

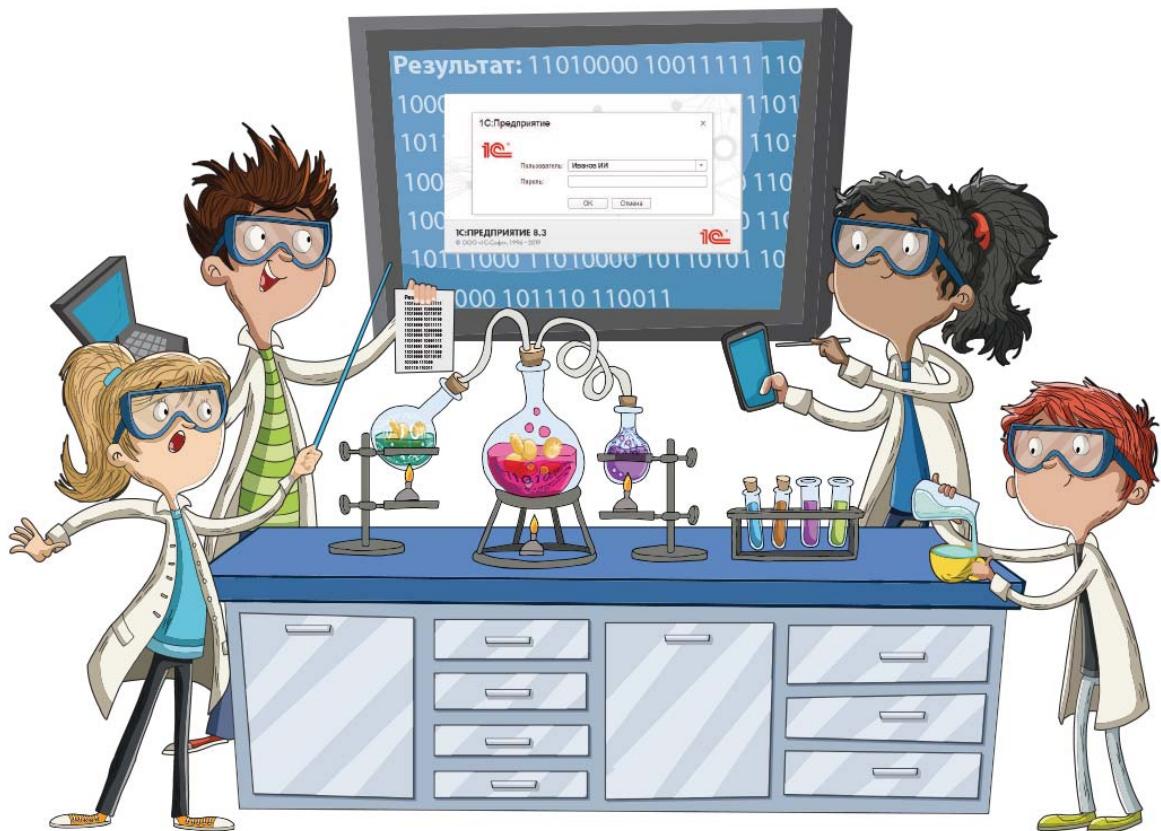


Павел Чистов, Алина Мальгинова

Москва
«1С-Паблишинг»
2021

Сборник лабораторных работ для студентов учебных заведений, изучающих программирование

в системе 1С:Предприятие 8
(1C:Enterprise 8)



Чистов П. А., Мальгинова А. А.

Сборник лабораторных работ для студентов учебных заведений, изучающих программирование в системе 1С:Предприятие 8 (1C:Enterprise 8)

Электронная книга в формате pdf; ISBN 978-5-9677-3063-4.

Электронный аналог печатного издания

«Сборник лабораторных работ для студентов учебных заведений, изучающих программирование в системе 1С:Предприятие 8 (1C:Enterprise 8)» (ISBN 978-5-9677-3035-1) М.: ООО «1С-Паблишинг», 2021.

артикул печатной книги по прейс-листу фирмы «1С»: 4601546145208;

по вопросам приобретения печатных изданий издательства «1С-Паблишинг» обращайтесь к партнеру «1С», обслуживающему вашу организацию, или к другим партнерам фирмы «1С», в магазины «1С Интерес», а также в книжные и интернет-магазины).

В издании представлены лабораторные работы для студентов учебных заведений, изучающих программирование в системе 1С:Предприятие 8 (1C:Enterprise 8), рекомендованные к использованию в учебном процессе в образовательных программах высшего и среднего профессионального образования по ИТ-направлениям с учетом требований ФГОС и профессиональных стандартов. Книга ориентирована на преподавателей ИТ-дисциплин, методистов и других сотрудников образовательных организаций ВО и СПО.

Сборник задач рекомендован Федеральными учебно-методическими объединениями в системе высшего и среднего профессионального образования по укрупненной группе "Информатика и вычислительная техника" в качестве учебного пособия.

АВТОРЫ СБОРНИКА



Чистов Павел Анатольевич

- Методист 1С:Учебного центра № 1.
- Сертифицированный преподаватель 1С:ЦСО.
- Независимый эксперт WorldSkills в компетенции «IT-решения для бизнеса».
- Сертифицированный преподаватель Lego Education LME EV3.



Мальгинова Алина Андреевна

- Выпускница Физико-математического лицея.
- Призер и победитель Олимпиад по математике, информатике, русскому языку.
- Обладатель Диплома бакалавра с отличием по направлению подготовки «Информационные системы и технологии» Московского технического университета связи и информатики.
- Методист и преподаватель 1С:Учебного центра № 1.

ОГЛАВЛЕНИЕ

Введение	7
Как читать этот сборник?	8
Лабораторная работа № 1. Установка системы 1С:Предприятие 8	9
Лабораторная работа № 2. Основные принципы работы с платформой	17
Добавление пустой информационной базы	18
Режимы запуска «1С:Предприятие» и «Конфигуратор»	21
Открытие окна конфигурации	22
Добавление нового объекта конфигурации.....	24
Удаление существующего объекта конфигурации.....	25
Окно редактирования объектов конфигурации	27
Палитра свойств объекта конфигурации	28
Важные свойства объектов конфигурации	29
Обновление конфигурации.....	30
Запуск режима «1С:Предприятие» из режима «Конфигуратор»	30
Выгрузка информационной базы	32
Загрузка информационной базы	32
Лабораторная работа № 3. Разработка конфигурации для организации хранения информации о студентах и изучаемых ими предметах.....	33
Лабораторная работа № 4. Разработка информационной системы для хранения информации о сотрудниках предприятия.....	51

Лабораторная работа № 5. Разработка конфигурации для учета посещений клиентами экскурсий	71
Лабораторная работа № 6. Разработка учетной системы для ведения информации о кассовых операциях.....	91
Лабораторная работа № 7. Разработка информационной системы, регистрирующей изменение курсов валют.....	124
Лабораторная работа № 8. Разработка информационной системы, регистрирующей изменение цен купли и продажи валют.....	140
Лабораторная работа № 9. Создать небольшую информационную систему для регистрации продаж в студенческом киоске	155
Лабораторная работа № 10. Разработка конфигурации для учета работы студентов на занятиях	176
Лабораторная работа № 11. Автоматизировать систему пункта проката электросамокатов в учебном заведении	208
Лабораторная работа № 12. Разработка информационной системы для библиотеки.....	238
Лабораторная работа № 13. Разработка информационной системы для небольшого торгового павильона.....	265
Лабораторная работа № 14. Разработка конфигурации для учета продаж товаров с сопутствующими услугами покупателям.....	286
Лабораторная работа № 15. Разработка конфигурации для учета доходов от продаж товаров	315
Лабораторная работа № 16. «Отловить» первый запуск информационной системы	352

Лабораторная работа № 17. Разработка конфигурации для учета товаров. Самая простая задача.....	368
Лабораторная работа № 18. Разработка конфигурации для учета товаров. Продажа товаров с одного склада	393
Лабораторная работа № 19. Разработка конфигурации для учета товаров. Продажа товаров с разных складов.....	426
Лабораторная работа № 20. Разработка конфигурации для учета товаров. Контроль срока годности товаров	460

ВВЕДЕНИЕ

Приветствуем, коллеги!

В руках у вас – сборник лабораторных работ для дисциплин, в которых изучают программирование в среде 1С:Предприятие.

О создании такого сборника неоднократно просили преподаватели. Но мы отказывались его писать, так как у каждого преподавателя – свои программы и свой стиль подачи материала.

На написание сборника нас подтолкнули изучение некоторых работ коллег и понимание, что образец для изучения все же должен быть.

Мы благодарны всем, кто принял участие в проверке работ этого сборника, и кто присыпал свои «лабы» для изучения.

Все вопросы, пожелания и замечания вы можете отправлять нам на электронную почту edu@1c.ru.

КАК ЧИТАТЬ ЭТОТ СБОРНИК?

Уважаемые коллеги, этот сборник предполагается использовать в учебном процессе по специальностям, в которых преподают программирование в среде 1С:Предприятие (1C:Enterprise):

- Каждая лабораторная работа имеет описание темы, на которую делается акцент при решении задачи.
- Лабораторные работы мы старались сделать независимыми друг от друга, так как в разных учебных заведениях программы обучения могут отличаться.
- Материалы в сборнике имеют теги, которые предназначены для понимания полного списка механизмов, демонстрируемых в лабораторных работах. Также работы имеют три уровня сложности.
- В материалах вы найдете ссылки на ресурсы фирмы «1С» – с описанием различных объектов и механизмов. На них можно ссылаться при подготовке к занятию.
- Некоторые лабораторные работы имеют вариативность – таким образом, группу в процессе обучения можно разделить на несколько потоков.
- Зачастую текст задачи имеет неформальный стиль – он написан так, как обычно ставят задачи пользователи (четкого технического задания с описанием структур таблиц баз данных и отрисовки форм вы тут не встретите). В некоторых работах приведено умение анализировать задачу.
- В целом, любую лабораторную работу можно «вырвать» из книжки и отдать слушателям занятий в работу. Это готовый материал, рассчитанный по времени на стандартную «пару». Думаем, что такой формат подачи материала особенно пригодится при удаленной работе со слушателями.
- Первые лабораторные работы предназначены для описания общих принципов работы с системой, в том числе – с установкой, первым запуском и ориентированием в терминах и окружении системы.
- Далее лабораторные работы затрагивают стандартные модели создания объектов для хранения информации; показывают, какими способами можно извлекать информацию, как формировать отчеты.
- В сборнике нет лабораторных работ с тематикой бухгалтерского учета и не рассматриваются объекты сложных периодических расчетов. При его составлении мы ориентировались на знакомство слушателей с платформой 1С:Предприятие.

Лабораторная работа № 1

УСТАНОВКА СИСТЕМЫ 1С:ПРЕДПРИЯТИЕ 8

Откроем браузер и перейдем на сайт 1С:Учебного центра № 1: <https://uc1.1c.ru>. Найдем кнопку «Учебная версия 1С».



Открывается окно авторизации на сайте.

Бесплатная актуальная учебная версия платформы 1С.

Учебная версия 1С специально созданная для отработки навыков программирования в учебных целях. Идеально подойдёт для первичного ознакомления с платформой 1С. В учебной версии программы сохранены все основные функции, которые могут потребоваться при обучении разработке в 1С. Для доступа к скачиванию учебной версии платформы 1С авторизуйтесь или зарегистрируйтесь на сайте.

Пожалуйста авторизуйтесь, чтобы получить ссылки на Учебные версии

Email *

Пароль *

[Забыли пароль?](#)

[Регистрация](#)

[Войти](#)

Инструкции по установке в формате видео доступны по ссылкам

Если у вас нет *учетной записи*, тогда вам необходимо пройти процедуру регистрации. Для этого нажмите на кнопку «Регистрация».

The image shows a registration form with a light gray background and a thin gray border. Inside, there are two input fields: one for 'Email *' and one for 'Пароль *'. Below these fields is a red link labeled 'Забыли пароль?'. At the bottom is a red rectangular button labeled 'Регистрация'.

В открывшемся окне заполните все поля, поставьте галочку и нажмите на кнопку «Зарегистрироваться».

Если был введен номер телефона, то на ваш телефон придет СМС с кодом подтверждения, введите его в соответствующее поле. Нажмите на кнопку «Зарегистрироваться» еще раз.

Дождитесь письма с подтверждением регистрации, которое должно прийти на вашу почту, и авторизуйтесь в системе.

При успешной авторизации на странице для получения *учебной версии* появится ссылка на скачивание *учебной версии* для разных операционных систем: Windows, Linux, MacOS. Ниже на данной странице имеются видео-инструкции по установке платформы для разных операционных систем.

В данном случае будет рассмотрен вариант установки для ОС Windows.

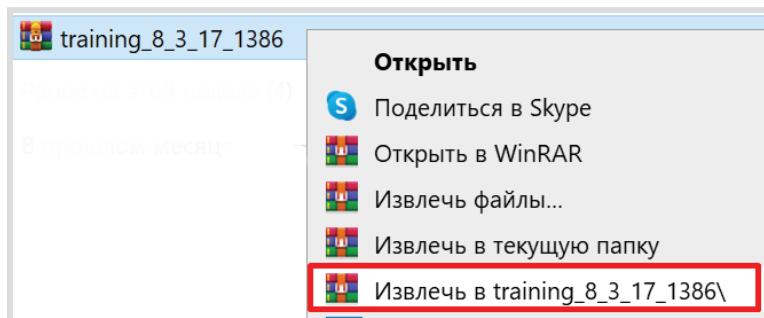
Найдите на странице раздел «Для Windows» и щелкните по ссылке для скачивания *учебной версии*.

Версия для программирования

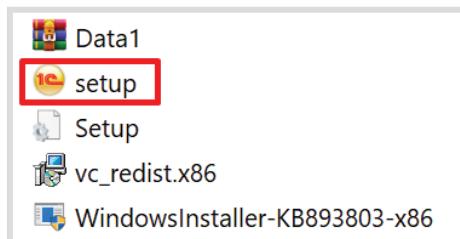
Для Windows

Скачать учебную версию можно по ссылке - https://uc1.1c.ru/files/platform/training_8_3_17_1386.zip

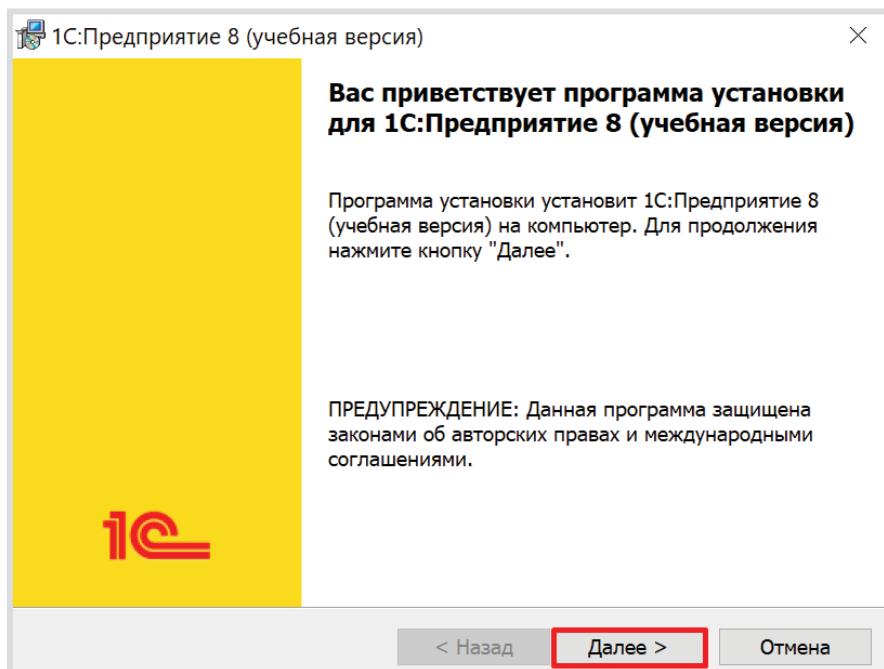
При нажатии на ссылку начнется скачивание архива. По окончании скачивания данный архив нужно распаковать с помощью любого доступного архиватора:



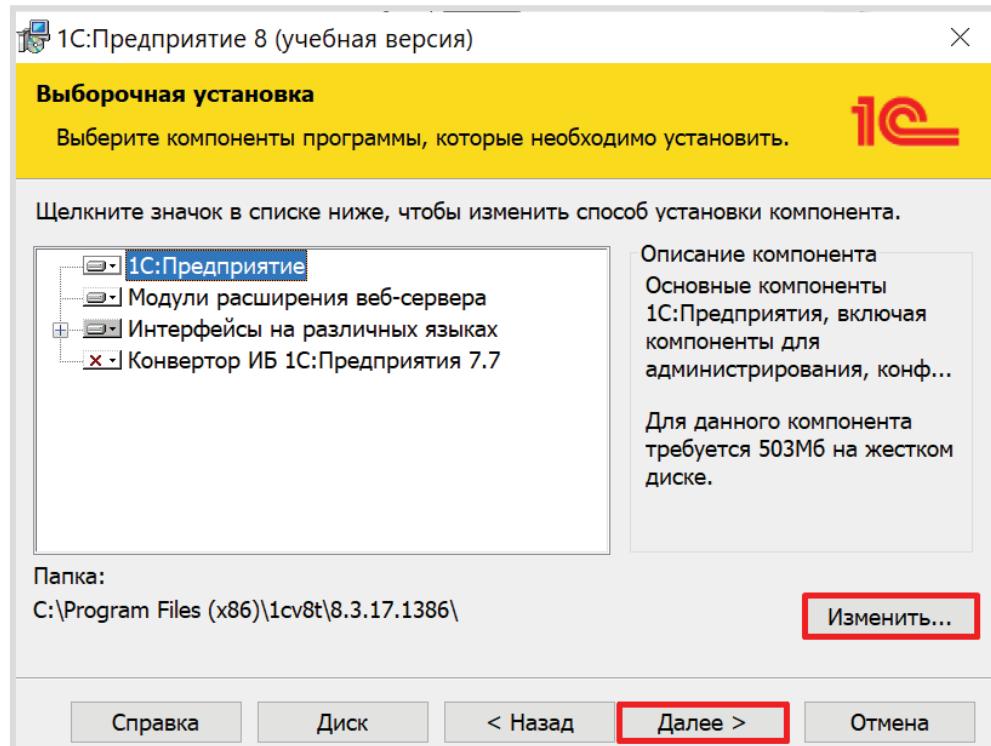
По завершении работы архиватора появится папка «training_8_3_17_1386». Откройте эту папку и запустите исполняемый файл «setup.exe».



Откроется окно установки 1С:Предприятия. Нажимаем на кнопку «Далее»:



На следующем этапе установки необходимо выбрать компоненты, которые будут установлены на ваш компьютер.



Внимание!

Проверьте, что компонент «1С:Предприятие» не отмечен знаком «крестика»!

Если вы планируете изучать работу *веб-клиента* или *мобильной платформы*, то дополнительно нужно установить «Модули расширения веб-сервера». Причем обратите внимание, что веб-сервер должен уже быть установлен заранее!

Если вы хотите выбрать язык интерфейса, отличный от стандартных (русский, английский), то раскройте ветку «Интерфейсы на различных языках» и отметьте галочками нужные языки.

Информация

Платформа «1С:Предприятие 8» локализована на 22 языка, включая английский, немецкий, французский, китайский, вьетнамский.

Механизмы локализации, заложенные в платформу, позволяют использовать различные языки как при разработке прикладного решения, так и при работе пользователей прикладного решения. Кроме этого, на уровне платформы поддерживаются различные национальные стандарты представления дат, чисел и т. д.

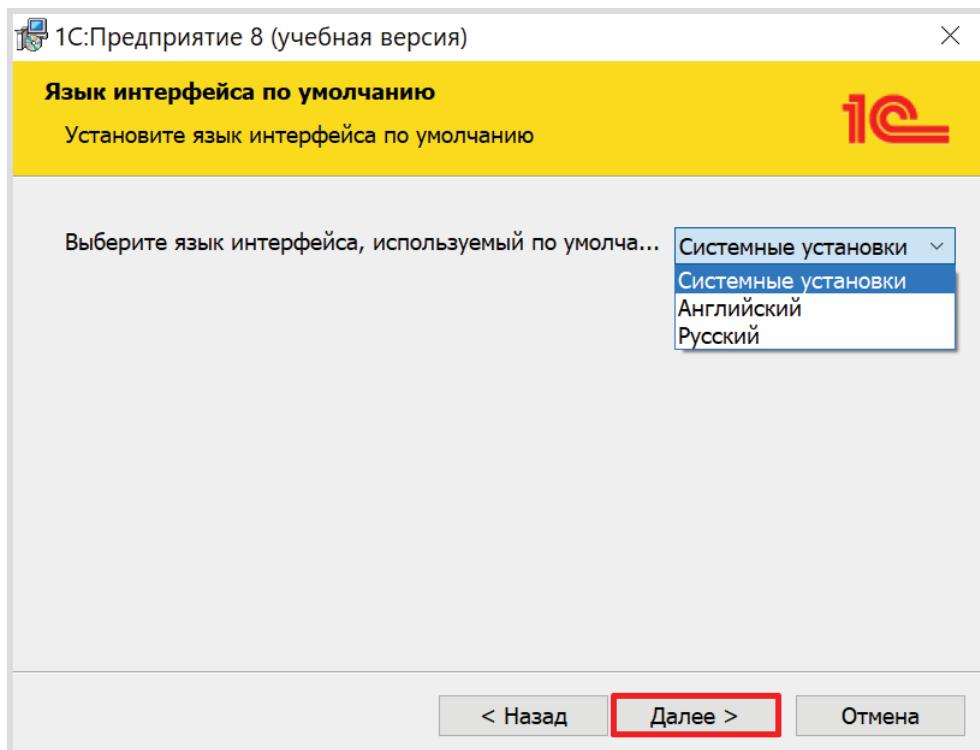
Если вы хотите изменить путь для установки файлов того или иного компонента, нажмите на кнопку «Изменить» и выберите новую папку.

Обратите внимание, что можно выбрать путь установки для каждого компонента по отдельности.

Для перехода к следующему шагу установки нажмите кнопку «Далее».

На следующем этапе необходимо выбрать язык интерфейса *по умолчанию*. Если выбрать «Системные настройки», то система будет запускаться на языке пользователя, установленном в операционной системе.

По окончании выбора языка интерфейса нажмите на кнопку «Далее».

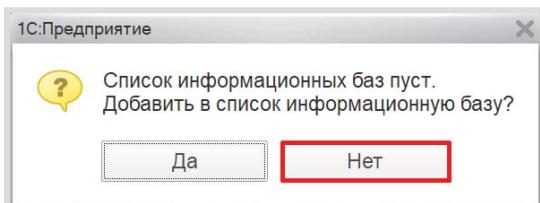


На последнем этапе установки нужно нажать на кнопку «Установить». Начнется процесс установки 1C:Предприятия на компьютер.

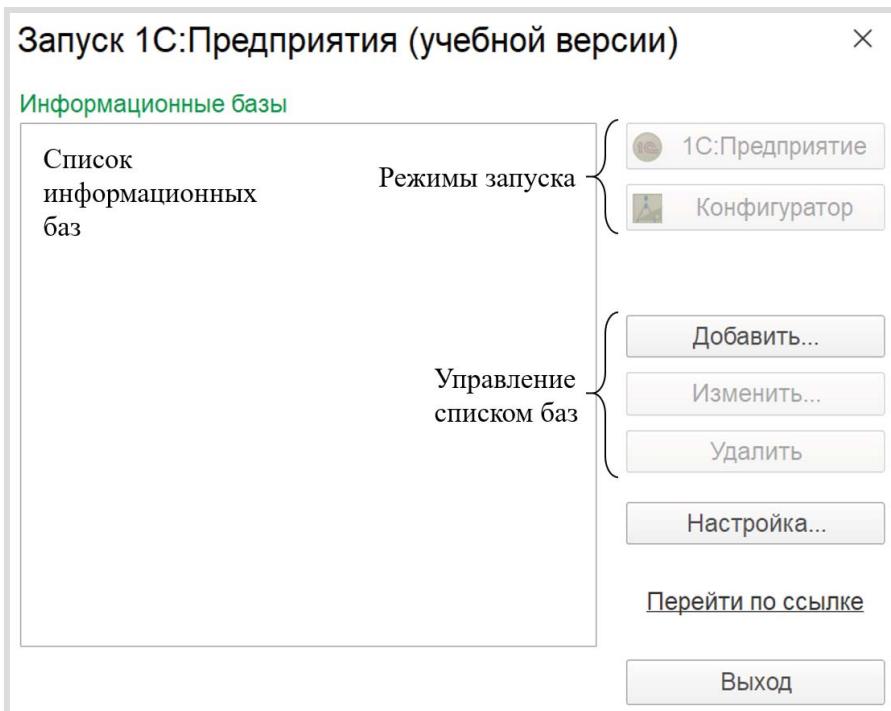
По завершении установки нажмите на кнопку «Готово». На рабочем столе появится значок запуска учебной версии системы 1C:Предприятие.



Запустим программу. При первом запуске система предложит добавить новую информационную базу, но мы этого делать не будем.



Откроется окно запуска 1С:Предприятия.

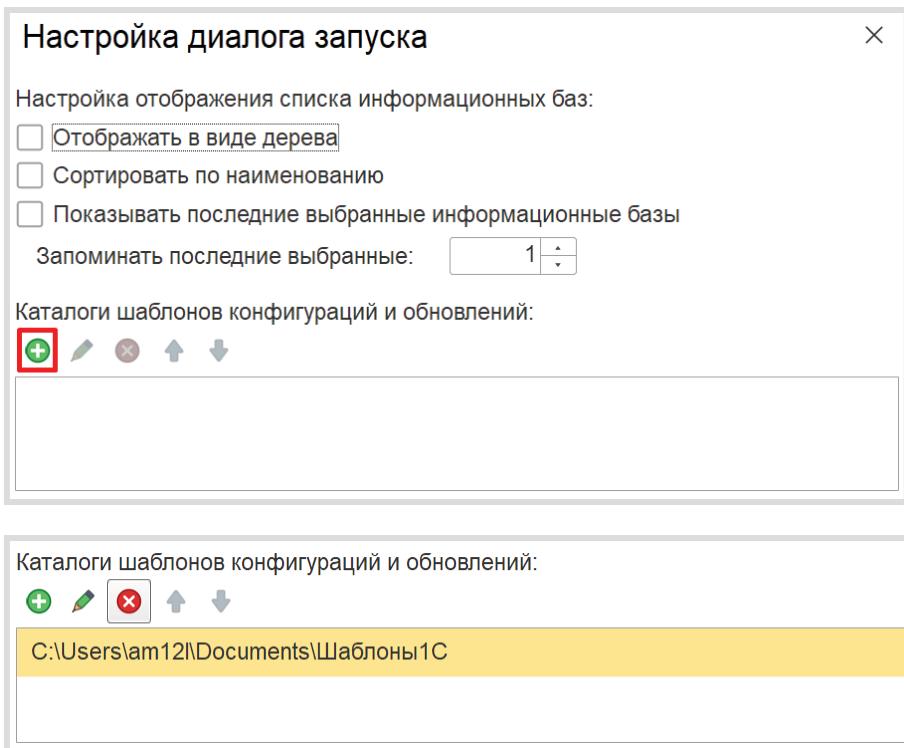


Далее перед нами стоит задача *переопределить путь*, где платформа будет искать шаблоны для создания новых информационных баз.

Информация

Информационные базы могут создаваться как с нуля, так и из шаблонов. Установка шаблонов выполняется специальной программой установки, которая образуется автоматически при создании комплекта поставки из конфигуратора. Шаблон представляет собой совокупность файлов поставки, файла манифеста и сопутствующих файлов, из которых производится создание информационной базы.

Откроем настройку окна запуска системы. Для этого нажмите на кнопку «Настройка». В открывшемся окне добавьте удобный для вас путь к каталогу шаблонов.



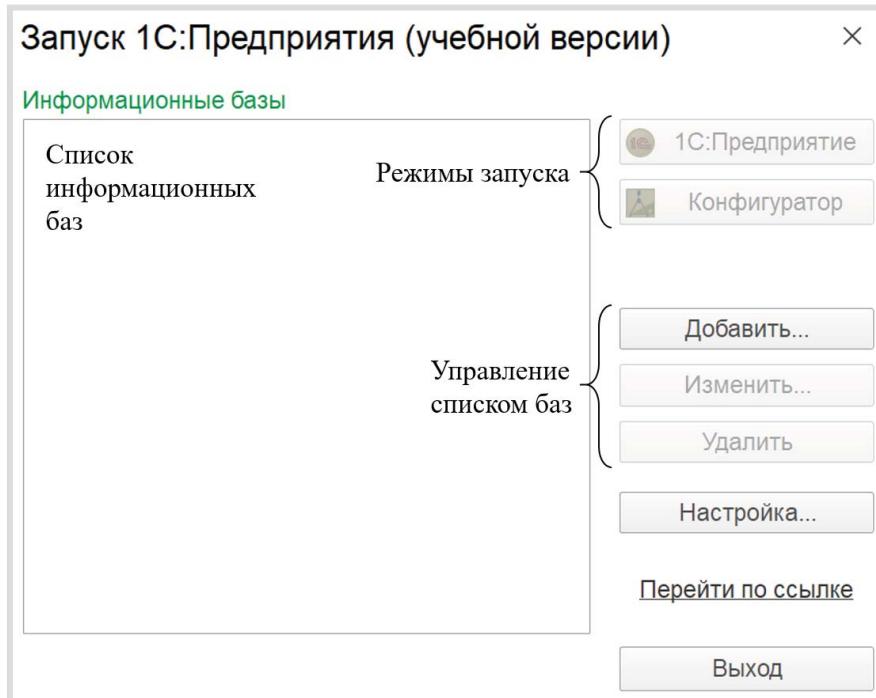
На этом установка и первичная настройка платформы завершена.

Лабораторная работа № 2

ОСНОВНЫЕ ПРИНЦИПЫ РАБОТЫ С ПЛАТФОРМОЙ

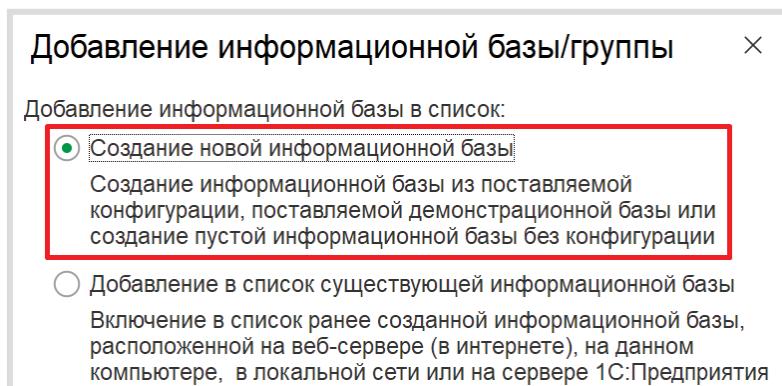
Добавление пустой информационной базы

Запустите *учебную версию 1С:Предприятия*. Перед вами появится окно запуска.



Добавим новую информационную базу. Для этого нужно нажать на кнопку «Добавить».

