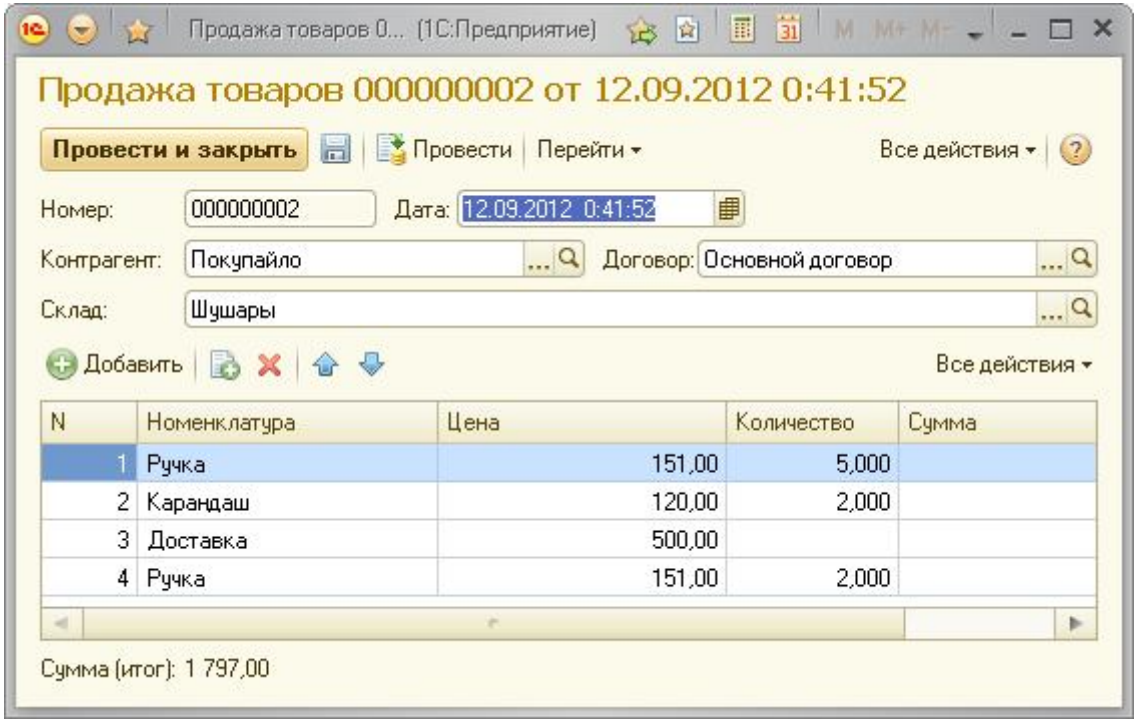


Нажимаем «ОК».


Продажа товаров 000000001 от 02.09.2012 1:43:12 - Конфигурация * (1С:Предприятие)
Продажа товаров 000000001 от 02.09.2012 1:43:12 *
Провести и закрыть
Провести
Перейти
Все действия
Номер: 000000001
Дата: 02.09.2012 1:43:12
Контрагент: Покупайло
Договор: Основной договор
Склад: Шушары
Добавить
Все действия
N
Номенклатура
Цена
Количество
Сумма
1 Ручка 151,00 8,000
2 Карандаш 120,00
3 Доставка 500,00
4 Ручка 151,00
Ошибка:
Не хватает товара "Ручка" из необходимых 10 в наличии осталось только 0
Сумма (итог): 2 250,00
Сообщения
Не хватает товара "Ручка" из необходимых 10 в наличии осталось только 0
Не хватает товара "Карандаш" из необходимых 2 в наличии осталось только 0

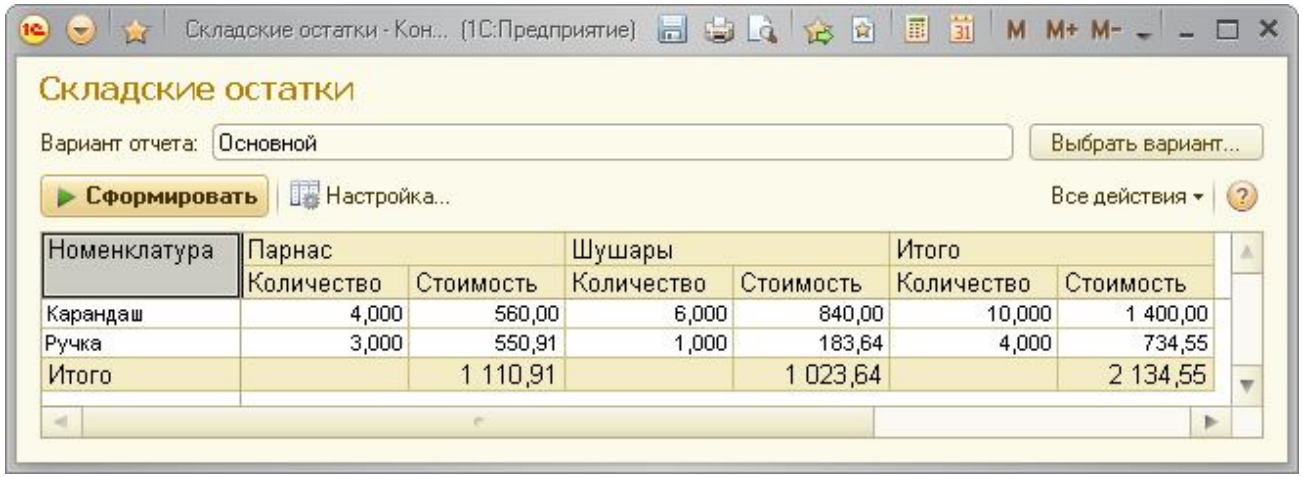

Продажа товаров 000000001 от 02.09.2012 1:43:12 - Конфигурация * (1С:Предприятие)
Продажа товаров 000000001 от 02.09.2012 1:43:12 *
Провести и закрыть
Провести
Перейти
Все действия
Номер: 000000001
Дата: 02.09.2012 1:43:12
Контрагент: Покупайло
Договор: Основной договор
Склад: Шушары
Добавить
Все действия
N
Номенклатура
Цена
Количество
Сумма
1 Ручка 151,00 8,000
2 Карандаш 120,00 2,000
3 Доставка 500,00
4 Ручка 151,00
Ошибка:
Не хватает товара "Карандаш" из необходимых 2 в наличии осталось только 0
Сумма (итог): 2 250,00
Сообщения
Не хватает товара "Ручка" из необходимых 10 в наличии осталось только 0
Не хватает товара "Карандаш" из необходимых 2 в наличии осталось только 0

Дата первого документа раньше, чем дата приходного документа, в котором на склад «Шушары» пришли ручки и карандаши:



А дата второго документа позже, поэтому документ провелся.

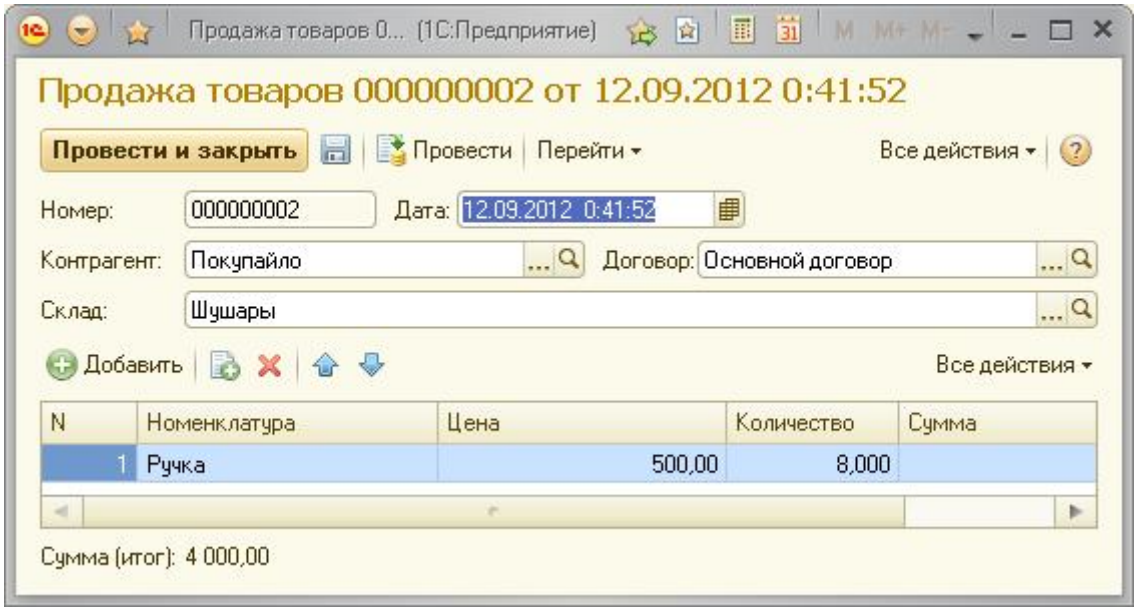
Посмотрим в отчете остатки по складам:



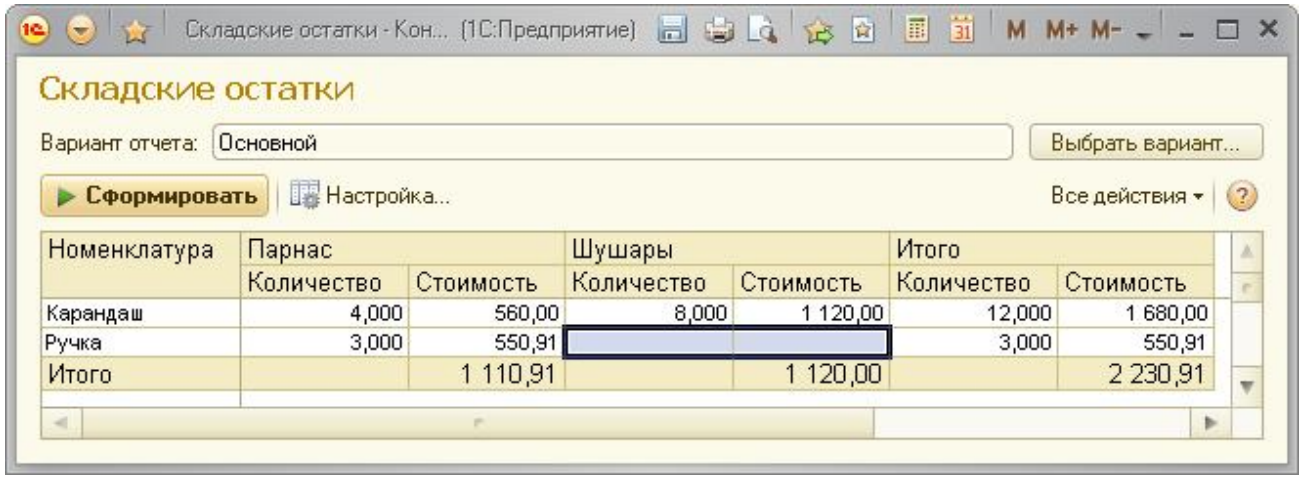
Номенклатура	Парнас		Шушары		Итого	
	Количество	Стоимость	Количество	Стоимость	Количество	Стоимость
Карандаш	4,000	560,00	6,000	840,00	10,000	1 400,00
Ручка	3,000	550,91	1,000	183,64	4,000	734,55
Итого		1 110,91		1 023,64		2 134,55

Добьемся полного списания ручек с одного склада. При этом остатком в суммовом выражении быть не должно:

Продадим все ручки:



И посмотрим отчет остатков на складах:



Номенклатура	Парнас		Шушары		Итого	
	Количество	Стоимость	Количество	Стоимость	Количество	Стоимость
Карандаш	4,000	560,00	8,000	1 120,00	12,000	1 680,00
Ручка	3,000	550,91			3,000	550,91
Итого		1 110,91		1 120,00		2 230,91

Занятие 1.13

При учете методом партий появляется понятие «Признак партии». Например, ящик, фура, договор, приходный документ и т.д. Чаще всего Признаком партии является документ, на основании которого оприходован товар.

В нашем примере «Признак партии» - конкретный документ «Приходная».

Таблица поступления и хранения партий товара:

Товар	Партия	Количество	Стоимость
Ложка	№1	10	100
Ложка	№2	10	200
Ложка	№3	10	300

Списание 17 штук товара по методам FIFO и LIFO:

	FIFO	LIFO
Партия	Партия №1, Партия №2	Партия №3, Партия №2
Списываемое количество	10, 7	10, 7
Списываемая себестоимость партии	100, 140	300, 140
Себестоимость оставшегося товара	360	160

FIFO – на складе остаются последние купленные товары. По данным учета товар оценен по ценам, максимально приближенным к рыночным. Используется в основном экономистами.

LIFO – при инфляционной политике государства (повышение цен), Прибыль псевдозанижается. Списываются самые дорогие товары (купленные последними), Себестоимость становится высокой и Прибыль уменьшается, а соответственно, уменьшается и Налог на прибыль.

В РФ применение метода LIFO для налогового учета запрещено.

Метод LIFO сейчас используется для управленческого учета.

В регистр накопления «ОстаткиТоваров» можно добавить признак Партии как дополнительное измерение:

Закладка «Данные»	
Ресурсы	
Партия	
Имя	Партия
Синоним	Партия
Тип	ДокументСсылка.Приходная

В больших конфигурация признаком партии могут быть несколько документов.

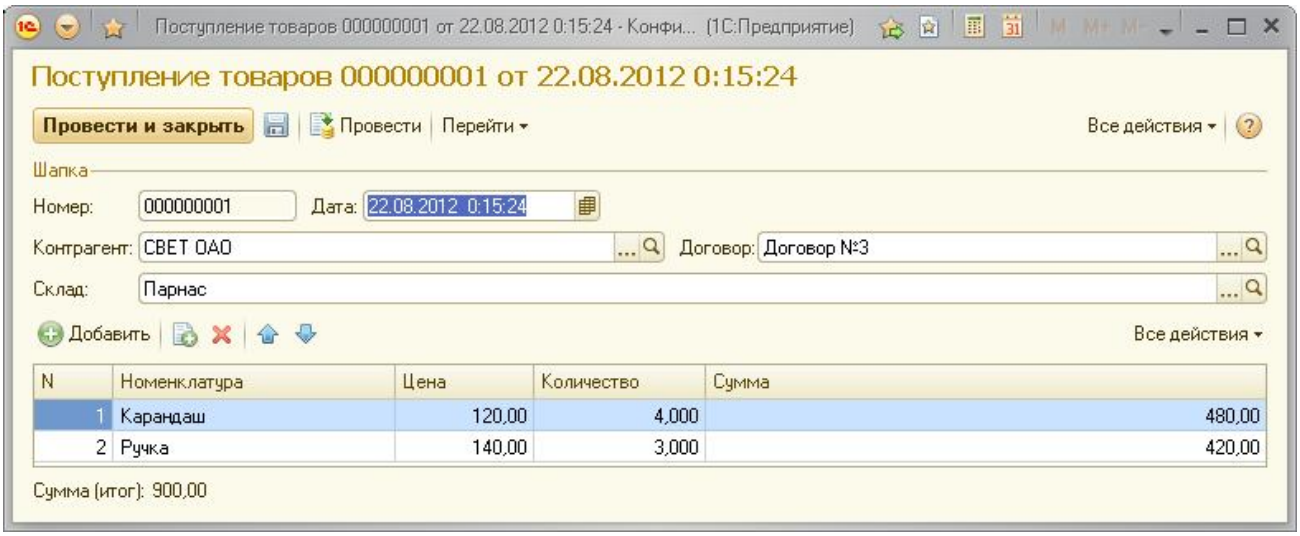
Определим, как будут разрезаться остатки по партиям:
В документе «Приходная» включим учет партий – при формировании движений внесем признак партий:

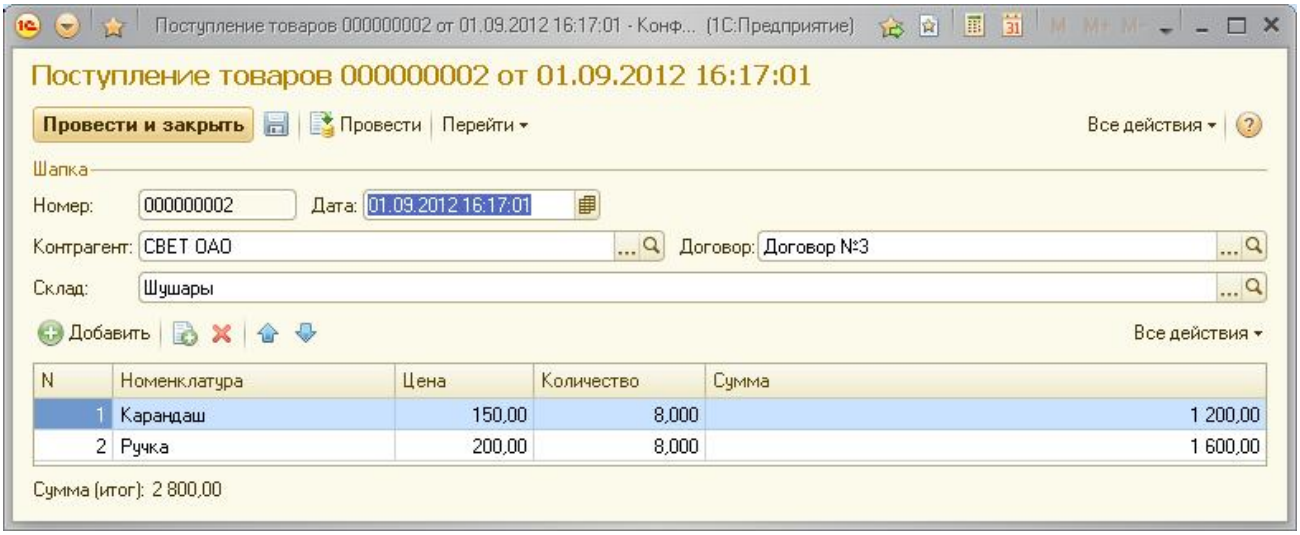
```
Процедура ОбработкаПроведения(Отказ, Режим)

    Движения.ЗакупочныеЦены.Записывать   = Истина;
    Движения.ОстаткиТоваров.Записывать   = Истина;
    Движения.ОстаткиПоСкладам.Записывать = Истина;

    Движения.ЗакупочныеЦены.Очистить();
    Движения.ОстаткиТоваров.Очистить();
    Движения.ОстаткиПоСкладам.Очистить();

    Для Каждого ТекСтрокаТовары Из Товары Цикл
        Движение = Движения.ЗакупочныеЦены.Добавить();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Контрагент = Контрагент;
        Движение.Цена = ТекСтрокаТовары.Цена;

        Движение = Движения.ОстаткиТоваров.ДобавитьПриход();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Количество = ТекСтрокаТовары.Количество;
        Движение.Стоимость = ТекСтрокаТовары.Сумма;
        Движение.Партия = Ссылка;

        Движение = Движения.ОстаткиПоСкладам.ДобавитьПриход();
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Количество = ТекСтрокаТовары.Количество;
        Движение.Склад = Склад;

    КонецЦикла;

КонецПроцедуры
```

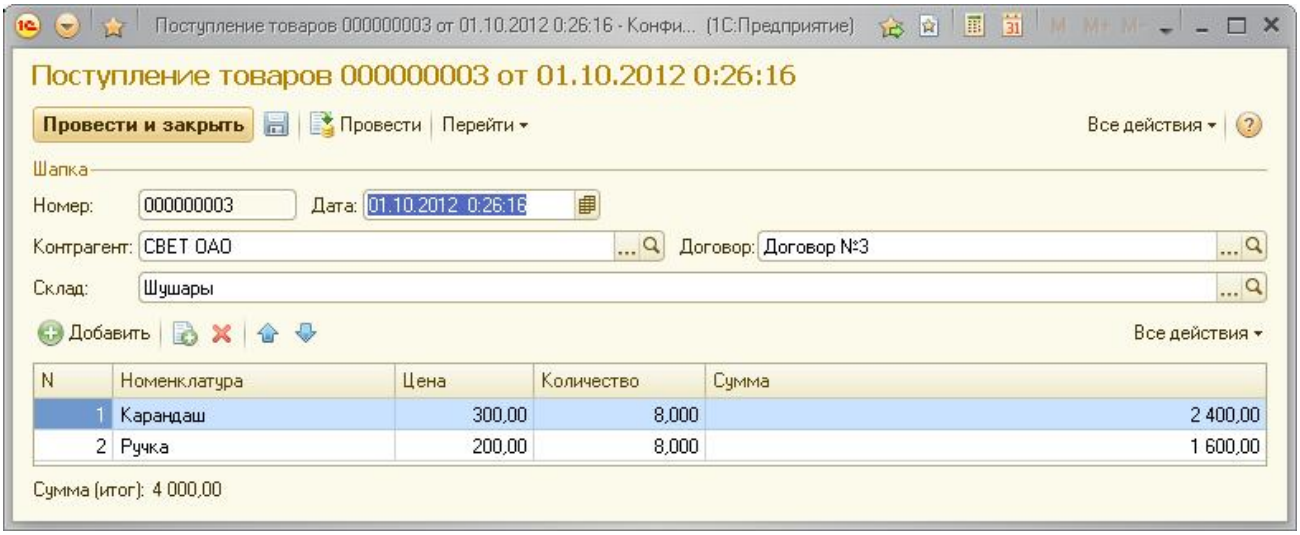
В режиме исполнения:

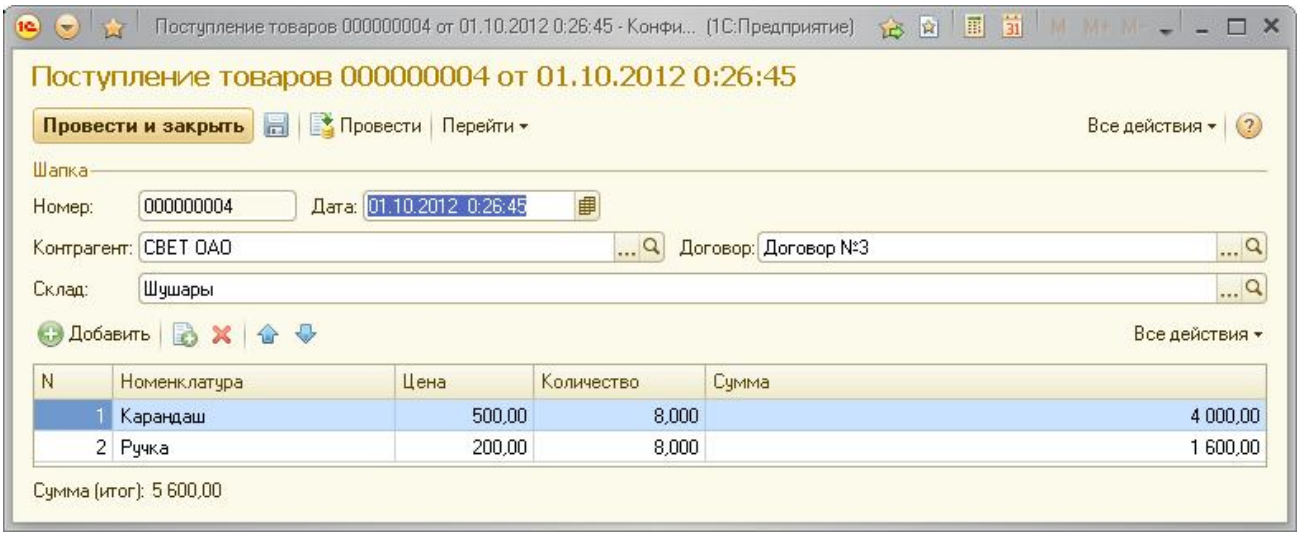
Тестовые наборы:

Перепроведем все приходные документы и создадим новые:

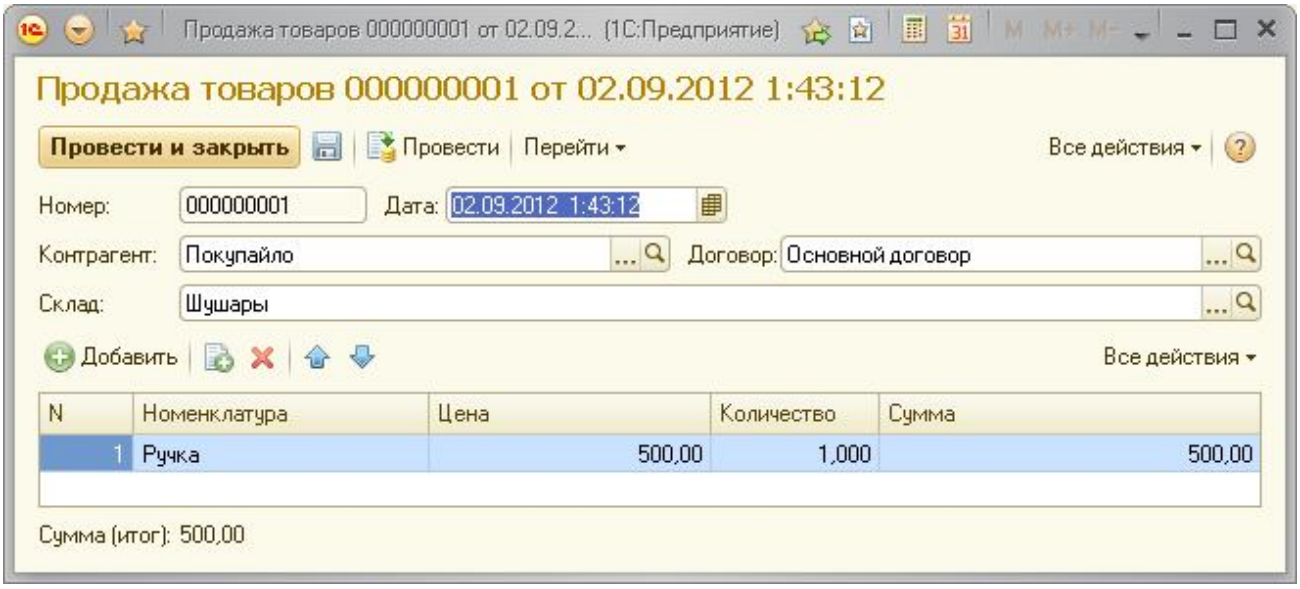

Поступление товаров 000000001 от 22.08.2012 0:15:24 - Конфи... (1С:Предприятие)
Поступление товаров 000000001 от 22.08.2012 0:15:24
Провести и закрыть
Провести
Перейти
Все действия
Шапка
Номер: 000000001
Дата: 22.08.2012 0:15:24
Контрагент: СВЕТ ОАО
Договор: Договор №3
Склад: Парнас
Добавить
Все действия
N | Номенклатура | Цена | Количество | Сумма
1 | Карандаш | 120,00 | 4,000 | 480,00
2 | Ручка | 140,00 | 3,000 | 420,00
Сумма (итог): 900,00

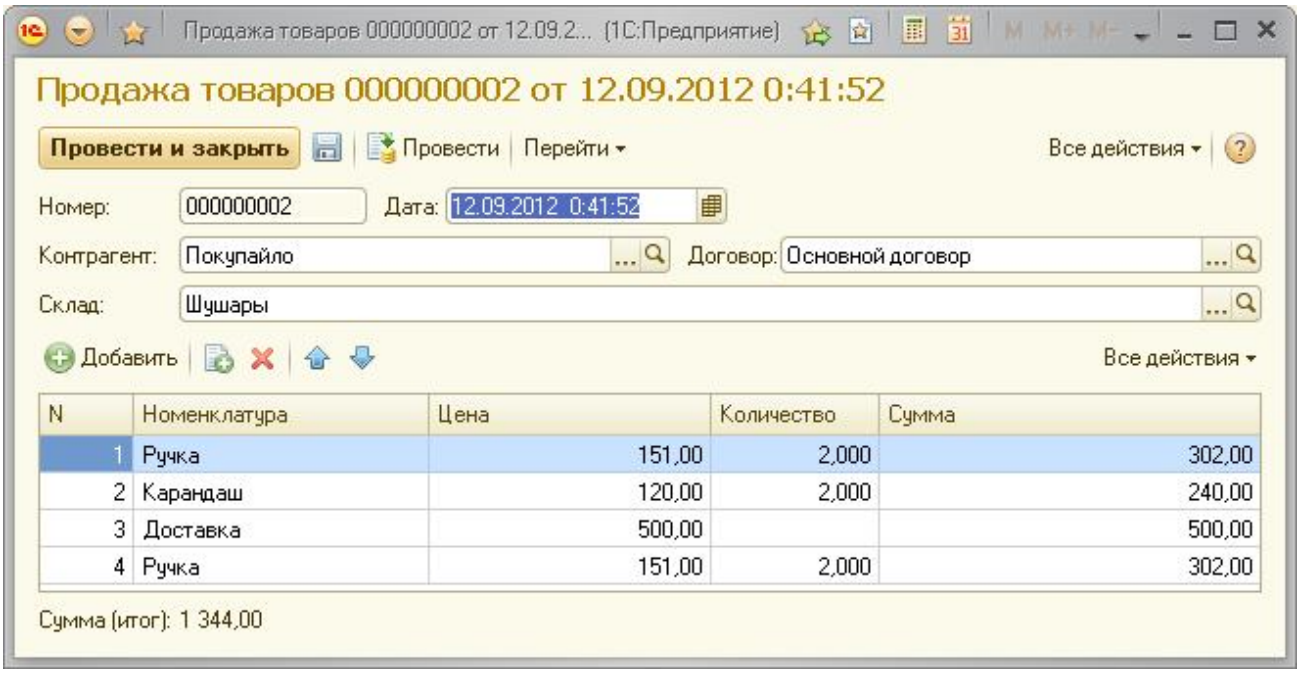

Поступление товаров 000000002 от 01.09.2012 16:17:01 - Конф... (1С:Предприятие)
Поступление товаров 000000002 от 01.09.2012 16:17:01
Провести и закрыть
Провести
Перейти
Все действия
Шапка
Номер: 000000002
Дата: 01.09.2012 16:17:01
Контрагент: СВЕТ ОАО
Договор: Договор №3
Склад: Шушары
Добавить
Все действия
N | Номенклатура | Цена | Количество | Сумма
1 | Карандаш | 150,00 | 8,000 | 1 200,00
2 | Ручка | 200,00 | 8,000 | 1 600,00
Сумма (итог): 2 800,00


Поступление товаров 000000003 от 01.10.2012 0:26:16 - Конфи... (1С:Предприятие)
Поступление товаров 000000003 от 01.10.2012 0:26:16
Провести и закрыть
Провести
Перейти
Все действия
Шапка
Номер: 000000003
Дата: 01.10.2012 0:26:16
Контрагент: СВЕТ ОАО
Договор: Договор №3
Склад: Шушары
Добавить
Все действия
N | Номенклатура | Цена | Количество | Сумма
1 | Карандаш | 300,00 | 8,000 | 2 400,00
2 | Ручка | 200,00 | 8,000 | 1 600,00
Сумма (итог): 4 000,00

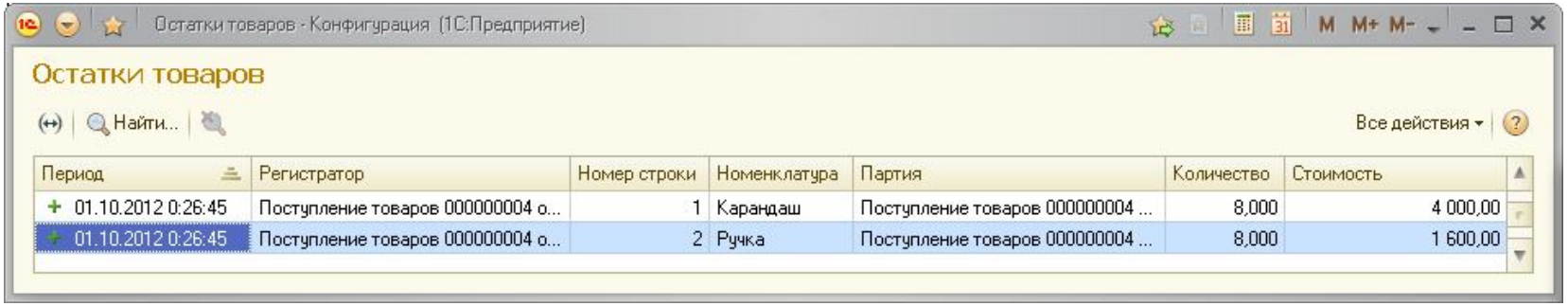


Отменим проведение у всех расходных документов:





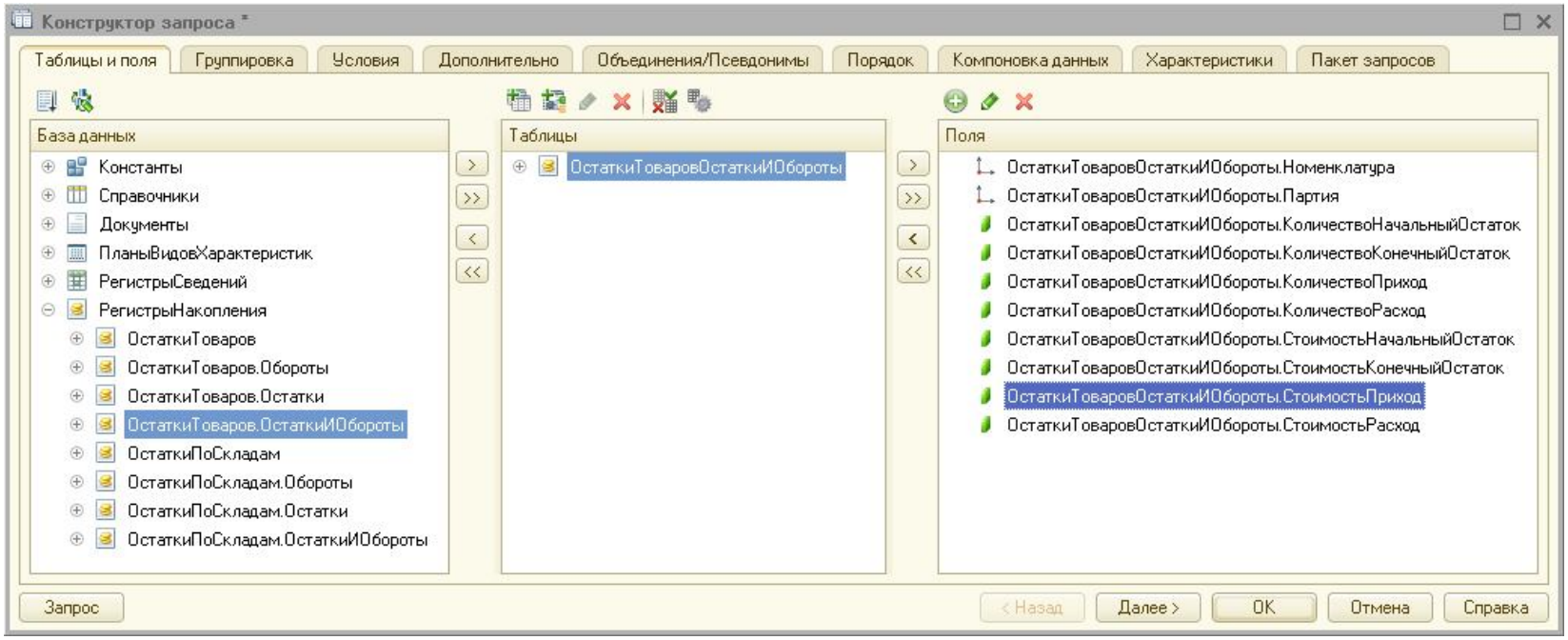
Посмотрим Остатки товаров по документу Расходная («Перейти – Остатки товаров»):



Создадим отчет-ведомость по состоянию остатков партий товаров – «Остатки партий товаров».

Свойство	Значение
Имя	ОстаткиПартийТоваров
Синоним	Остатки партий товаров

Откроем «Схему компоновки данных», добавим «Набор данных-запрос» и откроем «Конструктор запроса»:

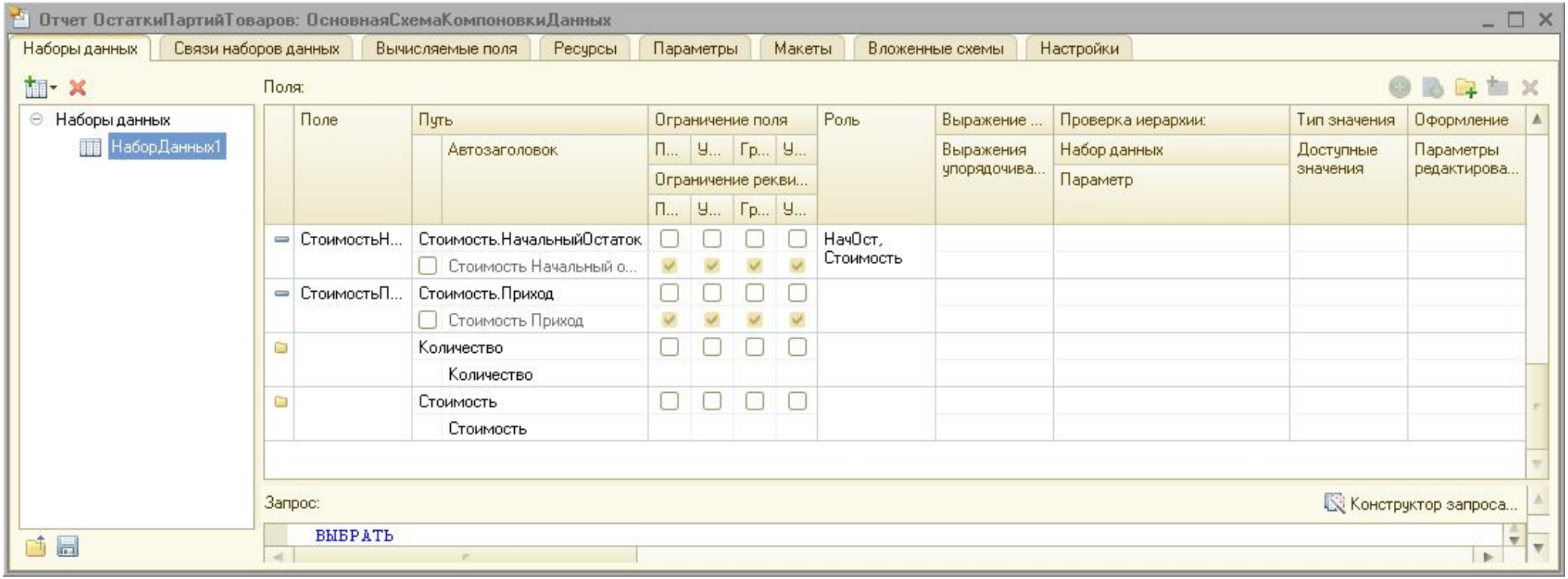


В СКД, на закладке «Наборы данных» к полям добавим две группы:

- Количество.
- Стоимость

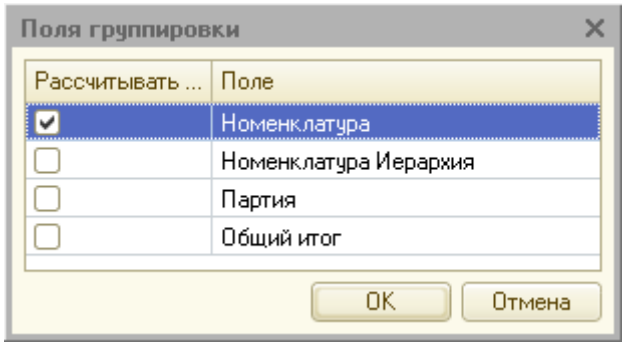
Чтобы пользователь мог одной галочкой включать/отключать стоимостное/количественное отображение.

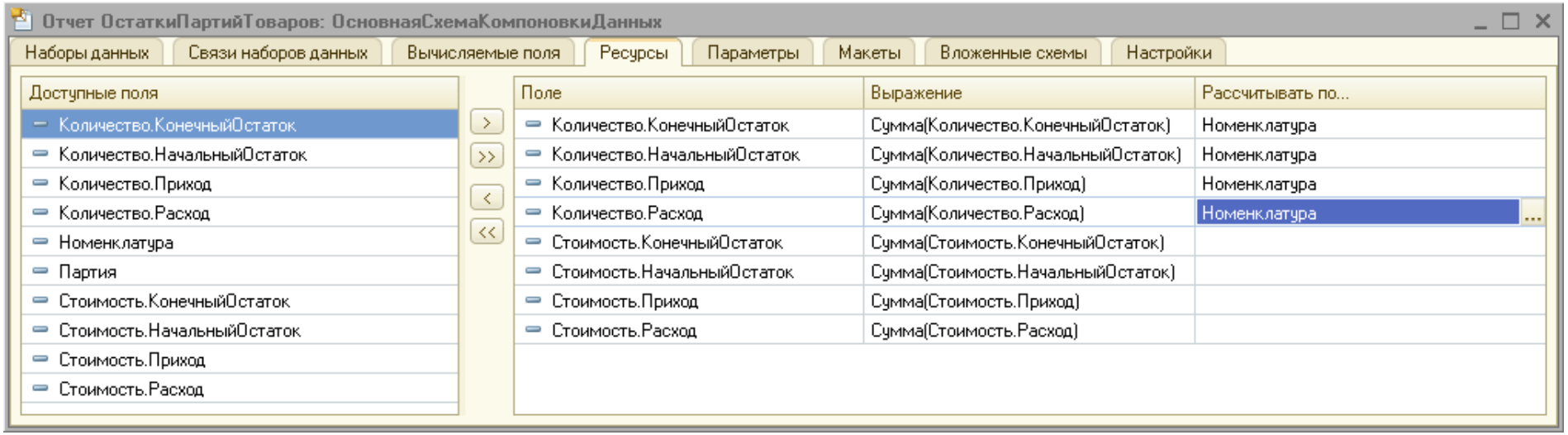
Во всех полях колонки «Путь» после слов «Количество» и «Стоимость» поставим точку – тем самым эти поля мы делаем подчиненными созданным группам.



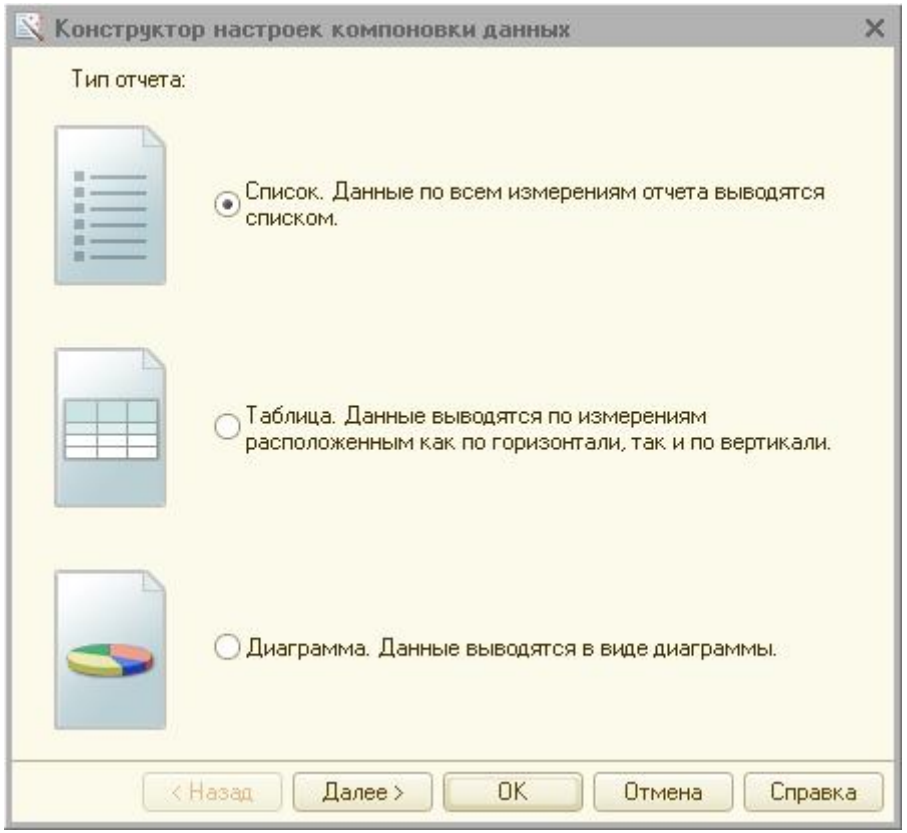
Закладка «Ресурсы»

Установим, чтобы количественный расчет устанавливался только по номенклатуре – если в одной партии есть и вилки и ложки, то количественный итог по партии и общий не имеют смысла:



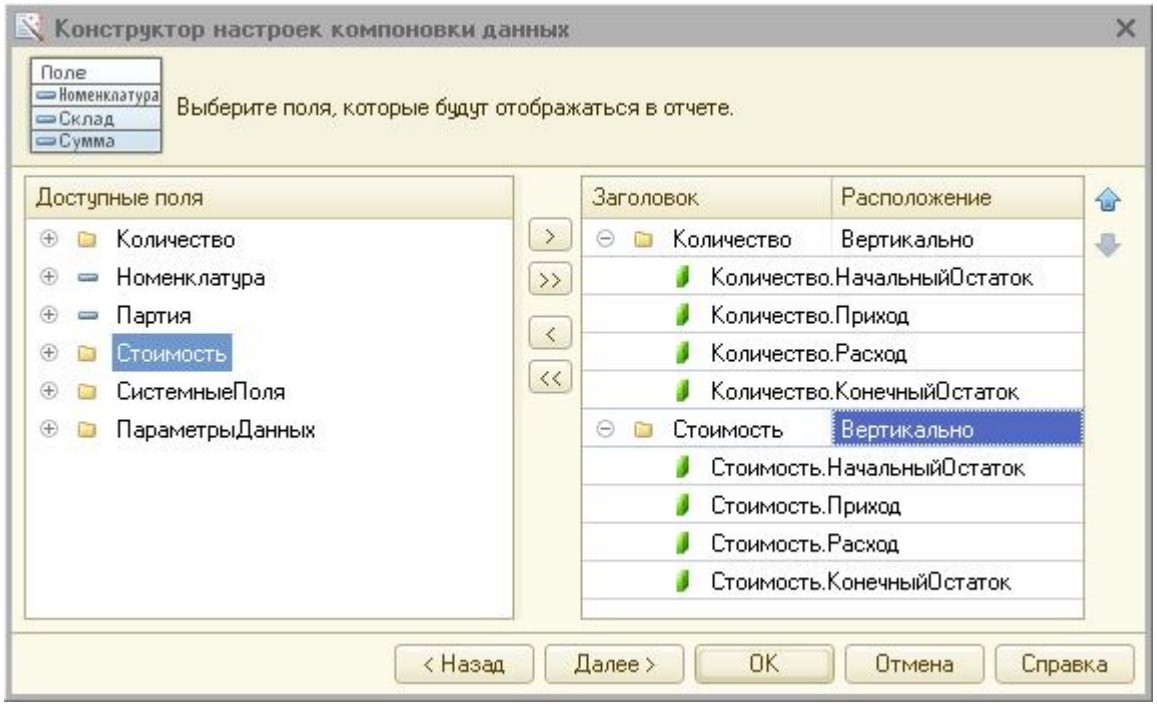


Закладка «Настройки» - сформируем основной вариант отчета с помощью «Конструктора настроек»

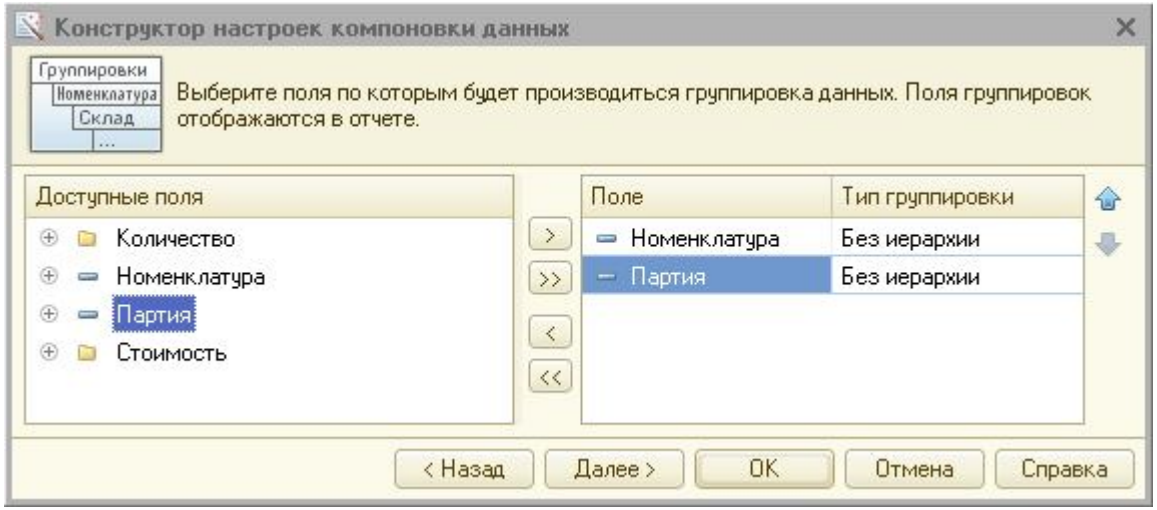


Что хотим отображать – количественный и стоимостной учет:

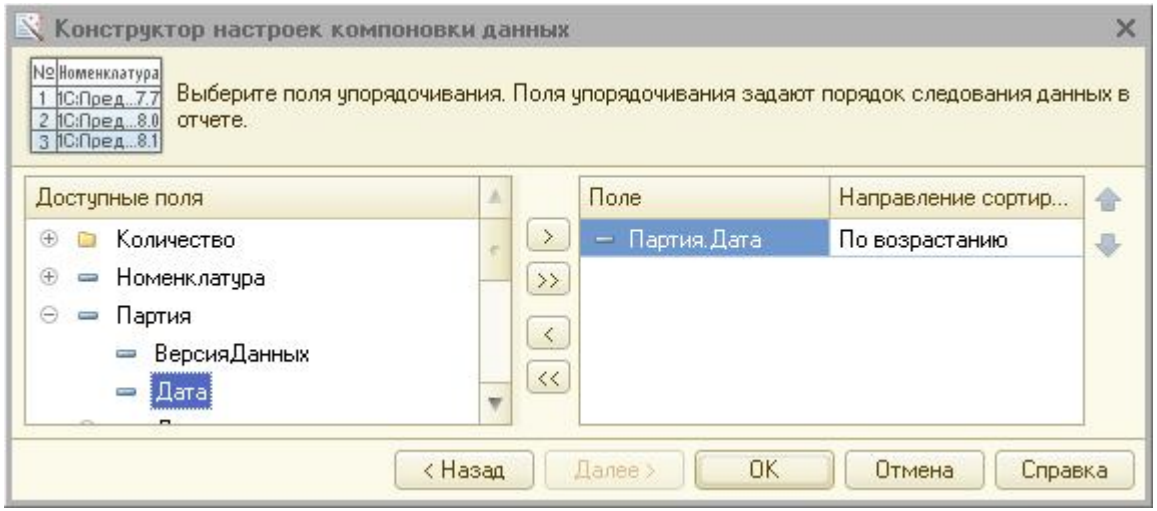
Для отображения групп «Количество» и «Стоимость» мы и создавали аналогичные группы на закладке «Наборы данных». Здесь же можно задать, каким образом они будут отображаться:



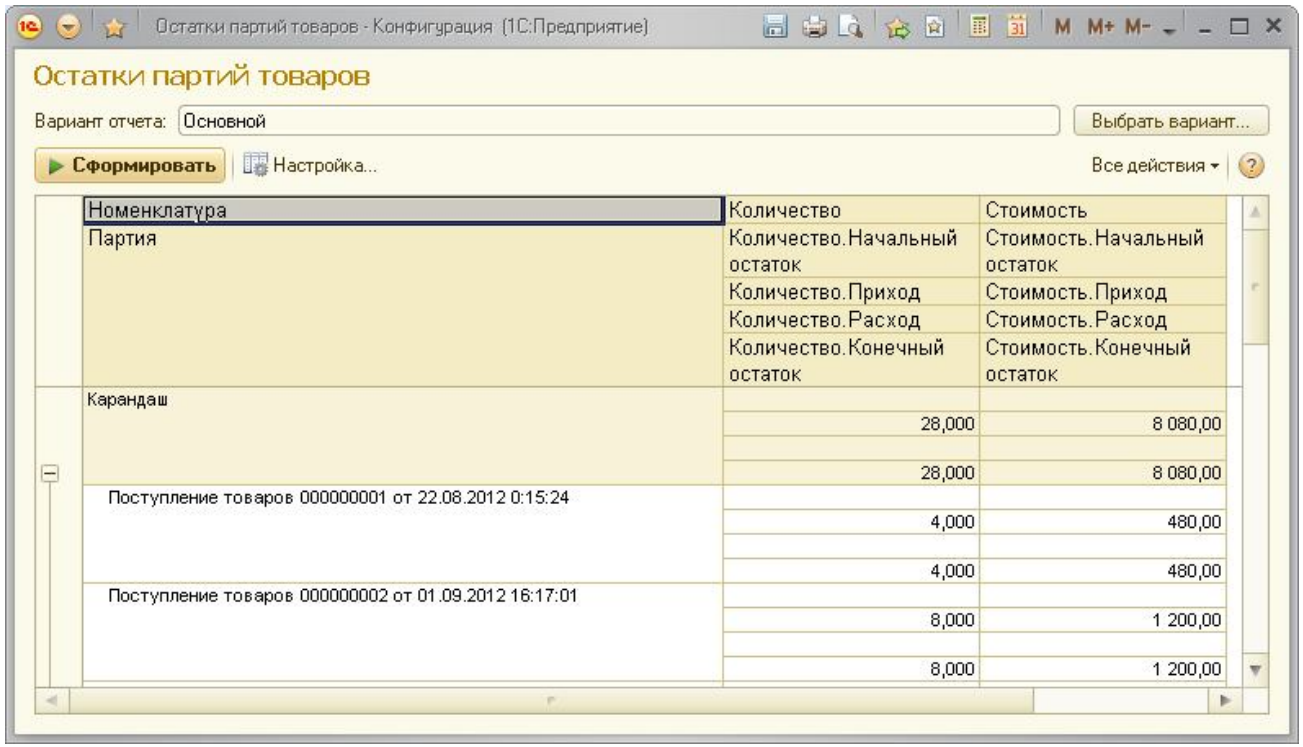
Группировка:



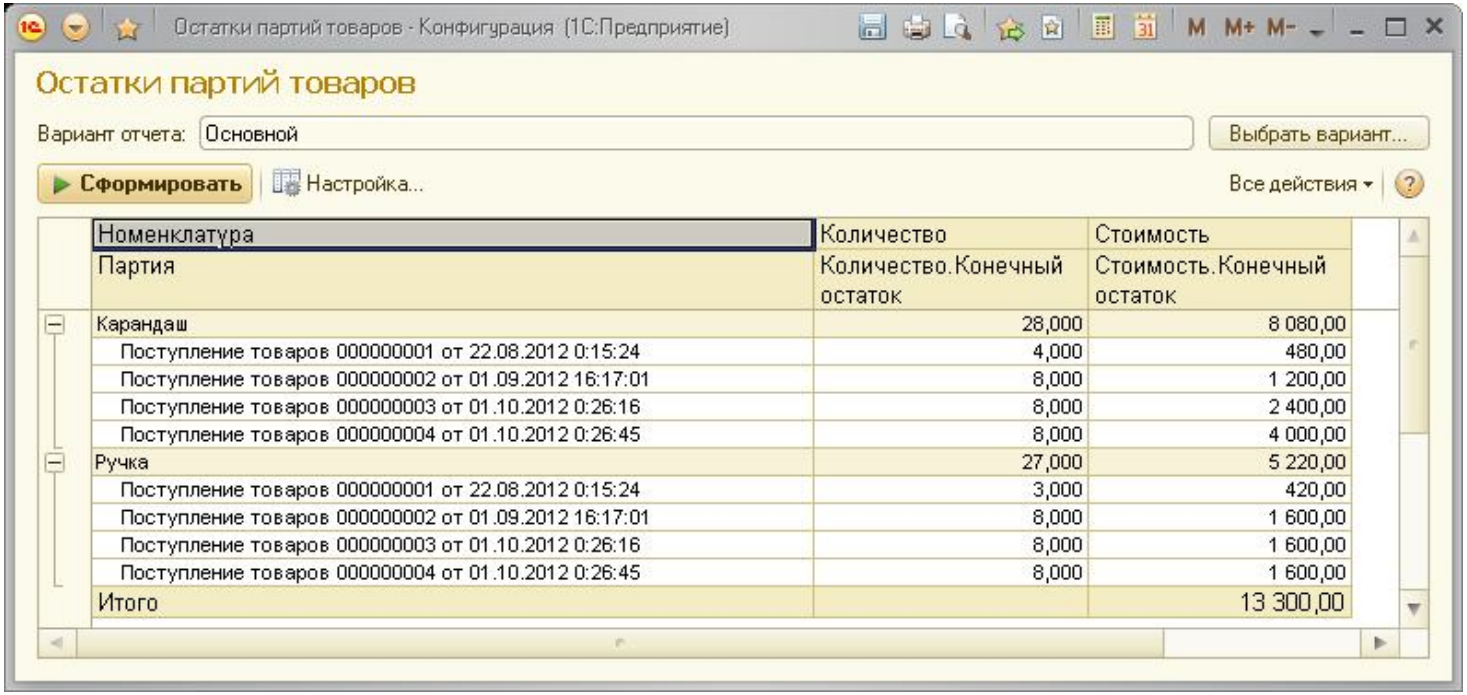
Сортировка – поле «Дата» партии будет являться полем для сортировки:



В режиме исполнения:



Отобразим в отчете только конечные остатки: «Все действия – Изменить вариант - Поля»



Номенклатура / Партия	Количество / Количество.Конечный остаток	Стоимость / Стоимость.Конечный остаток
Карандаш	28,000	8 080,00
Поступление товаров 000000001 от 22.08.2012 0:15:24	4,000	480,00
Поступление товаров 000000002 от 01.09.2012 16:17:01	8,000	1 200,00
Поступление товаров 000000003 от 01.10.2012 0:26:16	8,000	2 400,00
Поступление товаров 000000004 от 01.10.2012 0:26:45	8,000	4 000,00
Ручка	27,000	5 220,00
Поступление товаров 000000001 от 22.08.2012 0:15:24	3,000	420,00
Поступление товаров 000000002 от 01.09.2012 16:17:01	8,000	1 600,00
Поступление товаров 000000003 от 01.10.2012 0:26:16	8,000	1 600,00
Поступление товаров 000000004 от 01.10.2012 0:26:45	8,000	1 600,00
Итого		13 300,00

Допустим, надо продать 20 карандашей по методу FIFO. Партии отсортированы по возрастанию. Посмотрев на итог, делаем вывод, что можем продать 20 карандашей. Далее начинаем списывать карандаши по партиям. Четвертую партию списывать не будем, т.к. за три предыдущих партии мы списали нужное количество карандашей.

Допустим, надо продать 5 ручек по методу FIFO. Партии отсортированы по возрастанию. Посмотрев на итог, делаем вывод, что можем продать 5 ручек. Далее начинаем списывать ручки по партиям. Из второй партии спишем 2 ручки. Себестоимость этих двух ручек будем высчитывать по средней внутри партии.

Алгоритм списания товара по партиям при проведении документа «Расходная»:
Допустим, продаем 5 карандашей. Берем первую партию, смотрим какое количество товара в ней. Количество карандашей – 4, значит стоимость первой партии списываем полностью. Берем вторую партию. В ней 8 карандашей, а нам осталось списать один. Рассчитываем себестоимость одного карандаша внутри этой партии.

Из процедуры проведения документа «Расходная» возьмем запрос и перенесем в «Консоль запросов». Зададим следующие параметры:

Результат запроса:

Теперь мы можем обратиться к измерению «Партия» регистра накопления «Остатки товаров»

```
ВЫБРАТЬ
     РасходнаяТовары.Номенклатура,
     СУММА(РасходнаяТовары.Количество) КАК Количество,
     МИНИМУМ(РасходнаяТовары.НомерСтроки) КАК НомерСтроки
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары КАК РасходнаяТовары
ГДЕ
```

```
     РасходнаяТовары.Ссылка = &Ссылка
     И НЕ РасходнаяТовары.Номенклатура.Услуга

СГРУППИРОВАТЬ ПО
     РасходнаяТовары.Номенклатура
;
ВЫБРАТЬ
     ДокТЧ.НомерСтроки,
     ДокТЧ.Номенклатура,
     ДокТЧ.Количество,
     Остатки.Партия,
     ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
     ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
     ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
ИЗ
     ДокТЧ КАК ДокТЧ
          ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки(
                    &МоментВремени,
                    Номенклатура В
                         (ВЫБРАТЬ
                              ДокТЧ.Номенклатура
                         ИЗ
                              ДокТЧ КАК ДокТЧ)) КАК Остатки
          ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
          ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиПоСкладам.Остатки(
                    &МоментВремени,
                    Склад = &Склад
                         И Номенклатура В
                              (ВЫБРАТЬ
                                   ДокТЧ.Номенклатура
                              ИЗ
                                   ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
          ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
```

	1	2	3	4	5	6	7
1	НомерСтроки	Номенклатура	Количество	Партия	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
2	1	Ручка	4	Поступление товаров 000000002	8	1 600	3
3	1	Ручка	4	Поступление товаров 000000001	3	420	3
4	1	Ручка	4	Поступление товаров 000000003	8	1 600	3
5	1	Ручка	4	Поступление товаров 000000004	8	1 600	3
6	2	Карандаш	2	Поступление товаров 000000002	8	1 200	4
7	2	Карандаш	2	Поступление товаров 000000001	4	480	4
8	2	Карандаш	2	Поступление товаров 000000003	8	2 400	4
9	2	Карандаш	2	Поступление товаров 000000004	8	4 000	4

Цели:

Теперь видно стоимость и остаток партии, т.е. можно рассчитать себестоимость партии и реализовать партионное списание.
Контроль отрицательных остатков по складам – поля «Количество» и «ОстатокНаСкладе».

Для удобства анализа подведем итоги по полям «Количество» и «ОстатокНаСкладе

```
ВЫБРАТЬ
     РасходнаяТовары.Номенклатура,
     СУММА(РасходнаяТовары.Количество) КАК Количество,
     МИНИМУМ(РасходнаяТовары.НомерСтроки) КАК НомерСтроки
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары КАК РасходнаяТовары
ГДЕ
     РасходнаяТовары.Ссылка = &Ссылка
     И НЕ РасходнаяТовары.Номенклатура.Услуга

СГРУППИРОВАТЬ ПО
     РасходнаяТовары.Номенклатура
;

////////////////////////////////////////////////////////////////////////////////
ВЫБРАТЬ
     ДокТЧ.НомерСтроки,
```

```
    ДокТЧ.Номенклатура,
    ДокТЧ.Количество,
    Остатки.Партия,
    ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
    ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
    ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
ИЗ
    ДокТЧ КАК ДокТЧ
        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки(
                &МоментВремени,
                Номенклатура В
                    (ВЫБРАТЬ
                        ДокТЧ.Номенклатура
                    ИЗ
                        ДокТЧ КАК ДокТЧ)) КАК Остатки
        ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиПоСкладам.Остатки(
                &МоментВремени,
                Склад = &Склад
                    И Номенклатура В
                        (ВЫБРАТЬ
                            ДокТЧ.Номенклатура
                        ИЗ
                            ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
        ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
ИТОГИ МИНИМУМ(Количество), МИНИМУМ(ОстатокНаСкладе)
ПО ДокТЧ.Номенклатура
```

Запрос пакета 1

Запрос пакета 2

НомерСтроки	Номенклатура	Количество	Партия	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
NULL	Ручка	4	NULL	NULL	NULL	3
1	Ручка	4	Поступление товаров 000000	8	1 600	3
1	Ручка	4	Поступление товаров 000000	3	420	3
1	Ручка	4	Поступление товаров 000000	8	1 600	3
1	Ручка	4	Поступление товаров 000000	8	1 600	3
NULL	Карандаш	2	NULL	NULL	NULL	4
2	Карандаш	2	Поступление товаров 000000	8	1 200	4
2	Карандаш	2	Поступление товаров 000000	4	480	4
2	Карандаш	2	Поступление товаров 000000	8	2 400	4
2	Карандаш	2	Поступление товаров 000000	8	4 000	4

Таким запросом удобно сначала проанализировать хватает ли товара на списание, а потом уже списывать партии.

В данном примере надо продать 4 ручки, но остаток по складу – 3 ручки, хотя всего на складах их 8. Т.е. с этого склада мы ручки продать не сможем, документ не проведется.

Теперь в результате запроса есть два уровня:
По первому уровню будем контролировать отрицательные остатки – хватает или нет товара.

Запрос пакета 1

Запрос пакета 2

НомерСтроки	Номенклатура	Количество	Партия	КоличествоОстаток	СтоимостьОстаток	ОстатокНаСкладе
NULL	Ручка	4	NULL	NULL	NULL	3
NULL	Карандаш	2	NULL	NULL	NULL	4

Если товара хватает, то перебираем детальные записи и последовательно списываем партии.

Перенесем запрос в документ «Расходная».

Чтобы выборка перебирала только итоги (группировки):

```
Выборка = Результат.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
```

На этом этапе проверяем, хватает товара или нет:

```
Пока Выборка.Следующий() Цикл
     Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура + """ из необходимых " +
Выборка.Количество + " в наличии осталось только " + Выборка.ОстатокНаСкладе;
          Сообщение.Поле = "Товары[" + (Выборка.НомерСтроки - 1) + "].Количество";
          Сообщение.УстановитьДанные(ЭтотОбъект);
          Сообщение.Сообщить();
          Отказ = Истина;
          Движения.ОстаткиТоваров.Записывать = Ложь;
          Движения.ОстаткиПоСкладам.Записывать = Ложь;
     КонецЕсли;
КонецЦикла;
```

Если нет, то процедура дальше не выполняется:

```
Если Отказ Тогда
     Возврат;
КонецЕсли;
```

Если товара хватает, то сделаем сброс выборки и организуем второй цикл, внутри которого будем производить партионное списание:

Т.к. выборка опять на уровне итогов, то, до входа в цикл по партиям, запомним в переменной, сколько всего товара надо списать. Эту переменную будем уменьшать. Далее организуем цикл по детальным записям. В нем проверим, сколько надо списать товара из партии:

```
Списать = Мин(ОсталосьСписать, ВыборкаПартии.КоличествоОстаток);
```

Рассчитаем себестоимость списания – по методу средней, но в пределах партии:

```
Себестоимость = Списать / ВыборкаПартии.КоличествоОстаток * ВыборкаПартии.СтоимостьОстаток;
```

Создадим движения по регистру:

```
Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
Движение.Период = Дата;
Движение.Номенклатура = Выборка.Номенклатура;
Движение.Количество = Списать;
Движение.Стоимость = Себестоимость;
Движение.Партия = ВыборкаПартии.Партия;
```

Уменьшим количество, которое осталось списать:

```
ОсталосьСписать = ОсталосьСписать - Списать;
```

Если списывать больше нечего, то цикл больше не запускаем:

```
Пока ВыборкаПартии.Следующий() И ОсталосьСписать <> 0 Цикл
```

Для реализации метода списания FIFO надо в запросе отсортировать записи по реквизиту «Дата» колонки «Партия» по возрастанию:

```
|УПОРЯДОЧИТЬ ПО Партия.Дата ВОЗР
```

Для реализации метода списания LIFO вместо «ВОЗР» надо поставить «УБЫВ».

Итого код проведения документа «Расходная»:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

     Движения.ОстаткиТоваров.Записывать   = Истина;
     Движения.ОстаткиПоСкладам.Записывать = Истина;
```

```
Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
|    РасходнаяТовары.Номенклатура,
|    СУММА(РасходнаяТовары.Количество) КАК Количество,
|    МИНИМУМ(РасходнаяТовары.НомерСтроки) КАК НомерСтроки
|ПОМЕСТИТЬ ДокТЧ
|ИЗ
|    Документ.Расходная.Товары КАК РасходнаяТовары
|ГДЕ
|    РасходнаяТовары.Ссылка = &Ссылка
|    И НЕ РасходнаяТовары.Номенклатура.Услуга

|СГРУППИРОВАТЬ ПО
|    РасходнаяТовары.Номенклатура
|;

|ВЫБРАТЬ
|    ДокТЧ.НомерСтроки,
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Количество,
|    Остатки.Партия,
|    ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
|    ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
|    ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
|ИЗ
|    ДокТЧ КАК ДокТЧ
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки(
|                &МоментВремени,
|                Номенклатура В
|                    (ВЫБРАТЬ
|                        ДокТЧ.Номенклатура
|                    ИЗ
|                        ДокТЧ КАК ДокТЧ)) КАК Остатки
|        ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиПоСкладам.Остатки(
```

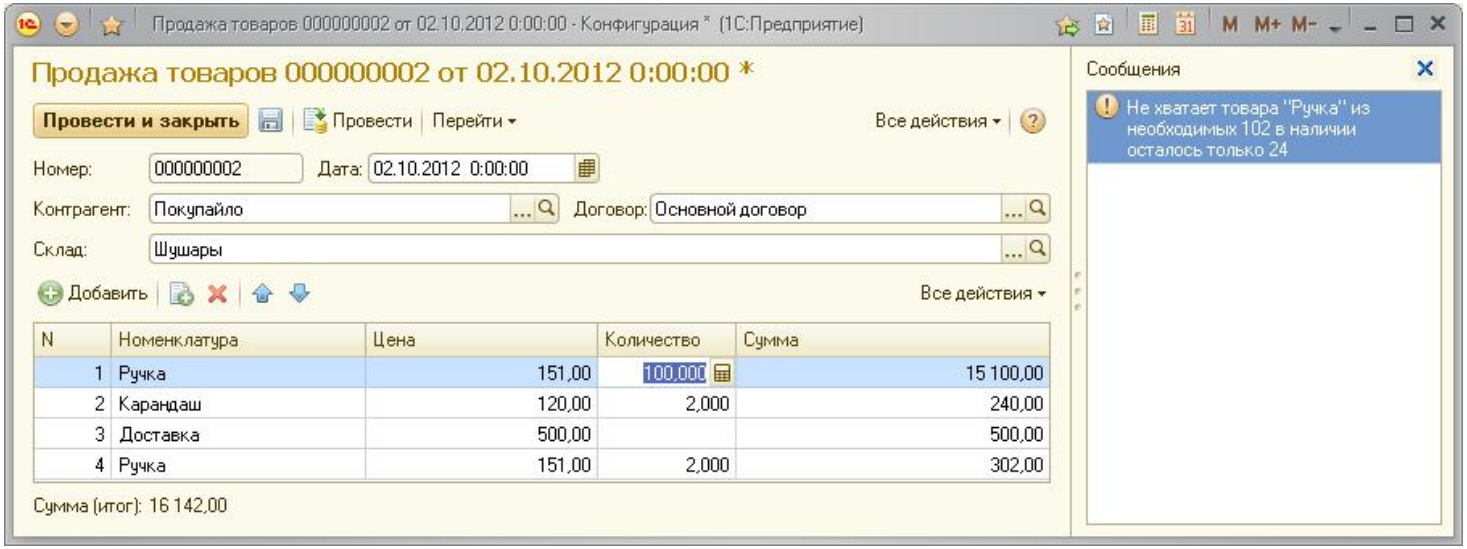
```
    |                    &МоментВремени,
    |                    Склад = &Склад
    |                         И Номенклатура В
    |                              (ВЫБРАТЬ
    |                                   ДокТЧ.Номенклатура
    |                              ИЗ
    |                                   ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
    |         ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
    |УПОРЯДОЧИТЬ ПО Партия.Дата ВОЗР
    |ИТОГИ МИНИМУМ(Количество), МИНИМУМ(ОстатокНаСкладе), МИНИМУМ(НомерСтроки)
    |ПО ДокТЧ.Номенклатура";

    Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
    Запрос.УстановитьПараметр("Ссылка", Ссылка);
    Запрос.УстановитьПараметр("Склад", Склад);

    Результат = Запрос.Выполнить();

    Выборка = Результат.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

    Пока Выборка.Следующий() Цикл
         Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
              Сообщение = Новый СообщениеПользователю;
              Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура + """ из необходимых " +
Выборка.Количество + " в наличии осталось только " + Выборка.ОстатокНаСкладе;
              Сообщение.Поле = "Товары[" + (Выборка.НомерСтроки - 1) + "].Количество";
              Сообщение.УстановитьДанные(ЭтотОбъект);
              Сообщение.Сообщить();
              Отказ = Истина;
              Движения.ОстаткиТоваров.Записывать   = Ложь;
              Движения.ОстаткиПоСкладам.Записывать = Ложь;
         КонецЕсли;
    КонецЦикла;

    Если Отказ Тогда
         Возврат;
```

```
    КонецЕсли;

    Выборка.Сбросить();

    Пока Выборка.Следующий() Цикл

        ОсталосьСписать = Выборка.Количество;

        ВыборкаПартии = Выборка.Выбрать();
        Пока ВыборкаПартии.Следующий() И ОсталосьСписать <> 0 Цикл

            Списать       = Мин(ОсталосьСписать, ВыборкаПартии.КоличествоОстаток);
            Себестоимость = Списать / ВыборкаПартии.КоличествоОстаток * ВыборкаПартии.СтоимостьОстаток;

            Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
            Движение.Период       = Дата;
            Движение.Номенклатура = Выборка.Номенклатура;
            Движение.Количество   = Списать;
            Движение.Стоимость    = Себестоимость;
            Движение.Партия       = ВыборкаПартии.Партия;

            ОсталосьСписать = ОсталосьСписать - Списать;
        КонецЦикла;

        Движение = Движения.ОстаткиПоСкладам.ДобавитьРасход();
        Движение.Период       = Дата;
        Движение.Номенклатура = Выборка.Номенклатура;
        Движение.Количество   = Выборка.Количество;
        Движение.Склад        = Склад;
    КонецЦикла;
КонецПроцедуры
```

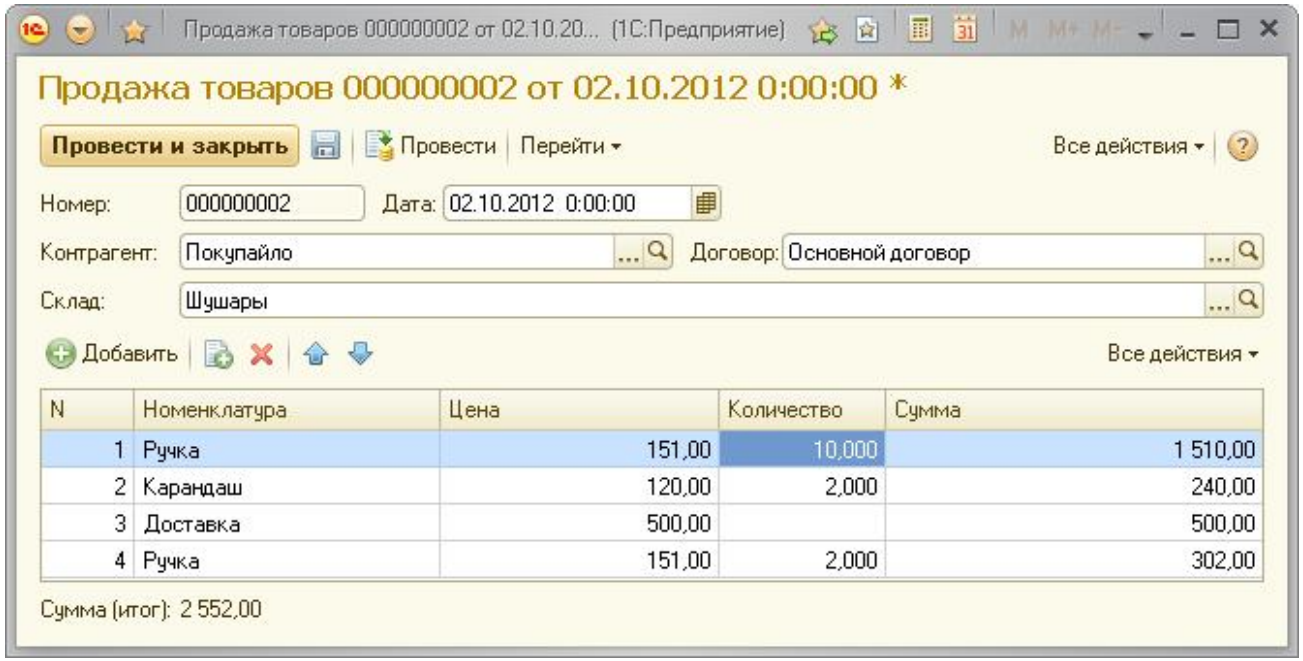
!!! «Выборка.Количество» - это поле, а «Выборка.Количество()» - это количество строк в выборке !!!

В режиме исполнения:

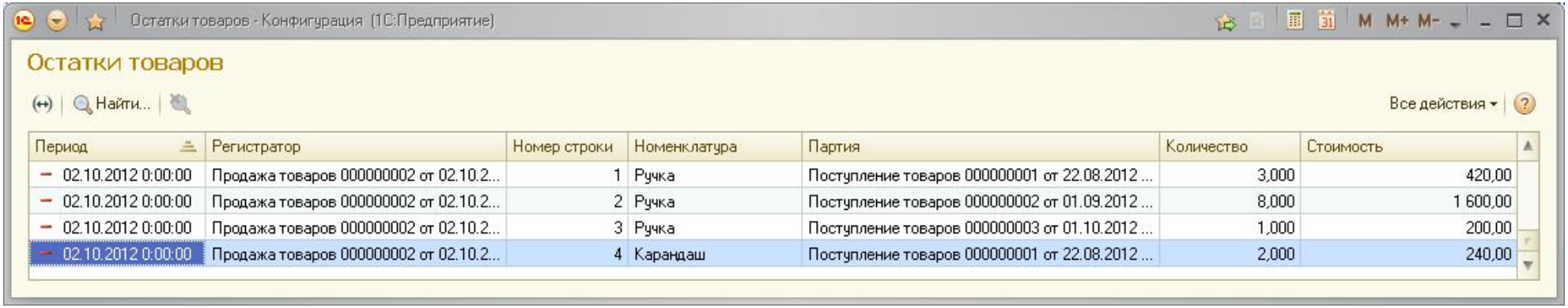
Т.к. мы хотим продать ручек больше, чем есть на складе, то система выдает ошибку:



Продадим 12 ручек:



И посмотрим отчет «Остатки товаров» - через меню «Перейти» документа:



Период	Регистратор	Номер строки	Номенклатура	Партия	Количество	Стоимость
02.10.2012 0:00:00	Продажа товаров 000000002 от 02.10.2...	1	Ручка	Поступление товаров 000000001 от 22.08.2012 ...	3,000	420,00
02.10.2012 0:00:00	Продажа товаров 000000002 от 02.10.2...	2	Ручка	Поступление товаров 000000002 от 01.09.2012 ...	8,000	1 600,00
02.10.2012 0:00:00	Продажа товаров 000000002 от 02.10.2...	3	Ручка	Поступление товаров 000000003 от 01.10.2012 ...	1,000	200,00
02.10.2012 0:00:00	Продажа товаров 000000002 от 02.10.2...	4	Карандаш	Поступление товаров 000000001 от 22.08.2012 ...	2,000	240,00

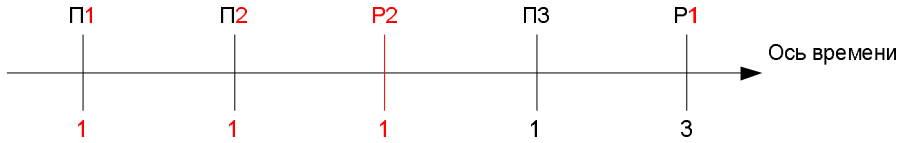
Видно, что первая и вторая партии списались полностью, и из третьей партии списалась еще одна ручка.

Занятие 1.14

Последовательности

Проблемы при работе с последовательностями:

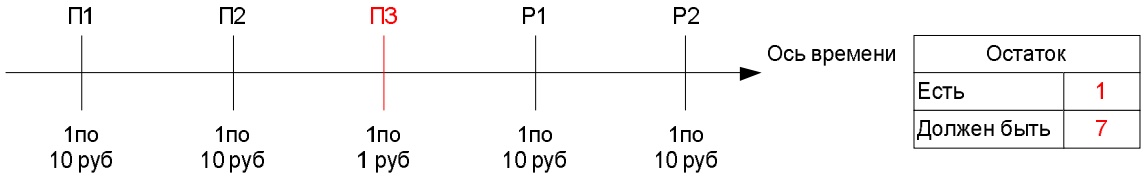
Первая ситуация:
Есть ось времени, на которой последовательно вводят Приходные и Расходные документы. Ввели 3 Приходных (П1, П2, П3) с одним и тем же товаром, а затем Расходную (Р1), в которой продают все 3 товара. Т.е. итого на складе ноль штук. Через некоторое время пользователь задним числом вводит еще один документ Расходная (Р2). Если в нем продается хотя бы одна штука товара, то на конец оси времени появляется отрицательный остаток (-1 штука). Система позволила провести Р2, т.к. система проверяет наличие товара на Момент времени, а на Момент времени Р2 было 2 штуки товара.



Т.е. если сейчас попробовать перепровести Р1, то система не даст этого сделать, т.к. товара не хватает.

Для исправления надо найти документ, в котором начинает не хватать товара или документ, нарушивший последовательность. Что делать с этими документами, будет решать Пользователь.

Вторая ситуация:
Есть ось времени, на которой последовательно вводят 2 Приходные (П1, П2) и 2 Расходные (Р1, Р2). В Приходных покупается один и тот же товар по 10 рублей. Обе Расходные продают по одной единице товара по 10 рублей. Задним числом вводят Приходную (П3), в которой покупается одна единица товара на один рубль. На конец оси времени есть Остаток – одна единица товара на один рубль. При этом Себестоимость товара рассчитана неправильно, Остаток должен быть равен 7 рублей. Получается, что мы завысили Себестоимость товара, а, следовательно, занизили Прибыль, т.е. Налог на прибыль заплатим меньше. А неправильно начисленный Налог на прибыль ведет к уголовной ответственности.



Остаток	
Есть	1
Должен быть	7

Решение: перепровести Расходные Р1 и Р2, после чего Себестоимость и Остатки будут верны.

Чтобы исправить последствия непоследовательного ввода данных, надо найти точку, которая нарушила последовательность. В первом примере это Расходная Р2, а во втором примере – Приходная П3.

Объект «Последовательности» отслеживает непоследовательный ввод данных в регистры. Последовательность состоит из двух настроек:

- По каким регистрам отслеживать движения.
- Какие документы входят в последовательность.

В примере надо отслеживать движения по регистру накопления «Остатки товаров», а документ, влияющий на последовательность – Расходная.

Документы, влияющие на последовательность – это документы, проведение по регистрам которых обусловлено.
Пример: Приходная как проводилась, так и проводится (себестоимость не рассчитывается, отрицательные остатки не проверяются), в отличие от проведения Расходной (рассчитывается себестоимость, проверяются остатки).

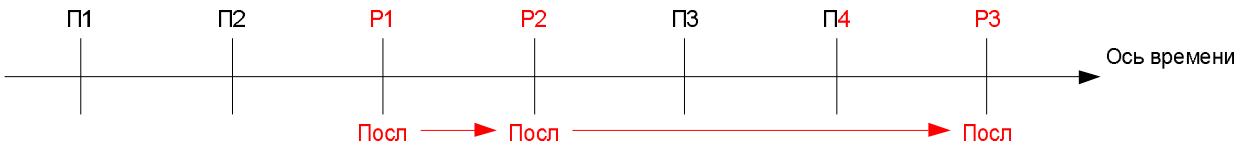
«Документы - Последовательности». Добавим последовательность «Себестоимость партий товаров»

Свойство	**Значение**
Имя	СебестоимостьПартийТоваров
Синоним	Себестоимость партий товаров
Закладка «Использование»	
Перемещение границы при проведении	Перемещать
Движения, влияющие на последовательность	РегистрНакопления.ОстаткиТоваров
Входящие документы	Документ.Расходная

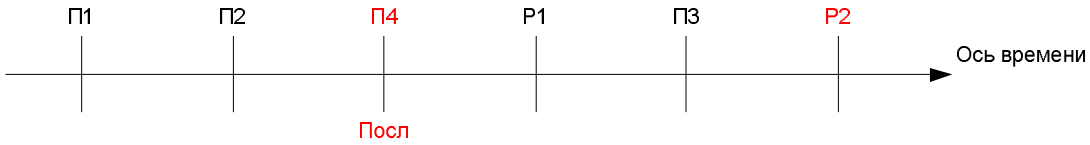
Движения, влияющие на последовательность – по каким регистрам отслеживается последовательный ввод данных.
Входящие документы – какие документы входят в последовательность.

В примере документ «Расходная» имеет обусловленное проведение по регистру – т.е. для него важно, чтобы при проведении данные по регистру «Остатки товаров» были актуальны и введены последовательно.

«Перемещение границы при проведении - Перемещать» - система автоматически перемещает последовательность вперед/назад.
Пример:
Последовательно введем Приходные (П1, П2), Расходные (Р1, Р2) и опять Приходные (П3, П4). Последовательность сдвигается вперед с каждым документом Расходная:

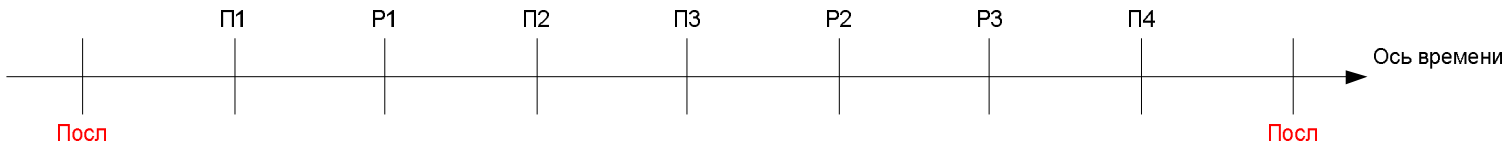


Допустим, последовательно вводим Приходные (П1, П2), Расходную (Р1) и опять Приходную (П3). Далее непоследовательно вводим Приходную (П4) – последовательность при этом автоматом переместится на нее (назад). И останется там до тех пор, пока последовательно не перепроведем все Расходные от этой точки и вперед. Т.е. если сейчас введем Расходную (Р2), то система не перебросит на нее границу последовательности, т.к. между границей последовательности и Расходной Р2 есть еще одна Расходная – Р1.



«Перемещение границы при проведении – Не перемещать» - последовательность перемещаться вперед автоматом не будет.
Пример:
Последовательность установлена вначале. Вводим документы П1, Р1, П2, П3, Р2, Р3, П4. При этом последовательность остается на месте. В конце месяца надо закрыть месяц и поставить последовательность в конец. При этом система перепроведет документы Р1, Р2,

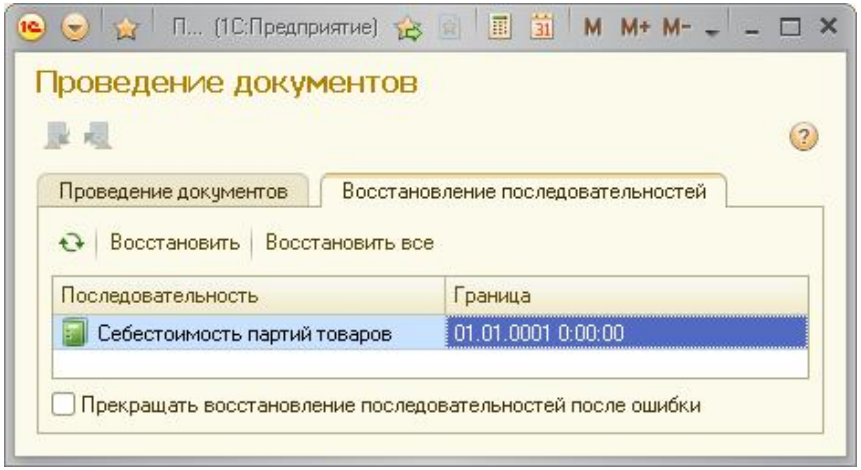
Р3 и установит последовательность на указанную дату, т.е. в конце месяца автоматом перепроведутся документы, входящие в последовательность. Но если задним числом введут или перепроведут Приходную или Расходную, то последовательность установится на нее.



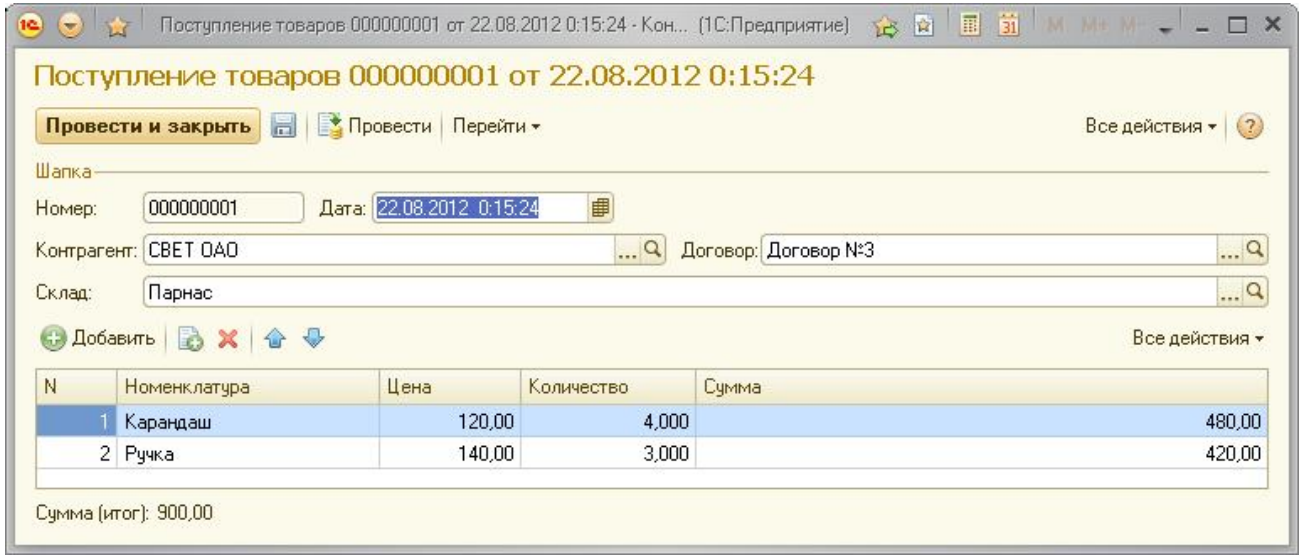
Что выбирать «Перемещать» или «Не перемещать». Перемещение последовательности вперед – это постоянный анализ при проведении каждого документа - это нагружает систему. «Перемещать» надо выбирать только если при каждом проведении документа «Расходная» важна его последовательность (например, при передаче материалов в производство, чтобы сразу же верно рассчитывалась их себестоимость).
Обычно последовательность перемещать автоматом не надо, т.к. бухгалтера смотрят себестоимость в конце месяца (при закрытии месяца).

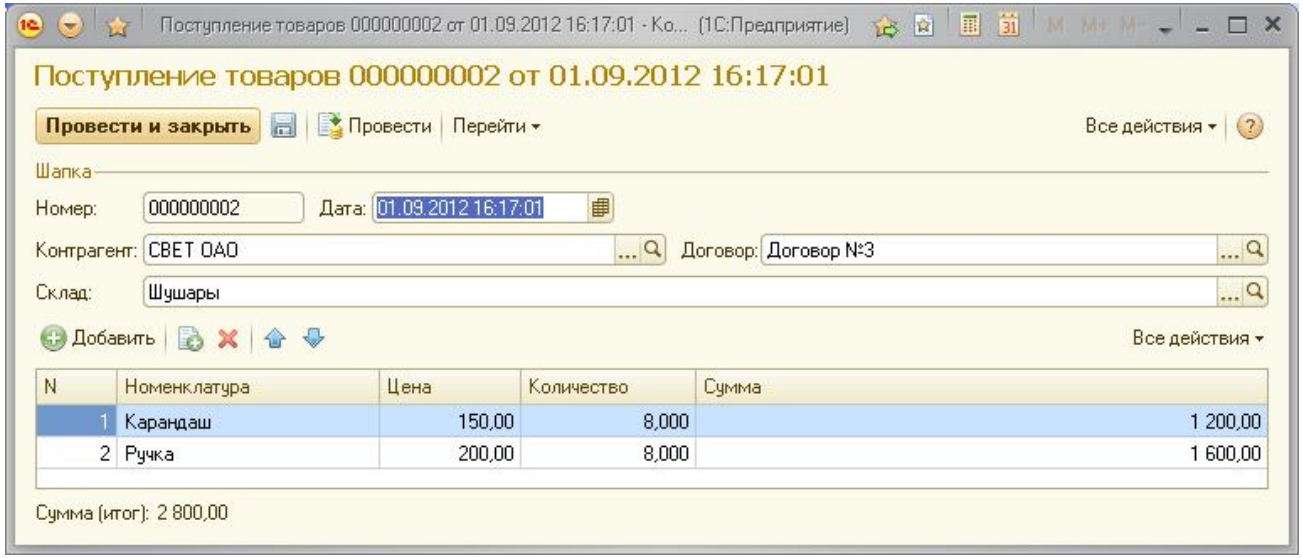
В режиме исполнения:

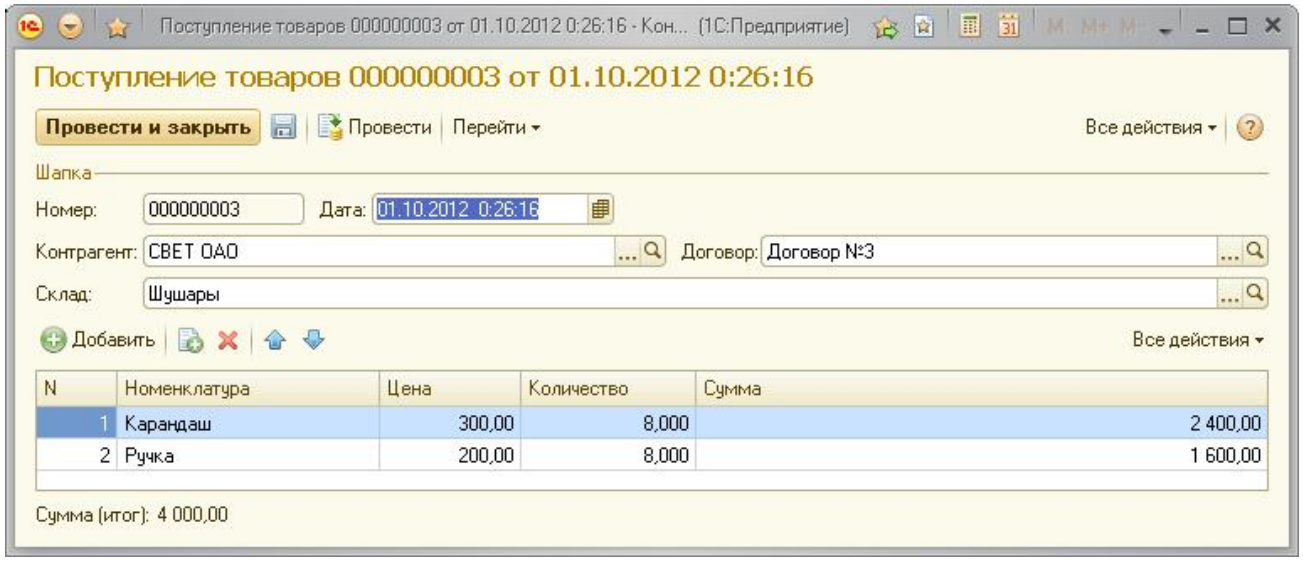
«Все функции – Стандартные – Проведение документов – Восстановление последовательностей». Сейчас граница последовательности установлена на 01.01.0001 0:00:00.

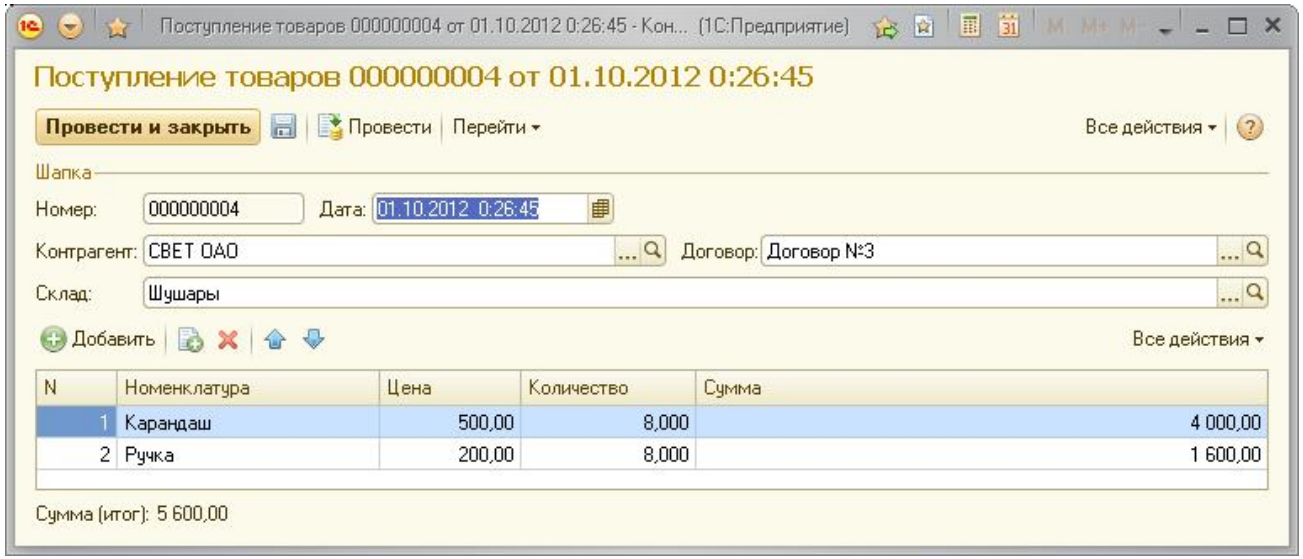


Проведем документы Приходная. При этом с последовательностью ничего не произойдет.

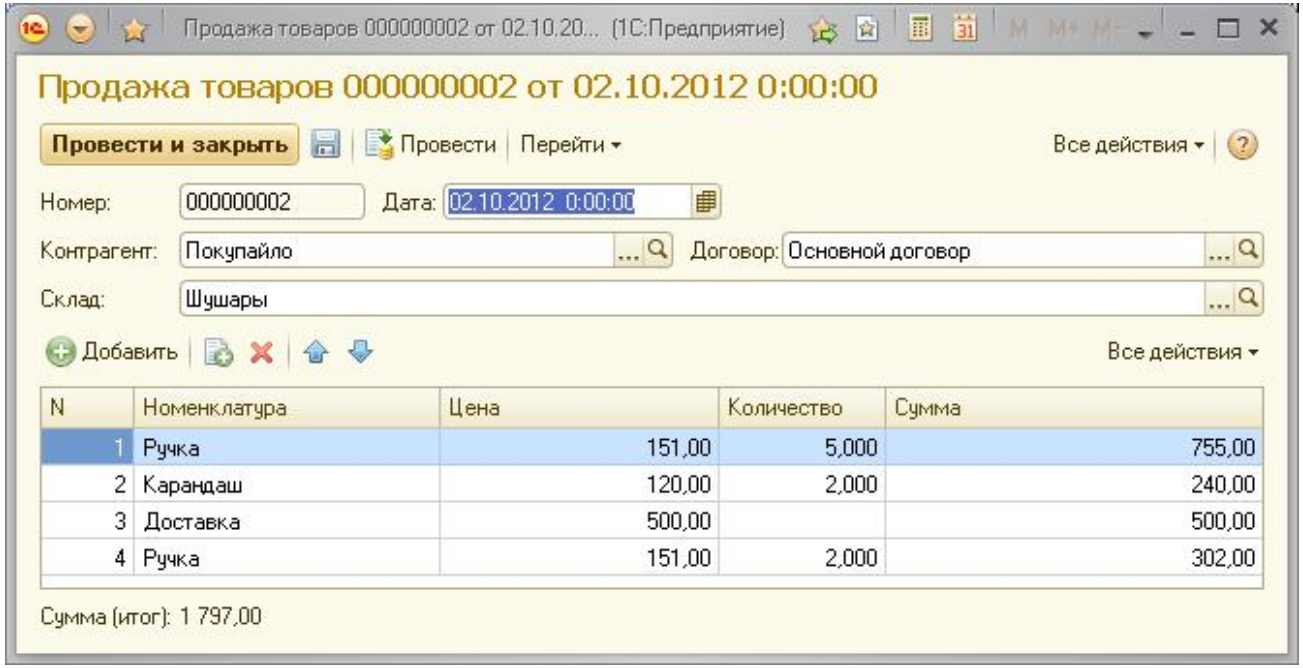


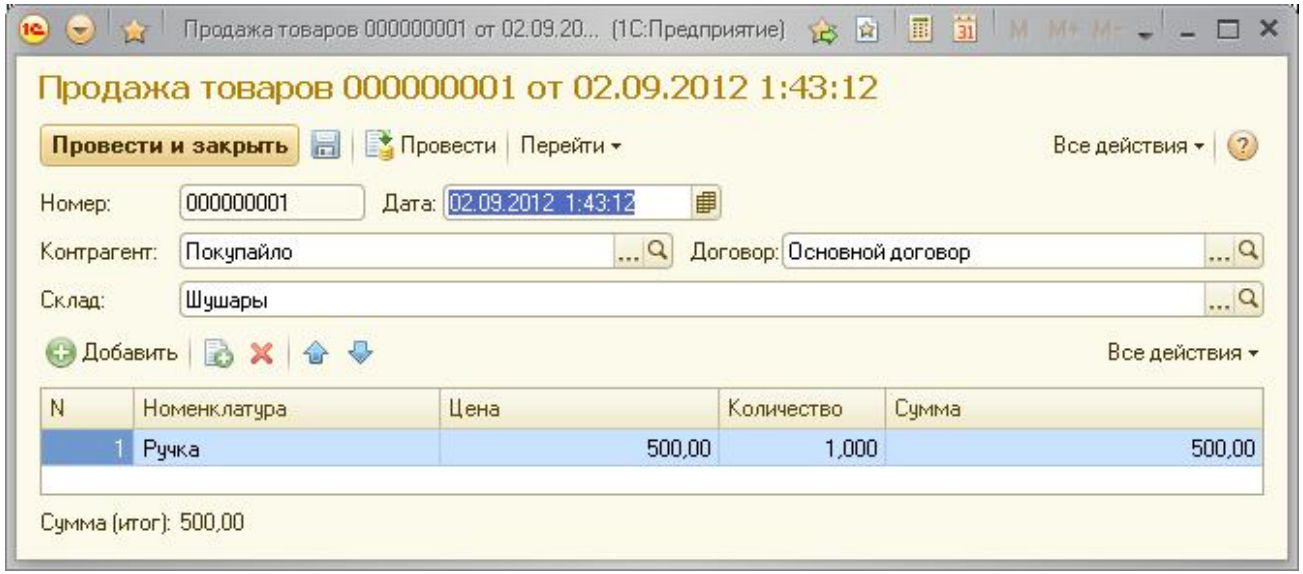


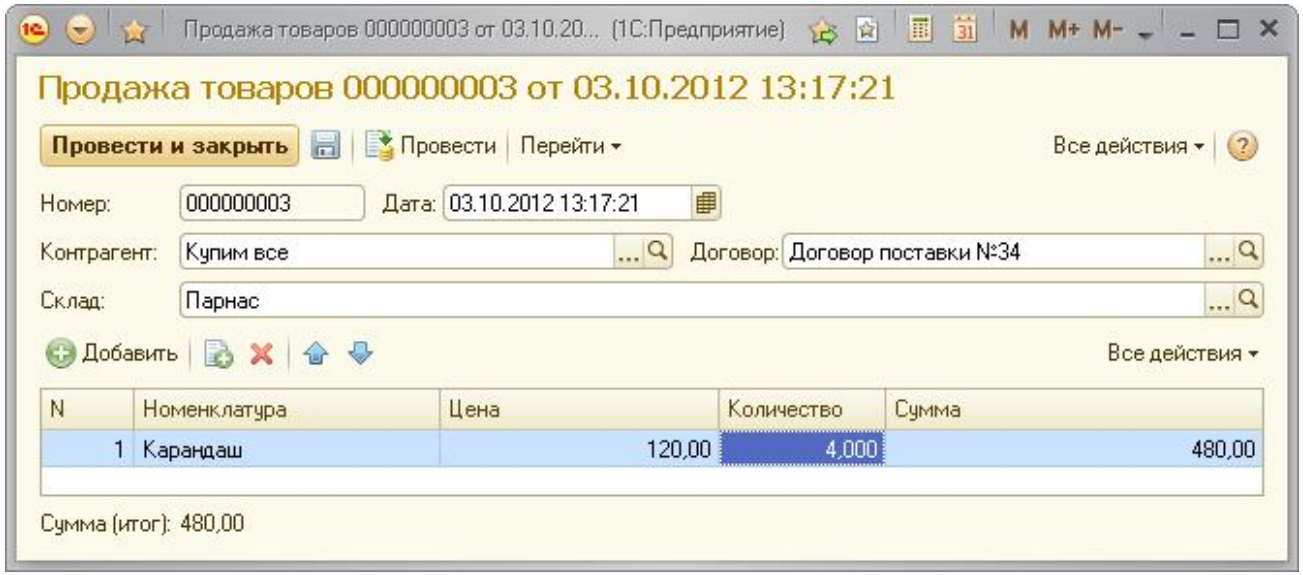

Поступление товаров 000000003 от 01.10.2012 0:26:16 - Кон... (1С:Предприятие)
Поступление товаров 000000003 от 01.10.2012 0:26:16
Провести и закрыть
Провести
Перейти
Все действия
Шапка
Номер: 000000003
Дата: 01.10.2012 0:26:16
Контрагент: СВЕТ ОАО
Договор: Договор №3
Склад: Шушары
Добавить
Все действия
N | Номенклатура | Цена | Количество | Сумма
1 | Карандаш | 300,00 | 8,000 | 2 400,00
2 | Ручка | 200,00 | 8,000 | 1 600,00
Сумма (итог): 4 000,00

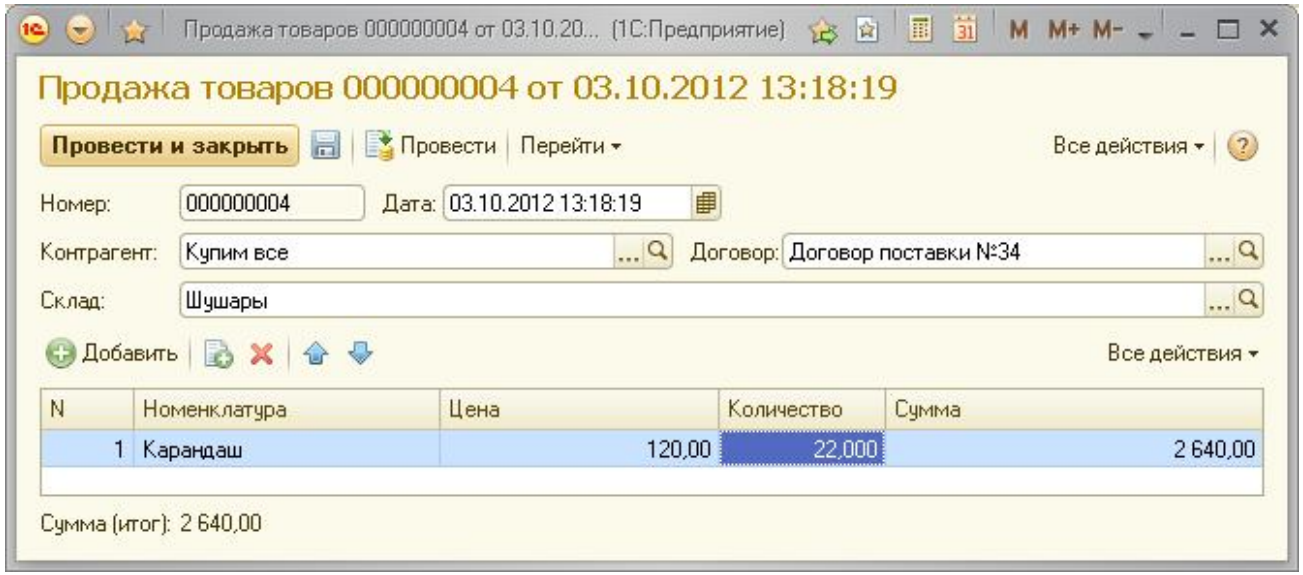

Поступление товаров 000000004 от 01.10.2012 0:26:45 - Кон... (1С:Предприятие)
Поступление товаров 000000004 от 01.10.2012 0:26:45
Провести и закрыть
Провести
Перейти
Все действия
Шапка
Номер: 000000004
Дата: 01.10.2012 0:26:45
Контрагент: СВЕТ ОАО
Договор: Договор №3
Склад: Шушары
Добавить
Все действия
N | Номенклатура | Цена | Количество | Сумма
1 | Карандаш | 500,00 | 8,000 | 4 000,00
2 | Ручка | 200,00 | 8,000 | 1 600,00
Сумма (итог): 5 600,00

Последовательно проведем документы Расходная:

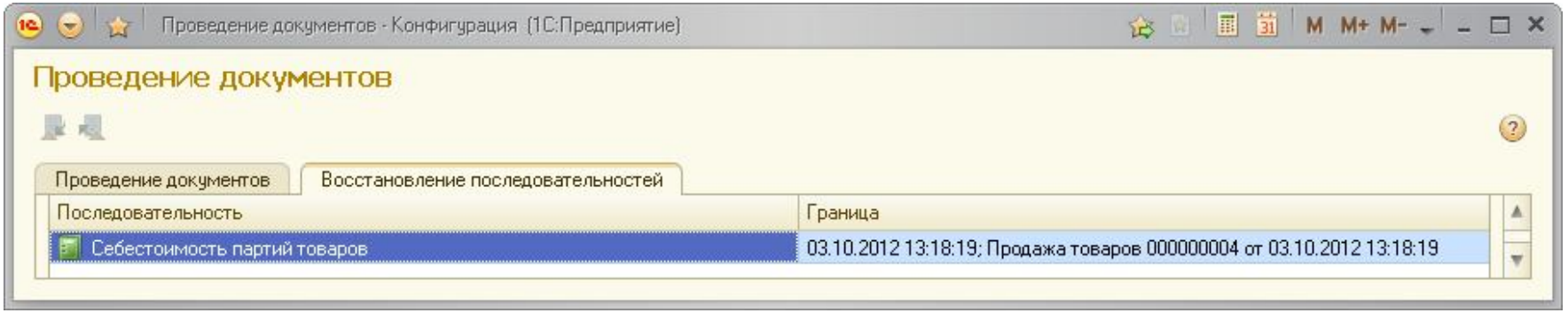




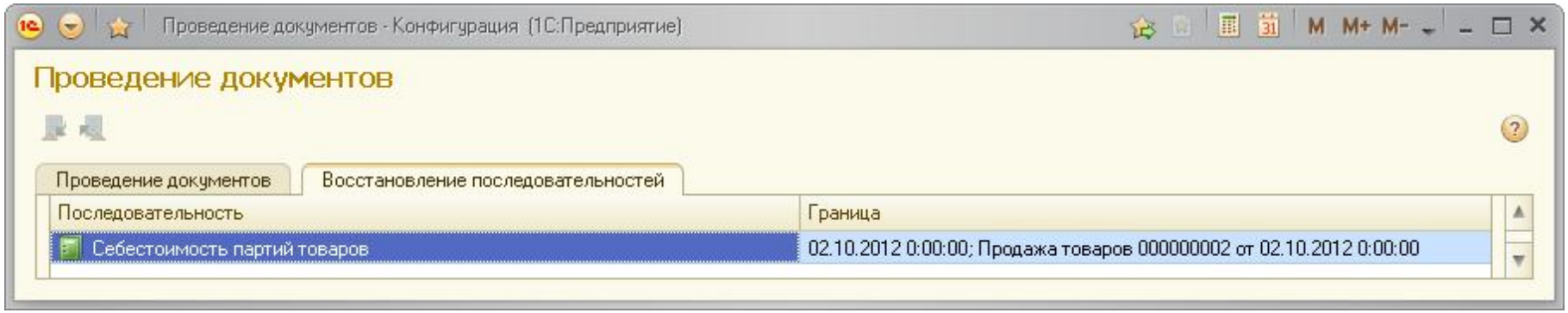

Продажа товаров 000000003 от 03.10.20... (1С:Предприятие)
Продажа товаров 000000003 от 03.10.2012 13:17:21
Провести и закрыть
Провести
Перейти
Все действия
Номер: 000000003
Дата: 03.10.2012 13:17:21
Контрагент: Купим все
Договор: Договор поставки №34
Склад: Парнас
Добавить
Все действия
N
Номенклатура
Цена
Количество
Сумма
1
Карандаш
120,00
4,000
480,00
Сумма (итог): 480,00


Продажа товаров 000000004 от 03.10.20... (1С:Предприятие)
Продажа товаров 000000004 от 03.10.2012 13:18:19
Провести и закрыть
Провести
Перейти
Все действия
Номер: 000000004
Дата: 03.10.2012 13:18:19
Контрагент: Купим все
Договор: Договор поставки №34
Склад: Шушары
Добавить
Все действия
N
Номенклатура
Цена
Количество
Сумма
1
Карандаш
120,00
22,000
2 640,00
Сумма (итог): 2 640,00

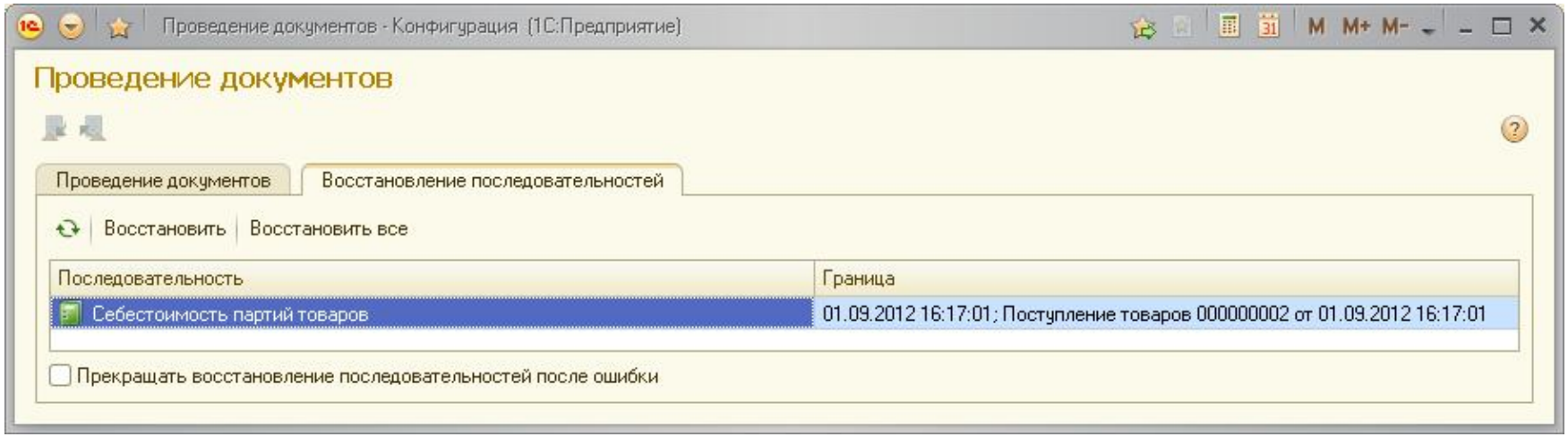
Последовательность должна установиться на документ Расходная №4 от 03.10.2012:



А теперь непоследовательно проведем Расходную №2 от 02.10.2012. При этом последовательность отодвинется назад, на нее:



А теперь, если изменить и перепровести любую Приходную, например №2 от 01.09.2012, то последовательность также отодвинется назад, на нее, причем только в том случае, если **Расходные вводятся последовательно**:



Если сейчас перепровести Расходную №4, то точка последовательности не изменится. А если начать последовательно проводить Расходные, то точка последовательности будет двигаться вместе с ними.

Кнопка «Восстановить» - автоматическое восстановление последовательности. Последовательно перепроводятся все документы Расходная и точка последовательности устанавливается на последнем.

Отчет по продажам

Вопрос: можно ли сейчас сделать отчет «Анализ продаж»:

Анализ продаж					
Контрагент	**Товар**	**Себестоимость продаж**	**Выручка**	**Количество**	**Прибыль**

Откуда берем поля:

Себестоимость продаж – из Регистра накопления «Остатки товаров», а именно, движения «Расход».
Выручка – из документа Расходная.
Количество - из документа Расходная.
Прибыль = Выручка – Себестоимость.

Себестоимость продаж надо отображать в разрезе Контрагента и Товара. В Регистре накопления «Остатки товаров» Контрагента можно взять из поля «Регистратор», т.к. Регистратор это и есть документ «Расходная». Но надо перебирать данные таблицы документа «Расходная».

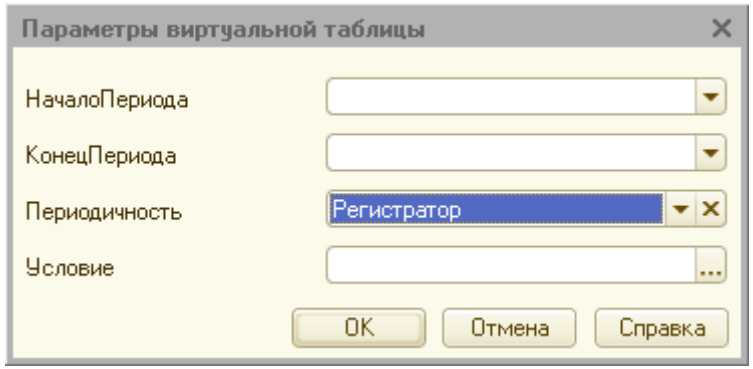
Создадим запрос в «Консоли запросов»:

Вытащим Себестоимость из Регистра накопления в разрезе Товаров и Контрагентов:

```
ВЫБРАТЬ
	*
ИЗ
	РегистрНакопления.ОстаткиТоваров.Обороты
```

Номенклатура	Партия	КоличествоОборот	КоличествоПриход	КоличествоРасход	СтоимостьОборот	СтоимостьПриход	СтоимостьРасход
Ручка	Поступление товаров 0	0	3	3	0	420	420
Карандаш	Поступление товаров 0	0	4	4	0	480	480
Ручка	Поступление товаров 0	3	8	5	600	1 600	1 000
Карандаш	Поступление товаров 0	0	8	8	0	1 200	1 200
Ручка	Поступление товаров 0	8	8	0	1 600	1 600	0
Ручка	Поступление товаров 0	8	8	0	1 600	1 600	0
Карандаш	Поступление товаров 0	0	8	8	0	2 400	2 400
Карандаш	Поступление товаров 0	0	8	8	0	4 000	4 000

По умолчанию таблица Оборотов разворачивается только по Измерениям, которые есть в регистре. Поэтому в таблице нет поля «Регистратор». Но «Регистратор» можно указать как параметр «Периодичность» виртуальной таблицы, т.е. дополнительный период, в разрезе которого надо получать данные. В Конструкторе запросов форма для задания параметров данной виртуальной таблицы выглядит так:




```
ВЫБРАТЬ
	*
ИЗ
	РегистрНакопления.ОстаткиТоваров.Обороты( , , Регистратор)
```

Номенклатура	Партия	Период	Регистратор	КоличествоОборот	КоличествоПриход	КоличествоРасход	СтоимостьОборот	СтоимостьПриход	СтоимостьРасход
Карандаш	Поступление товаро	03.10.2012 13:17:21	Продажа товаров 00	-2	0	2	-300	0	300
Карандаш	Поступление товаро	03.10.2012 13:17:21	Продажа товаров 00	-2	0	2	-240	0	240
Ручка	Поступление товаро	01.10.2012 0:26:16	Поступление товаро	8	8	0	1 600	1 600	0
Карандаш	Поступление товаро	01.10.2012 0:26:16	Поступление товаро	8	8	0	2 400	2 400	0
Ручка	Поступление товаро	01.09.2012 16:17:01	Поступление товаро	8	8	0	1 600	1 600	0
Карандаш	Поступление товаро	01.09.2012 16:17:01	Поступление товаро	8	8	0	1 200	1 200	0
Карандаш	Поступление товаро	03.10.2012 13:18:19	Продажа товаров 00	-6	0	6	-900	0	900
Карандаш	Поступление товаро	03.10.2012 13:18:19	Продажа товаров 00	-8	0	8	-2 400	0	2 400
Карандаш	Поступление товаро	03.10.2012 13:18:19	Продажа товаров 00	-8	0	8	-4 000	0	4 000
Ручка	Поступление товаро	22.08.2012 0:15:24	Поступление товаро	3	3	0	420	420	0
Карандаш	Поступление товаро	22.08.2012 0:15:24	Поступление товаро	4	4	0	480	480	0
Ручка	Поступление товаро	02.10.2012 0:00:00	Продажа товаров 00	-5	0	5	-1 000	0	1 000
Ручка	Поступление товаро	02.09.2012 1:43:12	Продажа товаров 00	-1	0	1	-140	0	140
Ручка	Поступление товаро	02.10.2012 0:00:00	Продажа товаров 00	-2	0	2	-280	0	280
Карандаш	Поступление товаро	02.10.2012 0:00:00	Продажа товаров 00	-2	0	2	-240	0	240
Ручка	Поступление товаро	01.10.2012 0:26:45	Поступление товаро	8	8	0	1 600	1 600	0
Карандаш	Поступление товаро	01.10.2012 0:26:45	Поступление товаро	8	8	0	4 000	4 000	0

Выберем нужные поля:

```
ВЫБРАТЬ
	Номенклатура,
	Регистратор.Контрагент,
	СтоимостьРасход,
```

```
    КоличествоРасход
ИЗ
    РегистрНакопления.ОстаткиТоваров.Обороты( , , Регистратор)
```

Номенклатура	РегистраторКонтраг	СтоимостьРасход	КоличествоРасход
Ручка	Покупайло	140	1
Карандаш	Купим все	7 300	22
Карандаш	Купим все	540	4
Ручка	Покупайло	1 280	7
Карандаш	Покупайло	240	2

Тип значения поля «Регистратор» в регистре составной: «Приходная» и «Расходная» и в таблицу запроса сейчас включены оба эти вида документа, а это лишняя таблица. Кроме того, нет гарантии, что в обоих этих видах документов есть поле «Контрагент». Т.е. запрос не оптимален.

Опишем второй запрос – получим выручку:

```
ВЫБРАТЬ
    *
ИЗ
    Документ.Расходная.Товары
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров	2	Карандаш	120	2	240
Продажа товаров	1	Ручка	151	5	755
Продажа товаров	3	Доставка	500	0	500
Продажа товаров	4	Ручка	151	2	302
Продажа товаров	1	Карандаш	120	22	2 640
Продажа товаров	1	Карандаш	120	4	480
Продажа товаров	1	Ручка	500	1	500

Выберем нужные поля. Добавим условие – только проведенные документы. Сгруппируем поля.

```
ВЫБРАТЬ
    Номенклатура,
```

```
      Ссылка.Контрагент,
      СУММА(Сумма) КАК Выручка
ИЗ
      Документ.Расходная.Товары
ГДЕ
      Ссылка.Проведен
СГРУППИРОВАТЬ ПО Номенклатура, Ссылка.Контрагент
```

Номенклатура	Контрагент	Выручка
Ручка	Покупайло	1 557
Карандаш	Покупайло	240
Доставка	Покупайло	500
Карандаш	Купим все	3 120

Теперь склеим эти два запроса:

```
ВЫБРАТЬ
      Номенклатура,
      Ссылка.Контрагент КАК Контрагент,
      СУММА(Сумма) КАК Выручка
ПОМЕСТИТЬ Продажи
ИЗ
      Документ.Расходная.Товары
ГДЕ
      Ссылка.Проведен
СГРУППИРОВАТЬ ПО Номенклатура, Ссылка.Контрагент
;
ВЫБРАТЬ
      Номенклатура,
      Регистратор.Контрагент КАК Контрагент,
      СтоимостьРасход,
      КоличествоРасход
ПОМЕСТИТЬ СебестоимостьПродаж
ИЗ
      РегистрНакопления.ОстаткиТоваров.Обороты( , , Регистратор)
```

```
ГДЕ
	Регистратор ССЫЛКА Документ.Расходная
;
ВЫБРАТЬ
	*
ИЗ
	Продажи КАК Продажи
ЛЕВОЕ СОЕДИНЕНИЕ
	СебестоимостьПродаж КАК СебестоимостьПродаж
ПО
	Продажи.Номенклатура = СебестоимостьПродаж.Номенклатура И
Продажи.Контрагент = СебестоимостьПродаж.Контрагент
```

«`Регистратор` **`ССЫЛКА`** `Документ.Расходная`» - условие на приведение к типу. Оператор «ССЫЛКА» позволяет проверить, является ли значение выражения, указанного справа от него, ссылкой на таблицу, указанную слева. Если да – результатом оператора будет ИСТИНА, иначе – ЛОЖЬ.
Правильно это условие ставить до того, как получаем данные по Контрагенту. В данном случае это можно сделать вложенным запросом или еще одним пакетом.

Запрос пакета 1

Запрос пакета 2

Запрос пакета 3

Номенклатура	Контрагент	Выручка	Номенклатура1	Контрагент1	СтоимостьРасход	КоличествоРасход
Ручка	Покупайло	1 557	Ручка	Покупайло	140	1
Ручка	Покупайло	1 557	Ручка	Покупайло	1 280	7
Карандаш	Покупайло	240	Карандаш	Покупайло	240	2
Доставка	Покупайло	500	NULL	NULL	NULL	NULL
Карандаш	Купим все	3 120	Карандаш	Купим все	7 300	22
Карандаш	Купим все	3 120	Карандаш	Купим все	540	4

Выберем нужные поля:

```
ВЫБРАТЬ
```

```
     Номенклатура,
     Ссылка.Контрагент КАК Контрагент,
     СУММА(Сумма) КАК Выручка
ПОМЕСТИТЬ Продажи
ИЗ
     Документ.Расходная.Товары
ГДЕ
     Ссылка.Проведен
СГРУППИРОВАТЬ ПО Номенклатура, Ссылка.Контрагент
;
ВЫБРАТЬ
     Номенклатура,
     Регистратор.Контрагент КАК Контрагент,
     СтоимостьРасход,
     КоличествоРасход
ПОМЕСТИТЬ СебестоимостьПродаж
ИЗ
     РегистрНакопления.ОстаткиТоваров.Обороты( , , Регистратор)
ГДЕ
     Регистратор ССЫЛКА Документ.Расходная
;
ВЫБРАТЬ
     Продажи.Номенклатура,
     Продажи.Контрагент,
     Выручка,
     СтоимостьРасход КАК Себестоимость,
     КоличествоРасход КАК Количество
ИЗ
     Продажи КАК Продажи
ЛЕВОЕ СОЕДИНЕНИЕ
     СебестоимостьПродаж КАК СебестоимостьПродаж
ПО
     Продажи.Номенклатура = СебестоимостьПродаж.Номенклатура И
Продажи.Контрагент = СебестоимостьПродаж.Контрагент
```

Запрос пакета 1

Запрос пакета 2

Запрос пакета 3

Номенклатура	Контрагент	Выручка	Себестоимость	Количество
Ручка	Покупайло	1 557	140	1
Ручка	Покупайло	1 557	1 280	7
Карандаш	Покупайло	240	240	2
Доставка	Покупайло	500	NULL	NULL
Карандаш	Купим все	3 120	7 300	22
Карандаш	Купим все	3 120	540	4

Минусы, кроме того, что запрос будет выполняться сравнительно медленно:

- Ориентируемся на физическую структуру таблицы документа «Расходная» - обращаемся к определенным полям. Но конфигурация может изменяться, может добавиться еще один документ по продаже товаров, который этот отчет учитывать не будет, надо будет переписывать отчет.
- «Себестоимость» и «Количество» в регистре накопления «Остатки товаров» не обязательно будут показывать только результат Продаж. Это может быть Списание, Передача в переработку и т.д.

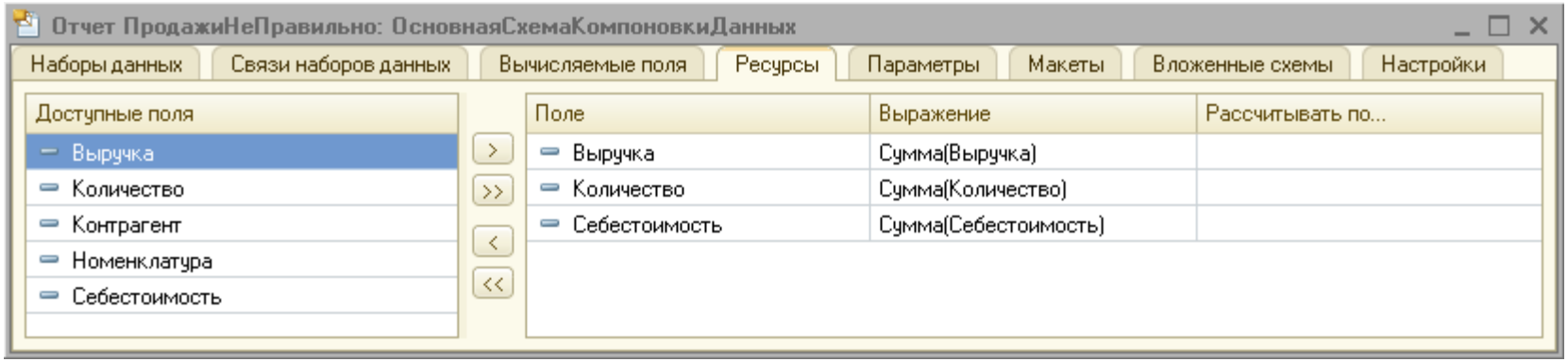
Вывод:
Сейчас в БД нигде не хранится показатель продаж. В запросе мы его сымитировали.

Добавим запрос в конфигурацию в виде отчета «Продажи не правильно»:

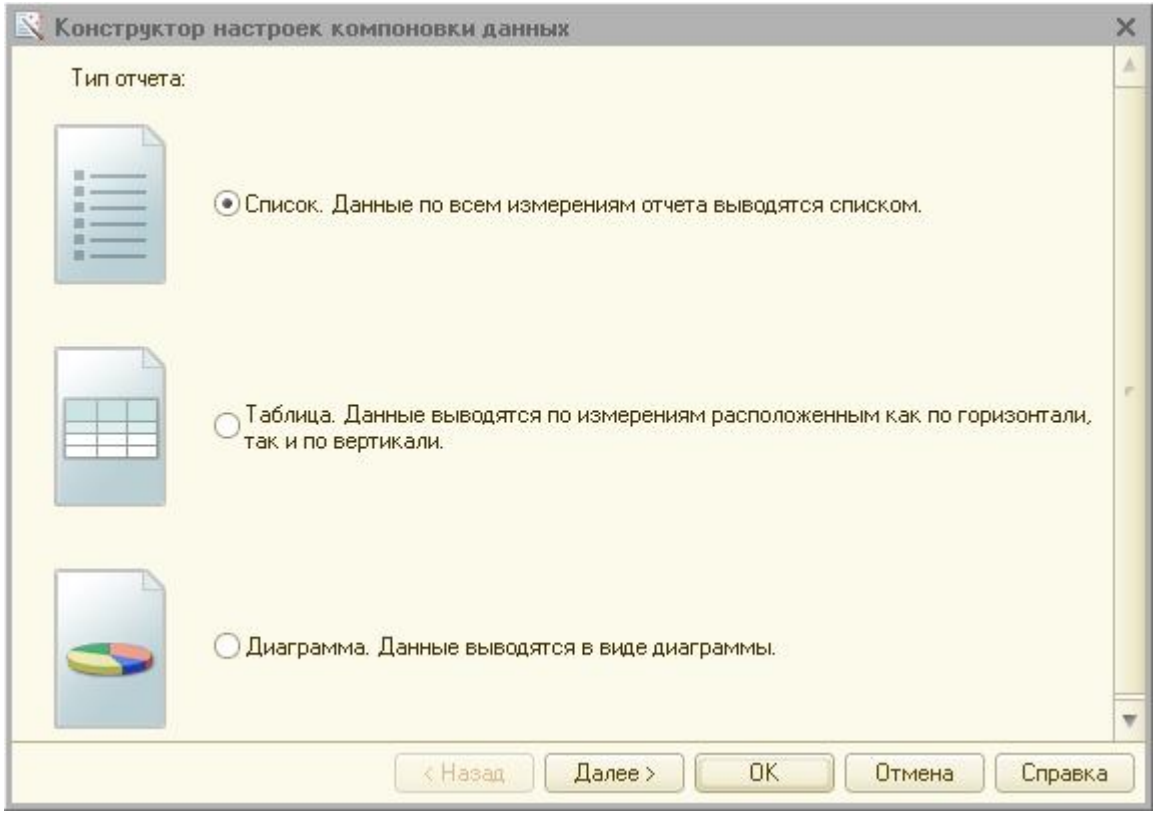
Свойство	Значение
Имя	ПродажиНеПравильно
Синоним	Продажи не правильно

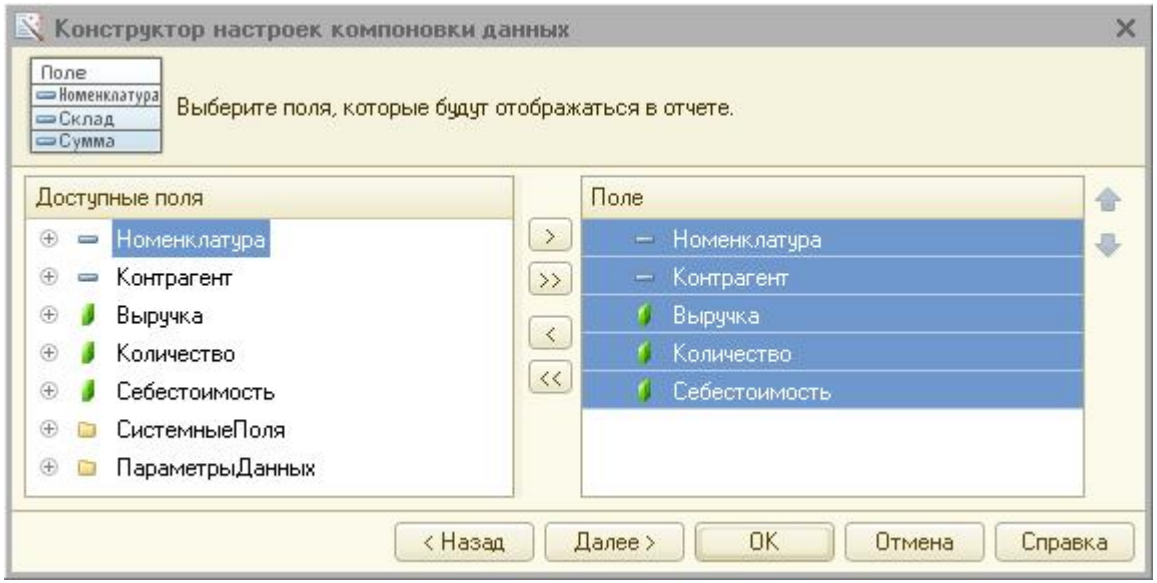
Откроем СКД, добавим «Набор данных-запрос», откроем «Конструктор запроса» и вставим туда наш запрос.

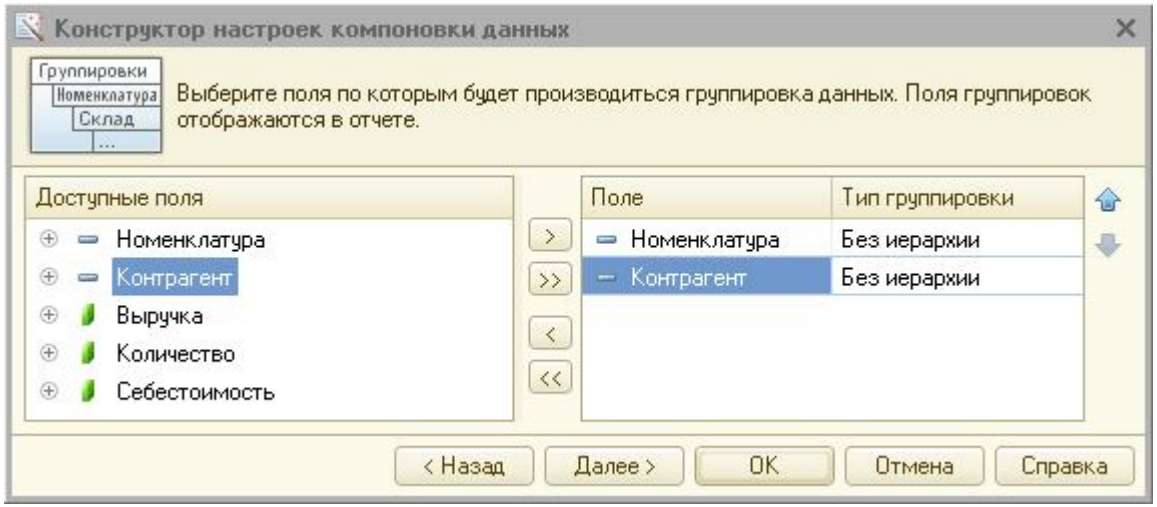
Закладка «Ресурсы» СКД:



На закладке «Настройки» сделаем настройку по умолчанию:







Показатель продаж будем хранить в регистре накопления, причем только ОБОРОТОВ (остатки не хранятся, накапливаются только обороты).

В Регистре накопления оборотов можно создать АГРЕГАТЫ – указать, в каких разрезах и за какие периоды чаще всего будем получать данные. Например, в нашем примере:

В регистре накопления по продажам будет два измерения:

- Контрагент.
- Номенклатура.

Можно добавить измерение «Менеджер», чтобы отслеживать, сколько менеджер продал товаров. На основании этих данных можно начислять ему зарплату. А зарплату можно начислять раз в месяц.

С помощью Агрегатов можно задать, чтобы по Менеджеру обороты накапливались за месяц, а по Продажам за день. Т.е. для каждого Измерения/Сочетания измерений период накапливания оборотов можно назначить отдельно.

Создадим регистр накопления «Продажи»:

Свойство	**Значение**
Имя	Продажи
Синоним	Продажи
Вид регистра	Обороты
Закладка «Данные»	
Измерения	
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Запрет незаполненных значений	Истина
Использование в итогах	Истина
Измерения	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	СправочникСсылка.Номенклатура
Запрет незаполненных значений	Истина
Использование в итогах	Истина

Если «Использование в итогах = Ложь», то расчета оборотов по этому измерению не будет и при запросе система будет анализировать реальную таблицу, что более трудозатратно.

Свойство	Значение
Закладка «Данные»	
Ресурсы	
Себестоимость	
Имя	Себестоимость
Синоним	Себестоимость
Тип	Число 15, 2
Выручка	
Имя	Выручка
Синоним	Выручка
Тип	Число 15, 2
Количество	
Имя	Количество
Синоним	Количество
Тип	Число 12, 3
Закладка «Регистраторы»	
Регистраторы	Расходная

Движения по регистру накопления «Продажи» будем формировать в документе «Расходная»:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

    Движения.ОстаткиТоваров.Записывать   = Истина;
    Движения.ОстаткиПоСкладам.Записывать = Истина;
    Движения.Продажи.Записывать          = Истина;

    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    РасходнаяТовары.Номенклатура,
    |    СУММА(РасходнаяТовары.Количество) КАК Количество,
    |    СУММА(Сумма) КАК Выручка,
    |    МИНИМУМ(РасходнаяТовары.НомерСтроки) КАК НомерСтроки
```

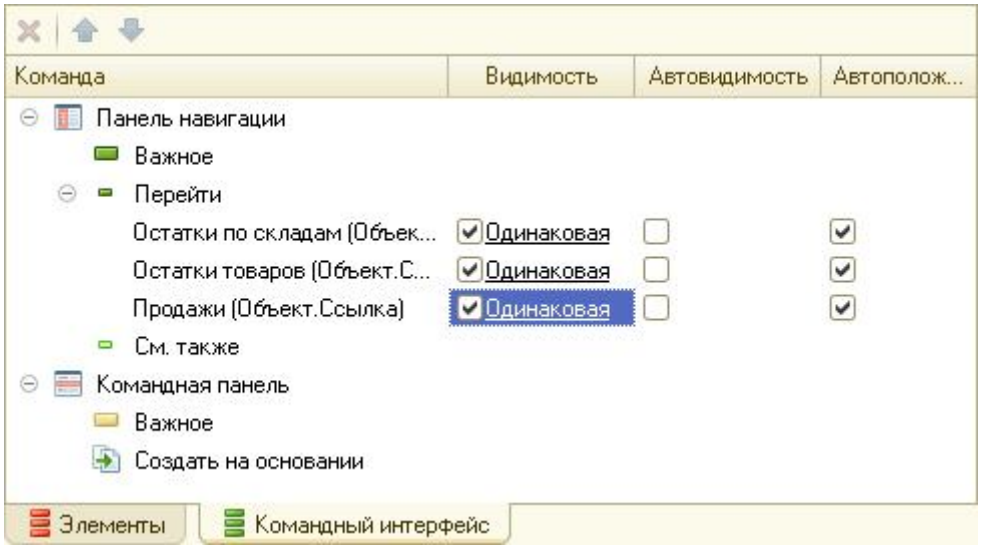
```
|ПОМЕСТИТЬ ДокТЧ
|ИЗ
|    Документ.Расходная.Товары КАК РасходнаяТовары
|ГДЕ
|    РасходнаяТовары.Ссылка = &Ссылка
|    И НЕ РасходнаяТовары.Номенклатура.Услуга

|СГРУППИРОВАТЬ ПО
|    РасходнаяТовары.Номенклатура
|;
|ВЫБРАТЬ
|    ДокТЧ.НомерСтроки,
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Выручка,
|    ДокТЧ.Количество,
|    Остатки.Партия,
|    ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
|    ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
|    ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
|ИЗ
|    ДокТЧ КАК ДокТЧ
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки(
|                &МоментВремени,
|                Номенклатура В
|                    (ВЫБРАТЬ
|                        ДокТЧ.Номенклатура
|                    ИЗ
|                        ДокТЧ КАК ДокТЧ)) КАК Остатки
|        ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиПоСкладам.Остатки(
|                &МоментВремени,
|                Склад = &Склад
|                    И Номенклатура В
|                        (ВЫБРАТЬ
|                            ДокТЧ.Номенклатура
|                        ИЗ
```

```
    |                                  ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
    |         ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
    |УПОРЯДОЧИТЬ ПО Партия.Дата ВОЗР
    |ИТОГИ МИНИМУМ(Количество), МИНИМУМ(Выручка), МИНИМУМ(ОстатокНаСкладе), МИНИМУМ(НомерСтроки)
    |ПО ДокТЧ.Номенклатура";

    Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
    Запрос.УстановитьПараметр("Ссылка", Ссылка);
    Запрос.УстановитьПараметр("Склад", Склад);

    Результат = Запрос.Выполнить();

    Выборка = Результат.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

    Пока Выборка.Следующий() Цикл
        Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
            Сообщение = Новый СообщениеПользователю;
            Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура + """ из необходимых " +
Выборка.Количество + " в наличии осталось только " + Выборка.ОстатокНаСкладе;
            Сообщение.Поле = "Товары[" + (Выборка.НомерСтроки - 1) + "].Количество";
            Сообщение.УстановитьДанные(ЭтотОбъект);
            Сообщение.Сообщить();
            Отказ = Истина;
            Движения.ОстаткиТоваров.Записывать   = Ложь;
            Движения.ОстаткиПоСкладам.Записывать = Ложь;
        КонецЕсли;
    КонецЦикла;

    Если Отказ Тогда
        Возврат;
    КонецЕсли;

    Выборка.Сбросить();

    Пока Выборка.Следующий() Цикл
```

```
    ОсталосьСписать = Выборка.Количество;

    СебестоимостьИтого = 0;

    ВыборкаПартии = Выборка.Выбрать();
    Пока ВыборкаПартии.Следующий() И ОсталосьСписать <> 0 Цикл

        Списать       = Мин(ОсталосьСписать, ВыборкаПартии.КоличествоОстаток);
        Себестоимость = Списать / ВыборкаПартии.КоличествоОстаток * ВыборкаПартии.СтоимостьОстаток;

        СебестоимостьИтого = СебестоимостьИтого + Себестоимость;

        Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
        Движение.Период       = Дата;
        Движение.Номенклатура = Выборка.Номенклатура;
        Движение.Количество   = Списать;
        Движение.Стоимость    = Себестоимость;
        Движение.Партия       = ВыборкаПартии.Партия;

        ОсталосьСписать = ОсталосьСписать - Списать;
    КонецЦикла;

    Движение = Движения.ОстаткиПоСкладам.ДобавитьРасход();
    Движение.Период       = Дата;
    Движение.Номенклатура = Выборка.Номенклатура;
    Движение.Количество   = Выборка.Количество;
    Движение.Склад        = Склад;

    Движение = Движения.Продажи.Добавить();
    Движение.Период        = Дата;
    Движение.Номенклатура  = Выборка.Номенклатура;
    Движение.Количество    = Выборка.Количество;
    Движение.Выручка       = Выборка.Выручка;
    Движение.Себестоимость = СебестоимостьИтого;
    Движение.Контаргент    = Контрагент;
КонецЦикла;
```

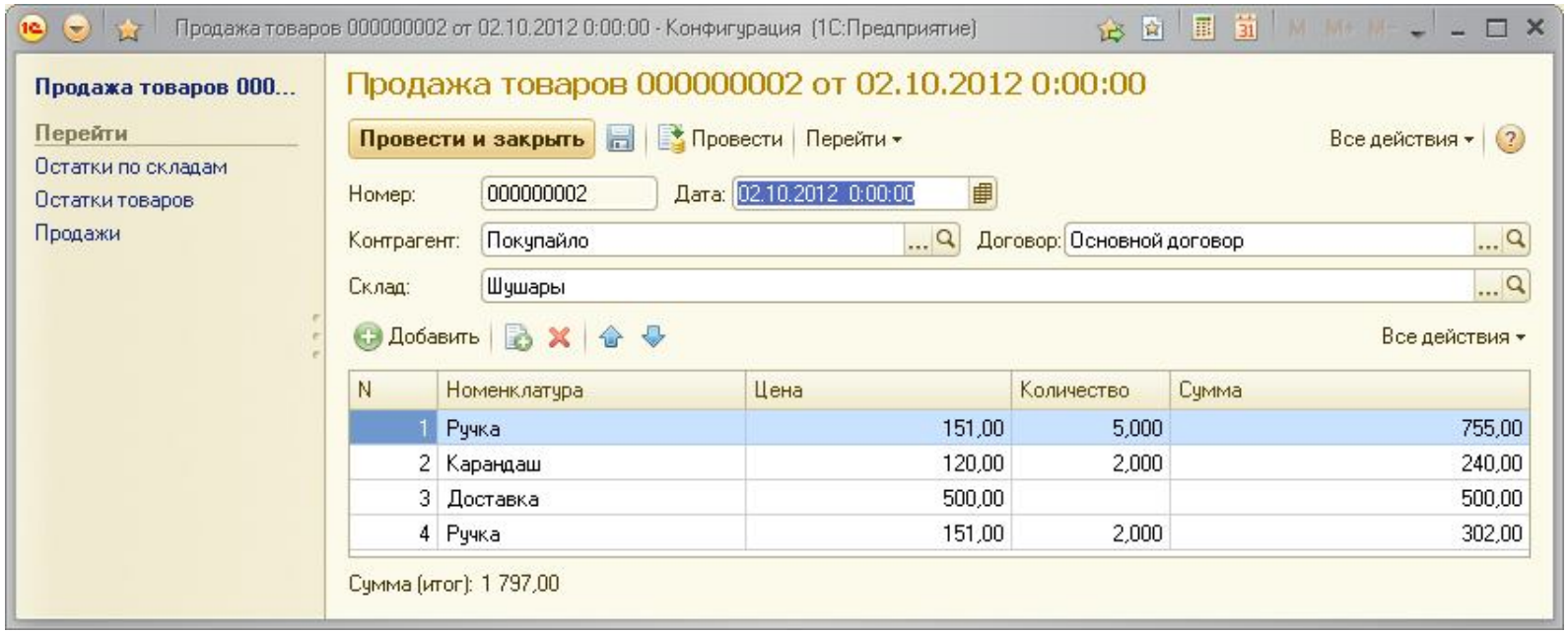
```
КонецПроцедуры
```

В форме документа «Расходная» перейдем на закладку «Командный интерфейс» и сделаем видимыми команды перехода к регистрам накопления:



В режиме исполнения:

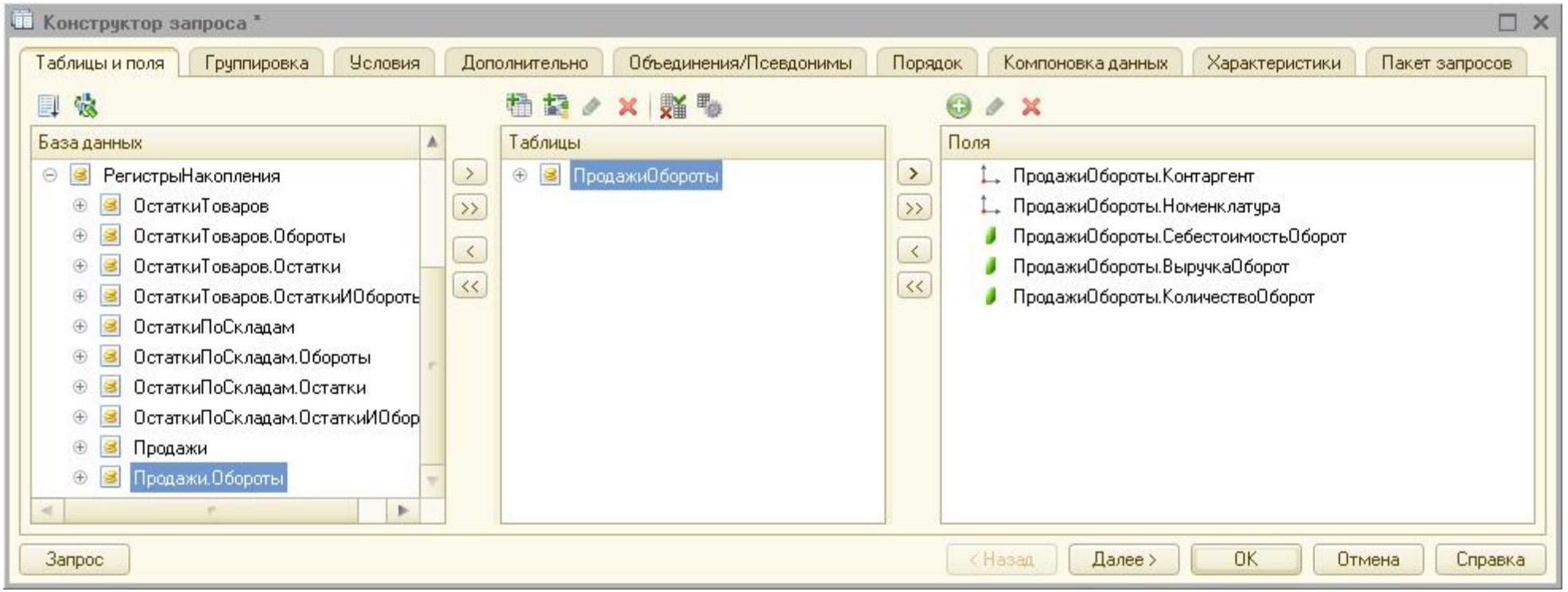
Перепроведем все документы «Расходная» и посмотрим «Продажи».
Например, Расходная №2:



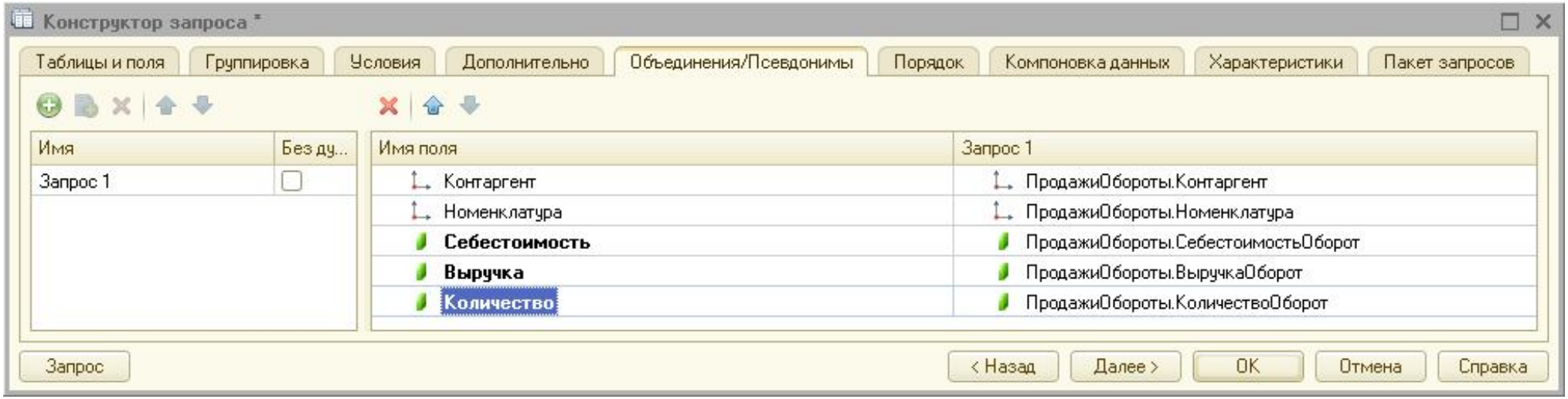
Теперь построим отчет «Продажи» оптимально:

Свойство	Значение
Имя	ПродажиТакКакНадо
Синоним	Продажи так как надо

Откроем СКД, добавим «Набор данных-запрос» и откроем «Конструктор запросов»:

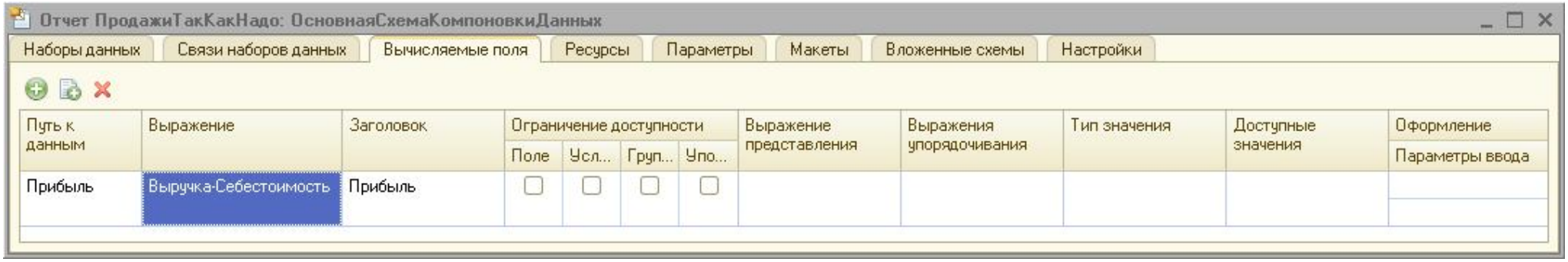


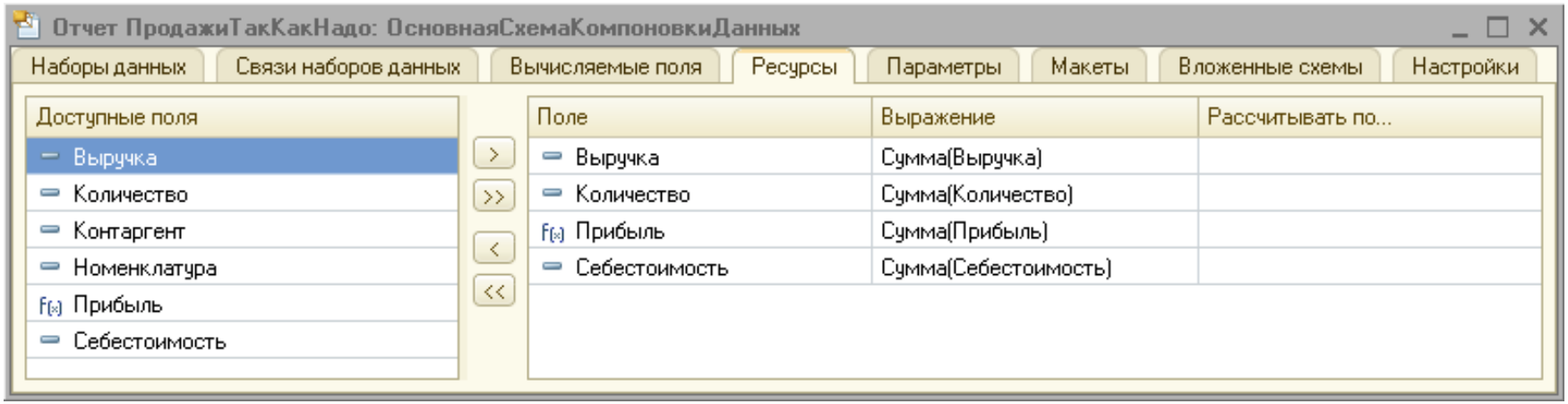
Переименуем поля (убрали слово «Оборот»):



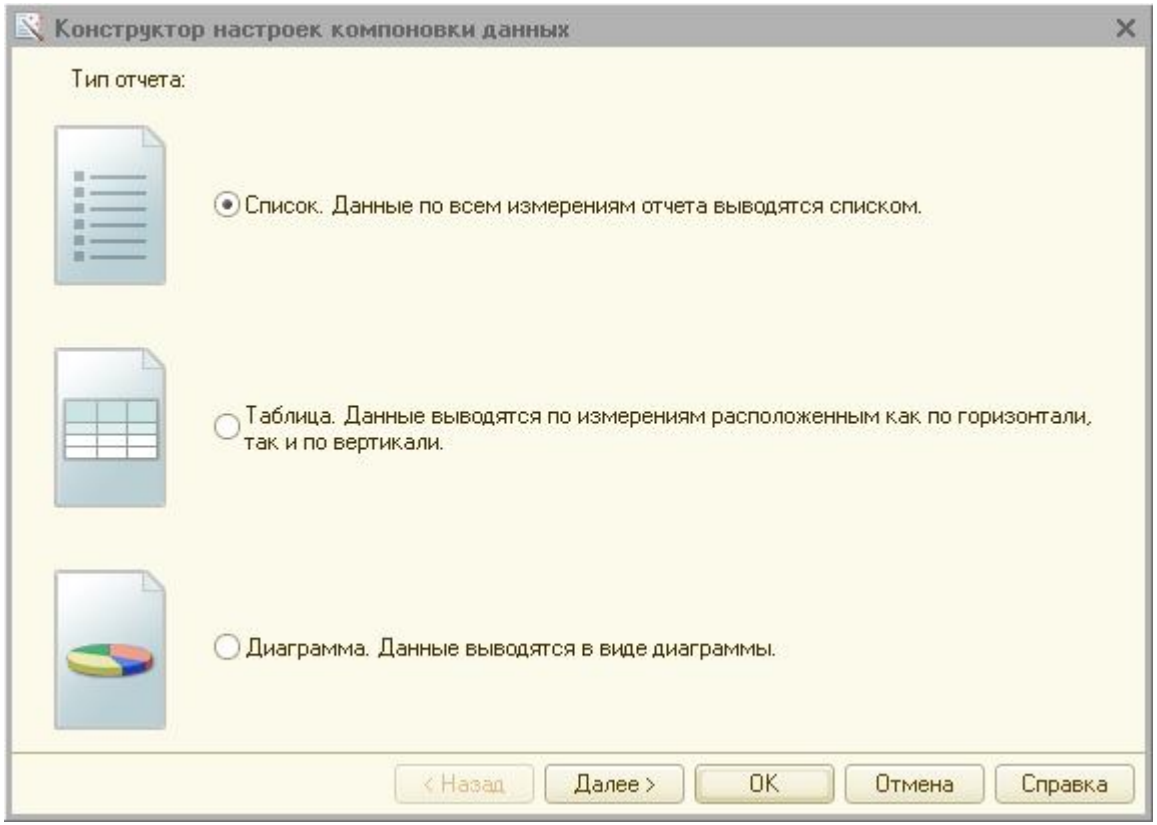
Нажимаем «ОК» и переходим опять в СКД:

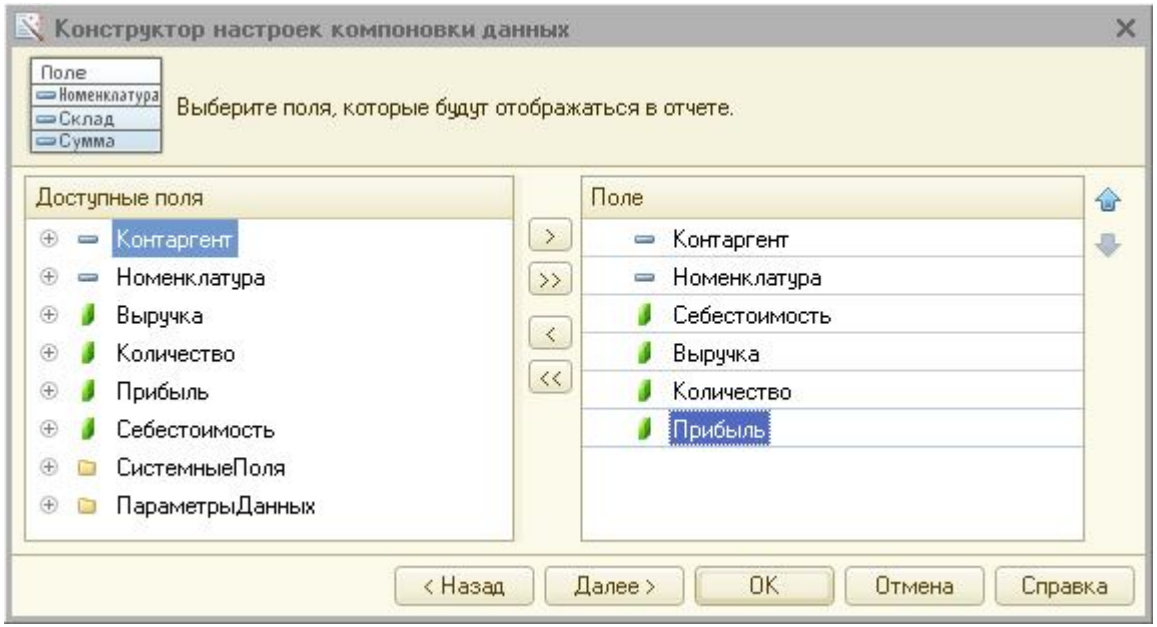
Опишем вычисляемое поле «Прибыль»:

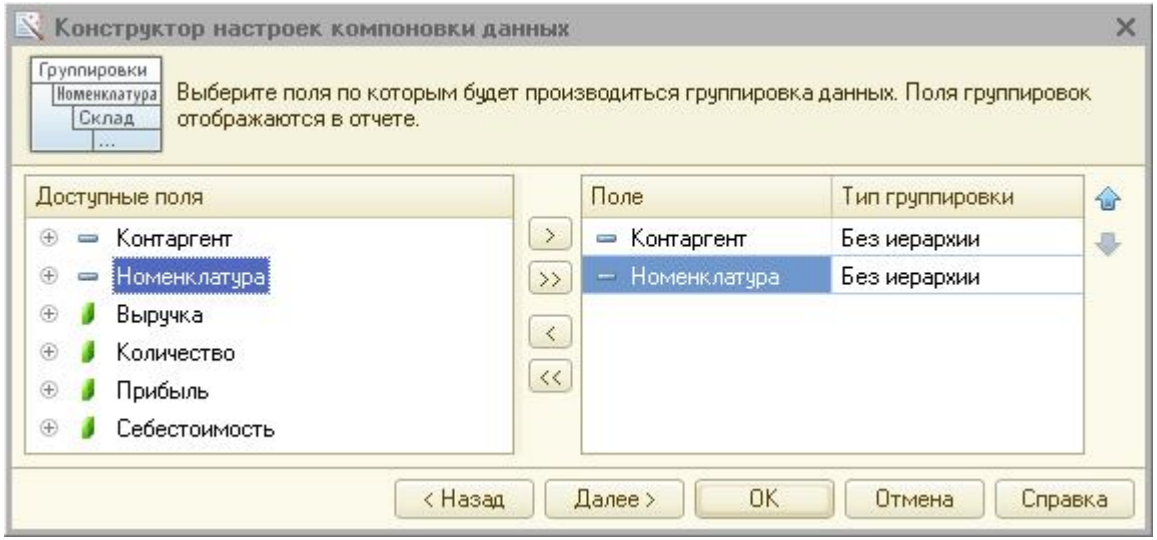




Закладка «Настройки» - сделаем настройку по умолчанию:


Конструктор настроек компоновки данных
Тип отчета:
Список. Данные по всем измерениям отчета выводятся списком.
Таблица. Данные выводятся по измерениям расположенным как по горизонтали, так и по вертикали.
Диаграмма. Данные выводятся в виде диаграммы.
< Назад
Далее >
OK
Отмена
Справка


Конструктор настроек компоновки данных
Поле
Номенклатура
Склад
Сумма
Выберите поля, которые будут отображаться в отчете.
Доступные поля
Контрагент
Номенклатура
Выручка
Количество
Прибыль
Себестоимость
СистемныеПоля
ПараметрыДанных
Поле
Контрагент
Номенклатура
Себестоимость
Выручка
Количество
Прибыль
< Назад
Далее >
OK
Отмена
Справка



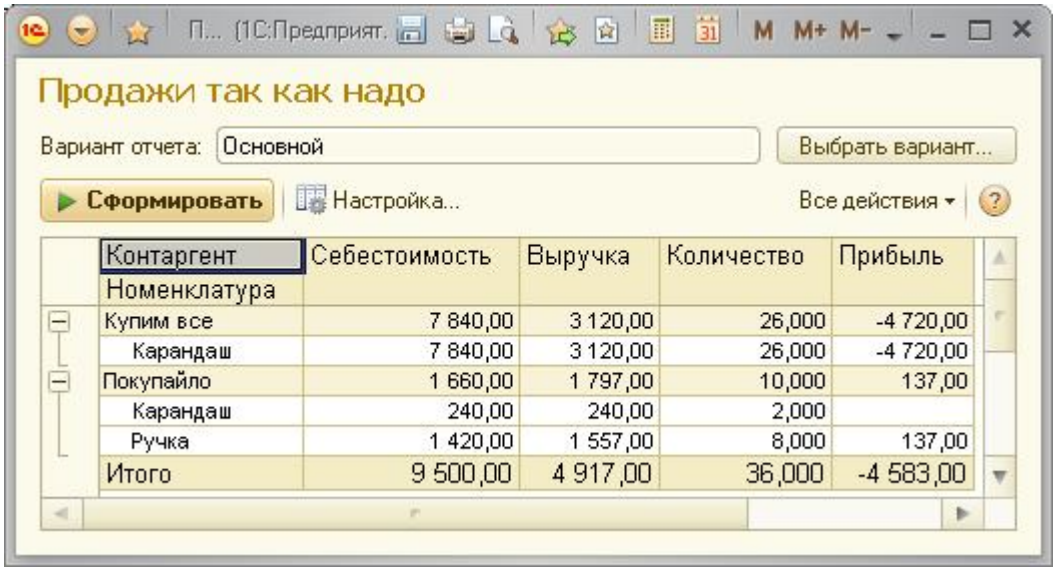
Отчет готов.

Посмотрим сам запрос:

```
ВЫБРАТЬ
      ПродажиОбороты.Контаргент,
      ПродажиОбороты.Номенклатура,
      ПродажиОбороты.СебестоимостьОборот КАК Себестоимость,
      ПродажиОбороты.ВыручкаОборот КАК Выручка,
      ПродажиОбороты.КоличествоОборот КАК Количество
ИЗ
      РегистрНакопления.Продажи.Обороты КАК ПродажиОбороты
```

Объем текста запроса намного меньше, используем только одну таблицу (виртуальную – механизм, группирующий данные автоматически). Виртуальная таблица обращается к служебным таблицам БД, использующим рассчитанные показатели в основных итогах или агрегатах. И главное, что в регистре накопления «Продажи» хранятся именно продажи, т.е. не надо имитировать показатели продаж.

В режиме исполнения:

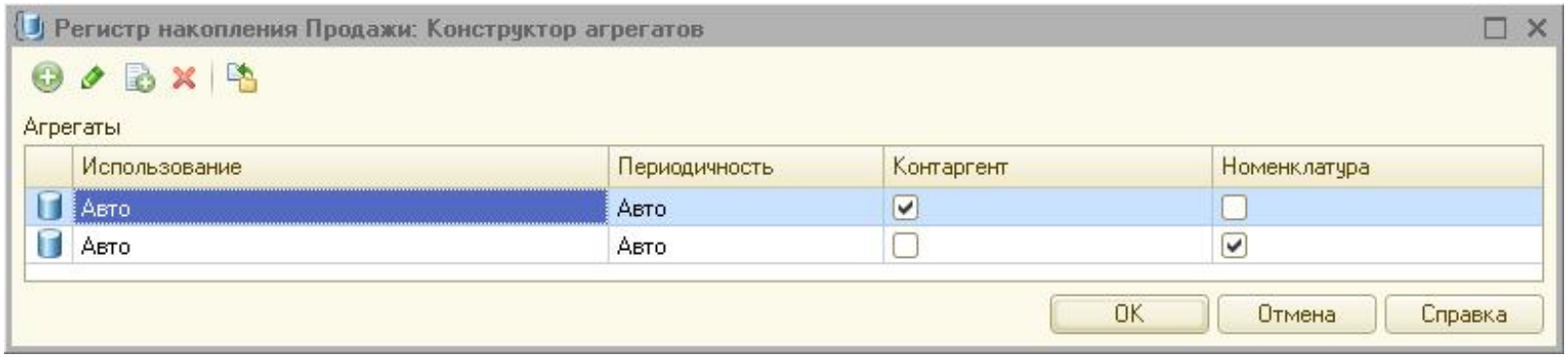


Контрагент / Номенклатура	Себестоимость	Выручка	Количество	Прибыль
Купим все	7 840,00	3 120,00	26,000	-4 720,00
Карандаш	7 840,00	3 120,00	26,000	-4 720,00
Покупайло	1 660,00	1 797,00	10,000	137,00
Карандаш	240,00	240,00	2,000	
Ручка	1 420,00	1 557,00	8,000	137,00
Итого	9 500,00	4 917,00	36,000	-4 583,00

Агрегаты:

Находятся в регистре накопления, на закладке «Данные». Только если вид регистра – «Обороты».

В регистре накопления «Продажи» построим агрегаты:



Использование	Периодичность	Контрагент	Номенклатура
Авто	Авто	☑	☐
Авто	Авто	☐	☑

«Периодичность - Авто» - накопление и периодичность агрегата устанавливается системой автоматом, исходя из количества и периодичности обращений к регистру.

Агрегаты используются для того, чтобы иметь рассчитанные служебные данные для быстрого построения отчетов.

Занятие 1.15

Для логического разделения интерфейса используют «Подсистемы». Создадим три подсистемы:

Подсистема «Закупки»

Свойство	Значение
Имя	Закупки
Синоним	Закупки
Закладка «Состав»	
Входящие в подсистему объекты	Справочники: ➢ Контрагенты ➢ Договоры Документы: ➢ Приходная Отчеты: ➢ ЗакупкиПоДокументам ➢ ЗакупкиПродажи_Объединение ➢ ЗакупкиПродажи_Соединение ➢ ВедомостьПоТоварам Регистры сведений: ➢ ЗакупочныеЦены

Подсистема «Продажи»

Свойство	Значение
Имя	Продажи
Синоним	Продажи

Подсистема «Продажи»

Свойство	Значение
Входящие в подсистему объекты	Справочники: ➢ Контрагенты ➢ Договоры ➢ Номенклатура Документы: ➢ Расходная Отчеты: ➢ ПрайсЛист ➢ ВедомостьПоТоварам ➢ ОстаткиТоваровСКД ➢ СкладскиеОстатки ➢ ОстаткиПартийТоваров ➢ ПродажиТакКакНадо Регистры накопления: ➢ ОстаткиТоваров ➢ ОстаткиПоСкладам ➢ Продажи

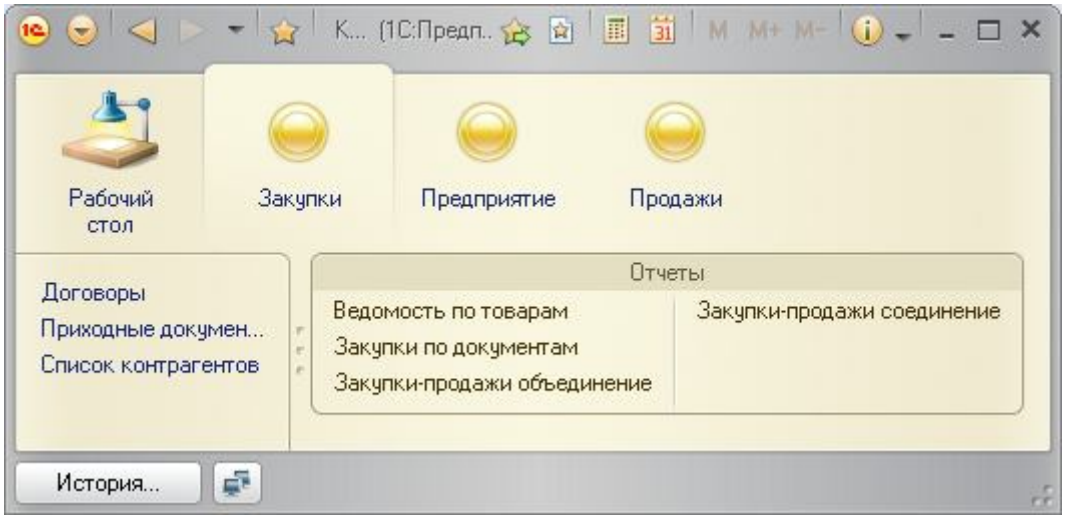
Подсистема «Предприятие»

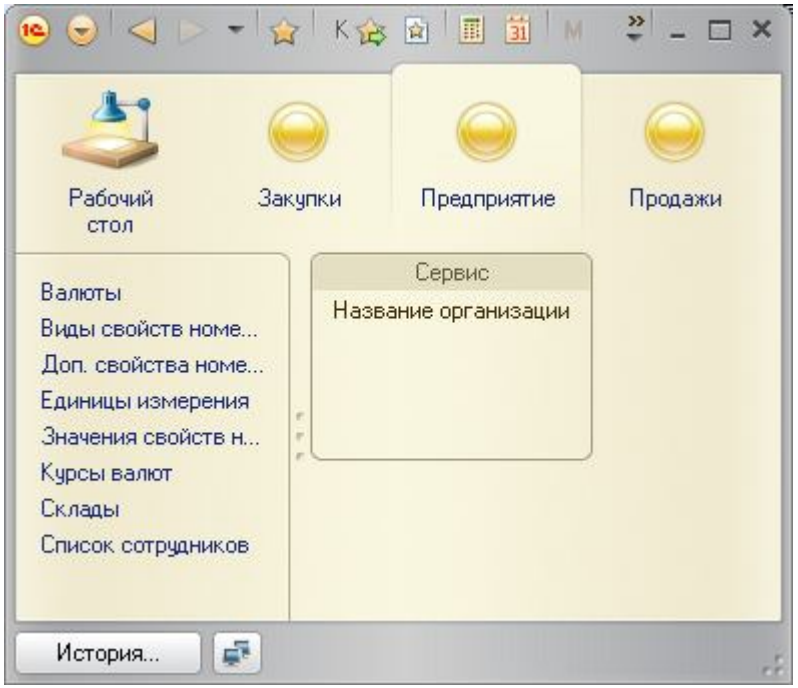
Свойство	Значение
Имя	Предприятие
Синоним	Предприятие

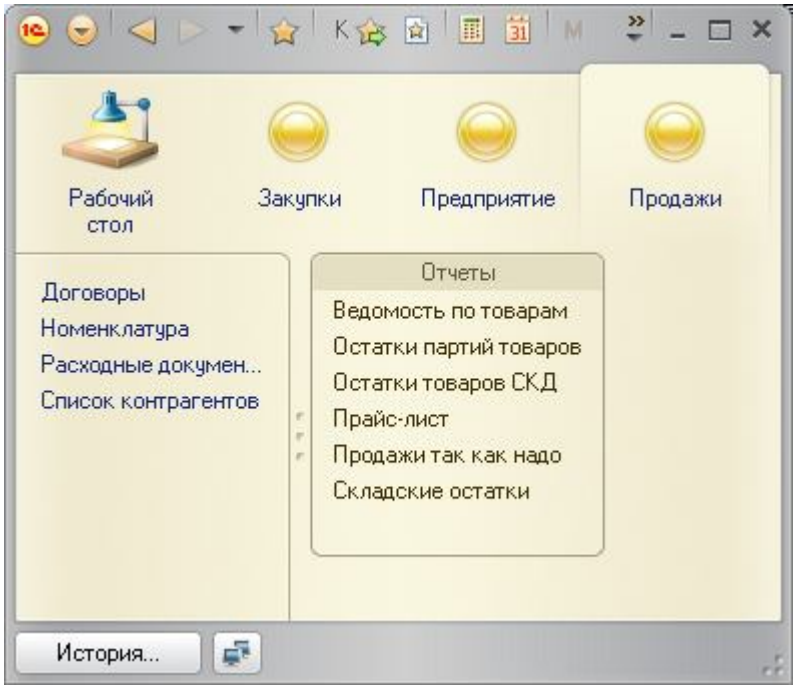
Подсистема «Предприятие»

Свойство	Значение
Входящие в подсистему объекты	Константы: ➢ НазваниеОрганизации Справочники: ➢ Сотрудники ➢ ЕдиницыИзмерения ➢ Валюты ➢ ДопСвойстваНоменклатуры ➢ Склады Планы видов характеристик: ➢ ВидыСвойствНоменклатуры Регистры сведений: ➢ КурсыВалют ➢ ЗначенияСвойствНоменклатуры

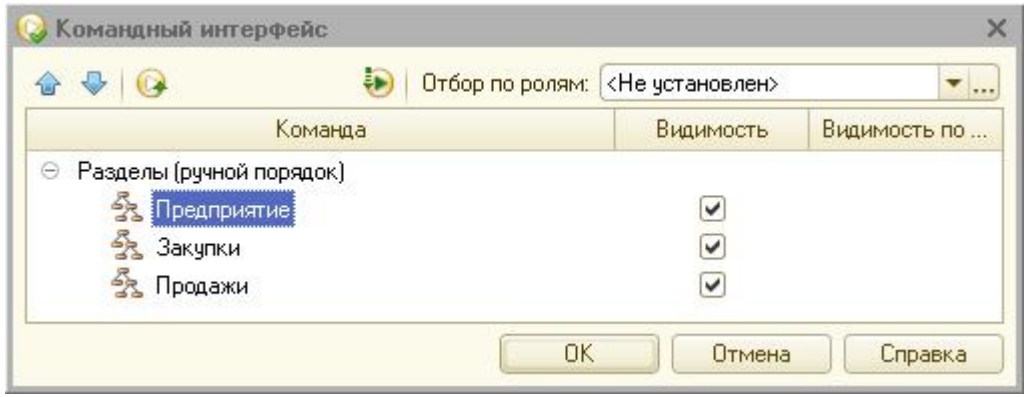
В режиме исполнения:




Рабочий стол
Закупки
Предприятие
Продажи
Валюты
Виды свойств номе...
Доп. свойства номе...
Единицы измерения
Значения свойств н...
Курсы валют
Склады
Список сотрудников
Сервис
Название организации
История...

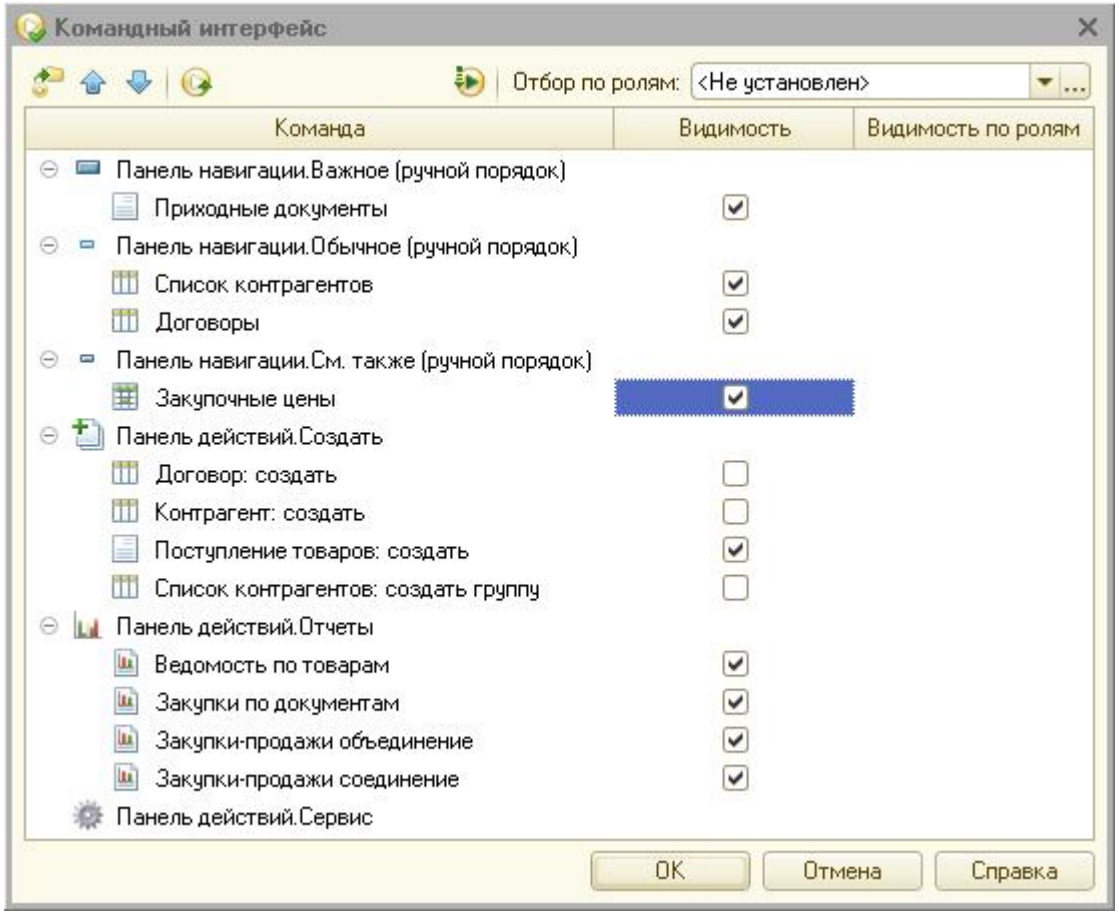

Рабочий стол
Закупки
Предприятие
Продажи
Договоры
Номенклатура
Расходные докумен...
Список контрагентов
Отчеты
Ведомость по товарам
Остатки партий товаров
Остатки товаров СКД
Прайс-лист
Продажи так как надо
Складские остатки
История...

«Конфигурация – ПКМ - Открыть командный интерфейс конфигурации» - можно определить очередность расположения подсистем.

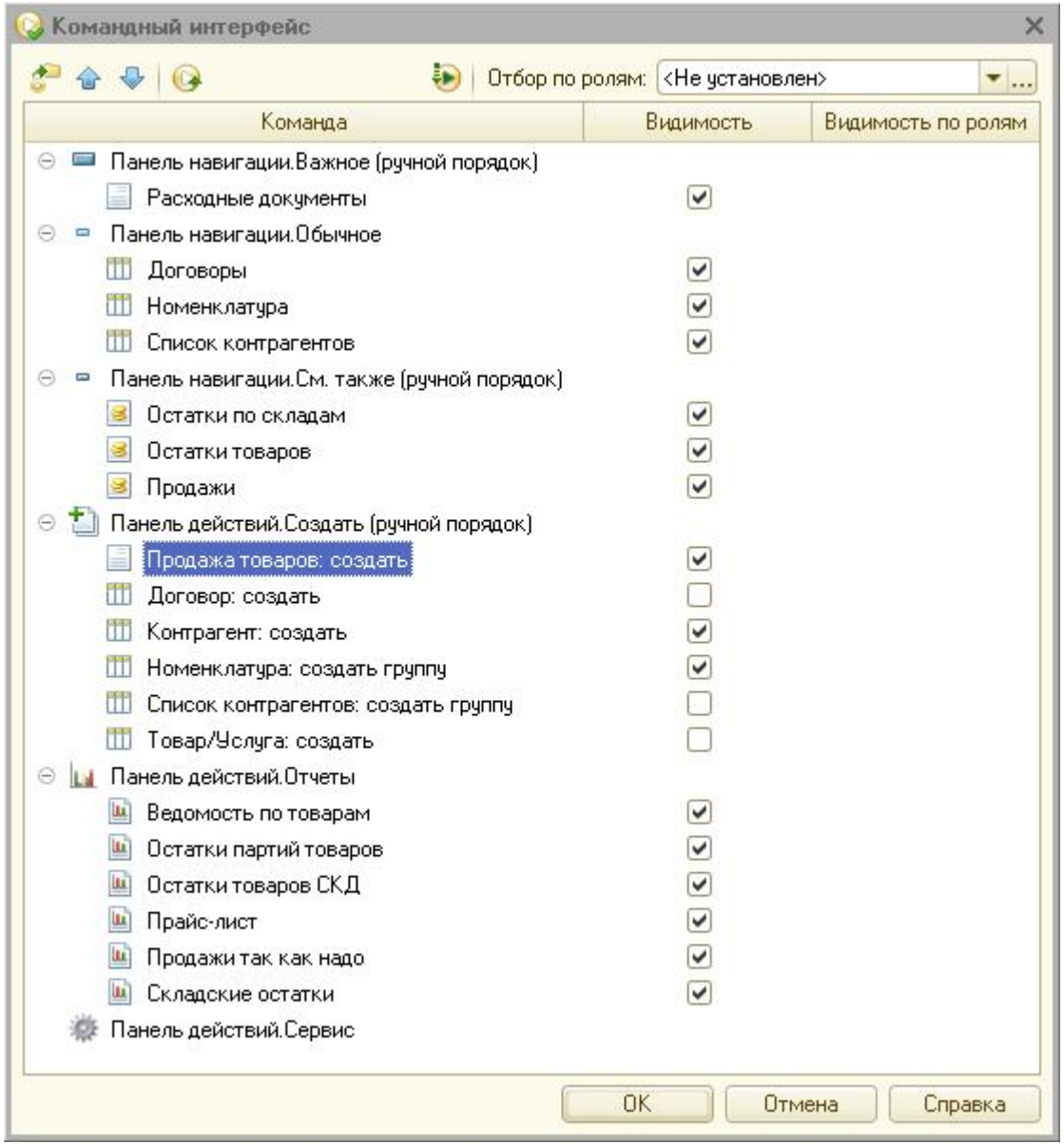


Определить интерфейс можно также для каждой подсистемы – «ПКМ по подсистеме – Открыть командный интерфейс».

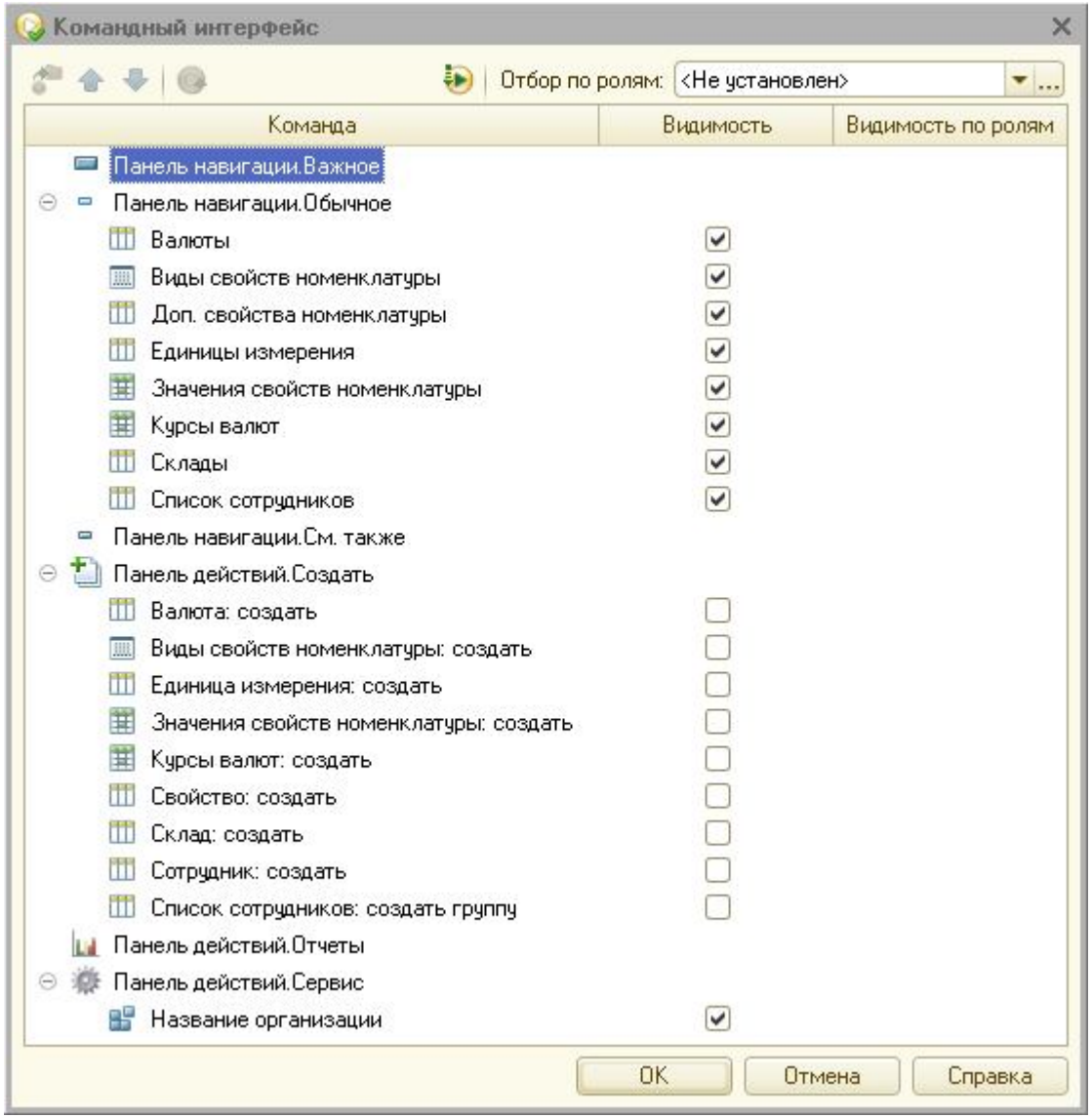
Настроим командный интерфейс для подсистемы «Закупки»:



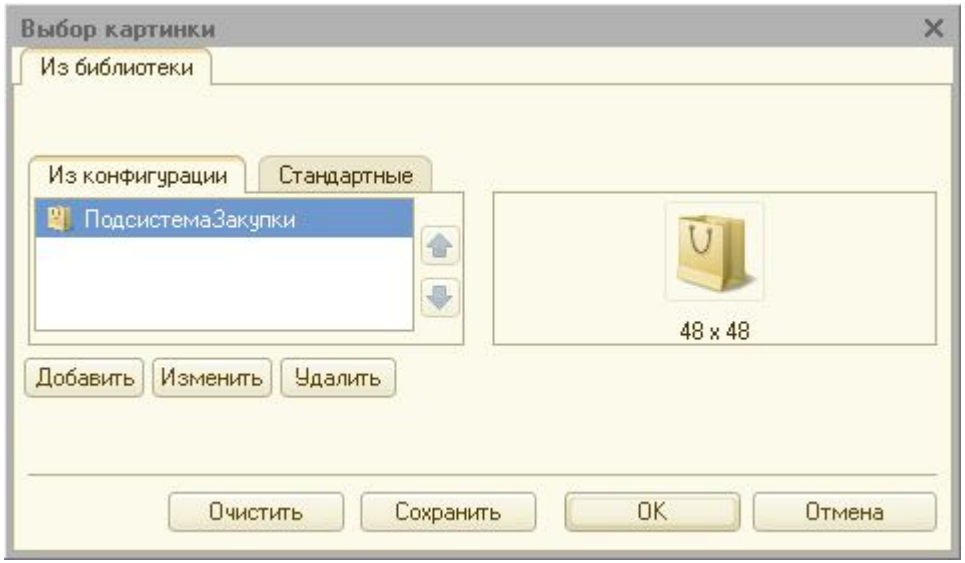
Настроим командный интерфейс для подсистемы «Продажи»:

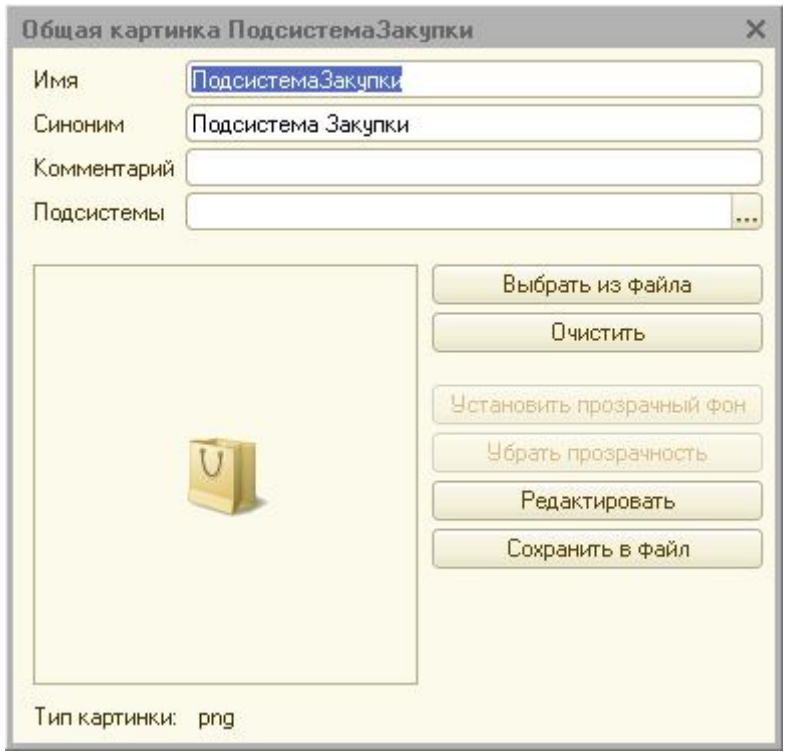


Командный интерфейс подсистемы «Предприятие» не меняем:



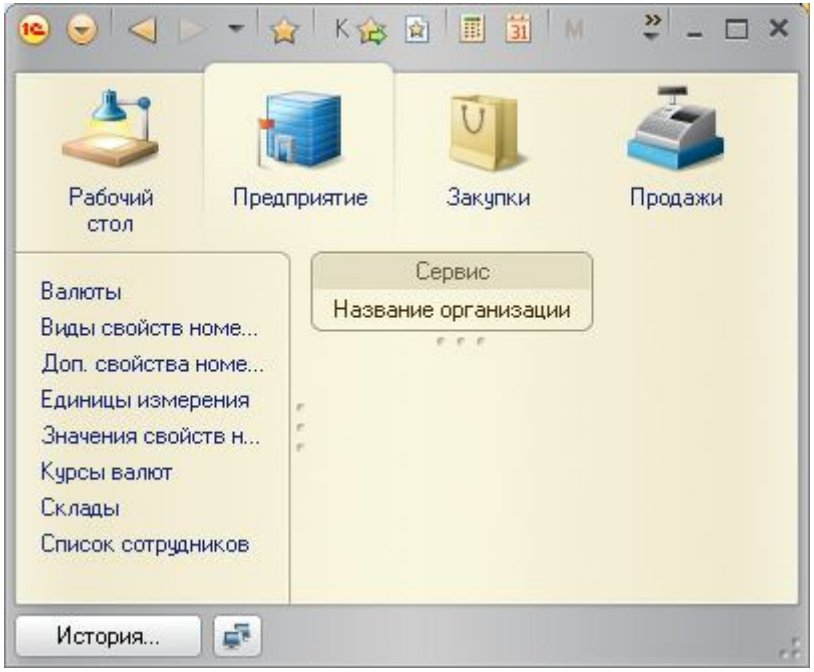
Для Подсистемы можно выбрать картинку (из файла, Конфигурации или стандартную) - «закладка Основные – свойство Картинка».

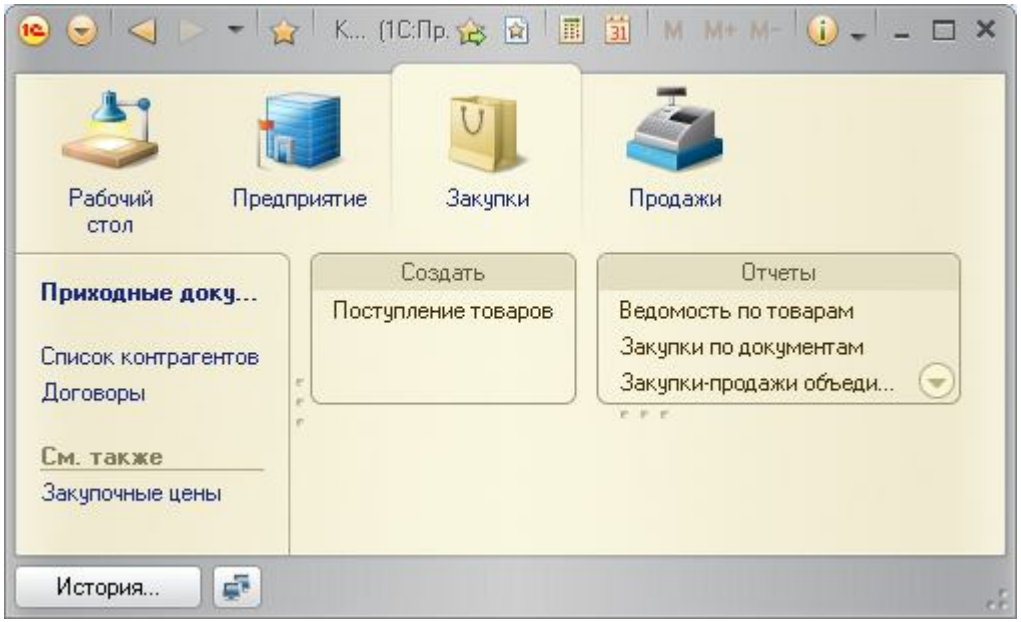


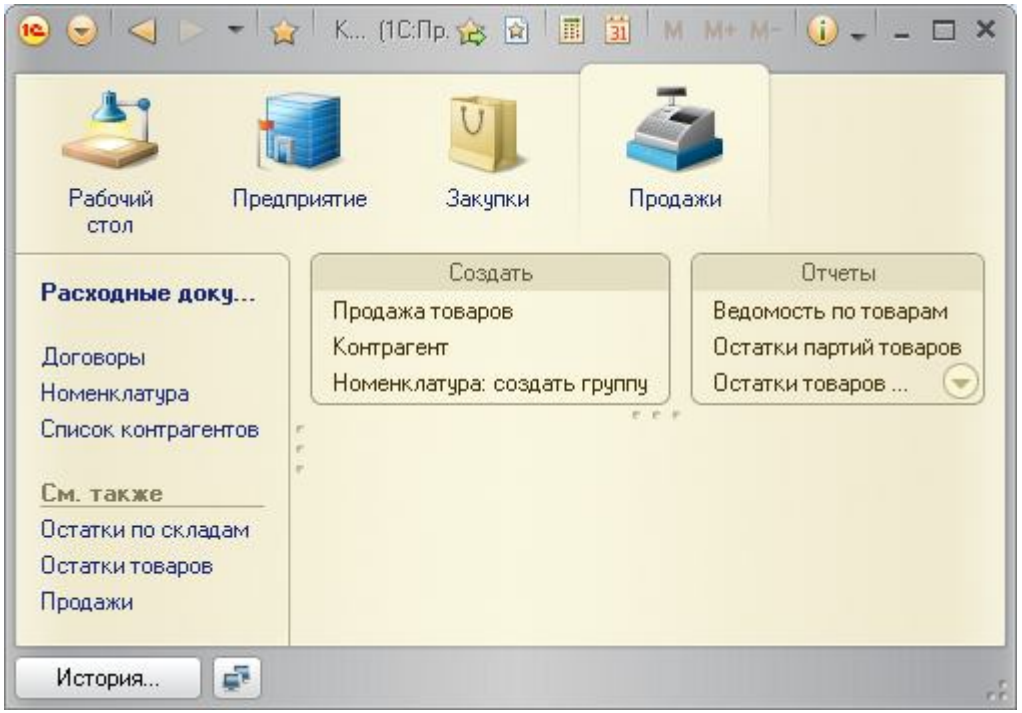


Аналогично добавим картинки в подсистемы «Продажи» и «Предприятие».

В режиме исполнения:







При этом все подписи можно настроить в свойствах нужного объекта.

У Подсистемы можно создать Подчиненную подсистему – будет отображаться также, как панель «См. также».

Подсистема «Рабочий стол» - создается автоматически. Для отображения того, что будет видеть пользователь, входя в программу.

Два способа создания «Рабочего стола»:

- Предопределенный.
- Создать собственную обработку, форму которой поместить на Рабочий стол – для наиболее гибких настроек.

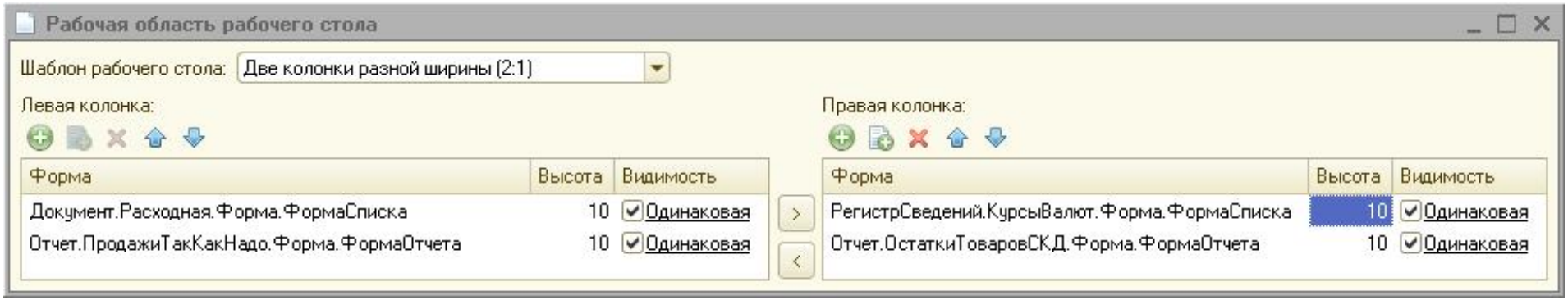
«Конфигурация – ПКМ – Открыть рабочую область рабочего стола»

Создадим форму списка документа «Расходная»:
Реквизиты:

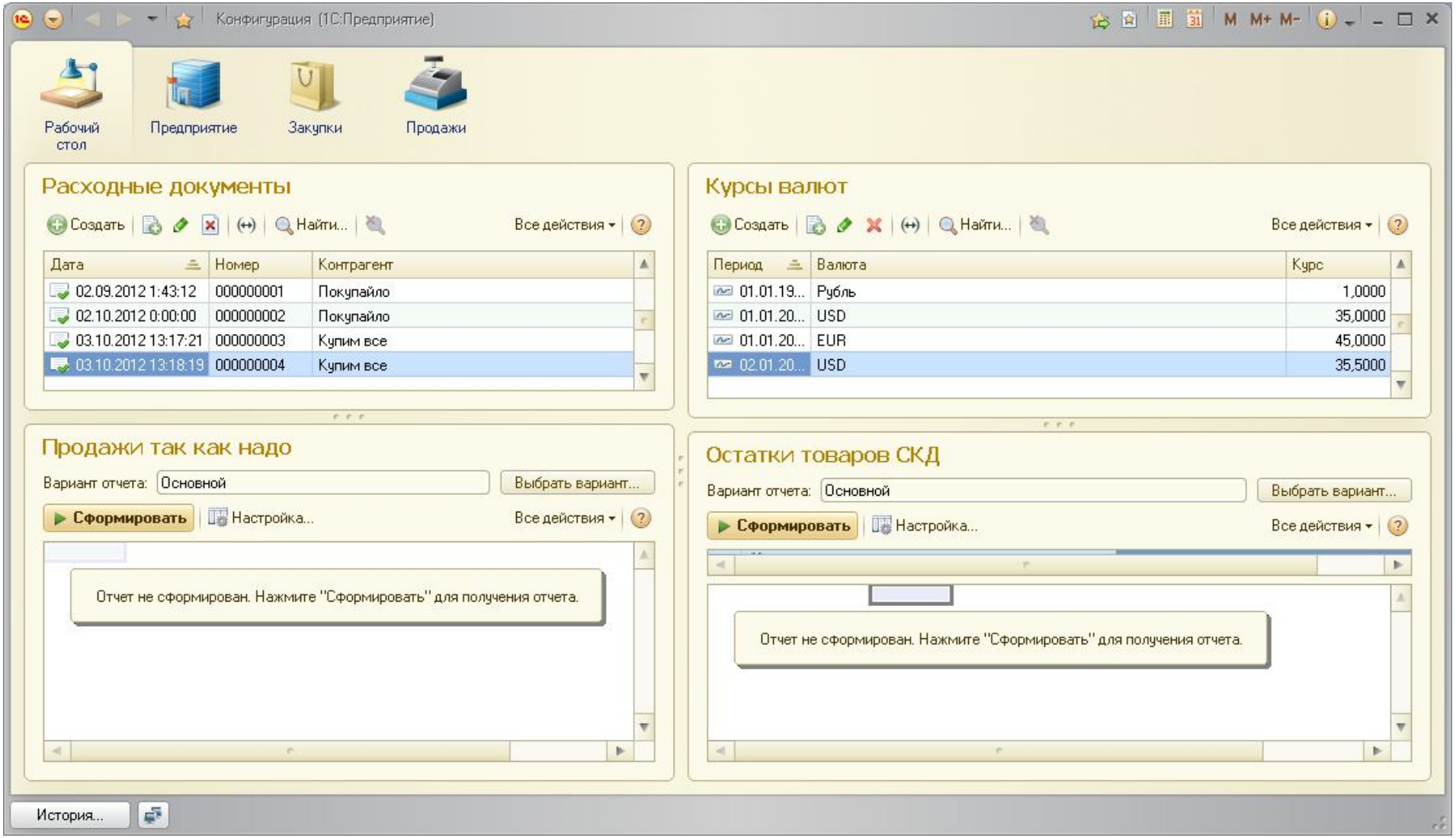
- Дата.
- Номер.
- Контрагент

Добавим форму отчета «ПродажиТакКакНадо».
Добавим форму списка регистра сведений «КурсыВалют».
Добавим форму отчета «ОстаткиТоваровСКД».

Зададим рабочую область Рабочего стола добавив элементы:



В режиме исполнения:



«Конфигурация – Общие – Функциональные опции» - возможность включить/отключить определенные механизмы системы.

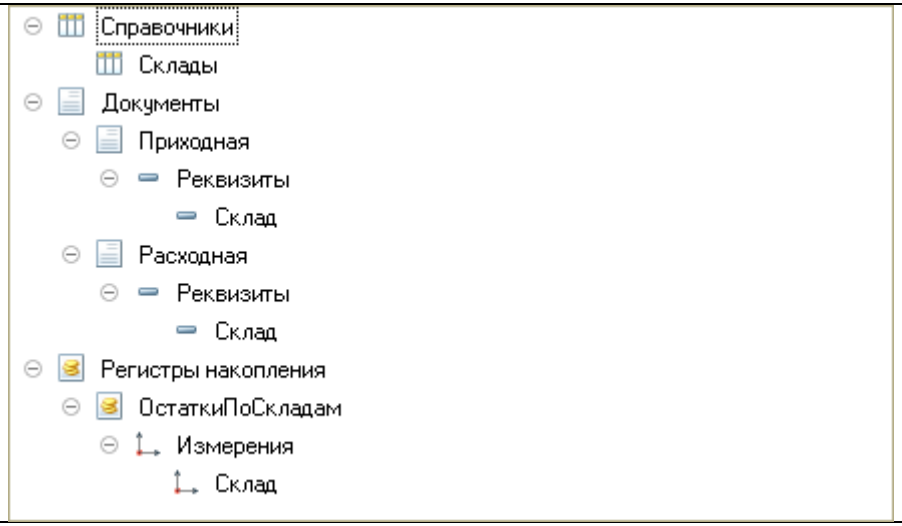
Например, есть на предприятии Складской учет. Для определенных целей складской учет может не вестись. Создается настройка, которая включает/отключает ведение складского учета.

Надо определить, где будет храниться значение опции. В примере складской учет будет включаться у всей конфигурации, поэтому будем сохранять значение настройки в Константе.

Добавим константу «Складской учет»:

Свойство	Значение
Имя	СкладскойУчет
Синоним	Включить учет остатков на складах
Тип	Булево

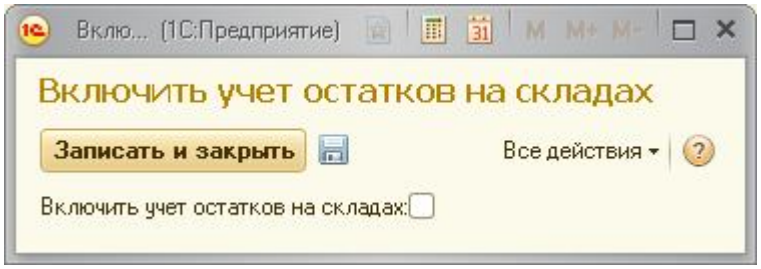
Добавим Функциональную опцию «Складской учет»:

Свойство	Значение
Имя	СкладскойУчет
Хранение	Константа.СкладскойУчет
Закладка «Состав»	

Включим Константу «СкладскойУчет» в подсистему «Предприятие».

В режиме исполнения:

Значение Константы по умолчанию «Ложь», т.е. учет по складам выключен, и склады нигде не указываются:



Если включить, то склады появятся.

«Функциональные опции» влияют только на интерфейс. Т.е. если реквизит «Склад» обязателен для заполнения, то логику зависимости от функциональной опции надо будет прописывать через код.

Более сложный пример работы с Функциональными опциями:

Создадим настройку, которая будет включать учет валюты в документе (Приходная, Расходная) в зависимости от типа Договора:

Настройку будем хранить в справочнике «Договоры» - добавим реквизит «Валютный учет»:

Свойство	Значение
Имя	ВалютныйУчет
Синоним	Валютный учет
Тип	Булево

Добавим реквизит «Валюта» в документы Приходная и Расходная:

Свойство	Значение
Имя	Валюта
Синоним	Валюта
Тип	СправочникСсылка.Валюты

И отобразим его на форме.

Добавим функциональную опцию «Валютный учет»:

Свойство	Значение
Имя	ВалютныйУчет
Синоним	Валютный учет
Хранение	Справочник.Договоры.Реквизит.ВалютныйУчет
Закладка «Состав»	

Надо узнать, по какому договору опция включена/выключена. Для этого добавим Параметр функциональной опции «Договор». В зависимости от него опция может быть включена/выключена:

Свойство	Значение
Имя	Договор
Синоним	Договор
Использование	Справочник.Договоры

Теперь, при конструировании управляемой формы, надо определить, отображать валюту или нет. Для этого при создании формы документа на Сервере, сделаем проверку опции:

```
&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
    ПараметрыФО = Новый Структура("Договор", Объект.Договор);
    УстановитьПараметрыФункциональныхОпцийФормы(ПараметрыФО);
КонецПроцедуры
```

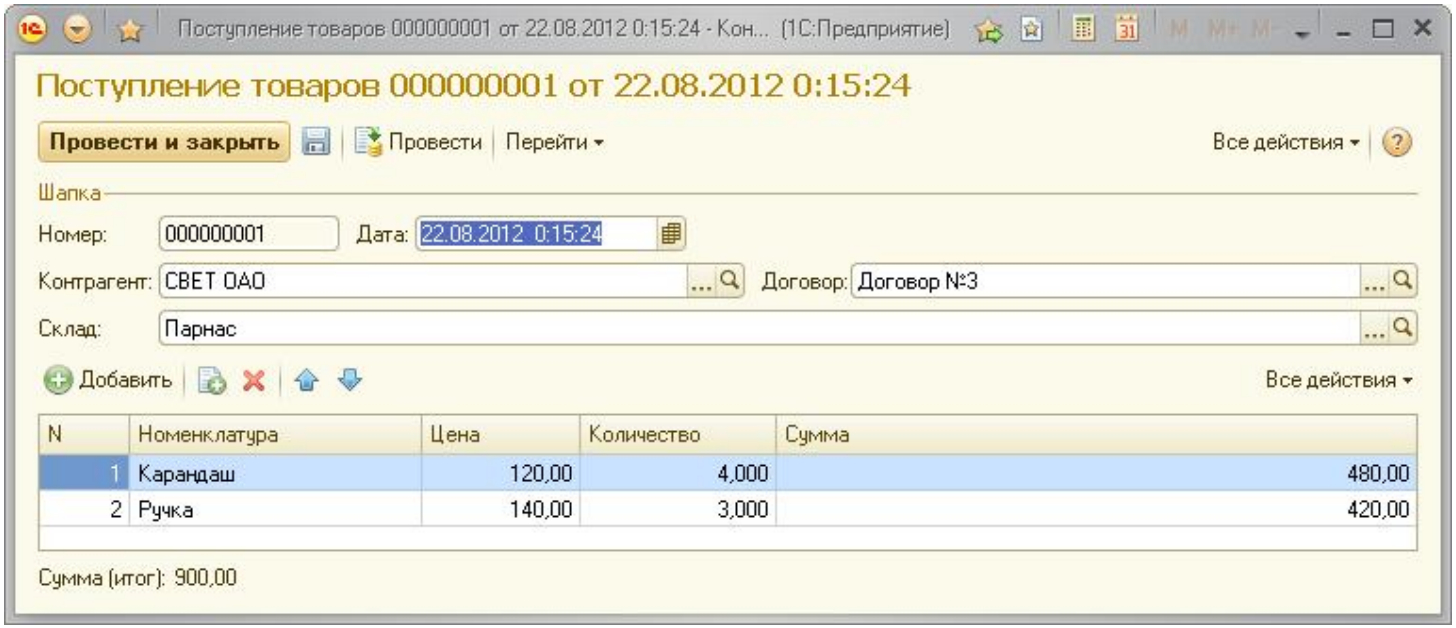
Аналогичные действия следует выполнять и при изменении договора в форме:

```
&НаКлиенте
Процедура ДоговорПриИзменении(Элемент)
    ПараметрыФО = Новый Структура("Договор", Объект.Договор);
    УстановитьПараметрыФункциональныхОпцийФормы(ПараметрыФО);
КонецПроцедуры
```

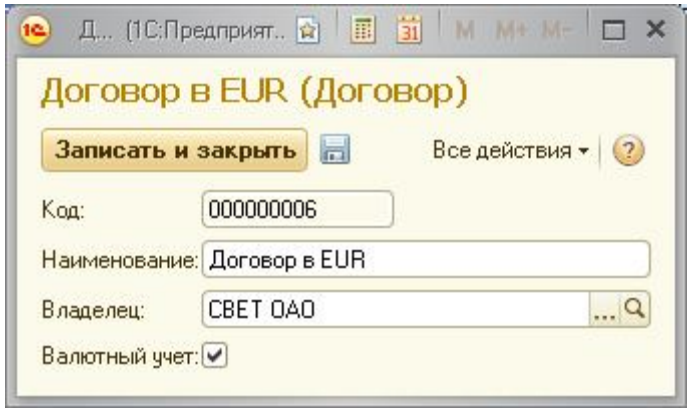
Аналогичные процедуры создадим и в документе Расходная.

В режиме исполнения:

Откроем Приходную №1:



Добавим новый договор (валютный):



В форме документа появилось поле «Валюта»:

Если в форме списка документа Приходная будет колонка «Валюта», то она будет отображаться, если хотя бы у одного документа включена функциональная опция «Валютный учет».

Также Функциональные опции могут размещаться в Регистрах сведений. Например, если надо хранить сложное сочетание настроек.
Пример:
Валютный учет может быть включен в зависимости от вида договора, с какой организацией договор заключен, каким менеджером оформлен договор. Условия будут «Параметрами функциональных опций» и «Измерениями» в регистре сведений, а «Ресурс» регистра сведений будет хранить значение опции (включена/выключена).

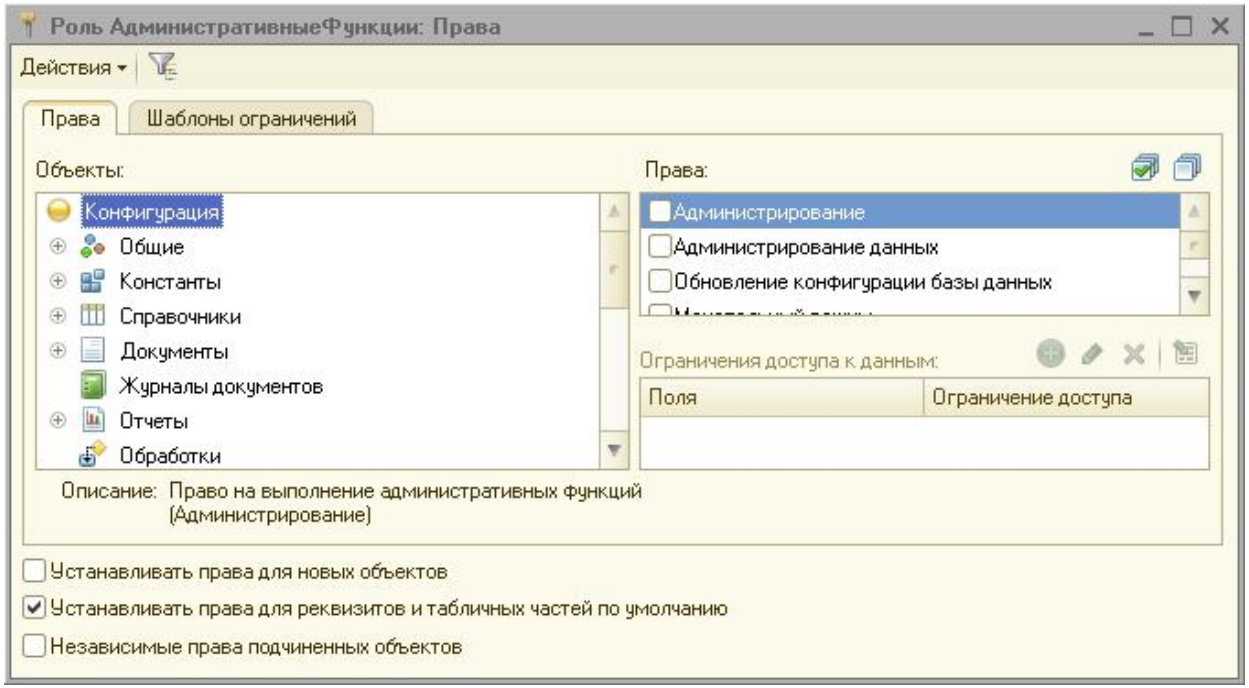
Роли

Это настройки прав пользователей.

Создадим роль «Административные функции»:

Свойство	Значение
Имя	АдминистративныеФункции
Синоним	Административные функции

Откроется «Конструктор ролей»:



«Действия – Установить все права» - установим все права.

Свойство	Значение
Устанавливать права для новых объектов	Истина

Права:
Просмотр – это интерактивное.
Чтение – это программное.

Настройки прав на Конфигурацию (в корне «дерева»):

- Администрирование – доступ к конфигуратору и к работе с метаданными.
- Обновление конфигурации базы данных – для возможности обновления БД пользователю, не обладающему правами Администратора.
- Монопольный режим – монопольный режим явно не задается, но он нужен для выполнения служебных действий (например, удаление помеченных объектов).

- Активные пользователи – возможность просмотра тех, кто работает в БД.
- Журнал регистрации – просмотр истории изменений в БД.

«Администрирование – Настройка журнала регистрации» - настройка отображаемой в журнале информации.

Журнал регистрации хранится:

- ✓ На локальной машине – если файловый вариант.
- ✓ На сервере БД – если клиент-серверный вариант.
- Вывод – если «Ложь», то пользователь не сможет вывести информацию из БД (только если screenshot)

Создадим роль «Менеджер»:

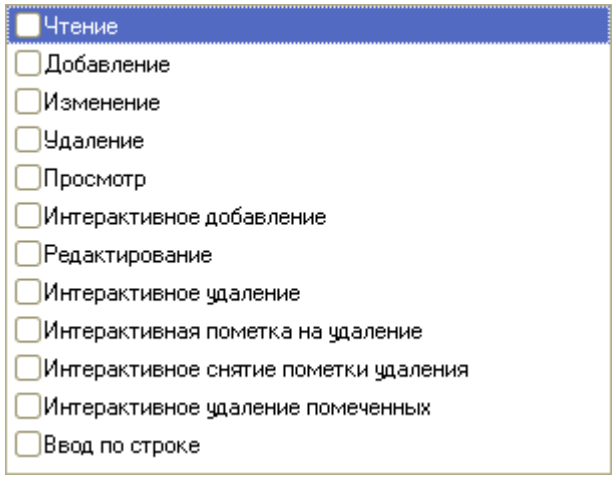
Свойство	Значение
Имя	Менеджер
Синоним	Менеджер

Сначала включим все права и установим свойство:

Свойство	Значение
Устанавливать права для новых объектов	Истина

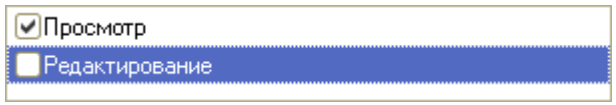
Константы:

Справочник «Сотрудники»:



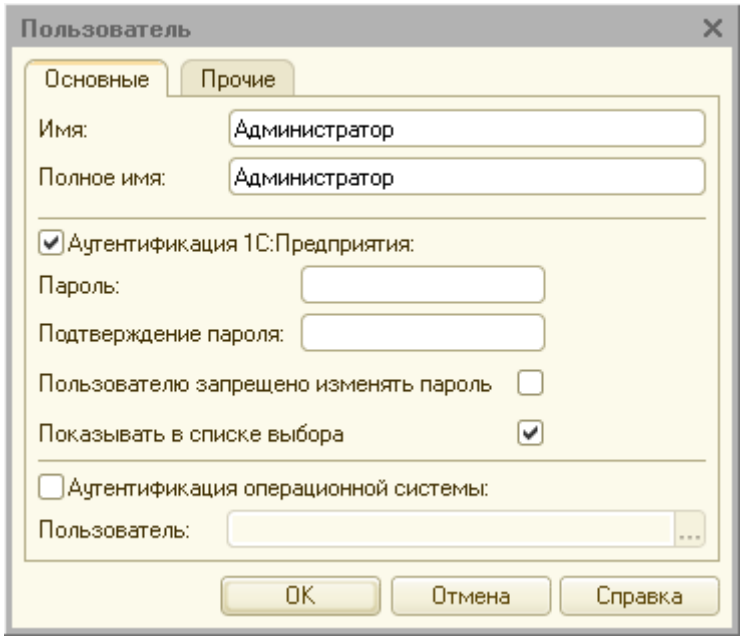
Справочник «Контрагенты» - реквизит «ОсновнойМенеджер»:

Справочник «Контрагенты» - реквизит «ВалютаВзаиморасчетов». Смотреть может, менять не может:



Документ «Расходная» - табличная часть «Товары» - реквизит «Цена». Цена продажи берется автоматом из справочника «Номенклатура»:

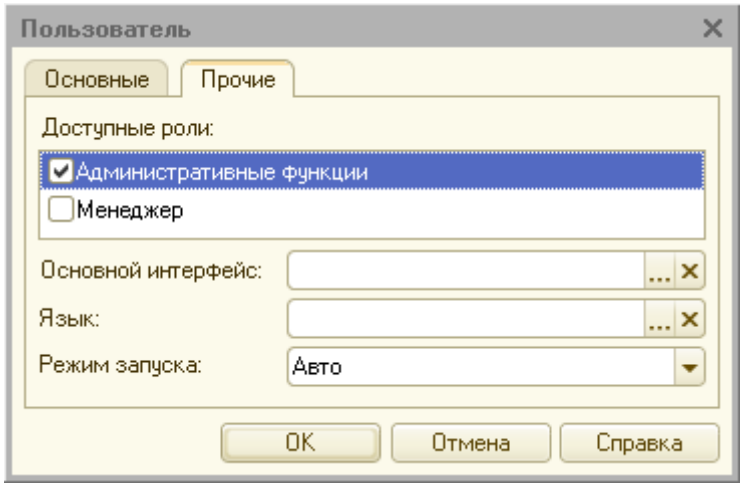
Создадим пользователей – реализуем авторизацию пользователей. «Администрирование - Пользователи»



Пароль пользователя сохраняется в таблицах БД в зашифрованном виде.

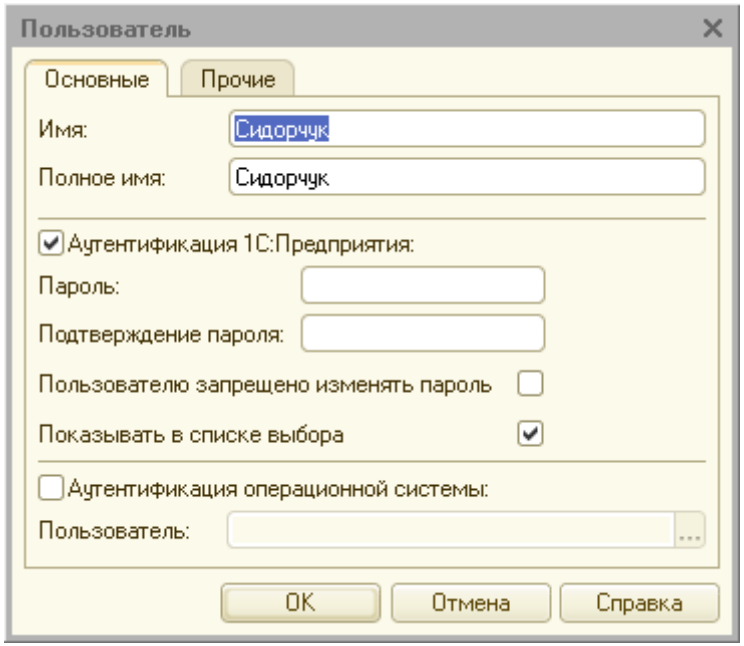
Аутентификация операционной системы – если пользователь авторизован в ОС, то он автоматически авторизуется в системе 1С, т.е. даже окно выбора появляться не будет.

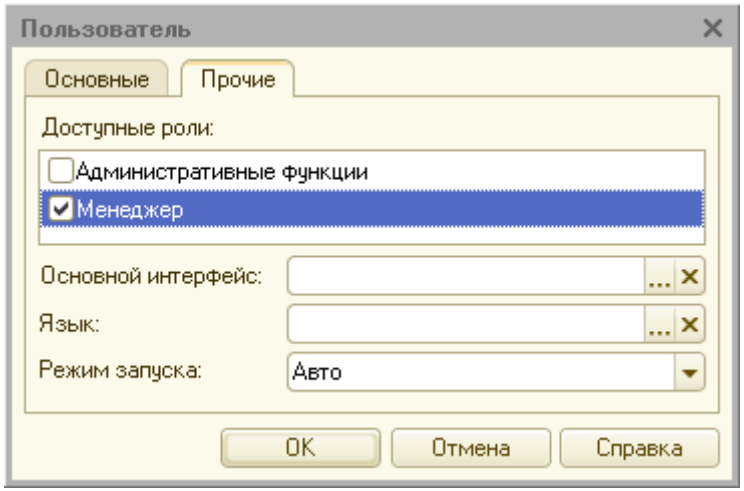
Закладка «Прочее»:



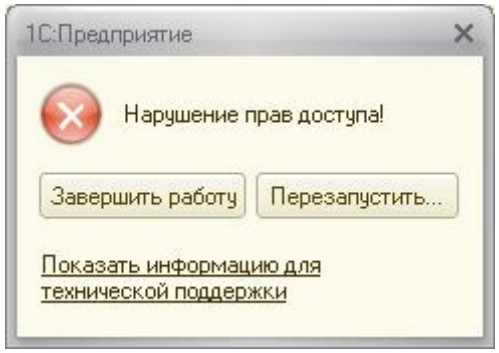
Если пользователю доступны несколько ролей, то права складываются, т.е. если в одной роли есть разрешение, а в другой нет, то доступ будет разрешен.

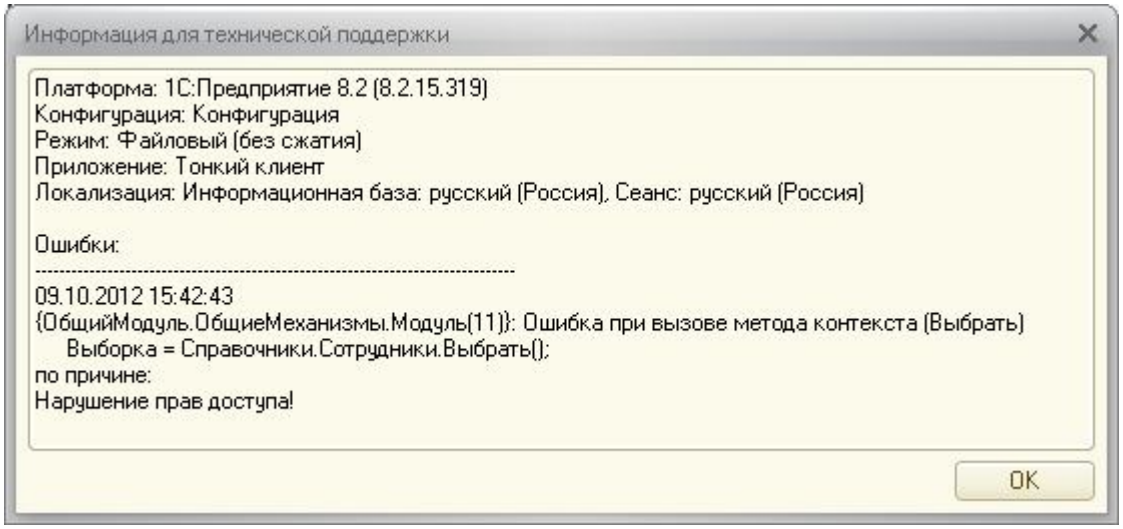
Создадим пользователя «Сидорчук»:





Запустим программу в режиме исполнения от имени Менеджера:





Справочник «Сотрудники» недоступен Менеджеру (даже программно), поэтому система выдает СОО. А обращение к справочнику идет из процедуры, которая при запуске показывает именинников. Процедура находится в общем модуле «ОбщиеМеханизмы».

Но если все же надо предоставить пользователю выполнение этой процедуры, то ее можно вынести в отдельный Общий модуль. Добавим Общий модуль «Привилегированный»:

Свойство	Значение
Имя	Привилегированный
Синоним	Привилегированный
Сервер	Истина
Вызов сервера	Истина

Свойство	Значение
Привилегированный	Истина

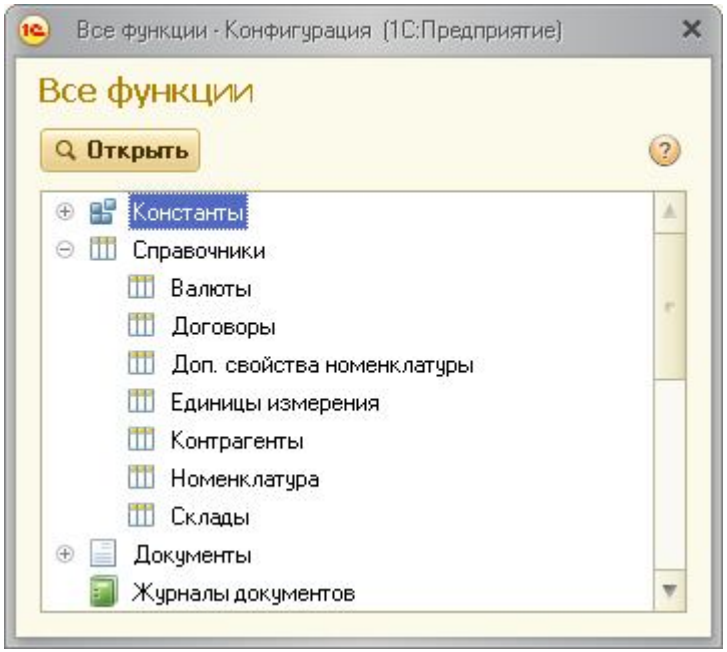
Свойство «Привилегированный» - при вызове процедур и функций из этого модуля система не будет учитывать настройки прав пользователя (запреты). Перенесем функцию «ПолучитьСписокИменинников» в этот модуль:

Вызывали эту функцию из «Модуля управляемого приложения» - «Конфигурация – Открыть модуль управляемого приложения»:

```
Процедура ПриНачалеРаботыСистемы()
     //Предупреждение("Добро пожаловать!", 5);
     //Сообщить("Сегодня " + ТекущаяДата());
     СписокИменинников = Привилегированный.ПолучитьСписокИменинников();

     Для Каждого ЭлементМассива Из СписокИменинников Цикл
          Сообщить("Сегодня ДР у " + ЭлементМассива);
     КонецЦикла;

КонецПроцедуры
```

В режиме исполнения:

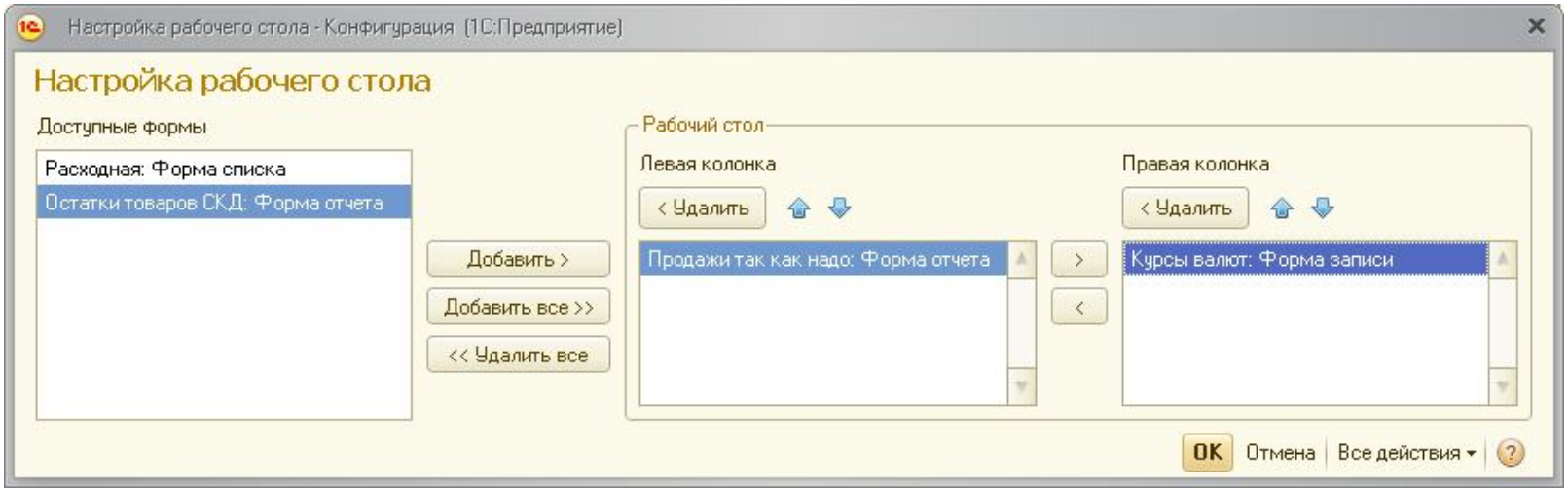
Теперь Менеджер входит в систему, обработка выполняется. Но если зайти «Все функции - Справочники»



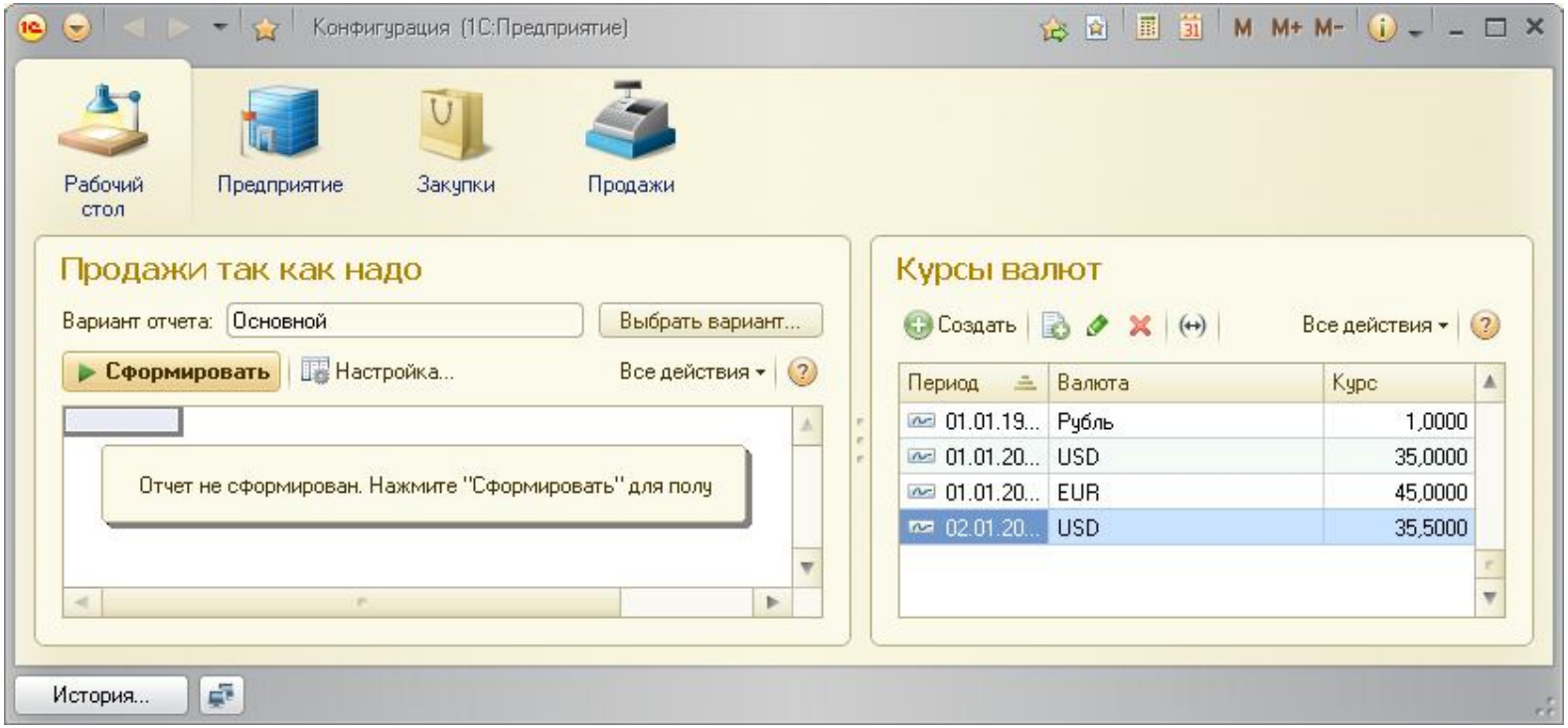
То справочник «Сотрудники» по-прежнему недоступен.

Пользователь может поменять настройки и в режиме исполнения. Допустим пользователю на Рабочем столе не нужна форма списка документа «Расходная» и отчет «Остатки товаров»:

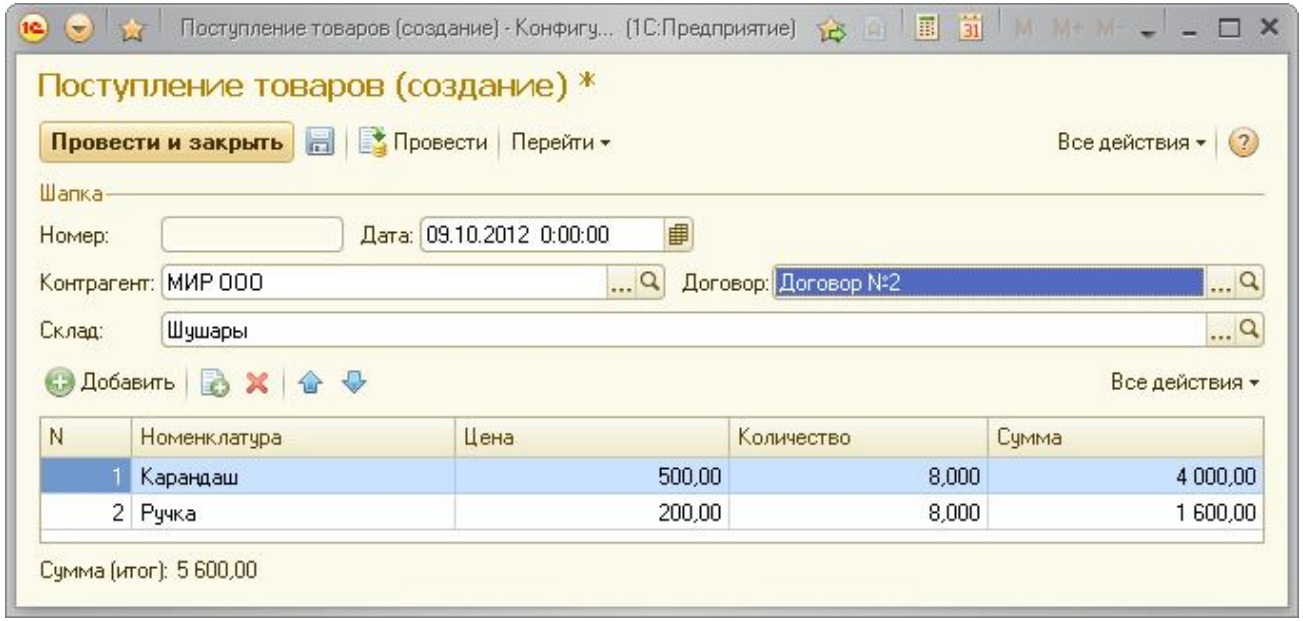
«вкладка Рабочий стол – ПКМ – Настройка рабочего стола»


Настройка рабочего стола - Конфигурация (1С:Предприятие)
Настройка рабочего стола
Доступные формы
Расходная: Форма списка
Остатки товаров СКД: Форма отчета
Добавить >
Добавить все >>
<< Удалить все
Рабочий стол
Левая колонка
< Удалить
Продажи так как надо: Форма отчета
Правая колонка
< Удалить
Курсы валют: Форма записи
OK
Отмена
Все действия

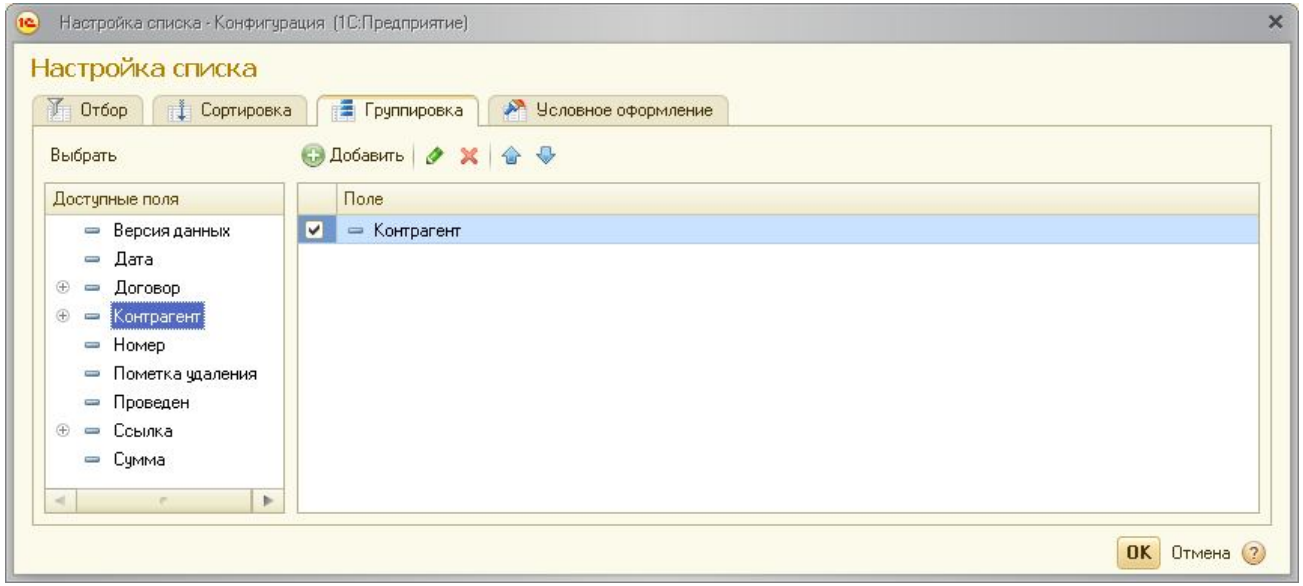
Результат:


Конфигурация (1С:Предприятие)
Рабочий стол
Предприятие
Закупки
Продажи
Продажи так как надо
Вариант отчета: Основной
Выбрать вариант...
Сформировать
Настройка...
Все действия
Отчет не сформирован. Нажмите "Сформировать" для полу
Курсы валют
Создать
Все действия
Период
Валюта
Курс
01.01.19... Рубль 1,0000
01.01.20... USD 35,0000
01.01.20... EUR 45,0000
02.01.20... USD 35,5000
История...

Аналогично можно менять и другие элементы. Например, табличная часть документа «Приходная». Сначала создадим еще один документ Приходная:



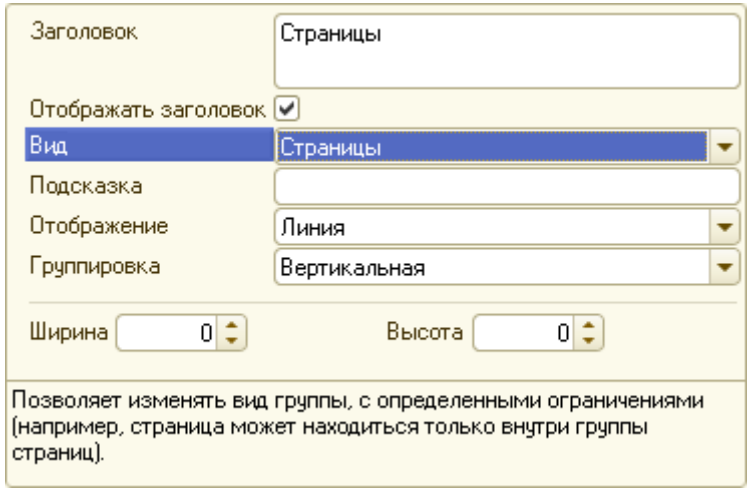
В списке документов «Приходная»: «Все действия – Настроить список» сделаем настройку:



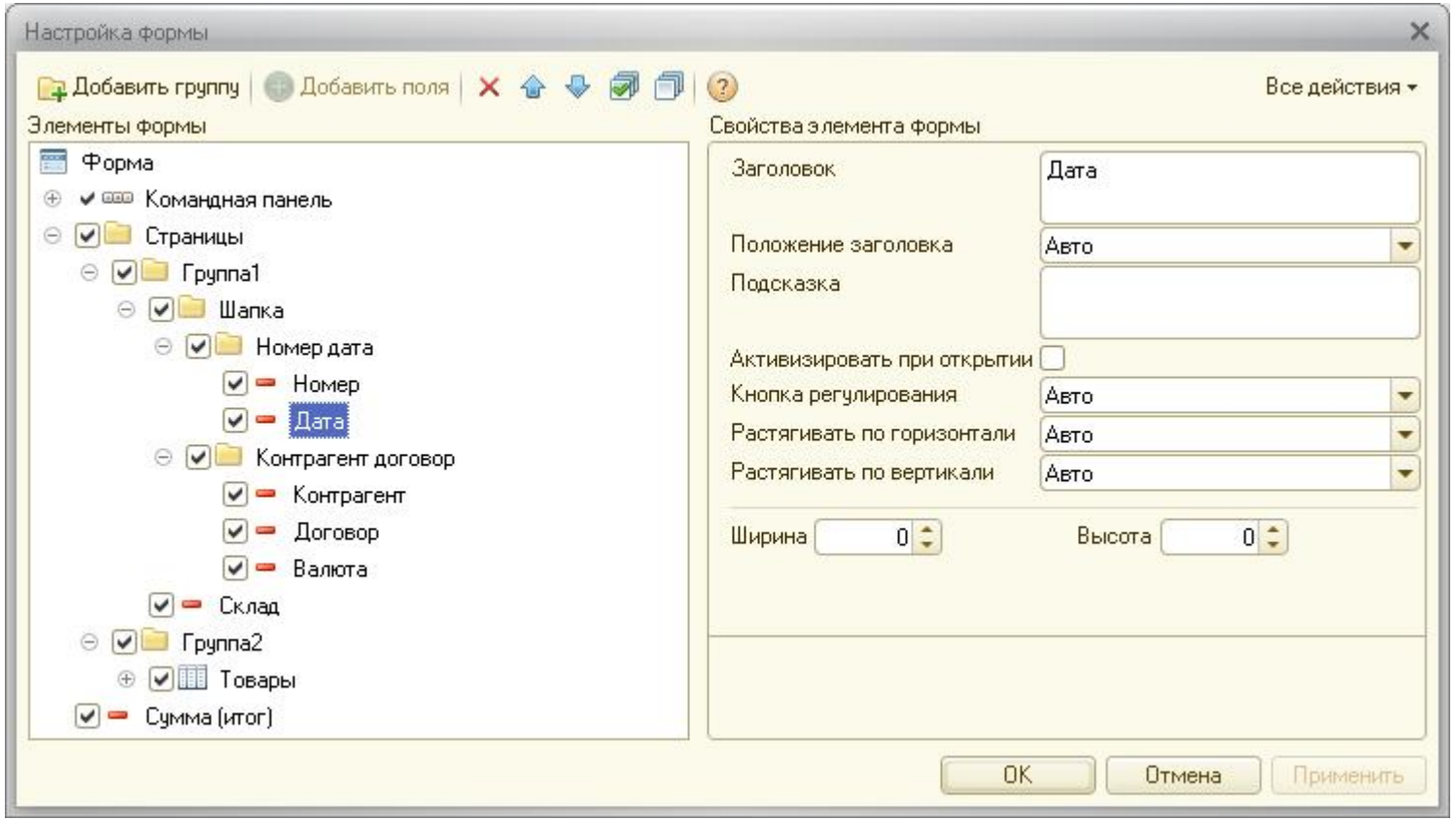
И теперь список Приходных документов выглядит так:

Также пользователь может произвести настройки вида и в самом документе - «Все действия – Изменить форму»:
Например, документ Расходная №2:

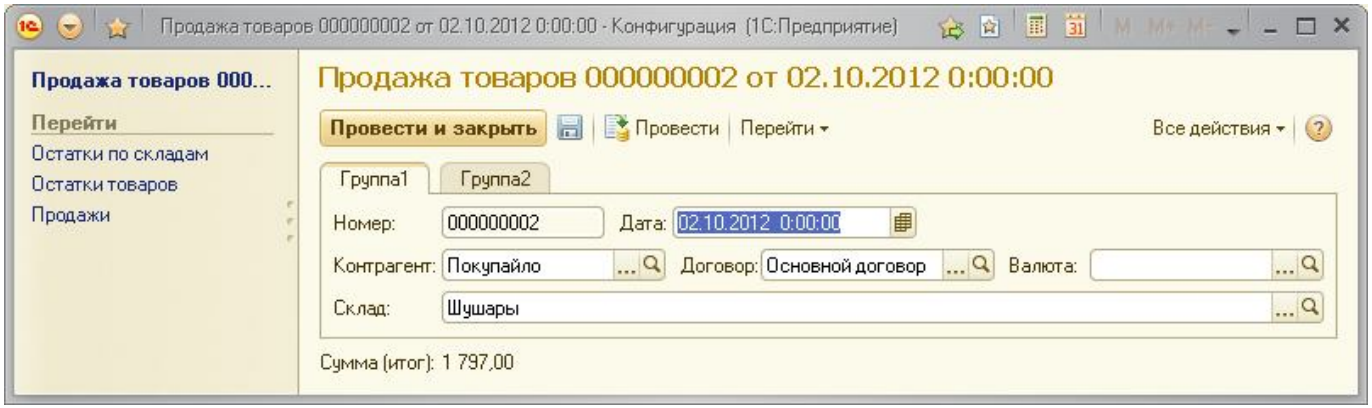
Создадим группу «Страницы»:

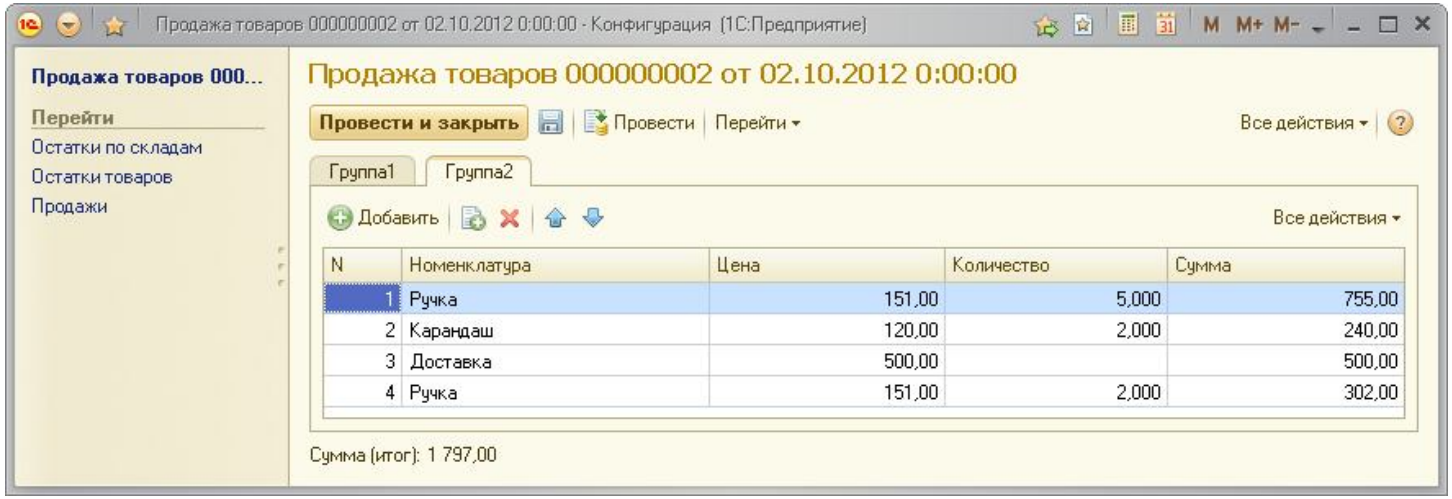


А в ней еще две группы. Разместим по этим группам элементы:



Итого:





Большинство пользовательских настроек сохранится. В примере не сохранится только группировка по Контрагенту в списке Приходных документов.

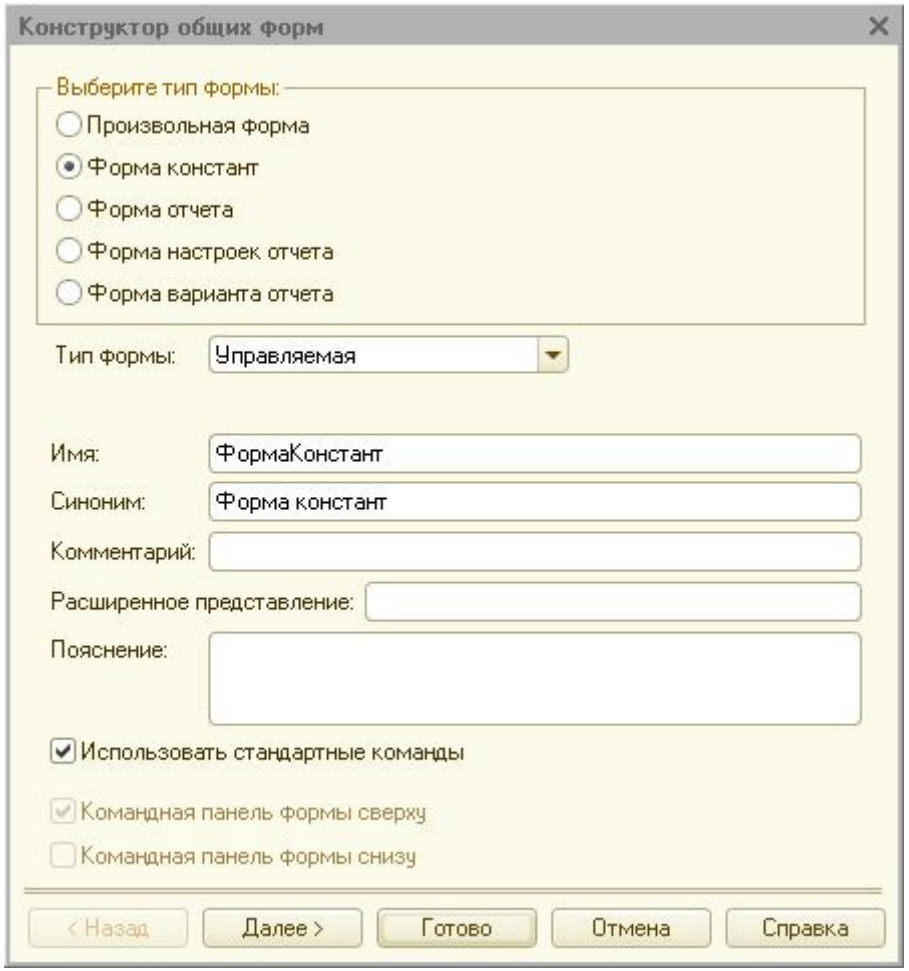
Занятие 1.16

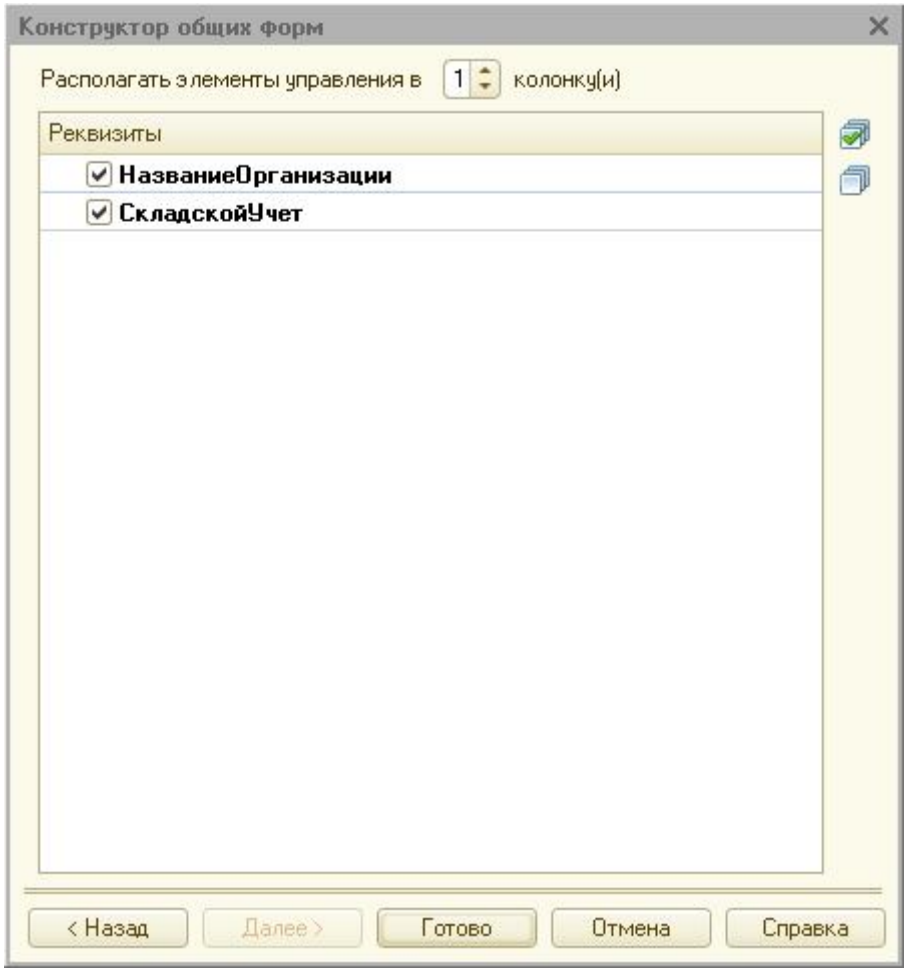
Команды

«Конфигурация – Общие – Общие команды» - команды, создаваемые для всей конфигурации. Например, открыть справку, настройки параметров учета и т.д.

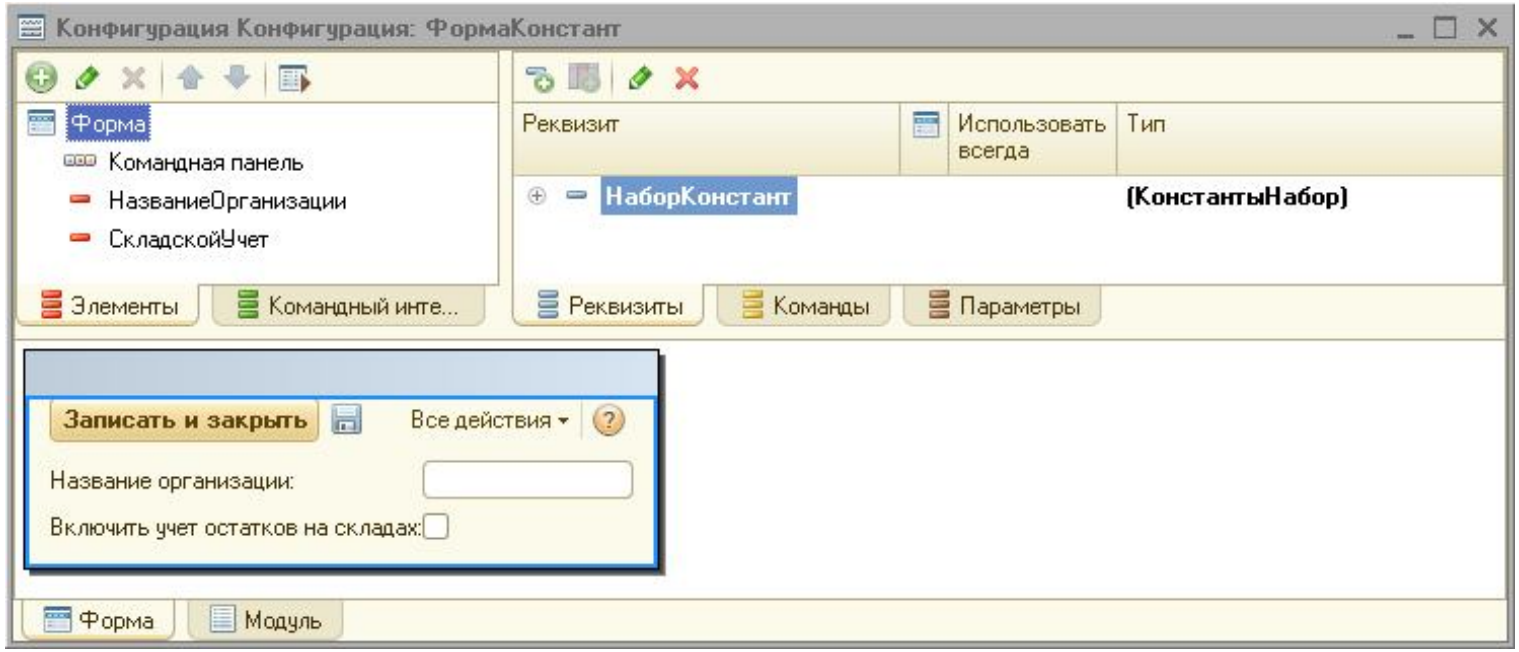
Пример:
Настройку констант «НазваниеОрганизации» и «СкладскойУчет» объединим в «Параметры учета» и создадим команду для их одновременного редактирования.

У Констант нет своих форм, поэтому, при создании формы Констант, она будет находиться в «Конфигурация – Общие – Общие формы».
По умолчанию Общие формы не включаются в командный интерфейс.
Создадим форму констант:


Конструктор общих форм
Выберите тип формы:
Произвольная форма
Форма констант
Форма отчета
Форма настроек отчета
Форма варианта отчета
Тип формы:
Управляемая
Имя:
ФормаКонстант
Синоним:
Форма констант
Комментарий:
Расширенное представление:
Пояснение:
Использовать стандартные команды
Командная панель формы сверху
Командная панель формы снизу
< Назад
Далее >
Готово
Отмена
Справка


Конструктор общих форм
Располагать элементы управления в 1 колонку(и)
Реквизиты
НазваниеОрганизации
СкладскойУчет
< Назад
Далее >
Готово
Отмена
Справка

Результат:



Создадим Общую команду «Открыть параметры учета»:

Свойство	Значение
Имя	ОткрытьПараметрыУчета
Синоним	Параметры учета
Группа	Панель действий.Сервис

Группа – куда будет входить команда.

При этом автоматом создается процедура:

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
    //Вставить содержимое обработчика.
    //ПараметрыФормы = Новый Структура("", );
```

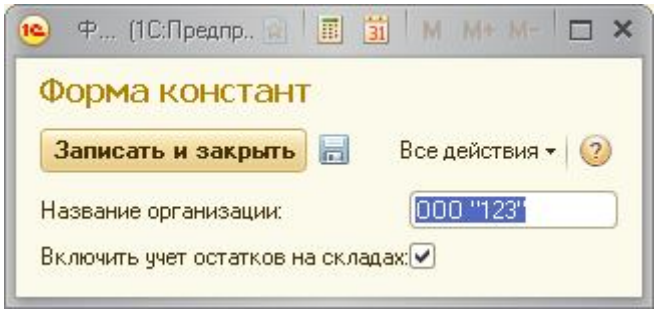
```
    //ОткрытьФорму("ОбщаяФорма.", ПараметрыФормы, ПараметрыВыполненияКоманды.Источник,
ПараметрыВыполненияКоманды.Уникальность, ПараметрыВыполненияКоманды.Окно);
КонецПроцедуры
```

В процедуре закомментирован код для создания форм (заготовка). Отредактируем код:

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
    ОткрытьФорму("ОбщаяФорма.ФормаКонстант");
КонецПроцедуры
```

Команду включим в Подсистему «Предприятие»

В режиме исполнения:

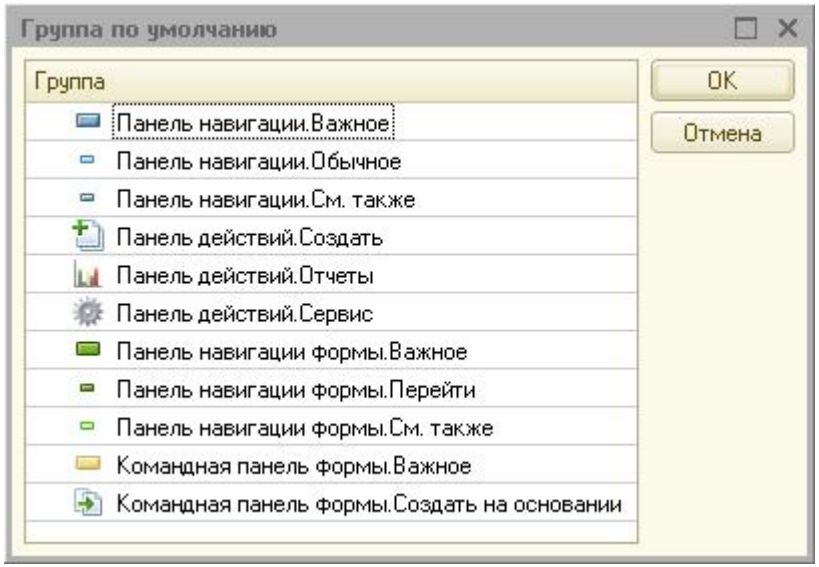


Также есть Команды объектов:

Можно создать команду, которая будет формировать печатную форму документа или нескольких документов, в зависимости от открытой формы (списка или документа). Т.е. команда одна, а параметры выполнения могут быть разные.

Создадим команду для документа «Расходная»:

При создании команды, надо указать, в какой группе она будет находиться:



Создадим отдельную группу команд «Печать» - для всей конфигурации в целом. «Конфигурация – Общие – Группы команд»:

Свойство	Значение
Имя	Печать
Синоним	Печать
Категория	Командная панель формы

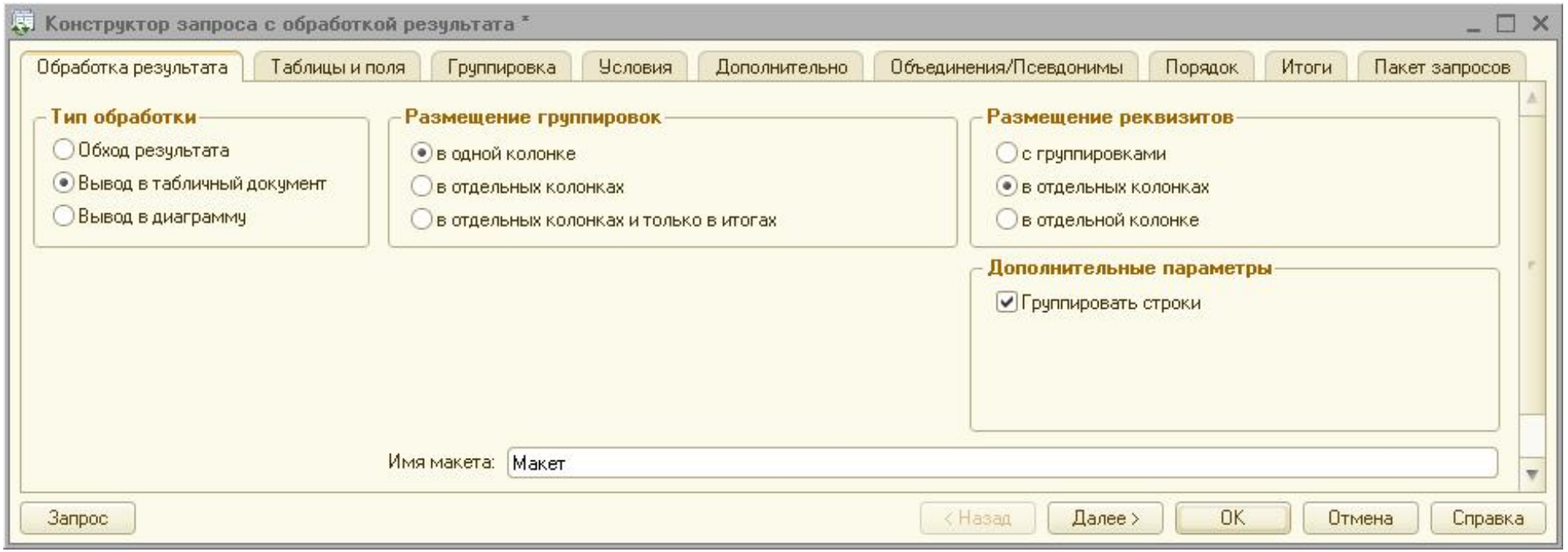
Команда «Печать накладной» документа «Расходная»:

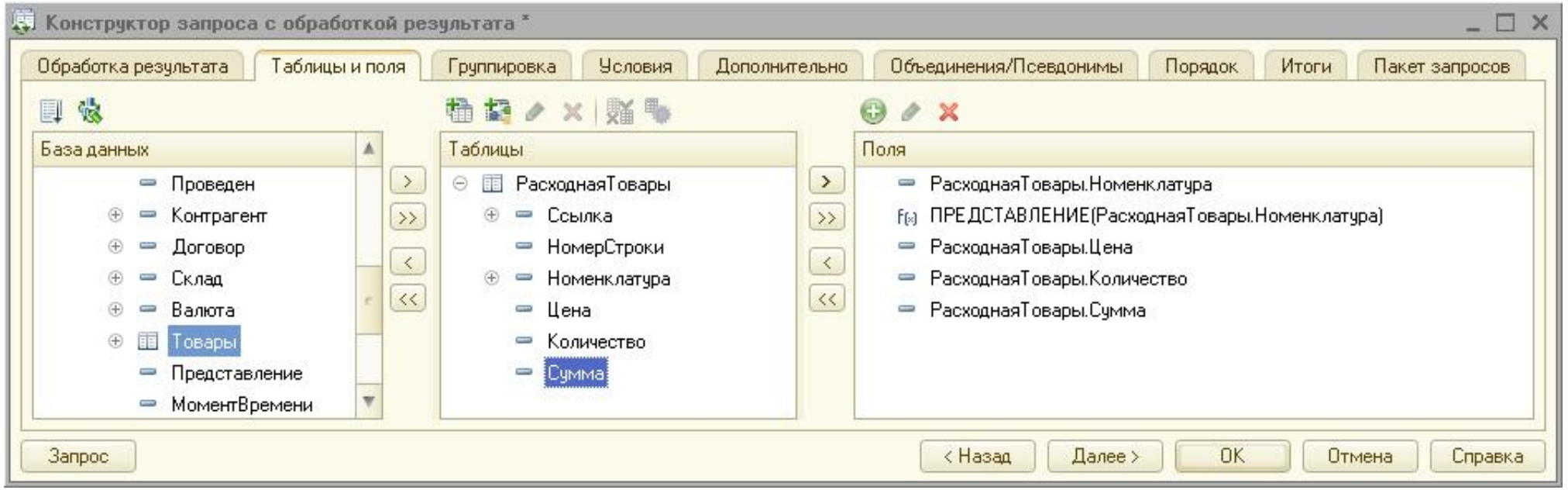
Свойство	Значение
Имя	ПечатьТОРГ12
Синоним	Печать накладной
Группа	Командная панель формы.Печать
Тип параметра команды	ДокументСсылка.Расходная

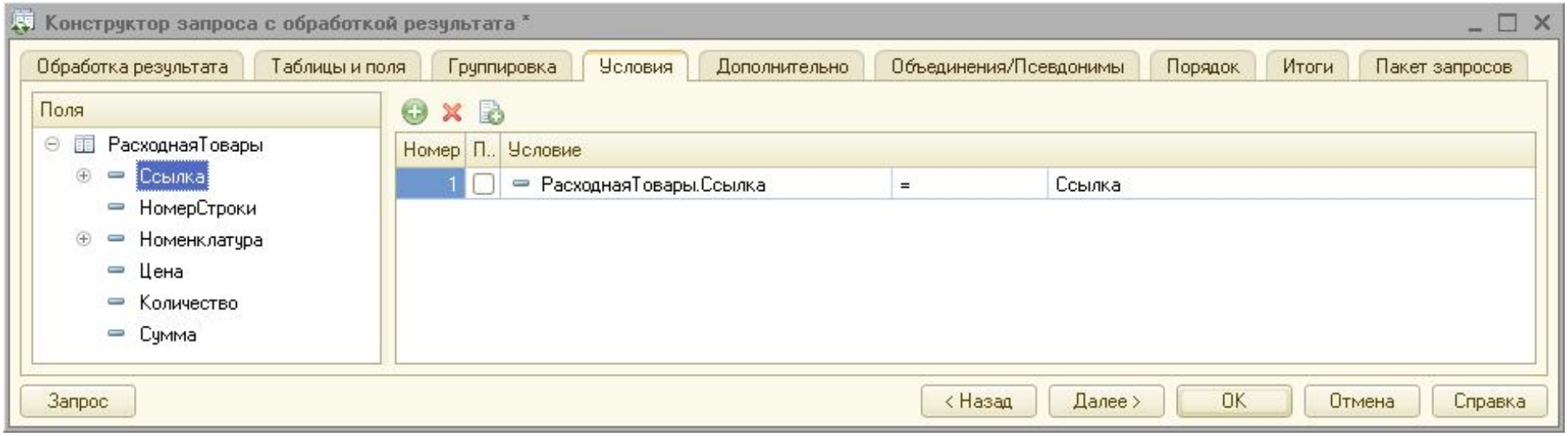
```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
	ТабДок = Новый ТабличныйДокумент;
	ПечатьТОРГ12(ПараметрыВыполнения, ТабДок)
```

```
КонецПроцедуры

&НаСервере
Процедура ПечатьТОРГ12(ПараметрыВыполнения, ТабличныйДокумент)

КонецПроцедуры
```

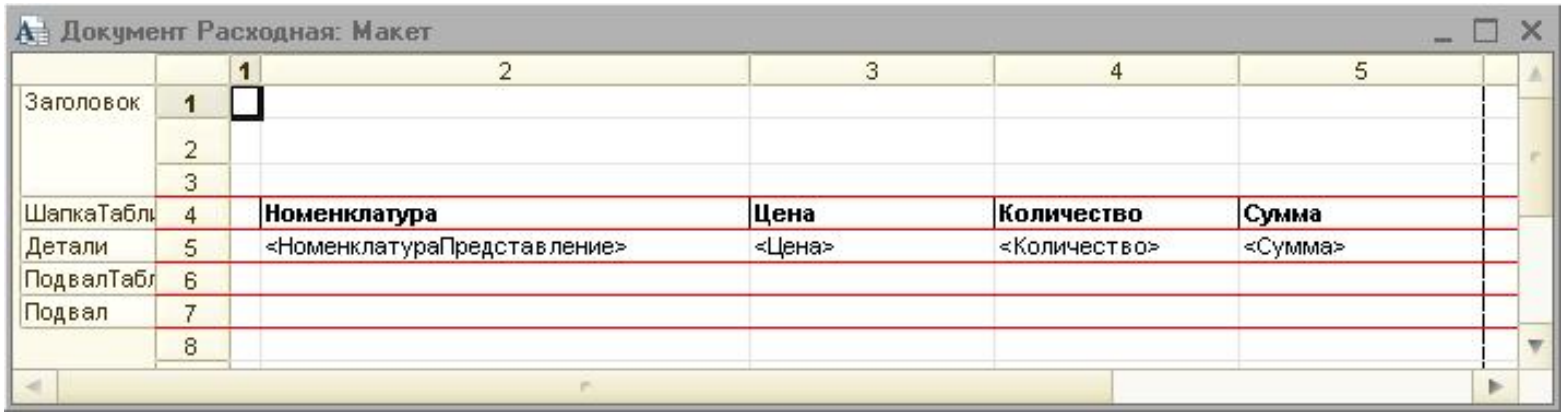
В процедуре «ПечатьТОРГ12» запустим «Конструктор запроса с обработкой результата»:




Конструктор запроса с обработкой результата *
Обработка результата
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Порядок
Итоги
Пакет запросов
База данных
Проведен
Контрагент
Договор
Склад
Валюта
Товары
Представление
МоментВремени
Таблицы
РасходнаяТовары
Ссылка
НомерСтроки
Номенклатура
Цена
Количество
Сумма
Поля
РасходнаяТовары.Номенклатура
ПРЕДСТАВЛЕНИЕ(РасходнаяТовары.Номенклатура)
РасходнаяТовары.Цена
РасходнаяТовары.Количество
РасходнаяТовары.Сумма
Запрос
< Назад
Далее >
OK
Отмена
Справка


Конструктор запроса с обработкой результата *
Обработка результата
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Порядок
Итоги
Пакет запросов
Поля
РасходнаяТовары
Ссылка
НомерСтроки
Номенклатура
Цена
Количество
Сумма
Номер
П..
Условие
1
РасходнаяТовары.Ссылка
=
Ссылка
Запрос
< Назад
Далее >
OK
Отмена
Справка

Результат:




```
&НаСервере
Процедура ПечатьТОРГ12(ПараметрыВыполнения, ТабДок)

    Макет = Документы.Расходная.ПолучитьМакет("Макет");
    Запрос = Новый Запрос;
    Запрос.Текст =
        "ВЫБРАТЬ
        |    РасходнаяТовары.Номенклатура,
        |    ПРЕДСТАВЛЕНИЕ(РасходнаяТовары.Номенклатура),
        |    РасходнаяТовары.Цена,
        |    РасходнаяТовары.Количество,
        |    РасходнаяТовары.Сумма
        |ИЗ
        |    Документ.Расходная.Товары КАК РасходнаяТовары
        |ГДЕ
        |    РасходнаяТовары.Ссылка = &Ссылка";

    Запрос.УстановитьПараметр("Ссылка", Ссылка);

    Результат = Запрос.Выполнить();

    ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
    ОбластьПодвал = Макет.ПолучитьОбласть("Подвал");
    ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");
    ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");
```

```
    ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

    ТабДок.Очистить();
    ТабДок.Вывести(ОбластьЗаголовок);
    ТабДок.Вывести(ОбластьШапкаТаблицы);
    ТабДок.НачатьАвтогруппировкуСтрок();

    ВыборкаДетальныеЗаписи = Результат.Выбрать();

    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        ОбластьДетальныхЗаписей.Параметры.Заполнить(ВыборкаДетальныеЗаписи);
        ТабДок.Вывести(ОбластьДетальныхЗаписей, ВыборкаДетальныеЗаписи.Уровень());
    КонецЦикла;

    ТабДок.ЗакончитьАвтогруппировкуСтрок();
    ТабДок.Вывести(ОбластьПодвалТаблицы);
    ТабДок.Вывести(ОбластьПодвал);

КонецПроцедуры
```

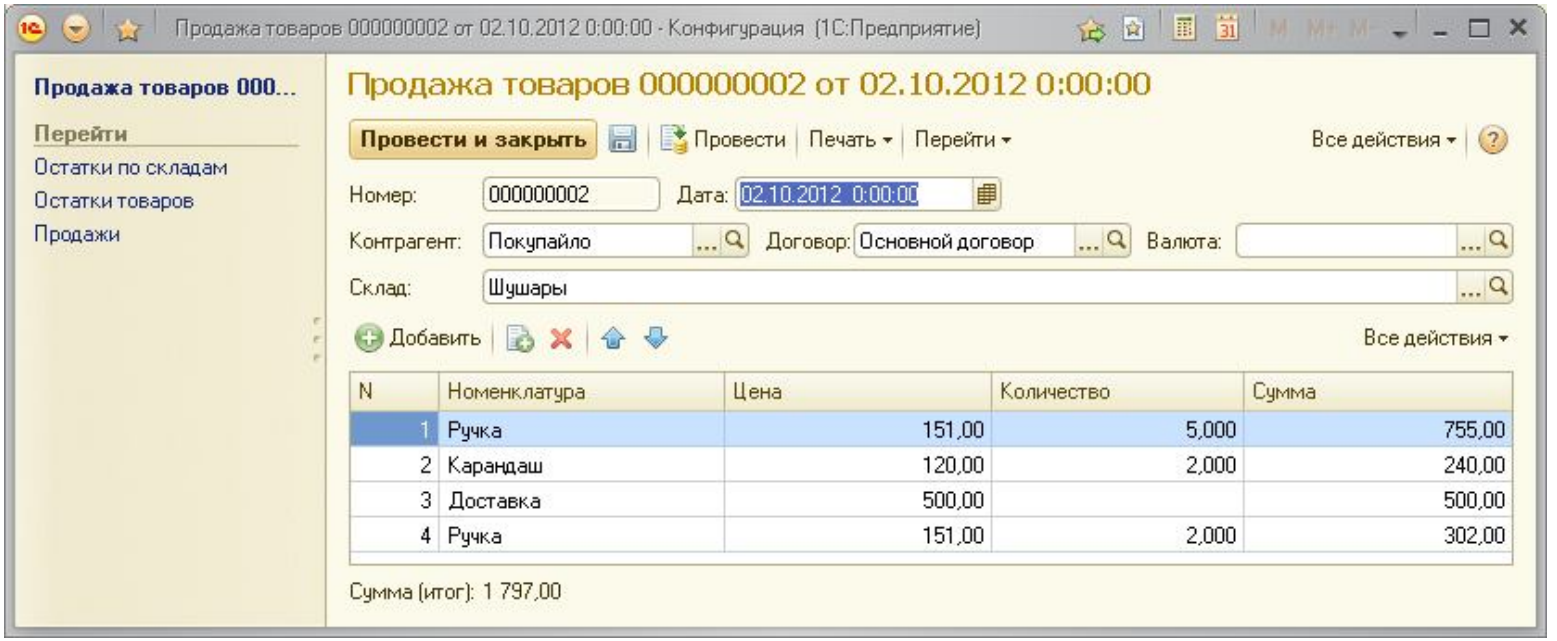
Поправим код. Итого модуль команды «Печать накладной»:

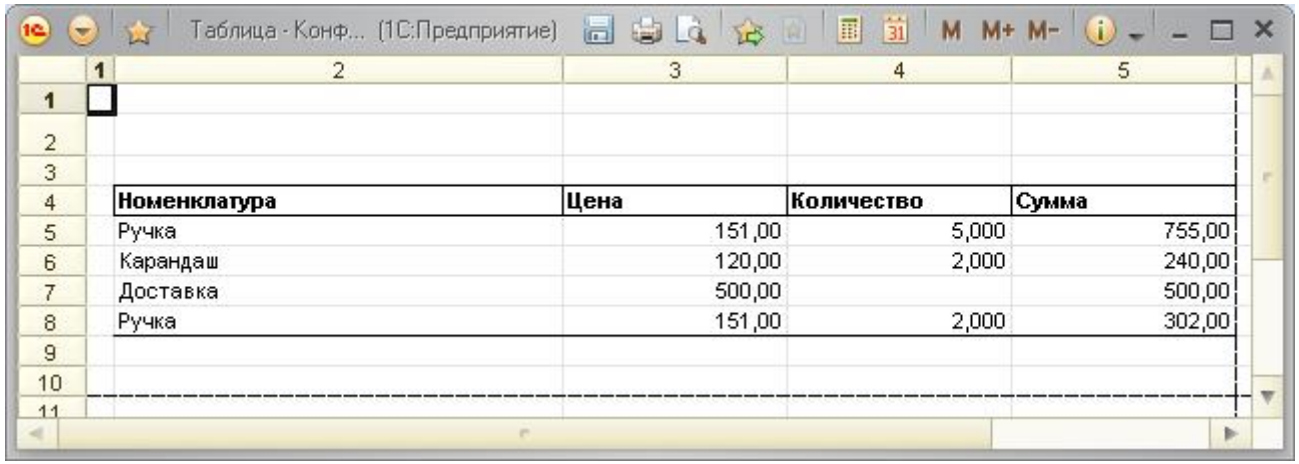
```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
    ТабДок = Новый ТабличныйДокумент;
    ПечатьТОРГ12(ПараметрКоманды, ТабДок);
    ТабДок.Показать();
КонецПроцедуры

&НаСервере
Процедура ПечатьТОРГ12(ПараметрыВыполнения, ТабДок)

    Макет = Документы.Расходная.ПолучитьМакет("Макет");
    Запрос = Новый Запрос;
    Запрос.Текст =
```

```
      "ВЫБРАТЬ
      |    РасходнаяТовары.Номенклатура,
      |    ПРЕДСТАВЛЕНИЕ(РасходнаяТовары.Номенклатура),
      |    РасходнаяТовары.Цена,
      |    РасходнаяТовары.Количество,
      |    РасходнаяТовары.Сумма
      |ИЗ
      |    Документ.Расходная.Товары КАК РасходнаяТовары
      |ГДЕ
      |    РасходнаяТовары.Ссылка = &Ссылка";

Запрос.УстановитьПараметр("Ссылка", ПараметрыВыполнения);

Результат = Запрос.Выполнить();

ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
ОбластьПодвал = Макет.ПолучитьОбласть("Подвал");
ОбластьШапкаТаблицы = Макет.ПолучитьОбласть("ШапкаТаблицы");
ОбластьПодвалТаблицы = Макет.ПолучитьОбласть("ПодвалТаблицы");
ОбластьДетальныхЗаписей = Макет.ПолучитьОбласть("Детали");

ТабДок.Очистить();
ТабДок.Вывести(ОбластьЗаголовок);
ТабДок.Вывести(ОбластьШапкаТаблицы);
ТабДок.НачатьАвтогруппировкуСтрок();

ВыборкаДетальныеЗаписи = Результат.Выбрать();

Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
     ОбластьДетальныхЗаписей.Параметры.Заполнить(ВыборкаДетальныеЗаписи);
     ТабДок.Вывести(ОбластьДетальныхЗаписей, ВыборкаДетальныеЗаписи.Уровень());
КонецЦикла;

ТабДок.ЗакончитьАвтогруппировкуСтрок();
ТабДок.Вывести(ОбластьПодвалТаблицы);
ТабДок.Вывести(ОбластьПодвал);
```

```
КонецПроцедуры
```

В режиме исполнения:





Теперь, во всех документах, где будут нужны печатные формы, команду «Печать» можно сделать одинаковой с помощью группы команд «Печать», а не рисовать отдельно для каждой формы.

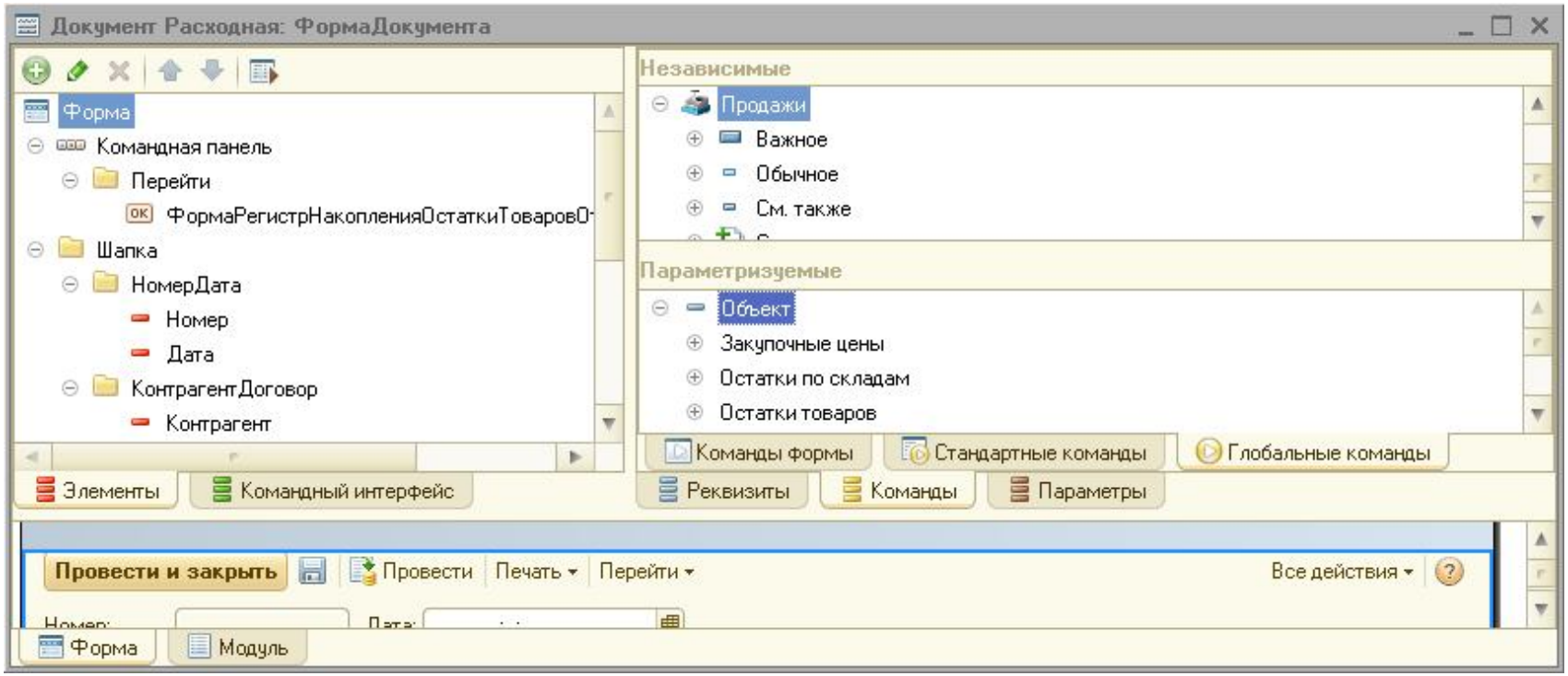
Групповые команды:

В свойствах команды:

Свойство	Значение
Режим использования параметров	Множественный

Тогда в точку вызова команды можно будет передавать несколько значений (массив). Используется чтобы формировать печатные формы нескольких документов сразу.

При редактировании управляемых форм можно обращаться к:
Глобальным командам (общим командам). Также в них можно обращаться к командам объектов, связанных с текущим объектом.



Стандартным командам.

Взаиморасчеты

Надо сформировать отчет «Ведомость (остатки) по взаиморасчетам»:

Контрагент	Наш долг	Долг контрагентов

Сначала создадим регистр накопления «Взаиморасчеты»:

Свойство	Значение
Имя	Взаиморасчеты
Синоним	Взаиморасчеты
Вид регистра	Остатки

Свойство	Значение
Закладка «Подсистемы»	Закупки Продажи
Закладка «Данные»	
Измерения	
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Запрет незаполненных значений	Истина
Ресурсы	
Сумма	
Имя	Сумма
Синоним	Сумма долга контрагента
Тип	Число 15, 2
Закладка «Регистраторы»	Приходная Расходная

В модуле объекта документа «Приходная» опишем формирование движений по регистру накопления «Взаиморасчеты»:

```
Процедура ОбработкаПроведения(Отказ, Режим)

    Движения.ЗакупочныеЦены.Записывать    = Истина;
    Движения.ОстаткиТоваров.Записывать    = Истина;
    Движения.ОстаткиПоСкладам.Записывать = Истина;
    Движения.Взаиморасчеты.Записывать     = Истина;

    Движения.ЗакупочныеЦены.Очистить();
    Движения.ОстаткиТоваров.Очистить();
    Движения.ОстаткиПоСкладам.Очистить();
    Движения.Взаиморасчеты.Очистить();

    Для Каждого ТекСтрокаТовары Из Товары Цикл
        Движение               = Движения.ЗакупочныеЦены.Добавить();
```

```
        Движение.Период       = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Контрагент   = Контрагент;
        Движение.Цена         = ТекСтрокаТовары.Цена;

        Движение              = Движения.ОстаткиТоваров.ДобавитьПриход();
        Движение.Период       = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Количество   = ТекСтрокаТовары.Количество;
        Движение.Стоимость    = ТекСтрокаТовары.Сумма;
        Движение.Партия       = Ссылка;

        Движение              = Движения.ОстаткиПоСкладам.ДобавитьПриход();
        Движение.Период       = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Количество   = ТекСтрокаТовары.Количество;
        Движение.Склад        = Склад;
    КонецЦикла;

    СуммаДокумента = Товары.Итог("Сумма");

    Если НЕ Валюта.Пустая() Тогда
        Курс = ОбщиеМеханизмы.ПолучитьКурсВалюты(Валюта, Дата);
        СуммаДокумента = СуммаДокумента * Курс;
    КонецЕсли;

    Запись = Движения.Взаиморасчеты.ДобавитьРасход();
    Запись.Период     = Дата;
    Запись.Контрагент = Контрагент;
    Запись.Сумма      = СуммаДокумента;

КонецПроцедуры
```

Сформируем аналогичные движения в документе «Расходная», только вместо «Расход» будет «Приход»:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

    Движения.ОстаткиТоваров.Записывать   = Истина;
    Движения.ОстаткиПоСкладам.Записывать = Истина;
    Движения.Продажи.Записывать          = Истина;
    Движения.Взаиморасчеты.Записывать    = Истина;

    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    РасходнаяТовары.Номенклатура,
    |    СУММА(РасходнаяТовары.Количество) КАК Количество,
    |    СУММА(Сумма) КАК Выручка,
    |    МИНИМУМ(РасходнаяТовары.НомерСтроки) КАК НомерСтроки
    |ПОМЕСТИТЬ ДокТЧ
    |ИЗ
    |    Документ.Расходная.Товары КАК РасходнаяТовары
    |ГДЕ
    |    РасходнаяТовары.Ссылка = &Ссылка
    |    И НЕ РасходнаяТовары.Номенклатура.Услуга

    |СГРУППИРОВАТЬ ПО
    |    РасходнаяТовары.Номенклатура
    |;
    |ВЫБРАТЬ
    |    ДокТЧ.НомерСтроки,
    |    ДокТЧ.Номенклатура,
    |    ДокТЧ.Выручка,
    |    ДокТЧ.Количество,
    |    Остатки.Партия,
    |    ЕСТЬNULL(Остатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
    |    ЕСТЬNULL(Остатки.СтоимостьОстаток, 0) КАК СтоимостьОстаток,
    |    ЕСТЬNULL(ОстаткиСКЛ.КоличествоОстаток, 0) КАК ОстатокНаСкладе
    |ИЗ
    |    ДокТЧ КАК ДокТЧ
    |         ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиТоваров.Остатки(
```

```
    |                    &МоментВремени,
    |                    Номенклатура В
    |                         (ВЫБРАТЬ
    |                              ДокТЧ.Номенклатура
    |                         ИЗ
    |                              ДокТЧ КАК ДокТЧ)) КАК Остатки
    |          ПО ДокТЧ.Номенклатура = Остатки.Номенклатура
    |          ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиПоСкладам.Остатки(
    |                    &МоментВремени,
    |                    Склад = &Склад
    |                         И Номенклатура В
    |                              (ВЫБРАТЬ
    |                                   ДокТЧ.Номенклатура
    |                              ИЗ
    |                                   ДокТЧ КАК ДокТЧ)) КАК ОстаткиСКЛ
    |          ПО ДокТЧ.Номенклатура = ОстаткиСКЛ.Номенклатура
    |УПОРЯДОЧИТЬ ПО Партия.Дата ВОЗР
    |ИТОГИ МИНИМУМ(Количество), МИНИМУМ(Выручка), МИНИМУМ(ОстатокНаСкладе), МИНИМУМ(НомерСтроки)
    |ПО ДокТЧ.Номенклатура";

    Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
    Запрос.УстановитьПараметр("Ссылка", Ссылка);
    Запрос.УстановитьПараметр("Склад", Склад);

    Результат = Запрос.Выполнить();

    Выборка = Результат.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

    Пока Выборка.Следующий() Цикл
         Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
              Сообщение = Новый СообщениеПользователю;
              Сообщение.Текст = "Не хватает товара """ + Выборка.Номенклатура + """ из необходимых " +
Выборка.Количество + " в наличии осталось только " + Выборка.ОстатокНаСкладе;
              Сообщение.Поле = "Товары[" + (Выборка.НомерСтроки - 1) + "].Количество";
              Сообщение.УстановитьДанные(ЭтотОбъект);
              Сообщение.Сообщить();
```

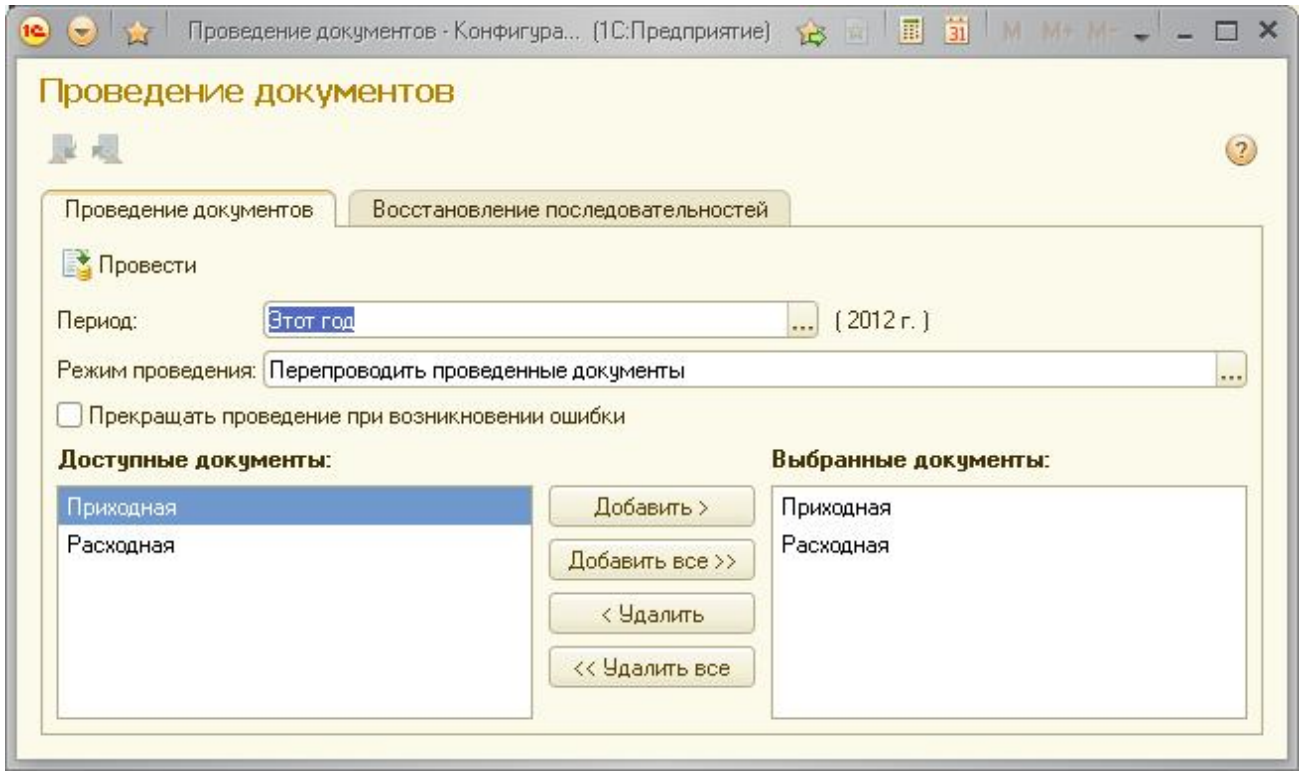
```
            Отказ = Истина;
            Движения.ОстаткиТоваров.Записывать    = Ложь;
            Движения.ОстаткиПоСкладам.Записывать = Ложь;
      КонецЕсли;
КонецЦикла;

Если Отказ Тогда
      Возврат;
КонецЕсли;

Выборка.Сбросить();

Пока Выборка.Следующий() Цикл

      ОсталосьСписать = Выборка.Количество;

      СебестоимостьИтого = 0;

      ВыборкаПартии = Выборка.Выбрать();
      Пока ВыборкаПартии.Следующий() И ОсталосьСписать <> 0 Цикл

            Списать       = Мин(ОсталосьСписать, ВыборкаПартии.КоличествоОстаток);
            Себестоимость = Списать / ВыборкаПартии.КоличествоОстаток * ВыборкаПартии.СтоимостьОстаток;

            СебестоимостьИтого = СебестоимостьИтого + Себестоимость;

            Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
            Движение.Период       = Дата;
            Движение.Номенклатура = Выборка.Номенклатура;
            Движение.Количество   = Списать;
            Движение.Стоимость    = Себестоимость;
            Движение.Партия       = ВыборкаПартии.Партия;

            ОсталосьСписать = ОсталосьСписать - Списать;
      КонецЦикла;
```

```
            Движение = Движения.ОстаткиПоСкладам.ДобавитьРасход();
            Движение.Период       = Дата;
            Движение.Номенклатура = Выборка.Номенклатура;
            Движение.Количество   = Выборка.Количество;
            Движение.Склад        = Склад;

            Движение = Движения.Продажи.Добавить();
            Движение.Период        = Дата;
            Движение.Номенклатура  = Выборка.Номенклатура;
            Движение.Количество    = Выборка.Количество;
            Движение.Выручка       = Выборка.Выручка;
            Движение.Себестоимость = СебестоимостьИтого;
            Движение.Контаргент    = Контрагент;
      КонецЦикла;

      СуммаДокумента = Товары.Итог("Сумма");

      Если НЕ Валюта.Пустая() Тогда
            Курс = ОбщиеМеханизмы.ПолучитьКурсВалюты(Валюта, Дата);
            СуммаДокумента = СуммаДокумента * Курс;
      КонецЕсли;

      Запись = Движения.Взаиморасчеты.ДобавитьПриход();
      Запись.Период     = Дата;
      Запись.Контрагент = Контрагент;
      Запись.Сумма      = СуммаДокумента;

КонецПроцедуры
```

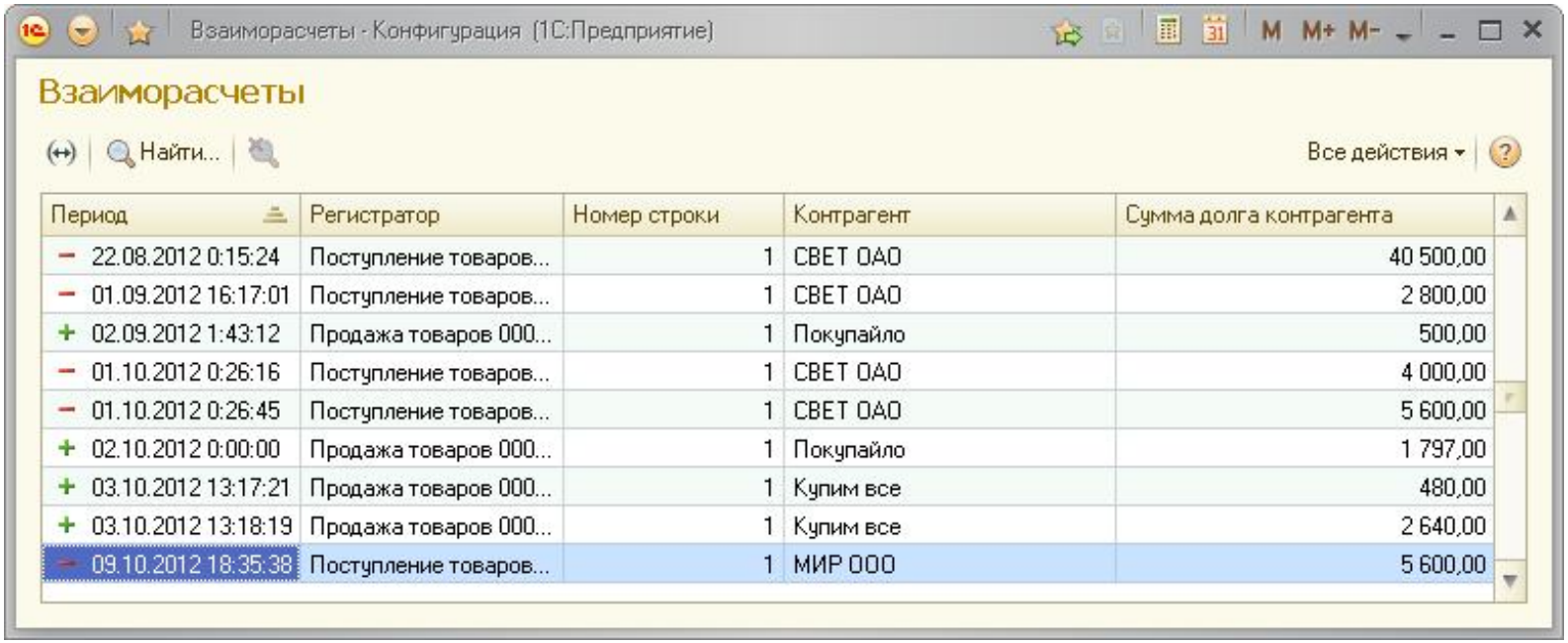
В типовых конфигурациях реализована универсальная технология проведения. В общем модуле есть специальная процедура, которая формирует движения по регистрам. В нее из документа передаются параметры, влияющие на формирование движения. Все остальные необходимые действия (контроль остатков, партионное списание и т.д.) производятся в данной универсальной процедуре.

В режиме исполнения:

Перепроведем все документы «Приходная» и «Расходная»:
«Все функции – Стандартные – Проведение документов»



Посмотрим регистр накопления «Взаиморасчеты»:



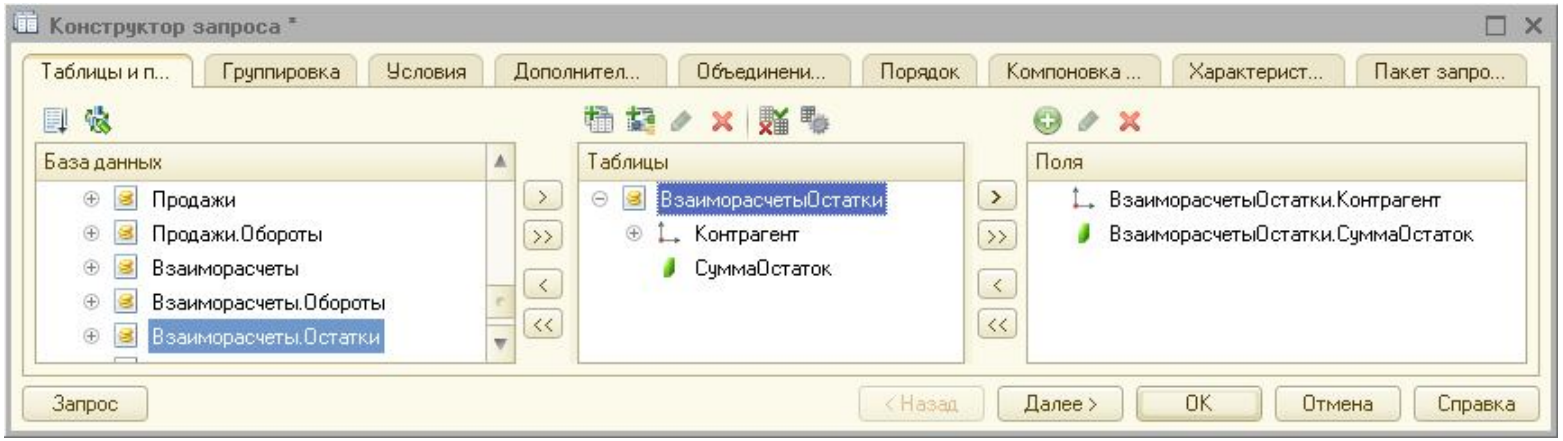
Период	Регистратор	Номер строки	Контрагент	Сумма долга контрагента
22.08.2012 0:15:24	Поступление товаров...	1	СВЕТ ОАО	40 500,00
01.09.2012 16:17:01	Поступление товаров...	1	СВЕТ ОАО	2 800,00
02.09.2012 1:43:12	Продажа товаров 000...	1	Покупайло	500,00
01.10.2012 0:26:16	Поступление товаров...	1	СВЕТ ОАО	4 000,00
01.10.2012 0:26:45	Поступление товаров...	1	СВЕТ ОАО	5 600,00
02.10.2012 0:00:00	Продажа товаров 000...	1	Покупайло	1 797,00
03.10.2012 13:17:21	Продажа товаров 000...	1	Купим все	480,00
03.10.2012 13:18:19	Продажа товаров 000...	1	Купим все	2 640,00
09.10.2012 18:35:38	Поступление товаров...	1	МИР ООО	5 600,00

По первой строке:
В данном документе указан валютный договор, т.е. цены в нем указаны в валюте. В регистре все цены должны указываться в рублях, поэтому сумма документа в валюте переведена в рубли по курсу на дату документа.

Создадим отчет «Взаиморасчеты»

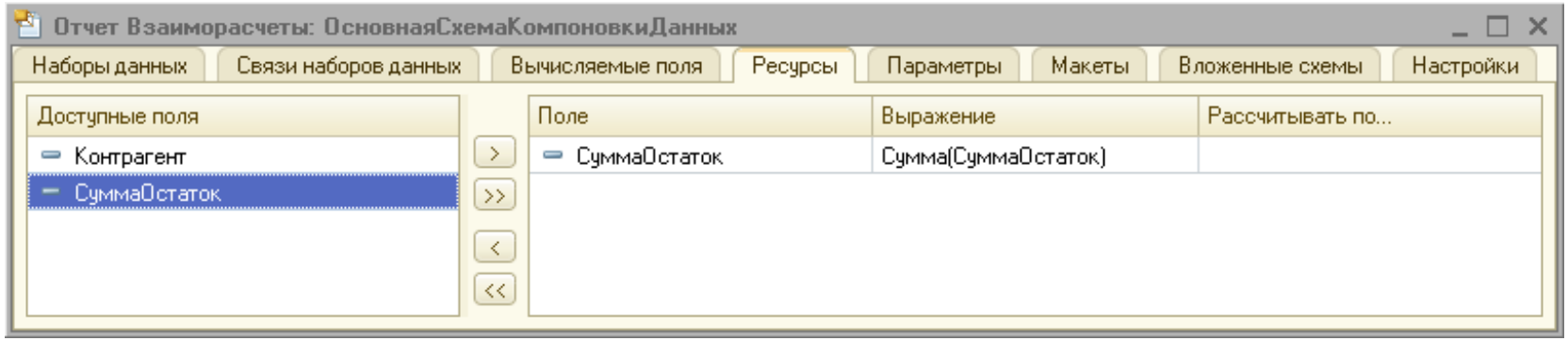
Свойство	Значение
Имя	Взаиморасчеты
Синоним	Взаиморасчеты
Закладка «Подсистемы»	Продажи

«Открыть СКД – Добавить набор данных-запрос – Конструктор запроса»

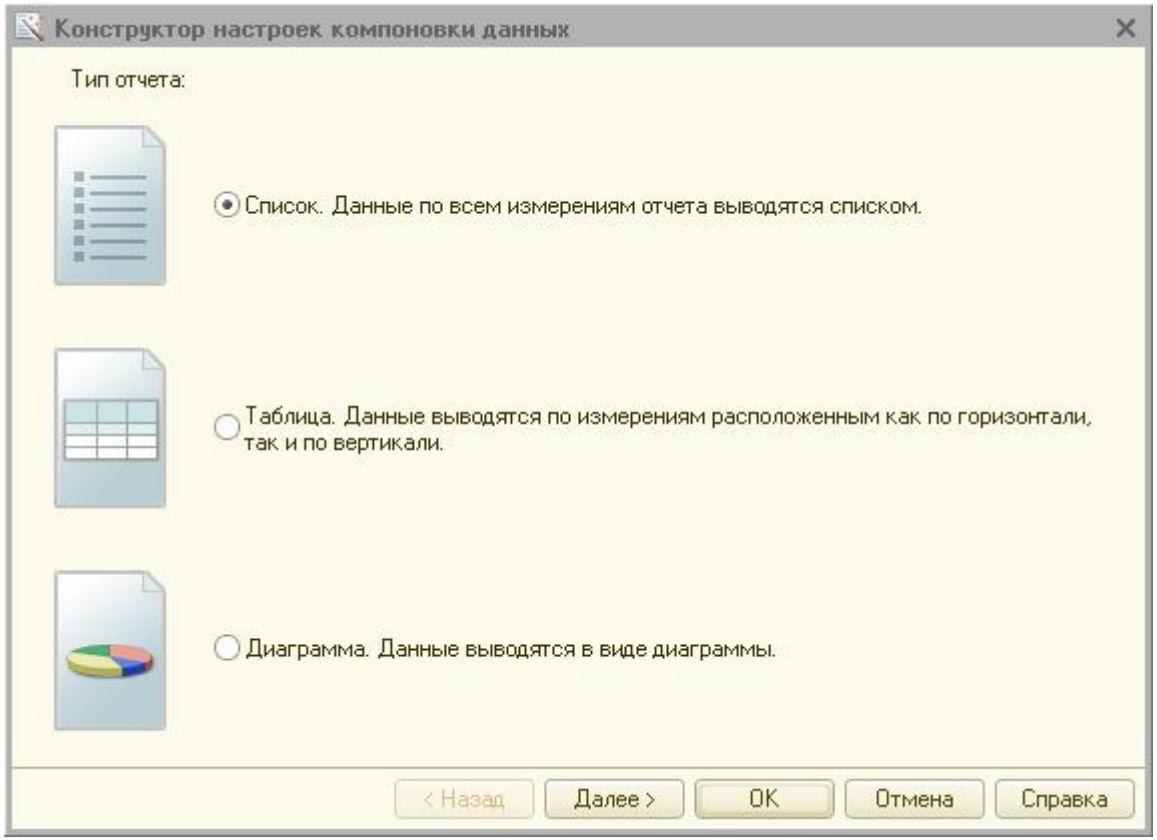


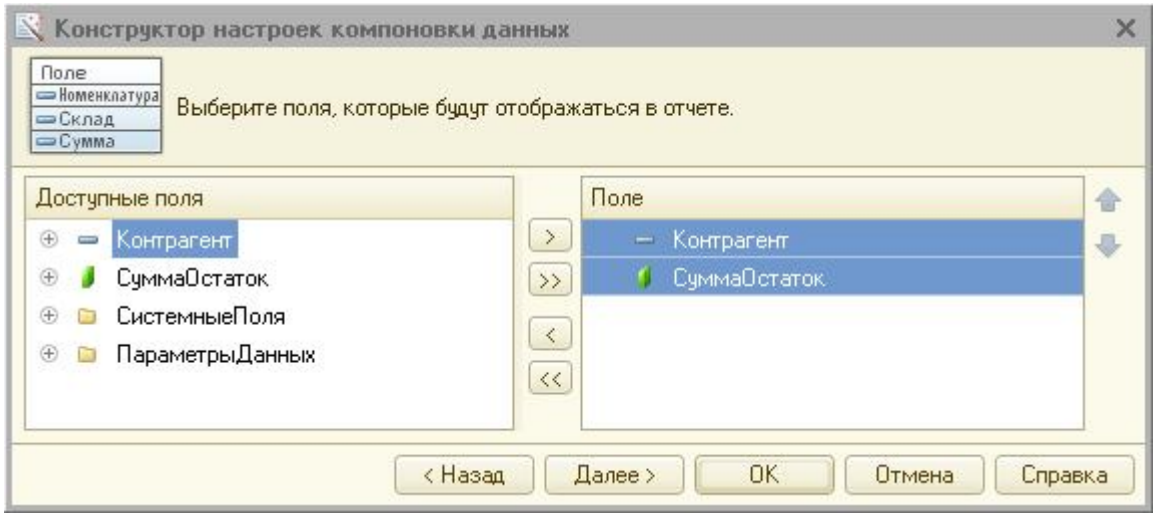
В данном случае остаток («+» или «-») будет отображаться в одной колонке. Оставим пока так, чтобы показать, как пользователь в режиме исполнения с помощью СКД может разбить «+» и «-» остатки по разным колонкам.

В СКД:

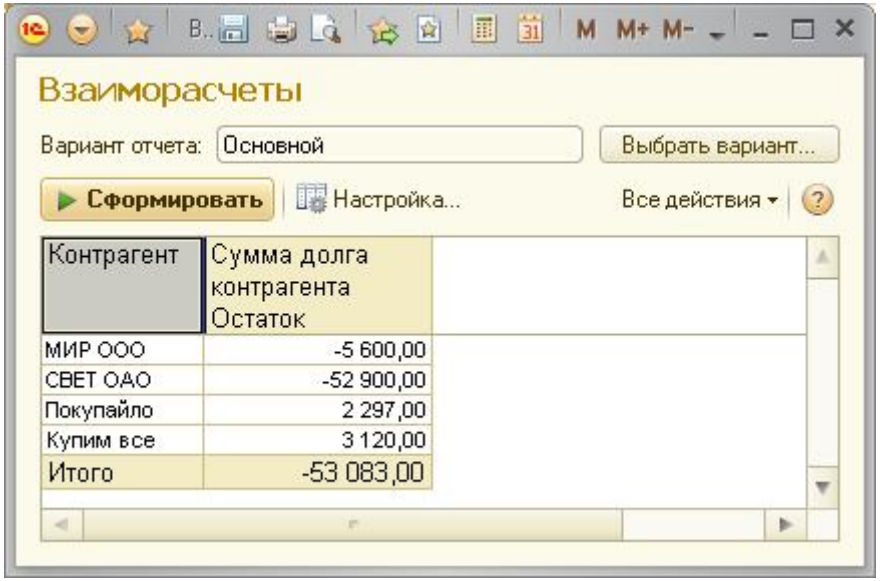


Закладка «Настройки» - сделаем настройку по умолчанию:


Конструктор настроек компоновки данных
Тип отчета:
Список. Данные по всем измерениям отчета выводятся списком.
Таблица. Данные выводятся по измерениям расположенным как по горизонтали, так и по вертикали.
Диаграмма. Данные выводятся в виде диаграммы.
< Назад
Далее >
OK
Отмена
Справка

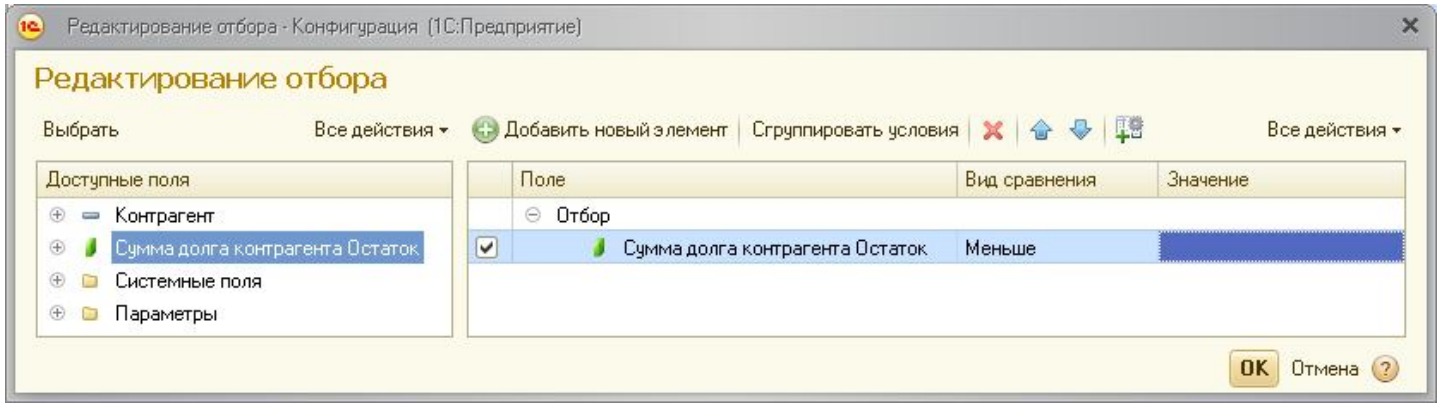

Конструктор настроек компоновки данных
Поле
Номенклатура
Склад
Сумма
Выберите поля, которые будут отображаться в отчете.
Доступные поля
Контрагент
СуммаОстаток
СистемныеПоля
ПараметрыДанных
Поле
Контрагент
СуммаОстаток
< Назад
Далее >
OK
Отмена
Справка

В режиме исполнения:



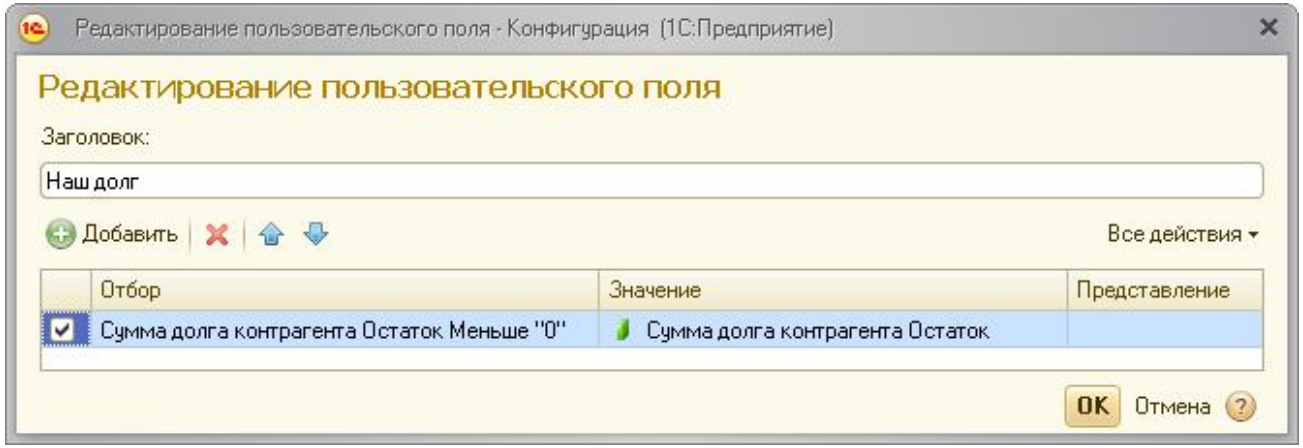
«Все действия – Изменить вариант – закладка Пользовательские поля – Добавить – Новое поле выбор»

Заполнение поля «Отбор»

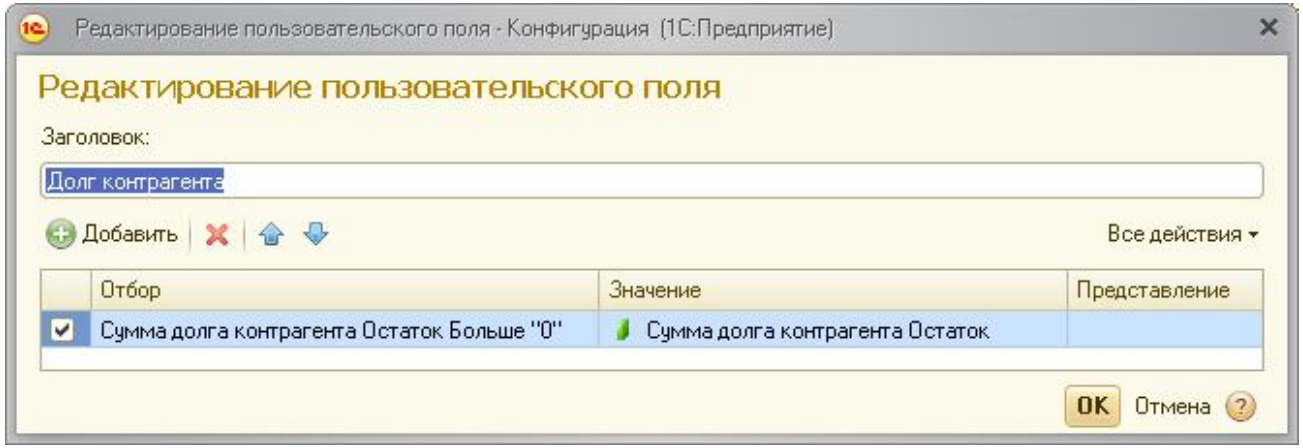


Заполнение поля «Значение»: «Поле компоновки данных - Сумма долга контрагента Остаток»

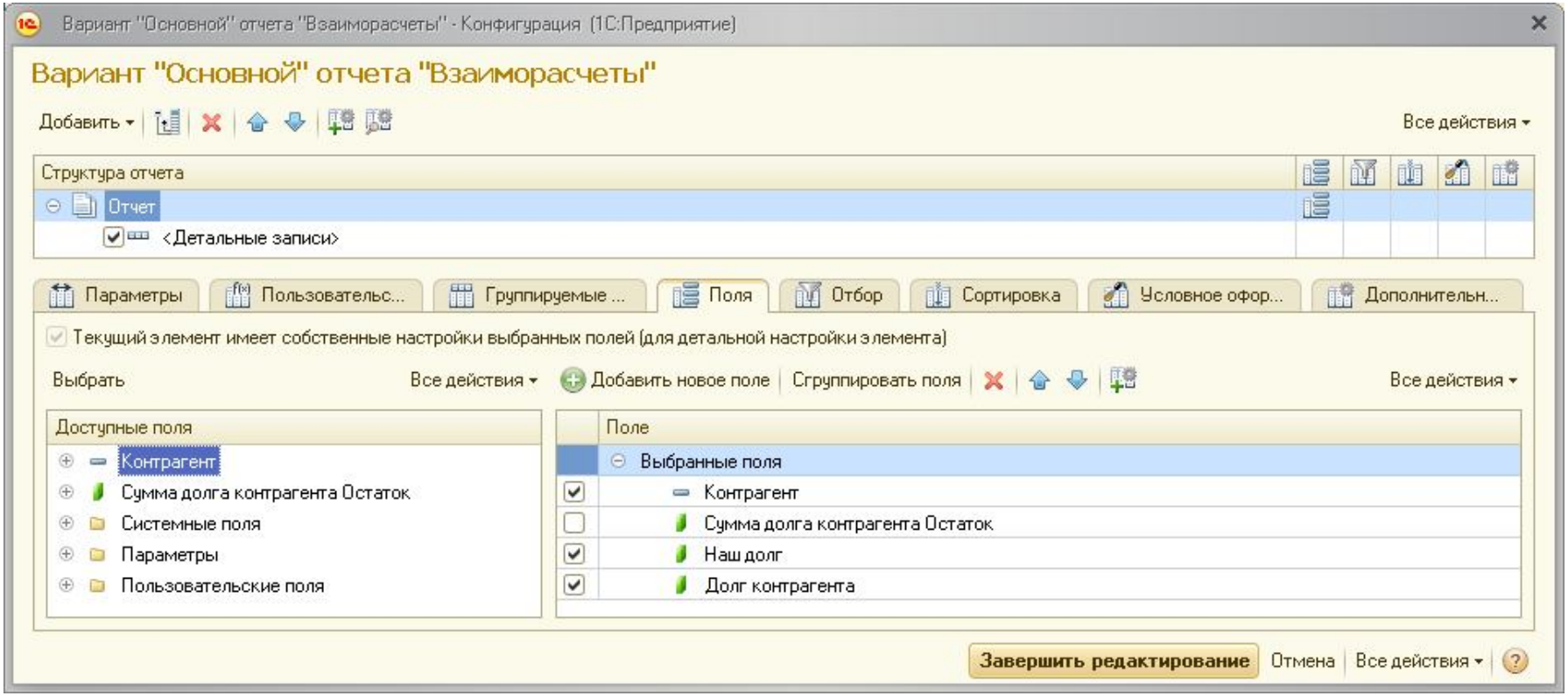
Итого пользовательское поле:



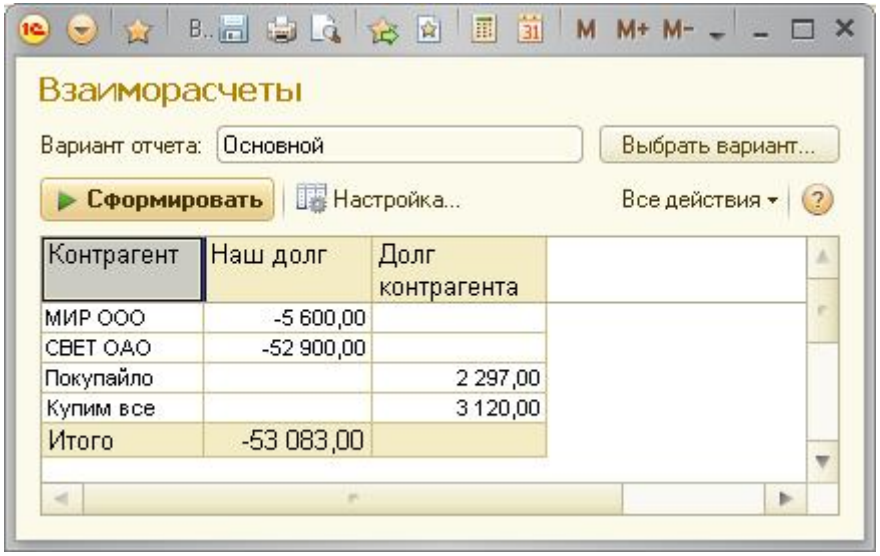
Аналогично создадим поле «Долг контрагента», только условие будет:



Далее «СКД – закладка «Поля» - выведем пользовательские поля:



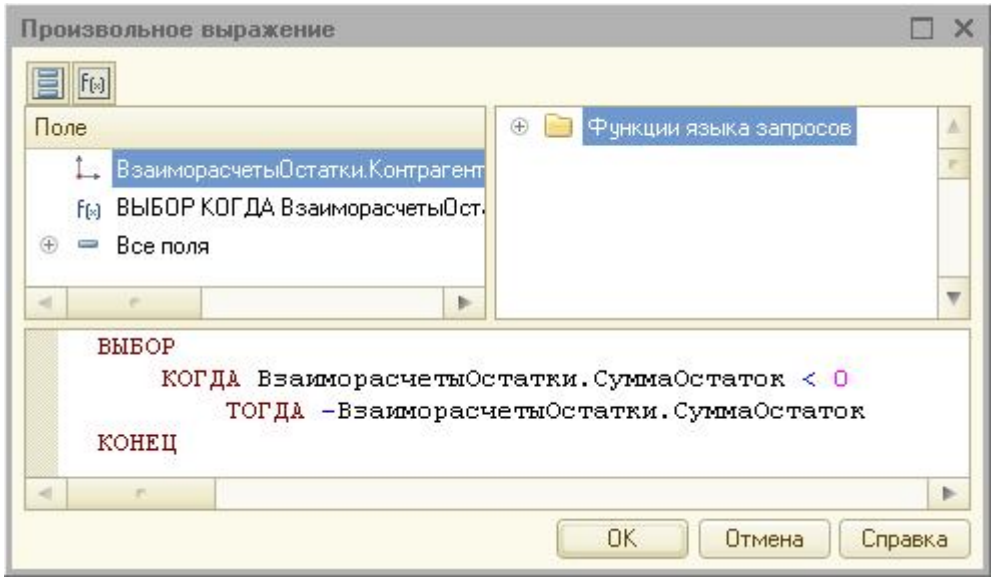
Сформируем отчет еще раз:

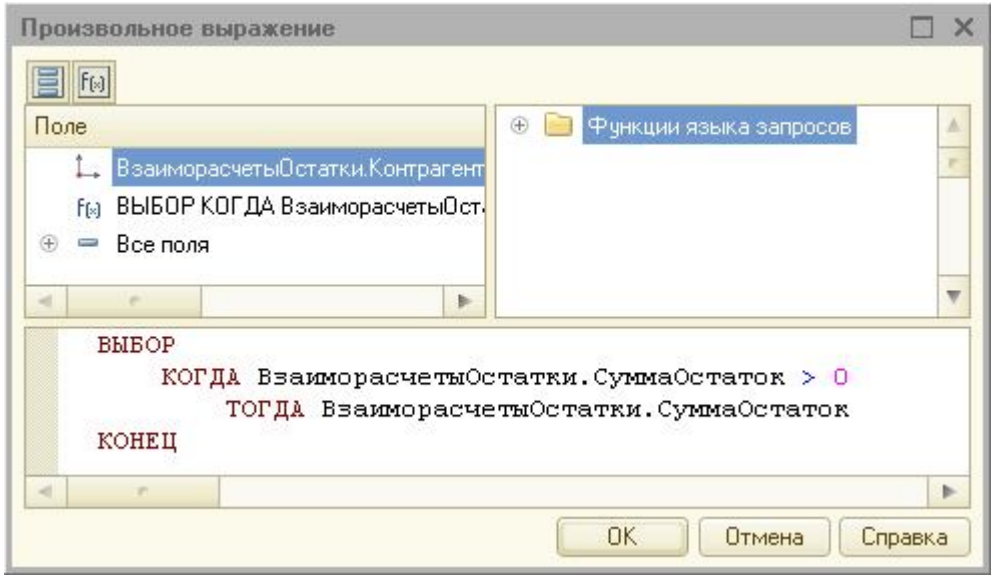


Контрагент	Наш долг	Долг контрагента
МИР ООО	-5 600,00	
СВЕТ ОАО	-52 900,00	
Покупайло		2 297,00
Купим все		3 120,00
Итого	-53 083,00	

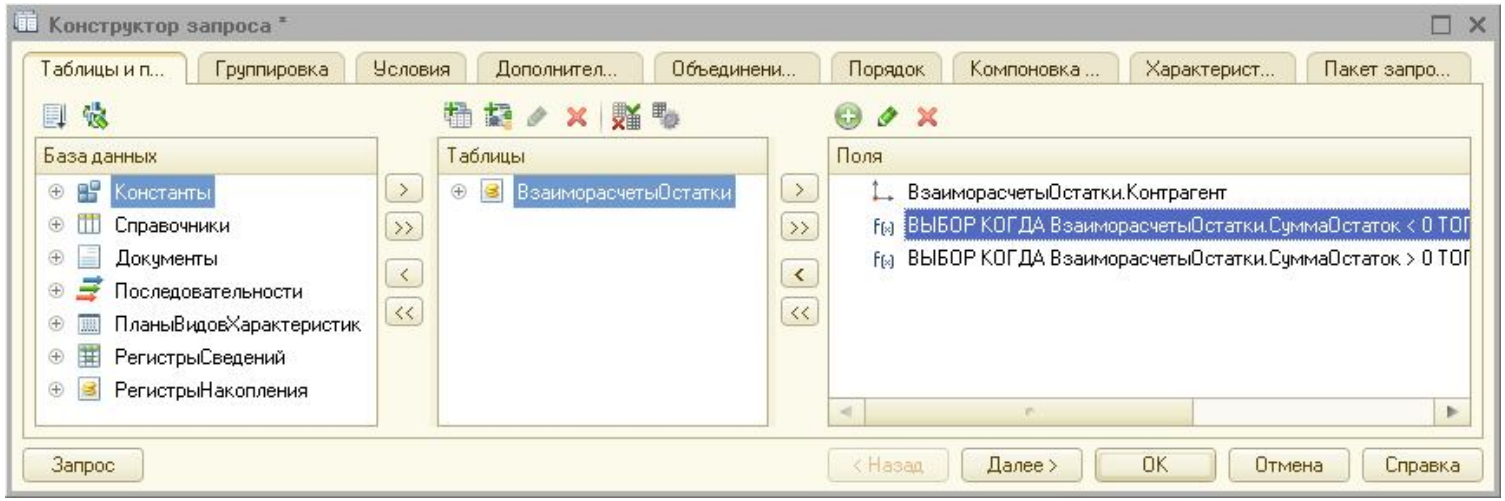
Доработаем отчет так, чтобы долг разделялся еще на этапе формирования запроса:

Создадим поля:

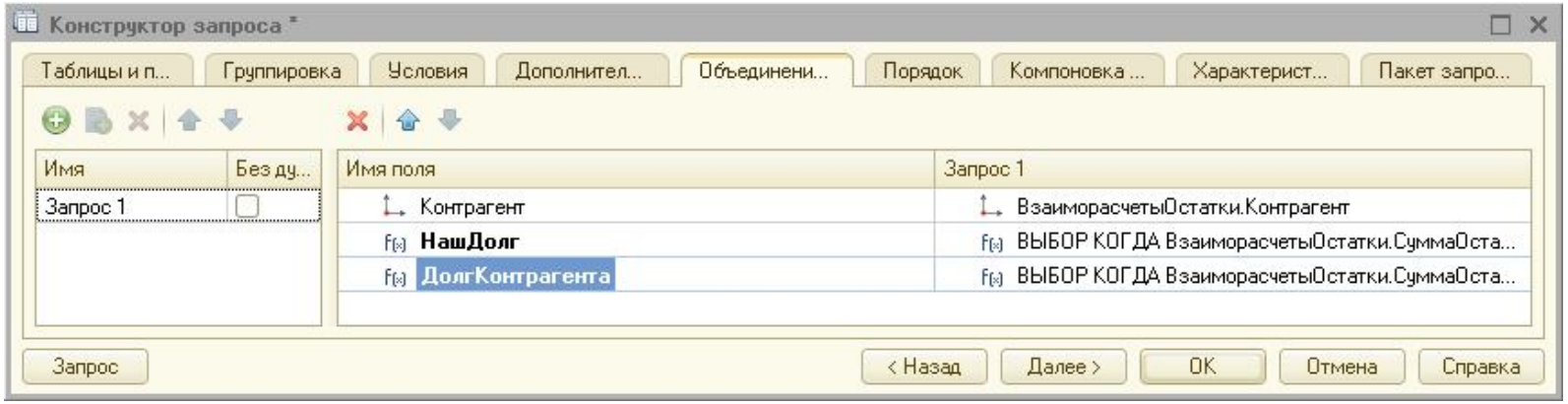




Полученные поля:

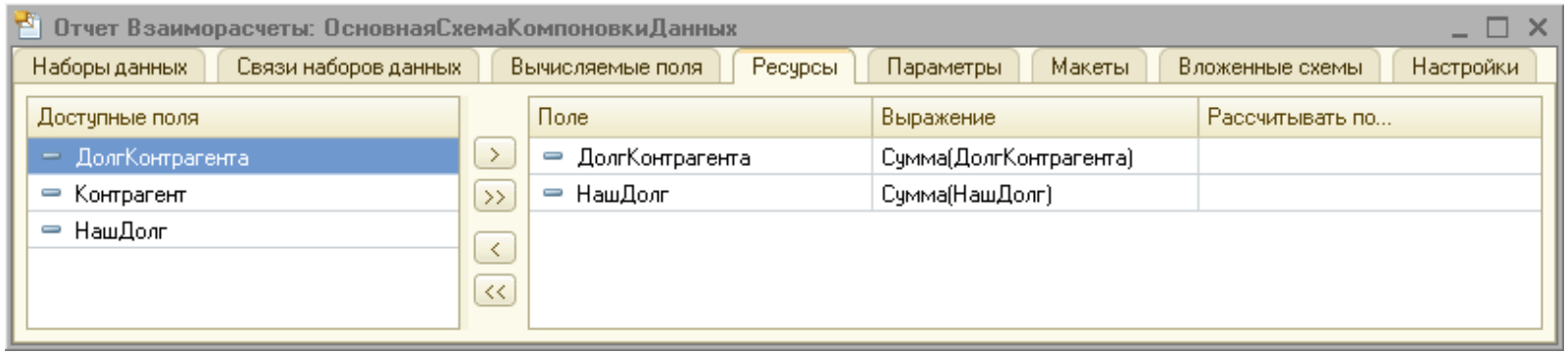


Переименуем поля:



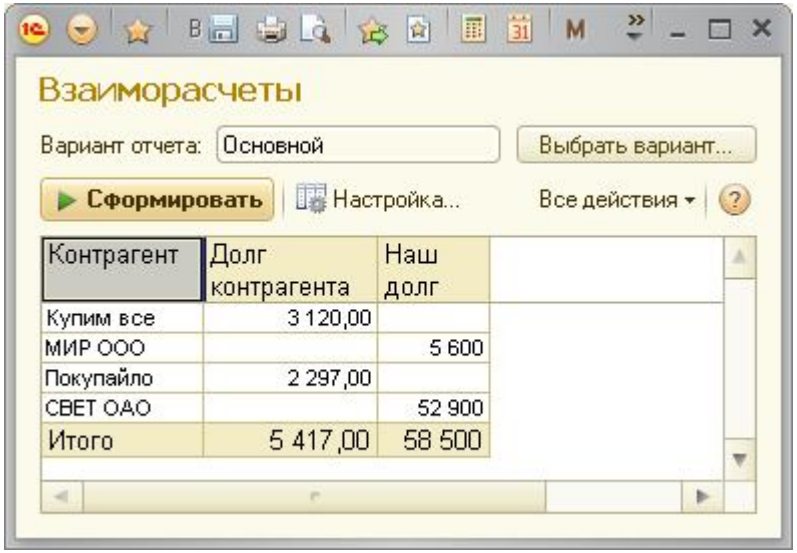
В СКД:

Добавим вновь созданные поля в Ресурсы:



И переделаем представление по умолчанию (закладка «Настройки»). Дополнительно только добавим группировку по Контрагенту.

В режиме исполнения:



Усложним отчет. Сделаем, чтобы отчет формировался по Контрагентам и по географическому признаку Контрагента:

Создадим План видов характеристик «Свойства контрагентов»:

Свойство	Значение
Имя	СвойстваКонтрагентов
Синоним	Свойства контрагентов
Тип значения характеристик	СправочникСсылка.СвойстваХарактеристик
Дополнительные значения характеристик	СвойстваХарактеристик
Закладка «Подсистемы»	Предприятие

Значения Плана видов характеристик будут храниться в справочнике «Свойства характеристик»:

Свойство	Значение
Имя	СвойстваХарактеристик
Синоним	Свойства характеристик
Закладка «Подсистемы»	Предприятие

Свойство	Значение
Закладка «Владельцы»	ПланВидовХарактеристик.СвойстваКонтрагентов

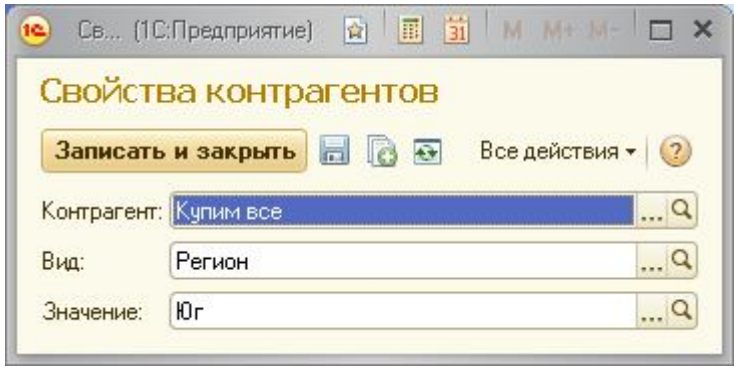
Сочетание Свойства и Контрагента будем хранить в Регистре сведений «Свойства контрагентов»:

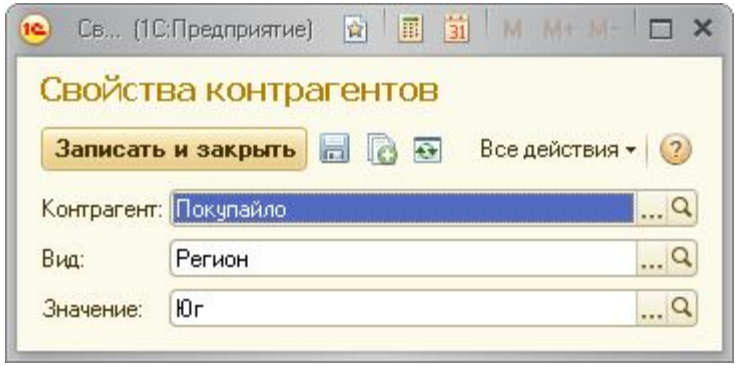
Свойство	Значение
Имя	СвойстваКонтрагентов
Синоним	Свойства контрагентов
Закладка «Подсистемы»	Предприятие
Закладка «Данные»	
Измерения	
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Ведущее	Истина
Основной отбор	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Вид	
Имя	Вид
Синоним	Вид
Тип	ПланВидовХарактеристикСсылка.СвойстваКонтрагентов
Ведущее	Истина
Основной отбор	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Ресурсы	
Значение	
Имя	Значение
Синоним	Значение
Тип	Характеристика.СвойстваКонтрагентов
Связи параметров выбора	Отбор.Владелец(Вид)

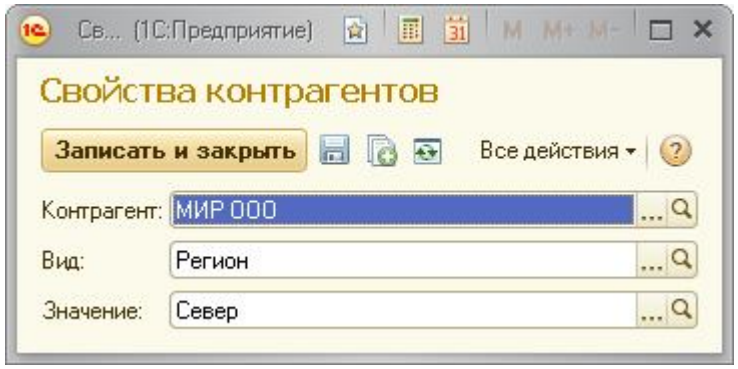
Свойство	Значение
Ресурсы	
Значение	
Связь по типу	Вид

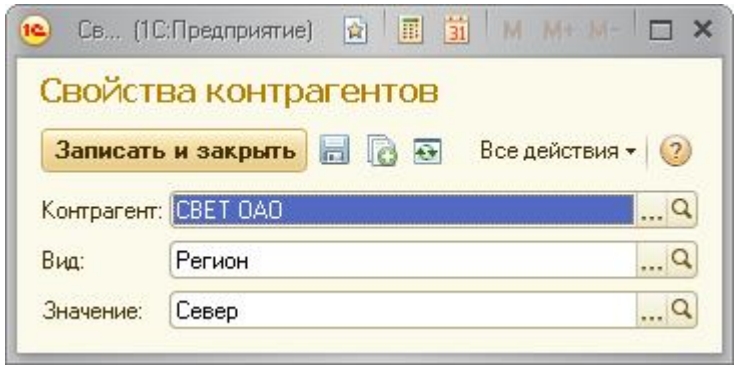
В режиме исполнения:

Создадим элемент в Регистре сведений «Свойства контрагентов»:







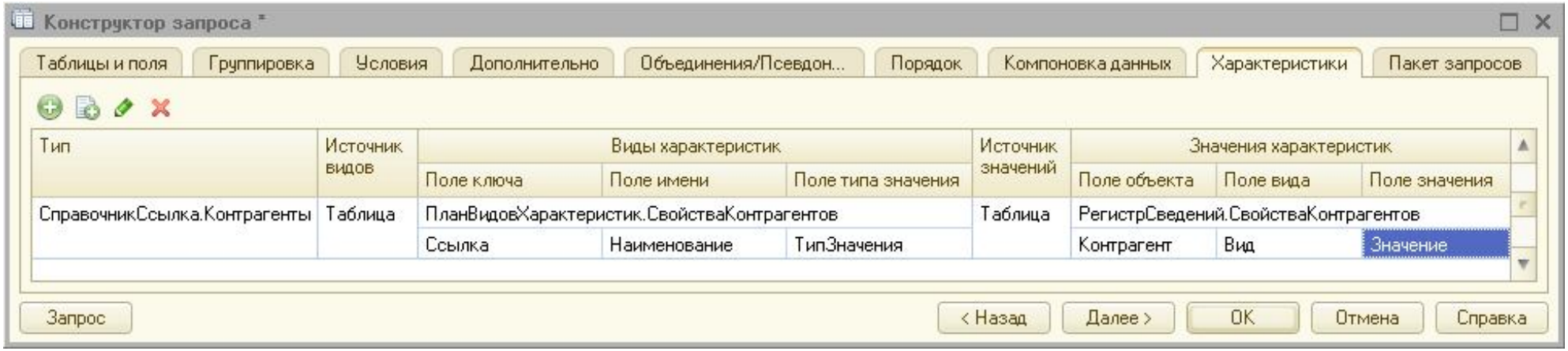


Теперь подключим данный механизм к отчету по взаиморасчетам.

«Отчет Взаиморасчеты – СКД - Конструктор запроса - Характеристики» - указывается связь текущего отчета с двумя таблицами:

- С видами характеристик (в примере это План видов характеристик «Свойства контрагентов») – левая сторона. В примере «Регион».
- Со значениями характеристик (в примере это Регистр сведений «Свойства контрагентов») – правая сторона. В примере это:

Контрагент	Вид	Значение
Купим все	Регион	Юг
Покупайло	Регион	Юг
МИР ООО	Регион	Север
СВЕТ ОАО	Регион	Север



Тип – для какого элемента (типа значения) задаются характеристики.

Итого текст запроса:

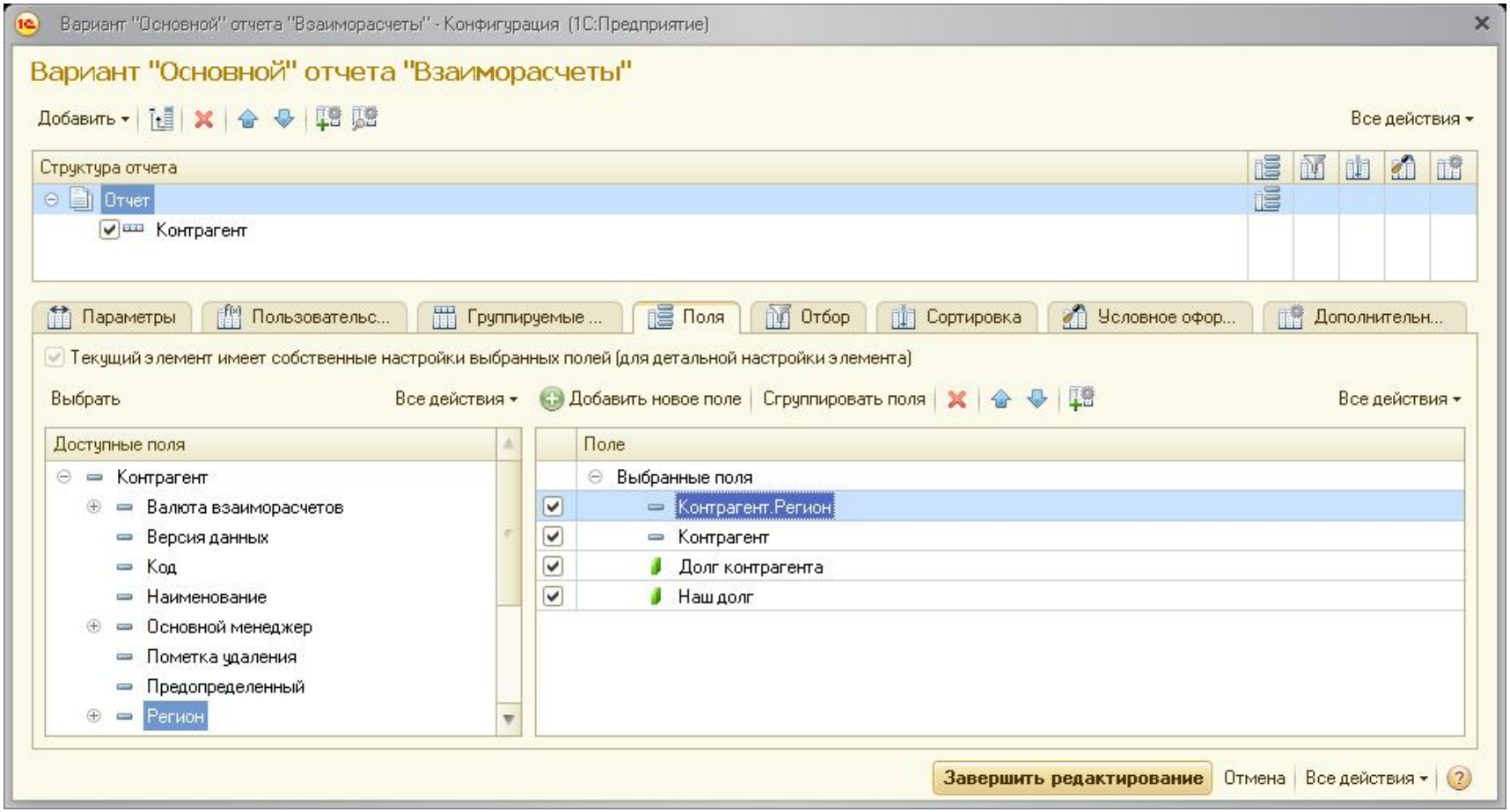
```
ВЫБРАТЬ
     ВзаиморасчетыОстатки.Контрагент,
     ВЫБОР
          КОГДА ВзаиморасчетыОстатки.СуммаОстаток < 0
               ТОГДА -ВзаиморасчетыОстатки.СуммаОстаток
     КОНЕЦ КАК НашДолг,
     ВЫБОР
          КОГДА ВзаиморасчетыОстатки.СуммаОстаток > 0
               ТОГДА ВзаиморасчетыОстатки.СуммаОстаток
     КОНЕЦ КАК ДолгКонтрагента
ИЗ
     РегистрНакопления.Взаиморасчеты.Остатки КАК ВзаиморасчетыОстатки
{ХАРАКТЕРИСТИКИ
     ТИП(Справочник.Контрагенты)
     ВИДЫХАРАКТЕРИСТИК ПланВидовХарактеристик.СвойстваКонтрагентов
     ПОЛЕКЛЮЧА Ссылка
     ПОЛЕИМЕНИ Наименование
     ПОЛЕТИПАЗНАЧЕНИЯ ТипЗначения
```

```
ЗНАЧЕНИЯХАРАКТЕРИСТИК РегистрСведений.СвойстваКонтрагентов
ПОЛЕОБЪЕКТА Контрагент
ПОЛЕВИДА Вид
ПОЛЕЗНАЧЕНИЯ Значение }
```

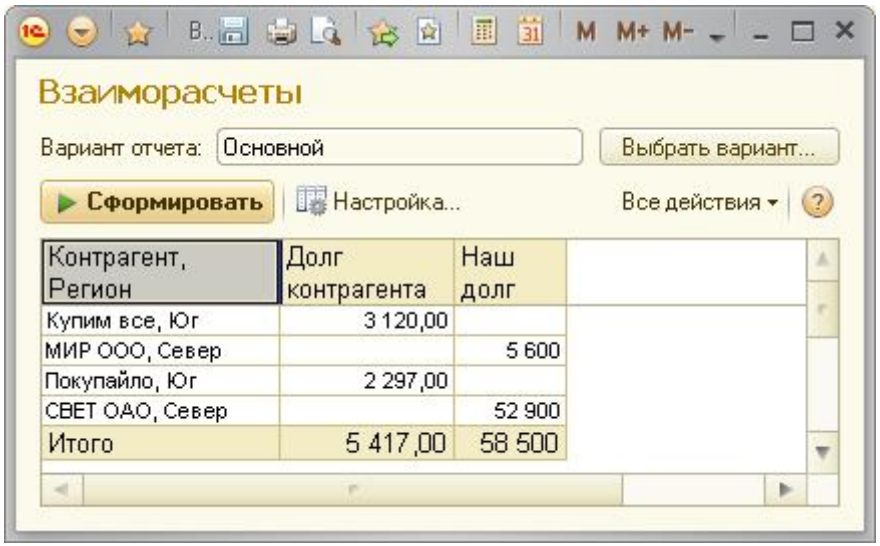
В режиме исполнения:

«Отчет Взаиморасчеты - Все действия – Изменить вариант - Поля»:

Будем выводить данные по Контрагенту и Региону:



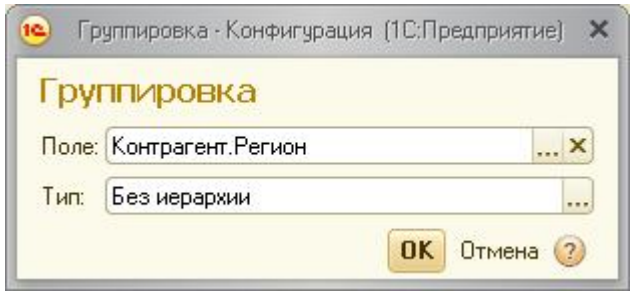
Результат:

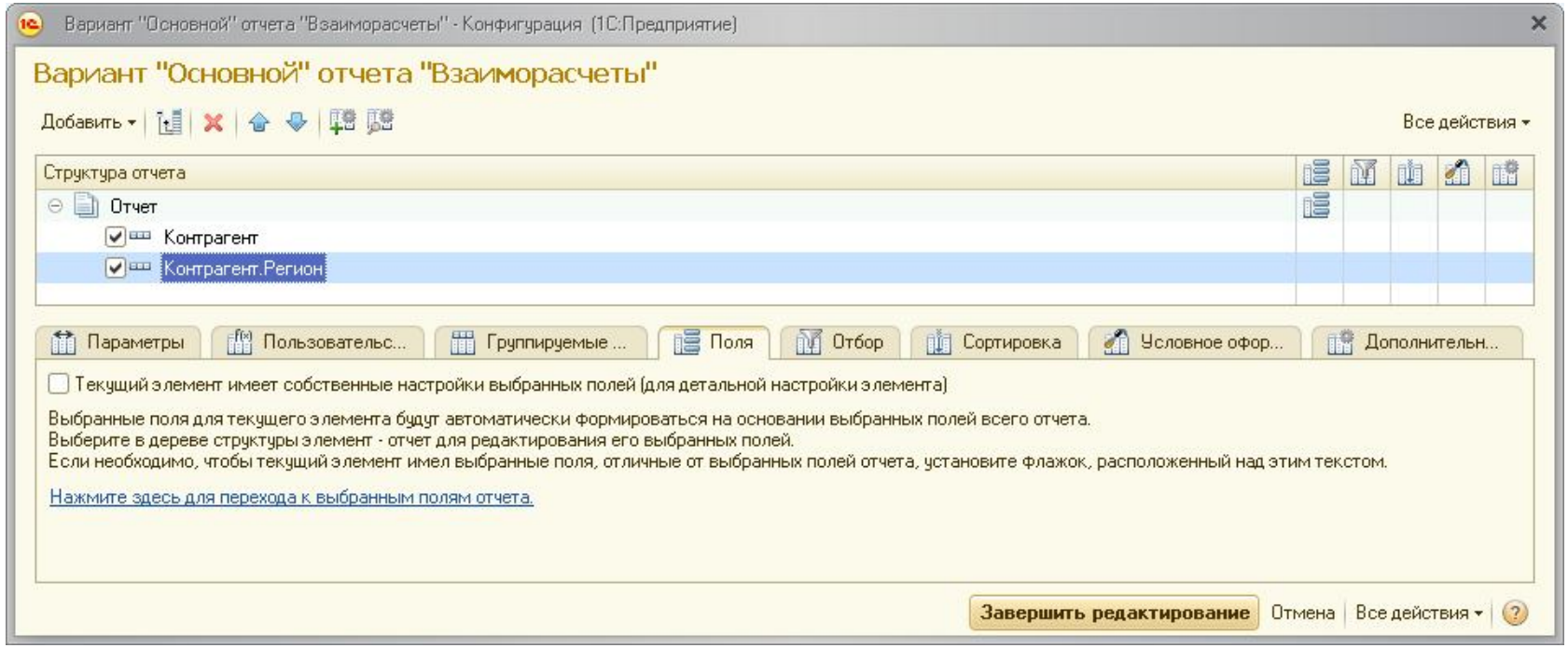


Контрагент, Регион	Долг контрагента	Наш долг
Купим все, Юг	3 120,00	
МИР ООО, Север		5 600
Покупайло, Юг	2 297,00	
СВЕТ ОАО, Север		52 900
Итого	5 417,00	58 500

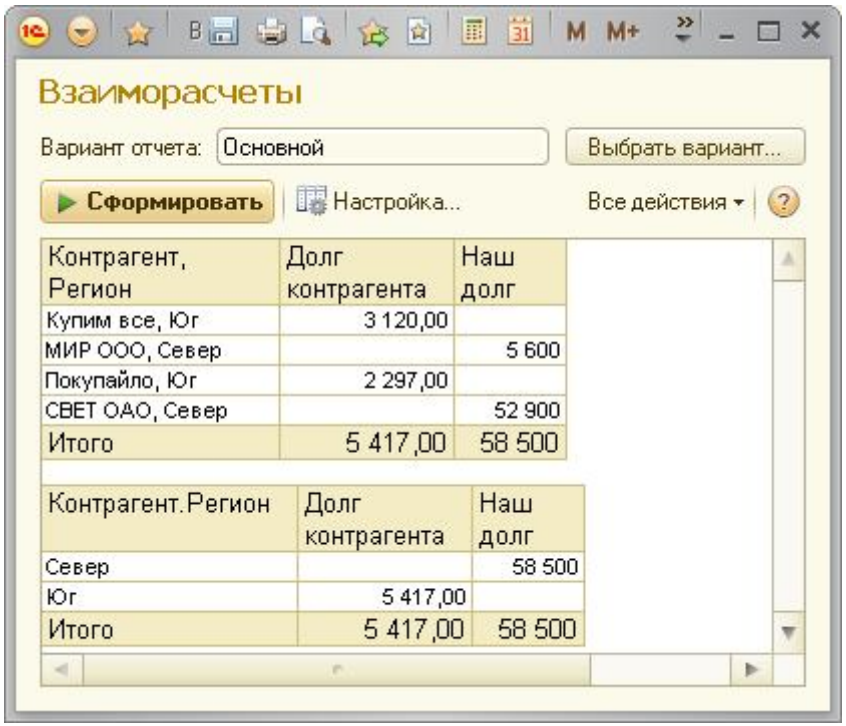
Можно сделать группировку:

«Все действия – Изменить вариант – Добавить – Новая группировка»:




Вариант "Основной" отчета "Взаиморасчеты" - Конфигурация (1С:Предприятие)
Вариант "Основной" отчета "Взаиморасчеты"
Добавить
Все действия
Структура отчета
Отчет
Контрагент
Контрагент.Регион
Параметры
Пользовательс...
Группируемые ...
Поля
Отбор
Сортировка
Условное офор...
Дополнительн...
Текущий элемент имеет собственные настройки выбранных полей (для детальной настройки элемента)
Выбранные поля для текущего элемента будут автоматически формироваться на основании выбранных полей всего отчета.
Выберите в дереве структуры элемент - отчет для редактирования его выбранных полей.
Если необходимо, чтобы текущий элемент имел выбранные поля, отличные от выбранных полей отчета, установите флажок, расположенный над этим текстом.
Нажмите здесь для перехода к выбранным полям отчета.
Завершить редактирование
Отмена
Все действия

Результат:

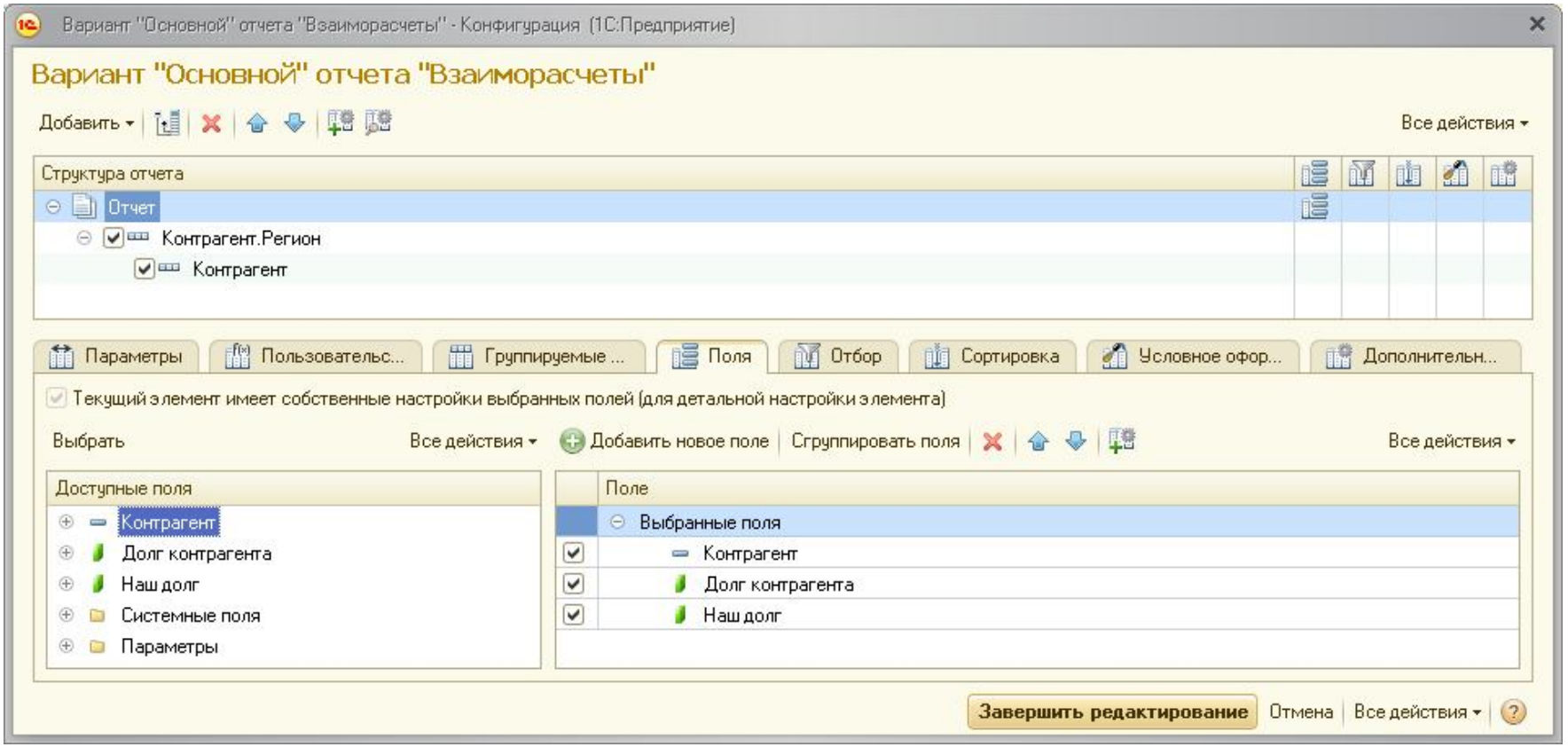


Контрагент, Регион	Долг контрагента	Наш долг
Купим все, Юг	3 120,00	
МИР ООО, Север		5 600
Покупайло, Юг	2 297,00	
СВЕТ ОАО, Север		52 900
Итого	5 417,00	58 500

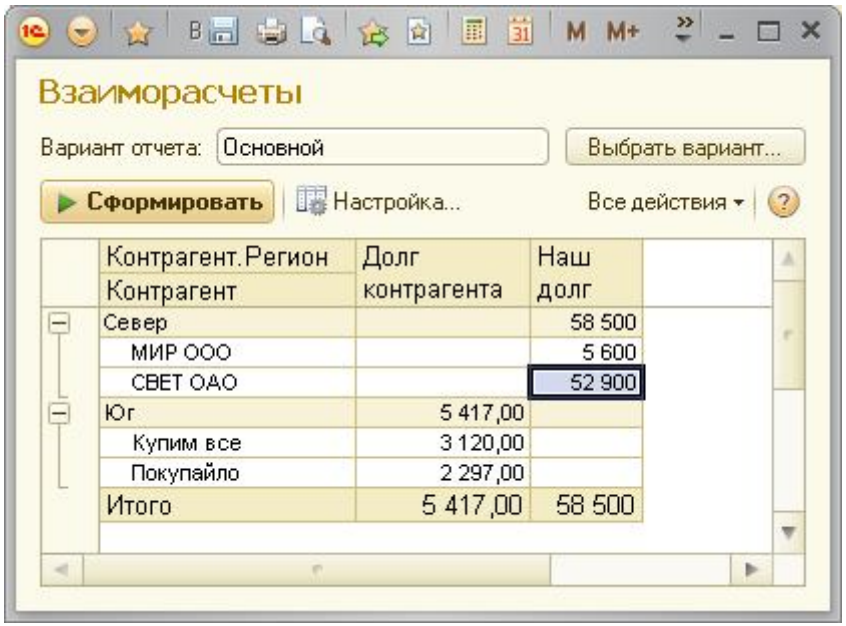
Контрагент.Регион	Долг контрагента	Наш долг
Север		58 500
Юг	5 417,00	
Итого	5 417,00	58 500

Можно поле «Контрагент» сделать подчиненным полю «Регион»:

При этом поле «Контрагент.Регион» можно убрать из отчета.



Результат:



Зачем закладка «Данные» в Последовательностях:

Чтобы соблюдать последовательность в разрезе измерений. Но их надо создавать только для небольшого количества элементов. Например, есть пять собственных юридических лиц и у каждого накапливается своя себестоимость.

Занятие 1.17

Создадим Перечисление «Роли исполнителей»:

Свойство	Значение
Имя	РолиИсполнителей
Синоним	Роли исполнителей
Закладка «Данные»	
Значения	
Менеджер	
Имя	Менеджер
Синоним	Менеджер
Руководитель	
Имя	Руководитель
Синоним	Руководитель
Бухгалтер	
Имя	Бухгалтер
Синоним	Бухгалтер

Создадим Регистр сведений «Адресация»:

Свойство	Значение
Имя	Адресация
Синоним	Адресация
Закладка «Подсистемы»	Предприятие
Закладка «Данные»	
Измерения	
Исполнитель	
Имя	Исполнитель
Синоним	Исполнитель
Тип	СправочникСсылка.Сотрудники
Ведущее	Истина
Основной отбор	Истина

Свойство	Значение
Закладка «Данные»	
Измерения	
Исполнитель	
Запрет незаполненных значений	Истина
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Ведущее	Истина
Основной отбор	Истина
Роль	
Имя	Роль
Синоним	Роль
Тип	ПеречислениеСсылка.РолиИсполнителей
Ведущее	Истина
Основной отбор	Истина

У измерения «Контрагент» свойство «Запрет незаполненных значений - Ложь», т.к. надо, чтобы можно было назначить задачу всем руководителям и не важно, по каким контрагентам. В созданном Регистре сведений будут храниться правила адресации.

Объект «Задачи» - содержит список задач.

Создадим Задачу «Задачи исполнителей»:

Свойство	Значение
Имя	ЗадачиИсполнителей
Синоним	Задачи исполнителей
Закладка «Подсистемы»	Предприятие

Закладка «Адресация»	
Адресация	Адресация
Основной реквизит адресации	Исполнитель
Текущий исполнитель	ТекущийПользователь
Реквизиты адресации	
Исполнитель	
Имя	Исполнитель
Синоним	Исполнитель
Тип	СправочникСсылка.Сотрудники
Измерение адресации	Исполнитель
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Измерение адресации	Контрагент
Роль	
Имя	Роль
Синоним	Роль
Тип	ПеречислениеСсылка.РолиИсполнителей
Измерение адресации	Роль

Таблица задач (состав):

Наименование	Исполнитель	Роль	Контрагент

Плюс дополнительные реквизиты, создаваемые самостоятельно.

Пример:

Если есть Задача и Исполнитель, то системе все равно, что записано в «Роль» и «Контрагент»:

Наименование	Исполнитель	Роль	Контрагент
Выписать счет	Иванов		

Если в Задаче указана только «Роль»:

Наименование	Исполнитель	Роль	Контрагент
Выписать счет		Менеджер	

Система обратится к Регистру сведений «Адресация», построит отбор по Роли «Менеджер» - найти всех исполнителей, имеющих право выполнить задачу и показать им ее.

На закладке «Адресация» надо указать связь создаваемой таблицы задач с другими объектами, в частности с Регистром сведений «Адресация».

Адресация – указывается место, где хранятся правила адресации задач.

Реквизиты адресации – как правило, аналогичны Измерениям Регистра сведений.

В Правилах адресации (регистр сведений «Адресация») обязательно должен быть указан «Исполнитель» (Запрет незаполненных значений - Истина), а в Задаче исполнителя можно не указывать (если он не указан, то система будет искать его в правилах адресации).

Мы указали, что правила адресации хранятся в регистре сведений «Адресация» - теперь мы можем каждый Реквизит адресации связать с соответствующим полем в Регистре сведений (свойство «Измерение адресации»). Т.е. мы указываем, где в Регистре сведений система должна наложить отбор, чтобы найти конкретного исполнителя.

Основной реквизит адресации – один из Реквизитов адресации, который в итоге должен быть найден, в нем должен храниться текущий исполнитель. Отсюда система берет Исполнителя Задачи. Если реквизит не заполнен, то система ищет конкретного исполнителя в Регистре сведений «Адресация».

Текущий исполнитель – хранит информацию о сотрудниках, авторизовавшихся (зашедших) в БД. Поле ссылается на «Параметр сеанса».

Параметр сеанса – это что-то типа глобальной публичной переменной, которая доступна в любом режиме исполнения. Это переменная, которой при старте системы присваивается значение.

«Конфигурация – Общие – Параметры системы»

Создадим Параметр сеанса «Текущий пользователь»:

Свойство	Значение
Имя	ТекущийПользователь
Синоним	Текущий пользователь
Тип	СправочникСсылка.Сотрудники

Укажем созданный Параметр сеанса в свойстве «Текущий исполнитель» Задачи «Задачи исполнителей»

Теперь надо определить, в какой момент Параметр сеанса «Текущий пользователь» примет значение (кто работает с БД).

По идее данное значение определяется при запуске системы. В примере с запуском системы мы работали в «Модуле управляемого приложения». Но «Модуль управляемого приложения» активен только когда пользователь входит в систему интерактивно в режиме управляемого приложения. Если пользователь входит в режиме обычного приложения (для совместимости с 8.1) или запускает базу программно (например, как COM-сервер), то «Модуль управляемого приложения» не запускается.

В этом случае используется «Модуль сеанса» - неважно как запускается БД, данный модуль будет отработан в любом случае. В данном модуле обрабатывается только одна процедура:

«УстановкаПараметровСеанса(ТребуемыеПараметры)»

В дальнейшем эта процедура будет вызываться только при обращении к Параметру сеанса, значение которого не инициализировано. Например, при старте системы можно не инициализировать сразу все Параметры сеанса. Во время работы можно затребовать необходимые Параметры сеанса, при этом будет запускаться данная процедура. Через массив Требуемых параметров производится поиск

нужного параметра и ему присваивается значение. В следующий раз, при обращении к Параметру сеанса, процедура запускаться не будет, т.к. значение Параметра сеанса уже установлено.

При первом запуске в Требуемые параметры передается «Неопределено», что означает, что процедура запускается впервые при запуске программы.

ИмяПользователя() – функция, возвращающая имя пользователя (как оно задано в «Администрирование - Пользователи»), авторизовавшегося в БД.

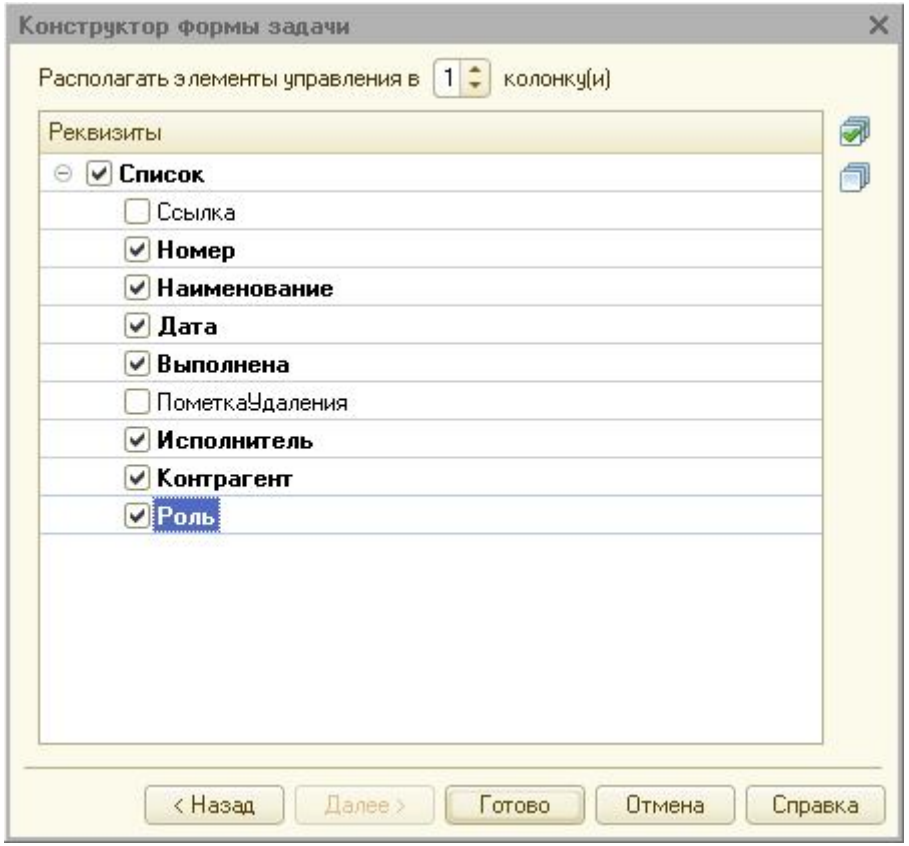
В зависимости от этого имени попробуем найти пользователя в БД, если такового нет, то создадим его.

```
Процедура УстановкаПараметровСеанса(ТребуемыеПараметры)
     Имя = ?(ПустаяСтрока(ИмяПользователя()), "НеАвторизован", ИмяПользователя());
     НайденныйЭлемент = Справочники.Сотрудники.НайтиПоНаименованию(Имя, Истина);
     Если НайденныйЭлемент.Пустая() Тогда
          НовыйЭлемент = Справочники.Сотрудники.СоздатьЭлемент();
          НовыйЭлемент.Наименование = Имя;
          НовыйЭлемент.Записать();
          НайденныйЭлемент = НовыйЭлемент.Ссылка;
     КонецЕсли;

     ПараметрыСеанса.ТекущийПользователь = НайденныйЭлемент;
КонецПроцедуры
```

«НайтиПоНаименованию(Имя, Истина)» - «Истина» означает, что ищем точное соответствие. Если «Ложь» или совсем не ставить этот параметр, то, например, если ищем по «Ива», то найдется и «Иванов» и «Иванюхин».

Теперь мы знаем, кто зашел в БД (в Параметре сеанса хранится ссылка на элемент справочника). В Задаче в Реквизитах адресации задано, где искать текущего исполнителя. Если в Реквизитах адресации он не найден, то указано где искать правила, по которым его можно найти (свойство «Адресация»), и с каким значением его надо сравнивать, когда найдем (свойство «Текущий исполнитель»).

Сделаем форму списка задач, чтобы показывать пользователю список его невыполненных задач. Реквизиты формы:



И в свойствах списка:

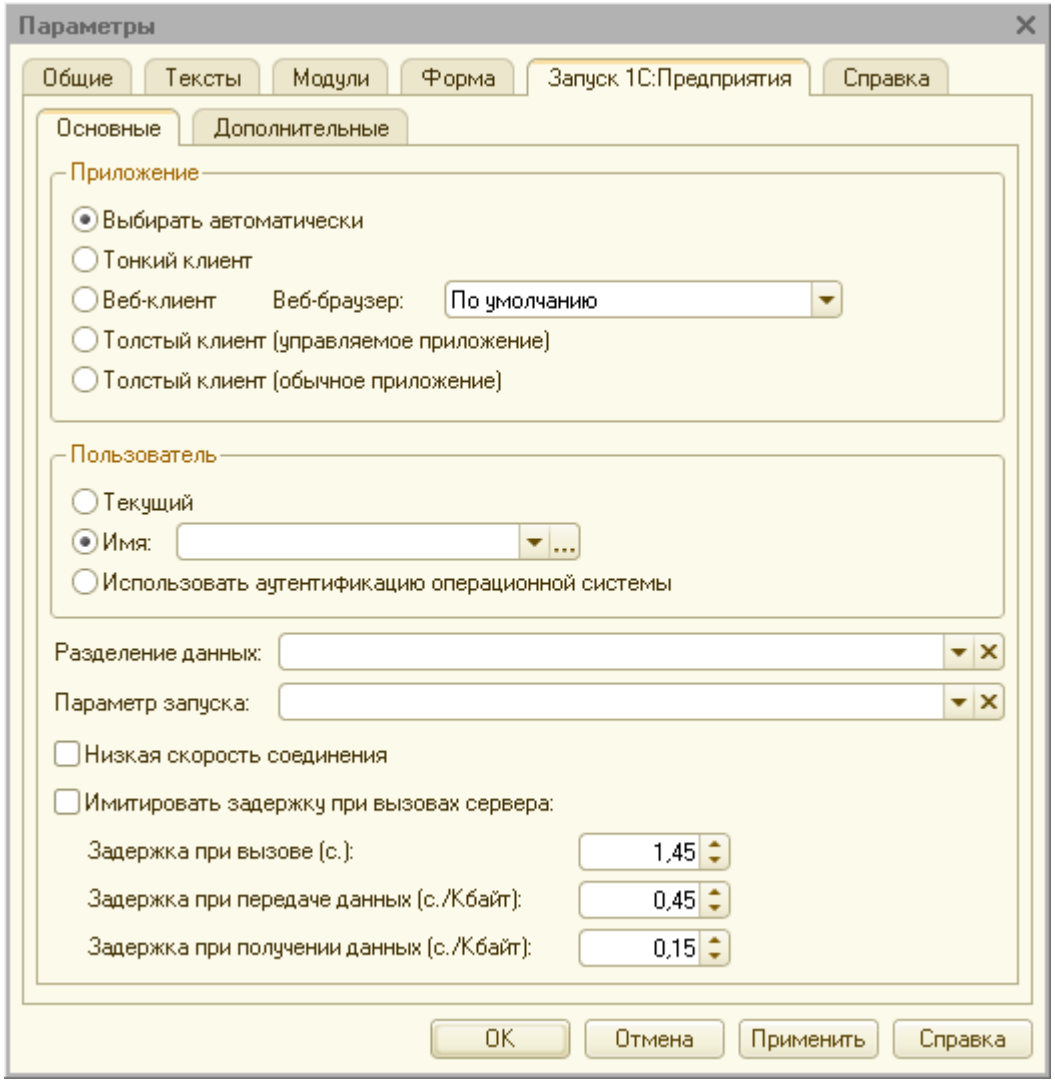
Свойство	Значение
ОсновнаяТаблица	Задача.ЗадачиИсполнителей.ЗадачиПоИсполнителю

Это виртуальная таблица, автоматом накладывающая отбор по правилам адресации (показывает задачи только конкретного пользователя).

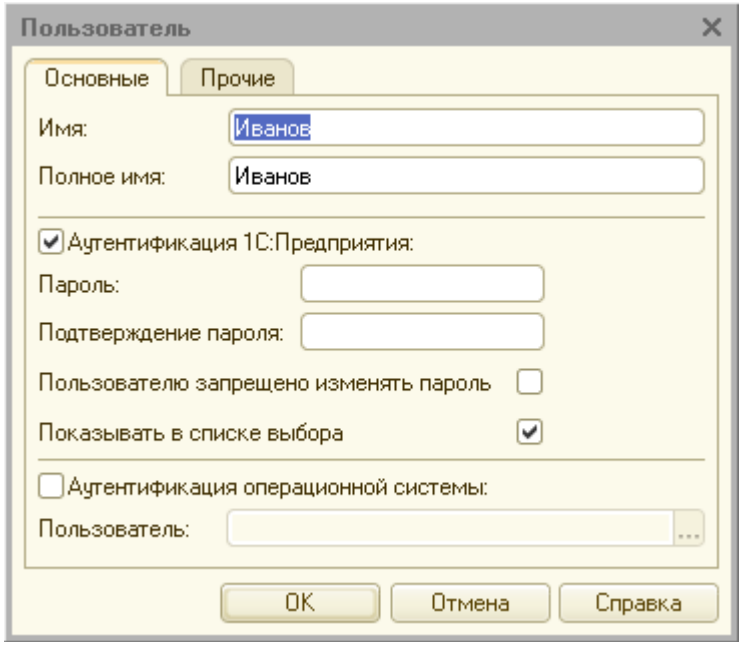
В режиме исполнения:

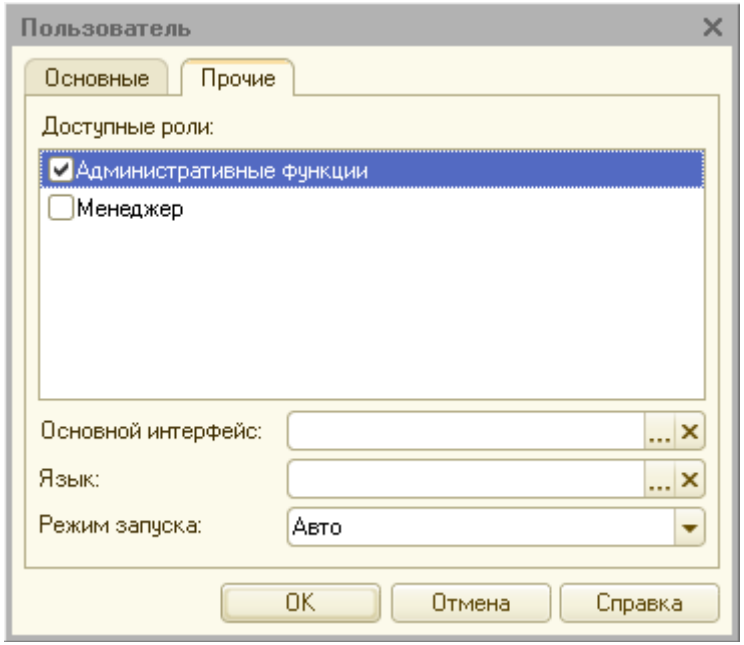
Сделаем, чтобы при запуске режима исполнения из конфигуратора, появлялось окно выбора пользователя:

«Сервис – Параметры – Запуск 1С:Предприятия – Пользователь - Имя»

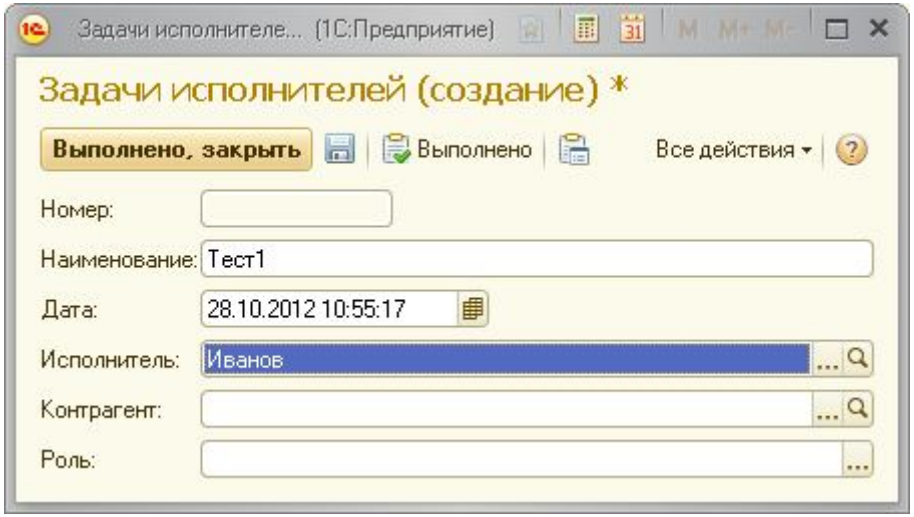


И создадим нового пользователя с правами администратора





Войдем в программу под Ивановым и создадим Задачу:

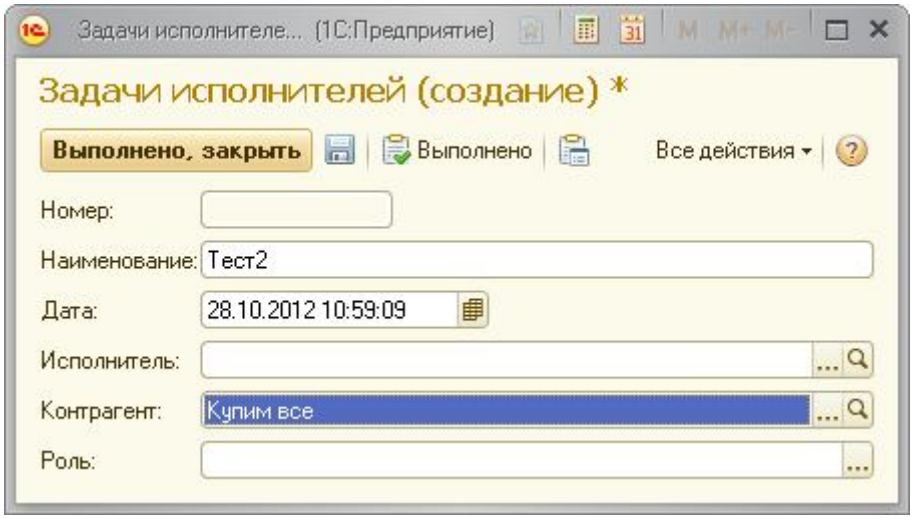


И запишем ее:

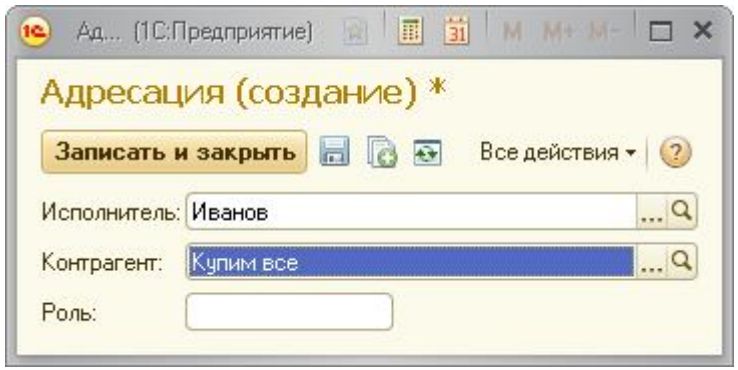
Номер	Наименование	Дата	Выполнена	Исполнитель	Контрагент	Роль
000000001	Тест1	28.10.2012 10:55:17		Иванов		

Пользователю будут показываться как не выполненные, так и выполненные Задачи.

Создадим еще одну Задачу:



После записи, в списке задач Иванова она отображаться не будет, т.к. не указано, что она связана с Ивановым. Но если указать связь в Адресации:



То в списке задач Иванова она станет доступной:

Номер	Наименование	Дата	Выполнена	Исполнитель	Контрагент	Роль
000000001	Тест1	28.10.2012 10:55:17	✔	Иванов		
000000002	Тест2	28.10.2012 10:59:09			Купим все	

Бизнес-процессы

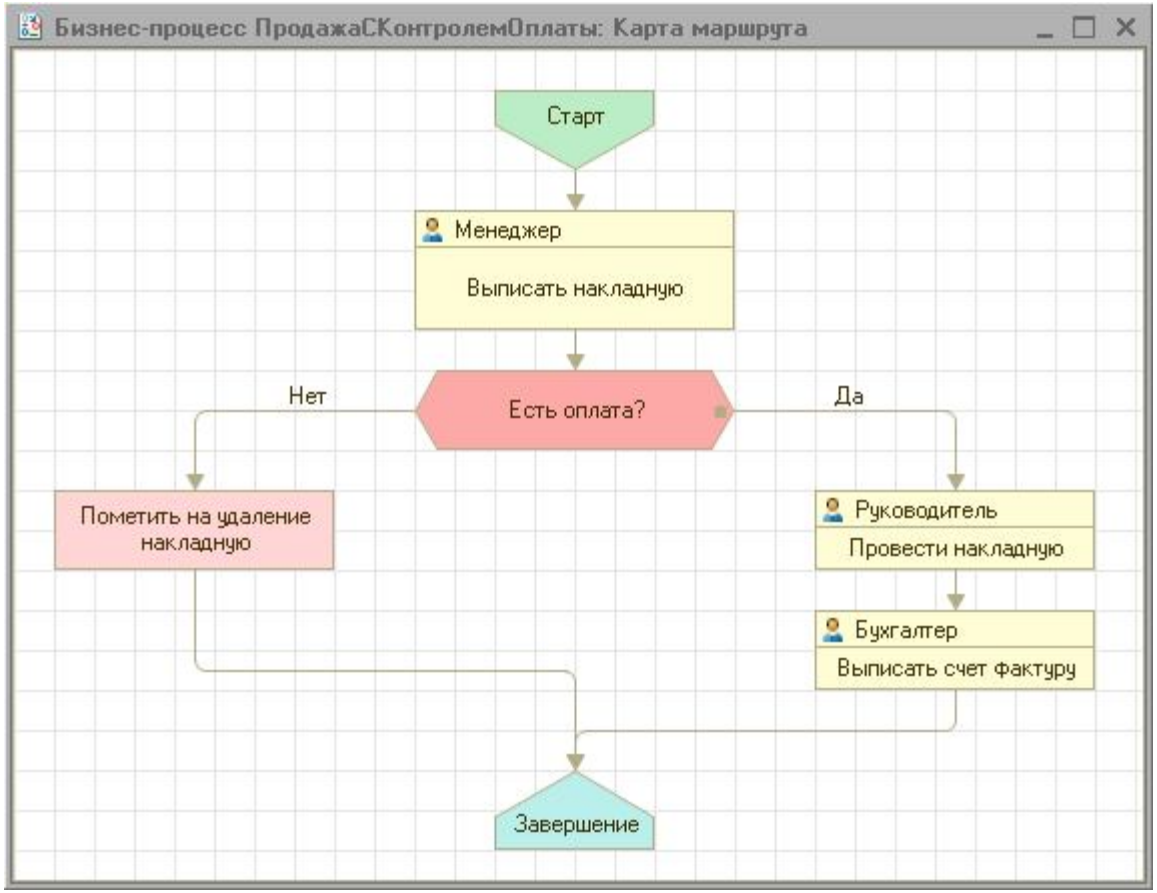
При создании бизнес-процесса задача создается автоматом.

Создадим бизнес-процесс «Продажа с контролем оплаты»

Свойство	Значение
Имя	ПродажаСКонтролемОплаты
Тип	Продажа с контролем оплаты
Задачи	ЗадачиИсполнителей
Закладка «Подсистемы»	Предприятие

Свойство	Значение
Закладка «Данные»	
Док продажи	
Имя	ДокПродажи
Синоним	Док продажи
Тип	ДокументСсылка.Расходная
Закладка «Прочее»	

Задачи – задается объект Задачи, в котором будут создаваться записи Задач бизнес-процесса.
Карта маршрута – описание логического поведения системы. Нарисуем карту маршрута:



Бизнес-процесс начинается с точки Старта. Имеет один обработчик события – «ПередСтартом». Влиять на Старт не рекомендуется, т.к. эта точка не интерактивная, т.е. пользователь не должен иметь возможность взаимодействовать с ней. Ее можно отметить программно – есть флаг «Отказ»:

```
Процедура СтартПередСтартом(ТочкаМаршрутаБизнесПроцесса, Отказ)
    // Вставить содержимое обработчика.
КонецПроцедуры
```

Точка Действия – по ним система формирует задачи.

Создадим точку Действия «Выписать накладную»:
Исполнитель задачи указывается на закладке «Адресация» - либо конкретный Исполнитель, либо Контрагент, либо Роль, либо сочетание. Значения признаков адресации можно задавать программно, например, в событии «ПриСозданииЗадач»

Свойство	Значение
Имя	ВыписатьНакладную
Заголовок	Выписать накладную
Роль	Менеджер

После выписки Накладной, будем ждать оплату. Создадим точку Условия «Есть оплата?»

Свойство	Значение
Имя	ЕстьОплата
Заголовок	Есть оплата?

Здесь есть событие «ПроверкаУсловия», которое в результат должно вернуть «Истина» или «Ложь».

```
Процедура ЕстьОплатаПроверкаУсловия(ТочкаМаршрутаБизнесПроцесса, Результат)
    // Вставить содержимое обработчика.
КонецПроцедуры
```

Если оплаты нет, то пометим Накладную на удаление. Создадим точку Обработки «Пометить на удаление накладную». Точка Обработки не интерактивная, т.е. это как функция/процедура:

Свойство	Значение
Имя	ПометитьНаУдалениеНакладную
Заголовок	Пометить на удаление накладную

Событие точки:

```
Процедура ПометитьНаУдалениеНакладнуюОбработка(ТочкаМаршрутаБизнесПроцесса)
    // Вставить содержимое обработчика.
КонецПроцедуры
```

Если оплата есть, то проведем Накладную. Создадим точку Действия «Провести накладную»:

Свойство	Значение
Имя	ПровестиНакладную
Заголовок	Провести накладную
Роль	Руководитель

По Накладной выпишем Счет-фактуру. Создадим точку Действия «Выписать счет фактуру»:

Свойство	Значение
Имя	ВыписатьСчетФактуру
Заголовок	Выписать счет фактуру
Роль	Бухгалтер

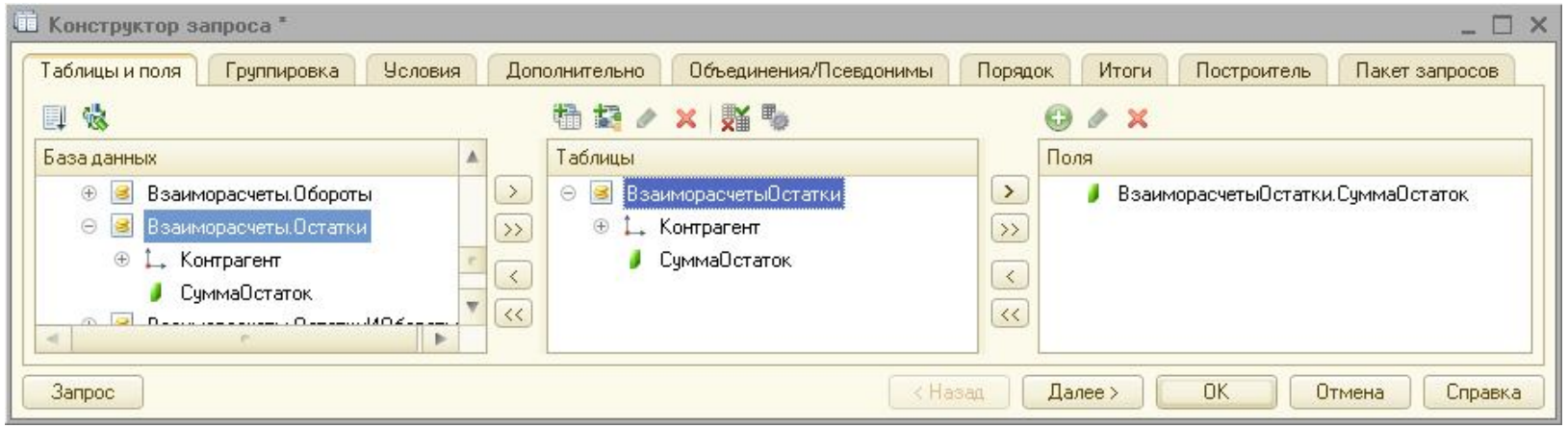
Теперь опишем поведение программы по точкам:

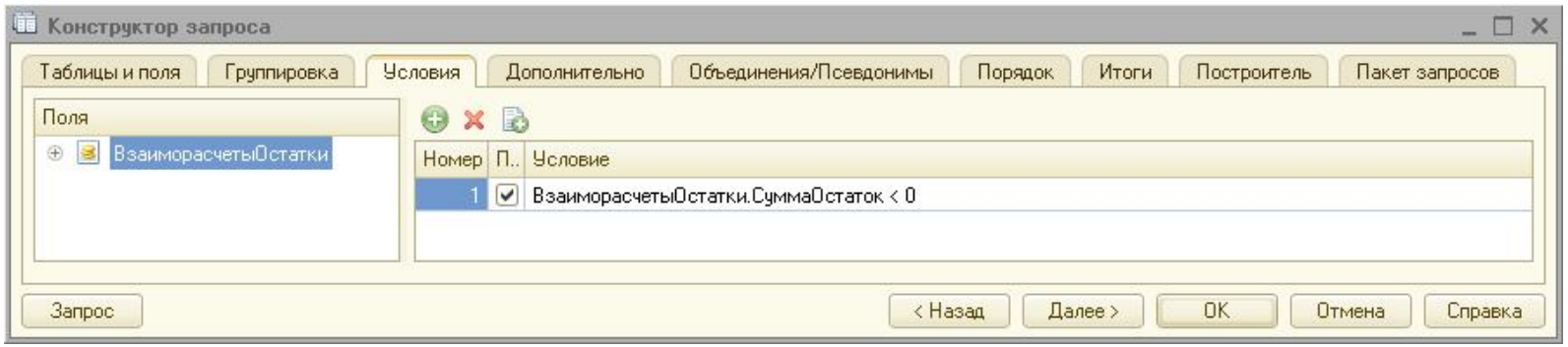
В точке «Выписать накладную» выполним проверку того, что документ-ссылка (из реквизита Задачи) указан (записан):

```
Процедура ВыписатьНакладнуюОбработкаПроверкиВыполнения(ТочкаМаршрутаБизнесПроцесса, Задача, Результат)
	Если ДокПродажи.Пустая() Тогда
		Результат = Ложь;
	Иначе
		Результат = Истина;
	КонецЕсли;
КонецПроцедуры
```

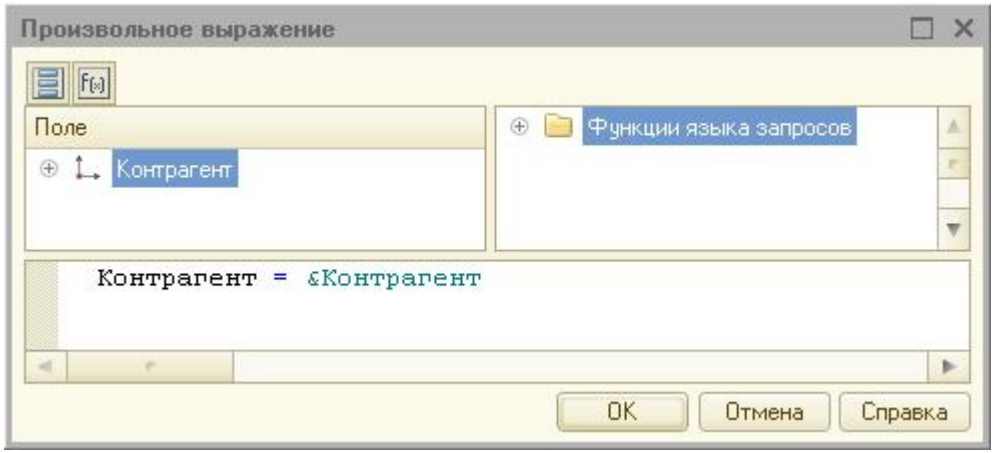
В точке «Есть оплата?» опишем событие:

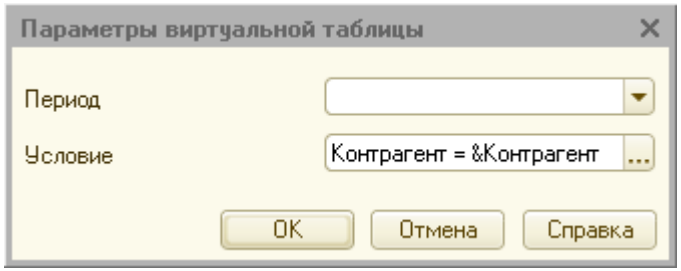
Запрос:





Установим параметр виртуальной таблицы:





Итого код:

```
Процедура ЕстьОплатаПроверкаУсловия(ТочкаМаршрутаБизнесПроцесса, Результат)
	Запрос = Новый Запрос;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	ВзаиморасчетыОстатки.СуммаОстаток
	|ИЗ
	|	РегистрНакопления.Взаиморасчеты.Остатки(, Контрагент = &Контрагент) КАК ВзаиморасчетыОстатки
	|ГДЕ
	|	ВзаиморасчетыОстатки.СуммаОстаток < 0";
	Запрос.УстановитьПараметр("Контрагент", ДокПродажи.Контрагент);
	РезультатЗапроса = Запрос.Выполнить();

	Если РезультатЗапроса.Пустой() Тогда
		Результат = Ложь;
	Иначе
		Результат = Истина;
	КонецЕсли;
КонецПроцедуры
```

В регистре накопления «Взаиморасчеты» учитываем долг контрагента, т.е. если долг есть, то сумма будет «+» и запрос ничего не вернет (действие не выполняем).

В точке «Пометить на удаление накладную» опишем событие:

Сделаем форму, где пользователь будет видеть выполнение шагов Карты маршрута бизнес-процесса:

Создадим обработку «Карта маршрута»:

Свойство	**Описание**
Имя	КартаМаршрута
Синоним	Карта маршрута
Закладка «Подсистемы»	Предприятие
Закладка «Данные»	
БП	
Имя	БП
Синоним	БП
Тип	БизнесПроцессСсылка.ПродажаСКонтролемОплаты

Форма:

Добавим реквизит «Карта»:

Свойство	**Описание**
Имя	Карта
Синоним	Карта
Тип	ГрафическаяСхема

И команду формы «Обновить»:

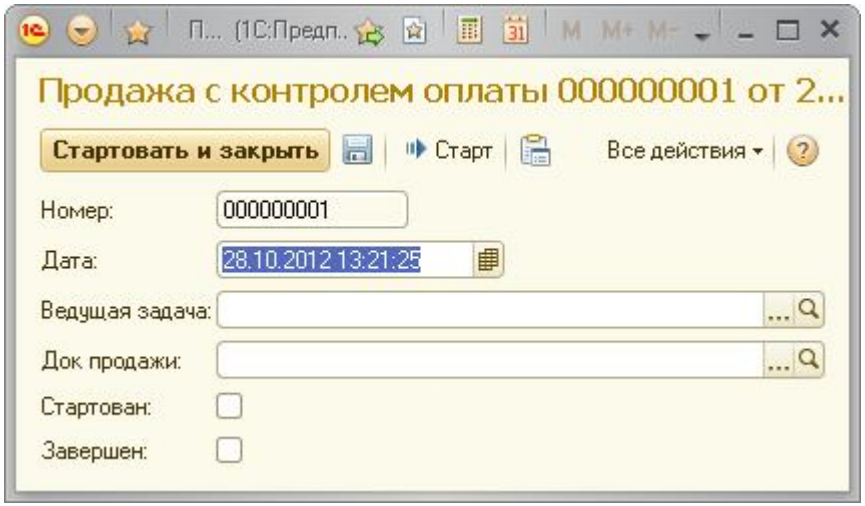
Свойство	Описание
Имя	Обновить
Заголовок	Обновить

Опишем свойство «Действие» команды формы «Обновить»:

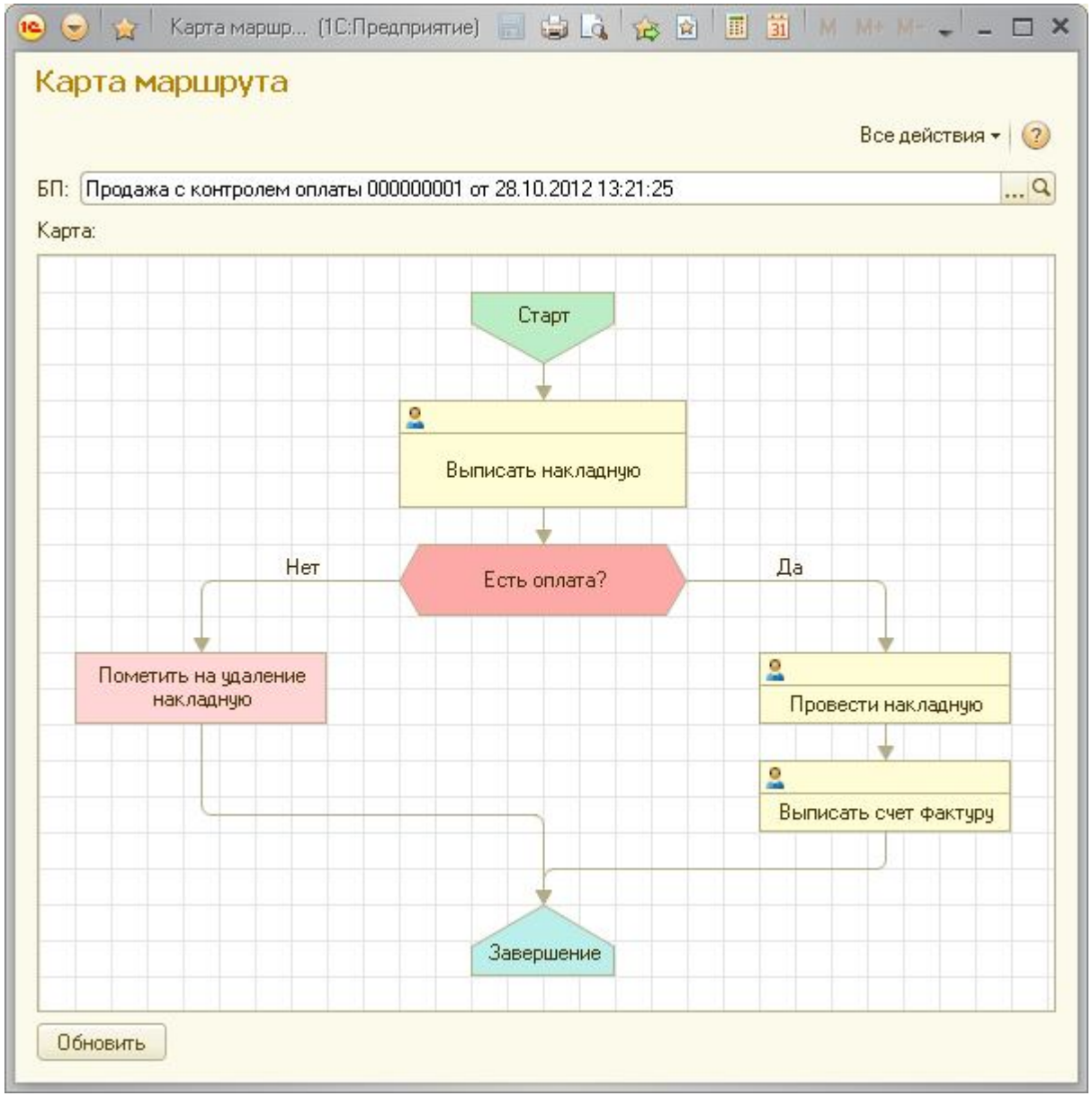
```
&НаКлиенте
Процедура Обновить(Команда)
	ОбновитьКарту();
КонецПроцедуры

&НаСервере
Процедура ОбновитьКарту()
	Карта = Объект.БП.ПолучитьОбъект().ПолучитьКартуМаршрута();
КонецПроцедуры
```

В режиме исполнения:

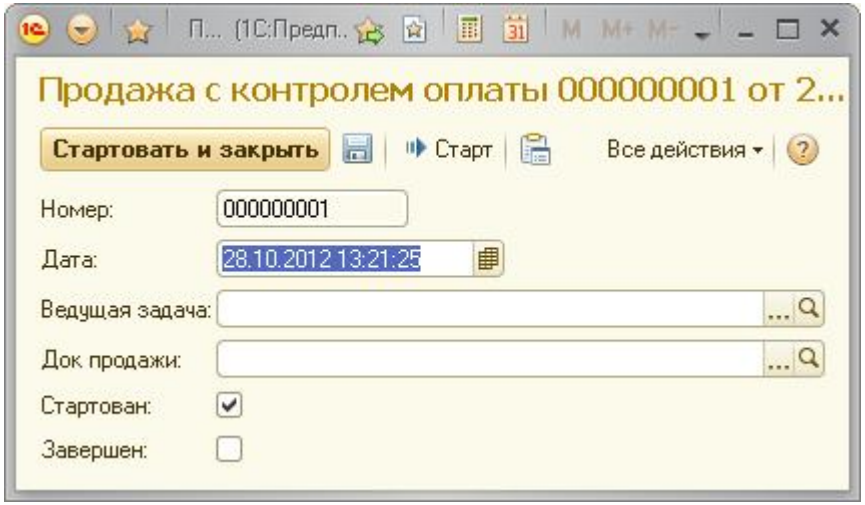
Создадим бизнес-процесс «Продажа с контролем оплаты» и запишем его:



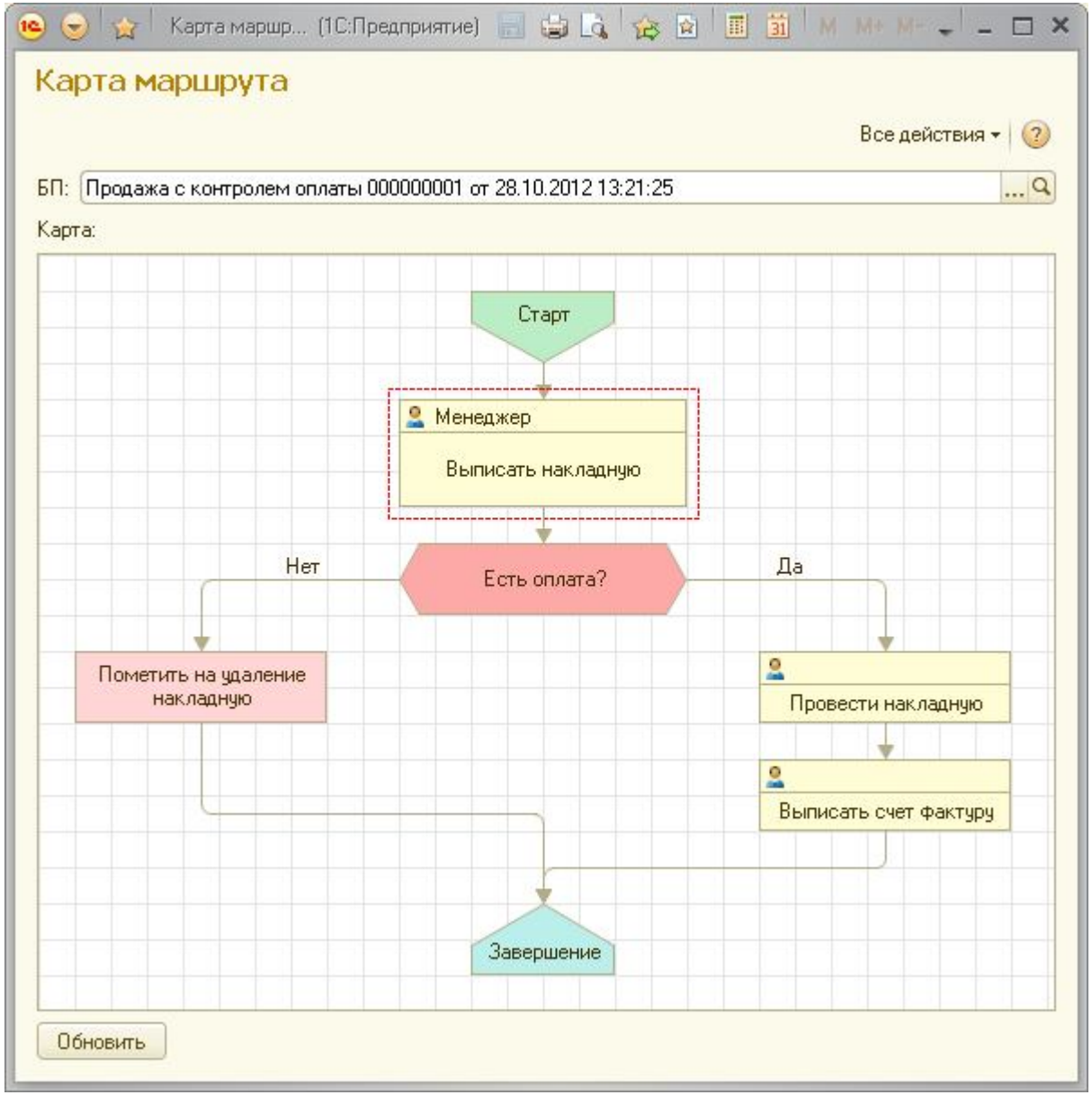
Т.е. создадим просто ссылку на бизнес-процесс, т.е. бизнес-процесс не стартован, ни одна из точек действия не активна. Откроем обработку «Карта маршрута» и выберем в ней наш бизнес-процесс:



Теперь стартанем бизнес-процесс:



И проверим обновление в карте:



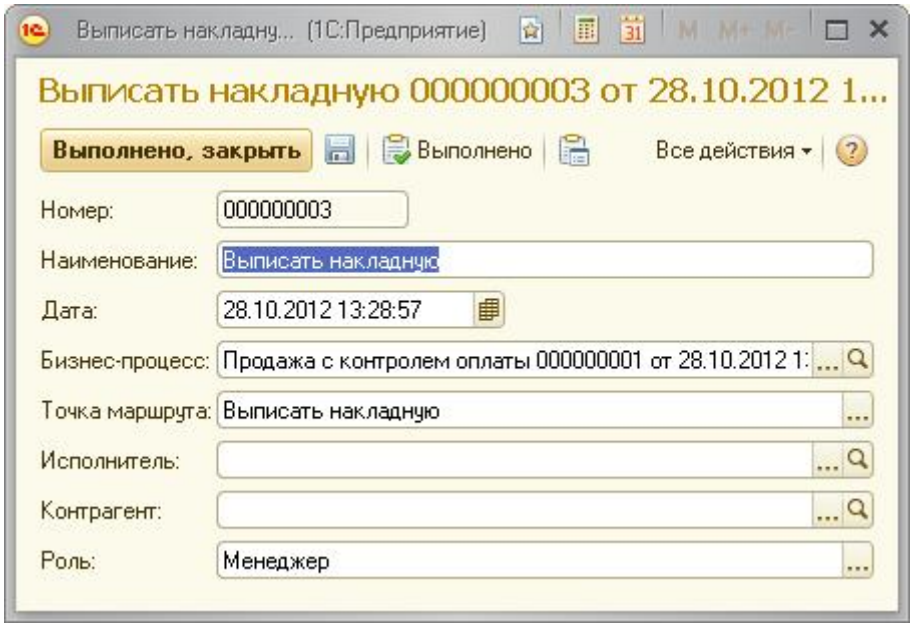
Точка «Выписать накладную» стала активной.

На время поменяем свойство «ОсновнаяТаблица» формы списка Задач, чтобы показывались все задачи, а не только активного пользователя:

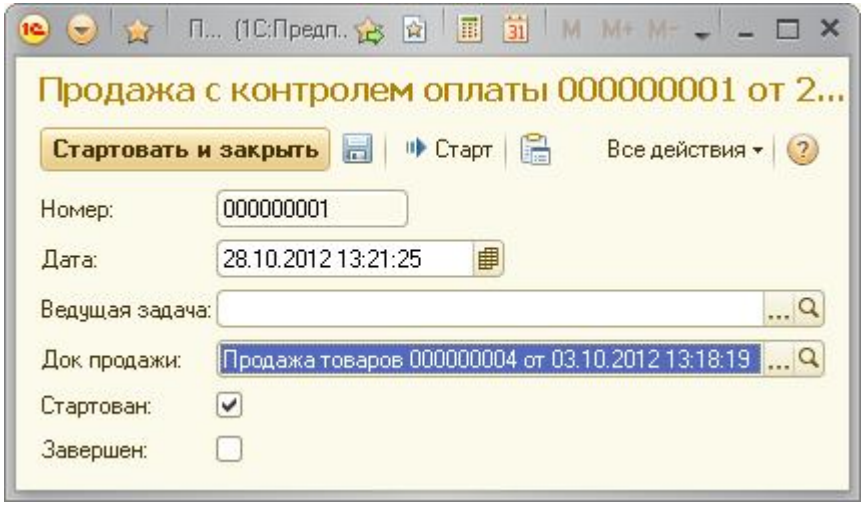
Свойство	Значение
ОсновнаяТаблица	Задача.ЗадачиИсполнителей

В режиме исполнения:

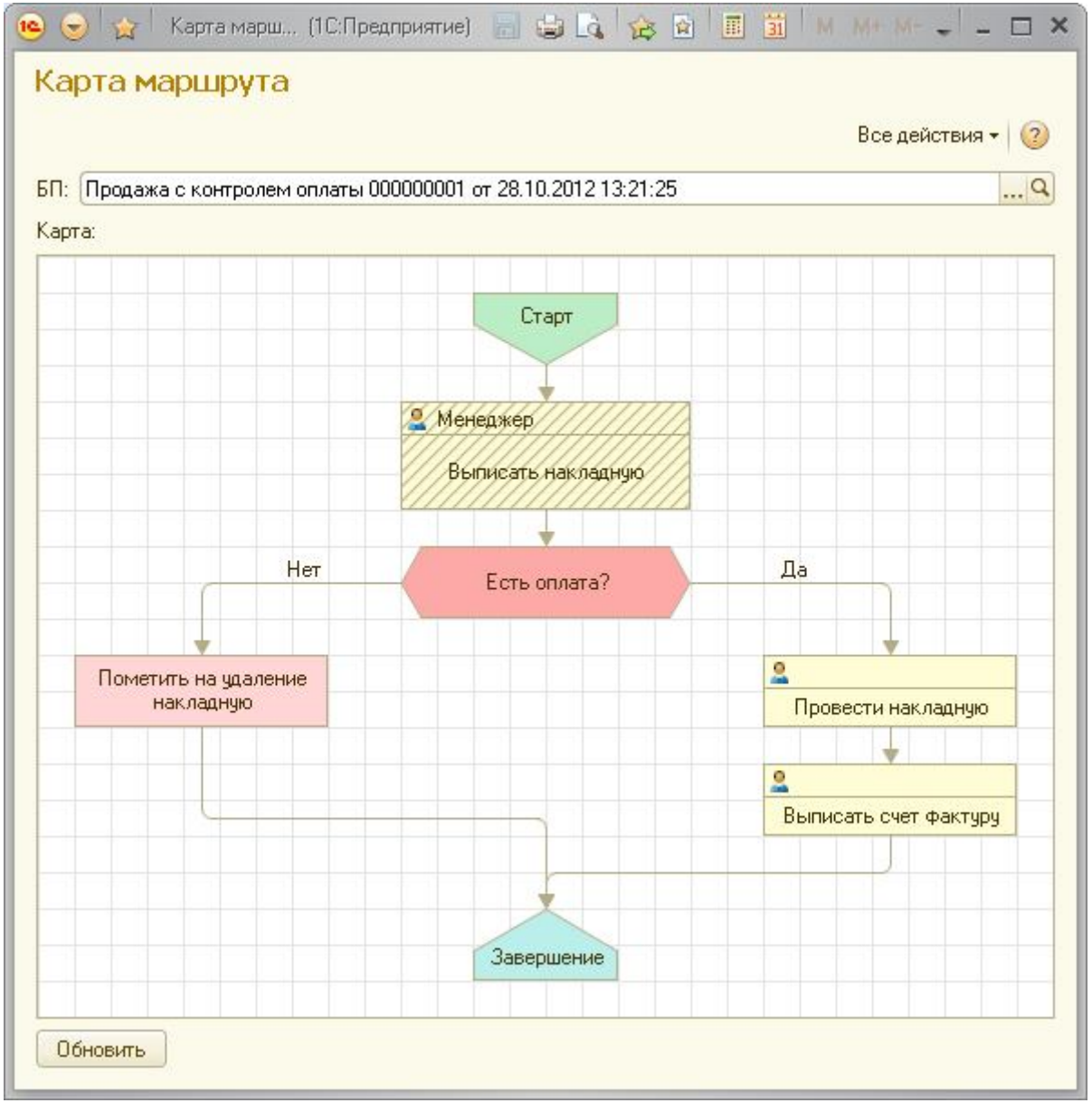
Смотрим задачу:



Мы не можем выполнить задачу, т.к. в бизнес-процессе не указан документ продажи. Укажем его:



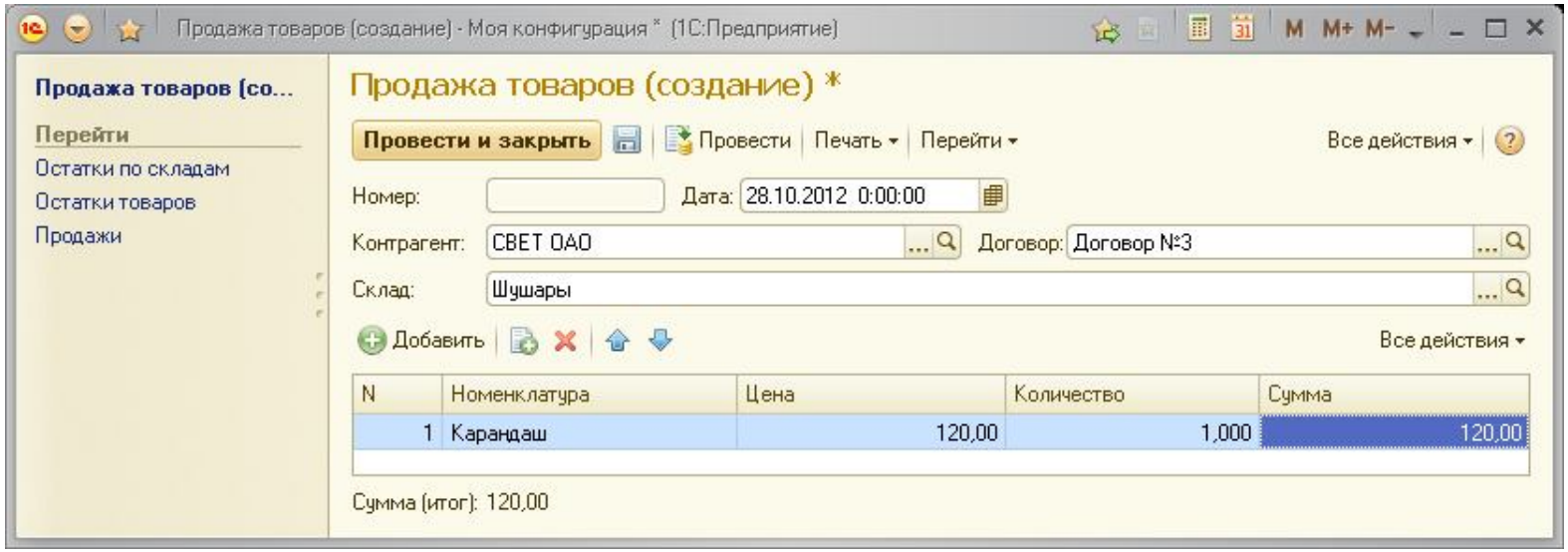
И выполним задачу и проверим карту маршрута:



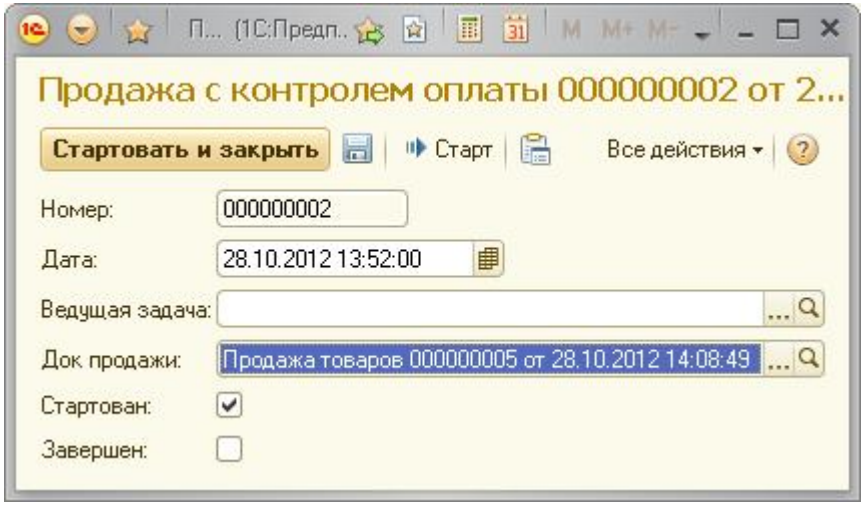
Точка «Выписать накладную» выполнилась. Оплата не была выполнена, поэтому пошли по ветке «Нет» и процесс завершился. При этом Расходная накладная должна была быть помечена на удаление. Проверим:

Дата	Номер	Контрагент
02.09.2012 1:43:12	000000001	Покупайло
02.10.2012 0:00:00	000000002	Покупайло
03.10.2012 13:17:21	000000003	Купим все
03.10.2012 13:18:19	000000004	Купим все

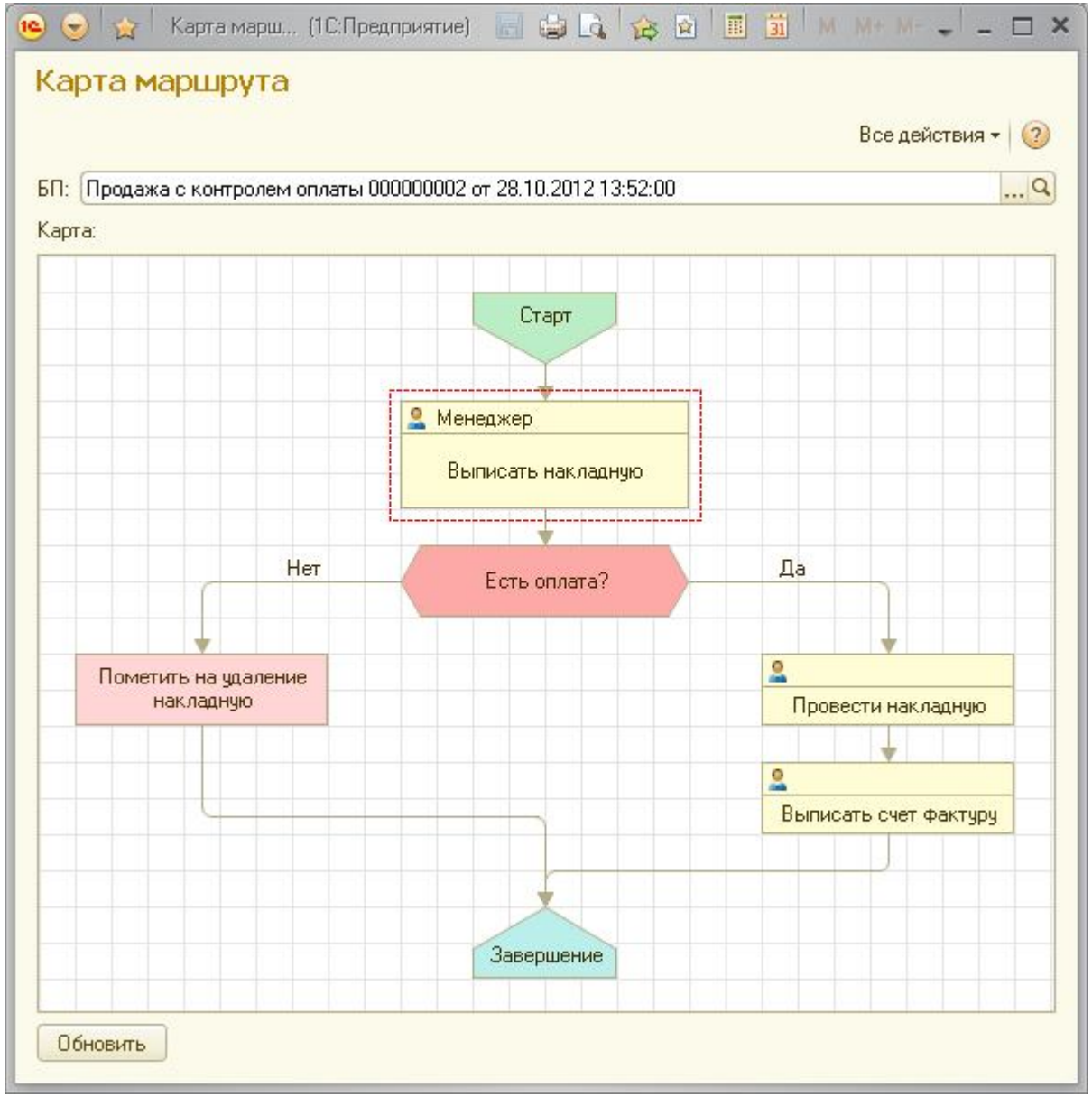
Сделаем пример, где оплата уже будет:
Создаем новый бизнес-процесс, запишем его. Создадим для него новый документ «Продажа товаров»:



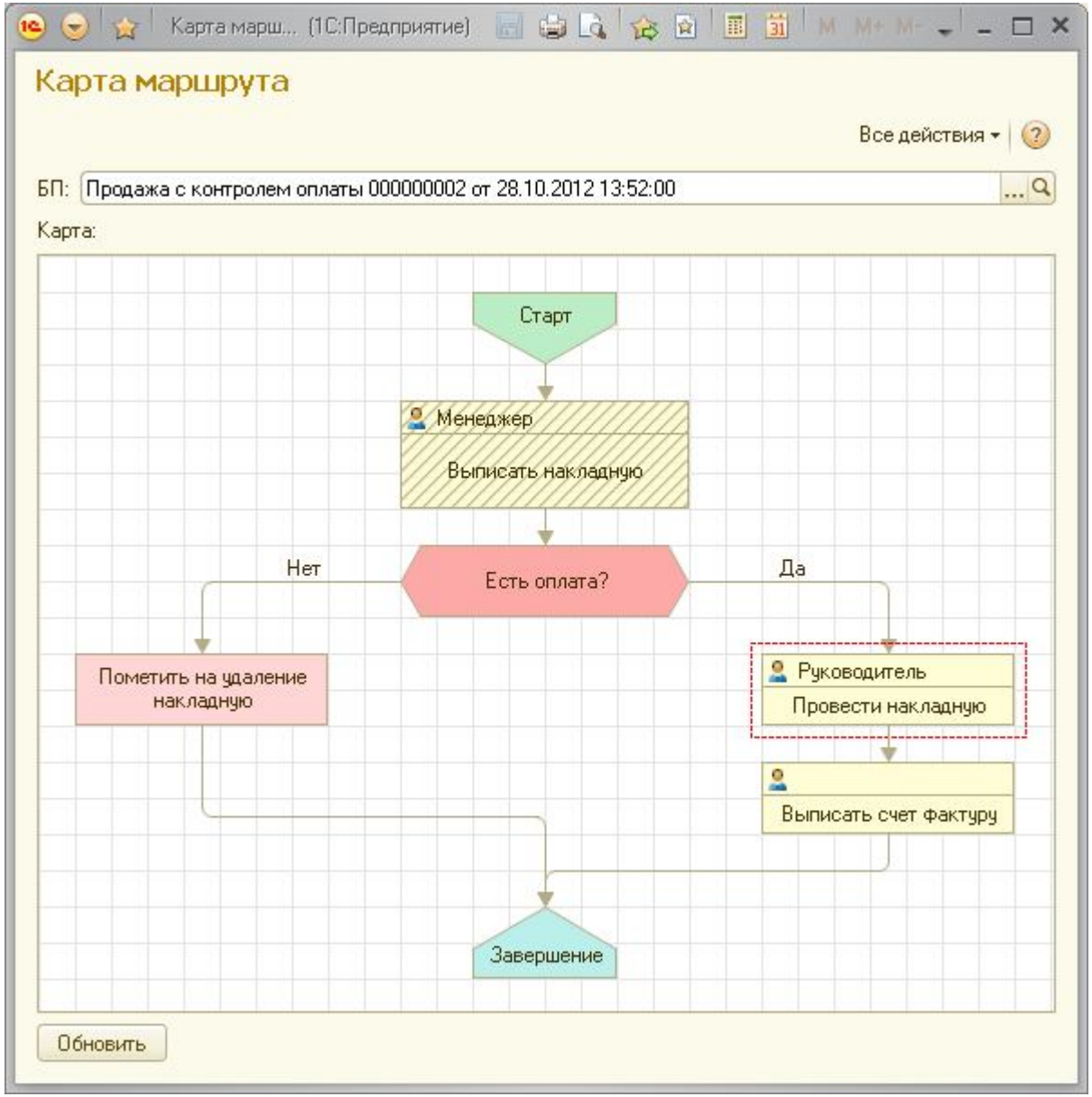
Запишем документ, выберем его в бизнес-процессе и стартуем бизнес-процесс:



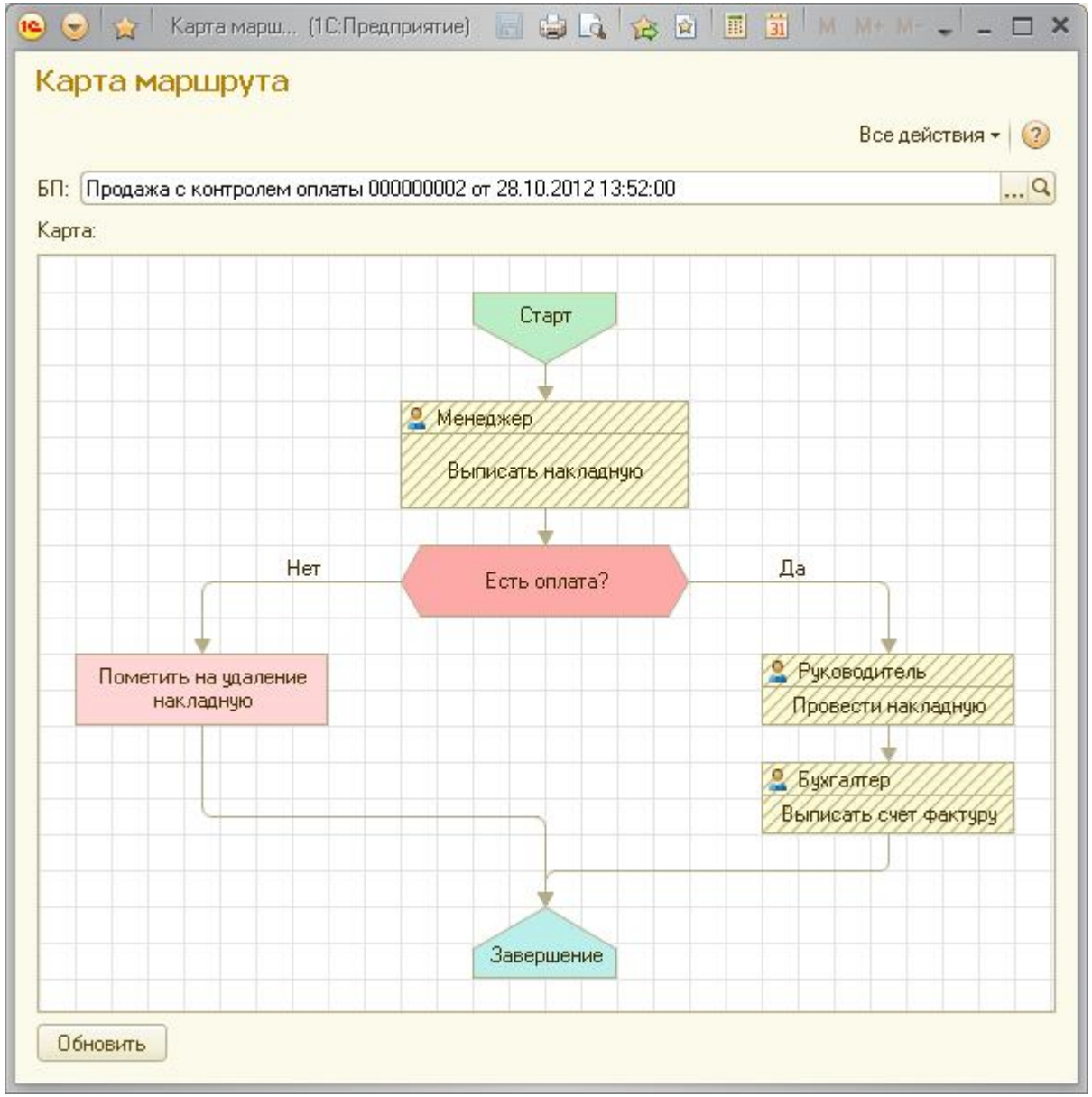
Проверим карту маршрута:



Надо выписать накладную. Открываем соответствующую задачу, выполняем ее и опять смотрим карту:



Теперь последовательно выполняем задачи «Провести накладную» и «Выписать счет-фактуру». Посмотрим карту маршрута:



Необходимые точки пройдены, бизнес-процесс завершен.

Занятие 1.18

Механизмы распределенных БД

В 8.2 практически не актуальны, т.к. идея 8.2 в клиент-серверной архитектуре, клиенты которой могут быть территориально удаленными.

«Конфигурация – Общие - Планы обмена» - для реализации механизмов распределенных БД. Две технологии работы:

1. Есть Центр и периферийные БД (технология «Звезда» - главный и удаленные узлы). Обмен данными происходит через Центр, при этом на периферийные БД могут быть наложены ограничения в получении/передаче данных.
2. Произвольный обмен данными. При этом обмен придется описывать самостоятельно. В первом варианте настройка обмена заложена в самой платформе. Настраивать придется только специфические тонкости.

Планы обмена позволяют накапливать информацию об измененных объектах, упорядочить процесс загрузки/выгрузки данных, контролировать процесс переноса данных

Теория:

Допустим, есть две БД (может быть и более), одна из которых – Центральная. В Центральной и Удаленной БД накапливается информация об объектах, которые входят в План обмена. По сути, План обмена – это таблица, фиксирующая несколько состояний записей:

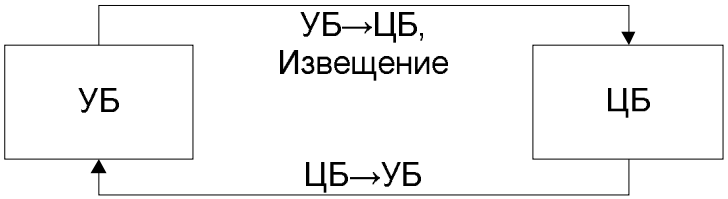
- Создан.
- Изменен.
- Удален.

Эту информацию накапливает каждый узел Плана обмена и при выгрузке/загрузке формируется XML-файл с этой информацией.

Допустим, ЦБ отправляет в УБ информацию об изменениях – формируется пакет данных №1. УБ загружает данные и при следующем соединении с ЦБ она формирует пакет, в котором хранятся:

- Данные о том, что надо загрузить из УБ в ЦБ.
- Извещение о том, что пакет данных №1 был принят.

Если Извещения нет, то ЦБ вместе с пакетом №2 выгрузит и передаст УБ данные пакета №1.



Ограничение технологии распределенных БД – конфигурации всех БД должны быть идентичны. Изменения допускаются только в ЦБ. Канал передачи данных пользователь выбирает самостоятельно (электронная почта, FTP-сервер и т.д.).

Пример:

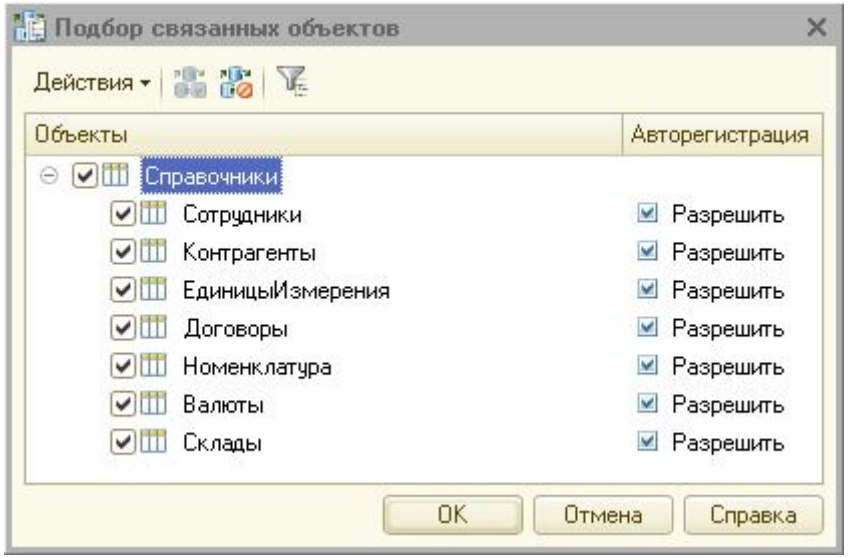
Создадим План обмена «Распределенка»

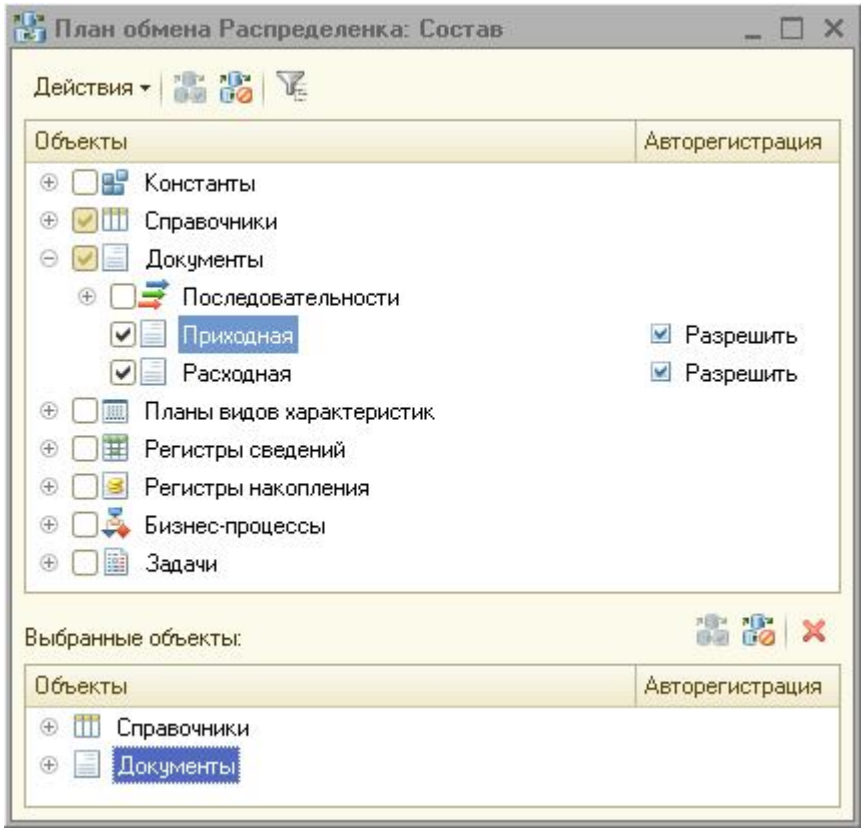
Свойство	**Значение**
Имя	Распределенка
Синоним	Распределенка
Распределенная база данных	Истина
Закладка «Подсистемы»	Предприятие

«Распределенная база данных – Истина» - включает механизм распределенных БД. Если «Ложь», то механизм придется описывать вручную.

Кнопка «Состав» - объекты передачи м/ду БД.
Например, надо чтобы передавались все документы: Приходная и Расходная. Если просто отметить их, то кроме документов ничего передаваться не будет. Но в документе есть реквизиты, которые тоже должны передаваться:

«Действия – Подобрать связанные для всех» - для передачи всех, связанных с отмеченными, данных.

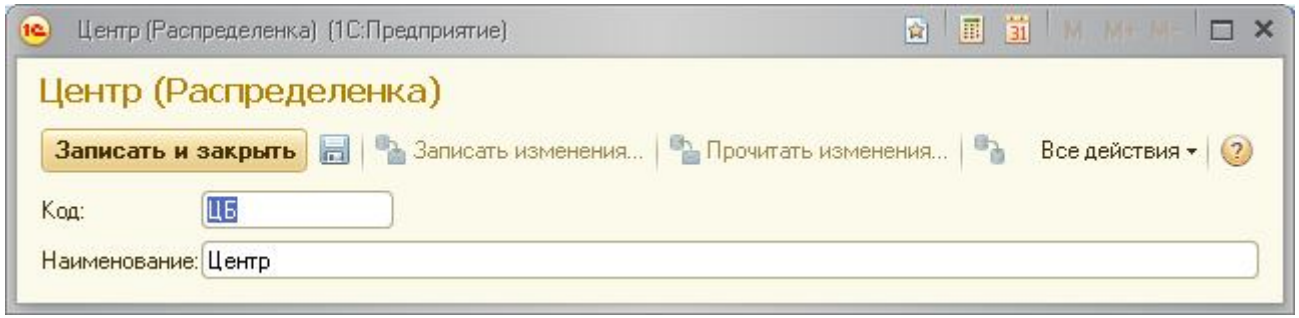




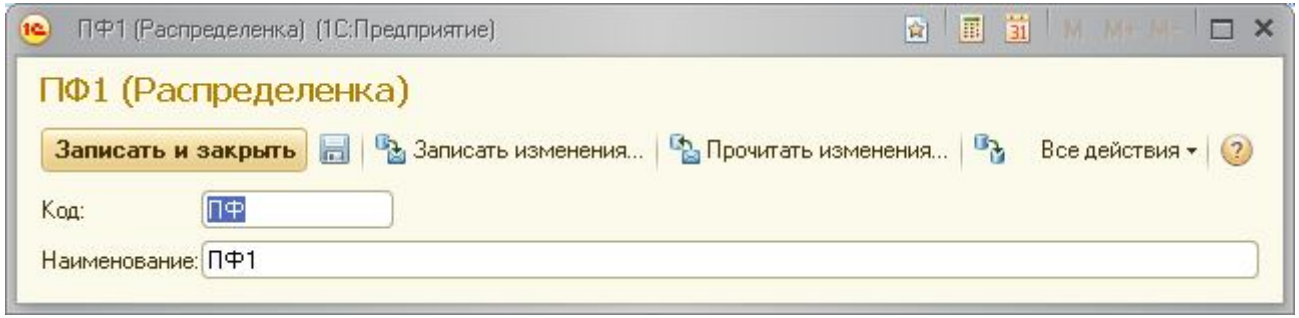
При этом автоматом не отмечаются для передачи движения по регистрам. Это объясняется тем, что движения по регистрам иногда строятся по данным учетной системы. В УБ передаваемые документы будут записаны, а могут быть еще и проведены, т.е. сформируются необходимые движения. Передача же существующих движений только увеличит объем передаваемых данных. В примере смысла в этом нет, т.к. данные по документам передаются полностью.

В режиме исполнения:

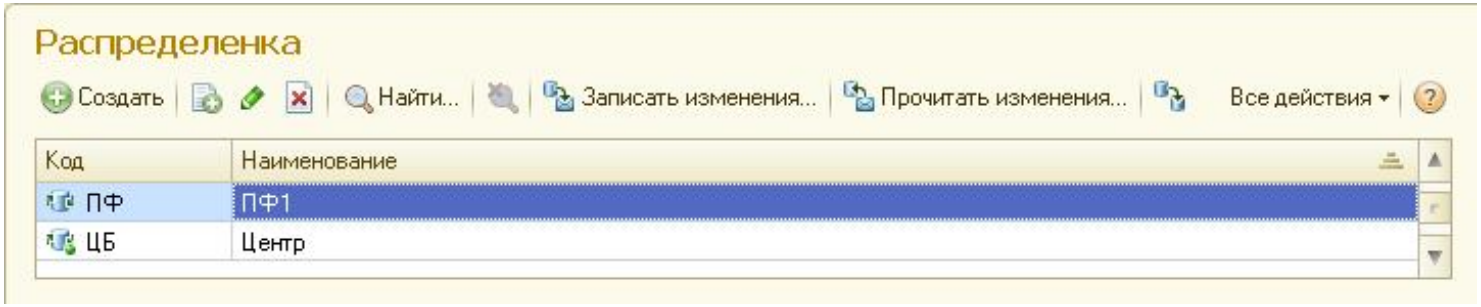
ЦБ уже есть, но поля в ней пустые. Заполним их:



Создадим УБ (периферийную БД):



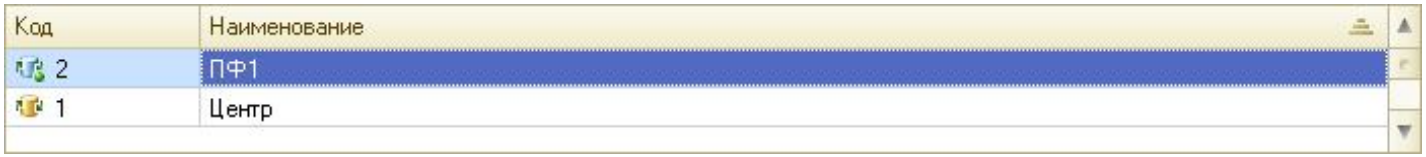
В итоге у нас есть БД:



Записать изменения – выгрузка изменений.
Прочитать изменения – загрузка изменений.
Создать начальный образ подчиненного узла РБД – начальная выгрузка БД. Создается файл формата «.CD», т.е. это файл БД (не сжатый, а уже полностью готовый к работе). Т.е. фактически это создание копии ЦБ. Перед созданием система спросит, в какое место сохранить файл.

Теперь, если мы, допустим, создадим еще одну Приходную, то система узнает, что для распределенной БД появился документ, который надо выгрузить. Далее в Плане обмена «Распределенка» открываем периферийную БД и нажимаем «Записать изменения», укажем место сохранения файла (это будет XML-файл, который будет находиться в ZIP-архиве, если «Сжимать сообщение - Истина»).

Загрузка данных производится в периферийной БД, где БД обозначены по-другому:



Встанем на БД «Центр», нажмем «Прочитать изменения» и выберем выгруженный из ЦБ файл.

Если номер принимаемого файла совпадает с номером уже принятого, то система выдаст ошибку и проигнорирует его.
При обмене возникают коллизии, которые приходится вычислять и исправлять вручную.

Конфигурации

Производить обновления и исправления БД, при этом каждый раз выгоняя пользователей, не совсем правильно.

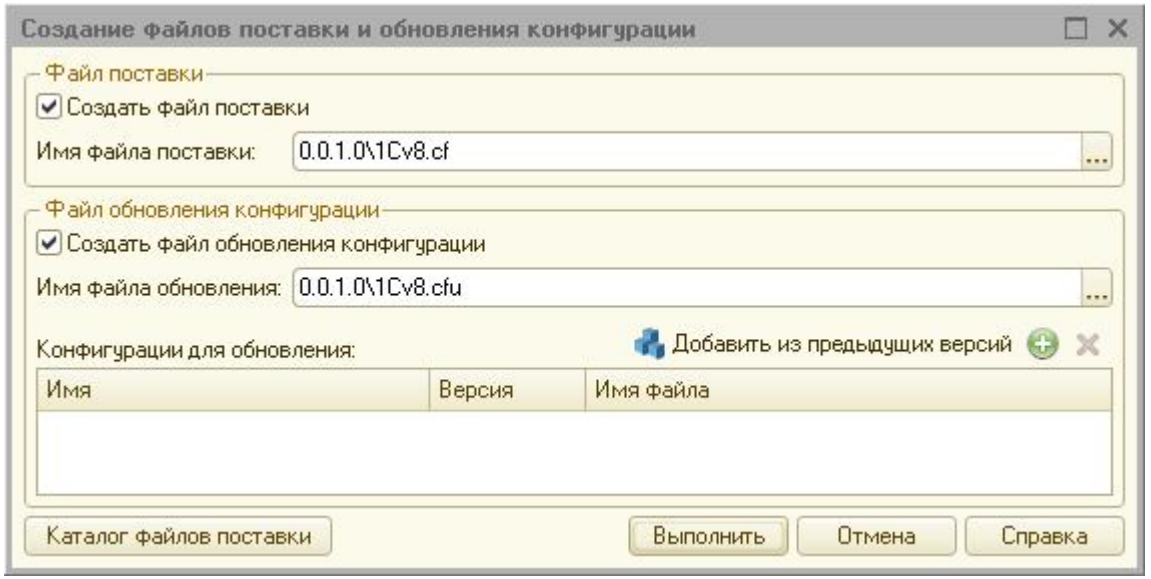
Если надо для клиента стать поставщиком конфигурации:

Надо обозначить свою конфигурацию – дать имя и указать информацию о поставщике. «Конфигурация – ПКМ - Свойства»:

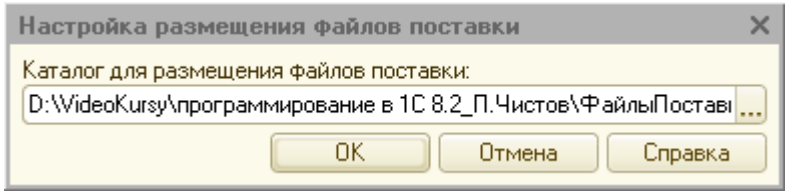
Свойство	Значение
Имя	ЧастьПервая
Синоним	Часть первая
Поставщик	edu1c.ru
Версия	0.0.1.0

Свойство «Адрес каталога обновлений» - это HTTP-каталог, или FTP-каталог, или локальный каталог и т.д., т.е. место, где платформа по умолчанию будет искать обновления конфигурации

«Конфигурация – Поставка конфигурации – Создать файлы поставки и обновления конфигурации» - можно создать файл поставки:
«.CF» - файл конфигурации без данных.
«.CFU» - файл результата сравнения новой конфигурации с предыдущими релизами.



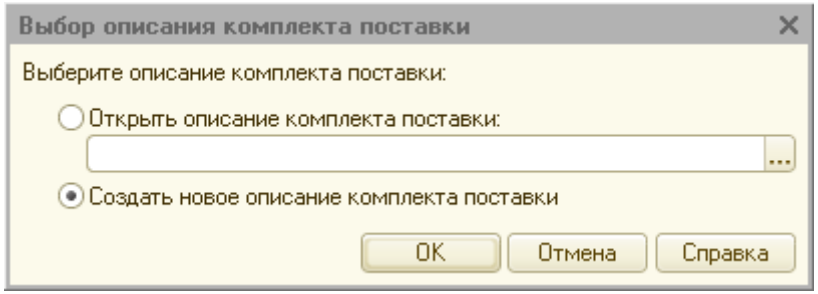
«Каталог файлов поставки» - задается путь к каталогу, в котором будут храниться файлы поставки.

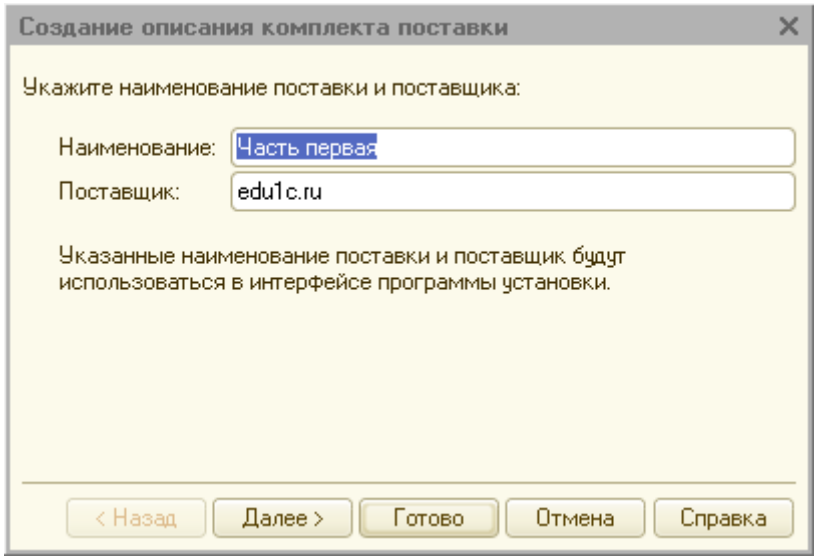


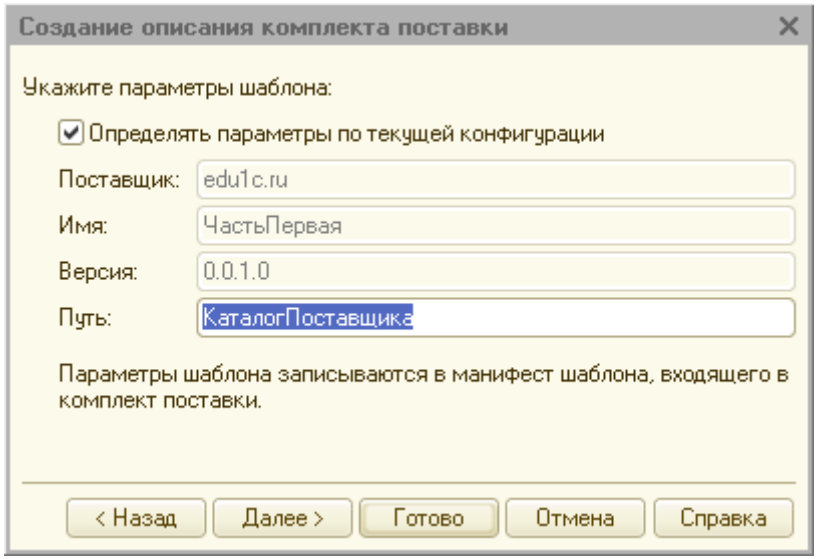
При нажатии на «Выполнить», будет выгружен CF-файл.
На основе выгрузки можно сформировать файл обновления, который не позволит пользователю менять конфигурацию самостоятельно, а позволит только обновлять ее.

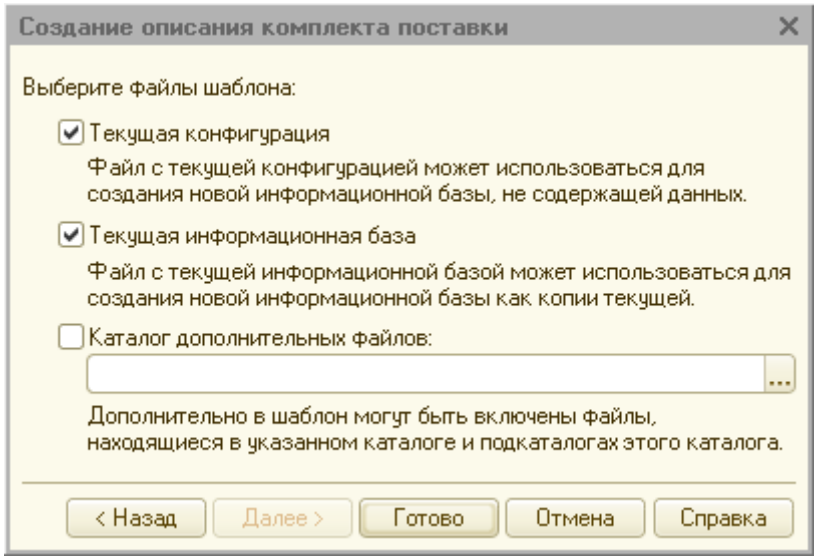
«Конфигурация – Поставка конфигурации – Комплект поставки» - создается для более удобного обновления конфигурации. Это такая же конфигурация, только упакованная в EXE-файл, при запуске которого запускается мастер шаблонов конфигураций, которые можно увидеть при запуске программы.

Создадим Комплект поставки:

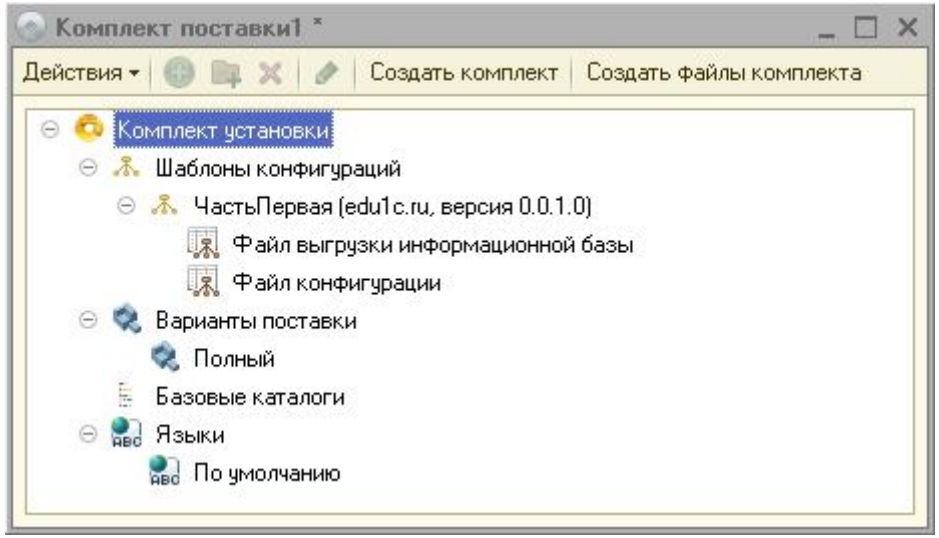



Создание описания комплекта поставки
Укажите наименование поставки и поставщика:
Наименование:
Часть первая
Поставщик:
edu1c.ru
Указанные наименование поставки и поставщик будут использоваться в интерфейсе программы установки.
< Назад
Далее >
Готово
Отмена
Справка


Создание описания комплекта поставки
Укажите параметры шаблона:
Определять параметры по текущей конфигурации
Поставщик:
edu1c.ru
Имя:
ЧастьПервая
Версия:
0.0.1.0
Путь:
КаталогПоставщика
Параметры шаблона записываются в манифест шаблона, входящего в комплект поставки.
< Назад
Далее >
Готово
Отмена
Справка



Итого получилось Описание шаблона:

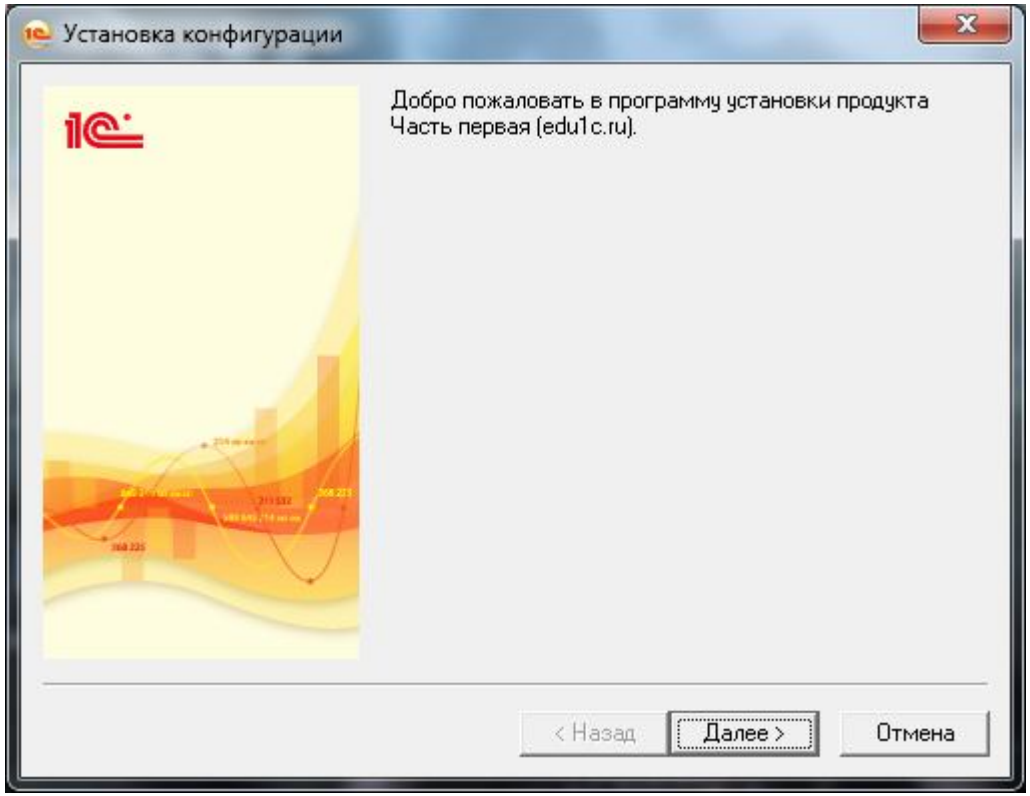


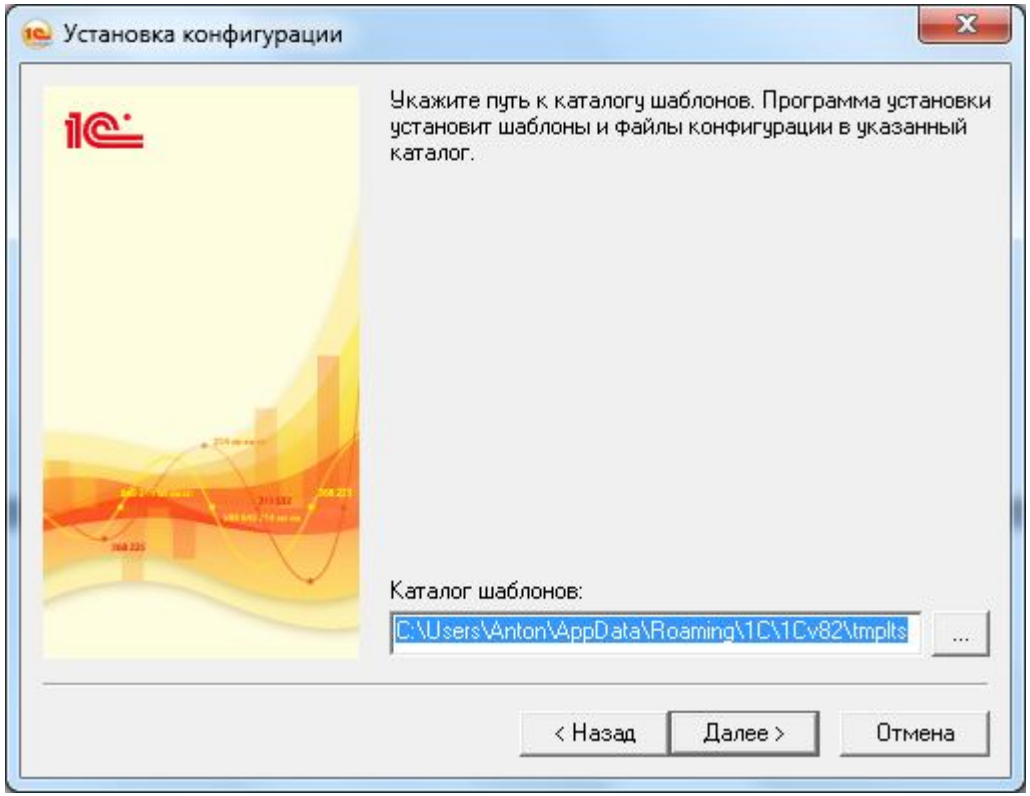
Нажмем кнопку «Создать комплект»:

Система предложить сохранить файл «Описания комплекта поставки» (EDF-файл) и сам Комплект, состоящий из двух файлов: EXE-файл – мастер установки.

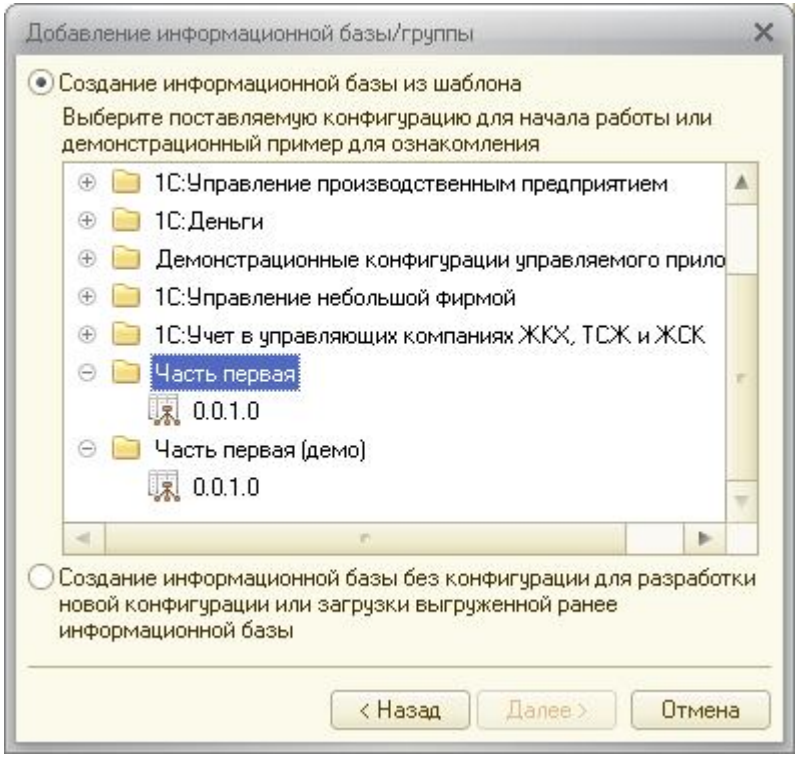
EFD-файл – в него упакована конфигурация, данные. Также в него можно упаковать другие объекты.

При запуске EXE-файла запускается мастер установки конфигурации:




Установка конфигурации
Укажите путь к каталогу шаблонов. Программа установки установит шаблоны и файлы конфигурации в указанный каталог.
Каталог шаблонов:
C:\Users\Anton\AppData\Roaming\1C\1Cv82\tmplts
...
< Назад
Далее >
Отмена

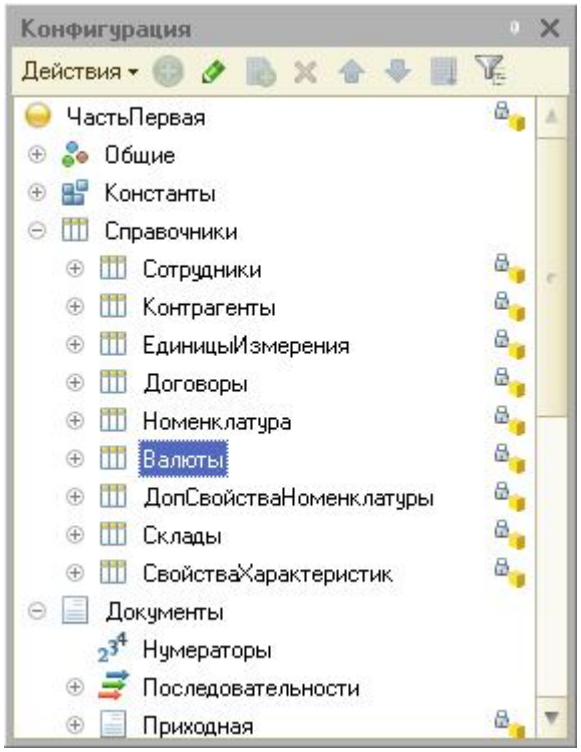
И при создании новой БД, система предложит созданный Комплект как шаблон:



«Часть первая» - это конфигурация без данных.
«Часть первая (демо)» – это конфигурация вместе с данными.

В данной конфигурации указывается состояние объекта поставки:

Желтый кубик – объект редактируется с сохранением поддержки, т.е. обозначает, что это поставка.
Замок – разрешен только просмотр объектов.



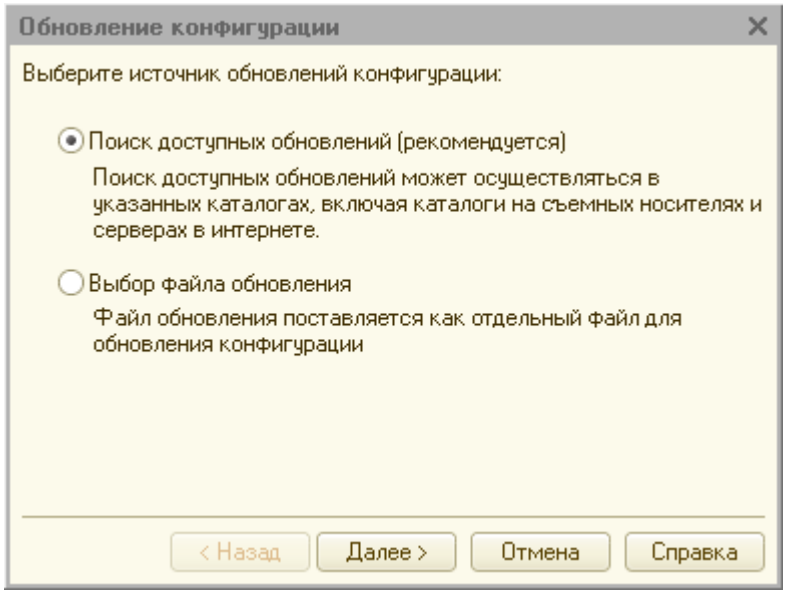
Сейчас поставка конфигурации находится на поддержке, с выключенной возможностью изменений. При этом, если надо что-либо обновить у Клиента, то просто выгрузим новую версию конфигурации:

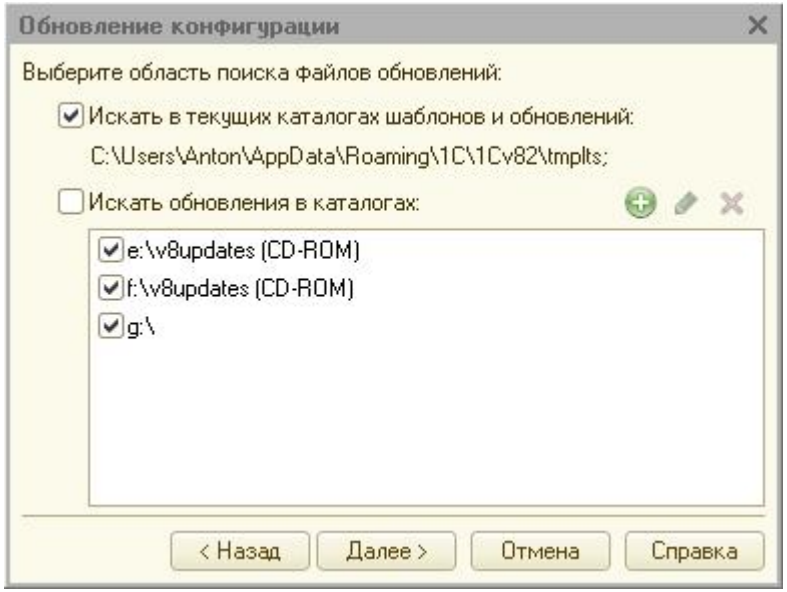
Свернем пока конфигурацию Клиента и откроем конфигурацию разработки. Создадим в ней новый справочник, при этом у нас получается как бы новая версия конфигурации: «Конфигурация – ПКМ – Свойства - Версия»:

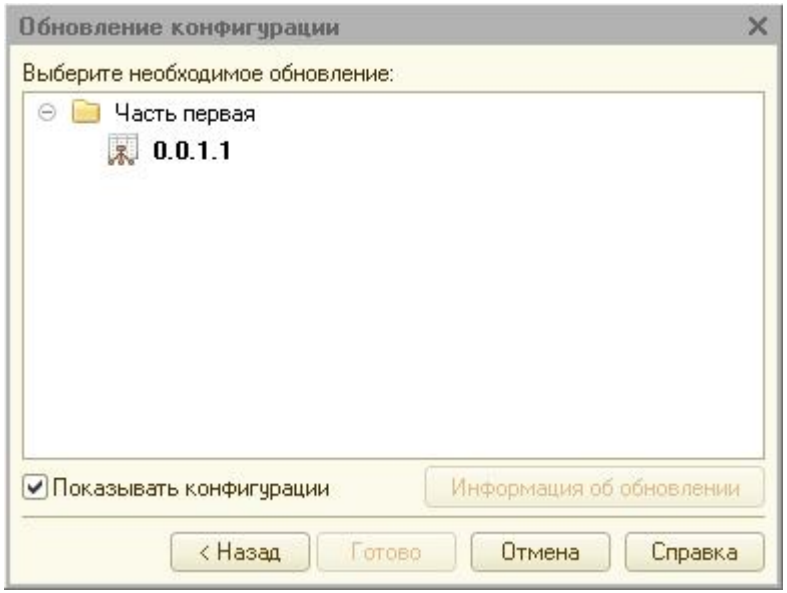
«0.0.1.1»

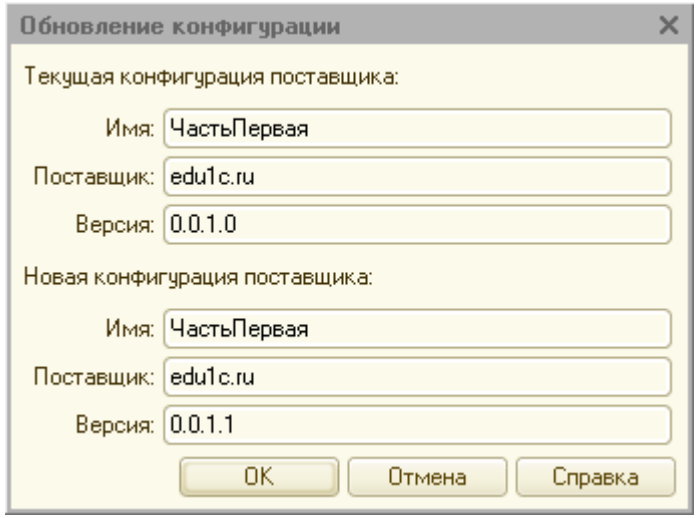
Создадим новый Комплект поставки и установим его.

Далее обновим конфигурацию Клиента – «Конфигурация – Поддержка – Обновить конфигурацию»:


Обновление конфигурации
Выберите источник обновлений конфигурации:
Поиск доступных обновлений (рекомендуется)
Поиск доступных обновлений может осуществляться в указанных каталогах, включая каталоги на съемных носителях и серверах в интернете.
Выбор файла обновления
Файл обновления поставляется как отдельный файл для обновления конфигурации
< Назад
Далее >
Отмена
Справка


Обновление конфигурации
Выберите область поиска файлов обновлений:
Искать в текущих каталогах шаблонов и обновлений:
C:\Users\Anton\AppData\Roaming\1C\1Cv82\tmplts;
Искать обновления в каталогах:
e:\v8updates (CD-ROM)
f:\v8updates (CD-ROM)
g:\
< Назад
Далее >
Отмена
Справка





При этом выполнились все новые изменения, и конфигурация по-прежнему осталась на поддержке, без права изменения вручную.

Динамическое обновление – допускается, только если не требуется реструктуризация таблиц БД. При Динамическом обновлении бывает «зависание» старого КЭШа метаданных, при этом результат обновления не виден.

Обновление можно сделать полностью автоматическим с помощью пакетного запуска программы (с указанными ключами). Можно создать Шедулер (планировщик), который будет в определенное время запускать процесс обновления.

Нетиповая конфигурация

«Конфигурация – Поддержка – Настройка поддержки – Включить возможность изменения» - разрешение на внесение изменений в конфигурацию.

Замки – дают уверенность, что конфигурация не изменялась. При этом система не проверяет, что изменено в конфигурации, а сразу обновляет.

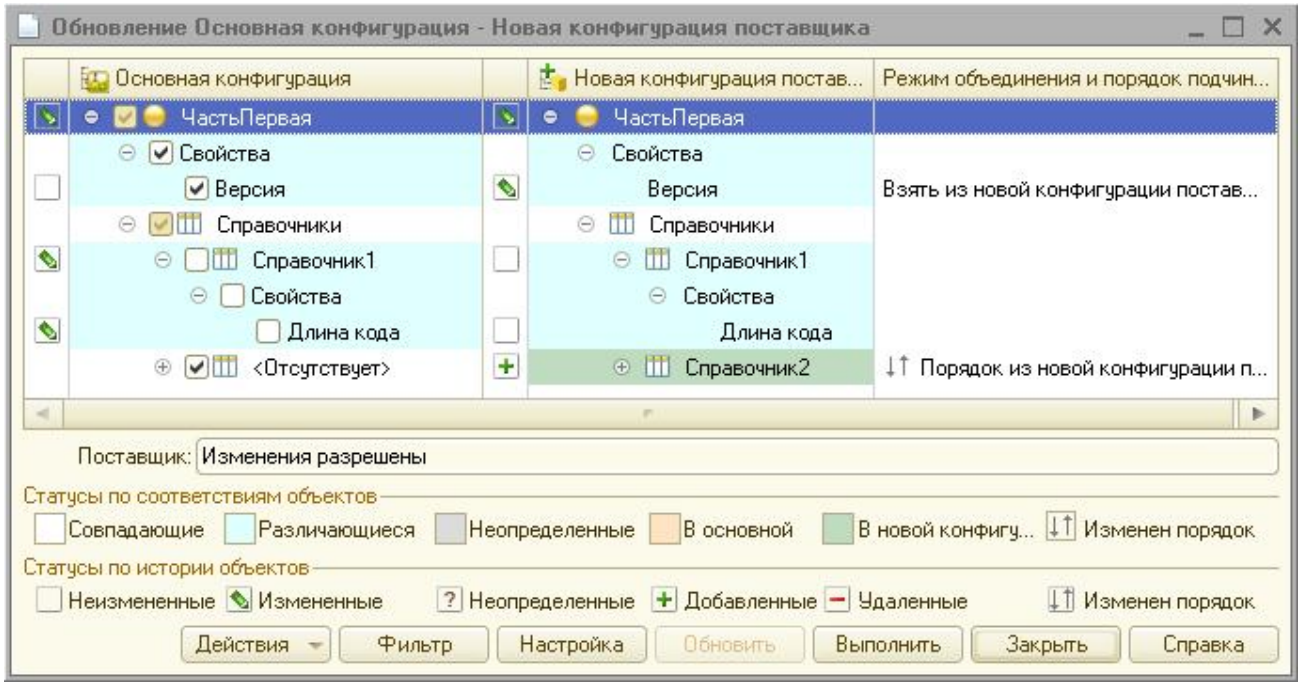
Желтые кубики – при каждом изменении объектов система запоминает, какие объекты изменились. В конфигурации образуются два файла метаданных:

- Поставщика – с замками и кубиками, т.е. на поддержке и запрещен для изменения.
- Разработчика (текущий) – с кубиками. С которым сейчас работаем.

При любом изменении система будет сравнивать конфигурацию Поставщика и Разработчика. На результат изменений наложится файл обновления. Если Разработчик менял то, что было реализовано Поставщиком, то автоматом обновление к измененным объектам применяться не будет.

Пример:
В конфигурации Клиента, в справочнике «Справочник1» изменим длину кода – сделаем 8. Сохраним конфигурацию
В оригинальной конфигурации добавим справочник «Справочник2», сохраним конфигурацию, изменим версию конфигурации, создадим Комплект поставки и установим его.

Обновим конфигурацию Клиента, используя метод обновления Поддержка. Но теперь автоматического обновления не происходит и система показывает таблицу сравнения объектов:



По умолчанию объекты измененные разработчиком не отмечены галочкой.

При включенной возможности изменения конфигурации на поддержке, размер БД будет больше, т.к. одновременно будут храниться конфигурация Поставщика и Разработчика.

Конфигурация не на поддержке

«Конфигурация – Поддержка – Настройка поддержки – Снять с поддержки»

В данном случае обновление конфигурации через Поддержку больше не работает.

Возврат конфигурации обратно на Поддержку

Допустим сняли конфигурацию с поддержки, исправили ошибки, а в следующем релизе такие же ошибки уже тоже исправлены. Надо поставить конфигурацию обратно на Поддержку:

Единственный правильный способ – «Конфигурация – Сравнить, объединить с конфигурацией из файла». После сравнения, когда конфигурация полностью идентична конфигурации поставки 1С, надо выполнить:

«Конфигурация – Загрузить конфигурацию из файла»

!!!Неправильно:

Сделать из своей конфигурации конфигурацию Поставщика. При этом для каждого объекта метаданных система создает новый идентификатор. При обновлении система проверяет объекты по идентификаторам. Тогда, при обновлении, пришедшем от 1С, идентификаторы не совпадают: а записи объектов, где идентификаторы не совпадают, при обновлении удаляются.
Итог: конфигурация обновится, но данные из БД исчезнут.

Занятие 2.19

План счетов – хранит список счетов БУ, который является предопределенным измерением регистров бухгалтерии. Аналог справочника. Представлен иерархией элементов.

Создадим новую пустую конфигурацию.

Создадим План счетов «Управленческий»:

Свойство	**Значение**
Имя	Управленческий
Синоним	Управленческий
Представление объекта	Счет
Представление списка	План счетов «Управленческий»
Закладка «Данные»	
Длина кода	5
Длина наименования	40
Маска кода	@@.@@
Автопорядок по коду	Истина
Длина порядка	5
Основное представление	В виде наименования

Свойства по умолчанию:

Длина кода – тип значения всегда «Строка». В длину кода входят разделители уровней счетов.

Автопорядок по коду – в отчетах, запросах и т.д. счета будут отсортированы по коду, а т.к. задано еще и поле «Порядок», то счета будут отсортированы по нему. При установке значения «Истина» длина порядка должна быть не меньше длины кода.

Допустим, бухгалтер ввел счета в следующем порядке:

10
10.1
10.2
10.3
10.11
10.12

По умолчанию счета будут отсортированы в следующем порядке (т.к. тип кода «Строка») и недостающие цифры заполняются пробелами:

10	10.__
10.1	10.1_
10.11	10.11
10.12	10.12
10.2	10.2_
10.3	10.3_

Вывод: План счетов по полю Код сортировать нельзя. Т.е. можно научить пользователя правильно вводить Код:

10.__ или 10.00
10._1 или 10.01
10._2 или 10.02
10._3 или 10.03
10.11
10.12

В платформе создан стандартный реквизит «Порядок». При этом пользователь может записывать коды счетов как угодно, а в поле «Порядок» они будут формироваться по определенному правилу:

Поле «Код» (пользовательское)	Поле «Порядок»
10	10.__
10.1	10._1
10.11	10._2
10.12	10._3
10.2	10._4
10.3	10._5

Автоматическое заполнение поля «Порядок» можно организовать с помощью свойства «Маска кода»:
Отвечает за формат ввода данных – пользовательский шаблон ввода данных в поле.

В примере не будем счетам давать коды в соответствии с правилами БУ, поэтому Представлять пользователю будем в виде Наименования.

Стандартные реквизиты (некоторые):
Вид – вид счета. Является системным перечислением «ВидСчетаБухгалтерскогоУчета»:

- Активный.
- Пассивный.
- АктивноПассивный.

На основе данного поля система будет возвращать обороты и остатки по счетам.

Забалансовый – булево. Означает, что счет не участвует в образовании баланса. По таким счетам не контролируется принцип двойной записи. Такие счета не могут образовывать корреспонденции с балансовыми счетами.

Использование забалансовых счетов не рекомендуется. Для хранения вспомогательных данных лучше создавать специализированные регистры накопления.

Создадим предопределенные счета (закладка «Прочее»):

План счетов Управленческий: Предопределенные счета

Действия

Имя	Код	Наименование	Вид	Забалансовый	Порядок
Счета					
Активы	01	Активы	Активный		01
Пассивы	02	Пассивы	Пассивный		02
УставныйКапитал	02.1	Уставный капитал	Пассивный		02. 1
Взаиморасчеты	03	Взаиморасчеты	Активный/Пассивный		03
РасчетыСУчредителями	03.1	Расчеты с учредителями	Активный/Пассивный		03. 1

В режиме исполнения:

Система не даст создать новый счет, т.к. поле «Порядок» при его использовании становится обязательным для заполнения, но оно не отображается (по умолчанию) в пользовательском режиме.

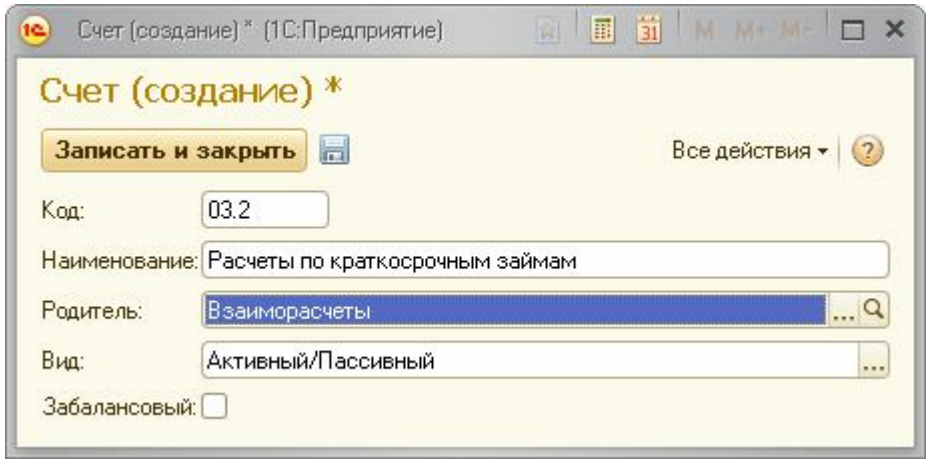
Сделаем, чтобы при записи элементов Плана счетов поле «Порядок» заполнялось автоматом. Модуль объекта:

```
Процедура ПередЗаписью(Отказ)
    Порядок = ПолучитьПорядокКода();
КонецПроцедуры
```

«`ПолучитьПорядокКода()`» - метод объекта Плана счетов, который на основе Маски кода и Кода заполняет порядок.

В режиме исполнения:

Создадим счет:



Создадим Регистр бухгалтерии «Управленческий»:

Свойство	**Значение**
Имя	Управленческий
Синоним	Управленческий
План счетов	Управленческий
Корреспонденция	Истина
Представление списка	Проводки «Управленческий»
Закладка «Данные»	
Ресурсы	
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2
Выделять отрицательные	Истина

План счетов – тип предопределенного измерения Регистра бухгалтерии (Счет).

Корреспонденция – определяет тип поведения Регистра бухгалтерии. Представление бухгалтерской записи:

С корреспонденцией

Счет Дт	Счет Кт	Сумма
Товары	Поставщики	10000

Без корреспонденции

Счет	Вид движения	Сумма
Товары	Дебет	10000
Поставщики	Кредит	10000

С точки зрения БУ эти записи идентичны и одинаково влияют на остатки. Независимо от выбора, система будет контролировать остатки и обороты по принципу двойной записи – обороты и остатки по Дт и Кт должны быть равны.

Но только в бухгалтерской записи с корреспонденцией можно получить «Корреспондирующие обороты» - обороты между конкретными счетами.

«Корреспонденция - Истина» - при этом система создает два поля «СчетДт» и «СчетКт», но измерение в регистре все равно одно – «Счет».

По ресурсу «Сумма»:
«Неотрицательное - Ложь» - могут быть проводки со знаком «-».
«Проверка заполнения – Не проверять» - нулевые записи по регистру бухгалтерии допускаются.
«Балансовый - Истина» - контроль баланса значений данного ресурса по Дт и Кт.

Создадим документ-регистратор, формирующий движения по данному регистру – «Операция бух»:

Свойство	Значение
Имя	ОперацияБух
Синоним	Операция бух
Представление объекта	Операция
Представление списка	Журнал операций

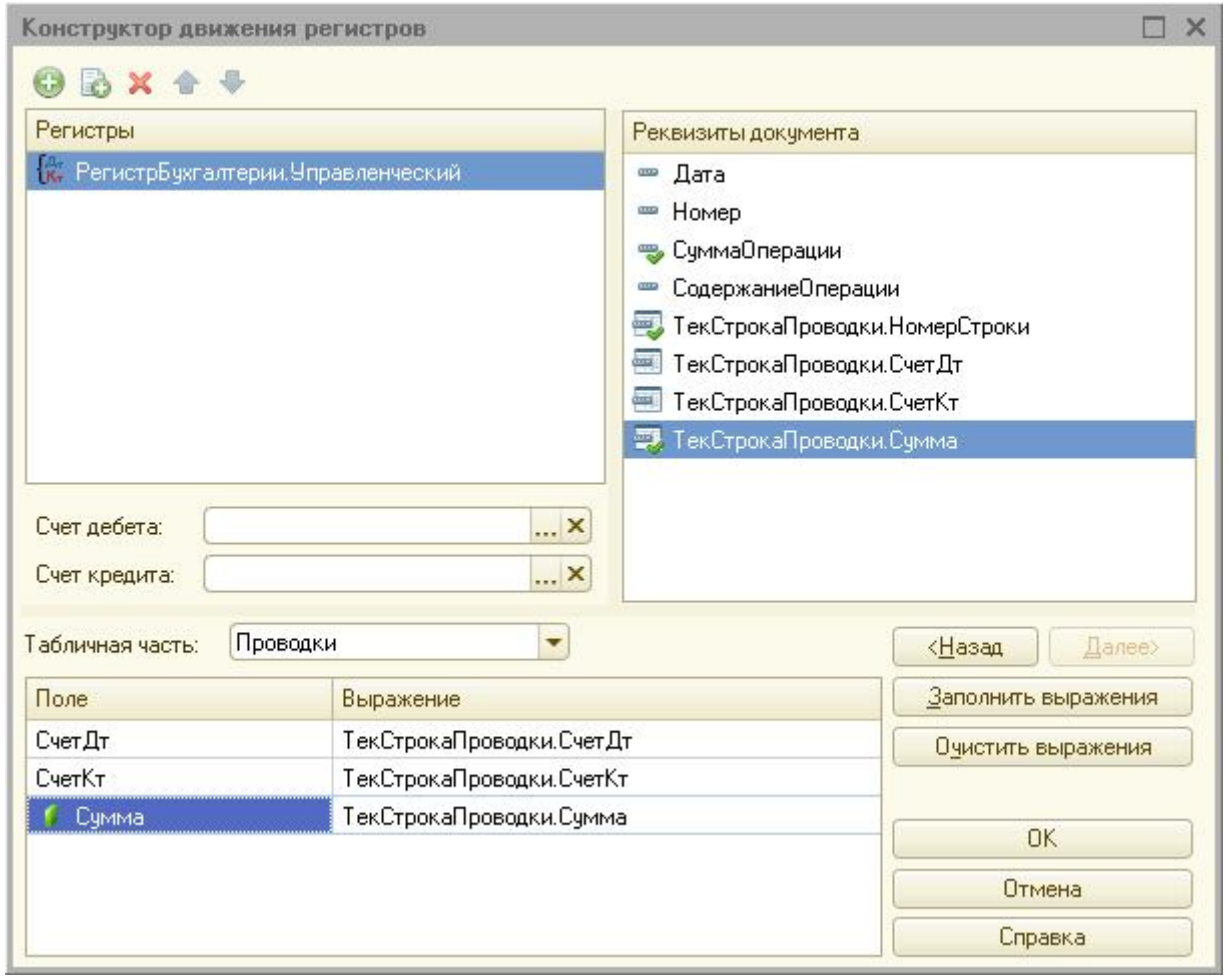
Структуру данных сделаем аналогичной структуре регистра бухгалтерии «Управленческий»:

Свойство	Значение
Закладка «Данные»	
Реквизиты	
Сумма операции	
Имя	СуммаОперации
Синоним	Сумма операции
Тип	Число 15, 2
Содержание операции	
Имя	СодержаниеОперации
Синоним	Содержание операции
Тип	Строка
Неограниченная длина	Истина
Табличные части	
Проводки	
Счет Дт	
Имя	СчетДт
Синоним	Счет Дт
Тип	ПланСчетовСсылка.Управленческий
Проверка заполнения	Выдавать ошибку
Счет Кт	
Имя	СчетКт
Синоним	Счет Кт
Тип	ПланСчетовСсылка.Управленческий
Проверка заполнения	Выдавать ошибку
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2

Сумма операции – это не сумма по регистру, данная сумма выводится для отображения общей суммы операции.

Свойство	Значение
Закладка «Движения»	
Оперативное проведение	Запретить
Удаление движений	Удалять автоматически

Конструктором создадим движения по регистру бухгалтерии «Управленческий»:

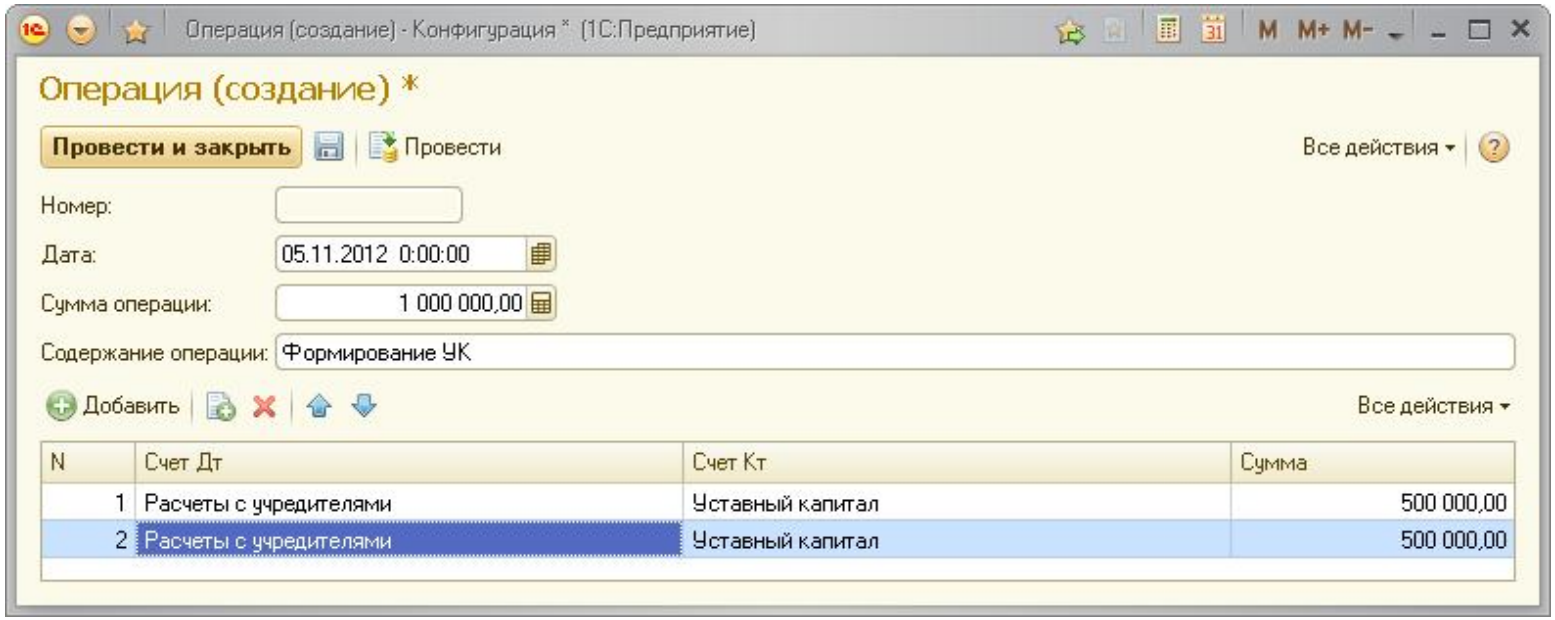


Итого код:

```
Процедура ОбработкаПроведения(Отказ, Режим)
	// регистр Управленческий
	Движения.Управленческий.Записывать = Истина;
	Для Каждого ТекСтрокаПроводки Из Проводки Цикл
		Движение = Движения.Управленческий.Добавить();
		Движение.Период = Дата;
		Движение.СчетДт = ТекСтрокаПроводки.СчетДт;
		Движение.СчетКт = ТекСтрокаПроводки.СчетКт;
		Движение.Сумма = ТекСтрокаПроводки.Сумма;
	КонецЦикла;
КонецПроцедуры
```

В режиме исполнения:

Создадим операцию:



Добавим отображение в Журнале проводок списка проводок текущего документа «Операция»:

Добавим форму списка документа «Операция». К выбранным по умолчанию реквизитам добавим:

- СуммаОперации.
- СодержаниеОперации.

Добавим таблицу, отображающую часть регистра бухгалтерии – реквизит формы «Проводки»:

Свойство	Значение
Имя	Проводки
Заголовок	Проводки
Тип	ДинамическийСписок
Основная таблица	РегистрБухгалтерии.Управленческий.ДвиженияССубконто
Произвольный запрос	Истина

ДвиженияССубконто – в движение включается отображение аналитики (в примере ее пока нет).

Настройка списка – отредактируем запрос:

```
ВЫБРАТЬ
     Управленческий.Период,
     Управленческий.Регистратор,
     Управленческий.НомерСтроки,
     Управленческий.Активность,
     Управленческий.СчетДт,
     Управленческий.СчетКт,
     Управленческий.Сумма,
     Управленческий.МоментВремени
ИЗ
     РегистрБухгалтерии.Управленческий КАК Управленческий
ГДЕ
     Управленческий.Регистратор = &Регистратор
```

И укажем порядок «НомерСтроки – По возрастанию»

Добавим реквизит «Проводки» в форму.

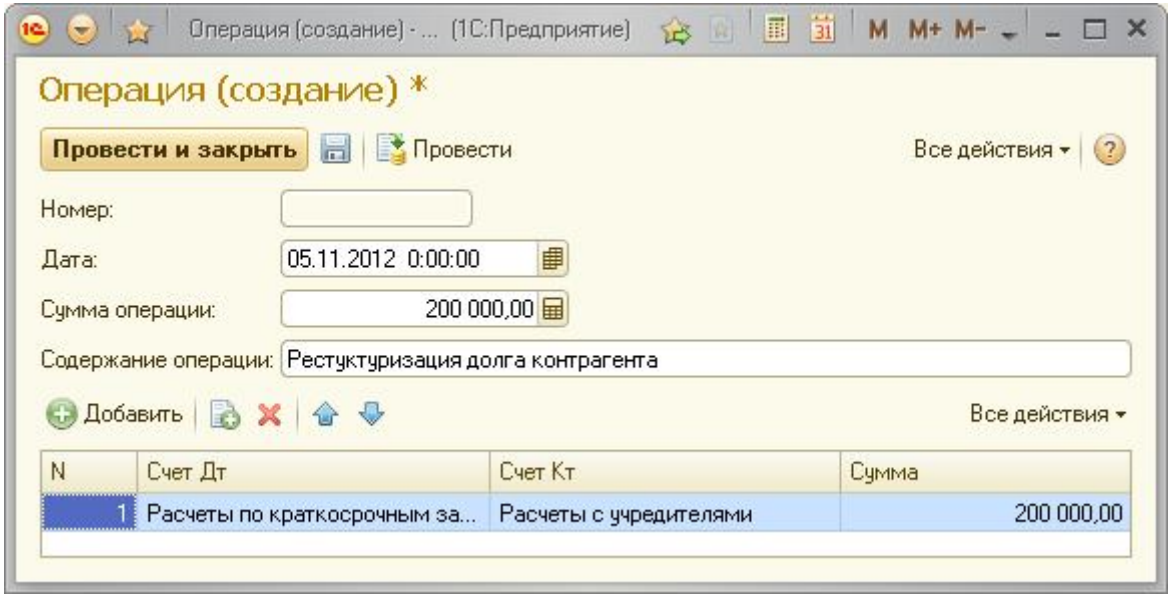
В «Списке» при переходе между элементами установим отбор в динамическом списке «Проводки»:

```
&НаКлиенте
Процедура СписокПриАктивизацииСтроки(Элемент)
    УстановитьОтборВПроводках(Элементы.Список.ТекущиеДанные.Ссылка);
КонецПроцедуры

&НаСервере
Процедура УстановитьОтборВПроводках(Регистратор)
    Проводки.Параметры.УстановитьЗначениеПараметра("Регистратор", Регистратор);
КонецПроцедуры
```

В реквизитах формы пометим, что реквизит «Ссылка» будет использоваться в форме всегда, чтобы избежать ошибки при выполнении процедуры «`УстановитьОтборВПроводках()`».

В режиме исполнения:

Создадим еще одну операцию:



Теперь при переходе по документам видно, что содержание таблицы с проводками обновляется.

Создадим отчет «Оборотно-сальдовая ведомость»:

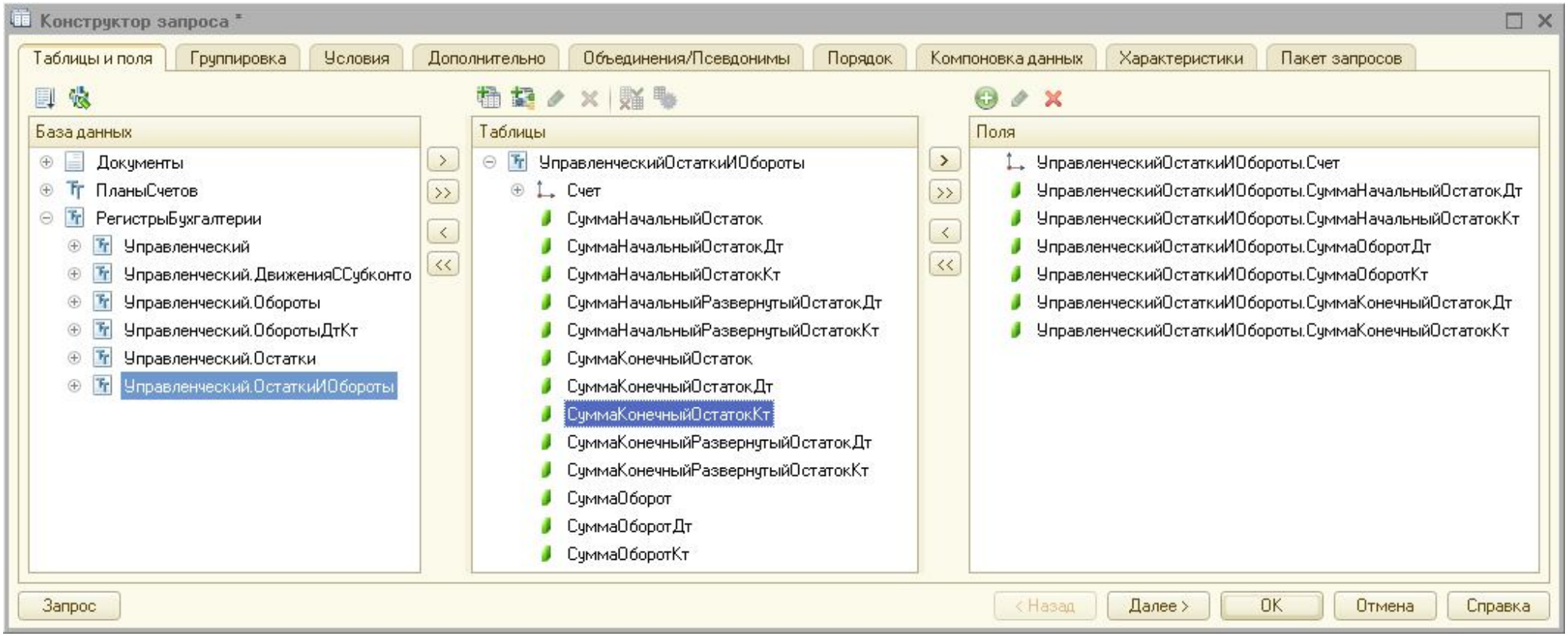
Свойство	Значение
Имя	ОСВ
Заголовок	Оборотно-сальдовая ведомость

Откроем СКД, добавим «Набор данных-запрос» и откроем «Конструктор запроса»:

В регистре бухгалтерии 3 «реальных» таблицы – в таких таблицах доступны неактивные записи (Активность - Ложь).
Таблица «Управленческий» - отображает синтетические данные (счета и сумма).
Таблица «Управленческий.ДвиженияССубконто» - отображает аналитические данные.
Таблица «Управленческий.Обороты».
Таблица «Управленческий.Остатки» - поле «Остаток» имеет несколько видов:

СуммаОстаток	В данном поле сумма хранится физически
СуммаОстатокДт СуммаОстатокКт	В каком из этих полей отобразится сумма, со знаком «+» или «-», зависит от Вида счета – если счет Активный, то остаток будет в Дт, если Пассивный, то в Кт. Причем значение «СуммаОстаток» будет возвращено с обратным знаком (если «-», то в Кт будет «+» и наоборот). Если счет Активно-пассивный, то остаток будет там, где Движение больше.
СуммаРазвернутыйОстатокДт СуммаРазвернутыйОстатокКт	Используется только для построения итогов. Показывает отдельно остаток по Дт и по Кт. Пример: Счет «Расчеты с поставщиками». ➢ СуммаОстаток – покажет итог: либо я должен, либо мне должны. ➢ СуммаРазвернутыйОстаток – покажет, и сколько я должен и сколько мне должны.

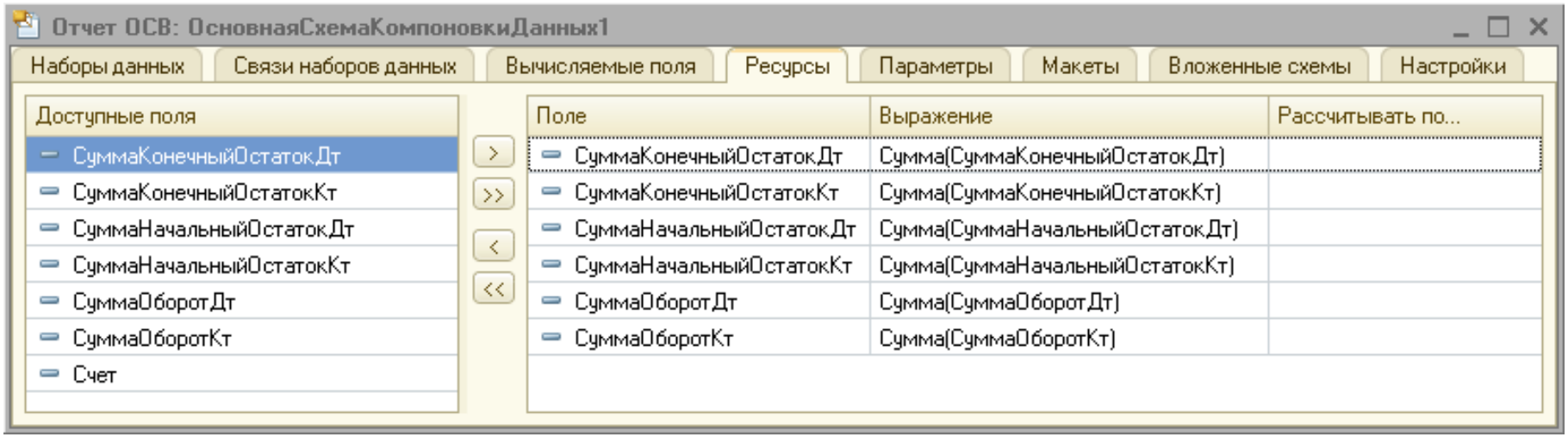
Желательно аналитику и служебные данные убирать из регистра бухгалтерии, т.к. сам по себе регистр бухгалтерии не очень быстрый механизм.



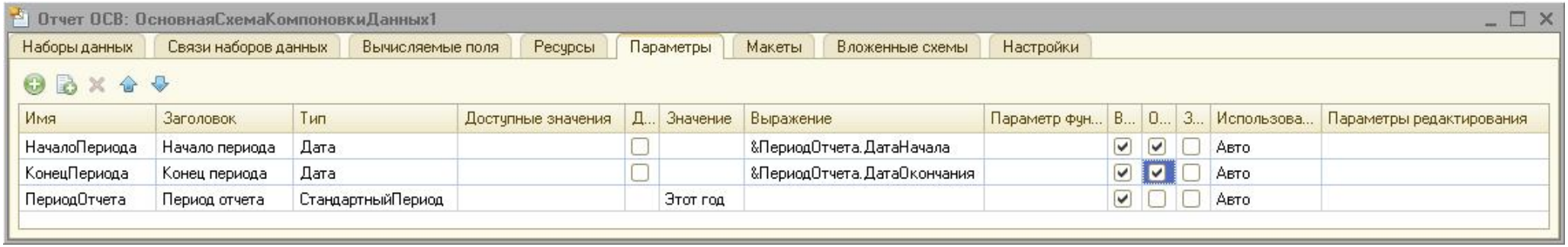
Вернемся в СКД:

СКД добавляет Роли, но из-за этого в данном случае неправильно формируются Итоги, очистим значения Ролей:

	Поле	Путь / Автозаголовок	Ограничение поля: По...	Ус...	Гр...	Уп...	Ограничение реквиз.: По...	Ус...	Гр...	Уп...	Роль	Выражение п... / Выражения упорядочиван...	Проверка иерархии: / Набор данных / Параметр	Тип значения / Доступные значения	Оформление / Параметры редактирова...
−	Счет	Счет	☐	☐	☐	☐					Измерение ...				
		☐ Счет	☐	☐	☐	☐									
−	СуммаКонечн...	СуммаКонечныйОстатокКт	☐	☐	☐	☐									
		☐ Сумма Конечный остаток Кт	☑	☑	☑	☑									
−	СуммаКонечн...	СуммаКонечныйОстатокДт	☐	☐	☐	☐									
		☐ Сумма Конечный остаток ...	☑	☑	☑	☑									
−	СуммаНачал...	СуммаНачальныйОстатокКт	☐	☐	☐	☐									
		☐ Сумма Начальный остато...	☑	☑	☑	☑									
−	СуммаНачал...	СуммаНачальныйОстатокДт	☐	☐	☐	☐									
		☐ Сумма Начальный остато...	☑	☑	☑	☑									
−	СуммаОборо...	СуммаОборотКт	☐	☐	☐	☐									
		☐ Сумма Оборот Кт	☑	☑	☑	☑									
−	СуммаОборо...	СуммаОборотДт	☐	☐	☐	☐									
		☐ Сумма Оборот Дт	☑	☑	☑	☑									



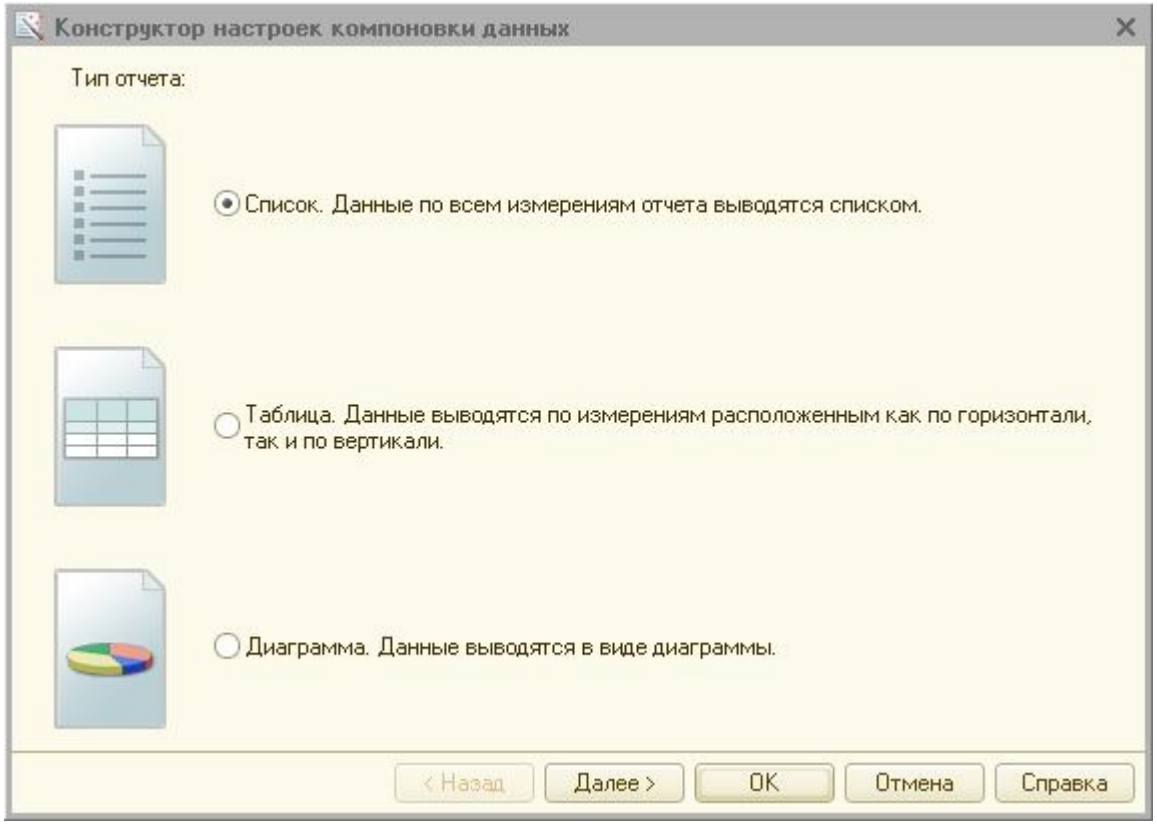
Настроим Параметры, чтобы пользователь указывал только одно значение параметра «Период отчета»:

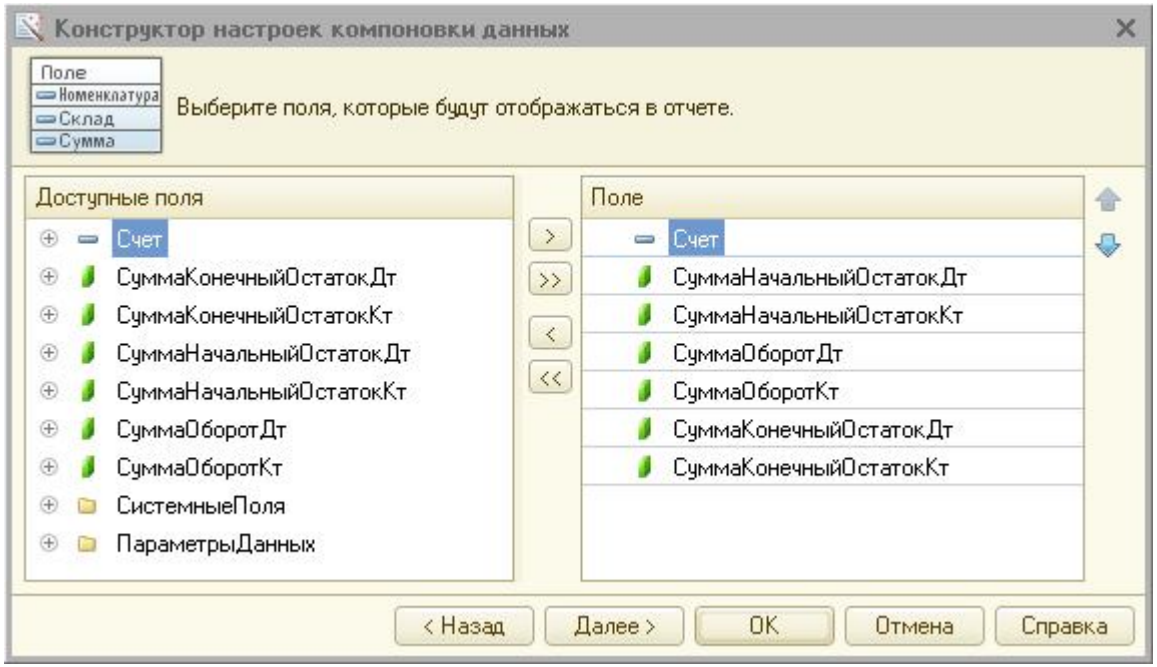


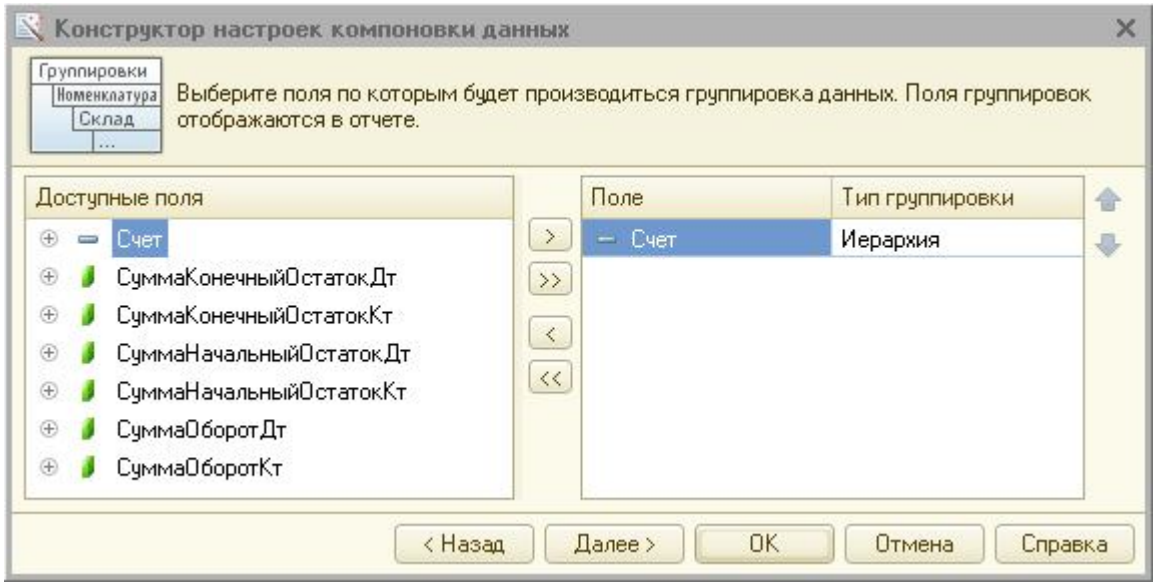
Имя	Заголовок	Тип	Доступные значения	Д...	Значение	Выражение	Параметр фун...	В...	О...	З...	Использова...	Параметры редактирования
НачалоПериода	Начало периода	Дата		☐		&ПериодОтчета.ДатаНачала		☑	☑	☐	Авто	
КонецПериода	Конец периода	Дата		☐		&ПериодОтчета.ДатаОкончания		☑	☑	☐	Авто	
ПериодОтчета	Период отчета	СтандартныйПериод			Этот год			☑	☐	☐	Авто	

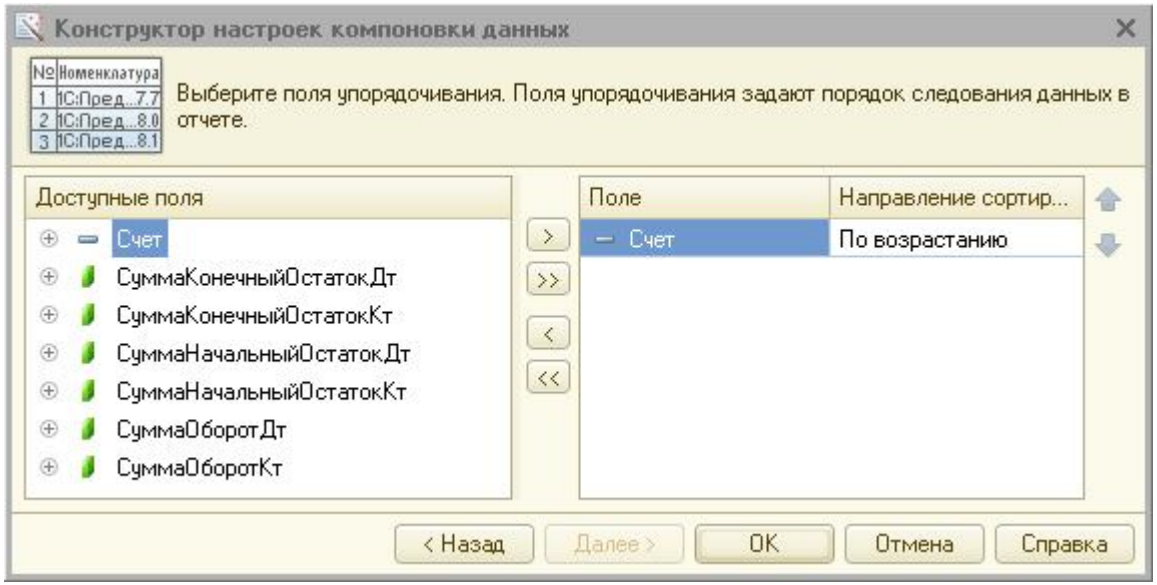
Значение – это значение по умолчанию.
Дата окончания в Стандартном периоде всегда равна концу дня.
«Ограничение доступности – Ложь» - у полей «НачалоПериода» и «КонецПериода», чтобы эти поля не отображались в пользовательском режиме.

Закладка «Настройки» - сформируем основной вариант отчета:

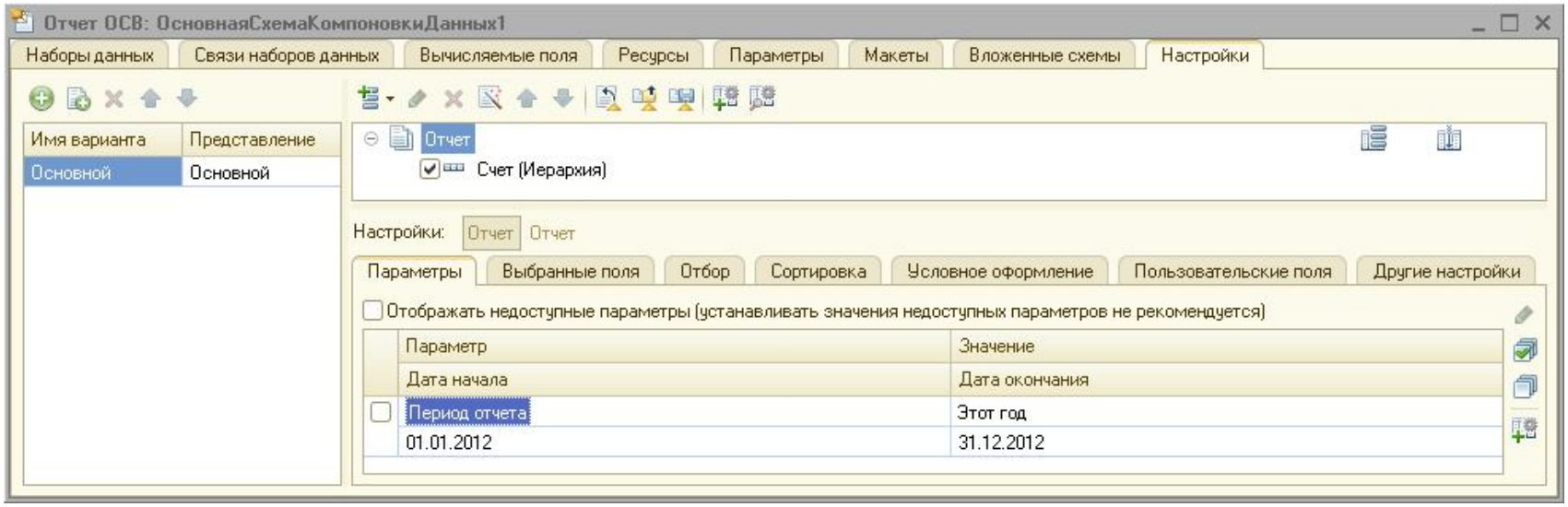



Конструктор настроек компоновки данных
Поле
Номенклатура
Склад
Сумма
Выберите поля, которые будут отображаться в отчете.
Доступные поля
Счет
СуммаКонечныйОстатокДт
СуммаКонечныйОстатокКт
СуммаНачальныйОстатокДт
СуммаНачальныйОстатокКт
СуммаОборотДт
СуммаОборотКт
СистемныеПоля
ПараметрыДанных
Поле
Счет
СуммаНачальныйОстатокДт
СуммаНачальныйОстатокКт
СуммаОборотДт
СуммаОборотКт
СуммаКонечныйОстатокДт
СуммаКонечныйОстатокКт
< Назад
Далее >
OK
Отмена
Справка

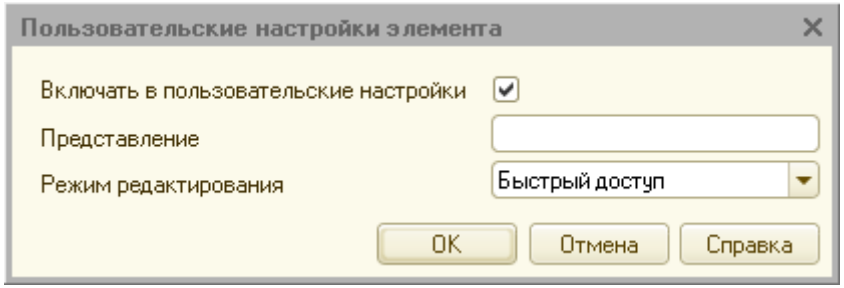

Конструктор настроек компоновки данных
Группировки
Номенклатура
Склад
...
Выберите поля по которым будет производиться группировка данных. Поля группировок отображаются в отчете.
Доступные поля
Счет
СуммаКонечныйОстатокДт
СуммаКонечныйОстатокКт
СуммаНачальныйОстатокДт
СуммаНачальныйОстатокКт
СуммаОборотДт
СуммаОборотКт
Поле
Тип группировки
Счет
Иерархия
< Назад
Далее >
OK
Отмена
Справка



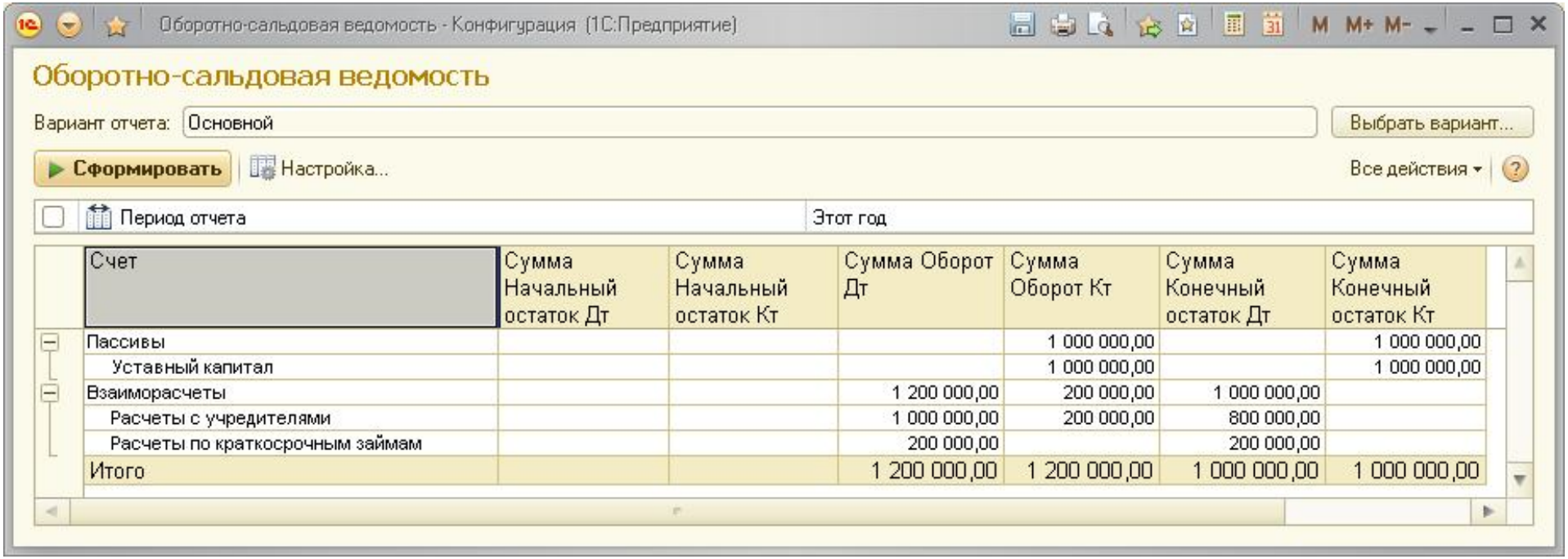
Период отчета включим в пользовательские настройки:



«Период отчета – ПКМ – Свойства элемента пользовательских настроек»:



В режиме исполнения:



Счет	Сумма Начальный остаток Дт	Сумма Начальный остаток Кт	Сумма Оборот Дт	Сумма Оборот Кт	Сумма Конечный остаток Дт	Сумма Конечный остаток Кт
Пассивы				1 000 000,00		1 000 000,00
Уставный капитал				1 000 000,00		1 000 000,00
Взаиморасчеты			1 200 000,00	200 000,00	1 000 000,00	
Расчеты с учредителями			1 000 000,00	200 000,00	800 000,00	
Расчеты по краткосрочным займам			200 000,00		200 000,00	
Итого			1 200 000,00	1 200 000,00	1 000 000,00	1 000 000,00

Занятие 2.20

Объект «План видов характеристик» - используется для описания видов аналитики. Похож на справочник, отличается тем, что создает новый тип значения – «Коллекция значений характеристик».

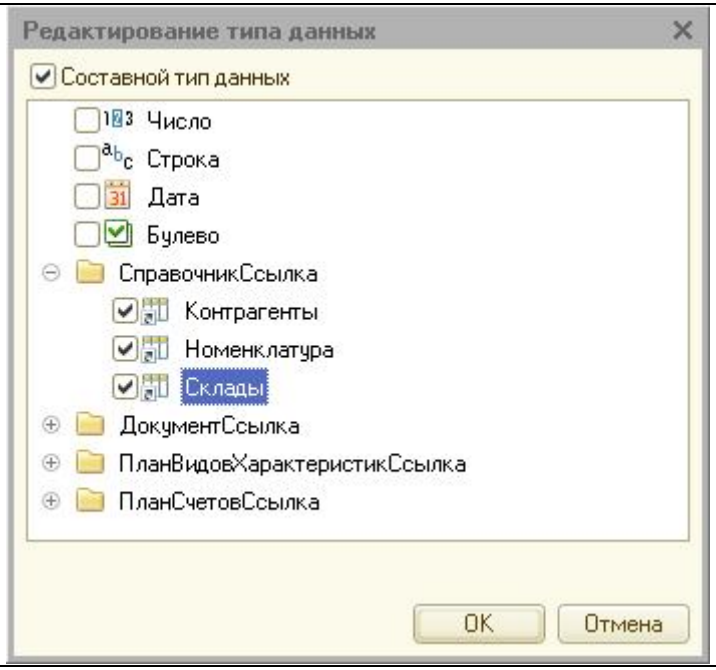
Виды аналитики в примере:

- Счета.
- Контрагенты.
- Товары.
- Склады.

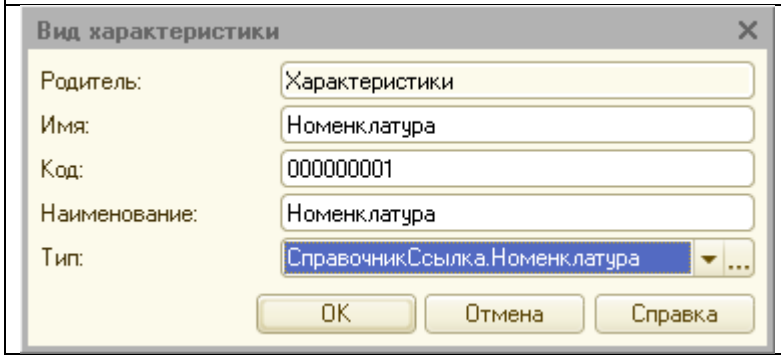
Для того чтобы эти виды аналитики были предопределены в системе, создадим справочники (без настройки, просто как объекты):

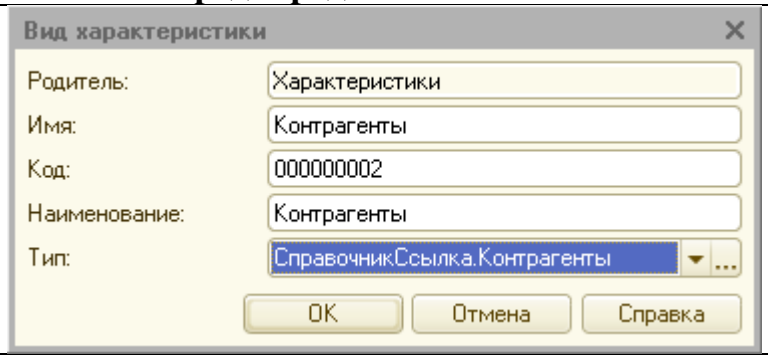
- Контрагенты.
- Номенклатура.
- Склады.

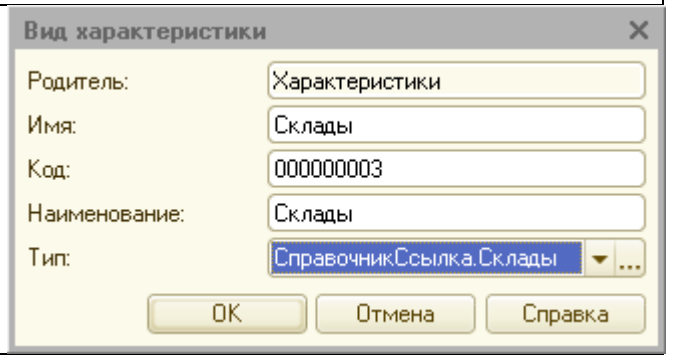
Создадим План видов характеристик «Виды субконто»:

Свойство	Значение
Имя	ВидыСубконто
Синоним	Виды субконто
Тип значения характеристик	

Свойство	Значение
Представление объекта	Вид субконто
Представление списка	Виды субконто
Закладка «Прочее»	
Предопределенные	







Влияние примитивных типов (строка, число, булево и т.д.) на регистры бухгалтерии:
Остатки по счетам в системе хранятся в разрезе счета и видов аналитики. Например, остатки по счету «Товары на складах»:

Таблицы регистров для извлечения остатков, оборотов проиндексированы. Состав основного индекса:

Период – 1 поле.
Счет – 1 поле.
Если в характеристики субконто входят только ссылочные значения, то система занимает под них по 3 поля индекса:

- Тип.
- Вид.
- Значение.

Например:

Тип	Вид	Значение
Справочник	Номенклатура	Ложка
Документ	Приходная	№55

Период	Счет	Субконто1 (Номенклатура)	Субконто2 (Склады)	СуммаОстаток
1	1	3	3	

Итого в индексе занято 8 полей.

У SQL-сервера есть ограничение – в основной индекс таблицы входят максимум 16 полей.
Если добавить еще одно субконто, то займем уже 11 полей.

Если, кроме ссылок, в Плане видов характеристик указать примитивный тип, например, число, то система под него займет на каждое субконто еще по одному полю:

Период	Счет	Субконто1 (Номенклатура)	Субконто2 (Склады)	Субконто3	СуммаОстаток
1	1	3	3	3	
	Число	1	1	1	

Итого 14 полей.

Допустим, укажем еще один примитивный тип – «Дата». Система под него займет на каждое субконто еще по одному полю. Кроме того, при сочетании «Число - Дата» система занимает еще по одному полю на каждое субконто:

Период	Счет	Субконто1 (Номенклатура)	Субконто2 (Склады)	Субконто3	СуммаОстаток
1	1	3	3	3	
	Число	1	1	1	
	Дата	1 + 1	1 + 1	1 + 1	

При этом превышено максимальное количество полей.

Если добавить еще один примитивный тип, то система займет на каждое субконто еще по одному полю и по одному полю на каждое сочетание.

Последствия превышения максимального количества полей:
Таблица будет проиндексирована не полностью и, как следствие, запросы будут выполняться неэффективно. Например:
Надо получить остаток по товару, который находится на определенном складе, в определенной ячейке, со сроком годности, который кончается позже определенной даты. При этом система возвращает не проиндексированный массив значений, который Сервер 1С будет обрабатывать перебором, т.к. нет индексации.

Привязка аналитики к счетам БУ:
Раньше, чтобы вести аналитику, открывались субсчета. Например:
60 «Поставщики».
60.1 «Авансы, выданные поставщикам».

Если нужна была аналитика по счету, то добавляли субсчета:

60.1.1 «Авансы ООО «Мир»».

Если по ООО «Мир» были взаиморасчеты в разрезе договоров, то добавляли еще субсчета:

60.1.1.1 «ООО «Мир» договор №3».
60.2 «Расчеты с поставщиками за товары».

и т.д.

Минусы:
Аналитика жестко привязана к счетам – например, расчеты с ООО «Мир» закончились, а счета «висят», а удалить их нельзя.
Линейная иерархия – например, Контрагенты хранятся только в разрезе Договоров, Товары только в разрезе Складов и т.д. При этом, если надо в отчете показать Склады в разрезе Товаров, то такая модель не позволит этого сделать быстро.

Субконто – многоуровневый аналитический учет.
В системе 1С учет по аналитике ведется следующим образом:
Одно измерение – это счета.

Счет	Субконто1 (Контрагент)	Субконто2 (Договор)
60.1 «Авансы, выданные поставщикам»	ООО «Мир»	№3

Субконто – это дополнительное, подчиненное измерение в регистре бухгалтерии.

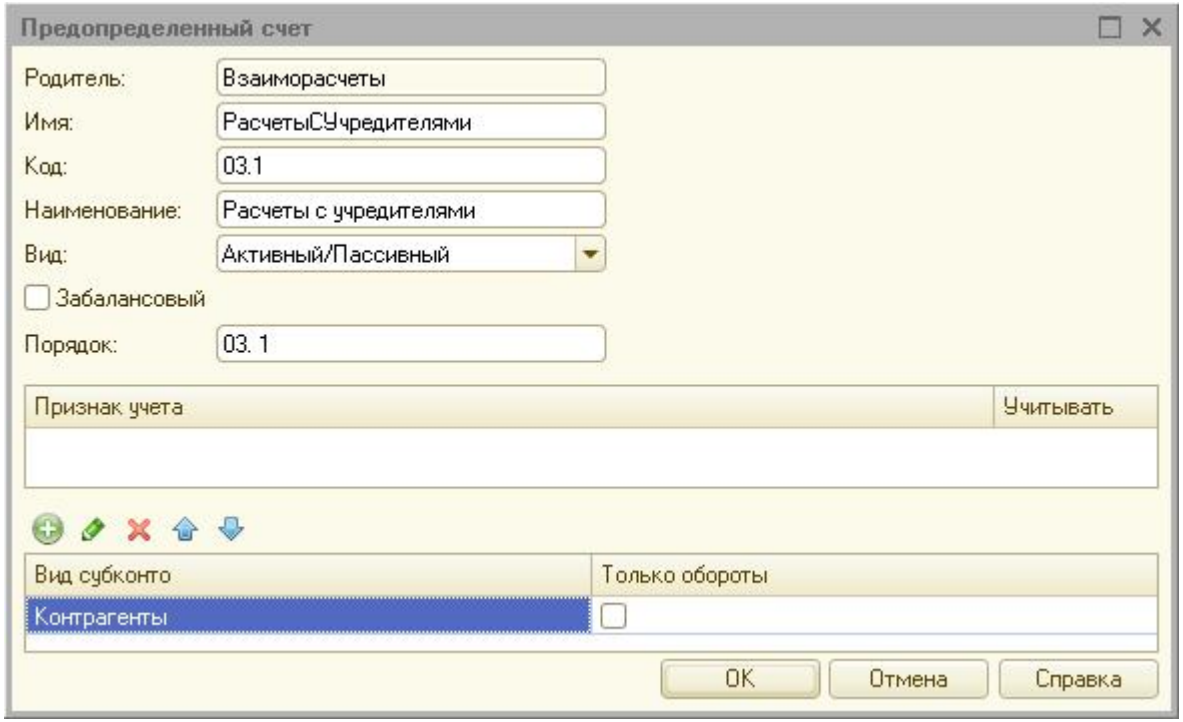
Значение по Субконто можно получать в разрезе счета или без него.

Свяжем План видов характеристик «ВидыСубконто» с Планом счетов «Управленческий».
План счетов «Управленческий», закладка «Субконто»:

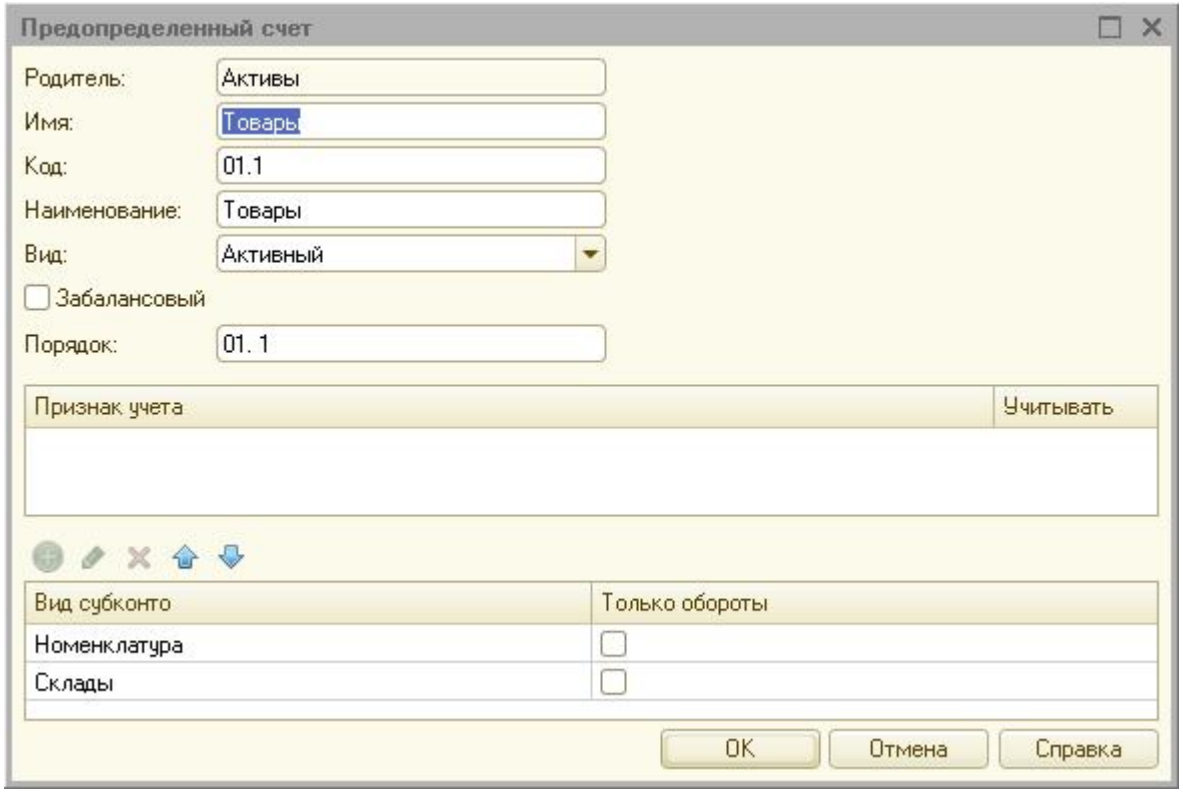
Свойство	Значение
Виды субконто	ВидыСубконто
Максимальное количество субконто	2

Данная настройка включила предопределенную табличную часть «Виды субконто» у счетов.