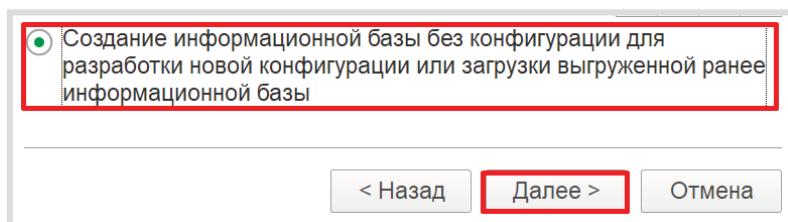
Откроется окно добавления информационной базы/группы. Выберите вариант создания новой информационной базы и нажмите на кнопку «Далее».



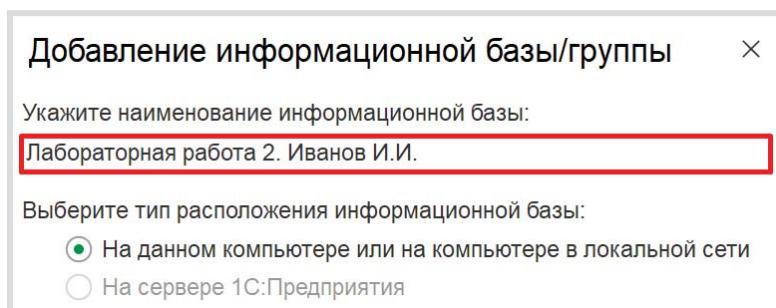
На следующем этапе можно создать новую информационную базу из заготовки (шаблона) или совершенно пустую.

Для выполнения лабораторных работ нам понадобятся только пустые информационные базы.

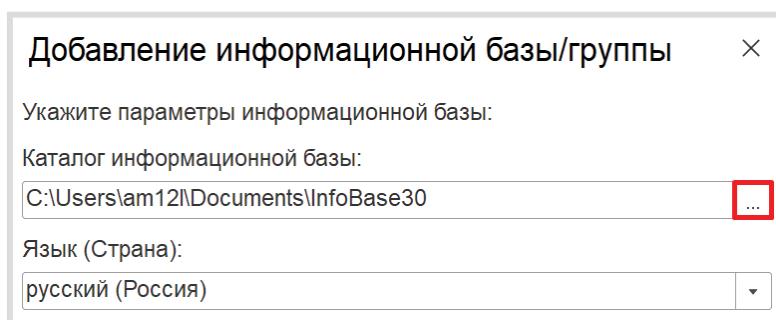
Выберите нужный пункт и нажмите на кнопку «Далее».



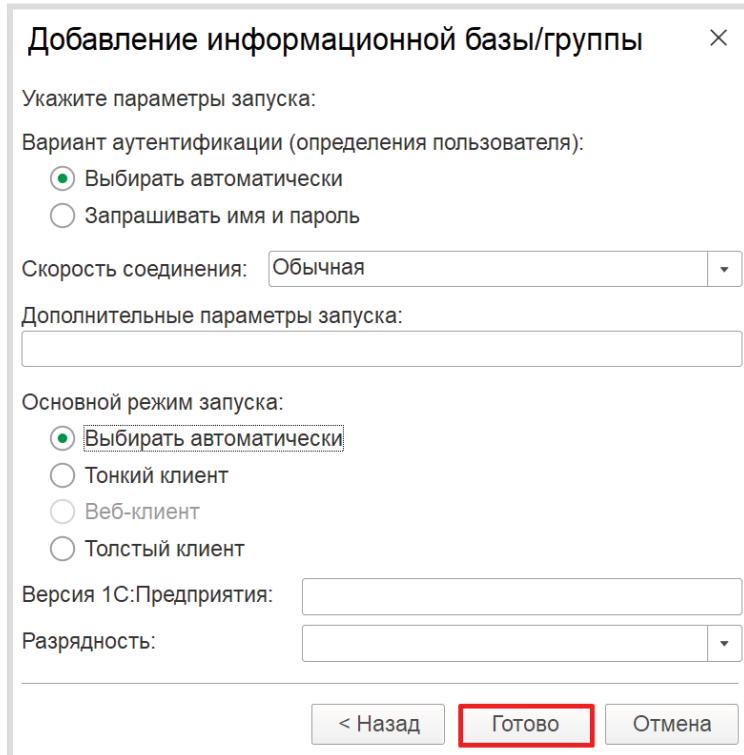
Дайте имя вашей информационной базе и нажмите кнопку «Далее».



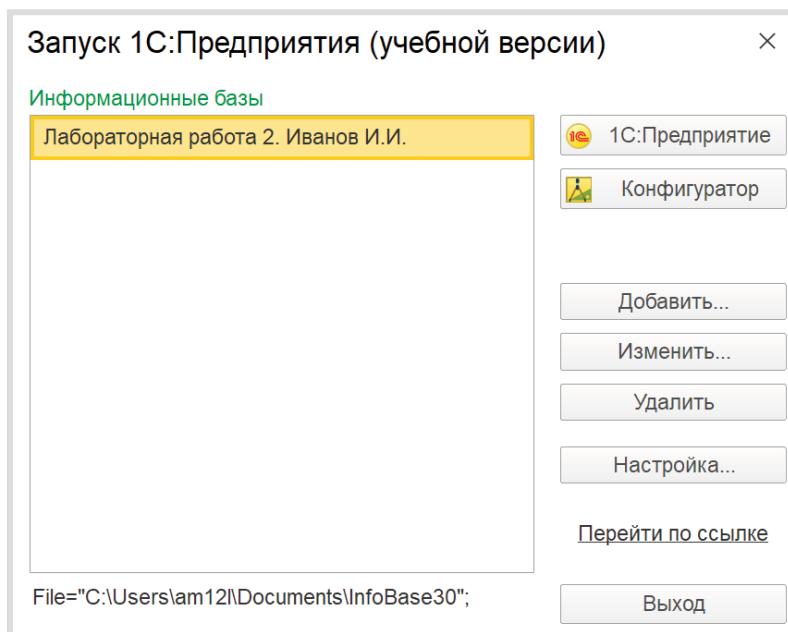
При необходимости можно изменить каталог для установки информационной базы. Нажмите на кнопку «Далее».



В последнем окне оставьте все настройки *по умолчанию* и нажмите на кнопку «Готово».



В списке информационных баз появится созданная вами информационная база.



Режимы запуска «1С:Предприятие» и «Конфигуратор»

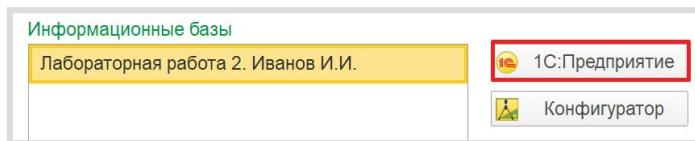
Любая информационная база может быть запущена в двух режимах: «1С:Предприятие» и «Конфигуратор».

Определение

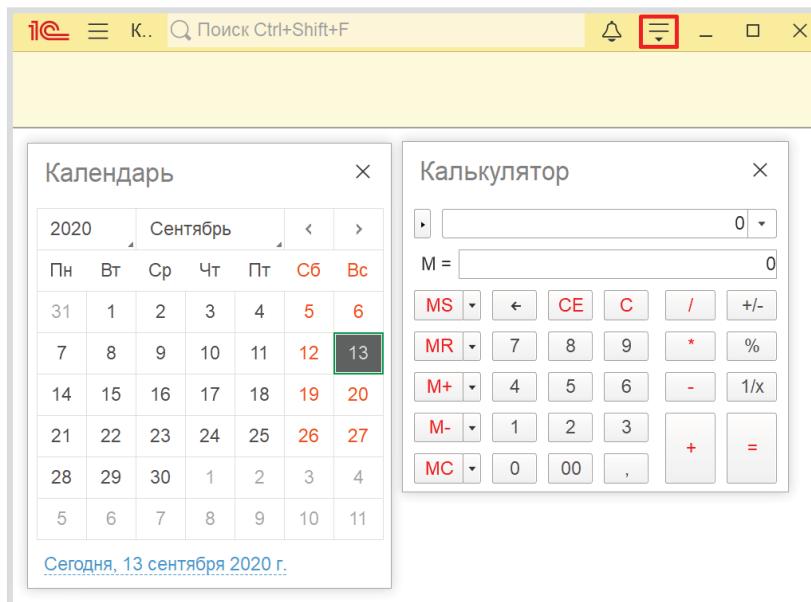
Режим «1С:Предприятие» — это пользовательский режим. В этом режиме пользователи добавляют, изменяют, удаляют данные, формируют отчеты и выполняют другие прикладные задачи.

Режим «Конфигуратор» — это режим для разработчика. В этом режиме разрабатываются прикладные решения и выполняется администрирование информационных баз.

Для начала запустим созданную информационную базу в режиме «1С:Предприятие». Для этого выберите свою базу в списке баз и нажмите на кнопку «1С:Предприятие».

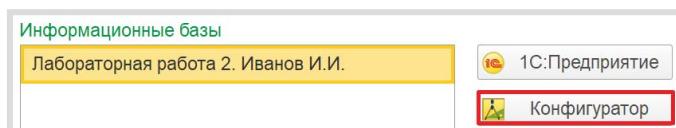


Запустится окно режима «1С:Предприятие». Система автоматически сформирует для пользователя форму, а также добавит некоторые дополнительные функции, например, калькулятор и календарь. Их можно вызвать при нажатии на кнопку, выделенную рамкой.



Закроем пользовательский режим с помощью «крестика» в правом верхнем углу.

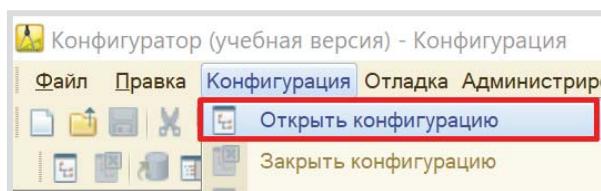
Откроем систему «1С:Предприятие» заново. На этот раз запустим информационную базу в режиме «Конфигуратор».



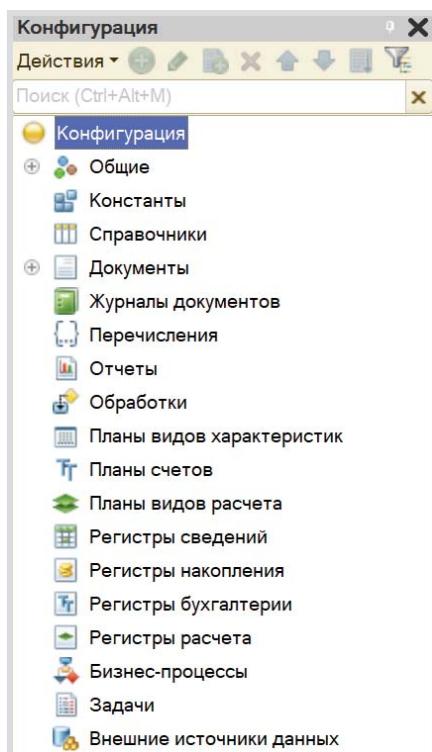
Откроется режим для разработчика. Здесь и будет производиться конфигурация (настройка) информационных баз.

Открытие окна конфигурации

Для разработки информационных баз нам понадобятся специальные объекты. Чтобы их включить, необходимо выбрать пункт меню «Конфигурация» → «Открыть конфигурацию».

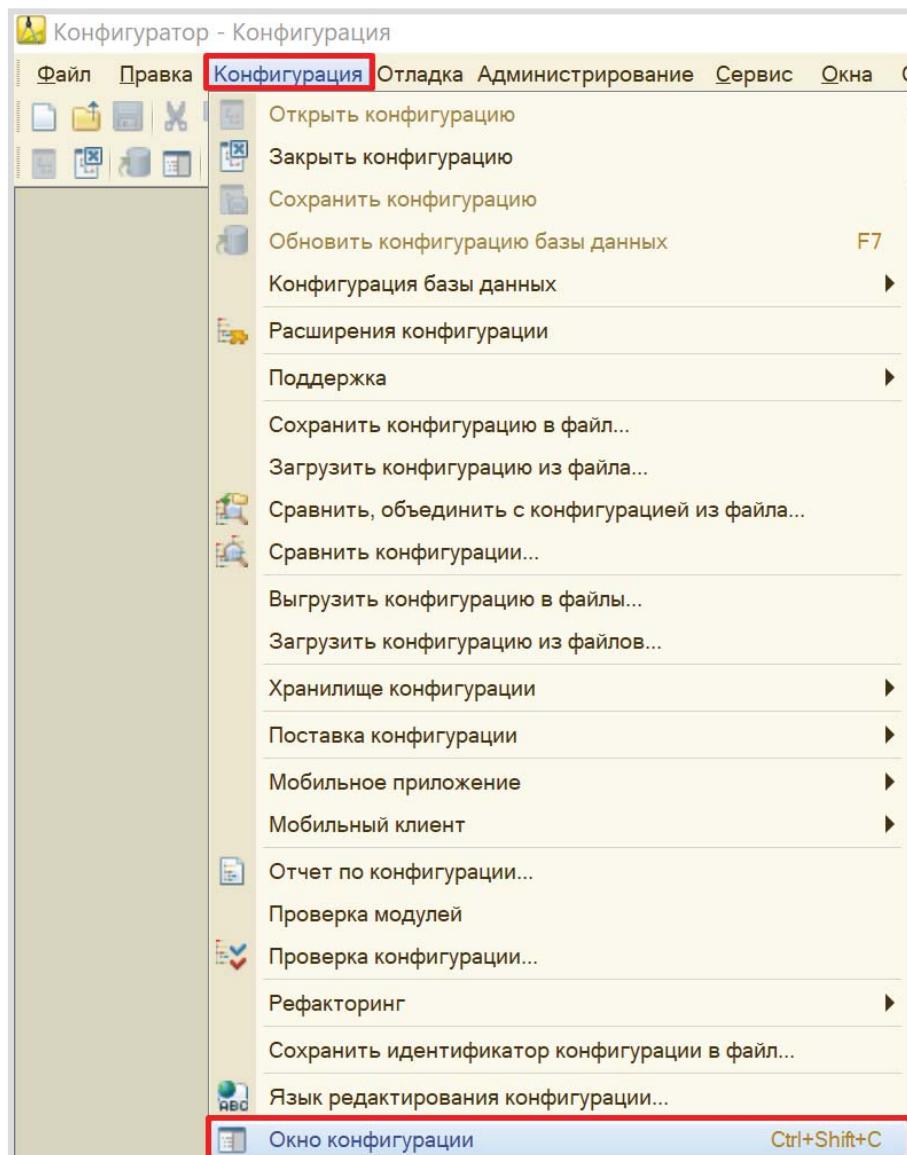


Откроется окно конфигурации.



В окне конфигурации находится дерево конфигурации. Дерево состоит из различных объектов конфигурации. Каждый из данных объектов выполняет определенную функцию, которые будут рассмотрены в следующих лабораторных работах.

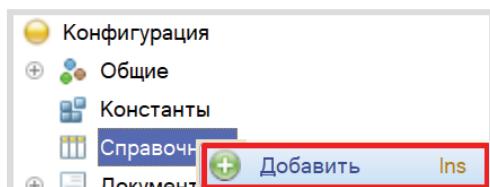
Если окно конфигурации было закрыто, то повторно открыть его можно, выбрав пункт меню «Конфигурация» → «Окно конфигурации».



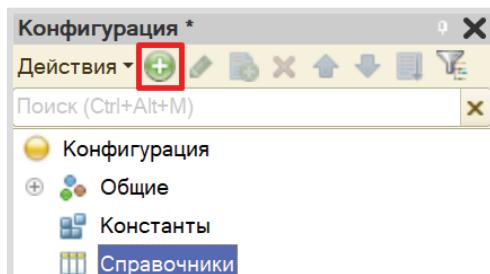
Добавление нового объекта конфигурации

Существует несколько способов добавления/создания нового объекта конфигурации. Рассмотрим на примере добавления нового *справочника*.

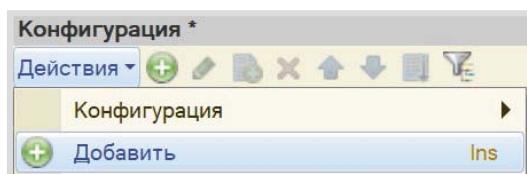
1. Щелкните по нужному объекту конфигурации левой кнопкой мыши и нажмите клавишу «Insert».
2. Щелкните по нужному объекту конфигурации правой кнопкой мыши и выберите пункт меню «Добавить».



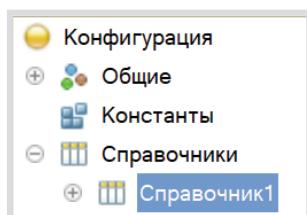
3. Щелкните по нужному объекту конфигурации левой кнопкой мыши и нажмите на кнопку «Добавить».



4. Щелкните по нужному объекту конфигурации левой кнопкой мыши и нажмите «Действия» → «Добавить».



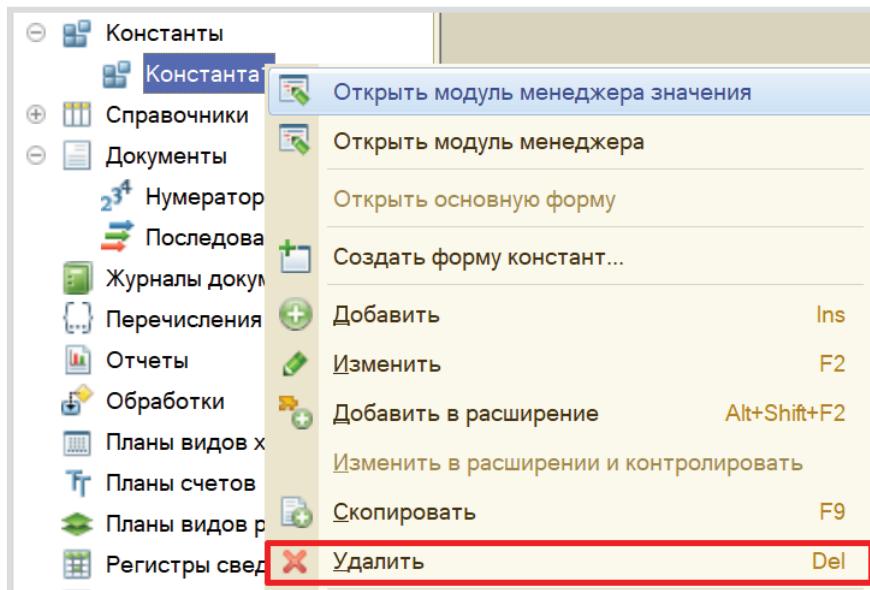
Все варианты приведут к одному результату: созданию нового объекта конфигурации вида *справочник*.



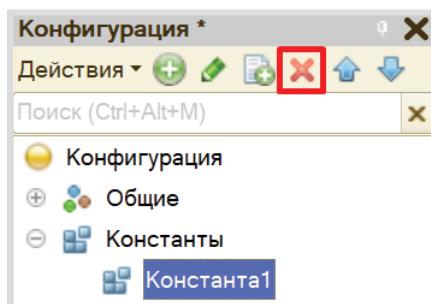
Удаление существующего объекта конфигурации

Удалить объект конфигурации можно также несколькими способами:

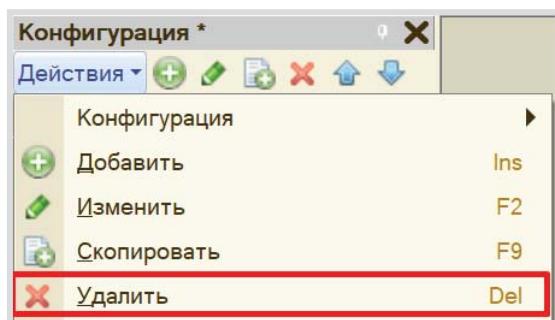
1. Щелкните по выбранному объекту левой кнопкой мыши и нажмите клавишу «Delete».
2. Щелкните по выбранному объекту правой кнопкой мыши и в контекстном меню выберите пункт «Удалить».



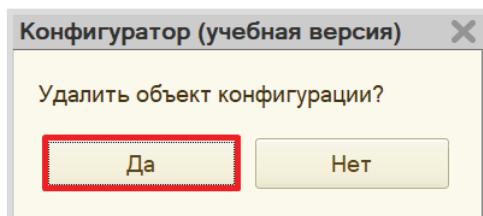
3. Щелкните по выбранному объекту левой кнопкой мыши и нажмите кнопку «Удалить».



4. Щелкните по выбранному объекту левой кнопкой мыши, выберите в меню «Действия» → «Удалить».

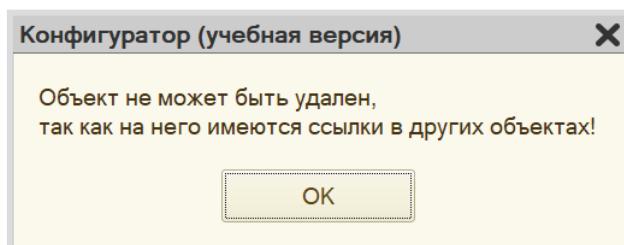


Любой из перечисленных методов даст одинаковый результат: появится окно, в котором необходимо подтвердить удаление объекта.



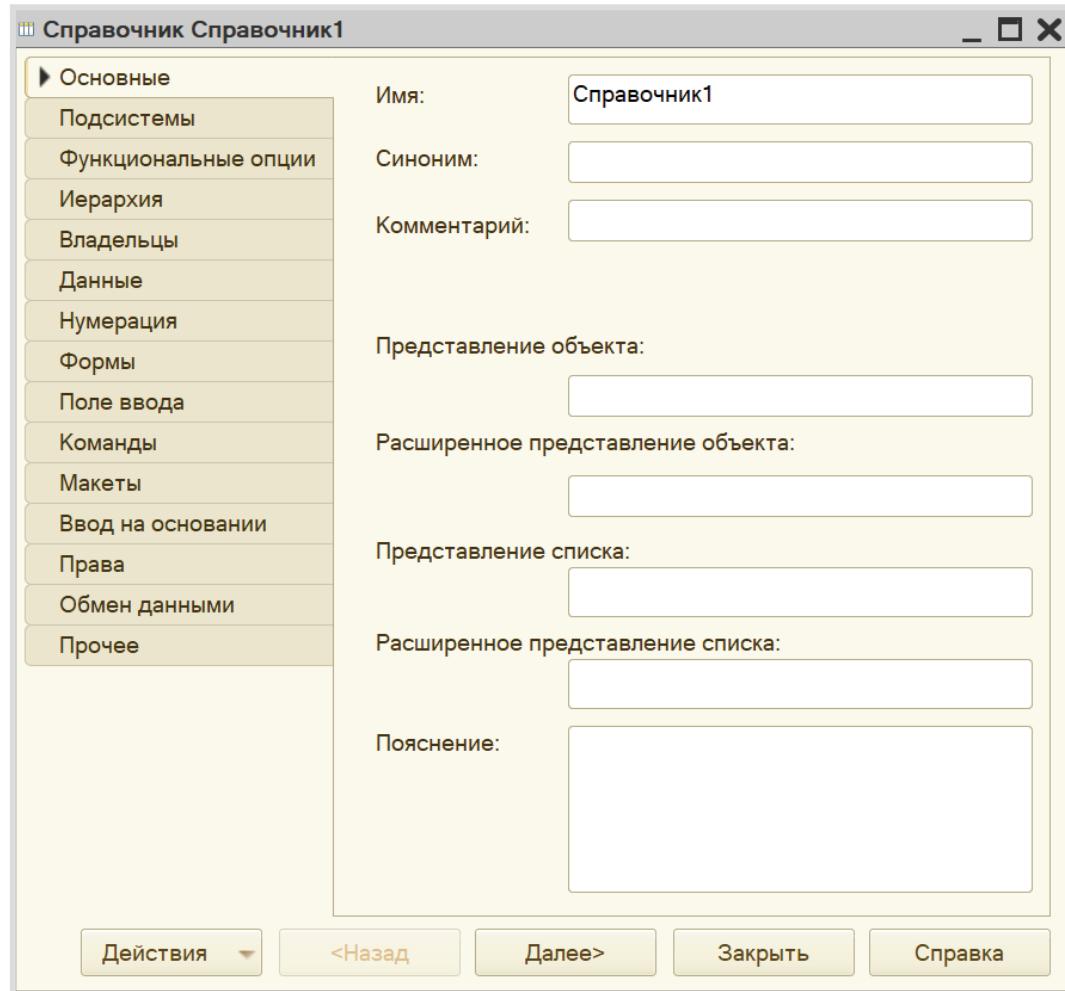
Если объект не связан ссылками с другими объектами, то он будет удален.

Если же данный объект связан с другим объектом с помощью ссылки, то появится окно с предупреждением о том, что объект не может быть удален. Объект удалить не получится, пока ссылочная связь не будет разорвана.



Окно редактирования объектов конфигурации

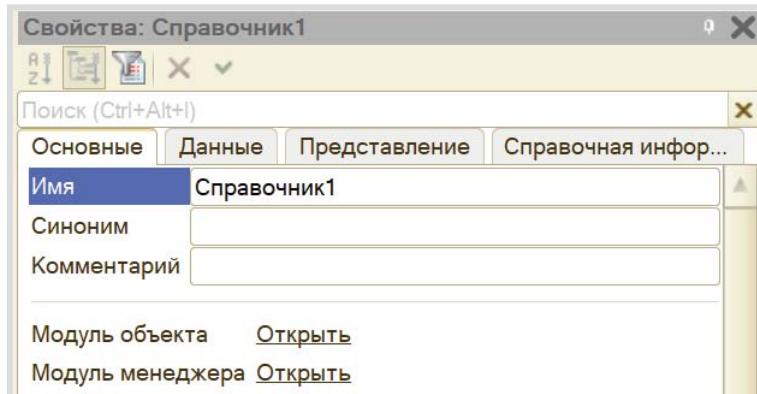
Редактирование объектов конфигурации может производиться в окне редактирования объекта конфигурации.



Для настроек того или иного объекта необходимо переключаться между вкладками.

Палитра свойств объекта конфигурации

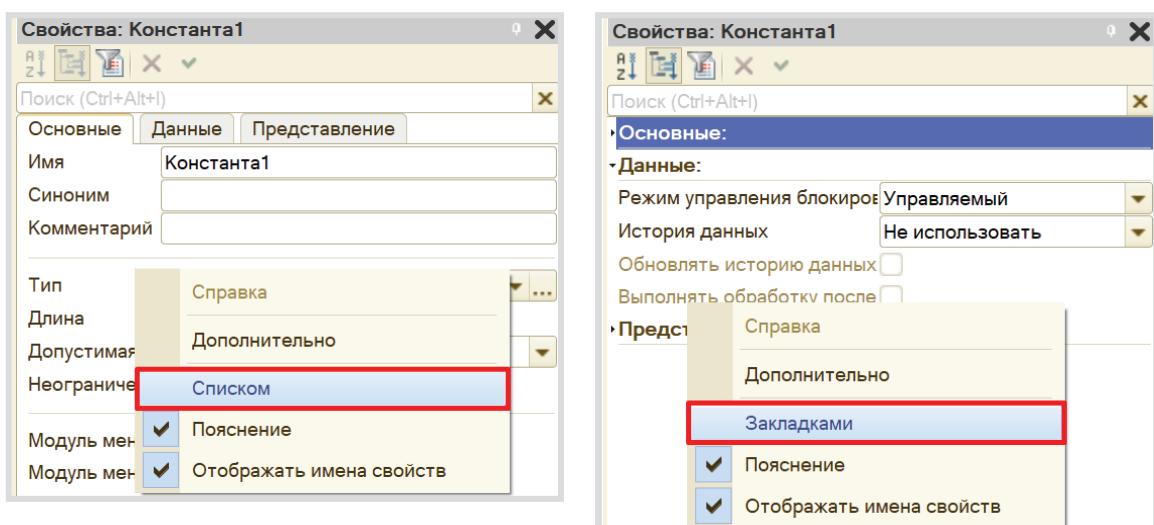
У любого объекта конфигурации может быть открыта палитра свойств. Для ее вызова нужно вызвать контекстное меню правой кнопкой мыши и выбрать пункт «Свойства».



Настройки в окне редактирования объекта конфигурации и в палитре свойств взаимосвязаны. Например, если изменить имя в палитре свойств, то оно обязательно изменится в окне редактирования объекта конфигурации.

У некоторых объектов (например, констант) нет окна редактирования конфигурации. Все настройки осуществляются исключительно в палитре свойств.

Палитра свойств может отображаться списком или закладками. Чтобы изменить отображение элементов, нужно в пустом месте палитры свойств щелкнуть правой кнопкой мыши и выбрать пункт «Списком» или «Закладками», в зависимости от предпочтений.



Важные свойства объектов конфигурации

Каждый объект конфигурации должен иметь *имя, синоним и тип*.

Определение

Имя – это идентификатор объекта в системе. Имя должно быть уникальным и записано по определенным правилам: имя должно состоять из одного слова (не иметь пробелов), начинаться с буквы и не содержать специальных символов, кроме «_».

Так, если мы хотим дать константе имя «*Дата создания организации*», то правильная запись будет выглядеть следующим образом:

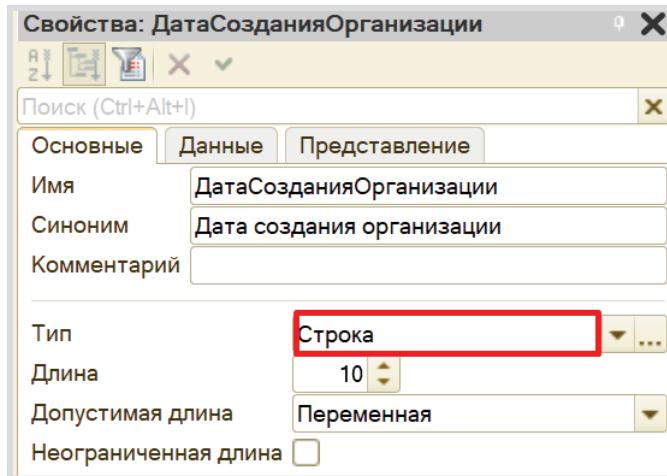
- «*ДатаСозданияОрганизации*»;
- «*Дата_создания_организации*».

Если при написании имени была допущена ошибка, то система выдаст сообщение об ошибке.

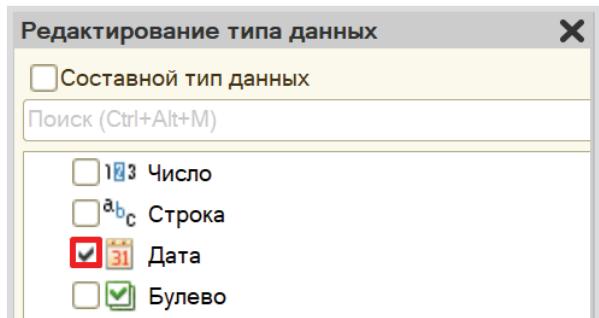
Определение

Синоним – удобное для пользователя название. В написании синонима нет никаких ограничений. Поле синонима заполняется автоматически после заполнения имени и может быть отредактировано.

Помимо имени и синонима необходимо обратить внимание на *тип данных*, который данная константа будет хранить. По умолчанию тип всегда заполнен одним из допустимых значений. Например, константа по умолчанию всегда имеет строковый тип с длиной 10 символов.