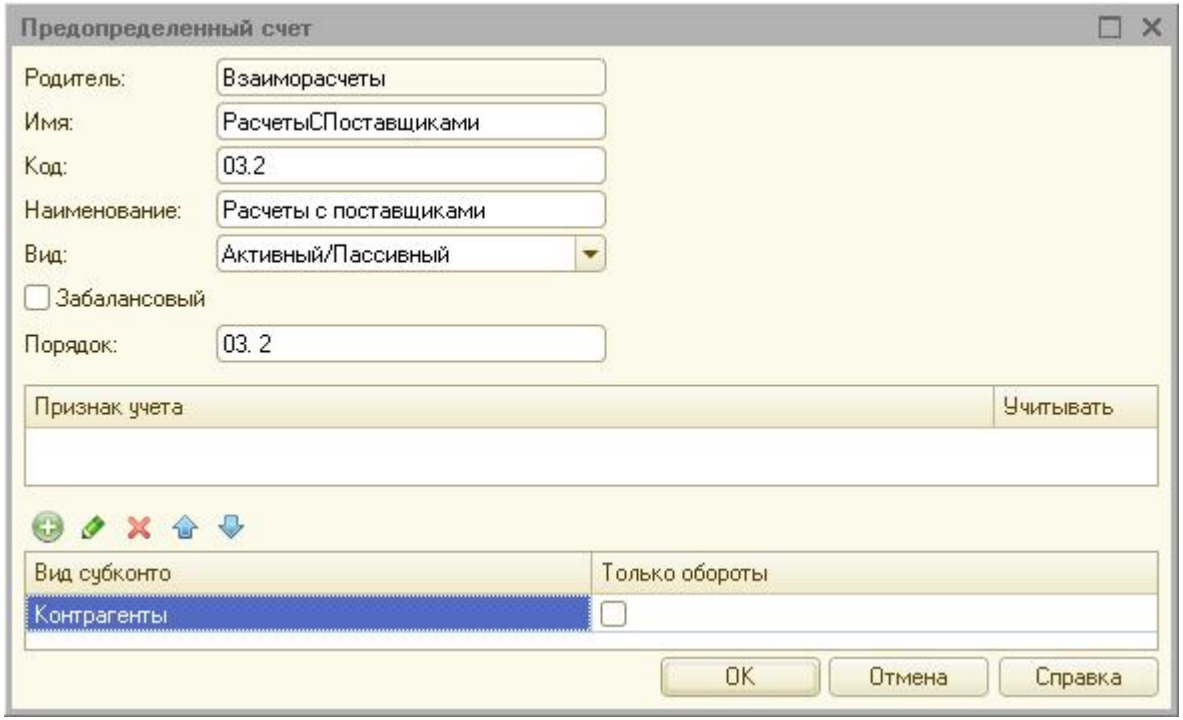
Отредактируем счета: «Закладка «Прочее» - Предопределенные»



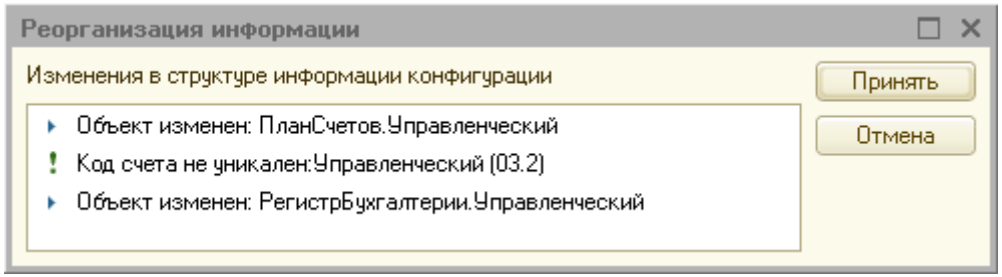
Создадим счет «Товары»:



Создадим счет «Расчеты с поставщиками»:



При сохранении изменений в конфигурации БД, система выдаст ошибку, но при этом конфигурацию сохранить можно:



Эта ошибка говорит о том, что у нас уже есть счет с таким кодом, причем система учитывает даже те счета, которые были созданы в пользовательском режиме.

Сформируем движения по Регистрам бухгалтерии:

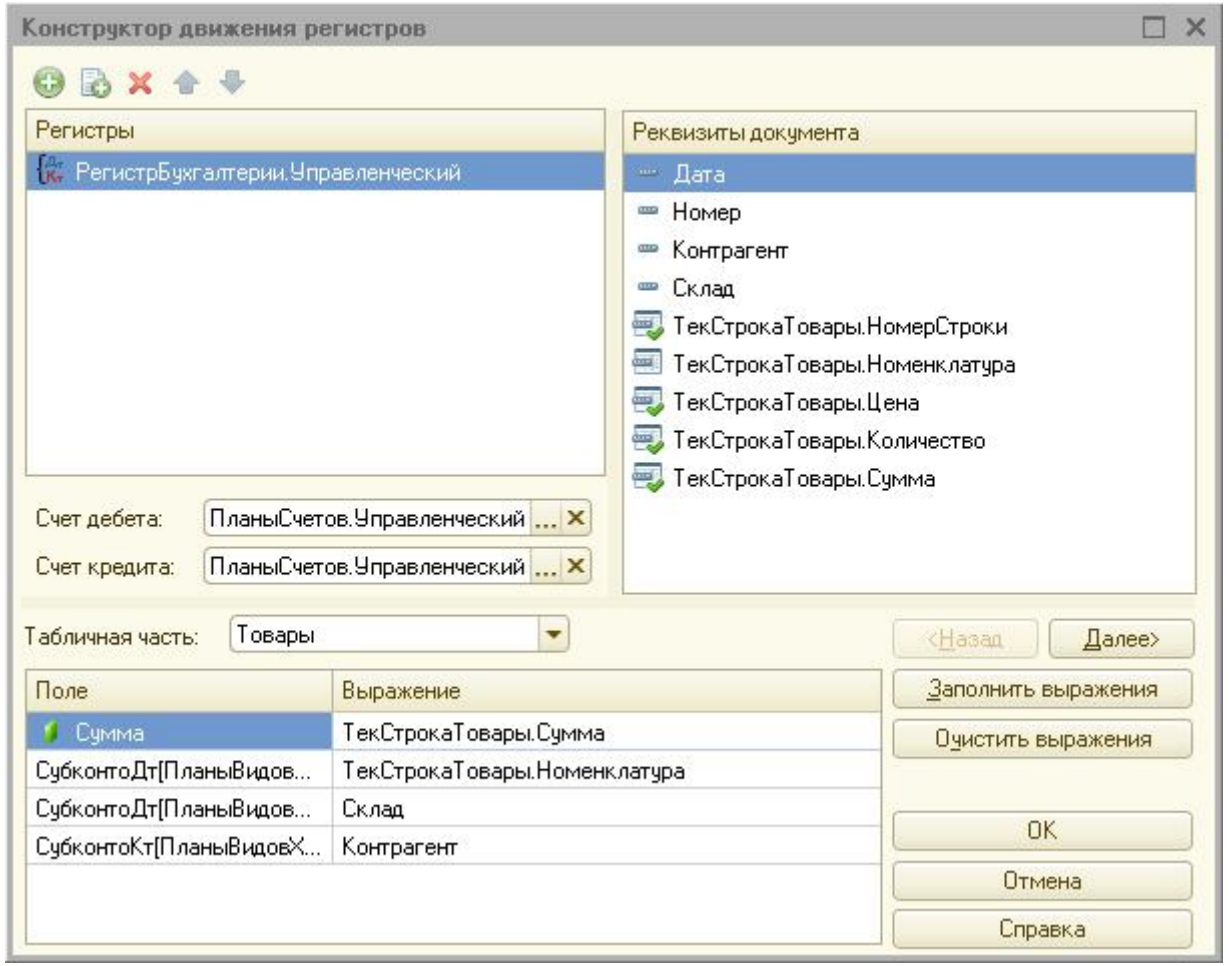
Создадим документ «Приходная», фиксирующий поступление товара в разрезе Номенклатуры, Складов и Контрагентов.

Свойство	Значение
Имя	Приходная
Синоним	Приходная
Представление объекта	Поступление товаров
Представление списка	Приходные
Закладка «Данные»	
Реквизиты	
Контрагент	
Имя	Контрагент
Синоним	Контрагент
Тип	СправочникСсылка.Контрагенты
Склад	
Имя	Склад
Синоним	Склад
Тип	СправочникСсылка.Склады
Табличная часть «Товары»	
Номенклатура	
Имя	Номенклатура
Синоним	Номенклатура
Тип	СправочникСсылка.Номенклатура
Цена	
Имя	Цена
Синоним	Цена
Тип	Число 15, 2
Количество	
Имя	Количество
Синоним	Количество
Тип	Число 12, 3

Свойство	Значение
Закладка «Данные»	
Табличная часть «Товары»	
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2
Закладка «Движения»	
Проведение	Разрешить
Оперативное проведение	Запретить
Удаление движений	Удалять автоматически

Отметим, что движения будут производиться по регистру бухгалтерии «Управленческий».

Откроем «Конструктор движений» и заполним данные:




```
Процедура ОбработкаПроведения(Отказ, Режим)
    // регистр Управленческий
    Движения.Управленческий.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.Товары;
        Движение.СчетКт = ПланыСчетов.Управленческий.РасчетыСПоставщиками;
        Движение.Период = Дата;
        Движение.Сумма = ТекСтрокаТовары.Сумма;
```

```
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = ТекСтрокаТовары.Номенклатура;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
    КонецЦикла;
КонецПроцедуры
```

Хранение движений в Регистре бухгалтерии на физическом уровне:

До добавления Субконто, таблица Регистра бухгалтерии выглядела так:

Период	Регистратор	НомерСтроки	СчетДт	СчетКт	Сумма
01.01.01.	Приходная №1	1	Товары	Поставщики	1000

При добавлении Субконто данная таблица не изменилась. Добавилась еще одна таблица – «Движения Субконто». Связь таблиц производится по Ключу «Регистратор – Номер строки»

Движения

Период	Регистратор	НомерСтроки	СчетДт	СчетКт	Сумма
01.01.01.	Приходная №1	1	Товары	Поставщики	1000

ДвиженияСубконто

Регистратор	НомерСтроки	ВидДвижения	ВидСубконто	Значение
Приходная №1	1	Дебет	Номенклатура	Ложка
Приходная №1	1	Дебет	Склады	Парнас
Приходная №1	1	Кредит	Контрагенты	ООО Мир

Такая форма оптимальна с точки зрения непостоянства количества Субконто по счетам. Синтаксис подогнан под строение таблицы:

```
Движение.ВидДвижения[ВидСубконто] = Значение;
```

Можно записать иначе, например:

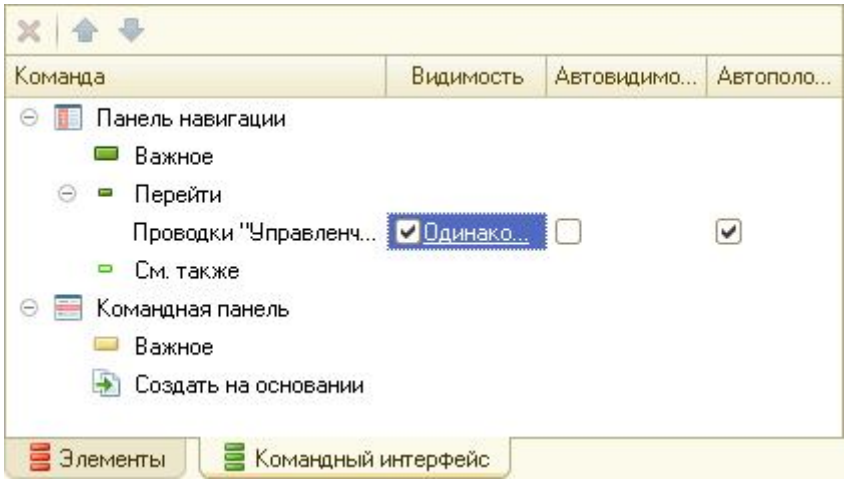
```
Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
```

как

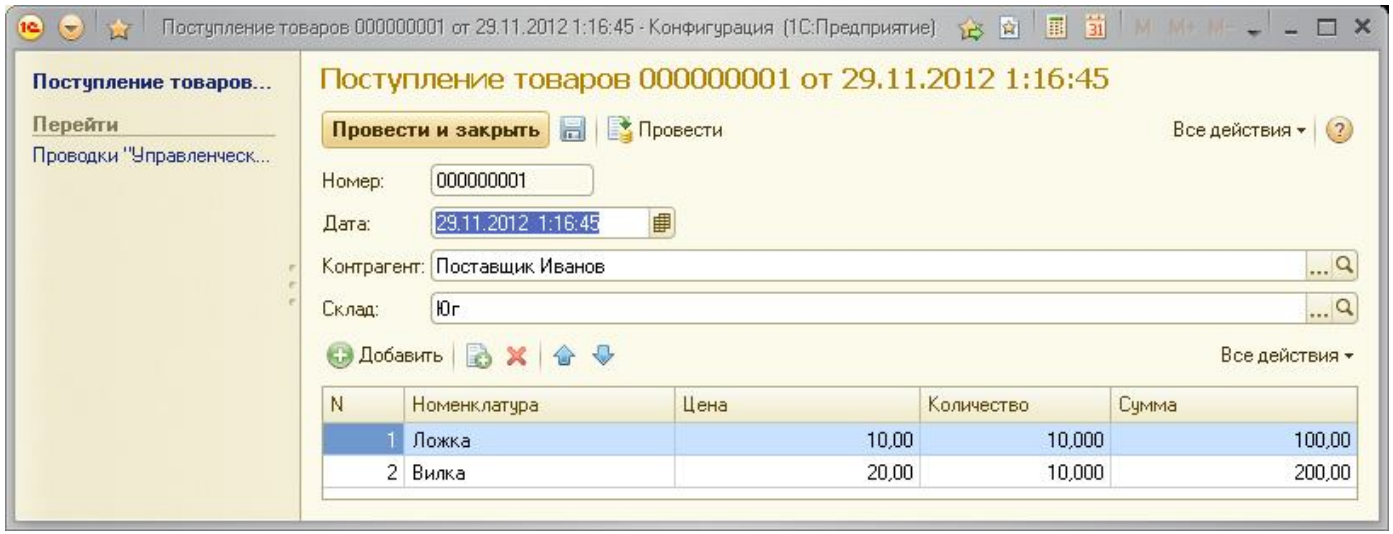
```
Движение.СубконтоКт.Контрагенты = Контрагент;
```

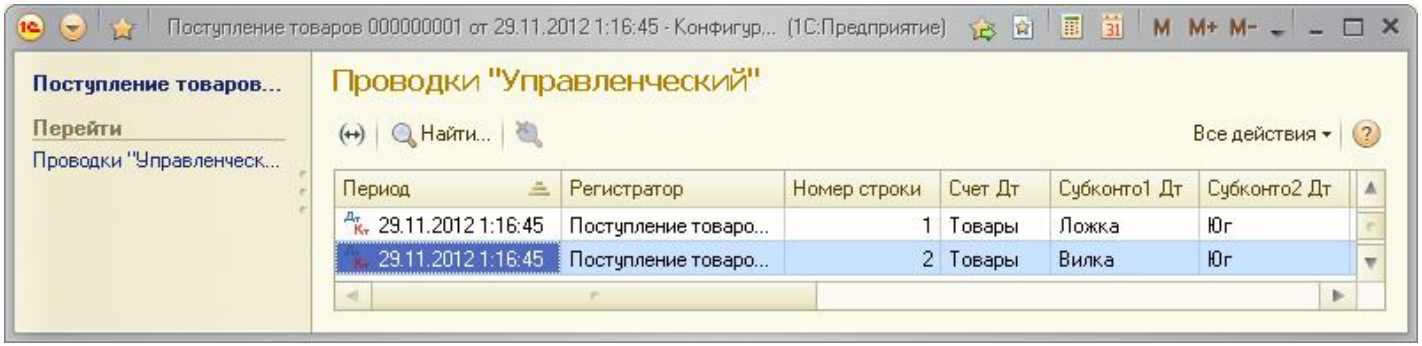
Но в этом случае система будет производить неявное преобразование типов, что влияет на производительность.

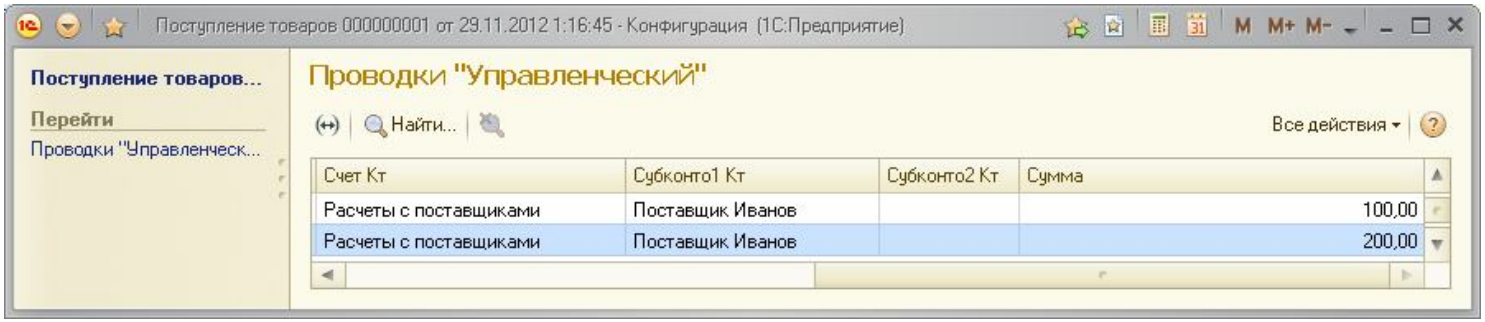
Создадим форму документа, чтобы в ней добавить переход к Регистру бухгалтерии «Управленческий»:



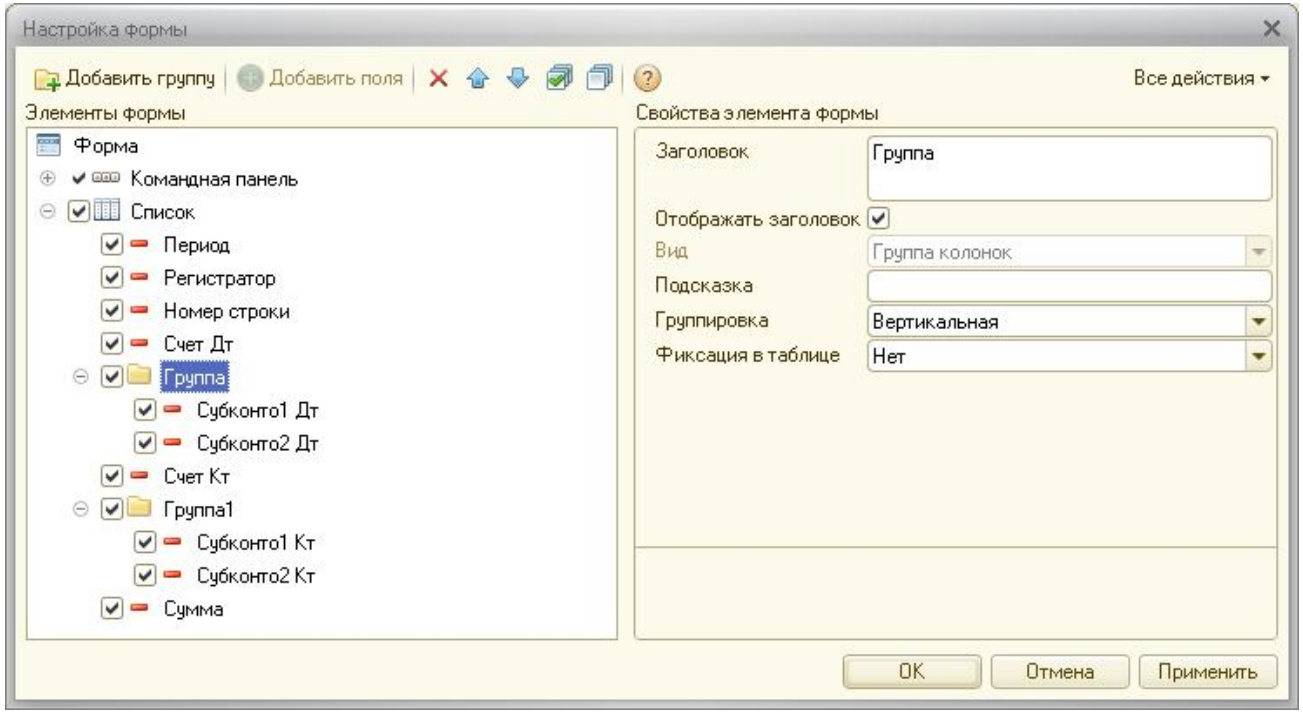
В режиме исполнения:
Добавим Приходную:

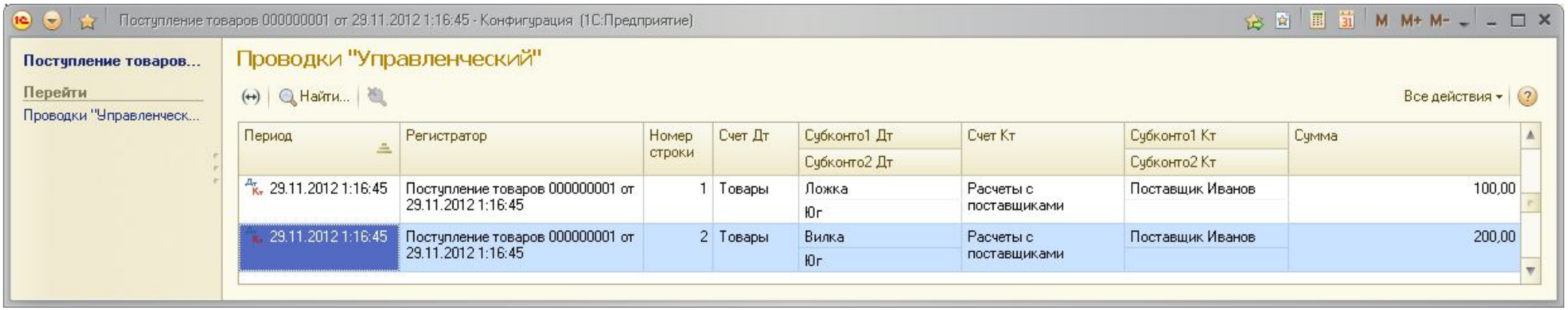






Можно настроить выведение полей: «Все действия – Изменить форму»:





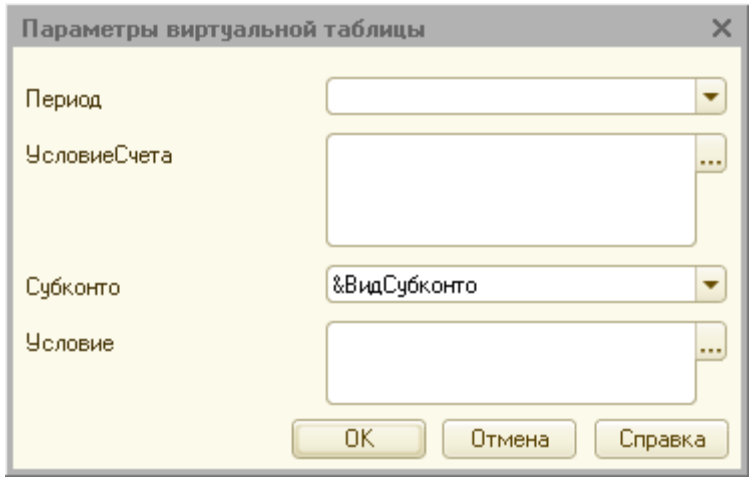
Создадим отчет «Остатки ТМЦ»

Свойство	Значение
Имя	ОстаткиТМЦ
Синоним	Остатки ТМЦ

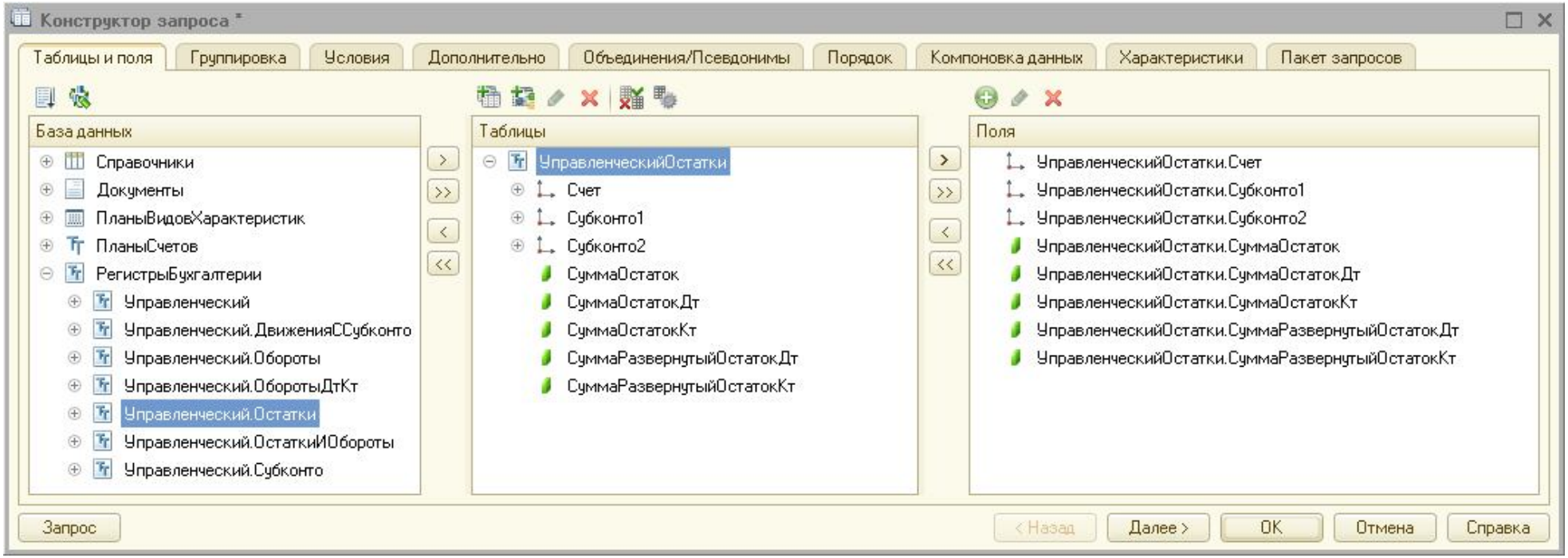
Откроем СКД, добавим «Набор данных-запрос» и откроем «Конструктор запроса».
Параметры таблицы «Остатки» регистра бухгалтерии:

- Период – дата для расчета остатков.
- Условие счета – наложение условий на счет или его свойства. Если ограничение не поставить, то система сформирует запрос по всем счетам.
- Субконто – отбор по виду аналитического учета. Поле Субконто в таблице итогов имеет Составной тип значения, т.е. при обращении к любому свойству этого поля, в запросе будут использоваться все таблицы, которые входят в тип значения этого поля. Субконто позволяет ограничить обращение к таблице итогов Регистра бухгалтерии с отбором по типу значения Субконто. Можно передать элемент Плана видов характеристик.

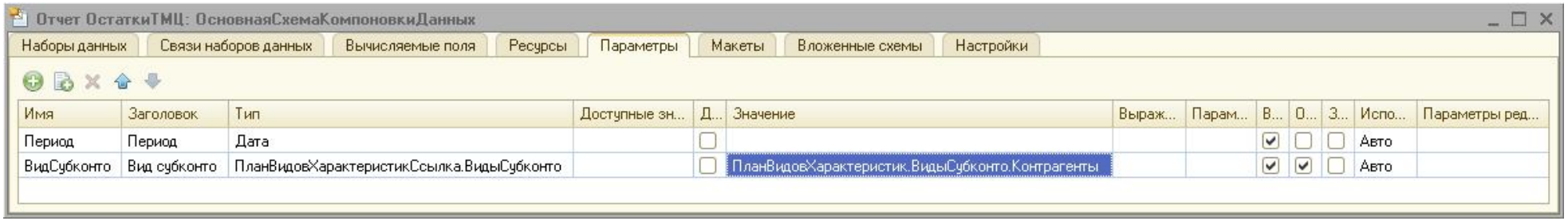
Передадим одно значение по Субконто:



Возьмем все поля:



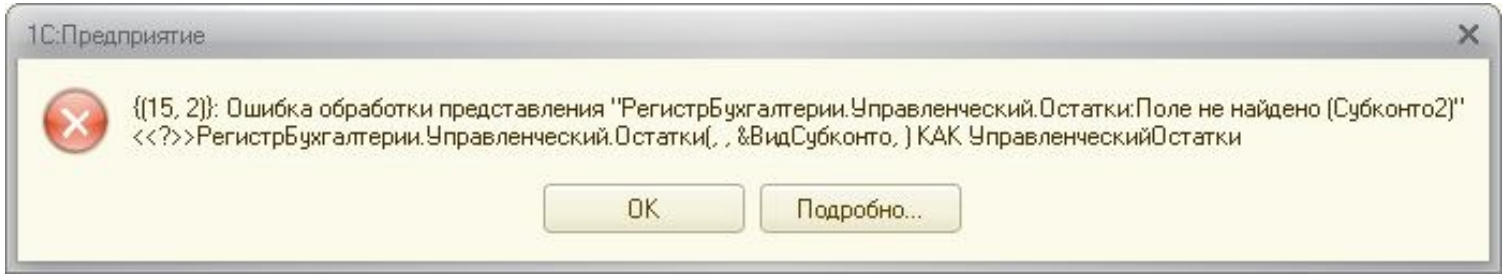
В СКД укажем параметр:



На закладке «Настройки» конструктором просто возьмем все поля.

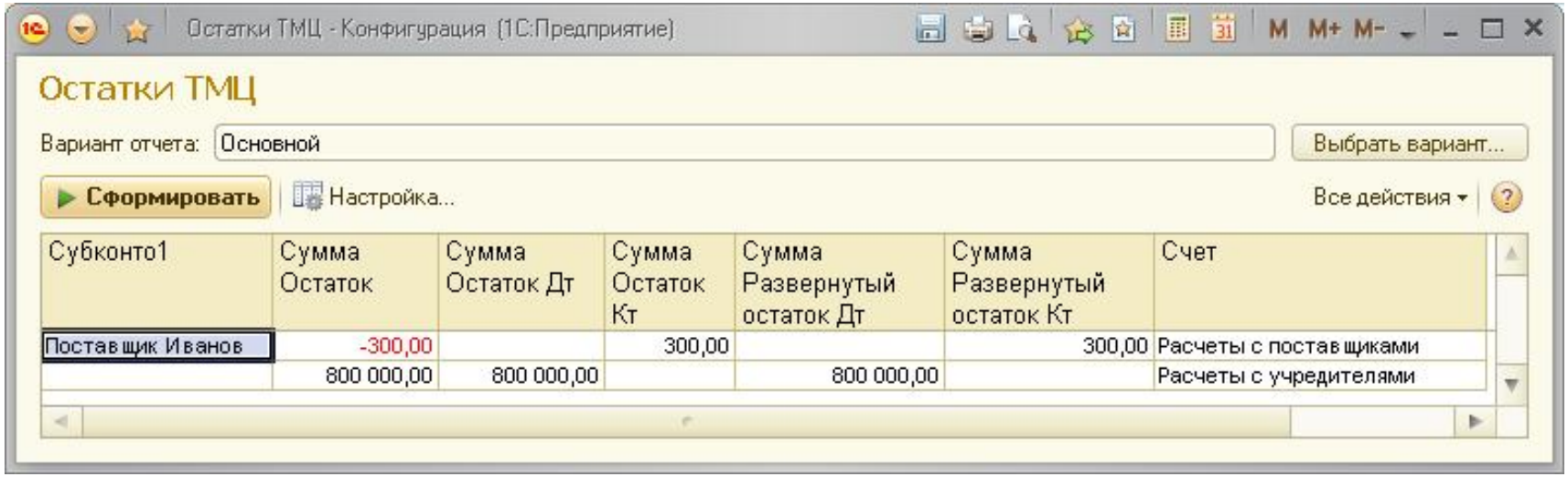
В режиме исполнения:

При попытке сформировать отчет, система выдаст ошибку:



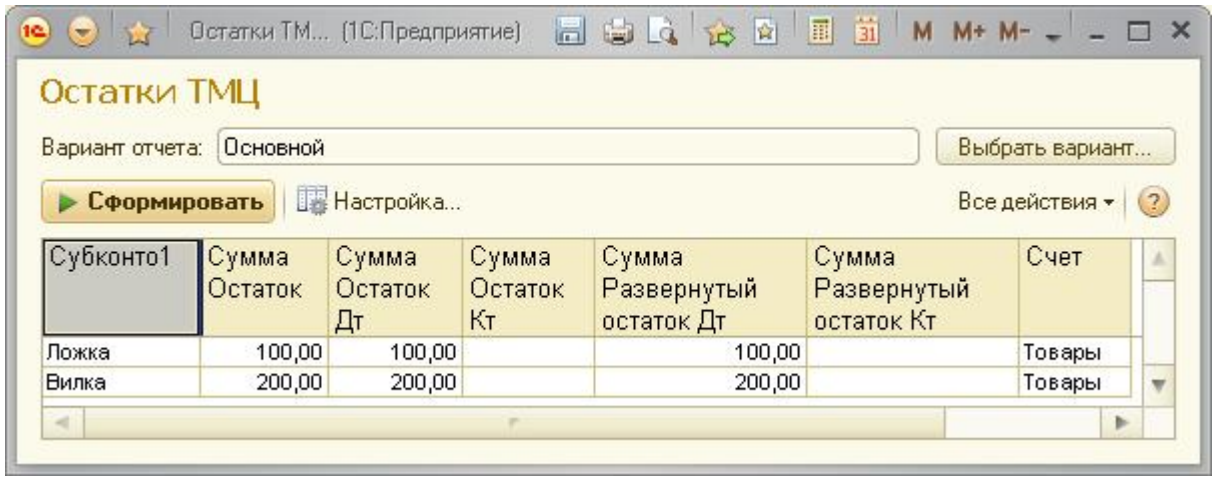
т.е. запрос пытается выбрать остатки из регистра, у которых в поле Субконто значение «&ВидСубконто» (Контрагенты). Чтобы отчет сформировался с таким условием, надо убрать из выбираемых полей «Субконто2».

В режиме исполнения:



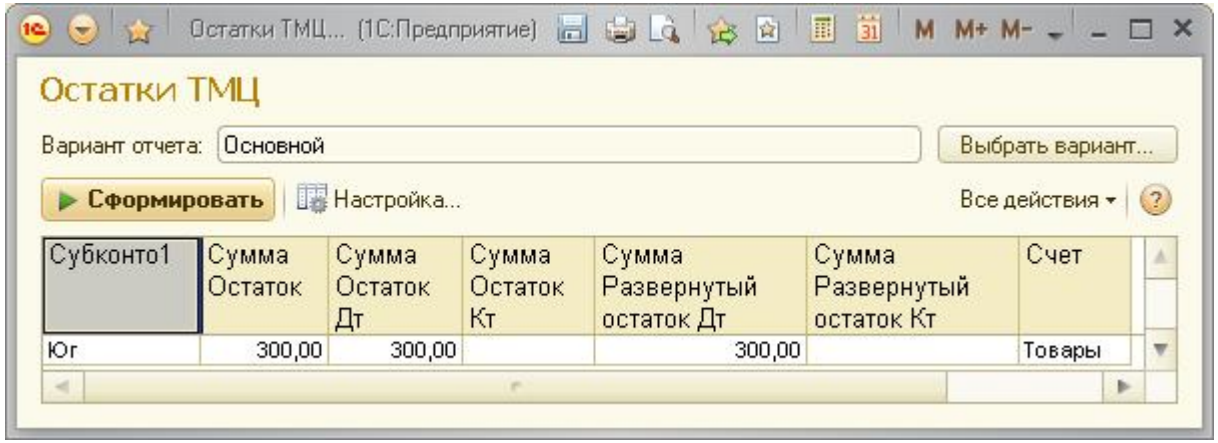
Субконто1	Сумма Остаток	Сумма Остаток Дт	Сумма Остаток Кт	Сумма Развернутый остаток Дт	Сумма Развернутый остаток Кт	Счет
Поставщик Иванов	-300,00		300,00		300,00	Расчеты с поставщиками
	800 000,00	800 000,00		800 000,00		Расчеты с учредителями

Аналогично можно получить остатки по Номенклатуре, поменяв Значение параметра «ВидСубконто» (Конфигуратор – СКД – закладка Параметры):



Субконто1	Сумма Остаток	Сумма Остаток Дт	Сумма Остаток Кт	Сумма Развернутый остаток Дт	Сумма Развернутый остаток Кт	Счет
Ложка	100,00	100,00		100,00		Товары
Вилка	200,00	200,00		200,00		Товары

Аналогично получаются остатки по Складам. Хотя Склад является вторым видом аналитики в счете «Товары» (Субконто 2), но в запросе он будет Субконто 1:



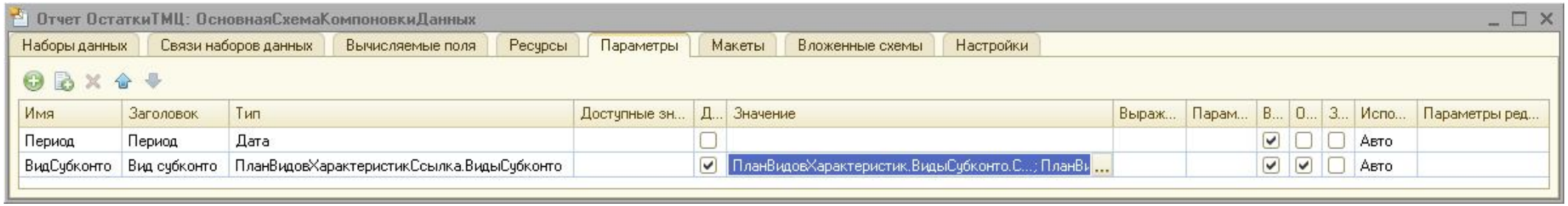
Субконто1	Сумма Остаток	Сумма Остаток Дт	Сумма Остаток Кт	Сумма Развернутый остаток Дт	Сумма Развернутый остаток Кт	Счет
Юг	300,00	300,00		300,00		Товары

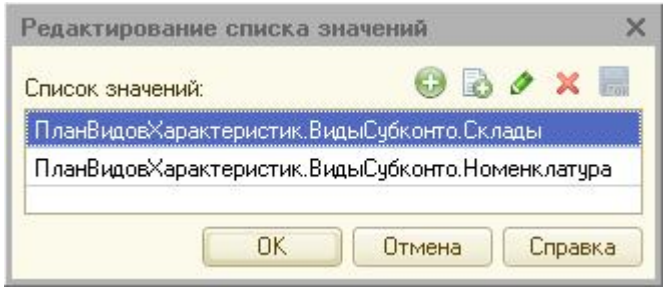
Т.е. параметр определяет отбор по Субконто и порядок Субконто в результате.

Но в данном параметре можно передавать множество значений в виде:

- Список значений.
- Массив.

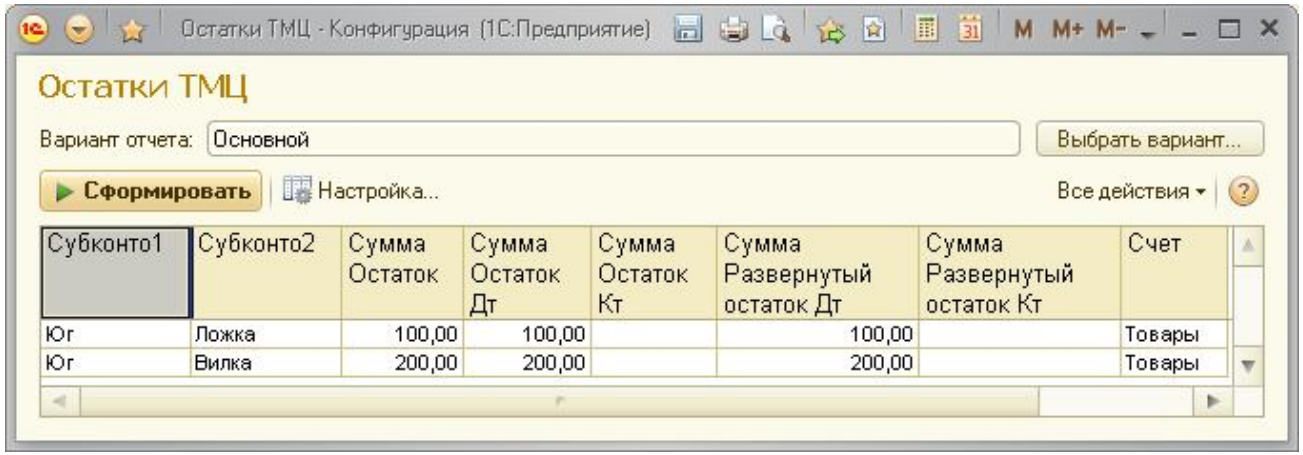
Вернем в запросе Субконто2. В параметрах:
«Доступен список значений – Истина»





В режиме исполнения:

Система покажет остатки по тем счетам, по которым ведется два вида аналитики – Склады и Номенклатура.



Субконто1	Субконто2	Сумма Остаток	Сумма Остаток Дт	Сумма Остаток Кт	Сумма Развернутый остаток Дт	Сумма Развернутый остаток Кт	Счет
Юг	Ложка	100,00	100,00		100,00		Товары
Юг	Вилка	200,00	200,00		200,00		Товары

Какой вид будет у Субконто1 и Субконто2 задается в параметрах (в конфигураторе), т.е. если вид первого параметра «Склады», то и вид Субконто1 будет «Склады».

Еще один параметр таблицы «Остатки» регистра бухгалтерии:

- Условие – наложение условий на сами Субконто (например, ложки, вилки, склады и т.д.). Порядок Субконто здесь также зависит от порядка, указанного в параметрах.

Документ «Расходная».
Проводки:

Счет Дт	Счет Кт	Сумма
Покупатели	Товары	Себестоимость Выручка

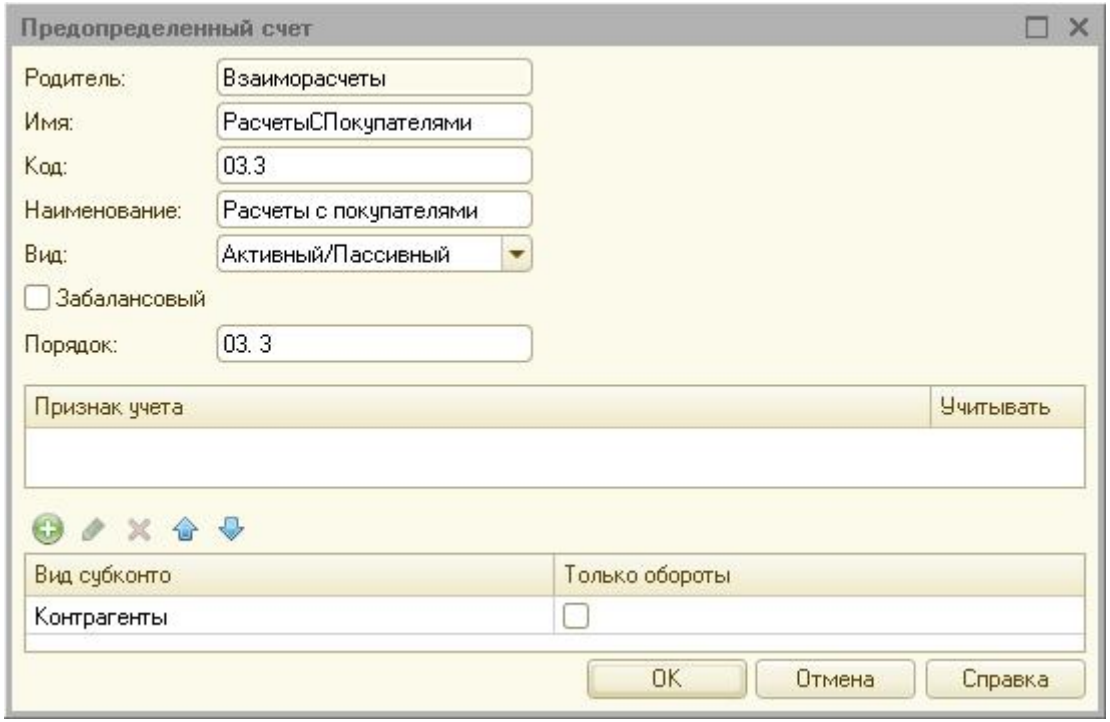
Если ставить «Выручка», то, например, если на складе есть вилок на 100 рублей, а продали их за 500 рублей, то на складе останется вилок на -400 рублей, что неверно. Товар нельзя уменьшить в продажных ценах, а только по себестоимости. Но при этом получается, что покупатели берут товар за такую же сумму, которую мы заплатили за товар, что тоже неверно.

Разделим проводку на две:

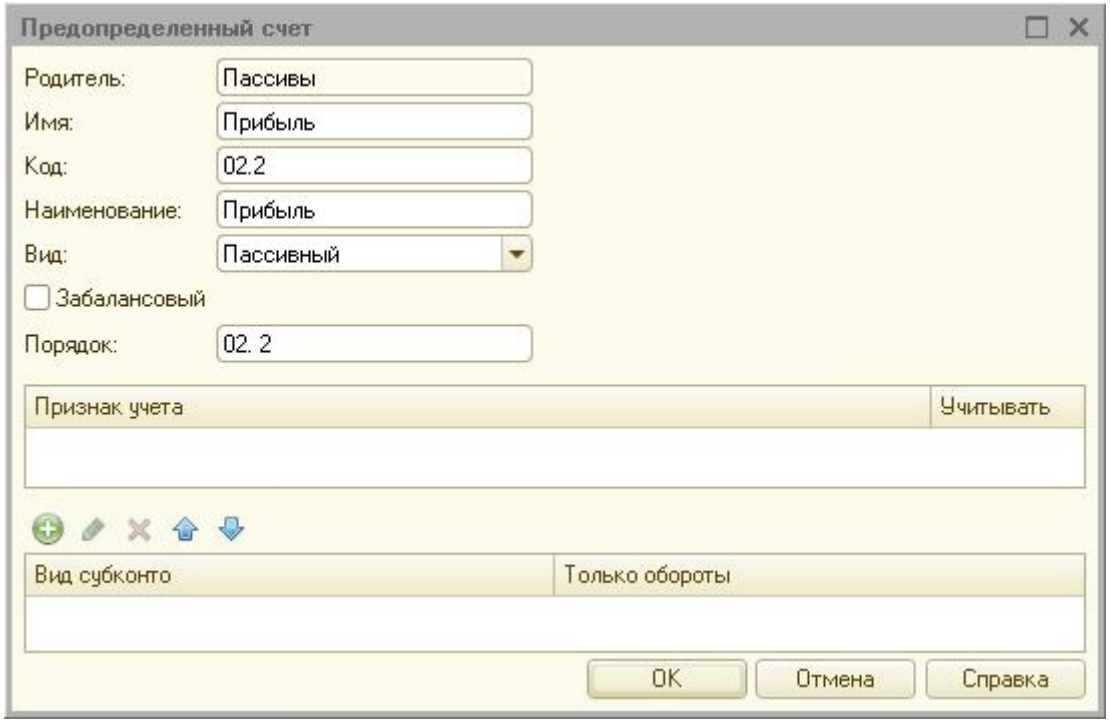
Счет Дт	Счет Кт	Сумма	Содержание
Прибыль	Товары	Себестоимость	Списание себестоимости
Покупатели	Прибыль	Выручка	Учет выручки

Будет добавлен новый счет, накапливающий результат операции – «Прибыль». В итоге на счете «Прибыль» отобразится Маржа - разница между Выручкой и Себестоимостью.

В группу «Взаиморасчеты» добавим предопределенный счет «Расчеты с покупателями»:



В группу «Пассивы» добавим предопределенный счет «Прибыль»:

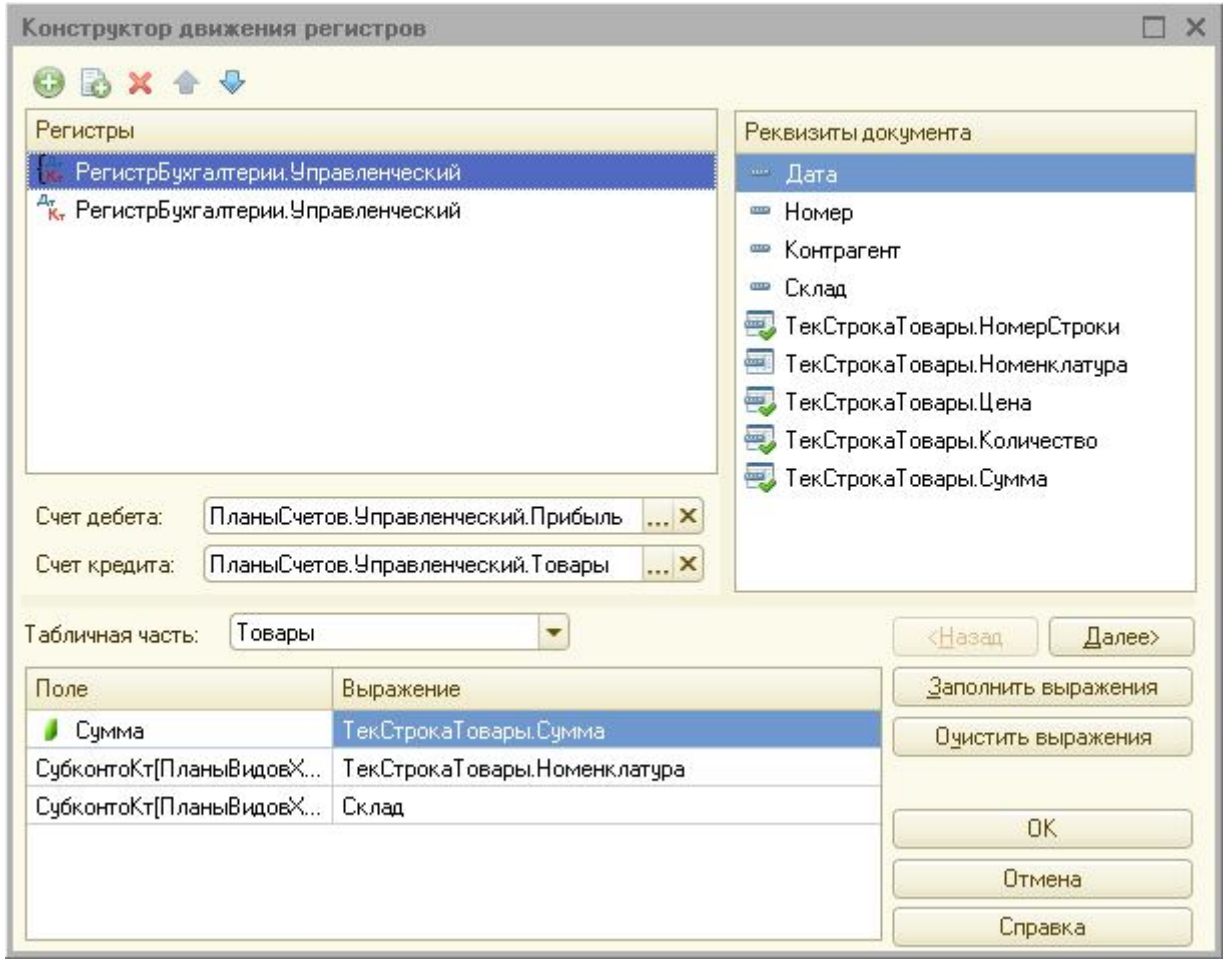


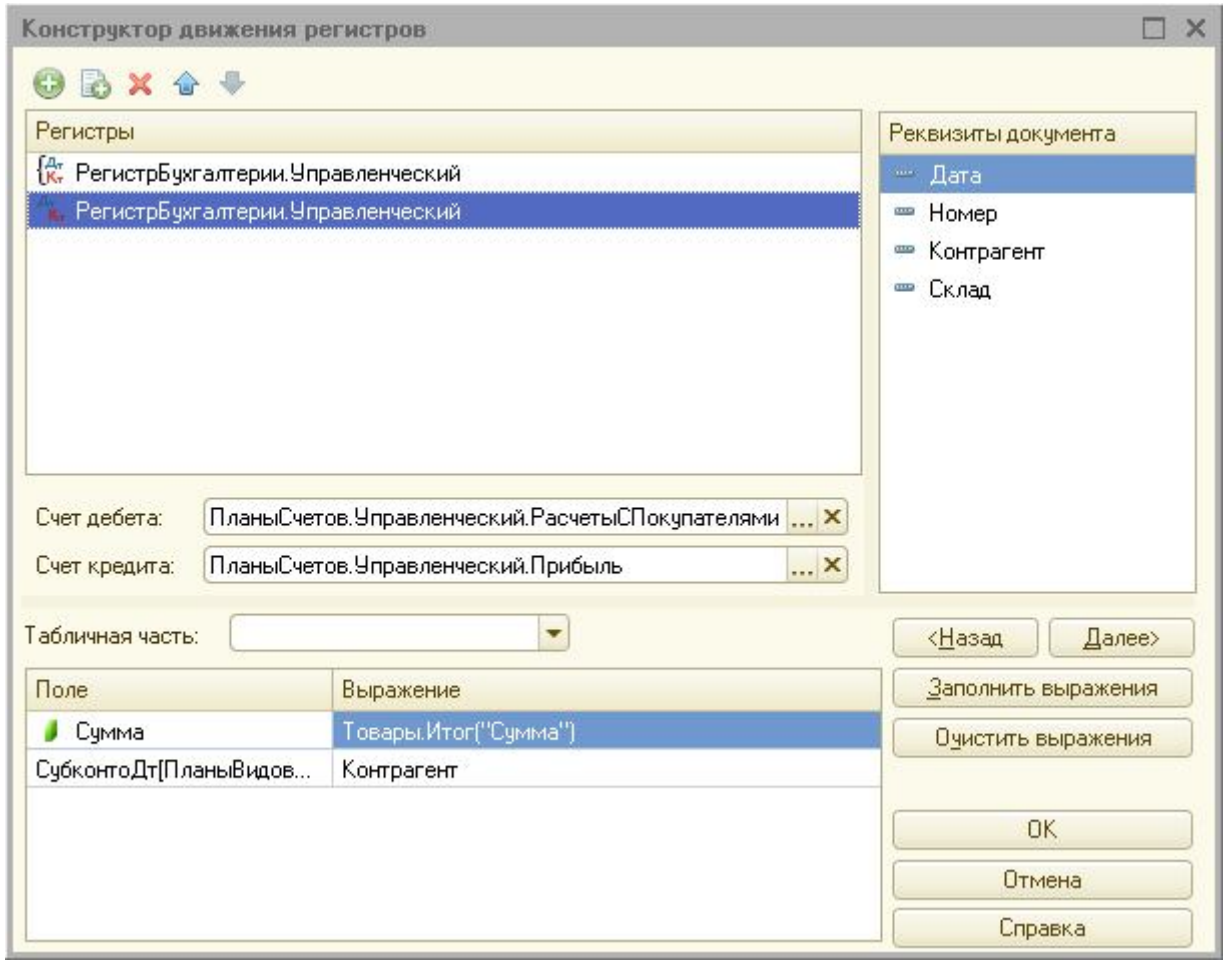
Создадим документ «Расходная»:

Свойство	Значение
Имя	Расходная
Синоним	Расходная
Представление объекта	Продажа товаров
Представление списка	Расходные

Данные Расходной аналогичны Приходной (реквизиты и табличная часть).

Движения:


Конструктор движения регистров
Регистры
РегистрБухгалтерии.Управленческий
РегистрБухгалтерии.Управленческий
Реквизиты документа
Дата
Номер
Контрагент
Склад
ТекСтрокаТовары.НомерСтроки
ТекСтрокаТовары.Номенклатура
ТекСтрокаТовары.Цена
ТекСтрокаТовары.Количество
ТекСтрокаТовары.Сумма
Счет дебета:
ПланыСчетов.Управленческий.Прибыль
Счет кредита:
ПланыСчетов.Управленческий.Товары
Табличная часть:
Товары
<Назад
Далее>
Поле
Выражение
Сумма
ТекСтрокаТовары.Сумма
СубконтоКт[ПланыВидовХ...
ТекСтрокаТовары.Номенклатура
СубконтоКт[ПланыВидовХ...
Склад
Заполнить выражения
Очистить выражения
OK
Отмена
Справка



Итого код:

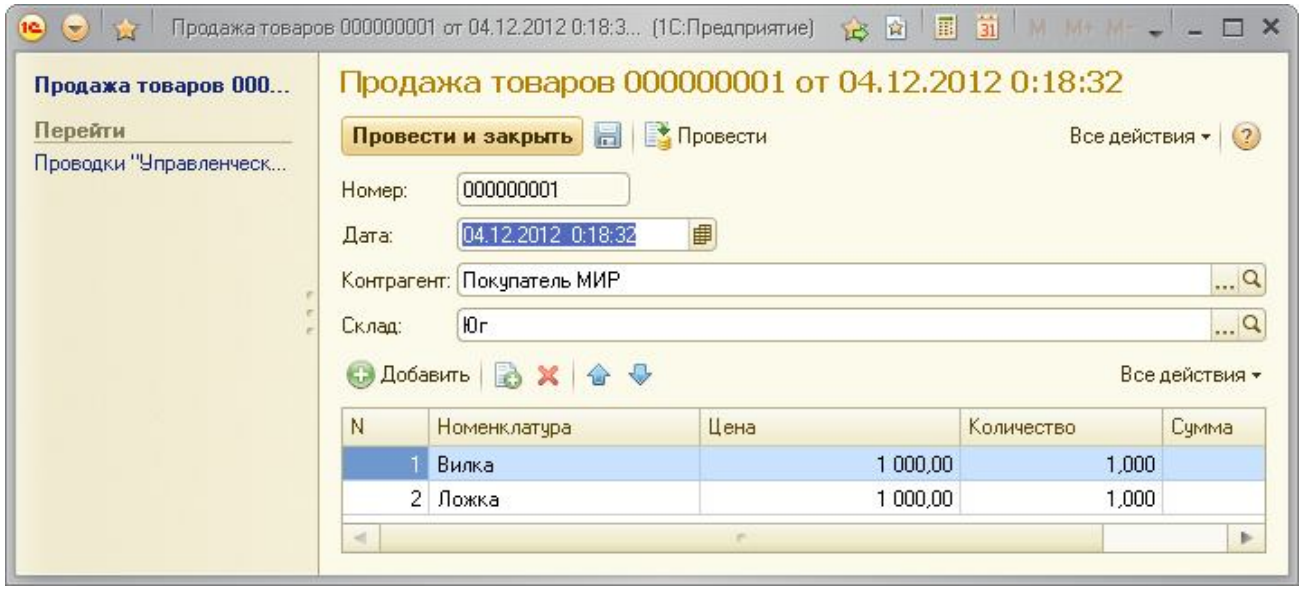
```
Процедура ОбработкаПроведения(Отказ, Режим)

     // регистр Управленческий
     Движения.Управленческий.Записывать = Истина;
     Для Каждого ТекСтрокаТовары Из Товары Цикл
          Движение = Движения.Управленческий.Добавить();
          Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
          Движение.СчетКт = ПланыСчетов.Управленческий.Товары;
          Движение.Период = Дата;
```

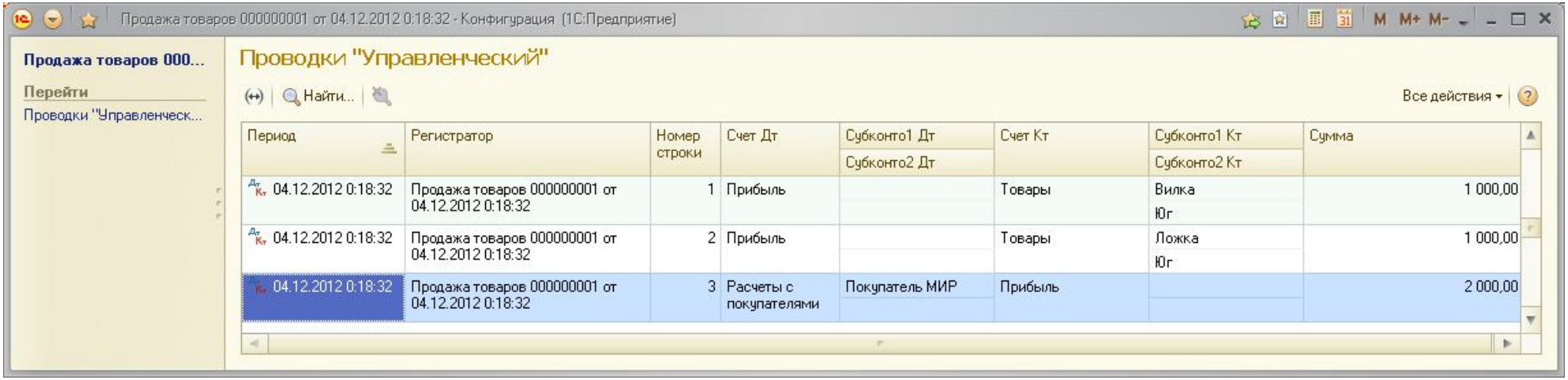
```
        Движение.Сумма = ТекСтрокаТовары.Сумма; // неверно
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = ТекСтрокаТовары.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
    КонецЦикла;

    // регистр Управленческий
    Движения.Управленческий.Записывать = Истина;
    Движение = Движения.Управленческий.Добавить();
    Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
    Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
    Движение.Период = Дата;
    Движение.Сумма = Товары.Итог("Сумма");
    Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;

КонецПроцедуры
```

Сейчас в примере нет аналитики для правильного расчета себестоимости («Количество»), поэтому сумма в первой проводке неправильная.

В режиме исполнения:

Создадим документ «Расходная», проведем его и проверим сформированные проводки:





Видно, что суммы первых двух проводок неверные, третья проводка правильная.

Занятие 2.21

Сделаем обусловленное проведение документа с контролем остатков и расчетом себестоимости.

Счет Дт	Счет Кт	Сумма	Количество
Товары	Поставщики	10000	20

Накапливать Количество по обоим счетам не правильно. При учете Количества по Поставщикам, этот ресурс никогда не будет сводиться в ноль. Надо реализовать количественный учет только по товарам.

В Плане счетов создадим Признак учета «Количественный» и проставим его у нужных счетов:

Свойство	Значение
Имя	Количественный
Синоним	Количественный
Тип	Булево

Сделаем «Количественный - Истина» только у счета «Товары».

Создадим ресурс «Количество» в Регистре бухгалтерии:

Свойство	Значение
Имя	Количество
Синоним	Количество
Тип	Число 12, 3
Признак учета	Количественный
Балансовый	Ложь

«Признак учета – Количественный» - связь ресурса с признаком учета. Если у счета стоит Признак учета «Количественный», то по нему накапливается количество и наоборот.

Свойство «Балансовый» - в проводке может быть один счет Количественный, а другой нет. Если при этом по ресурсу Количество будет накапливаться баланс (у одного из счетов проводки свойство «Балансовый - Истина»), то изменение значения ресурса будет зарегистрировано по обоим счетам.

Если Ресурс балансовый, то при формировании движений будет одно поле:

- Количество.

А если Ресурс не балансовый, то полей будет два:

- КоличествоДт.
- КоличествоКт.

Т.е. нам надо, чтобы количество учитывалось только по Дт:

Счет Дт	Счет Кт	Сумма	КоличествоДт	КоличествоКт
Товары	Поставщики	10000	20	0

При добавлении Ресурса в Регистр бухгалтерии надо ответить на вопросы:

- Будет ли ресурс накапливаться по всем счетам? Если нет, то нужен Признак учета по счету, который определит, по каким счетам должно накапливаться значение ресурса.
- В проводке могут быть счета с разными значениями Признака учета? Если да, то «Балансовый - Ложь».

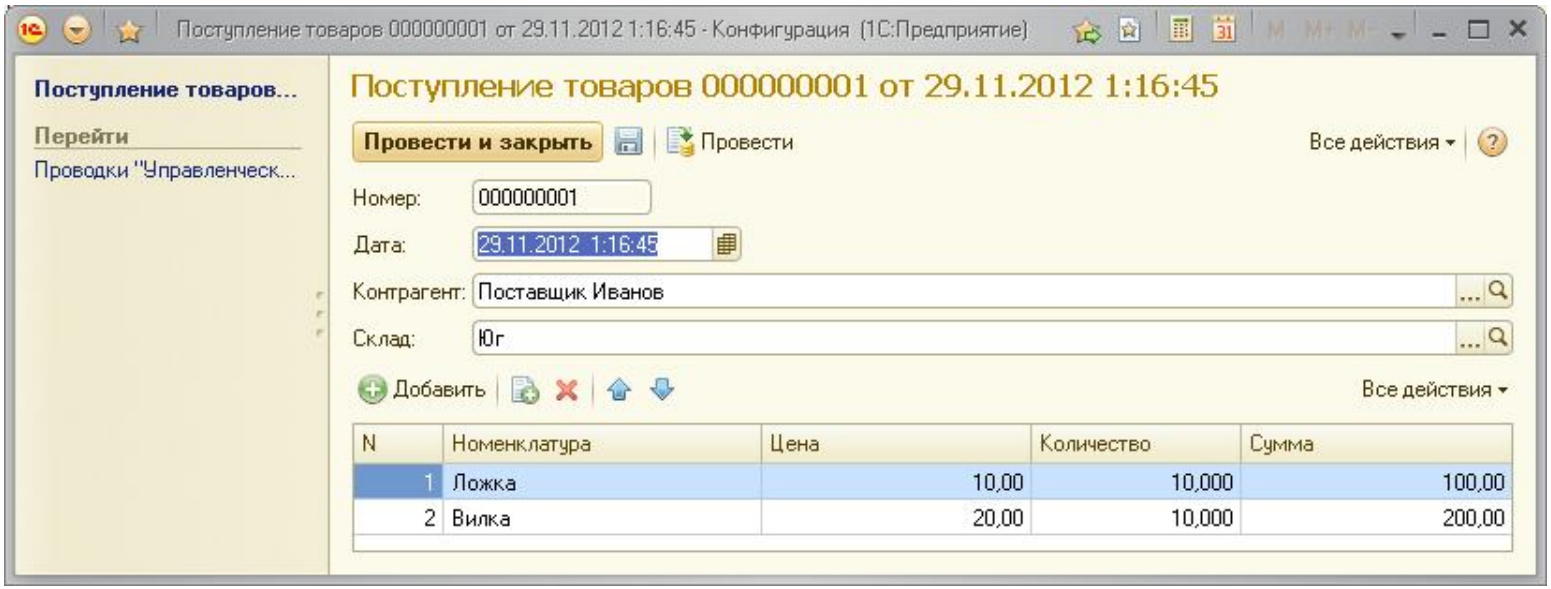
Поправим документ «Приходная»:

Модуль объекта:

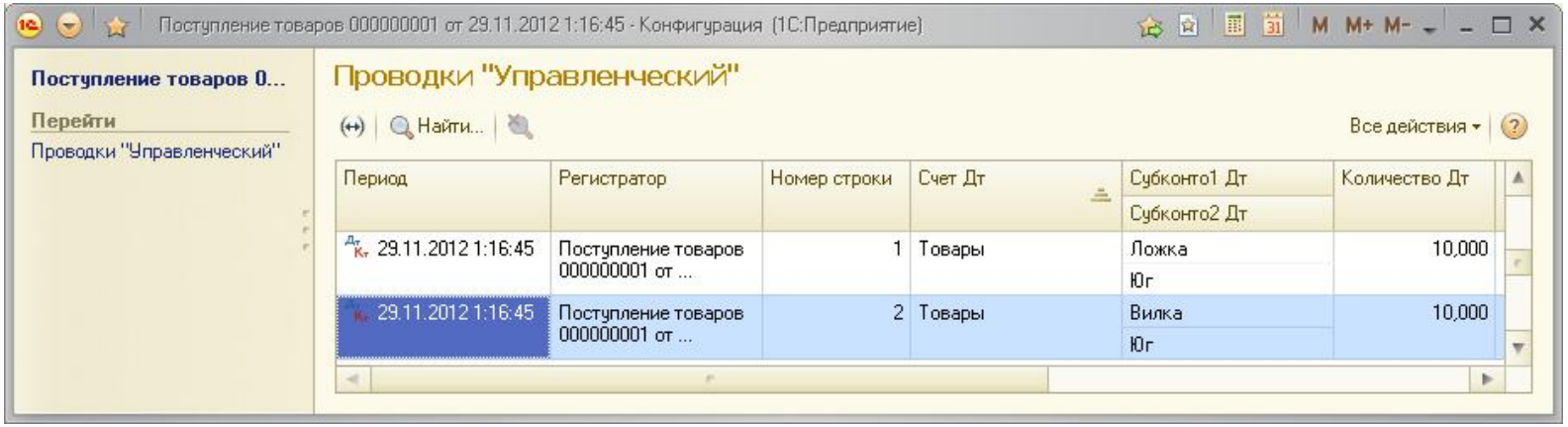
Обращение к Ресурсам в Регистре бухгалтерии возможно только после определения счетов, по которым формируются движения. Счета определяют Признаки учета, а в зависимости от значения Признака учета система разрешит или запретит формировать записи.

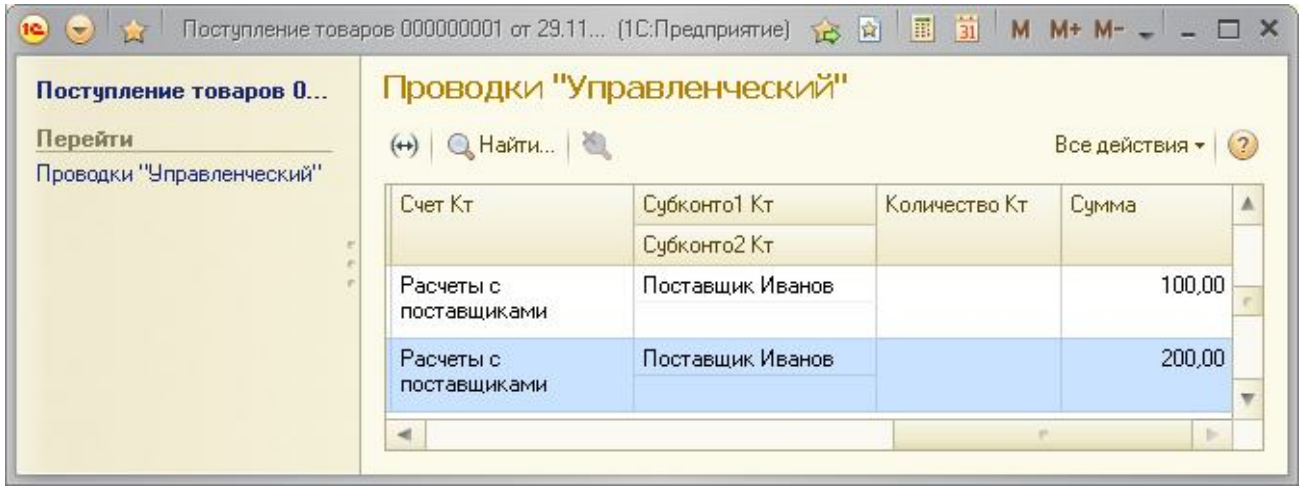
```
Процедура ОбработкаПроведения(Отказ, Режим)

    // регистр Управленческий
    Движения.Управленческий.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.Товары;
        Движение.СчетКт = ПланыСчетов.Управленческий.РасчетыСПоставщиками;
        Движение.Период = Дата;
        Движение.Сумма = ТекСтрокаТовары.Сумма;
        Движение.КоличествоДт = ТекСтрокаТовары.Количество;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = ТекСтрокаТовары.Номенклатура;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
    КонецЦикла;

КонецПроцедуры
```

В режиме исполнения:
Перепроведем документ «Приходная»



и проверим результат:



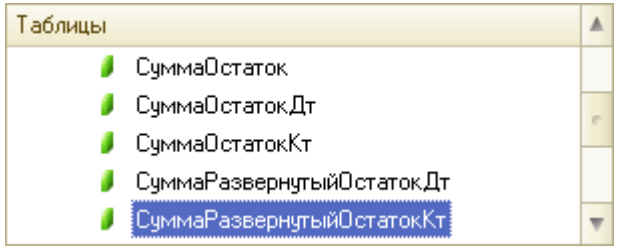


Доделаем отчет «Остатки ТМЦ»:

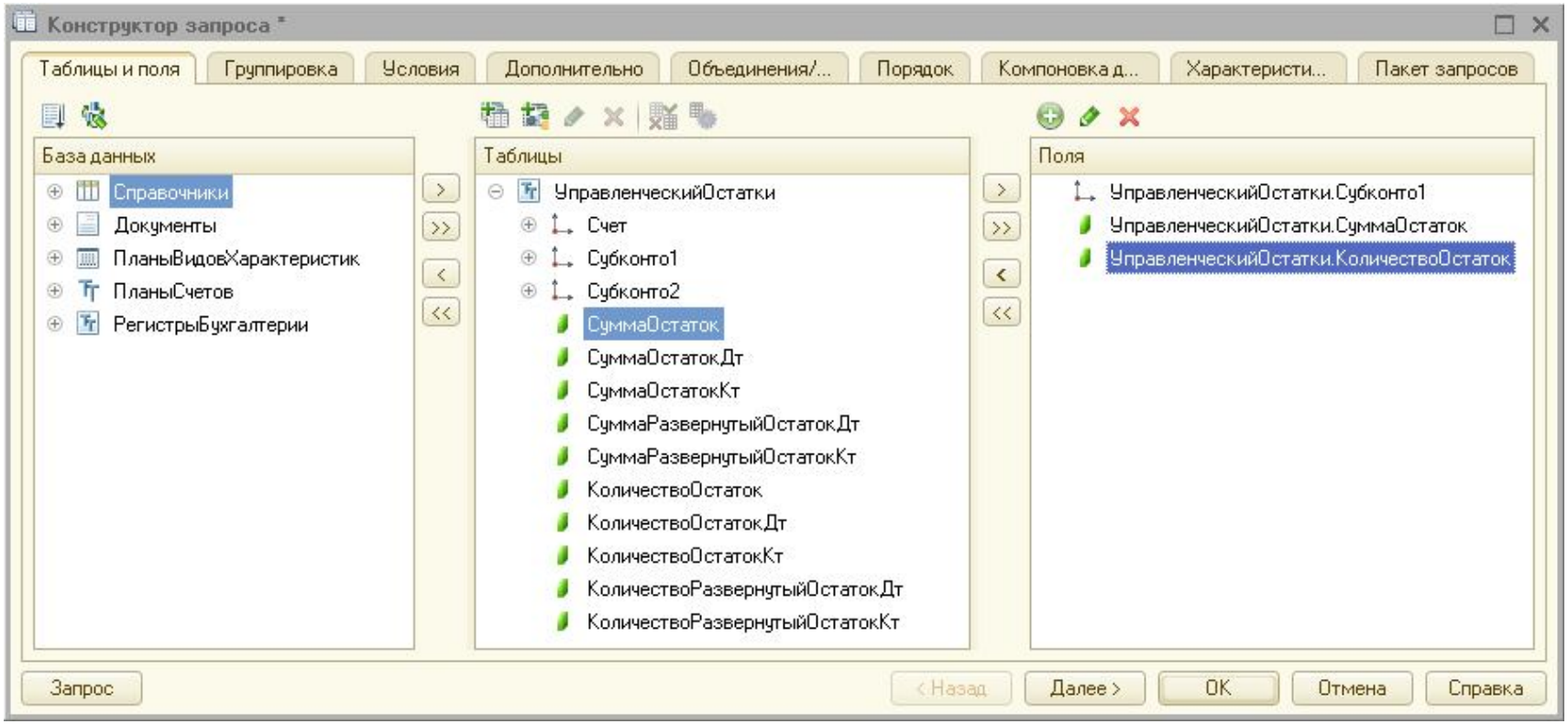
Сделаем отчет универсальным, чтобы он отображал информацию по всем счетам (которые существуют сейчас и которые будут добавлены позже), которые содержат информацию по ТМЦ. Признаки счета:

- Субконто – минимум, должна вестись аналитика по Номенклатуре (Вид субконто).
- Признак «Количественный».
- Вид счета «Активный» - только Активный счет содержит наши средства, средства на Пассивном счете (займы, отданные на переработку) уже не наши

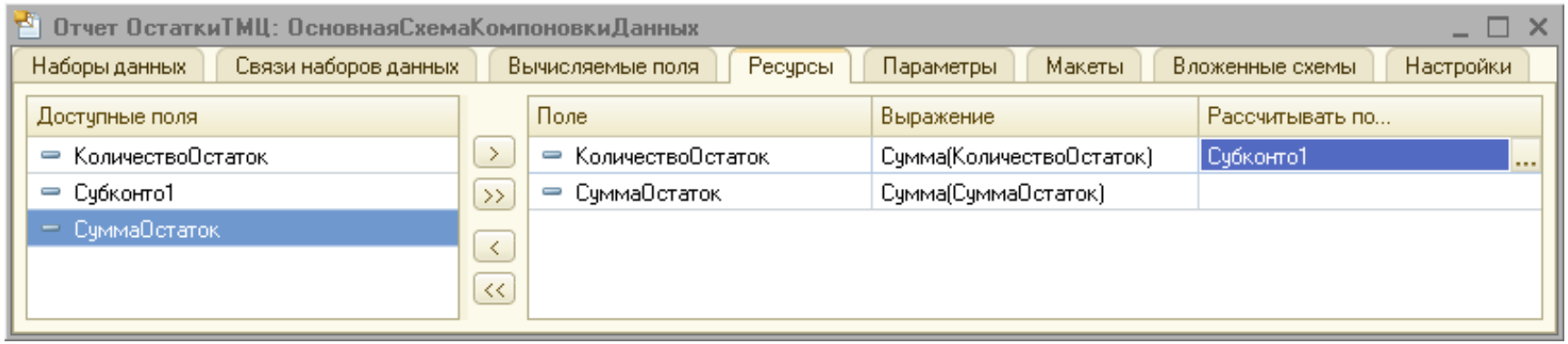
В запросе:
Какую сумму брать:

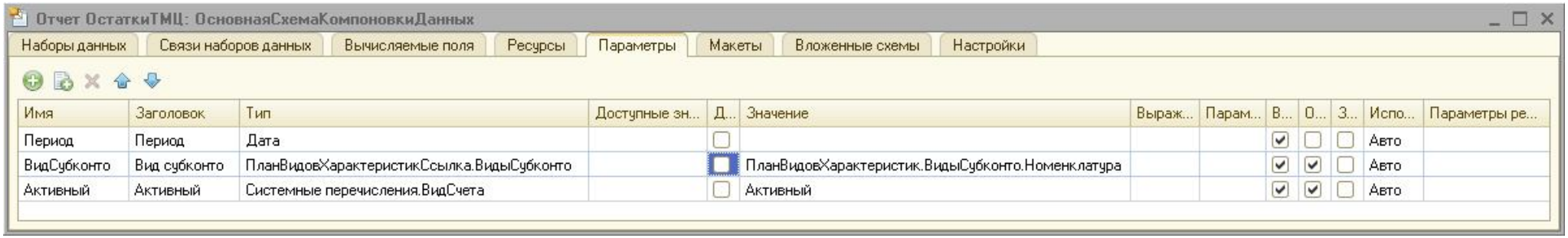


Т.к. счет Активный, то СуммаОстаток будет только в Дт. СуммаРазвернутыйОстаток используется только в итогах, в отчете нам итоги не нужны, кроме того, в итогах тоже не будет разных (Активных/Пассивных) счетов, т.е. РСуммаРазвернутыйОстаток не осличается от СуммаОстаток.
В поле БД остаток в суммовом выражении хранится в поле СуммаОстаток и, в зависимости от счета, система возвращает Дт или Кт. В отчете СуммаОстатокДт будет всегда равен СуммаОстоток. Если выбрать СуммаОстатокДт, то система будет анализировать Тип счета, что потребует дополнительных ресурсов, поэтому выберем просто СуммаОстаток.
Если бы счета были разного типа, то надо было бы выбирать счета по Дт и по Кт.

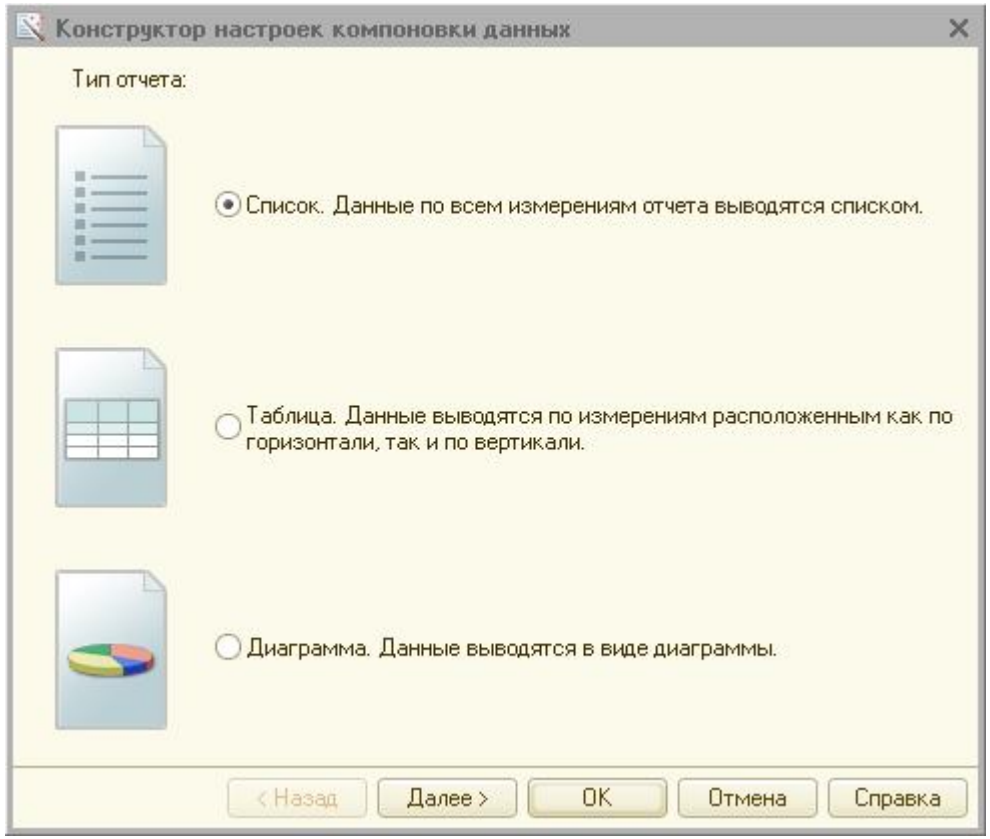

Конструктор запроса *
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/...
Порядок
Компоновка д...
Характеристи...
Пакет запросов
База данных
Справочники
Документы
ПланыВидовХарактеристик
ПланыСчетов
РегистрыБухгалтерии
Таблицы
УправленческийОстатки
Счет
Субконто1
Субконто2
СуммаОстаток
СуммаОстатокДт
СуммаОстатокКт
СуммаРазвернутыйОстатокДт
СуммаРазвернутыйОстатокКт
КоличествоОстаток
КоличествоОстатокДт
КоличествоОстатокКт
КоличествоРазвернутыйОстатокДт
КоличествоРазвернутыйОстатокКт
Поля
УправленческийОстатки.Субконто1
УправленческийОстатки.СуммаОстаток
УправленческийОстатки.КоличествоОстаток
Запрос
< Назад
Далее >
OK
Отмена
Справка

В СКД:

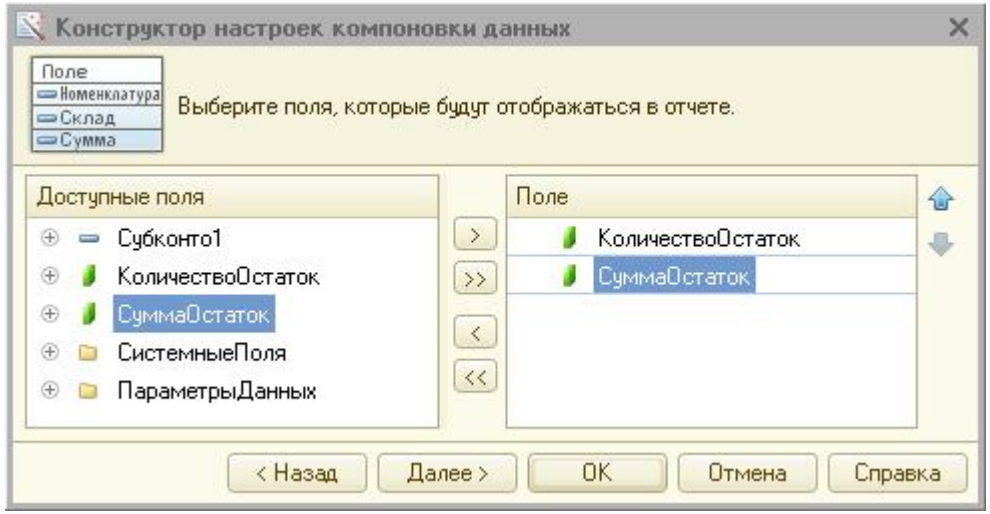

Отчет ОстаткиТМЦ: ОсновнаяСхемаКомпоновкиДанных
Наборы данных
Связи наборов данных
Вычисляемые поля
Ресурсы
Параметры
Макеты
Вложенные схемы
Настройки
Доступные поля
КоличествоОстаток
Субконто1
СуммаОстаток
Поле
Выражение
Рассчитывать по...
КоличествоОстаток
Сумма(КоличествоОстаток)
Субконто1
СуммаОстаток
Сумма(СуммаОстаток)



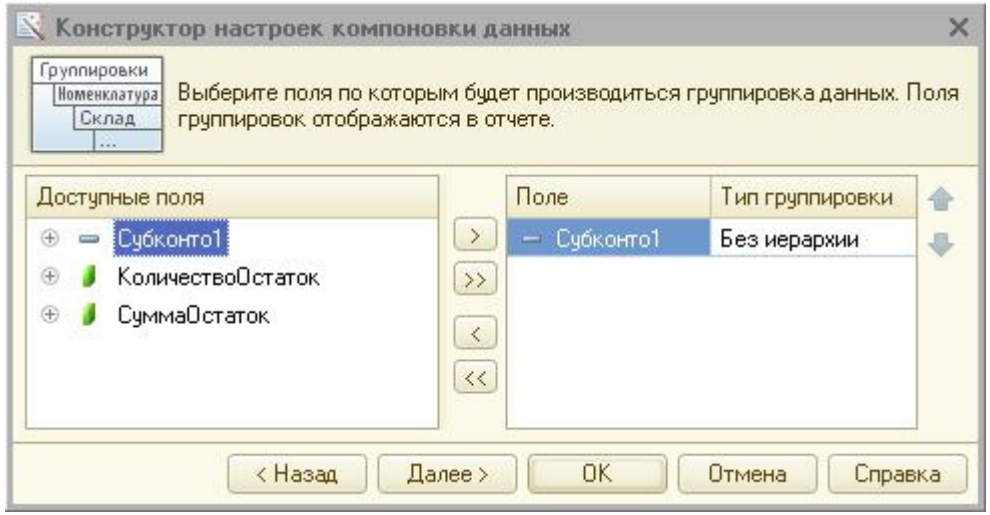
Закладка «Настройки» - установим настройки по умолчанию:



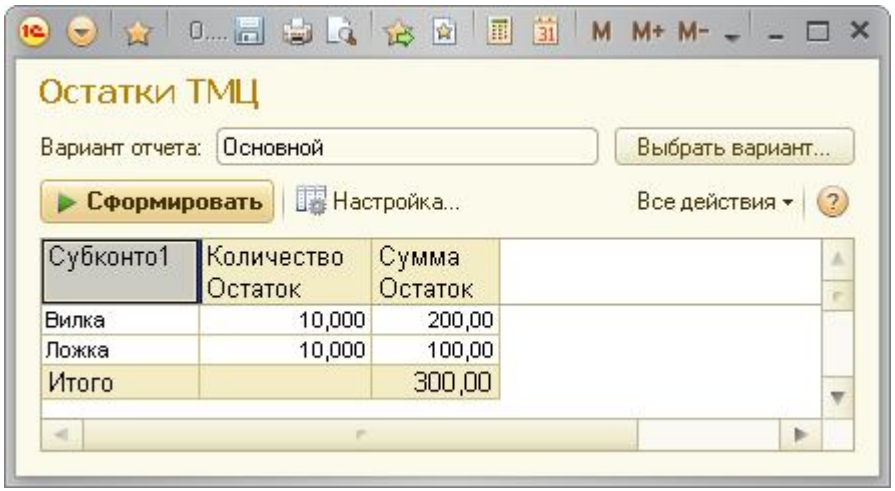
Выведем поля:



В разрезе:



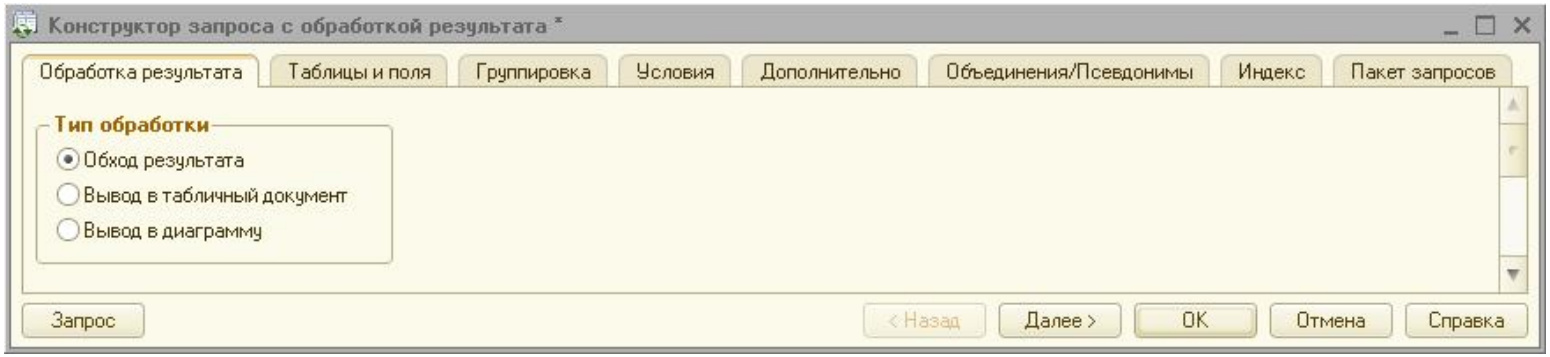
В режиме исполнения:
Пометим на удаление документ «Расходная» и сформируем отчет:

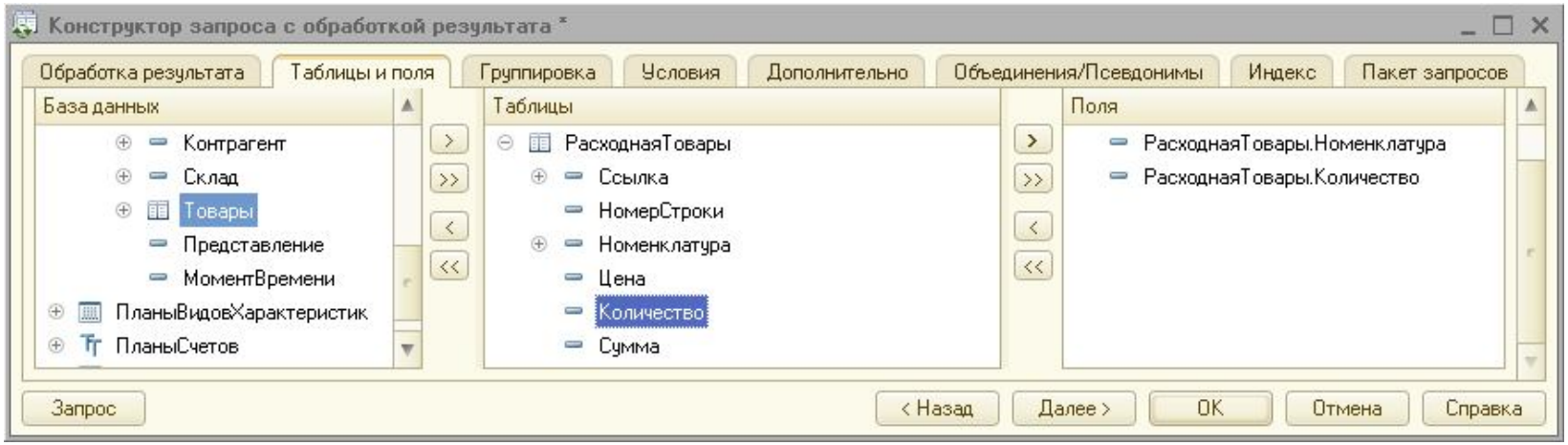


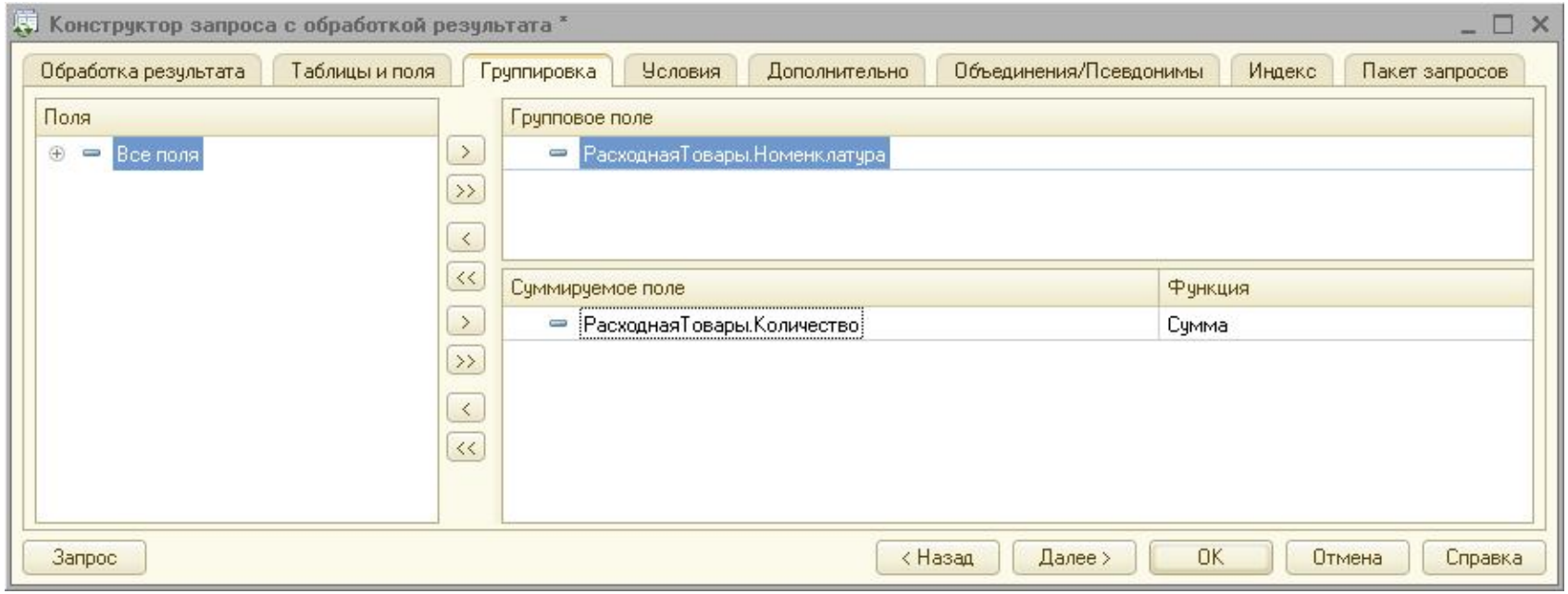
Доработаем документ «Расходная» - будем проверять, хватает товара или нет, если хватает, то будем рассчитывать себестоимость его списания по методу средней. Таблица, которую надо получить при проведении документа:

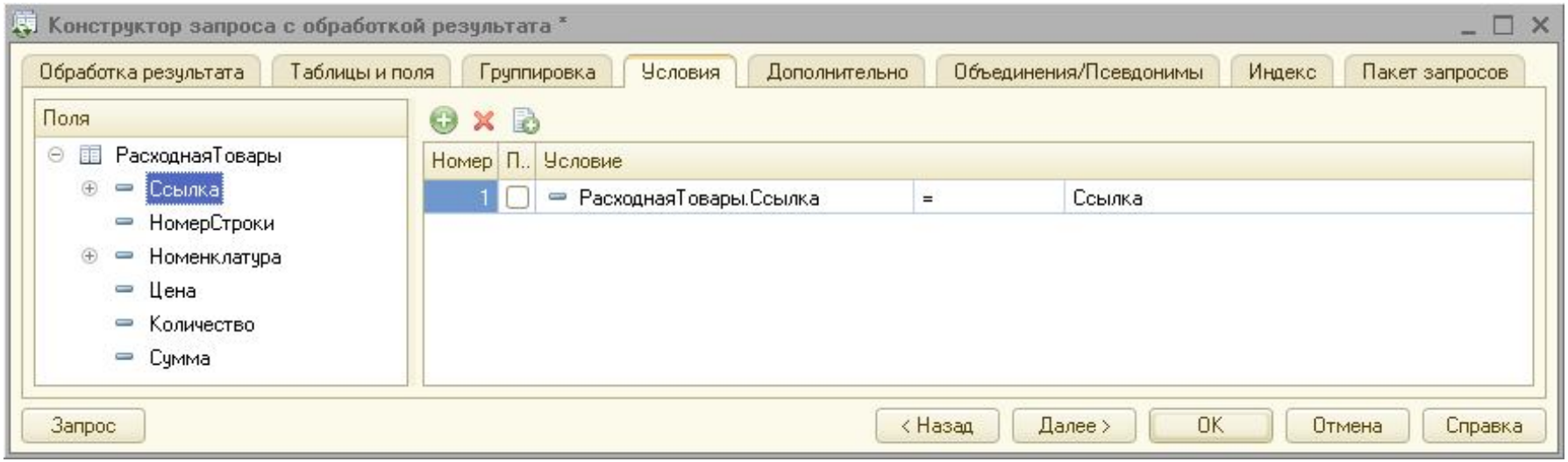
Номенклатура	Количество	КоличествоОстаток	СуммаОстаток

«Модуль объекта – Процедура ОбработкаПроведения()» - запустим «Конструктор запроса с обработкой результата:

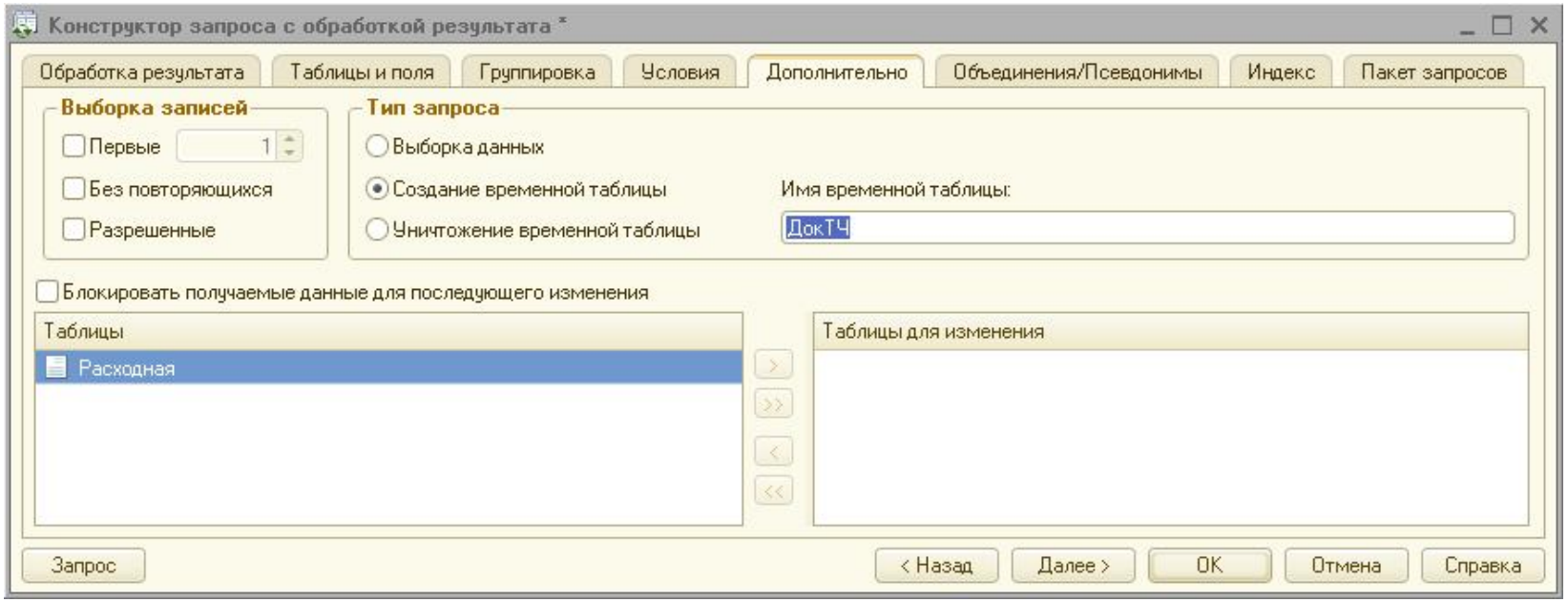



Конструктор запроса с обработкой результата *
Обработка результата
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Индекс
Пакет запросов
База данных
Контрагент
Склад
Товары
Представление
МоментВремени
ПланыВидовХарактеристик
ПланыСчетов
Таблицы
РасходнаяТовары
Ссылка
НомерСтроки
Номенклатура
Цена
Количество
Сумма
Поля
РасходнаяТовары.Номенклатура
РасходнаяТовары.Количество
Запрос
< Назад
Далее >
OK
Отмена
Справка


Конструктор запроса с обработкой результата *
Обработка результата
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Индекс
Пакет запросов
Поля
Все поля
Групповое поле
РасходнаяТовары.Номенклатура
Суммируемое поле
Функция
РасходнаяТовары.Количество
Сумма
Запрос
< Назад
Далее >
OK
Отмена
Справка



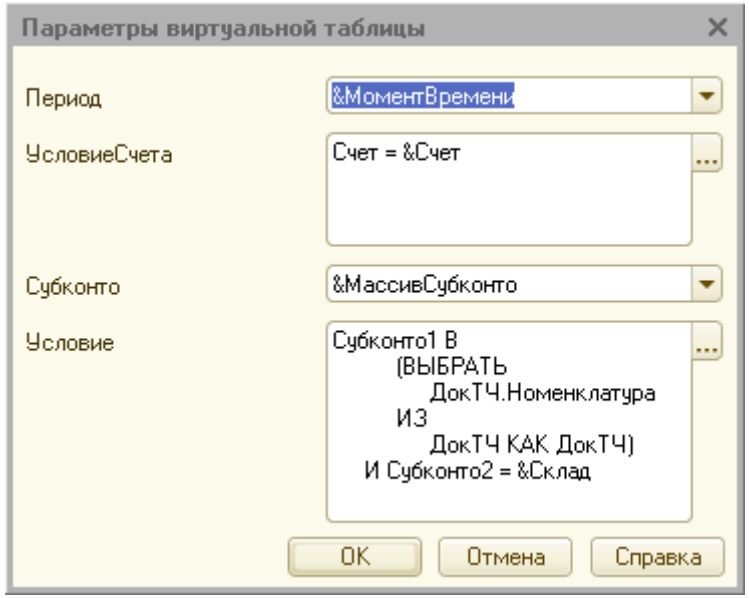
Поместим запрос во временную таблицу (пакет):

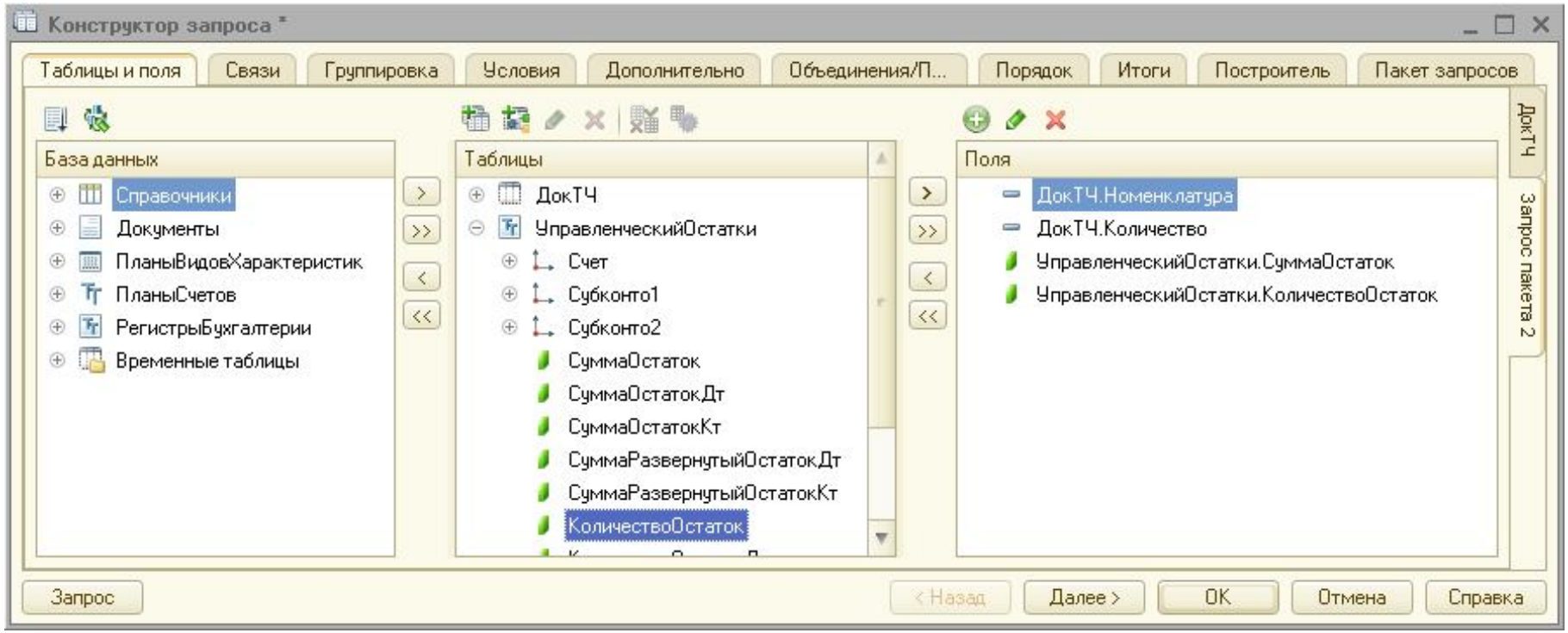


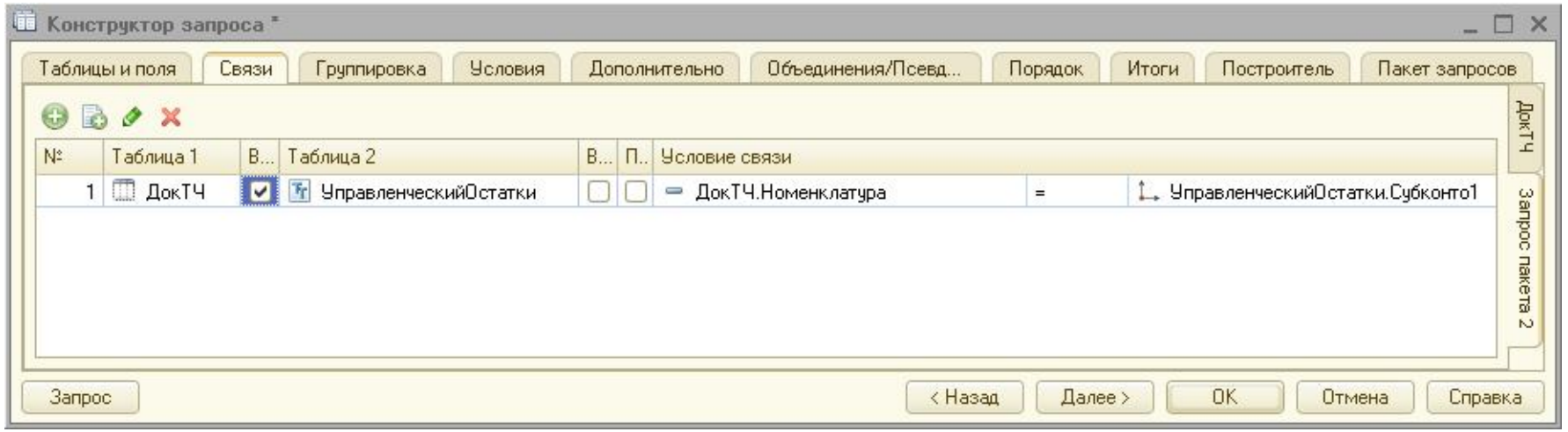
Если бы в результате вернулось большое количество данных, то можно было бы проиндексировать запрос по полю Номенклатура, для ускорения получения данных из результата запроса. Но в нашем примере это необязательно, т.к. данных не много.

На закладке «Пакет запросов» добавим еще один запрос:

В запросе участвует таблица «УправленческийОстатки». Важно правильно настроить ее параметры:
Период - момент проведения документа.
Счет – условия счета. В данном случае нас интересует счет «Товары».
Субконто – ограничим выборку двумя видами Субконто (Номенклатура и Склады).
Условие – наложим условия на сами Субконто (На Номенклатуру и Склады).







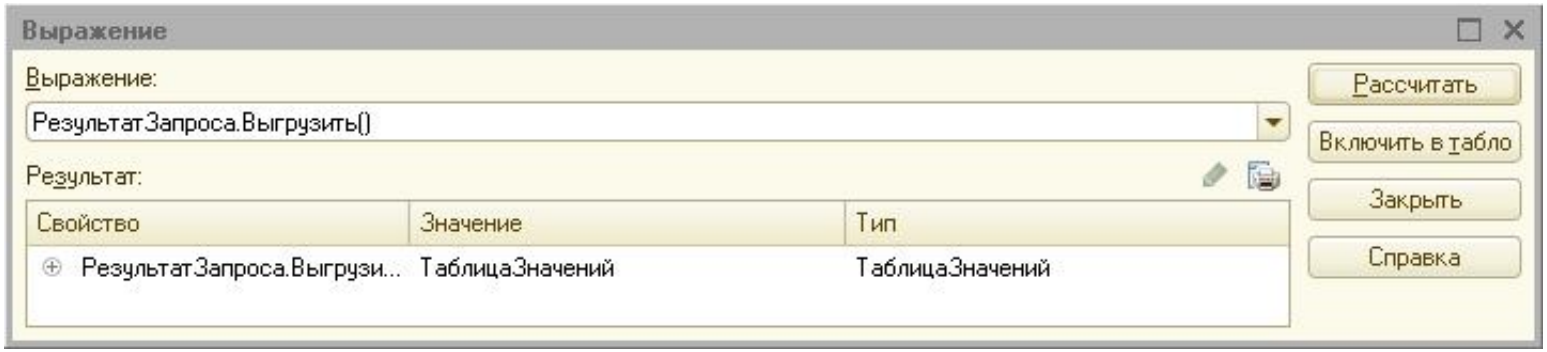
Запрос готов.

Условие по Субконто может быть массивом, Списком значений или одним элементов Плана видов характеристик

Определим параметры запроса:

```
Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.Товары);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Склад", Склад);
```

В режиме исполнения:
Проверим РезультатЗапроса (выделим переменную с результатом запроса и нажмем кнопку «Вычислить выражение»). В строке «Выражение» допишем «Выгрузить()»:



Можно посмотреть ТаблицуЗначений:

Поля СуммаОстаток и КоличествоОстаток могут иметь значения NULL (если по данной Номенклатуре вообще нет остатков в Регистре бухгалтерии). Избавимся от этого в запросе с помощью функции «ЕСТЬNULL».

Сделаем проверку по отрицательным остаткам с помощью выборки из запроса. Если товара не хватает, то выведем сообщение, отключим запись в Регистр бухгалтерии.
При этом можно не отключать формирование движений по другим регистрам (не устанавливать «Отказ»). В примере движения записываются по одному регистру, поэтому отключим формирование движения:

```
Движения.Управленческий.Записывать = Ложь;
```

и выключим транзакцию, чтобы система сообщила, что документ не проведен:

```
Отказ = Истина;
```

«Возврат» делать не будем, т.к. запрос уже сформирован. Пусть по тем строкам, по которым не хватило Товара, движения не формируются, а по строкам, где Товара хватает – формируются:

```
Если Отказ Тогда
Продолжить;
КонецЕсли;
```

Если Товара хватает, то рассчитаем его Себестоимость:

```
Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;
```

Сформируем движения.

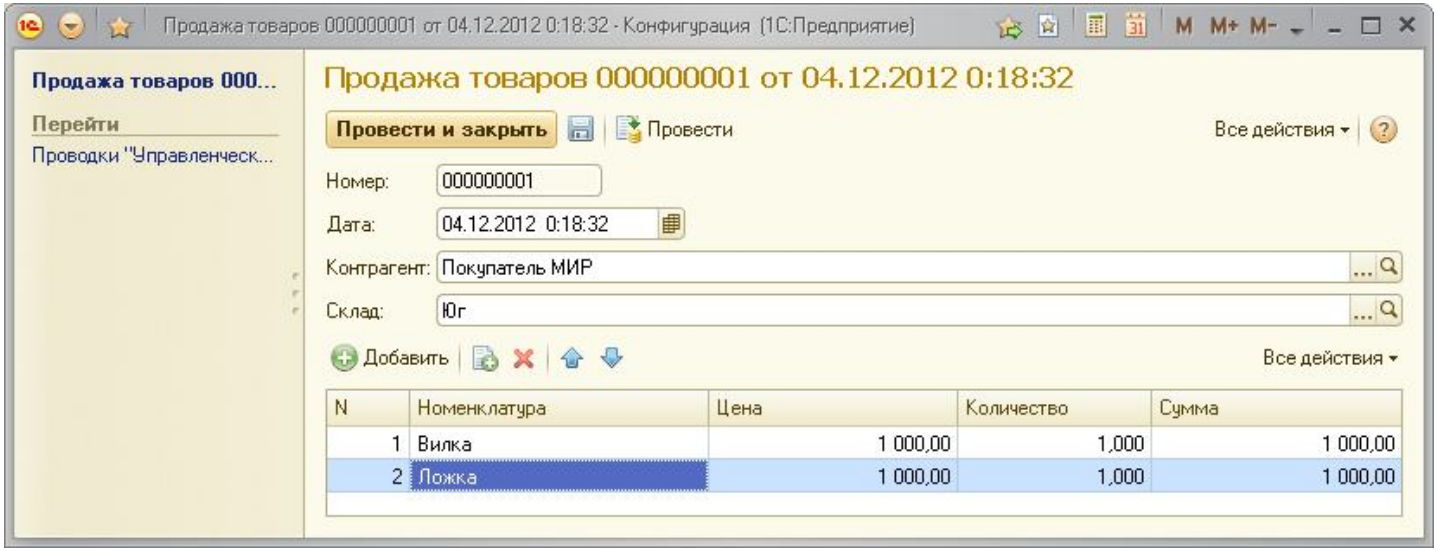
Итого код:

```
Процедура ОбработкаПроведения(Отказ, Режим)

	Запрос = Новый Запрос;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	РасходнаяТовары.Номенклатура,
	|	СУММА(РасходнаяТовары.Количество) КАК Количество
	|ПОМЕСТИТЬ ДокТЧ
	|ИЗ
	|	Документ.Расходная.Товары КАК РасходнаяТовары
	|ГДЕ
	|	РасходнаяТовары.Ссылка = &Ссылка
	|
	|СГРУППИРОВАТЬ ПО
	|	РасходнаяТовары.Номенклатура
	|;
	|
	|////////////////////////////////////////////////////////////////////////////////
	|ВЫБРАТЬ
	|	ДокТЧ.Номенклатура,
	|	ДокТЧ.Количество,
	|	ЕСТЬNULL(УправленческийОстатки.СуммаОстаток, 0) КАК СуммаОстаток,
	|	ЕСТЬNULL(УправленческийОстатки.КоличествоОстаток, 0) КАК КоличествоОстаток
	|ИЗ
	|	ДокТЧ КАК ДокТЧ
	|		ЛЕВОЕ СОЕДИНЕНИЕ РегистрБухгалтерии.Управленческий.Остатки(
	|				&МоментВремени,
	|				Счет = &Счет,
	|				&МассивСубконто,
	|				Субконто1 В
	|					(ВЫБРАТЬ
	|						ДокТЧ.Номенклатура
	|					ИЗ
	|						ДокТЧ КАК ДокТЧ)
```

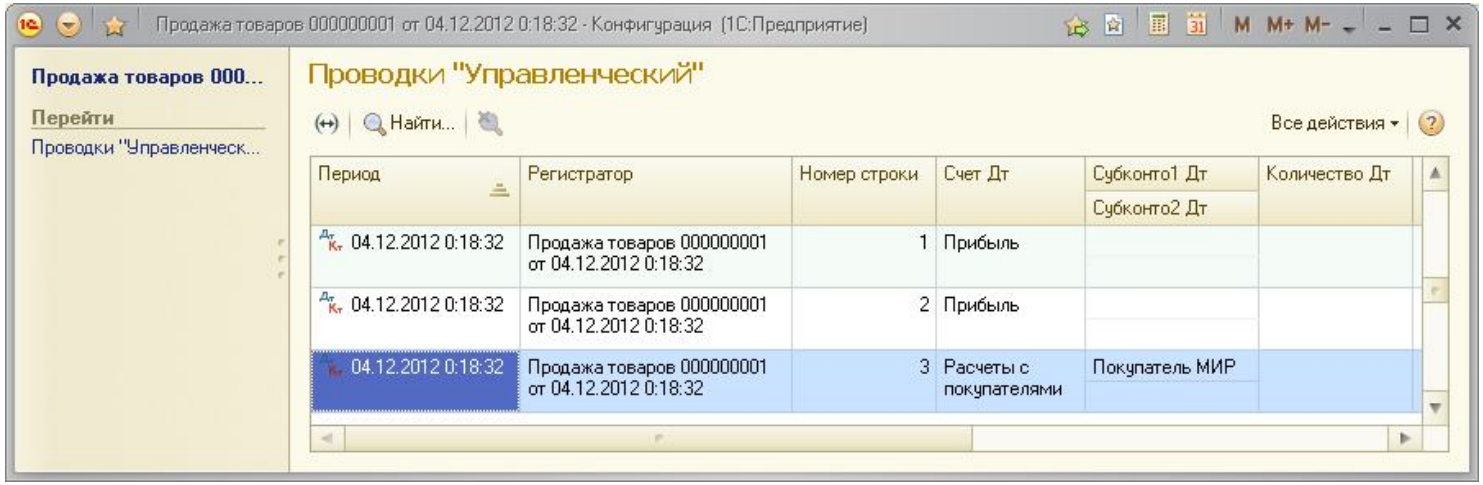
```
|                          И Субконто2 = &Склад) КАК УправленческийОстатки
|          ПО ДокТЧ.Номенклатура = УправленческийОстатки.Субконто1";

Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.Товары);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

     Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
          + ". Из необходимых " + Выборка.Количество +
          " в наличии имеется только " + Выборка.КоличествоОстаток;
          Сообщение.Сообщить();

          Движения.Управленческий.Записывать = Ложь;
          Отказ = Истина;
     КонецЕсли;

     Если Отказ Тогда
          Продолжить;
     КонецЕсли;

     Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;
```

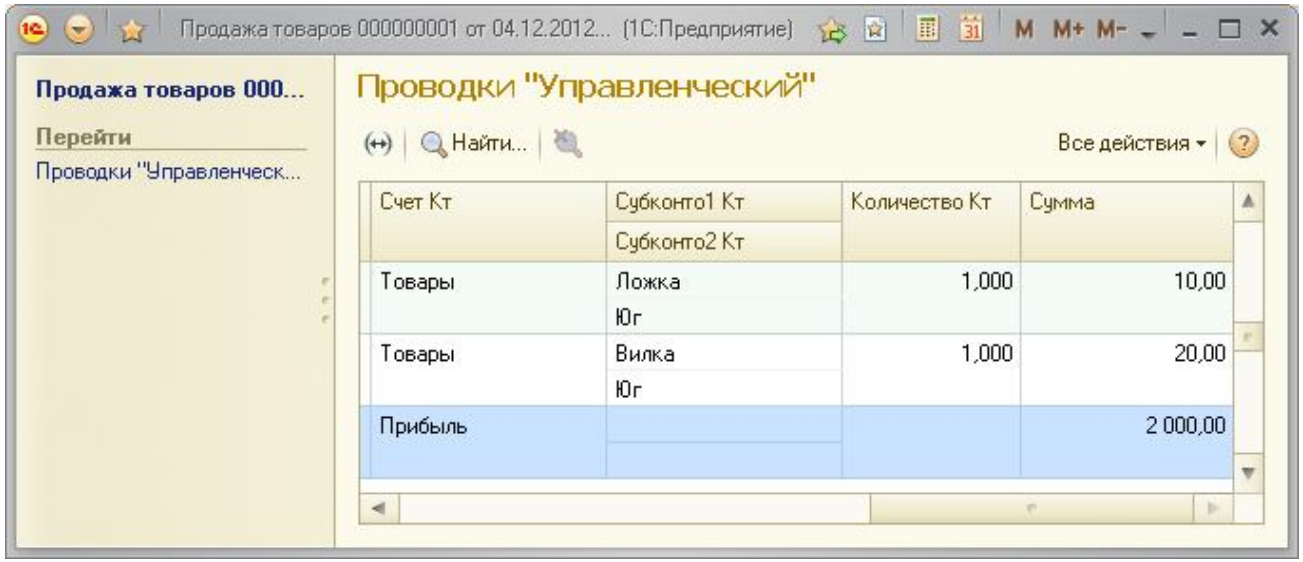
```
        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
        Движение.СчетКт = ПланыСчетов.Управленческий.Товары;
        Движение.Период = Дата;
        Движение.Сумма = Себестоимость;
        Движение.КоличествоКт = Выборка.Количество;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
    КонецЦикла;

    Движение = Движения.Управленческий.Добавить();
    Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
    Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
    Движение.Период = Дата;
    Движение.Сумма = Товары.Итог("Сумма");
    Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;

КонецПроцедуры
```

В режиме исполнения:
Проведем документ Расходная и посмотрим результат:



Результат:





Если надо посмотреть весь текст запроса, то надо поставить Точку останова на строке выполнения запроса и «Вычислить выражение»:

```
Запрос.Текст
```

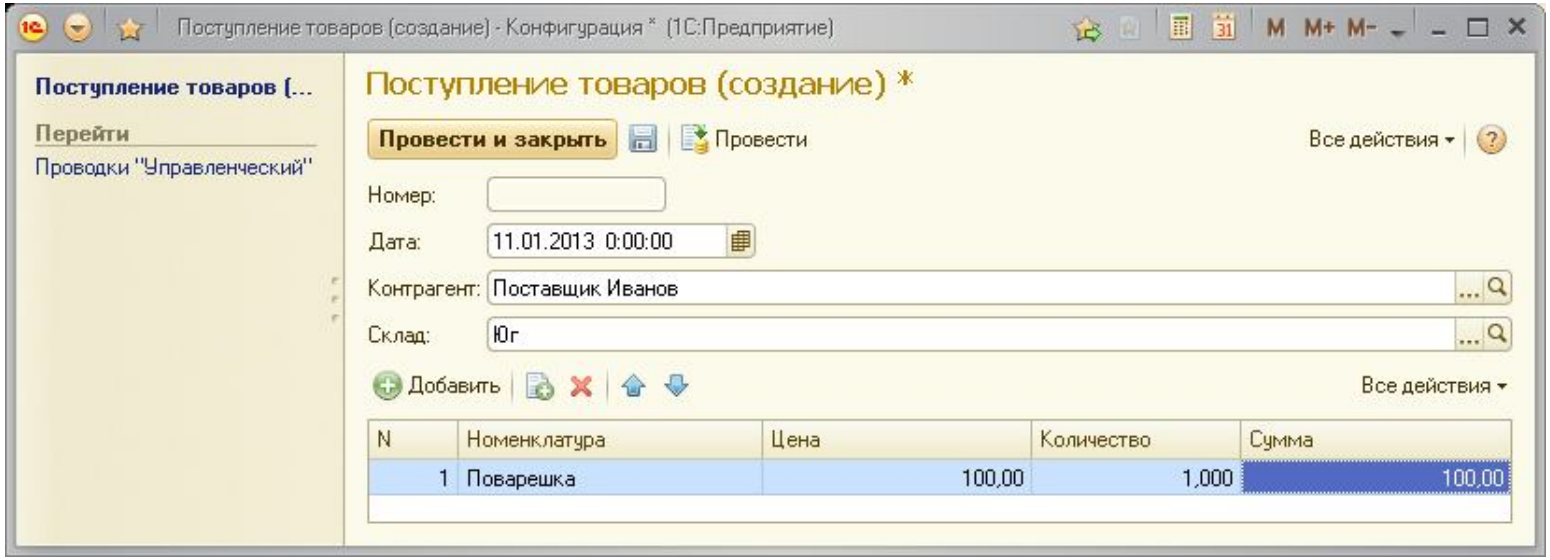
Можно скопировать текст запроса, создать Внешнюю обработку, открыть Модуль объекта, вставить туда запрос, поставить кавычки в начале и в конце. После этого можно открыть запрос в Конструкторе запроса.

Пока в примере расчет Себестоимости по-прежнему производится не совсем корректно.

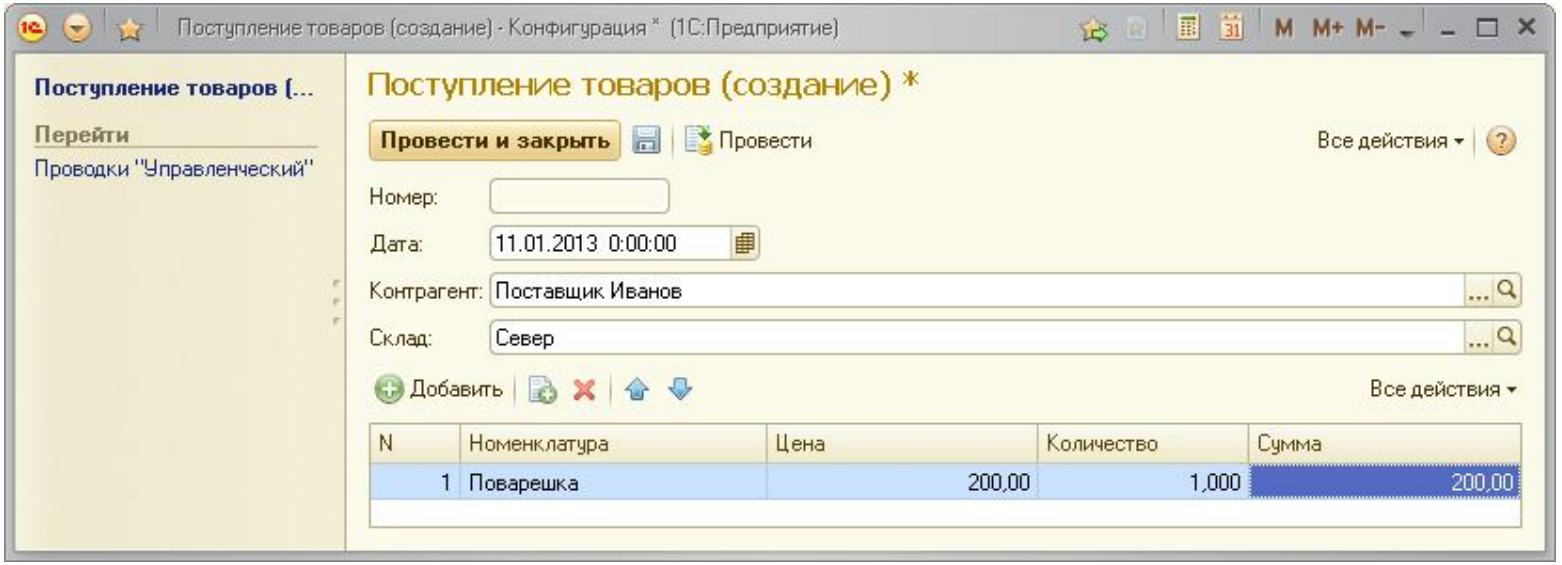
Занятие 2.22

Сейчас в примере Себестоимость считается по Складам, что не верно, если Себестоимость рассчитывается для целей налогообложения. Себестоимость по Складам, Обособленным подразделениям, Торговым точкам можно рассчитывать для целей Управленческого учета.

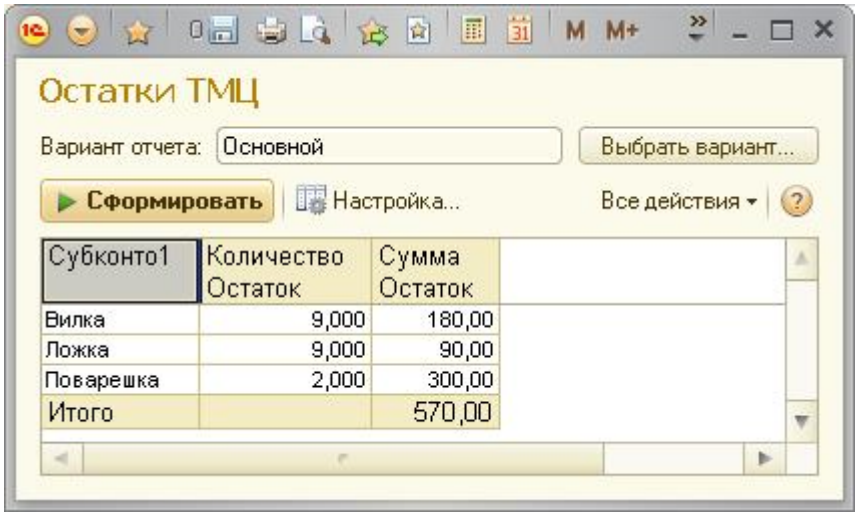
Пример:
Создадим Приходную накладную:



И вторую, аналогичную, только на другой Склад и по другой Цене:

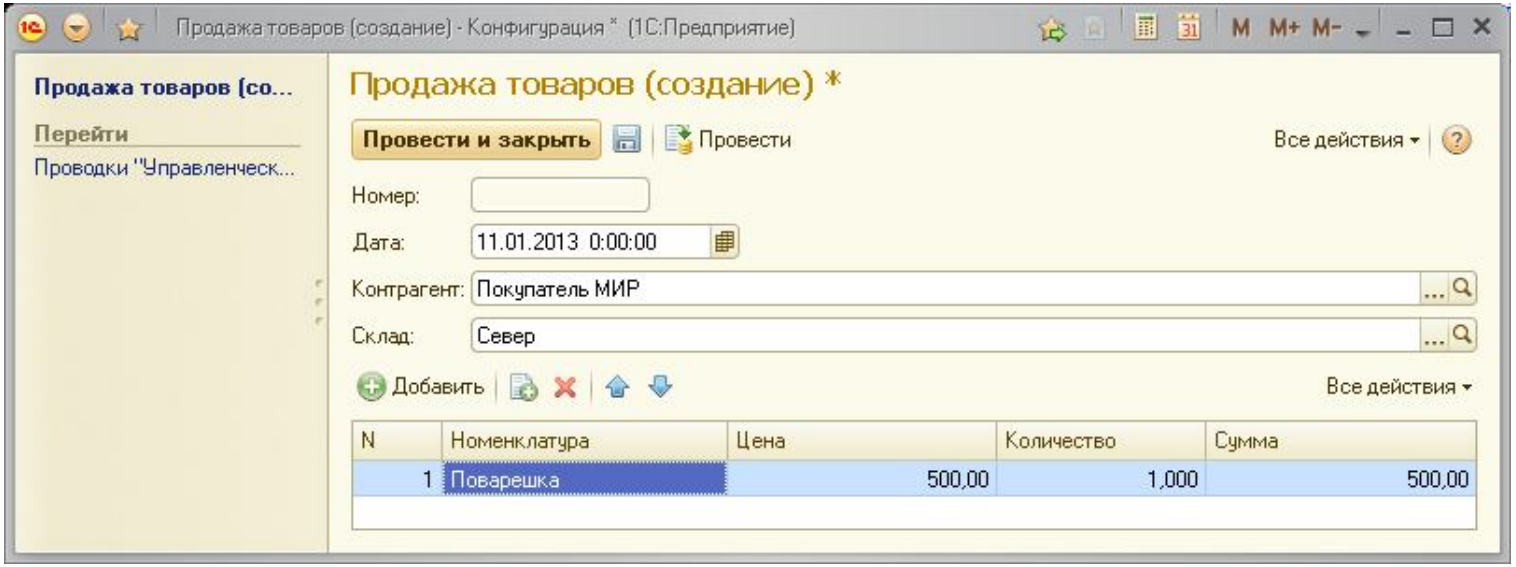


Проверим отчет «Остатки ТМЦ»:

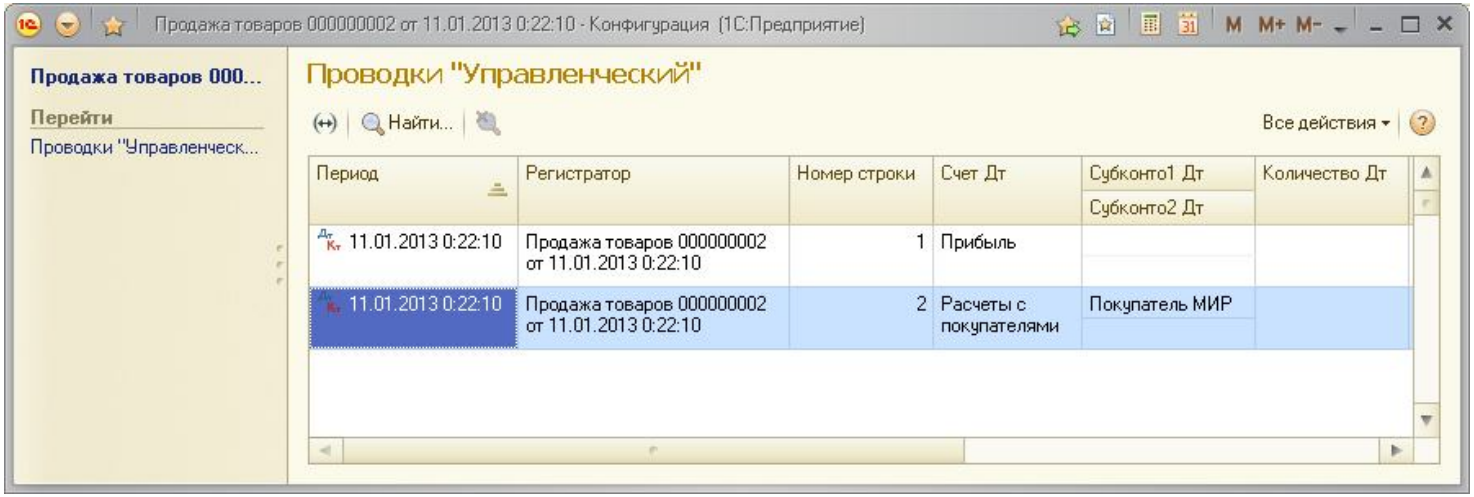


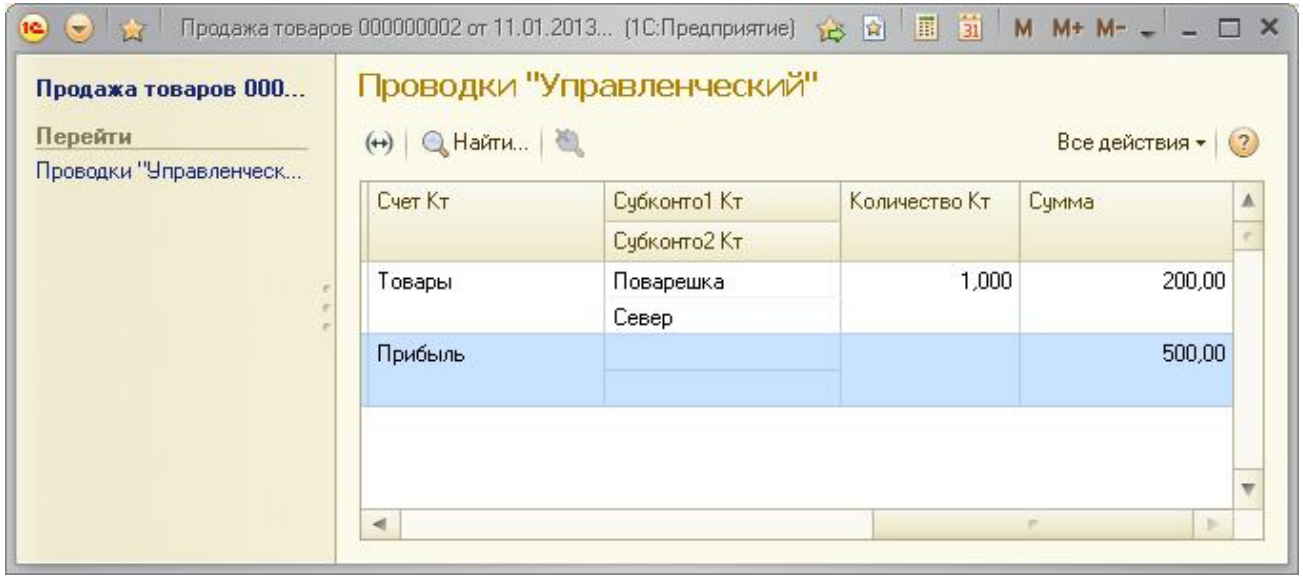
По отчету можно подумать, что Себестоимость поварешки:
300 / 2 = 150

Но при продаже, стоимость поварешки будет разниться, в зависимости от того, с какого Склада мы ее будем продавать:



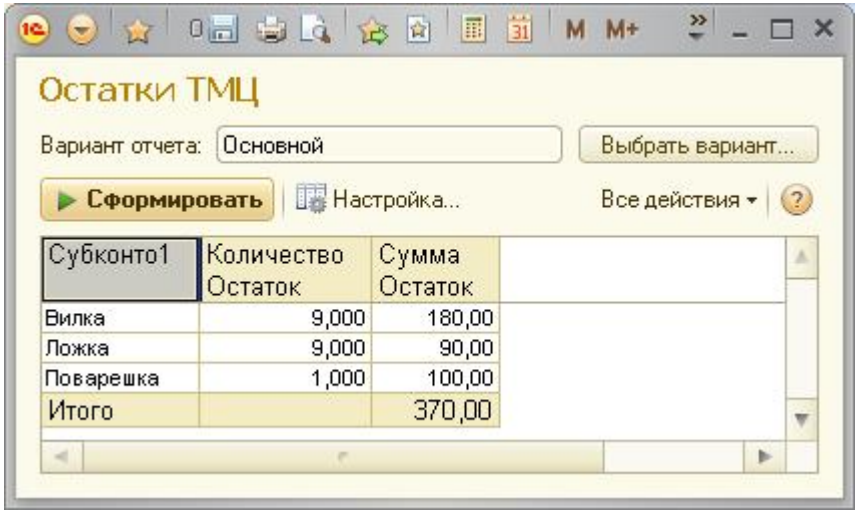
Проведем Расходную и проверим проводки:





Система рассчитала Себестоимость: 200.

Снова проверим отчет «Остатки ТМЦ»:



Субконто1	Количество Остаток	Сумма Остаток
Вилка	9,000	180,00
Ложка	9,000	90,00
Поварешка	1,000	100,00
Итого		370,00

Получается, что 2 поварешки стоили 300 рублей, а одна уже 100. Это приведет к путанице.

Если рассчитать Себестоимость без учета Склада, ничего не меняя, то «поплывут» остатки. Пример:

Движения

	Номенклатура	Склад	Количество	Цена
+	Ложка	Север	1	100
+	Ложка	Юг	1	200

Остатки

Номенклатура	Склад	Количество	Цена
Ложка	Север	1	100
Ложка	Юг	1	200

Надо списать По среднему по Предприятию. Допустим, спишем со Склада «Юг» по стоимости 150 рублей, тогда в Остатках система запишет остаток 0 на 50 рублей:

Движения

	Номенклатура	Склад	Количество	Цена
+	Ложка	Север	1	100
+	Ложка	Юг	1	200
-	Ложка	Юг	1	150

Остатки

Номенклатура	Склад	Количество	Цена
Ложка	Север	1	100
Ложка	Юг	0	50

Если бы списали с «Севера», то осталось бы 0 на «-50» рублей.

Сейчас по счету «Товары» ведется количественный и стоимостной учет. Отключим стоимостной учет по Складам (Субконто «Склады»).
При этом Движения будут формироваться следующим образом:

Движения

	Номенклатура	Склад	Количество	Цена
+	Ложка	Юг	1	
	Ложка			100

Это одно движение. При этом Остатки:

Остатки

Номенклатура	Склад	Количество	Цена
Ложка	Юг	1	
Ложка			100

Сформируем еще одно поступление:

Движения

	Номенклатура	Склад	Количество	Цена
+	Ложка	Юг	1	
	Ложка			100
+	Ложка	Север	1	
	Ложка			200

При этом Остатки:

Остатки

Номенклатура	Склад	Количество	Цена
Ложка	Юг	1	
Ложка	Север	1	
Ложка			300

Если сформируем Расход:

Движения

	Номенклатура	Склад	Количество	Цена
+	Ложка	Юг	1	
	Ложка			100
+	Ложка	Север	1	
	Ложка			200
-	Ложка	Юг	1	
	Ложка			150

При этом Остатки (на «Юге» ложек не осталось):

Остатки

Номенклатура	Склад	Количество	Цена
Ложка	Север	1	
Ложка			150

При этом для пользователя вид записей никак не изменится.

Итого:

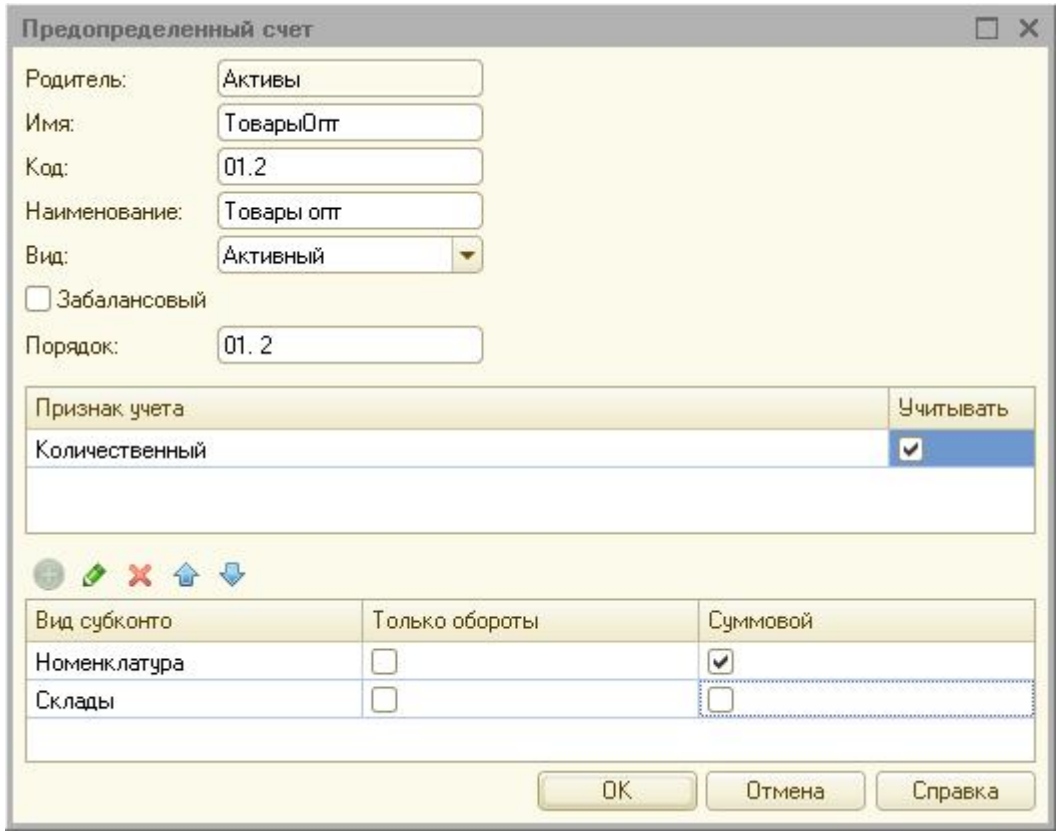
- По Субконто «Номенклатура» храним Количество и Сумму.
- По Субконто «Склады» храним только Количество.

Создадим Признак учета по Субконто. Т.е. по Счету накапливается Сумма и Количество, а внутри Счета по каждому виду аналитики (Субконто) ведется свой учет.

Закладка «Субконто» - Признаки учета субконто:

Свойство	**Значение**
Имя	Суммовой
Синоним	Суммовой

Определим счета, по которым будет вестись данный признак учета. Чтобы не менять предыдущие записи, сделаем еще один счет по «Товарам»:



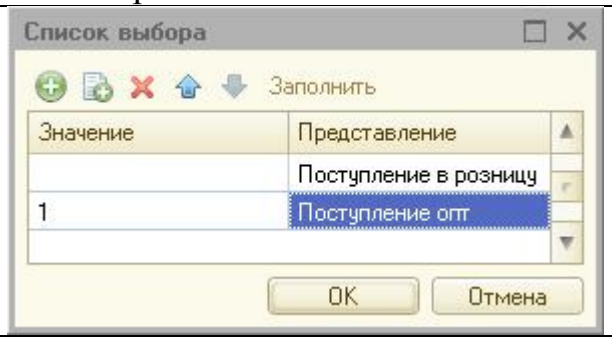
Опишем формирование Движений по Счету «Товары» документом Приходная:
Реализуем в Приходной два типа движений:

- По счету «Товары» с учетом Суммы по разрезу (Субконто) «Склады».
- По счету «Товары» без учета Суммы по разрезу (Субконто) «Склады».

Добавим реквизит «Тип операции»:

Свойство	Значение
Имя	ТипОперации
Синоним	Тип операции
Тип	Число 1, 0

Добавим реквизит на форму документа и настроим свойства:

Свойство	Значение
Вид	Поле переключателя
Список выбора	

Модуль объекта:

```
Процедура ОбработкаПроведения(Отказ, Режим)
	// регистр Управленческий

	СчетУчета = ?(ТипОперации = 0,
		ПланыСчетов.Управленческий.Товары,
		ПланыСчетов.Управленческий.ТоварыОпт);

	Движения.Управленческий.Записывать = Истина;
	Для Каждого ТекСтрокаТовары Из Товары Цикл
		Движение = Движения.Управленческий.Добавить();
		Движение.СчетДт = СчетУчета;
		Движение.СчетКт = ПланыСчетов.Управленческий.РасчетыСПоставщиками;
```

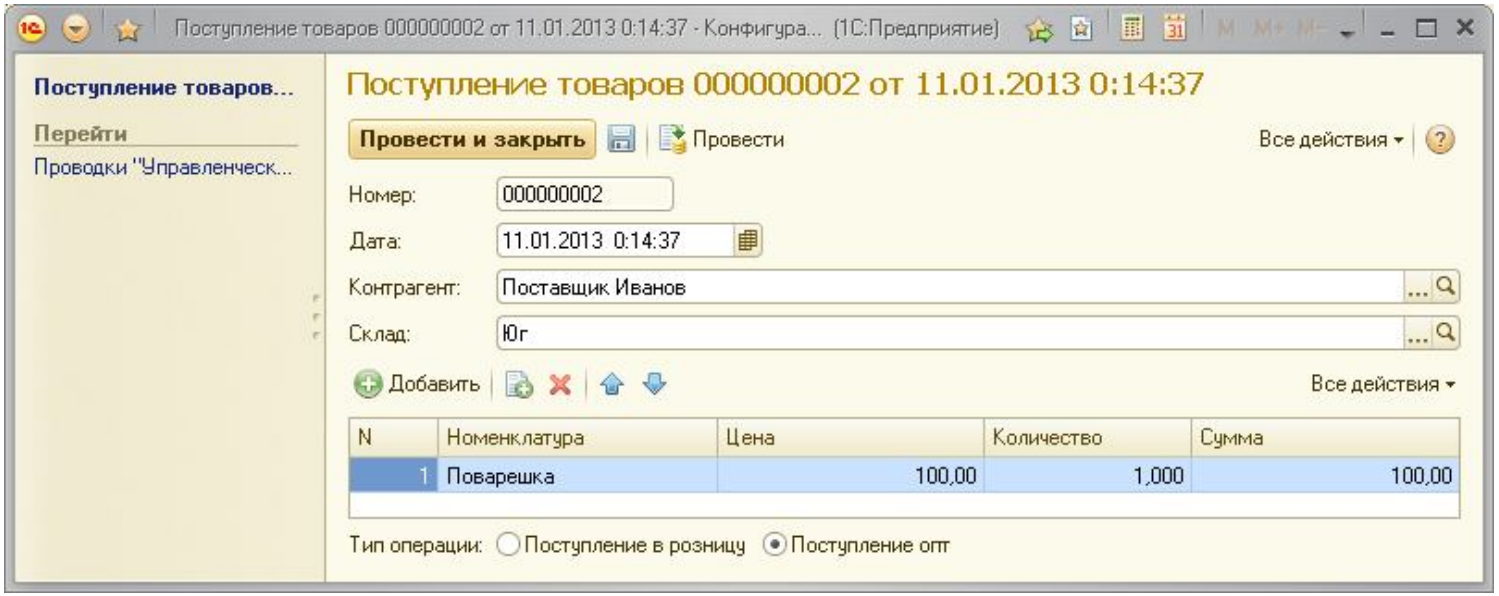
```
        Движение.Период = Дата;
        Движение.Сумма = ТекСтрокаТовары.Сумма;
        Движение.КоличествоДт = ТекСтрокаТовары.Количество;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = ТекСтрокаТовары.Номенклатура;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
    КонецЦикла;

КонецПроцедуры
```

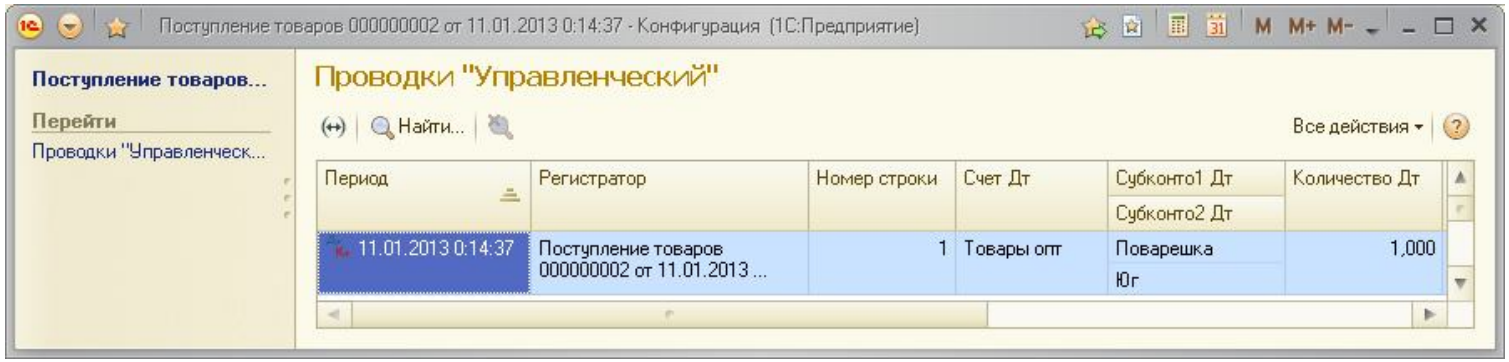
В Регистре бухгалтерии «Управленческий» Ресурс «Сумма» свяжем с Признаком учета субконто:

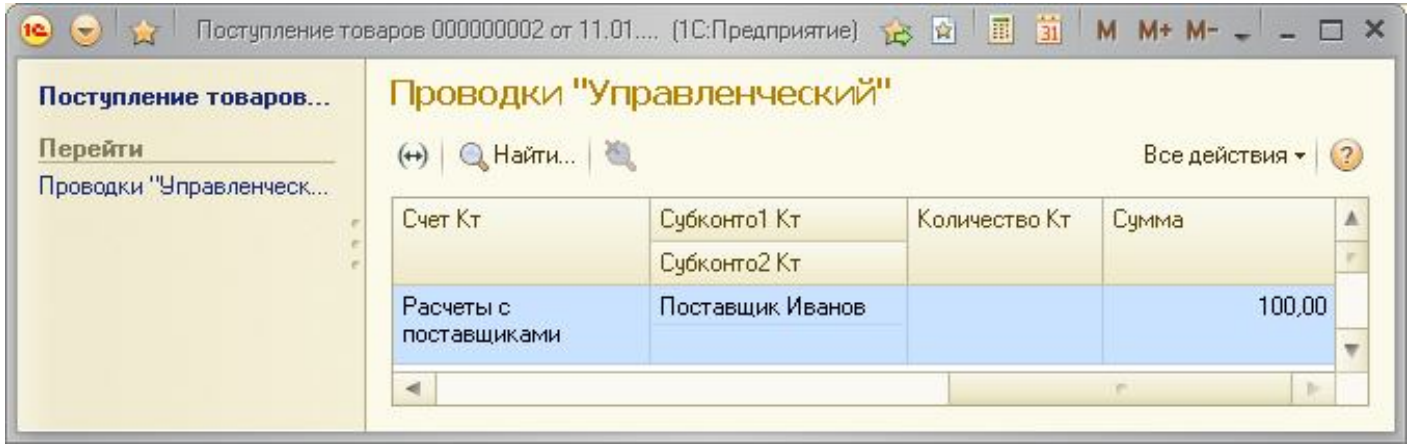
Свойство	Значение
Признак учета субконто	Суммовой

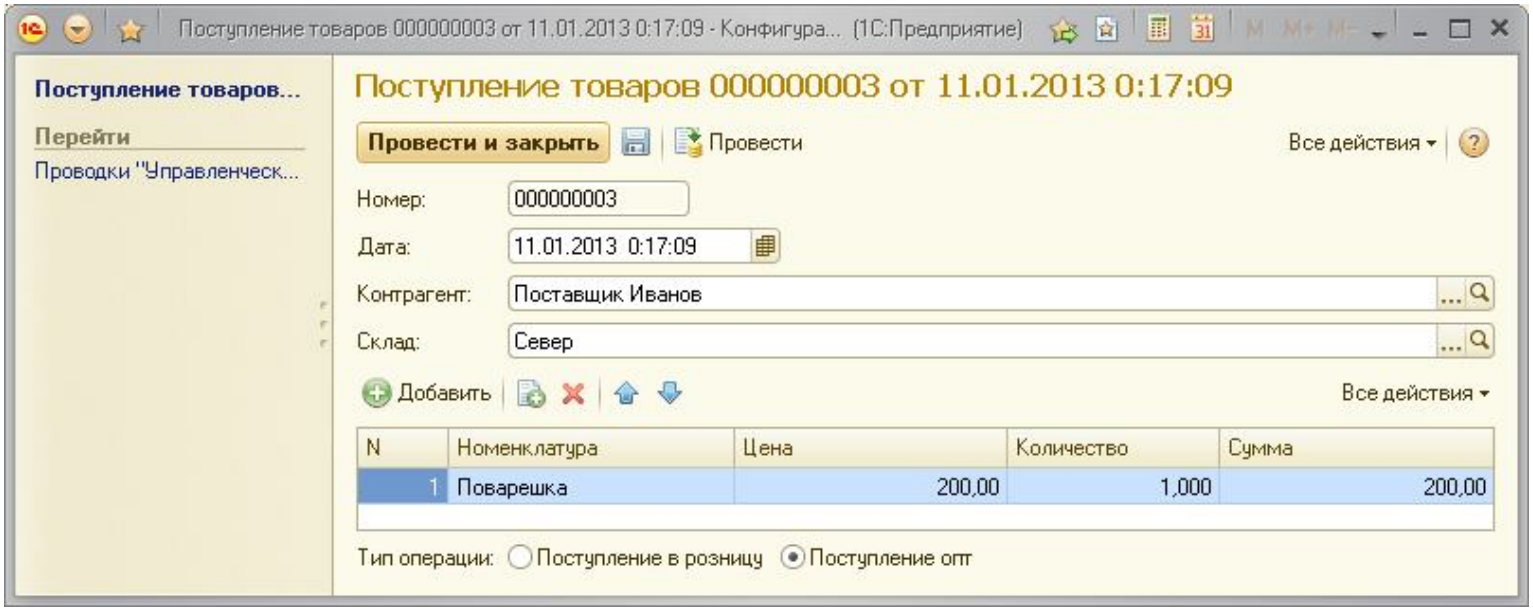
В режиме исполнения:

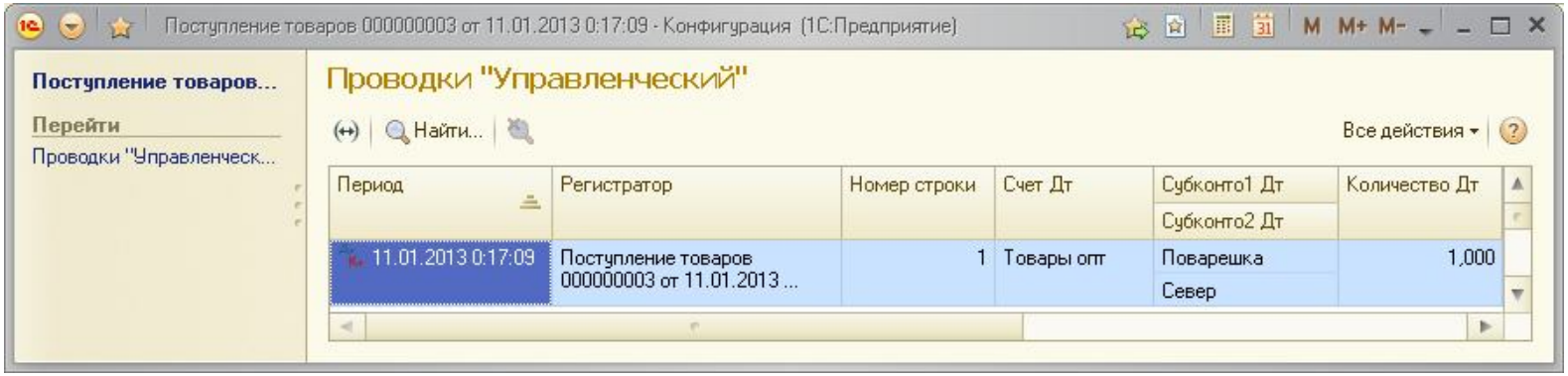
В документах Приходная поменяем «Тип операции» и перепроведем их. В Движениях все останется как прежде:

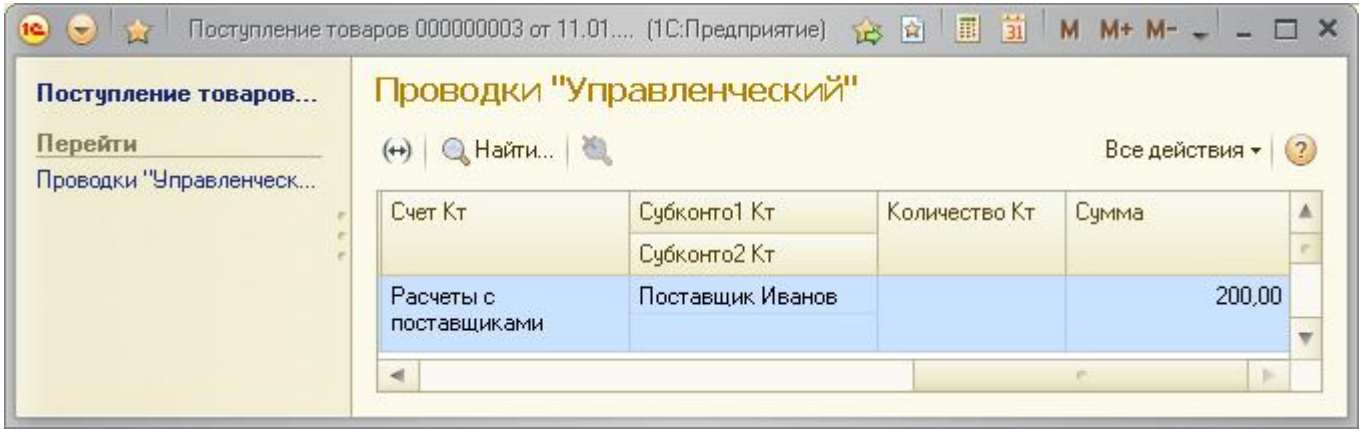



Поступление товаров 000000002 от 11.01.2013 0:14:37 - Конфигурация (1С:Предприятие)
Поступление товаров...
Перейти
Проводки "Управленческ...
Проводки "Управленческий"
Найти...
Все действия
Период
Регистратор
Номер строки
Счет Дт
Субконто1 Дт
Субконто2 Дт
Количество Дт
11.01.2013 0:14:37
Поступление товаров 000000002 от 11.01.2013 ...
1
Товары опт
Поварешка
Юг
1,000


Поступление товаров 000000002 от 11.01.... (1С:Предприятие)
Поступление товаров...
Перейти
Проводки "Управленческ...
Проводки "Управленческий"
Найти...
Все действия
Счет Кт
Субконто1 Кт
Субконто2 Кт
Количество Кт
Сумма
Расчеты с поставщиками
Поставщик Иванов
100,00


Поступление товаров 000000003 от 11.01.2013 0:17:09 - Конфигура... (1С:Предприятие)
Поступление товаров...
Перейти
Проводки "Управленческ...
Поступление товаров 000000003 от 11.01.2013 0:17:09
Провести и закрыть
Провести
Все действия
Номер:
000000003
Дата:
11.01.2013 0:17:09
Контрагент:
Поставщик Иванов
Склад:
Север
Добавить
Все действия
N
Номенклатура
Цена
Количество
Сумма
1
Поварешка
200,00
1,000
200,00
Тип операции:
Поступление в розницу
Поступление опт

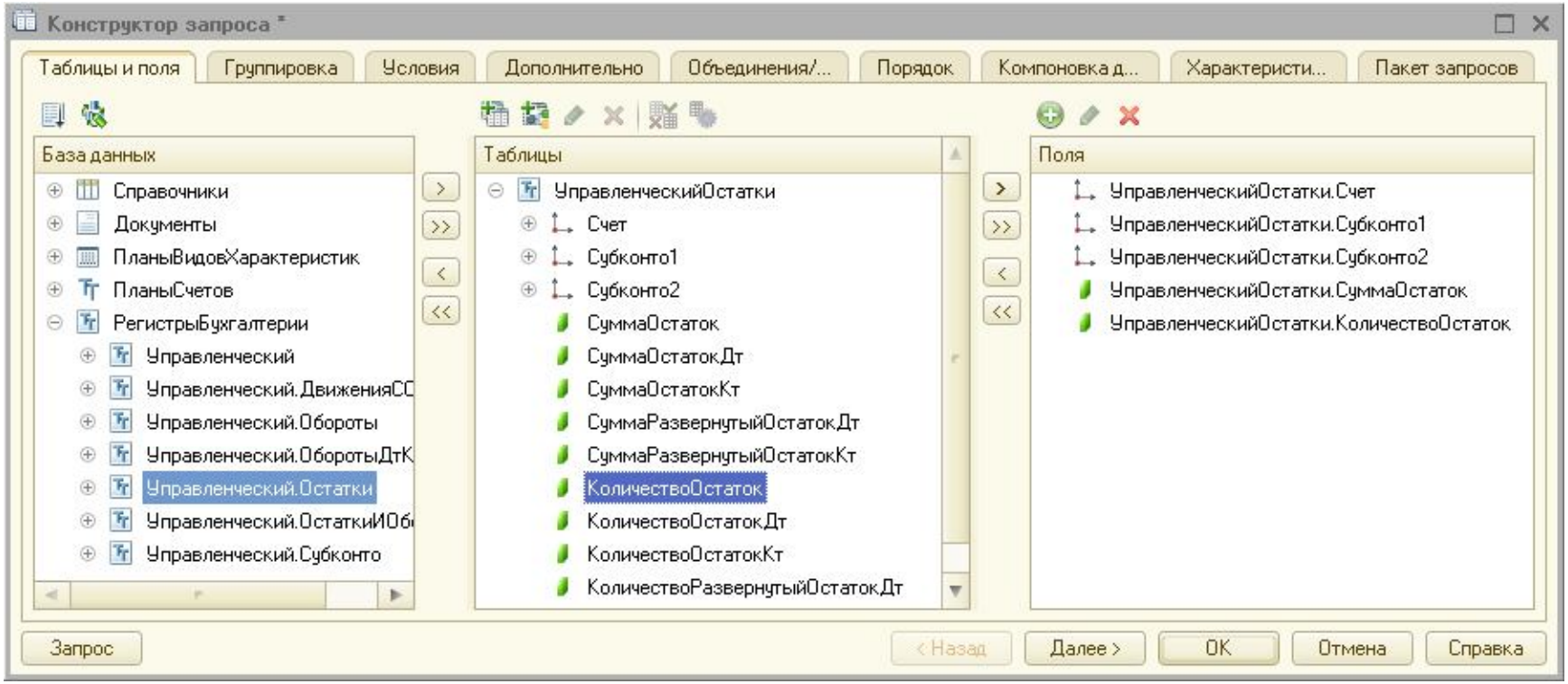




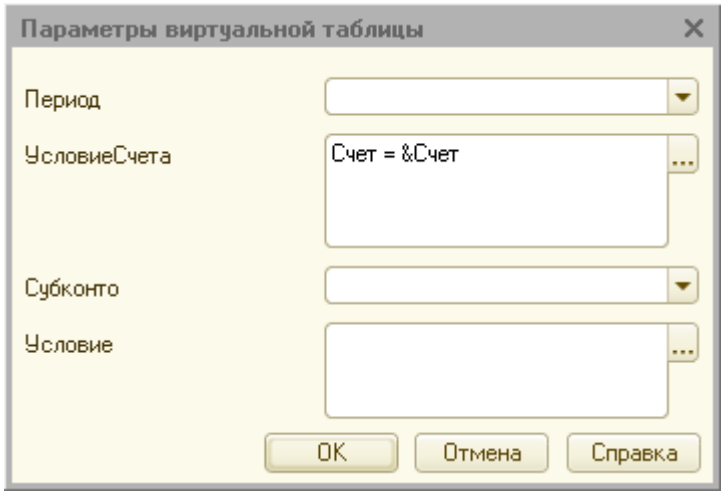
В отчете «Остатки ТМЦ» все максимально сгруппировано, т.е. в нем тоже не увидеть изменения. Сформируем отчет с «сырыми» данными, где будет видно, что аналитика по Складам не имеет Суммового выражения:

Свойство	Значение
Имя	ОтчетПоСчТоварыБезСуммыПоСкладам
Синоним	Отчет по счету Товары (без суммы по складам)

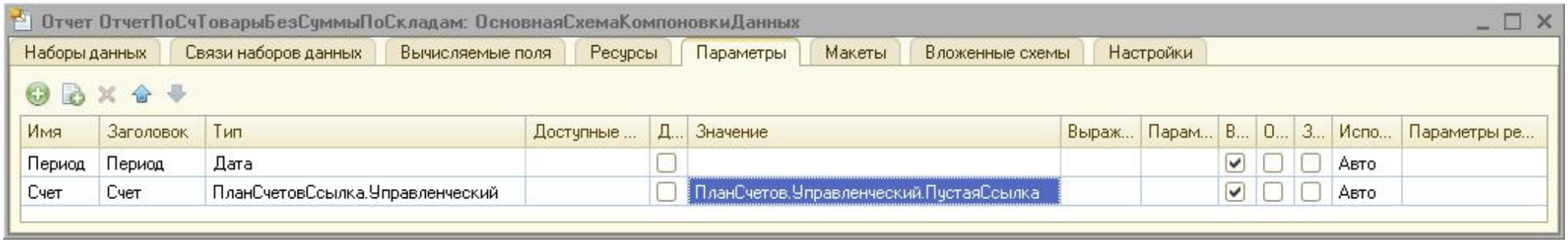
СКД – Добавить набор данных-запрос – Конструктор запроса:



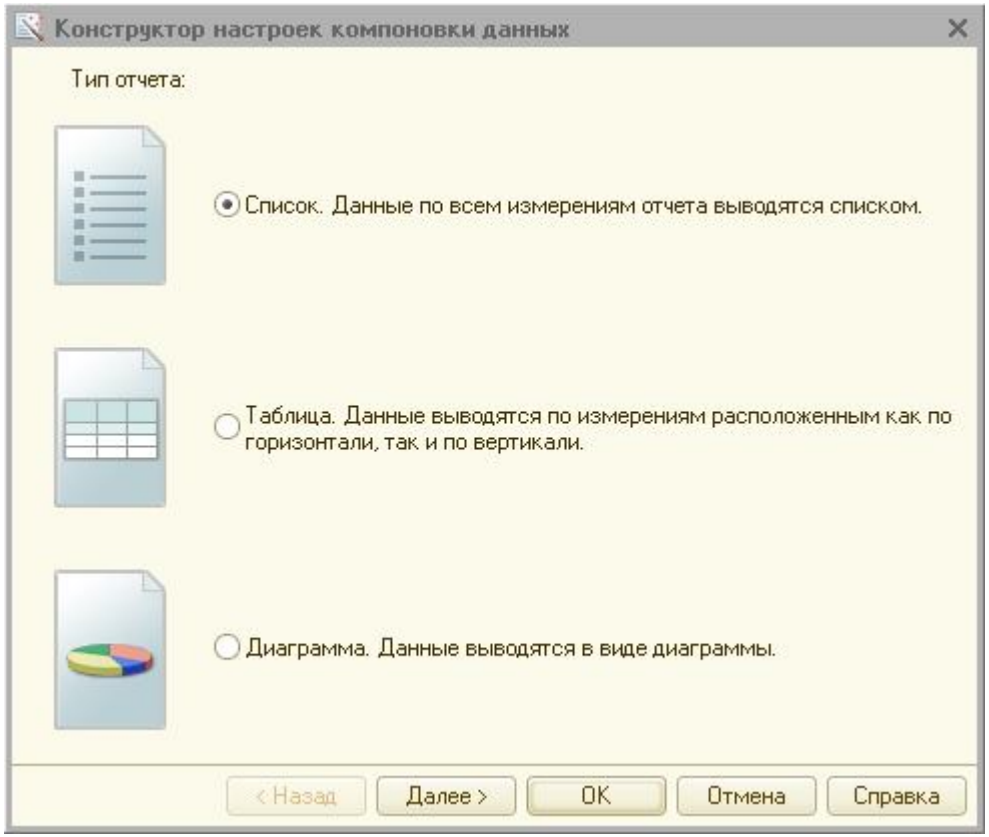
Параметры виртуальной таблицы «УправленческийОстатки»:

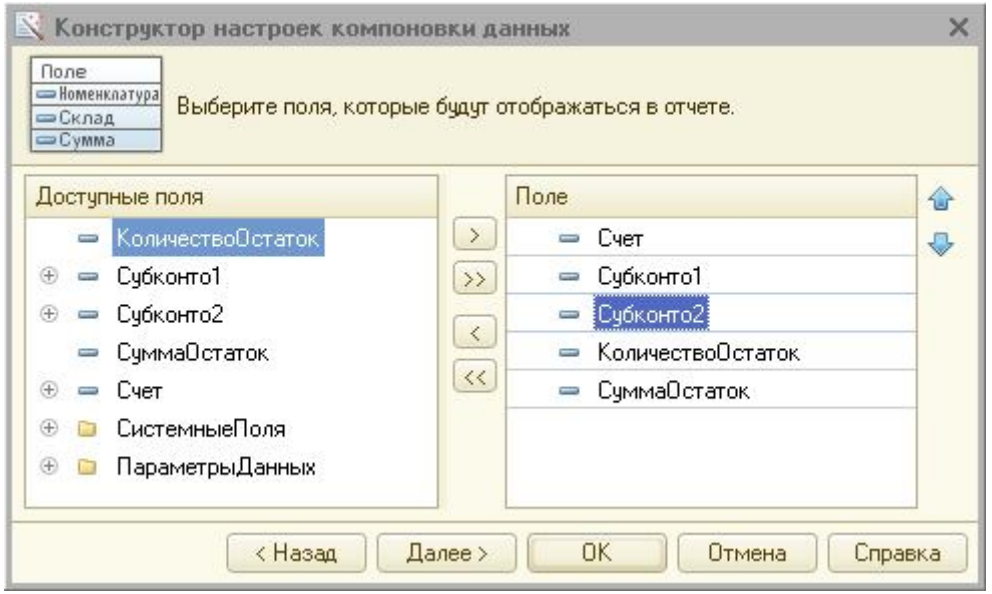


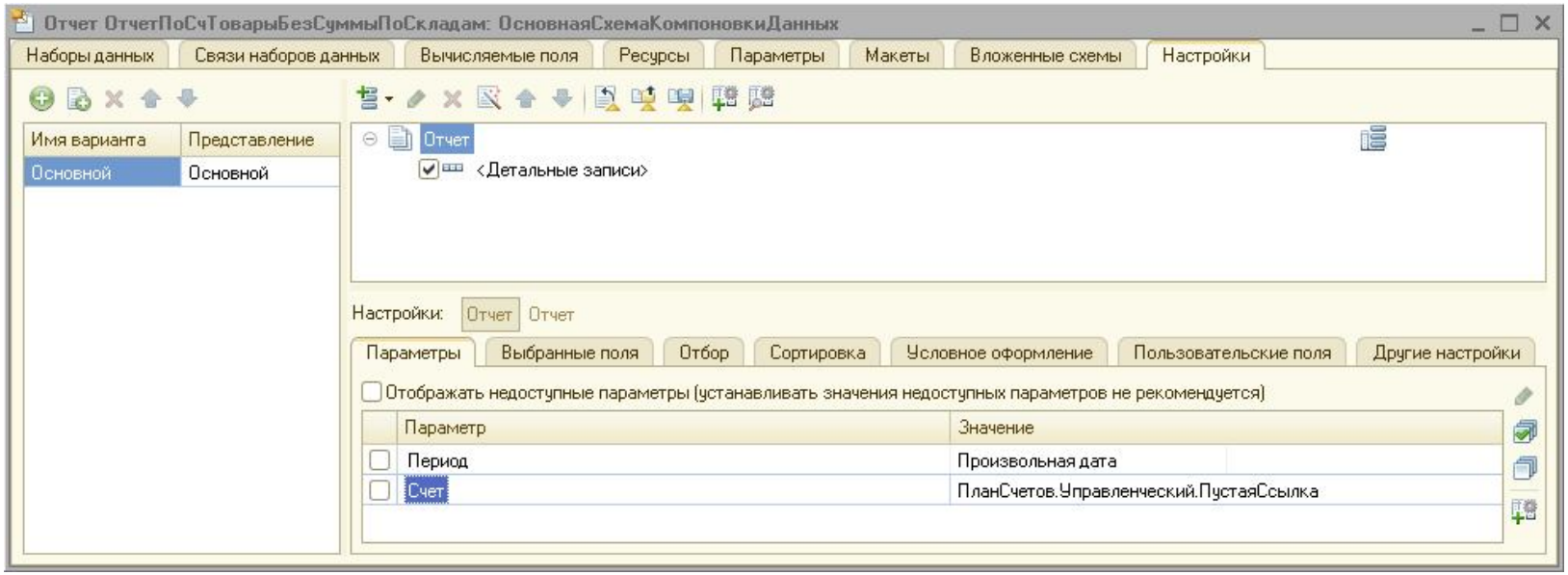
Настройки СКД:



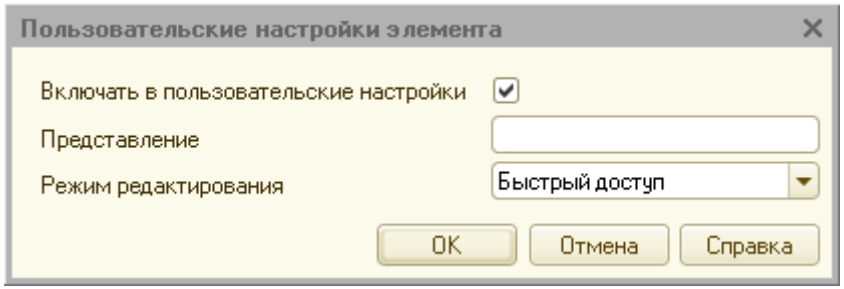
Конструктор настроек отображения (закладка «Настройки»):





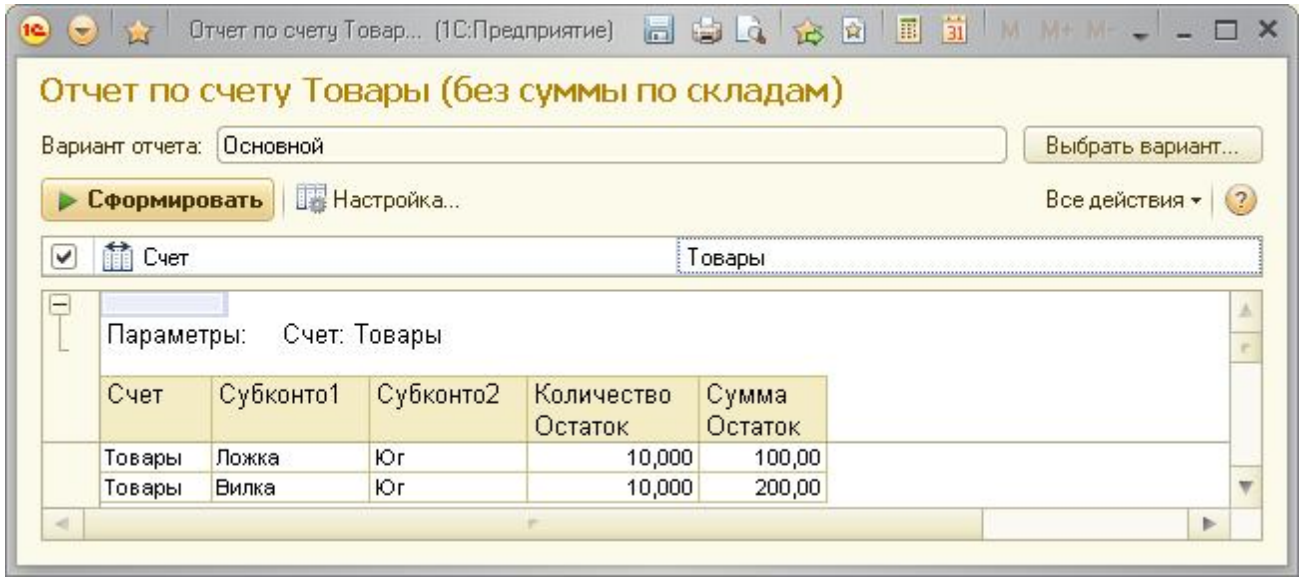


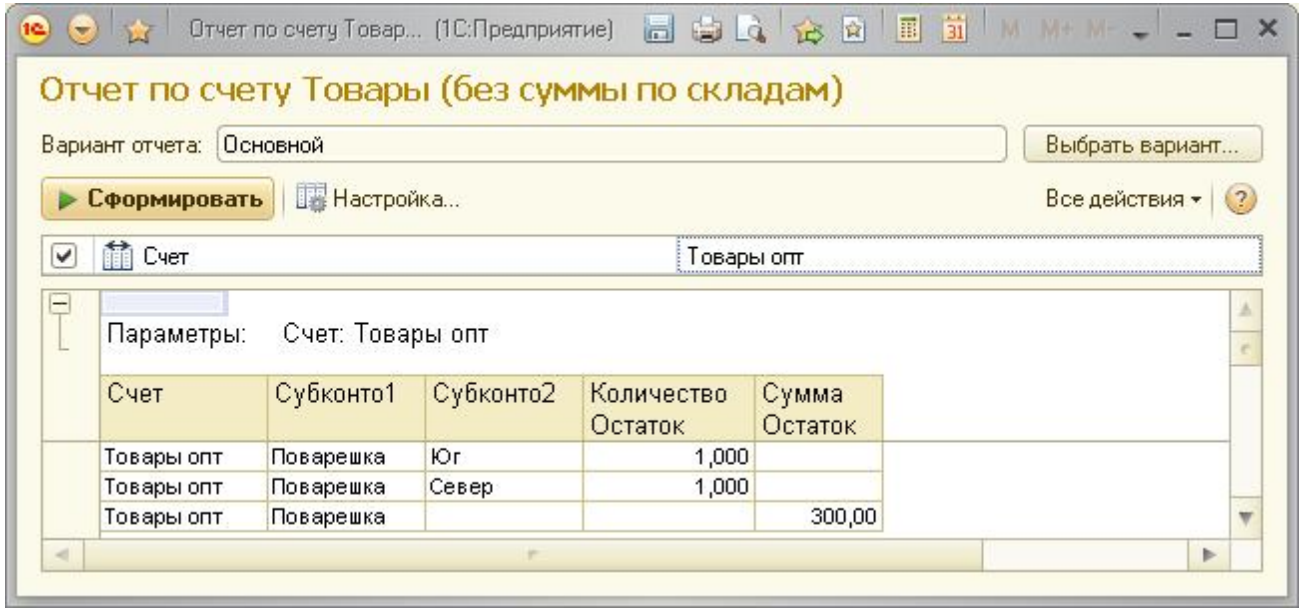
Счет – ПКМ – Свойства элемента пользовательских настроек:



В режиме исполнения:

Отменим проведение Расходных.

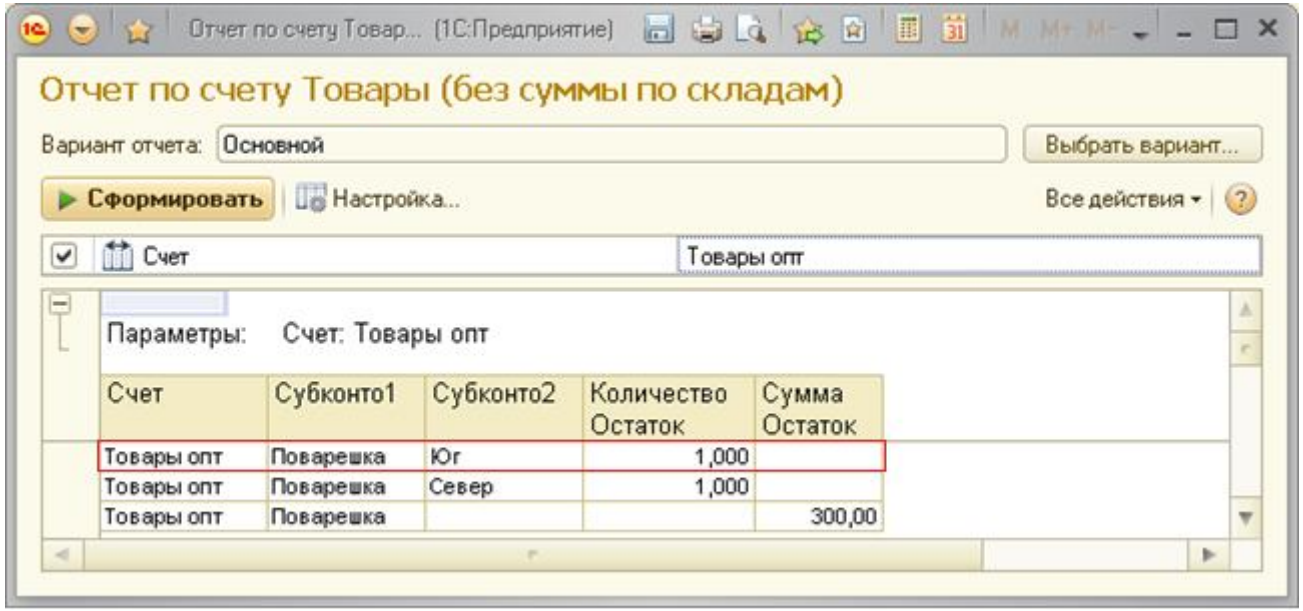




Чтобы запустить «Консоль запроса», сделанную под Обычное приложение, в свойствах конфигурации:

Свойство	Значение
Основной режим запуска	Обычное приложение

При продаже товара надо будет рассчитывать Себестоимость и производить контроль остатков по Складам. При этом, при получении остатков из Регистра бухгалтерии, мы не можем ограничить получение остатков по конкретному складу, информация будет недостаточной:



Нет Суммы и общего Количества (для расчета Себестоимости). Поэтому отбор по складу надо устанавливать не в параметрах виртуальной таблицы, а уже после получения таблицы вида, изображенного на рисунке выше. Также надо рассчитать итоговое количество товара по всем складам.
Сформируем запрос в консоли запросов:

Выберем все записи из документов Расходная:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 000000001 от 04.12.2012 0:18:32	1	Вилка	1 000,00	1,000	1 000,00
Продажа товаров 000000002 от 11.01.2013 0:22:10	1	Поварешка	500,00	1,000	500,00
Продажа товаров 000000001 от 04.12.2012 0:18:32	2	Ложка	1 000,00	1,000	1 000,00

Установим отбор по документу, например «Продажа товаров 000000002 от 11.01.2013 0:22:10»:

```
ВЫБРАТЬ
	*
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка
```

Ссылка	НомерСтроки	Номенклатура	Цена	Количество	Сумма
Продажа товаров 000000002 от 11.01.2013 0:22:10	1	Поварешка	500,00	1,000	500,00

Получим данные из регистра бухгалтерии:

```
ВЫБРАТЬ
      Счет,
      Субконто1,
      Субконто2,
      СуммаОстаток,
      КоличествоОстаток
ИЗ
      РегистрБухгалтерии.Управленческий.Остатки КАК УправленческийОстатки
```

Счет	Субконто1	Субконто2	СуммаОстаток	КоличествоОстаток
Товары опт	Поварешка	Юг		1,000
Расчеты по краткосрочн...			200 000,00	
Расчеты с учредителями			800 000,00	
Товары опт	Поварешка	Север		1,000
Товары опт	Поварешка		300,00	
Товары	Ложка	Юг	100,00	10,000
Расчеты с поставщиками	Поставщик Иванов		-600,00	
Товары	Вилка	Юг	200,00	10,000
Уставный капитал			-1 000 000,00	

Ограничим получение остатков конкретным счетом («Товары опт»):

```
ВЫБРАТЬ
      Счет,
      Субконто1,
      Субконто2,
      СуммаОстаток,
      КоличествоОстаток
ИЗ
      РегистрБухгалтерии.Управленческий.Остатки ( , Счет = &Счет) КАК УправленческийОстатки
```

Счет	Субконто1	Субконто2	СуммаОстаток	КоличествоОстаток
Товары опт	Поварешка	Юг		1,000
Товары опт	Поварешка	Север		1,000
Товары опт	Поварешка		300,00	

Теперь надо сгруппировать записи, но если их просто сгруппировать, то не узнать, какое количество товара хранится на конкретном складе. Поэтому, перед группировкой опишем еще одно (условное) поле:

```
ВЫБРАТЬ
      Счет,
      Субконто1,
      Субконто2,
      СуммаОстаток,
      КоличествоОстаток,
      ВЫБОР КОГДА Субконто2 = &Склад ТОГДА КоличествоОстаток КОНЕЦ
ИЗ
      РегистрБухгалтерии.Управленческий.Остатки ( , Счет = &Счет) КАК УправленческийОстатки
```

В качестве параметра «Склад» выберем склад «ЮГ».

Счет	Субконто1	Субконто2	СуммаОстаток	КоличествоОстаток	Поле1
Товары опт	Поварешка	Юг		1,000	1,000
Товары опт	Поварешка	Север		1,000	
Товары опт	Поварешка		300,00		

Теперь можно сгруппировать поля. Поля Субконто2 и Счет в запросе уже не нужны:

```
ВЫБРАТЬ
	Субконто1,
	СУММА(СуммаОстаток) КАК СуммаОстаток,
	СУММА(КоличествоОстаток) КАК КоличествоОстаток,
	СУММА(ВЫБОР КОГДА Субконто2 = &Склад ТОГДА КоличествоОстаток КОНЕЦ) КАК ОстатокНаСкладе
ИЗ
	РегистрБухгалтерии.Управленческий.Остатки ( , Счет = &Счет) КАК УправленческийОстатки
СГРУППИРОВАТЬ ПО Субконто1
```

Субконто1	СуммаОстаток	КоличествоОстаток	ОстатокНаСкладе
Поварешка	300	2	1

Оптимизируем запрос:

```
ВЫБРАТЬ
	Субконто1,
	СУММА(СуммаОстаток) КАК СуммаОстаток,
	СУММА(КоличествоОстаток) КАК КоличествоОстаток,
	СУММА(ВЫБОР КОГДА Субконто2 = &Склад ТОГДА КоличествоОстаток КОНЕЦ) КАК ОстатокНаСкладе
ИЗ
	РегистрБухгалтерии.Управленческий.Остатки (&МоментВремени, Счет = &Счет, &ВидыСубконто)
КАК УправленческийОстатки
СГРУППИРОВАТЬ ПО Субконто1
```

Зададим значения параметра «ВидыСубконто» (сначала установим тип выбираемых значений как «Список значений», тип каждого значения установим как «Вид субконто»):

- Номенклатура.
- Склады.

Значение параметра «МоментВремени» укажем в коде, а пока оно будет «11.01.2013 23:59:59».

Данный запрос максимально эффективно получает нужные значения.

Объединим запросы:
Поправим первый запрос:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка
СГРУППИРОВАТЬ ПО Номенклатура
```

Номенклатура	Количество
Поварешка	1

Результат запишем во временную таблицу:

```
ВЫБРАТЬ
	Номенклатура,
	СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары
ГДЕ
	Ссылка = &Ссылка
СГРУППИРОВАТЬ ПО Номенклатура
```

Вставим в запрос таблицу, полученную по данным регистра бухгалтерии, и тоже поместим ее во временную таблицу:

```
ВЫБРАТЬ
     Номенклатура,
     СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары
ГДЕ
     Ссылка = &Ссылка
СГРУППИРОВАТЬ ПО Номенклатура
;
ВЫБРАТЬ
     Субконто1,
     СУММА(СуммаОстаток) КАК СуммаОстаток,
     СУММА(КоличествоОстаток) КАК КоличествоОстаток,
     СУММА(ВЫБОР КОГДА Субконто2 = &Склад ТОГДА КоличествоОстаток КОНЕЦ) КАК ОстатокНаСкладе
ПОМЕСТИТЬ Остатки
ИЗ
     РегистрБухгалтерии.Управленческий.Остатки (&МоментВремени, Счет = &Счет, &ВидыСубконто) КАК УправленческийОстатки
СГРУППИРОВАТЬ ПО Субконто1
```

В параметрах второго запроса укажем условие по Субконто – на список товаров. Чтобы остатки рассчитывались только на те товары, которые указаны в документе:

```
ВЫБРАТЬ
     Номенклатура,
     СУММА(Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары
ГДЕ
     Ссылка = &Ссылка
СГРУППИРОВАТЬ ПО Номенклатура
```

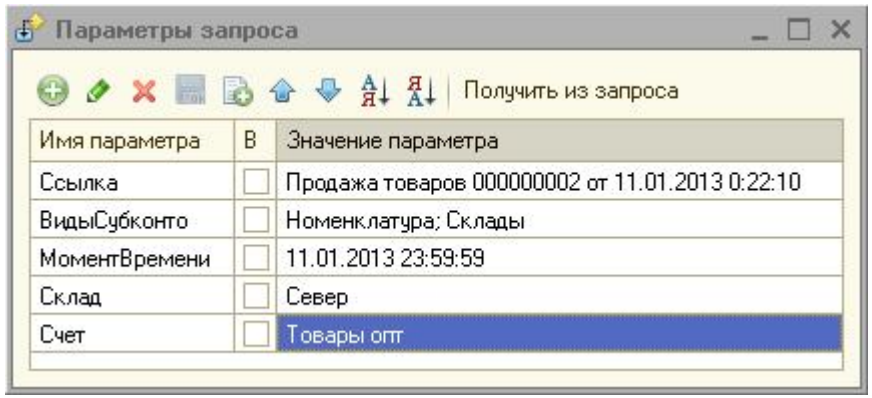
```
;
ВЫБРАТЬ
     Субконто1,
     СУММА(СуммаОстаток) КАК СуммаОстаток,
     СУММА(КоличествоОстаток) КАК КоличествоОстаток,
     СУММА(ВЫБОР КОГДА Субконто2 = &Склад ТОГДА КоличествоОстаток КОНЕЦ) КАК ОстатокНаСкладе
ПОМЕСТИТЬ Остатки
ИЗ
     РегистрБухгалтерии.Управленческий.Остатки (&МоментВремени, Счет = &Счет, &ВидыСубконто,
Субконто1 В (ВЫБРАТЬ Номенклатура ИЗ ДокТЧ КАК ДокТЧ)) КАК УправленческийОстатки
СГРУППИРОВАТЬ ПО Субконто1
```

Соединим данные в третьем запросе:

```
ВЫБРАТЬ
     РасходнаяТовары.Номенклатура,
     СУММА(РасходнаяТовары.Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
     Документ.Расходная.Товары КАК РасходнаяТовары
ГДЕ
     РасходнаяТовары.Ссылка = &Ссылка

СГРУППИРОВАТЬ ПО
     РасходнаяТовары.Номенклатура
;
ВЫБРАТЬ
     УправленческийОстатки.Субконто1,
     СУММА(УправленческийОстатки.СуммаОстаток) КАК СуммаОстаток,
     СУММА(УправленческийОстатки.КоличествоОстаток) КАК КоличествоОстаток,
     СУММА(ВЫБОР
               КОГДА УправленческийОстатки.Субконто2 = &Склад
                    ТОГДА УправленческийОстатки.КоличествоОстаток
          КОНЕЦ) КАК ОстатокНаСкладе
ПОМЕСТИТЬ Остатки
```

```
ИЗ
      РегистрБухгалтерии.Управленческий.Остатки(
                  &МоментВремени,
                  Счет = &Счет,
                  &ВидыСубконто,
                  Субконто1 В
                        (ВЫБРАТЬ
                              ДокТЧ.Номенклатура
                        ИЗ
                              ДокТЧ КАК ДокТЧ)) КАК УправленческийОстатки

СГРУППИРОВАТЬ ПО
      УправленческийОстатки.Субконто1
;
ВЫБРАТЬ
      *
ИЗ
      ДокТЧ КАК ДокТЧ
            ЛЕВОЕ СОЕДИНЕНИЕ Остатки КАК Остатки
            ПО ДокТЧ.Номенклатура = Остатки.Субконто1
```

Установим параметры запроса:



Результат:

Номенклатура	Количество	Субконто1	СуммаОстаток	КоличествоОстаток	ОстатокНаСкладе
Поварешка	1	Поварешка	300	2	1

Поле	Источник
Номенклатура	Документ
Количество	
Субконто1	Таблица остатков
СуммаОстаток	Итоговые значения из таблицы остатков
КоличествоОстаток	
ОстатокНаСкладе	Рассчитываемое поле. Остаток на конкретном, указанном складе

Вместо «*» опишем получаемые поля:

```
ВЫБРАТЬ
	РасходнаяТовары.Номенклатура,
	СУММА(РасходнаяТовары.Количество) КАК Количество
ПОМЕСТИТЬ ДокТЧ
ИЗ
	Документ.Расходная.Товары КАК РасходнаяТовары
ГДЕ
	РасходнаяТовары.Ссылка = &Ссылка

СГРУППИРОВАТЬ ПО
	РасходнаяТовары.Номенклатура
;
ВЫБРАТЬ
	УправленческийОстатки.Субконто1,
	СУММА(УправленческийОстатки.СуммаОстаток) КАК СуммаОстаток,
	СУММА(УправленческийОстатки.КоличествоОстаток) КАК КоличествоОстаток,
	СУММА(ВЫБОР
			КОГДА УправленческийОстатки.Субконто2 = &Склад
				ТОГДА УправленческийОстатки.КоличествоОстаток
```

```
          КОНЕЦ) КАК ОстатокНаСкладе
ПОМЕСТИТЬ Остатки
ИЗ
     РегистрБухгалтерии.Управленческий.Остатки(
               &МоментВремени,
               Счет = &Счет,
               &ВидыСубконто,
               Субконто1 В
                    (ВЫБРАТЬ
                         ДокТЧ.Номенклатура
                    ИЗ
                         ДокТЧ КАК ДокТЧ)) КАК УправленческийОстатки

СГРУППИРОВАТЬ ПО
     УправленческийОстатки.Субконто1
;
ВЫБРАТЬ
     ДокТЧ.Номенклатура,
     ДокТЧ.Количество,
     Остатки.КоличествоОстаток,
     Остатки.СуммаОстаток,
     ЕСТЬNULL(Остатки.ОстатокНаСкладе, 0) КАК ОстатокНаСкладе
ИЗ
     ДокТЧ КАК ДокТЧ
          ЛЕВОЕ СОЕДИНЕНИЕ Остатки КАК Остатки
          ПО ДокТЧ.Номенклатура = Остатки.Субконто1
```

Номенклатура	Количество	КоличествоОстаток	СуммаОстаток	ОстатокНаСкладе
Поварешка	1	2	300	1

В документе Расходная опишем альтернативное проведение:

Код из процедуры «ОбработкаПроведения» перенесем в процедуры:

- «ПроведениеРозница»
- «ПроведениеОпт» - изменим запрос и поправим код.

Добавим реквизит «ТипОперации», аналогичный одноименному реквизиту из документа Приходная.

Итого код:

```
Процедура ОбработкаПроведения(Отказ, Режим)
	Если ТипОперации = 0 Тогда
		ПроведениеРозница(Отказ);
	Иначе
		ПроведениеОпт(Отказ);
	КонецЕсли;
КонецПроцедуры

Процедура ПроведениеОпт(Отказ)
	Запрос = Новый Запрос;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	РасходнаяТовары.Номенклатура,
	|	СУММА(РасходнаяТовары.Количество) КАК Количество
	|ПОМЕСТИТЬ ДокТЧ
	|ИЗ
	|	Документ.Расходная.Товары КАК РасходнаяТовары
	|ГДЕ
	|	РасходнаяТовары.Ссылка = &Ссылка
	|
	|СГРУППИРОВАТЬ ПО
	|	РасходнаяТовары.Номенклатура
	|;
	|
	|ВЫБРАТЬ
```

```
|    УправленческийОстатки.Субконто1,
|    СУММА(УправленческийОстатки.СуммаОстаток) КАК СуммаОстаток,
|    СУММА(УправленческийОстатки.КоличествоОстаток) КАК КоличествоОстаток,
|    СУММА(ВЫБОР
|              КОГДА УправленческийОстатки.Субконто2 = &Склад
|                   ТОГДА УправленческийОстатки.КоличествоОстаток
|         КОНЕЦ) КАК ОстатокНаСкладе
|ПОМЕСТИТЬ Остатки
|ИЗ
|    РегистрБухгалтерии.Управленческий.Остатки(
|              &МоментВремени,
|              Счет = &Счет,
|              &МассивСубконто,
|              Субконто1 В
|                   (ВЫБРАТЬ
|                        ДокТЧ.Номенклатура
|                   ИЗ
|                        ДокТЧ КАК ДокТЧ)) КАК УправленческийОстатки
|
|СГРУППИРОВАТЬ ПО
|    УправленческийОстатки.Субконто1
|;
|
|ВЫБРАТЬ
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Количество,
|    Остатки.КоличествоОстаток,
|    Остатки.СуммаОстаток,
|    ЕСТЬNULL(Остатки.ОстатокНаСкладе, 0) КАК ОстатокНаСкладе
|ИЗ
|    ДокТЧ КАК ДокТЧ
|         ЛЕВОЕ СОЕДИНЕНИЕ Остатки КАК Остатки
|         ПО ДокТЧ.Номенклатура = Остатки.Субконто1";

Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.ТоварыОпт);
```

```
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

     Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
          + ". Из необходимых " + Выборка.Количество +
          " в наличии имеется только " + Выборка.КоличествоОстаток;
          Сообщение.Сообщить();

          Движения.Управленческий.Записывать = Ложь;
          Отказ = Истина;
     КонецЕсли;

     Если Отказ Тогда
          Продолжить;
     КонецЕсли;

     Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;

     Движение = Движения.Управленческий.Добавить();
     Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
     Движение.СчетКт = ПланыСчетов.Управленческий.ТоварыОпт;
     Движение.Период = Дата;
     Движение.Сумма = Себестоимость;
```

```
            Движение.КоличествоКт = Выборка.Количество;
            Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
            Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
      КонецЦикла;

      Движение = Движения.Управленческий.Добавить();
      Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
      Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
      Движение.Период = Дата;
      Движение.Сумма = Товары.Итог("Сумма");
      Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
КонецПроцедуры

Процедура ПроведениеРозница(Отказ)
      Запрос = Новый Запрос;
      Запрос.Текст =
      "ВЫБРАТЬ
      |    РасходнаяТовары.Номенклатура,
      |    СУММА(РасходнаяТовары.Количество) КАК Количество
      |ПОМЕСТИТЬ ДокТЧ
      |ИЗ
      |    Документ.Расходная.Товары КАК РасходнаяТовары
      |ГДЕ
      |    РасходнаяТовары.Ссылка = &Ссылка
      |
      |СГРУППИРОВАТЬ ПО
      |    РасходнаяТовары.Номенклатура
      |;
      |
      |ВЫБРАТЬ
      |    ДокТЧ.Номенклатура,
      |    ДокТЧ.Количество,
      |    ЕСТЬNULL(УправленческийОстатки.СуммаОстаток, 0) КАК СуммаОстаток,
      |    ЕСТЬNULL(УправленческийОстатки.КоличествоОстаток, 0) КАК КоличествоОстаток
      |ИЗ
      |    ДокТЧ КАК ДокТЧ
```

```
|          ЛЕВОЕ СОЕДИНЕНИЕ РегистрБухгалтерии.Управленческий.Остатки(
|                    &МоментВремени,
|                    Счет = &Счет,
|                    &МассивСубконто,
|                    Субконто1 В
|                              (ВЫБРАТЬ
|                                   ДокТЧ.Номенклатура
|                              ИЗ
|                                   ДокТЧ КАК ДокТЧ)
|                         И Субконто2 = &Склад) КАК УправленческийОстатки
|          ПО ДокТЧ.Номенклатура = УправленческийОстатки.Субконто1";

Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.Товары);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

     Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
          + ". Из необходимых " + Выборка.Количество +
          " в наличии имеется только " + Выборка.КоличествоОстаток;
          Сообщение.Сообщить();

          Движения.Управленческий.Записывать = Ложь;
```

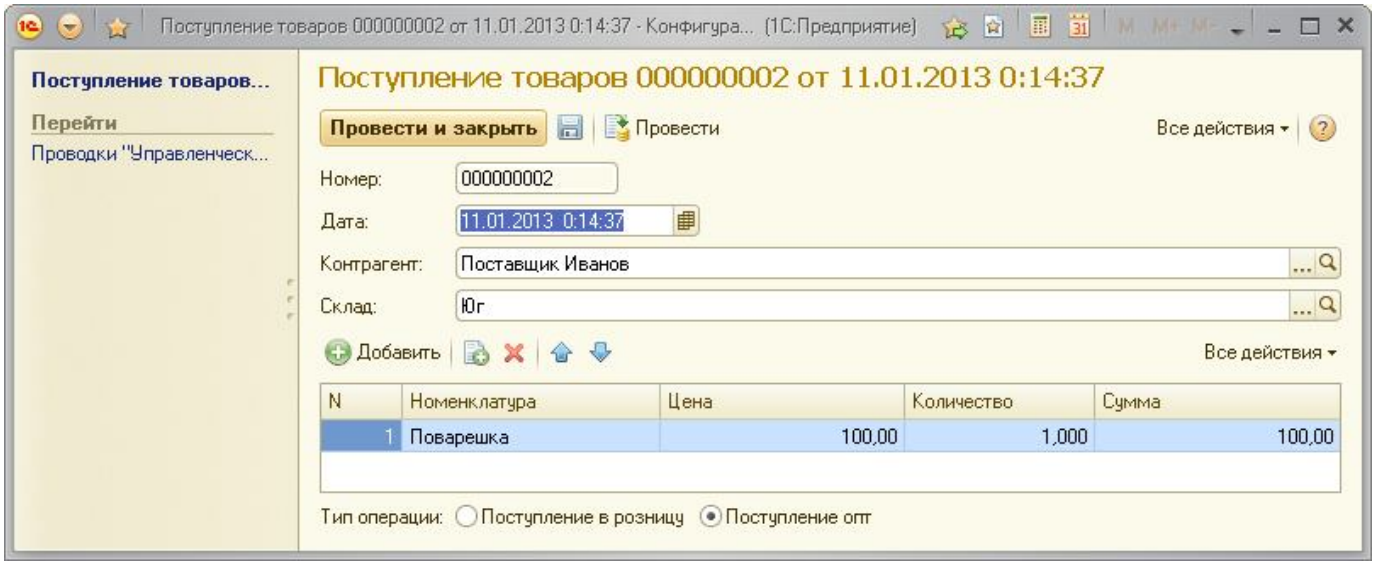
```
            Отказ = Истина;
        КонецЕсли;

        Если Отказ Тогда
            Продолжить;
        КонецЕсли;

        Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
        Движение.СчетКт = ПланыСчетов.Управленческий.Товары;
        Движение.Период = Дата;
        Движение.Сумма = Себестоимость;
        Движение.КоличествоКт = Выборка.Количество;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;
    КонецЦикла;

    Движение = Движения.Управленческий.Добавить();
    Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
    Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
    Движение.Период = Дата;
    Движение.Сумма = Товары.Итог("Сумма");
    Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
КонецПроцедуры
```

Чтобы проверить работоспособность, поправим форму документа Расходная (добавим реквизит «ТипОперации» аналогично документу Приходная):

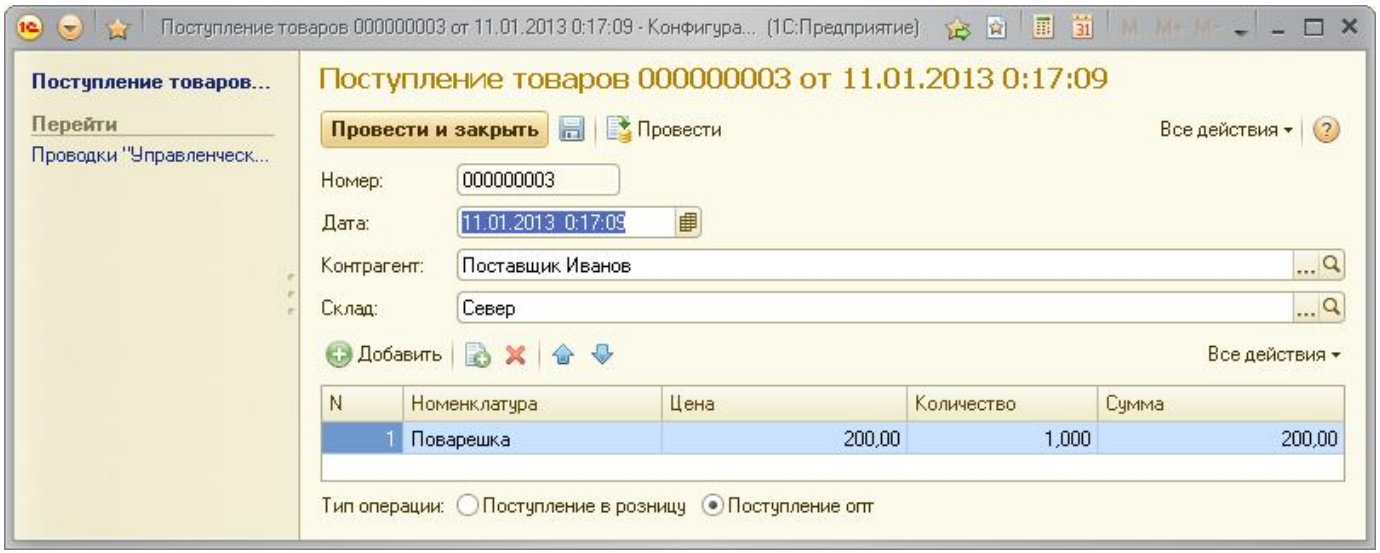
В режиме исполнения:

Купим одинаковые товары по одной стоимости, но на разные Склады и Счета:

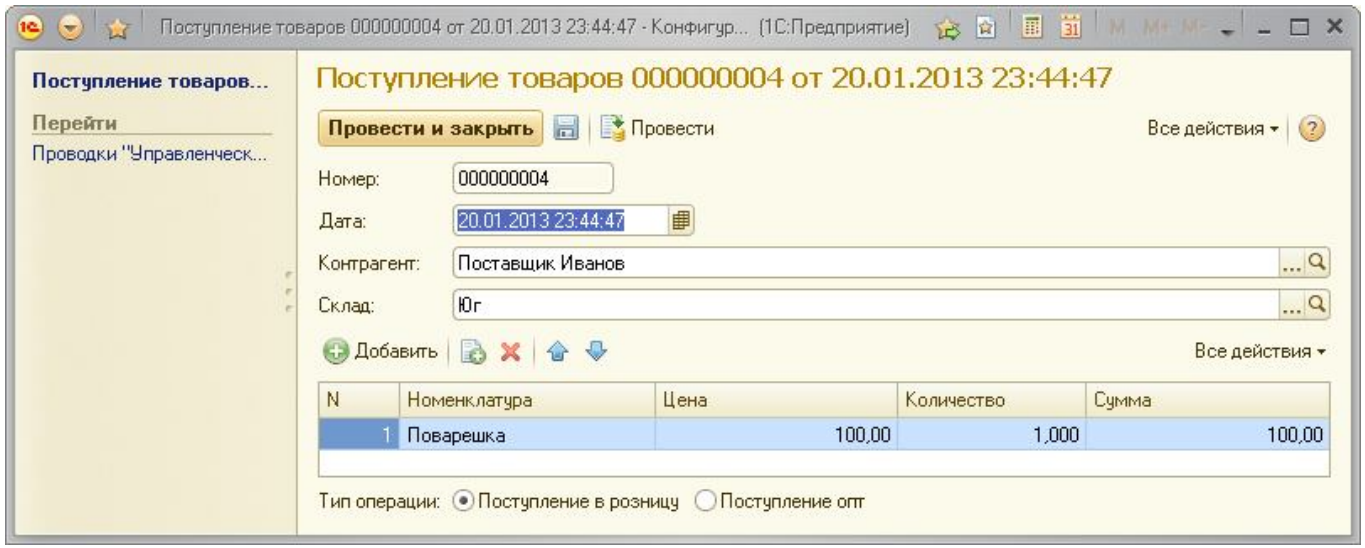
Поварешка, Поступление опт на склад «Юг»

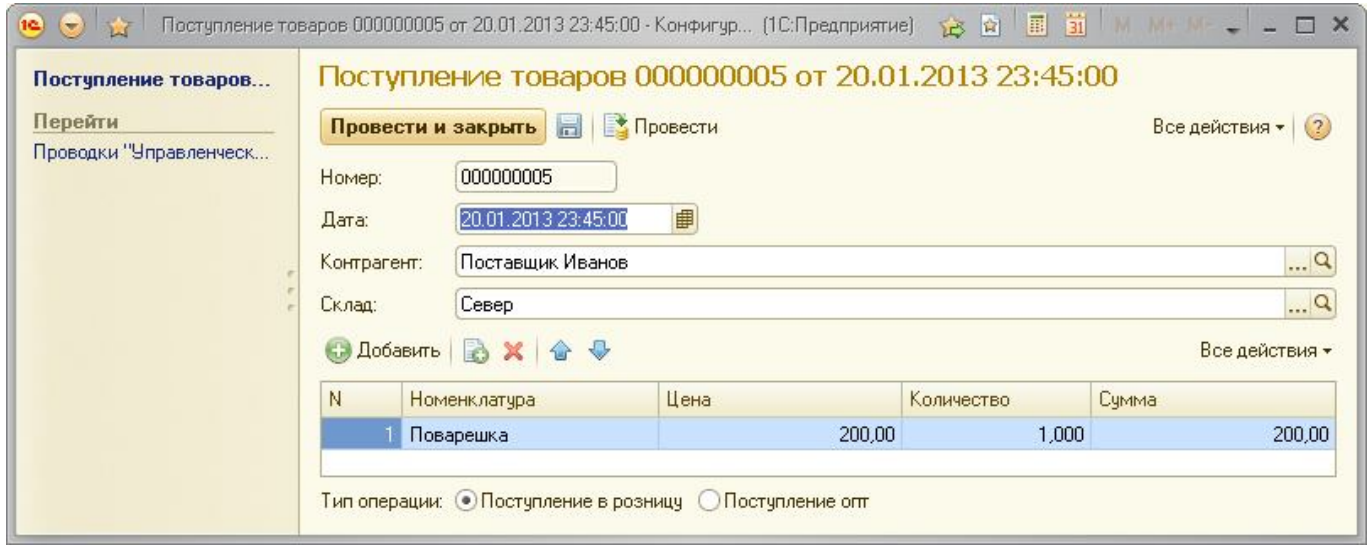


Поварешка, Поступление опт на склад «Север»

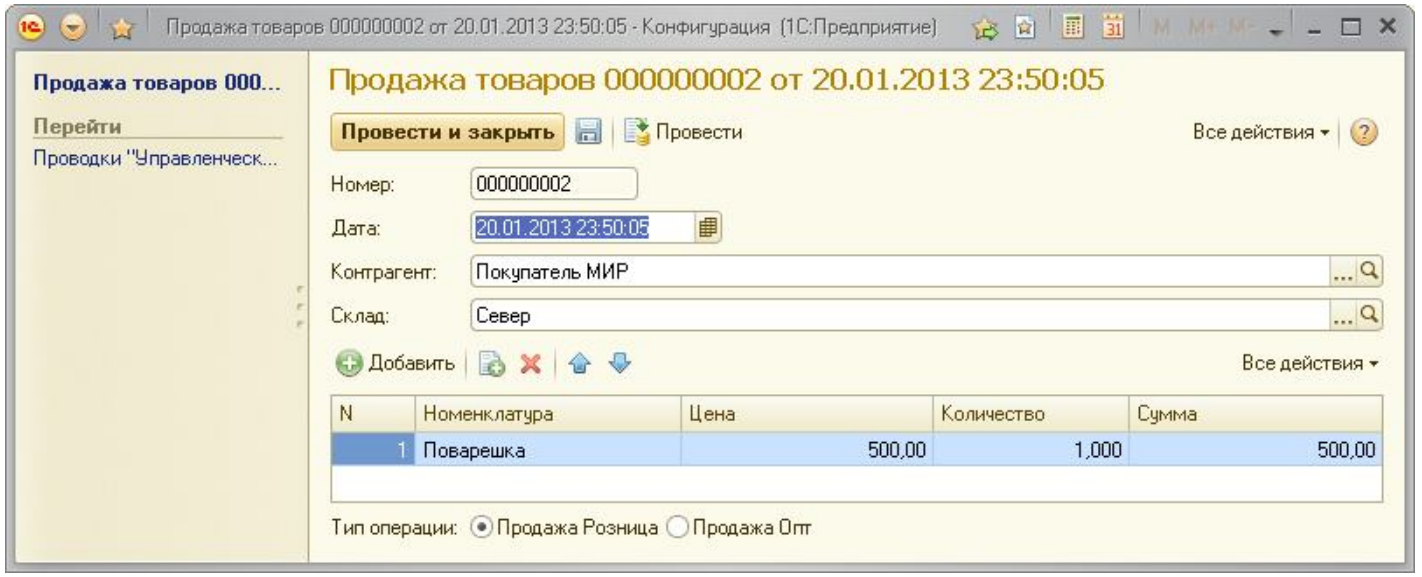


И такие же документы, только по Рознице:

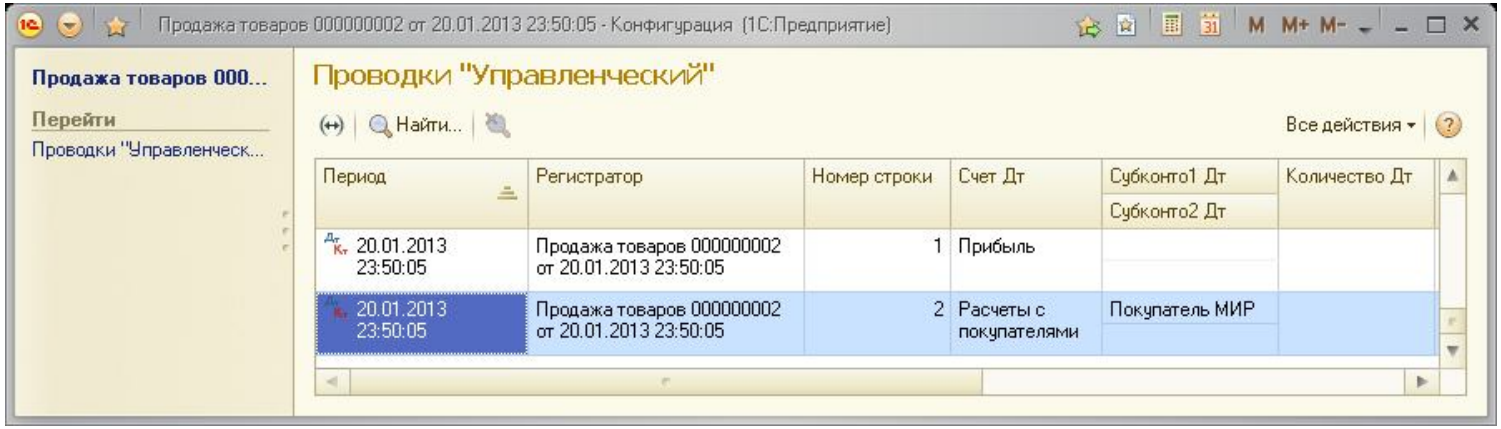


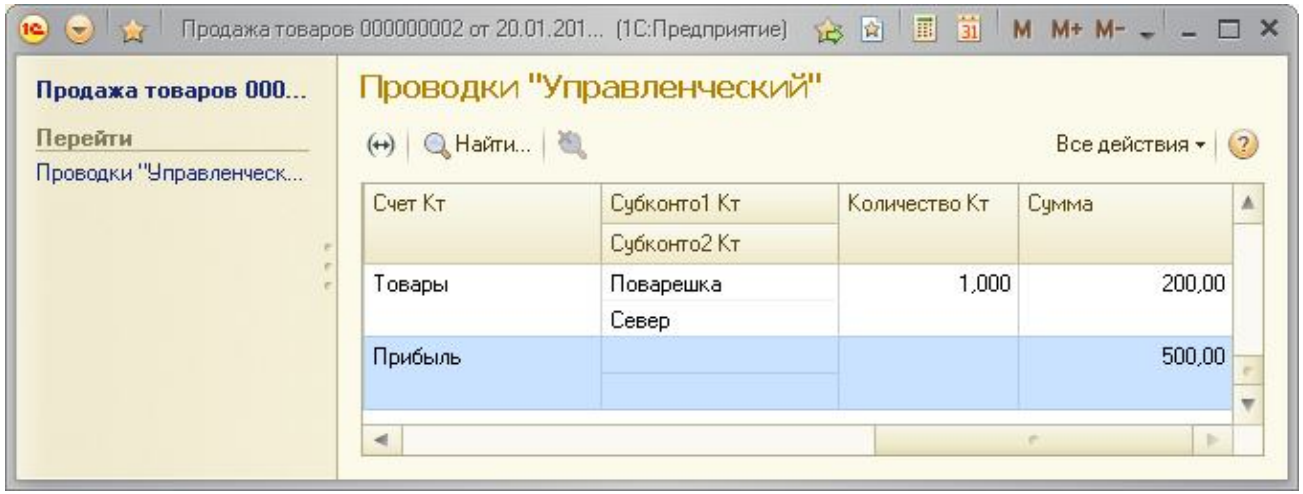


Создадим документы Расходная:
При продаже в розницу:



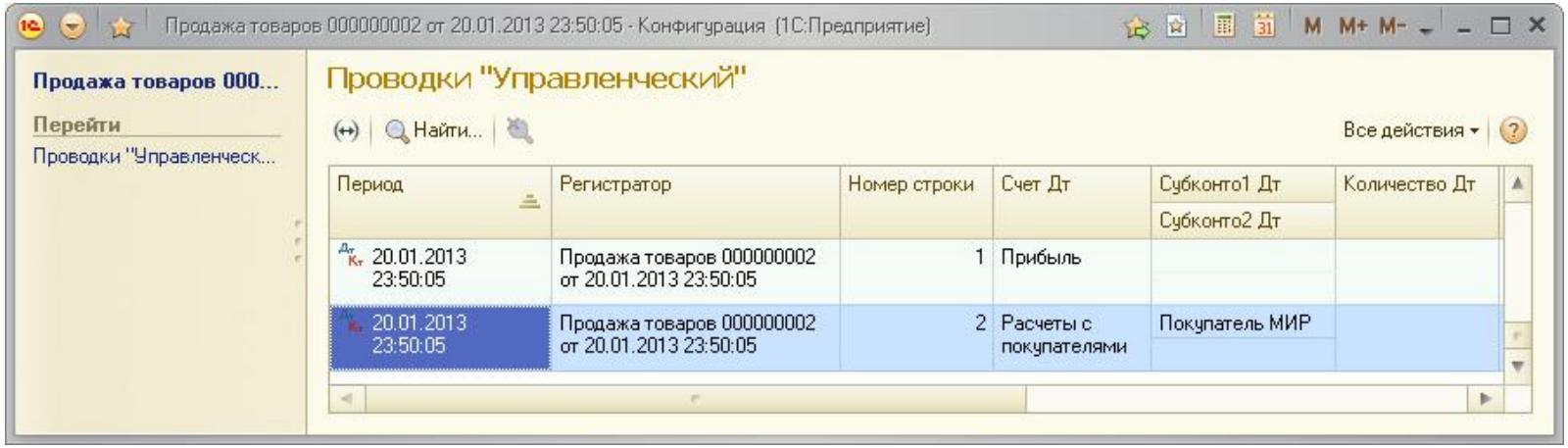
Результат проведения:

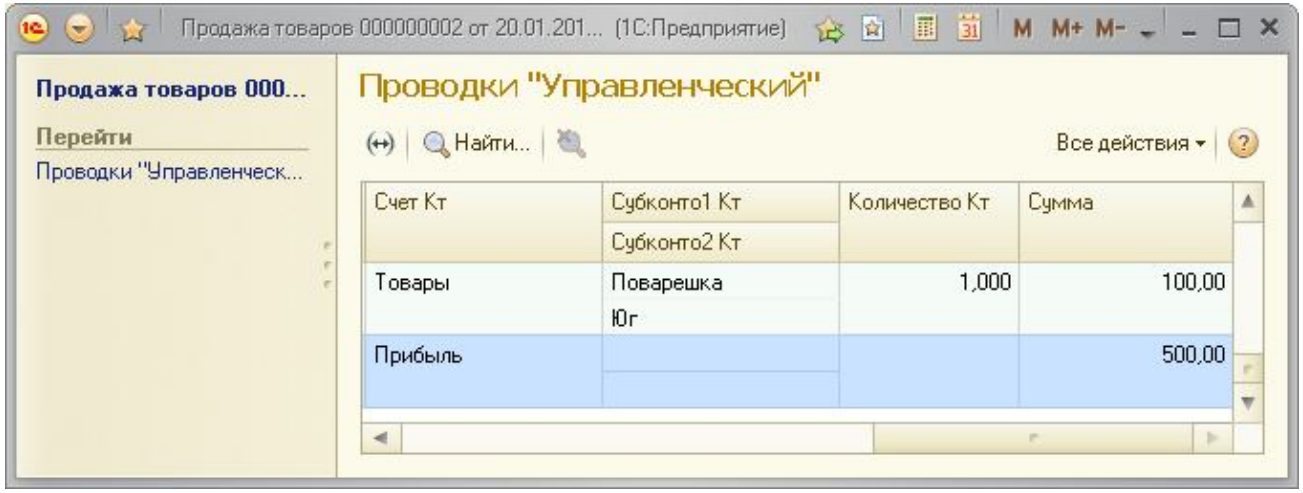




Себестоимость Поварешки 200 рублей.

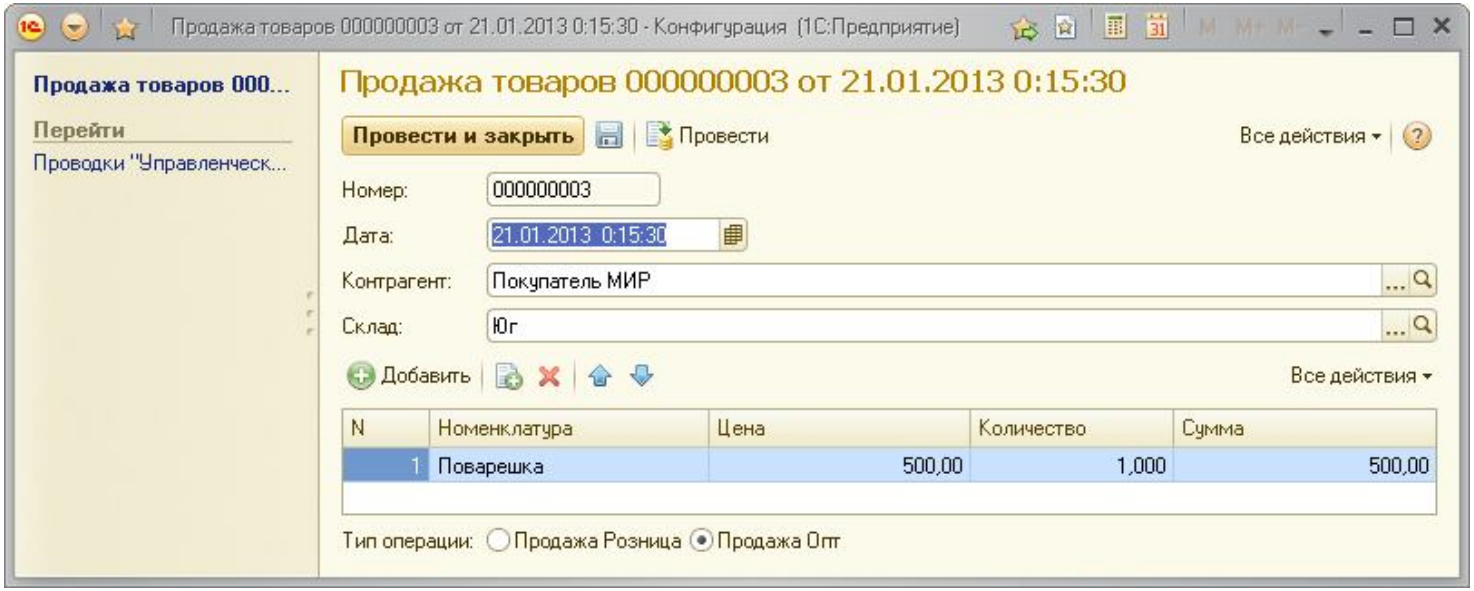
Поменяем Склад на «Юг»:



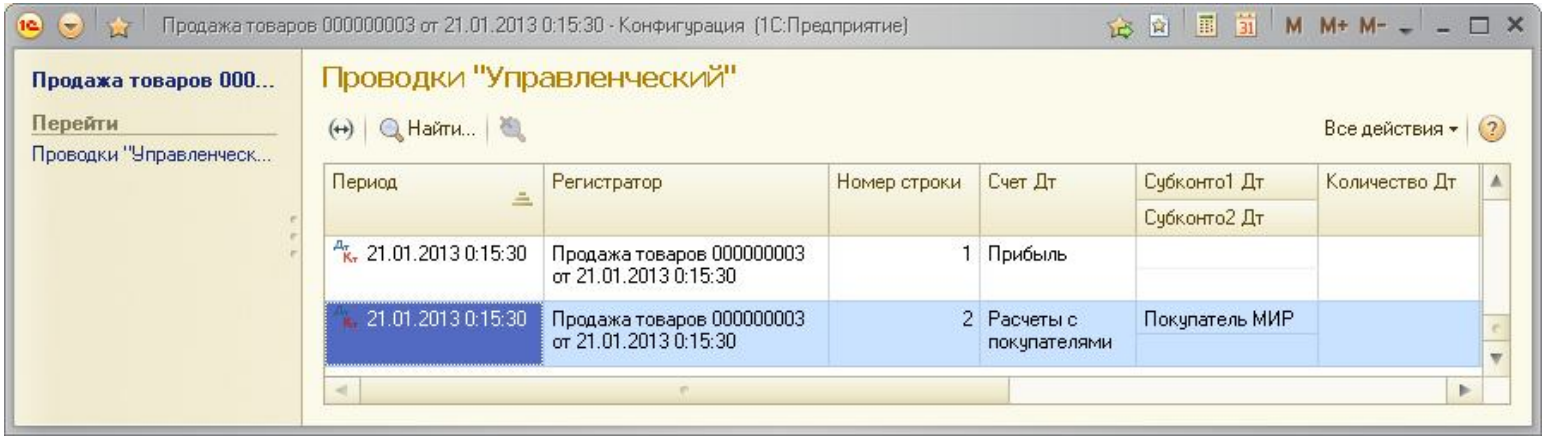


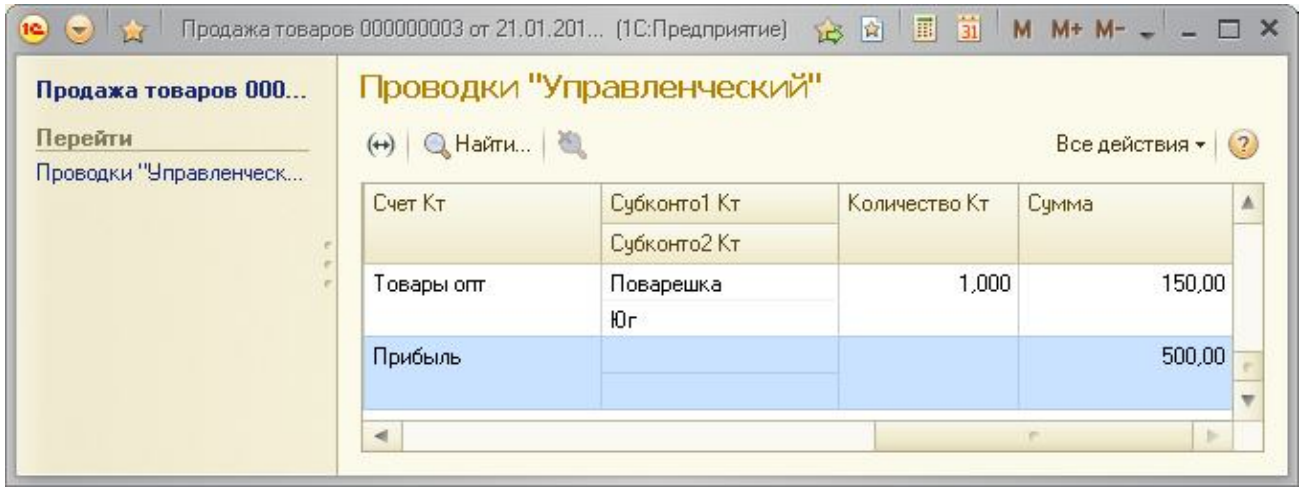
Себестоимость Поварешки 100 рублей.

При продаже Оптом:



Независимо от указанного Склада, Себестоимость всегда будет равна 150 рублей.





Занятие 2.23

Цель: для управленческих целей по некоторым счетам надо вести учет (по взаиморасчетам) в валюте. Валютный учет ведется чисто для Управленческих целей. Регламентированный учет все равно ведется в национальной валюте.

Создадим справочник «Валюты»:

Свойство	Значение
Имя	Валюты
Синоним	Валюты

Создадим регистр сведений «Курсы валют»:

Свойство	Значение
Имя	КурсыВалют
Синоним	Курсы валют
Периодичность	В пределах секунды
Закладка «Данные»	
Измерения	
Валюта	
Имя	Валюта
Синоним	Валюта
Тип	СправочникСсылка.Валюты
Ведущее	Истина
Запрет незаполненных значений	Истина
Проверка заполнения	Выдавать ошибку
Ресурсы	
Курс	
Имя	Курс
Синоним	Курс
Тип	Число 10,4
Проверка заполнения	Выдавать ошибку

Самая частая процедура с валютой – это пересчет (национальной валюты в иностранную и наоборот). Данная процедура будет нужна в различных объектах, поэтому опишем ее в Общем модуле:

Модуль «Общие механизмы»:

Свойство	Значение
Имя	ОбщиеМеханизмы
Синоним	Общие механизмы
Сервер	Истина
Вызов сервера	Истина

Повторное использование возвращаемых значений – например, есть функция, расположенная в модуле, пересчитывающая 50 долларов в евро. Ее вызвали, она отработала и возвратила некоторое значение. При повторном вызове, функция не отработает, а возвратит уже рассчитанное ранее значение. При условии, что входящие параметры будут равны предыдущим.

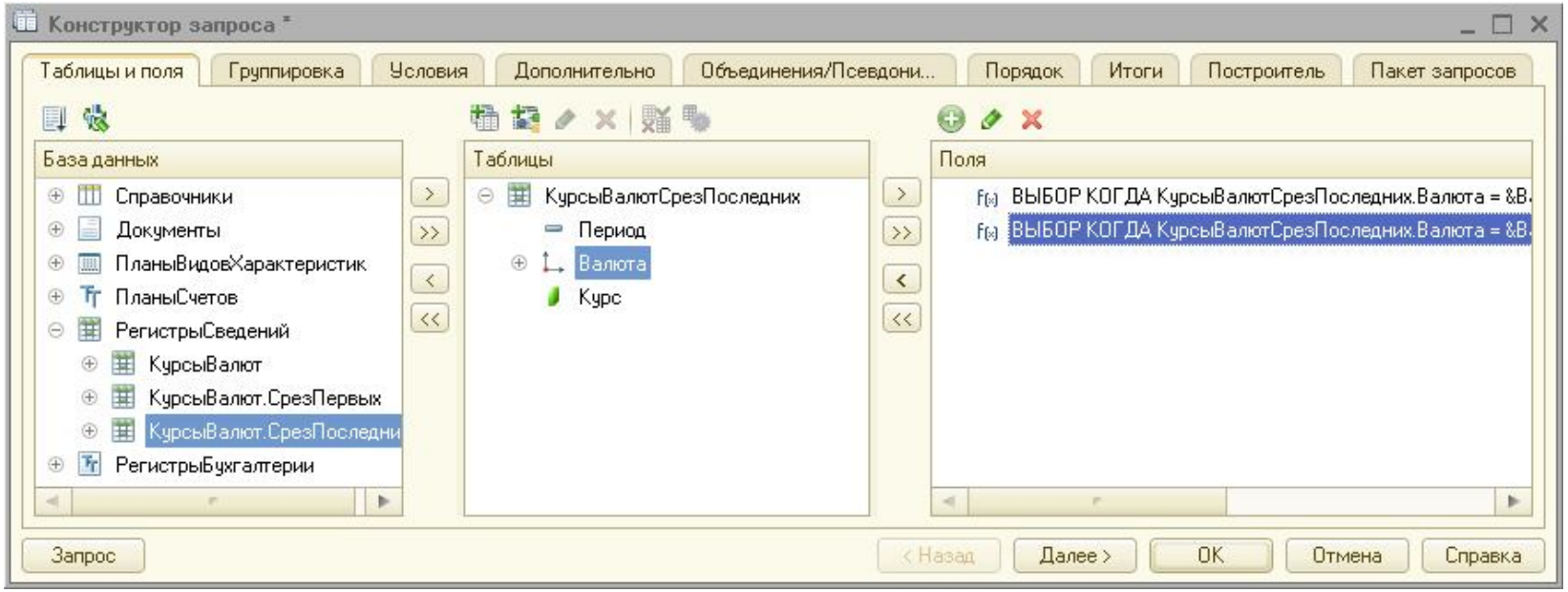
Но нам это не нужно, т.к. Курс с течением времени может поменяться.

Функция:
Параметры функции:
Сумма – сумма, которую будем пересчитывать.
ВалютаИз – валюта, которую будем пересчитывать.
ВалютаВ – валюта, в которую будем пересчитывать.
Дата – дата совершения операции.
Обязательным реквизитом будет только «Сумма», остальные сделаем дифференцированными («`ИмяПараметра=Неопределено`»).

Т.к. в функции дифференцированные параметры будут переопределены (в случае, если они не переданы), то при вызове функции будем получать не ссылки на параметры, а значения параметров – ключевое слово «`Знач`». Это дает возможность задавать определенные условия, например:

```
Дата = ?(Дата = Неопределено, ТекущаяДата(), Дата);
```

Опишем запрос:

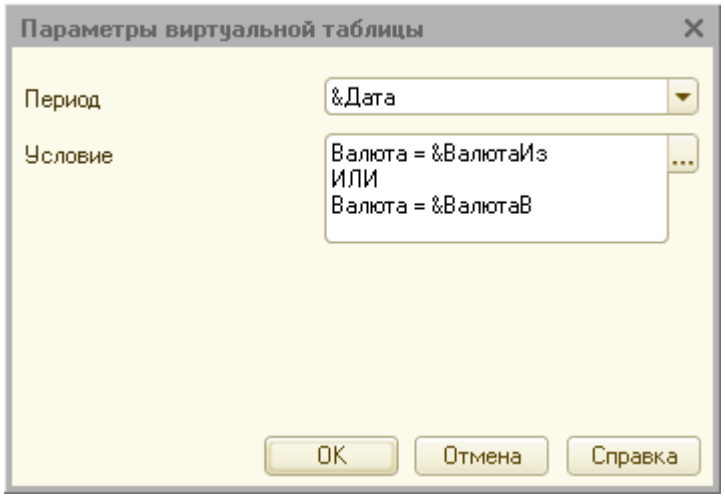


Поля:

```
ВЫБОР
     КОГДА КурсыВалютСрезПоследних.Валюта = &ВалютаИз
          ТОГДА КурсыВалютСрезПоследних.Курс
     ИНАЧЕ 0
КОНЕЦ
```

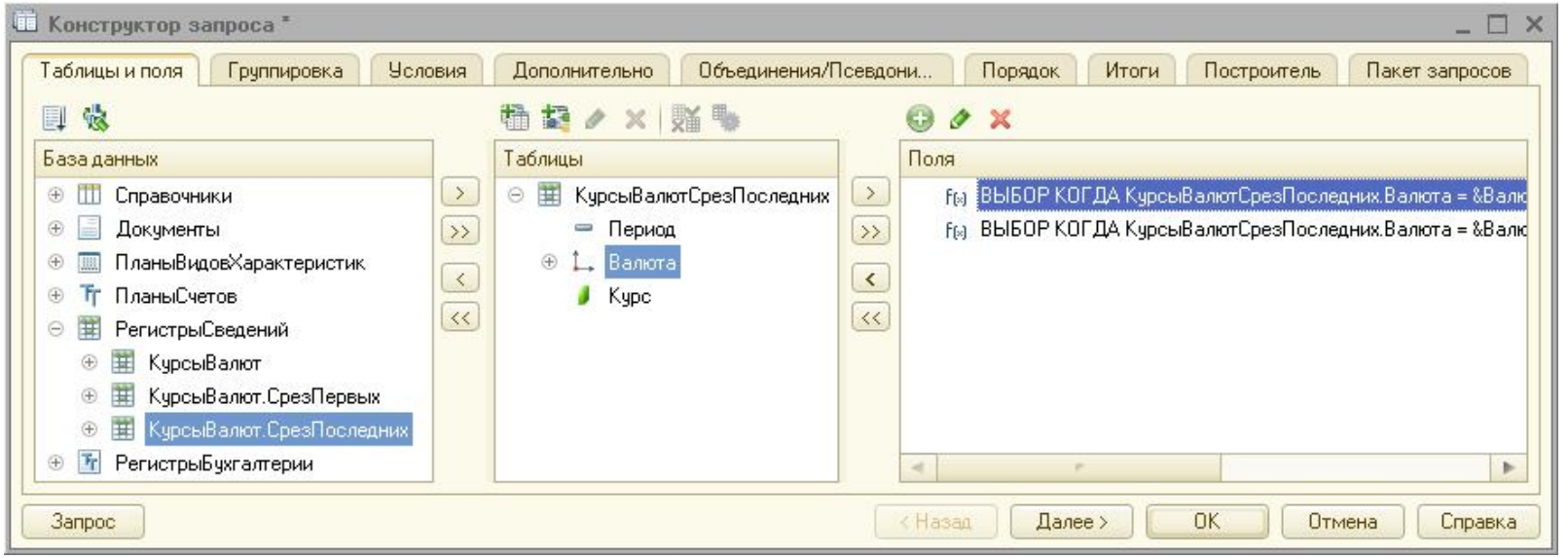
```
ВЫБОР
     КОГДА КурсыВалютСрезПоследних.Валюта = &ВалютаВ
          ТОГДА КурсыВалютСрезПоследних.Курс
     ИНАЧЕ 0
КОНЕЦ
```

Опишем условия на таблицу:

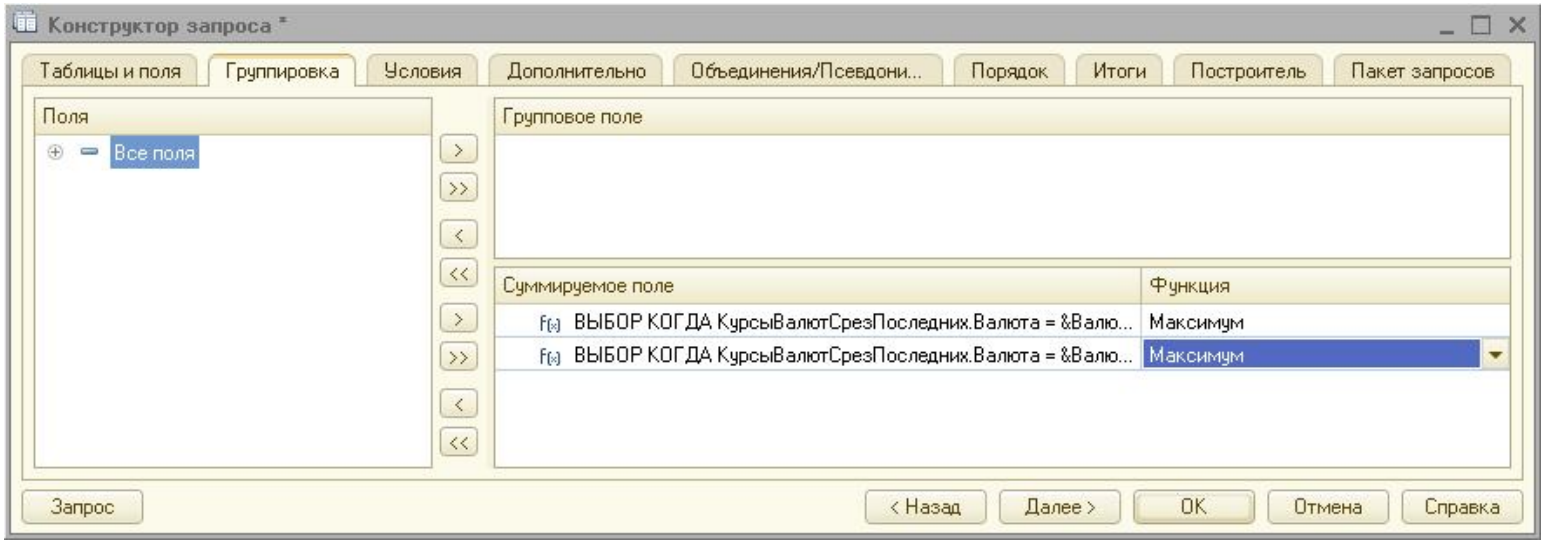


Получатся строки:

Курс	0
0	Курс

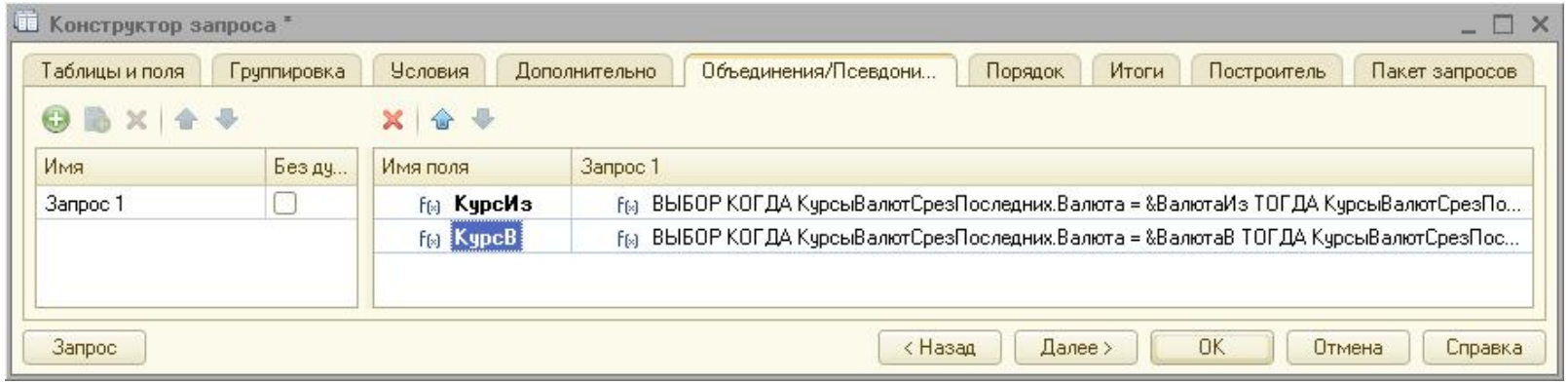


Сгруппируем полученные поля:



Выбрали «Максимум», т.к. в значениях получается «0» или «Курс», т.е. итог «Курс», а «Максимум» работает быстрее, чем «Сумма».

Переименуем поля:



Запрос готов. Итого код:

```
Функция ИзВалютыВВалюту(Сумма, Знач ВалютаИз = Неопределено,
      Знач ВалютаВ = Неопределено,
      Знач Дата = Неопределено) Экспорт
      Дата = ?(Дата = Неопределено, ТекущаяДата(), Дата);
      Запрос = Новый Запрос;
      Запрос.Текст =
      "ВЫБРАТЬ
      |     МАКСИМУМ(ВЫБОР
      |               КОГДА КурсыВалютСрезПоследних.Валюта = &ВалютаИз
      |                    ТОГДА КурсыВалютСрезПоследних.Курс
      |               ИНАЧЕ 0
      |          КОНЕЦ) КАК КурсИз,
      |     МАКСИМУМ(ВЫБОР
      |               КОГДА КурсыВалютСрезПоследних.Валюта = &ВалютаВ
      |                    ТОГДА КурсыВалютСрезПоследних.Курс
      |               ИНАЧЕ 0
      |          КОНЕЦ) КАК КурсВ
      |ИЗ
      |     РегистрСведений.КурсыВалют.СрезПоследних(
      |               &Дата,
      |               Валюта = &ВалютаИз
```

```
|                    ИЛИ Валюта = &ВалютаВ) КАК КурсыВалютСрезПоследних"

Запрос.УстановитьПараметр("ВалютаИЗ", ВалютаИз);
Запрос.УстановитьПараметр("ВалютаВ", ВалютаВ);
Запрос.УстановитьПараметр("Дата", Дата);

РезультатЗапроса = Запрос.Выполнить();
Если РезультатЗапроса.Пустой() Тогда
     Возврат Сумма;
КонецЕсли;

Данные = РезультатЗапроса.Выбрать();
Данные.Следующий();
Если ВалютаИз = Неопределено Тогда
     КурсИЗ = 1;
Иначе
     КурсИЗ = Данные.КурсИЗ;
КонецЕсли;

КурсВ = ?(ВалютаВ = Неопределено, 1, Данные.КурсВ);

Возврат Сумма * КурсИЗ / КурсВ;
```

КонецФункции // ИзВалютыВВалюту()

Реализуем механизмы бухгалтерских задач, связанные с валютным учетом.

Для хранения Суммы в валюте, добавим в Регистр бухгалтерии реквизит «Вал сумма»:

Свойство	Значение
Имя	ВалСумма
Синоним	Вал сумма
Тип	Число 15, 2

Не все счета будут являться валютными, т.е. хранить Сумму в валюте, поэтому в План счетов добавим Признак учета «Валютный»:

Свойство	Значение
Имя	Валютный
Синоним	Валютный
Тип	Булево

При этом в Регистре бухгалтерии у реквизита «ВалСумма» заполним еще одно свойство:

Свойство	Значение
Признак учета	Валютный

Это означает, что по этому ресурсу данные надо накапливать только по счетам с признаком «Валютный».

Свойство	Значение
Балансовый	Ложь

Т.е. допускается ситуация, когда по Дт(Кт) будет валютная сумма, а по Кт(Дт) будет ноль. Например, при покупке валюты.

«ВалСумма» есть, но не понятно в какой валюте (доллары, евро, франки...). Чтобы разделить Ресурс на разрезы, надо создать Измерение. Создадим Измерение Регистра бухгалтерии «Валюта»:

Свойство	Значение
Имя	Валюта
Синоним	Валюта
Тип	СправочникСсылка.Валюты
Запрет незаполненных значений	Истина
Признак учета	Валютный
Балансовый	Ложь

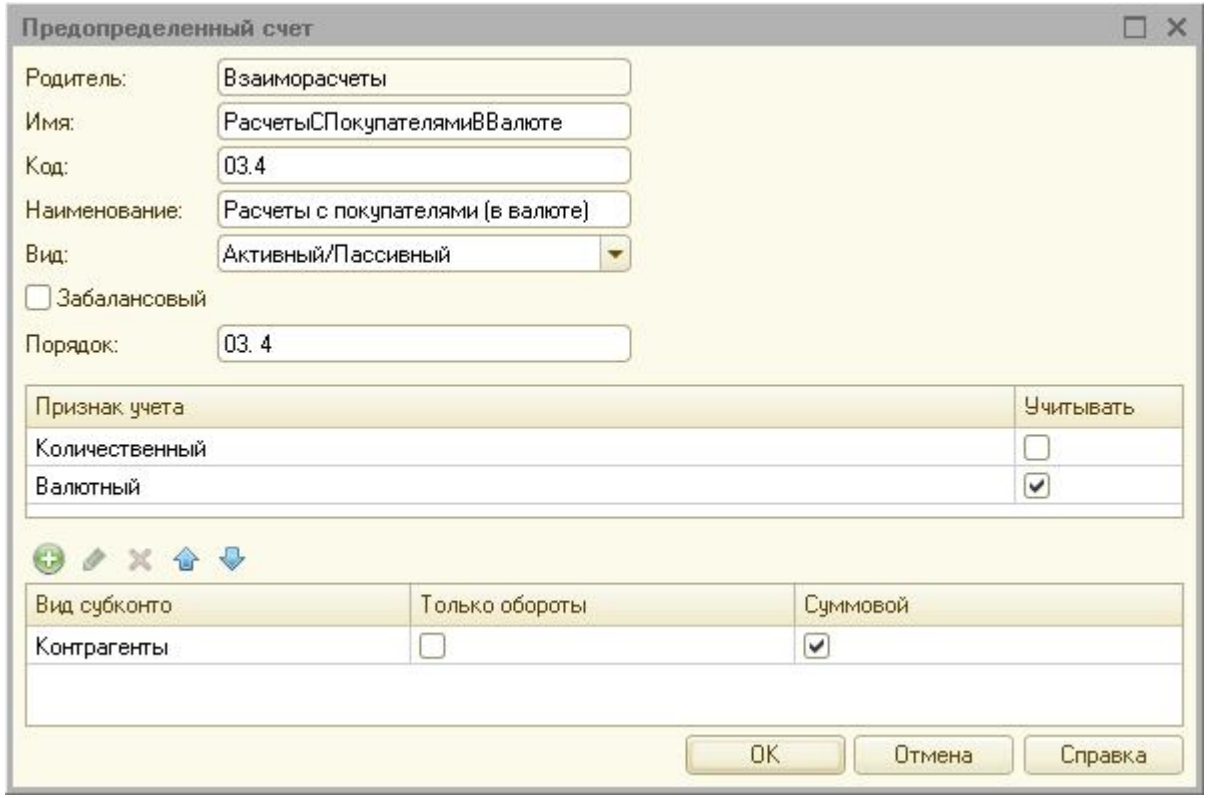
«Признак учета - Валютный» - чтобы значение данного разреза действовало только на те счета, по которым ведется валютный учет.

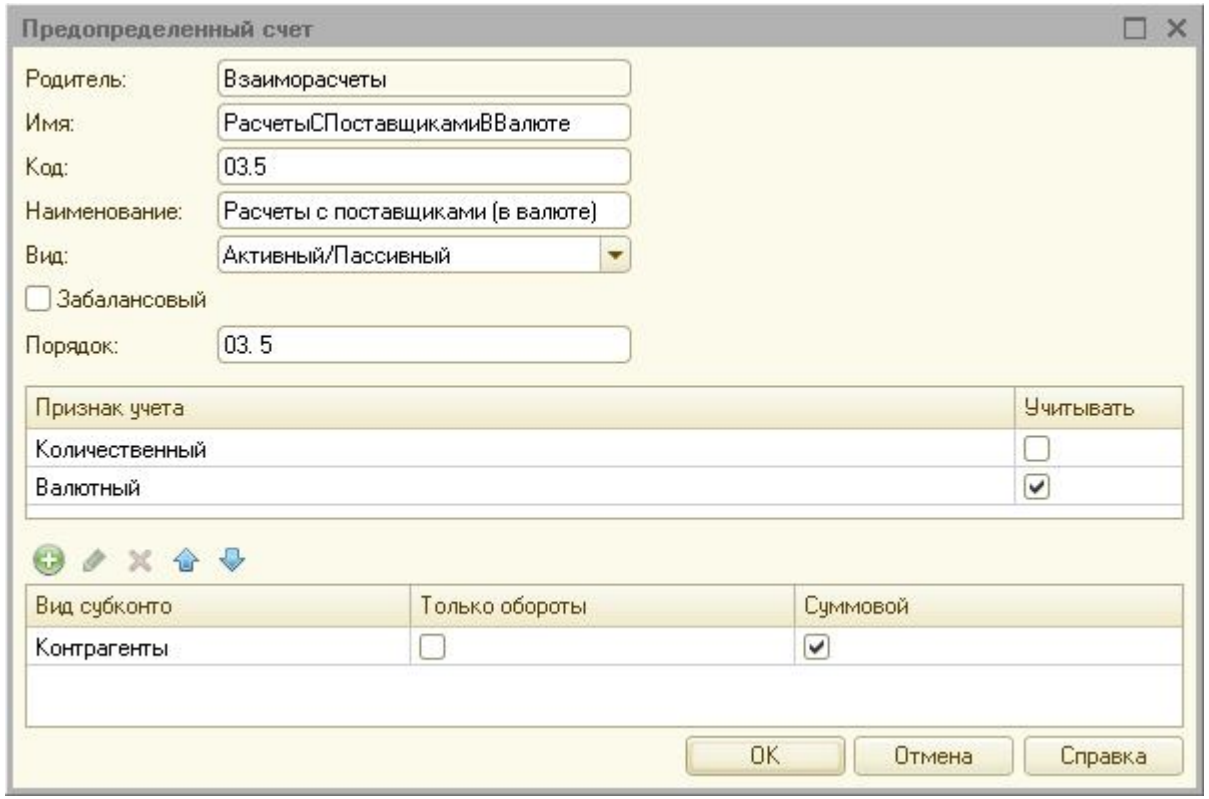
Чаще всего, если Измерение связанно с Ресурсом, то настройки:

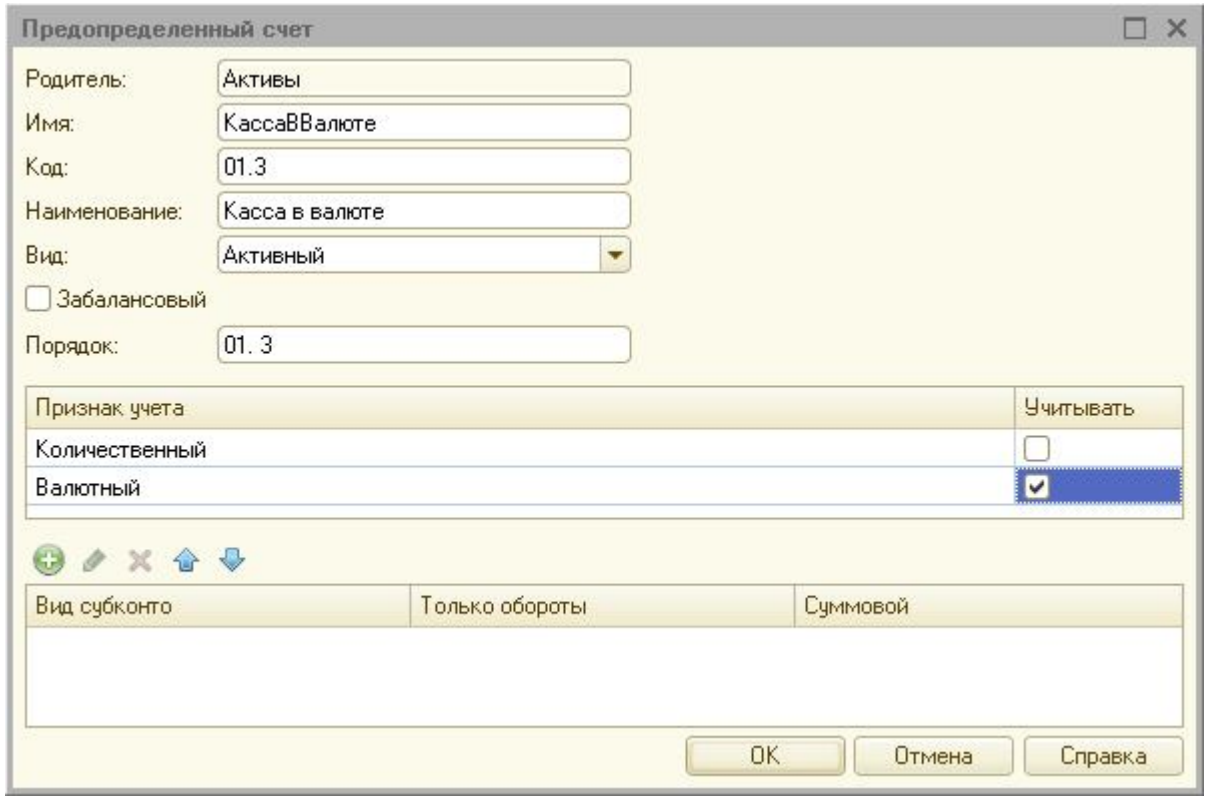
- Признак учета.
- Балансовый.

должны иметь одинаковое значение.

Создадим счета для валютного учета:
«План счетов Управленческий – Предопределенные»:




Предопределенный счет
Родитель: Взаиморасчеты
Имя: РасчетыСПоставщикамиВВалюте
Код: 03.5
Наименование: Расчеты с поставщиками (в валюте)
Вид: Активный/Пассивный
Забалансовый
Порядок: 03. 5
Признак учета
Учитывать
Количественный
Валютный
Вид субконто
Только обороты
Суммовой
Контрагенты
OK
Отмена
Справка



Добавим возможность совершения валютных операций.

В документ Приходная добавим возможность указания валюты:
Добавим реквизит «Валюта»:

Свойство	Значение
Имя	Валюта
Синоним	Валюта
Тип	СправочникСсылка.Валюты

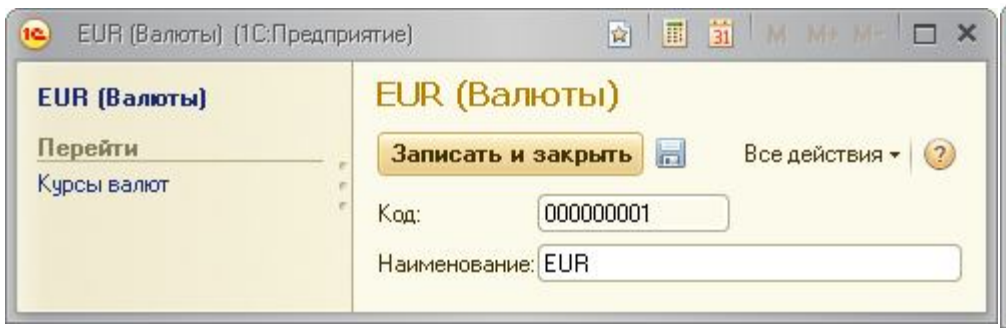
Если «Валюта» не заполнен, значит это национальная валюта.

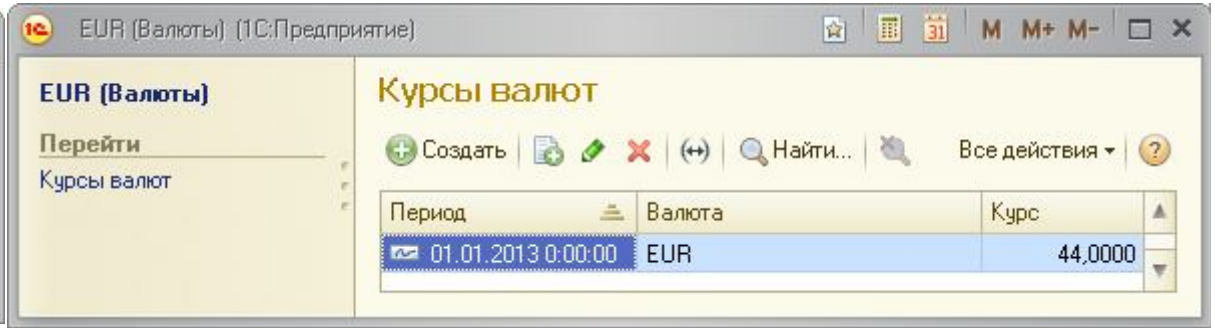
Добавим реквизит «Валюта» в форму документа.

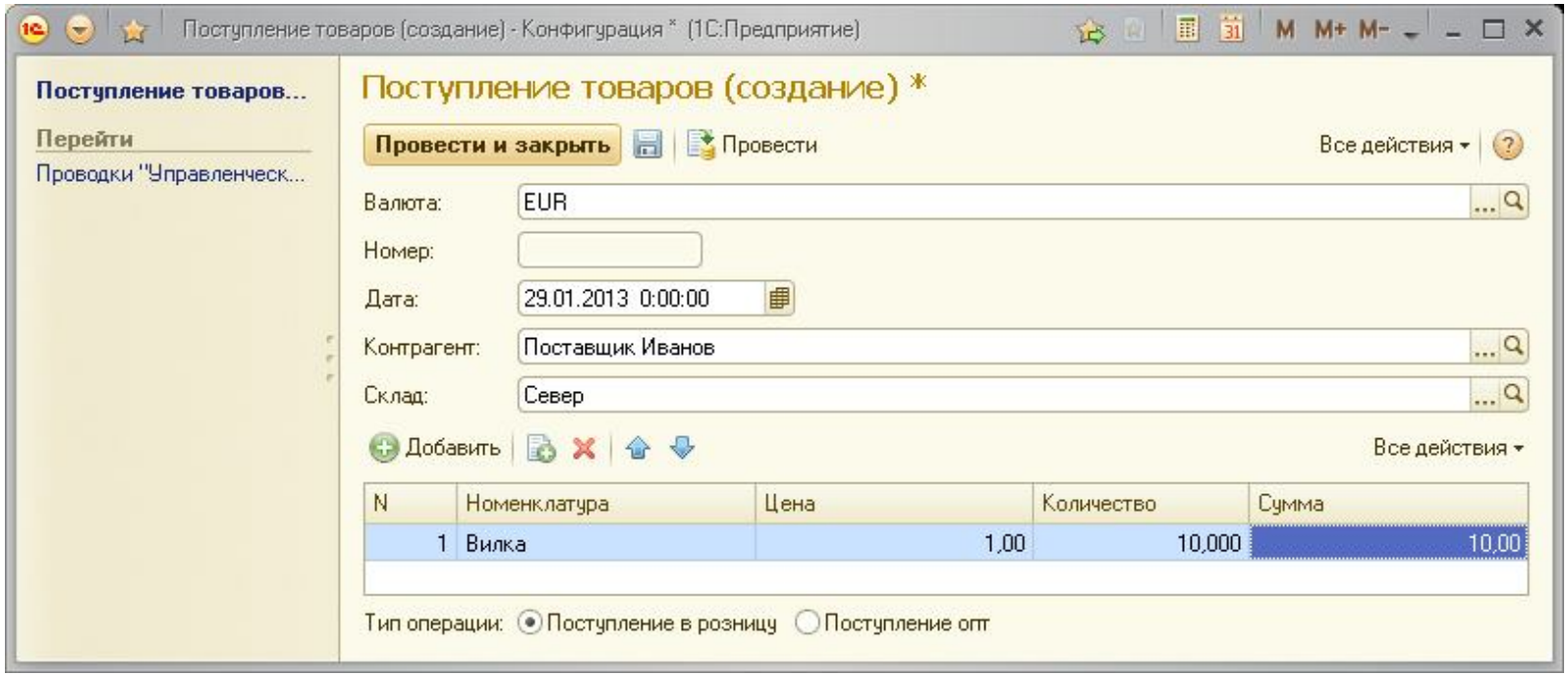
Если пользователь укажет валюту, то сформируем движение по счету «РасчетыСПоставщикамиВВалюте» (модуль объекта):

В режиме исполнения:

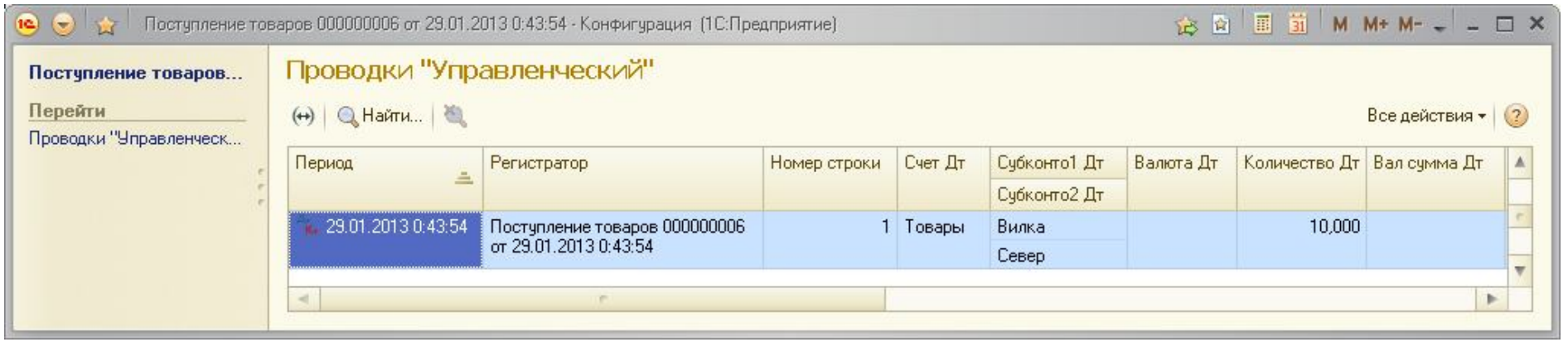
Создадим документ Приходная. Перед этим создадим валюту «ЕВРО» и зададим ее курс:

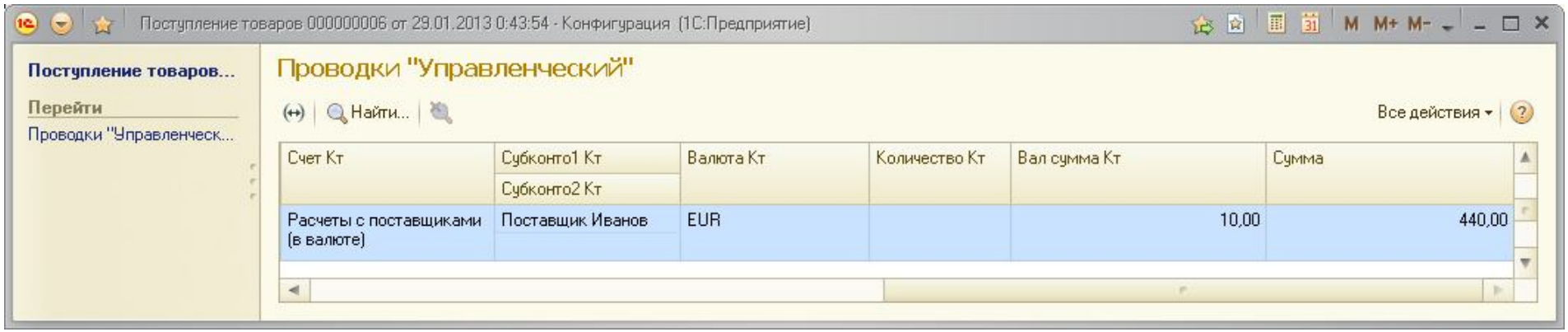




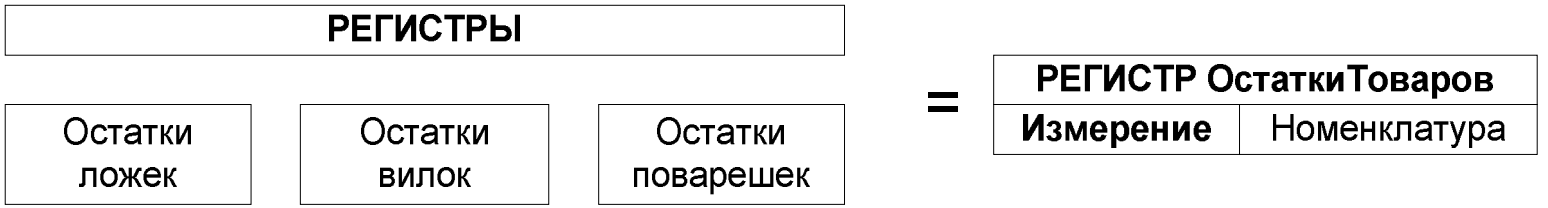


Проведем документ и посмотрим проводки:





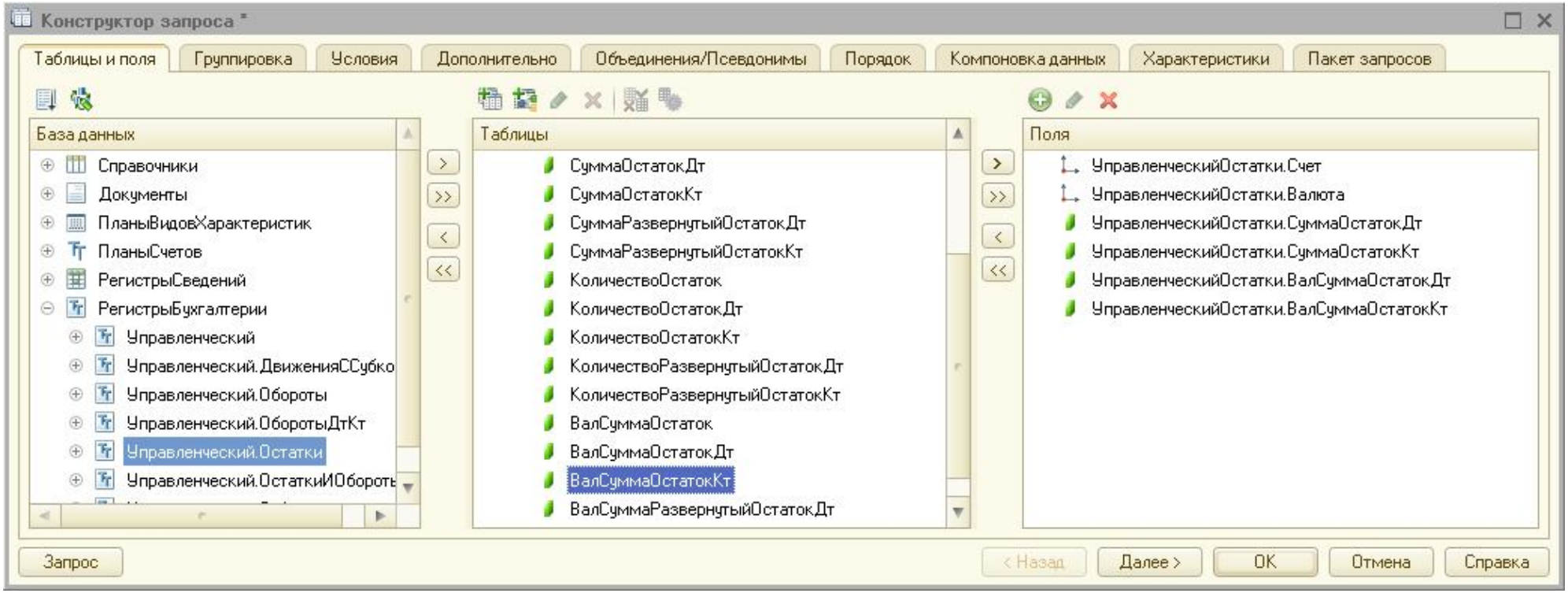
Почему «Валюта» измерение Регистра бухгалтерии, а не Субконто Плана счетов (из Плана видов характеристик):
Зачем бухгалтеру валютный учет? Отчетность ведется в национальной валюте. Допустим, по контрагенту надо накапливать долги в валюте, но расчет все равно будет производиться в национальной валюте, только дополнительно будет рассчитана сумма в валюте. Т.е. Основной баланс, участвующий в регламентированных отчетах, ведется в национальной валюте, а для Управленческих (служебных) целей бухгалтер может параллельно вести дополнительный баланс в валюте.
В примере Измерение «разрезало» баланс, т.е. каждое значение Измерения – это отдельный баланс (новый Разрез).
Субконто же используется для аналитики по счетам, чтобы уйти от Субсчетов.
Измерение любого регистра как бы создает новый регистр. Пример:
Можно создать Регистры «Остатки ложек», «Остатки вилок», «Остатки поварешек», а можно создать один регистр «Остатки товаров» с Измерением «Номенклатура».



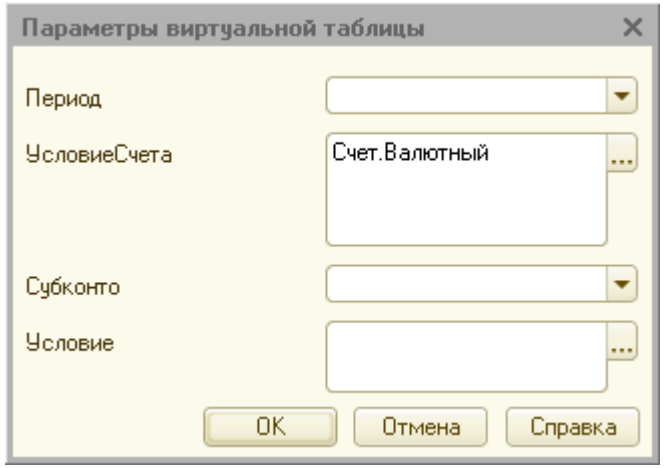
Сделаем отчет «Анализ валютных остатков», показывающий остатки в рублях и в валютах, если счет валютный.

Свойство	Значение
Имя	АнализВалютныхОстатков
Синоним	Анализ валютных остатков

Откроем «Схему компоновки данных», добавим «Набор данных-запрос» и откроем «Конструктор запроса»:

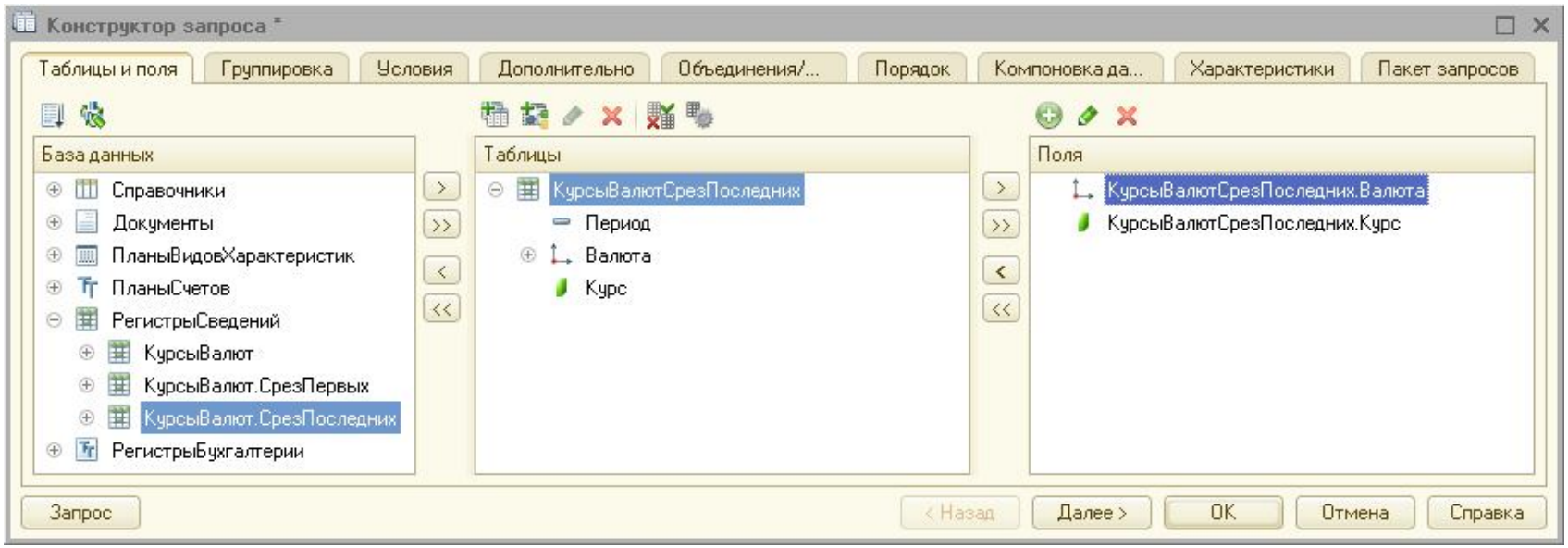


Условия виртуальной таблицы:



Т.е. данные берем только из валютных счетов.

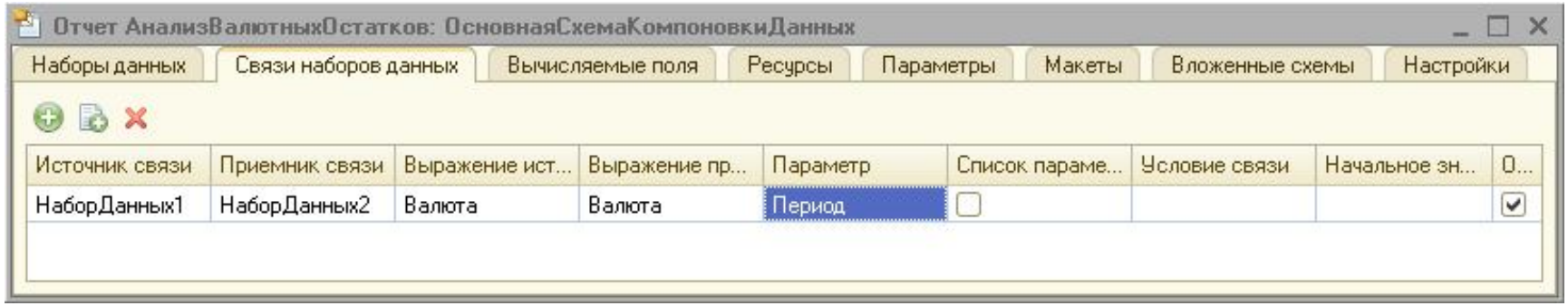
Остатки получили. Привяжем к ним Курс валюты:
Добавим второй набор в Наборы данных:



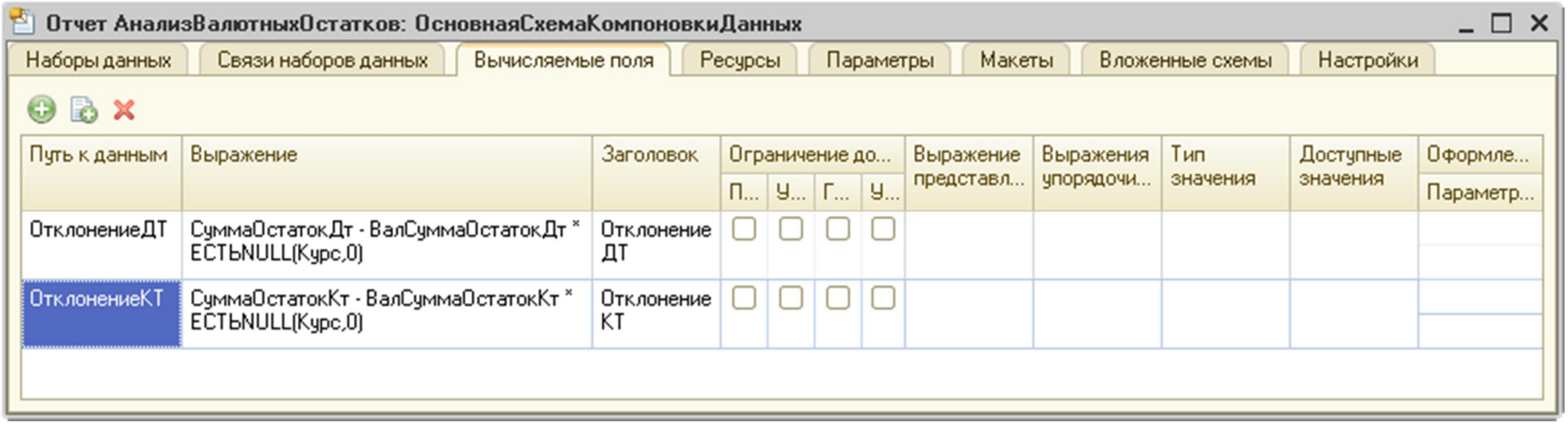
Свяжем Наборы данных:

Связи в СКД всегда ЛЕВОЕ СОЕДИНЕНИЕ.

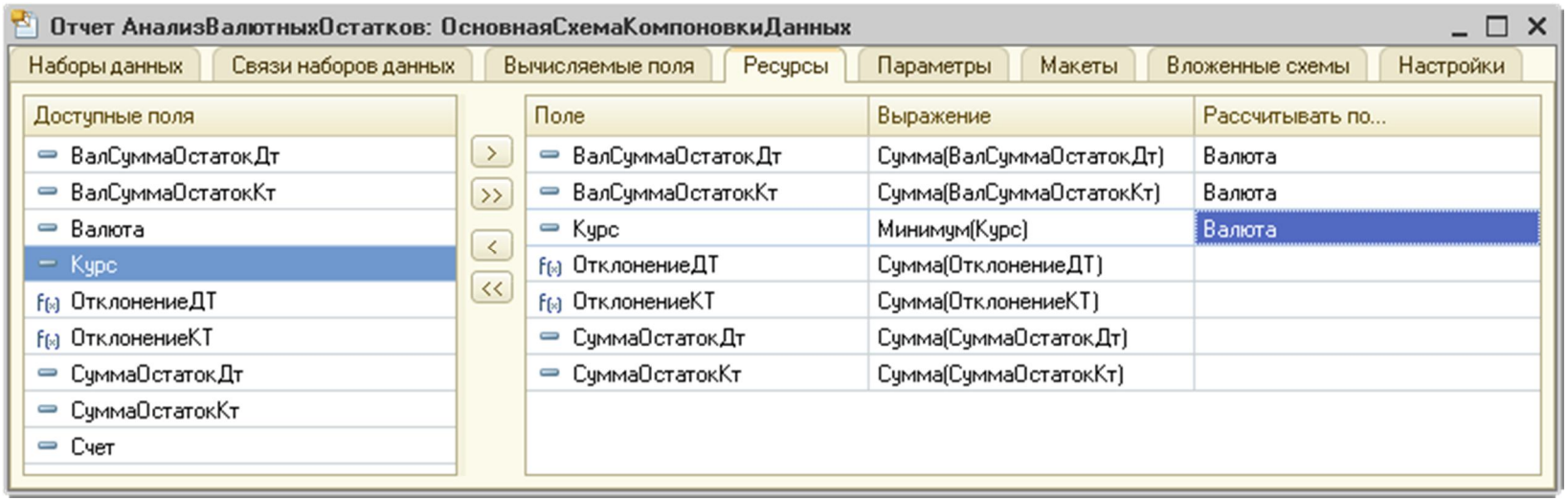
Связь по полю Валюта. Период должен совпадать, т.е. Период тоже как бы условие связи.



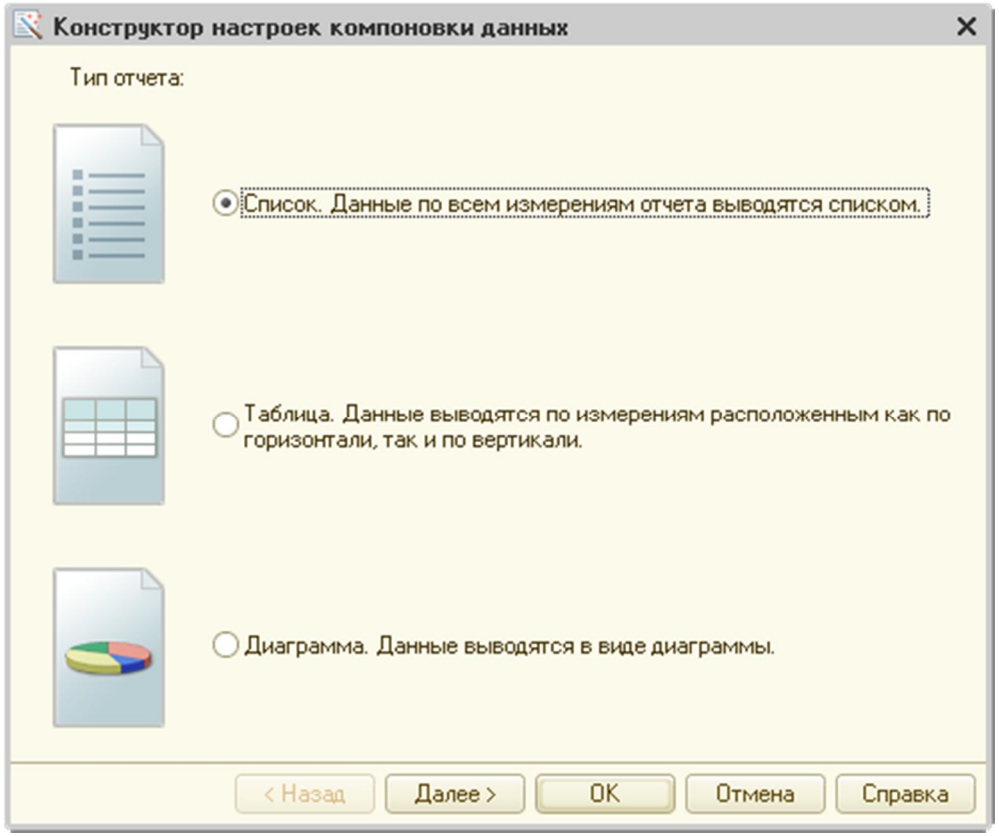
С времени формирования движения по Регистру бухгалтерии, курс валюты может поменяться, тогда в отчете следует показать, что на дату отчета есть отклонение от учетной ставки.

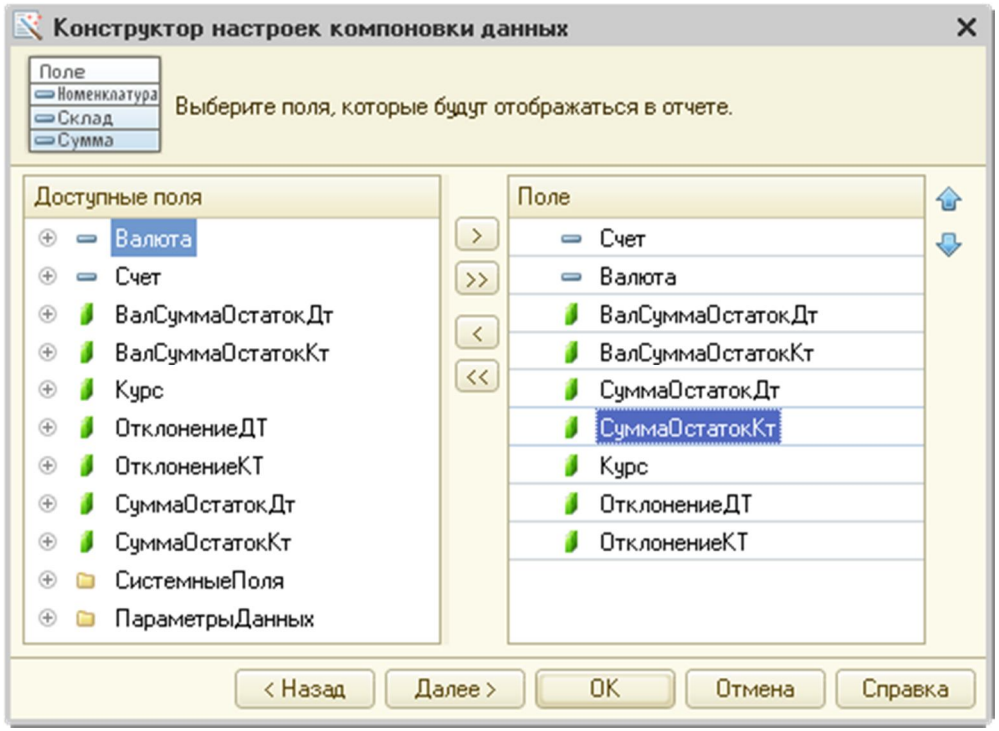


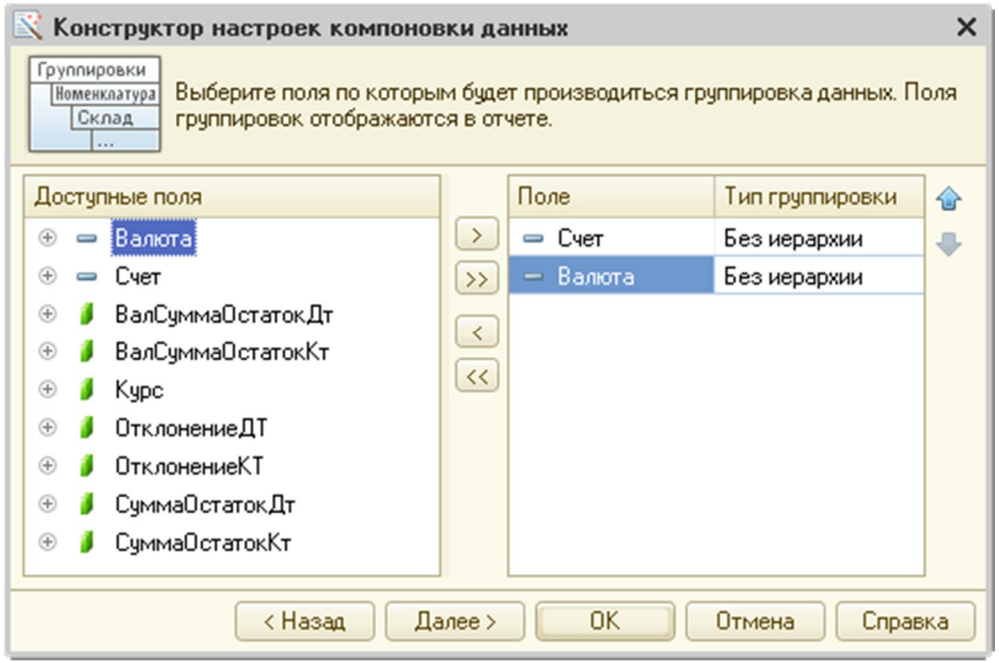
Валютные суммы будем рассчитывать только по Валюте. Курс – для значения по группировке возьмем Минимум (он не является Ресурсом).

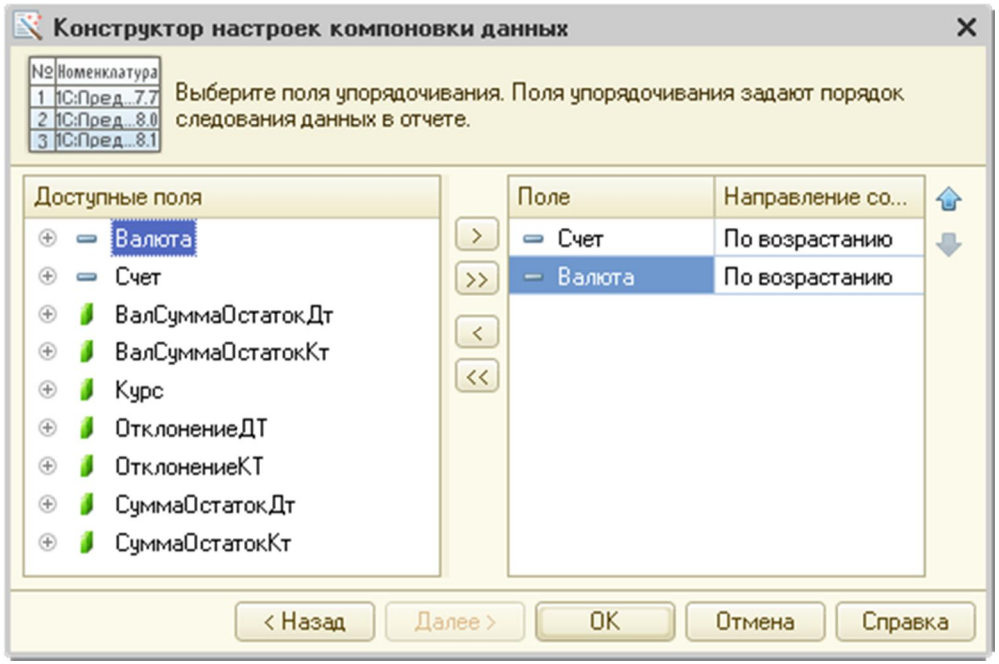

Отчет АнализВалютныхОстатков: ОсновнаяСхемаКомпоновкиДанных
Наборы данных
Связи наборов данных
Вычисляемые поля
Ресурсы
Параметры
Макеты
Вложенные схемы
Настройки
Доступные поля
ВалСуммаОстатокДт
ВалСуммаОстатокКт
Валюта
Курс
ОтклонениеДТ
ОтклонениеКТ
СуммаОстатокДт
СуммаОстатокКт
Счет
Поле
Выражение
Рассчитывать по...
ВалСуммаОстатокДт
Сумма(ВалСуммаОстатокДт)
Валюта
ВалСуммаОстатокКт
Сумма(ВалСуммаОстатокКт)
Валюта
Курс
Минимум(Курс)
Валюта
ОтклонениеДТ
Сумма(ОтклонениеДТ)
ОтклонениеКТ
Сумма(ОтклонениеКТ)
СуммаОстатокДт
Сумма(СуммаОстатокДт)
СуммаОстатокКт
Сумма(СуммаОстатокКт)

Настройки отчета:



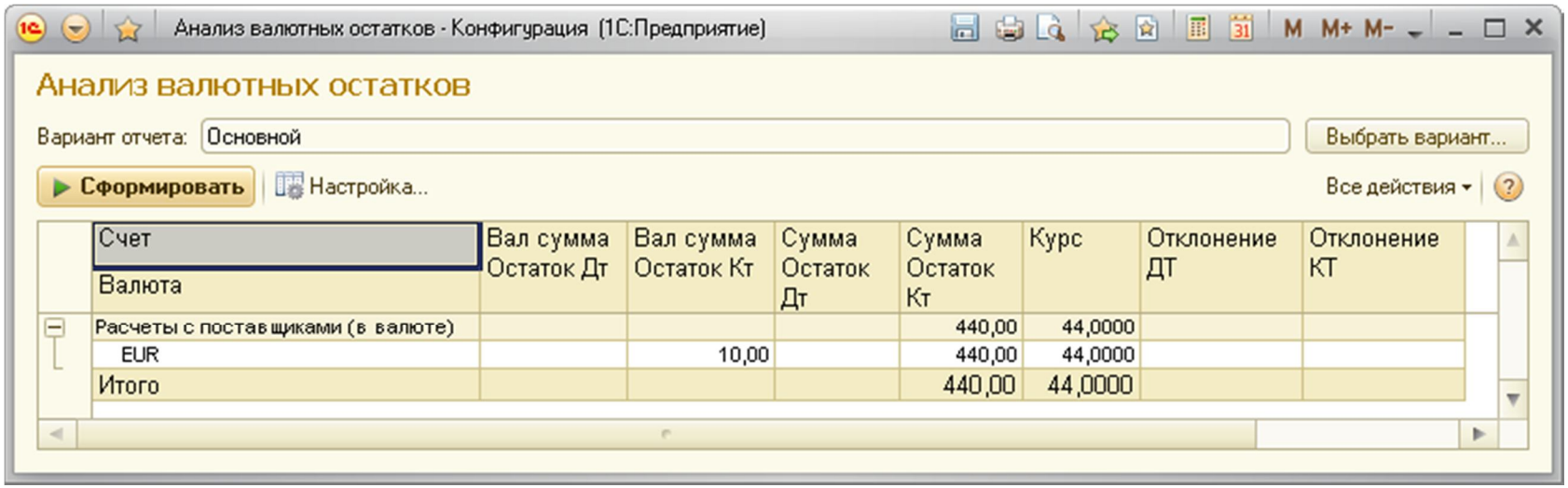

Конструктор настроек компоновки данных
Поле
Номенклатура
Склад
Сумма
Выберите поля, которые будут отображаться в отчете.
Доступные поля
Валюта
Счет
ВалСуммаОстатокДт
ВалСуммаОстатокКт
Курс
ОтклонениеДТ
ОтклонениеКТ
СуммаОстатокДт
СуммаОстатокКт
СистемныеПоля
ПараметрыДанных
Поле
Счет
Валюта
ВалСуммаОстатокДт
ВалСуммаОстатокКт
СуммаОстатокДт
СуммаОстатокКт
Курс
ОтклонениеДТ
ОтклонениеКТ
< Назад
Далее >
OK
Отмена
Справка


Конструктор настроек компоновки данных
Группировки
Номенклатура
Склад
...
Выберите поля по которым будет производиться группировка данных. Поля группировок отображаются в отчете.
Доступные поля
Валюта
Счет
ВалСуммаОстатокДт
ВалСуммаОстатокКт
Курс
ОтклонениеДТ
ОтклонениеКТ
СуммаОстатокДт
СуммаОстатокКт
Поле
Тип группировки
Счет
Без иерархии
Валюта
Без иерархии
< Назад
Далее >
OK
Отмена
Справка



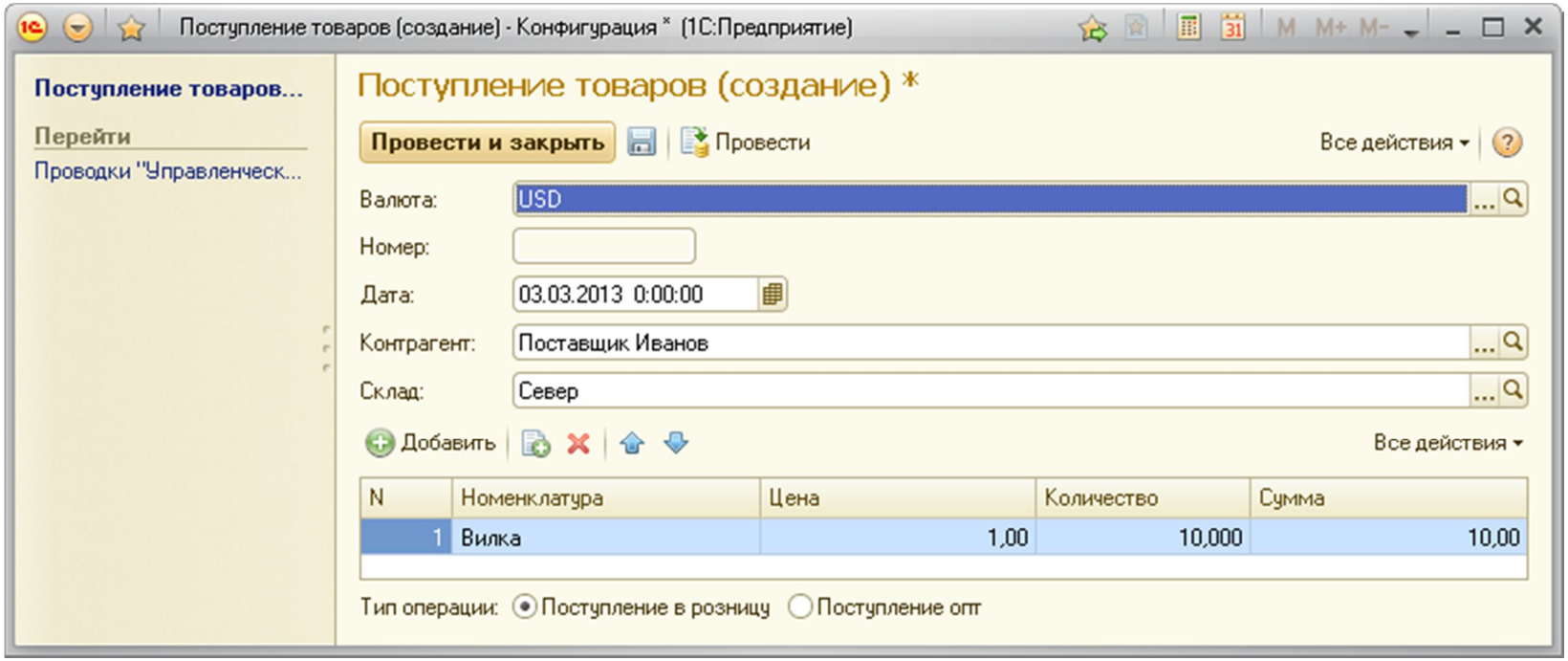
В режиме исполнения:

Сформируем отчет:

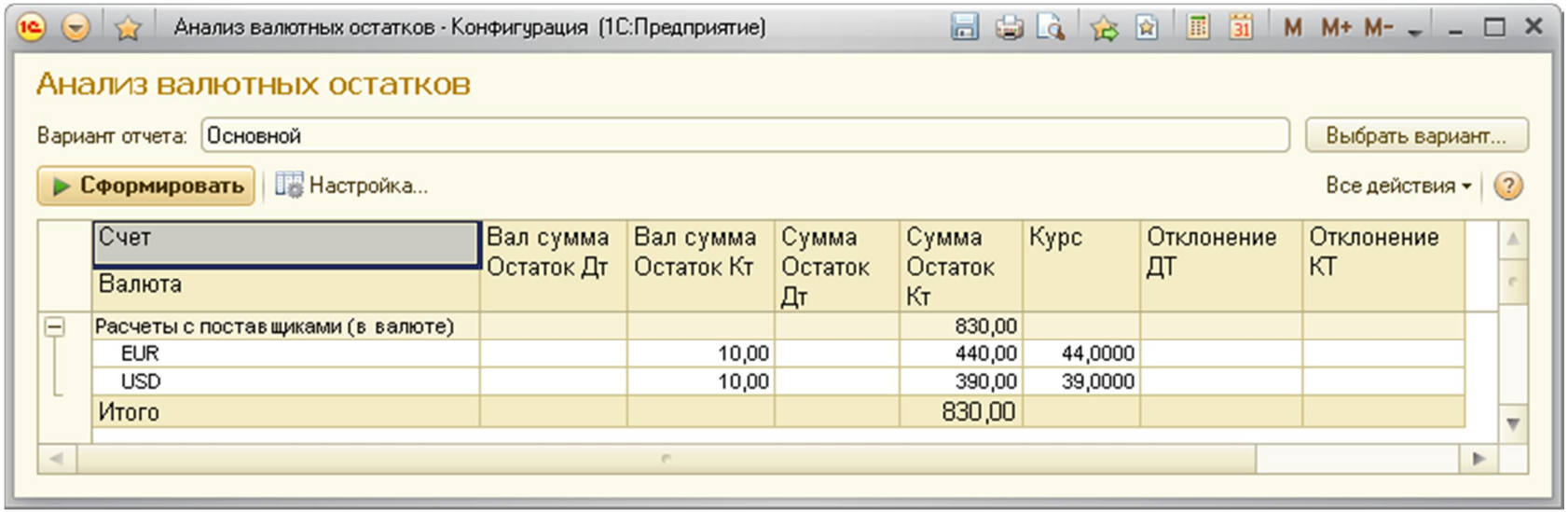


Счет / Валюта	Вал сумма Остаток Дт	Вал сумма Остаток Кт	Сумма Остаток Дт	Сумма Остаток Кт	Курс	Отклонение ДТ	Отклонение КТ
Расчеты с поставщиками (в валюте)				440,00	44,0000		
EUR		10,00		440,00	44,0000		
Итого				440,00	44,0000		

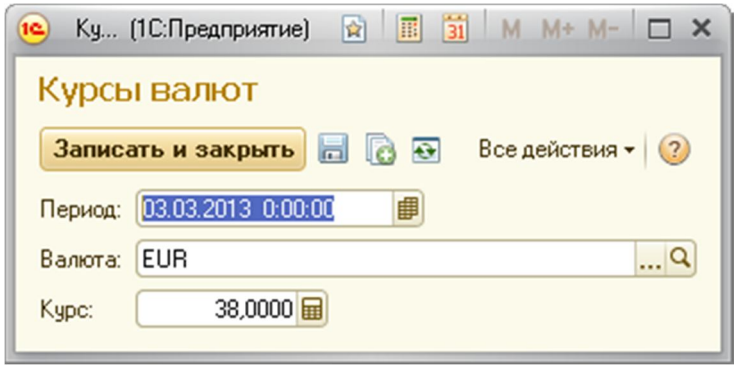
Сформируем Приходную накладную (в долларах). Курс доллара будет 39 рублей:

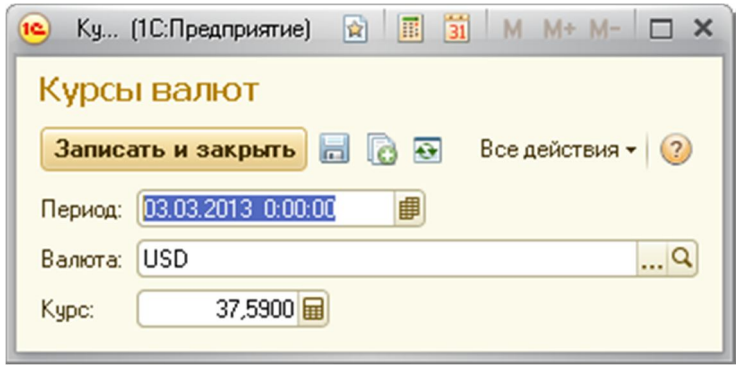


Проверим отчет:

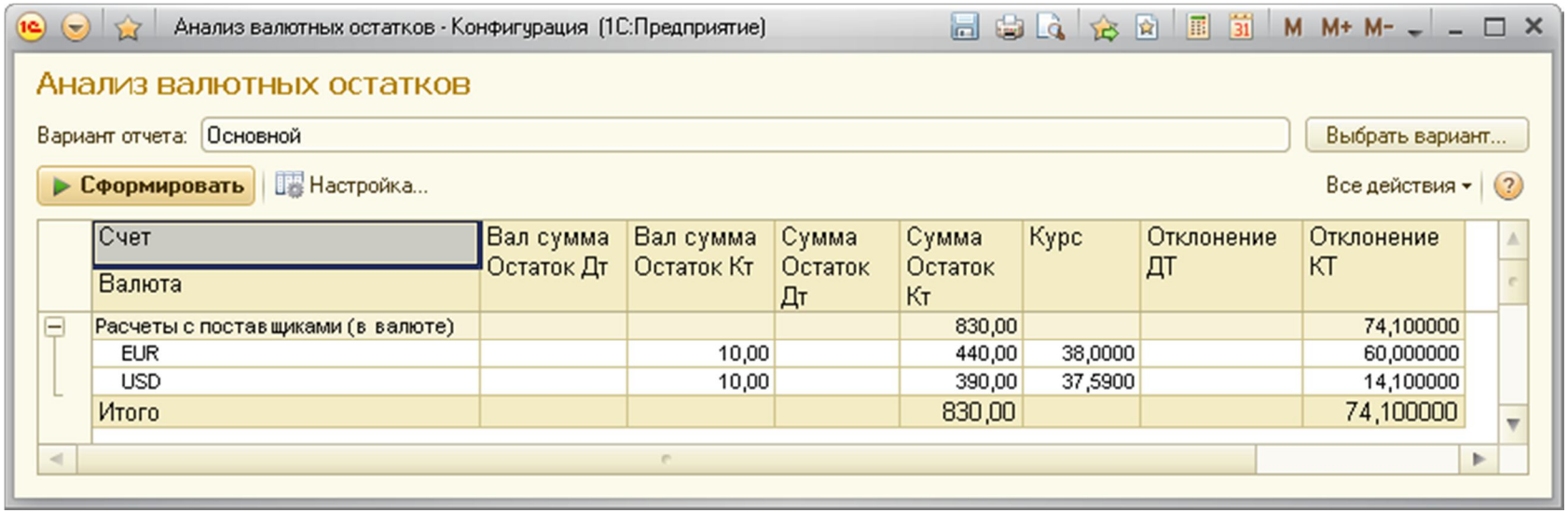


Заведем курсы валют на текущую дату:





Сформируем отчет:



Счет / Валюта	Вал сумма Остаток Дт	Вал сумма Остаток Кт	Сумма Остаток Дт	Сумма Остаток Кт	Курс	Отклонение ДТ	Отклонение КТ
Расчеты с поставщиками (в валюте)				830,00			74,100000
EUR		10,00		440,00	38,0000		60,000000
USD		10,00		390,00	37,5900		14,100000
Итого				830,00			74,100000

Появилась сумма отклонения, т.е. сумма в рублях, которая в данный момент числится по счету, не соответствует действительности на сумму отклонения.

Допустим, в кассе есть валюта и на текущую дату курс увеличился, то это обернется прибылью. А если эта валюта – заемные средства, то это обернется убытком. Обратная ситуация возникнет при понижении курса.

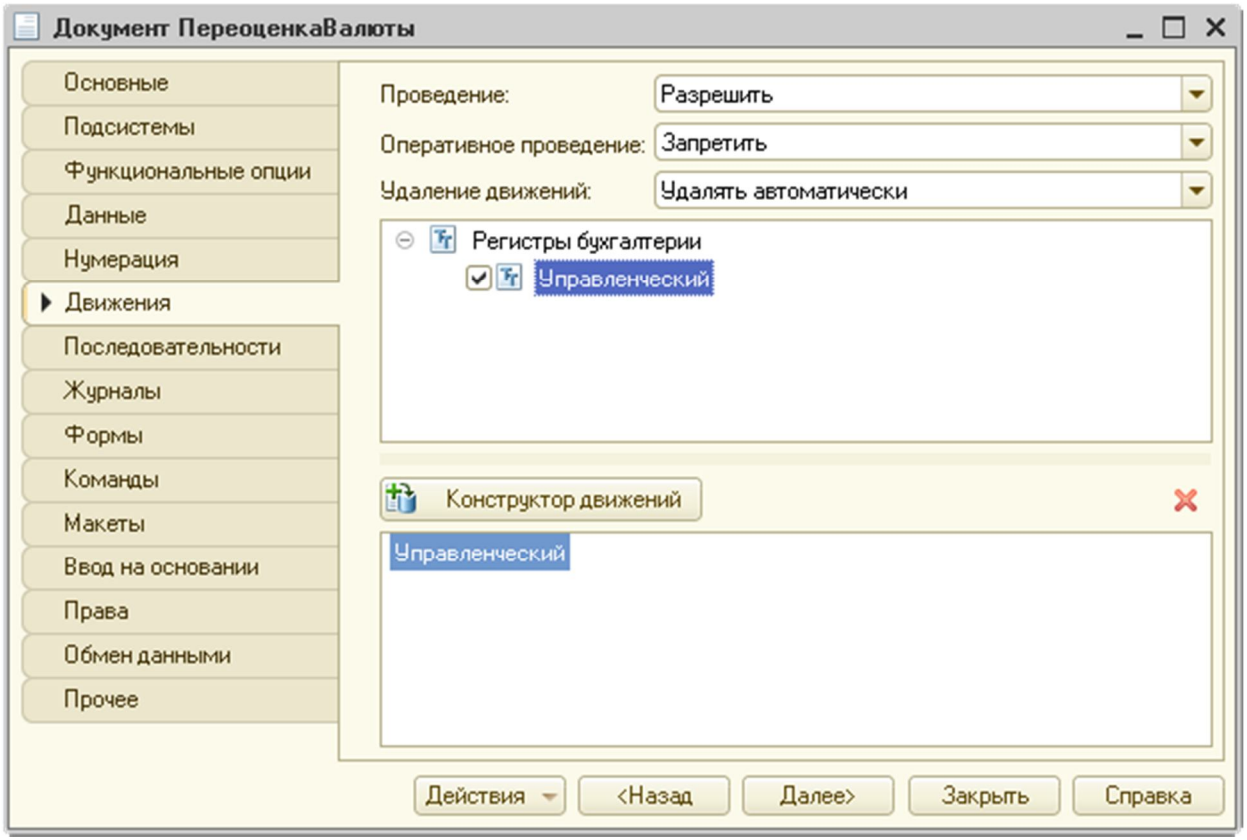
На суммы отклонений надо составить проводки, чтобы остаток в национальной валюте по счету «Расчеты с поставщиками в валюте» стал актуальным. В зависимости от счета:

Дт (прибыль) – Кт (счет)	ИЛИ	Дт (счет) – Кт (прибыль)

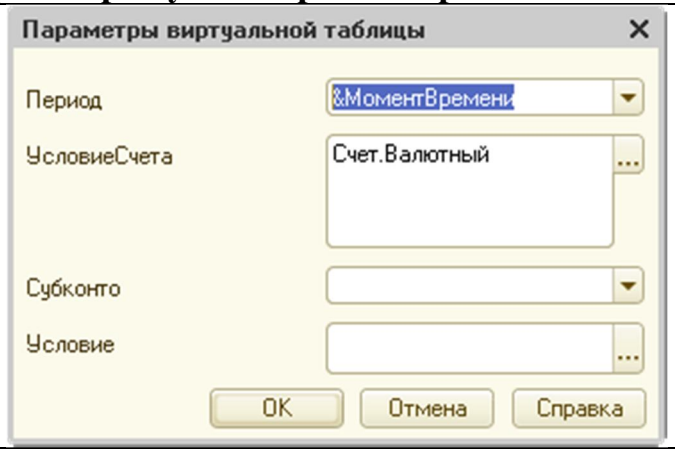
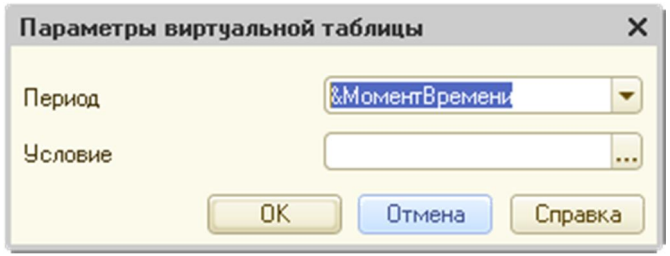
Например, счет «Касса»:
Дт «Касса» - Кт «Прибыль» - денег в кассе стало больше.

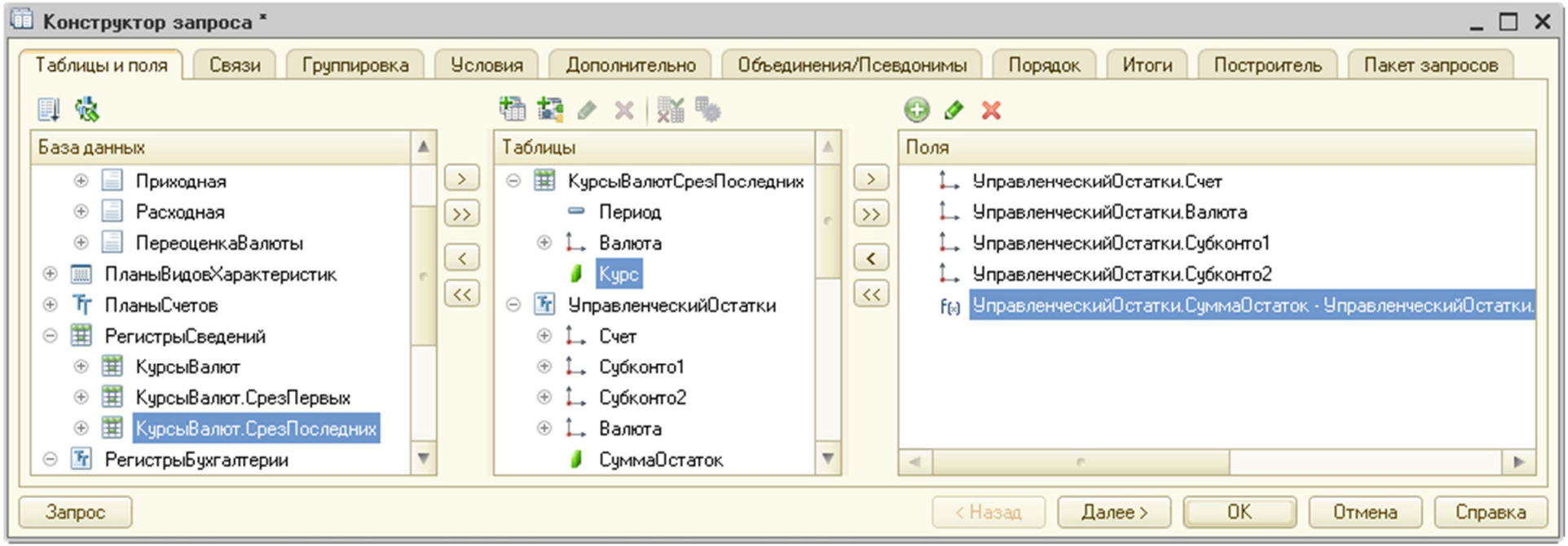
Создадим служебный документ «Переоценка валюты», который будет формировать проводки на суммы отклонений.

Свойство	**Значение**
Имя	ПереоценкаВалюты
Синоним	Переоценка валюты



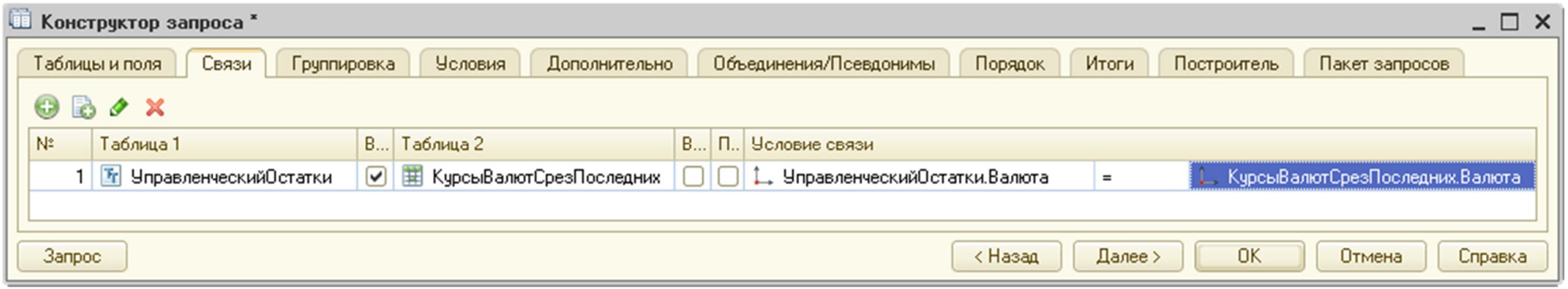
Модуль объекта. Запрос аналогичен запросу из отчета «АнализВалютныхОстатков»:

Условия регистра бухгалтерии «УправленческийОстатки»	Условия регистра сведений «КурсыВалютСрезПоследних»
	

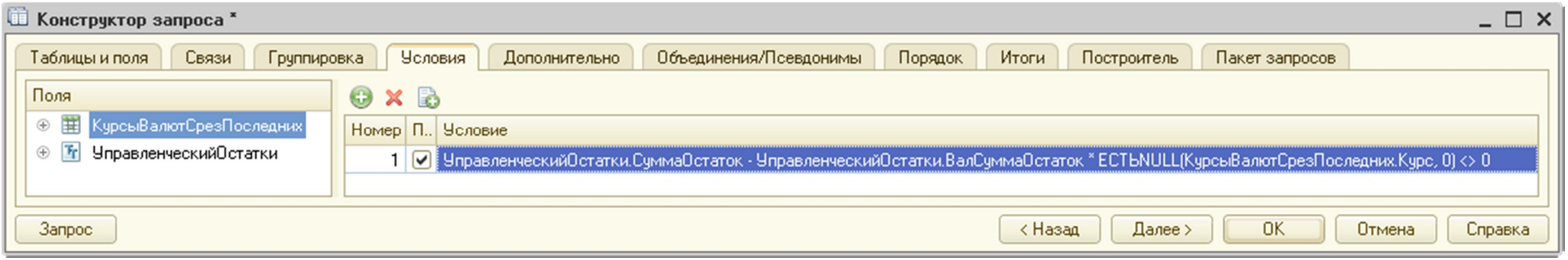


Поле «Отклонение»:

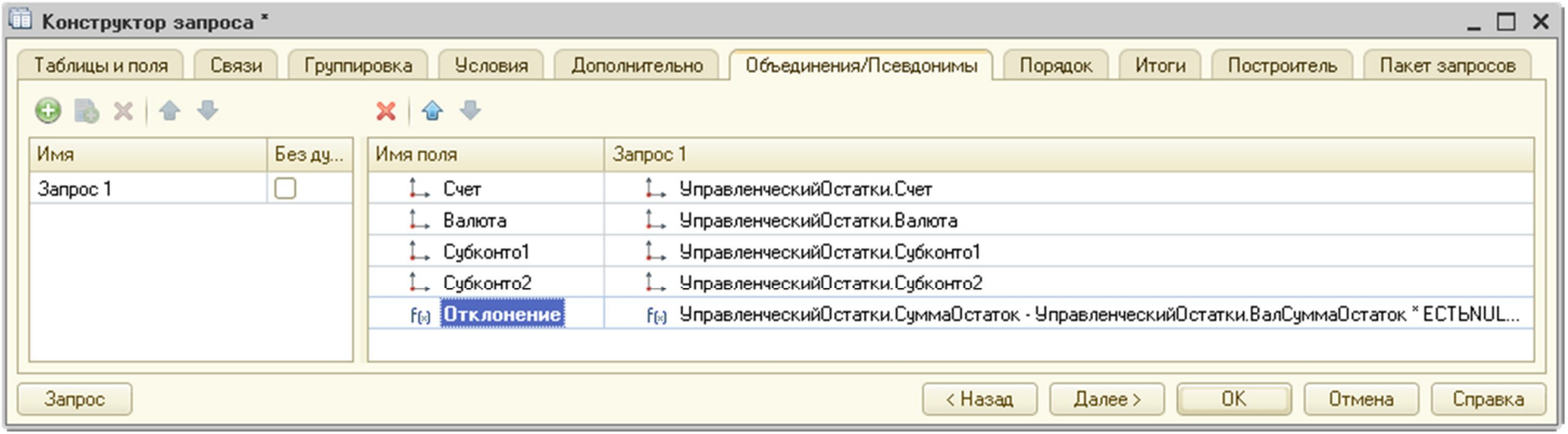
```
СуммаОстаток - ВалСуммаОстаток * ЕСТЬNULL(Курс, 0)
```



Условие:



Вычисляемому полю зададим псевдоним:



Запрос вернет отклонения по счетам, курс которых был изменен.

Итого код:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)

     Движения.Управленческий.Записывать = Истина;

     Запрос = Новый Запрос;
     Запрос.Текст =
     "ВЫБРАТЬ
     |    УправленческийОстатки.Счет,
     |    УправленческийОстатки.Валюта,
     |    УправленческийОстатки.Субконто1,
     |    УправленческийОстатки.Субконто2,
     |    УправленческийОстатки.СуммаОстаток - УправленческийОстатки.ВалСуммаОстаток *
ЕСТЬNULL(КурсыВалютСрезПоследних.Курс, 0) КАК Отклонение
     |ИЗ
     |    РегистрБухгалтерии.Управленческий.Остатки(&МоментВремени, Счет.Валютный, , ) КАК УправленческийОстатки
     |         ЛЕВОЕ СОЕДИНЕНИЕ РегистрСведений.КурсыВалют.СрезПоследних(&МоментВремени, )
КАК КурсыВалютСрезПоследних
     |         ПО УправленческийОстатки.Валюта = КурсыВалютСрезПоследних.Валюта
     |ГДЕ
     |    УправленческийОстатки.СуммаОстаток - УправленческийОстатки.ВалСуммаОстаток *
ЕСТЬNULL(КурсыВалютСрезПоследних.Курс, 0) <> 0";

     Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
     РезультатЗапроса = Запрос.Выполнить();

     Выборка = РезультатЗапроса.Выбрать();
     Пока Выборка.Следующий() Цикл

          Проводка        = Движения.Управленческий.Добавить();
          Проводка.Период = Дата;
          Проводка.Сумма  = Макс(Выборка.Отклонение, -Выборка.Отклонение);

          Если (Выборка.Отклонение < 0 И Выборка.Счет <> ВидСчета.Пассивный)
               ИЛИ (Выборка.Отклонение > 0 И Выборка.Счет = ВидСчета.Пассивный) Тогда
```

```
            // прибыль
            Проводка.СчетДт   = Выборка.Счет;
            Проводка.ВалютаДт = Выборка.Валюта;
            Проводка.СчетКт   = ПланыСчетов.Управленческий.Прибыль;
            ТЧСубконто        = Проводка.СубконтоДт;
        Иначе
            // убыток
            Проводка.СчетКт   = Выборка.Счет;
            Проводка.ВалютаКт = Выборка.Валюта;
            Проводка.СчетДт   = ПланыСчетов.Управленческий.Прибыль;
            ТЧСубконто        = Проводка.СубконтоКт;
        КонецЕсли;

        Для Каждого Стр Из Выборка.Счет.ВидыСубконто Цикл
            ТЧСубконто[Стр.ВидСубконто] = Выборка["Субконто" + Стр.НомерСтроки];
        КонецЦикла;

    КонецЦикла;
```

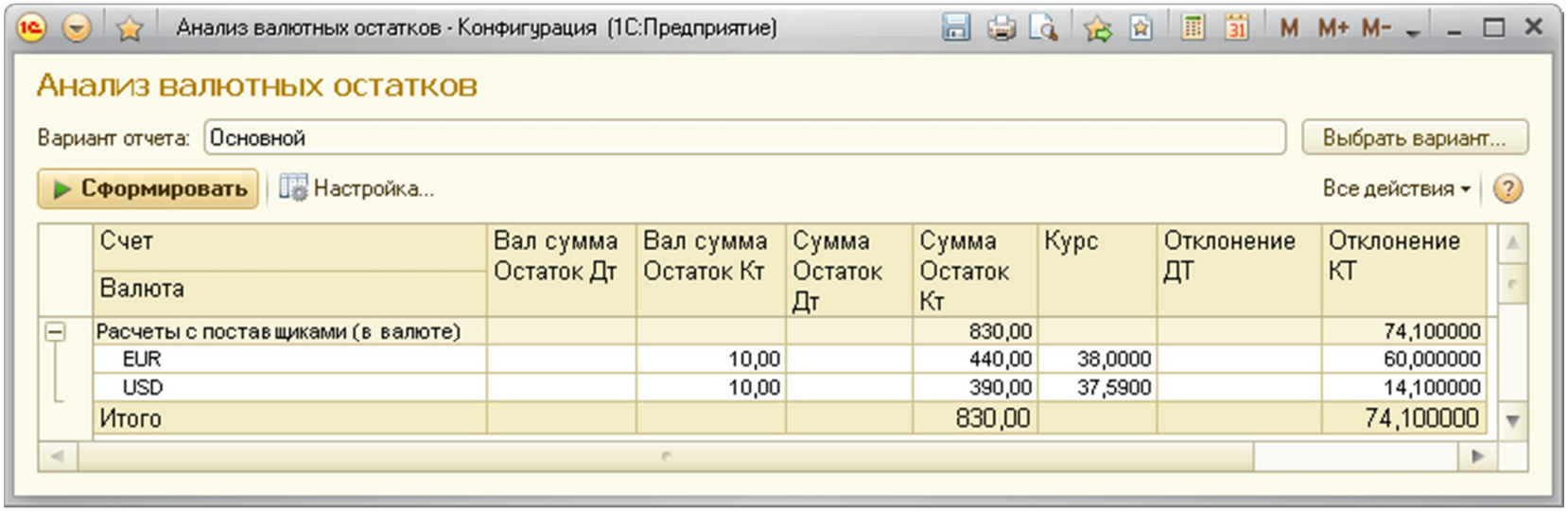
КонецПроцедуры

Цикл по Субконто организован, т.к. мы заранее не знаем сколько субконто у счета, поэтому переберем все виды субконто счета и присвоим им значения.

`Выборка["Субконто" + Стр.НомерСтроки]` – это значит «Субконто1» или «Субконто2», т.е. это обращение к значению по идентификатору (альтернатива обращения «через точку»).

В режиме исполнения:

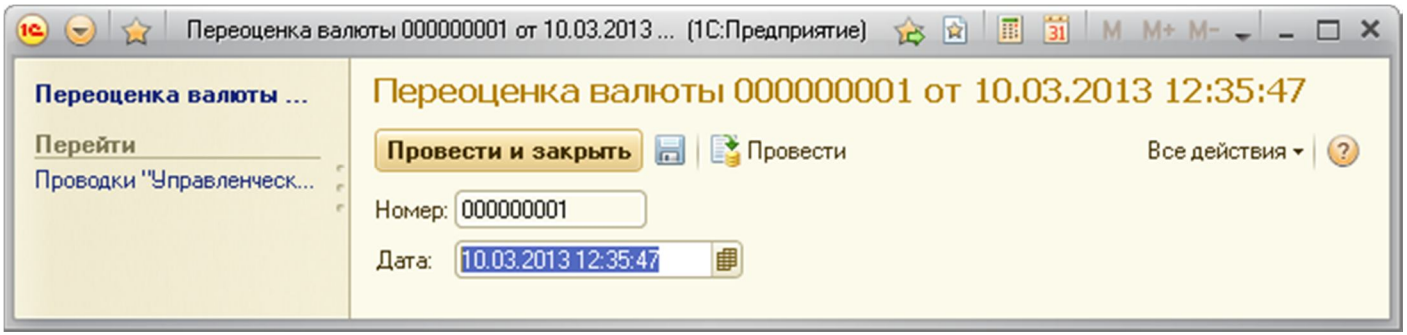
Проверим отчет «Анализ валютных остатков»:



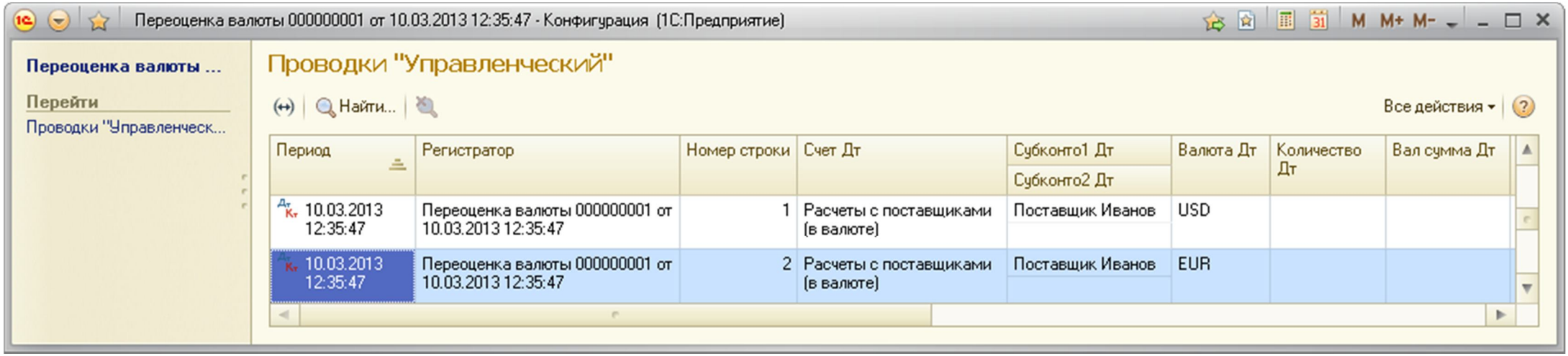
Счет / Валюта	Вал сумма Остаток Дт	Вал сумма Остаток Кт	Сумма Остаток Дт	Сумма Остаток Кт	Курс	Отклонение ДТ	Отклонение КТ
Расчеты с поставщиками (в валюте)				830,00			74,100000
EUR		10,00		440,00	38,0000		60,000000
USD		10,00		390,00	37,5900		14,100000
Итого				830,00			74,100000

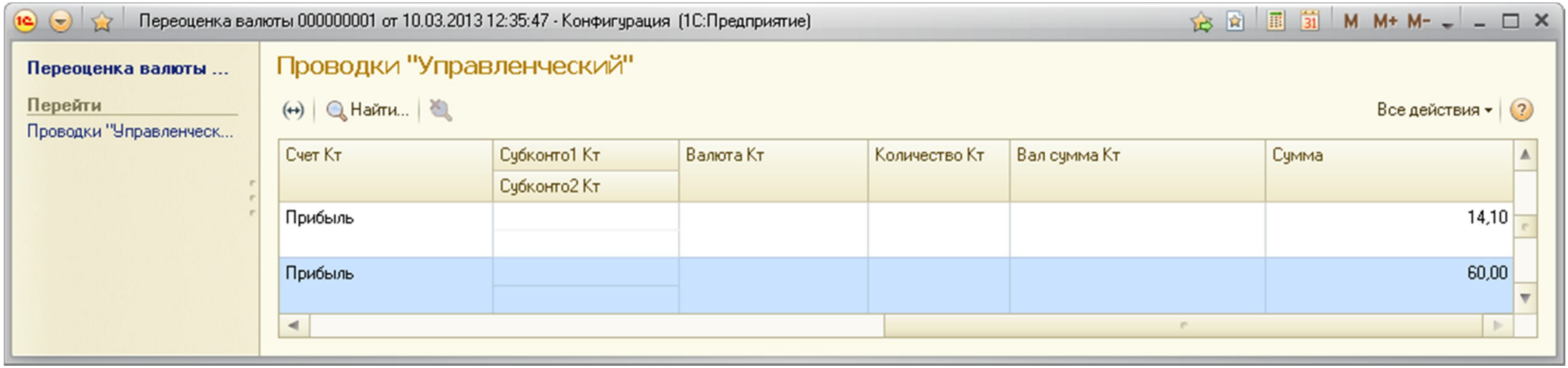
Т.к. с момента проведения документов курс изменился, то есть отклонение.

Создадим и проведем документ «Переоценка валюты»:

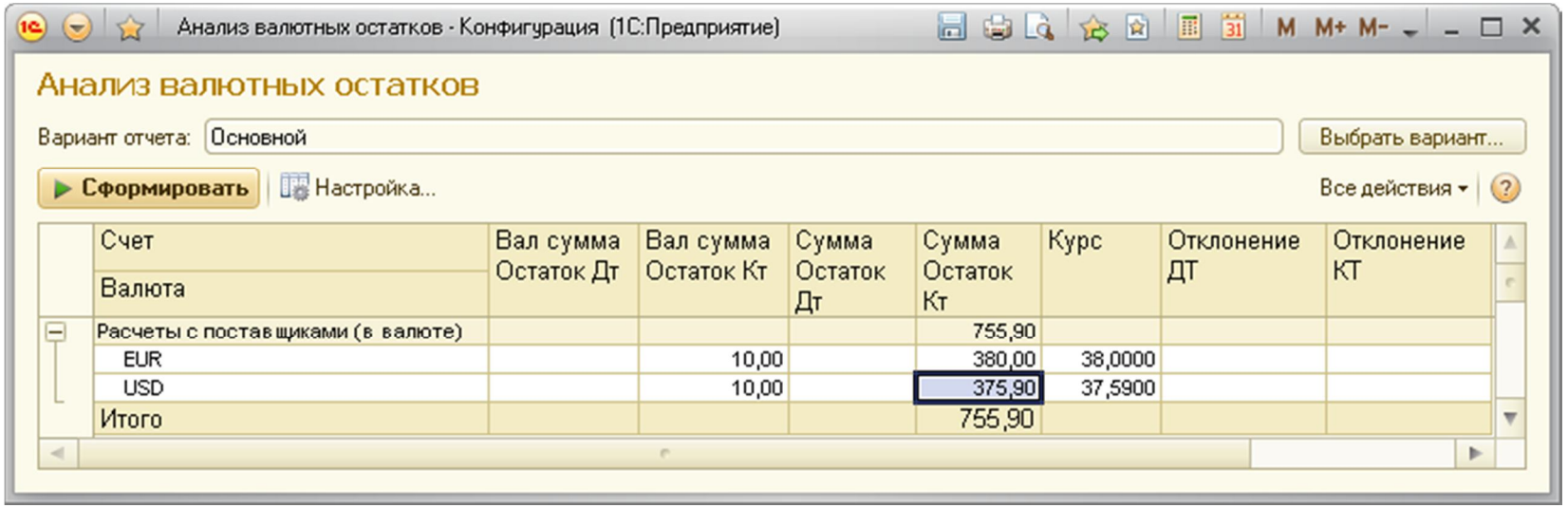


Проверим движения:





И проверим отчет «Анализ валютных остатков»:



Счет / Валюта	Вал сумма Остаток Дт	Вал сумма Остаток Кт	Сумма Остаток Дт	Сумма Остаток Кт	Курс	Отклонение ДТ	Отклонение КТ
Расчеты с поставщиками (в валюте)				755,90			
EUR		10,00		380,00	38,0000		
USD		10,00		375,90	37,5900		
Итого				755,90			

Отклонения исчезли, т.к. их суммы мы перевели, в данном примере, на счет прибыли.

Если сделать документ «Переоценка валюты» на следующий день, то движений не будет, т.к. за это время (в примере) курс не менялся.

Занятие 2.24

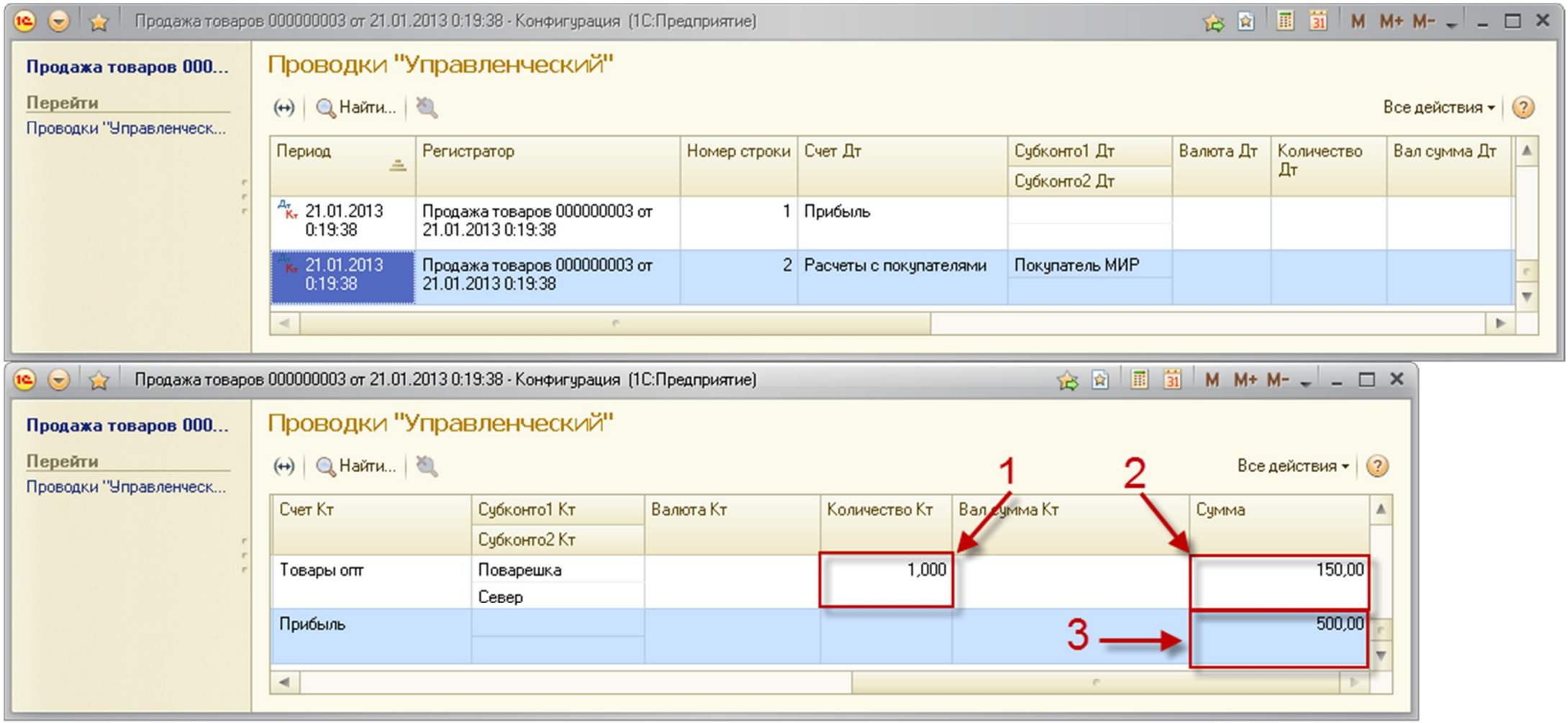
Отчет «Анализ продаж»:

Продажи

Номенклатура	Количество	Себестоимость	Выручка	Прибыль

Показатель – то, что показывает отчет. В примере, в отчете 3 показателя и 1 рассчитываемый показатель – «Прибыль».

Продажи регистрируются документом «Расходная», проверим движения, формируемые документом, и посмотрим, достаточно ли их для построения отчета:



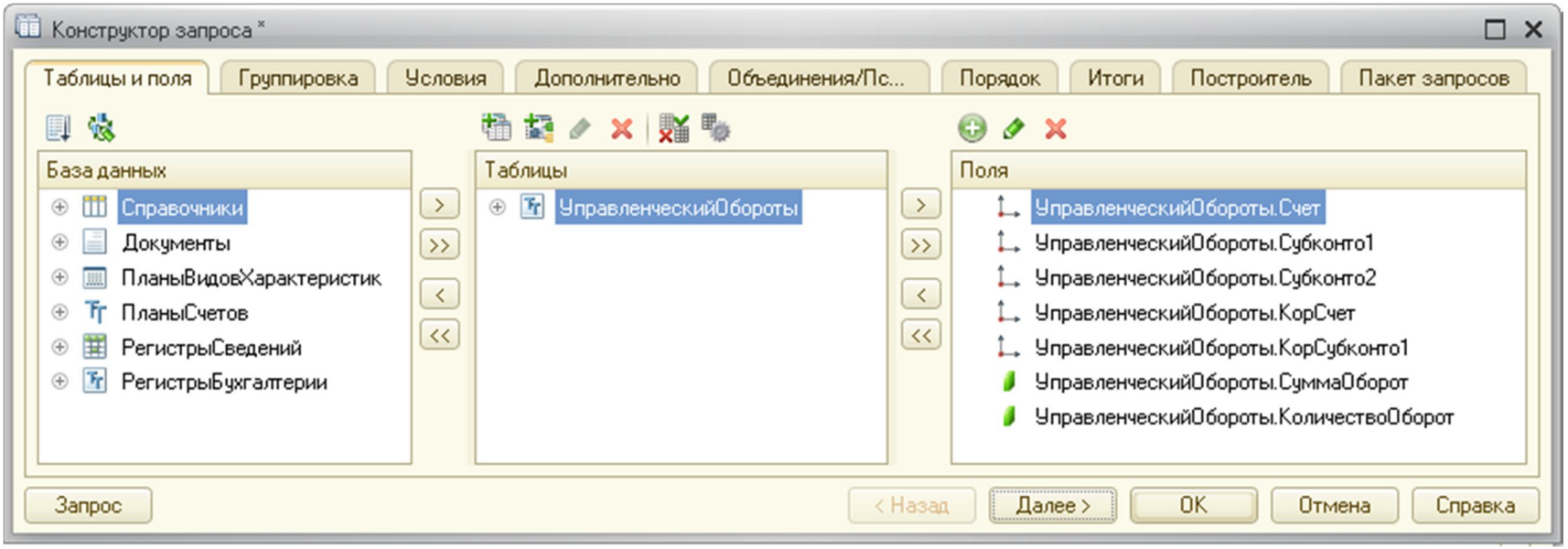
1 – Количество. Оборот по Кт счета «Товары опт» в количественном выражении.
2 – Себестоимость. Оборот по Кт счета «Товары опт» в суммовом выражении.
3 – Выручка. Сумма есть, но она не разрезается по товарам.

Попробуем построить запрос:
Количество продаж и Выручка являются оборотными показателями, т.е. обращаться надо к таблице Обороты Регистра бухгалтерии.

Таблица Обороты имеет 2 вида:

Обороты.

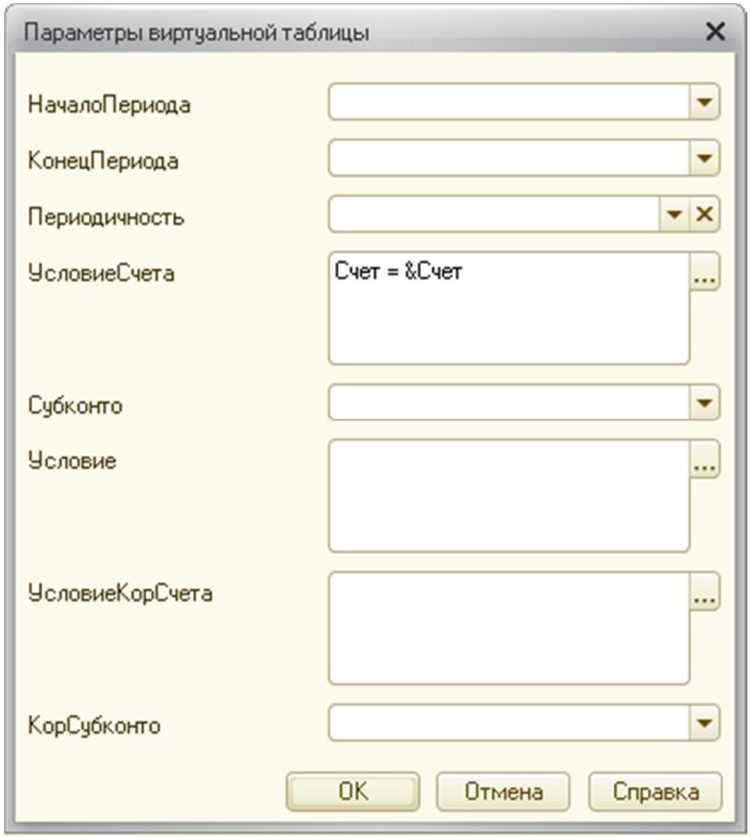
В консоли сформируем запрос:



Счет	Субконто1	Субконто2	КорСчет	КорСубконто1	КорСубконто2	СуммаОборот	КоличествоОборот
Товары	Поварешка	Юг	Прибыль	NULL	NULL	-100	-1
Расчеты с учредителями	NULL	NULL	Расчеты по краткосрочны	NULL	NULL	-200 000	0
Расчеты с поставщиками	Поставщик Иванов	NULL	Товары	Поварешка	Север	-200	0
Товары опт	Поварешка	NULL	Прибыль	NULL	NULL	-150	0
Уставный капитал	NULL	NULL	Расчеты с учредителями	NULL	NULL	-1 000 000	0
Расчеты по краткосрочны	NULL	NULL	Расчеты с учредителями	NULL	NULL	200 000	0
Расчеты с покупателями	Покупатель МИР	NULL	Прибыль	NULL	NULL	1 000	0
Расчеты с поставщиками	Поставщик Иванов	NULL	Товары опт	Поварешка	Юг	-100	0
Расчеты с поставщиками	Поставщик Иванов	NULL	Товары	Вилка	Север	-830	0
Прибыль	NULL	NULL	Товары опт	Поварешка	Север	150	0
Расчеты с поставщиками	Поставщик Иванов	NULL	Прибыль	NULL	NULL	74,1	0
Расчеты с поставщиками	Поставщик Иванов	NULL	Товары опт	Поварешка	Север	-200	0
Товары опт	Поварешка	Север	Расчеты с поставщиками	Поставщик Иванов	NULL	0	1
Товары	Вилка	Север	Расчеты с поставщиками	Поставщик Иванов	NULL	830	20

Выберем счета, отвечающие за Продажи:

Выберем движения только со счетами «Товары», они будут показывать нам и Приход и Расход:



Счет	Субконто1	Субконто2	КорСчет	КорСубконто1	КорСубконто2	СуммаОборот	КоличествоОборот
Товары	Поварешка	Юг	Прибыль	NULL	NULL	-100	-1
Товары	Вилка	Север	Расчеты с поставщиками	Поставщик Иванов	NULL	830	20
Товары	Поварешка	Север	Расчеты с поставщиками	Поставщик Иванов	NULL	200	1
Товары	Поварешка	Юг	Расчеты с поставщиками	Поставщик Иванов	NULL	100	1
Товары	Вилка	Юг	Расчеты с поставщиками	Поставщик Иванов	NULL	200	10
Товары	Ложка	Юг	Расчеты с поставщиками	Поставщик Иванов	NULL	100	10

Но все же счет «Товары» отражает Остатки товаров. Уменьшение остатков, это не всегда продажи, это может быть и закупки и т.д.

Проанализируем счет «Прибыль»:

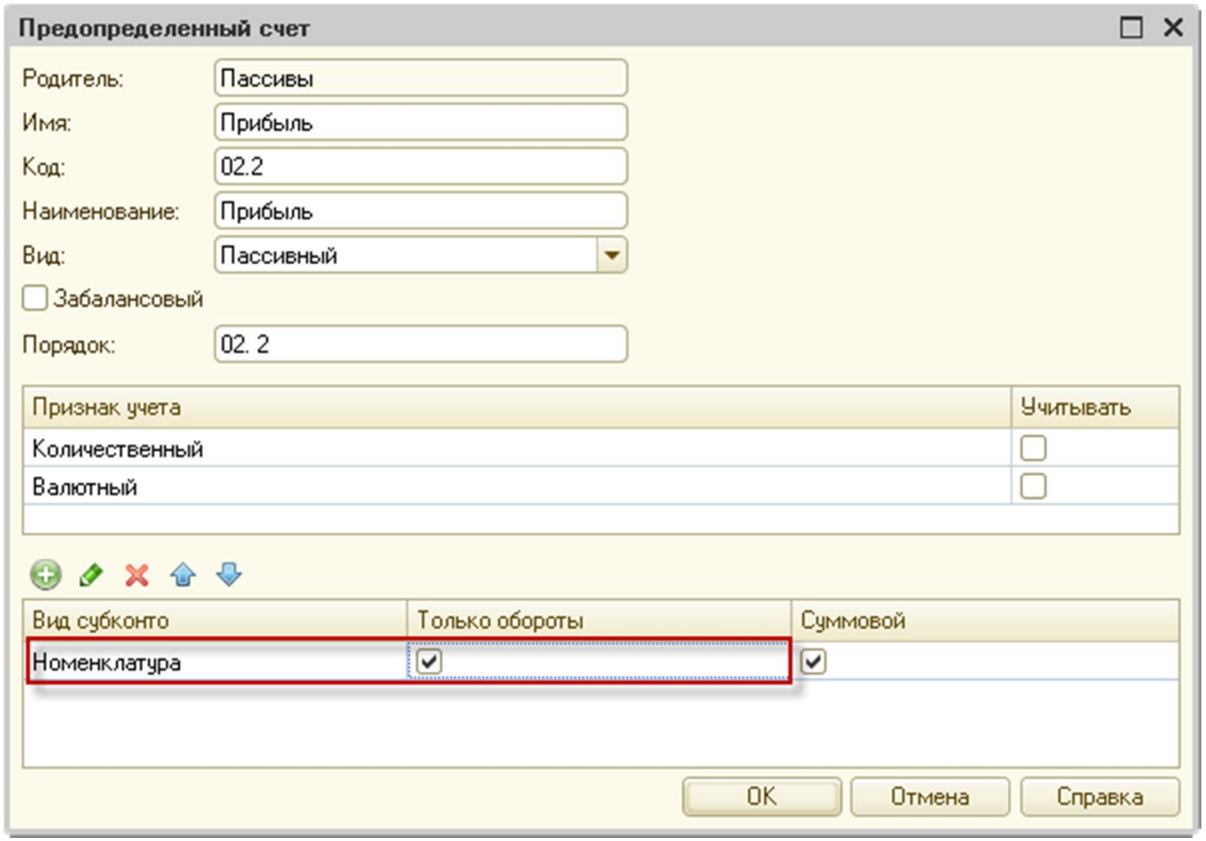
Счет	Субконто1	Субконто2	КорСчет	КорСубконто1	КорСубконто2	СуммаОборот	КоличествоОборот
Прибыль	NULL	NULL	Расчеты с покупателями	Покупатель МИР	NULL	-1 000	0
Прибыль	NULL	NULL	Расчеты с поставщиками	Поставщик Иванов	NULL	-74,1	0
Прибыль	NULL	NULL	Товары	Поварешка	Юг	100	0
Прибыль	NULL	NULL	Товары опт	Поварешка	Север	150	0

т.е. когда продаем товар (себестоимость):
Дт Прибыль Кт Товары

когда учитываем выручку:
Дт Покупатели Кт Прибыль
(потому в результате запроса суммы оборота идут с минусом)

В данной ситуации Выручка не разрезана по Номенклатуре, т.е. это единственный показатель, который на данный момент мы не можем получить для отчета.

Решение: по счету Прибыль надо организовать аналитику, а т.к. счет Прибыль накопительный (накапливаются обороты), то добавим только оборотную аналитику (остатки храниться не будут):



Чтобы по этой аналитике накапливались значения, документ «Расходная» должен формировать движения в разрезе Номенклатуры по Выручке.

Перенесем движения по Выручке в цикл по табличной части документа. Вместо итога по графе «Сумма» надо записать сумму выручки, которую надо получить в запросе из документа и вытащить ее в основной запрос. В движения по счету Прибыль добавим аналитику. Итого код:

```
Процедура ПроведениеОпт(Отказ)
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    РасходнаяТовары.Номенклатура,
    |    СУММА(РасходнаяТовары.Количество) КАК Количество,
    |    СУММА(РасходнаяТовары.Сумма) КАК Сумма
```

```
|ПОМЕСТИТЬ ДокТЧ
|ИЗ
|    Документ.Расходная.Товары КАК РасходнаяТовары
|ГДЕ
|    РасходнаяТовары.Ссылка = &Ссылка
|
|СГРУППИРОВАТЬ ПО
|    РасходнаяТовары.Номенклатура
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|    УправленческийОстатки.Субконто1,
|    СУММА(УправленческийОстатки.СуммаОстаток) КАК СуммаОстаток,
|    СУММА(УправленческийОстатки.КоличествоОстаток) КАК КоличествоОстаток,
|    СУММА(ВЫБОР
|              КОГДА УправленческийОстатки.Субконто2 = &Склад
|                   ТОГДА УправленческийОстатки.КоличествоОстаток
|         КОНЕЦ) КАК ОстатокНаСкладе
|ПОМЕСТИТЬ Остатки
|ИЗ
|    РегистрБухгалтерии.Управленческий.Остатки(
|              &МоментВремени,
|              Счет = &Счет,
|              &МассивСубконто,
|              Субконто1 В
|                   (ВЫБРАТЬ
|                        ДокТЧ.Номенклатура
|                   ИЗ
|                        ДокТЧ КАК ДокТЧ)) КАК УправленческийОстатки
|
|СГРУППИРОВАТЬ ПО
|    УправленческийОстатки.Субконто1
|;
|
|////////////////////////////////////////////////////////////////////////////
```

```
|ВЫБРАТЬ
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Количество,
|    ДокТЧ.Сумма,
|    Остатки.КоличествоОстаток,
|    Остатки.СуммаОстаток,
|    ЕСТЬNULL(Остатки.ОстатокНаСкладе, 0) КАК ОстатокНаСкладе
|ИЗ
|    ДокТЧ КАК ДокТЧ
|        ЛЕВОЕ СОЕДИНЕНИЕ Остатки КАК Остатки
|        ПО ДокТЧ.Номенклатура = Остатки.Субконто1";

Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.ТоварыОпт);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

    Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
        + ". Из необходимых " + Выборка.Количество +
        " в наличии имеется только " + Выборка.КоличествоОстаток;
        Сообщение.Сообщить();

        Движения.Управленческий.Записывать = Ложь;
```

```
            Отказ = Истина;
        КонецЕсли;

        Если Отказ Тогда
            Продолжить;
        КонецЕсли;

        Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
        Движение.СчетКт = ПланыСчетов.Управленческий.ТоварыОпт;
        Движение.Период = Дата;
        Движение.Сумма = Себестоимость;
        Движение.КоличествоКт = Выборка.Количество;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
        Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.Период = Дата;
        Движение.Сумма = Выборка.Сумма;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
    КонецЦикла;
КонецПроцедуры
```

Аналогично поправляем процедуру проведения по Рознице:

```
Процедура ПроведениеРозница(Отказ)
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
```

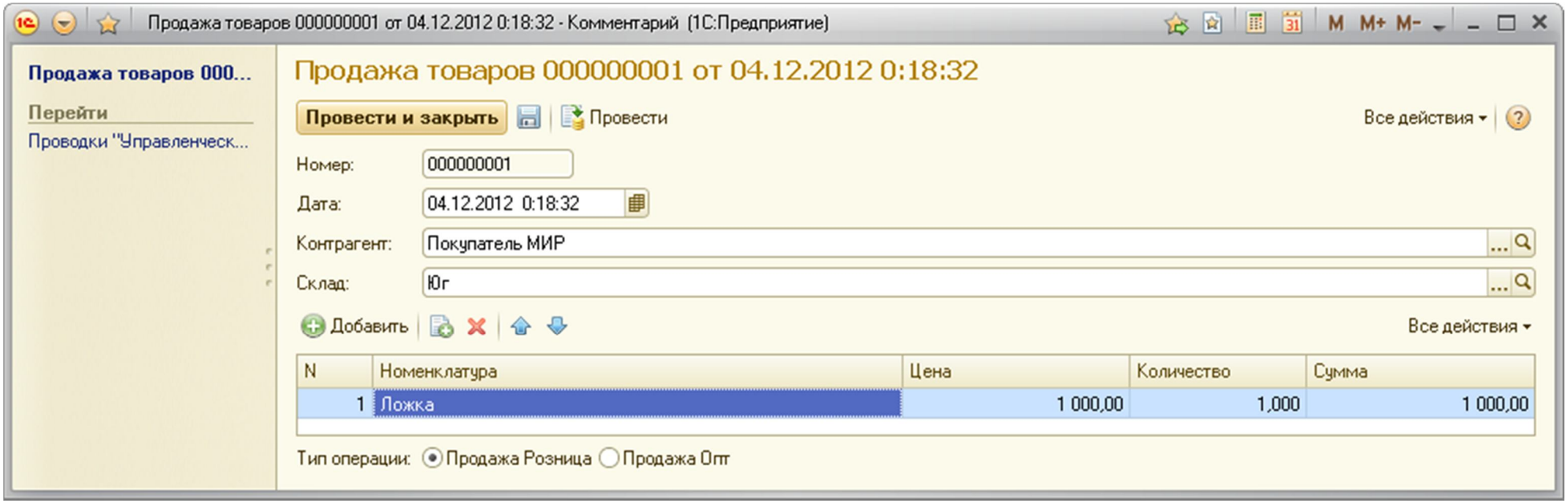
```
|    РасходнаяТовары.Номенклатура,
|    СУММА(РасходнаяТовары.Количество) КАК Количество,
|    СУММА(Сумма) КАК Сумма
|ПОМЕСТИТЬ ДокТЧ
|ИЗ
|    Документ.Расходная.Товары КАК РасходнаяТовары
|ГДЕ
|    РасходнаяТовары.Ссылка = &Ссылка
|
|СГРУППИРОВАТЬ ПО
|    РасходнаяТовары.Номенклатура
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Количество,
|    ДокТЧ.Сумма,
|    ЕСТЬNULL(УправленческийОстатки.СуммаОстаток, 0) КАК СуммаОстаток,
|    ЕСТЬNULL(УправленческийОстатки.КоличествоОстаток, 0) КАК КоличествоОстаток
|ИЗ
|    ДокТЧ КАК ДокТЧ
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрБухгалтерии.Управленческий.Остатки(
|                &МоментВремени,
|                Счет = &Счет,
|                &МассивСубконто,
|                Субконто1 В
|                        (ВЫБРАТЬ
|                            ДокТЧ.Номенклатура
|                        ИЗ
|                            ДокТЧ КАК ДокТЧ)
|                    И Субконто2 = &Склад) КАК УправленческийОстатки
|        ПО ДокТЧ.Номенклатура = УправленческийОстатки.Субконто1";

Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
```

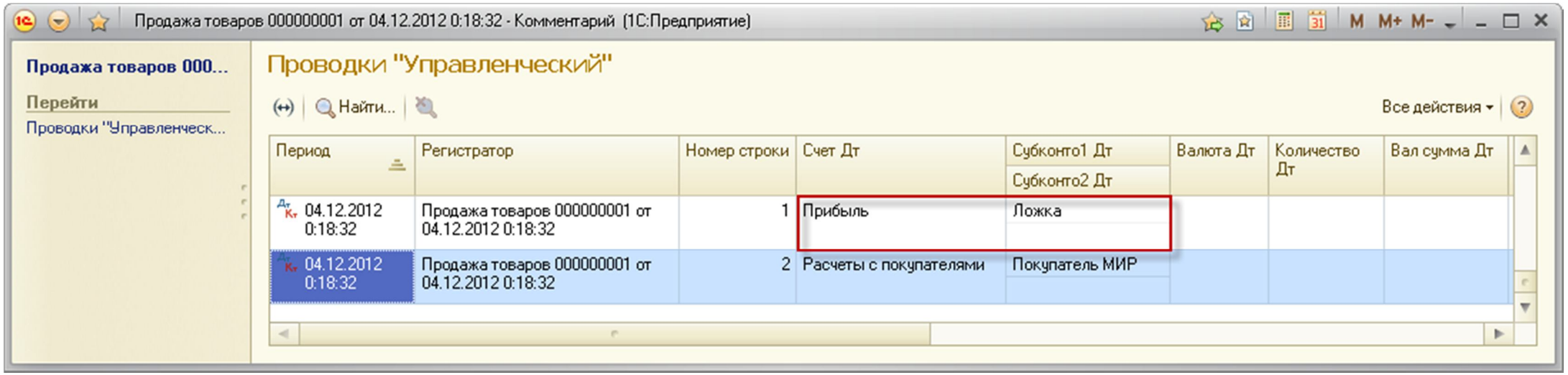
```
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.Товары);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

     Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
          + ". Из необходимых " + Выборка.Количество +
          " в наличии имеется только " + Выборка.КоличествоОстаток;
          Сообщение.Сообщить();

          Движения.Управленческий.Записывать = Ложь;
          Отказ = Истина;
     КонецЕсли;

     Если Отказ Тогда
          Продолжить;
     КонецЕсли;

     Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;

     Движение = Движения.Управленческий.Добавить();
     Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
     Движение.СчетКт = ПланыСчетов.Управленческий.Товары;
     Движение.Период = Дата;
     Движение.Сумма = Себестоимость;
```

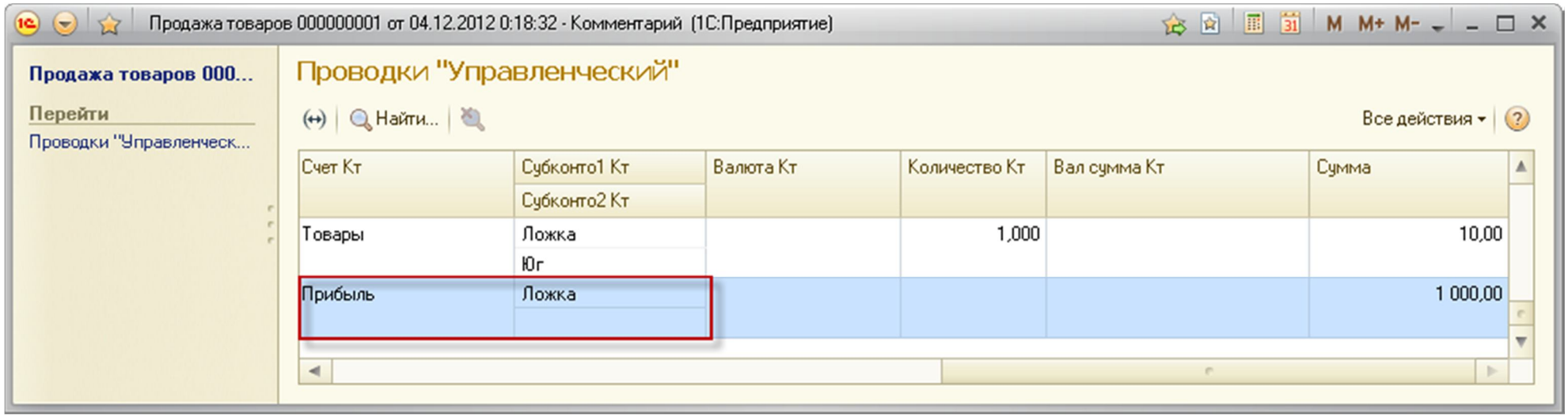
```
        Движение.КоличествоКт = Выборка.Количество;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
        Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
        Движение.Период = Дата;
        Движение.Сумма = Товары.Итог("Сумма");
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;
    КонецЦикла;
КонецПроцедуры
```

В режиме исполнения:

Перепроведем документы Расходная:

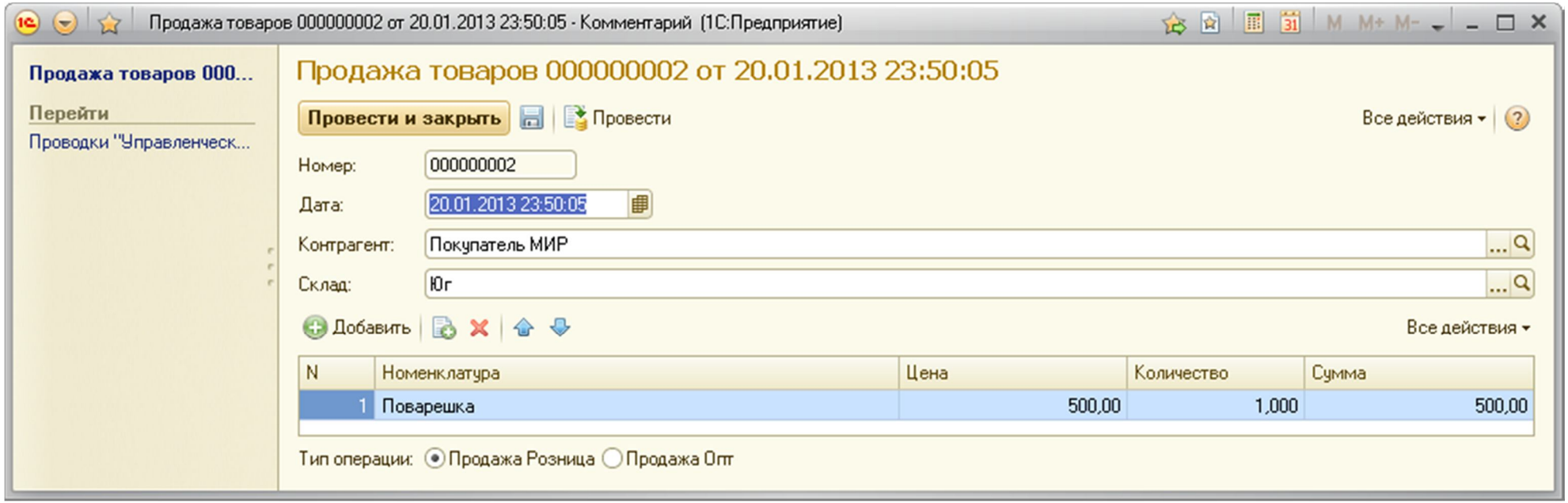


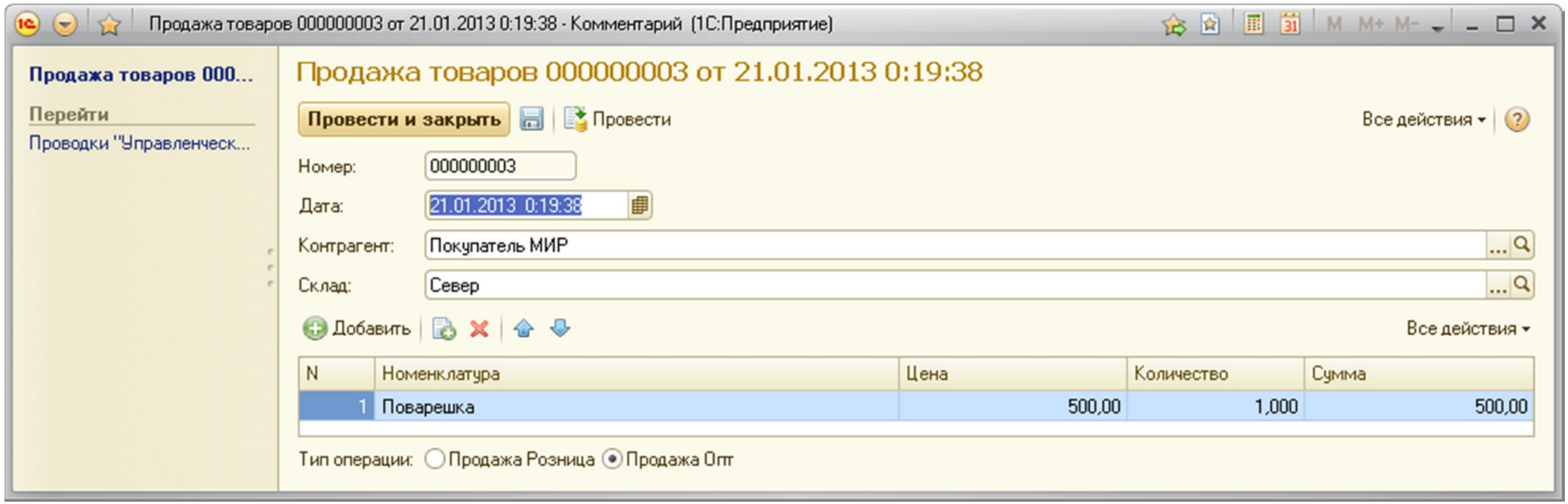




Теперь и Себестоимость и Выручка имеют разрез по Номенклатуре.