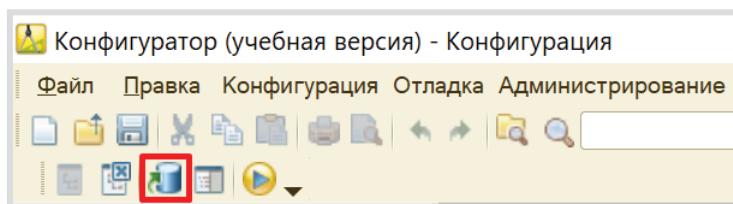


Тип данных может быть изменен. Для этого нужно нажать на «многоточие» и выбрать интересующий тип данных, поставив напротив него галочку.

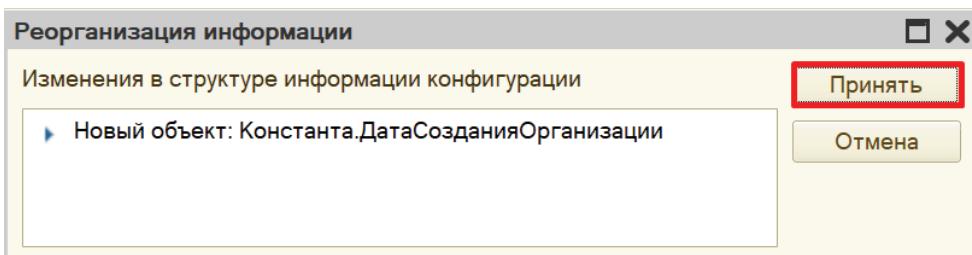


Обновление конфигурации

Для того чтобы обновить конфигурацию, нужно нажать на кнопку «Обновить конфигурацию базы данных» или использовать горячую клавишу «F7».

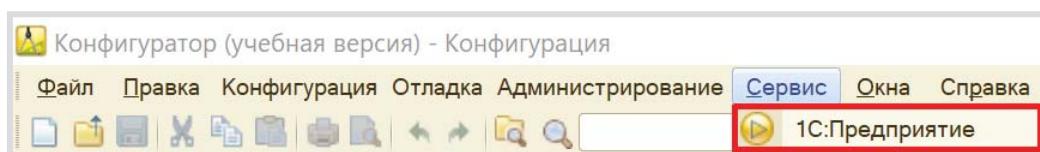


Появится окно, в котором необходимо подтвердить, что вы согласны с внесением изменений в структуру конфигурации.

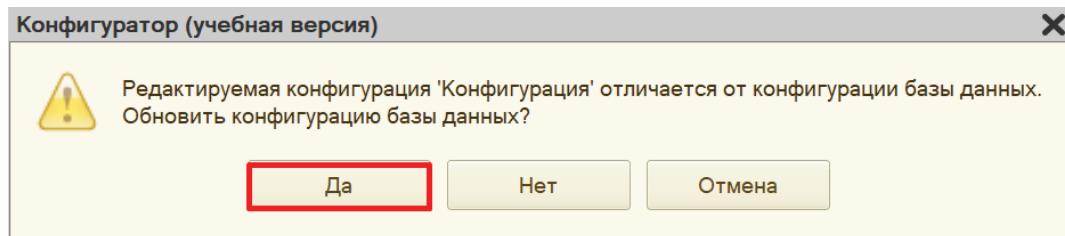


Запуск режима «1С:Предприятие» из режима «Конфигуратор»

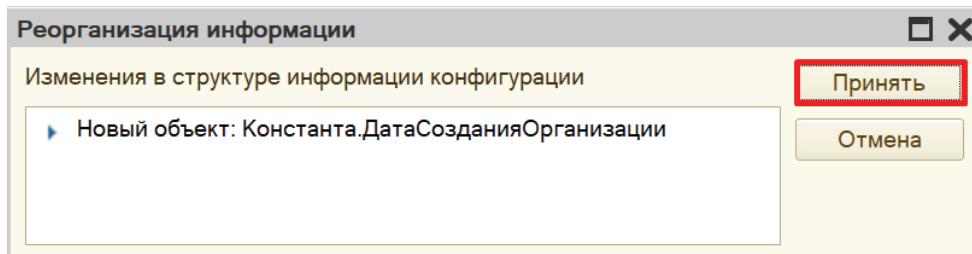
Для мгновенного запуска пользовательского режима выберите в меню «Сервис» → «1С:Предприятие».



Если вы еще не произвели обновление конфигурации, то система спросит, хотите ли вы обновить конфигурацию.



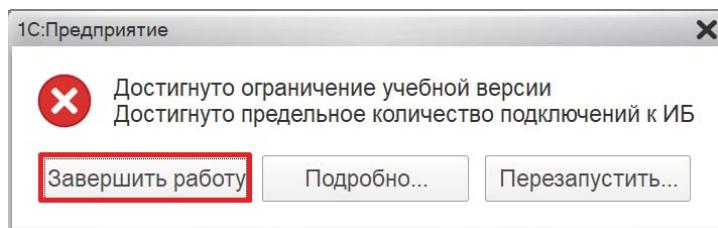
Затем появится окно реорганизации информации.



Начнется запуск пользовательского режима.

Внимание!

Обязательно закрывайте пользовательский режим по окончании работы с ним! При попытке обновить конфигурацию с открытым пользовательским режимом появится сообщение об ошибке:

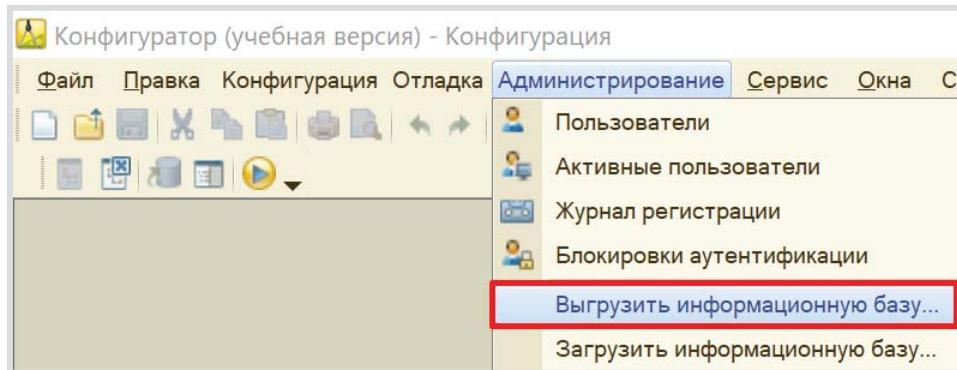


Попытку запустить режим «1С:Предприятие» повторно система рассматривает как попытку подключения еще одного пользователя. Но учебная версия платформы накладывает ограничение: с информационной базой может работать только один пользователь, в связи с чем и возникает ошибка.

Выгрузка информационной базы

При необходимости информационная база может быть выгружена в файл. Нужно сделать следующее:

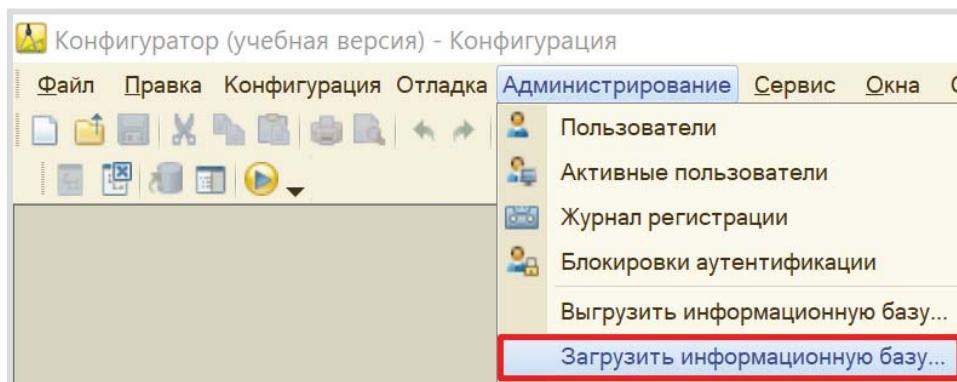
1. Обновить конфигурацию.
2. Выбрать в меню «Администрирование» → «Выгрузить информационную базу...».



Затем выберите папку, в которую хотите выгрузить свою информационную базу. При необходимости переименуйте файл. Нажмите на кнопку «Сохранить». В указанном месте должен появиться файл с расширением «dtt».

Загрузка информационной базы

Загрузка информационной базы происходит аналогично. Выберите в меню «Администрирование» → «Загрузить информационную базу».



Выберите на своем компьютере файл с разрешением «dtt» и нажмите на кнопку «Открыть».

Внимание!

Все данные текущей информационной базы будут удалены!

Лабораторная работа № 3

РАЗРАБОТКА КОНФИГУРАЦИИ ДЛЯ ОРГАНИЗАЦИИ ХРАНЕНИЯ ИНФОРМАЦИИ О СТУДЕНТАХ И ИЗУЧАЕМЫХ ИМИ ПРЕДМЕТАХ

Сложность: *

Теги: справочник, иерархия групп и элементов,
схема компоновки данных

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для организации хранения информации о студентах и изучаемых ими предметах. Студенты должны быть разделены по группам.

1. Необходимо хранить следующую информацию о каждом студенте:

- ФИО;
- номер телефона в формате +7(999)999-99-99;
- перечень изучаемых предметов.

2. Нужно построить отчет, формирующий список студентов по предметам. А также разработать возможность устанавливать отбор по конкретному предмету.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

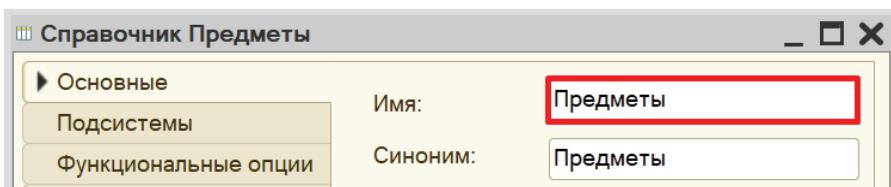
«Заказчик просит разработать конфигурацию для организации хранения информации о студентах и изучаемых ими предметах.»

Данная часть условия говорит нам о том, что в информационной системе должна храниться информация о студентах и предметах. Для этой цели необходимо использовать *справочник*.

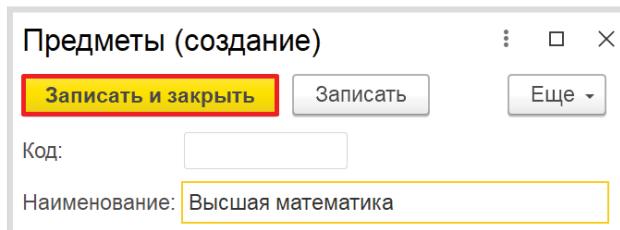
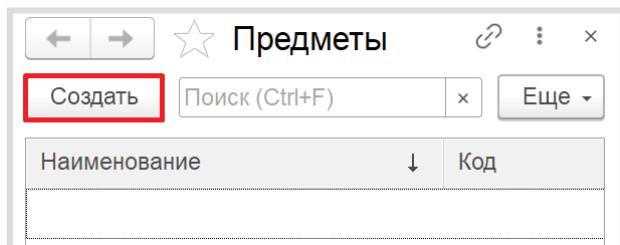
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Добавим справочник «Предметы».

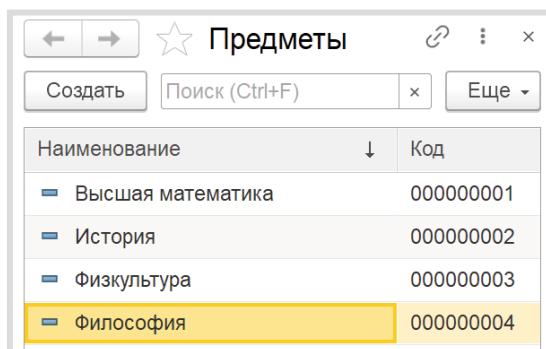


Перейдем в режим «1С:Предприятие» и добавим несколько предметов в справочник.



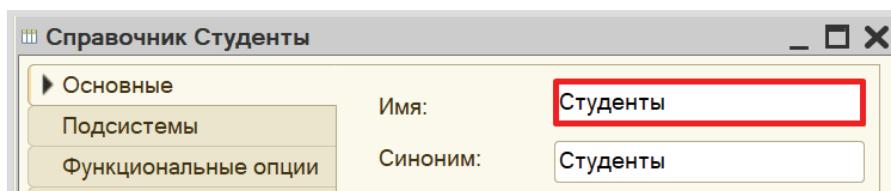
Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.

Аналогично добавим еще несколько предметов в список.



Таким образом, мы реализовали хранение информации об учебных предметах.

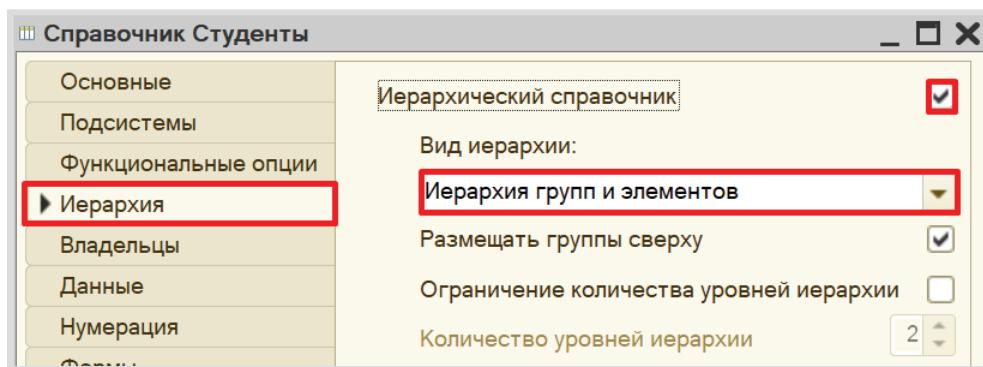
Добавим еще один справочник – «Студенты».



В системе 1С:Предприятие справочники могут быть определенным образом структурированы. Близкие по смыслу (в том или ином понимании) элементы можно объединять в одну группу, чтобы не путать их с элементами другого содержания. В качестве примера можно привести справочник «Номенклатура», в котором могут быть созданы группы «Товары» и «Услуги». Для этого используется *иерархия*.

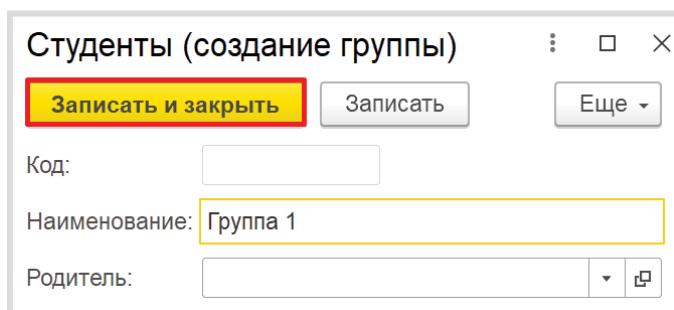
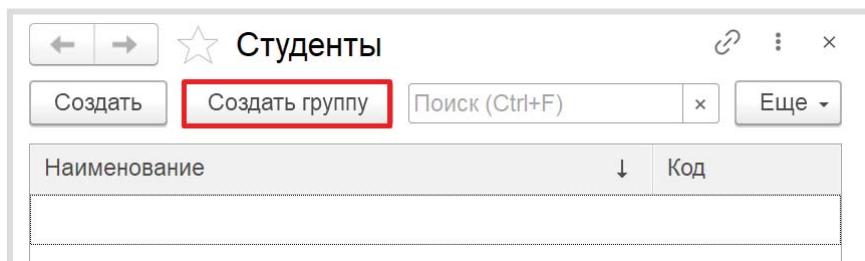
У нас есть необходимость разделить элементы справочника «Студенты» по группам.

Переходим на вкладку «Иерархия» и устанавливаем галочку «Иерархический справочник». Выбираем вид иерархии «Иерархия групп и элементов». Данный вид иерархии позволит создавать отдельно группы элементов и сами элементы.

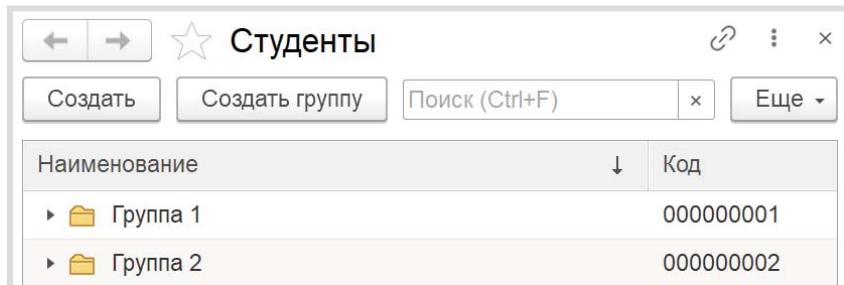


Теперь помимо элементов справочника мы сможем создавать группы. Группы справочника и будут группами студентов.

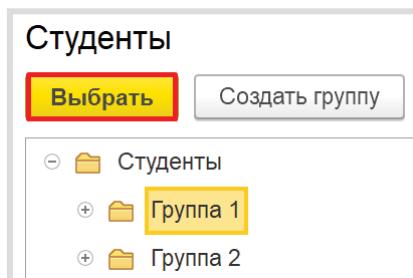
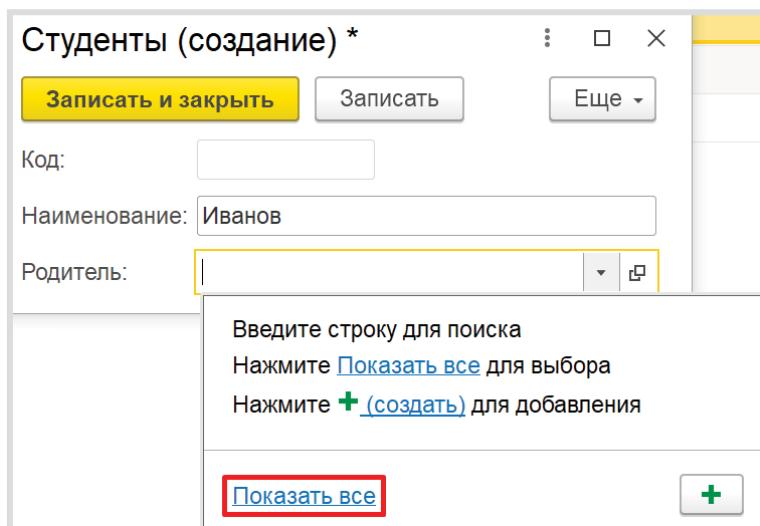
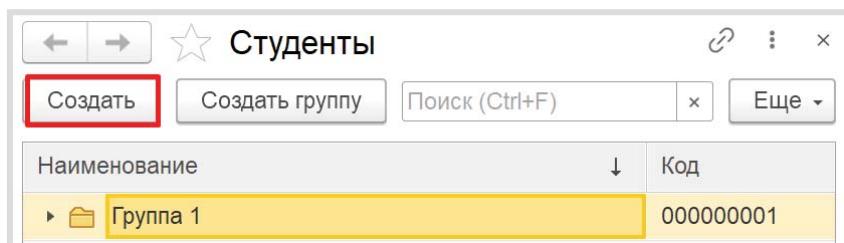
Запустим режим «1С:Предприятие» и попробуем создать несколько групп студентов.



Аналогично создадим еще одну группу.



Теперь создадим студента и добавим его в первую группу, указав ее в поле «Родитель».



Студенты (создание) *

Записать и закрыть **Записать** **Еще**

Код:

Наименование: Иванов

Родитель: Группа 1

Аналогично добавим студента «Петров» в группу 2.

Студенты (создание)

Записать и закрыть **Записать** **Еще**

Код:

Наименование: Петров

Родитель: Группа 2

Самостоятельно добавьте несколько студентов в созданные группы.

Перейдем к формированию структуры справочника «Студенты». Перейдем на вкладку «Данные». В первую очередь, необходимо изменить длину наименования, поскольку 25 символов на ввод ФИО может не хватить. Увеличим его до 60.

Затем откроем стандартные реквизиты.

Справочник Студенты

Основные **Подсистемы** **Функциональные опции** **Иерархия** **Владельцы** **Данные** **Нумерация** **Формы** **Поле ввода** **Команды** **Макеты** **Ввод на основании** **Права** **Обмен данными** **Прочее**

Длина кода: 9
Длина наименования: 60

Тип кода: Число / Стока
Основное представление: В виде кода / В виде наименования

Реквизиты

Габличные части

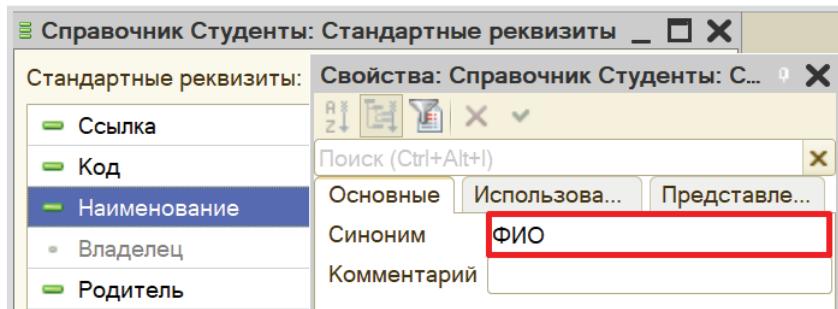
Стандартные реквизиты **Характеристики**

Общие реквизиты

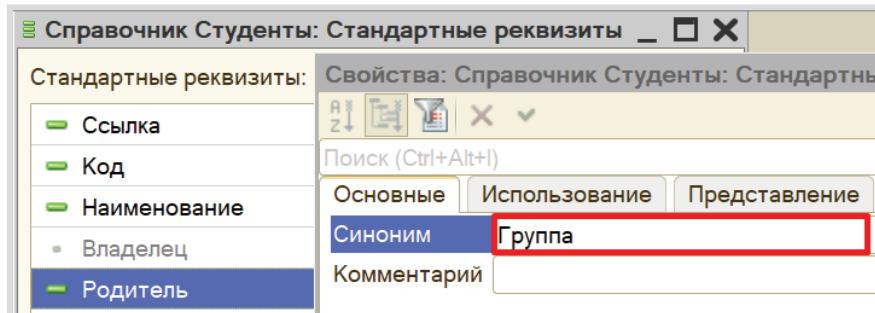
Действия **<Назад** **Далее>** **Закрыть** **Справка**

Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Мы можем добавить синоним для изменения названий данных полей для пользователя.

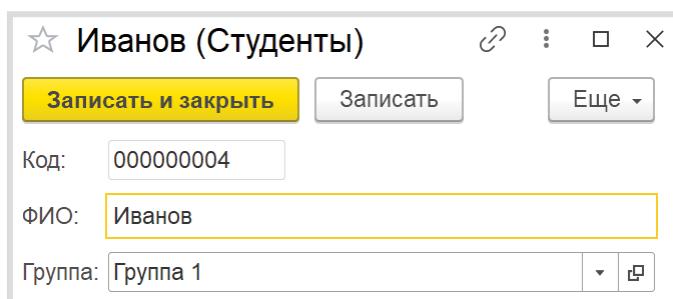
Изменим синоним реквизита «Наименование» на «ФИО».



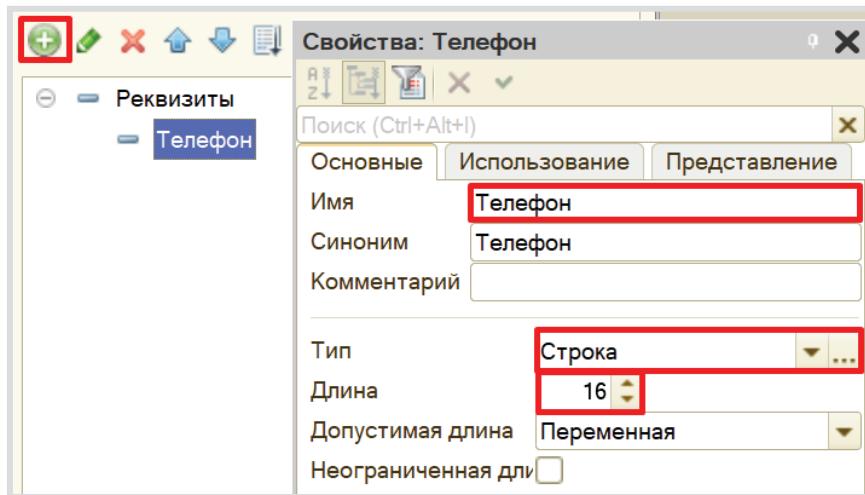
Аналогично изменим синоним реквизита «Родитель» на «Группа».



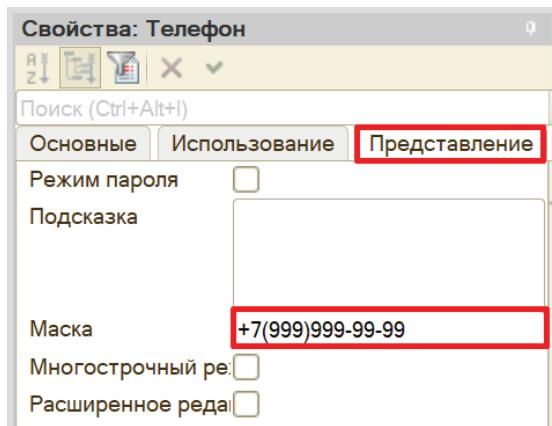
Теперь карточка студента должна выглядеть следующим образом:



Помимо *ФИО* и *группы* карточка должна хранить информацию о номере телефона студента. Добавим новый реквизит «Телефон» на вкладке «Данные».



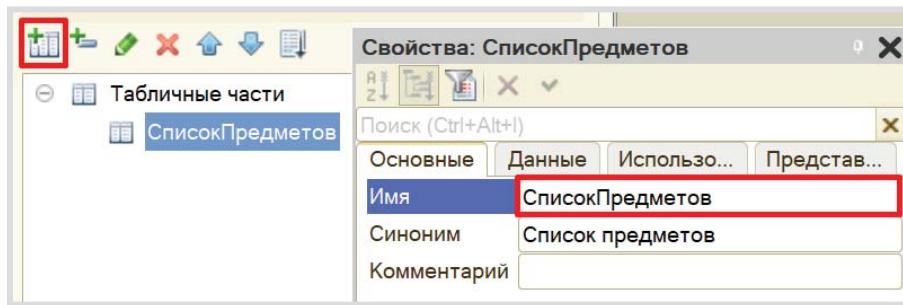
Для фиксирования формата ввода номера телефона воспользуемся свойством «Маска». Мaska нужна для ограничения формата ввода информации. Заполните маску следующим образом:



Вместо символа «9» в режиме «1С:Предприятие» пользователь сможет ввести свои цифры. Теперь карточка студента позволяет ввести номер телефона по шаблону.

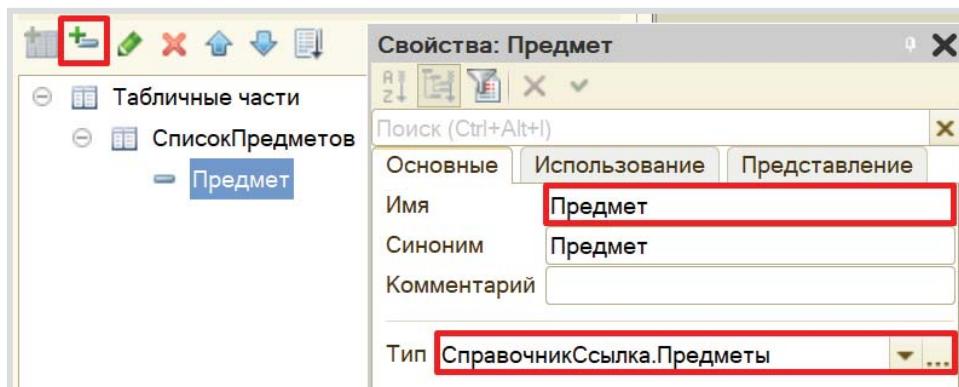
В карточке студента необходимо также отметить изучаемые студентом предметы.

Конечно, мы можем создать несколько отдельных реквизитов, но может получиться такая ситуация, что студент изучает большее количество предметов, чем добавлено реквизитов. Идея с добавлением реквизитов нам не подходит. Поэтому список предметов логично разместить в табличной части справочника. Добавим новую табличную часть «СписокПредметов».



На данный момент табличная часть совершенно пуста, в нее нельзя будет добавить какие-либо данные. Чтобы это исправить, нужно добавить реквизит табличной части, которая представляет собой колонку таблицы.

Добавим реквизит табличной части «Предмет», тип – «СправочникСсылка.Предметы». Таким образом, в реквизит «Предмет» может быть занесена только ссылка на элемент справочника «Предметы».



После обновления конфигурации карточка студента будет выглядеть следующим образом:

N	Предмет
1	Высшая математика
2	История
3	Философия

В табличную часть может быть внесено неограниченное количество строк для каждого элемента справочника. Таким образом, мы обеспечили хранение перечня изучаемых предметов для каждого студента.

Самостоятельно заполните карточки остальных студентов.

Теперь, когда у нас есть справочник, хранящий информацию о студентах и изучаемых ими предметах, мы можем создать отчет, формирующий список студентов по предметам.

Для создания отчета воспользуемся соответствующим объектом конфигурации.

Определение

Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platfroma/otchet/>).

Добавим отчет «СписокСтудентовПоПредметам». Для наполнения отчета воспользуемся конструктором схемы компоновки данных.

Отчет СписокСтудентовПоПредметам

- ▶ Основные
- Подсистемы
- Функциональные опции
- Данные
- Формы
- Команды
- Макеты
- Права
- Прочее

Имя: СписокСтудентовПоПредметам

Синоним: Список студентов по предметам

Комментарий:

Основная схема компоновки да...

... x

Открыть схему компоновки данных

Конструктор макета

Имя: ОсновнаяСхемаКомпоновкиДанных

Синоним: Основная схема компоновки данных

Комментарий:

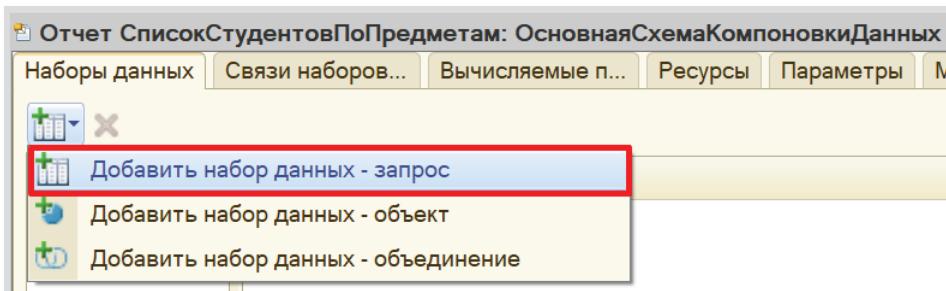
Выберите тип макета:

- Табличный документ
- Текстовый документ
- Двоичные данные
- Active document
- HTML документ
- Географическая схема
- Графическая схема
- Схема компоновки данных
- Макет оформления компоновки данных
- Внешняя компонента

Загрузить из файла:

Готово **Отмена** **Справка**

Все созданные нами объекты конфигурации представляют собой таблицы базы данных. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Для составления запроса воспользуемся *конструктором запроса*.

Поле	Путь	Ограничение п...	Роль	Выра...	Проверка иера...	Тип з...	Офор...
	Заголовок	П... У... Г... У...		Выраж упоряд	Набор данных Параметр	Доступ значен	Параме редакт
		Ограничение р...					
		П... У... Г... У...					

Запрос:

Конструктор запроса...

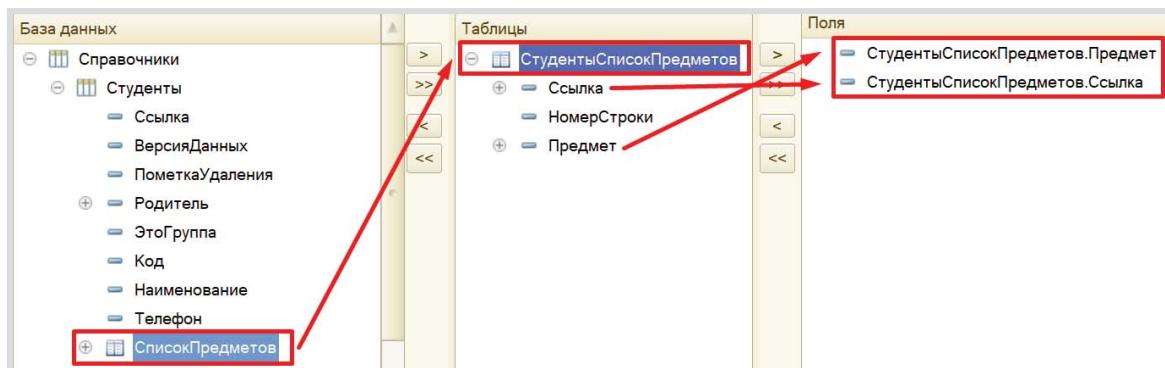
Открывается *конструктор запроса*. Эта вкладка имеет три части:

- ❑ Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Необходимо выбрать лишь те объекты, из которых мы хотим получать данные.
- ❑ Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- ❑ Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

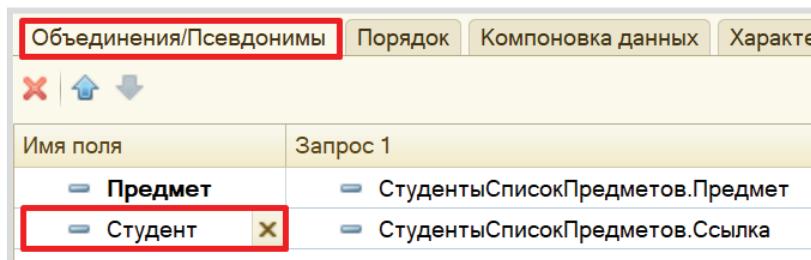
Данные будем брать из табличной части «СписокПредметов» справочника «Студенты».

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно выглядеть следующим образом:

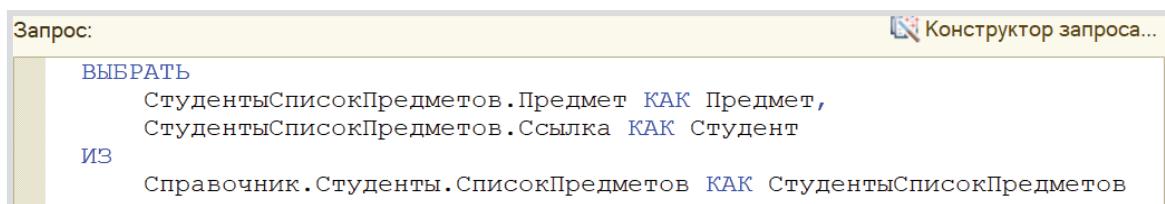


Перейдем на вкладку «Объединения/Псевдонимы» и изменим имя реквизита «Ссылка» на «Студент». Так пользователю будет проще понять, что отображено в отчете.

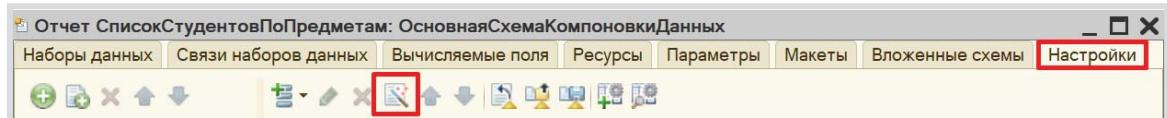


По завершении работы с *конструктором запроса* нажмите на кнопку «OK».

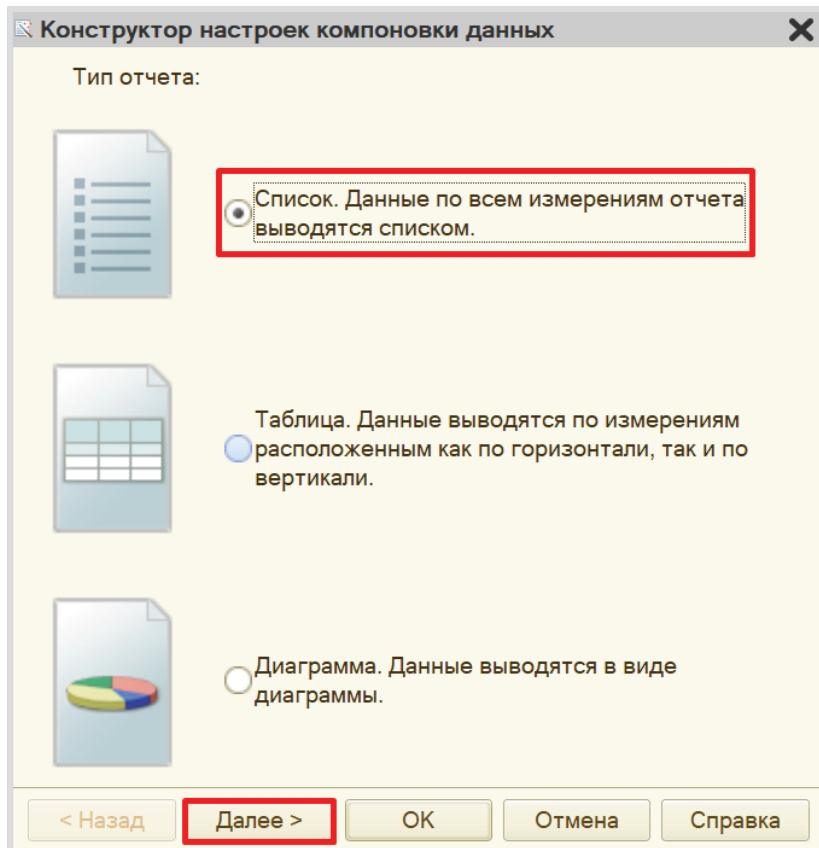
Получившийся запрос должен выглядеть следующим образом:



Для настроек отображения отчета перейдем на вкладку «Настройки» и воспользуемся конструктором настроек отчета.

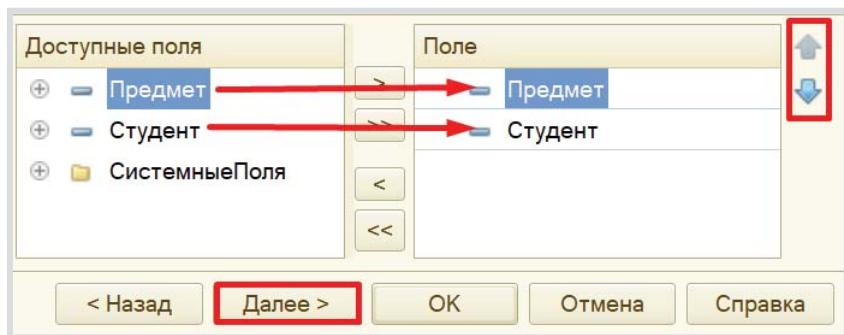


Наш отчет будет иметь вид списка.

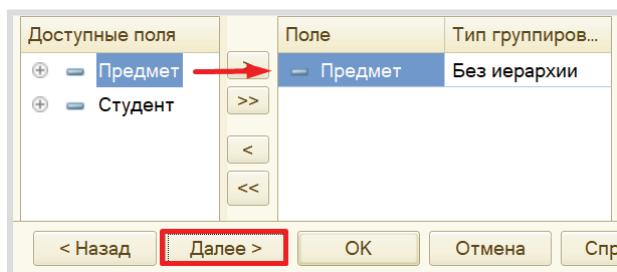


Выберем реквизиты «Студент» и «Предмет» для отображения в отчете.

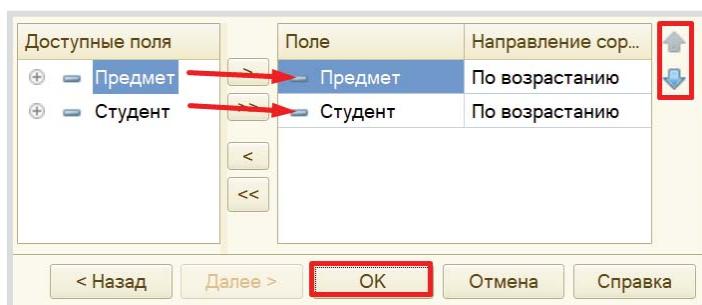
Обратите внимание, что в данном окне определяется порядок расположения реквизитов в отчете. Для изменения порядка воспользуйтесь стрелочками справа от выбранных полей.



Установим группировку элементов по предметам.



Установим сортировку. Пусть сначала будут расположены предметы по алфавиту, а внутри группировки – студенты будут расположены по алфавиту. *Обратите внимание* на порядок реквизитов, измените его с помощью стрелочек при необходимости.



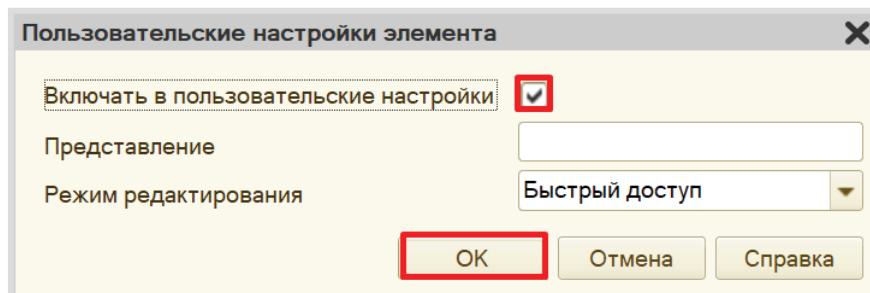
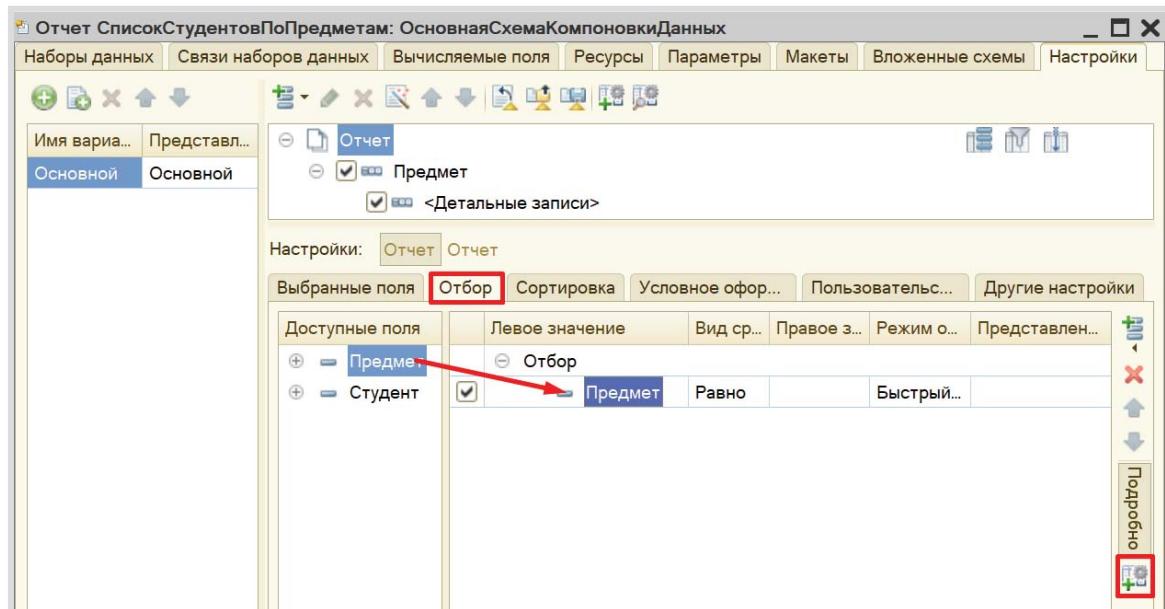
Проверим работу отчета в режиме «1С:Предприятие».

Предмет	Студент
Высшая математика	Иванов
	Филиппов
История	Иванов
	Мартынов
	Филиппов
Физкультура	Мартынов
	Петров
	Филиппов
Философия	Иванов
	Мартынов
	Петров

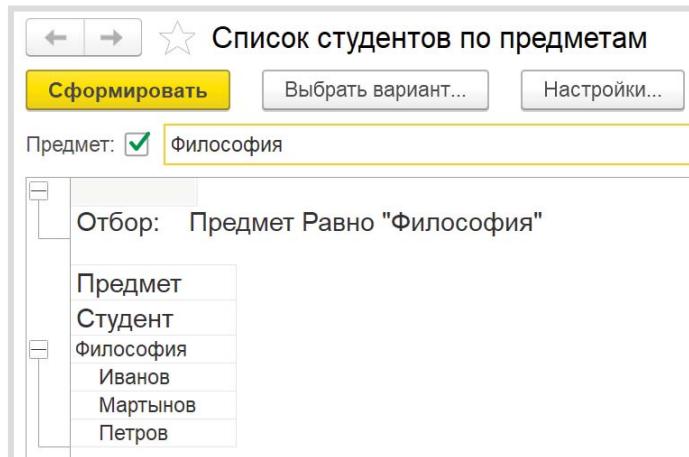
В получившемся отчете произведена группировка по предметам. В каждой группировке перечислены студенты, изучающие конкретный предмет.

Предположим, что нам необходимо получить список студентов, изучающих конкретный предмет. В текущем варианте отчета придется пролистывать весь список в поисках нужного предмета. Но можно добавить в отчет отбор по конкретному предмету, который пользователь может выбрать самостоятельно.

Для этого на вкладке «Настройки» найдем вкладку «Отбор» и установим реквизит «Предмет» для установки отбора. Откроем свойства элемента и установим галочку, чтобы данный элемент был доступен для пользователя.



Проверим работу отчета в режиме «1С:Предприятие». Установим отбор по предмету.



Кстати, вы можете перейти к карточке конкретного студента, просто щелкнув по нему в отчете двойным щелчком. Это возможно потому, что в качестве одного из полей для отчета (на этапе составления запроса к базе данных) мы выбрали поле «Ссылка».

Список студентов по предметам

Сформировать Выбрать вариант... Настройки...

Предмет: Философия

Отбор: Предмет Равно "Философия"

Предмет Студент Философия Иванов Мартынов Петров

Иванов (Студенты)

Записать и закрыть Записать Еще ▾

Код: 00000004

ФИО: Иванов

Группа: Группа 1

Телефон: +7(734)910-47-18

Добавить Поиск (Ctrl+F) Еще ▾

N	Предмет
1	Высшая математика
2	История
3	Философия

Поставленная задача решена.

Лабораторная работа № 4

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ХРАНЕНИЯ ИНФОРМАЦИИ О СОТРУДНИКАХ ПРЕДПРИЯТИЯ

*Сложность: **

Теги: справочник, регистр сведений, табличная
часть справочника, подчиненный справочник

ЗАДАНИЕ

Заказчик просит разработать информационную систему для хранения информации о сотрудниках предприятия.

В данной информационной системе необходимо хранить:

1. Список сотрудников.
2. Информацию о трудовой деятельности каждого сотрудника:
 - место работы;
 - дату начала работы;
 - дату увольнения;
 - должность.
3. Информацию о детях сотрудников:
 - ФИО ребенка;
 - год рождения.
4. Информацию о текущем окладе сотрудника.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать, смотрите в Лабораторной работе № 2 (стр. 17).

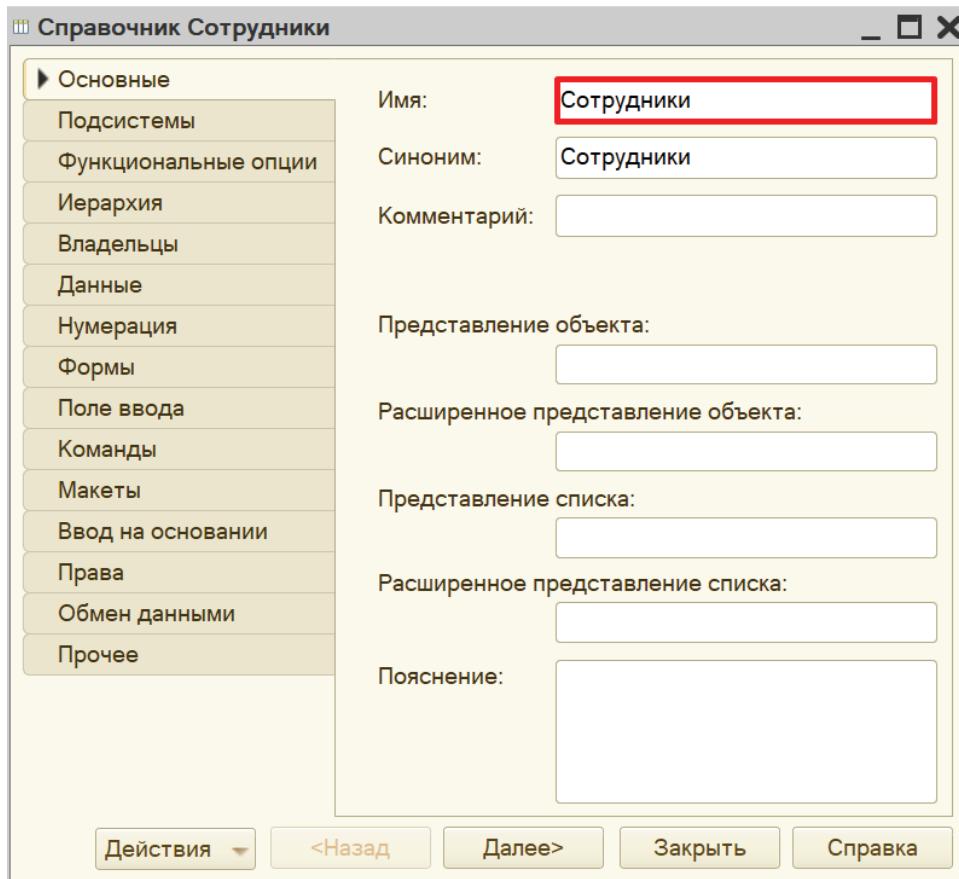
Выполнение

Чтобы хранить в информационной системе список сотрудников, необходимо добавить **справочник**.

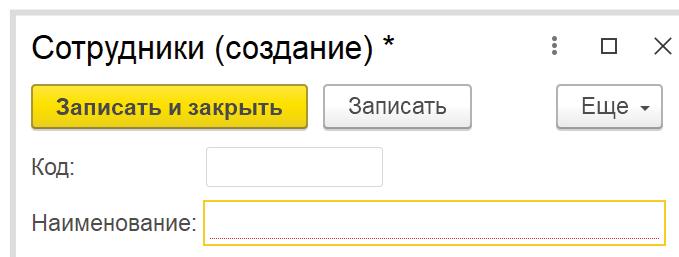
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список подразделений организации (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Создадим справочник «Сотрудники».



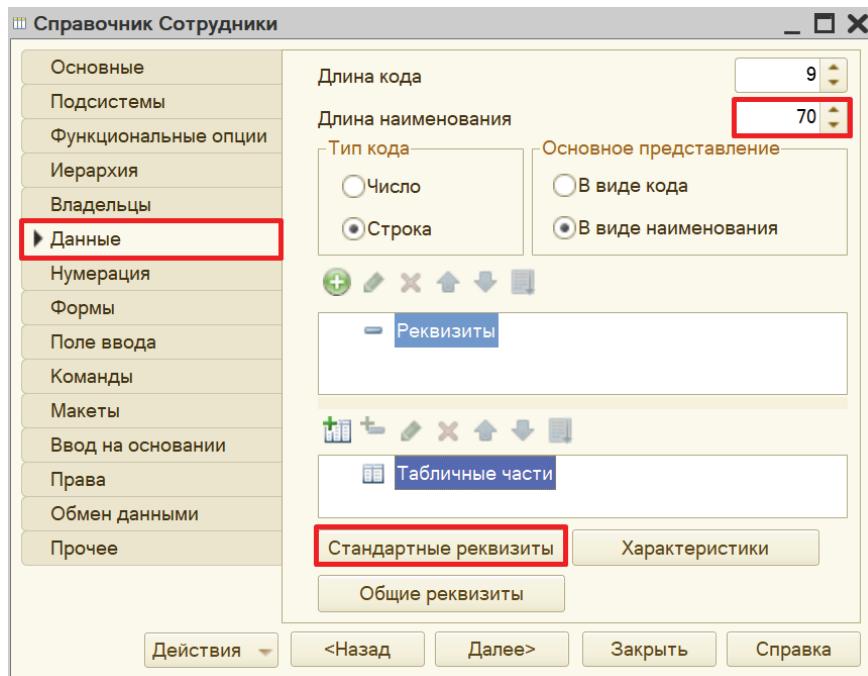
Если открыть данный справочник в режиме «1С:Предприятие», то он будет выглядеть следующим образом:



- Обратите внимание**, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.

Для добавления полей на карточку сотрудника перейдем на вкладку «Данные».

Длина поля «Наименование» ограничена, что помешает нам при попытке ввести длинное ФИО сотрудника. Необходимо увеличить длину наименования до 70 знаков.

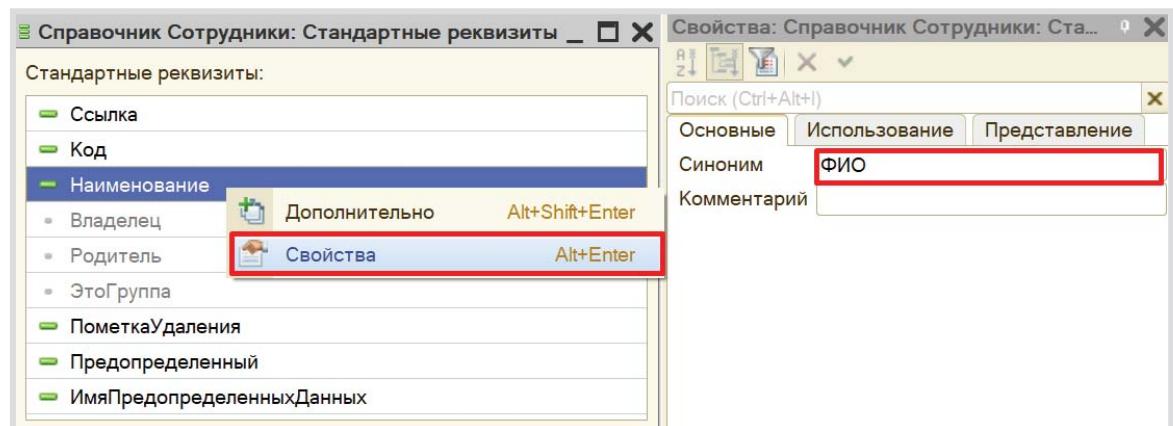


В этом же окне можно посмотреть стандартные реквизиты справочника. В открывшемся окне можно изменить синоним стандартного реквизита.

Определение

Синоним – это название реквизита, удобное для пользователя.

Добавим синоним для реквизита «Наименование» – «ФИО».

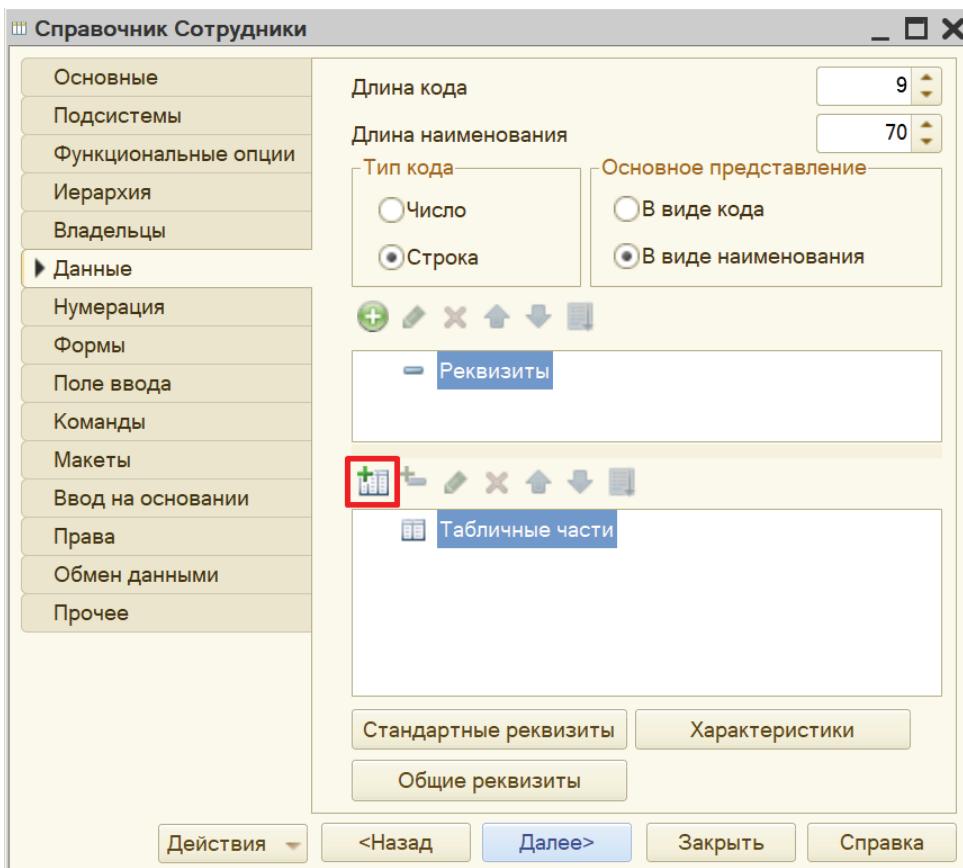


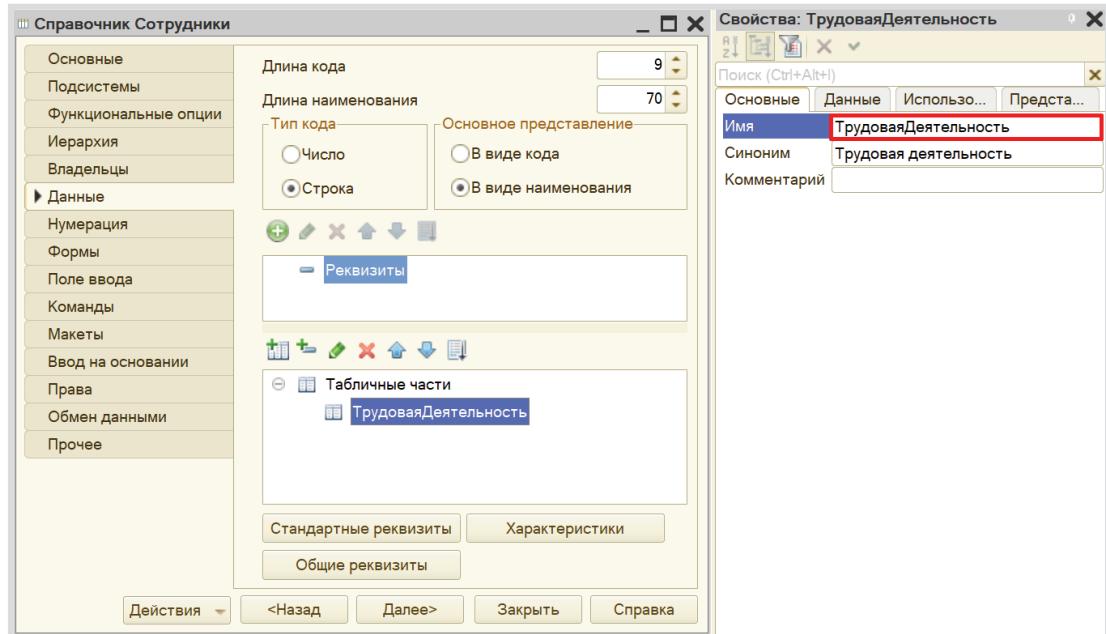
Теперь справочник в режиме «1С:Предприятие» будет выглядеть следующим образом:

Таким образом, была реализована возможность хранения списка сотрудников.

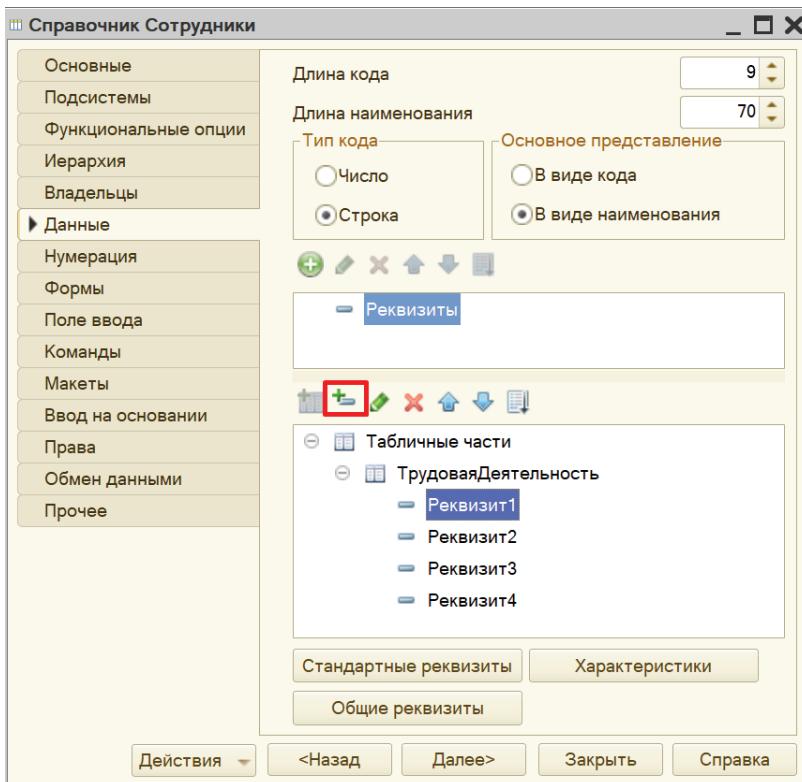
Но по условию задачи нам необходимо хранить еще множество информации о сотрудниках, и стандартных реквизитов нам недостаточно. Нужно создать свои собственные реквизиты.

Как в данной системе организовать хранение информации о трудовой деятельности сотрудника? Для этого добавим в справочник табличную часть «ТрудоваяДеятельность» с помощью кнопки «Добавить табличную часть».