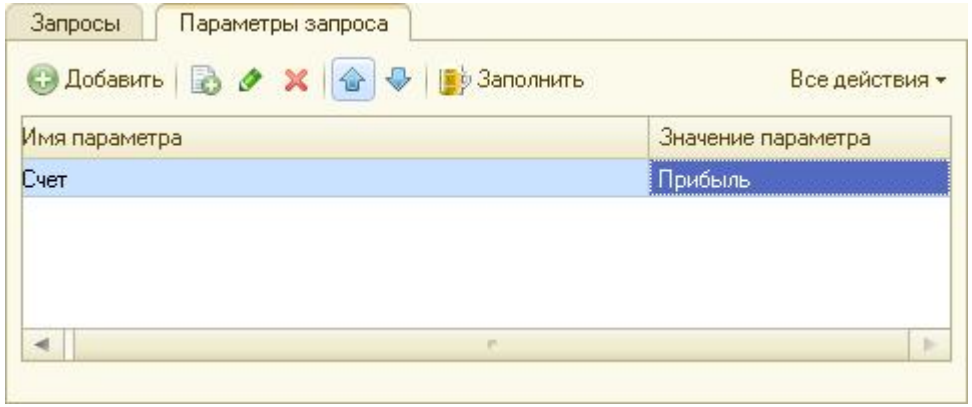
Перепроведем остальные документы:





Проверим таблицу оборотов на наличие всех необходимых для формирования отчета данных:

```
ВЫБРАТЬ
	*
ИЗ
	РегистрБухгалтерии.Управленческий.Обороты(, , , Счет = &Счет, , , , ) КАК УправленческийОбороты
```



Дт	Кт	Содержание
Прибыль	Товары	Себестоимость продаж
Расчеты с покупателями	Прибыль	Выручка

КоличествоКорОборот – по счету Прибыль количество не хранится, но в корреспонденции с ним Количество есть, т.к. корреспондирующий счет Товары является Количественным.

Получим информацию о продажах:
Субконто1 – сделаем отбор только по Номенклатуре (параметры регистра).
СуммаОборотДт – себестоимость продаж.
СуммаОборотКт – выручка.
КоличествоКорОборотДт – количество продаж.

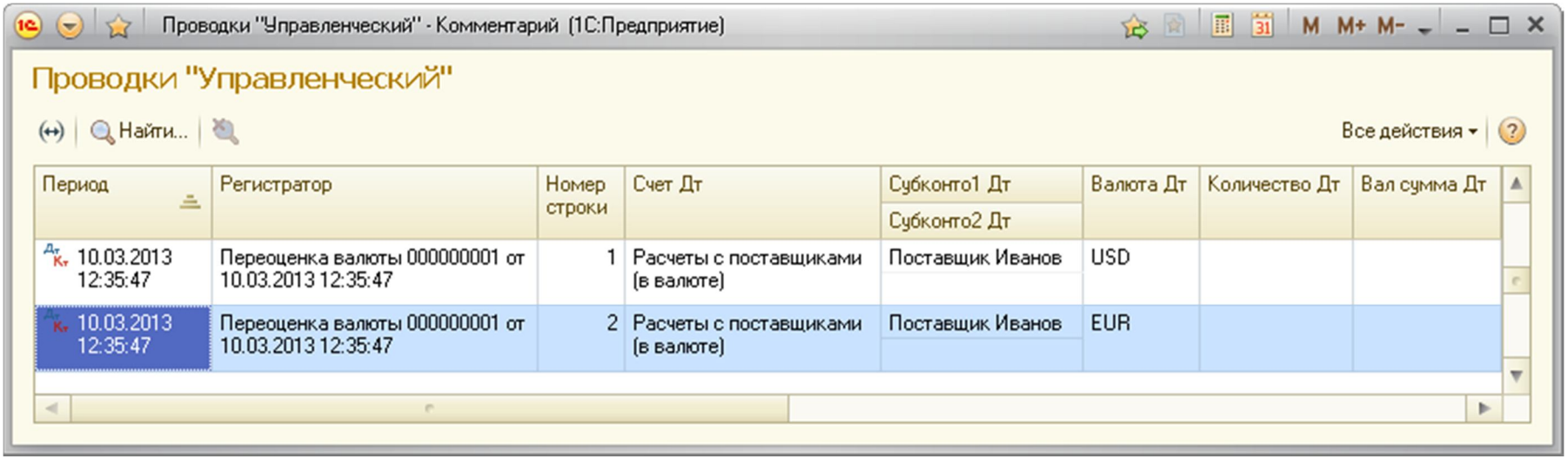
Итого:

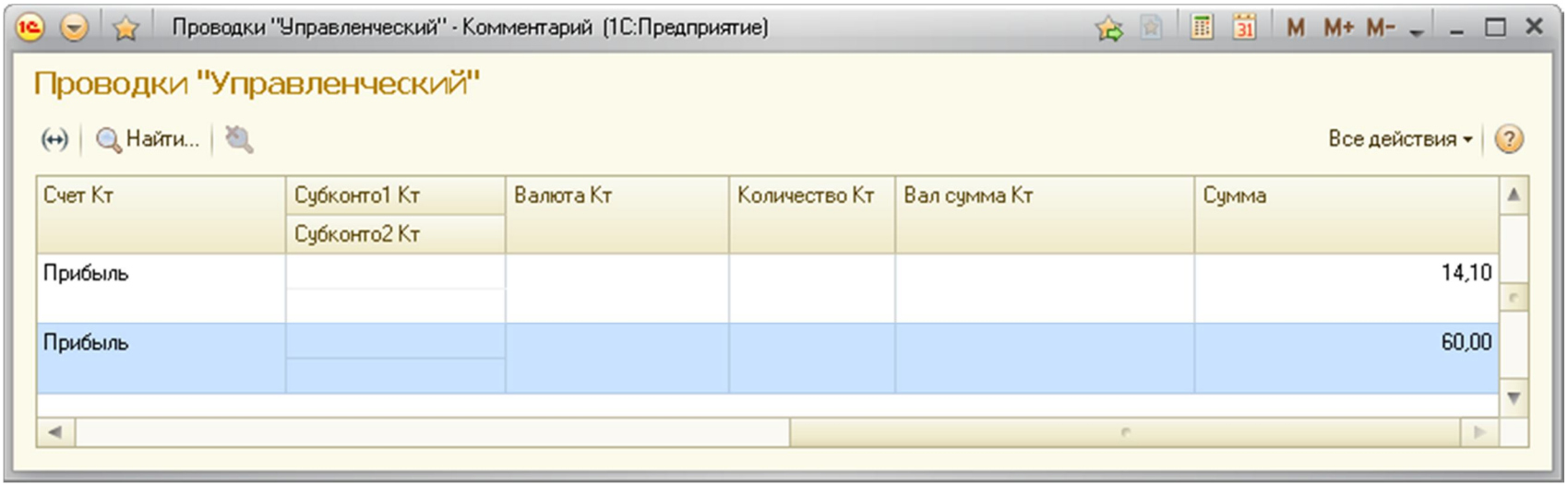
```
ВЫБРАТЬ
      УправленческийОбороты.Субконто1,
      СуммаОборотДт,
      СуммаОборотКт,
      КоличествоКорОборотДт
ИЗ
      РегистрБухгалтерии.Управленческий.Обороты(, , , Счет = &Счет, &ВидСубконто, , , ) КАК УправленческийОбороты
```

Субконто1	СуммаОборотДт	СуммаОборотКт	КоличествоКорОб
	0	74,1	8
Поварешка	250	1 000	2
Ложка	10	1 000	1

курсовая разница

Откуда она взялась?
Откроем Регистр бухгалтерии:

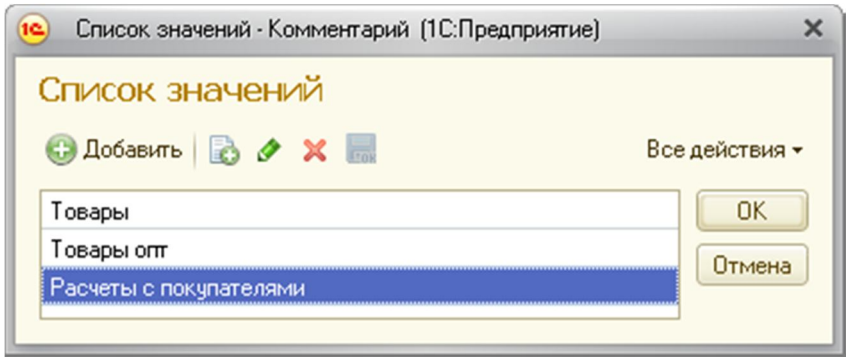




Чтобы курсовая разница не отображалась в отчете, надо:

- Наложить условия на корреспондирующие счета:

```
ВЫБРАТЬ
      УправленческийОбороты.Субконто1,
      УправленческийОбороты.СуммаОборотДт,
      УправленческийОбороты.СуммаОборотКт,
      УправленческийОбороты.КоличествоКорОборотДт
ИЗ
      РегистрБухгалтерии.Управленческий.Обороты(, , , Счет = &Счет, &ВидСубконто, , КорСчет В (&КорСчета), )
КАК УправленческийОбороты
```



Итого:

Субконто1	СуммаОборотДт	СуммаОборотКт	КоличествоКорОборотДт
Поварешка	250	1 000	2
Ложка	10	1 000	1

или

- Ввести перечисление «Тип операции».

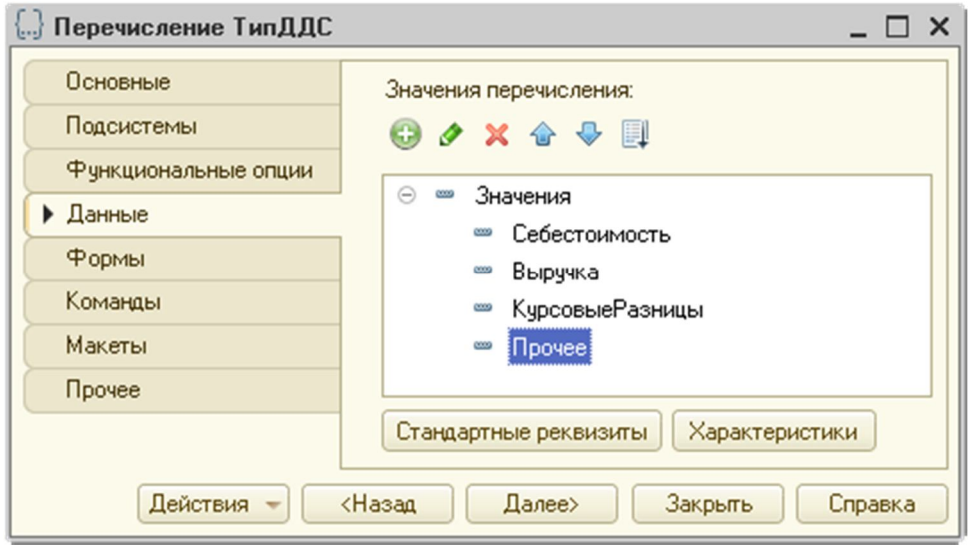
Если счет расчетов с покупателями или поставщиками будет в валюте, то в результат дополнительно попадут движения по переоценке. Например:
Дт Расчеты с поставщиками (в валюте) Кт Прибыль – это увеличение выручки и переоценка.

Чтобы не попадали лишние движения, надо к счету «Прибыль» привязать еще одно субконто – по типу операции.

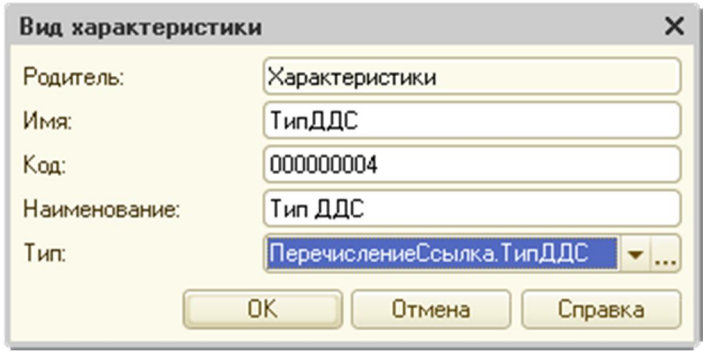
Добавим перечисление:

Свойство	Значение
Имя	ТипДДС
Синоним	Тип ДДС

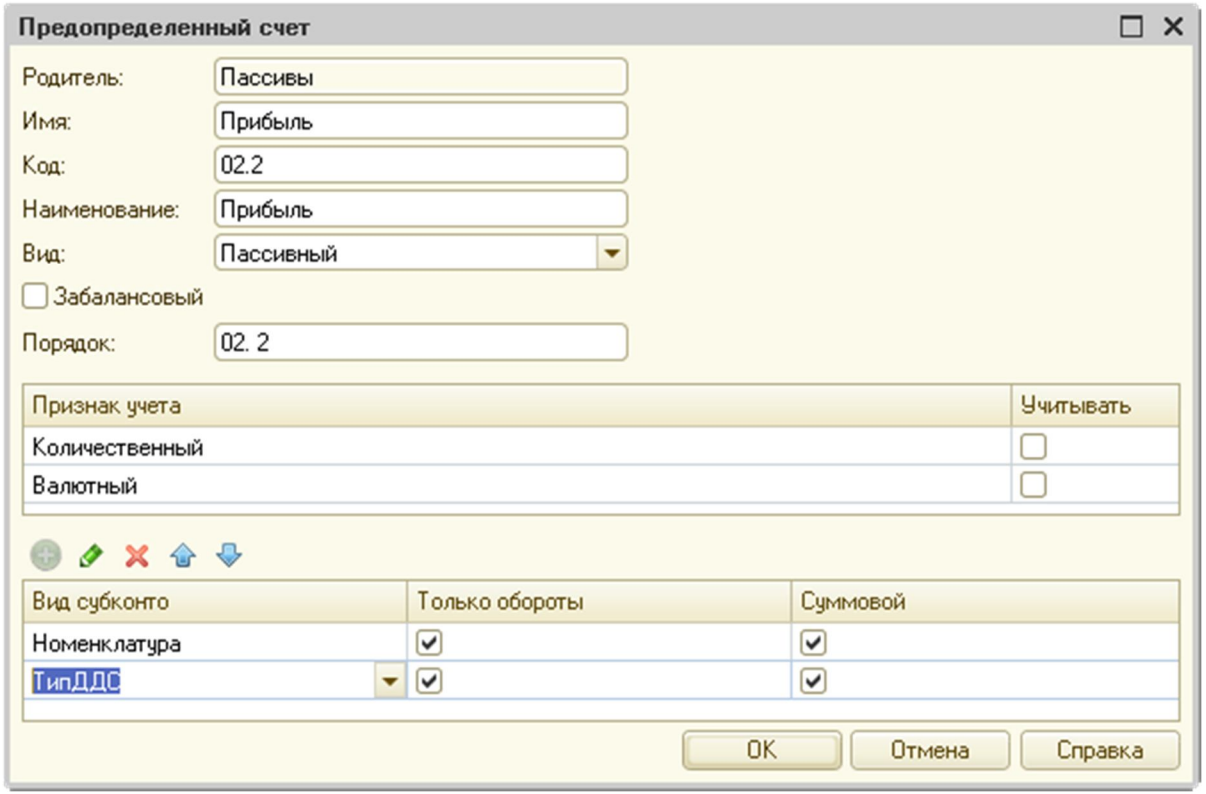
В Данные добавим значения:



Добавим перечисление в Виды субконто (свойство «Тип значения характеристик»). И предопределенное субконто:



Привяжем этот вид аналитики к счету «Прибыль»:



Поправим движения по документу «Расходная»:

```
Процедура ПроведениеОпт(Отказ)
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    РасходнаяТовары.Номенклатура,
    |    СУММА(РасходнаяТовары.Количество) КАК Количество,
    |    СУММА(РасходнаяТовары.Сумма) КАК Сумма
    |ПОМЕСТИТЬ ДокТЧ
    |ИЗ
    |    Документ.Расходная.Товары КАК РасходнаяТовары
    |ГДЕ
```

```
|    РасходнаяТовары.Ссылка = &Ссылка
|
|СГРУППИРОВАТЬ ПО
|    РасходнаяТовары.Номенклатура
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|    УправленческийОстатки.Субконто1,
|    СУММА(УправленческийОстатки.СуммаОстаток) КАК СуммаОстаток,
|    СУММА(УправленческийОстатки.КоличествоОстаток) КАК КоличествоОстаток,
|    СУММА(ВЫБОР
|              КОГДА УправленческийОстатки.Субконто2 = &Склад
|                   ТОГДА УправленческийОстатки.КоличествоОстаток
|         КОНЕЦ) КАК ОстатокНаСкладе
|ПОМЕСТИТЬ Остатки
|ИЗ
|    РегистрБухгалтерии.Управленческий.Остатки(
|              &МоментВремени,
|              Счет = &Счет,
|              &МассивСубконто,
|              Субконто1 В
|                   (ВЫБРАТЬ
|                        ДокТЧ.Номенклатура
|                   ИЗ
|                        ДокТЧ КАК ДокТЧ)) КАК УправленческийОстатки
|
|СГРУППИРОВАТЬ ПО
|    УправленческийОстатки.Субконто1
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Количество,
|    ДокТЧ.Сумма,
```

```
|    Остатки.КоличествоОстаток,
|    Остатки.СуммаОстаток,
|    ЕСТЬNULL(Остатки.ОстатокНаСкладе, 0) КАК ОстатокНаСкладе
|ИЗ
|    ДокТЧ КАК ДокТЧ
|         ЛЕВОЕ СОЕДИНЕНИЕ Остатки КАК Остатки
|         ПО ДокТЧ.Номенклатура = Остатки.Субконто1";

Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.ТоварыОпт);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

     Если Выборка.Количество > Выборка.ОстатокНаСкладе Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
          + ". Из необходимых " + Выборка.Количество +
          " в наличии имеется только " + Выборка.КоличествоОстаток;
          Сообщение.Сообщить();

          Движения.Управленческий.Записывать = Ложь;
          Отказ = Истина;
     КонецЕсли;

     Если Отказ Тогда
```

```
            Продолжить;
        КонецЕсли;

        Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
        Движение.СчетКт = ПланыСчетов.Управленческий.ТоварыОпт;
        Движение.Период = Дата;
        Движение.Сумма = Себестоимость;
        Движение.КоличествоКт = Выборка.Количество;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.ТипДДС] = Перечисления.ТипДДС.Себестоимость;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
        Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.ТипДДС] = Перечисления.ТипДДС.Выручка;
        Движение.Период = Дата;
        Движение.Сумма = Выборка.Сумма;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;

    КонецЦикла;

КонецПроцедуры

Процедура ПроведениеРозница(Отказ)
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    РасходнаяТовары.Номенклатура,
    |    СУММА(РасходнаяТовары.Количество) КАК Количество,
    |    СУММА(Сумма) КАК Сумма
```

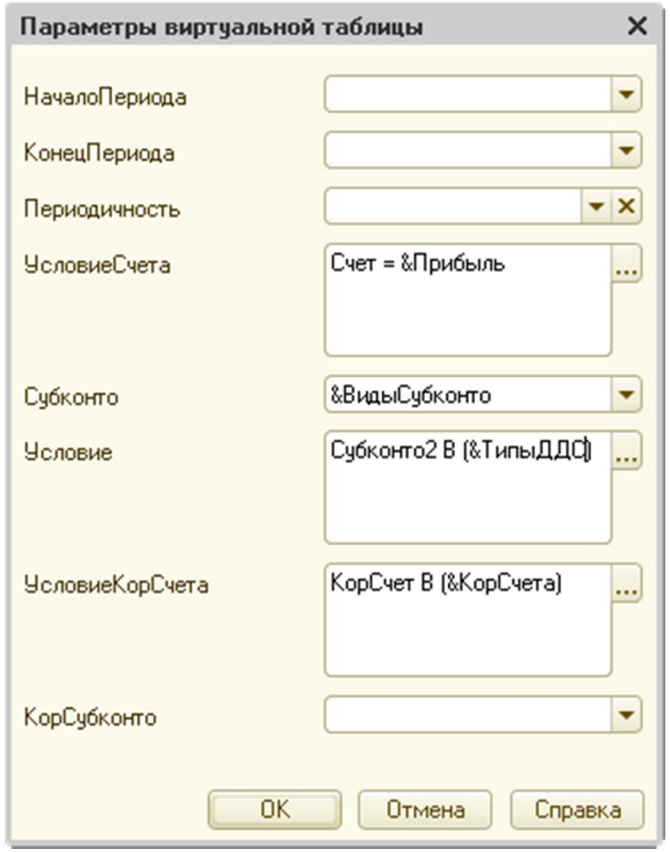
```
|ПОМЕСТИТЬ ДокТЧ
|ИЗ
|    Документ.Расходная.Товары КАК РасходнаяТовары
|ГДЕ
|    РасходнаяТовары.Ссылка = &Ссылка
|
|СГРУППИРОВАТЬ ПО
|    РасходнаяТовары.Номенклатура
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|    ДокТЧ.Номенклатура,
|    ДокТЧ.Количество,
|    ДокТЧ.Сумма,
|    ЕСТЬNULL(УправленческийОстатки.СуммаОстаток, 0) КАК СуммаОстаток,
|    ЕСТЬNULL(УправленческийОстатки.КоличествоОстаток, 0) КАК КоличествоОстаток
|ИЗ
|    ДокТЧ КАК ДокТЧ
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрБухгалтерии.Управленческий.Остатки(
|                &МоментВремени,
|                Счет = &Счет,
|                &МассивСубконто,
|                Субконто1 В
|                        (ВЫБРАТЬ
|                            ДокТЧ.Номенклатура
|                        ИЗ
|                            ДокТЧ КАК ДокТЧ)
|                    И Субконто2 = &Склад) КАК УправленческийОстатки
|        ПО ДокТЧ.Номенклатура = УправленческийОстатки.Субконто1";

Запрос.УстановитьПараметр("Ссылка", Ссылка);
Запрос.УстановитьПараметр("МоментВремени", МоментВремени());
Запрос.УстановитьПараметр("Счет", ПланыСчетов.Управленческий.Товары);
МассивСубконто = Новый Массив(2);
МассивСубконто[0] = ПланыВидовХарактеристик.ВидыСубконто.Номенклатура;
```

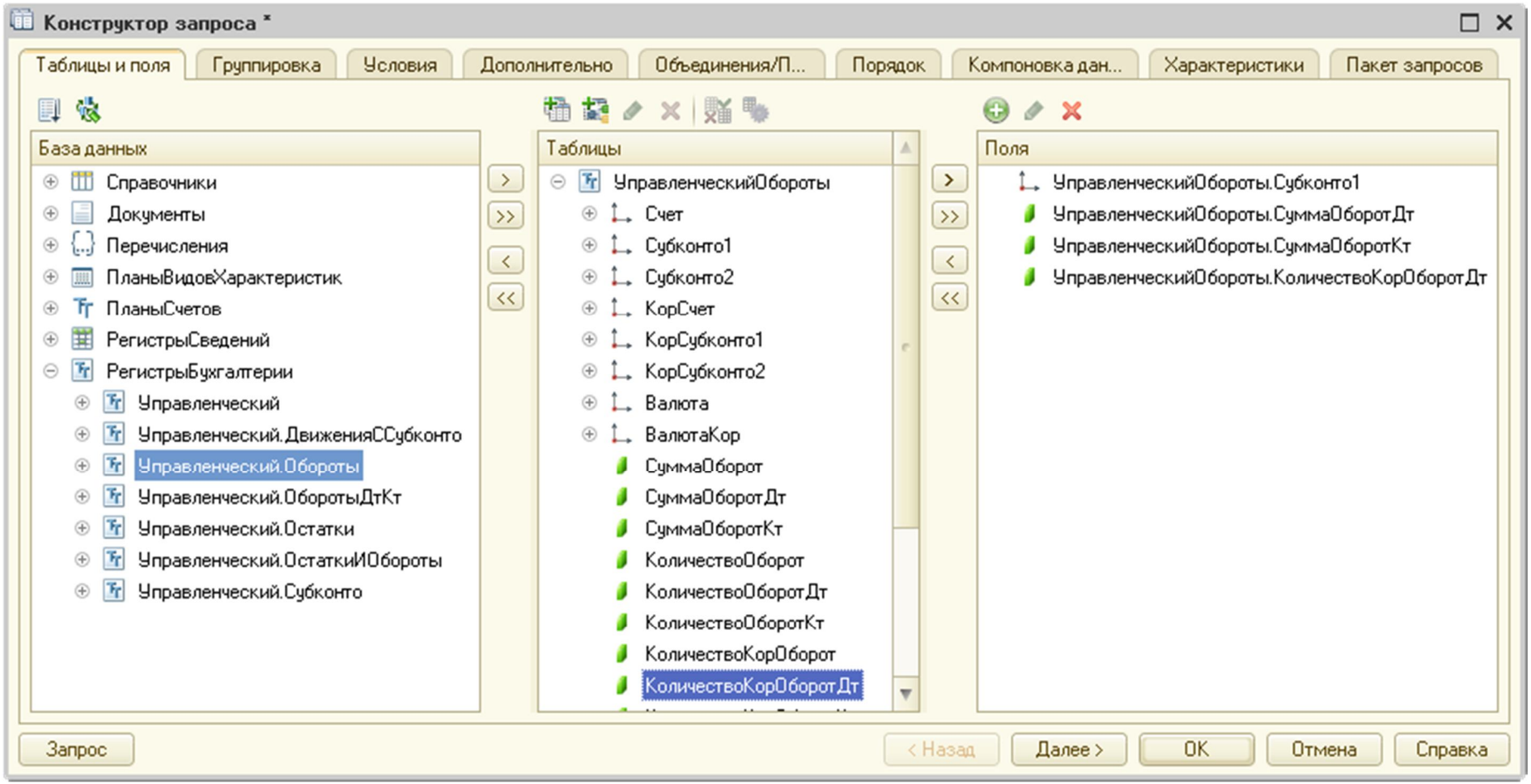
```
МассивСубконто[1] = ПланыВидовХарактеристик.ВидыСубконто.Склады;
Запрос.УстановитьПараметр("МассивСубконто", МассивСубконто);
Запрос.УстановитьПараметр("Склад", Склад);

РезультатЗапроса = Запрос.Выполнить();

Движения.Управленческий.Записывать = Истина;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл

     Если Выборка.Количество > Выборка.КоличествоОстаток Тогда
          Сообщение = Новый СообщениеПользователю;
          Сообщение.Текст = "Не хватает товара " + Выборка.Номенклатура
          + ". Из необходимых " + Выборка.Количество +
          " в наличии имеется только " + Выборка.КоличествоОстаток;
          Сообщение.Сообщить();

          Движения.Управленческий.Записывать = Ложь;
          Отказ = Истина;
     КонецЕсли;

     Если Отказ Тогда
          Продолжить;
     КонецЕсли;

     Себестоимость = Выборка.Количество / Выборка.КоличествоОстаток * Выборка.СуммаОстаток;

     Движение = Движения.Управленческий.Добавить();
     Движение.СчетДт = ПланыСчетов.Управленческий.Прибыль;
     Движение.СчетКт = ПланыСчетов.Управленческий.Товары;
     Движение.Период = Дата;
     Движение.Сумма = Себестоимость;
     Движение.КоличествоКт = Выборка.Количество;
     Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.ТипДДС] = Перечисления.ТипДДС.Себестоимость;
     Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
```

```
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Склады] = Склад;

        Движение = Движения.Управленческий.Добавить();
        Движение.СчетДт = ПланыСчетов.Управленческий.РасчетыСПокупателями;
        Движение.СчетКт = ПланыСчетов.Управленческий.Прибыль;
        Движение.Период = Дата;
        Движение.Сумма = Товары.Итог("Сумма");
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.ТипДДС] = Перечисления.ТипДДС.Выручка;
        Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Номенклатура] = Выборка.Номенклатура;
        Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Контрагенты] = Контрагент;

    КонецЦикла;

КонецПроцедуры
```

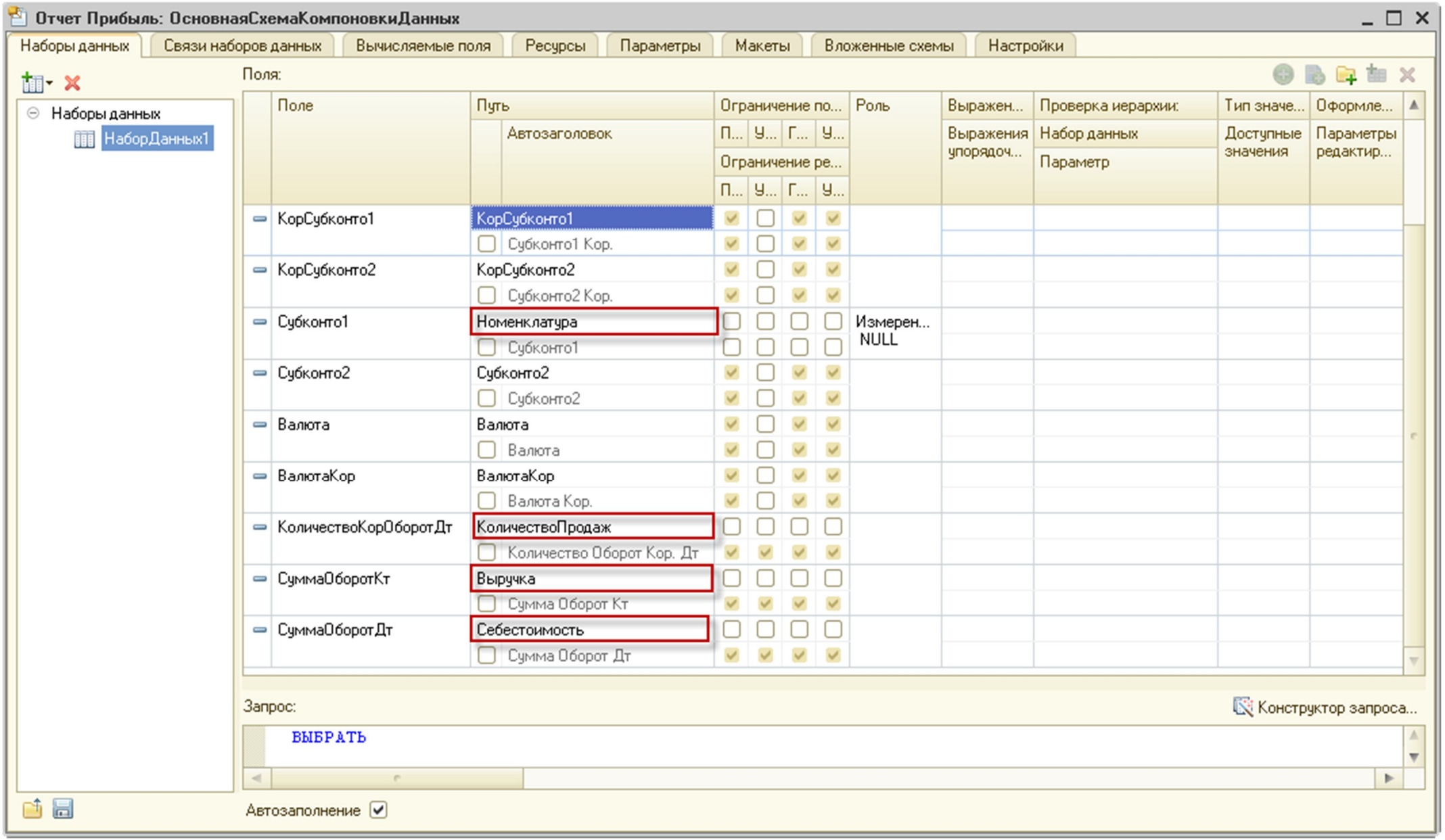
Теперь все готово для формирования отчета «Прибыль»:

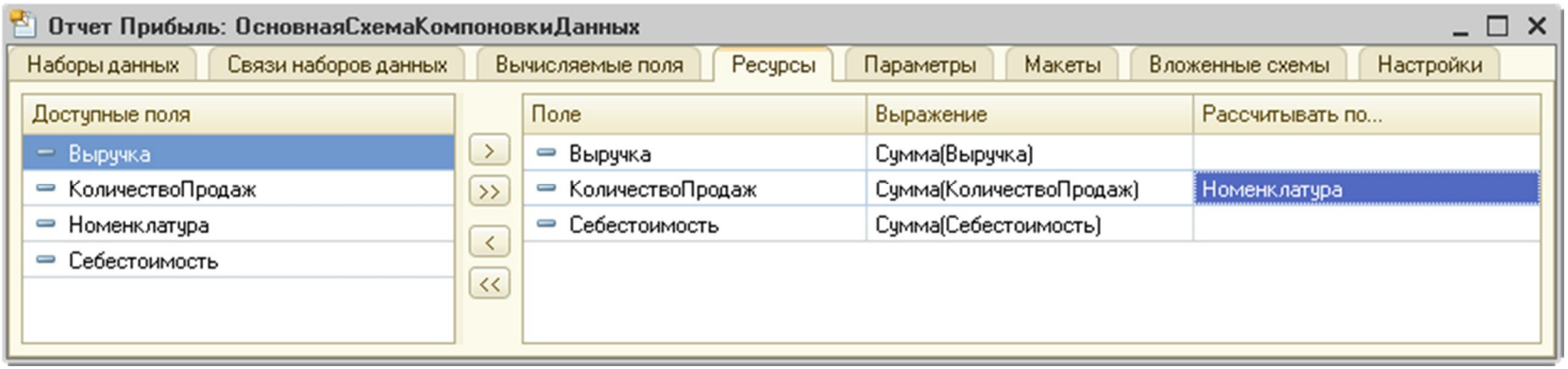
Откроем СКД – «Добавить набор данных-запрос» - «Конструктор запроса»:
Параметры виртуальной таблицы:



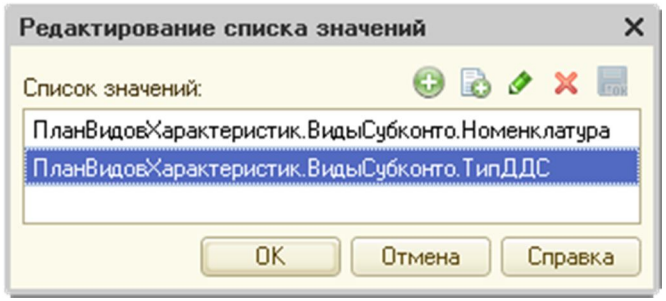

Конструктор запроса *
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/П...
Порядок
Компоновка дан...
Характеристики
Пакет запросов
База данных
Справочники
Документы
Перечисления
ПланыВидовХарактеристик
ПланыСчетов
РегистрыСведений
РегистрыБухгалтерии
Управленческий
Управленческий.ДвиженияССубконто
Управленческий.Обороты
Управленческий.ОборотыДтКт
Управленческий.Остатки
Управленческий.ОстаткиИОбороты
Управленческий.Субконто
Таблицы
УправленческийОбороты
Счет
Субконто1
Субконто2
КорСчет
КорСубконто1
КорСубконто2
Валюта
ВалютаКор
СуммаОборот
СуммаОборотДт
СуммаОборотКт
КоличествоОборот
КоличествоОборотДт
КоличествоОборотКт
КоличествоКорОборот
КоличествоКорОборотДт
Поля
УправленческийОбороты.Субконто1
УправленческийОбороты.СуммаОборотДт
УправленческийОбороты.СуммаОборотКт
УправленческийОбороты.КоличествоКорОборотДт
Запрос
< Назад
Далее >
OK
Отмена
Справка

В СКД изменим название некоторых полей, чтобы было нагляднее:

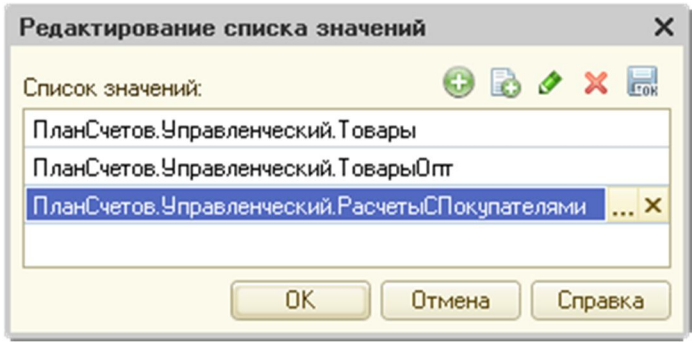




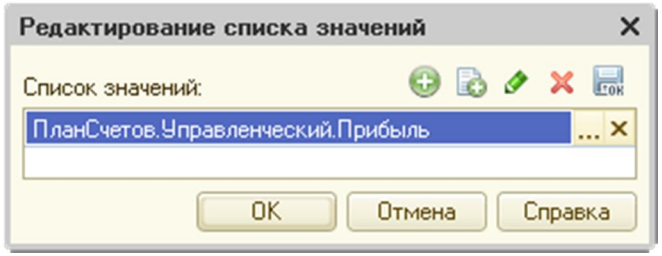
Закладка «Параметры»:
Виды субконто:



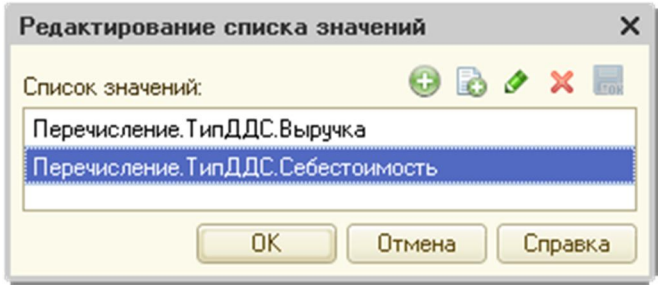
КорСчета:

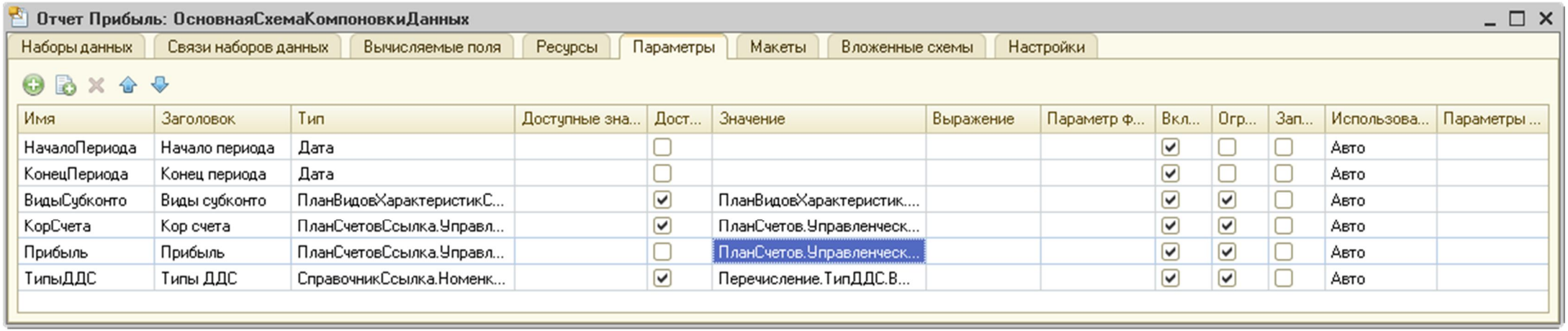


Прибыль:

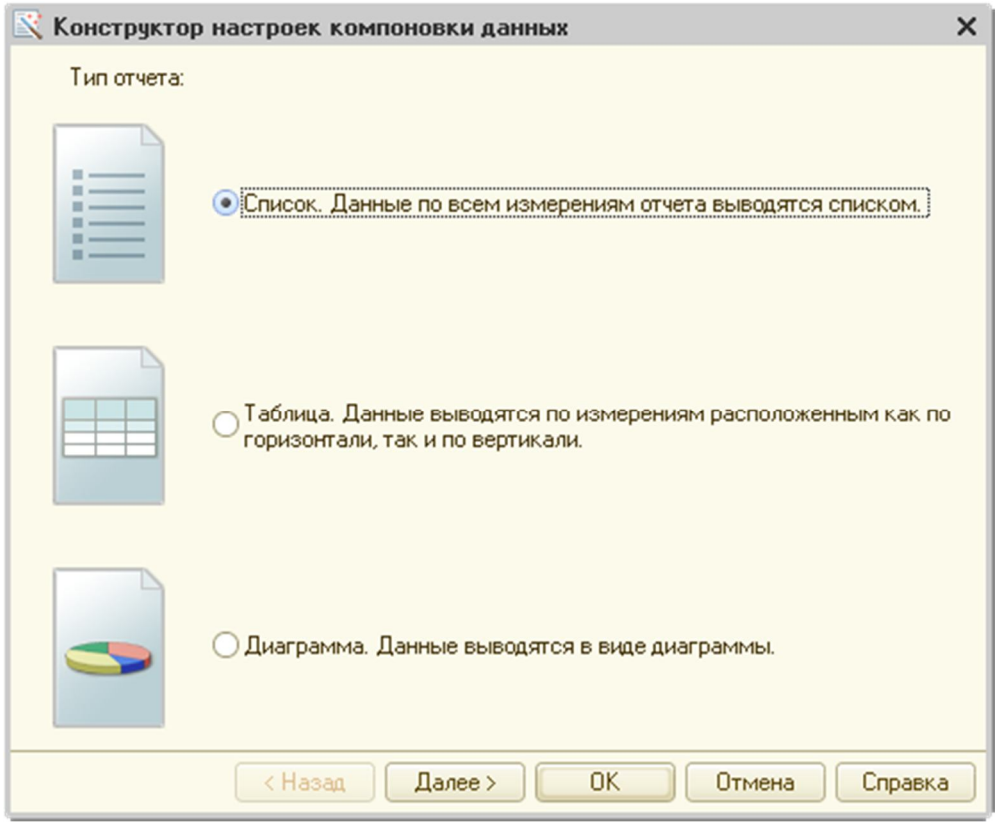


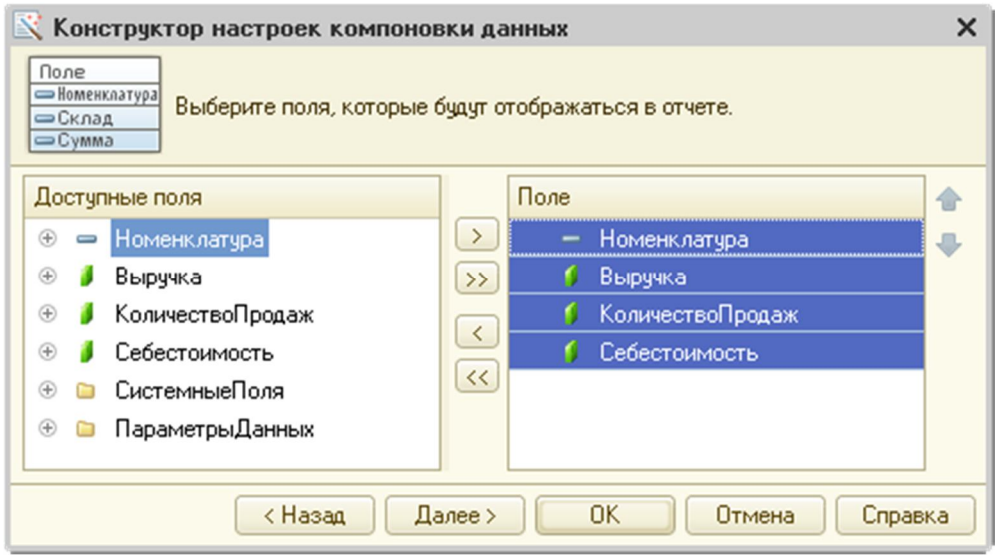
ТипыДДС:

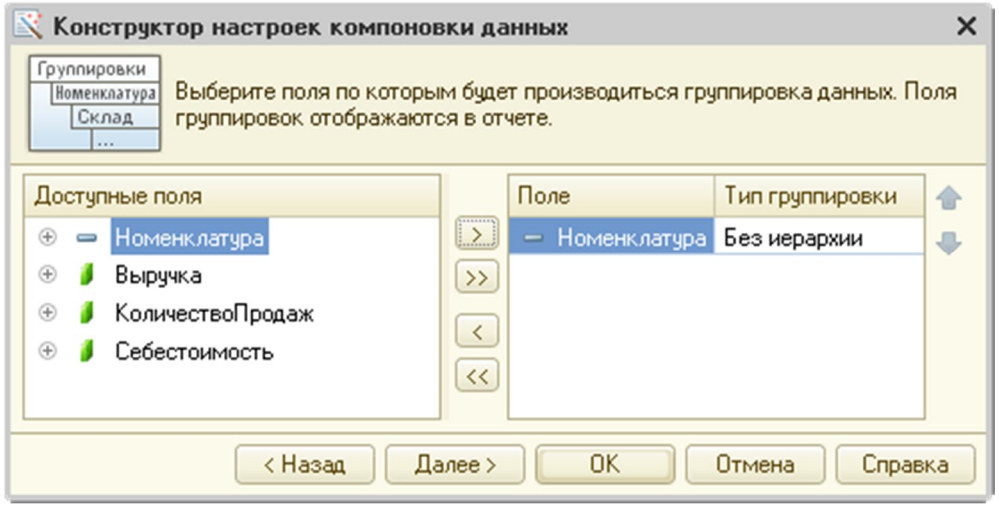




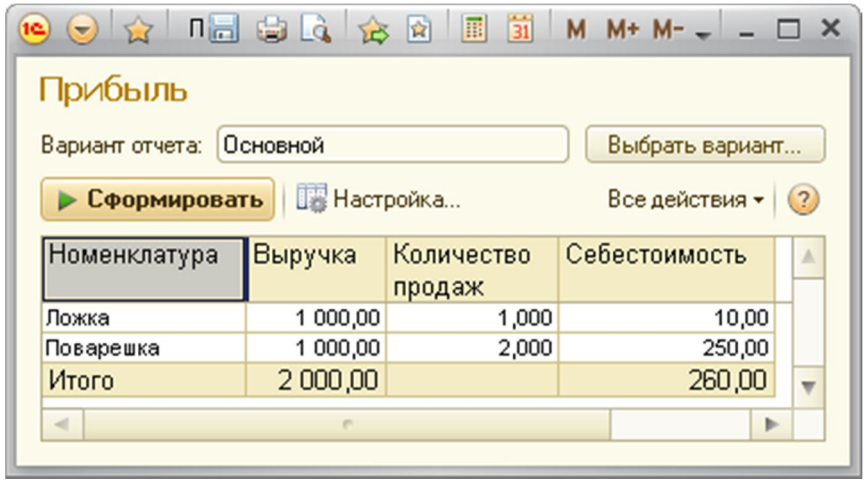
Закладка «Настройки»:



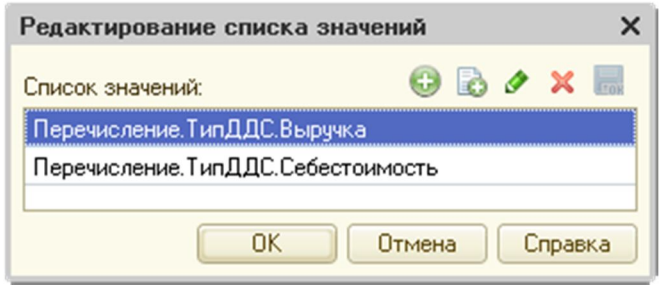

Конструктор настроек компоновки данных
Поле
Номенклатура
Склад
Сумма
Выберите поля, которые будут отображаться в отчете.
Доступные поля
Номенклатура
Выручка
КоличествоПродаж
Себестоимость
СистемныеПоля
ПараметрыДанных
Поле
Номенклатура
Выручка
КоличествоПродаж
Себестоимость
< Назад
Далее >
OK
Отмена
Справка


Конструктор настроек компоновки данных
Группировки
Номенклатура
Склад
...
Выберите поля по которым будет производиться группировка данных. Поля группировок отображаются в отчете.
Доступные поля
Номенклатура
Выручка
КоличествоПродаж
Себестоимость
Поле
Тип группировки
Номенклатура
Без иерархии
< Назад
Далее >
OK
Отмена
Справка

В режиме исполнения:
Перепроведем документы «Расходная» и сформируем отчет по прибыли:

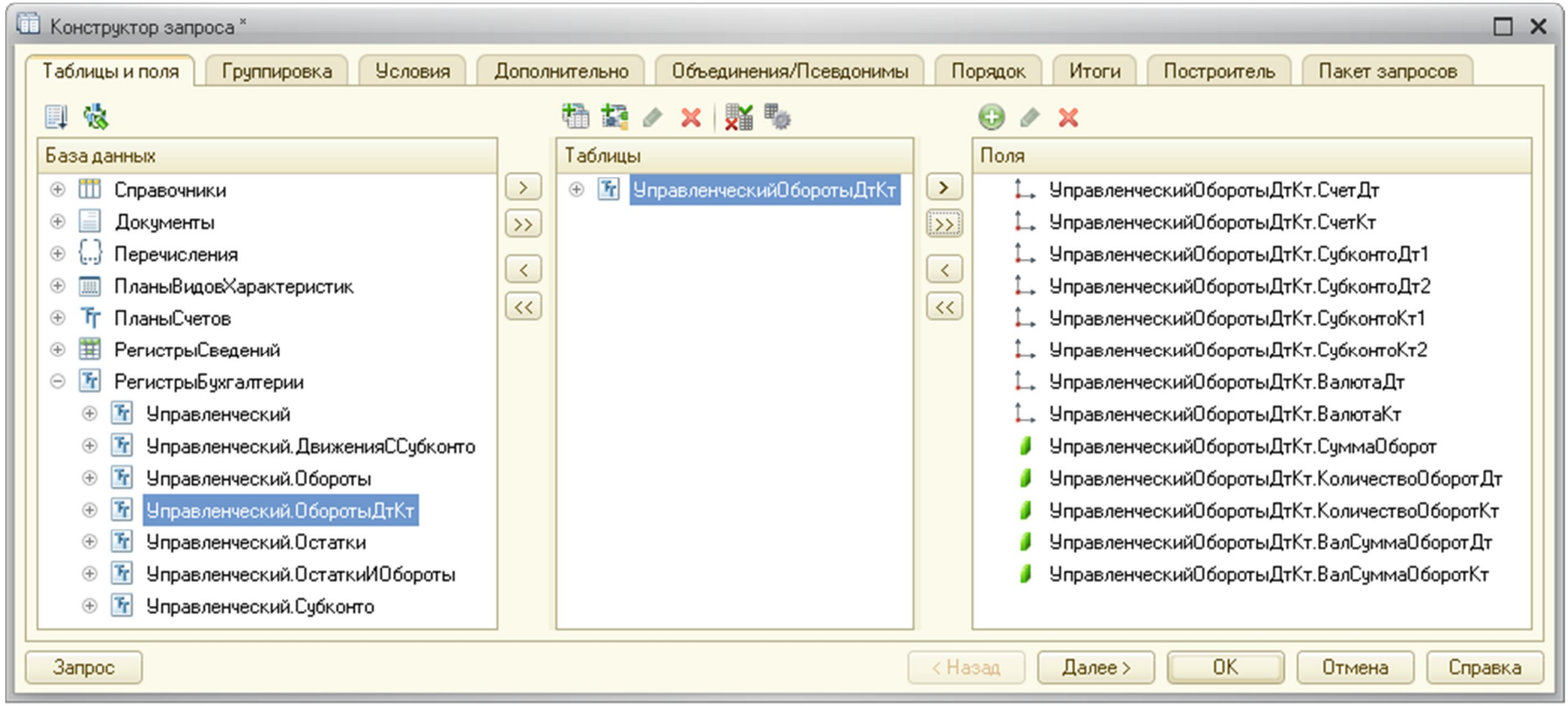


Без ограничения субконто по типам ДДС:



отчет может формироваться некорректно. Будут попадать лишние для данного отчета данные (например, от операции списания товара по инвентаризации, в брак и т.д.), особенно, если пользователю разрешено вводить операции вручную.

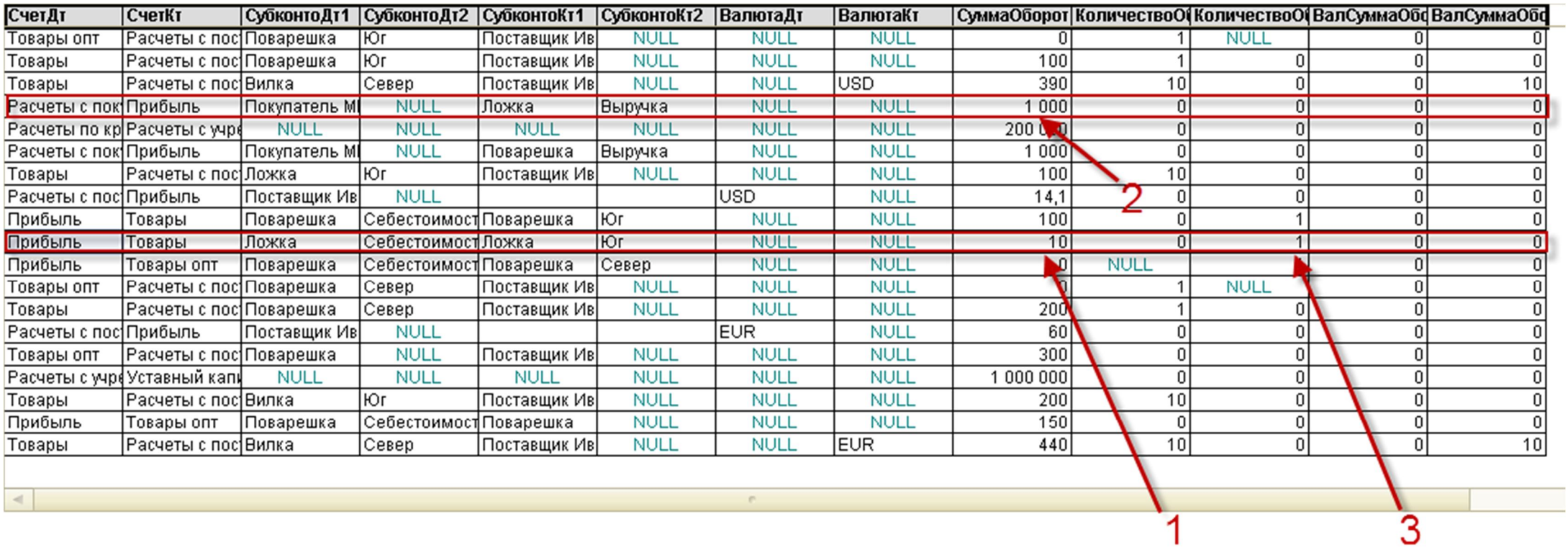
Таблица «ОборотыДтКт»:



Можно ли с помощью данной таблицы сделать наш отчет?

1 – себестоимость.
2 – выручка.
3 – количество.

СчетДт	СчетКт	СубконтоДт1	СубконтоДт2	СубконтоКт1	СубконтоКт2	ВалютаДт	ВалютаКт	СуммаОборот	КоличествоОб	КоличествоОб	ВалСуммаОбо	ВалСуммаОбо
Товары опт	Расчеты с пос	Поварешка	Юг	Поставщик Ив	NULL	NULL	NULL	0	1	NULL	0	0
Товары	Расчеты с пос	Поварешка	Юг	Поставщик Ив	NULL	NULL	NULL	100	1	0	0	0
Товары	Расчеты с пос	Вилка	Север	Поставщик Ив	NULL	NULL	USD	390	10	0	0	10
Расчеты с пок	Прибыль	Покупатель МИ	NULL	Ложка	Выручка	NULL	NULL	1 000	0	0	0	0
Расчеты по кр	Расчеты с учре	NULL	NULL	NULL	NULL	NULL	NULL	200 000	0	0	0	0
Расчеты с пок	Прибыль	Покупатель МИ	NULL	Поварешка	Выручка	NULL	NULL	1 000	0	0	0	0
Товары	Расчеты с пос	Ложка	Юг	Поставщик Ив	NULL	NULL	NULL	100	10	0	0	0
Расчеты с пос	Прибыль	Поставщик Ив	NULL			USD	NULL	14,1	0	0	0	0
Прибыль	Товары	Поварешка	Себестоимост	Поварешка	Юг	NULL	NULL	100	0	1	0	0
Прибыль	Товары	Ложка	Себестоимост	Ложка	Юг	NULL	NULL	10	0	1	0	0
Прибыль	Товары опт	Поварешка	Себестоимост	Поварешка	Север	NULL	NULL	0	NULL		0	0
Товары опт	Расчеты с пос	Поварешка	Север	Поставщик Ив	NULL	NULL	NULL	0	1	NULL	0	0
Товары	Расчеты с пос	Поварешка	Север	Поставщик Ив	NULL	NULL	NULL	200	1	0	0	0
Расчеты с пос	Прибыль	Поставщик Ив	NULL			EUR	NULL	60	0	0	0	0
Товары опт	Расчеты с пос	Поварешка	NULL	Поставщик Ив	NULL	NULL	NULL	300	0	0	0	0
Расчеты с учре	Уставный капи	NULL	NULL	NULL	NULL	NULL	NULL	1 000 000	0	0	0	0
Товары	Расчеты с пос	Вилка	Юг	Поставщик Ив	NULL	NULL	NULL	200	10	0	0	0
Прибыль	Товары опт	Поварешка	Себестоимост	Поварешка	NULL	NULL	NULL	150	0	0	0	0
Товары	Расчеты с пос	Вилка	Север	Поставщик Ив	NULL	NULL	EUR	440	10	0	0	10



Но, чтобы получить необходимые значения, данные нужно будет дополнительно сгруппировать.
Данная таблица больше предназначена для получения сводных проводок (например, в какой счет списывались товары), анализа поступления товаров (только где Дт Товары), т.е. для обращения к счету, находящемуся либо в Дт, либо в Кт

Таблица «ДвиженияССубконто» и «Субконто»:

Таблица «Субконто» раскрывает аналитику таблицы движений, если соединить эти таблицы по регистратору и номеру строки, то получим виртуальную таблицу «ДвиженияССубконто».

Таблица «Субконто» (результат):

Период1	Регистратор1	МоментВремени1	НомерСтроки1	ВидДвижения	Вид	Значение
29.01.2013 0:00:00	Поступление товаров 0000(	29.01.2013 0:00:00; Поступл	1	Дебет	Номенклатура	Вилка
29.01.2013 0:00:00	Поступление товаров 0000(	29.01.2013 0:00:00; Поступл	1	Дебет	Склады	Север
29.01.2013 0:00:00	Поступление товаров 0000(	29.01.2013 0:00:00; Поступл	1	Кредит	Контрагенты	Поставщик Иванов
11.01.2013 0:14:37	Поступление товаров 0000(	11.01.2013 0:14:37; Поступл	1	Дебет	Склады	Юг
11.01.2013 0:14:37	Поступление товаров 0000(	11.01.2013 0:14:37; Поступл	1	Дебет	Номенклатура	Поварешка
11.01.2013 0:14:37	Поступление товаров 0000(	11.01.2013 0:14:37; Поступл	1	Кредит	Контрагенты	Поставщик Иванов
11.01.2013 0:17:09	Поступление товаров 0000(	11.01.2013 0:17:09; Поступл	1	Дебет	Склады	Север
11.01.2013 0:17:09	Поступление товаров 0000(	11.01.2013 0:17:09; Поступл	1	Дебет	Номенклатура	Поварешка
11.01.2013 0:17:09	Поступление товаров 0000(	11.01.2013 0:17:09; Поступл	1	Кредит	Контрагенты	Поставщик Иванов
20.01.2013 23:44:47	Поступление товаров 0000(	20.01.2013 23:44:47; Поступ	1	Дебет	Номенклатура	Поварешка
20.01.2013 23:44:47	Поступление товаров 0000(	20.01.2013 23:44:47; Поступ	1	Дебет	Склады	Юг
20.01.2013 23:44:47	Поступление товаров 0000(	20.01.2013 23:44:47; Поступ	1	Кредит	Контрагенты	Поставщик Иванов
20.01.2013 23:45:00	Поступление товаров 0000(	20.01.2013 23:45:00; Поступ	1	Дебет	Склады	Север
20.01.2013 23:45:00	Поступление товаров 0000(	20.01.2013 23:45:00; Поступ	1	Дебет	Номенклатура	Поварешка
20.01.2013 23:45:00	Поступление товаров 0000(	20.01.2013 23:45:00; Поступ	1	Кредит	Контрагенты	Поставщик Иванов
29.01.2013 0:43:54	Поступление товаров 0000(	29.01.2013 0:43:54; Поступл	1	Дебет	Склады	Север
29.01.2013 0:43:54	Поступление товаров 0000(	29.01.2013 0:43:54; Поступл	1	Дебет	Номенклатура	Вилка
29.01.2013 0:43:54	Поступление товаров 0000(	29.01.2013 0:43:54; Поступл	1	Кредит	Контрагенты	Поставщик Иванов
29.11.2012 1:16:45	Поступление товаров 0000(	29.11.2012 1:16:45; Поступл	1	Дебет	Склады	Юг
29.11.2012 1:16:45	Поступление товаров 0000(	29.11.2012 1:16:45; Поступл	1	Дебет	Номенклатура	Ложка

Такая форма записи позволяет не изменять физически таблицы БД при увеличении максимального количества Субконто по счетам (субконто развернуты по вертикали).

Занятие 3.25

РАСЧЕТ

Задача: разработаем конфигурацию, которая позволит пользователю вводить Управленческий оклад («по-черному», сколько решат, столько и будет). По большому счету такой оклад ни от чего не зависит, поэтому будет просто вводиться пользователем в базу. В конце месяца начальники отделов вводят информацию о премировании сотрудников, у каждого сотрудника свой процент премии:

Премия = Управленческий оклад * %

Создадим объект, который будет содержать информацию о видах начислений. Он будет использоваться, в частности, для создания отчета вида:

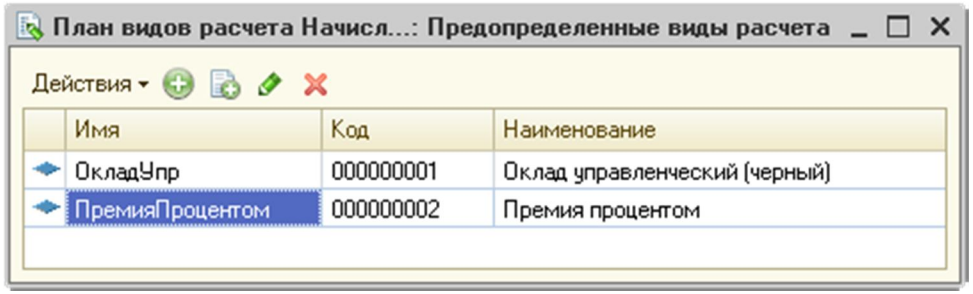
Начисление	Сумма
Оклад	70000
Премия	30000

План видов расчета – специальный объект системы для хранения видов расчета. Аналог справочника, отличается тем, что хранит виды зависимостей одного вида расчета от другого.

Свойство	Значение
Имя	Начисления
Синоним	Начисления
Представление объекта	Начисление
Представление списка	Список начислений

Закладка «Расчет» - главное отличие «Плана видов расчета» от «Справочника».

Создадим предопределенные виды расчета:



При расчете начисления надо знать какая сумма оклада была начислена сотруднику – оптимальный объект для хранения информации об окладах – Регистр накопления (вид регистра - Обороты). Пояснение: надо начислить премию за месяц, взял оборот регистра за месяц, аналогично за год и другие периоды.
Но в системе 1С есть специальный объект для хранения расчетных данных – «Регистры расчета». Предназначен для накопления 2-х показателей учета:

- Показатель базы – например, информация об окладе за какой-то прошлый период, т.е. сумма начисленного оклада в прошлом периоде. База будет накапливаться в разрезе сотрудников. База аналогична Оборотам (постоянно растет), но в Регистре расчета связь шире – можно устанавливать зависимость (например, если база изменилась, то все связанные с ней расчеты надо пересчитать).

Создадим Регистр расчета. В примере в Регистре расчета будет храниться информация вида:

Сотрудник	Начислено
Иванов	5000
Петров	3000

Свойство	Значение
Имя	НачисленияСотрудников
Синоним	Начисления сотрудников
План видов расчета	Начисления

Регистр расчета связан с Планом видов расчета, на основе этого регистр расчета будет приобретать определенные свойства. В примере, за счет связи, регистр наследует информацию о типе значения предопределенного измерения регистра расчета, т.е. Вид расчета в регистре расчета уже будет существовать.

Период	ВидРасчета	Сотрудник	Сумма
29.04.2013	Оклад	Иванов	100000

Создадим справочник «Сотрудники»:

Свойство	Значение
Имя	Сотрудники
Синоним	Сотрудники
Представление объекта	Сотрудник
Представление списка	Сотрудники

Вернемся к Регистру расчета «НачисленияСотрудников» и перейдем на закладку «Данные»:
База должна сохраняться в Ресурсах. В примере: если в дальнейшем сумма Оклада сотрудника должна стать Базой для расчета премии, то сумма должна находиться в Ресурсах.
Другие чистовые данные должны храниться как Реквизиты.

Свойство	Значение
Измерения	
Сотрудник	
Имя	Сотрудник
Синоним	Сотрудник
Тип	СправочникСсылка.Сотрудники
Запрет незаполненных значений	Истина
Ресурсы	
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2

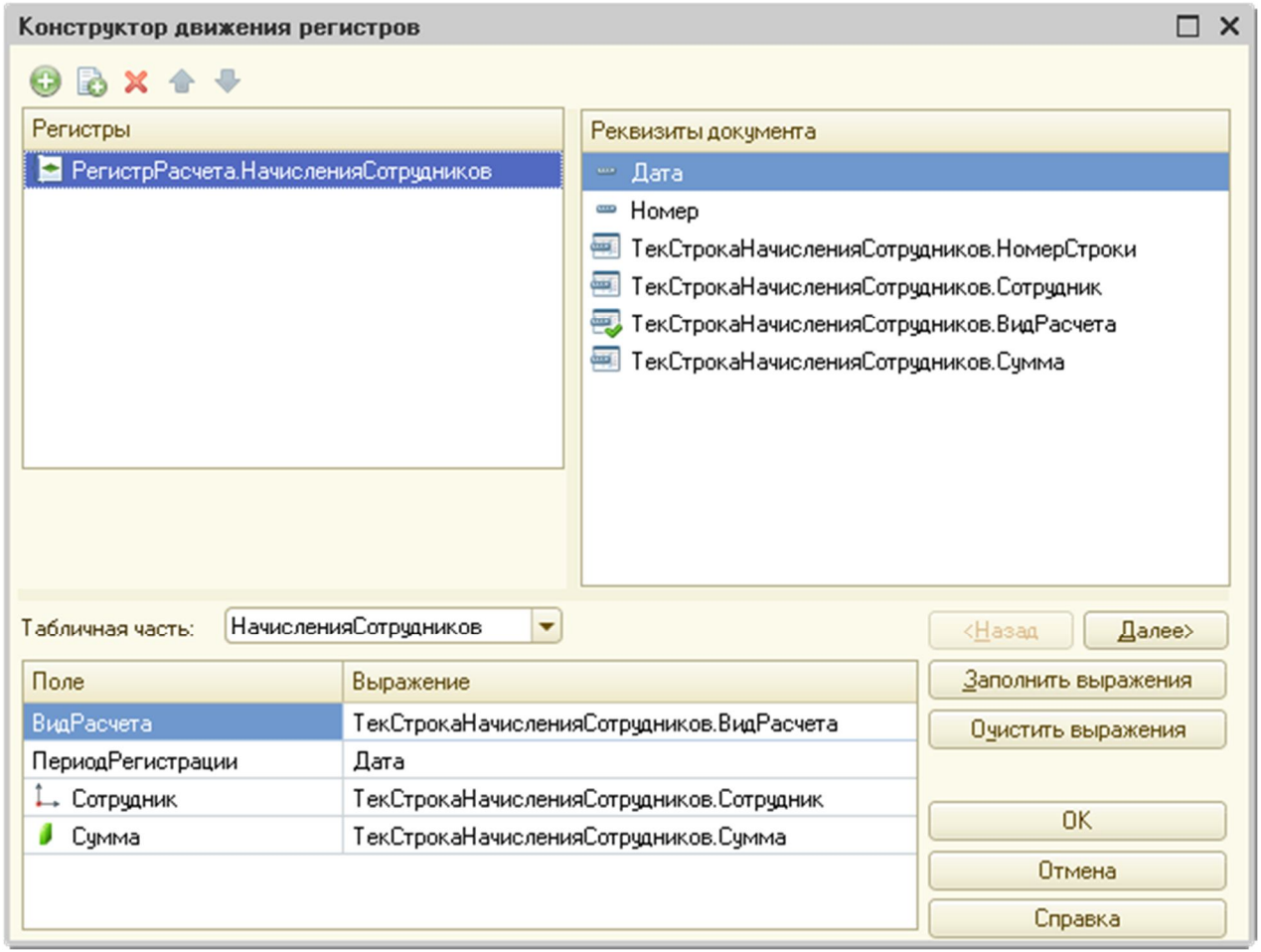
Для формирования движений по регистру расчета создадим документ «Ввод произвольных начислений»

Свойство	Значение
Имя	ВводПроизвольныхНачислений
Синоним	Ввод произвольных начислений
Закладка «Данные»	
Табличные части	
Начисления сотрудников	
Имя	НачисленияСотрудников
Синоним	Начисления сотрудников
Проверка заполнения	Выдавать ошибку
Реквизиты табличной части	
Сотрудник	
Имя	Сотрудник
Синоним	Сотрудник
Тип	СправочникСсылка.Сотрудники
Проверка заполнения	Выдавать ошибку
Вид расчета	
Имя	ВидРасчета
Синоним	Вид расчета
Тип	ПланВидовРасчетаСсылка.Начисления
Проверка заполнения	Выдавать ошибку
Сумма	
Имя	Сумма
Синоним	Сумма
Тип	Число 15, 2
Закладка «Движения»	
Оперативное проведение	Запретить

Движения будут проводиться по Регистру расчета «НачисленияСотрудников». Откроем конструктор движений:

Выберем табличную часть «НачисленияСотрудников» и нажмем «Заполнить выражения». Система заполнит все поля автоматом, кроме «ПериодРегистрации»:

ПериодРегистрации – аналог поля «Период» регистров – дата регистрации записи в регистре.
Период регистрации приводится к первому числу интервала, установленного в свойстве «Периодичность» Регистра расчета.



Результат:

```
Процедура ОбработкаПроведения(Отказ, Режим)

     Движения.НачисленияСотрудников.Записывать = Истина;
     Для Каждого ТекСтрокаНачисленияСотрудников Из НачисленияСотрудников Цикл
          Движение = Движения.НачисленияСотрудников.Добавить();
          Движение.ВидРасчета = ТекСтрокаНачисленияСотрудников.ВидРасчета;
          Движение.ПериодРегистрации = Дата;
          Движение.Сотрудник = ТекСтрокаНачисленияСотрудников.Сотрудник;
          Движение.Сумма = ТекСтрокаНачисленияСотрудников.Сумма;
     КонецЦикла;

КонецПроцедуры
```

Будем очищать набор записей при перепроведении:

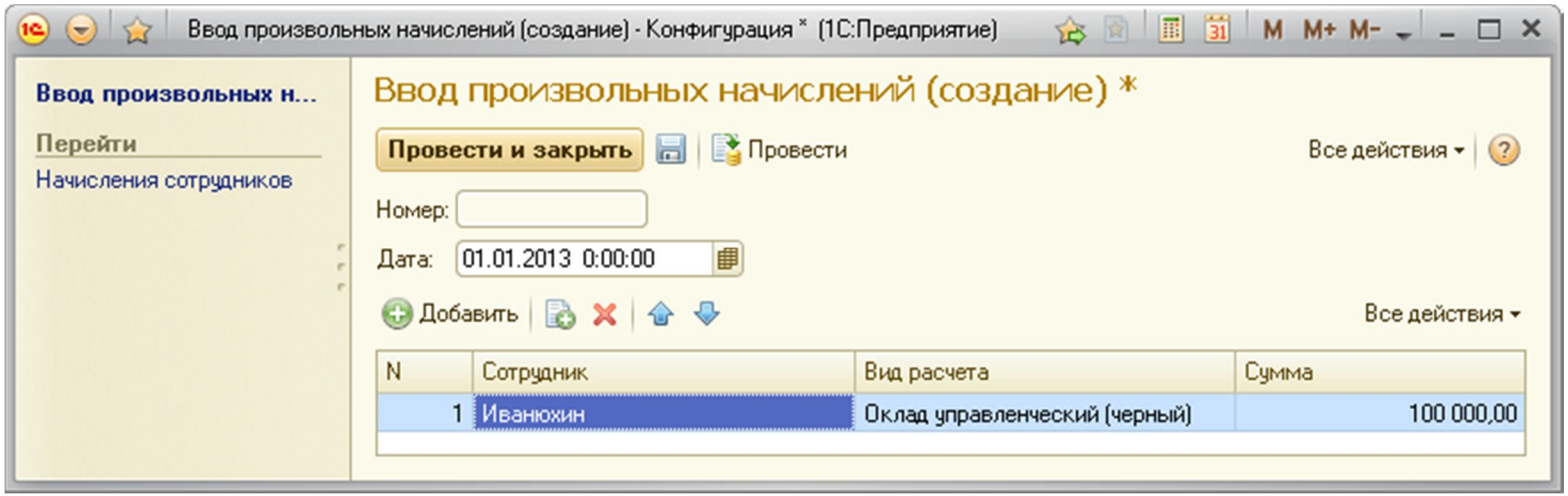
```
Процедура ОбработкаПроведения(Отказ, Режим)

     Движения.НачисленияСотрудников.Записывать = Истина;
     Движения.НачисленияСотрудников.Очистить();
     Для Каждого ТекСтрокаНачисленияСотрудников Из НачисленияСотрудников Цикл
          Движение = Движения.НачисленияСотрудников.Добавить();
          Движение.ВидРасчета = ТекСтрокаНачисленияСотрудников.ВидРасчета;
          Движение.ПериодРегистрации = Дата;
          Движение.Сотрудник = ТекСтрокаНачисленияСотрудников.Сотрудник;
          Движение.Сумма = ТекСтрокаНачисленияСотрудников.Сумма;
     КонецЦикла;

КонецПроцедуры
```

В режиме исполнения:

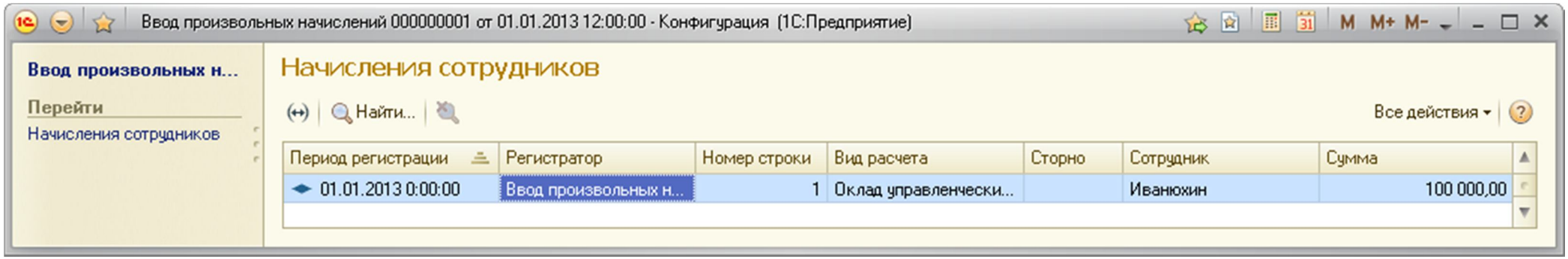
Создадим сотрудников:

- Иванюхин
- Каменский

Введем произвольные начисления:



Проведем документ и посмотрим движения:



Аналогично назначим оклад, равный 90000, Каменскому.

Начисление премии:
Для автоматического расчета премии, надо хранить информацию о проценте (%) премии. Т.к. вводить данную информацию будет расчетчик, то будем хранить ее в документе «ВводПроизвольныхНачислений». Создадим реквизит в табличной части документа:

Свойство	Значение
Имя	Параметр
Синоним	Параметр
Тип	Число 15, 2

Вид хранимых параметром данных может меняться в зависимости от вида расчета. Пример:

- ставка в час для оклада.
- процент больничного, оплачиваемый работодателем.

Значение Параметра обязательно должно попасть в Регистр расчета, т.к. расчет результата записи должен производиться не в документе (будем часто программно перерасчитывать записи).

ПРАВИЛО: данные для расчета должны храниться в Регистре расчета, чтобы не обращаться к документам, сформировавшим движения с данными, при расчетах, перерасчетах и т.д.

В Регистре расчета «Начисления сотрудников» добавим Реквизит «Параметр», аналогичный реквизиту «Параметр» документа «ВводПроизвольныхНачислений».

Сформируем движения по Параметру: документ «ВводПроизвольныхНачислений», модуль объекта:

```
Процедура ОбработкаПроведения(Отказ, Режим)
     Движения.НачисленияСотрудников.Записывать = Истина;
     Движения.НачисленияСотрудников.Очистить();
     Для Каждого Стр Из НачисленияСотрудников Цикл
          Движение = Движения.НачисленияСотрудников.Добавить();
          Движение.ВидРасчета        = Стр.ВидРасчета;
          Движение.ПериодРегистрации = Дата;
          Движение.Сотрудник         = Стр.Сотрудник;
```

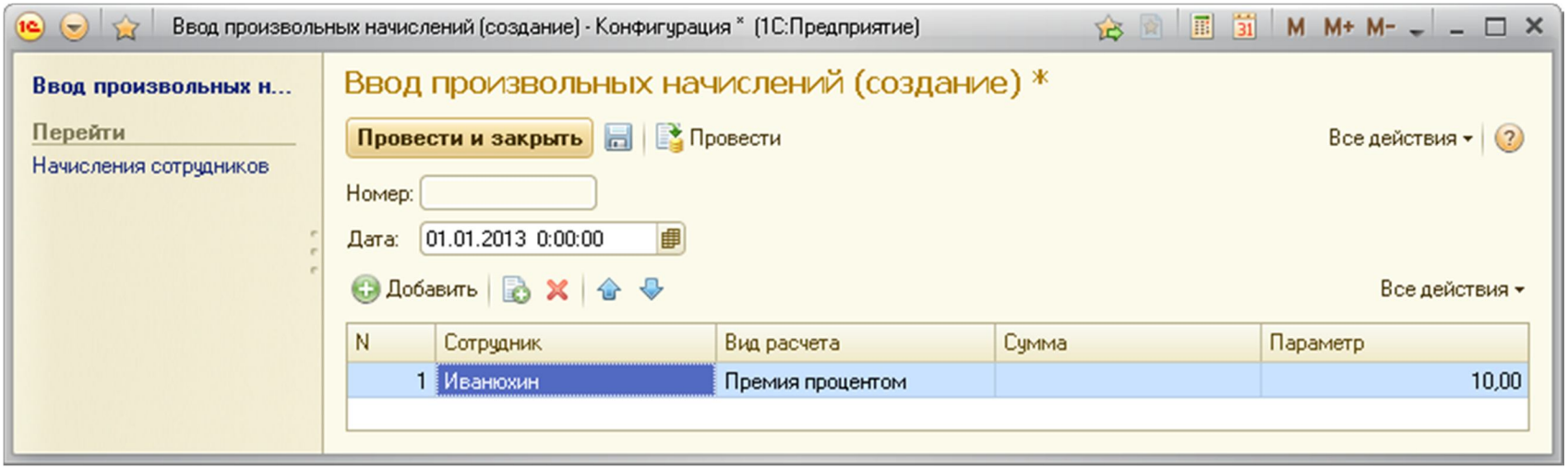
```
        Движение.Сумма              = Стр.Сумма;
        Движение.Параметр           = Стр.Параметр;
    КонецЦикла;
КонецПроцедуры
```

Другой вариант:
Реквизиты Регистра расчета и табличной части документа называются одинаково, поэтому таблицу Регистра можно автоматически заполнить значениями таблицы документа:

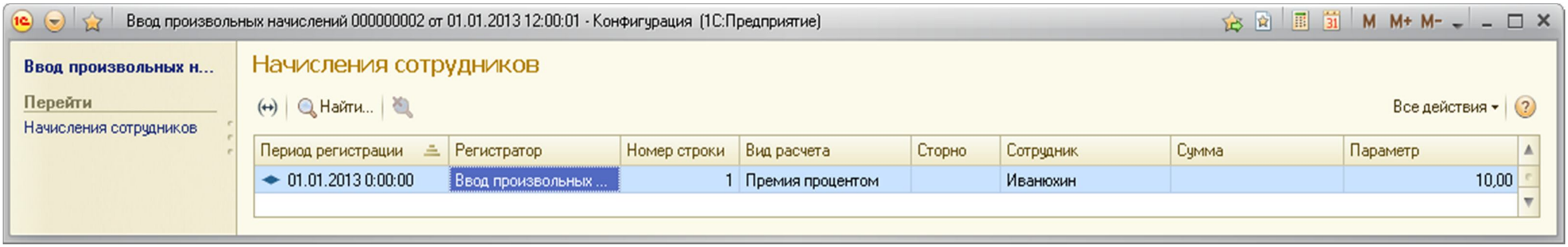
```
Процедура ОбработкаПроведения(Отказ, Режим)
    Движения.НачисленияСотрудников.Записывать = Истина;
    Движения.НачисленияСотрудников.Очистить();
    Для Каждого Стр Из НачисленияСотрудников Цикл
        Движение = Движения.НачисленияСотрудников.Добавить();
        //Движение.ВидРасчета        = Стр.ВидРасчета;
        Движение.ПериодРегистрации = Дата;
        //Движение.Сотрудник         = Стр.Сотрудник;
        //Движение.Сумма             = Стр.Сумма;
        //Движение.Параметр          = Стр.Параметр;
        ЗаполнитьЗначенияСвойств(Движение, Стр);
    КонецЦикла;
КонецПроцедуры
```

Это удобно, т.к. при увеличении количества реквизитов документа и данных регистра, не надо будет постоянно дописывать код в движениях.

В режиме исполнения:
Создадим документ, начисляющий премию Иванюхину:



Проведем документ и посмотрим движения:



Рассчитаем сумму премии:

Надо определить, за какой период рассчитываем премию. Если алгоритм расчета премии одинаков для любого периода, то правильно будет сделать так, чтобы пользователь мог указывать период расчета. Для этого в регистре надо определить «Период для получения базы». Т.е. чтобы автоматом получать «Базу» в регистр расчета надо добавить еще 2 поля:

- ДатаНачалаПолученияБАЗЫ.
- ДатаОкончанияПолученияБАЗЫ.

Например, есть запись о начислении Иванову оклада:

Период	ВидРасчета	Сотрудник	Сумма	Параметр	ДатаНачалаПолученияБАЗЫ	ДатаОкончанияПолученияБАЗЫ
29.04.2013	Оклад	Иванов	1000000			

Начислим премию:

Период	ВидРасчета	Сотрудник	Сумма	Параметр	ДатаНачалаПолученияБАЗЫ	ДатаОкончанияПолученияБАЗЫ
29.04.2013	Оклад	Иванов	1000000			
29.04.2013	Премия	Иванов		10	01.01.2013	31.01.2013

В данном случае Иванов не получит премию, т.к. Оклад рассчитан за АПРЕЛЬ, а премия рассчитана за МАЙ.

Если же дата начисления оклада будет входить в период получения БАЗЫ, то премия будет рассчитана

Поля «ДатаНачалаПолученияБАЗЫ» и «ДатаОкончанияПолученияБАЗЫ» включаются в регистр расчета свойством «Базовый период».
В примере:

Свойство	Значение
Базовый период	Истина
Периодичность	Месяц

Чтобы вводить данные появившихся полей через документ, добавим их в документ:

База начало:

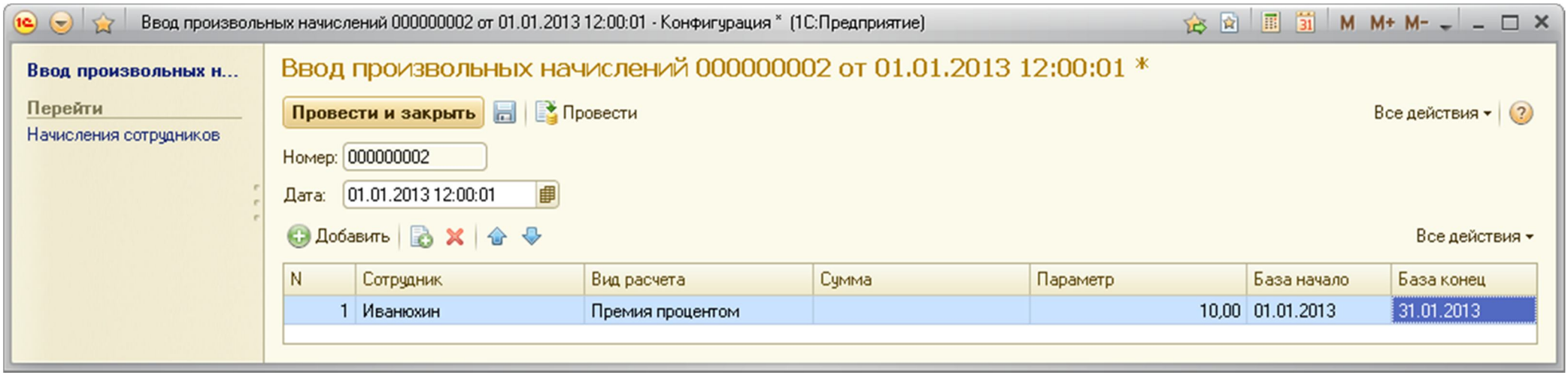
Свойство	Значение
Имя	БазовыйПериодНачало
Синоним	База начало
Тип	Дата

База конец:

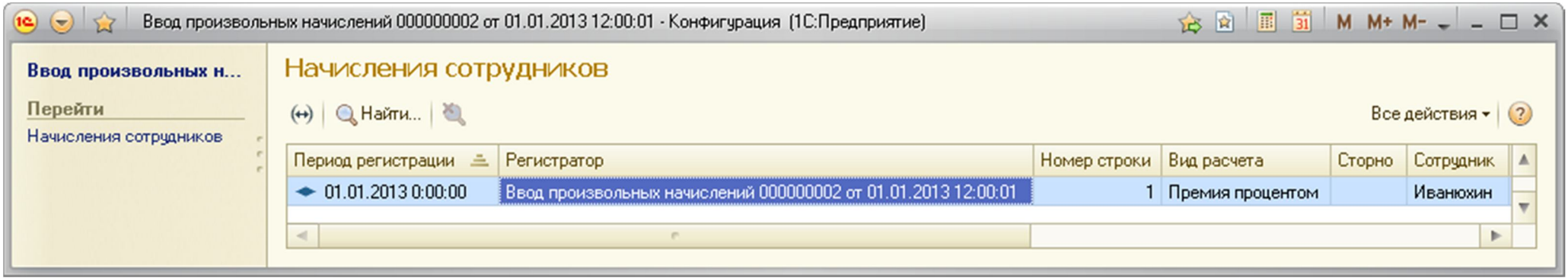
Свойство	Значение
Имя	БазовыйПериодКонец
Синоним	База конец
Тип	Дата

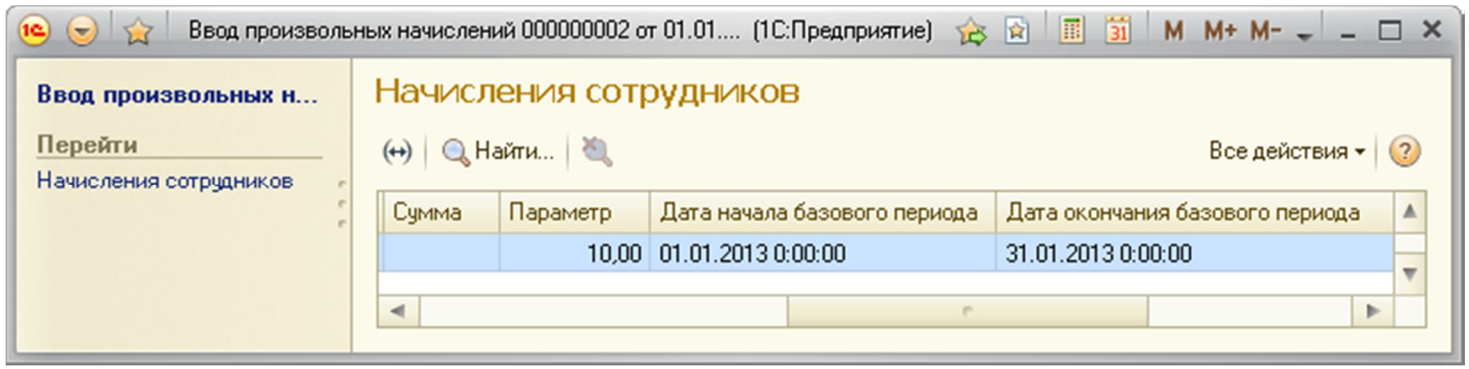
В режиме исполнения:

В документе расчета премии заполним добавленные поля:

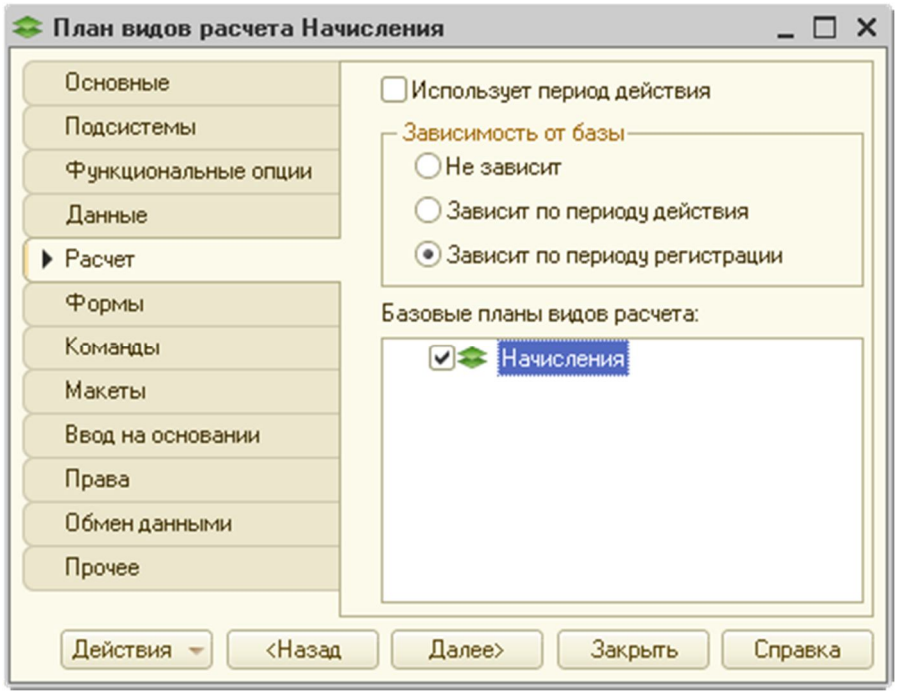


Проведем документ и посмотрим движения:



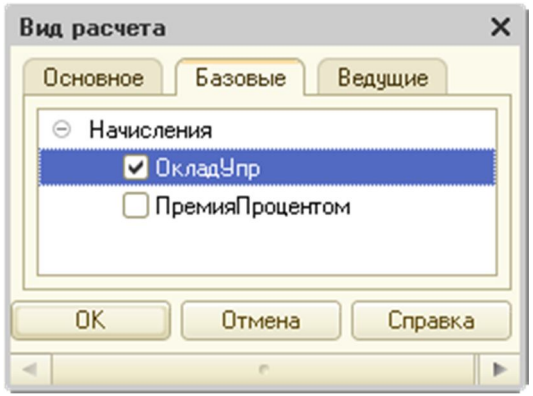


% премии указали, теперь надо наладить связь с показателями, от которых будет зависеть премия:
Связь указывается в свойствах Плана видов расчета, т.е. в Плане видов расчета «Начисления» можно включить понятие «Накопление БАЗЫ» (понятие связи – ПРЕМИЯ считается ПРОЦЕНТОМ от ВИДА РАСЧЕТА. БАЗУ для ПРЕМИИ могут образовывать различные виды расчета).



Базовые планы видов расчета – указывается, откуда можно выбирать базовые виды.

Данная настройка позволит указать, процентом от чего считается премия. В свойствах Вида расчета появилась закладка «Базовые» - в системе это таблица «Базовые виды расчета»:



В результате:

Метод расчета ПРЕМИИ системой:

Период	ВидРасчета	Сотрудник	Сумма	Параметр	ДатаНачалаПолученияБАЗЫ	ДатаНачалаПолученияБАЗЫ
01.01.2013	Оклад	Иванов	1000000			
01.01.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Петров	1000000			
25.05.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Иванов	1000000			
25.05.2013	Премия	Сидоров		10	01.05.2013	31.05.2013

Системе надо взять премию за ПЕРИОД – май, т.е. первые 2 записи она игнорирует:

Период	ВидРасчета	Сотрудник	Сумма	Параметр	ДатаНачалаПолученияБАЗЫ	ДатаНачалаПолученияБАЗЫ
01.01.2013	Оклад	Иванов	1000000			
01.01.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Петров	1000000			
25.05.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Иванов	1000000			
25.05.2013	Премия	Сидоров		10	01.05.2013	31.05.2013

Системе задано, какие виды расчета образуют БАЗУ для ПРЕМИИ – в примере это ОКЛАД, т.е. последняя строка тоже исключается:

Период	ВидРасчета	Сотрудник	Сумма	Параметр	ДатаНачалаПолученияБАЗЫ	ДатаНачалаПолученияБАЗЫ
01.01.2013	Оклад	Иванов	1000000			
01.01.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Петров	1000000			
25.05.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Иванов	1000000			
25.05.2013	Премия	Сидоров		10	01.05.2013	31.05.2013

Чтобы окончательно сформировать БАЗУ, надо наложить дополнительный отбор по Сотруднику, после чего исключается 3 строка:

Период	ВидРасчета	Сотрудник	Сумма	Параметр	ДатаНачалаПолученияБАЗЫ	ДатаНачалаПолученияБАЗЫ
01.01.2013	Оклад	Иванов	1000000			
01.01.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Петров	1000000			
25.05.2013	Премия	Иванов		10	01.05.2013	31.05.2013
25.05.2013	Оклад	Иванов	1000000			
25.05.2013	Премия	Сидоров		10	01.05.2013	31.05.2013

Производить расчет в документе нежелательно, т.к. чтобы перерасчитать прошлые записи, надо будет перепроводить документ, при этом блокируется таблица регистра расчета и таблица документа.

ПРАВИЛО: Движения Регистра расчета рассчитываются вне документа.

Следовательно, расчет будем выполнять в ОБЩЕМ МОДУЛЕ. Создадим модуль «Расчеты»:

Свойство	Значение
Имя	Расчеты
Синоним	Расчеты
Привилегированный	Истина

«Вызов сервера = ЛОЖЬ» - модуль будет исполняться только на сервере, т.е. вызывать его можно будет только с серверной стороны.
«Привилегированный = ИСТИНА» - действия в модуле будут наделены полномочными правами с БД, не будут проверяться права и, как следствие, расчет будет производиться быстрее.

В свойствах документа есть аналогичный реквизит – «Прив. режим при проведении».

В примере включили режим Привилегированности в общем модуле, т.к. его процедуры будут вызываться не только из документа, но и из внешней обработки, которая будет создана позже.

В процедуру передаем ссылку на документ «ВводПроизвольныхНачислений», из которого и будет вызываться данная процедура. По ссылке можно обратиться к регистру и получить из него движения.

```
Процедура Рассчитать(Регистратор) Экспорт

КонецПроцедуры
```

Запишем движения документа «ВводПроизвольныхНачислений»

```
Процедура ОбработкаПроведения(Отказ, Режим)

     Движения.НачисленияСотрудников.Записывать = Истина;
     Движения.НачисленияСотрудников.Очистить();
     Для Каждого Стр Из НачисленияСотрудников Цикл
          Движение = Движения.НачисленияСотрудников.Добавить();
          Движение.ПериодРегистрации = Дата;
          ЗаполнитьЗначенияСвойств(Движение, Стр);
     КонецЦикла;

     Движения.Записать();

КонецПроцедуры
```

Метод «Записать()» относится ко всей коллекции движений документа, инициирует запись всех движений с установленным признаком «Записывать»:

```
Движения.НачисленияСотрудников.Записывать = Истина;
```

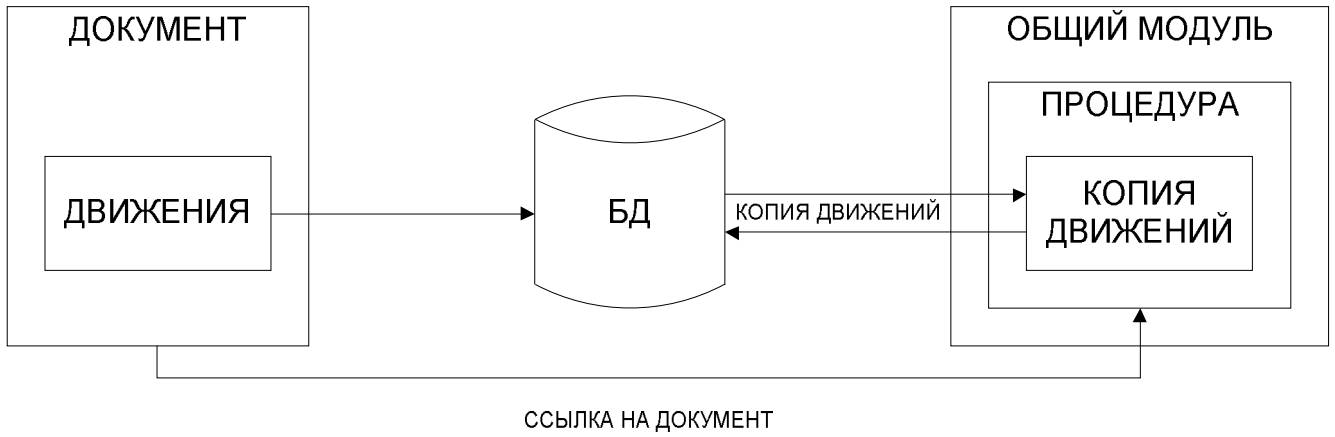
Если применить метод «Записать()» 2 раза подряд, то второй раз движения записаны не будут, т.к. метод дополнительно сбразывает флаг «Записывать» в ЛОЖЬ:

```
Движения.НачисленияСотрудников.Записывать = Ложь;
```

Вызовем процедуру общего модуля:

```
Процедура ОбработкаПроведения(Отказ, Режим)

    Движения.НачисленияСотрудников.Записывать = Истина;
    Движения.НачисленияСотрудников.Очистить();
    Для Каждого Стр Из НачисленияСотрудников Цикл
        Движение = Движения.НачисленияСотрудников.Добавить();
        Движение.ПериодРегистрации = Дата;
        ЗаполнитьЗначенияСвойств(Движение, Стр);
    КонецЦикла;

    Движения.Записать();

    Расчеты.Рассчитать(Ссылка);

КонецПроцедуры
```

В общем модуле получим копию набора записей документа:



Есть ДОКУМЕНТ, у которого сформированы ДВИЖЕНИЯ и записаны в БД. Из ДОКУМЕНТА вызываем ПРОЦЕДУРУ ОБЩЕГО МОДУЛЯ и передаем туда ССЫЛКУ на ДОКУМЕНТ. В ПРОЦЕДУРЕ получаем КОПИЮ ДВИЖЕНИЙ (по ССЫЛКЕ). КОПИЮ ДВИЖЕНИЙ рассчитываем и записываем обратно в БД (в регистр).

```
Процедура Рассчитать(Регистратор) Экспорт
     // получим копию набора записей регистра
     НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
     НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
     НаборЗаписей.Прочитать();

     ОтборПоИзмерениям = Новый Структура;
     ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

     СписокРесурсов = Новый Массив(1);
     СписокРесурсов[0] = "НачисленияСотрудноков.Сумма";

     Для Каждого Запись Из НаборЗаписей Цикл
          Если Запись.ВидРасчета = ПланыВидовРасчета.Начисления.ПремияПроцентом Тогда
               ТЗ_База = Запись.ПолучитьБазу(СписокРесурсов, ОтборПоИзмерениям);
               База = ТЗ_База[0].Сумма;

               Запись.Сумма = База * Запись.Параметр / 100;
          КонецЕсли;
     КонецЦикла;
     НаборЗаписей.Записать();
КонецПроцедуры
```

Создаем набор записей. Устанавливаем в нем отбор по документу и читаем записи. После этого в переменной «НаборЗаписей» сохраняются движения из регистра расчета «НачисленияСотрудников», которые до этого были записаны с помощью документа «ВводПроизвольныхНачислений».
Далее перебираем полученные записи в цикле.
Нам надо рассчитывать только премию, поэтому проверяем, какая запись считывается:

```
Если Запись.ВидРасчета = ПланыВидовРасчета.Начисления.ПремияПроцентом Тогда
```

Метод «ПолучитьБазу(<Ресурсы>, <Измерения>, <Разрезы>)» - собирает данные из БД по определенным параметрам

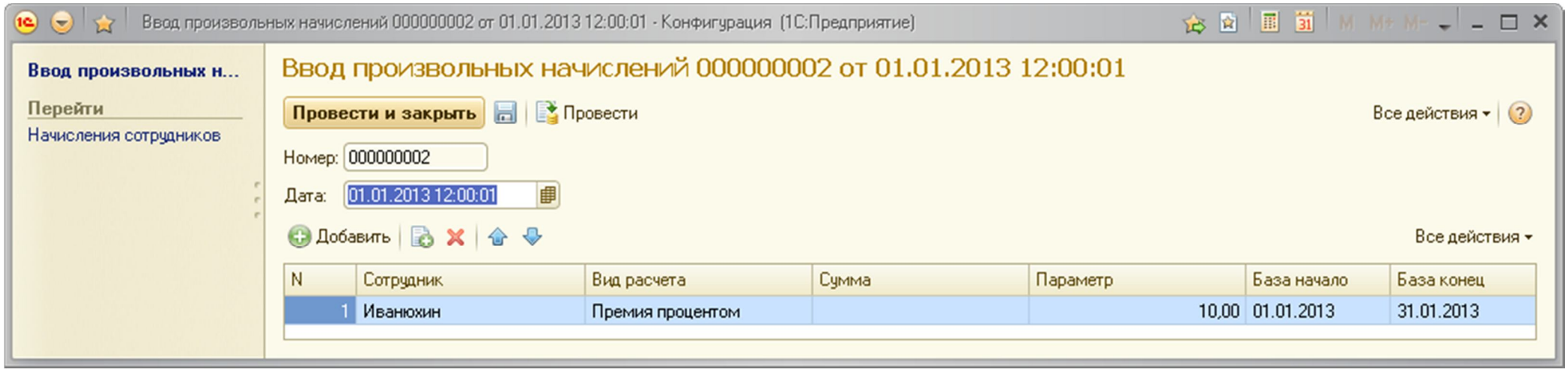
<Ресурсы> - какие ресурсы надо суммировать. Задается массивом. Ресурсы должны быть заданы в формате:

<Имя регистра расчета>.<Имя ресурса>

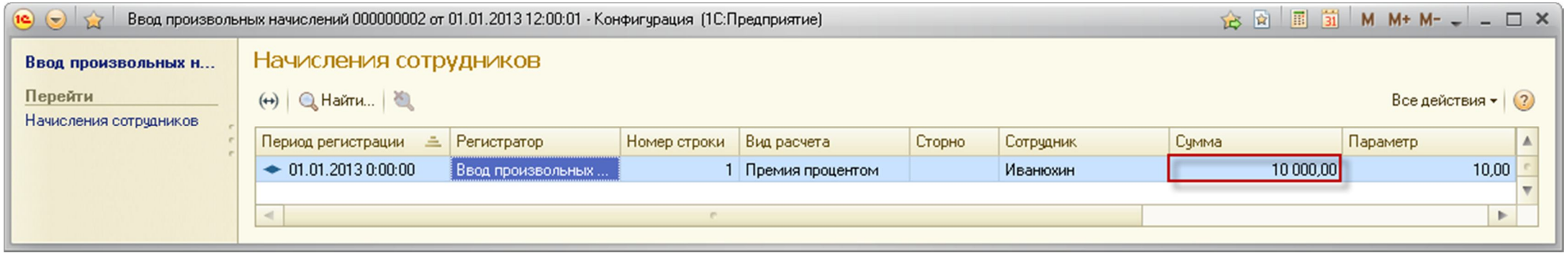
<Измерения> - значения отбора при получении базы. Задается структурой.

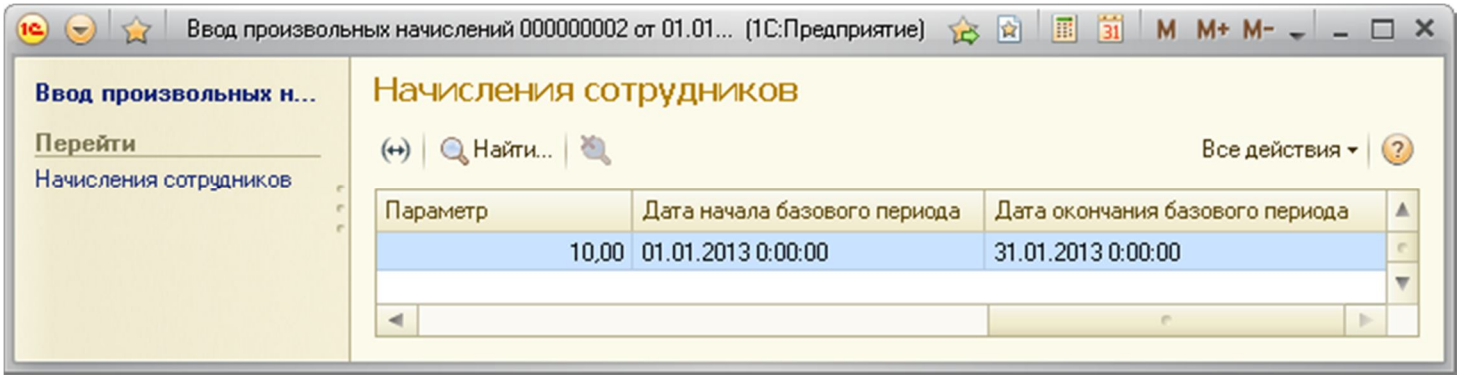
В режиме исполнения:

Перепроведем документ «Ввод произвольных начислений»:

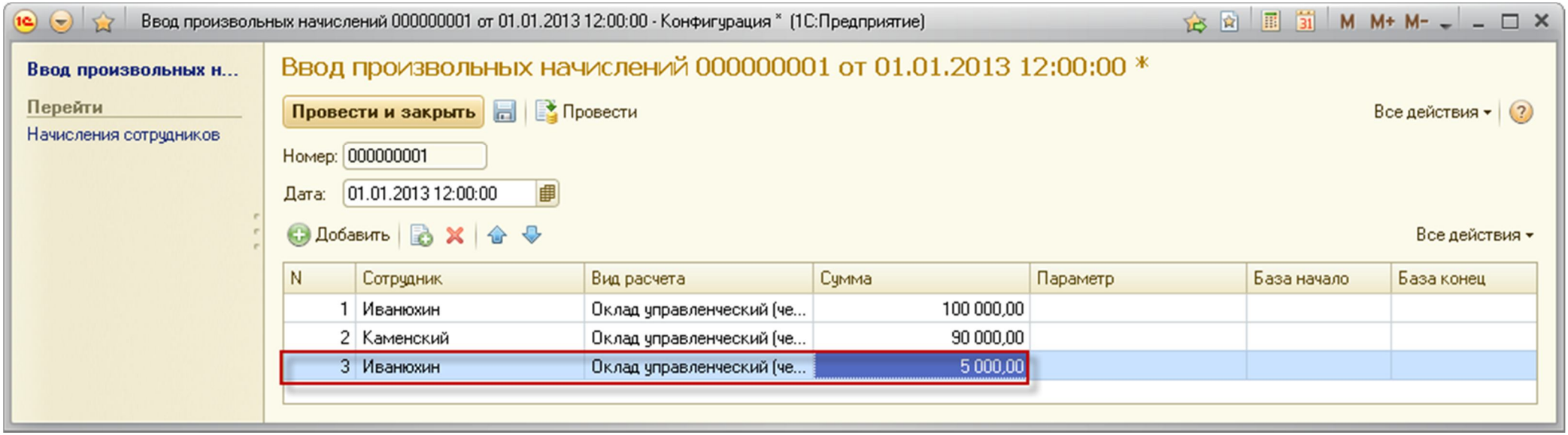


Проверим движения:

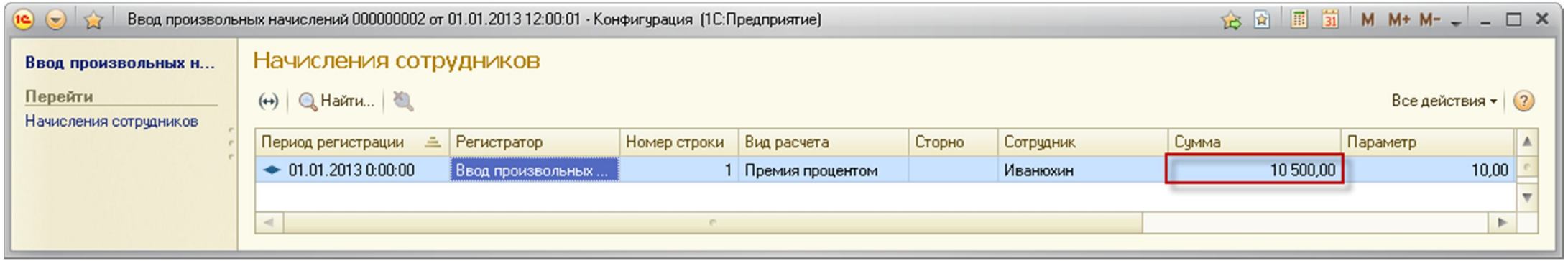


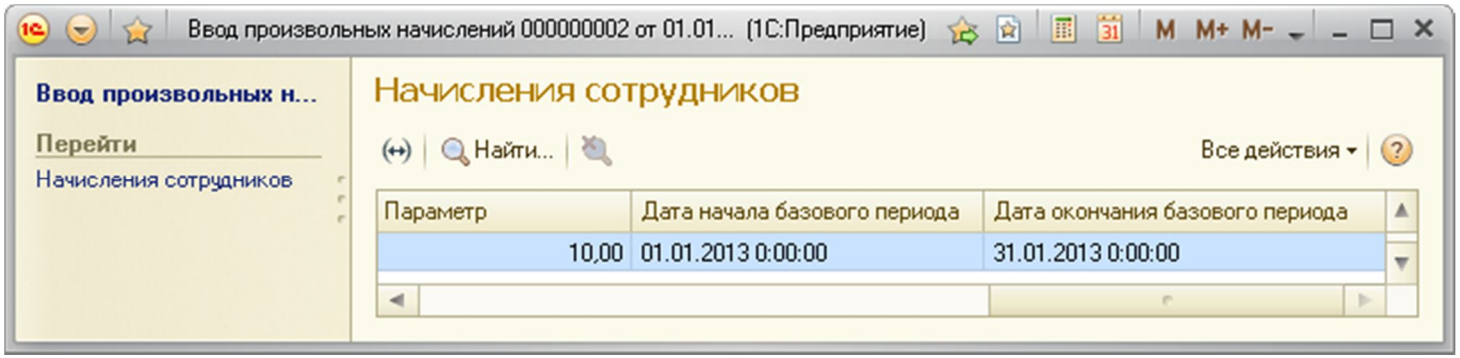


Усложним задачу. Введем Иванюхину еще один оклад:

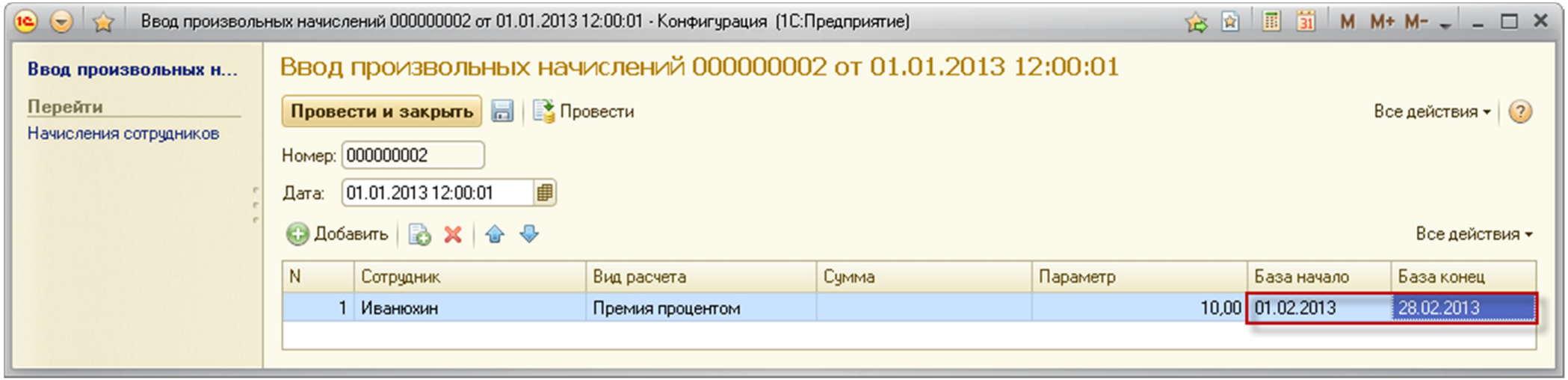


Пересчитаем премию:

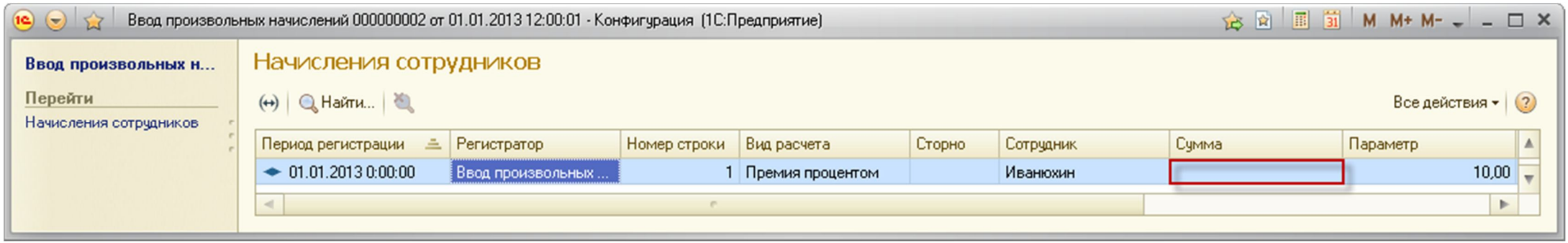


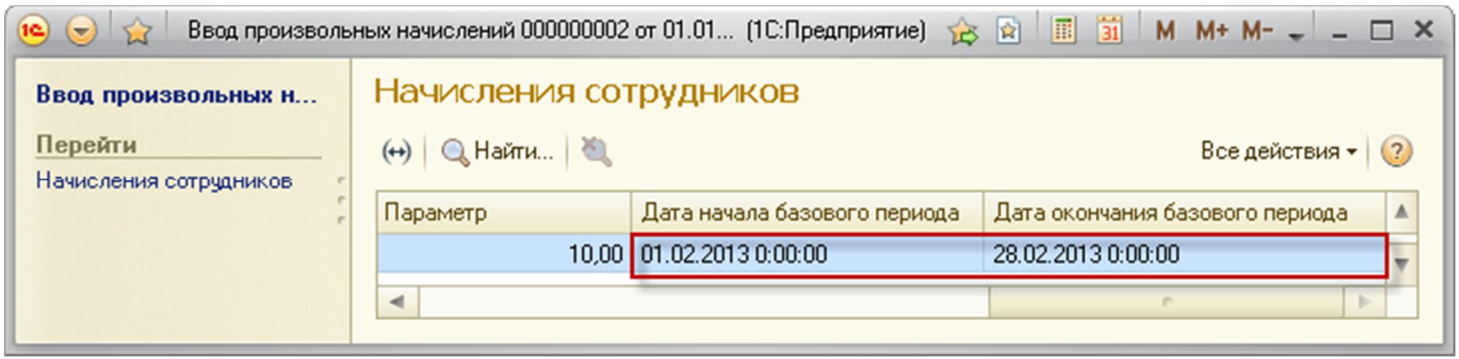


Поменяем базовый период расчета премии:



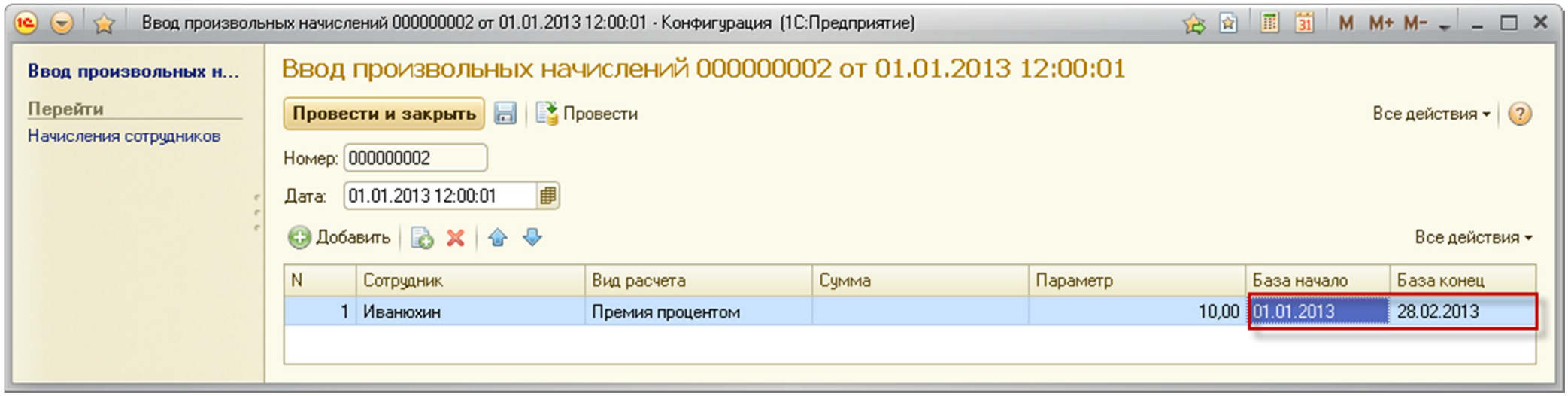
Проверим движения:



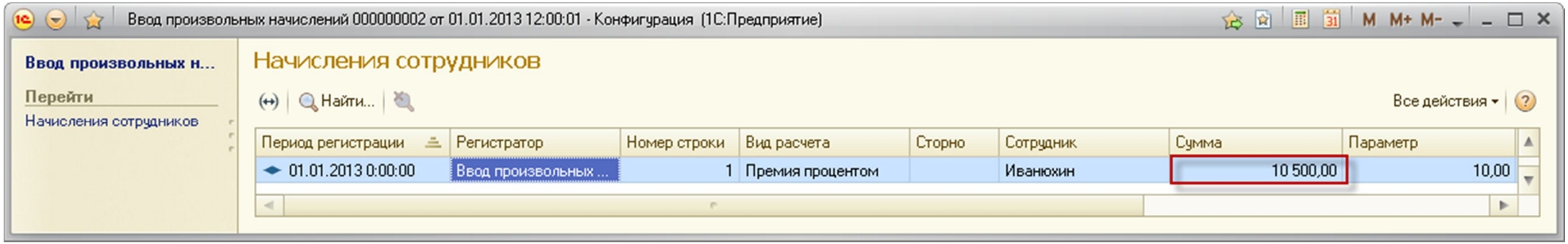


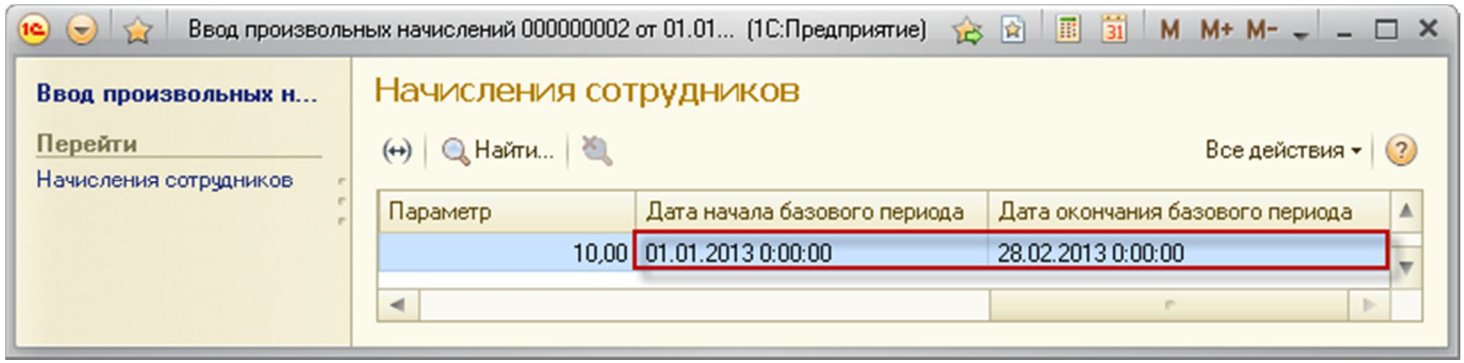
Премия не рассчиталась, т.к. за этот период нет базы для ее расчета.

Еще раз поменяем базовый период расчета премии (за 2 месяца):



Проверим движения:





Если бы был оклад за февраль, то он бы тоже посчитался.

- Показатель данных графика

Занятие 3.26

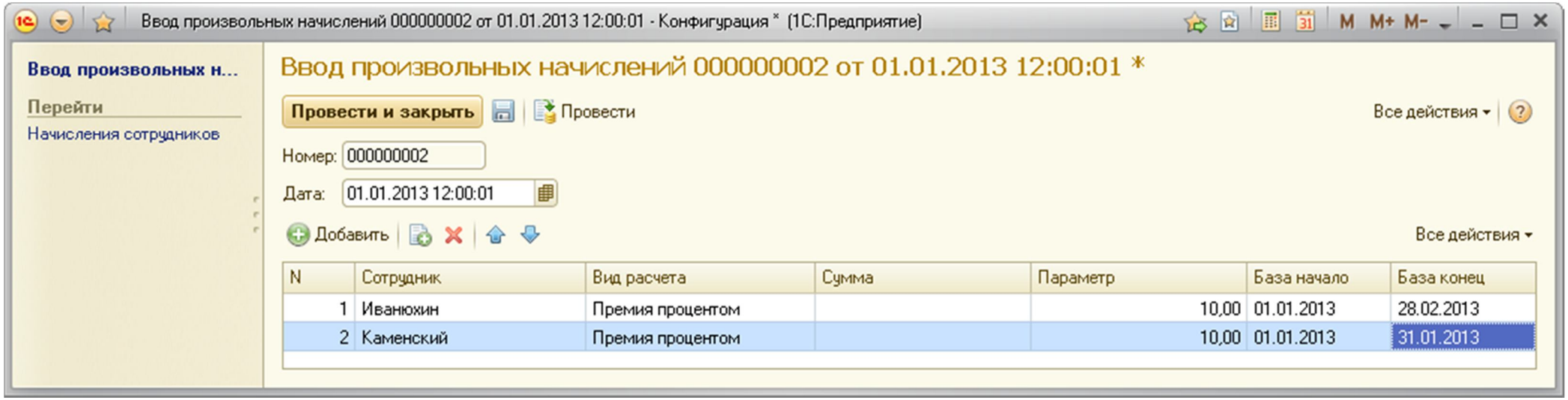
Оптимизируем процедуру «Рассчитать»:

Метод «ПолучитьБазу()» строит запрос к регистру расчета. Сейчас запрос строится в цикле, что не совсем хорошо. Получим БАЗУ для всех записей сразу. Обратимся не к записи регистра, а к самому регистру и получим все записи по определенным отборам (в примере – по текущему документу):

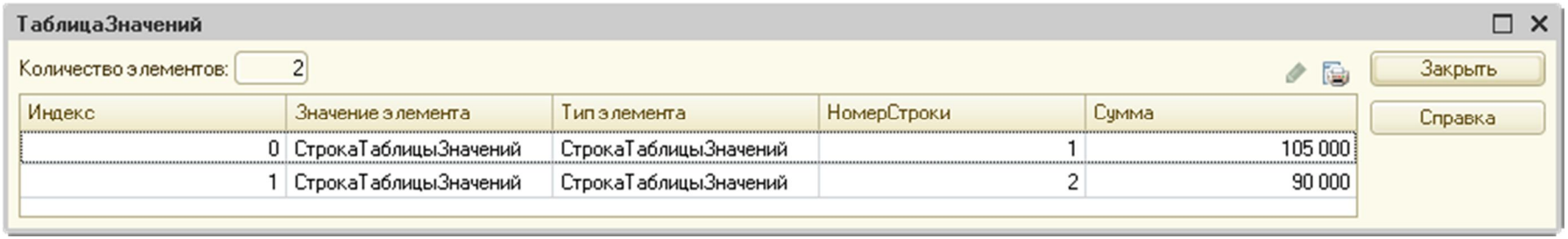
```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
                        ОтборПоРегистратору,
                        СписокРесурсов,
                        ОтборПоИзмерениям);

    Для Каждого Запись Из НаборЗаписей Цикл
        Если Запись.ВидРасчета = ПланыВидовРасчета.Начисления.ПремияПроцентом Тогда
            ТЗ_База = Запись.ПолучитьБазу(СписокРесурсов, ОтборПоИзмерениям);
            База = ТЗ_База[0].Сумма;

            Запись.Сумма = База * Запись.Параметр / 100;
```

```
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

Поставим точку останова и посмотрим, что из себя представляет значение «ТЗ_База». В режиме исполнения:
В документ расчета премии добавим еще одного сотрудника:



Проведем документ и посмотрим, что находится в переменной «ТЗ_База»:



Количество строк документа (набора записей регистра расчета) и количество строк в таблице БАЗА всегда будет совпадать.
Нумерация строк тоже будет совпадать – для первой строки набора записей регистра, БАЗА будет содержаться в первой строке таблицы «БАЗА».

Количество строк и их порядок совпадают, следовательно, чтобы прочитать определенную строку БАЗЫ, к ней можно обратиться по индексу.

Итого код:

```
Процедура Рассчитать(Регистратор) Экспорт
	// получим копию набора записей регистра
	НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
	НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
	НаборЗаписей.Прочитать();

	ОтборПоИзмерениям = Новый Структура;
	ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

	СписокРесурсов = Новый Массив(1);
	СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

	ОтборПоРегистратору = Новый Структура;
	ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

	ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
	ОтборПоРегистратору,
	СписокРесурсов,
	ОтборПоИзмерениям);

	Для Каждого Запись Из НаборЗаписей Цикл
		//Индекс = Запись.НомерСтроки - 1;
		Индекс = НаборЗаписей.Индекс(Запись);
		База = ТЗ_База[Индекс].Сумма;

		Если Запись.ВидРасчета = ПланыВидовРасчета.Начисления.ПремияПроцентом Тогда
			//ТЗ_База = Запись.ПолучитьБазу(СписокРесурсов, ОтборПоИзмерениям);
			//База = ТЗ_База[0].Сумма;

			Запись.Сумма = База * Запись.Параметр / 100;
```

```
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

Индекс – переменная, содержащая индекс записываемой записи. Можно получить, обратившись к записи:

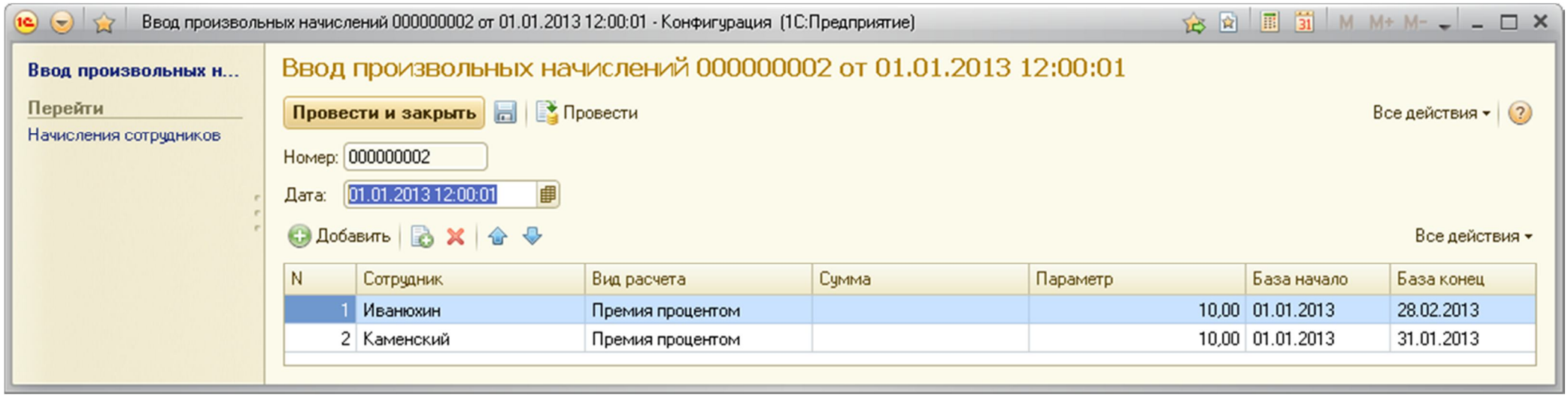
```
Индекс = Запись.НомерСтроки - 1;
```

или обратившись к набору записей (более правильный вариант):

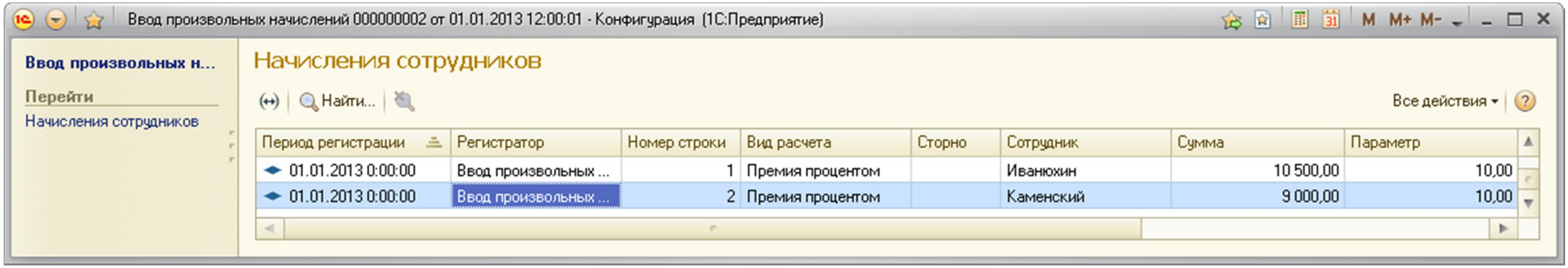
```
Индекс = НаборЗаписей.Индекс(Запись);
```

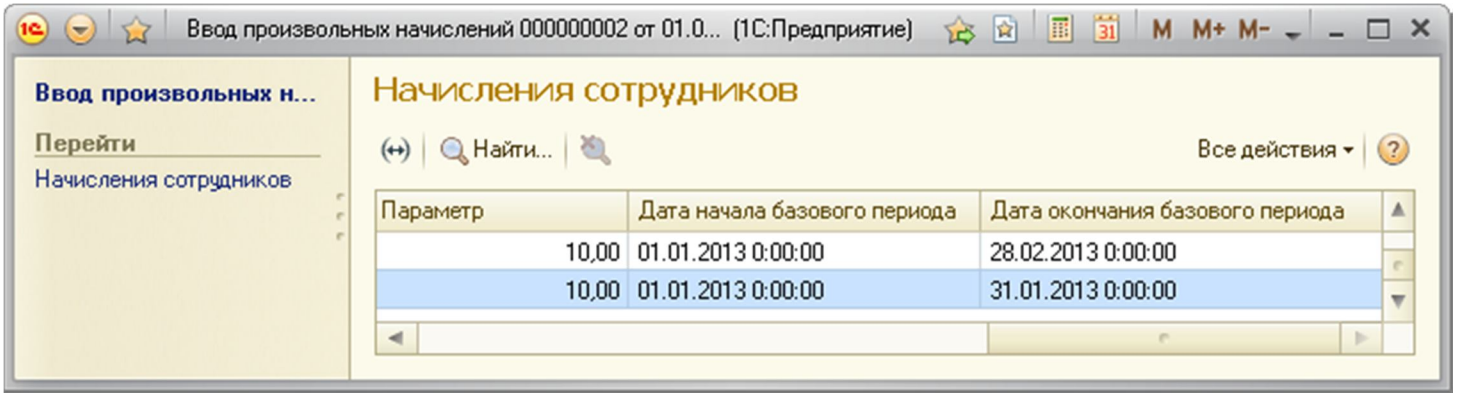
Если точно известно, что в наборе записей только один элемент, то можно использовать метод «ПолучитьБазу()»записи, в других случаях используется метод «ПолучитьБазу()» самого регистра.

В режиме исполнения:

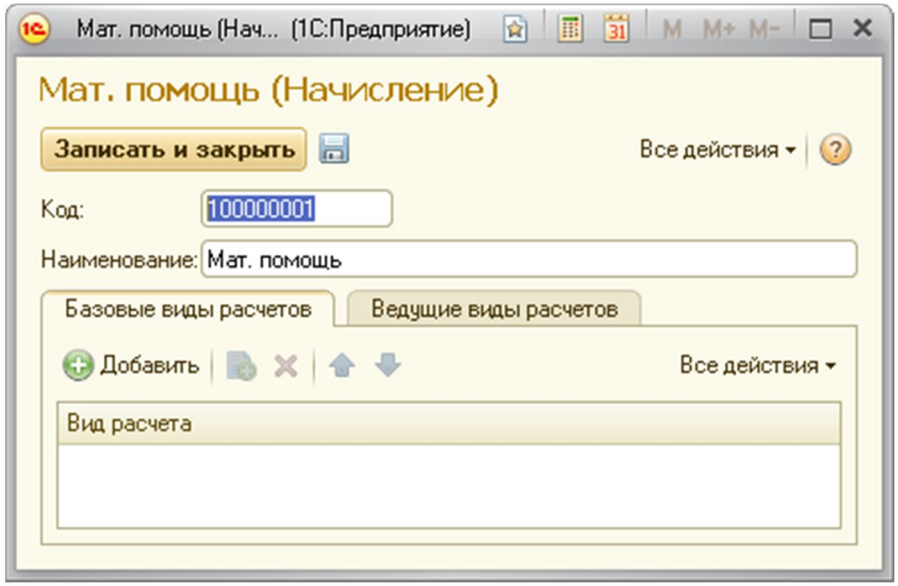


Перепроведем документ и проверим движения:

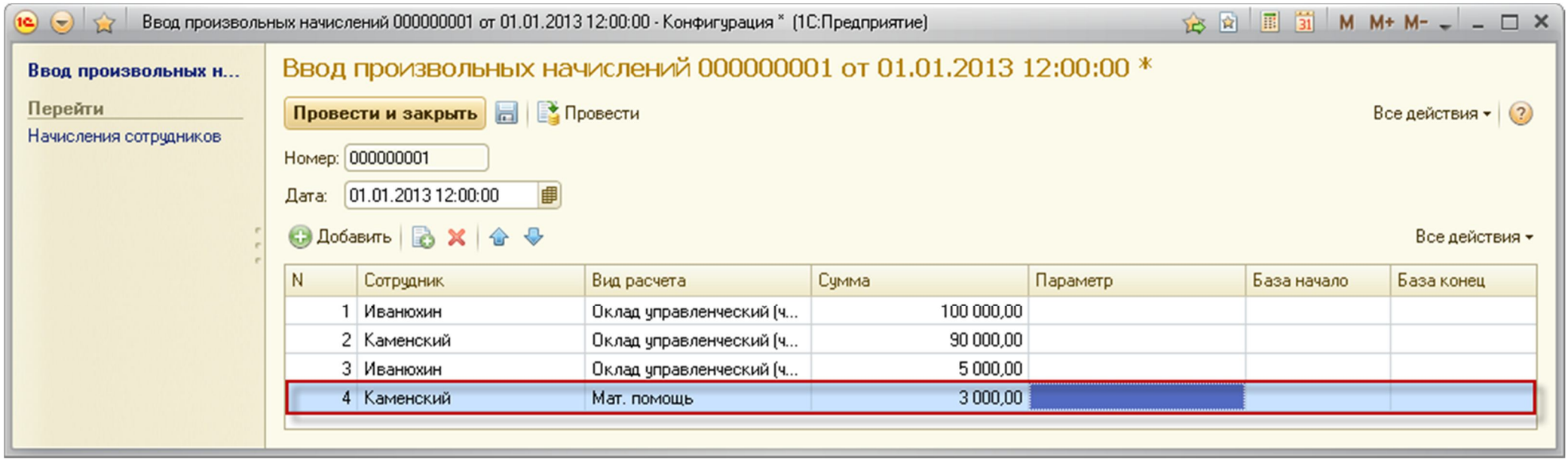




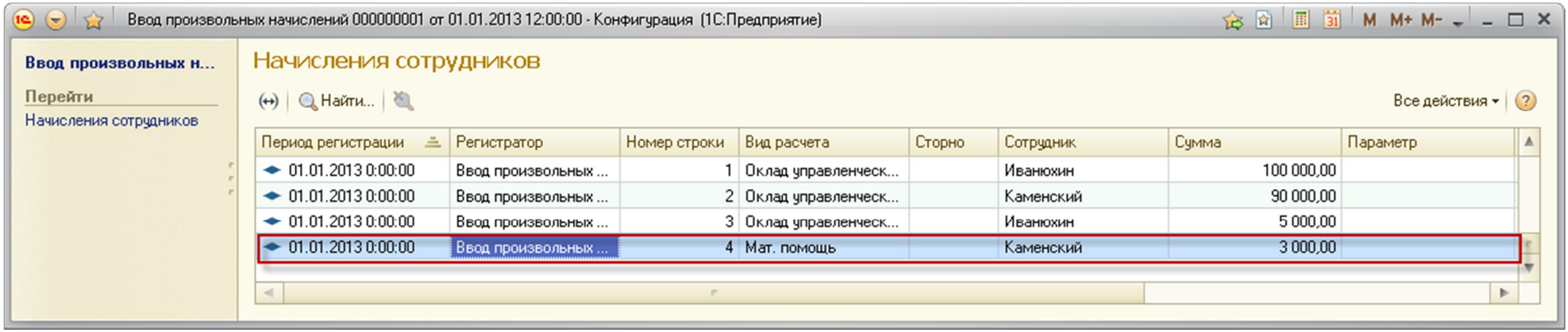
Пользователь может ввести данные вручную – создать вид расчета, который тоже должен рассчитываться.
В режиме исполнения создадим новый вид расчета «Материальная помощь»:

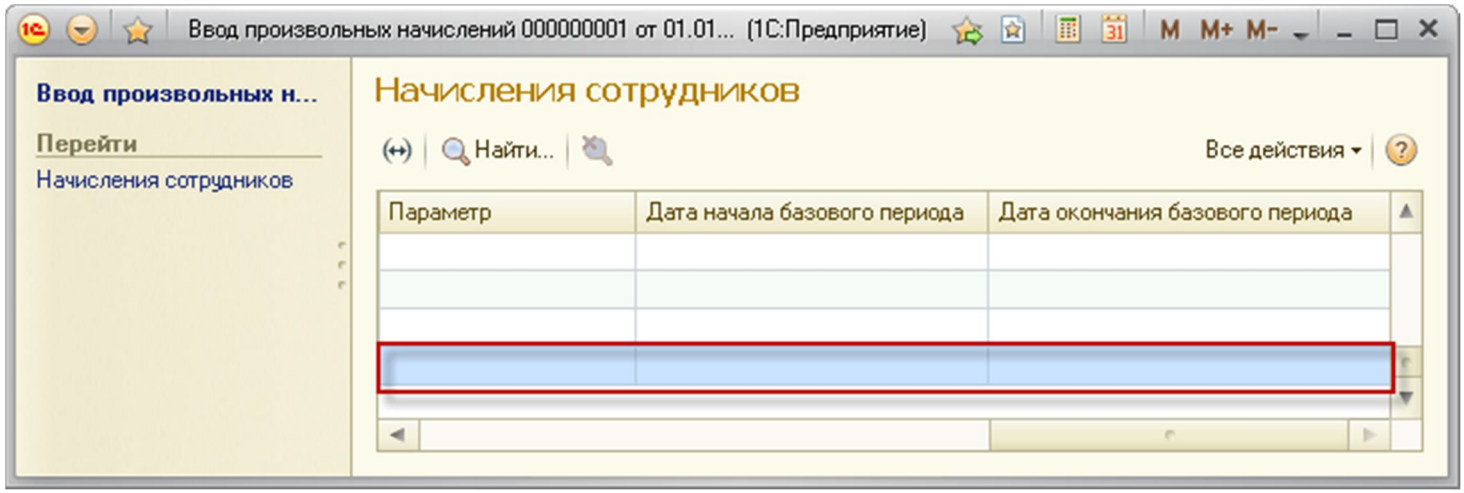


Начислим материальную помощь:



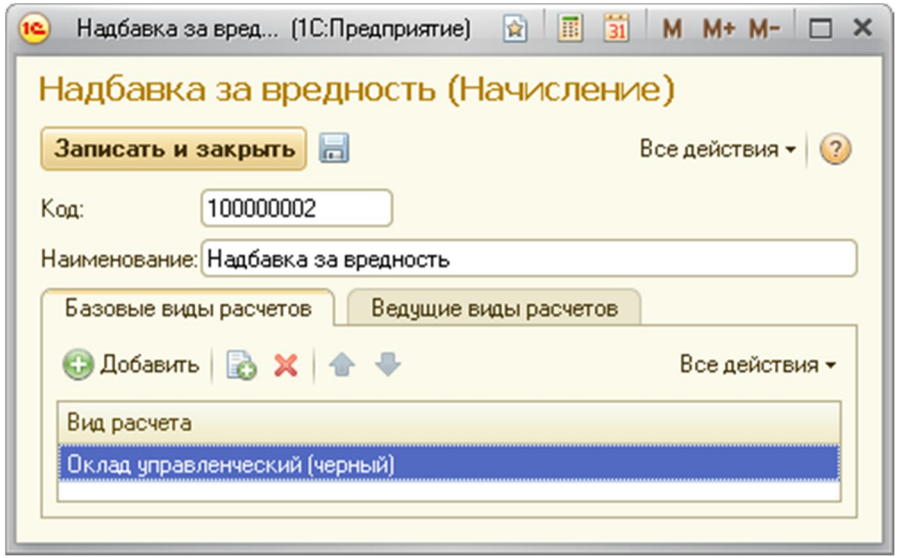
Проверим движения:



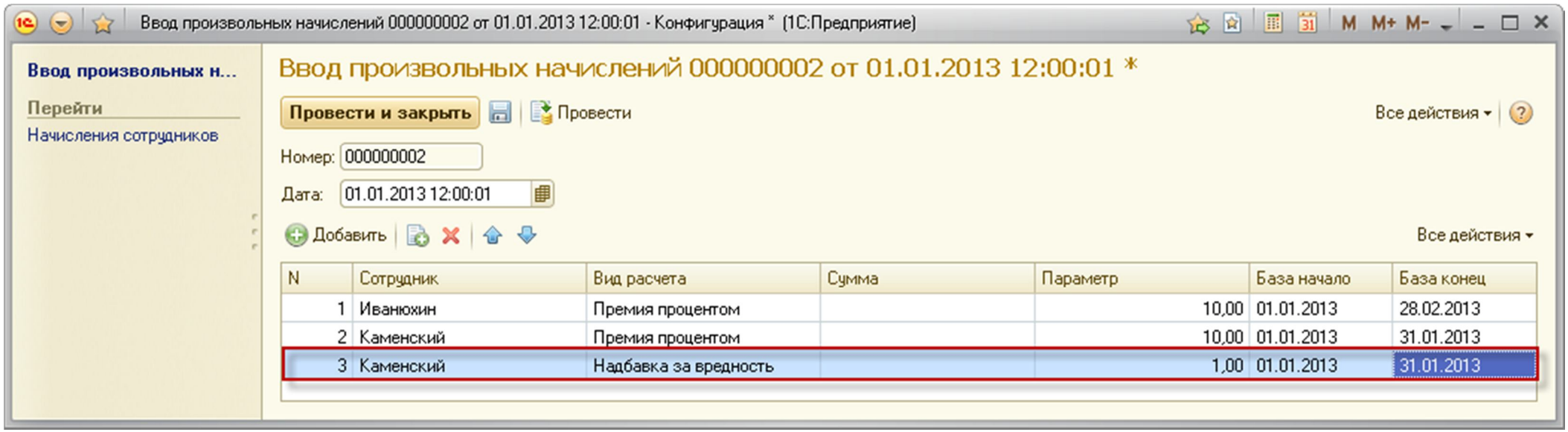


Видим, что материальная помощь рассчиталась.

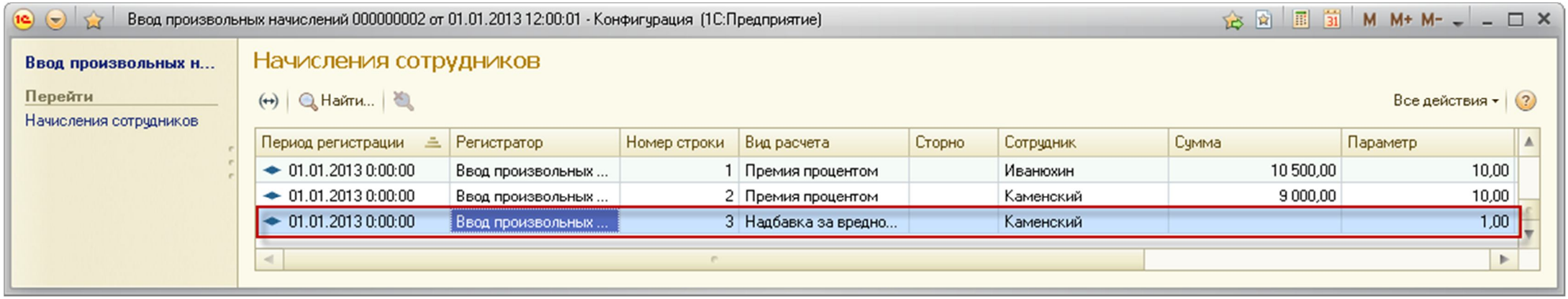
Введем еще один вид начислений:

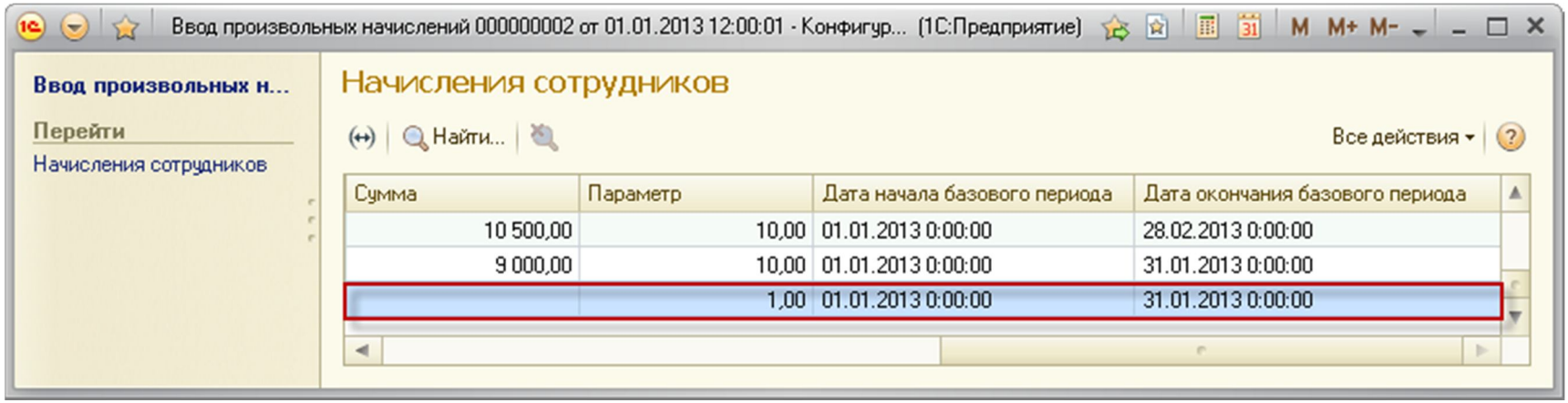


Применим начисление:



Проверим движения:





Начисление не рассчитается, т.к. не описан алгоритм расчета.
Вывод: необходимо описание стандартных алгоритмов расчета, например «Премия процентом» и «Надбавка за вредность» рассчитываются по одному алгоритму.

Создадим перечисление «Способы расчета», которое будет хранить названия стандартных алгоритмов расчета:

Свойство	Значение
Имя	СпособыРасчета
Синоним	Способы расчета

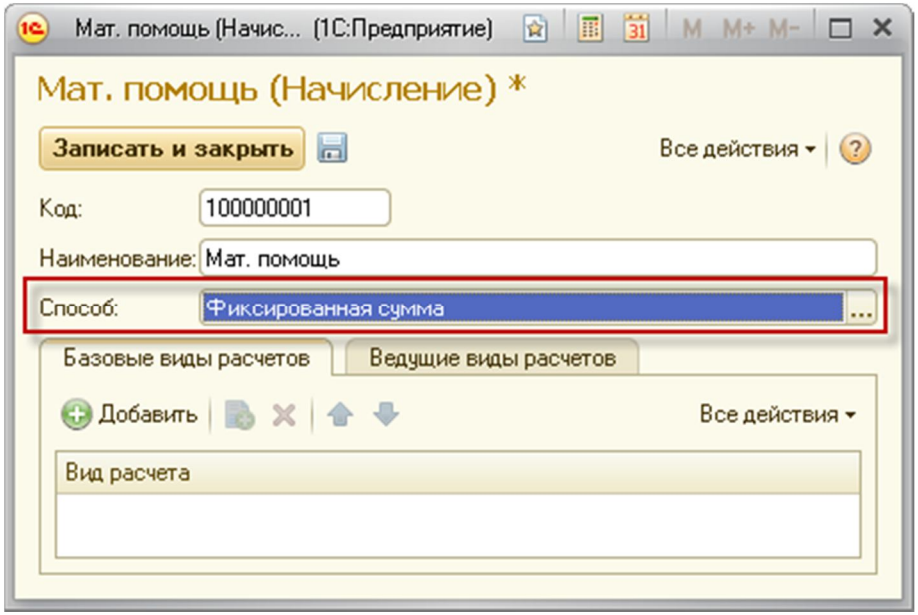
Закладка «Данные»	
Значения	
ФиксированнаяСумма	
Имя	ФиксированнаяСумма
Синоним	Фиксированная сумма
Процентом	
Имя	Процентом
Синоним	Процентом

Теперь к каждому виду расчета привяжем способ расчета:

«План видов расчета – закладка Данные» - создадим реквизит «Способ»:

Свойство	Значение
Имя	Способы
Синоним	Способы
Тип	ПеречислениеСсылка.СпособыРасчета

В режиме исполнения:



Код	Наименование	Способ
100000001	Мат. помощь	Фиксированная сумма
100000002	Надбавка за вредность	Процентом
000000001	Оклад управленческий (черный)	Фиксированная сумма
000000002	Премия процентом	Процентом

Теперь изменим процедуру «Рассчитать()» общего модуля «Расчеты»:

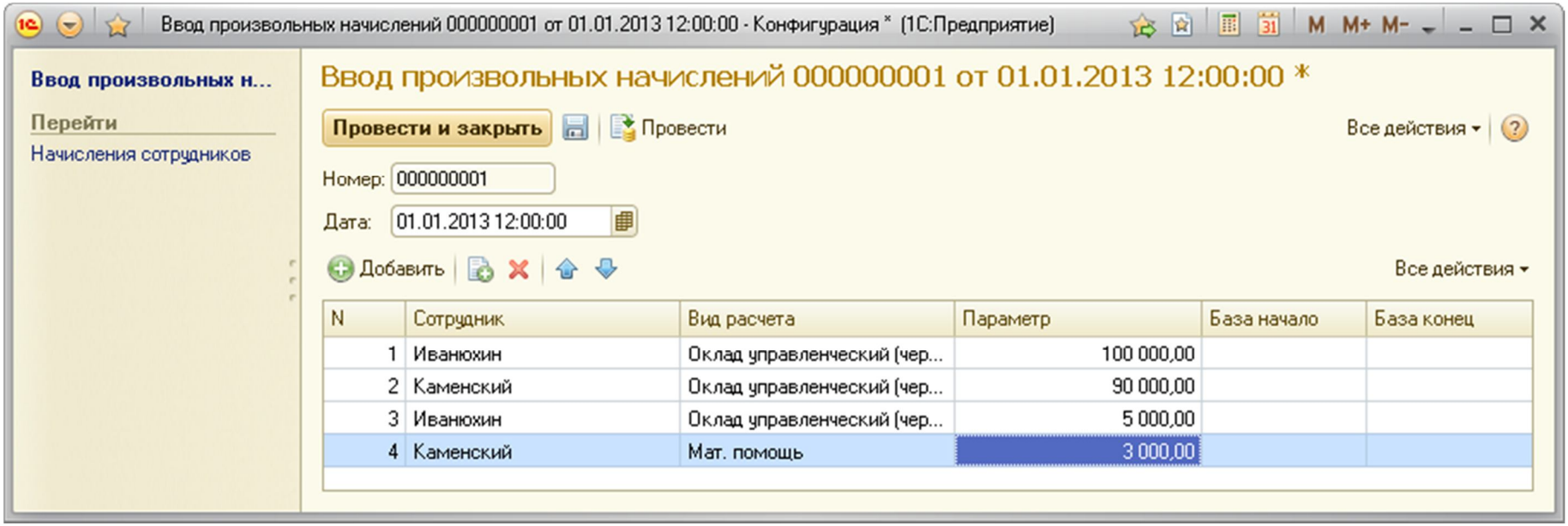
```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");
```

```
    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    Для Каждого Запись Из НаборЗаписей Цикл
        Индекс = НаборЗаписей.Индекс(Запись);
        База = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Запись.Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
            Запись.Сумма = Запись.Параметр;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

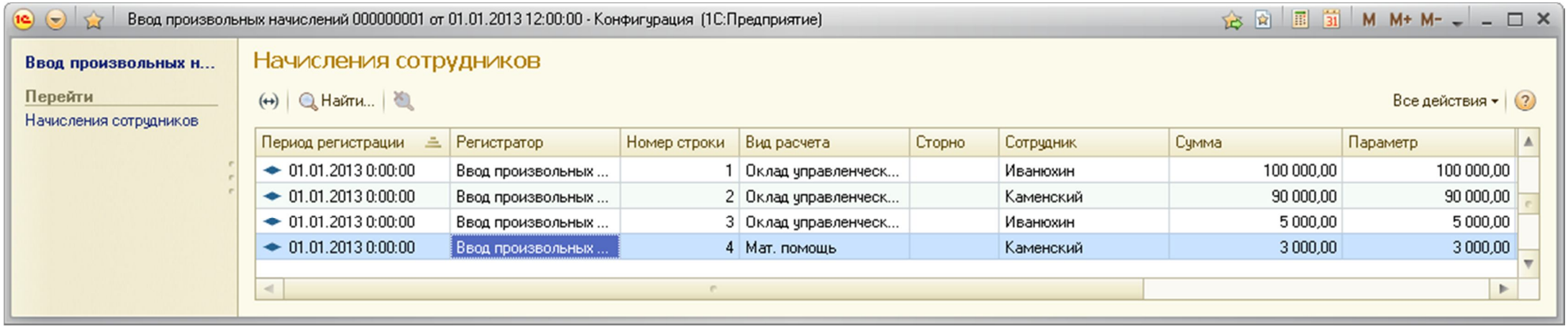
Теперь реквизит «Сумма» можно вообще удалить из документа «ВводПроизвольныхНачислений», он больше там не будет использоваться, т.к. сумму пользователь сможет вводить в реквизит «Параметр».

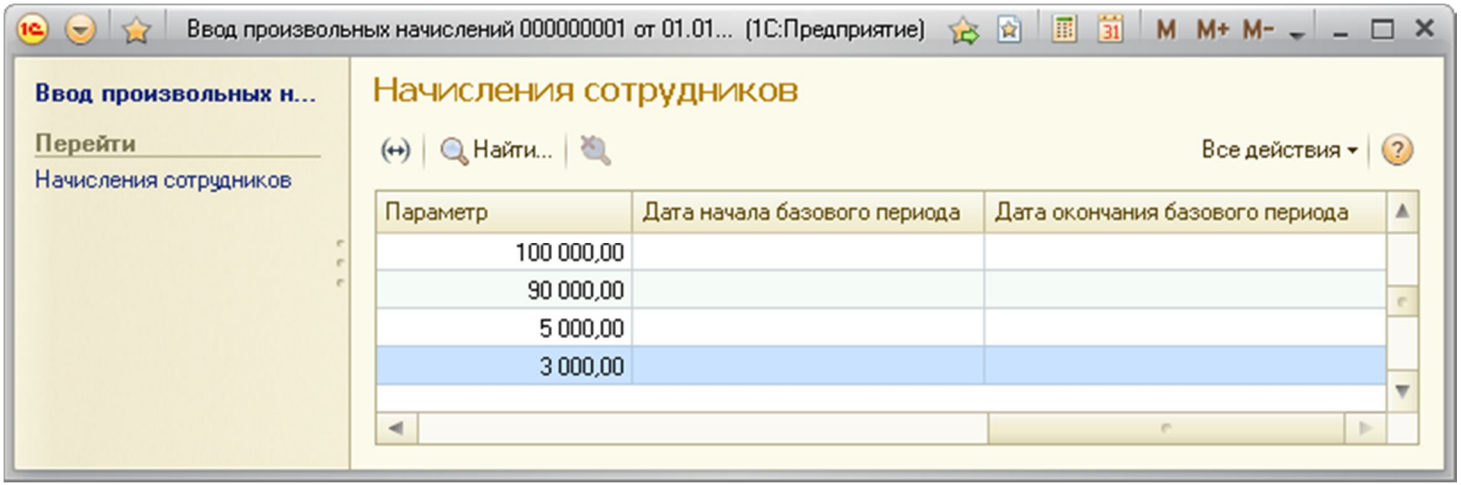
В режиме исполнения:

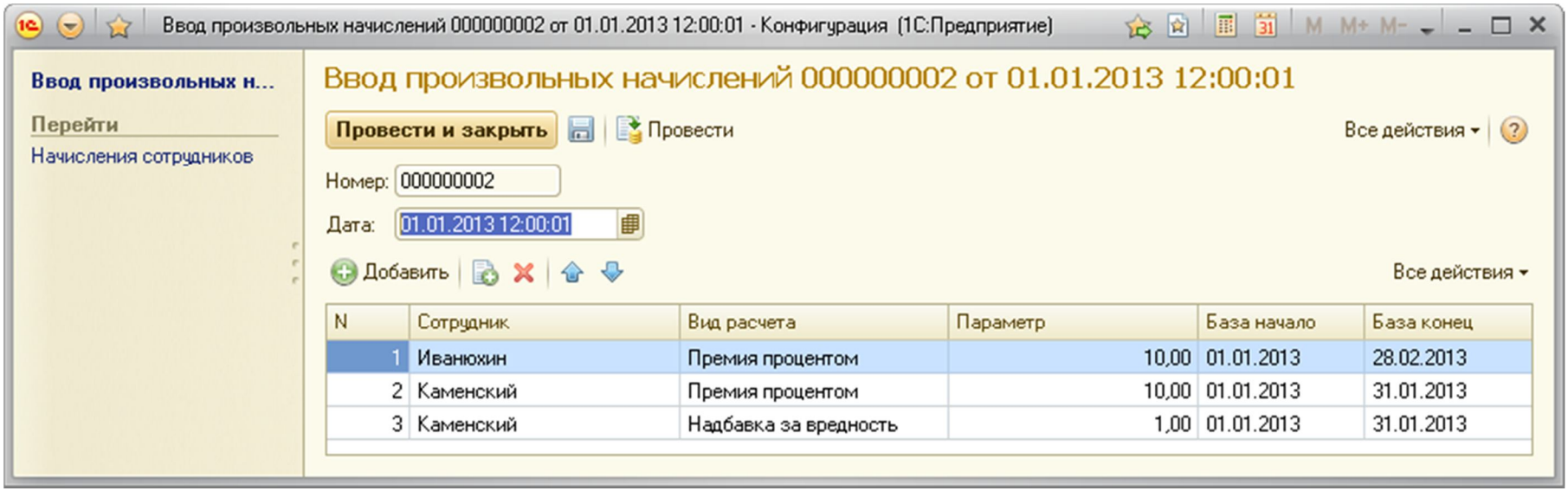
Введем оклады сотрудникам в документе «Ввод произвольных начислений»:

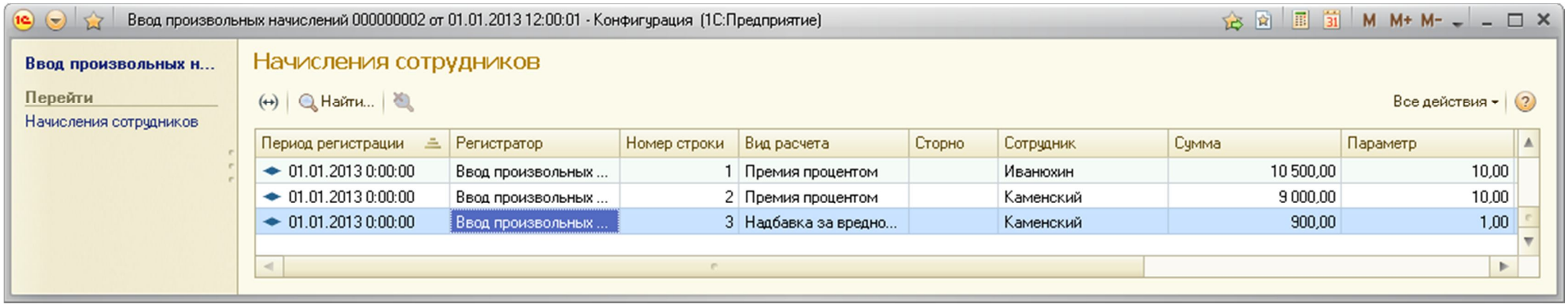


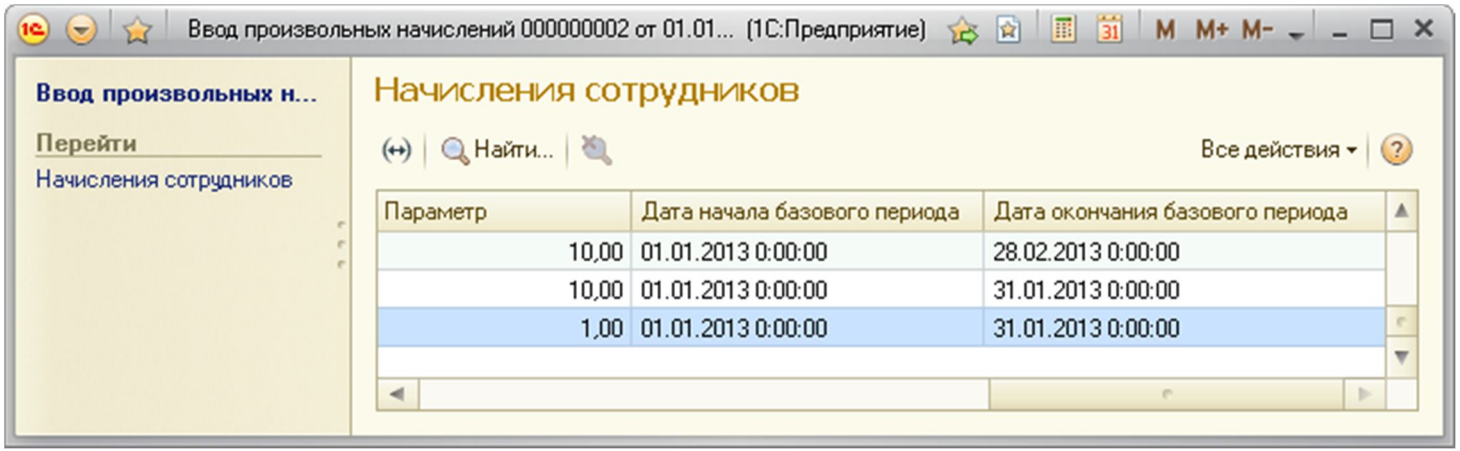
Проведем документы и посмотрим движения:






Ввод произвольных начислений 000000002 от 01.01.2013 12:00:01 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000002 от 01.01.2013 12:00:01
Провести и закрыть
Провести
Все действия
Номер: 000000002
Дата: 01.01.2013 12:00:01
Добавить
Все действия
N
Сотрудник
Вид расчета
Параметр
База начало
База конец
1 Иванюхин Премия процентом 10,00 01.01.2013 28.02.2013
2 Каменский Премия процентом 10,00 01.01.2013 31.01.2013
3 Каменский Надбавка за вредность 1,00 01.01.2013 31.01.2013





Надо рассчитывать ЗП по отработанному времени:
1. По дням – т.е. ЗП начисляется за отработанный день. Например, 1000 рублей в день. Расчетчик при этом ведет Табель отработанных дней.
2. Оклад. Если сотрудник отработал месяц (независимо от количества рабочих дней в месяце), то он полностью получит сумму оклада. Называется «Расчет пропорционально отработанного времени».

Для расчета по обоим способам надо знать, какое количество дней отработал сотрудник. Причем, если прогул сотрудника выпал на выходной, то он не считается прогулом. Т.е. надо хранить информацию о рабочих и выходных днях. Для этого создадим подобие «Производственного календаря»:

Дата	Признак
11.07.2013	V
12.07.2013	V
13.07.2013	X

Создадим регистр сведений «Календарь»:

Свойство	Значение
Имя	Календарь
Синоним	Календарьрасчета
Периодичность	Непериодический
Закладка «Данные»	
Измерения	
Дата	
Имя	Дата
Синоним	Дата
Тип	Дата
Запрет незаполненных значений	Истина
Ресурсы	
Признак	
Имя	Признак
Синоним	Признак
Тип	Число 1, 0
Неотрицательное	Истина

Тип Ресурса «Признак» можно было бы сделать «Булево», но при этом неудобно будет считать количество рабочих дней за период (при выполнении расчета будет производиться конвертация данных из «Булево» в «Число», что потребует больше ресурсов).

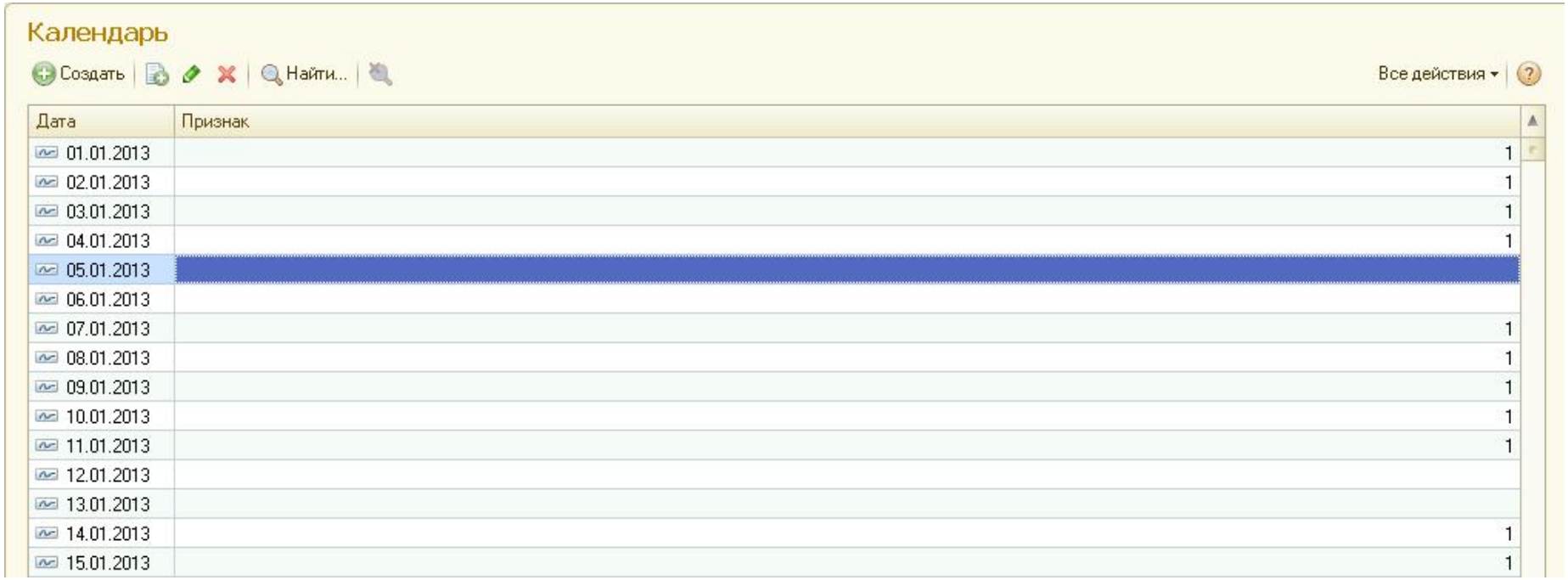
Чтобы не заполнять регистр «Календарь» вручную, напишем обработку:

Свойство	Значение
Имя	ЗаполнениеКалендаря
Синоним	Заполнение календаря

В форме будет только одна команда (команды формы) «Заполнить». Код модуля формы:

```
&НаКлиенте
Процедура Заполнить(Команда)
	ЗаполнитьНаСервере();
	Предупреждение(НСтр("ru = 'Обработка завершена'"), 60);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура ЗаполнитьНаСервере()
	НачатьТранзакцию();
	//переберем текущий год
	Для Ном = 0 По 365 Цикл
		//рассчитаем текущую дату
		ТекДата = '20130101' + 86400 * Ном;
		//сформируем записи в регистре сведений
		Запись = РегистрыСведений.Календарь.СоздатьМенеджерЗаписи();
		Запись.Дата = ТекДата;
		//по признаку пятидневки
		Запись.Признак = ?(ДеньНедели(ТекДата) < 6, 1, 0);
		Запись.Записать();
	КонецЦикла;
	ЗафиксироватьТранзакцию();
КонецПроцедуры // ЗаполнитьНаСервере()
```

В режиме исполнения:
Выполним обработку:



Добавим новый способ расчета (перечисление «СпособыРасчета») «По дням»:
Закладка Данные – Значения перечисления:

Свойство	Значение
Имя	ПоДням
Синоним	По дням

Теперь, чтобы иметь возможность рассчитывать записи регистра расчета «НачисленияСотрудников» в разрезе дней, свяжем регистр расчета с регистром сведений «Календарь»:

Свойства регистра расчета «НачисленияСотрудников»:

Свойство	Значение
Период действия	Истина
График	Календарь
Значение графика	Признак
Дата графика	Дата

«Период действия» - в регистре расчета создаются новые поля, главные из которых:

- ПериодДействияНАЧАЛО
- ПериодДействияКОНЕЦ

Поля:

- График – регистр сведений, в котором хранится информация о графиках.
- Значение графика – ресурс.
- Дата графика – измерение.

это связь регистра расчета с регистром сведений.

		Период действия	
		Начало	Конец
Иванов	Оклад по дням	с 01.01.2013	по 18.01.2013

Календарь	
Дата	Признак
01.01.2013	1
02.01.2013	0
03.01.2013	0
04.01.2013	1

Слева – запись регистра расчета.
Справа – записи регистра сведений «Календарь».

При связи регистра расчета с регистром сведений, можно получать данные графика, указав только период, за который они нужны. Пример: Сколько дней отработал Иванов? Система возьмет поля «ПериодДействияНачало/ ПериодДействияКонец», наложит отбор на Измерение «Дата» регистра сведений и просуммирует поле «Признак» по этому отбору, вернув количество рабочих дней в указанном периоде.

Исправим документ «ВводПроизвольныхНачислений» так, чтобы он записывал поля:

- ПериодДействияНАЧАЛО
- ПериодДействияКОНЕЦ

Закладка «Данные» - ТЧ «НачисленияСотрудников» - добавим поля:

Свойство	Значение
Период действия НАЧАЛО	
Имя	ПериодДействияНАЧАЛО
Синоним	Период действия НАЧАЛО
Тип	Дата
Период действия КОНЕЦ	
Имя	ПериодДействияКОНЕЦ
Синоним	Период действия КОНЕЦ
Тип	Дата

При проведении документа движения в регистр расчета запишутся автоматом, т.к. при проведении мы используем метод:

```
ЗаполнитьЗначенияСвойств(Движение, Стр);
```

Надо только рассчитать записи в общем модуле «Расчеты»:

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ФактическийПериодДействия);

    Для Каждого Запись Из НаборЗаписей Цикл
        //Индекс = Запись.НомерСтроки - 1;
        Индекс = НаборЗаписей.Индекс(Запись);
        База = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;
        ФактДней = ТЗ_ФактДней[Индекс].Признак;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Запись.Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
```

```
            Запись.Сумма = Запись.Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
            Запись.Сумма = ФактДней * Запись.Параметр;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

Добавляем еще один способ расчета:

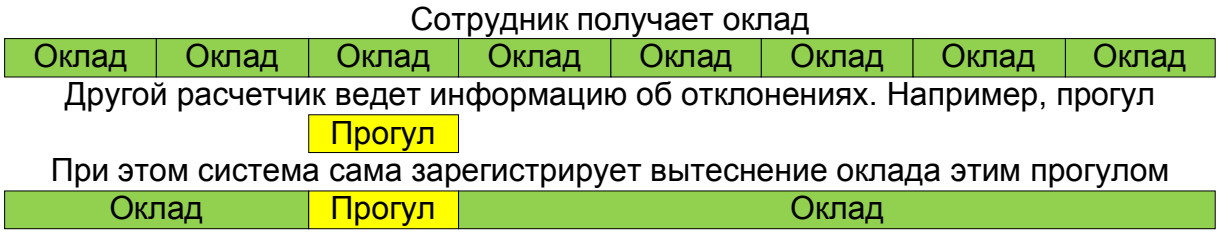
```
ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
```

Получим данные графика - сколько дней отработал сотрудник за определенный период:

```
ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ФактическийПериодДействия);
```

«ВидПериодаРегистраРасчета» - за какой период надо посчитать количество дней:

- БазовыйПериод – вернет количество дней в базовом периоде.
- ПериодДействия – вернет количество дней в том месяце, в котором действует запись.
- ПериодРегистрации – вернет количество дней в том месяце, в котором зарегистрирована запись.
- ФактическийПериодДействия – информация о реально отработанных днях. Например:



В итоге будет две записи по окладу (с разрывом).
Информация о том, что произошло в результате всех вытеснений и есть «ФактическийПериодДействия».

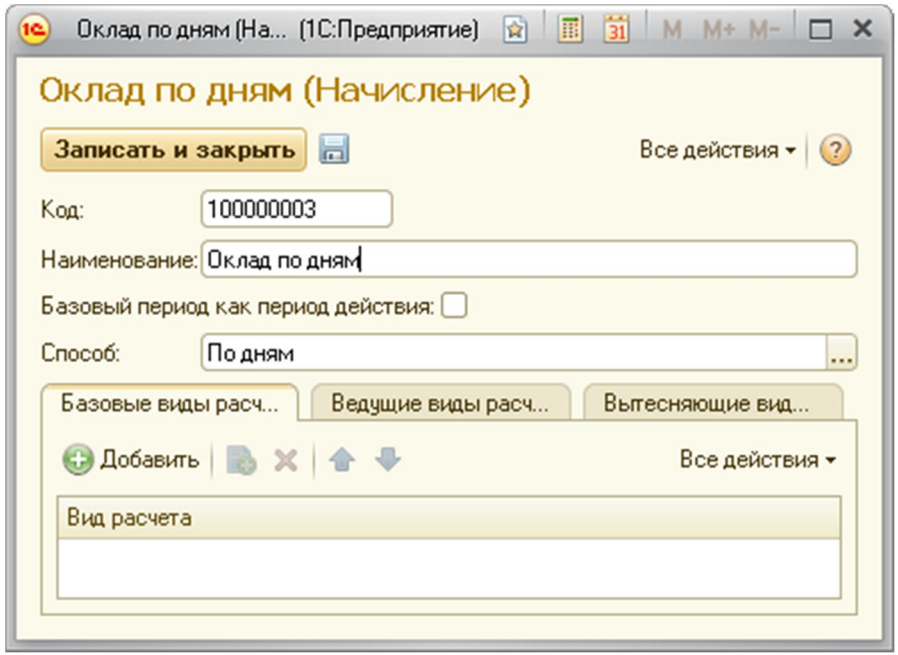
Отличие «ПериодДействия» от «ПериодРегистрации» - например, в феврале задним числом регистрируется информация о больничном сотрудника, т.е.
«ПериодДействия» - январь.
«ПериодРегистрации» - фквраль.

Получим количество фактически отработанных дней:

```
ФактДней = ТЗ_ФактДней[Индекс].Признак;
```

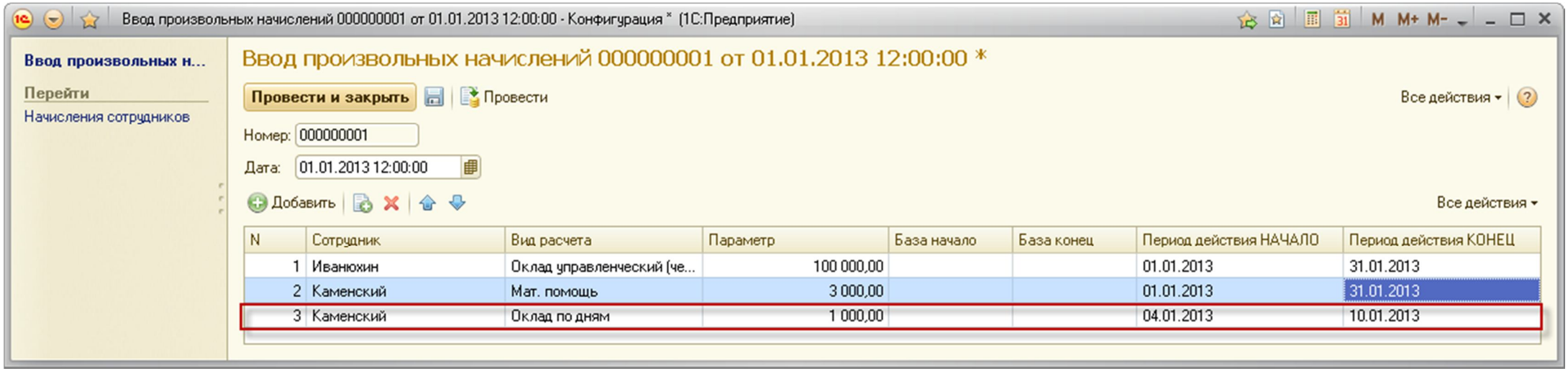
Чтобы проверить расчет в режиме исполнения, надо включить параметр «Использует период действия» в Плане видов расчета «Начисления», на закладке Данные. Позволяет указывать, что вытесняет виды расчета

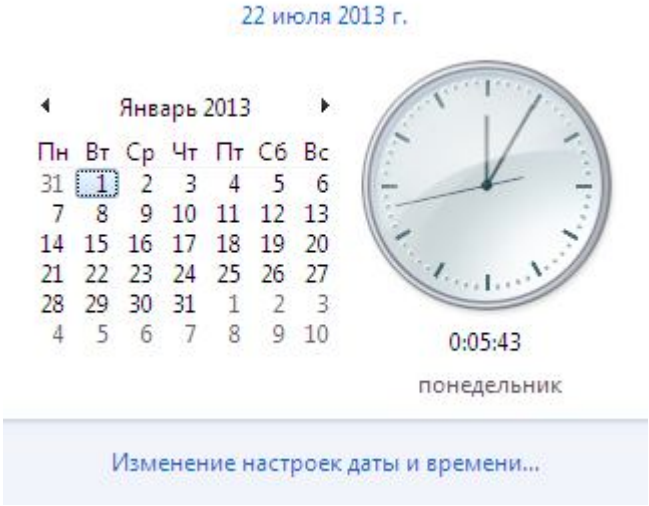
Добавим еще один вид начисления:



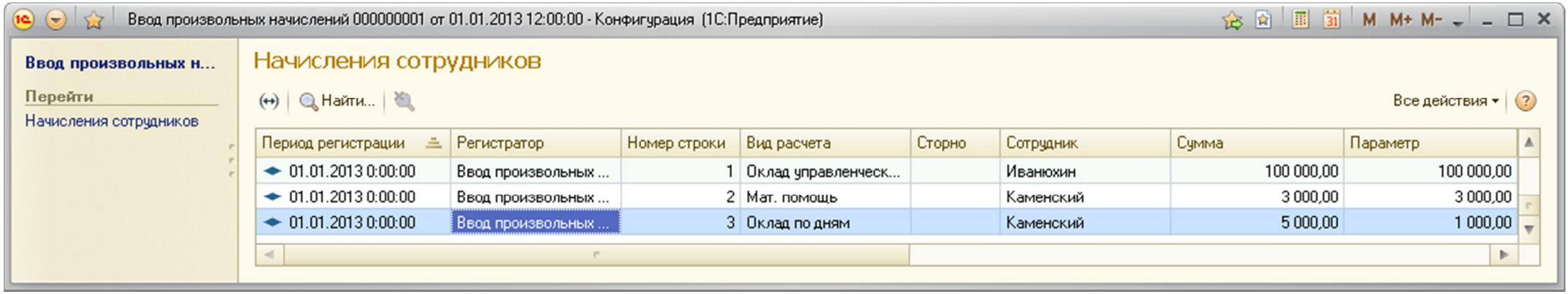
Откроем и отредактируем документ «Ввод произвольных начислений»:

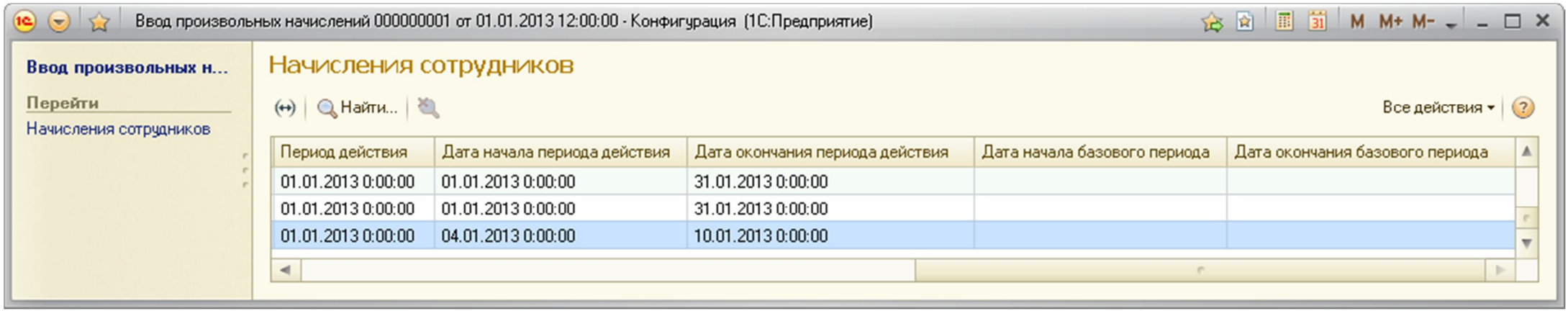
Сотруднику «Каменскому» назначим оклад по дням и будем платить 1000 рублей в день:



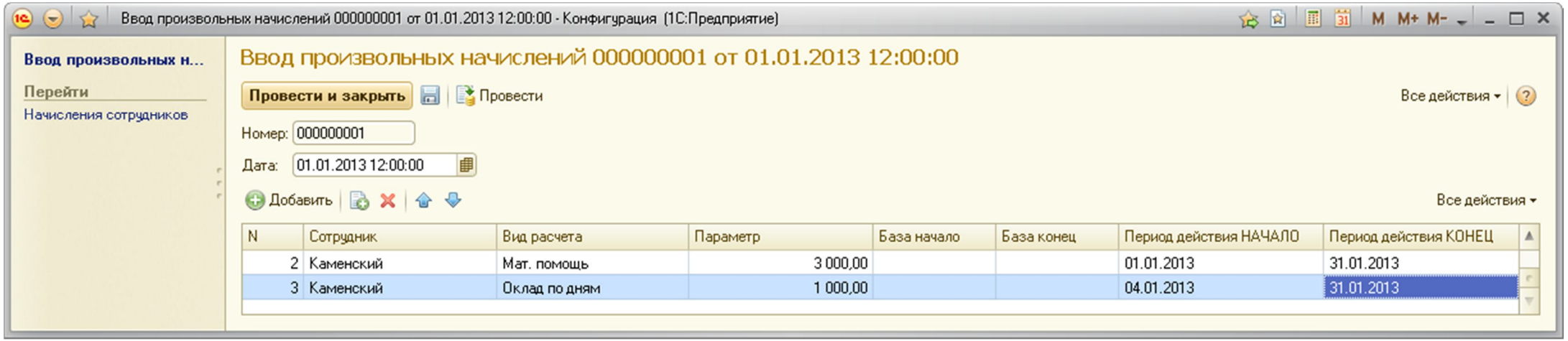


Проведем документ и проверим результат:

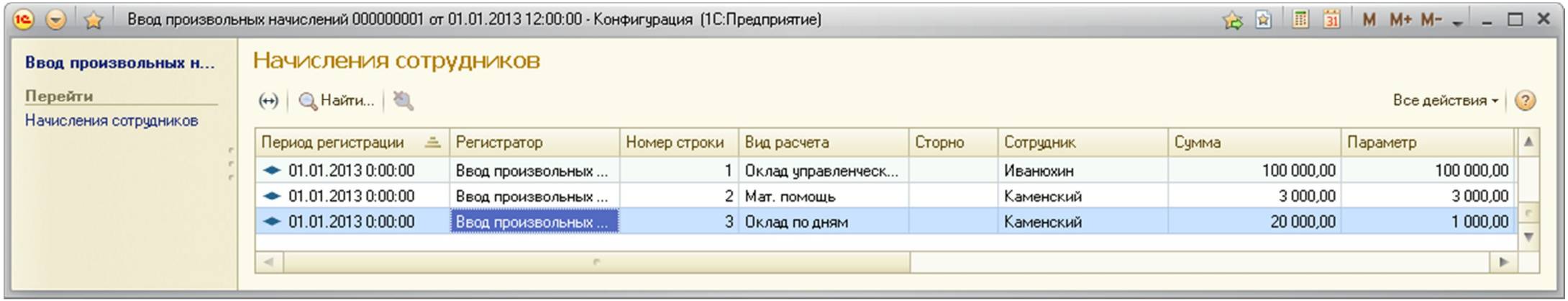


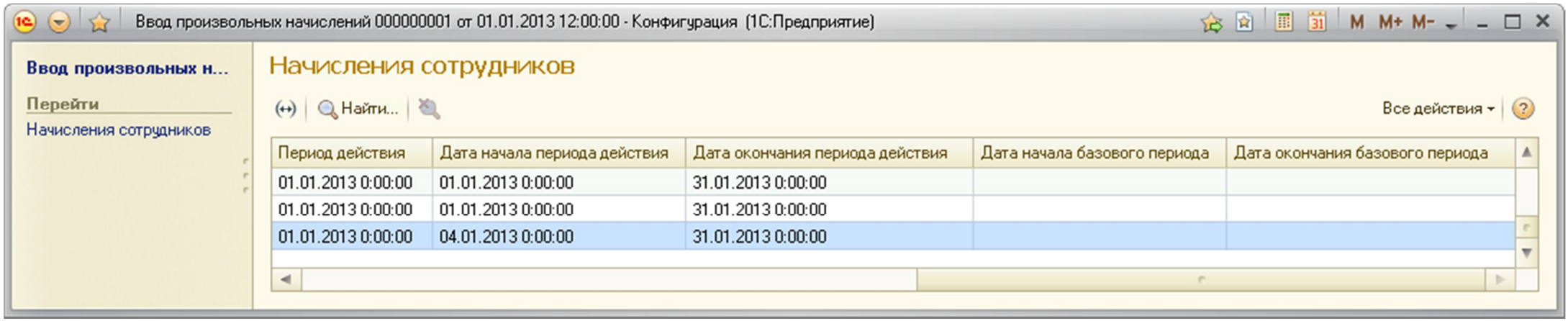


Произведем расчет полностью за январь:



Результат:





Мы рассмотрели показатель Регистра расчета – «Данные графика», для его работы обязательно наличие Регистра сведений, в котором хранится этот самый график (в примере «Календарь»).

Занятие 3.27

Реализуем вид расчета, рассчитывающий записи пропорционально отработанному времени – пользователь будет указывать в записи сумму оклада за месяц, а система будет рассчитывать количество отработанных сотрудником дней, которые необходимо будет оплатить.

Необходимо получить таблицу с нормой дней за месяц:

```
ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
                    ОтборПоРегистратору,
                    ВидПериодаРегистраРасчета.ПериодДействия);
```

И норму дней:

```
НормаДней = ТЗ_НормаДней[Индекс].Признак;
```

Опишем расчет записей:

В перечисление «Способы расчета» добавим значение «Пропорционально отработанному времени»

Свойство	Значение
Имя	ПропорциональноОтработанномуВремени
Синоним	Пропорционально отработанному времени

И опишем расчет:

```
Запись.Сумма = ФактДней / НормаДней * Параметр;
```

Если РС «Календарь» не заполнен, то «НормаДней» может быть равна нулю.

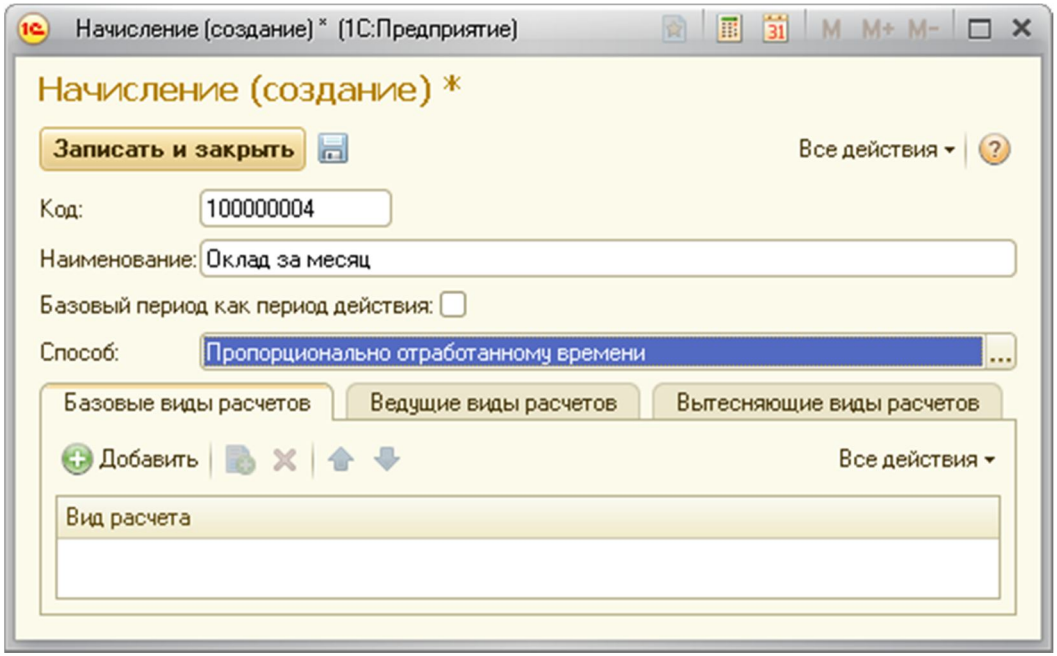
Итого код (общий модуль «Расчеты»):

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ФактическийПериодДействия);

    ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ПериодДействия);

    Для Каждого Запись Из НаборЗаписей Цикл
        //Индекс = Запись.НомерСтроки - 1;
        Индекс = НаборЗаписей.Индекс(Запись);
        База = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;
        ФактДней = ТЗ_ФактДней[Индекс].Признак;
```

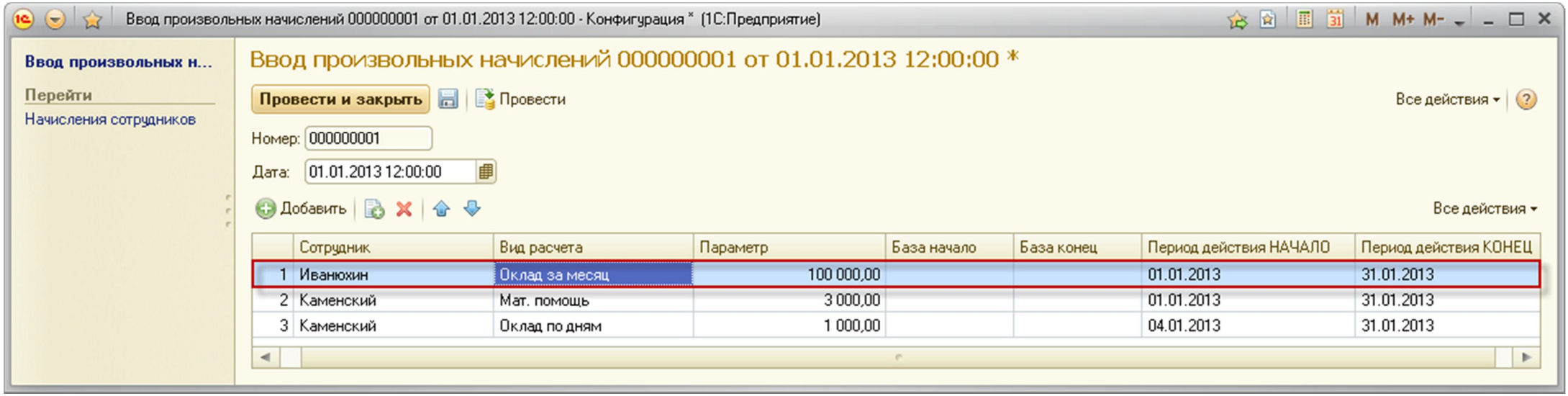
```
        НормаДней = ТЗ_НормаДней[Индекс].Признак;
        Параметр = Запись.Параметр;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
            Запись.Сумма = Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
            Запись.Сумма = ФактДней * Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
            Запись.Сумма = ФактДней / НормаДней * Параметр;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

В режиме исполнения:

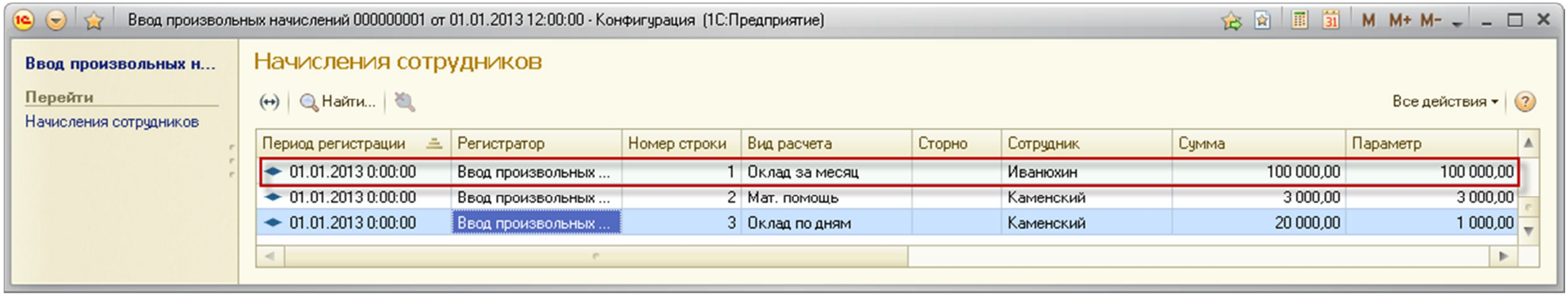
Добавим новое начисление:

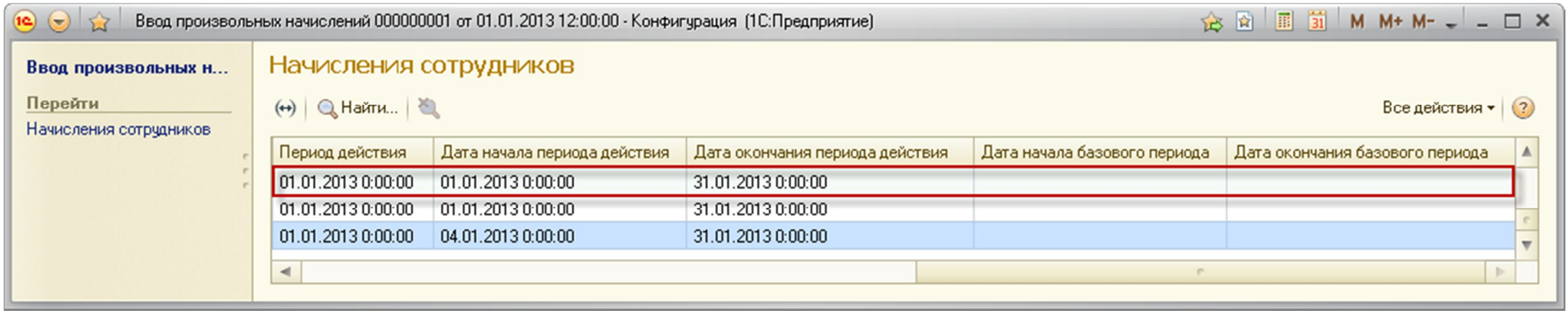


Изменим документ «Ввод произвольных начислений»:



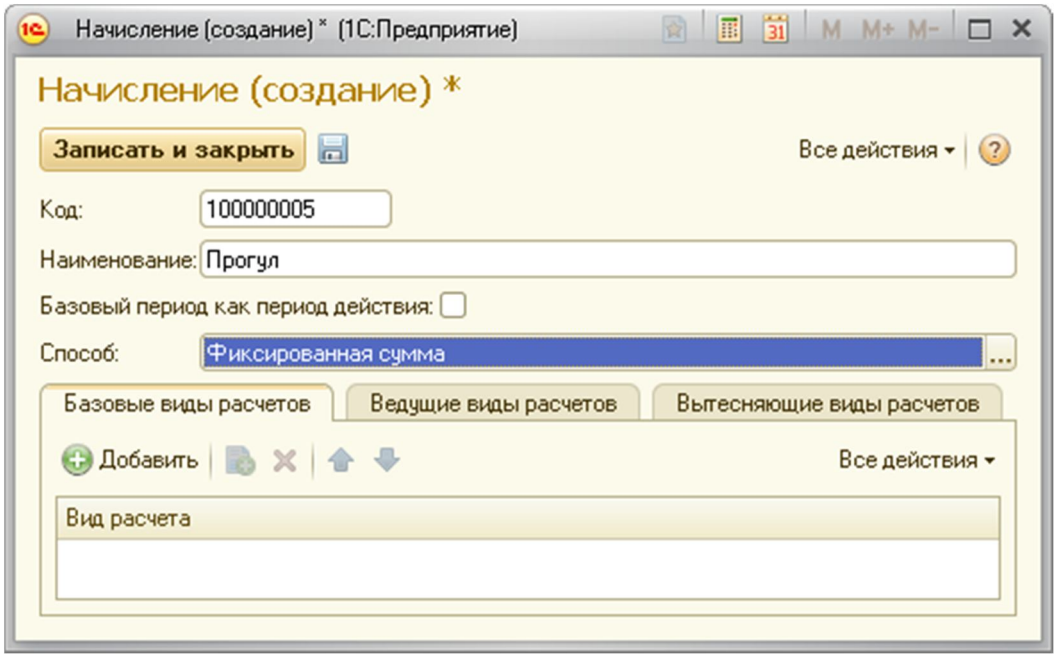
Проведем документ и посмотрим результат (Иванюхин проработал полный месяц, поэтому сумма к начислению должна быть равна 100 000):



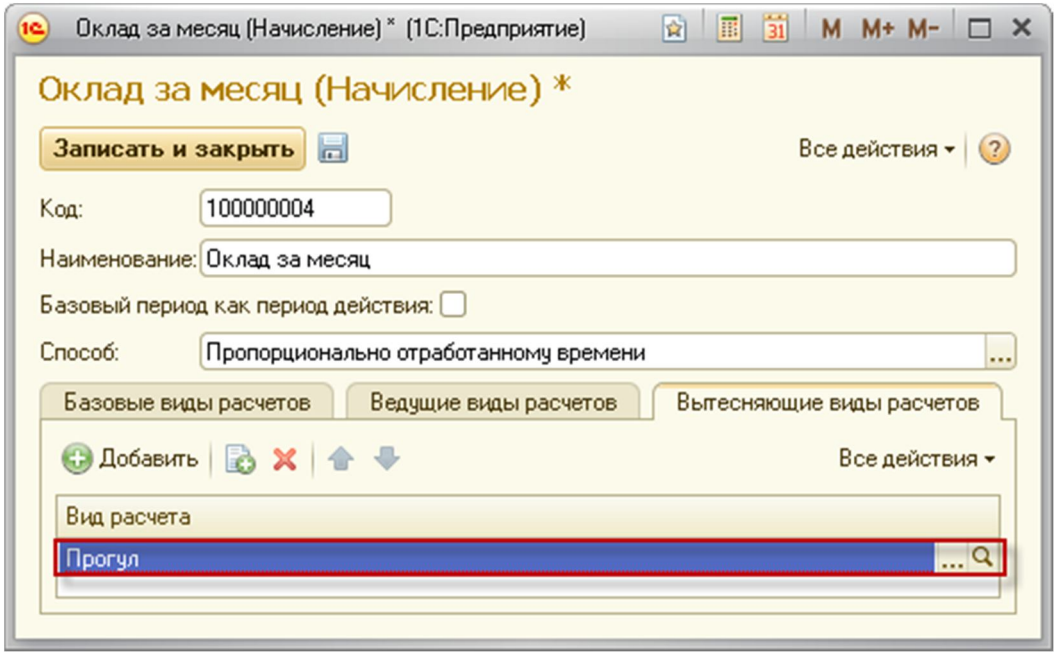


Теперь реализуем возможность ввода отклонений (прогулов, больничных и т.д.) – для этого даже не надо делать каких-либо настроек в конфигураторе:

Создадим вид расчета «Прогул»:



Чтобы Прогул уменьшил Оклад за месяц, надо в свойствах соответствующего вида расчета указать вытесняющие свойства:



Вытеснение – это конкуренция за период действия. Пример:

Оклад	Оклад	Оклад	Оклад	Оклад	Оклад	Оклад	Оклад
		Прогул			Прогул		
Оклад	Оклад	Прогул	Оклад	Оклад	Прогул	Оклад	Оклад

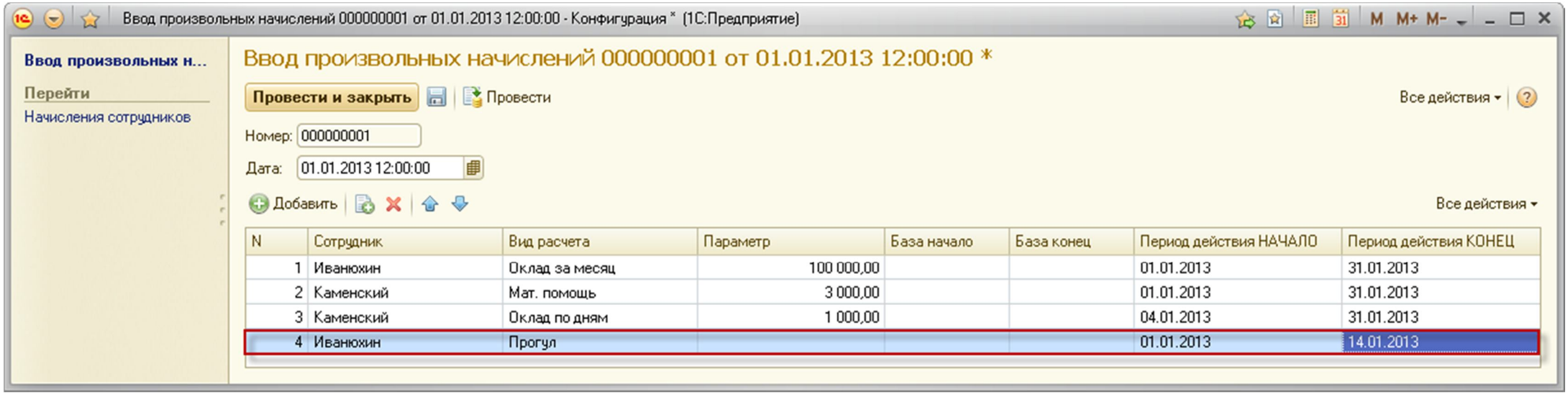
Если записи Оклада и Прогула пересекаются по хронологии, то система разобьет Оклад, т.е. вытеснит у Оклада период действия, который занял Прогул. При этом количество записей оклада станет 3 (в примере):

- с начала 1-го оклада по начало 1-го прогула.
- с конца 1-го прогула до начала 2-го прогула.
- с конца 2-го прогула до конца записи.

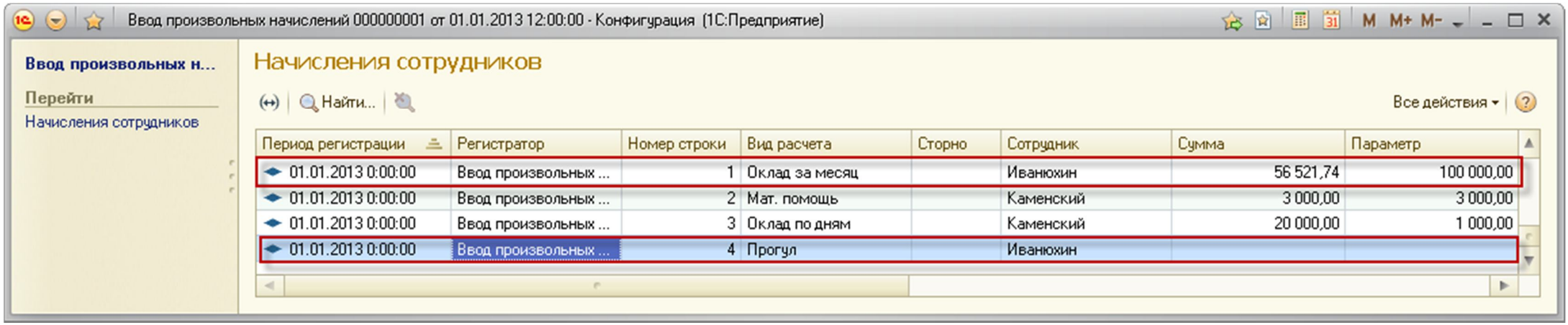
Важно помнить, что запись Оклада помнит:

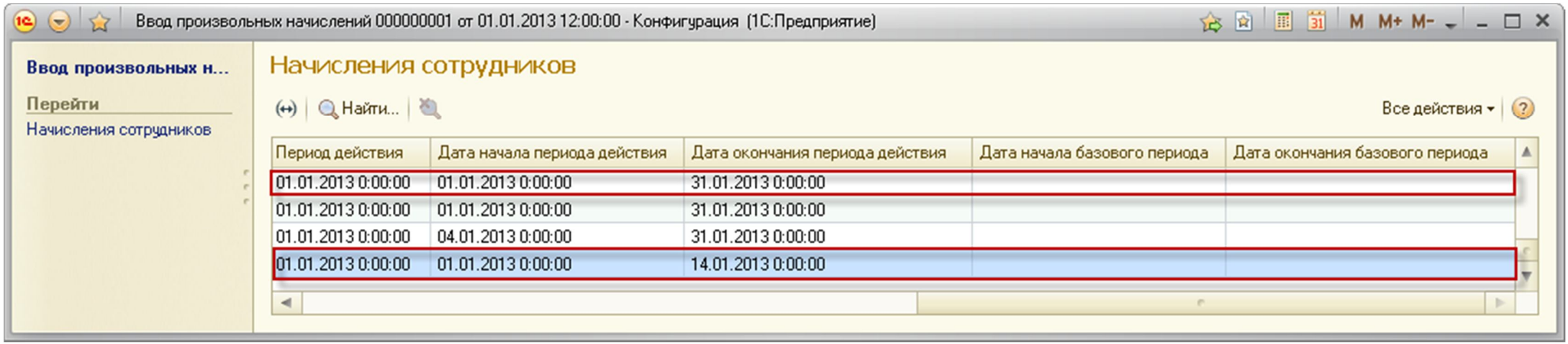
- Эталон – сколько она занимала времени при вводе.
- Результат вытеснения – сколько стало записи Оклада (в количестве дней) – фактический период действия.

Возьмем документ «Ввод произвольных начислений» и зарегистрируем Прогул Иванюхину:



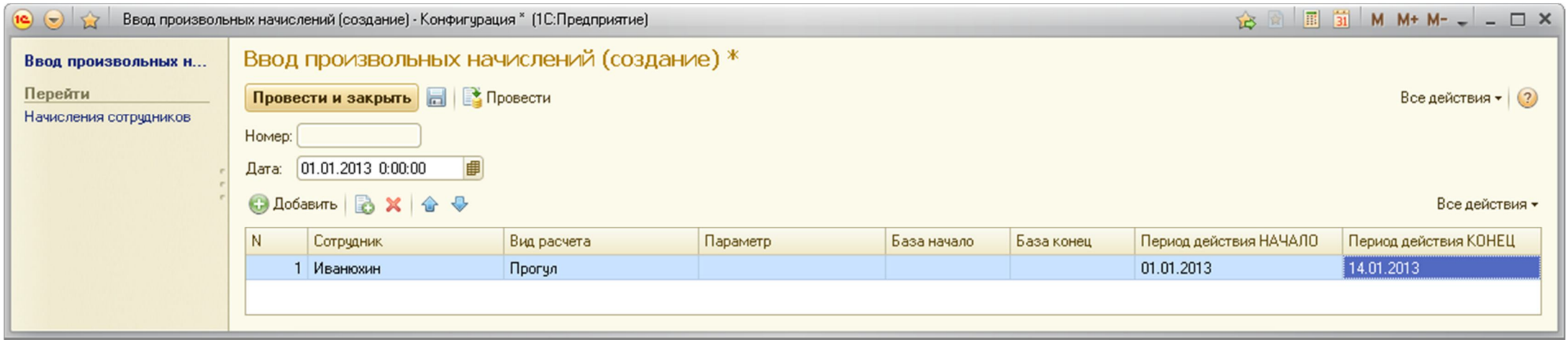
Проведем документ и посмотрим результат:



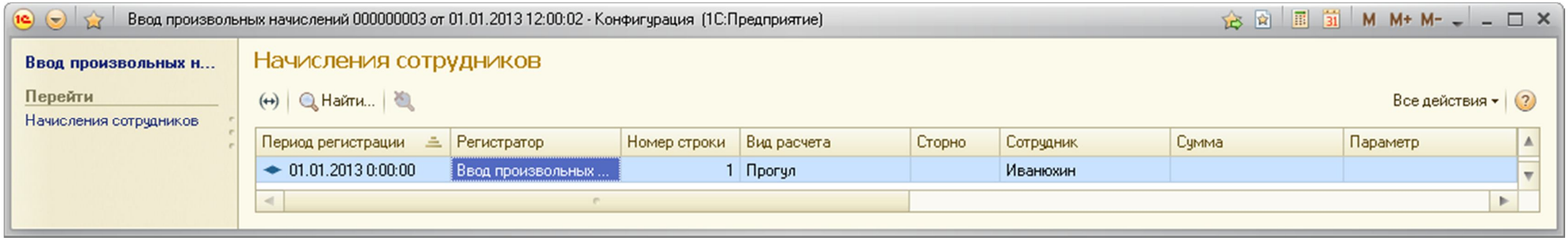


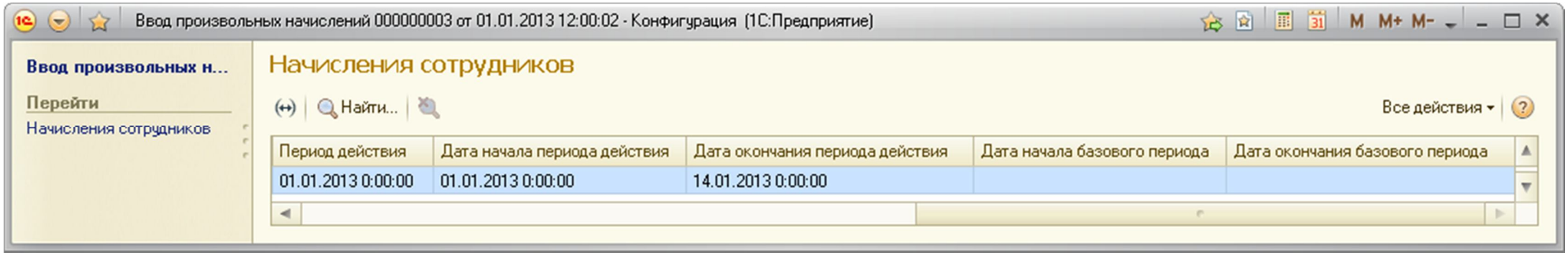
Вытеснять записи можно и другими документами, т.е. не обязательно, чтобы Оклад и Прогул были в одном документе:

Удалим Прогул в документе «Ввод произвольных начислений» № 000000001, проведем его, Оклад будет равен 100 000. Создадим еще один документ:

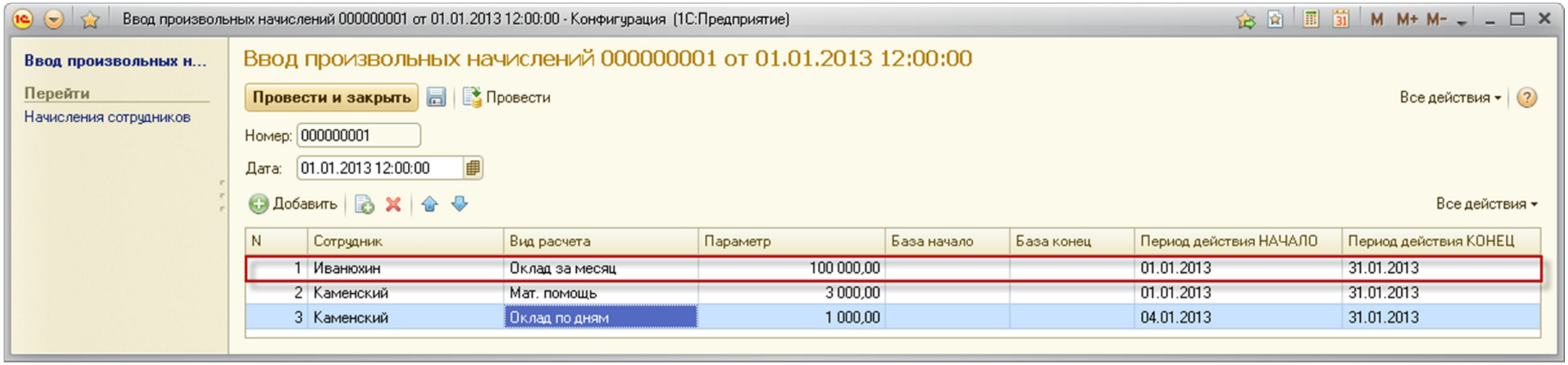


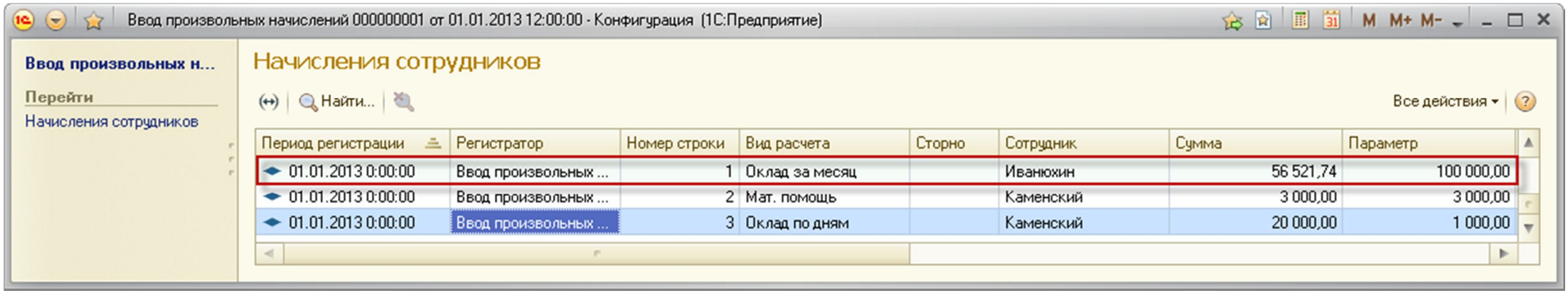
Проведем его и проверим результат:

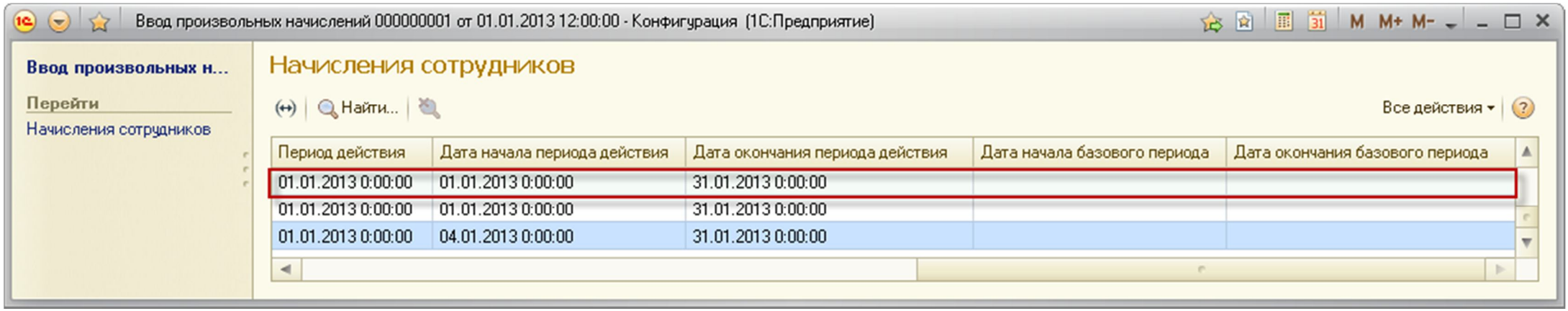




Видим, что сам Прогул не рассчитывается, т.е. Сумма = 0. А вот при перепроведении документа с Окладом за месяц (автоматизируем позже), Сумма будет рассчитана пропорционально:







Представим механизм вытеснений визуально с помощью отчета «Диаграмма Ганта» - будет показывать отношение записей на оси времени относительно друг друга:

Свойство	Значение
Имя	ДиаграммаГанта
Синоним	Диаграмма Ганта

Создадим форму отчета и добавим в нее реквизит:

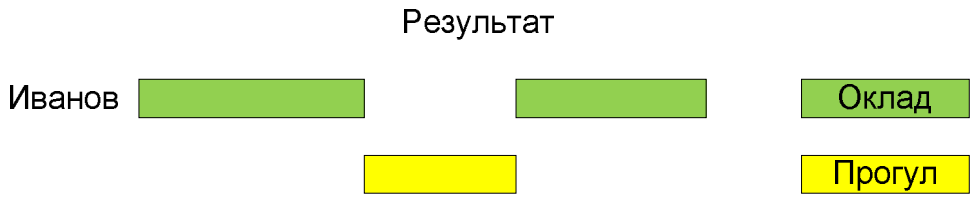
Свойство	Значение
Диаграмма	
Имя	Диаграмма
Синоним	Диаграмма
Тип	ДиаграммаГанта

и поместим на форму.

Любая диаграмма состоит из Точек и Серий.
В примере Точками будут Сотрудники, т.е. одному Сотруднику будет соответствовать раздел, состоящий из полосок, обозначающий протяженность по времени того или иного Вида начисления.
Серией будет Вид начисления (оклад, прогул, отпуск и т.д.). Серии будут обозначены разными цветами.
В диаграмму выводятся значения, которые находятся на пересечении Точек и Серий.

Чтобы вывести диаграмму, надо получить таблицу вида:

Сотрудник	ВидРасчета	ДатаНачала	ДатаОкончания
Иванов	Оклад	01.01.2013	08.01.2013
Иванов	Оклад	15.01.2013	31.01.2013
Иванов	Прогул	09.01.2013	14.01.2013



Добавим кнопку (команду формы) «Сформировать отчет»:

Свойство	Значение
Имя	СформироватьОтчет
Синоним	Сформировать отчет

Получим таблицу на Сервере и выведем в диаграмму.

«Текст – Конструктор запроса с обработкой результата» - для быстрого создания шаблона запроса с кодом обработки результата. Результат можно вывести в диаграмму вида Гистограмма (столбики), а нам нужна диаграмма Ганта.

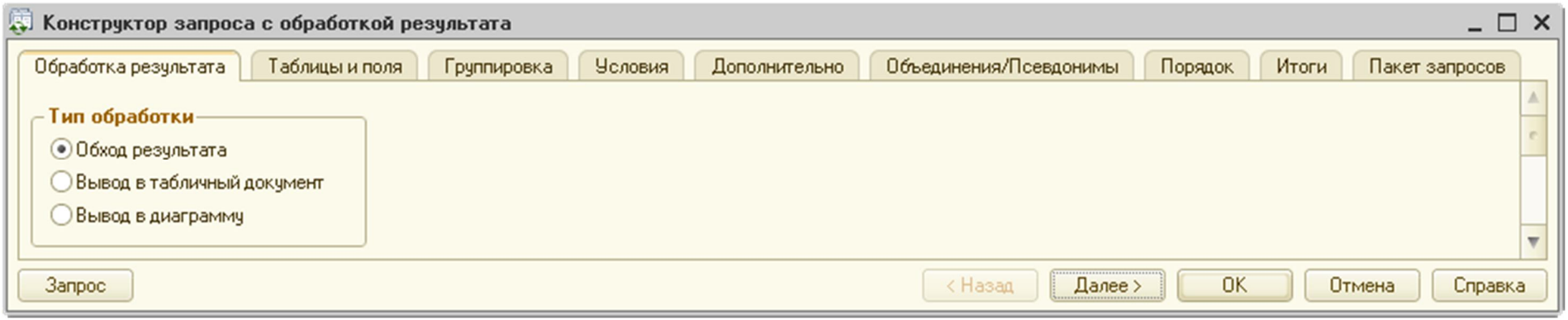


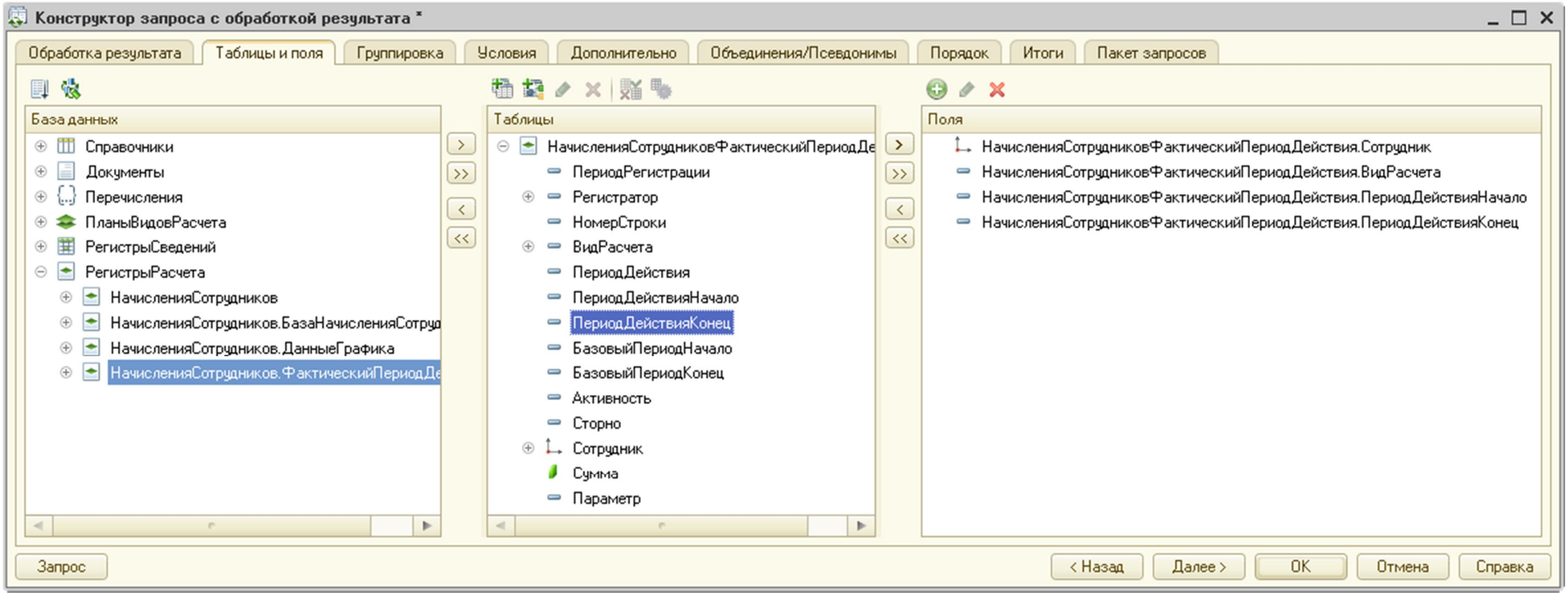
Таблица «НачисленияСотрудников.ФактическийПериодДействия» аналогична реальной таблице «НачисленияСотрудников», но в таблице «НачисленияСотрудников.ФактическийПериодДействия» поля:

- ПериодДействияНачало.
- ПериодДействияКонец.

отображают то, что получилось после вытеснения. А в реальной таблице «НачисленияСотрудников» эти поля отображают то, что было в оригинале.

Например, в реальной таблице будет информация о периоде действия оклада с 01 по 31 число, а в таблице «ФактическийПериодДействия» будет несколько записей (результат вытеснения):

- с 1 по 6.
- с 8 по 15.
- с 17 по 31.



Результат работы конструктора:

```
&НаСервере
Процедура СформироватьОтчетНаСервере(ДГ)

	//{{КОНСТРУКТОР_ЗАПРОСА_С_ОБРАБОТКОЙ_РЕЗУЛЬТАТА
// Данный фрагмент построен конструктором.
// При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!

Запрос = Новый Запрос;
Запрос.Текст =
	"ВЫБРАТЬ
	|	НачисленияСотрудниковФактическийПериодДействия.Сотрудник,
```

```
            |    НачисленияСотрудниковФактическийПериодДействия.ВидРасчета,
            |    НачисленияСотрудниковФактическийПериодДействия.ПериодДействияНачало,
            |    НачисленияСотрудниковФактическийПериодДействия.ПериодДействияКонец
            |ИЗ
            |    РегистрРасчета.НачисленияСотрудников.ФактическийПериодДействия КАК НачисленияСотрудниковФактиче-
скийПериодДействия";

      Результат = Запрос.Выполнить();

      ВыборкаДетальныеЗаписи = Результат.Выбрать();

      Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
            // Вставить обработку выборки ВыборкаДетальныеЗаписи
      КонецЦикла;

      //}}КОНСТРУКТОР_ЗАПРОСА_С_ОБРАБОТКОЙ_РЕЗУЛЬТАТА

КонецПроцедуры
```

До цикла сначала выключим обновление диаграммы, очистим, а после цикла включим обновление.

Итого код:

```
&НаКлиенте
Процедура СформироватьОтчет(Команда)
      СформироватьОтчетНаСервере(Диаграмма);
КонецПроцедуры

&НаСервереБезКонтекста
Процедура СформироватьОтчетНаСервере(ДГ)

      Запрос = Новый Запрос;
      Запрос.Текст =
      "ВЫБРАТЬ
      |    НачисленияСотрудниковФактическийПериодДействия.Сотрудник,
```

```
|    НачисленияСотрудниковФактическийПериодДействия.ВидРасчета,
|    НачисленияСотрудниковФактическийПериодДействия.ПериодДействияНачало,
|    НачисленияСотрудниковФактическийПериодДействия.ПериодДействияКонец
|ИЗ
|    РегистрРасчета.НачисленияСотрудников.ФактическийПериодДействия
         КАК НачисленияСотрудниковФактическийПериодДействия";

Результат = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = Результат.Выбрать();

ДГ.Обновление = Ложь;
ДГ.Очистить();

Пока ВыборкаДетальныеЗаписи.Следующий() Цикл

     Точка    = ДГ.УстановитьТочку(ВыборкаДетальныеЗаписи.Сотрудник);
     Серия    = ДГ.УстановитьСерию(ВыборкаДетальныеЗаписи.ВидРасчета);
     Значение = ДГ.ПолучитьЗначение(Точка, Серия);
     Интервал = Значение.Добавить();

     Интервал.Начало = ВыборкаДетальныеЗаписи.ПериодДействияНачало;
     Интервал.Конец  = ВыборкаДетальныеЗаписи.ПериодДействияКонец;

КонецЦикла;

ДГ.Обновление = Истина;
```

КонецПроцедуры

Сотрудник в таблице может дублироваться, т.е. если в точку просто добавить Сотрудника, то на каждое повторение будет создаваться новое значение диаграммы. Чтобы получить существующую Запись, есть метод «УстановитьТочку» Диаграммы Ганта, в который передается значение, по которому устанавливается точка. Аналогично с Серией

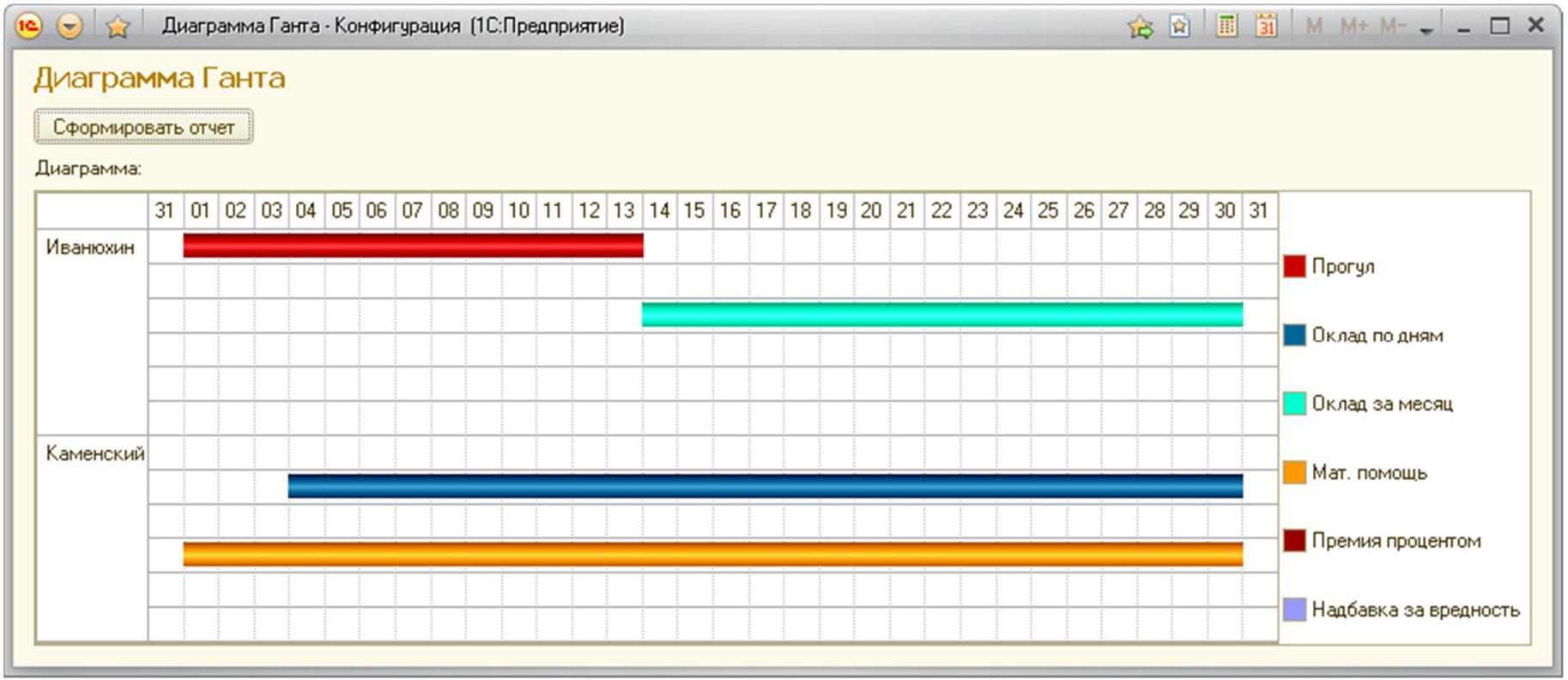
На пересечении Точки и Серии получаем Значение диаграммы

```
Значение = ДГ.ПолучитьЗначение(Точка, Серия);
```

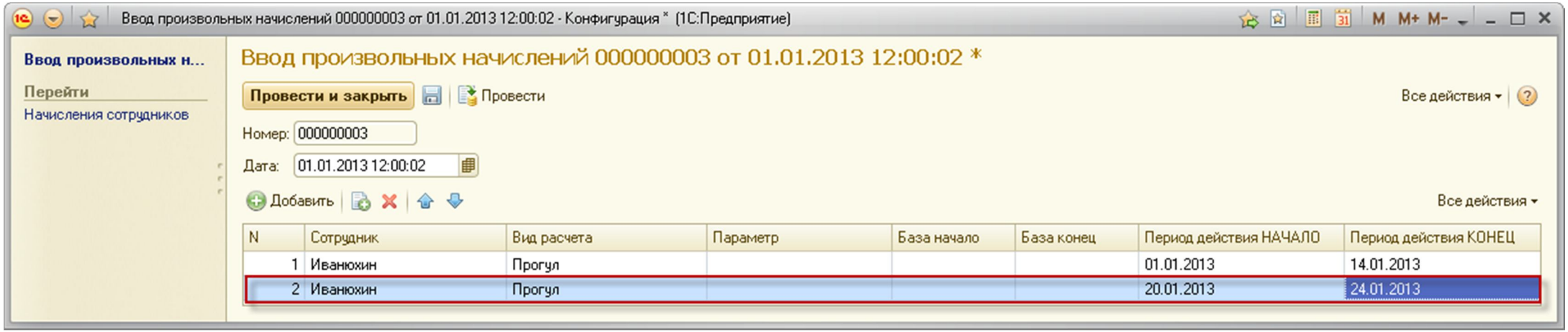
Для Интервала надо установить границы:

```
Интервал.Начало = ВыборкаДетальныеЗаписи.ПериодДействияНачало;
Интервал.Конец  = ВыборкаДетальныеЗаписи.ПериодДействияКонец;
```

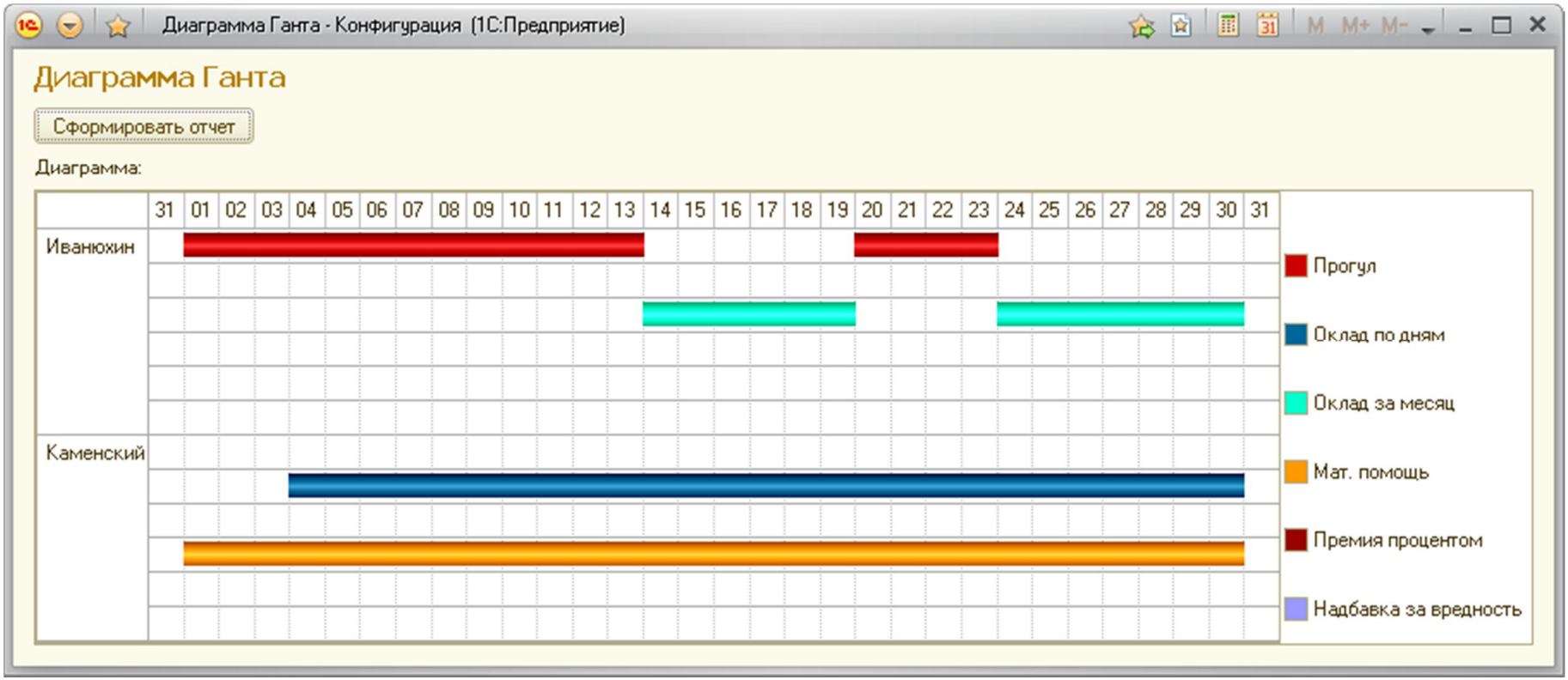
В режиме исполнения:



Добавим Иванюхину еще один Прогул:



Сформируем отчет:



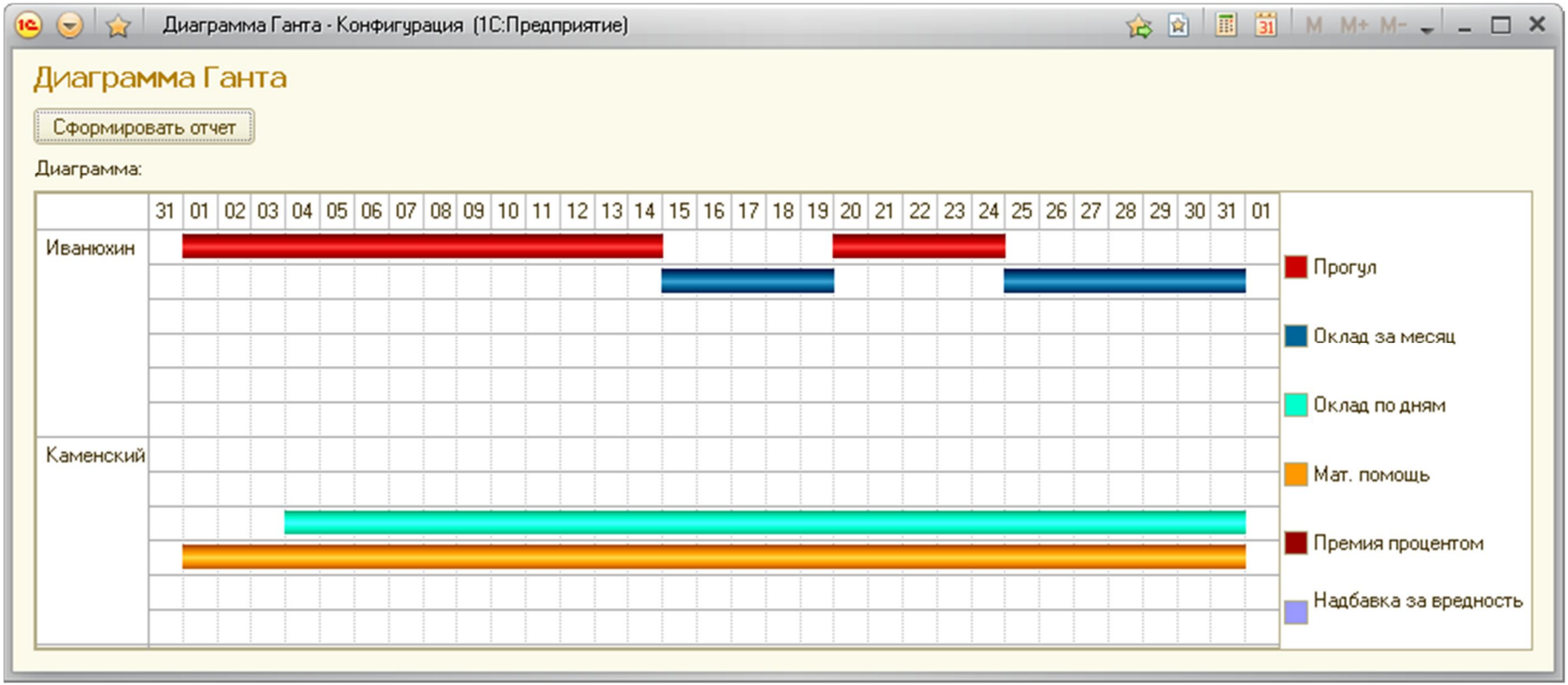
Записи в Регистр расчета можно вводить только внутри одного периода действия, т.е. если сотрудник в отпуске с середины прошлого месяца до середины текущего, то надо вводить 2 записи.

Ошибка в диаграмме: например, ввели Прогул по 24-е, а в диаграмме отобразилось по 23-е. Дело в дате, т.е. Прогул считается по 24.01.2013. 00:00:00, а с 24.01.2013. 00:00:01 уже считается, что сотрудник снова на работе. Поэтому надо всегда делать запись по конец дня.

Поправим сразу для всех записей: регистр расчета «Начисления сотрудников» - Модуль набора записей:

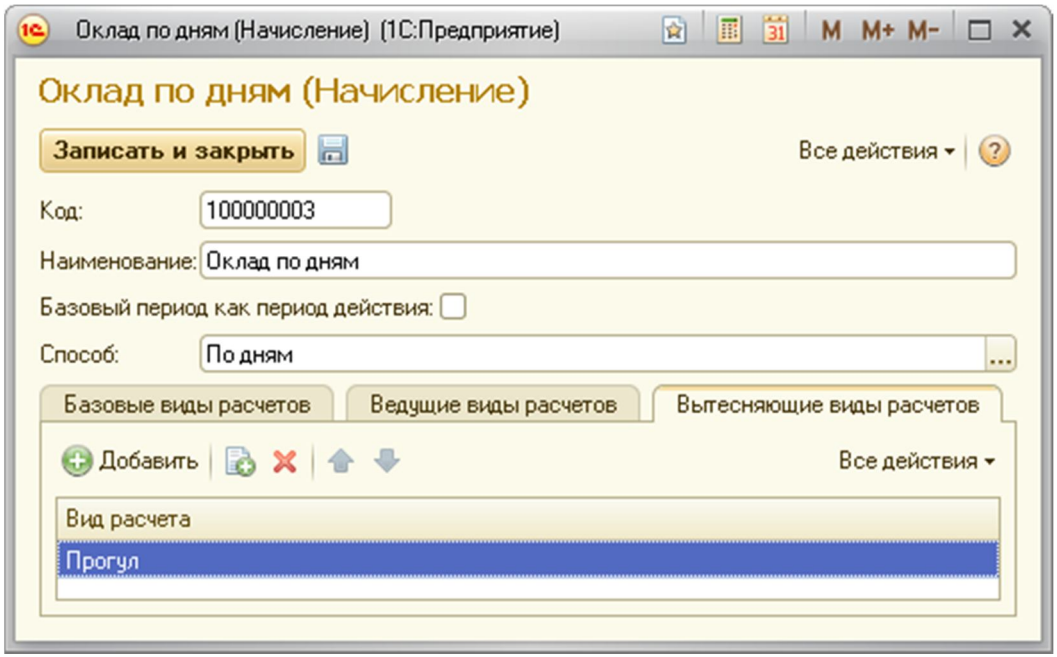
```
Процедура ПередЗаписью(Отказ, Замещение, ТолькоЗапись, ЗаписьФактическогоПериодаДействия, ЗаписьПерерасчетов)
     Для Каждого Запись Из ЭтотОбъект Цикл
          Запись.ПериодДействияКонец = КонецДня(Запись.ПериодДействияКонец);
     КонецЦикла;
КонецПроцедуры
```

В режиме исполнения:
Перепроведем документы «Ввод произвольных начислений» №№ 000000001, 000000003 и сформируем отчет:

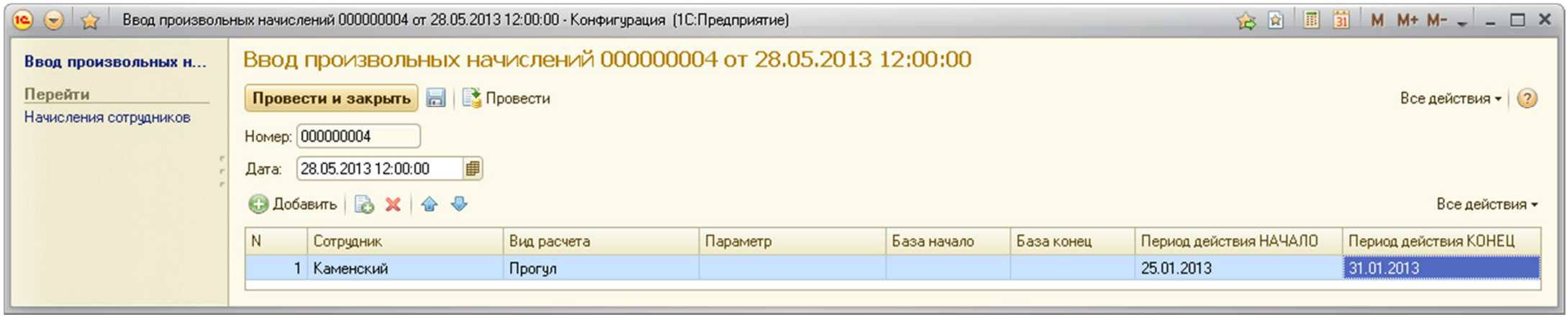


Введем еще одно вытеснение:

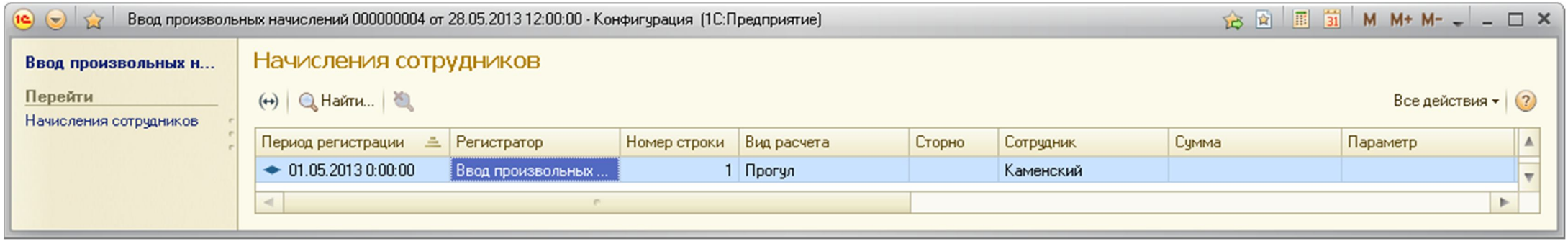
Пометим на удаление документ «Ввод произвольных начислений» № 000000002.
Укажем, что вид начисления «Оклад по дням» вытесняется видом начисления «Прогул»:

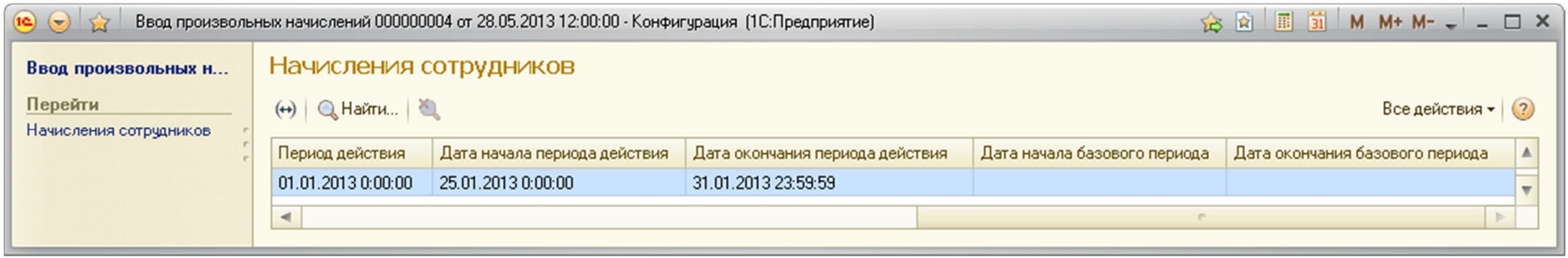


Введем новый документ «Ввод произвольных начислений»:



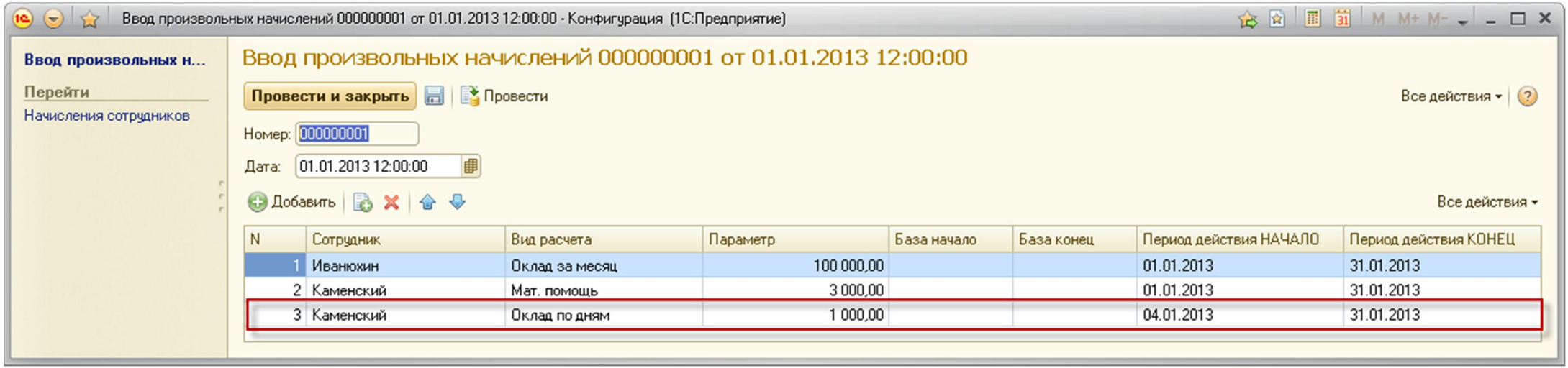
Результат:

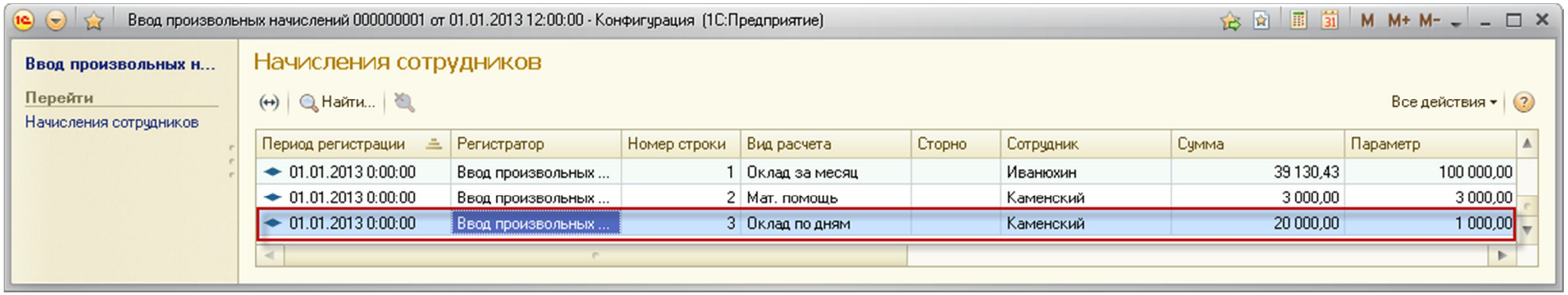


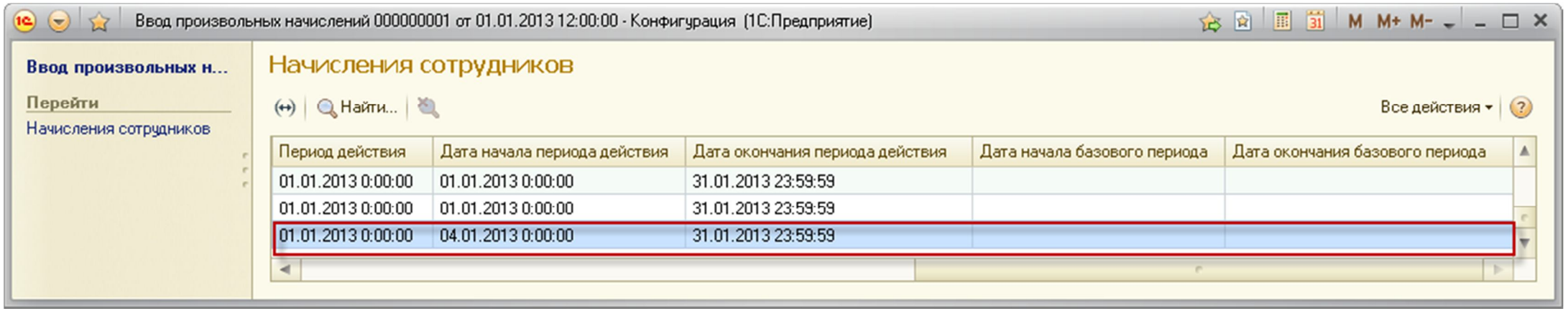


Т.е. Прогул в системе зарегистрировался.

Перепроведем документ «Ввод произвольных начислений» № 000000001, в котором есть вытесненные записи и посмотрим результат:

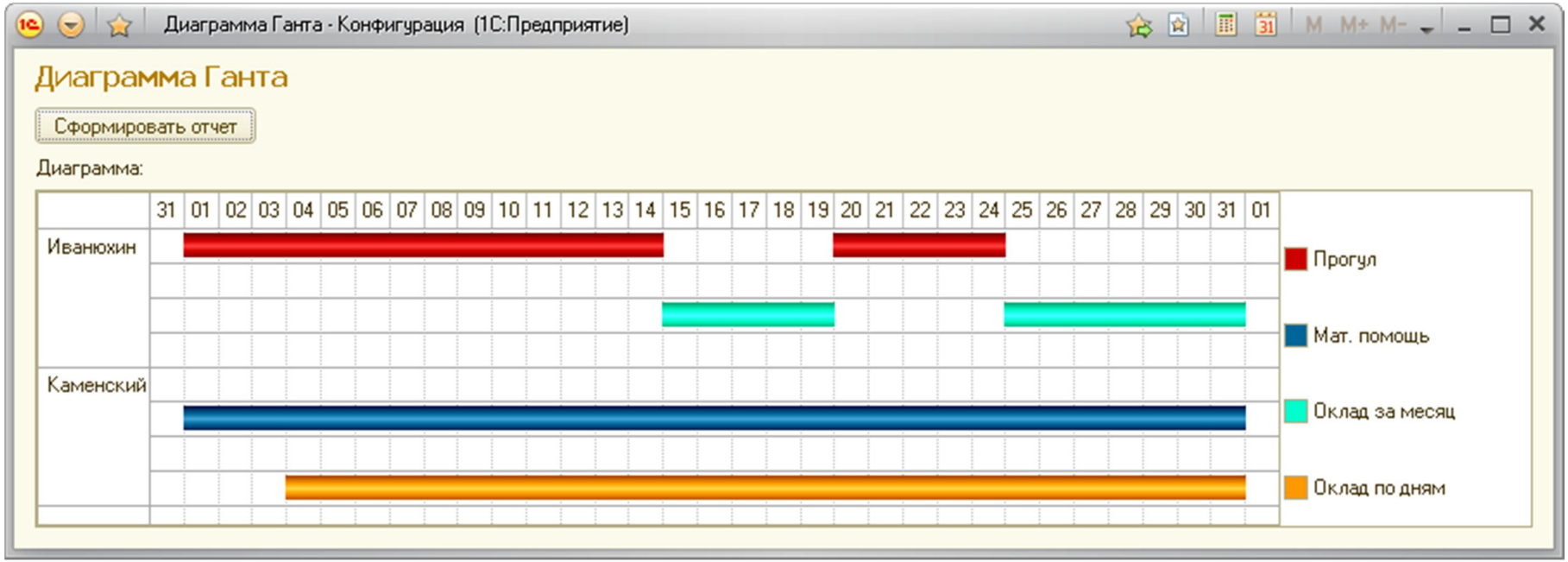






Оклад по дням у Каменского меньше не стал.

В отчете тоже не видно никаких изменений:



Запись не вытиснилась из-за даты периода регистрации. Мы в мае, т.е. задним числом вводим информацию о Прогуле, совершенном в январе. Данные за январь уже рассчитаны, начислены налоги, выдана зарплата и т.д., поэтому исправление задним числом недопустимо – система не позволяет произвести вытеснение.
Запись Оклад и запись Прогул конкурируют за период действия, но Оклад имеет приоритет, поэтому Прогул не может получить данные по фактическому периоду действия.

Надо принудительно вытеснить оклад, но вытеснять нужно в мае – сформируем запись СТОРНО – принудительное вытеснение записи прошлого периода в текущем.
Сформируем запись Оклада повторно, но с признаком СТОРНО, в итоге 2 записи Оклада дадут фактический период действия, равный нулю.

Чтобы сформировать СТОРНО записи, надо при вводе данных в регистр расчета «НачисленияСотрудников» получить таблицу с записями СТОРНО.
Документ «ВводПроизвольныхНачислений» - модуль объекта:

Обратимся к движению «НачисленияСотрудников» и прочитаем записи Дополнений, которые сформирует регистр. Система знает какие записи надо вытеснить, за какой период, но самостоятельно вытеснение не производит, а предлагает информацию о вытеснении, чтобы на ее основе мы могли предпринять какие-либо действия.

После формирования записей в регистре, но до записи, получим записи Дополнения:

```
Дополнения = Движения.НачисленияСотрудников.ПолучитьДополнение();
```

Посмотрим получившиеся Дополнения – в режиме исполнения проведем документ «Ввод произвольных начислений» № 000000004 с введенным прогулом по Каменскому. В таблице значений Дополнения получаем запись, которая не дает рассчитаться Прогулу

Новый2 *

	1	2	3	4	5	6
1	Индекс	Значение элемента	Тип элемента	ПериодРегистрации	Регистратор	НомерСтроки
2	0	СтрокаТаблицыЗначений	СтрокаТаблицыЗначений	01.01.2013 0:00:00	Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00	3
3						

Новый2 *

	7	8	9	10	11	12	13	14
1	ВидРасчета	ПериодДействия	ПериодДействияНачало	ПериодДействияКонец	БазовыйПериодНачало	БазовыйПериодКонец	Активность	Сторно
2	Оклад по дням	01.01.2013 0:00:00	04.01.2013 0:00:00	31.01.2013 23:59:59	01.01.0001 0:00:00	01.01.0001 0:00:00	Истина	Ложь
3								

Новый2 *

	15	16	17	18	19	20
1	Сотрудник	Сумма	Параметр	ПериодРегистрацииСторно	ПериодДействияНачалоСторно	ПериодДействияКонецСторно
2	Каменский	20 000	1 000	01.05.2013 0:00:00	25.01.2013 0:00:00	31.01.2013 0:00:00
3						

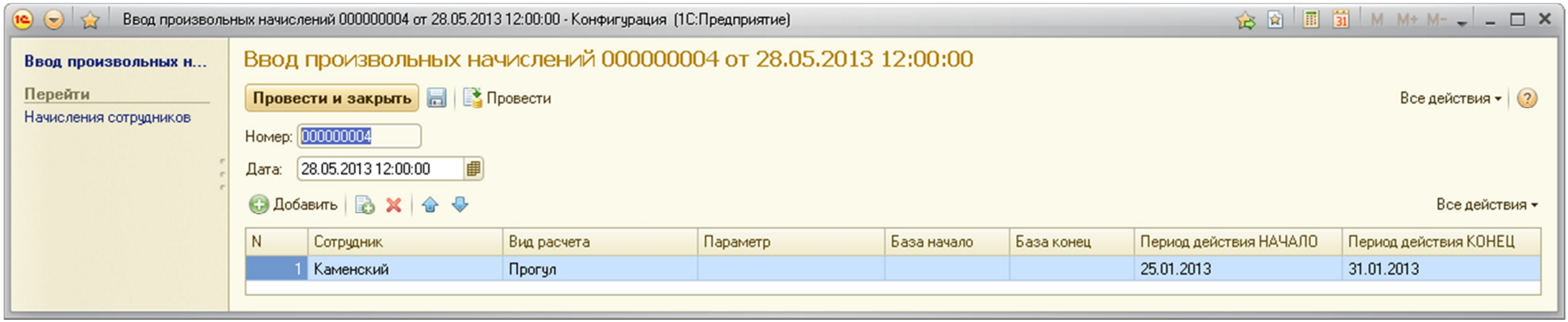
Информация о периоде вытеснения (последние 3 колонки):

ПериодРегистрацииСторно – когда необходимо сформировать запись СТОРНО.
ПериодДействияНачалоСторно и ПериодДействияКонецСторно – пересекающийся, конфликтный период записи, которая вводится, с записью, которая уже есть.
Создадим новую запись в регистре. Переберем таблицу Дополнений, запишем движения. Главное, установить флаг, что это Движение СТОРНО:

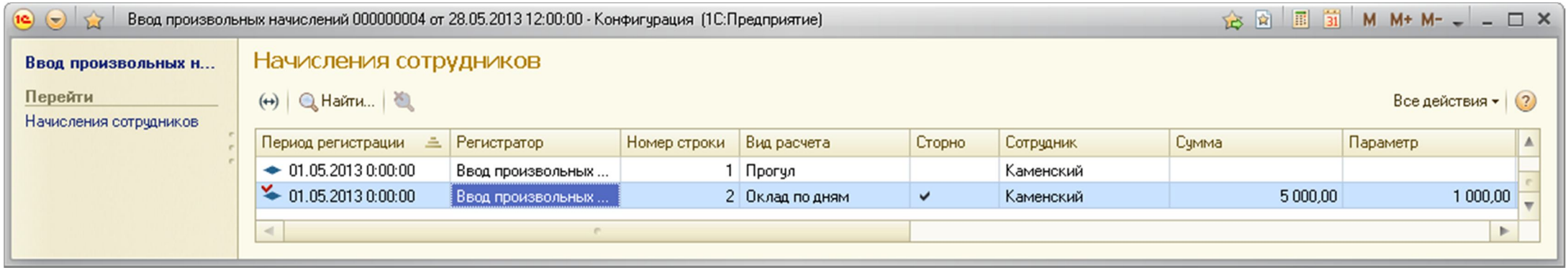
```
Для Каждого Стр Из Дополнения Цикл
      Движение = Движения.НачисленияСотрудников.Добавить();
      ЗаполнитьЗначенияСвойств(Движение, Стр);

      Движение.ПериодРегистрации = Стр.ПериодРегистрацииСторно;
      Движение.ПериодДействияНачало = Стр.ПериодДействияНачалоСторно;
      Движение.ПериодДействияКонец = Стр.ПериодДействияКонецСторно;
      Движение.Сторно = Истина
КонецЦикла;
```

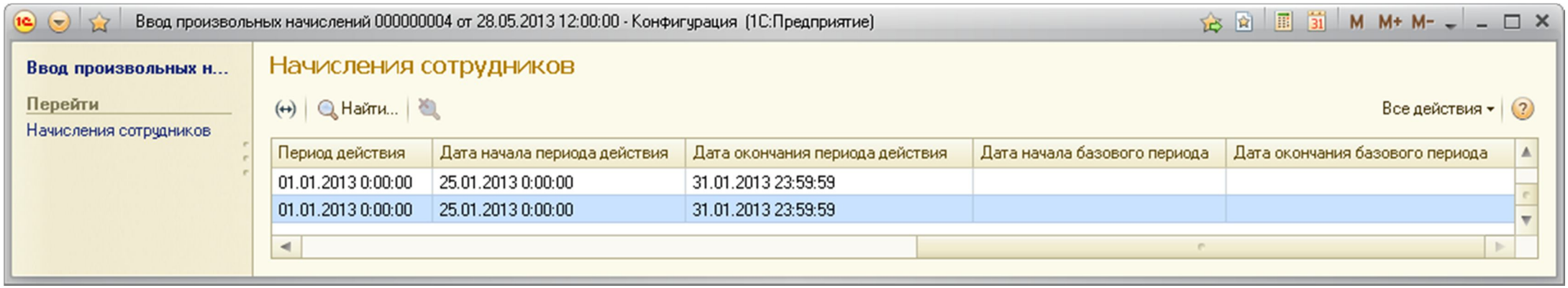
В режиме исполнения:

Перепроведем документ «Ввод произвольных начислений» № 000000004:



Результат:





Появилась вторая запись – по Окладу, с признаком СТОРНО. Начислено 5000, но это сумма, которую надо вычесть из оклада сотрудника, поэтому, при расчете записей, все записи СТОРНО будем записывать со знаком «Минус»:
Перейдем в процедуру расчета (Общий модуль «Расчеты» - процедура «Рассчитать») и сделаем проверку:

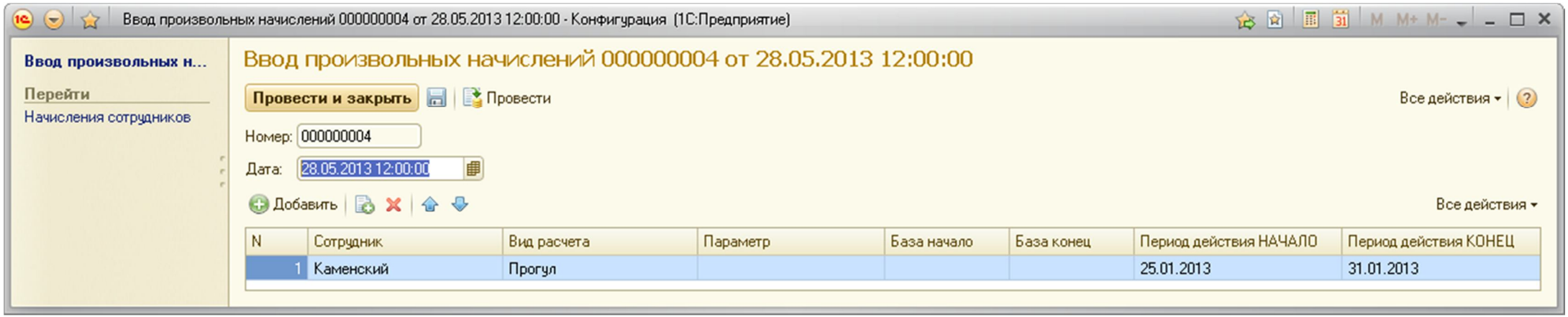
```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");
```

```
СписокРесурсов = Новый Массив(1);
СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

ОтборПоРегистратору = Новый Структура;
ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
ОтборПоРегистратору,
СписокРесурсов,
ОтборПоИзмерениям);

ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ФактическийПериодДействия);

ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ПериодДействия);

Для Каждого Запись Из НаборЗаписей Цикл
     //Индекс = Запись.НомерСтроки - 1;
     Индекс = НаборЗаписей.Индекс(Запись);
     База = ТЗ_База[Индекс].Сумма;
     СпособРасчета = Запись.ВидРасчета.Способ;
     ФактДней = ТЗ_ФактДней[Индекс].Признак;
     НормаДней = ТЗ_НормаДней[Индекс].Признак;
     Параметр = Запись.Параметр;

     Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
          Запись.Сумма = База * Параметр / 100;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
          Запись.Сумма = Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
          Запись.Сумма = ФактДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
```

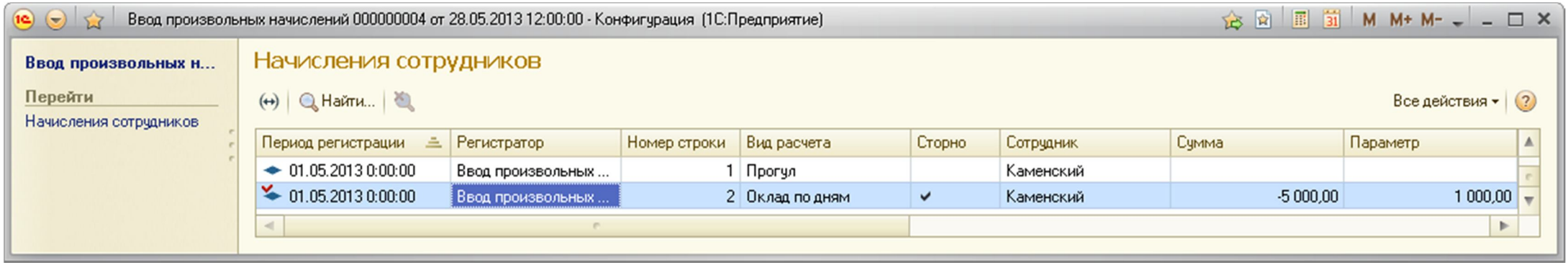
```
            Запись.Сумма = ФактДней / НормаДней * Параметр;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

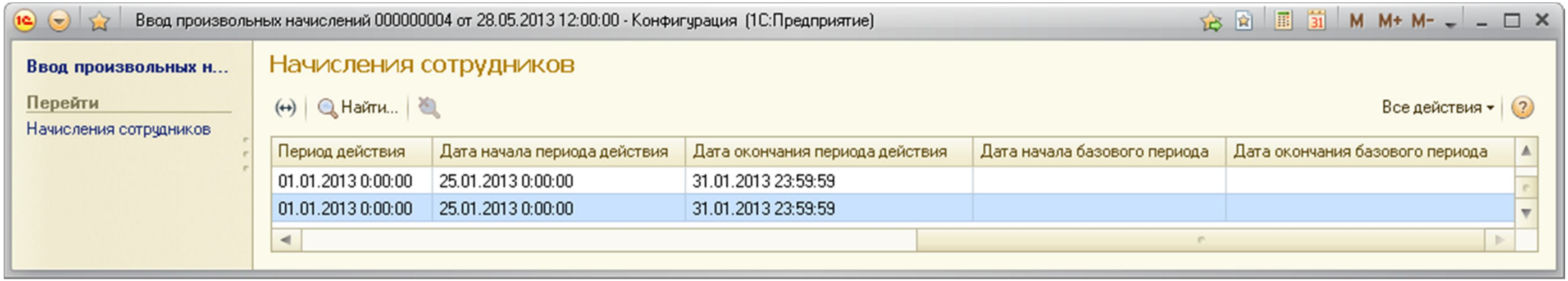
В режиме исполнения:

Перепроведем документ «Ввод произвольных начислений» № 000000004:

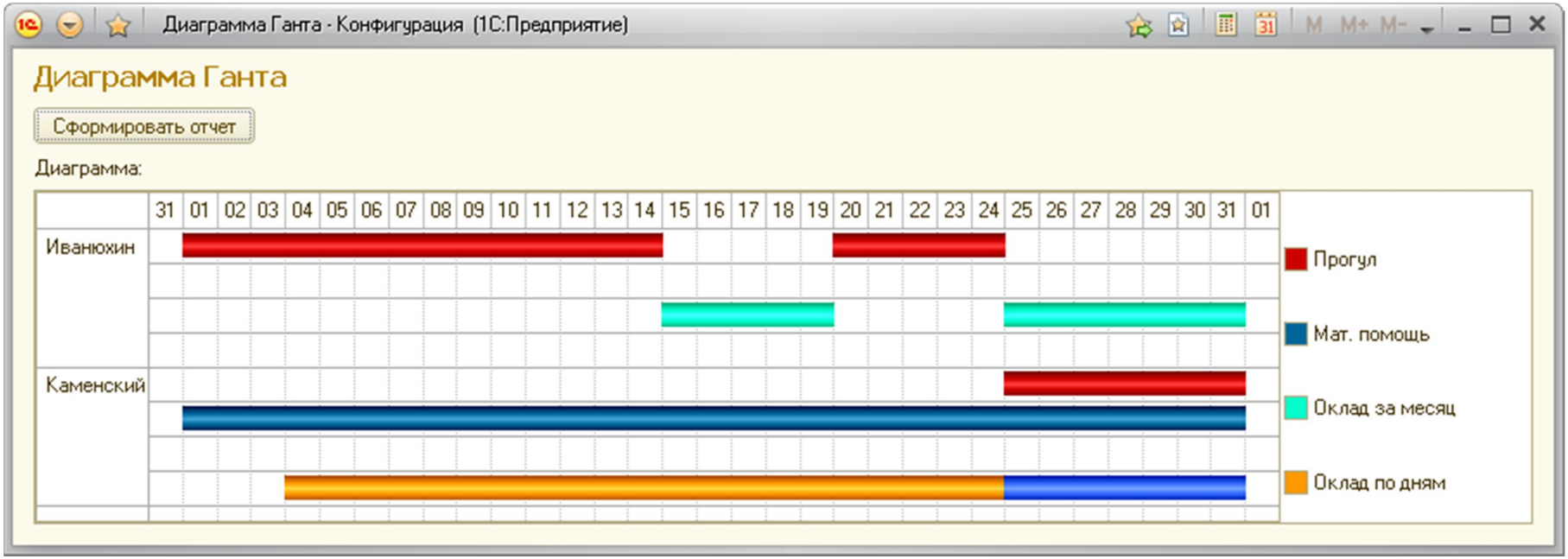


Результат:


Ввод произвольных начислений 000000004 от 28.05.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации
Регистратор
Номер строки
Вид расчета
Сторно
Сотрудник
Сумма
Параметр
01.05.2013 0:00:00
Ввод произвольных ...
1
Прогул
Каменский
01.05.2013 0:00:00
Ввод произвольных ...
2
Оклад по дням
Каменский
-5 000,00
1 000,00


Ввод произвольных начислений 000000004 от 28.05.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия
Дата начала периода действия
Дата окончания периода действия
Дата начала базового периода
Дата окончания базового периода
01.01.2013 0:00:00
25.01.2013 0:00:00
31.01.2013 23:59:59
01.01.2013 0:00:00
25.01.2013 0:00:00
31.01.2013 23:59:59

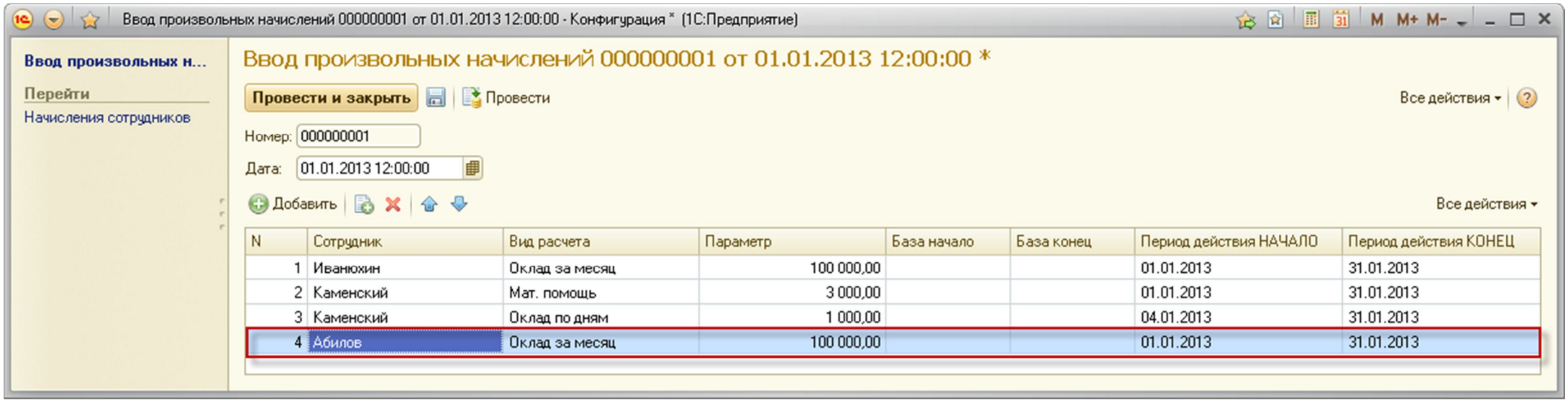
Посмотрим СТОРНО-запись в диаграмме Ганта:



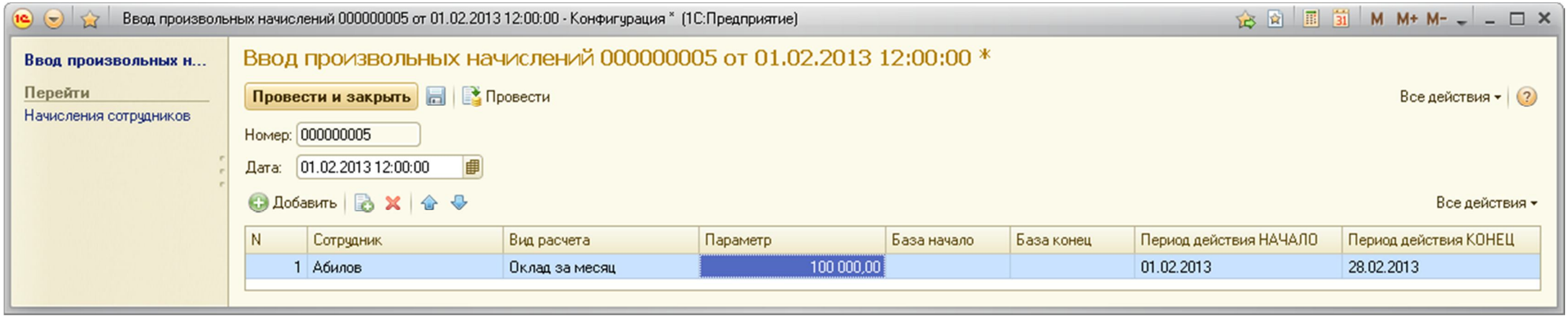
Занятие 3.28

Разработаем вид начисления типа «Отпуск». Правило для расчета – отпуск начисляется по среднедневному заработку работника за предыдущие 3 месяца.
Добавим тестовые данные:

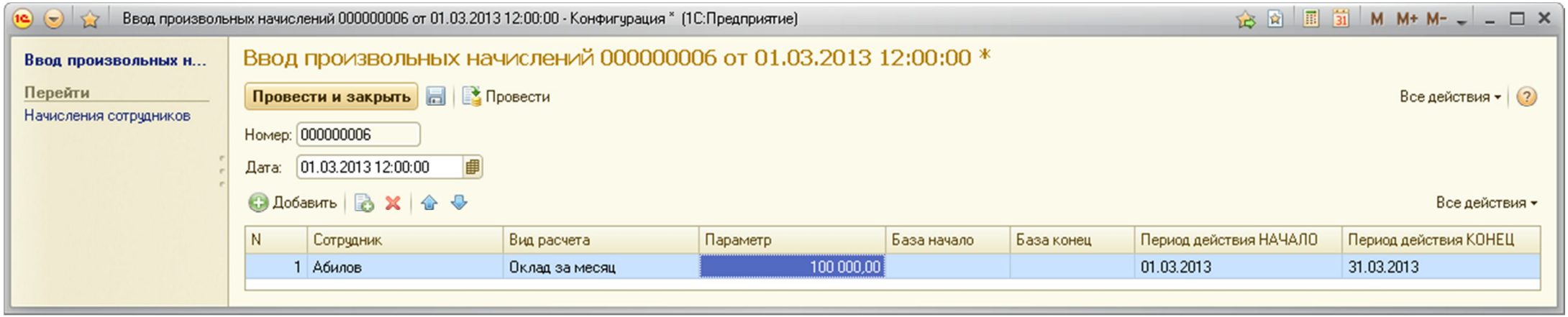
Новый сотрудник Абилов полностью отработал январь:



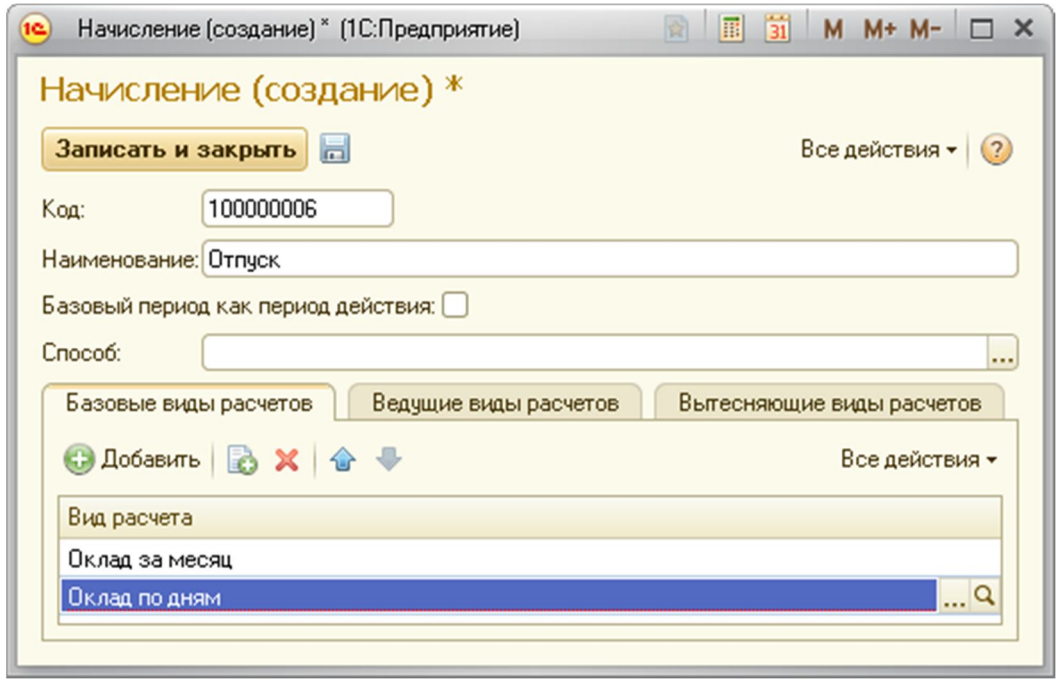
полностью отработал февраль (отражаем это в документе за февраль):



полностью отработал март (отражаем это в документе за март):



В апреле Абилов может уйти в отпуск:
Создадим новый Вид начисления – «Отпуск»:

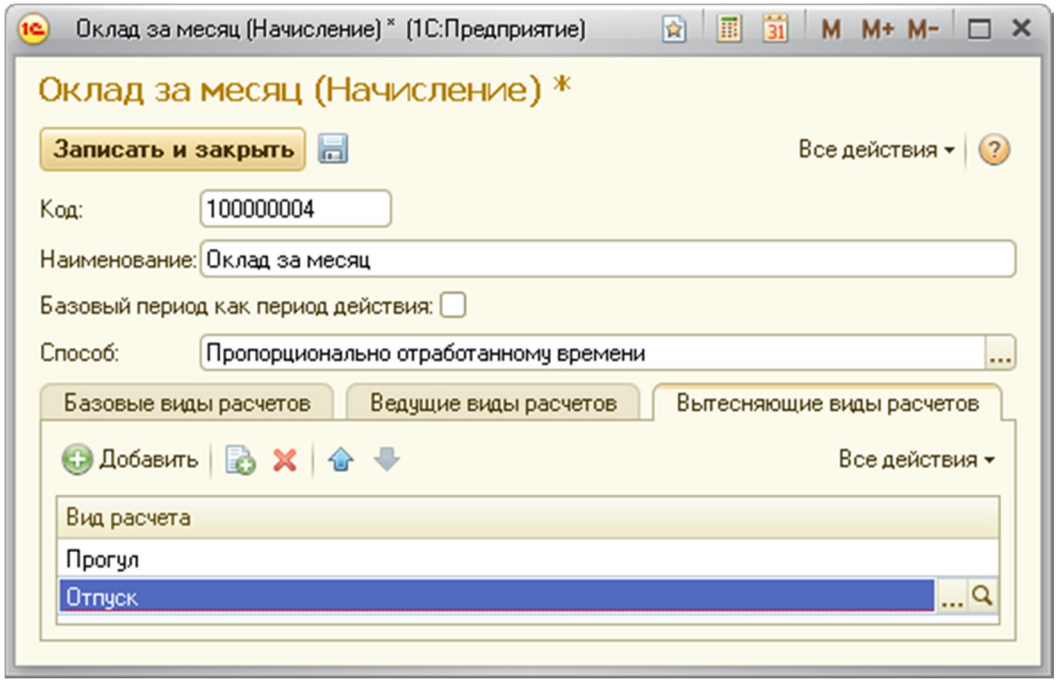


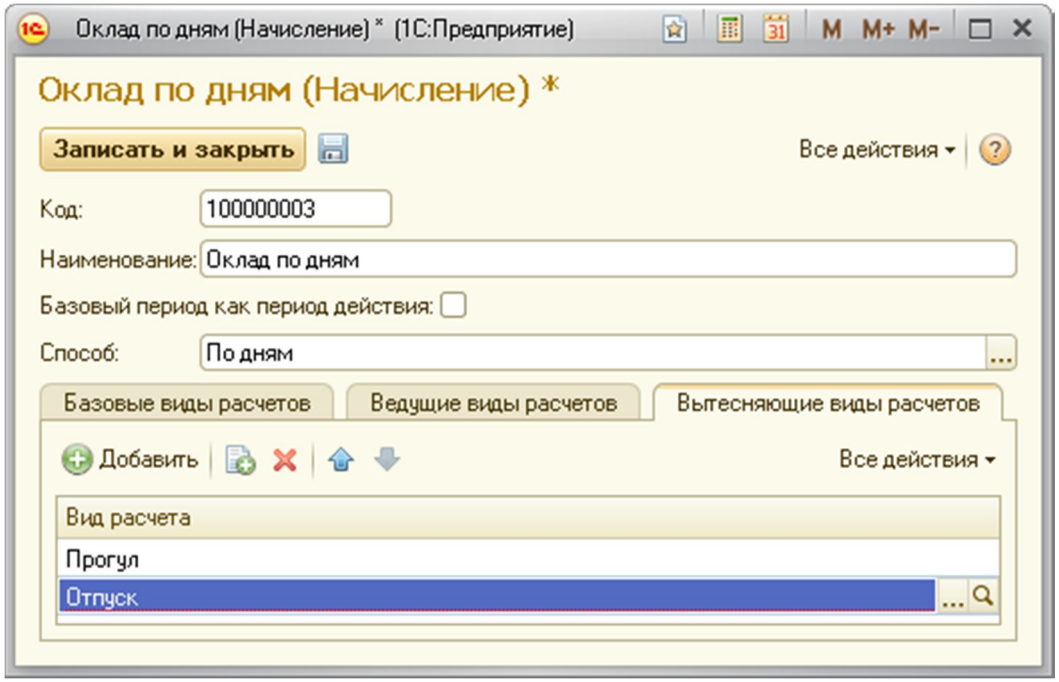
«Способ» - мы пока не создали необходимый способ расчета, поэтому это поле оставим пока пустым.
Базой для расчета среднедневного заработка являются:

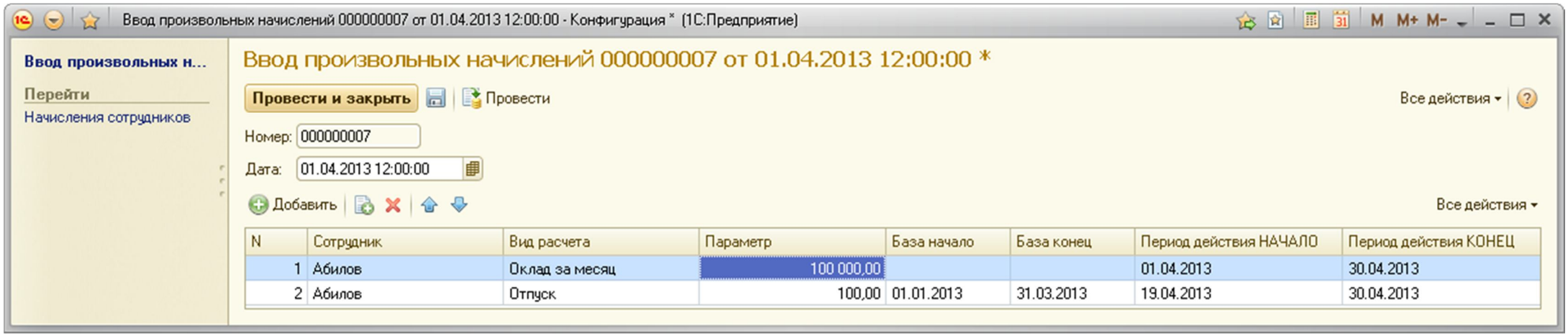
- Оклад за месяц.
- Оклад по дням.

Отпуск, в свою очередь, является вытесняющим для:

- Оклад за месяц.
- Оклад по дням.







Параметр – будем указывать процент расчета среднедневного заработка. В примере 100%.
ПериодДействияНАЧАЛО/ПериодДействияКОНЕЦ – период отпуска.
База начало/База конец – база для расчета отпуска.

Среднедневной заработок = количество заработанного за период / количество дней в периоде.

Создадим новый способ расчета – «По среднему»:

Свойство	Значение
Имя	ПоСреднему
Синоним	По среднему

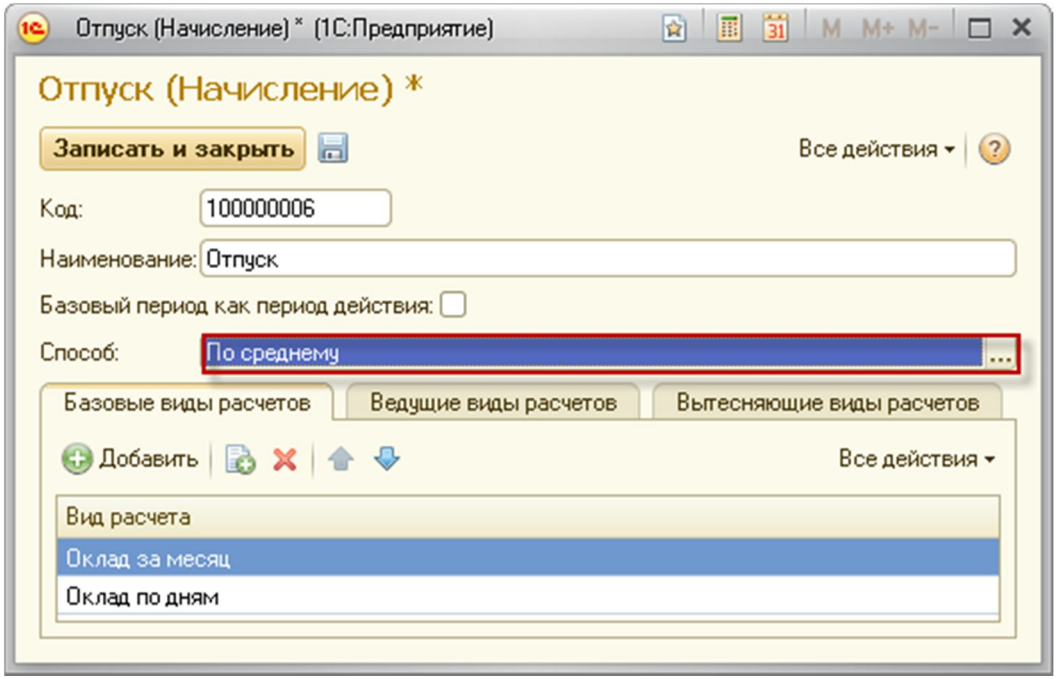
Опишем условия расчета (Общий модуль «Расчеты» - процедура «Рассчитать»)

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);
```

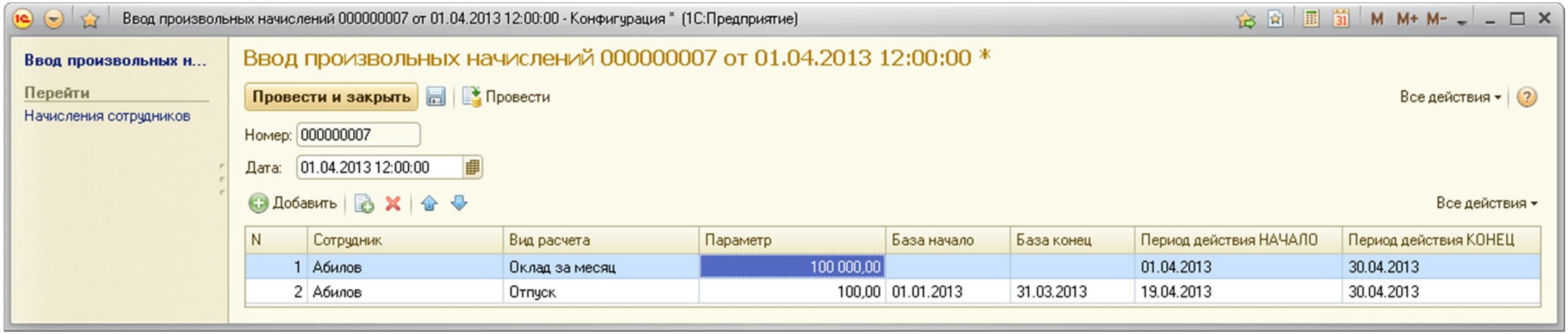
```
ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ФактическийПериодДействия);

ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ПериодДействия);

ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.БазовыйПериод);

Для Каждого Запись Из НаборЗаписей Цикл
     //Индекс = Запись.НомерСтроки - 1;
     Индекс = НаборЗаписей.Индекс(Запись);
     База = ТЗ_База[Индекс].Сумма;
     СпособРасчета = Запись.ВидРасчета.Способ;
     ФактДней = ТЗ_ФактДней[Индекс].Признак;
     НормаДней = ТЗ_НормаДней[Индекс].Признак;
     БазаДней = ТЗ_БазаДней[Индекс].Признак;
     Параметр = Запись.Параметр;

     Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
          Запись.Сумма = База * Параметр / 100;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
          Запись.Сумма = Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
          Запись.Сумма = ФактДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
          Запись.Сумма = ФактДней / НормаДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
          Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
     КонецЕсли;

     Если Запись.Сторно Тогда
          Запись.Сумма = - Запись.Сумма;
```

```
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

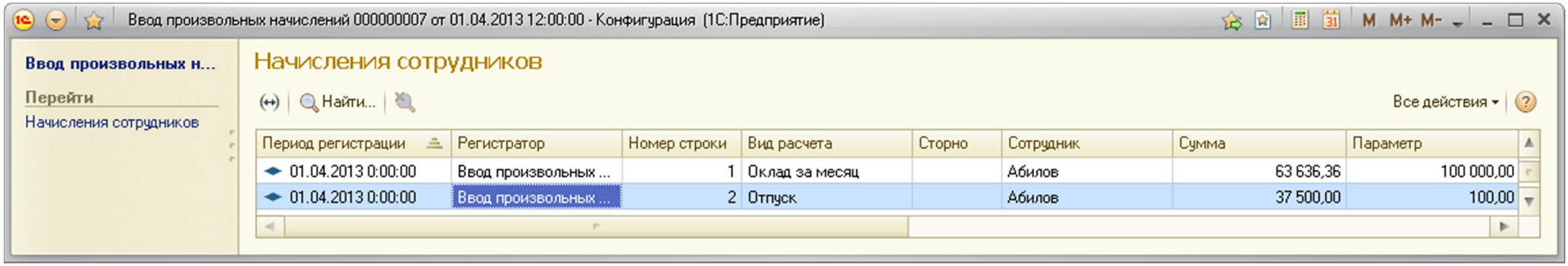
В Начислении «Отпуск» укажем Способ расчета:

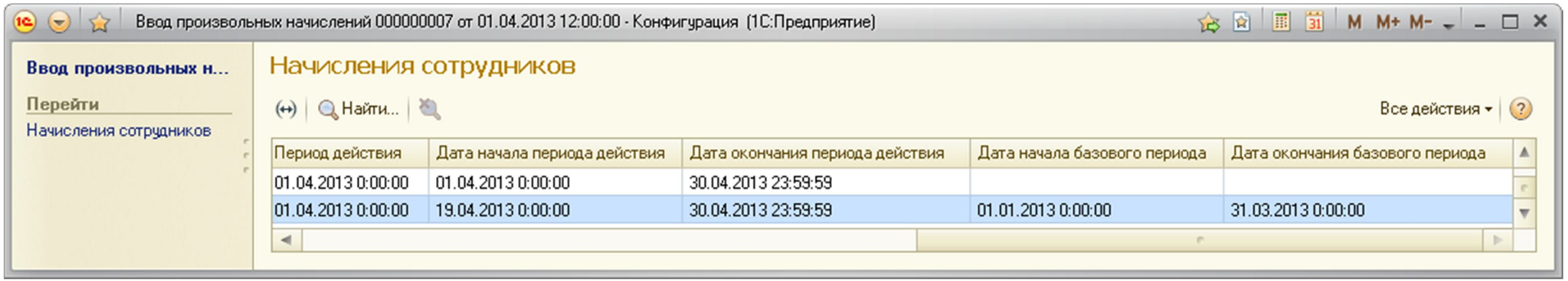


Перепроведем документ «Ввод произвольных начислений» № 000000007:

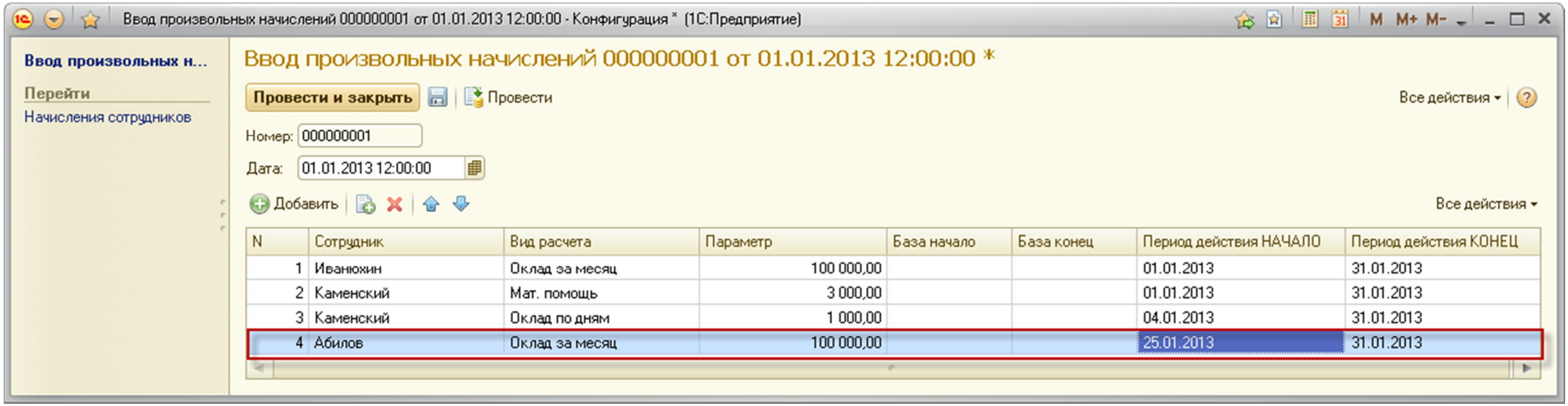


И проверим результат:

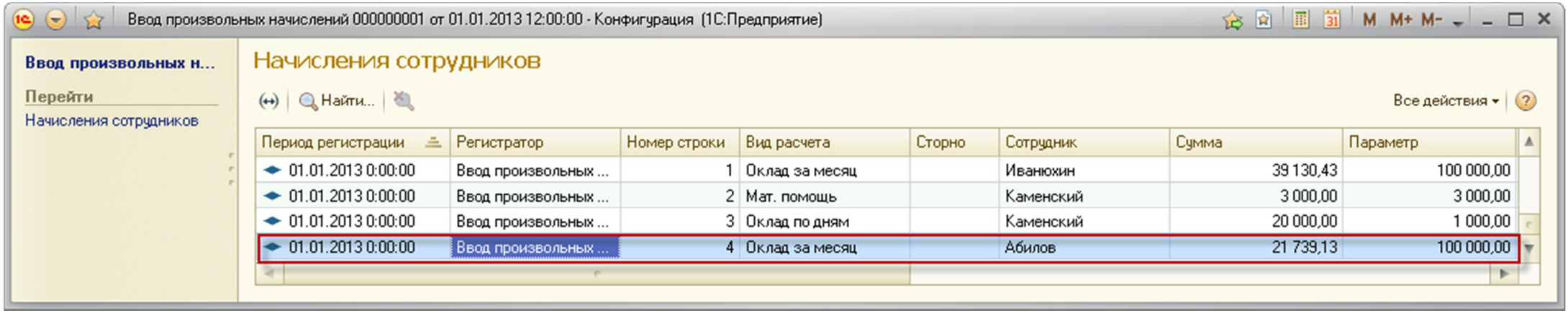


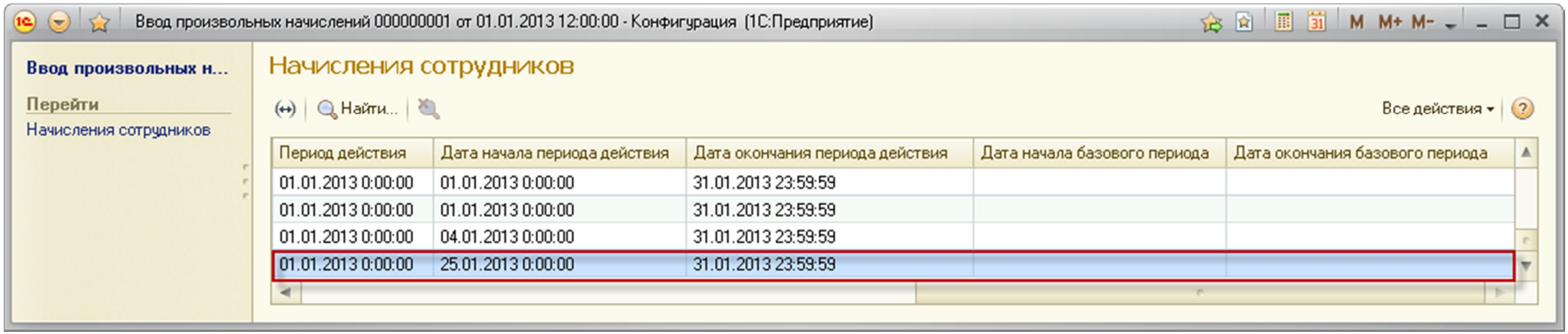


Изменим входящие данные, чтобы увидеть ошибку в расчете:
Допустим, сотрудник принят на работу не с 1-го числа, а с 25-го:

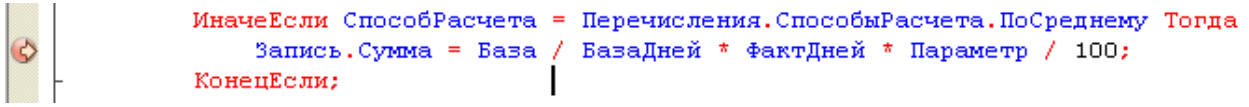


Результат проведения:





Теперь перепроведем документ с начислением отпуска и проверим результат в отладке:



```
ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
    Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
КонецЕсли;
```

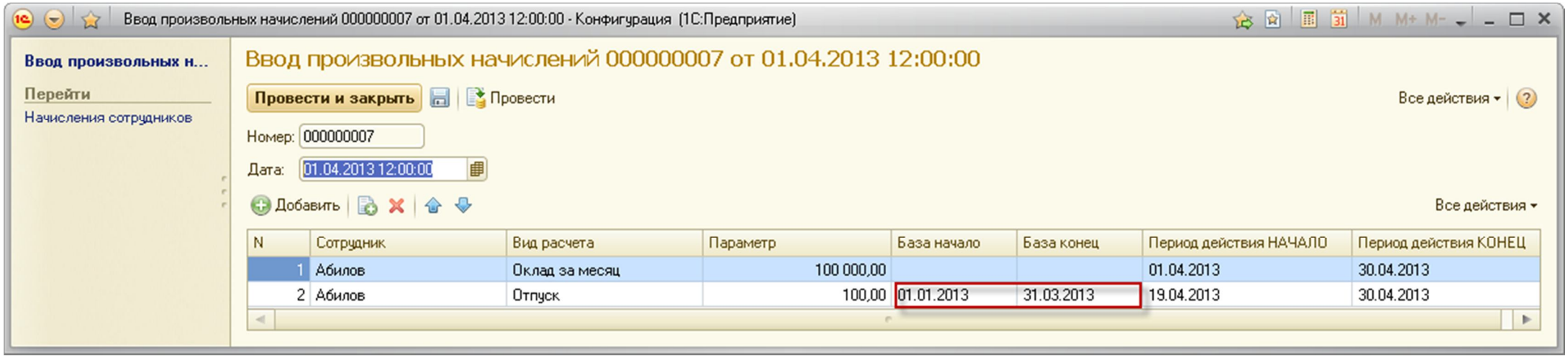
Выражение	Значение
База	221 739,13
БазаДней	64

База - сумма Базы стала меньше (это правильно).
БазаДней – осталась неизменной. Система рассчитала количество рабочих дней в базовом периоде, не учитывая, что работник принят не с 1-го января, а с 25-го. Вывод: БазаДней рассчитана неверно, она должна быть меньше.

Как рассчитывается:
При получении количества дней в базовом периоде:

```
ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.БазовыйПериод);
```

передается условие отбора «По базовому периоду». В записи, где рассчитывается отпуск указано, что базовый период с 01.01.2013 по 31.03.2013:



т.е. отбор на Календарь накладывается именно этими датами. Можно сказать, что мы получаем Плановое количество рабочих дней в базовом периоде.

Вывод: сейчас нигде не хранится информация о фактически отработанном сотрудником времени.
Количество заработанных сотрудником денег и отработанных дней – показатели базы. Регистр расчета дает возможность пользователю самостоятельно описывать Виды расчета и включать Виды расчета в Базу другого расчета.
Пример:
Для вида расчета «Оклад по дням» будет интересно накапливать базу «Отработано дней» - можно будет посмотреть, сколько дней проболел сотрудник, был в отпуске и т.д., получая Базу в разрезе Видов расчета.

Добавим в Регистр расчета еще один ресурс - «Отработано дней»:

Свойство	Значение
Имя	ОтработаноДней
Синоним	Отработано дней
Тип	Число, 2

Количество отработанных сотрудником дней за месяц не может быть больше 31, поэтому Длина – 2.

Не все виды расчета будут увеличивать ресурс «Отработано дней», поэтому в Плане видов расчета «Начисления» добавим реквизит «Увеличивает отработанные дни»:

Свойство	Значение
Имя	УвеличиваетОтработанныеДни
Синоним	Увеличивает отработанные дни
Тип	Булево

Допишем код расчета (Общий модуль «Расчеты» - процедура «Рассчитать»):

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);
```

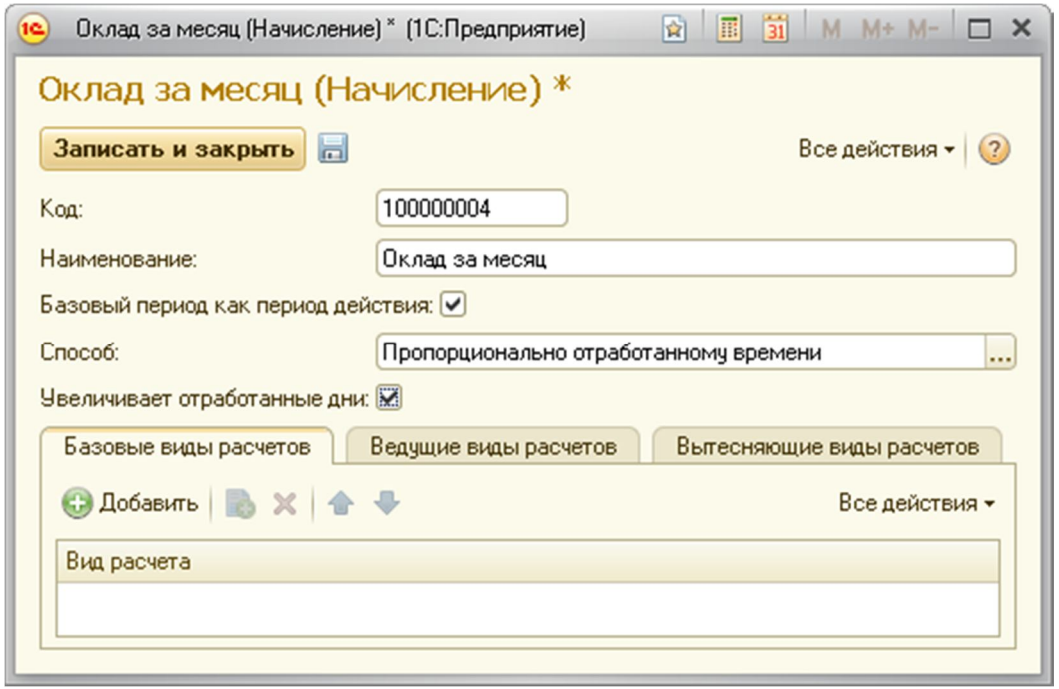
```
ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
ОтборПоРегистратору,
СписокРесурсов,
ОтборПоИзмерениям);

ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ФактическийПериодДействия);

ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ПериодДействия);

ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.БазовыйПериод);

Для Каждого Запись Из НаборЗаписей Цикл
     //Индекс = Запись.НомерСтроки - 1;
     Индекс = НаборЗаписей.Индекс(Запись);
     База = ТЗ_База[Индекс].Сумма;
     СпособРасчета = Запись.ВидРасчета.Способ;
     ФактДней = ТЗ_ФактДней[Индекс].Признак;
     НормаДней = ТЗ_НормаДней[Индекс].Признак;
     БазаДней = ТЗ_БазаДней[Индекс].Признак;
     Параметр = Запись.Параметр;
     УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

     Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
          Запись.Сумма = База * Параметр / 100;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
          Запись.Сумма = Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
          Запись.Сумма = ФактДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
```

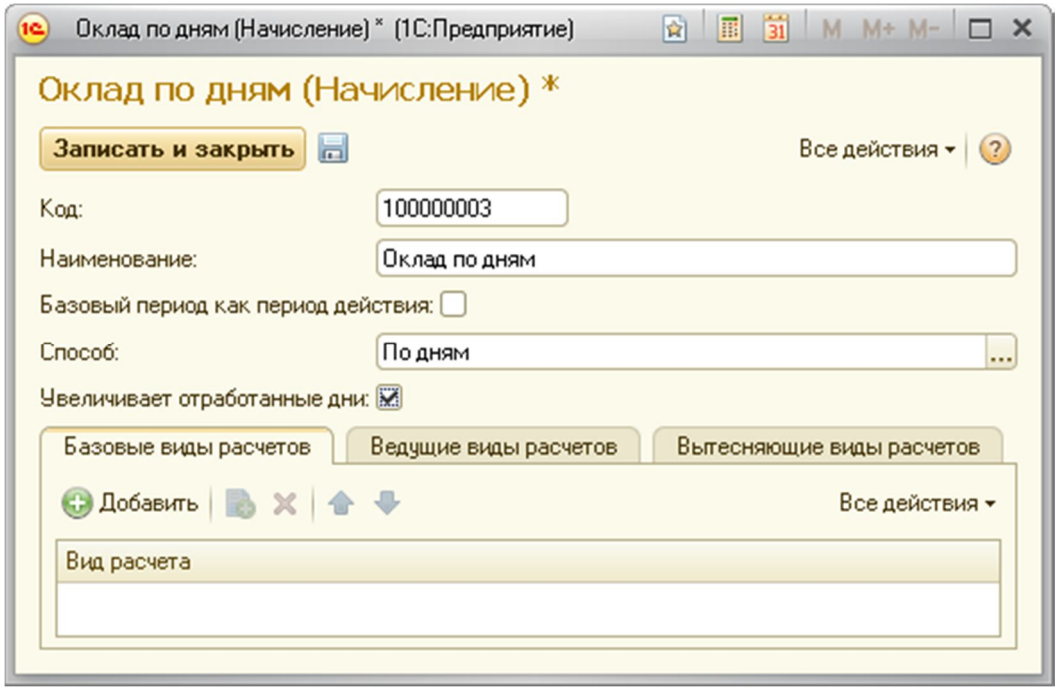
```
            Запись.Сумма = ФактДней / НормаДней * Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
            Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
        КонецЕсли;

        Если УвеличиваетОтработанныеДни тогда
            Запись.ОтработаноДней = ФактДней;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
            Запись.ОтработаноДней = - Запись.ОтработаноДней;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

Теперь будет накапливаться значение «Отработано дней».

В режиме исполнения:

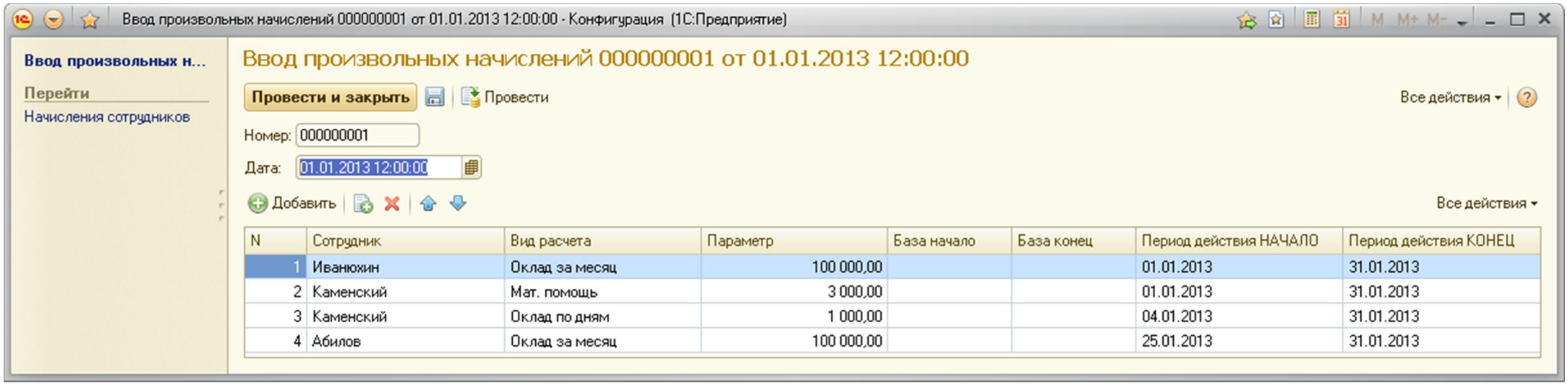
Отметим виды начислений, которые увеличивают количество отработанных дней:

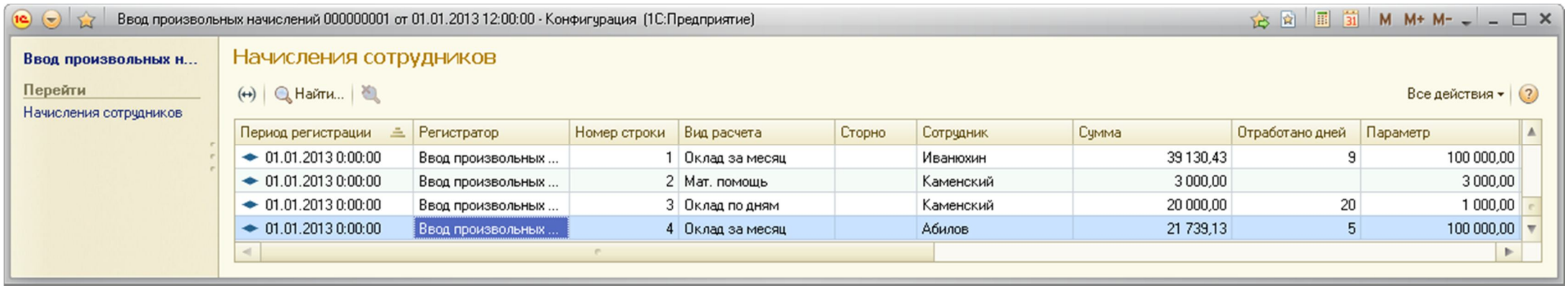


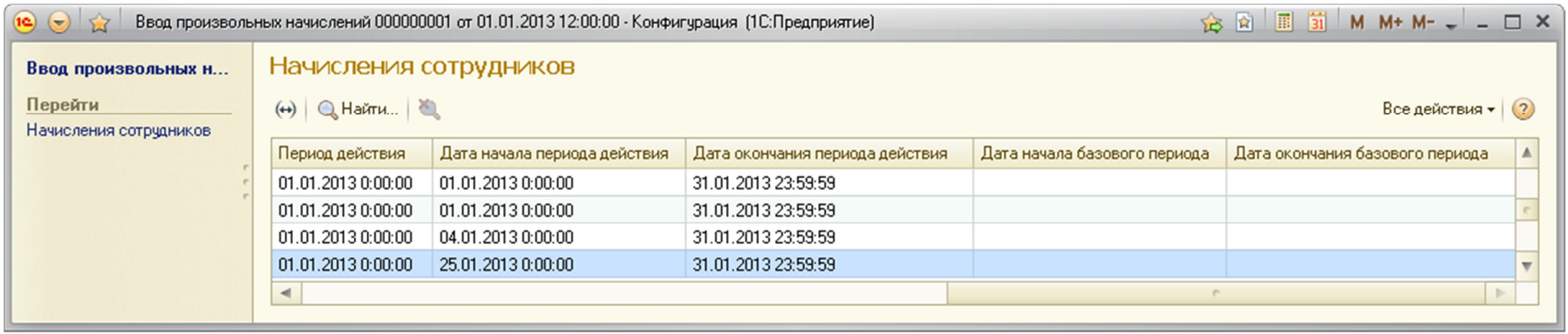


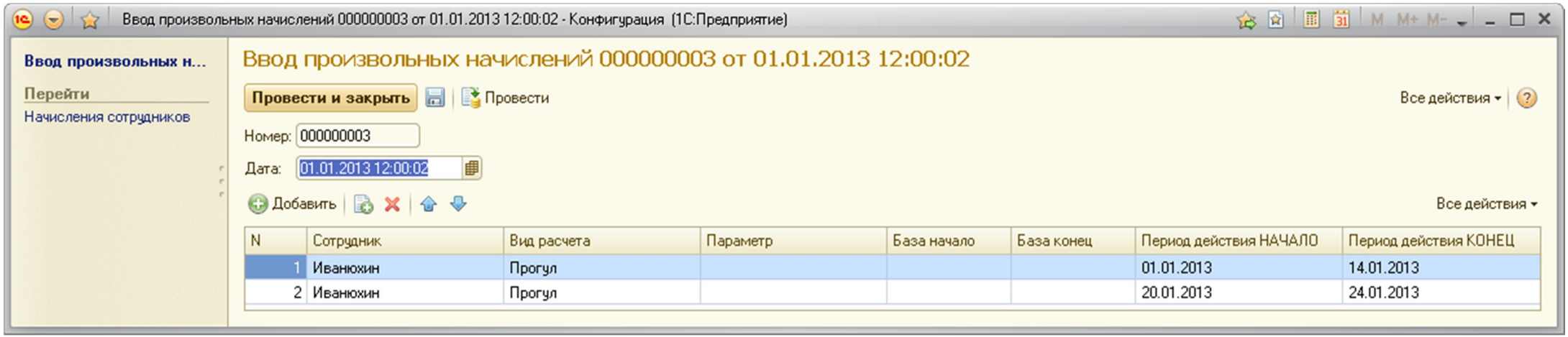
Перепроведем документы «Ввод произвольных начислений», в которых есть Виды начислений:

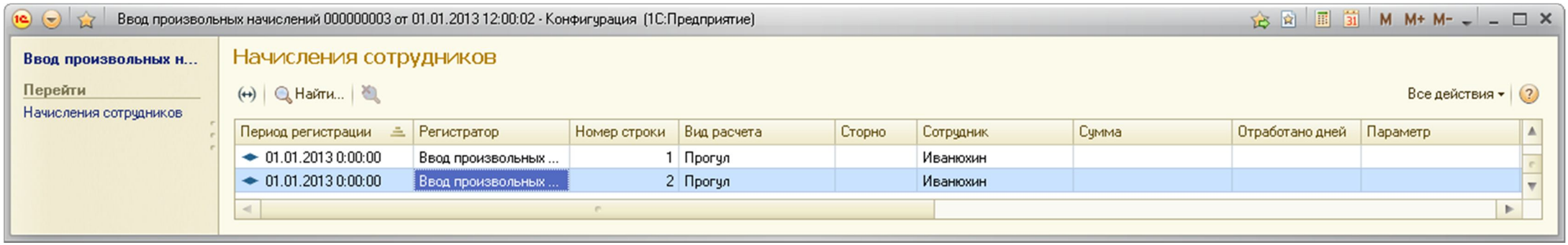
- Оклад за месяц.
- Оклад по дням.

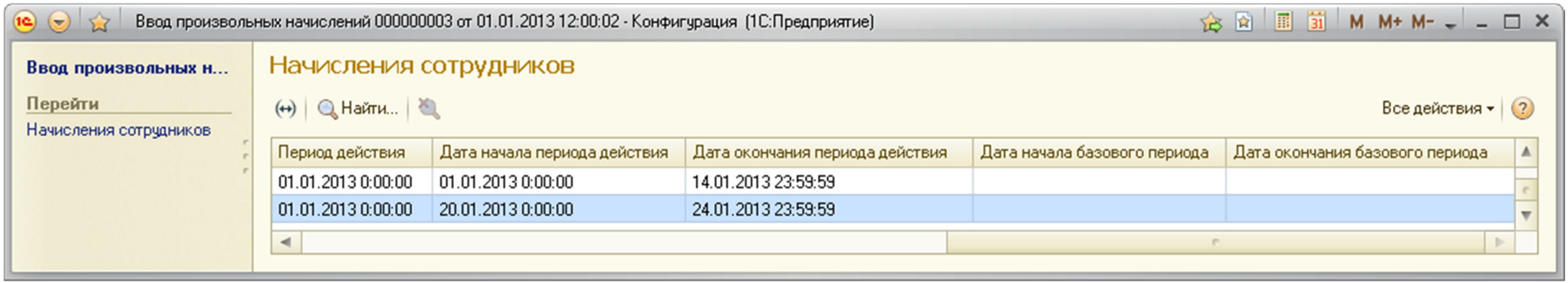

Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00
Провести и закрыть
Провести
Все действия
Номер: 000000001
Дата: 01.01.2013 12:00:00
Добавить
Все действия
N | Сотрудник | Вид расчета | Параметр | База начало | База конец | Период действия НАЧАЛО | Период действия КОНЕЦ
1 | Иванюхин | Оклад за месяц | 100 000,00 | | | 01.01.2013 | 31.01.2013
2 | Каменский | Мат. помощь | 3 000,00 | | | 01.01.2013 | 31.01.2013
3 | Каменский | Оклад по дням | 1 000,00 | | | 04.01.2013 | 31.01.2013
4 | Абилов | Оклад за месяц | 100 000,00 | | | 25.01.2013 | 31.01.2013

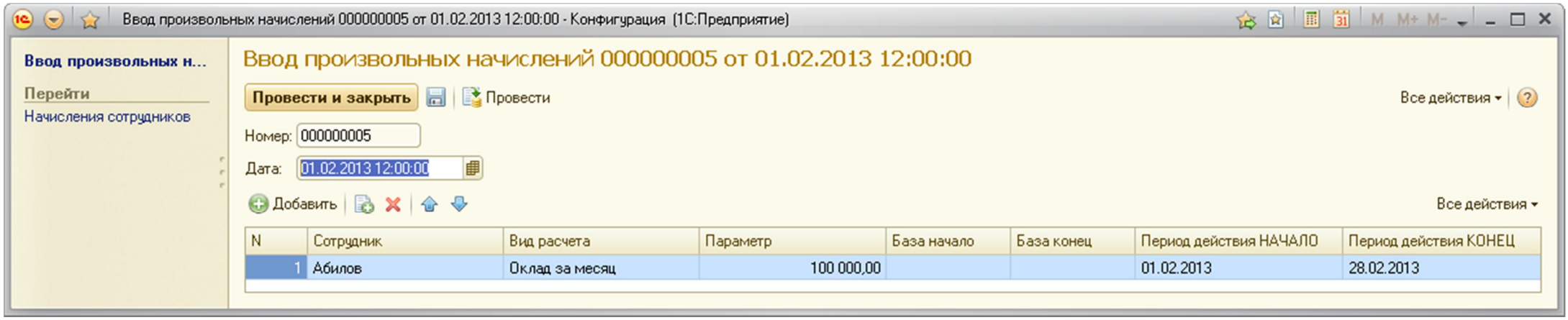

Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации | Регистратор | Номер строки | Вид расчета | Сторно | Сотрудник | Сумма | Отработано дней | Параметр
01.01.2013 0:00:00 | Ввод произвольных ... | 1 | Оклад за месяц | | Иванюхин | 39 130,43 | 9 | 100 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 2 | Мат. помощь | | Каменский | 3 000,00 | | 3 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 3 | Оклад по дням | | Каменский | 20 000,00 | 20 | 1 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 4 | Оклад за месяц | | Абилов | 21 739,13 | 5 | 100 000,00

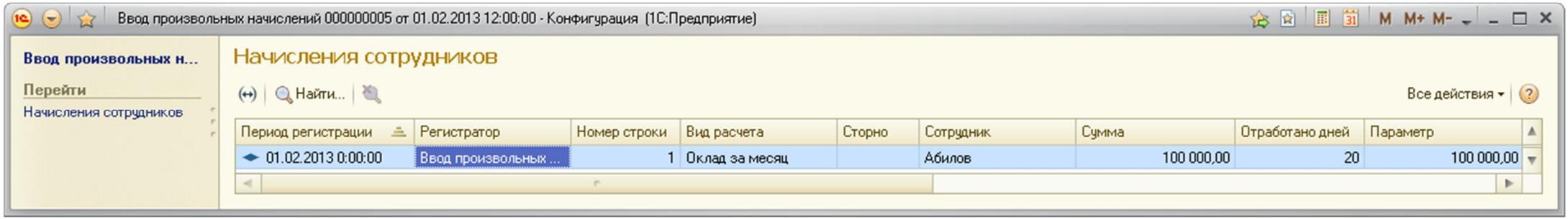

Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия | Дата начала периода действия | Дата окончания периода действия | Дата начала базового периода | Дата окончания базового периода
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 04.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 25.01.2013 0:00:00 | 31.01.2013 23:59:59

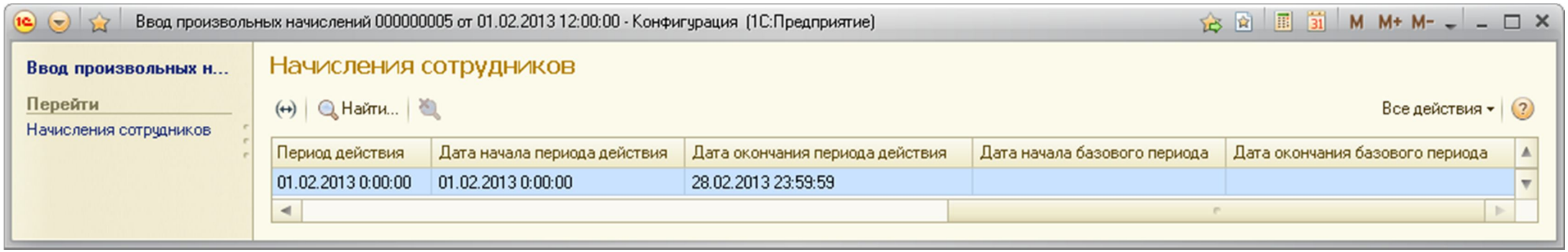

Ввод произвольных начислений 000000003 от 01.01.2013 12:00:02 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000003 от 01.01.2013 12:00:02
Провести и закрыть
Провести
Все действия
Номер: 000000003
Дата: 01.01.2013 12:00:02
Добавить
Все действия
N | Сотрудник | Вид расчета | Параметр | База начало | База конец | Период действия НАЧАЛО | Период действия КОНЕЦ
1 | Иванюхин | Прогул | | | | 01.01.2013 | 14.01.2013
2 | Иванюхин | Прогул | | | | 20.01.2013 | 24.01.2013

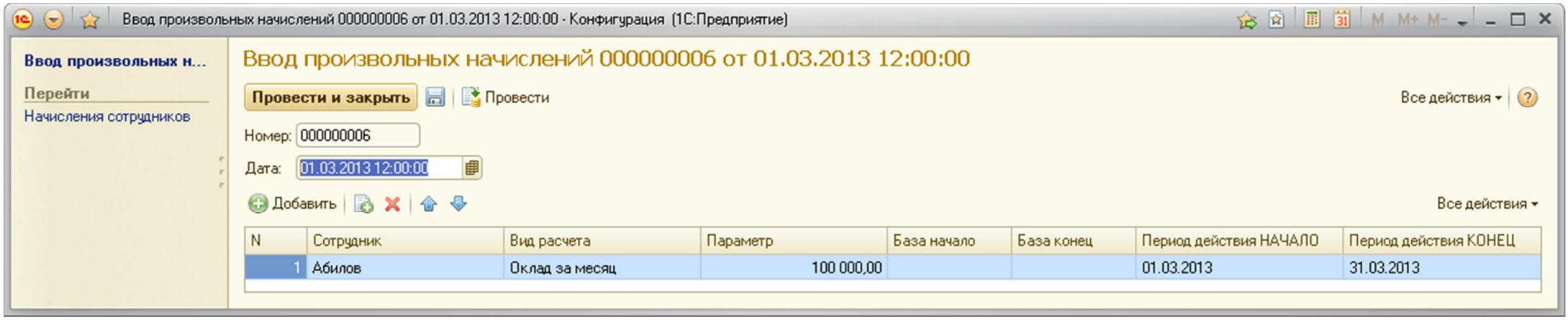

Ввод произвольных начислений 000000003 от 01.01.2013 12:00:02 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации | Регистратор | Номер строки | Вид расчета | Сторно | Сотрудник | Сумма | Отработано дней | Параметр
01.01.2013 0:00:00 | Ввод произвольных ... | 1 | Прогул | | Иванюхин
01.01.2013 0:00:00 | Ввод произвольных ... | 2 | Прогул | | Иванюхин

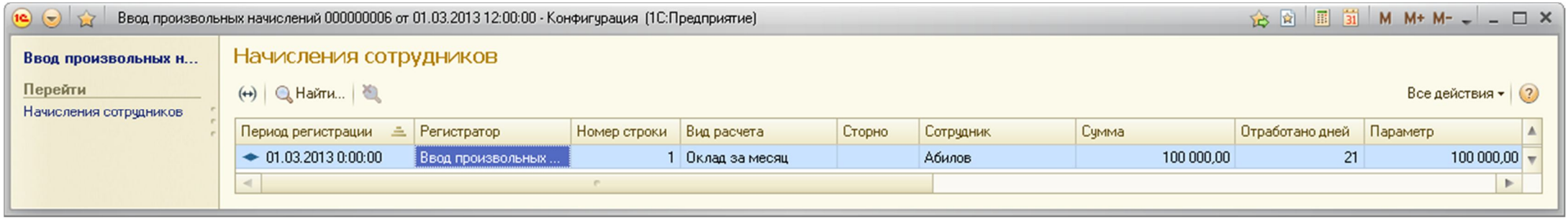

Ввод произвольных начислений 000000003 от 01.01.2013 12:00:02 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия | Дата начала периода действия | Дата окончания периода действия | Дата начала базового периода | Дата окончания базового периода
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 14.01.2013 23:59:59
01.01.2013 0:00:00 | 20.01.2013 0:00:00 | 24.01.2013 23:59:59

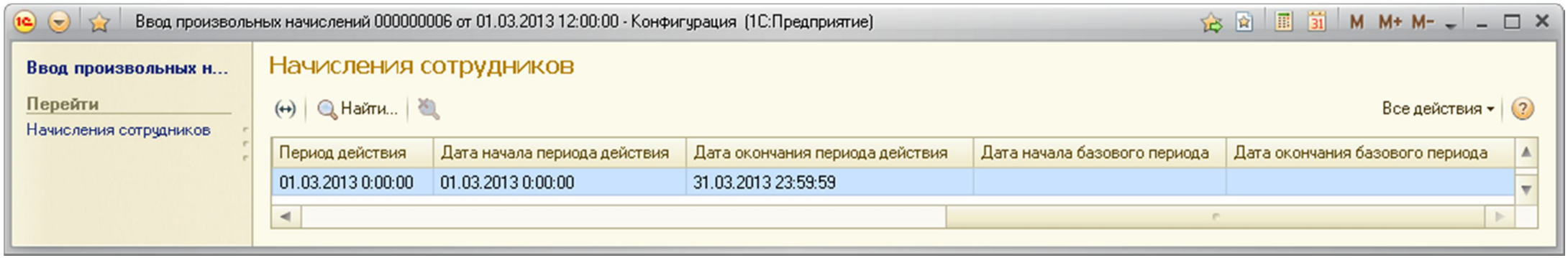

Ввод произвольных начислений 000000005 от 01.02.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000005 от 01.02.2013 12:00:00
Провести и закрыть
Провести
Все действия
Номер: 000000005
Дата: 01.02.2013 12:00:00
Добавить
Все действия
N
Сотрудник
Вид расчета
Параметр
База начало
База конец
Период действия НАЧАЛО
Период действия КОНЕЦ
1
Абилов
Оклад за месяц
100 000,00
01.02.2013
28.02.2013

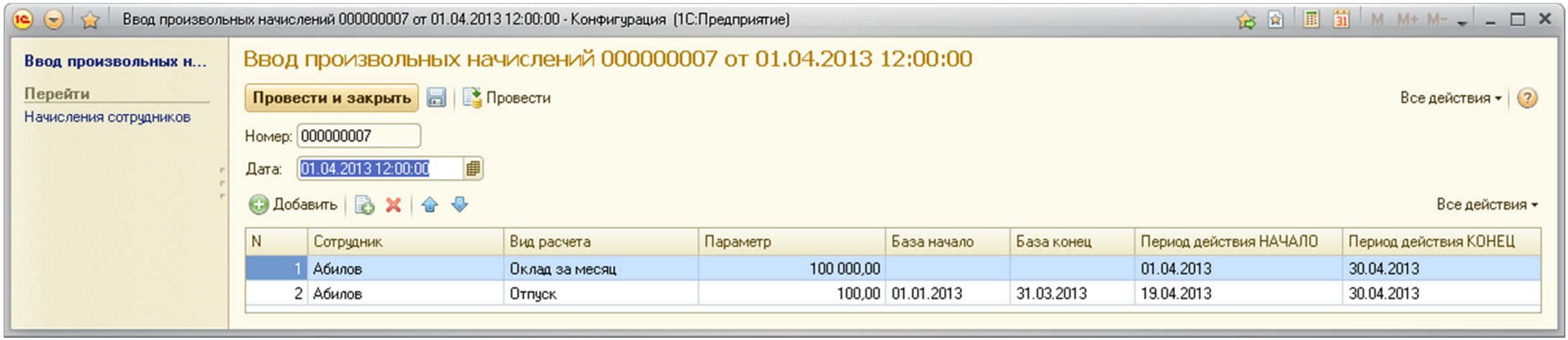

Ввод произвольных начислений 000000005 от 01.02.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации
Регистратор
Номер строки
Вид расчета
Сторно
Сотрудник
Сумма
Отработано дней
Параметр
01.02.2013 0:00:00
Ввод произвольных ...
1
Оклад за месяц
Абилов
100 000,00
20
100 000,00

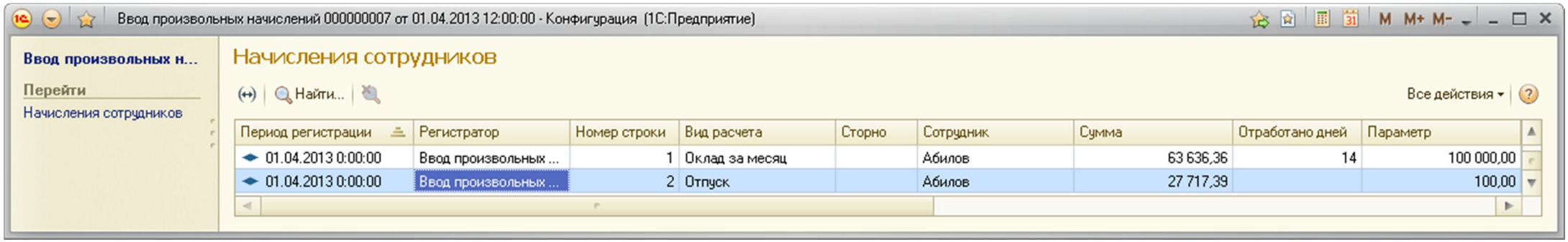

Ввод произвольных начислений 000000005 от 01.02.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия
Дата начала периода действия
Дата окончания периода действия
Дата начала базового периода
Дата окончания базового периода
01.02.2013 0:00:00
01.02.2013 0:00:00
28.02.2013 23:59:59

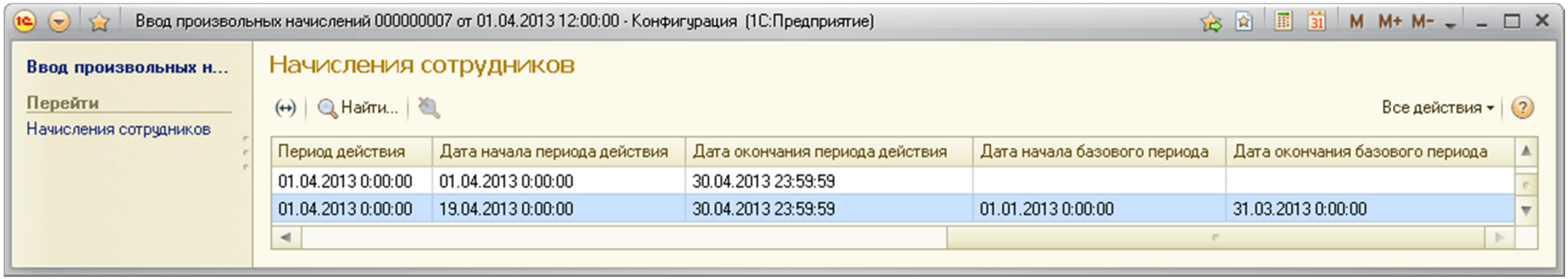

Ввод произвольных начислений 000000006 от 01.03.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000006 от 01.03.2013 12:00:00
Провести и закрыть
Провести
Все действия
Номер: 000000006
Дата: 01.03.2013 12:00:00
Добавить
Все действия
N
Сотрудник
Вид расчета
Параметр
База начало
База конец
Период действия НАЧАЛО
Период действия КОНЕЦ
1
Абилов
Оклад за месяц
100 000,00
01.03.2013
31.03.2013


Ввод произвольных начислений 000000006 от 01.03.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации
Регистратор
Номер строки
Вид расчета
Сторно
Сотрудник
Сумма
Отработано дней
Параметр
01.03.2013 0:00:00
Ввод произвольных ...
1
Оклад за месяц
Абилов
100 000,00
21
100 000,00


Ввод произвольных начислений 000000006 от 01.03.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия
Дата начала периода действия
Дата окончания периода действия
Дата начала базового периода
Дата окончания базового периода
01.03.2013 0:00:00
01.03.2013 0:00:00
31.03.2013 23:59:59


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00
Провести и закрыть
Провести
Все действия
Номер: 000000007
Дата: 01.04.2013 12:00:00
Добавить
Все действия
N | Сотрудник | Вид расчета | Параметр | База начало | База конец | Период действия НАЧАЛО | Период действия КОНЕЦ
1 | Абилов | Оклад за месяц | 100 000,00 | | | 01.04.2013 | 30.04.2013
2 | Абилов | Отпуск | 100,00 | 01.01.2013 | 31.03.2013 | 19.04.2013 | 30.04.2013


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации | Регистратор | Номер строки | Вид расчета | Сторно | Сотрудник | Сумма | Отработано дней | Параметр
01.04.2013 0:00:00 | Ввод произвольных ... | 1 | Оклад за месяц | | Абилов | 63 636,36 | 14 | 100 000,00
01.04.2013 0:00:00 | Ввод произвольных ... | 2 | Отпуск | | Абилов | 27 717,39 | | 100,00


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия | Дата начала периода действия | Дата окончания периода действия | Дата начала базового периода | Дата окончания базового периода
01.04.2013 0:00:00 | 01.04.2013 0:00:00 | 30.04.2013 23:59:59 | | |
01.04.2013 0:00:00 | 19.04.2013 0:00:00 | 30.04.2013 23:59:59 | 01.01.2013 0:00:00 | 31.03.2013 0:00:00

Изменим расчет (Общий модуль «Расчеты» - процедура «Рассчитать»):

Теперь мы не будем обращаться к данным графика, т.к. оттуда мы получаем только данные по плану, а надо по факту.
Чтобы получить информацию о количестве отработанных сотрудником дней (теперь эта информация хранится как Ресурс Регистра расчета, а Ресурс образует Базу), получаем Базу (метод «ПолучитьБазу»):

```
ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
ОтборПоРегистратору,
СписокРесурсов,
ОтборПоИзмерениям);
```

А что является Базой, определяется списком Ресурсов, которые суммируются.