Теперь создадим реквизиты табличной части – колонки таблицы – с помощью кнопки «Добавить реквизит». Добавляем четыре реквизита.



Меняем свойства созданных реквизитов. *Обратите внимание* на тип создаваемых реквизитов.

Свойства: МестоРаботы

Имя	МестоРаботы
Синоним	Место работы
Комментарий	
Тип	Строка
Длина	50
Допустимая длина	Переменная
Неограниченная длина	<input type="checkbox"/>

Свойства: Должность

Имя	Должность
Синоним	Должность
Комментарий	
Тип	Строка
Длина	30
Допустимая длина	Переменная
Неограниченная длина	<input type="checkbox"/>

Свойства: ДатаНачалаРаботы

Имя	ДатаНачалаРаботы
Синоним	Дата начала работы
Комментарий	
Тип	Дата
Состав даты	Дата

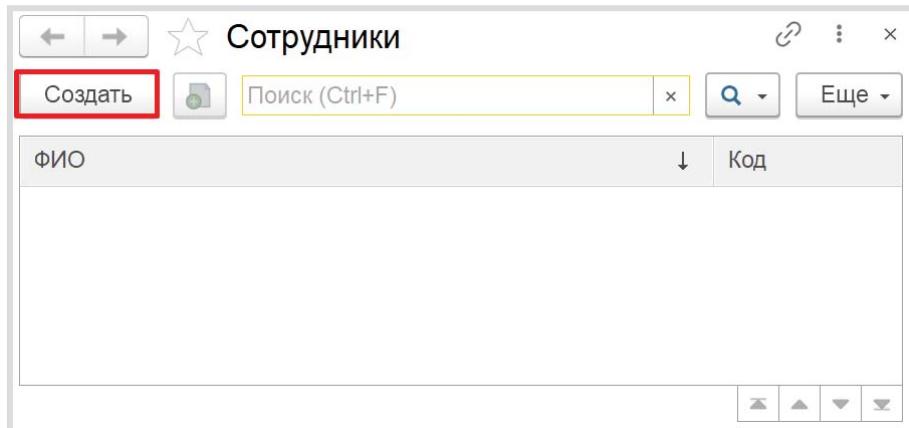
Свойства: ДатаУвольнения

Имя	ДатаУвольнения
Синоним	Дата увольнения
Комментарий	
Тип	Дата
Состав даты	Дата

Табличная часть справочника должна выглядеть следующим образом:

- (-) Табличные части
- (-) Трудовая Деятельность
 - МестоРаботы
 - ДатаНачалаРаботы
 - ДатаУвольнения
 - Должность

Откроем справочник «Сотрудники» в режиме «1С:Предприятие» и заполним данные о двух сотрудниках.



Сотрудники (создание) *

Записать и закрыть	Записать	Еще		
Код:	<input type="text"/>			
ФИО:	Иванюхин Иван Иванович			
Добавить	↑ ↓	Поиск (Ctrl+F)		
N	Место работы	Дата начала работы	Дата увольнения	Должность
1	ООО "Ромашка"	10.09.2003	17.12.2010	Менеджер
2	ООО "Гвоздика"	15.01.2010	03.07.2018	Старший менеджер

Сотрудники (создание) *

Записать и закрыть	Записать	Еще		
Код:	<input type="text"/>			
ФИО:	Петренко Петр Петрович			
Добавить	↑ ↓	Поиск (Ctrl+F)		
N	Место работы	Дата начала работы	Дата увольнения	Должность
1	ООО "Роза"	12.10.2007	09.03.2014	Директор
2	ООО "Незабудка"	15.05.2014	24.11.2019	Директор

Теперь справочник «Сотрудники» может хранить данные о трудовой деятельности сотрудников.

Но нам нужно также сохранять данные об их детях, чтобы отдел кадров мог подготовить к Новому году сладкие подарки для детей младше 18 лет.

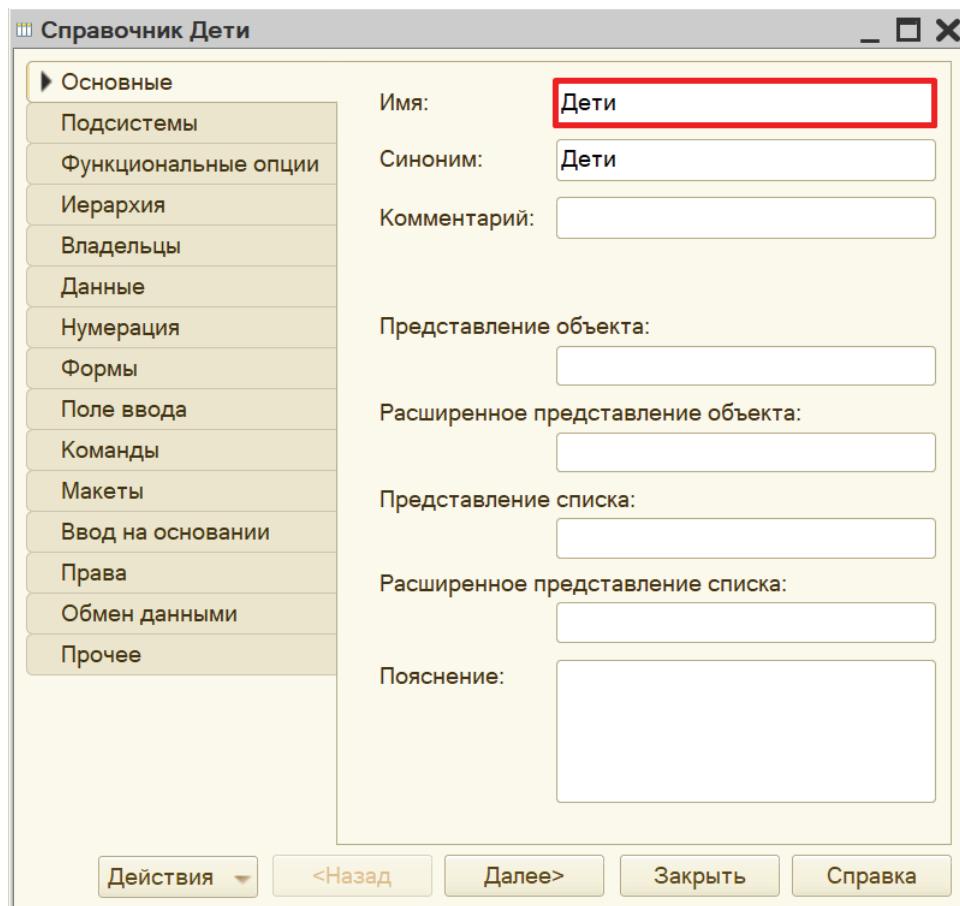
Можно по аналогии создать табличную часть и заполнить ее информацией о детях сотрудников, но мы рассмотрим еще один вариант хранения информации.

Создадим *подчиненный справочник*.

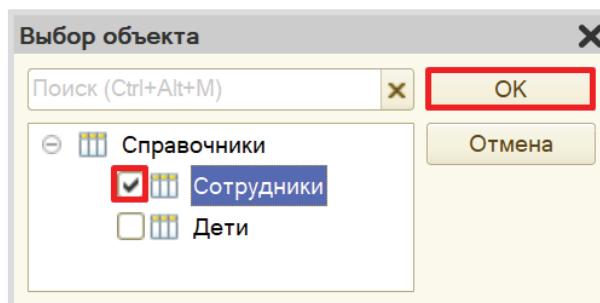
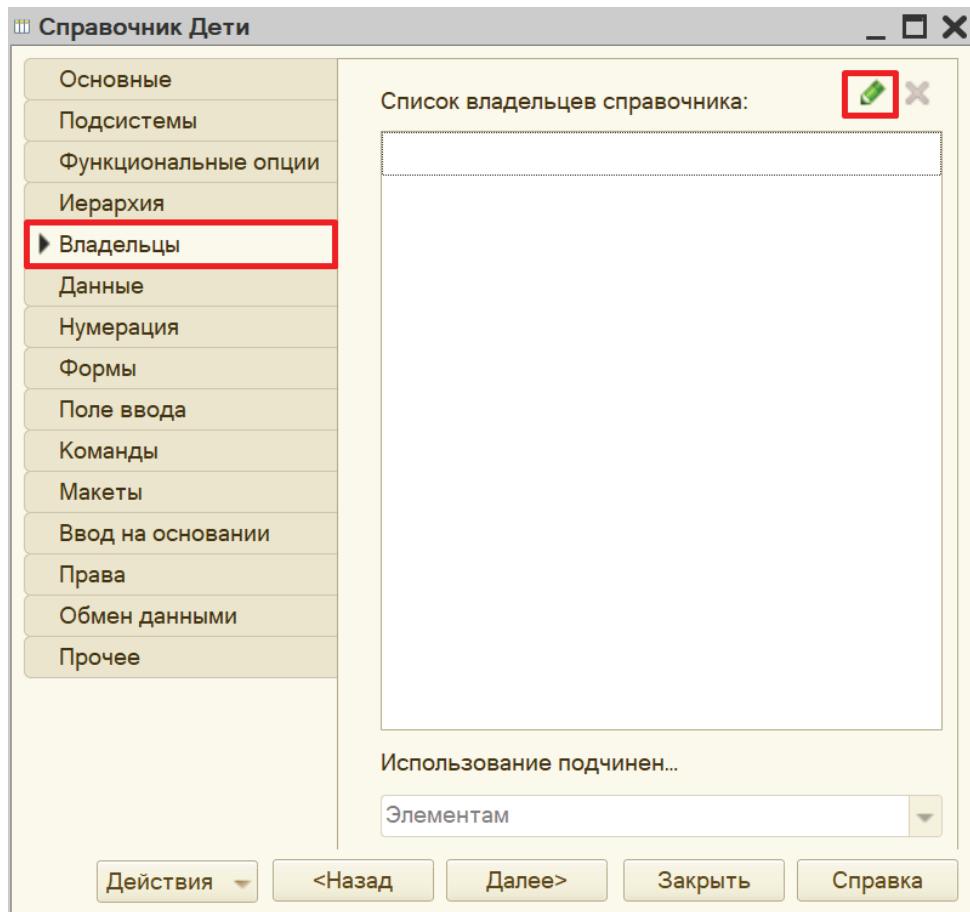
Определение

Подчиненный справочник – это справочник, значения которого зависят от значения другого – родительского – справочника.

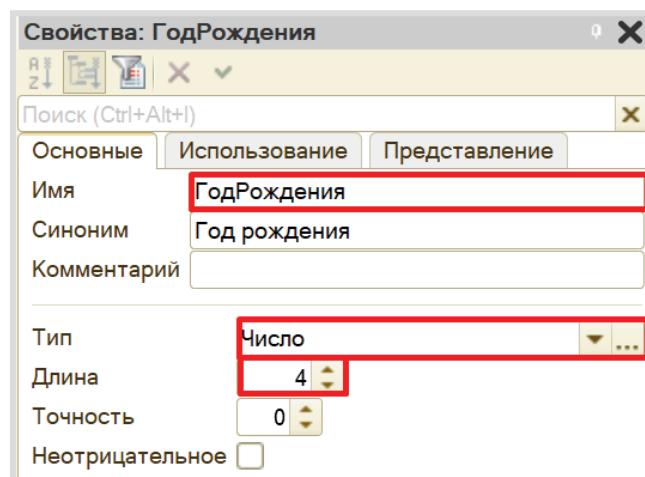
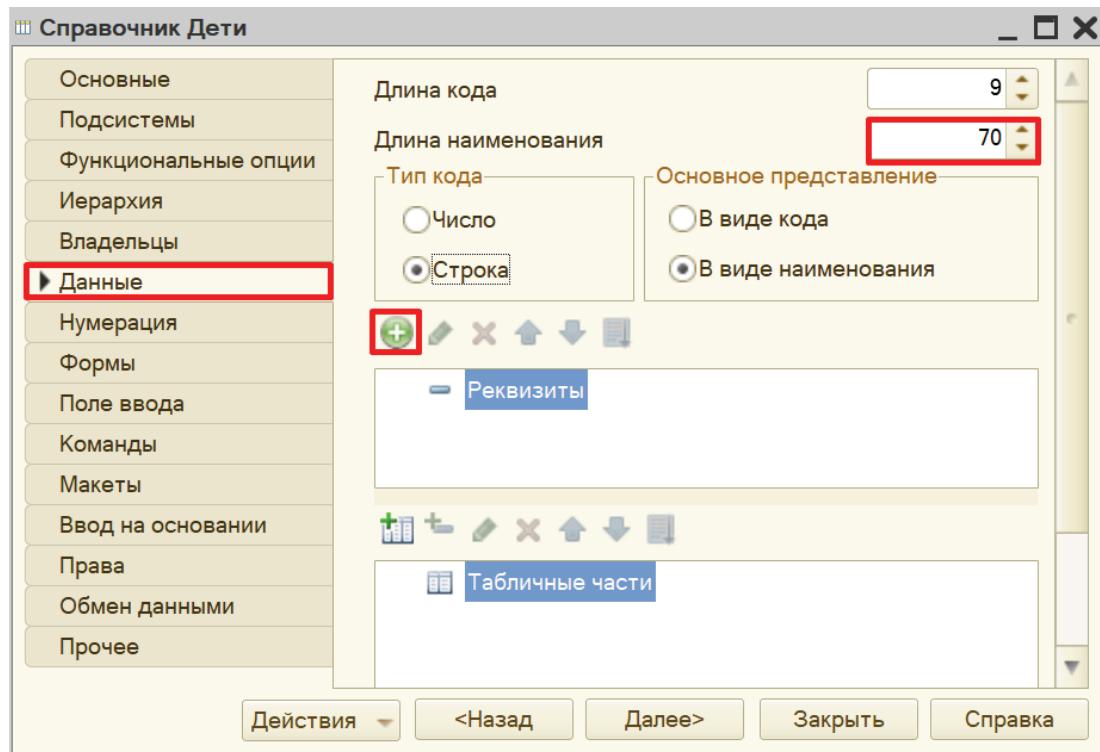
Создадим новый справочник «Дети».



Справочник «Дети» будет подчинен справочнику «Сотрудники». Для настройки подчинения необходимо перейти на вкладку «Владельцы» и из списка справочников выбрать справочник «Сотрудники».



Для настройки структуры справочника переходим на вкладку «Данные» и добавляем реквизит «ГодРождения» с типом «Число».



Самостоятельно измените синоним стандартного реквизита «Наименование» на «ФИО» справочника «Дети».

Теперь можно запустить программу в режиме «1С:Предприятие» и ввести данные о детях сотрудников. Причем мы можем это сделать прямо из карточки сотрудника.

☆ Иванюхин Иван Иванович (Сотрудники)

Основное Дети

Записать и закрыть **Записать** **Еще**

Код: 00000001

ФИО: Иванюхин Иван Иванович

Добавить **Поиск (Ctrl+F)** **Еще**

N	Место работы	Дата начала работы	Дата увольнения	Должность
1	ООО "Ромашка"	10.09.2003	17.12.2010	Менеджер
2	ООО "Гвоздика"	15.01.2010	03.07.2018	Старший менеджер

☆ Иванюхин Иван Иванович (Сотрудники)

Основное Дети

Дети

Создать **Поиск (Ctrl+F)** **Еще**

Наименование	Код	Владелец	Год рождения

Дети (создание) *

Записать и закрыть **Записать** **Еще**

Код:

ФИО: Иванюхина Анна Ивановна

Владелец: Иванюхин Иван Иванович

Год рождения: 2 005

Аналогично можно добавлять информацию о детях напрямую в справочник «Дети».

В таком случае поле «Владелец» необходимо заполнить вручную.

Сотрудники

ФИО	Код
Иванюхин Иван Иванович	000000001
Петренко Петр Петрович	000000002

Дети (создание) *

Код:	<input type="text"/>
ФИО:	Петренко Анастасия Петровна
Владелец:	Петренко Петр Петрович
Год рождения:	2 007

Убедимся в том, что ребенок был добавлен в карточку сотрудника.

☆ Петренко Петр Петрович (Сотрудники)

Основное	Дети
Дети	
Создать	<input type="button"/>
Поиск (Ctrl+F)	
ФИО	Код
Петренко Анастасия Петровна	000000002
Петренко Петр Петрович	2 007

Внимание!

Создание новой карточки ребенка невозможно без заполнения поля «Владелец». Именно в этом и заключается суть подчиненных справочников: мы исключаем возможность добавления в базу объекта без владельца (ребенка без родителя, который является сотрудником нашей организации).

Кроме того, как вы могли убедиться, можно посмотреть карточки детей либо в справочнике «Дети», либо в карточке сотрудника.

Чем же такой вариант хранения информации отличается от хранения данных в табличной части справочника?

Дети сотрудников, как и сами сотрудники – это *объекты аналитики*. Это значит, что данные объекты и их реквизиты могут в дальнейшем быть использованы, например, для построения отчетов. Каждый объект аналитики (отдельный сотрудник или ребенок) в системе имеет уникальную ссылку. А вот данные о трудовой деятельности сотрудника, занесенные в табличную часть, несут только информативный характер, и в дальнейшем, при работе программы, использоваться никак не будут, поэтому у них нет уникальной ссылки.

Рассмотрим еще один вариант хранения данных в системе.

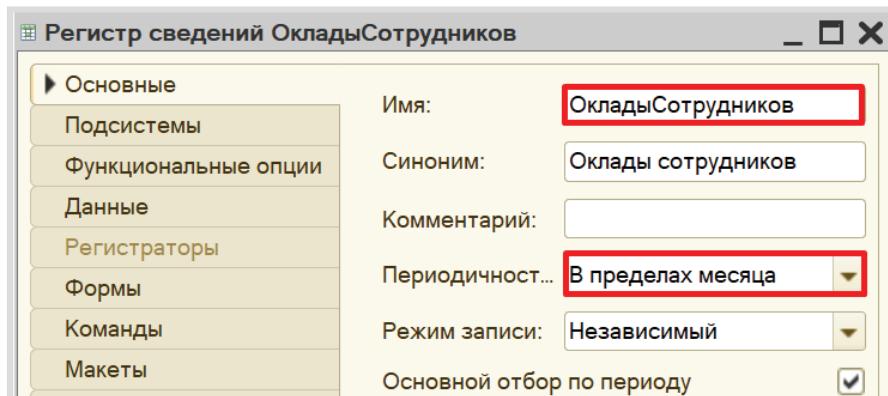
Перед нами стоит задача учитывать оклад сотрудника. Но оклад может меняться в зависимости от условий выполняемой работы. Следовательно, хранить данные об окладе сотрудника в справочнике нам невыгодно.

Для хранения истории изменения данных в системе «1С:Предприятие» используют *регистры сведений*.

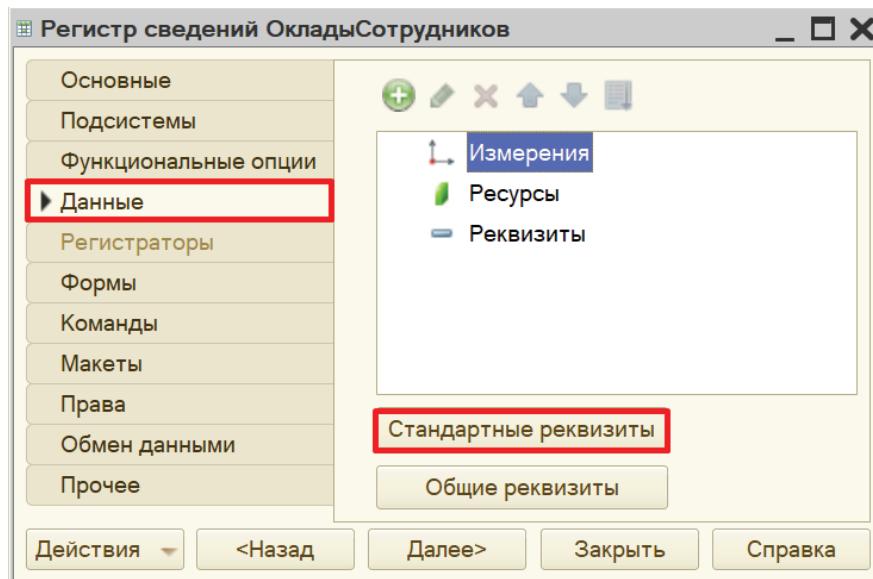
Определение

Регистр сведений позволяет хранить информацию об изменении каких-либо показателей с течением времени, например, хранить данные о курсах валют (подробнее о регистрах сведений можно прочитать здесь: <https://v8.1c.ru/platforma/registr-svedeniy/>).

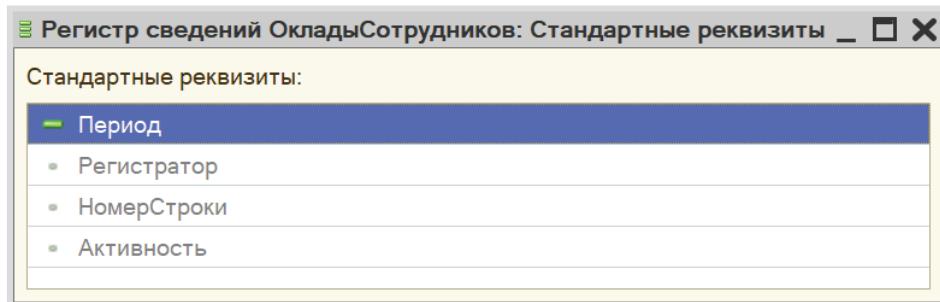
Добавим *регистр сведений*. Назовем его «ОкладыСотрудников». Установим периодичность «В пределах месяца». Таким образом, вносить данные об окладе сотрудников в *регистр сведений* можно будет только один раз в месяц.



Структура регистра, как и прочих объектов конфигурации, настраивается на вкладке «Данные».



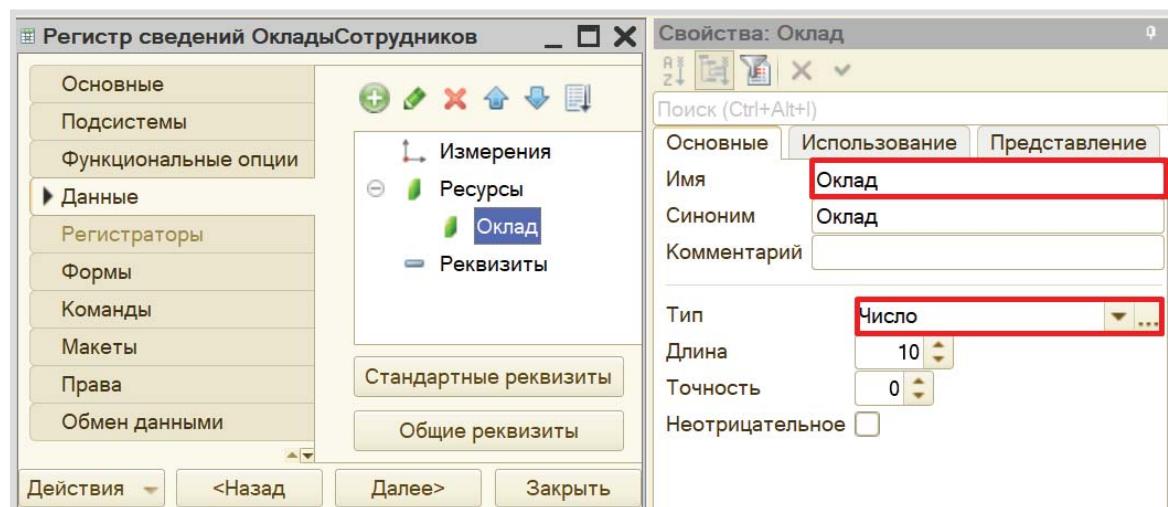
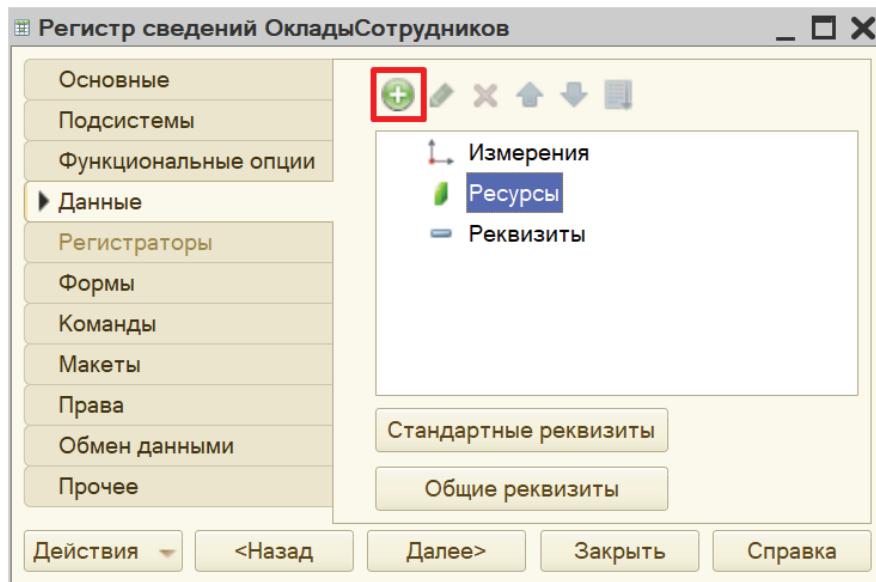
Здесь же можно заглянуть в список стандартных реквизитов.



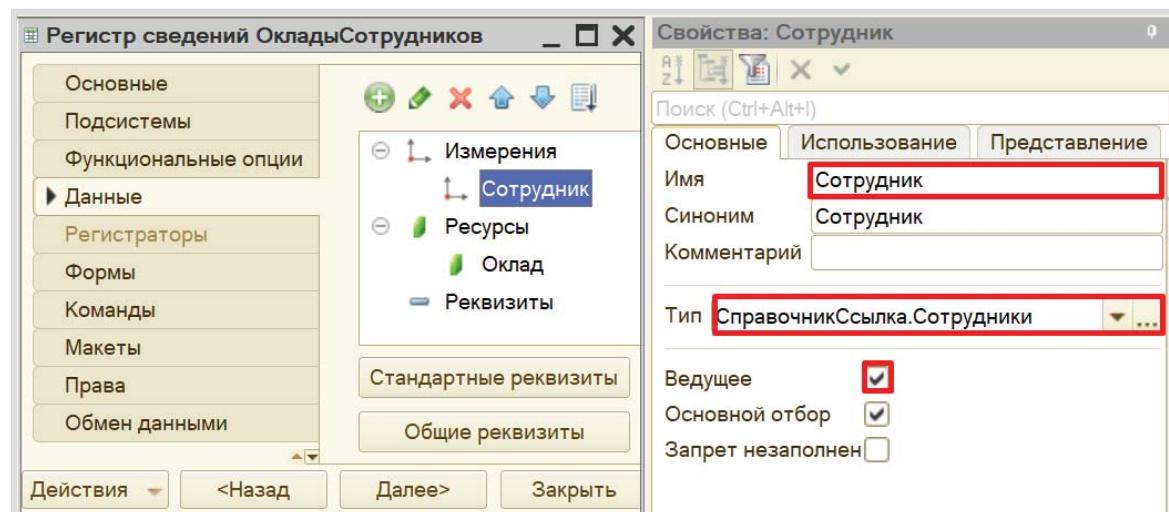
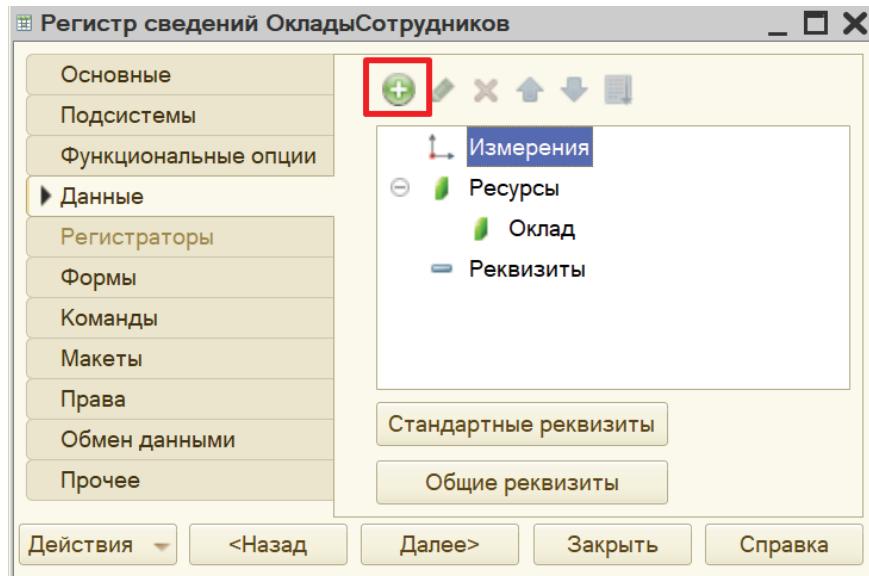
Изменив периодичность регистра на «В пределах месяца», мы активировали стандартный реквизит «Период». В него будет заноситься дата внесения данных в регистр. Это сделано для того, чтобы система могла отслеживать попытки внесения данных чаще, чем один раз в месяц.

Возвращаемся к окну редактирования *регистра сведений*.

Заполнение данного окна всегда проще всего начинать с добавления *ресурса*. Чтобы понять, что использовать в качестве ресурса, необходимо задать вопрос: «Что мы хотим хранить в данном регистре?». Мы хотим хранить данные об окладах. Следовательно, оклад и будет являться ресурсом. Тип данного реквизита – «Число».



Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим хранить оклад? Мы хотим хранить оклад (чей?) сотрудников. Значит, в качестве измерения необходимо добавить реквизит «Сотрудник». Тип данного реквизита – «СправочникСсылка.Сотрудники». В этом поле будут храниться ссылки на элементы справочника «Сотрудники».



Если у измерения свойство «Ведущее» включено, это значит, что запись *регистра сведений* имеет смысл только пока существует объект, на который он ссылается. То есть если удалить сотрудника из справочника «Сотрудники», то все записи, связанные с этим сотрудником, из *регистра сведений* будут удалены. Плюс данная галочка создаст быстрый переход к регистру прямо из карточки сотрудника.

Откроем *регистр сведений* в режиме «1С:Предприятие» и введем данные об окладе сотрудника за несколько месяцев.

Добавим оклад сотруднику прямиком из карточки сотрудника.

Иванюхин Иван Иванович (Сотрудники)

Основное Дети **Оклады сотрудников**

Записать и закрыть **Записать** Еще ▾

Код: 000000001

ФИО: Иванюхин Иван Иванович

Добавить **↑** **↓** Поиск (Ctrl+F) Еще ▾

N	Место работы	Дата начала работы	Дата увольнения	Должность
1	ООО "Ромашка"	10.09.2003	17.12.2010	Менеджер
2	ООО "Гвоздика"	15.01.2010	03.07.2018	Старший менеджер

Иванюхин Иван Иванович (Сотрудники)

Основное Дети **Оклады сотрудников**

Оклады сотрудников

Создать Поиск (Ctrl+F) Еще ▾

Период	Сотрудник	Оклад

Оклады сотрудников (со...)

Записать и закрыть **Записать** Еще ▾

Период: 01.08.2020

Сотрудник: Иванюхин Иван Иванович

Оклад: 50 000

Период	Сотрудник	Оклад
01.08.2020	Иванюхин Иван Иванович	50 000
01.09.2020	Иванюхин Иван Иванович	55 000

Проверим, что эти данные действительно попали в *регистр сведений*.

Период	Сотрудник	Оклад
01.08.2020	Иванюхин Иван Иванович	50 000
01.09.2020	Иванюхин Иван Иванович	55 000

Таким образом, была реализована возможность хранения информации об изменении оклада сотрудника.

Поставленная задача решена.

Лабораторная работа № 5

**РАЗРАБОТКА КОНФИГУРАЦИИ
ДЛЯ УЧЕТА ПОСЕЩЕНИЙ
КЛИЕНТАМИ ЭКСКУРСИЙ**

Сложность: *

Теги: справочник, документ, ввод на основании,
схема компоновки данных

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета посещений клиентами экскурсий.

1. В системе необходимо регистрировать посещения экскурсий на основании оформленной брони.

Пользователь системы по телефону с клиентом оформляет бронь выбранной экскурсии. Затем при посещении клиент оплачивает забронированную экскурсию наличными деньгами или банковской картой.

2. Нужно построить отчет о доходах с экскурсий.

Форма отчета:

Способ оплаты	Сумма
Экскурсия	
Наличные	19 500
Купеческая усадьба	10 000
Реконструкция битвы	7 500
Царские палаты	2 000
Банковская карта	2 000
Царские палаты	2 000
Итого	21 500

Отчет группирует информацию по способу оплаты экскурсии, а также подводит общий итог.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать, смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

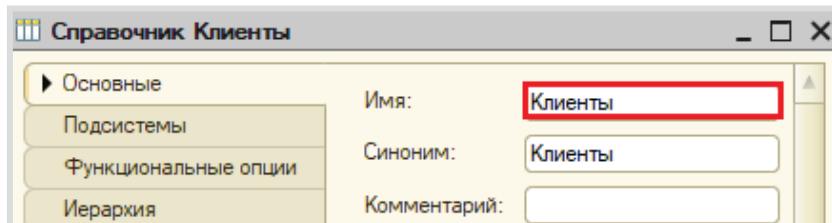
«Заказчик просит разработать конфигурацию для учета посещений клиентами экскурсий.»

Из условия следует, что необходимо хранить информацию о клиентах и проводимых экскурсиях. Для решения этой задачи нам понадобятся *справочники*.

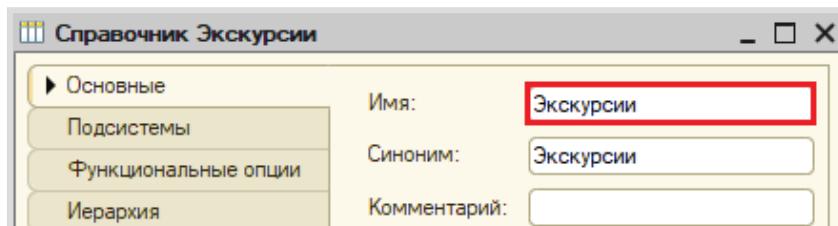
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

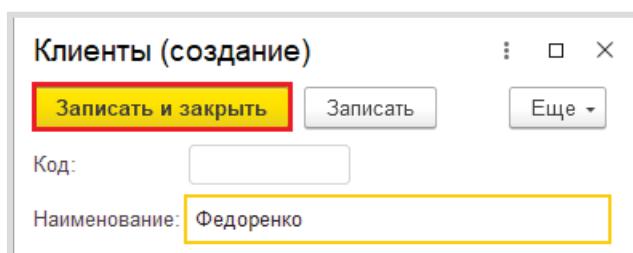
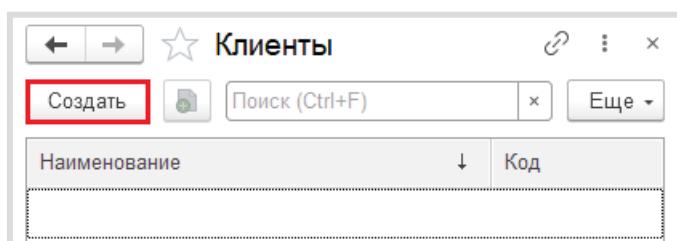
Создадим справочник «Клиенты».



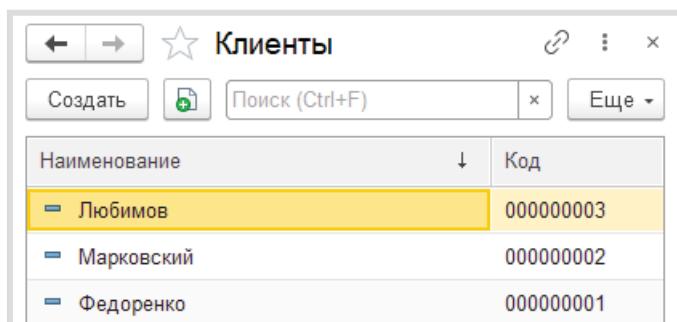
Создадим справочник «Экскурсии».



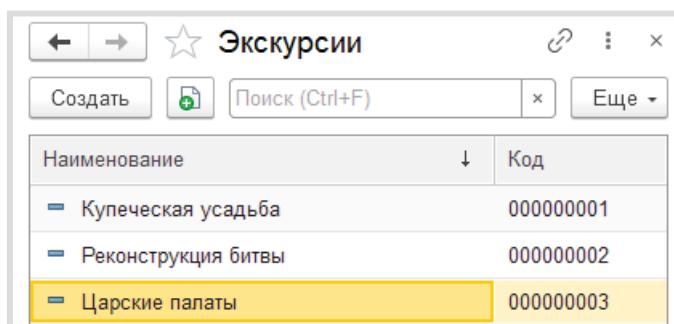
Откроем программу в режиме «1С:Предприятие» и добавим в каждый *справочник* несколько элементов.



Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



Аналогично – со справочником «Экскурсии».



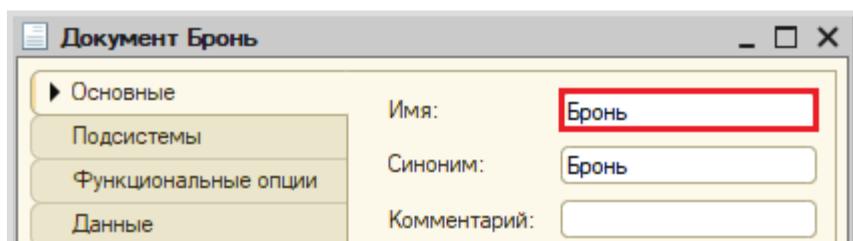
«В системе необходимо регистрировать посещения экскурсий на основании оформленной брони.»

Для регистрации посещения экскурсий и брони необходимо воспользоваться объектом конфигурации *документ*.

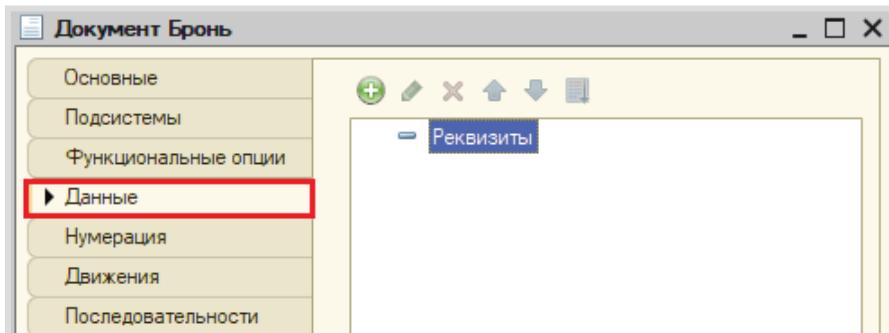
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим новый документ «Бронь».



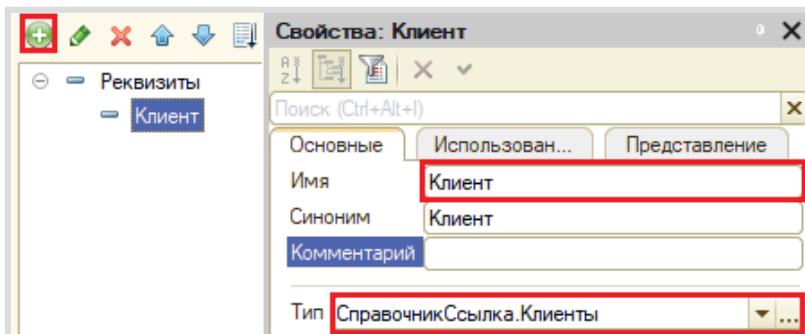
Для настройки структуры документа переходим на вкладку «Данные».



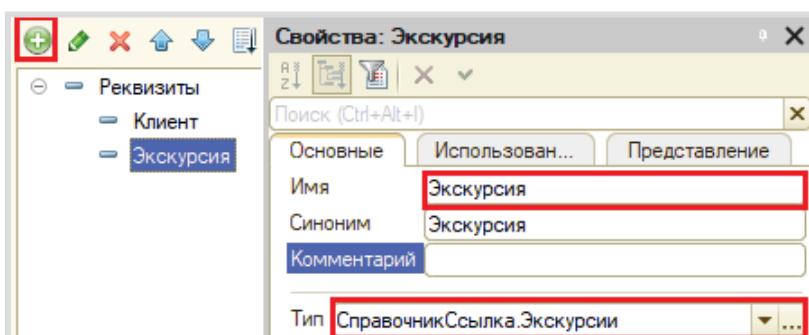
«Пользователь системы по телефону с клиентом оформляет бронь выбранной экскурсии.»

Из условия следует, что при оформлении брони необходимо указывать клиента и бронируемую экскурсию.

Добавим реквизит «Клиент», тип – «СправочникСсылка.Клиенты». Данный реквизит будет хранить ссылку на элемент справочника «Клиенты».



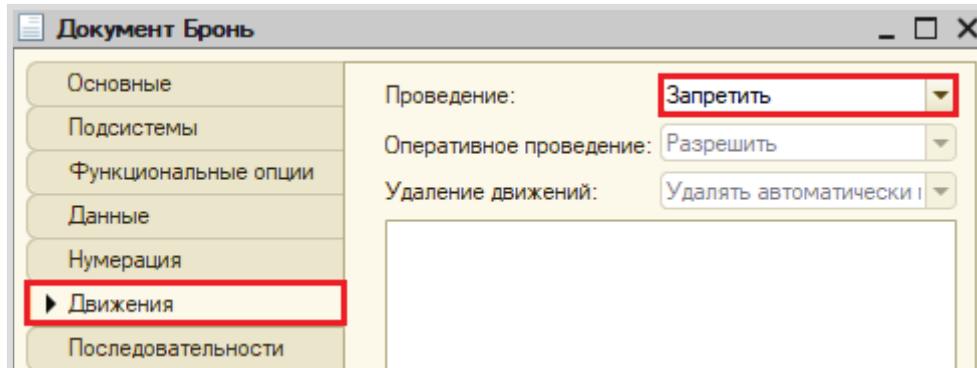
Далее добавим реквизит «Экскурсия», тип – «СправочникСсылка.Экскурсии».



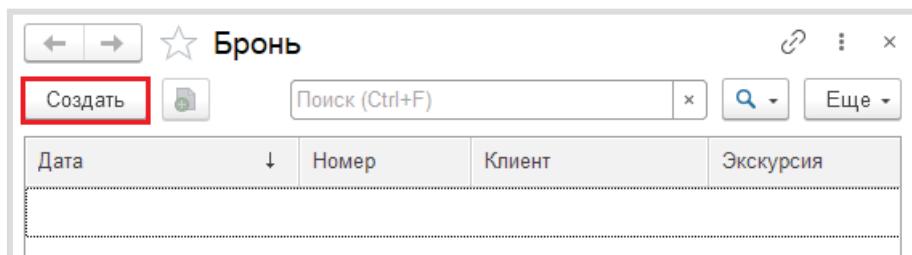
Любой документ может находиться в одном из двух состояний: *подготовленный к свершению* или *совершенный*:

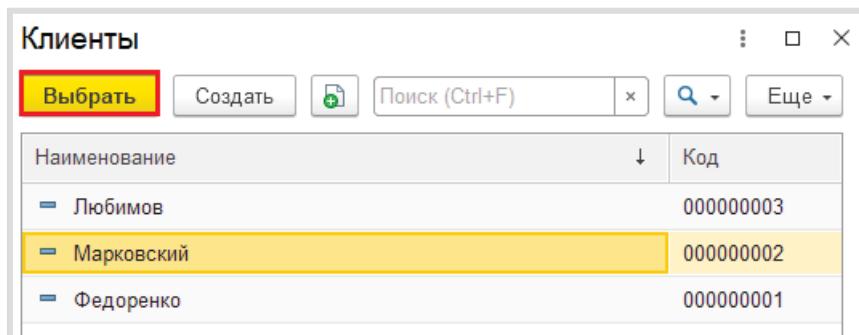
- чтобы подготовить документ для использования в будущем, необходимо его записать;
- чтобы отметить документ как совершенный – провести.

Но для документа «Бронь» нам не нужно два этапа, поскольку данный документ будет заполняться пользователем сразу в момент диалога с клиентом и в должен считаться уже совершенным действием. Отключим проведение у документа на вкладке «Движения».



Запустим режим «1С:Предприятие» и попробуем создать несколько документов.



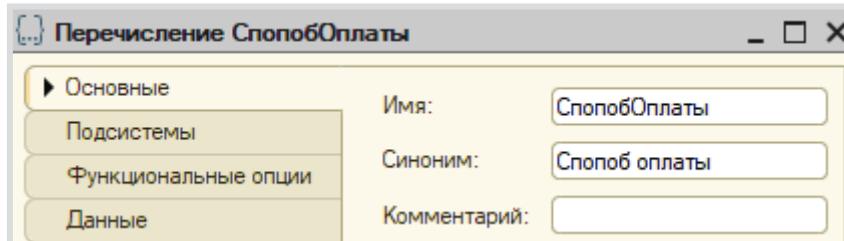


Аналогичным образом укажите экскурсию и нажмите на кнопку «Записать и закрыть».

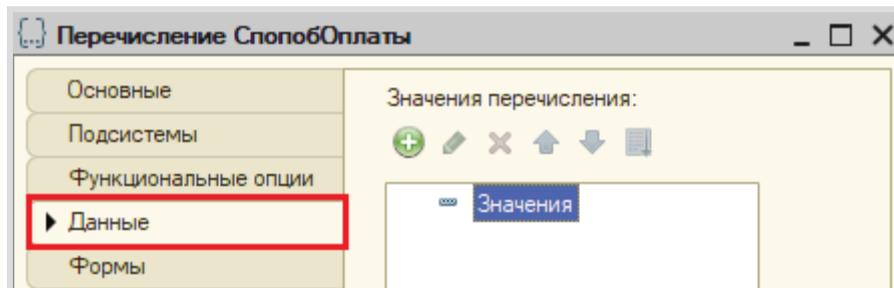
«...клиент оплачивает забронированную экскурсию наличными деньгами или банковской картой.»

Из условия следует, что способы оплаты определены заранее и представляют собой выбор из двух вариантов. Для решения задачи хранения информации, которая представляет собой фиксированный набор альтернатив, нам понадобится новый объект, который называется **перечисление** (подробнее про перечисления можно прочитать здесь: <https://v8.1c.ru/platforma/perechisleniya/>).

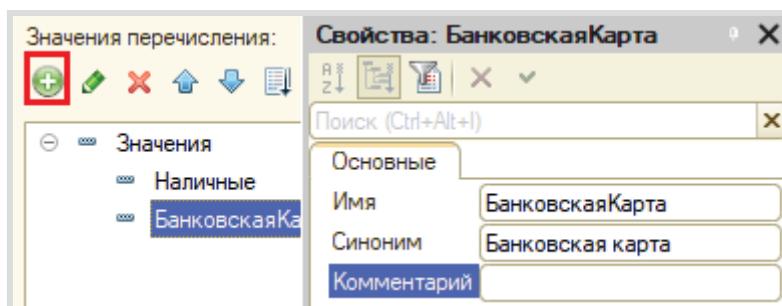
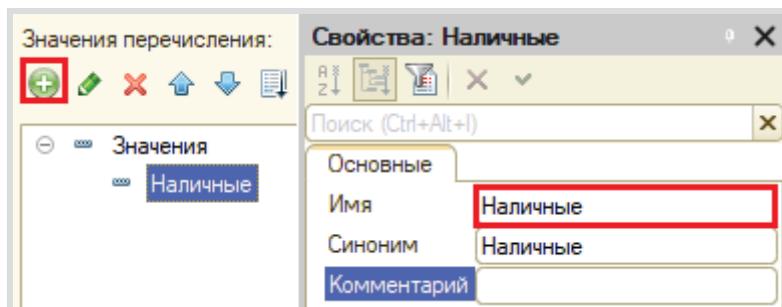
Добавим новое перечисление «СпособОплаты».



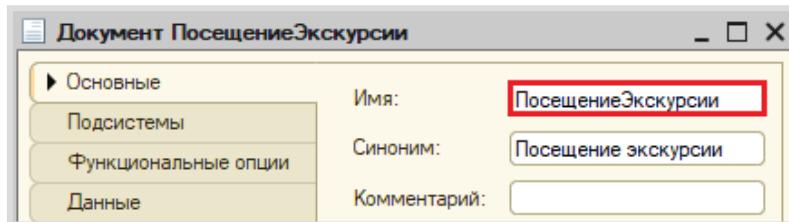
Значения перечисления (заготовленный список выбора) заполним на вкладке «Данные».



Добавим два способа оплаты: *наличными* и *банковской картой*.



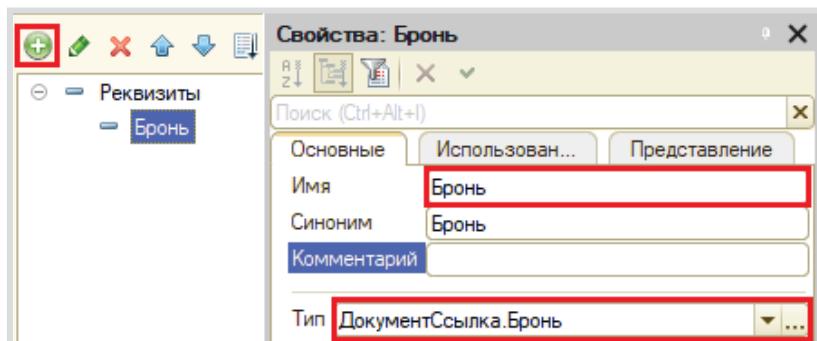
Для реализации посещения экскурсии нам также потребуется *документ*. Добавим новый документ «ПосещениеЭкскурсии».



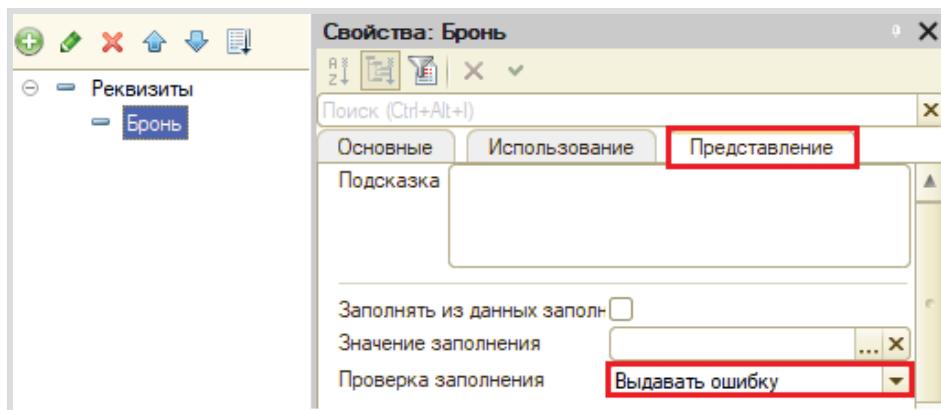
«Затем при посещении клиент оплачивает забронированную экскурсию наличными деньгами или банковской картой.»

Из условия следует, что в новом документе потребуется хранить информацию о номере брони, клиенте, выбранной экскурсии, а также сумме экскурсии и способе ее оплаты.

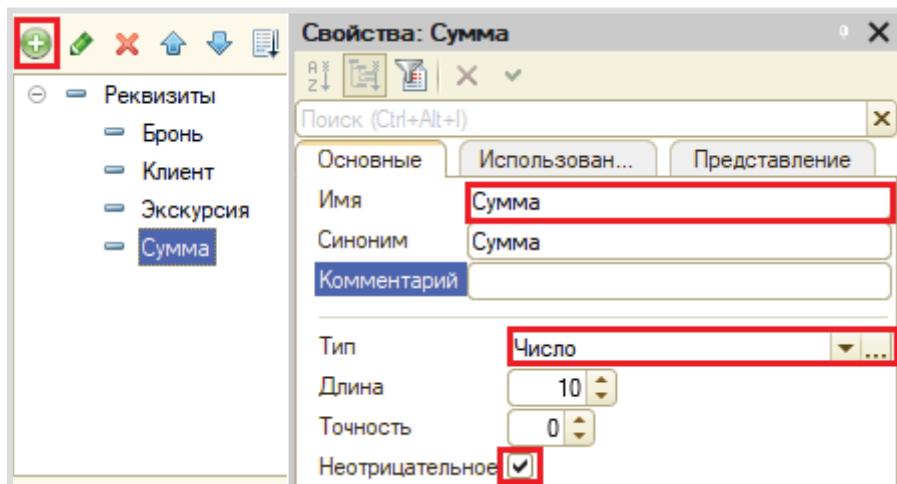
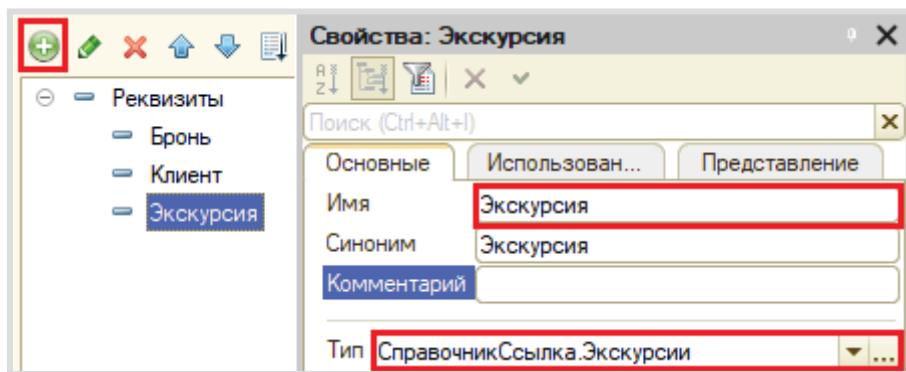
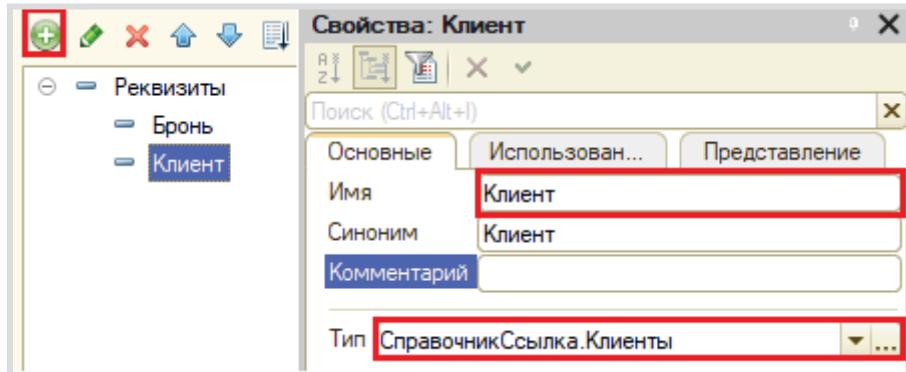
Аналогично процессу создания реквизитов в документе «Бронь» добавим реквизиты в документ «ПосещениеЭкскурсии» экскурсии на вкладке «Данные».

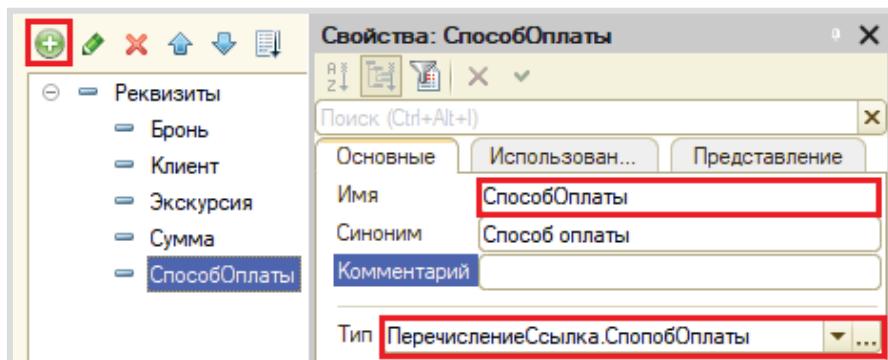


Чтобы у пользователя не было возможности создавать посещения экскурсий, не указав бронь, настроим проверку заполнения.



Добавим реквизиты «Клиент», «Экскурсия», «Сумма» и «СпособОплаты».



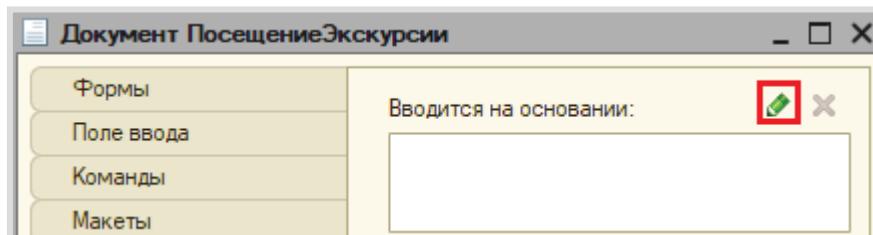


«... посещения экскурсий на основании оформленной брони.»

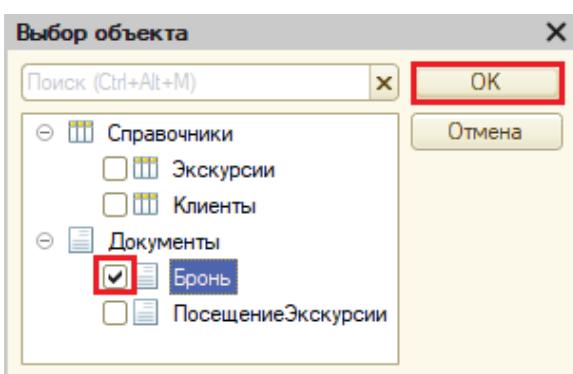
Условие дает нам понять, что документ «ПосещениеЭкскурсии» должен формироваться на основании документа другого вида. Таким образом, часть информации будет «перетекать» из одного документа в другой.

Для реализации такого функционала воспользуемся *конструктором ввода на основании*. (Более подробно про *конструктор ввода на основании* можно прочитать здесь: <https://v8.1c.ru/platforma/konstruktor-vvoda-na-osnovanii/>).

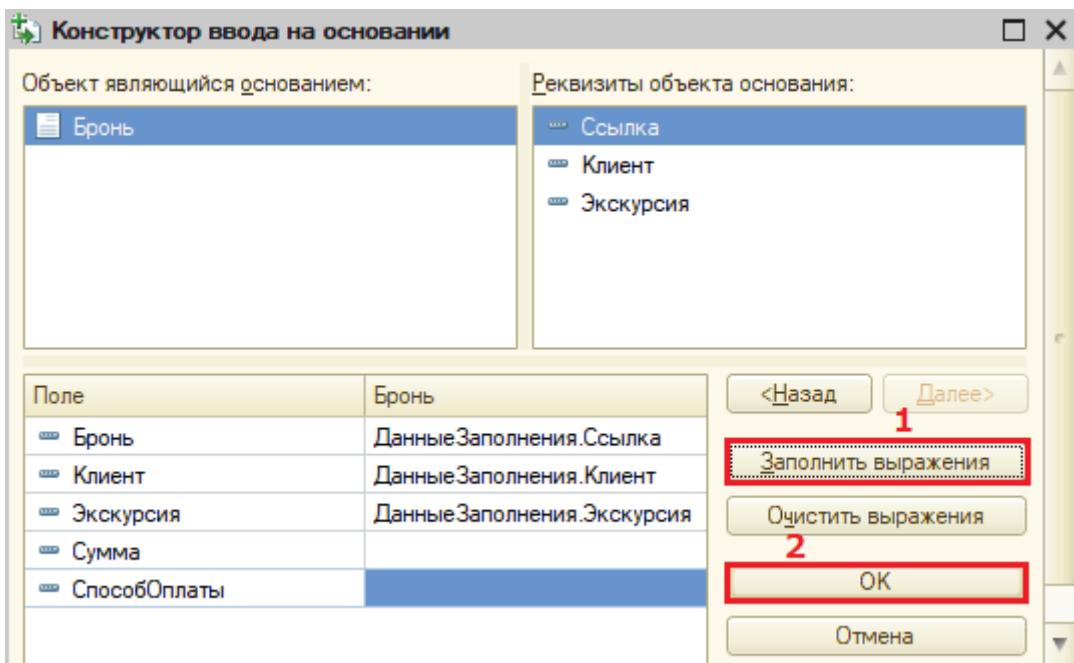
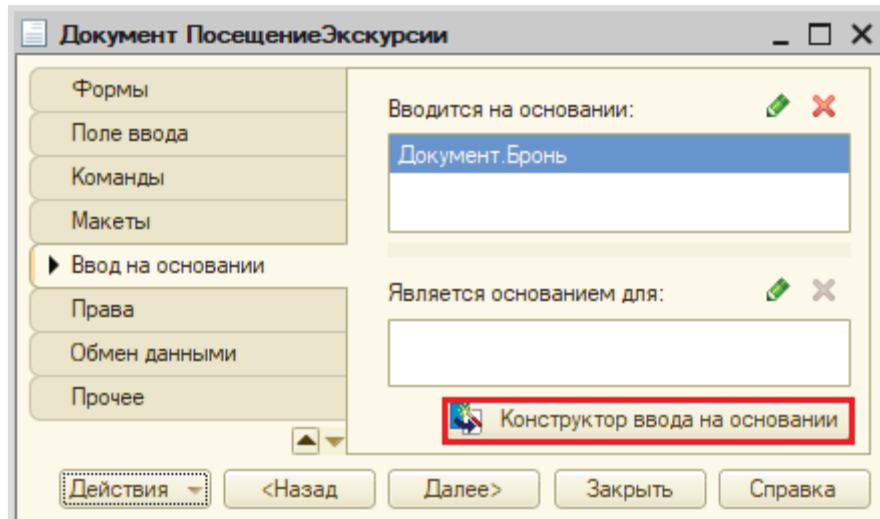
Чтобы им воспользоваться, перейдем на вкладку «Ввод на основании» документа «ПосещениеЭкскурсии».



Поскольку информация будет поступать из документа «Бронь» в документ «ПосещениеЭкскурсии», то второй будет вводиться на основании первого. Укажем это.