Итого код:

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(2);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";
    СписокРесурсов[1] = "НачисленияСотрудников.ОтработаноДней";

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
```

```
ОтборПоИзмерениям);

ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ФактическийПериодДействия);

ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
ОтборПоРегистратору,
ВидПериодаРегистраРасчета.ПериодДействия);

//ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
//ОтборПоРегистратору,
//ВидПериодаРегистраРасчета.БазовыйПериод);

Для Каждого Запись Из НаборЗаписей Цикл
     //Индекс = Запись.НомерСтроки - 1;
     Индекс = НаборЗаписей.Индекс(Запись);
     База = ТЗ_База[Индекс].Сумма;
     СпособРасчета = Запись.ВидРасчета.Способ;
     ФактДней = ТЗ_ФактДней[Индекс].Признак;
     НормаДней = ТЗ_НормаДней[Индекс].Признак;
     //БазаДней = ТЗ_БазаДней[Индекс].Признак;
     БазаДней = ТЗ_База[Индекс].ОтработаноДней;
     Параметр = Запись.Параметр;
     УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

     Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
          Запись.Сумма = База * Параметр / 100;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
          Запись.Сумма = Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
          Запись.Сумма = ФактДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
          Запись.Сумма = ФактДней / НормаДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
          Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
```

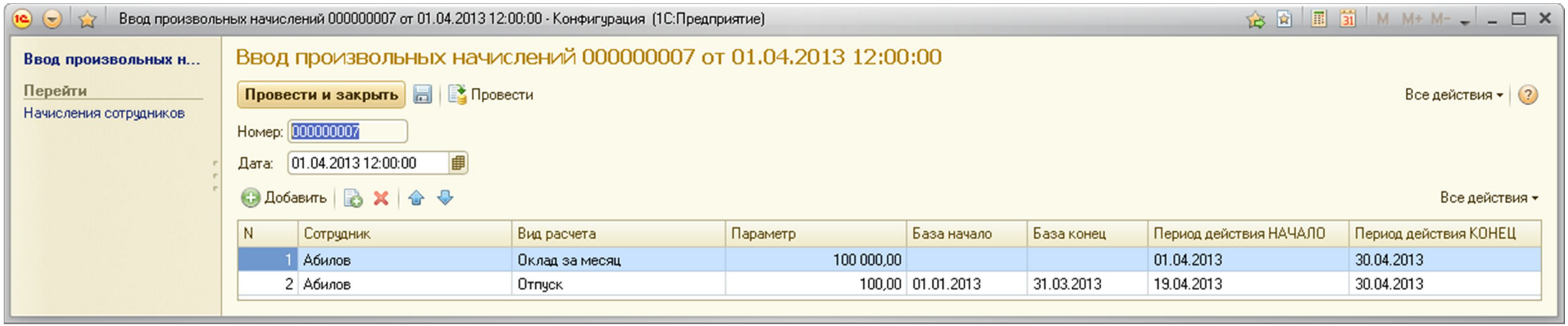
```
        КонецЕсли;

        Если УвеличиваетОтработанныеДни тогда
            Запись.ОтработаноДней = ФактДней;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
            Запись.ОтработаноДней = - Запись.ОтработаноДней;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры
```

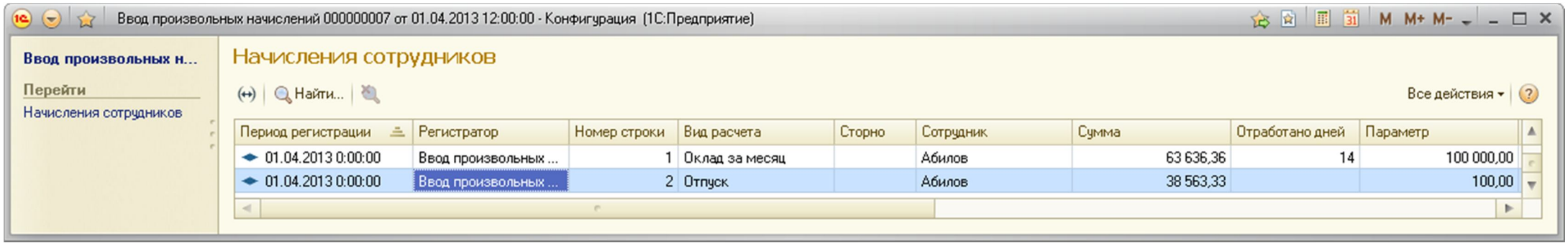
В режиме исполнения (перепроведем и проверим на отладке документ «Ввод произвольных начислений» № 000000007):

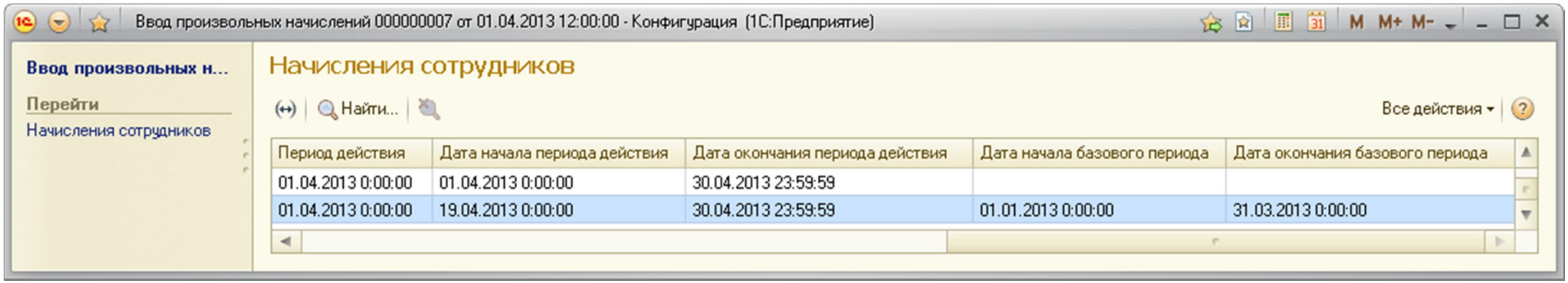
```
ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
    Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
КонецЕсли;
```

Выражение	Значение
База	221 739,13
БазаДней	46

БазаДней стала меньше.


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00
Провести и закрыть
Провести
Все действия
Номер: 000000007
Дата: 01.04.2013 12:00:00
Добавить
Все действия
N | Сотрудник | Вид расчета | Параметр | База начало | База конец | Период действия НАЧАЛО | Период действия КОНЕЦ
1 | Абилов | Оклад за месяц | 100 000,00 | | | 01.04.2013 | 30.04.2013
2 | Абилов | Отпуск | 100,00 | 01.01.2013 | 31.03.2013 | 19.04.2013 | 30.04.2013


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации | Регистратор | Номер строки | Вид расчета | Сторно | Сотрудник | Сумма | Отработано дней | Параметр
01.04.2013 0:00:00 | Ввод произвольных ... | 1 | Оклад за месяц | | Абилов | 63 636,36 | 14 | 100 000,00
01.04.2013 0:00:00 | Ввод произвольных ... | 2 | Отпуск | | Абилов | 38 563,33 | | 100,00


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия | Дата начала периода действия | Дата окончания периода действия | Дата начала базового периода | Дата окончания базового периода
01.04.2013 0:00:00 | 01.04.2013 0:00:00 | 30.04.2013 23:59:59 | | |
01.04.2013 0:00:00 | 19.04.2013 0:00:00 | 30.04.2013 23:59:59 | 01.01.2013 0:00:00 | 31.03.2013 0:00:00

Усложним пример: каждый вид начисления рассчитывается исходя из выбранного графика.
«Календарь» - это у нас Регистр сведений. В Регистрах сведений есть Измерения, которые, по сути, являются новыми разрезами значений показателя.
Добавим справочник, в котором будут храниться виды графиков – «Графики»:

Свойство	Значение
Имя	Графики
Синоним	Графики
Представление объекта	График

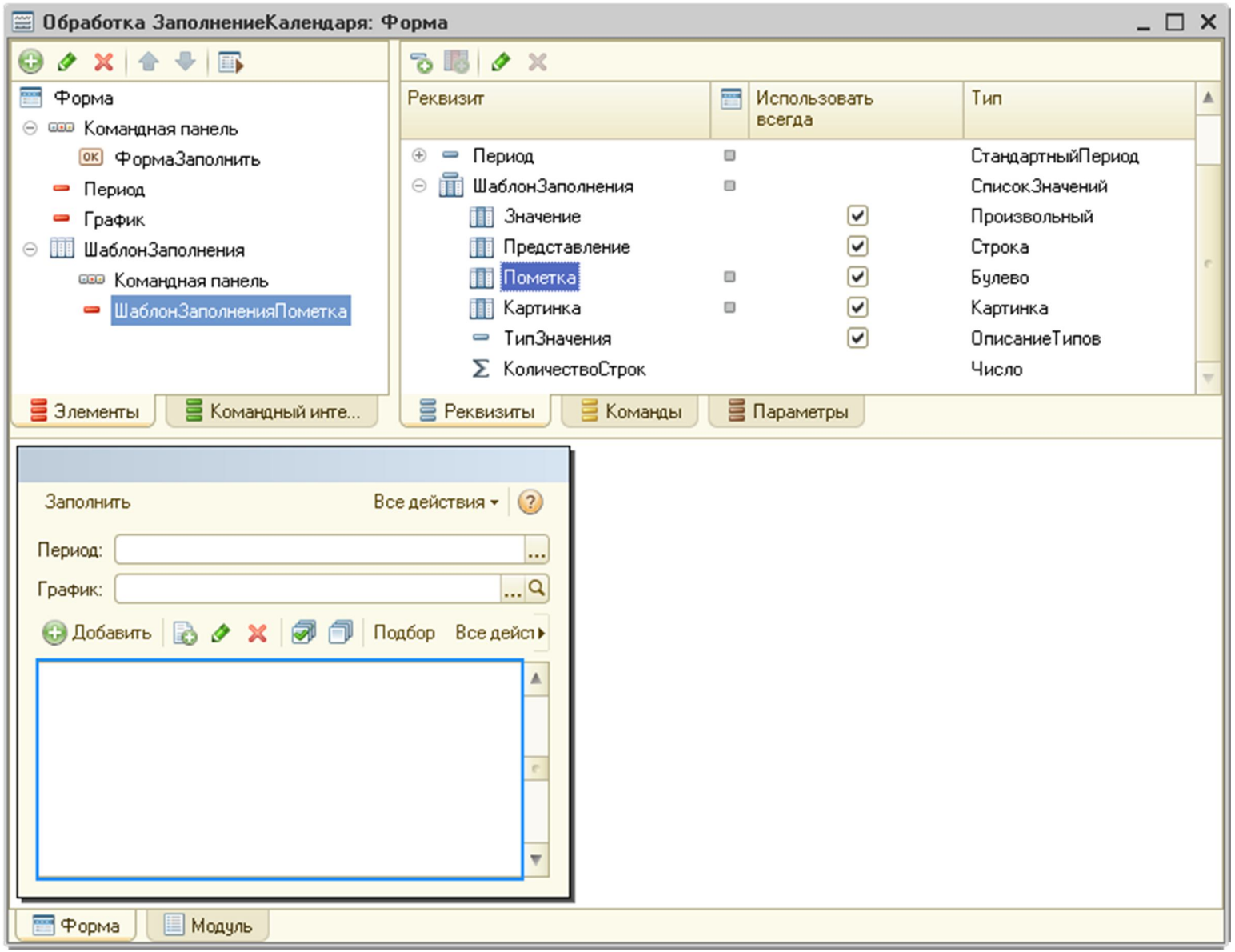
В Регистр сведений «Календарь» добавим Измерение «График»:

Свойство	Значение
Имя	График
Синоним	График
Тип	СправочникСсылка.Графики
Ведущее	Истина
Запрет незаполненных значений	Истина

Доработаем обработку «Заполнение календаря»:
Добавим реквизиты формы и поместим их на форму:

Свойство	Значение
График	
Имя	График
Синоним	График
Тип	СправочникСсылка.Графики
Период	
Имя	Период
Синоним	Период
Тип	СтандартныйПериод
Шаблон заполнения	
Имя	ШаблонЗаполнения
Синоним	Шаблон заполнения
Тип	СписокЗначений

Тип Шаблона заполнения на фирме – Таблица. Поле таблицы – ШаблонЗаполненияПометка.


Обработка ЗаполнениеКалендаря: Форма
Форма
Командная панель
ФормаЗаполнить
Период
График
ШаблонЗаполнения
Командная панель
ШаблонЗаполненияПометка
Элементы
Командный инте...
Реквизит
Использовать всегда
Тип
Период
СтандартныйПериод
ШаблонЗаполнения
СписокЗначений
Значение
Произвольный
Представление
Строка
Пометка
Булево
Картинка
Картинка
ТипЗначения
ОписаниеТипов
КоличествоСтрок
Число
Реквизиты
Команды
Параметры
Заполнить
Все действия
Период:
График:
Добавить
Подбор
Все дейст
Форма
Модуль

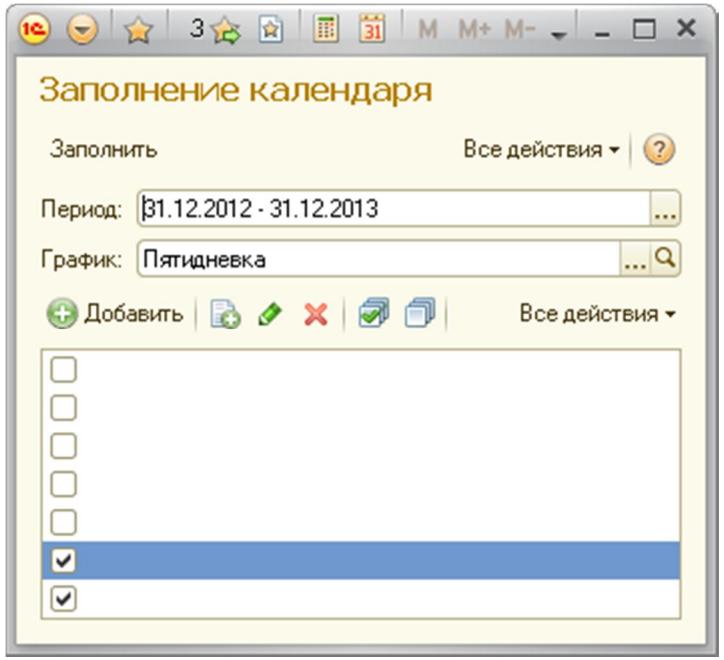
Модуль формы:

```
&НаКлиенте
Процедура Заполнить(Команда)
	ЗаполнитьНаСервере();
	Предупреждение(НСтр("ru = 'Обработка завершена'"), 60);
КонецПроцедуры

&НаСервере
Процедура ЗаполнитьНаСервере()

	КолДней = (НачалоДня(Период.ДатаОкончания) - НачалоДня(Период.ДатаНачала)) / 86400;
	          // 86400 - количество секунд в одном дне

	НачатьТранзакцию();
	//переберем текущий год
	Для Ном = 0 По КолДней Цикл
		//рассчитаем текущую дату
		ТекДата = Период.ДатаНачала + 86400 * Ном;
		//сформируем записи в регистре сведений
		Запись = РегистрыСведений.Календарь.СоздатьМенеджерЗаписи();
		Запись.График = График;
		Запись.Дата = ТекДата;
		Запись.Признак = ПолучитьЗначениеПоДате(ТекДата);
		Запись.Записать();
	КонецЦикла;
	ЗафиксироватьТранзакцию();
КонецПроцедуры // ЗаполнитьНаСервере()

&НаСервере
Функция ПолучитьЗначениеПоДате(Дата)
	Разность = (Дата - Период.ДатаНачала) / 86400;
	Индекс = Разность % ШаблонЗаполнения.Количество();
	Возврат ?(ШаблонЗаполнения[Индекс].Пометка, 0, 1);
КонецФункции
```

Сначала очистим старые записи:

```
Процедура ЗаполнитьНаСервере()

	КолДней = (НачалоДня(Период.ДатаОкончания) - НачалоДня(Период.ДатаНачала)) / 86400;
	          // 86400 - количество секунд в одном дне

	НачатьТранзакцию();
	//переберем текущий год
	Для Ном = 0 По КолДней Цикл
		//рассчитаем текущую дату
		ТекДата = Период.ДатаНачала + 86400 * Ном;
		//сформируем записи в регистре сведений
		Запись = РегистрыСведений.Календарь.СоздатьМенеджерЗаписи();
		//Запись.График = График;
		Запись.Дата = ТекДата;
		//Запись.Признак = ПолучитьЗначениеПоДате(ТекДата);
		//Запись.Записать();
		Запись.Удалить();
	КонецЦикла;
	ЗафиксироватьТранзакцию();
КонецПроцедуры // ЗаполнитьНаСервере()
```

В режиме исполнения:
Удалим все записи за этот год. Вернем процедуру к исходному коду и заполним Календарь по-новому:
Пятидневка:

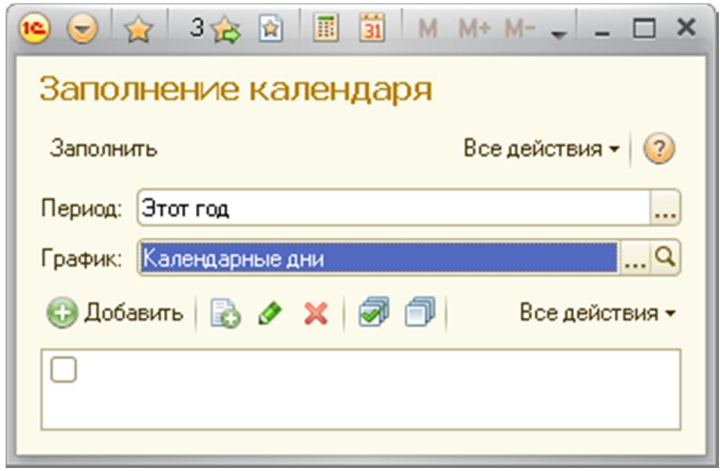


Чтобы все заполнилось верно, надо, чтобы Дата начала всегда была Понедельником.

Результат:

Дата	График	Признак
31.12.20...	Пятидневка	1
01.01.20...	Пятидневка	1
02.01.20...	Пятидневка	1
03.01.20...	Пятидневка	1
04.01.20...	Пятидневка	1
05.01.20...	Пятидневка	
06.01.20...	Пятидневка	
07.01.20...	Пятидневка	1
08.01.20...	Пятидневка	1
09.01.20...	Пятидневка	1
10.01.20...	Пятидневка	1
11.01.20...	Пятидневка	1
12.01.20...	Пятидневка	
13.01.20...	Пятидневка	

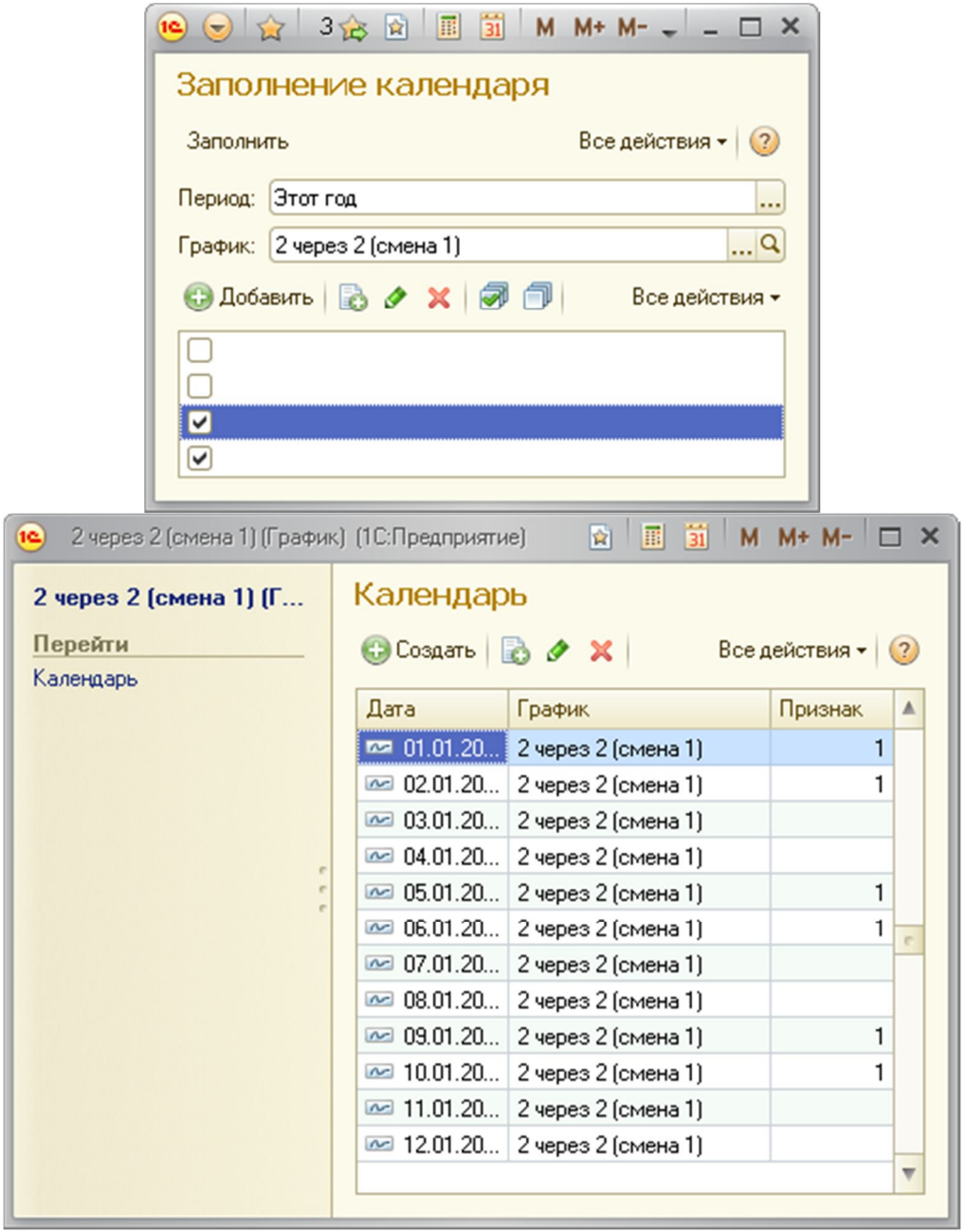
Календарные дни (все рабочие):

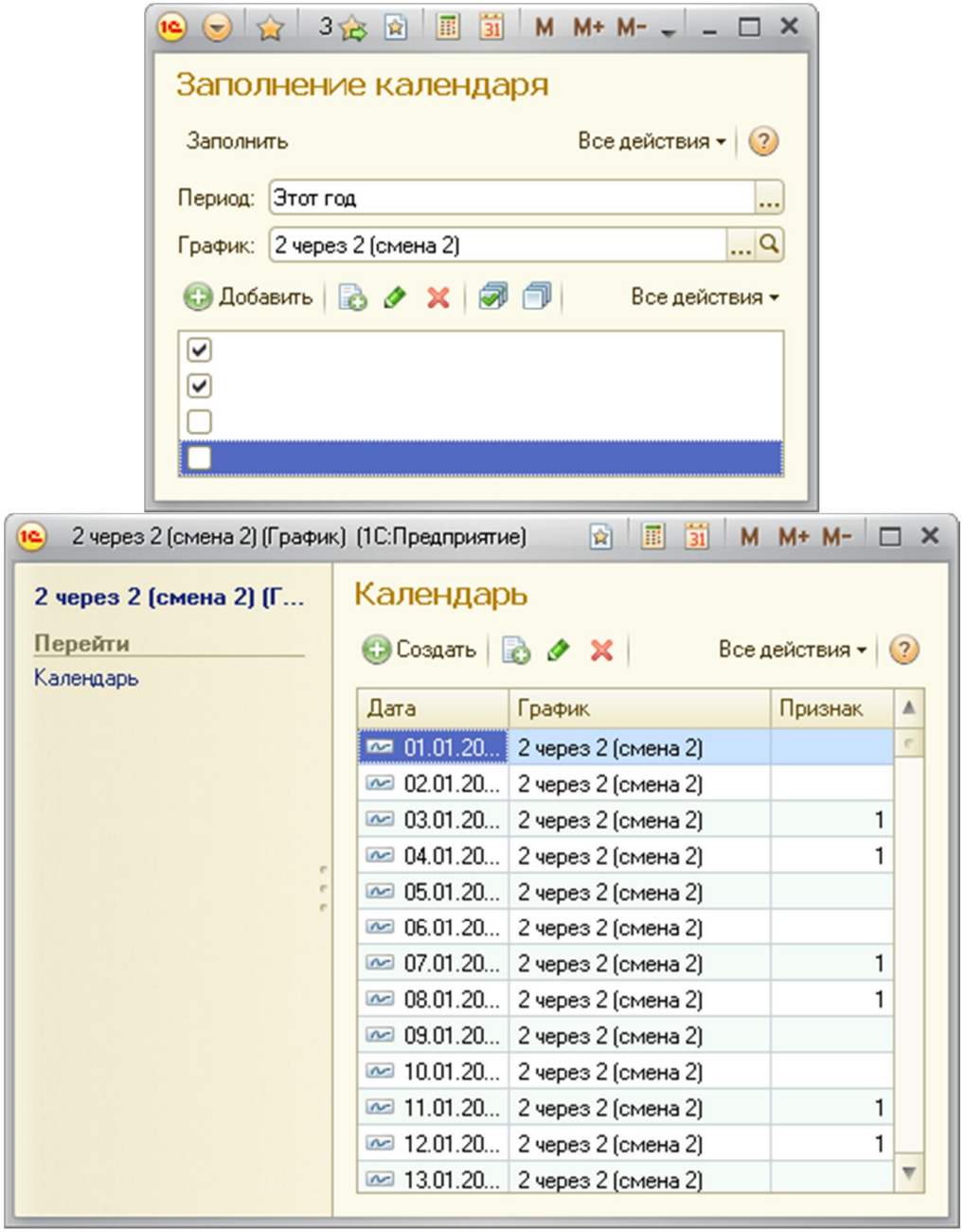


Результат:

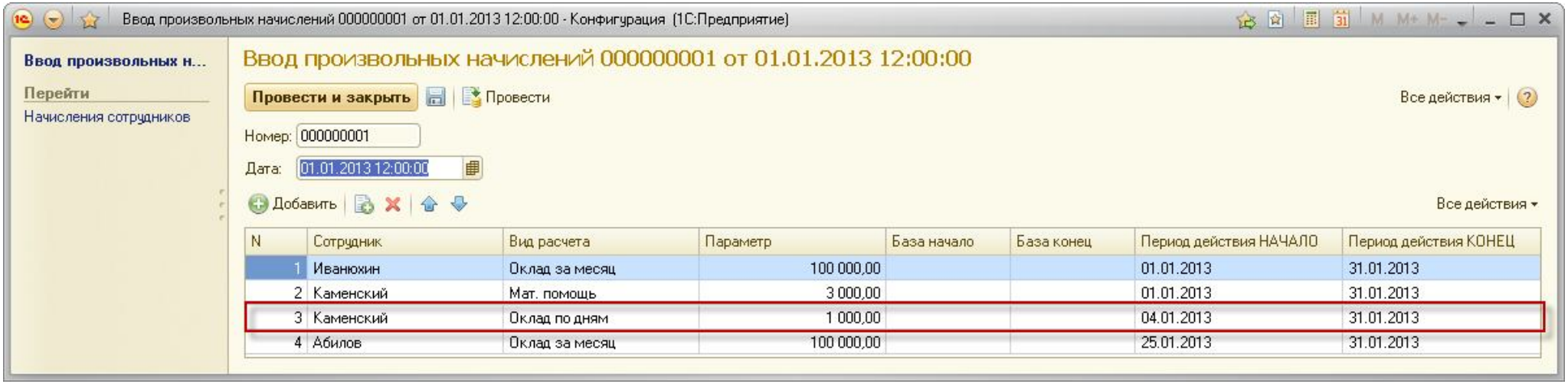
Дата	График	Признак
04.01.20...	Пятидневка	1
04.01.20...	Календарные дни	1
05.01.20...	Пятидневка	
05.01.20...	Календарные дни	1
06.01.20...	Пятидневка	
06.01.20...	Календарные дни	1
07.01.20...	Пятидневка	1
07.01.20...	Календарные дни	1
08.01.20...	Пятидневка	1
08.01.20...	Календарные дни	1
09.01.20...	Пятидневка	1
09.01.20...	Календарные дни	1
10.01.20...	Пятидневка	1
10.01.20...	Календарные дни	1
11.01.20...	Пятидневка	1
11.01.20...	Календарные дни	1
12.01.20...	Пятидневка	
12.01.20...	Календарные дни	1
13.01.20...	Пятидневка	
13.01.20...	Календарные дни	1

2 через 2 (смена 1 или смена 2):



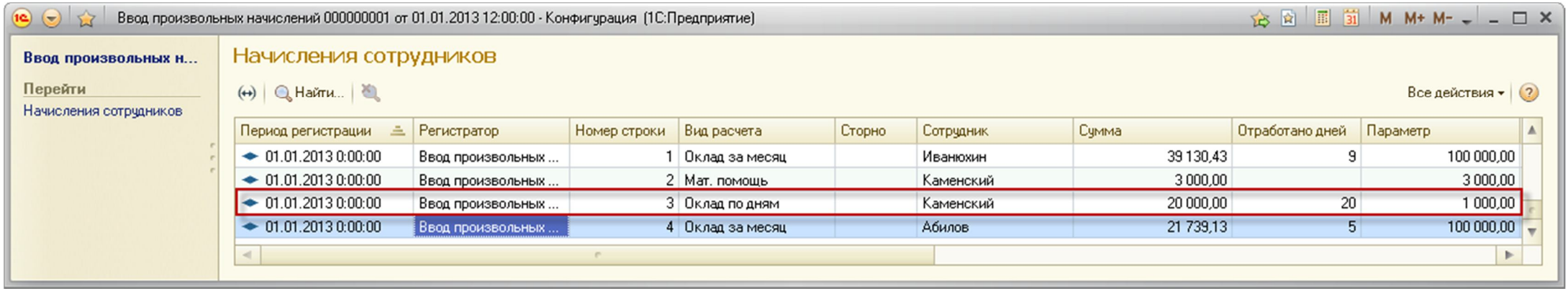


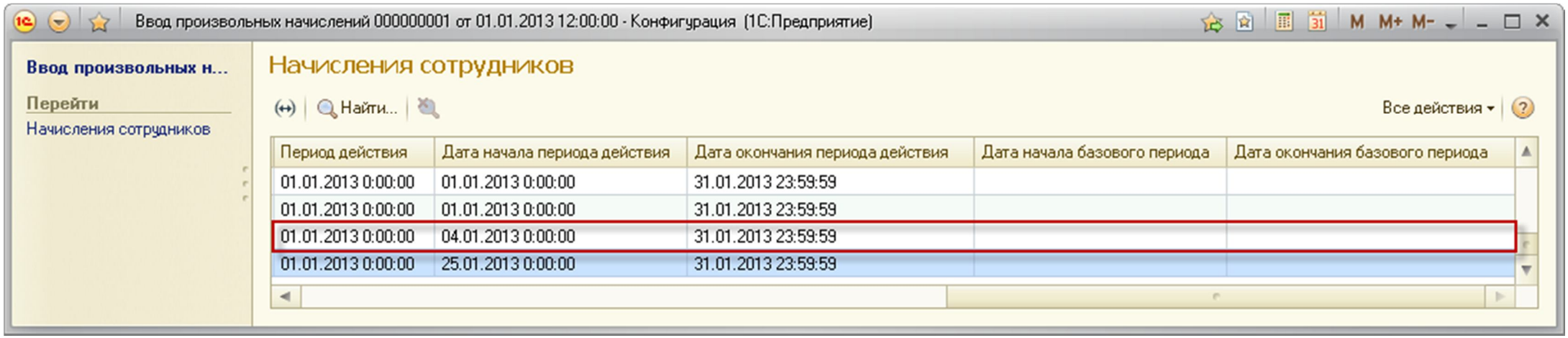
Теперь мы можем к каждой записи Регистра расчета привязать информацию о том, к какому календарю она относится. Перепроведем документ «Ввод произвольных начислений», в котором есть Оклад по дням до привязки:



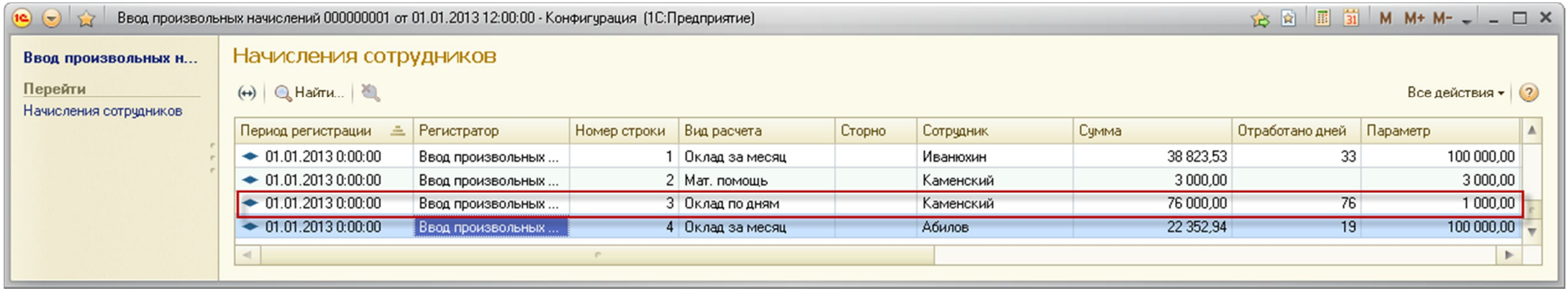
N	Сотрудник	Вид расчета	Параметр	База начало	База конец	Период действия НАЧАЛО	Период действия КОНЕЦ
1	Иванюхин	Оклад за месяц	100 000,00			01.01.2013	31.01.2013
2	Каменский	Мат. помощь	3 000,00			01.01.2013	31.01.2013
3	Каменский	Оклад по дням	1 000,00			04.01.2013	31.01.2013
4	Абилов	Оклад за месяц	100 000,00			25.01.2013	31.01.2013

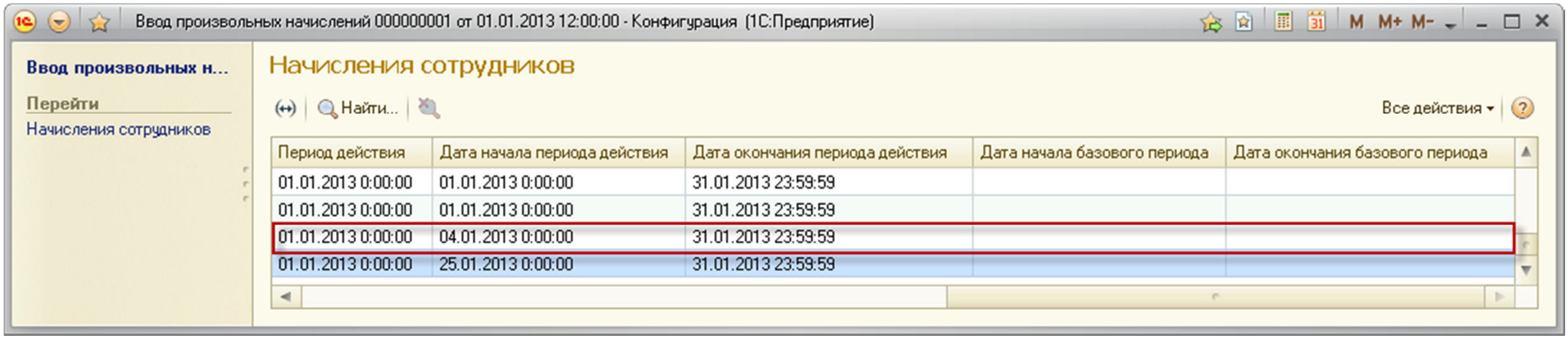
Было:


Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации | Регистратор | Номер строки | Вид расчета | Сторно | Сотрудник | Сумма | Отработано дней | Параметр
01.01.2013 0:00:00 | Ввод произвольных ... | 1 | Оклад за месяц | | Иванюхин | 39 130,43 | 9 | 100 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 2 | Мат. помощь | | Каменский | 3 000,00 | | 3 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 3 | Оклад по дням | | Каменский | 20 000,00 | 20 | 1 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 4 | Оклад за месяц | | Абилов | 21 739,13 | 5 | 100 000,00


Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия | Дата начала периода действия | Дата окончания периода действия | Дата начала базового периода | Дата окончания базового периода
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 04.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 25.01.2013 0:00:00 | 31.01.2013 23:59:59

Стало:


Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации | Регистратор | Номер строки | Вид расчета | Сторно | Сотрудник | Сумма | Отработано дней | Параметр
01.01.2013 0:00:00 | Ввод произвольных ... | 1 | Оклад за месяц | | Иванюхин | 38 823,53 | 33 | 100 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 2 | Мат. помощь | | Каменский | 3 000,00 | | 3 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 3 | Оклад по дням | | Каменский | 76 000,00 | 76 | 1 000,00
01.01.2013 0:00:00 | Ввод произвольных ... | 4 | Оклад за месяц | | Абилов | 22 352,94 | 19 | 100 000,00


Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период действия | Дата начала периода действия | Дата окончания периода действия | Дата начала базового периода | Дата окончания базового периода
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 01.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 04.01.2013 0:00:00 | 31.01.2013 23:59:59
01.01.2013 0:00:00 | 25.01.2013 0:00:00 | 31.01.2013 23:59:59

Как это получилось:

Количество дней считается методом «ПолучитьДанныеГрафика()» - значение получаем как отбор, наложенный на две даты. Сейчас у нас в Регистре сведений Календарь очень много записей (4 вида календаря). Поэтому при расчете Количества дней надо, чтобы система отбирала дни определенного Графика:
В Регистр расчета «НачисленияСотрудников» добавим реквизит «График»:

Свойство	Значение
Имя	График
Синоним	График
Тип	СправочникСсылка.Графики

и свяжем его с Регистром сведений «Календарь»:

Свойство	Значение
Связь с графиком	График

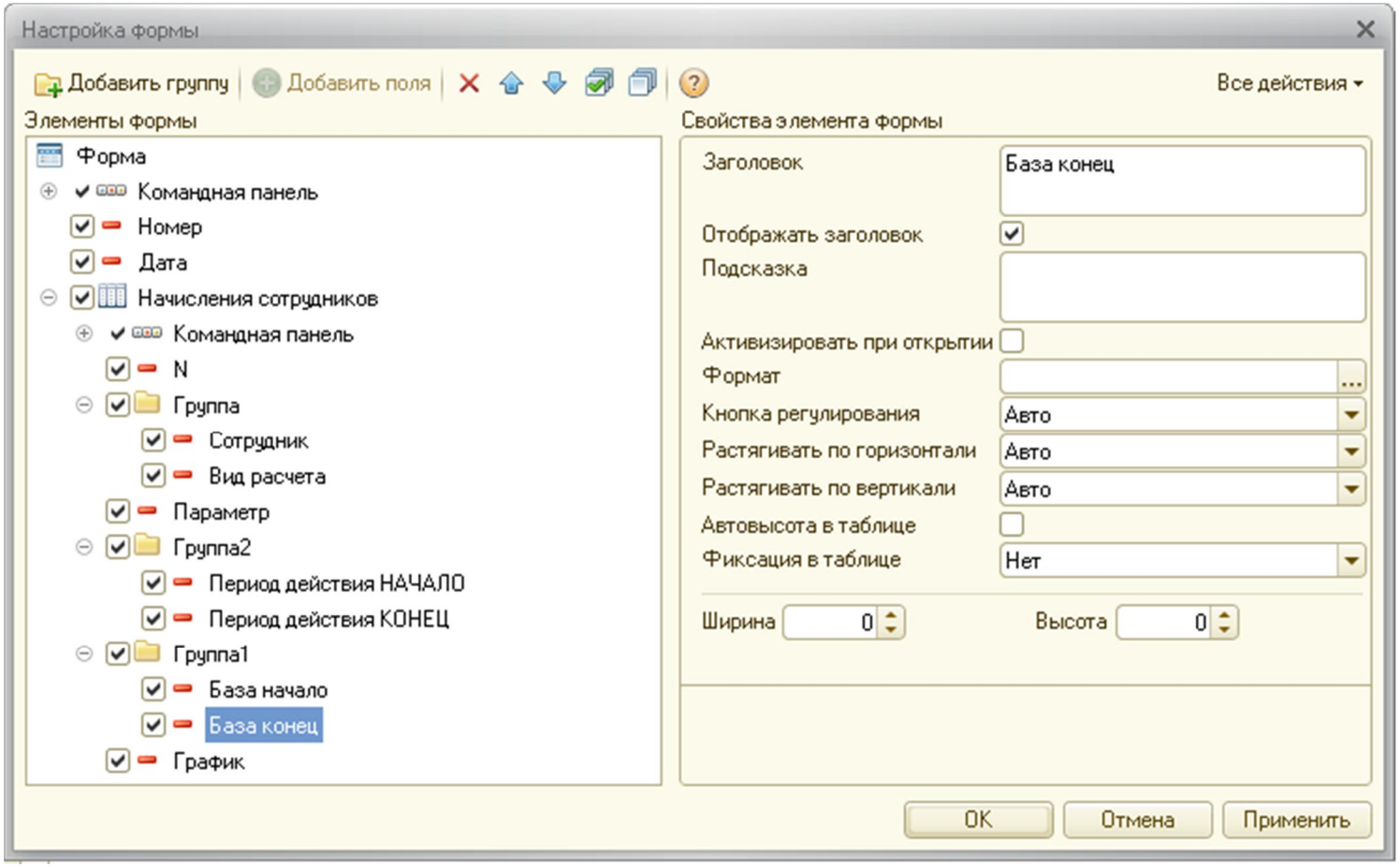
Каждая запись должна быть привязана к Графикам:
Документ «Ввод произвольных начислений» - табличная часть «НачисленияСотрудников» - добавим новый реквизит - «График»:

Свойство	Значение
Имя	График
Синоним	График
Тип	СправочникСсылка.Графики

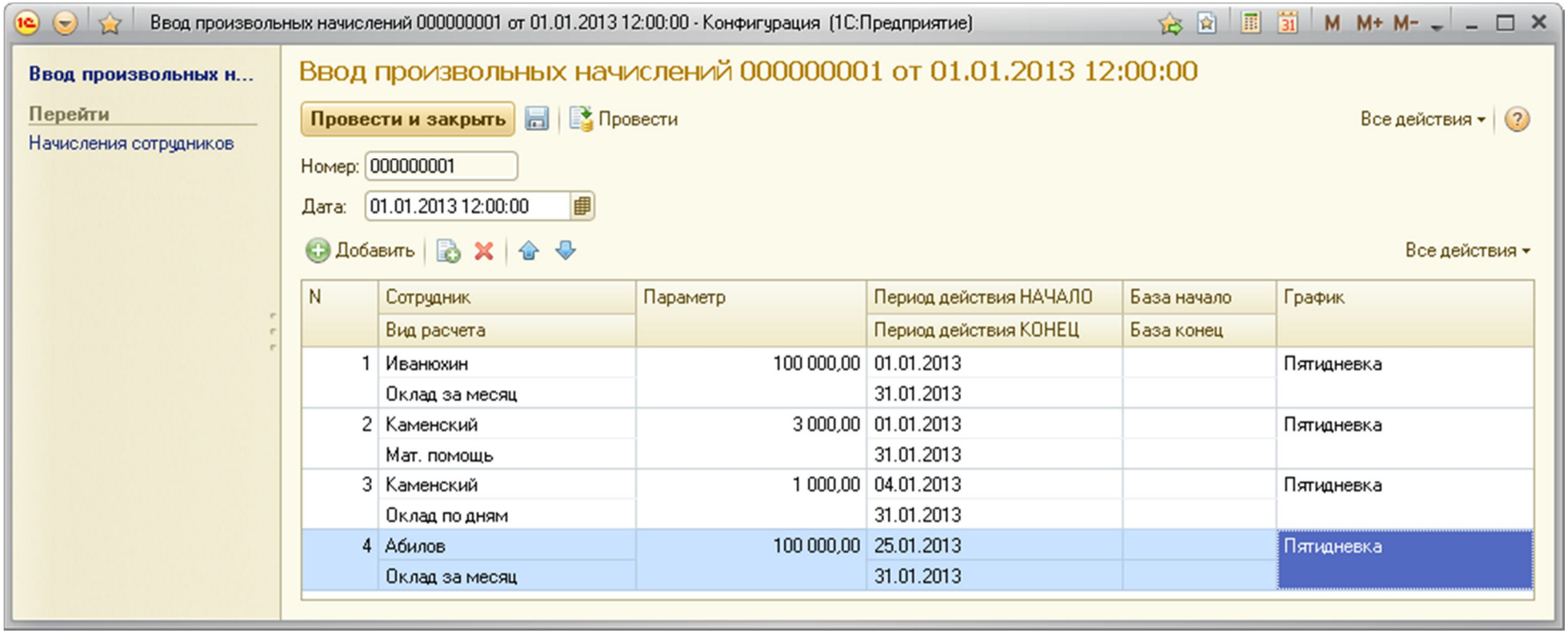
В режиме исполнения:

Поправим внешний вид формы:

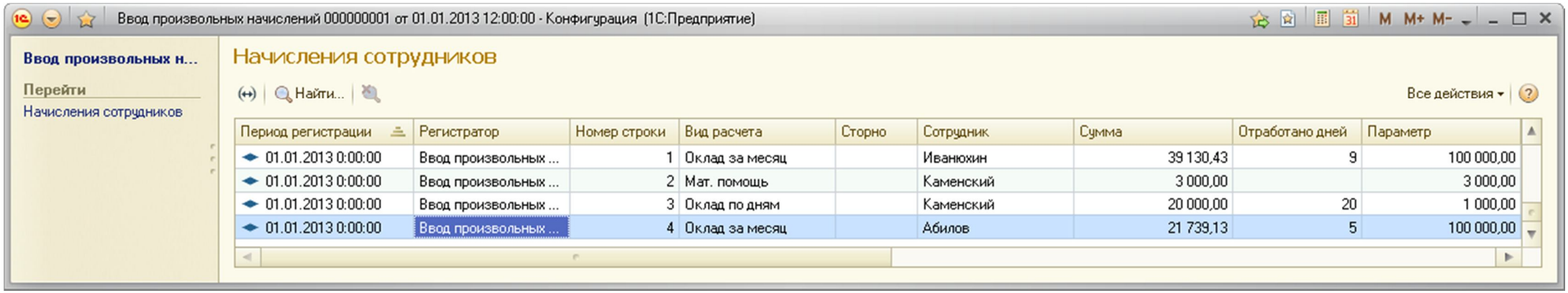
На форме «Все действия – Изменить форму»:

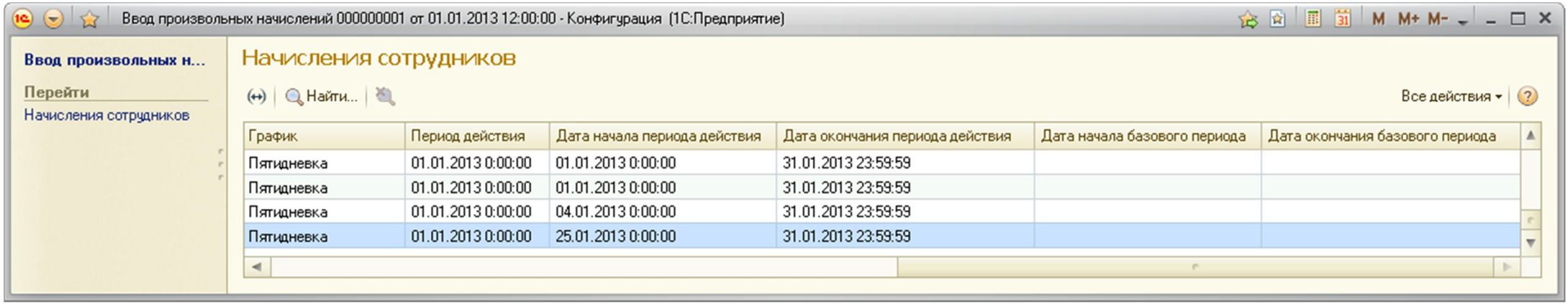


Привяжем График к каждой записи:

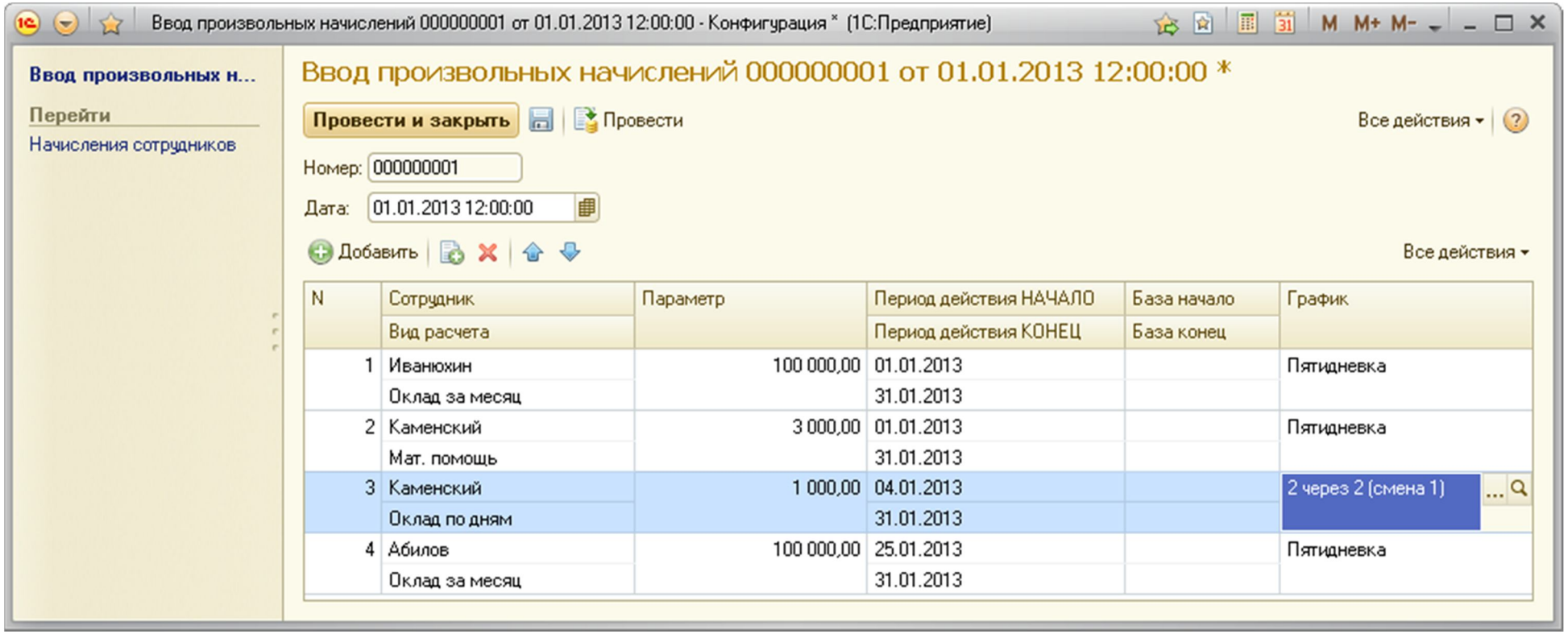


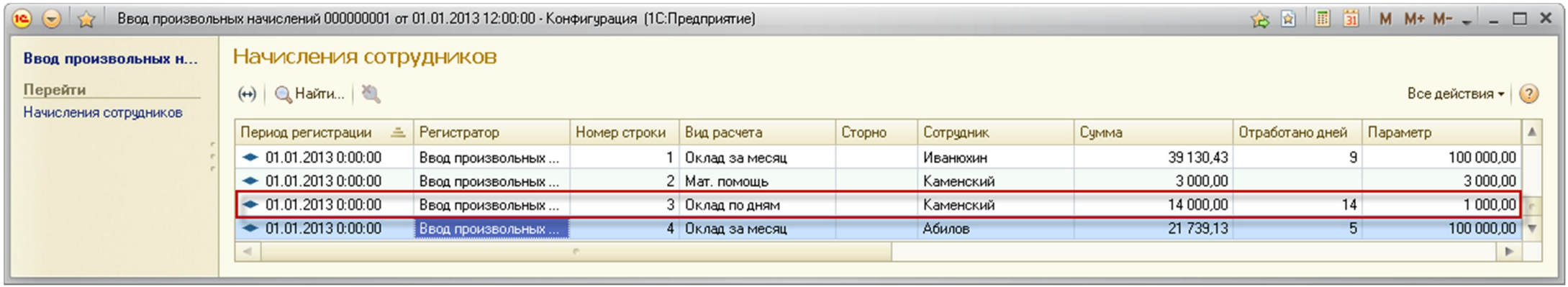
N	Сотрудник / Вид расчета	Параметр	Период действия НАЧАЛО / КОНЕЦ	База начало / конец	График
1	Иванюхин	100 000,00	01.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		
2	Каменский	3 000,00	01.01.2013		Пятидневка
	Мат. помощь		31.01.2013		
3	Каменский	1 000,00	04.01.2013		Пятидневка
	Оклад по дням		31.01.2013		
4	Абилов	100 000,00	25.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		

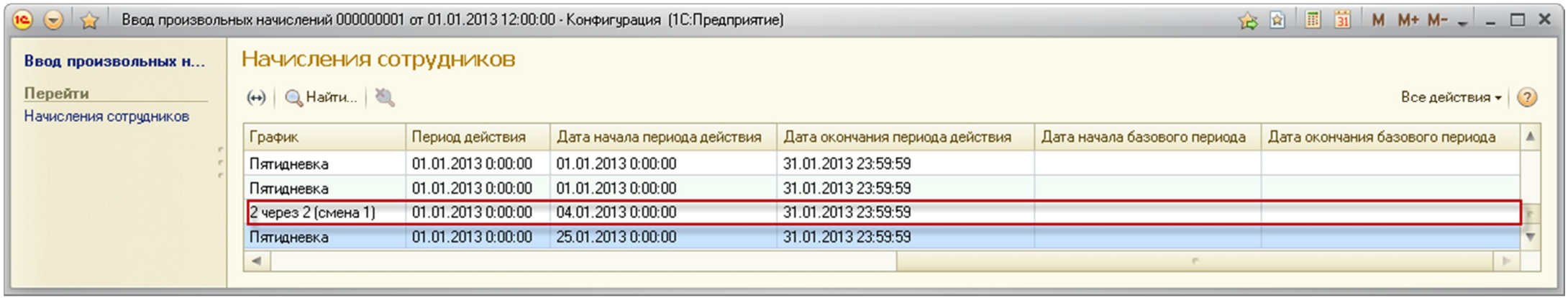

Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации
Регистратор
Номер строки
Вид расчета
Сторно
Сотрудник
Сумма
Отработано дней
Параметр
01.01.2013 0:00:00 Ввод произвольных ... 1 Оклад за месяц Иванюхин 39 130,43 9 100 000,00
01.01.2013 0:00:00 Ввод произвольных ... 2 Мат. помощь Каменский 3 000,00 3 000,00
01.01.2013 0:00:00 Ввод произвольных ... 3 Оклад по дням Каменский 20 000,00 20 1 000,00
01.01.2013 0:00:00 Ввод произвольных ... 4 Оклад за месяц Абилов 21 739,13 5 100 000,00


Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
График
Период действия
Дата начала периода действия
Дата окончания периода действия
Дата начала базового периода
Дата окончания базового периода
Пятидневка 01.01.2013 0:00:00 01.01.2013 0:00:00 31.01.2013 23:59:59
Пятидневка 01.01.2013 0:00:00 01.01.2013 0:00:00 31.01.2013 23:59:59
Пятидневка 01.01.2013 0:00:00 04.01.2013 0:00:00 31.01.2013 23:59:59
Пятидневка 01.01.2013 0:00:00 25.01.2013 0:00:00 31.01.2013 23:59:59

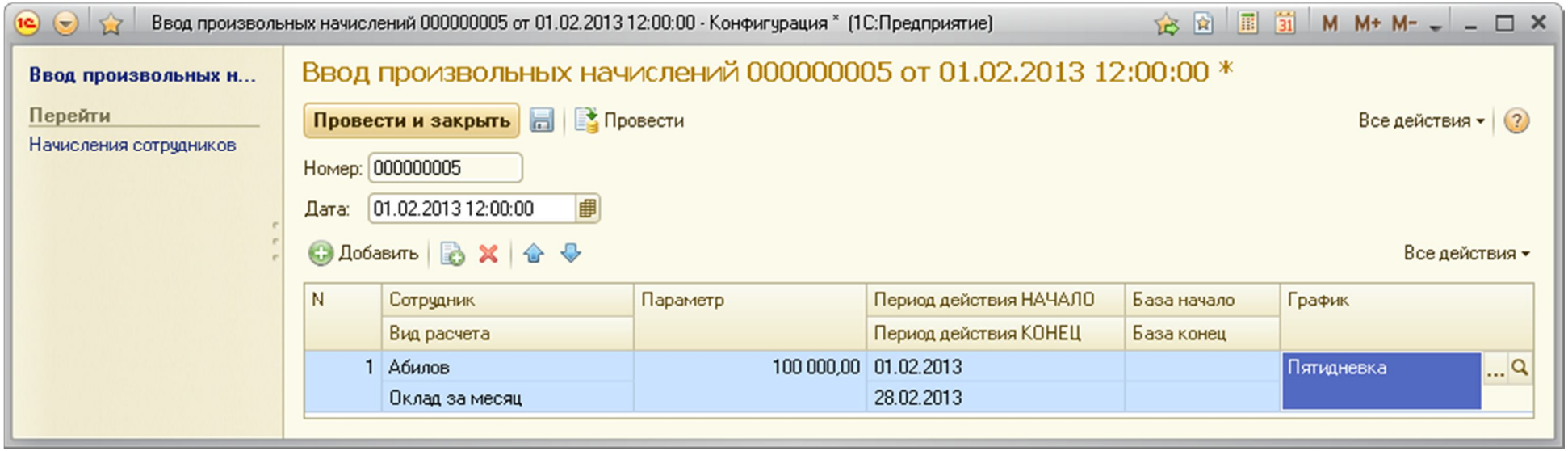
Изменим данные одной записи:

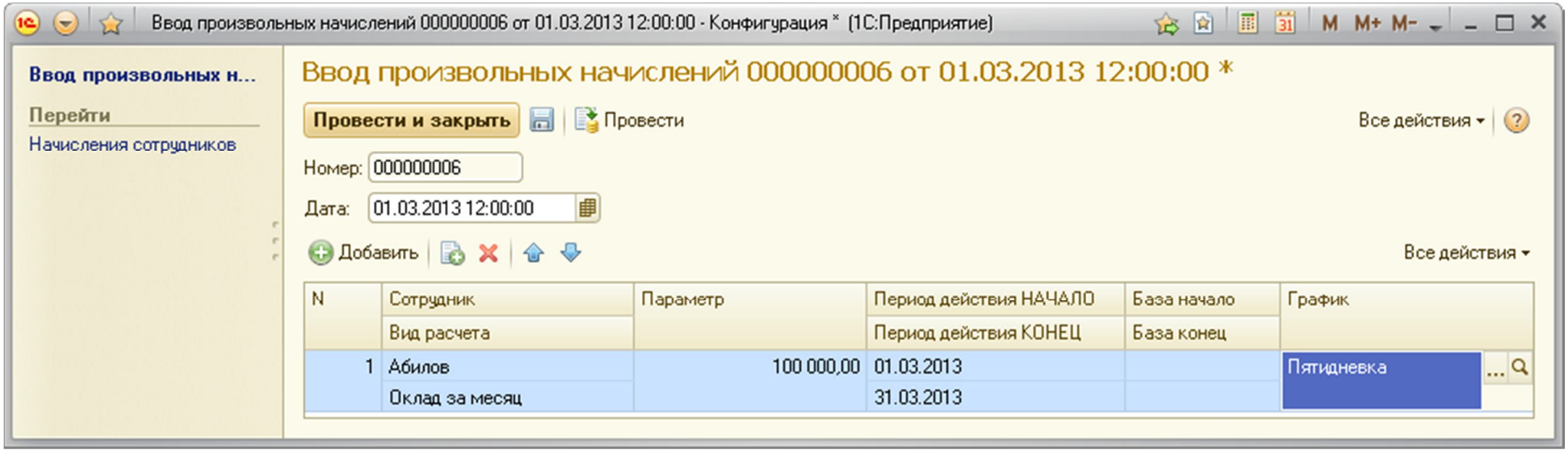


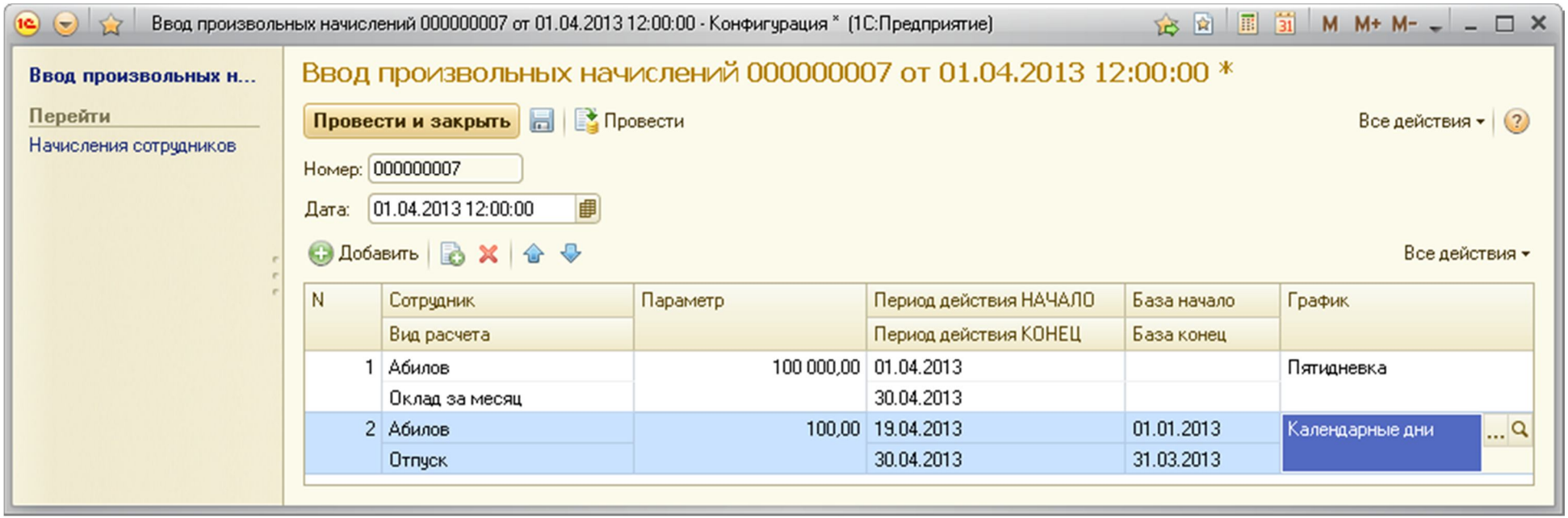

Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации
Регистратор
Номер строки
Вид расчета
Сторно
Сотрудник
Сумма
Отработано дней
Параметр
01.01.2013 0:00:00 Ввод произвольных ... 1 Оклад за месяц Иванюхин 39 130,43 9 100 000,00
01.01.2013 0:00:00 Ввод произвольных ... 2 Мат. помощь Каменский 3 000,00 3 000,00
01.01.2013 0:00:00 Ввод произвольных ... 3 Оклад по дням Каменский 14 000,00 14 1 000,00
01.01.2013 0:00:00 Ввод произвольных ... 4 Оклад за месяц Абилов 21 739,13 5 100 000,00

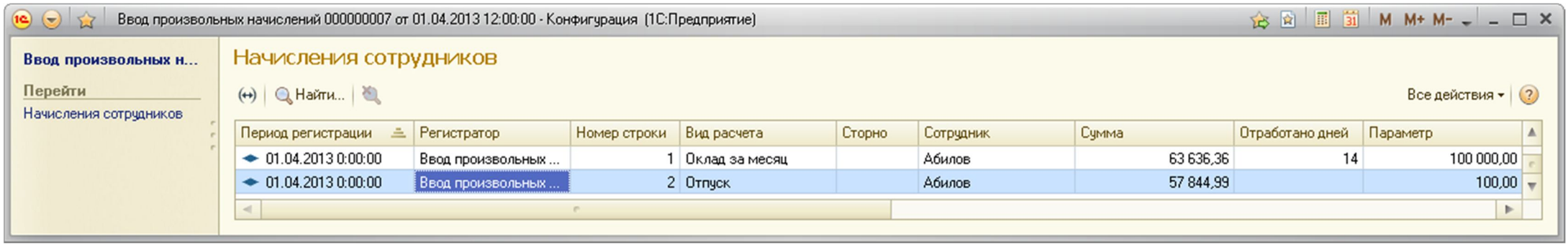

Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
График
Период действия
Дата начала периода действия
Дата окончания периода действия
Дата начала базового периода
Дата окончания базового периода
Пятидневка 01.01.2013 0:00:00 01.01.2013 0:00:00 31.01.2013 23:59:59
Пятидневка 01.01.2013 0:00:00 01.01.2013 0:00:00 31.01.2013 23:59:59
2 через 2 (смена 1) 01.01.2013 0:00:00 04.01.2013 0:00:00 31.01.2013 23:59:59
Пятидневка 01.01.2013 0:00:00 25.01.2013 0:00:00 31.01.2013 23:59:59

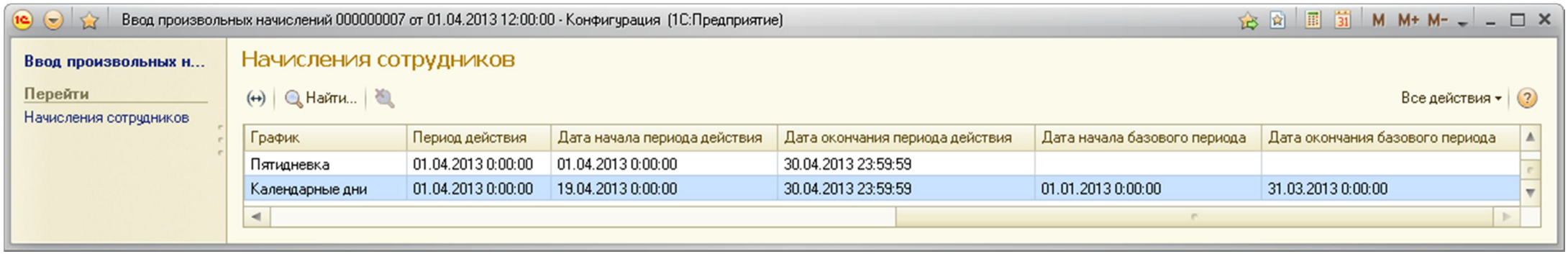
Дополним данные остальных документов и перепроведем их:






Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация * (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 *
Провести и закрыть
Провести
Все действия
Номер: 000000007
Дата: 01.04.2013 12:00:00
Добавить
Все действия
N
Сотрудник
Вид расчета
Параметр
Период действия НАЧАЛО
Период действия КОНЕЦ
База начало
База конец
График
1 Абилов 100 000,00 01.04.2013 Пятидневка
Оклад за месяц 30.04.2013
2 Абилов 100,00 19.04.2013 01.01.2013 Календарные дни
Отпуск 30.04.2013 31.03.2013


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
Период регистрации
Регистратор
Номер строки
Вид расчета
Сторно
Сотрудник
Сумма
Отработано дней
Параметр
01.04.2013 0:00:00 Ввод произвольных ... 1 Оклад за месяц Абилов 63 636,36 14 100 000,00
01.04.2013 0:00:00 Ввод произвольных ... 2 Отпуск Абилов 57 844,99 100,00


Ввод произвольных начислений 000000007 от 01.04.2013 12:00:00 - Конфигурация (1С:Предприятие)
Ввод произвольных н...
Перейти
Начисления сотрудников
Начисления сотрудников
Найти...
Все действия
График
Период действия
Дата начала периода действия
Дата окончания периода действия
Дата начала базового периода
Дата окончания базового периода
Пятидневка 01.04.2013 0:00:00 01.04.2013 0:00:00 30.04.2013 23:59:59
Календарные дни 01.04.2013 0:00:00 19.04.2013 0:00:00 30.04.2013 23:59:59 01.01.2013 0:00:00 31.03.2013 0:00:00

Отпускных стало еще больше, т.к. теперь оплачиваются не только рабочие дни, а все дни отпуска:

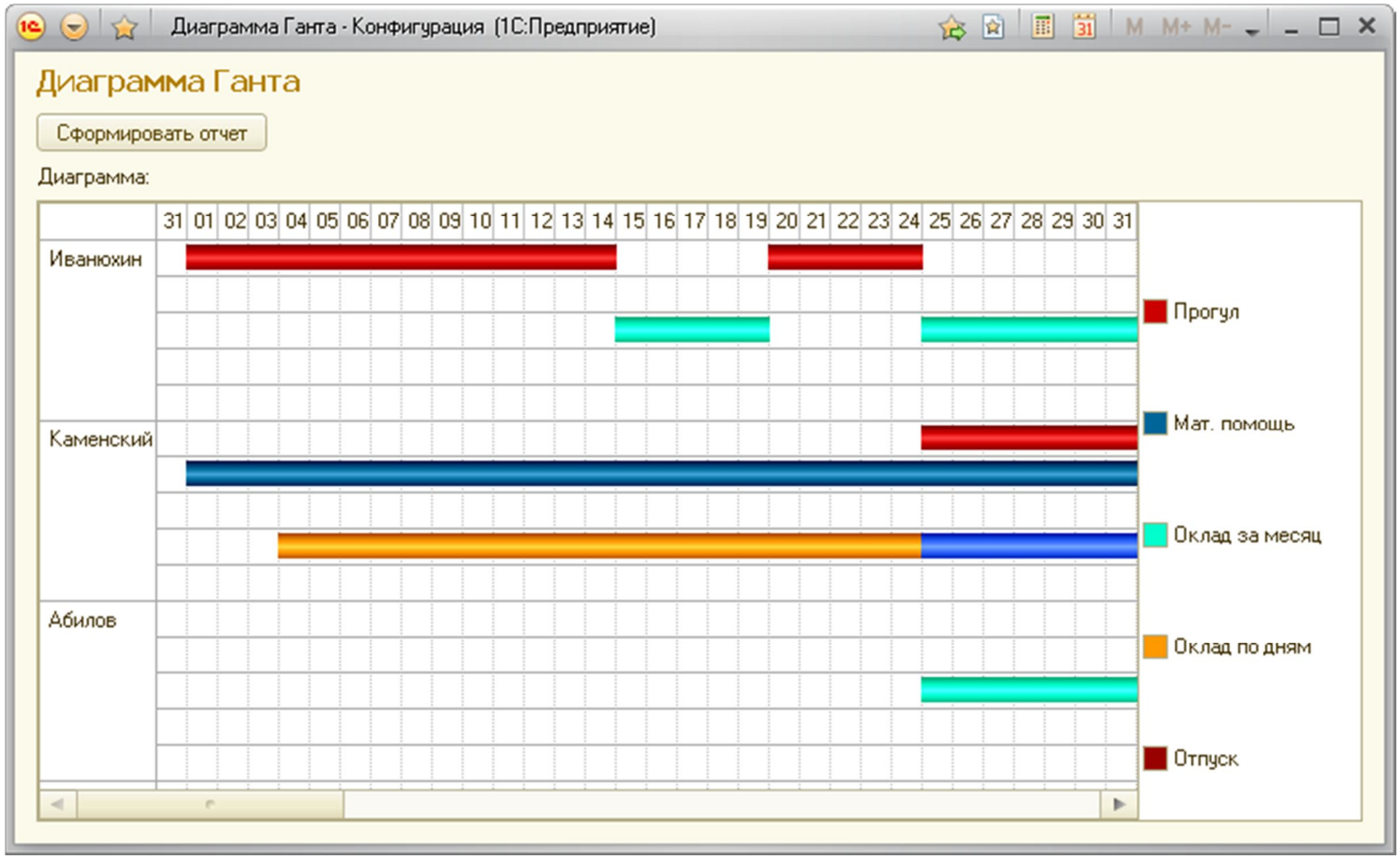
```
ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
    Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
КонецЕсли;
```

Выражение	Значение
ФактДней	12

Занятие 3.29

Оптимизация механизма вытеснения:
В момент завершения транзакции записи в регистре расчета, система анализирует таблицу периода действия помещаемого набора записей и сравнивает ее с записями, уже находящимися в регистре, по периоду действия – система вытеснения.
Вывод: чем больше записей в регистре и чем больше вносимых записей, тем медленнее работа механизма.

Проанализируем диаграмму Ганта:



Например, мат. помощь – не действует целый месяц, не зависит от количества дней начисления. Мат. помощь можно зарегистрировать одним днем.
Вывод: надо убрать из регистра расчета «НачисленияСотрудников» записи, для которых понятие протяженности во времени не имеет смысла. В данном регистре расчета останутся записи, которые вытесняют другие записи и записи, на которых действует вытеснение.
В конфигурации должно быть два регистра расчета:

- с включенным вытеснением.
- с выключенным вытеснением.

Многие свойства Регистра расчета наследуются из Плана видов расчета. Создадим План видов расчета «Доп. начисления» (путем копирования Плана видов расчета «Начисления»):

Свойство	Значение
Имя	ДопНачисления
Синоним	Доп. начисления
Представление объекта	Доп. начисление
Представление списка	Список доп. начислений

Закладка «Данные»:
Реквизит «Способ» - оставляем. С его помощью в общем модуле «Расчеты» задается алгоритм расчета для конкретного способа.
Реквизит «УвеличиваетОтработанныеДни» - удаляем. Использовался для записи Ресурса «ОтработаноДней», который, в свою очередь, использовался для расчета среднедневного заработка сотрудников.

Закладка «Расчет»:
Настройки на закладке «Расчет» влияют на табличные части Плана видов расчета.
Если «Использует период действия = Истина», то в Плане видов расчета у видов расчета появится вкладка «Вытесняющие». Т.е. свойство включает отношения записей по вытеснению.

Свойство	Значение
Использует период действия	Ложь

«Зависимость от базы» - включает закладку «Базовые»:

Свойство	Значение
Зависит по периоду регистрации	Истина
Базовые планы видов расчета	Начисления ДопНачисления

Например, у нас будет вид расчета «Премия», в состав которого входит Мат. помощь и Оклад – они будут находиться в разных Планах видов расчета, т.к. Мат. помощь не зависит от протяженности во времени, а Оклад зависит.

Удалим предопределенные виды расчета.

Создадим регистр расчета «Доп. начисления» - путем копирования регистра расчета «Начисления сотрудников»:

Свойство	Значение
Имя	ДопНачисления
Синоним	Доп. начисления
План видов расчета	ДопНачисления
Период действия	Ложь
Базовый период	Истина
Периодичность	Месяц

Период действия – включает механизм вытеснения, в записях регистра расчета появляются поля:
- ПериодДействияНачало.
- ПериодДействияКонец.

Базовый период – в записях регистра расчета появляются поля:
- БазовыйПериодНачало.
- БазовыйПериодКонец.

Т.е. мы будем получать базу, т.к. нам надо указывать, за какой период производится начисление (например, премия за месяц).

Закладка «Данные» - удаляем Ресурс «ОтработаноДней» и Реквизит «График»
Закладка «Регистраторы»:

Свойство	Значение
Регистратор	ВводПроизвольныхНачислений

Опишем в регистраторе формирование движений по регистру расчета «ДопНачисления»:
В документе «ВводПроизвольныхНачислений» на закладке «Данные» добавим табличную часть «Доп. начисления» - путем копирования табличной части «НачисленияСотрудников»:

Свойство	Значение
Имя	ДопНачисления
Синоним	Доп. начисления

В ТЧ поменяем данные реквизита «ВидРасчета»:

Свойство	Значение
Тип	ПланВидовРасчетаСсылка.ДопНачисления

И удалим реквизиты, относящиеся к периоду действия:

- ПериодДействияНАЧАЛО.
- ПериодДействияКОНЕЦ.
- График.

Модуль объекта – сформируем движения. Отредактируем код:

```
Процедура ОбработкаПроведения(Отказ, Режим)

     Движения.НачисленияСотрудников.Записывать = Истина;
     Движения.НачисленияСотрудников.Очистить();
     Для Каждого Стр Из НачисленияСотрудников Цикл
          Движение = Движения.НачисленияСотрудников.Добавить();
```

```
        Движение.ПериодРегистрации = Дата;
        ЗаполнитьЗначенияСвойств(Движение, Стр);
    КонецЦикла;

    Движения.ДопНачисления.Записывать = Истина;
    Движения.ДопНачисления.Очистить();
    Для Каждого Стр Из ДопНачисления Цикл
        Движение = Движения.ДопНачисления.Добавить();
        Движение.ПериодРегистрации = Дата;

        ЗаполнитьЗначенияСвойств(Движение, Стр);
    КонецЦикла;

    Дополнения = Движения.НачисленияСотрудников.ПолучитьДополнение();

    Для Каждого Стр Из Дополнения Цикл
        Движение = Движения.НачисленияСотрудников.Добавить();
        ЗаполнитьЗначенияСвойств(Движение, Стр);

        Движение.ПериодРегистрации = Стр.ПериодРегистрацииСторно;
        Движение.ПериодДействияНачало = Стр.ПериодДействияНачалоСторно;
        Движение.ПериодДействияКонец = Стр.ПериодДействияКонецСторно;
        Движение.Сторно = Истина
    КонецЦикла;

    Движения.Записать();

    Расчеты.Рассчитать(Ссылка);

КонецПроцедуры
```

Отредактируем расчет – общий модуль «Расчеты» - процедура «Рассчитать»:

Сейчас расчет «заточен» под один регистр. Разделим расчет на две части: «Основную» и «Дополнительную».

Перенесем цикл расчета записей в процедуру «РассчитатьОсновные». Базу также будем рассчитывать внутри этой процедуры. Т.е. теперь в процедуре «Рассчитать» только получаем данные графика.

Теперь в документе «Ввод произвольных начислений» две табличные части, поэтому, если рассчитываются Основные начисления, Доп. начисления могут быть пустыми, и наоборот. Изменим свойства табличных частей документа:

Итого код:

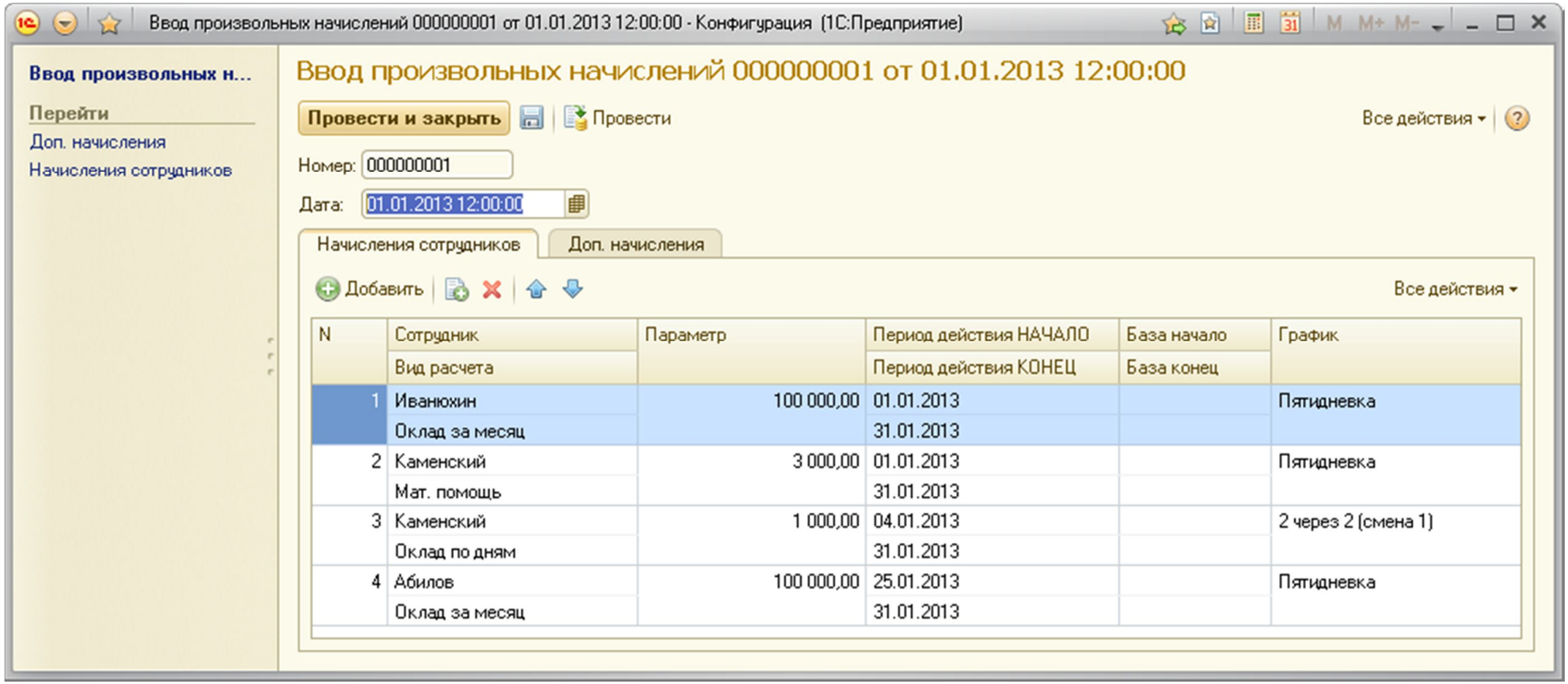
```
Процедура Рассчитать(Регистратор) Экспорт
	// получим копию набора записей регистра
	НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
	НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
	НаборЗаписей.Прочитать();

	ОтборПоРегистратору = Новый Структура;
	ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

	ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
	ОтборПоРегистратору,
	ВидПериодаРегистраРасчета.ФактическийПериодДействия);

	ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
	ОтборПоРегистратору,
	ВидПериодаРегистраРасчета.ПериодДействия);

	//ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
	//ОтборПоРегистратору,
	//ВидПериодаРегистраРасчета.БазовыйПериод);

	РассчитатьОсновные(НаборЗаписей, ТЗ_НормаДней, ТЗ_ФактДней);

КонецПроцедуры
```

```
Процедура РассчитатьОсновные(НаборЗаписей, ТЗ_НормаДней, ТЗ_ФактДней)

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(2);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";
    СписокРесурсов[1] = "НачисленияСотрудников.ОтработаноДней";

    ОтборПоРегистратору = Новый Структура;
    Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    Для Каждого Запись Из НаборЗаписей Цикл
        Индекс = НаборЗаписей.Индекс(Запись);
        База = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;
        ФактДней = ТЗ_ФактДней[Индекс].Признак;
        НормаДней = ТЗ_НормаДней[Индекс].Признак;
        БазаДней = ТЗ_База[Индекс].ОтработаноДней;
        Параметр = Запись.Параметр;
        УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
            Запись.Сумма = Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
            Запись.Сумма = ФактДней * Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
            Запись.Сумма = ФактДней / НормаДней * Параметр;
```

```
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
            Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
        КонецЕсли;

        Если УвеличиваетОтработанныеДни тогда
            Запись.ОтработаноДней = ФактДней;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
            Запись.ОтработаноДней = - Запись.ОтработаноДней;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры

Процедура РассчитатьДополнительные(НаборЗаписей)

КонецПроцедуры
```

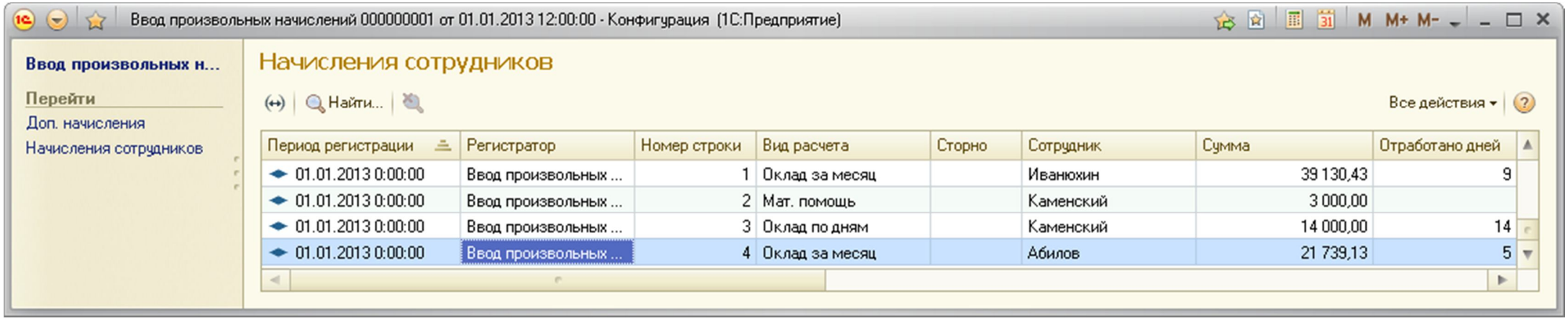
Свойство	Значение
Проверка заполнения	Не проверять

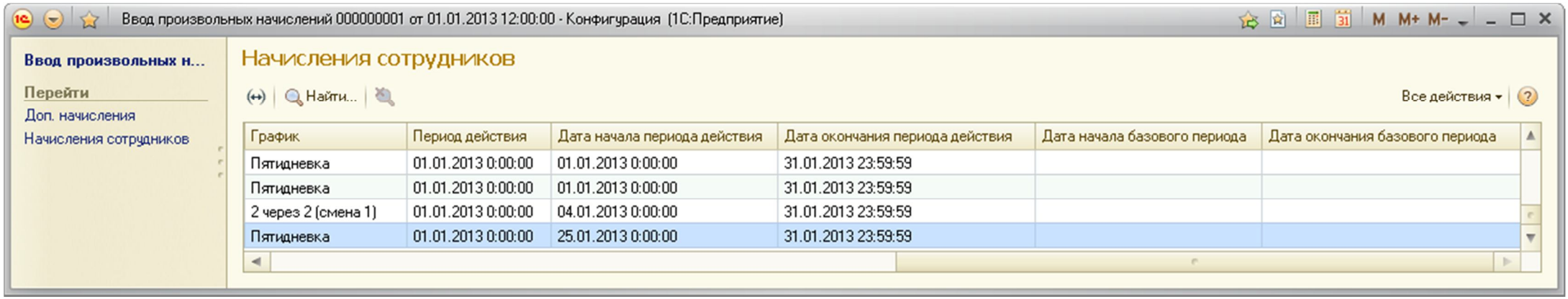
В режиме исполнения:
Проверим, не нарушился ли расчет Основных начислений из-за реконструкции кода:



N	Сотрудник / Вид расчета	Параметр	Период действия НАЧАЛО / КОНЕЦ	База начало / конец	График
1	Иванюхин	100 000,00	01.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		
2	Каменский	3 000,00	01.01.2013		Пятидневка
	Мат. помощь		31.01.2013		
3	Каменский	1 000,00	04.01.2013		2 через 2 (смена 1)
	Оклад по дням		31.01.2013		
4	Абилов	100 000,00	25.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		

Перепроведем документ и проверим результат:





Результат не изменился.

Алгоритм расчета Дополнительных начислений аналогичен алгоритму расчета Основных начислений. Итого код:

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();
```

```
    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ФактическийПериодДействия);

    ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ПериодДействия);

    //ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    //ОтборПоРегистратору,
    //ВидПериодаРегистраРасчета.БазовыйПериод);

    //Получим копию набора записей регистра
    ДопНаборЗаписей = РегистрыРасчета.ДопНачисления.СоздатьНаборЗаписей();
    ДопНаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    ДопНаборЗаписей.Прочитать();

    РассчитатьОсновные(НаборЗаписей, ТЗ_НормаДней, ТЗ_ФактДней);
    РассчитатьДополнительные(ДопНаборЗаписей);

КонецПроцедуры

Процедура РассчитатьОсновные(НаборЗаписей, ТЗ_НормаДней, ТЗ_ФактДней)

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(2);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";
    СписокРесурсов[1] = "НачисленияСотрудников.ОтработаноДней";

    ОтборПоРегистратору = Новый Структура;
    Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
```

```
ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
ОтборПоРегистратору,
СписокРесурсов,
ОтборПоИзмерениям);

Для Каждого Запись Из НаборЗаписей Цикл
     //Индекс = Запись.НомерСтроки - 1;
     Индекс = НаборЗаписей.Индекс(Запись);
     База = ТЗ_База[Индекс].Сумма;
     СпособРасчета = Запись.ВидРасчета.Способ;
     ФактДней = ТЗ_ФактДней[Индекс].Признак;
     НормаДней = ТЗ_НормаДней[Индекс].Признак;
     //БазаДней = ТЗ_БазаДней[Индекс].Признак;
     БазаДней = ТЗ_База[Индекс].ОтработаноДней;
     Параметр = Запись.Параметр;
     УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

     Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
          Запись.Сумма = База * Параметр / 100;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
          Запись.Сумма = Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
          Запись.Сумма = ФактДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
          Запись.Сумма = ФактДней / НормаДней * Параметр;
     ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
          Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
     КонецЕсли;

     Если УвеличиваетОтработанныеДни тогда
          Запись.ОтработаноДней = ФактДней;
     КонецЕсли;

     Если Запись.Сторно Тогда
```

```
            Запись.Сумма = - Запись.Сумма;
            Запись.ОтработаноДней = - Запись.ОтработаноДней;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры

Процедура РассчитатьДополнительные(НаборЗаписей)
    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник,
    |ДопНачисления.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма, ДопНачисления.Сумма";

    ОтборПоРегистратору = Новый Структура;
    Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.ДопНачисления.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    Для Каждого Запись Из НаборЗаписей Цикл
        Индекс        = НаборЗаписей.Индекс(Запись);
        База          = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;
        Параметр      = Запись.Параметр;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
            Запись.Сумма = Параметр;
        КонецЕсли;
```

```
            Если Запись.Сторно Тогда
                Запись.Сумма = - Запись.Сумма;
            КонецЕсли;
        КонецЦикла;
        НаборЗаписей.Записать();
КонецПроцедуры
```

Базу надо получать из двух регистров, т.е. отбор надо установить на два регистра:

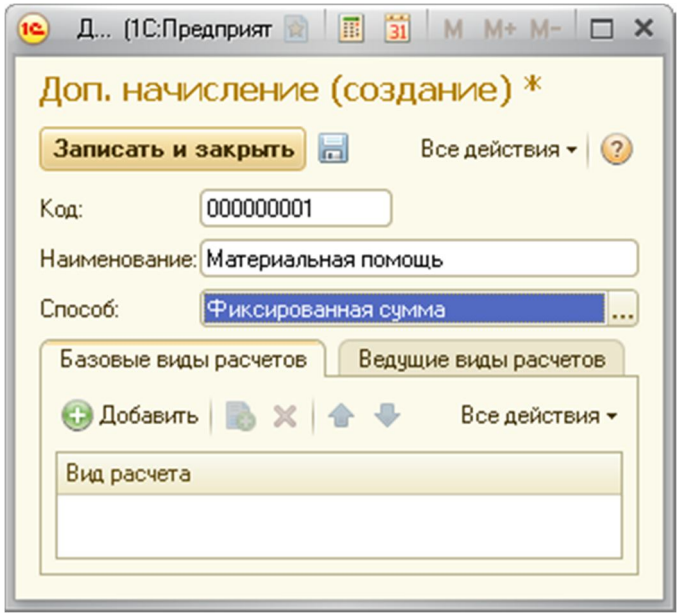
```
ОтборПоИзмерениям = Новый Структура;
ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник,
                           |ДопНачисления.Сотрудник");
```

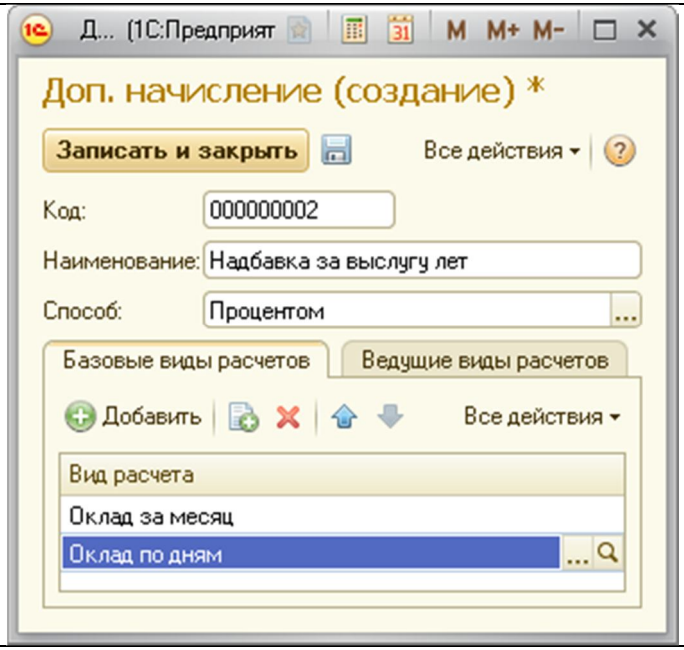
Просуммировать надо Начисления и Доп. начисления

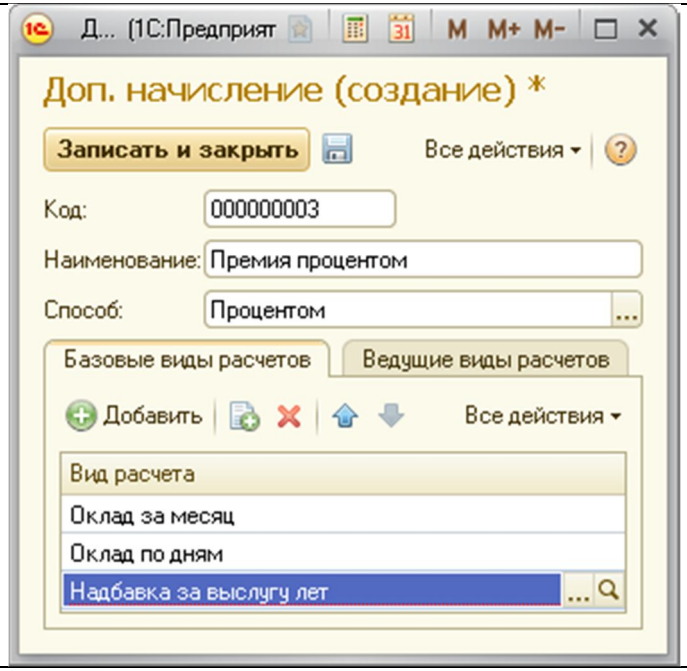
```
СписокРесурсов = Новый Массив(1);
СписокРесурсов[0] = "НачисленияСотрудников.Сумма, ДопНачисления.Сумма";
```

т.к. суммы записаны в виде одного параметра (элемента массива), то Начисления и Доп. начисления просуммируются в одну колонку.

В режиме исполнения:
Создадим Доп. начисления:





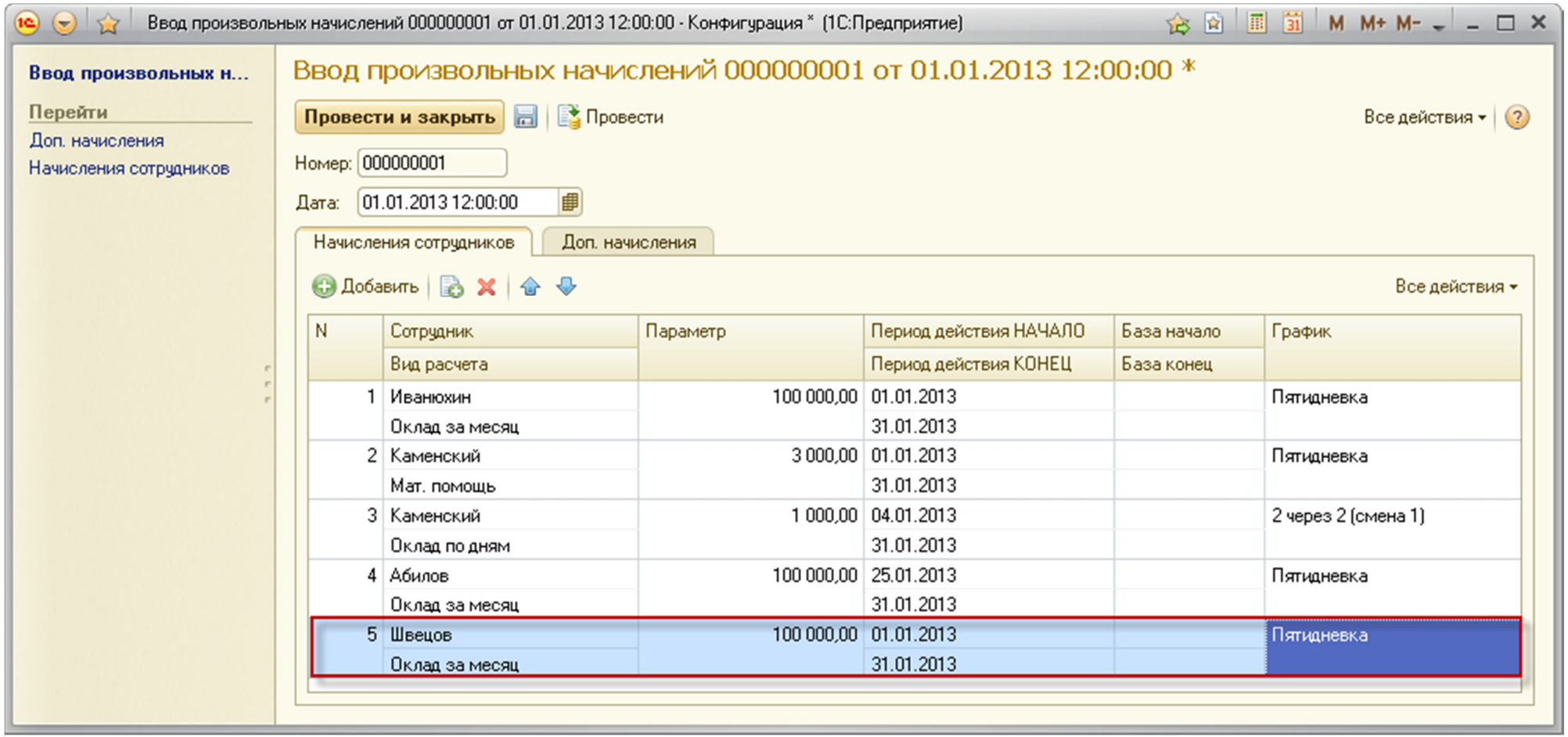


Здесь будут использоваться Базы двух регистров:
Оклад за месяц, Оклад по дням – Начисления.
Надбавка за выслугу лет – Доп. начисления.

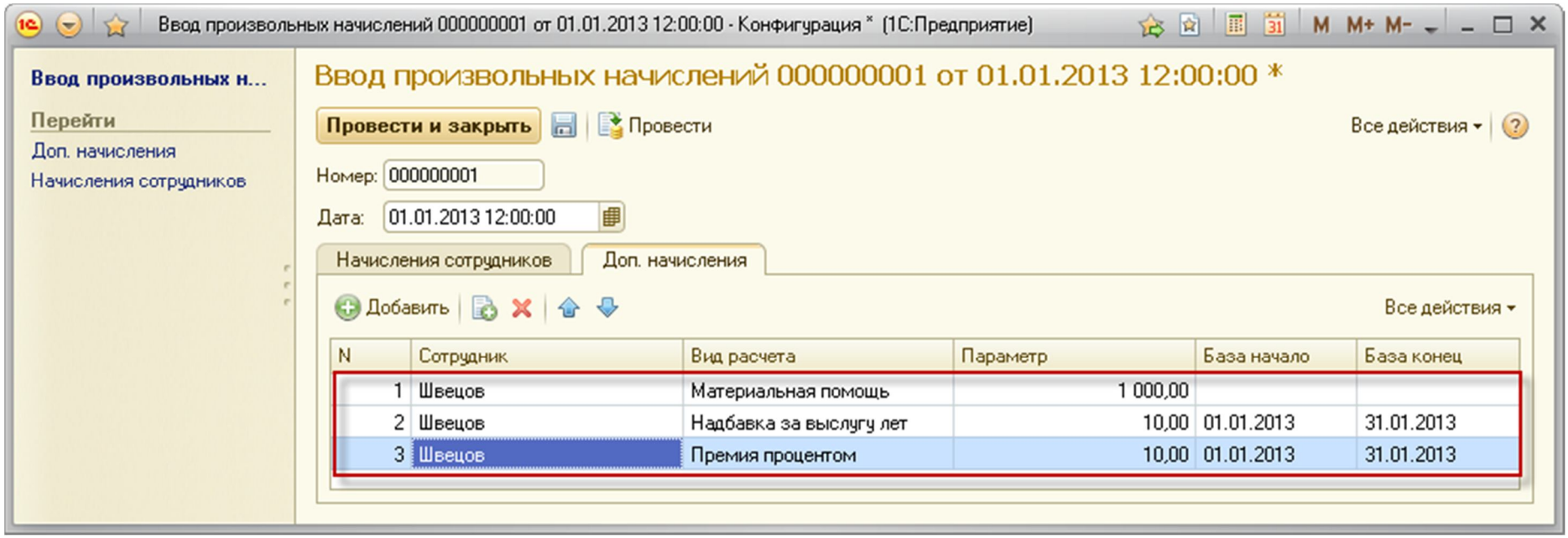
Из Списка начислений удалим ненужные элементы:

- Мат. помощь.
- Надбавка за вредность.

Документы «Ввод произвольных начислений»:
№ 000000001:

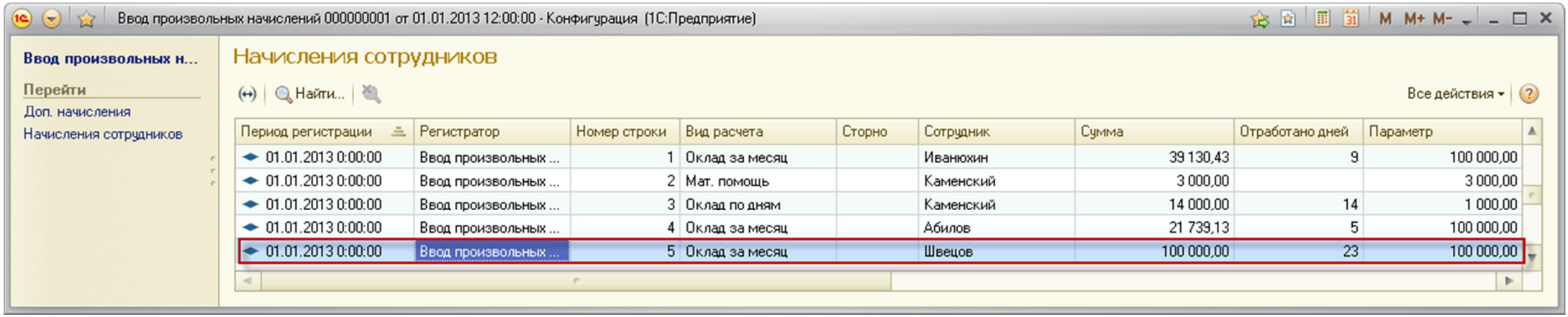


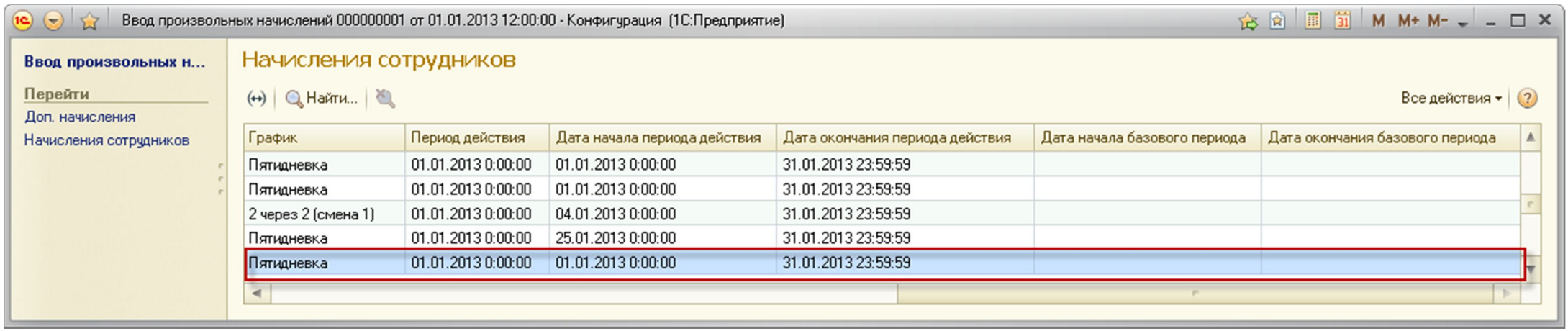
N	Сотрудник / Вид расчета	Параметр	Период действия НАЧАЛО / Период действия КОНЕЦ	База начало / База конец	График
1	Иванюхин	100 000,00	01.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		
2	Каменский	3 000,00	01.01.2013		Пятидневка
	Мат. помощь		31.01.2013		
3	Каменский	1 000,00	04.01.2013		2 через 2 (смена 1)
	Оклад по дням		31.01.2013		
4	Абилов	100 000,00	25.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		
5	Швецов	100 000,00	01.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		


Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 - Конфигурация * (1С:Предприятие)
Ввод произвольных н...
Перейти
Доп. начисления
Начисления сотрудников
Ввод произвольных начислений 000000001 от 01.01.2013 12:00:00 *
Провести и закрыть
Провести
Все действия
Номер: 000000001
Дата: 01.01.2013 12:00:00
Начисления сотрудников
Доп. начисления
Добавить
Все действия
N
Сотрудник
Вид расчета
Параметр
База начало
База конец
1 Швецов Материальная помощь 1 000,00
2 Швецов Надбавка за выслугу лет 10,00 01.01.2013 31.01.2013
3 Швецов Премия процентом 10,00 01.01.2013 31.01.2013

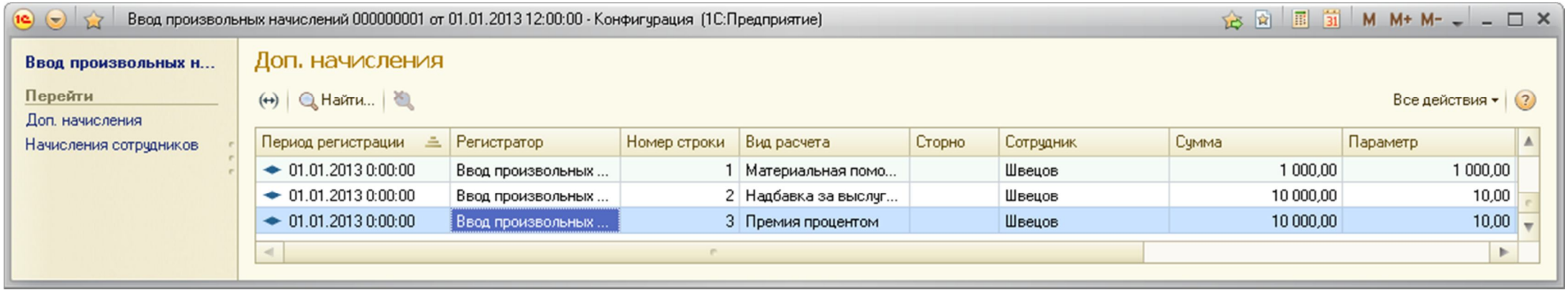
Перепроведем документ:

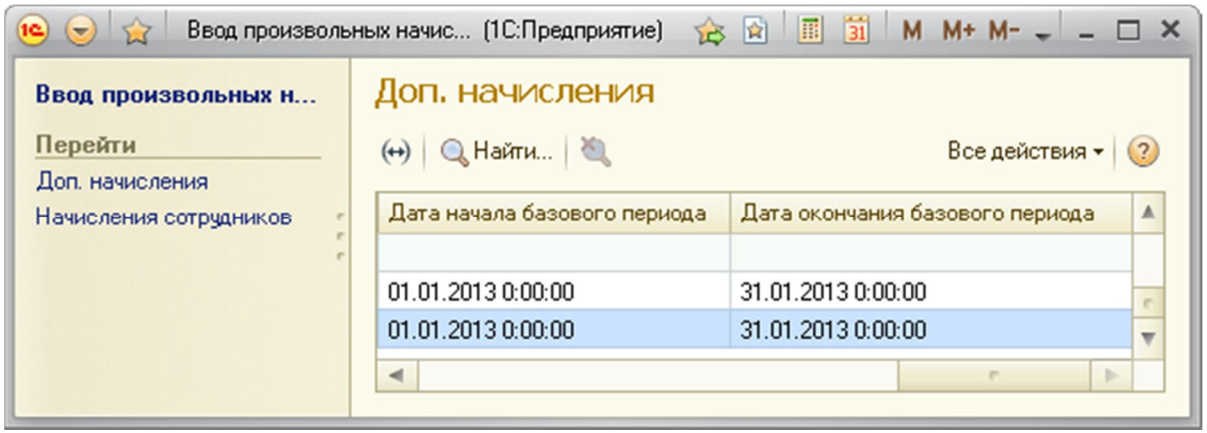
Начисления:





Доп. начисления:





Премия процентом посчиталась неверно, т.к. при ее расчете в базу входит не только Оклад, но и Надбавка за выслугу лет, т.е. Премия должна быть 11000 рублей, а не 10000.

Почему так посчиталось?
База для вычисления Премии процентом сейчас получается методом «ПолучитьБазу()», который строит запрос к таблице регистра расчета. Но Надбавка за выслугу лет рассчитывается в одном документе (в одной транзакции) с Премией, т.е. на момент записи документа записи с Надбавкой за выслугу лет еще нет в Регистре расчета.
Если сейчас ввести Премию процентом другим документом, то она рассчитается корректно.

Надо либо научить пользователя вводить зависящие друг от друга начисления разными документами (что нереально), либо получать Базу из только что рассчитанных записей. А это можно сделать только предварительно записав набор записей в регистр расчета.

Необходимо определить порядок расчета записей. Для этого создадим приоритет расчета начислений (тип - число):

План видов расчета «Начисления» - закладка «Данные» - создадим реквизит «Приоритет»:

Свойство	Значение
Имя	Приоритет
Синоним	Приоритет
Тип	Число 3, 0

Аналогичный реквизит добавим в План видов расчета «ДопНачисления».

Приоритеты будут сквозные для обоих регистров начисления. Общий модуль «Расчеты»:

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    ОтборПоРегистратору = Новый Структура;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ФактическийПериодДействия);

    ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    ОтборПоРегистратору,
    ВидПериодаРегистраРасчета.ПериодДействия);
```

```
    //ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    //ОтборПоРегистратору,
    //ВидПериодаРегистраРасчета.БазовыйПериод);

    //Получим копию набора записей регистра
    ДопНаборЗаписей = РегистрыРасчета.ДопНачисления.СоздатьНаборЗаписей();
    ДопНаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    ДопНаборЗаписей.Прочитать();

    СписокПриоритетов = ПолучитьСписокПриоритетов(Регистратор);

    Для Каждого Приоритет Из СписокПриоритетов Цикл
        РассчитатьОсновные(НаборЗаписей, ТЗ_НормаДней, ТЗ_ФактДней, Приоритет);
        РассчитатьДополнительные(ДопНаборЗаписей, Приоритет);
    КонецЦикла;

КонецПроцедуры

Процедура РассчитатьОсновные(НаборЗаписей, ТЗ_НормаДней, ТЗ_ФактДней, Приоритет)

    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

    СписокРесурсов = Новый Массив(2);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма";
    СписокРесурсов[1] = "НачисленияСотрудников.ОтработаноДней";

    ОтборПоРегистратору = Новый Структура;
    Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);
```

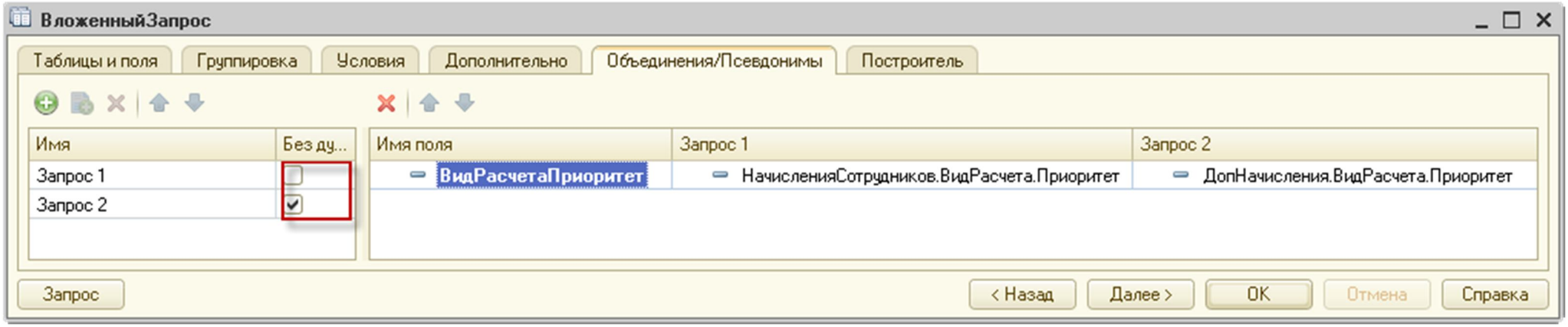
```
Для Каждого Запись Из НаборЗаписей Цикл

    Если Запись.ВидРасчета.Приоритет <> Приоритет Тогда
        Продолжить;
    КонецЕсли;

    //Индекс = Запись.НомерСтроки - 1;
    Индекс = НаборЗаписей.Индекс(Запись);
    База = ТЗ_База[Индекс].Сумма;
    СпособРасчета = Запись.ВидРасчета.Способ;
    ФактДней = ТЗ_ФактДней[Индекс].Признак;
    НормаДней = ТЗ_НормаДней[Индекс].Признак;
    //БазаДней = ТЗ_БазаДней[Индекс].Признак;
    БазаДней = ТЗ_База[Индекс].ОтработаноДней;
    Параметр = Запись.Параметр;
    УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

    Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
        Запись.Сумма = База * Параметр / 100;
    ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
        Запись.Сумма = Параметр;
    ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
        Запись.Сумма = ФактДней * Параметр;
    ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
        Запись.Сумма = ФактДней / НормаДней * Параметр;
    ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
        Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
    КонецЕсли;

    Если УвеличиваетОтработанныеДни тогда
        Запись.ОтработаноДней = ФактДней;
    КонецЕсли;

    Если Запись.Сторно Тогда
        Запись.Сумма = - Запись.Сумма;
        Запись.ОтработаноДней = - Запись.ОтработаноДней;
```

```
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать( , , Ложь);
КонецПроцедуры

Процедура РассчитатьДополнительные(НаборЗаписей, Приоритет)
    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник,
    |ДопНачисления.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма, ДопНачисления.Сумма";

    ОтборПоРегистратору = Новый Структура;
    Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.ДопНачисления.ПолучитьБазу(
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    Для Каждого Запись Из НаборЗаписей Цикл

        Если Запись.ВидРасчета.Приоритет <> Приоритет Тогда
            Продолжить;
        КонецЕсли;

        Индекс        = НаборЗаписей.Индекс(Запись);
        База          = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;
        Параметр      = Запись.Параметр;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
```

```
            Запись.Сумма = Параметр;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры

Функция ПолучитьСписокПриоритетов(Регистратор)
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    ВложенныйЗапрос.Приоритет КАК Приоритет
    |ИЗ
    |    (ВЫБРАТЬ
    |        НачисленияСотрудников.ВидРасчета.Приоритет КАК Приоритет
    |    ИЗ
    |        РегистрРасчета.НачисленияСотрудников КАК НачисленияСотрудников
    |    ГДЕ
    |        НачисленияСотрудников.Регистратор = &Регистратор
    |
    |    ОБЪЕДИНИТЬ
    |
    |    ВЫБРАТЬ
    |        ДопНачисления.ВидРасчета.Приоритет
    |    ИЗ
    |        РегистрРасчета.ДопНачисления КАК ДопНачисления
    |    ГДЕ
    |        ДопНачисления.Регистратор = &Регистратор) КАК ВложенныйЗапрос
    |
    |УПОРЯДОЧИТЬ ПО
    |    Приоритет";
    Запрос.УстановитьПараметр("Регистратор", Регистратор);
    РезультатЗапроса = Запрос.Выполнить().Выгрузить();
```

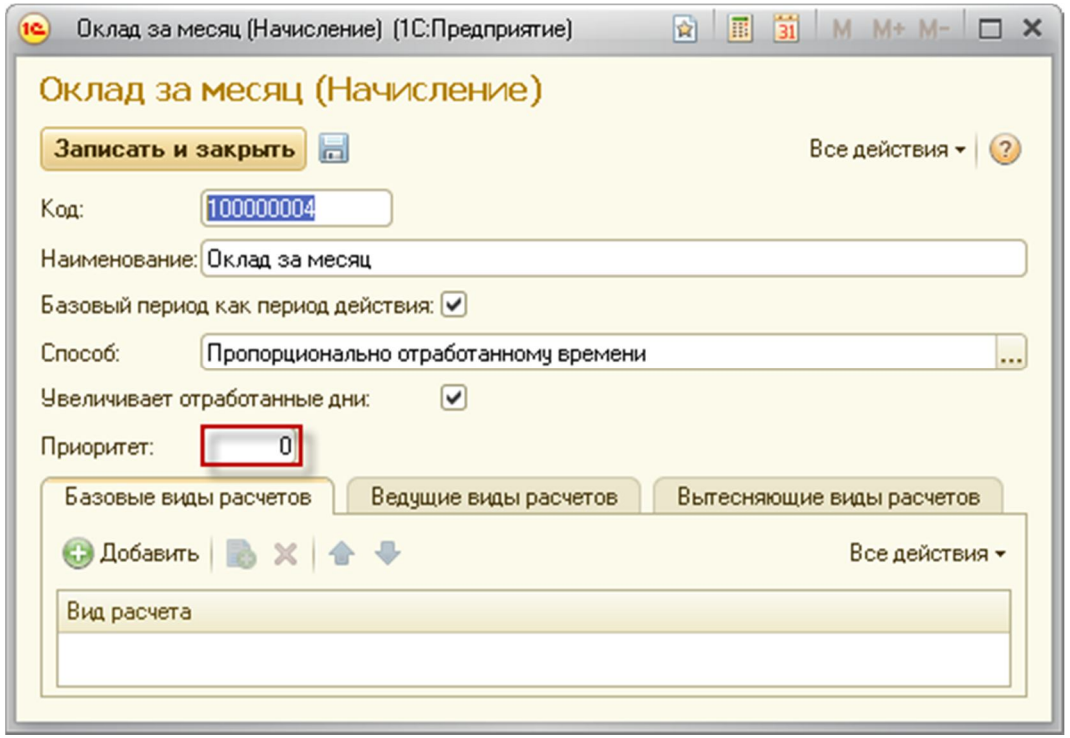
```
    Возврат РезультатЗапроса.ВыгрузитьКолонку("Приоритет");

КонецФункции
```

Пояснение кода:
До запуска процедур расчета получим список Приоритетов. Процедуры расчета будем запускать для каждого Приоритета.
Функция «ПолучитьСписокПриоритетов()»:
В конструкторе запроса галочка «Без дублей» убирает слово «ВСЕ» после слова «ОБЪЕДИНИТЬ» 0 для получения только уникальных значений из результатов объединяемых запросов.



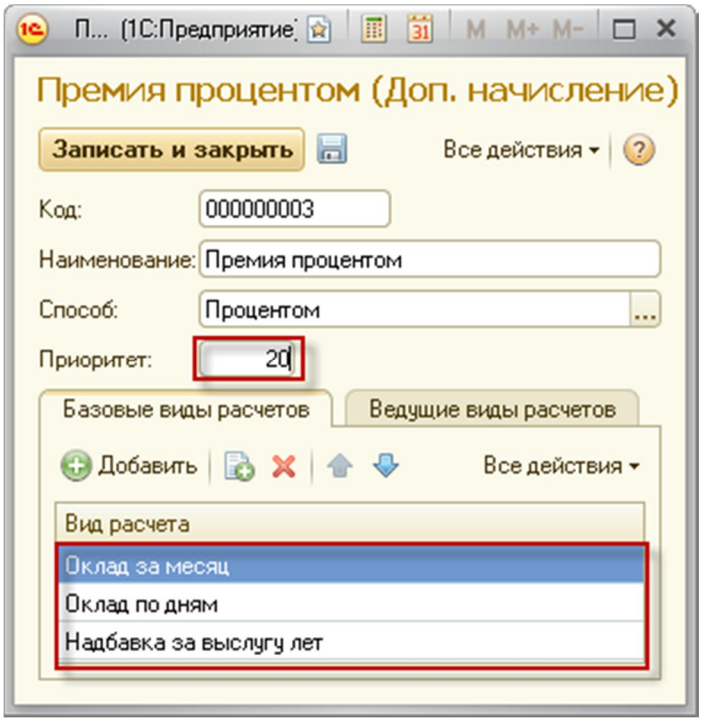
Вложенный запрос нужен для применения Сортировки к результату.
Функция вернет МАССИВ со списком Приоритетов.
В процедурах расчета, после входа в цикл по Наборам записей выполним отбор по Приоритету

В режиме исполнения:
Определим Приоритеты для Видов расчета:



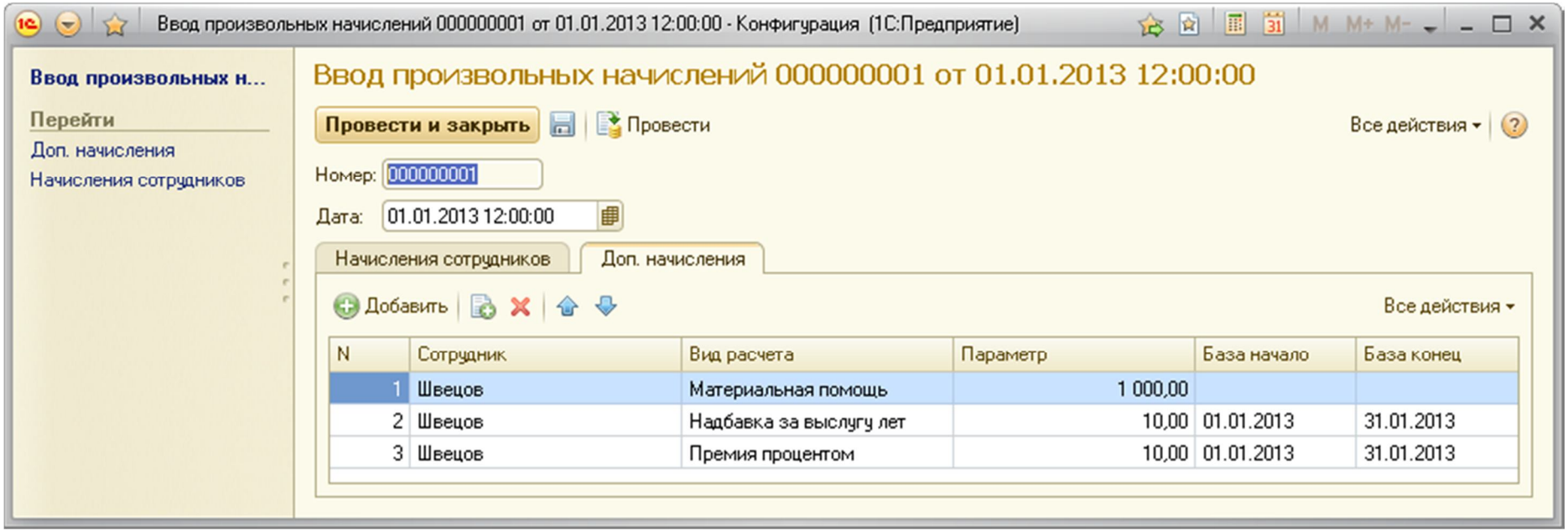
Основные начисления:
Оклад за месяц – 0.
Оклад по дням – 0.
Отпуск – 0. Т.к. Базой для Отпуска являются Оклад за месяц и Оклад по дням за ПРОШЛЫЕ периоды, то Приоритет расчета отпуска идет наравне с Окладами.

Доп. начисления:
Материальная помощь – 0.
Надбавка за выслугу лет – 10. Должна рассчитываться после Окладов
Премия процентом – 20. Должна рассчитываться по результатам расчета Окладов и Надбавки за выслугу лет:



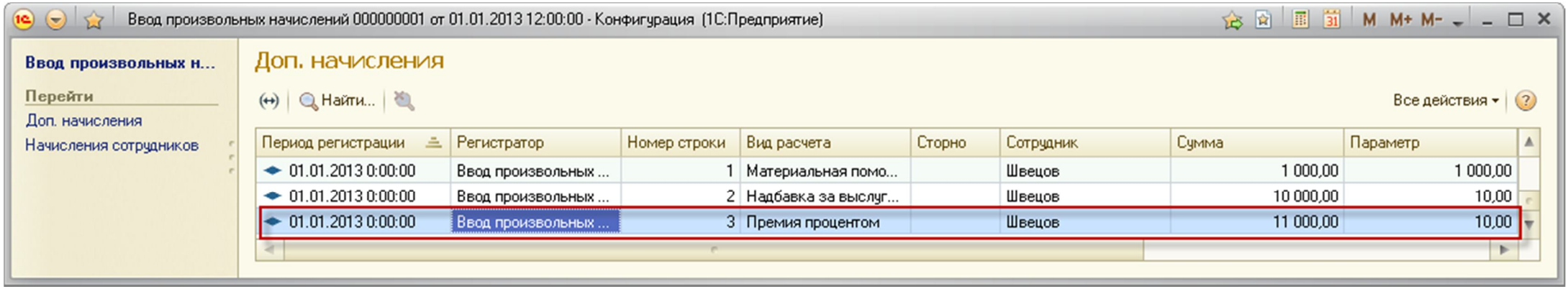
Шаг сделали большим, чтобы оставить диапазон на случай, если появятся новые начисления.

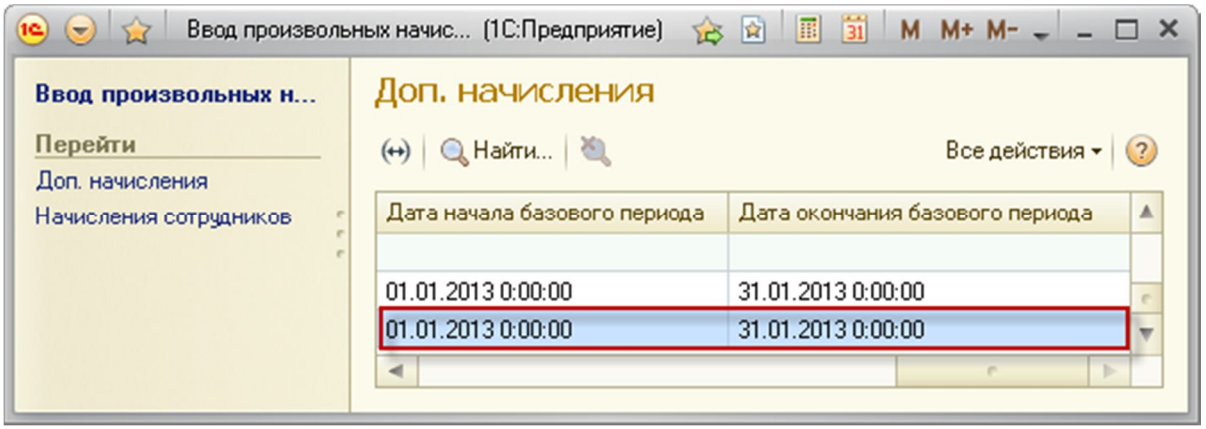
Перепроведем документ «Ввод произвольных начислений» № 000000001:



N	Сотрудник	Вид расчета	Параметр	База начало	База конец
1	Швецов	Материальная помощь	1 000,00		
2	Швецов	Надбавка за выслугу лет	10,00	01.01.2013	31.01.2013
3	Швецов	Премия процентом	10,00	01.01.2013	31.01.2013

и проверим результат:





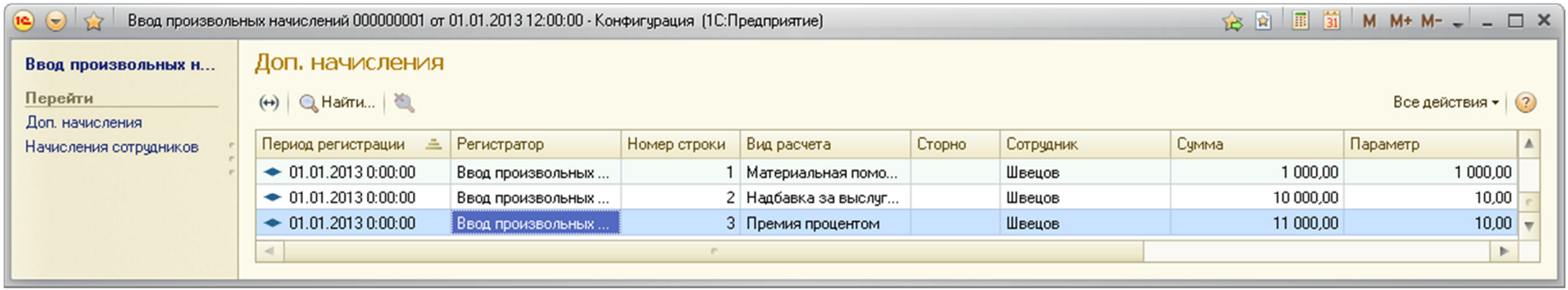
Теперь Премия процентом рассчиталась верно.

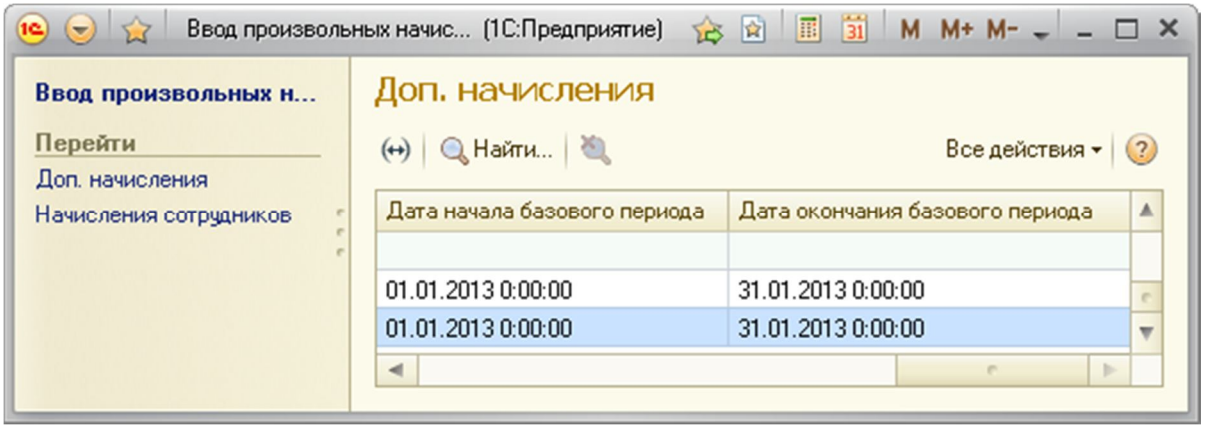
Можно оптимизировать процедуру записи в регистр расчета:

```
Записать(<Замещать>, <ТолькоЗапись>, <ЗаписьФактическогоПериодаДействия>, <ЗаписьПерерасчетов>)
```

Фактический период действия записи у нас уже рассчитан, т.е. механизм вытеснения уже задействован и после расчета сумм данные вытеснения уже не меняются. Например, отпуск уменьшил рабочие дни на 10 и их стало 15. После расчета сумм, рабочих дней не станет еще меньше. Запись уже есть в регистре, а в цикле рассчитывается результат записи. Т.е. в цикле нам не надо каждый раз записывать информацию о фактическом периоде действия:

Итого:
Сейчас неоптимально считается База. Например процедура «РассчитатьДополнительные()». Внутри цикла мы получаем Базу для всех трех записей (документ «Ввод произвольных начислений» № 000000001), а пользуемся только одной.





Но с помощью объектной модели нет возможности рассчитать Базу только для одной записи. В данной ситуации надо использовать Запрос.

Занятие 3.30

Оптимизация работы с Регистрами расчета

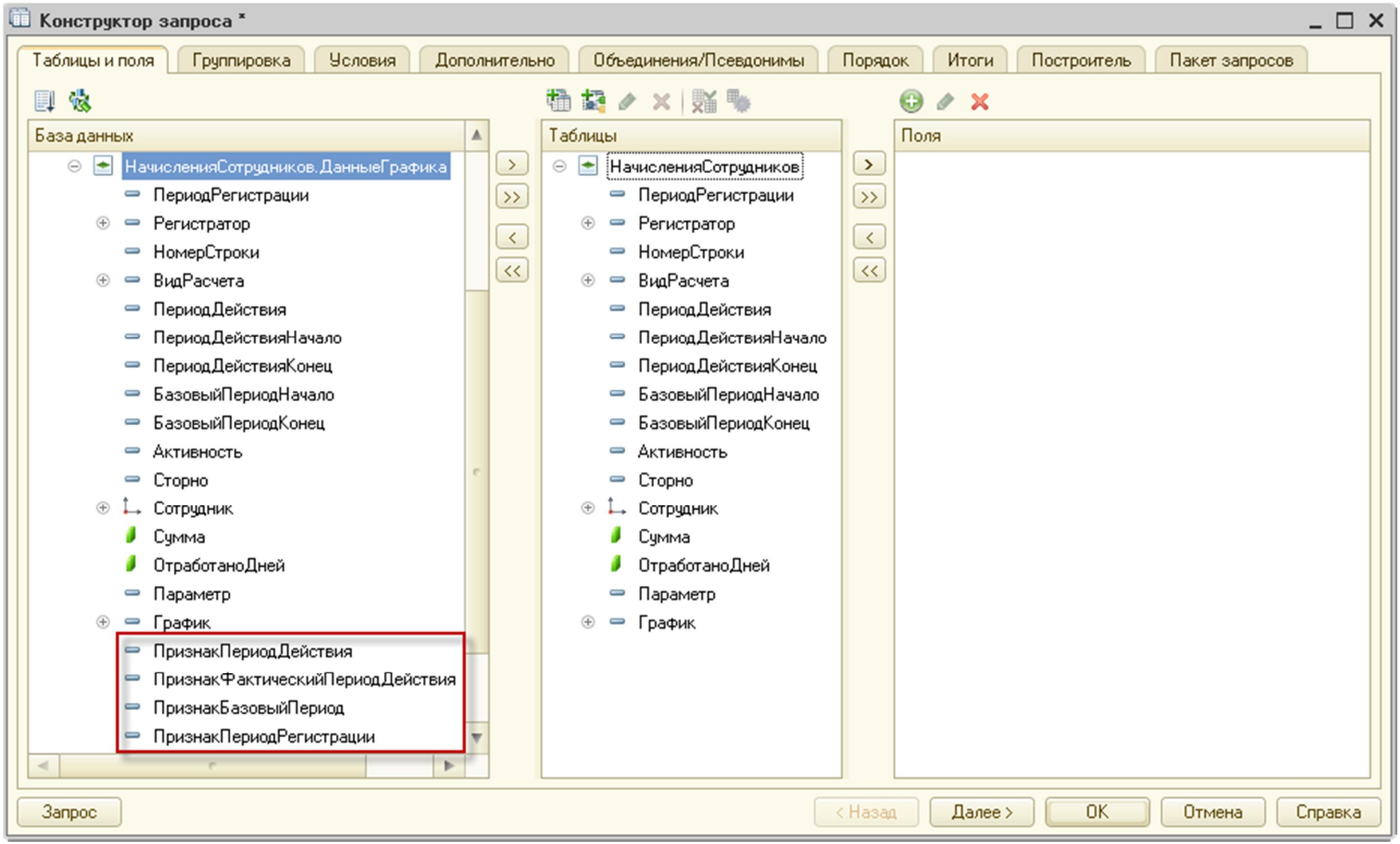
Общий модуль «Расчеты» - процедура «Рассчитать»:

Данные по Норме и Факту дней, а также дополнительные данные, которые нам понадобятся для расчета записей регистра, получим с помощью запроса:

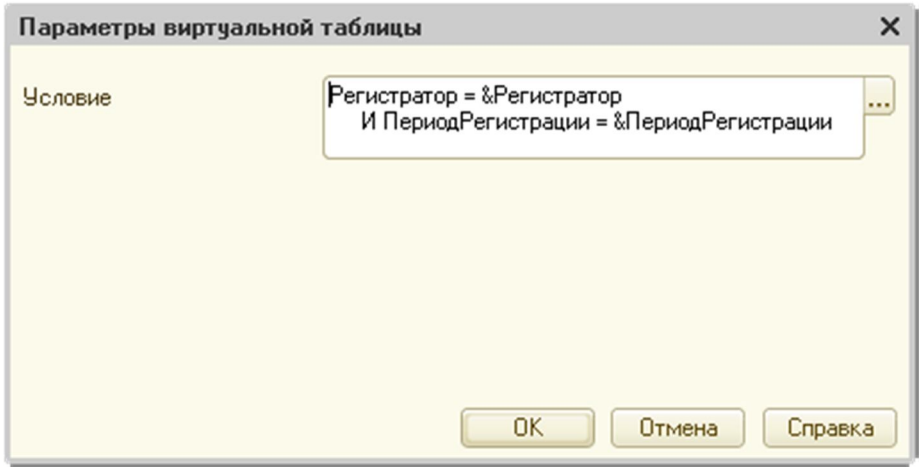
Чтобы между этим запросом и запросом, который будет получать Базу, можно было передавать данные, получим **Менеджер временных таблиц**. В запросе нам надо получить то же самое, что мы получали при расчете Основных начислений:

Можно обратиться к реальной таблице или таблице «ДанныеГрафика». Они практически идентичны, но у таблицы «ДанныеГрафика» есть дополнительные поля. Они автоматически рассчитываются системой и представляют собой Количество отработанных дней за указанный период:

- Действия.
- Фактический.
- Базовый.
- Регистрации.


Конструктор запроса *
Таблицы и поля
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Порядок
Итоги
Построитель
Пакет запросов
База данных
НачисленияСотрудников.ДанныеГрафика
ПериодРегистрации
Регистратор
НомерСтроки
ВидРасчета
ПериодДействия
ПериодДействияНачало
ПериодДействияКонец
БазовыйПериодНачало
БазовыйПериодКонец
Активность
Сторно
Сотрудник
Сумма
ОтработаноДней
Параметр
График
ПризнакПериодДействия
ПризнакФактическийПериодДействия
ПризнакБазовыйПериод
ПризнакПериодРегистрации
Таблицы
НачисленияСотрудников
ПериодРегистрации
Регистратор
НомерСтроки
ВидРасчета
ПериодДействия
ПериодДействияНачало
ПериодДействияКонец
БазовыйПериодНачало
БазовыйПериодКонец
Активность
Сторно
Сотрудник
Сумма
ОтработаноДней
Параметр
График
Поля
Запрос
< Назад
Далее >
OK
Отмена
Справка

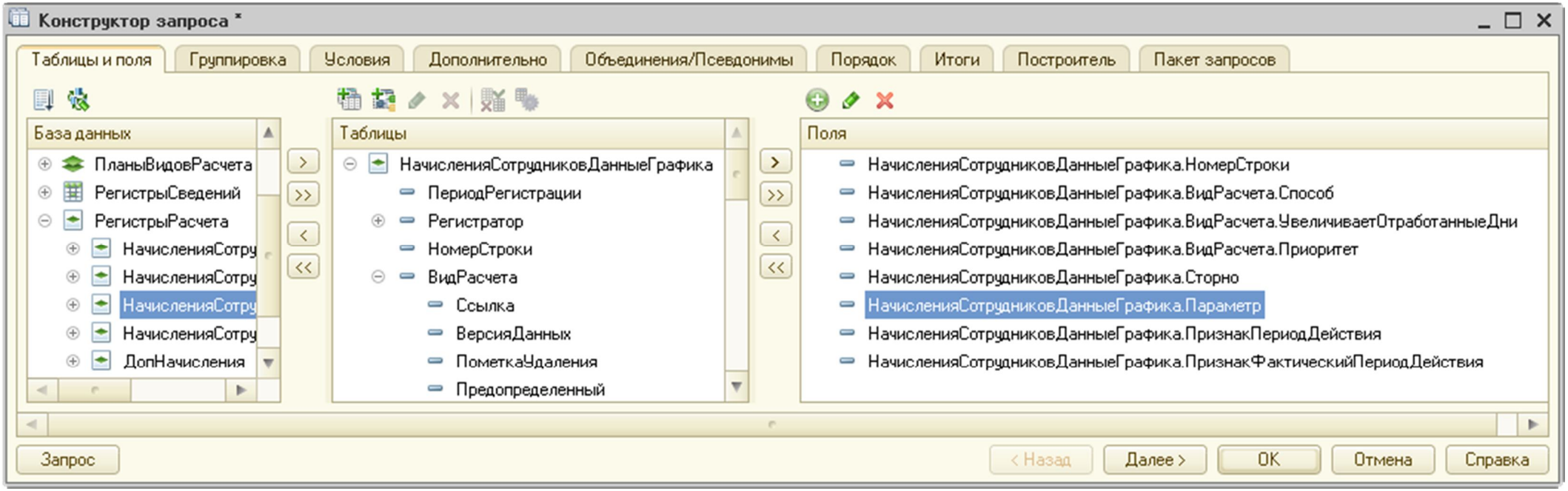
Установим параметры виртуальной таблицы «НачисленияСотрудниковДанныеГрафика»:



В параметрах установим отбор по документу-регистратору и по периоду регистрации. Отбор по периоду регистрации не повлияет на результат выборки, но увеличит скорость отработки, т.к. Индекс таблицы Данные графика составной:

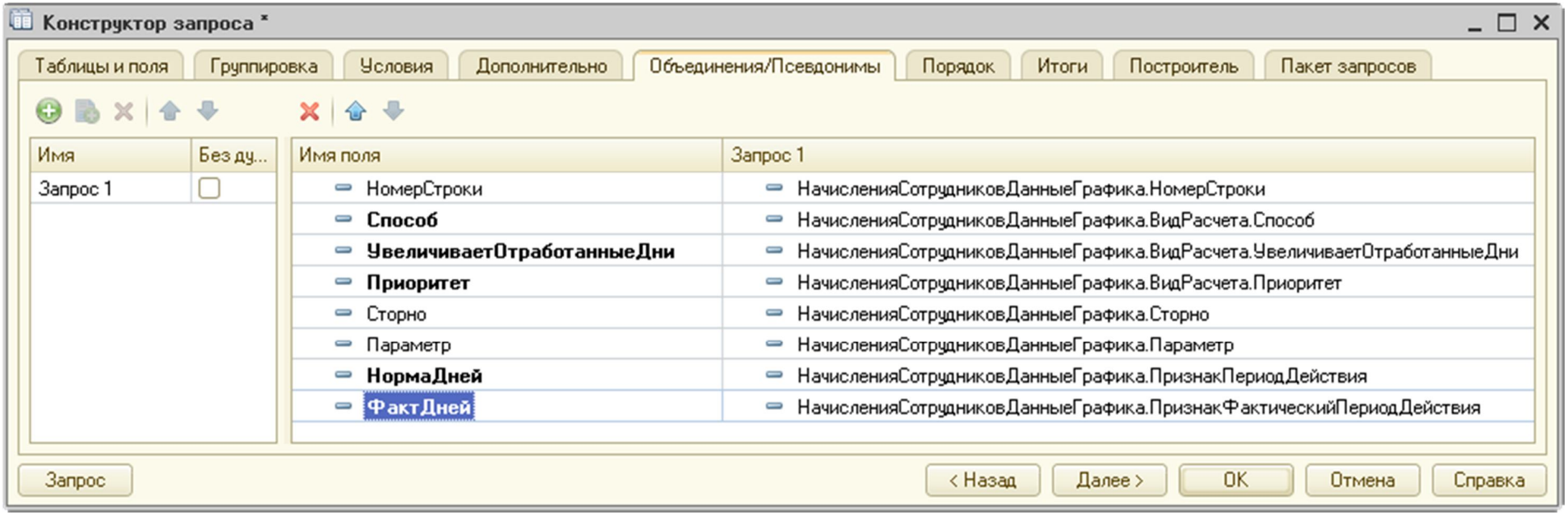
- Документ + ПериодРегистрации.

Поля:



НомерСтроки – нужен для получения результата записи – суммы.
Приоритет – нужен для отбора данных при получении таблицы Базы (ограничение данных Базы только нужным Приоритетом).
ПризнакПериодДействия – норма дней.
ПризнакФактическийПериодДействия – факт дней.

Назначим псевдонимы:



Итого процедура «Рассчитать»:

```
Процедура Рассчитать(Регистратор) Экспорт
	// получим копию набора записей регистра
	НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
	НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
	НаборЗаписей.Прочитать();

	Запрос = Новый Запрос;
	МВТ = Новый МенеджерВременныхТаблиц;
	Запрос.МенеджерВременныхТаблиц = МВТ;
	Запрос.Текст =
	"ВЫБРАТЬ
	|	НачисленияСотрудниковДанныеГрафика.НомерСтроки,
	|	НачисленияСотрудниковДанныеГрафика.ВидРасчета.Способ КАК Способ,
	|	НачисленияСотрудниковДанныеГрафика.ВидРасчета.УвеличиваетОтработанныеДни
		КАК УвеличиваетОтработанныеДни,
```

```
|    НачисленияСотрудниковДанныеГрафика.ВидРасчета.Приоритет КАК Приоритет,
|    НачисленияСотрудниковДанныеГрафика.Сторно,
|    НачисленияСотрудниковДанныеГрафика.Параметр,
|    НачисленияСотрудниковДанныеГрафика.ПризнакПериодДействия КАК НормаДней,
|    НачисленияСотрудниковДанныеГрафика.ПризнакФактическийПериодДействия КАК ФактДней
|ПОМЕСТИТЬ ДанныеГрафика
|ИЗ
|    РегистрРасчета.НачисленияСотрудников.ДанныеГрафика(
|                Регистратор = &Регистратор
|                    И ПериодРегистрации = &ПериодРегистрации) КАК НачисленияСотрудниковДанныеГрафика";
Запрос.УстановитьПараметр("Регистратор", Регистратор);
Запрос.УстановитьПараметр("ПериодРегистрации", НачалоМесяца(Регистратор.Дата));
Запрос.Выполнить();

//Получим копию набора записей регистра
ДопНаборЗаписей = РегистрыРасчета.ДопНачисления.СоздатьНаборЗаписей();
ДопНаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
ДопНаборЗаписей.Прочитать();

СписокПриоритетов = ПолучитьСписокПриоритетов(Регистратор);

Для Каждого Приоритет Из СписокПриоритетов Цикл
    РассчитатьОсновные(НаборЗаписей, МВТ, Приоритет);
    РассчитатьДополнительные(ДопНаборЗаписей, Приоритет);
КонецЦикла;

КонецПроцедуры
```

Параметр «ПериодРегистрации» - берем дату документа-регистратора. Значение параметра должно быть равно первому числу периода, за который будут вычисляться данные.
Результат запроса поместим во временную таблицу «ДанныеГрафика»:

```
|ПОМЕСТИТЬ ДанныеГрафика
```

Поместим объект с результатом запроса (МВТ) в память – выполним запрос:

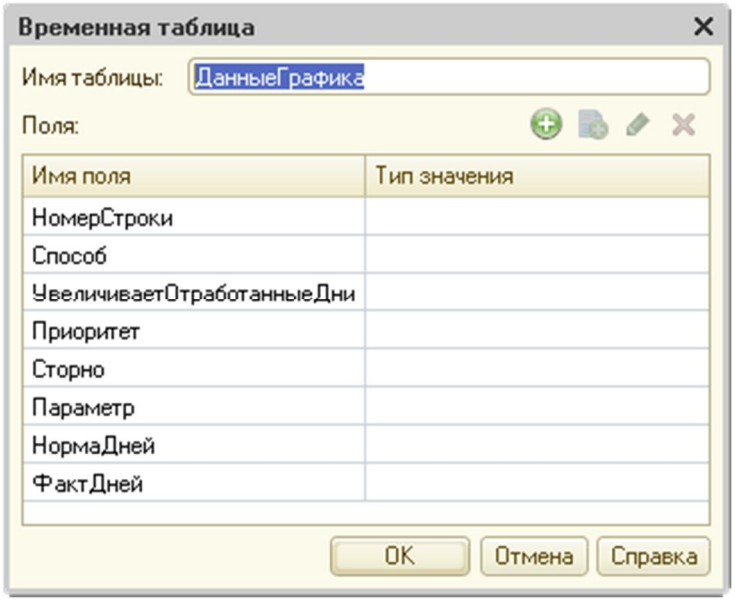
```
Запрос.Выполнить();
```

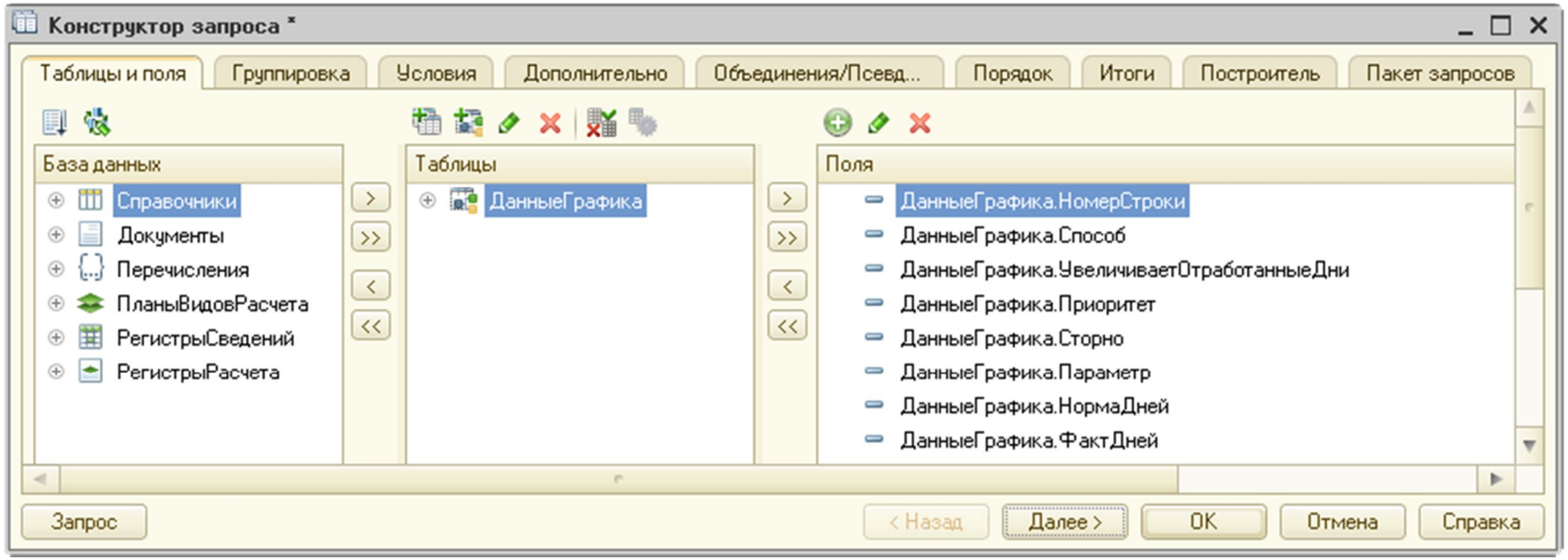
МВТ передадим в процедуру «РассчитатьОсновные()».
В процедуре «РассчитатьОсновные()» объектный метод получения Базы заменим на запрос:

Было:

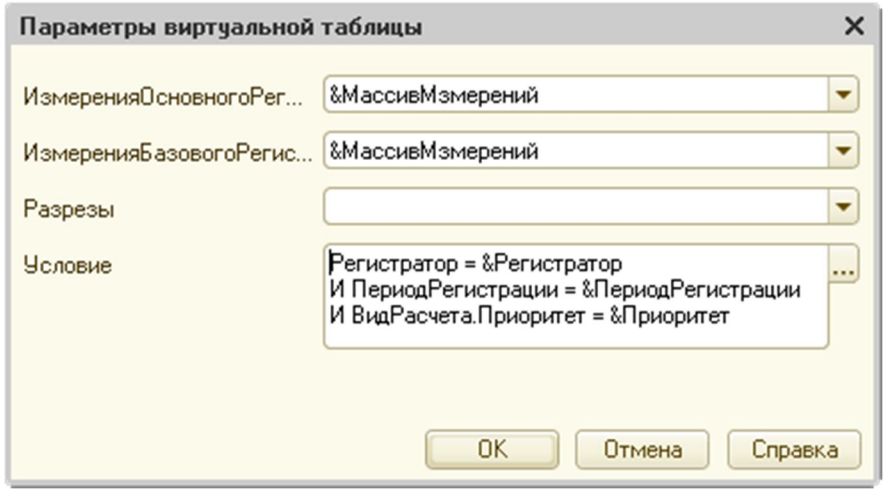
```
ОтборПоИзмерениям = Новый Структура;
ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");

СписокРесурсов = Новый Массив(2);
СписокРесурсов[0] = "НачисленияСотрудников.Сумма";
СписокРесурсов[1] = "НачисленияСотрудников.ОтработаноДней";

ОтборПоРегистратору = Новый Структура;
Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
ОтборПоРегистратору,
СписокРесурсов,
ОтборПоИзмерениям);
```

В конструкторе запроса опишем виртуальную таблицу «ДанныеГрафика», которую мы передали в процедуру:



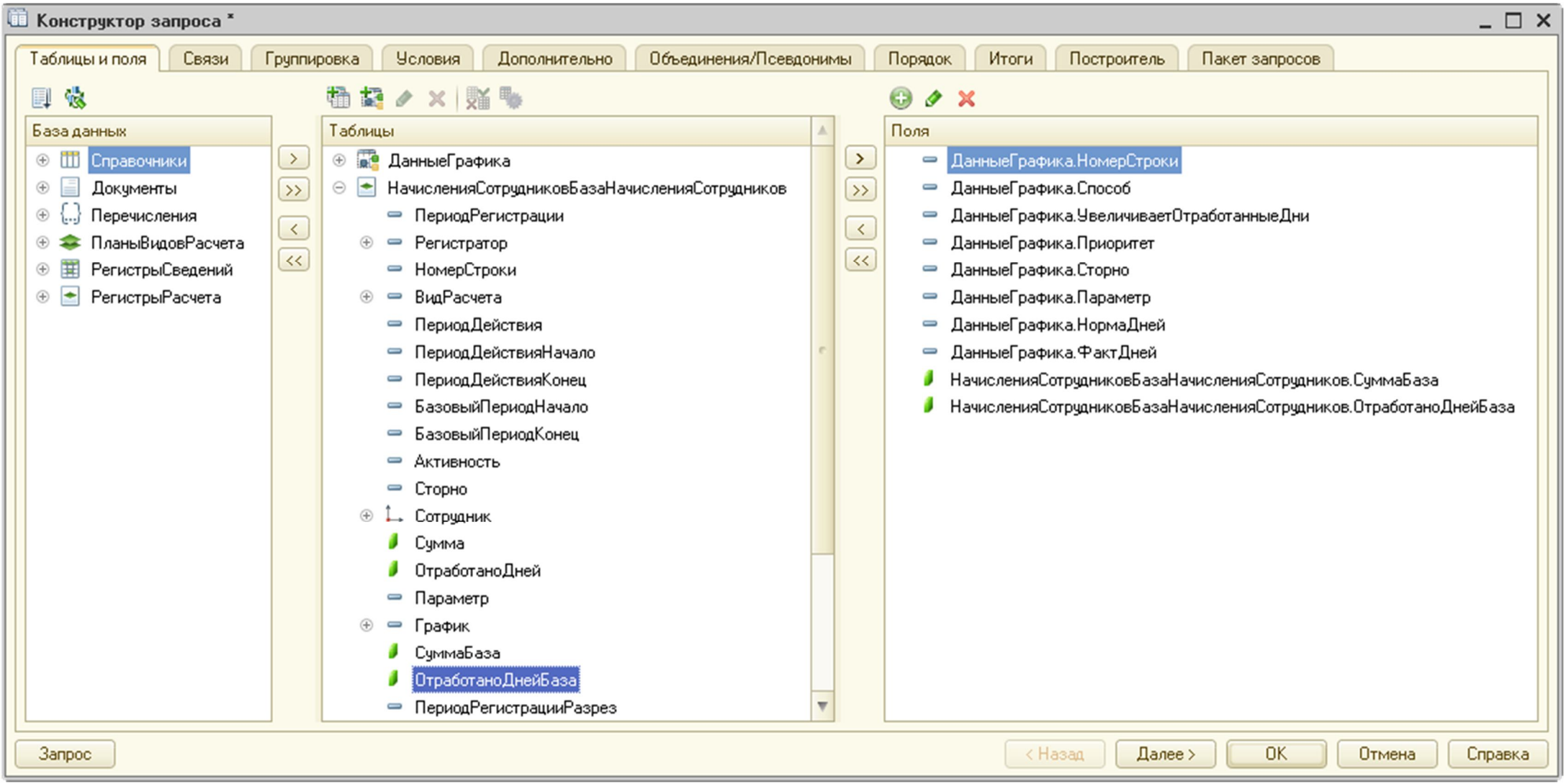


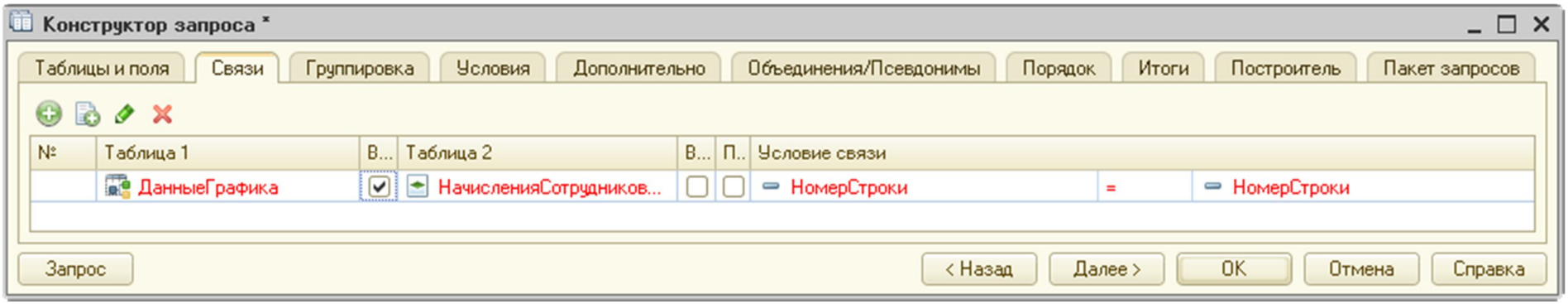
К ней надо присоединить таблицу с Базой регистра расчета «НачисленияСотрудников». Опишем параметры данной виртуальной таблицы:



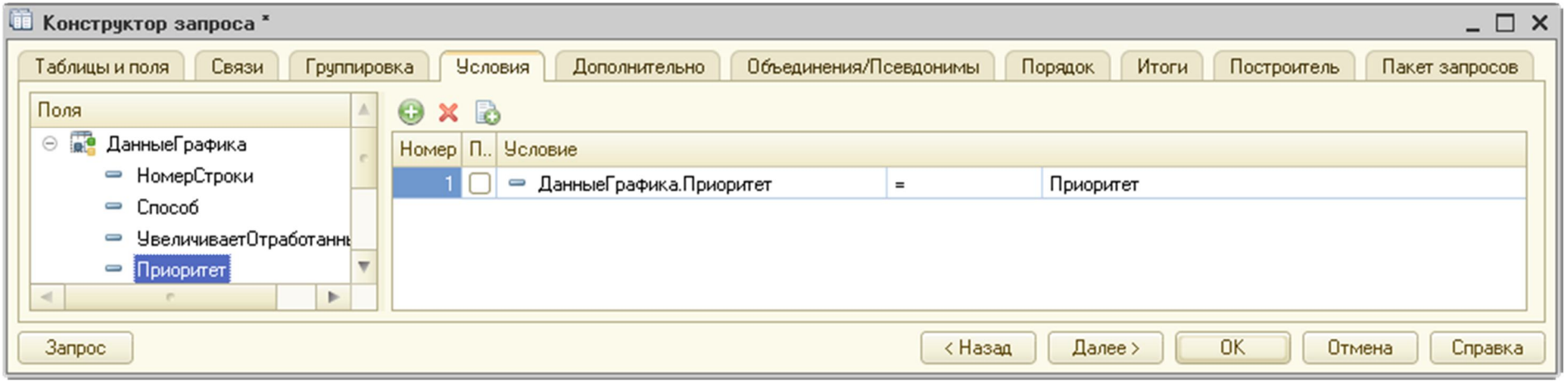
Первые два параметра – это как отбор по измерениям в объектной модели. Базу надо получать только по одному сотруднику, поэтому в измерения Основного и Базового регистра надо передать два массива:
1. «ИзмеренияОсновногоРегистра» - названия полей набора записей регистра, который содержит значения отбора. В примере это поле «Сотрудник».
2. «ИзмеренияБазовогоРегистра» - названия измерений в Базовом регистре, в котором установится отбор.
Т.к. в примере Базу получаем саму из себя, то отбор устанавливается также по Измерению «Сотрудник».

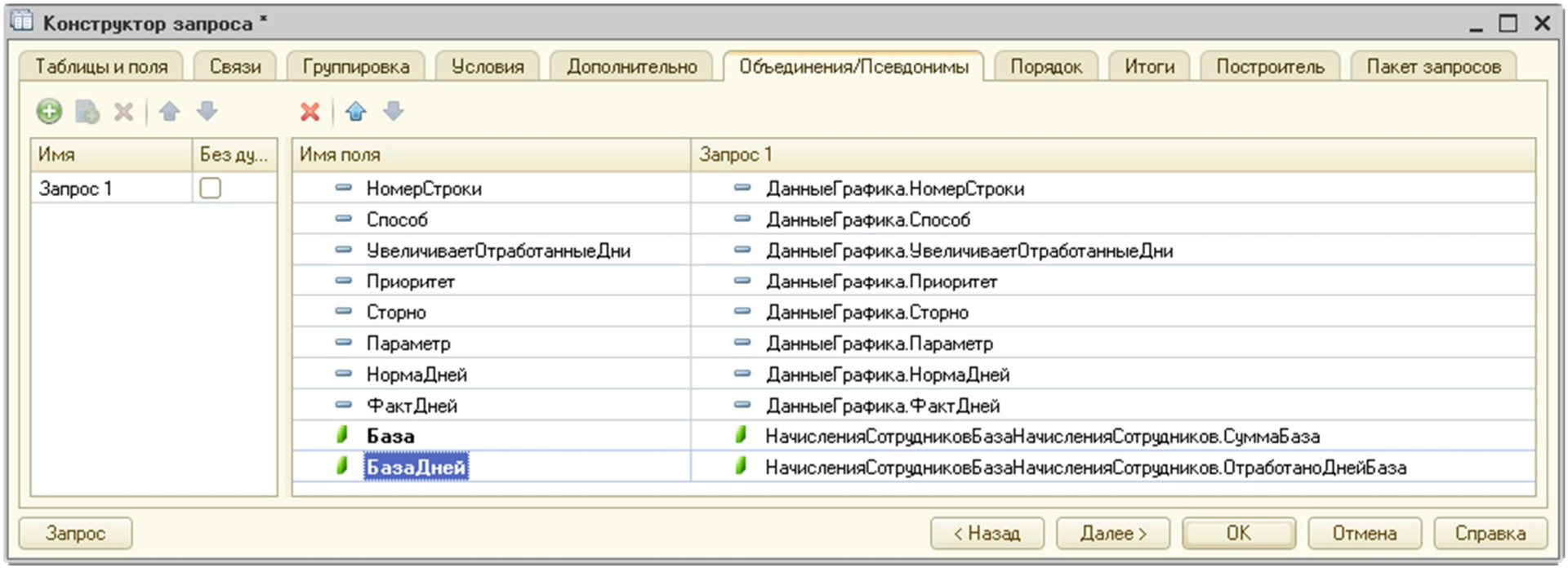
Условия:
«`Регистратор = &Регистратор`» - это отбор по документу, т.к. База нам нужна только по одному документу.
«`ПериодРегистрации = &ПериодРегистрации`» - для увеличения быстродействия. Поле «ПериодРегистрации» состоит в индексе таблицы регистра.
«`ВидРасчета.Приоритет = &Приоритет`» - это отбор по определенному Приоритету.


Конструктор запроса *
Таблицы и поля
Связи
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Порядок
Итоги
Построитель
Пакет запросов
База данных
Справочники
Документы
Перечисления
ПланыВидовРасчета
РегистрыСведений
РегистрыРасчета
Таблицы
ДанныеГрафика
НачисленияСотрудниковБазаНачисленияСотрудников
ПериодРегистрации
Регистратор
НомерСтроки
ВидРасчета
ПериодДействия
ПериодДействияНачало
ПериодДействияКонец
БазовыйПериодНачало
БазовыйПериодКонец
Активность
Сторно
Сотрудник
Сумма
ОтработаноДней
Параметр
График
СуммаБаза
ОтработаноДнейБаза
ПериодРегистрацииРазрез
Поля
ДанныеГрафика.НомерСтроки
ДанныеГрафика.Способ
ДанныеГрафика.УвеличиваетОтработанныеДни
ДанныеГрафика.Приоритет
ДанныеГрафика.Сторно
ДанныеГрафика.Параметр
ДанныеГрафика.НормаДней
ДанныеГрафика.ФактДней
НачисленияСотрудниковБазаНачисленияСотрудников.СуммаБаза
НачисленияСотрудниковБазаНачисленияСотрудников.ОтработаноДнейБаза
Запрос
< Назад
Далее >
OK
Отмена
Справка


Конструктор запроса *
Таблицы и поля
Связи
Группировка
Условия
Дополнительно
Объединения/Псевдонимы
Порядок
Итоги
Построитель
Пакет запросов
№
Таблица 1
В...
Таблица 2
В...
П..
Условие связи
ДанныеГрафика
НачисленияСотрудников...
НомерСтроки
=
НомерСтроки
Запрос
< Назад
Далее >
OK
Отмена
Справка

НомерСтроки – это единственное поле, точно идентифицирующее запись по позиции регистратора.





Поля:

- НормаДней
- ФактДней
- База
- БазаДней

могут быть не рассчитаны, т.е. вернут NULL, поэтому воспользуемся функцией ЕСТЬNULL.

Итого запрос:

```
ВЫБРАТЬ
	ДанныеГрафика.НомерСтроки,
	ДанныеГрафика.Способ,
	ДанныеГрафика.УвеличиваетОтработанныеДни,
	ДанныеГрафика.Приоритет,
	ДанныеГрафика.Сторно,
	ДанныеГрафика.Параметр,
	ЕСТЬNULL(ДанныеГрафика.НормаДней, 0) КАК НормаДней,
	ЕСТЬNULL(ДанныеГрафика.ФактДней, 0) КАК ФактДней,
	ЕСТЬNULL(НачисленияСотрудниковБазаНачисленияСотрудников.СуммаБаза, 0) КАК База,
	ЕСТЬNULL(НачисленияСотрудниковБазаНачисленияСотрудников.ОтработаноДнейБаза, 0) КАК БазаДней
ИЗ
	ДанныеГрафика КАК ДанныеГрафика
		ЛЕВОЕ СОЕДИНЕНИЕ РегистрРасчета.НачисленияСотрудников.БазаНачисленияСотрудников(
				&МассивМзмерений,
				&МассивМзмерений,
				,
				Регистратор = &Регистратор
					И ПериодРегистрации = &ПериодРегистрации
					И ВидРасчета.Приоритет = &Приоритет) КАК НачисленияСотрудниковБазаНачисленияСотрудников
		ПО ДанныеГрафика.НомерСтроки = НачисленияСотрудниковБазаНачисленияСотрудников.НомерСтроки
ГДЕ
	ДанныеГрафика.Приоритет = &Приоритет
```

Итого код:

```
Процедура Рассчитать(Регистратор) Экспорт
    // получим копию набора записей регистра
    НаборЗаписей = РегистрыРасчета.НачисленияСотрудников.СоздатьНаборЗаписей();
    НаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    НаборЗаписей.Прочитать();

    //ОтборПоРегистратору = Новый Структура;
    //ОтборПоРегистратору.Вставить("Регистратор", Регистратор);
    //
    //ТЗ_ФактДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    //ОтборПоРегистратору,
    //ВидПериодаРегистраРасчета.ФактическийПериодДействия);
    //
    //ТЗ_НормаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    //ОтборПоРегистратору,
    //ВидПериодаРегистраРасчета.ПериодДействия);

    //ТЗ_БазаДней = РегистрыРасчета.НачисленияСотрудников.ПолучитьДанныеГрафика(
    //ОтборПоРегистратору,
    //ВидПериодаРегистраРасчета.БазовыйПериод);

    Запрос = Новый Запрос;
    МВТ = Новый МенеджерВременныхТаблиц;
    Запрос.МенеджерВременныхТаблиц = МВТ;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    НачисленияСотрудниковДанныеГрафика.НомерСтроки,
    |    НачисленияСотрудниковДанныеГрафика.ВидРасчета.Способ КАК Способ,
    |    НачисленияСотрудниковДанныеГрафика.ВидРасчета.УвеличиваетОтработанныеДни
            КАК УвеличиваетОтработанныеДни,
    |    НачисленияСотрудниковДанныеГрафика.ВидРасчета.Приоритет КАК Приоритет,
    |    НачисленияСотрудниковДанныеГрафика.Сторно,
    |    НачисленияСотрудниковДанныеГрафика.Параметр,
    |    НачисленияСотрудниковДанныеГрафика.ПризнакПериодДействия КАК НормаДней,
```

```
    |    НачисленияСотрудниковДанныеГрафика.ПризнакФактическийПериодДействия КАК ФактДней
    |ПОМЕСТИТЬ ДанныеГрафика
    |ИЗ
    |    РегистрРасчета.НачисленияСотрудников.ДанныеГрафика(
    |                Регистратор = &Регистратор
    |                    И ПериодРегистрации = &ПериодРегистрации) КАК НачисленияСотрудниковДанныеГрафика";
    Запрос.УстановитьПараметр("Регистратор", Регистратор);
    Запрос.УстановитьПараметр("ПериодРегистрации", НачалоМесяца(Регистратор.Дата));
    Запрос.Выполнить();

    //Получим копию набора записей регистра
    ДопНаборЗаписей = РегистрыРасчета.ДопНачисления.СоздатьНаборЗаписей();
    ДопНаборЗаписей.Отбор.Регистратор.Установить(Регистратор);
    ДопНаборЗаписей.Прочитать();

    СписокПриоритетов = ПолучитьСписокПриоритетов(Регистратор);

    Для Каждого Приоритет Из СписокПриоритетов Цикл
        РассчитатьОсновные(НаборЗаписей, МВТ, Приоритет);
        РассчитатьДополнительные(ДопНаборЗаписей, Приоритет);
    КонецЦикла;

КонецПроцедуры

Процедура РассчитатьОсновные(НаборЗаписей, МВТ, Приоритет)

    //ОтборПоИзмерениям = Новый Структура;
    //ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник");
    //
    //СписокРесурсов = Новый Массив(2);
    //СписокРесурсов[0] = "НачисленияСотрудников.Сумма";
    //СписокРесурсов[1] = "НачисленияСотрудников.ОтработаноДней";
    //
    //ОтборПоРегистратору = Новый Структура;
    //Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
    //ОтборПоРегистратору.Вставить("Регистратор", Регистратор);
```

```
//
//ТЗ_База = РегистрыРасчета.НачисленияСотрудников.ПолучитьБазу(
//ОтборПоРегистратору,
//СписокРесурсов,
//ОтборПоИзмерениям);

Запрос = Новый Запрос;
Запрос.МенеджерВременныхТаблиц = МВТ;
Запрос.Текст =
"ВЫБРАТЬ
|    ДанныеГрафика.НомерСтроки,
|    ДанныеГрафика.Способ,
|    ДанныеГрафика.УвеличиваетОтработанныеДни,
|    ДанныеГрафика.Приоритет,
|    ДанныеГрафика.Сторно,
|    ДанныеГрафика.Параметр,
|    ЕСТЬNULL(ДанныеГрафика.НормаДней, 0) КАК НормаДней,
|    ЕСТЬNULL(ДанныеГрафика.ФактДней, 0) КАК ФактДней,
|    ЕСТЬNULL(НачисленияСотрудниковБазаНачисленияСотрудников.СуммаБаза, 0) КАК База,
|    ЕСТЬNULL(НачисленияСотрудниковБазаНачисленияСотрудников.ОтработаноДнейБаза, 0) КАК БазаДней
|ИЗ
|    ДанныеГрафика КАК ДанныеГрафика
|        ЛЕВОЕ СОЕДИНЕНИЕ РегистрРасчета.НачисленияСотрудников.БазаНачисленияСотрудников(
|                &МассивМзмерений,
|                &МассивМзмерений,
|                ,
|                Регистратор = &Регистратор
|                    И ПериодРегистрации = &ПериодРегистрации
|                    И ВидРасчета.Приоритет = &Приоритет)
                        КАК НачисленияСотрудниковБазаНачисленияСотрудников
|        ПО ДанныеГрафика.НомерСтроки = НачисленияСотрудниковБазаНачисленияСотрудников.НомерСтроки
|ГДЕ
|    ДанныеГрафика.Приоритет = &Приоритет";

Массив = Новый Массив(1);
Массив[0] = "Сотрудник";
```

```
Запрос.УстановитьПараметр("МассивМзмерений", Массив);
Запрос.УстановитьПараметр("Регистратор", НаборЗаписей.Отбор.Регистратор.Значение);
Запрос.УстановитьПараметр("ПериодРегистрации", НачалоМесяца(НаборЗаписей.Отбор.Регистратор.Значение.Дата));
Запрос.УстановитьПараметр("Приоритет", Приоритет);

РезультатЗапроса = Запрос.Выполнить();
Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл
     Запись                       = НаборЗаписей[Выборка.НомерСтроки - 1];
     База                         = Выборка.База;
     СпособРасчета                = Выборка.Способ;
     ФактДней                     = Выборка.ФактДней;
     НормаДней                    = Выборка.НормаДней;
     БазаДней                     = Выборка.БазаДней;
     Параметр                     = Выборка.Параметр;
     УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

//Для Каждого Запись Из НаборЗаписей Цикл
//
//   Если Запись.ВидРасчета.Приоритет <> Приоритет Тогда
//        Продолжить;
//   КонецЕсли;
//
//   //Индекс = Запись.НомерСтроки - 1;
//   Индекс = НаборЗаписей.Индекс(Запись);
//   База = ТЗ_База[Индекс].Сумма;
//   СпособРасчета = Запись.ВидРасчета.Способ;
//   ФактДней = ТЗ_ФактДней[Индекс].Признак;
//   НормаДней = ТЗ_НормаДней[Индекс].Признак;
//   //БазаДней = ТЗ_БазаДней[Индекс].Признак;
//   БазаДней = ТЗ_База[Индекс].ОтработаноДней;
//   Параметр = Запись.Параметр;
//   УвеличиваетОтработанныеДни = Запись.ВидРасчета.УвеличиваетОтработанныеДни;

     Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
```

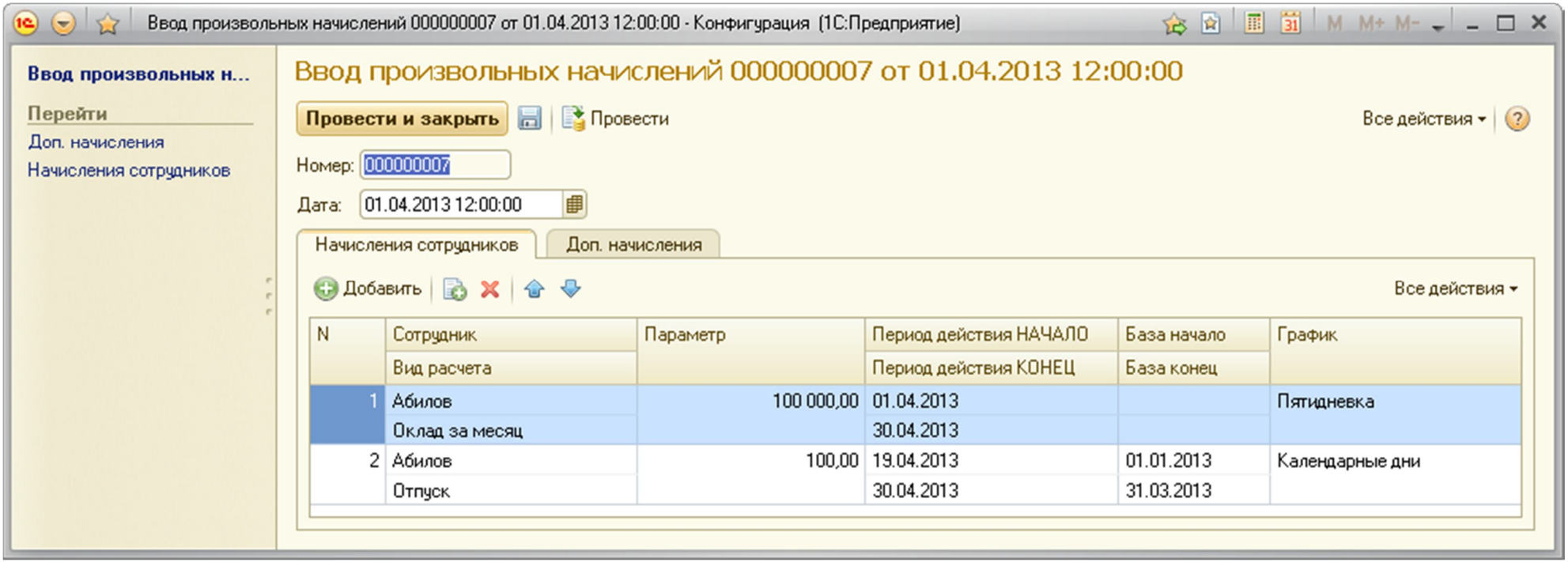
```
            Запись.Сумма = База * Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
            Запись.Сумма = Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоДням Тогда
            Запись.Сумма = ФактДней * Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПропорциональноОтработанномуВремени Тогда
            Запись.Сумма = ФактДней / НормаДней * Параметр;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ПоСреднему Тогда
            Запись.Сумма = База / БазаДней * ФактДней * Параметр / 100;
        КонецЕсли;

        Если УвеличиваетОтработанныеДни тогда
            Запись.ОтработаноДней = ФактДней;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
            Запись.ОтработаноДней = - Запись.ОтработаноДней;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать( , , Ложь);
КонецПроцедуры

Процедура РассчитатьДополнительные(НаборЗаписей, Приоритет)
    ОтборПоИзмерениям = Новый Структура;
    ОтборПоИзмерениям.Вставить("Сотрудник", "НачисленияСотрудников.Сотрудник,
    |ДопНачисления.Сотрудник");

    СписокРесурсов = Новый Массив(1);
    СписокРесурсов[0] = "НачисленияСотрудников.Сумма, ДопНачисления.Сумма";

    ОтборПоРегистратору = Новый Структура;
    Регистратор = НаборЗаписей.Отбор.Регистратор.Значение;
    ОтборПоРегистратору.Вставить("Регистратор", Регистратор);

    ТЗ_База = РегистрыРасчета.ДопНачисления.ПолучитьБазу(
```

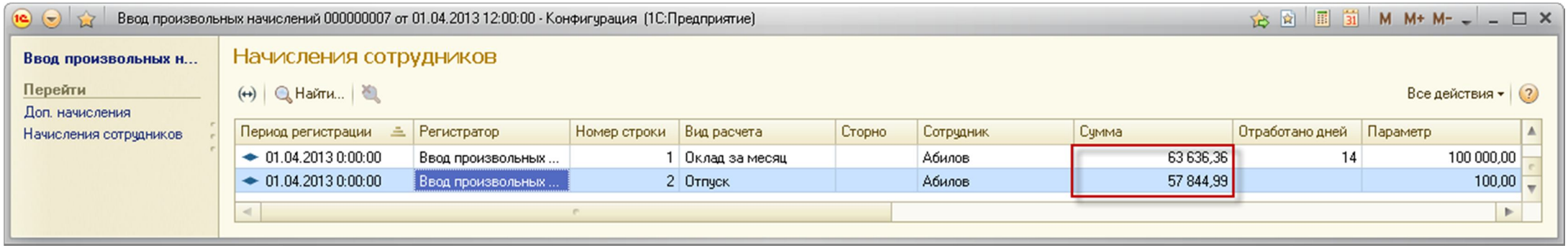
```
    ОтборПоРегистратору,
    СписокРесурсов,
    ОтборПоИзмерениям);

    Для Каждого Запись Из НаборЗаписей Цикл

        Если Запись.ВидРасчета.Приоритет <> Приоритет Тогда
            Продолжить;
        КонецЕсли;

        Индекс        = НаборЗаписей.Индекс(Запись);
        База          = ТЗ_База[Индекс].Сумма;
        СпособРасчета = Запись.ВидРасчета.Способ;
        Параметр      = Запись.Параметр;

        Если СпособРасчета = Перечисления.СпособыРасчета.Процентом Тогда
            Запись.Сумма = База * Параметр / 100;
        ИначеЕсли СпособРасчета = Перечисления.СпособыРасчета.ФиксированнаяСумма Тогда
            Запись.Сумма = Параметр;
        КонецЕсли;

        Если Запись.Сторно Тогда
            Запись.Сумма = - Запись.Сумма;
        КонецЕсли;
    КонецЦикла;
    НаборЗаписей.Записать();
КонецПроцедуры

Функция ПолучитьСписокПриоритетов(Регистратор)
    Запрос = Новый Запрос;
    Запрос.Текст =
    "ВЫБРАТЬ
    |    ВложенныйЗапрос.Приоритет КАК Приоритет
    |ИЗ
    |    (ВЫБРАТЬ
    |        НачисленияСотрудников.ВидРасчета.Приоритет КАК Приоритет
```

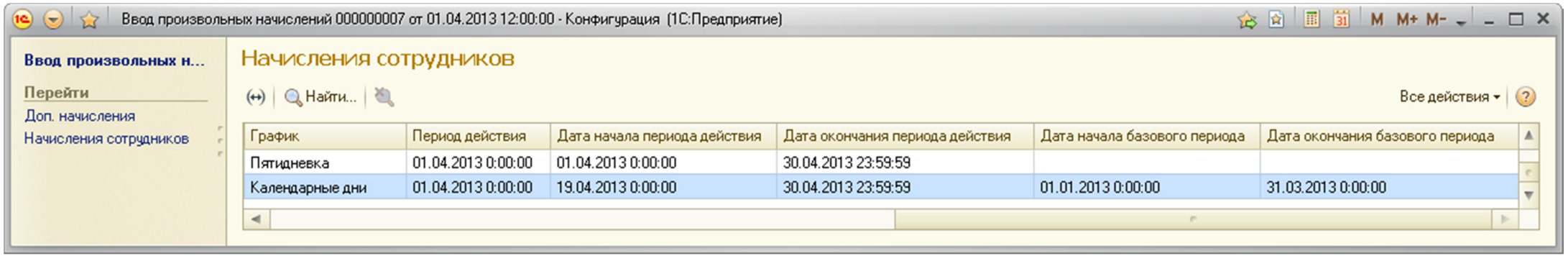
```
|    ИЗ
|         РегистрРасчета.НачисленияСотрудников КАК НачисленияСотрудников
|    ГДЕ
|         НачисленияСотрудников.Регистратор = &Регистратор
|
|    ОБЪЕДИНИТЬ
|
|    ВЫБРАТЬ
|         ДопНачисления.ВидРасчета.Приоритет
|    ИЗ
|         РегистрРасчета.ДопНачисления КАК ДопНачисления
|    ГДЕ
|         ДопНачисления.Регистратор = &Регистратор) КАК ВложенныйЗапрос
|
|УПОРЯДОЧИТЬ ПО
|    Приоритет";
Запрос.УстановитьПараметр("Регистратор", Регистратор);
РезультатЗапроса = Запрос.Выполнить().Выгрузить();
Возврат РезультатЗапроса.ВыгрузитьКолонку("Приоритет");
```

КонецФункции

В режиме исполнения:
Перепроведем документ «Ввод произвольных начислений» № 000000007:
Было:







Стало:

Аналогично. Вывод: начисление считается правильно.

Перерасчеты

Автоматический перерасчет записей Регистра расчетов. Механизм делится на две спецификации:

1.

*

Оклад	Оклад	Прогул	Оклад	Оклад

100000

Допустим, рассчитан Оклад. Прошло время, и результат расчета Оклада изменился из-за ввода Вытеснений. При этом система для этой записи автоматом установит необходимость перерасчета (пометка). Это «Перерасчет по вытеснению».

2.

Оклад 100000

Налог 13000

Алименты (Оклад - Налог) *Y%

Если изменится сумма Оклада, то надо пересчитать Налог и Алименты, но не в одной транзакции, т.к. расчет Алиментов требует завершенного расчета Налога от настоящего Оклада. Это «Зависимость перерасчета по Базе». Здесь уже требуется ручное описание Зависимостей.

Перерасчет – это таблица с информацией о том, что надо пересчитать. Система может заполнять таблицу автоматом. Использовать таблицу или нет – зависит от пользователя и/или разработчика, т.е. механизм по умолчанию имеет уведомительный характер.
Стандартная таблица перерасчетов состоит из двух колонок:

- ОбъектПерерасчета.
- ВидРасчета.

Пример:
Если оставить только стандартные колонки:

ОбъектПерерасчета	ВидРасчета
Документ «Начисление премии»	Премия

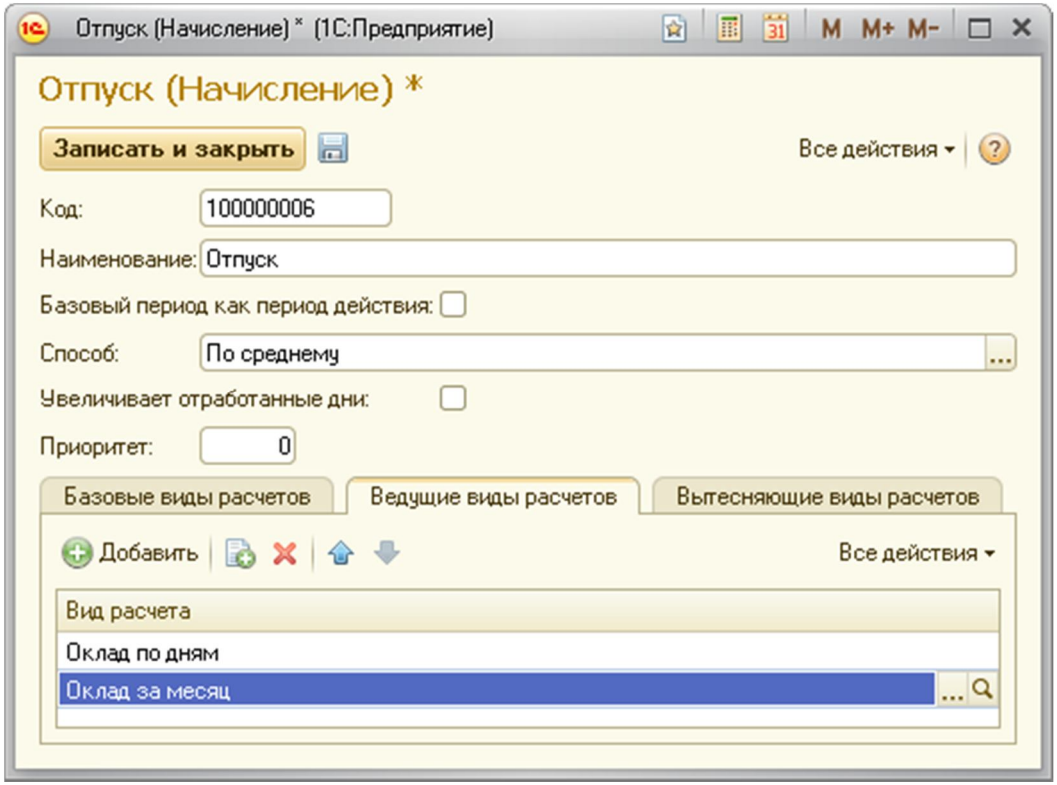
то непонятно, кому надо перерасчитать Премию. Поэтому к данной таблице добавим колонку «Сотрудник»

В Регистре расчета «НачисленияСотрудников» создадим перерасчет «Перерасчет1» и добавим измерение «Сотрудник»:

Свойство	**Значение**
Имя	Сотрудник
Синоним	Сотрудник
Измерение регистра	Сотрудник
Данные ведущих регистров	Сотрудник

Надо указать связь текущего регистра («Измерение регистра») с Базовым («Данные ведущих регистров»). Т.к. База тоже из этого регистра, то связь одинаковая. Т.е. при изменении Оклада по Иванову, Налог и Премию надо пересчитать тоже только по Иванову.

Таблица накапливается автоматом, если наладить отбор по Базовой таблице. В режиме исполнения:



На закладке «Ведущие» указываются Виды расчета, от которых зависит текущий Вид расчета, т.е. при изменении каких видов расчета в Базовом периоде, надо пересчитать текущие записи.

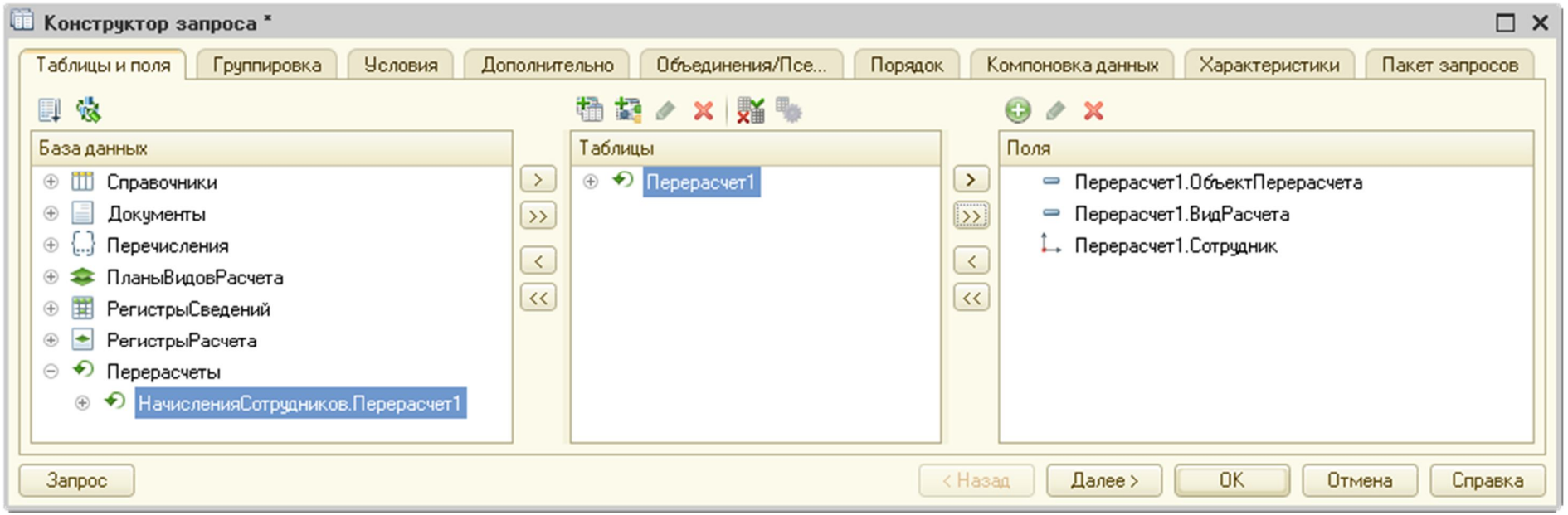
Таблица накапливается автоматом, при изменении Вытесняющий или Ведущих Видов расчета.
Удаляются записи из таблицы при фиксировании записей в регистре. Например, при перепроведении документа «Начисление отпускных» заново фиксируется запись в Регистре расчета и система «смотрит», по какому документу эти записи зафиксировались. Если этот документ находится в таблице Перерасчета, то все записи по этому документу автоматом из нее удаляются.

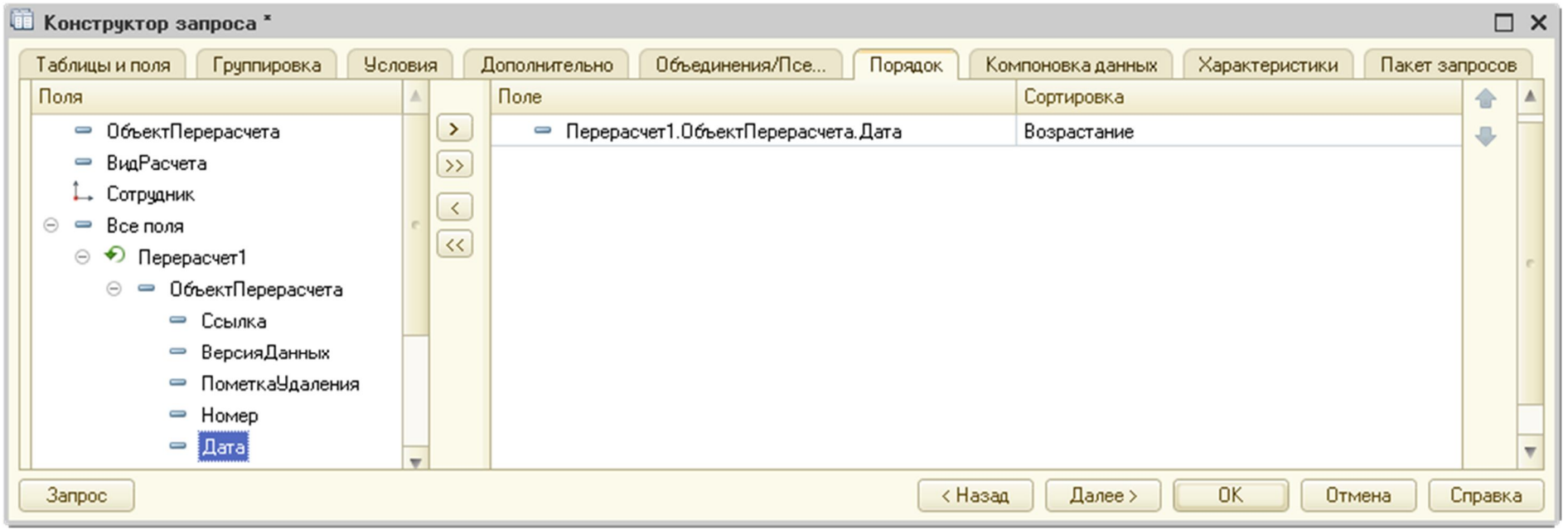
Но сам механизм носит уведомительный характер, т.е. для осуществления перерасчета нужна обработка.

Разработаем отчет, показывающий записи, которые надо перерасчитать – «Отчет по перерасчетам»:

Свойство	Значение
Имя	ОтчетПоПерерасчетам
Синоним	Отчет по перерасчетам

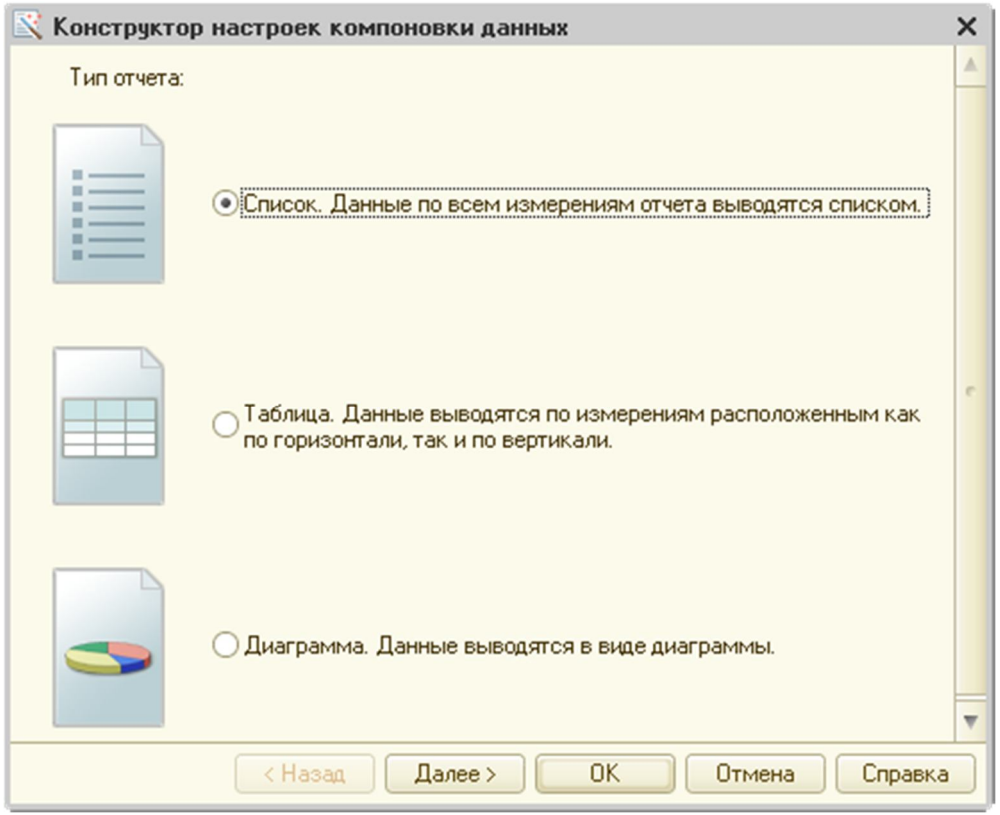
Откроем СКД, добавим «Набор данных-запрос»:

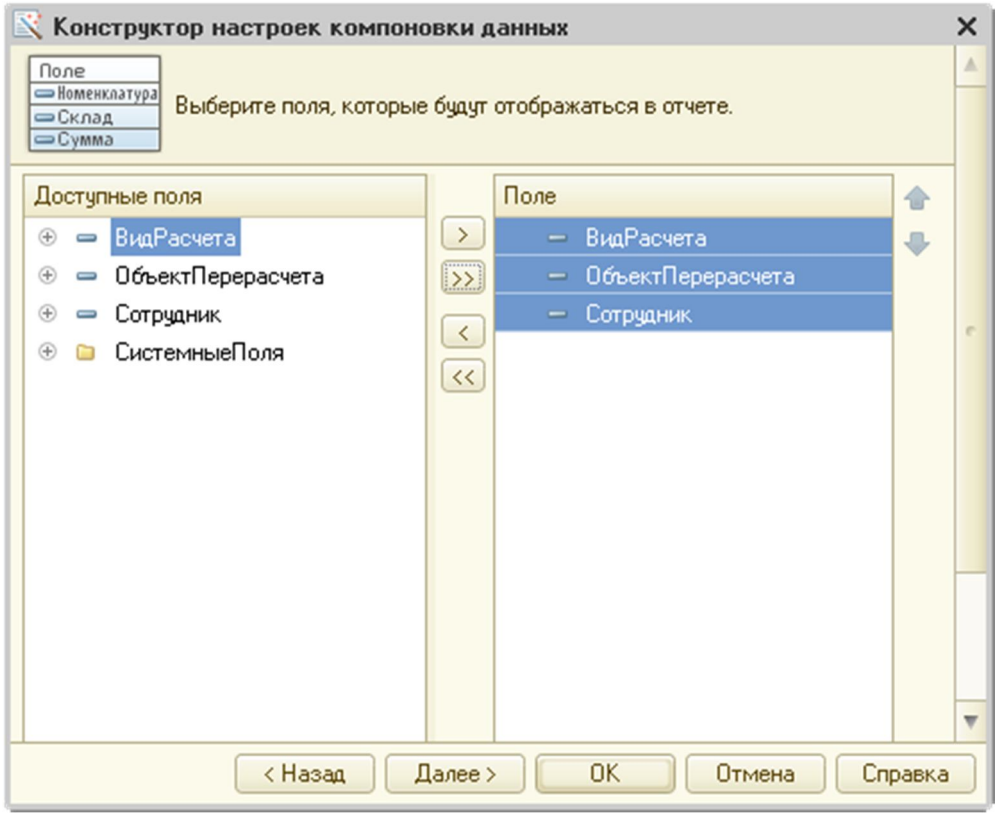




Запрос готов.

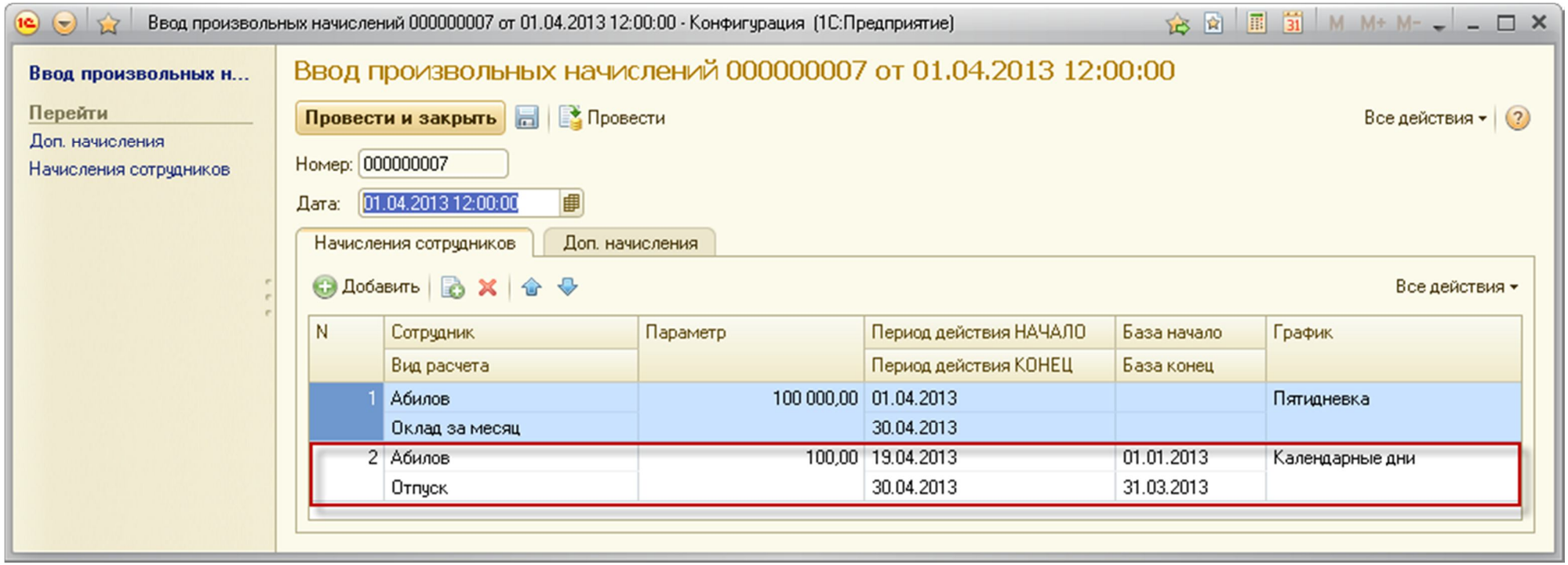
СКД – Закладка «Настройки» - настроим вывод отчета:



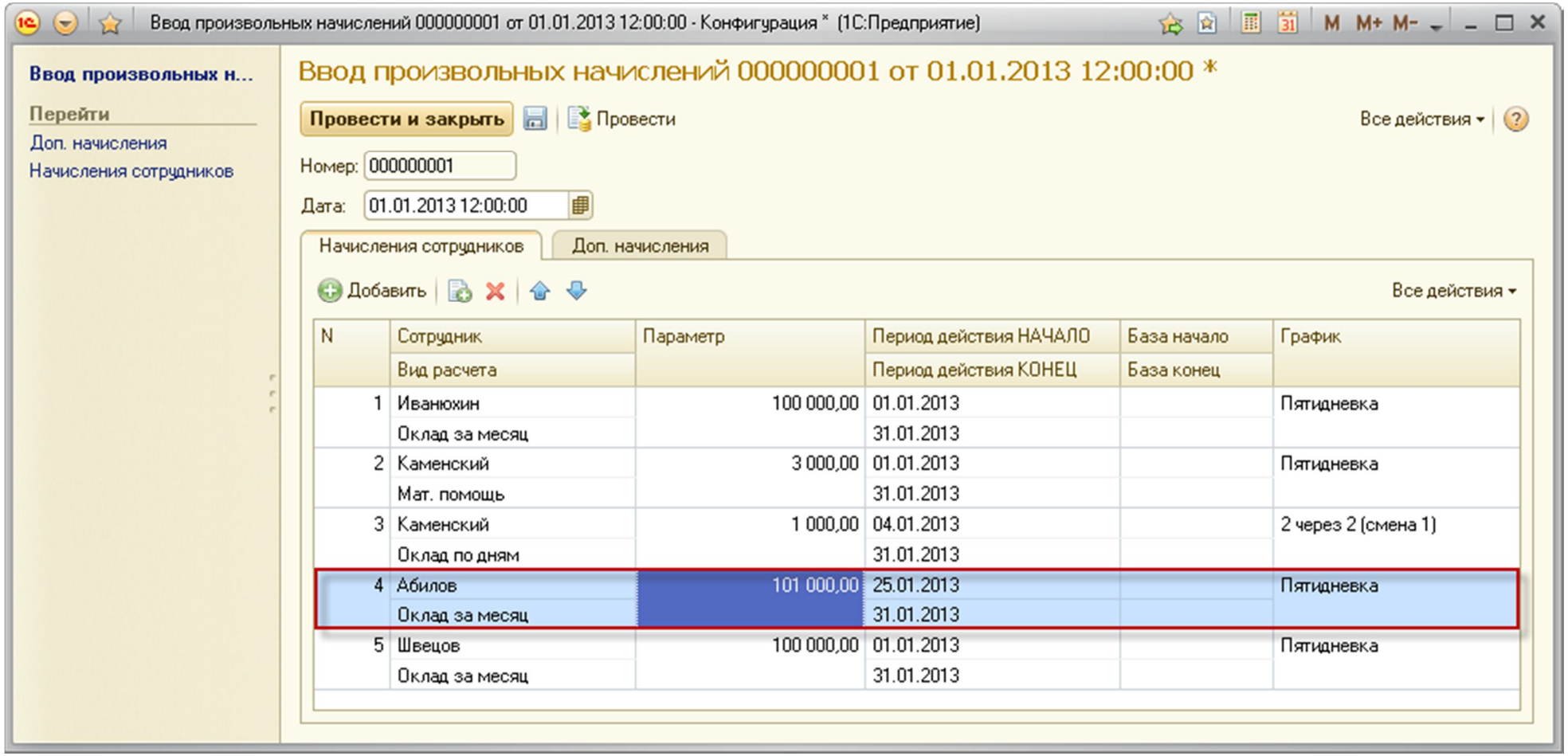


Отчет готов. В режиме исполнения:

По умолчанию таблица Перерасчетов пустая.
База Отпуска Обилова формируется за 1-3 месяцы:

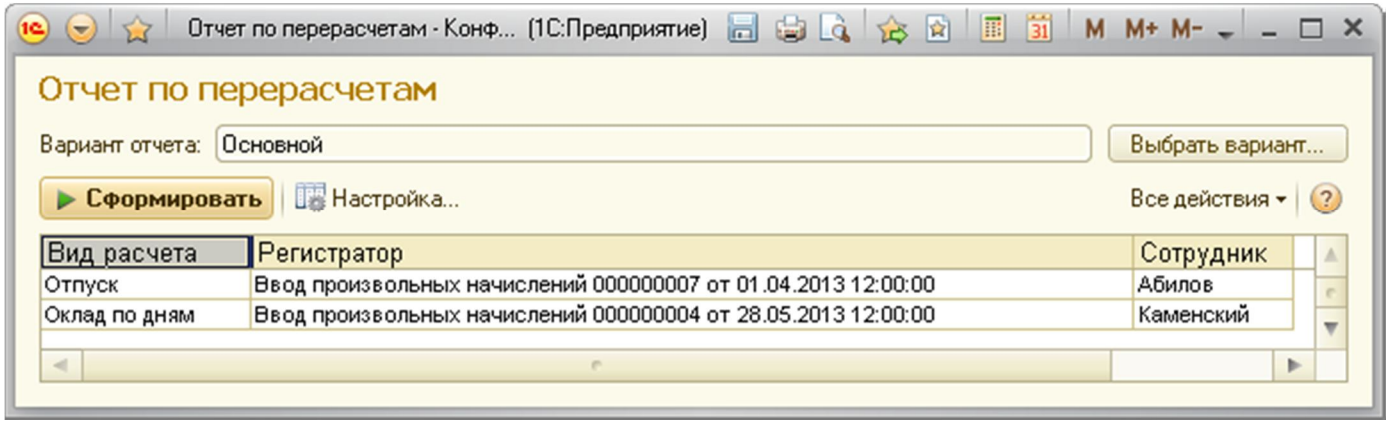


Изменим ему Оклад за первый месяц:



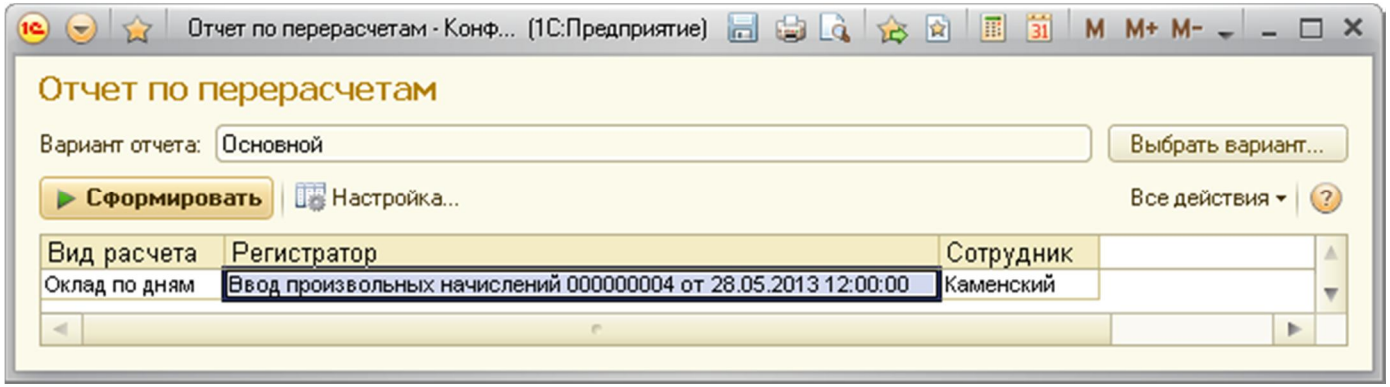
N	Сотрудник / Вид расчета	Параметр	Период действия НАЧАЛО / Период действия КОНЕЦ	База начало / База конец	График
1	Иванюхин	100 000,00	01.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		
2	Каменский	3 000,00	01.01.2013		Пятидневка
	Мат. помощь		31.01.2013		
3	Каменский	1 000,00	04.01.2013		2 через 2 (смена 1)
	Оклад по дням		31.01.2013		
4	Абилов	101 000,00	25.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		
5	Швецов	100 000,00	01.01.2013		Пятидневка
	Оклад за месяц		31.01.2013		

И проверим отчет:



Две записи появились потому, что в документе, где меняли Оклад Обилову, были и другие сотрудники, но нас сейчас интересует только Обилов.

Перепроведем документ с Отпуском и проверим отчет:



Запись о необходимости перепроведения документа с Отпуском, исчезла.