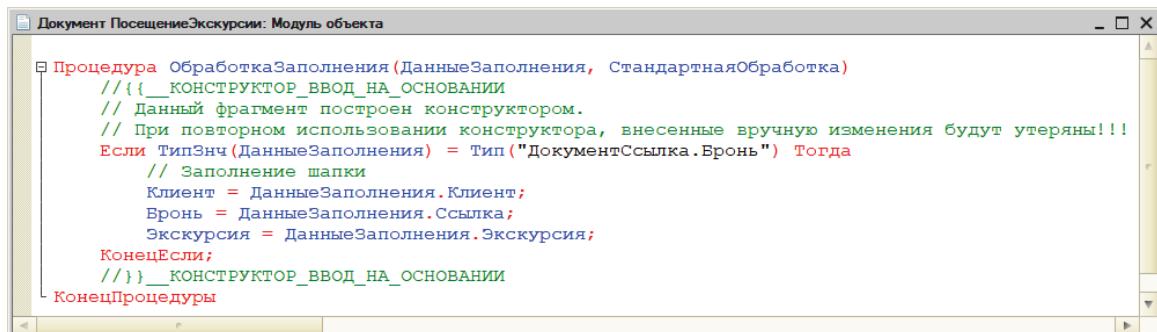
Далее нам необходимо описать *Обработку заполнения* нашего документа. Делать это будем с помощью конструктора ввода на основании.



Нажмите на кнопку «Заполнить выражения». Автозаполнение происходит по принципу совпадения имен реквизитов и типов данных. Также есть возможность заполнить поля вручную.

По завершении работы с конструктором ввода на основании нажмите на кнопку «OK».

В результате будет сформирован программный код Обработки заполнения в модуле объекта документа «ПосещениеЭкскурсии». Данный код описывает, какие данные из документа «Бронь» попадут в текущий документ.



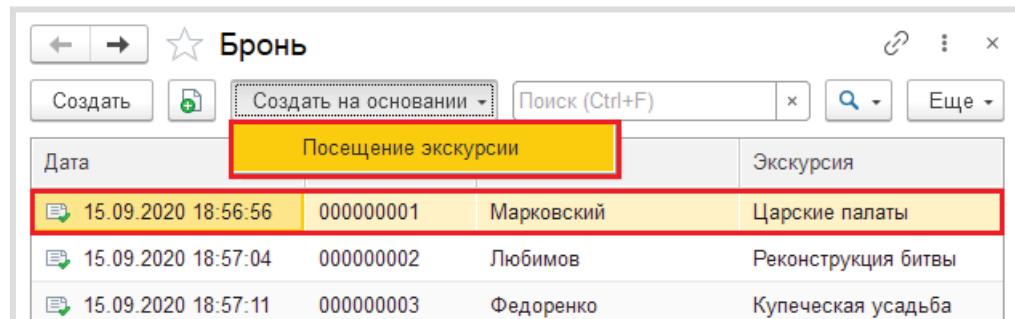
```

Документ ПосещениеЭкскурсии: Модуль объекта

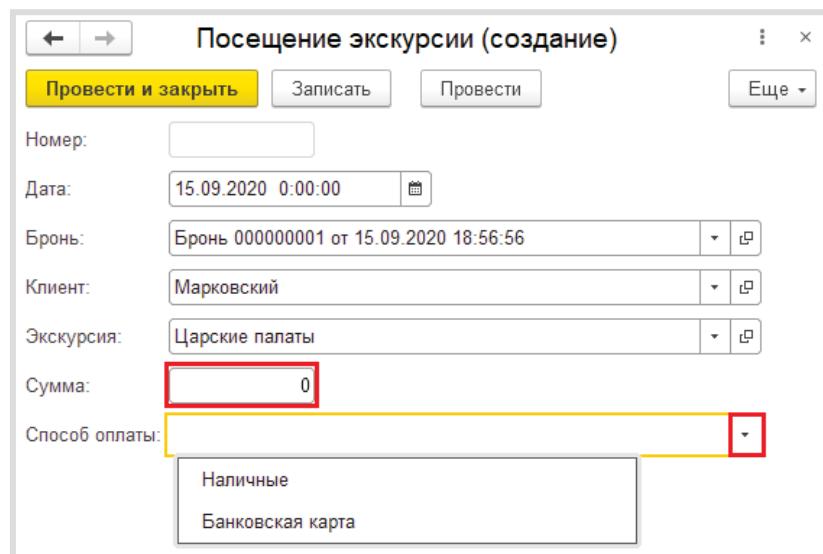
Процедура ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)
    //{{_КОНСТРУКТОР_ВВОД_НА_ОСНОВАНИИ
    //  Данный фрагмент построен конструктором.
    //  При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
    Если ТипЗнач(ДанныеЗаполнения) = Тип("ДокументСсылка.Бронь") Тогда
        // Заполнение шапки
        Клиент = ДанныеЗаполнения.Клиент;
        Бронь = ДанныеЗаполнения.Ссылка;
        Экскурсия = ДанныеЗаполнения.Экскурсия;
    КонецЕсли;
    //}} _КОНСТРУКТОР_ВВОД_НА_ОСНОВАНИИ
КонецПроцедуры

```

После проделанных настроек запустим режим «1С:Предприятие» и создадим несколько документов «Посещение экскурсии» на основании уже созданной брони.



Мы вводим документ «на основании», поэтому большая часть данных уже будет указана.



Посещение экскурсии (создание)	
<input type="button" value="Провести и закрыть"/> <input type="button" value="Записать"/> <input type="button" value="Провести"/> <input type="button" value="Еще"/>	
Номер:	<input type="text"/>
Дата:	15.09.2020 0:00:00 <input type="button" value="..."/>
Бронь:	Бронь 000000001 от 15.09.2020 18:56:56 <input type="button" value="..."/>
Клиент:	Марковский <input type="button" value="..."/>
Экскурсия:	Царские палаты <input type="button" value="..."/>
Сумма:	<input type="text" value="0"/>
Способ оплаты:	<input type="button" value="Наличные"/> <input type="button" value="Банковская карта"/>

Посещение экскурсии (создание) *

← → Провести и закрыть Записать Провести Еще ▾

Номер:	<input type="text"/>
Дата:	15.09.2020 0:00:00 <input type="button" value=""/>
Бронь:	Бронь 00000001 от 15.09.2020 18:56:56 <input type="button" value=""/>
Клиент:	Марковский <input type="button" value=""/>
Экскурсия:	Царские палаты <input type="button" value=""/>
Сумма:	2 000
Способ оплаты:	Наличные <input type="button" value=""/>

Аналогичным способом создайте еще несколько документов «Посещение экскурсии» на основании документов «Бронь».

Внимание! Обязательно укажите разные способы оплаты!

В результате список документов «Посещение экскурсии» должен выглядеть подобным образом:

← → ★ Посещение экскурсии

Создать

Поиск (Ctrl+F) Еще ▾

Дата	Номер	Бронь	Клиент	Экскурсия	Сумма	Способ оплаты
15.09.2020 19:55:57	00000001	Бронь 00000001 от 15.09.2020 18:56:56	Марковский	Царские палаты	2 000	Наличные
15.09.2020 19:56:28	00000002	Бронь 00000002 от 15.09.2020 19:56:28	Любимов	Реконструкция битвы	7 500	Наличные
15.09.2020 19:56:35	00000003	Бронь 00000003 от 15.09.2020 19:56:35	Федоренко	Купеческая усадьба	10 000	Наличные
15.09.2020 19:57:40	00000004	Бронь 00000004 от 15.09.2020 19:57:40	Марковский	Царские палаты	2 000	Банковская карта

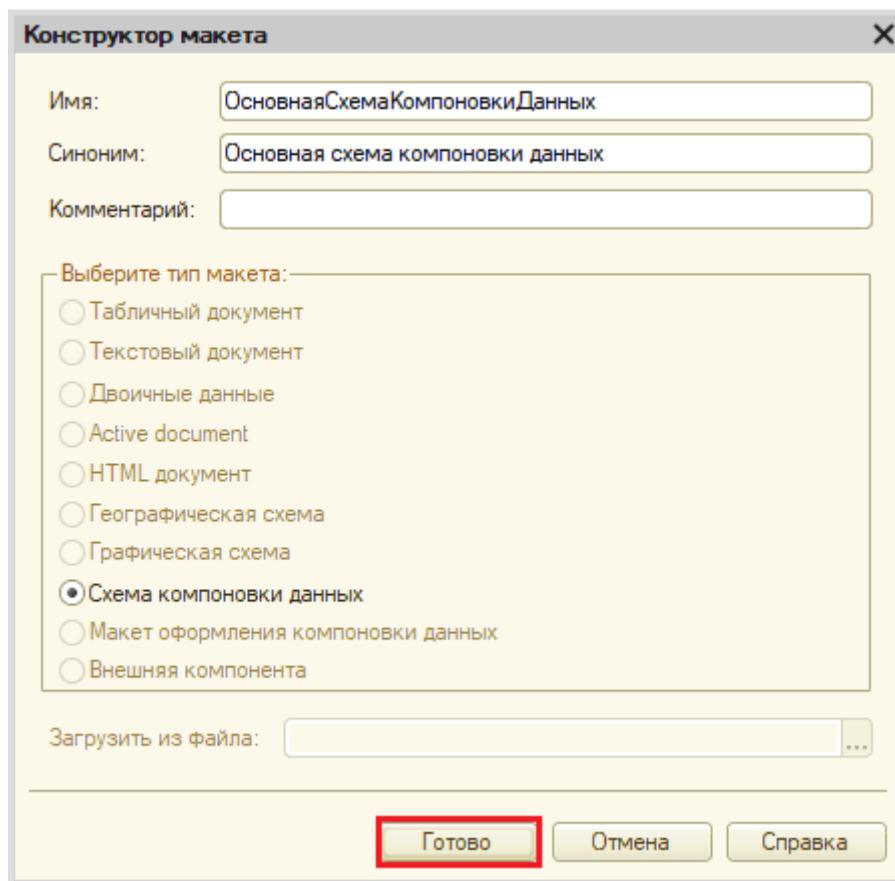
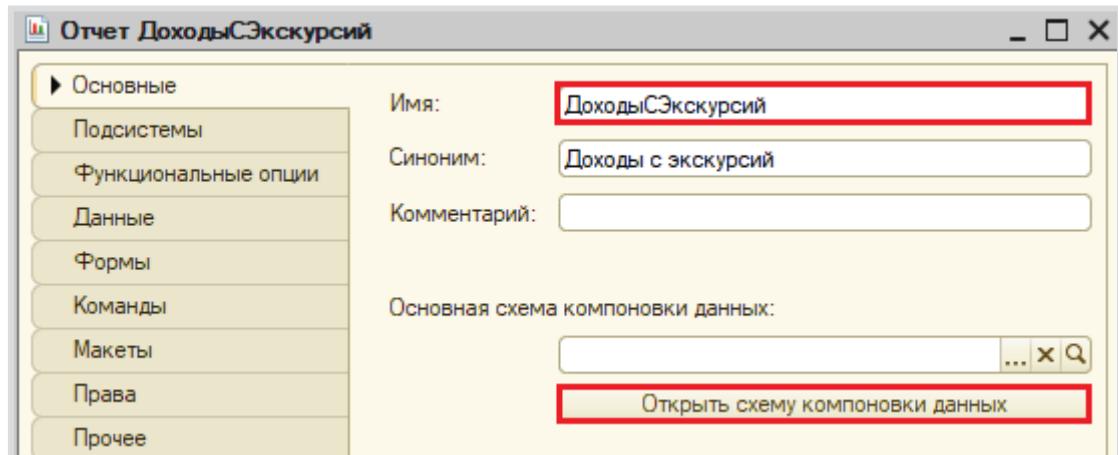
«Нужно построить Отчет о доходах с экскурсий.»

Построим *отчет*. Для этого воспользуемся соответствующим объектом конфигурации.

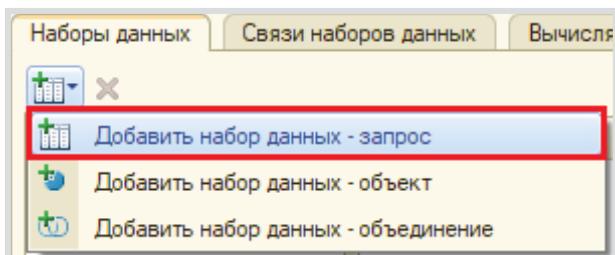
Определение

Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: [https://v8.1c.ru/platforma/отчет/](https://v8.1c.ru/platforma/otchet/)).

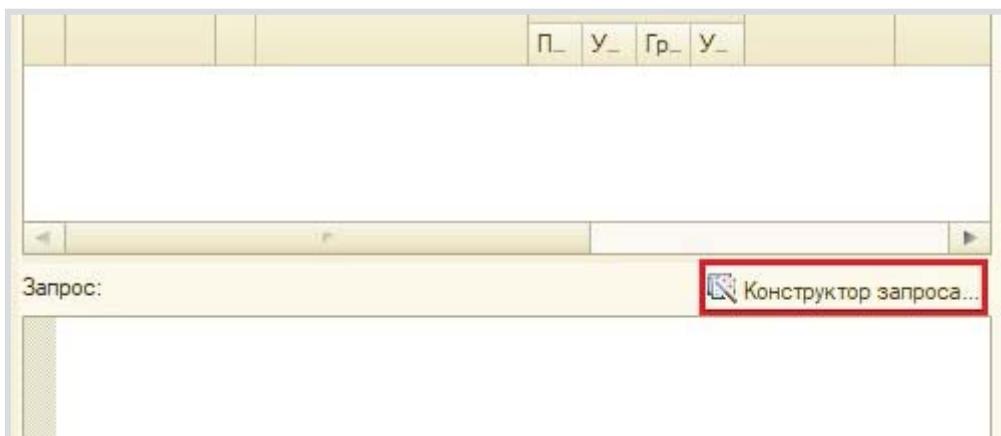
Добавим отчет «ДоходыСЭкскурсий». Воспользуемся схемой компоновки данных.



Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Для формирования запроса воспользуемся *конструктором запроса*.



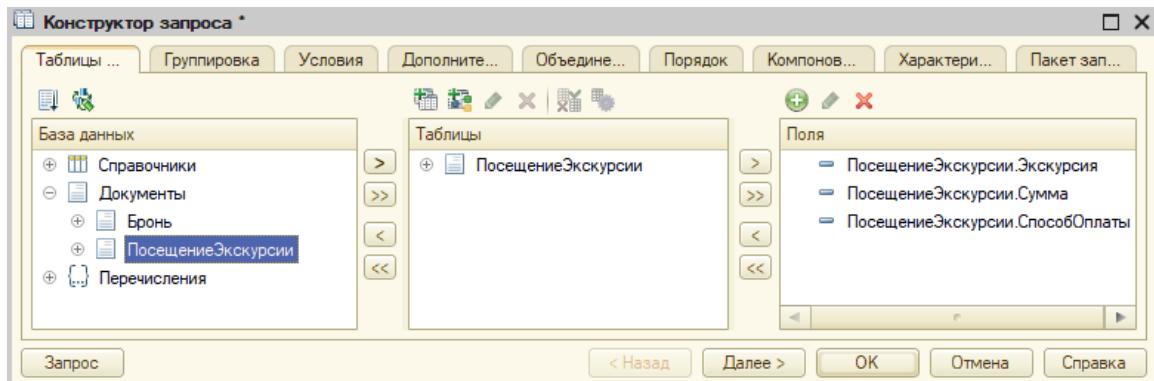
Открывается *конструктор запроса*. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Необходимо выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

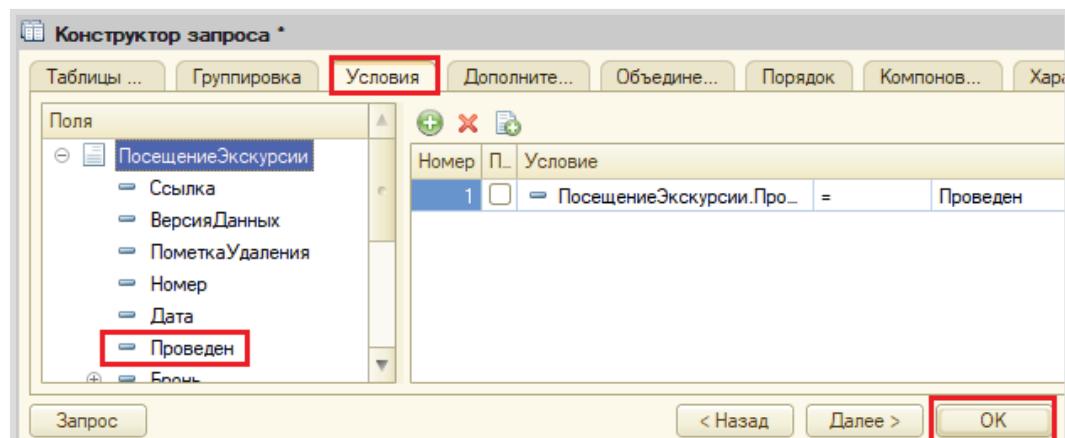
Данные будем брать из *регистра накоплений* напрямую, чтобы иметь возможность рассчитывать средний балл.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



Поскольку в отчет должны попадать только совершенные (проведенные) посещения экскурсий – перейдем на вкладку «Условия».

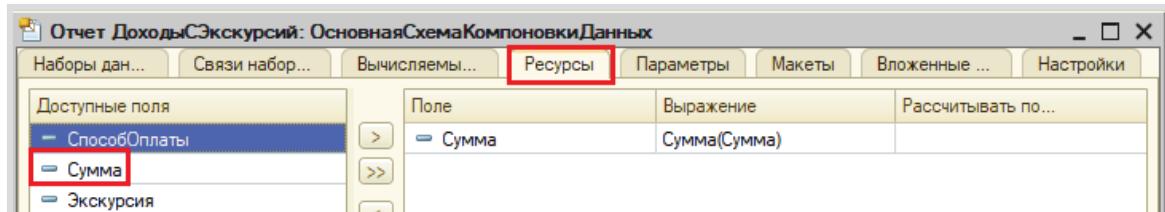


Нажмите на кнопку «OK».

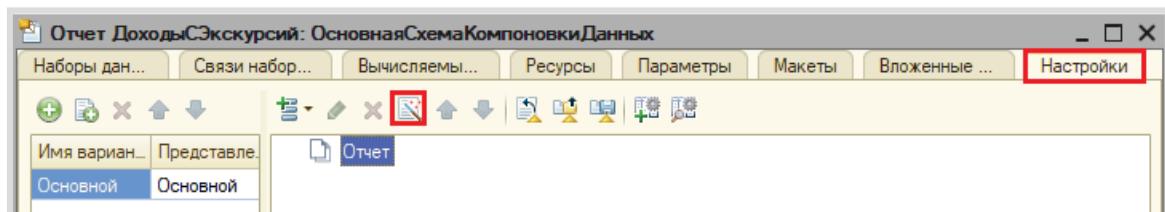
Конструктор сформировал запрос на встроенном языке запросов 1С. Поскольку документ может быть *проведен* или *не проведен*, значит, нужно установить пометку проведения в значение *Истина*. Это нужно для того, чтобы в отчет не попали непроведенные (не совершенные) документы.

```
Запрос: Конструктор запроса...
ВЫБРАТЬ
    ПосещениеЭкскурсии.Экскурсия КАК Экскурсия,
    ПосещениеЭкскурсии.Сумма КАК Сумма,
    ПосещениеЭкскурсии.СпособОплаты КАК СпособОплаты
ИЗ
    Документ.ПосещениеЭкскурсии КАК ПосещениеЭкскурсии
ГДЕ
    ПосещениеЭкскурсии.Проведен = Истина
```

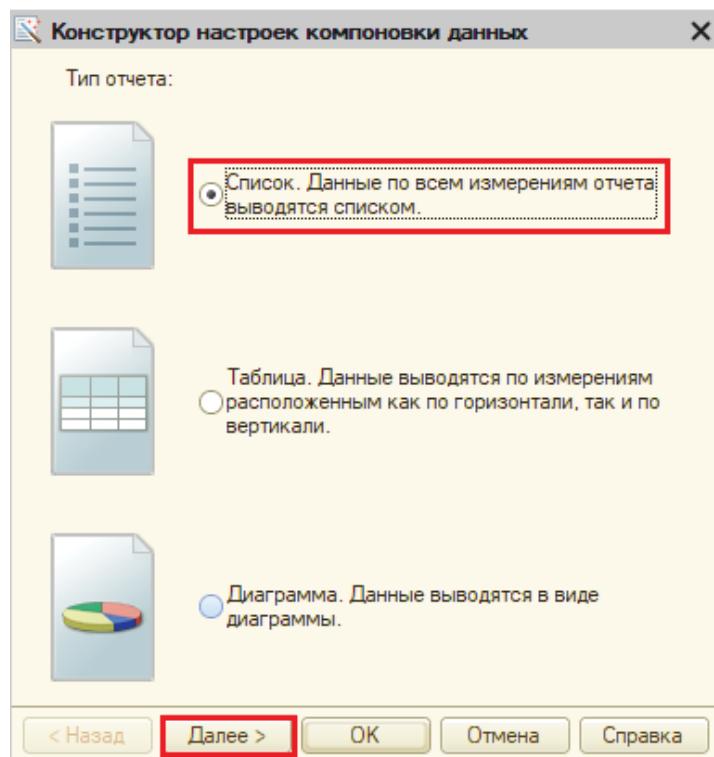
Заказчика интересуют доходы как по конкретным экскурсиям, так и вообще. Для решения этой задачи укажем поле «Сумма» в качестве ресурса на соответствующей вкладке. Теперь для данного поля система будет вести подсчет итогов.



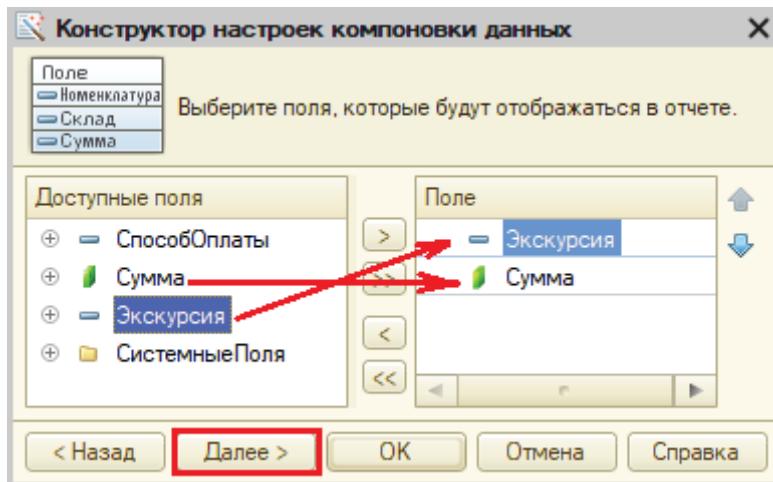
Теперь переходим на вкладку «Настройки» для оформления внешнего вида отчета. Воспользуемся конструктором настроек.



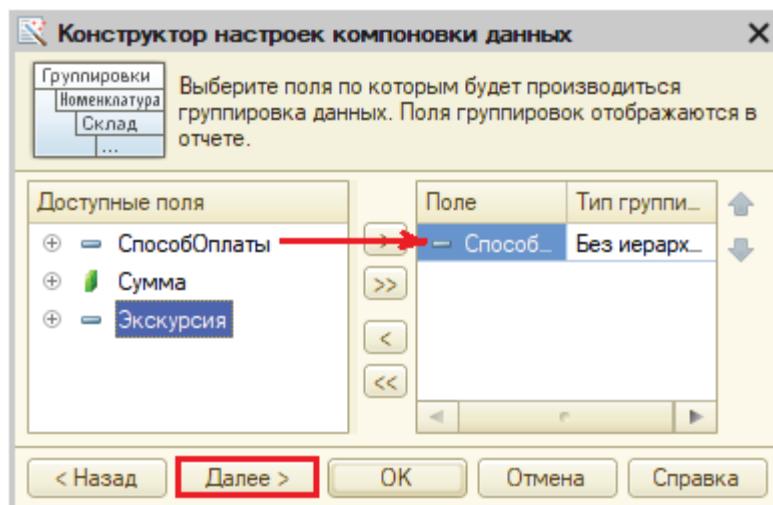
Построим отчет в виде списка.



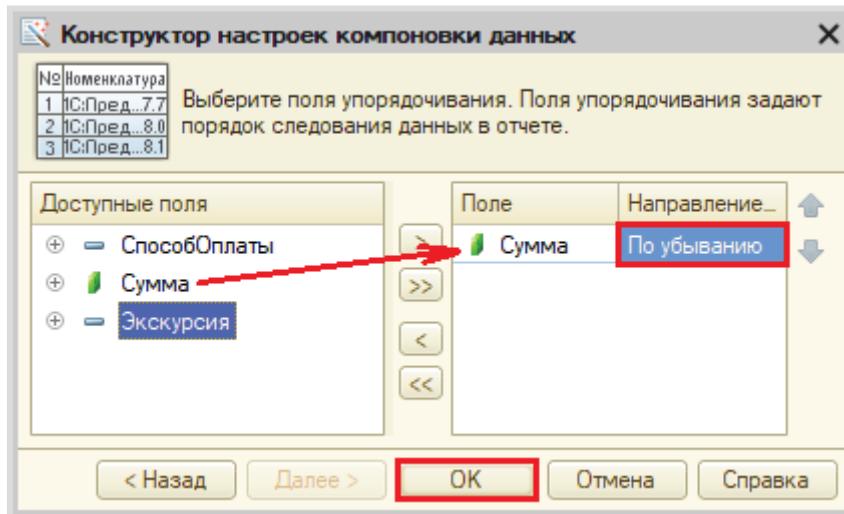
Выберем поля, которые будут отображаться в отчете.



Сгруппируем отображаемые поля по способу оплаты.



Отсортируем наш отчет по убыванию суммы доходов.



Отчет готов. Запустим систему в режиме «1С:Предприятие» и построим его.

★ Доходы с экскурсий		
Сформировать		Выбрать вариант...
	Способ оплаты	Сумма
Экскурсия		
Наличные		19 500
Купеческая усадьба		10 000
Реконструкция битвы		7 500
Царские палаты		2 000
Банковская карта		2 000
Царские палаты		2 000
Итого		21 500

Поставленная задача решена.

Лабораторная работа № 6

РАЗРАБОТКА УЧЕТНОЙ СИСТЕМЫ ДЛЯ ВЕДЕНИЯ ИНФОРМАЦИИ О КАССОВЫХ ОПЕРАЦИЯХ

Сложность: *

Теги: индексы, нумераторы, журналы документов, регистр накопления остатков, работа с управляемой формой

ЗАДАЧА

Заказчик просит разработать учетную систему для ведения информации о кассовых операциях. Кассовые операции включают в себя приход и расход денежных средств с обязательным указанием контрагента и суммы.

Необходимо предусмотреть возможность выбирать из списка только те документы, в которых фигурирует выбранный пользователем контрагент.

Кроме того, нужно реализовать возможность просматривать остаток денежных средств в кассе.

Примерный вид журнала кассовых операций:

Дата		Номер	Тип документа	Контрагент	Сумма
01.01.2020		001	Приход	ООО «Мак»	3000
03.01.2020		002	Расход	ООО «Мак»	7000

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

В первую очередь, следует разобраться с *объектами аналитики*, которые необходимо хранить в информационной системе. Однозначно нужно хранить информацию о контрагентах – это частные или юридические лица, с которыми будут вестись любые отношения. Для хранения контрагентов будем использовать *справочник*.

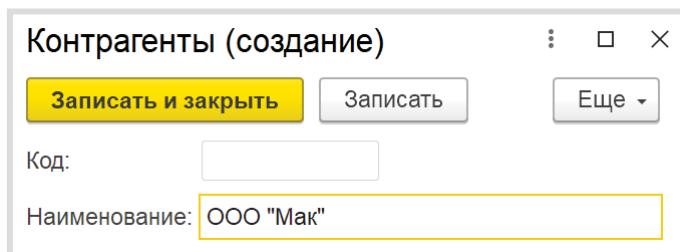
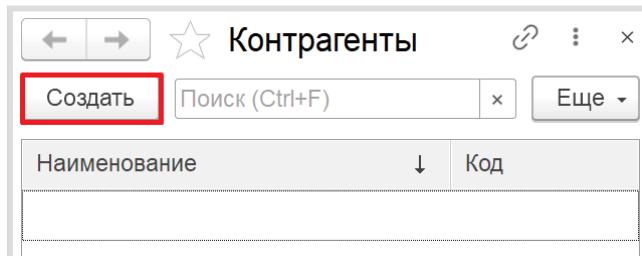
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Создайте справочник «Контрагенты».



Запустите режим «1С:Предприятие» и добавьте несколько контрагентов.

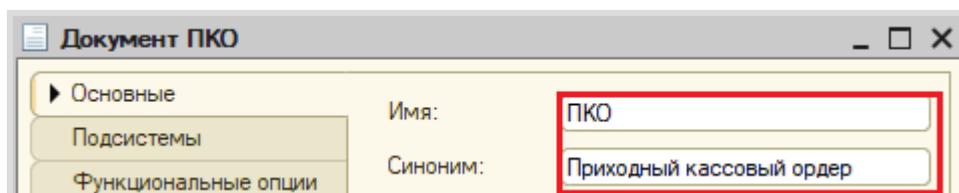


Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.

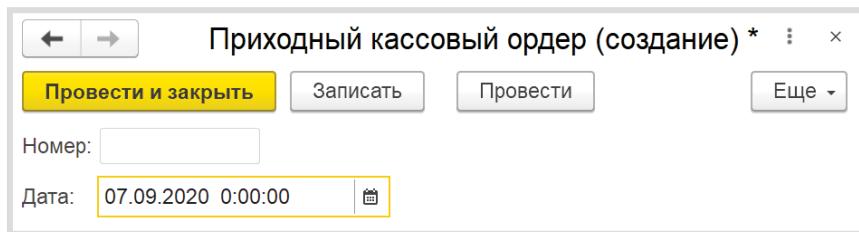
Таким образом, была реализована возможность хранения информации о контрагентах.

Как же регистрировать движения денежных средств? Для отражения денежных операций, как и любых других событий, в информационной системе потребуется объект конфигурации **документ**. Подробнее про документы можно прочитать [здесь](https://v8.1c.ru/platforma/dokumenty/): <https://v8.1c.ru/platforma/dokumenty/>.

Добавьте новый документ «ПКО» (Приходный кассовый ордер, или другими словами – поступление денег в кассу).



Запустите конфигурацию в режиме «1С:Предприятие». Попробуйте добавить документ «ПКО».

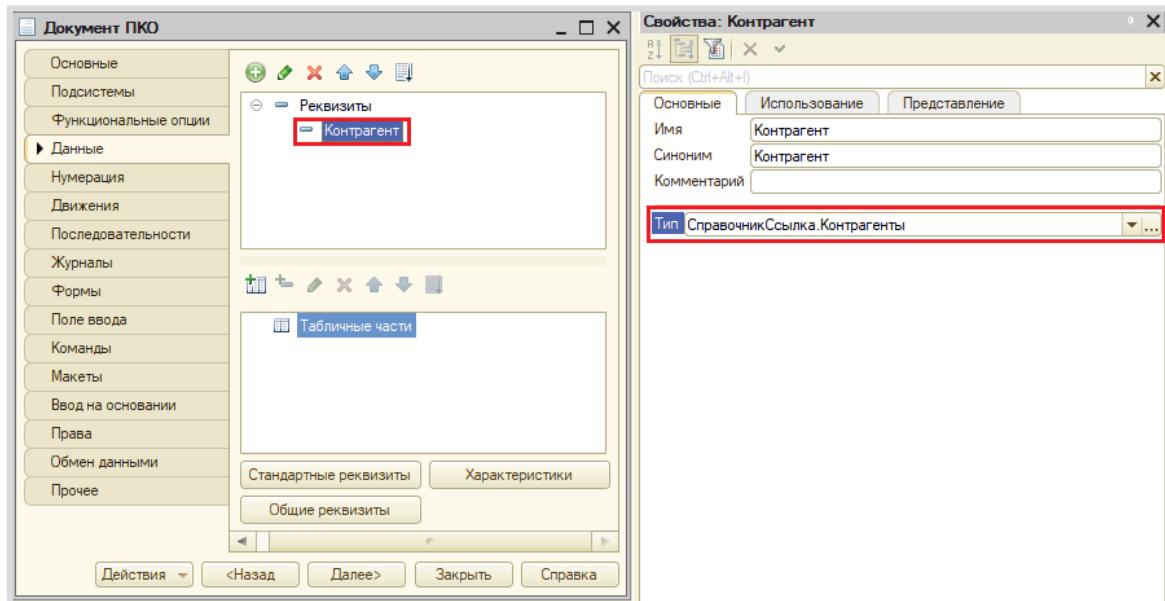


Любой документ может находиться в одном из двух состояний: *подготовленный к свершению* или *совершенный*:

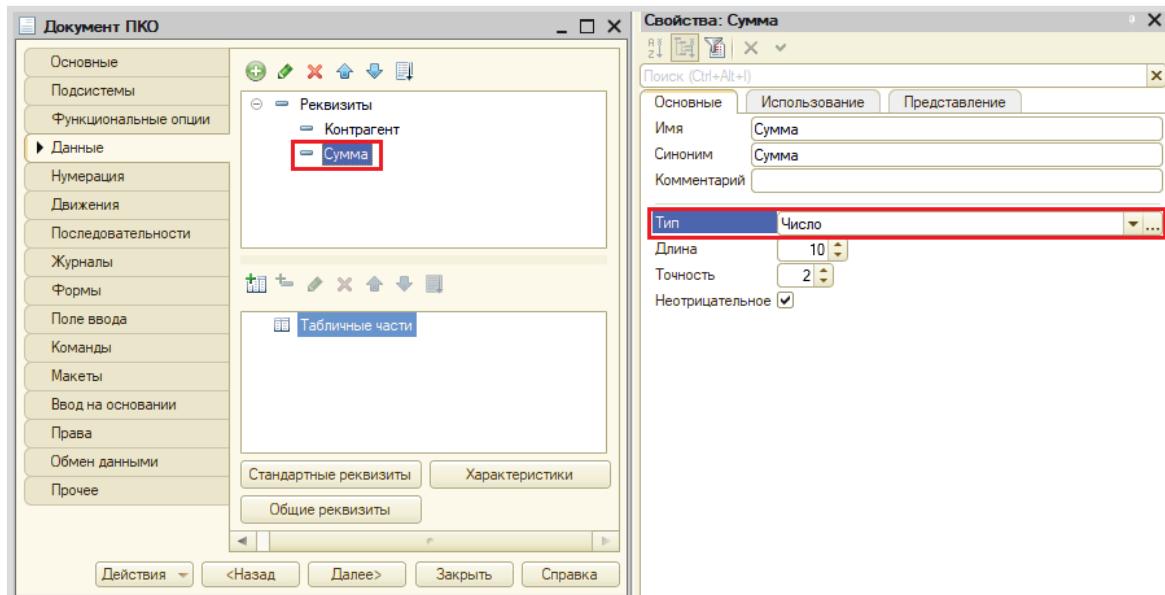
- чтобы подготовить документ для использования в будущем, нужно его записать;
- чтобы отметить документ как совершенный – провести.

Легко заметить, что система сгенерировала для документа другие стандартные реквизиты: «Номер» и «Дата». Оба поля заполняются автоматически, дата может быть изменена.

Одних лишь стандартных реквизитов для решения задачи недостаточно. Необходимо добавить свои реквизиты. Для этого откройте окно редактирования документа на вкладке «Данные» и добавьте реквизит «Контрагент» (тип – «СправочникСсылка.Контрагенты»). Данный реквизит будет хранить ссылку на элемент справочника «Контрагенты».



Далее нам необходимо добавить еще один реквизит, чтобы отслеживать, какую денежную сумму мы отдали или получили от контрагента. Для этого нужно добавить реквизит «Сумма», тип – «Число», точность – «2».



Запустите конфигурацию в режиме «1С:Предприятие» и посмотрите на изменения в документе.

The screenshot shows the 'Receipt Cash Order (creation)' document form. The 'Post and Close' button is highlighted in yellow. The 'Contractor' field has a dropdown menu open, showing a search input field and a 'Show All' button ('Показать все') which is highlighted with a red border.

The screenshot shows the 'Contractors' selection dialog. The 'Select' button is highlighted in yellow. A contractor named 'ООО "Мак"' (OOO "Mak") is selected and highlighted with a yellow background. The 'Search' and 'More' buttons are also visible.

Приходный кассовый ордер (создание) *

Провести и закрыть Записать Провести Еще

Номер:

Дата: 01.09.2020 0:00:00

Контрагент: ООО "Мак"

Сумма: 1 200,00

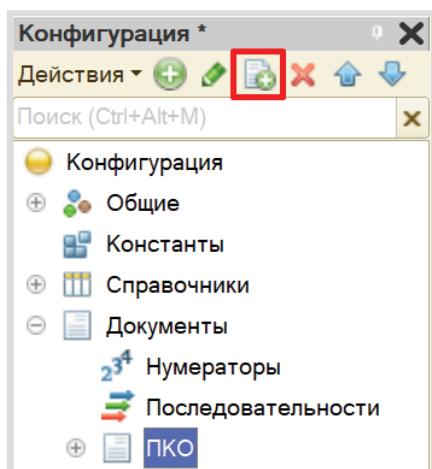
Добавьте еще несколько документов ПКО.

Приходный кассовый ордер

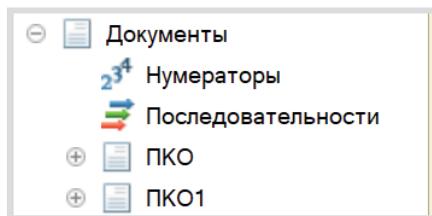
Дата	Номер	Контрагент	Сумма
01.09.2020 10:12:12	000000001	ОАО "Мак"	1 200,00
01.09.2020 10:12:56	000000002	ООО "Василёк"	6 000,00
01.09.2020 10:13:33	000000005	ОАО "Мак"	3 756,00

Аналогично документу «ПКО» необходимо создать и документ «РКО» (Расходный кассовый ордер). Можно повторить действия, описанные для документа «ПКО», а можно скопировать этот документ со всем его содержимым.

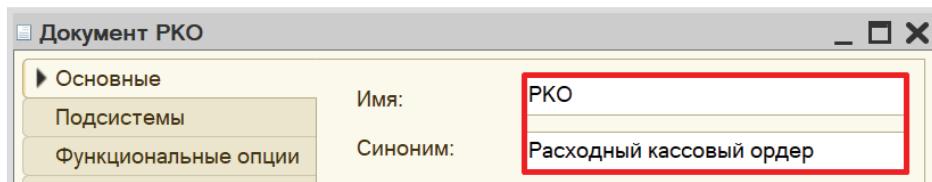
Для этого выделите нужный документ в окне конфигурации и нажмите на кнопку «Добавить копированием».



Появится документ, полностью копирующий документ «ПКО», – «ПКО1».



Необходимо переименовать документ «ПКО1» в «ПКО».



Перейдите на вкладку «Данные» и убедитесь в том, что реквизиты «Контрагент» и «Сумма» существуют и имеют правильные типы данных.

Теперь можно запустить систему в режиме «1С:Предприятие» и добавить несколько документов «РКО».

Расходный кассовый ордер				
Дата	Номер	Контрагент	Сумма	
01.09.2020 14:00:00	000000003	ООО "Василёк"	3 000,00	
01.09.2020 15:00:00	000000004	ОАО "Мак"	1 500,00	
01.09.2020 16:00:00	000000006	ООО "Василёк"	1 234,00	

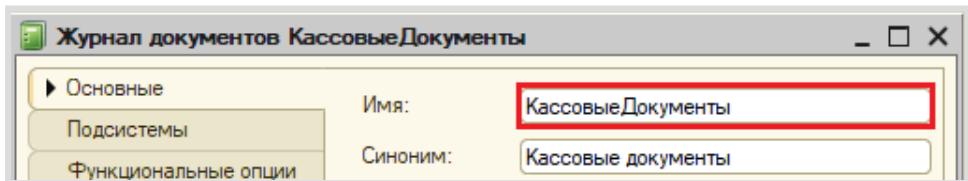
Таким образом, были реализованы документы, фиксирующие движение денежных средств. Но задача была поставлена так: все документы ПКО и РКО должны находиться в одной таблице. Причем добавляться эти документы должны по мере их создания.

Для решения этой задачи необходимо использовать объект конфигурации *Журнал документов*.

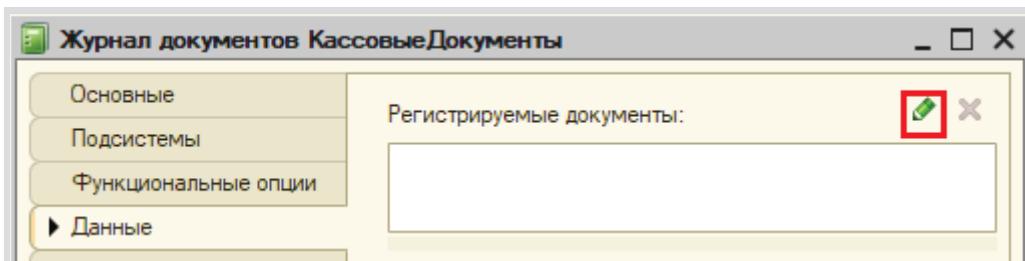
Определение

Журнал представляет собой отдельную таблицу, в которую будут попадать данные из указанных документов. В этом случае одинаковые данные будут храниться и в документах своего вида, и в журнале, тем самым будут дублироваться. Более подробно про журнал документов можно узнать здесь: <https://v8.1c.ru/platforma/zhurnal-dokumentov/>.

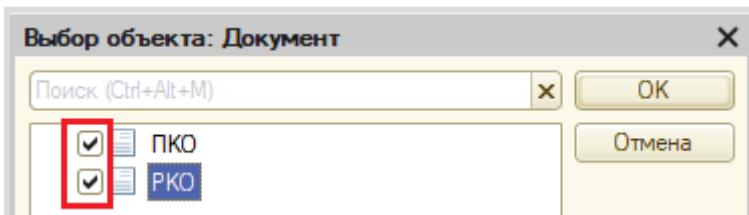
Добавьте новый журнал «Кассовые документы».



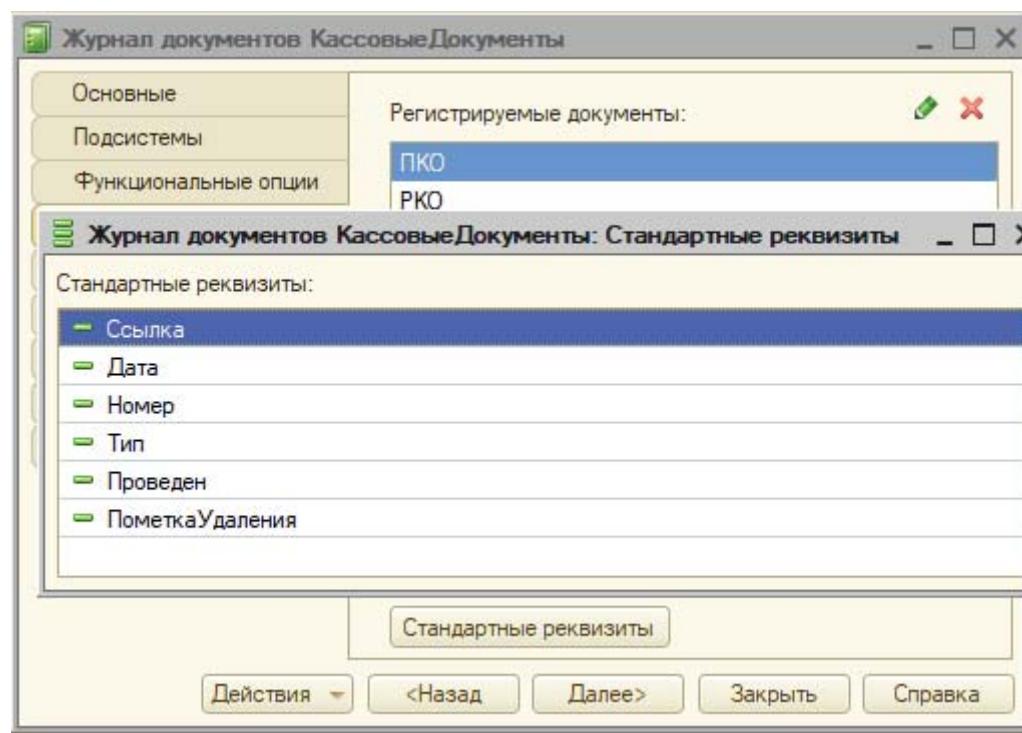
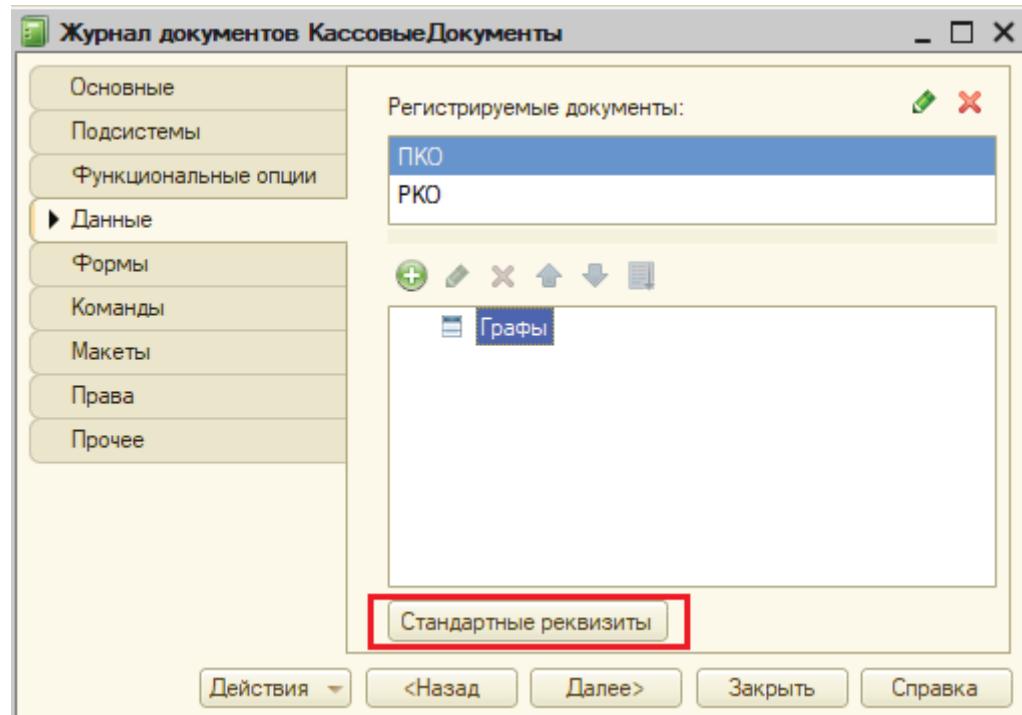
Переходим на вкладку «Данные». Здесь нужно указать, какие документы будут попадать в Журнал документов. Чтобы добавить документ, нажмите на иконку «зеленого карандаша».



В открывшемся окне выберите два созданных ранее документа. Нажмите кнопку «OK».

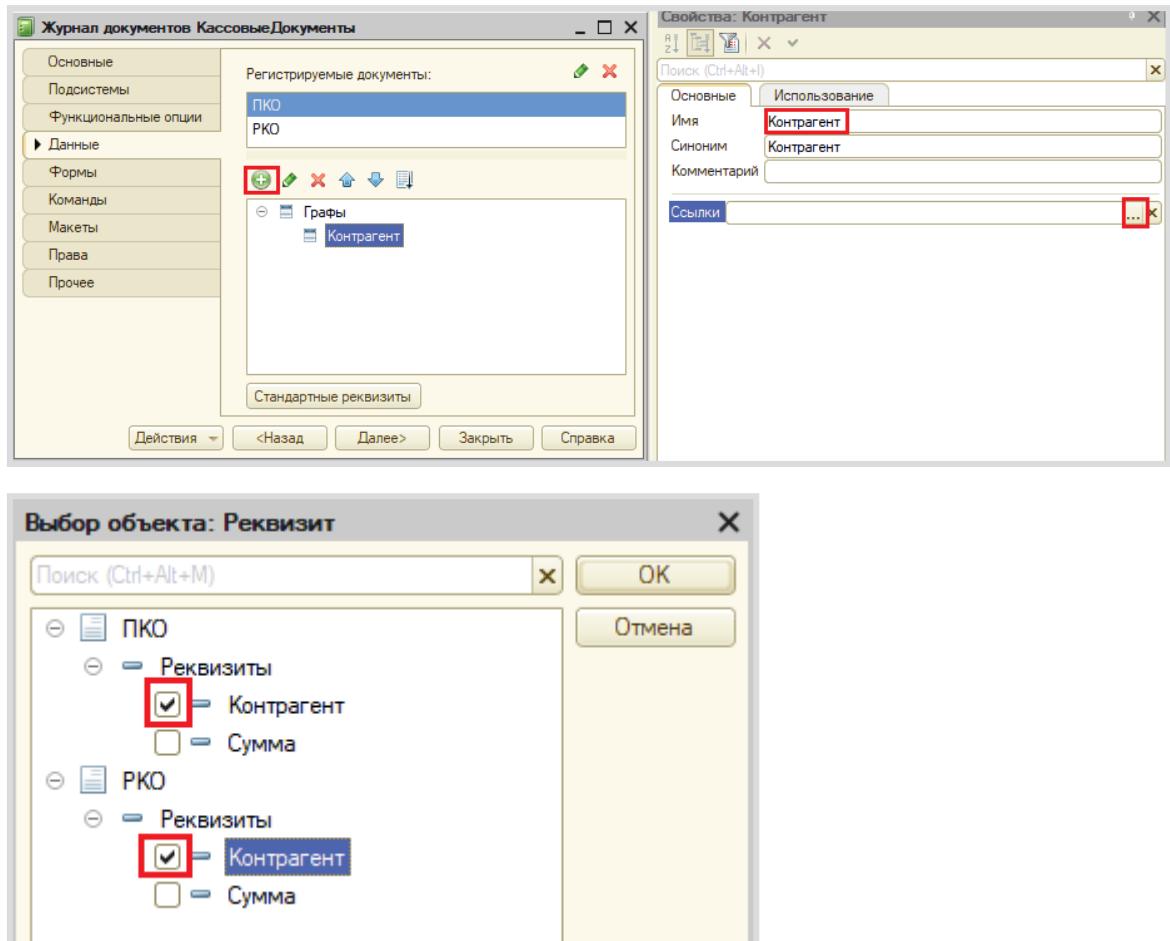


Давайте посмотрим на стандартные реквизиты журнала документов.

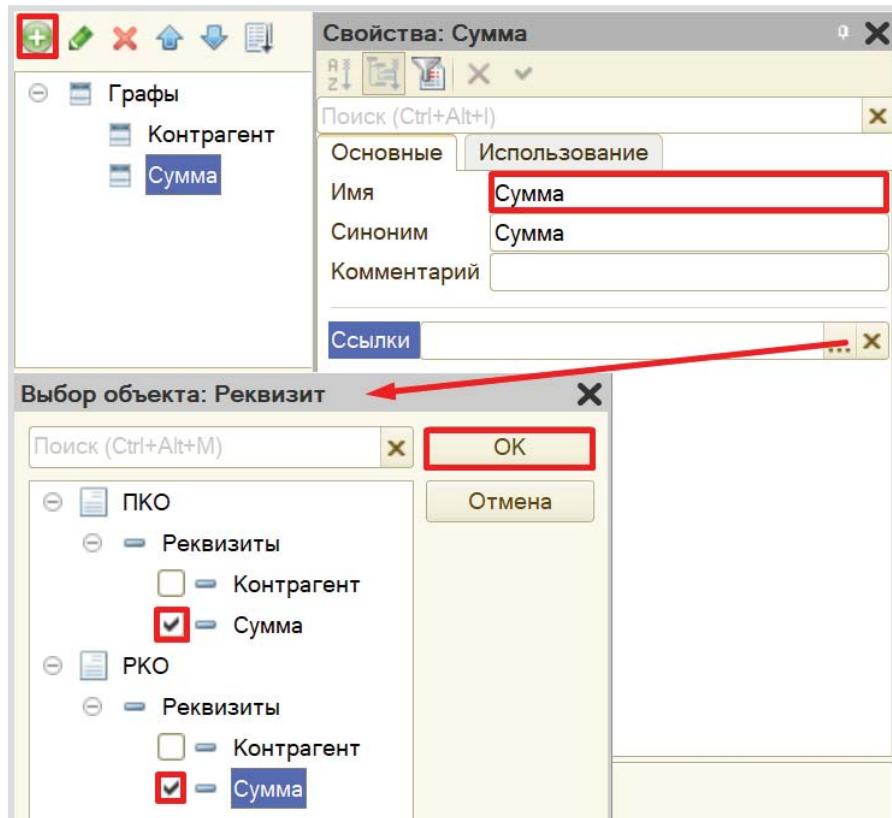


Интересующими нас реквизитами являются «Дата», «Номер» и «Тип» документа. Явно не хватает еще двух полей: «Контрагент» и «Сумма». В журнале это называется *графа*. Добавьте графу «Контрагент».

Далее нужно указать, из каких реквизитов данные будут попадать в графу. Для этого нажмите на кнопку «многоточия» свойства *Ссылки*. Выберите реквизит «Контрагент» из обоих документов.



Аналогичным образом добавьте графу «Сумма». Ссылки настройте так же: сделайте источником графы реквизит «Сумма» обоих документов.

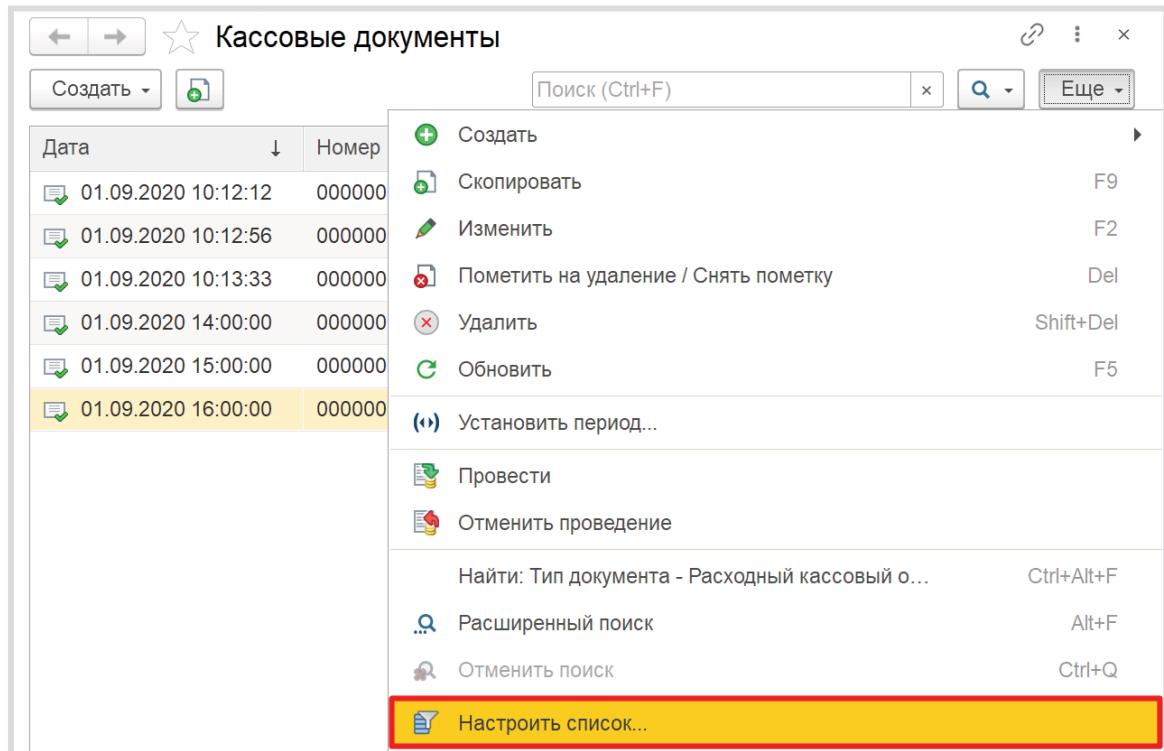


Запустите систему в режиме «1С:Предприятие» и проверьте работу журнала.

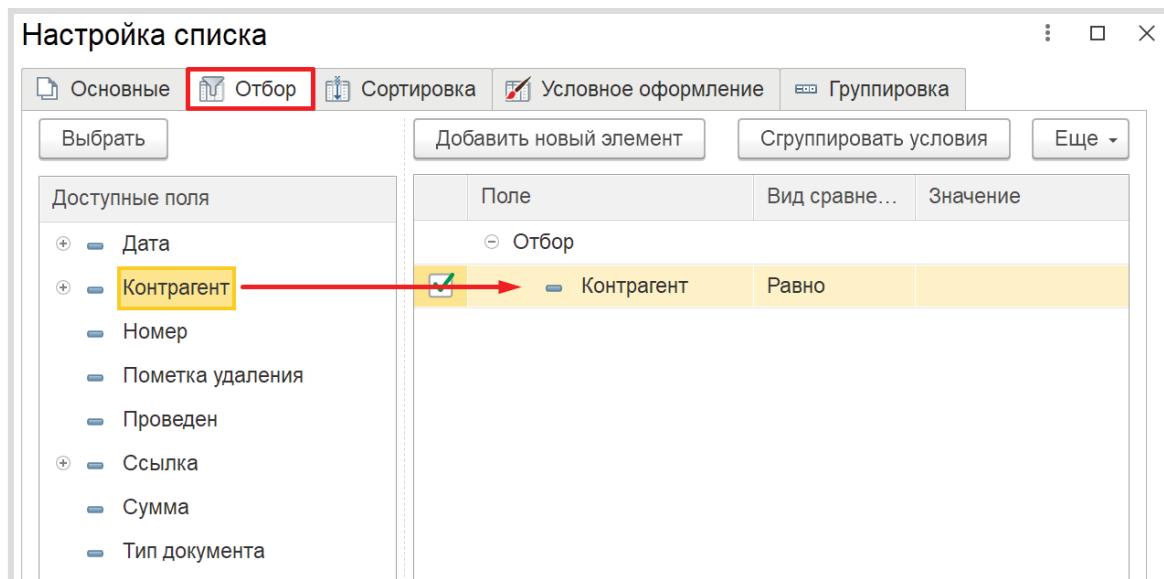
Кассовые документы					
Дата	Номер	Тип документа	Контрагент	Сумма	
01.09.2020 10:12:12	000000001	Приходный кассовый ордер	ОАО "Мак"	1 200,00	
01.09.2020 10:12:56	000000002	Приходный кассовый ордер	ООО "Василёк"	6 000,00	
01.09.2020 10:13:33	000000005	Приходный кассовый ордер	ОАО "Мак"	3 756,00	
01.09.2020 14:00:00	000000003	Расходный кассовый ордер	ООО "Василёк"	3 000,00	
01.09.2020 15:00:00	000000004	Расходный кассовый ордер	ОАО "Мак"	1 500,00	
01.09.2020 16:00:00	000000006	Расходный кассовый ордер	ООО "Василёк"	1 234,00	

Можно настроить журнал документов и добавить отбор по контрагенту. Таким образом, можно быстро посмотреть на все документы, в которых фигурирует выбранный пользователем контрагент.

Для этого нажмите на кнопку «Еще» и выберите пункт меню «Настроить список...».



В открывшемся окне перейдите на вкладку «Отбор» и установите отбор по полю «Контрагент».



По окончании работы нажмите на кнопку «Завершить редактирование».

Попробуйте отобразить документы с каким-либо контрагентом.

Дата	Номер	Тип документа	Контрагент	Сумма
01.09.2020 10:12:12	000000001	Приходный кассовый ордер	ОАО "Мак"	1 200,00
01.09.2020 10:13:33	000000005	Приходный кассовый ордер	ОАО "Мак"	3 756,00
01.09.2020 15:00:00	000000004	Расходный кассовый ордер	ОАО "Мак"	1 500,00

Таким образом, мы реализовали возможность хранить несколько видов документов в одном списке, а также делать отбор по контрагенту.

Можно ли теперь на основе этих документов узнать остаток денег в кasse? Можно, но для этого придется прибегнуть к грубому перебору всех существующих документов. Данный вариант является неправильным, потому что, если таких документов окажется очень много, система будет требовать большого количества ресурсов и времени.

Для решения данной проблемы и ускорения процесса извлечения данных создадим еще один объект – *регистр накопления*.

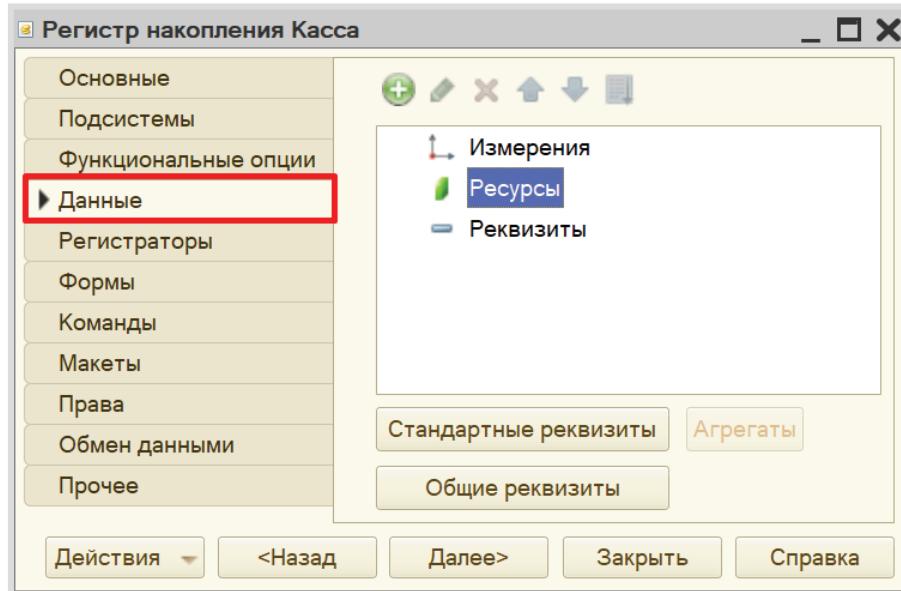
Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Создайте регистр и назовите его «Касса».

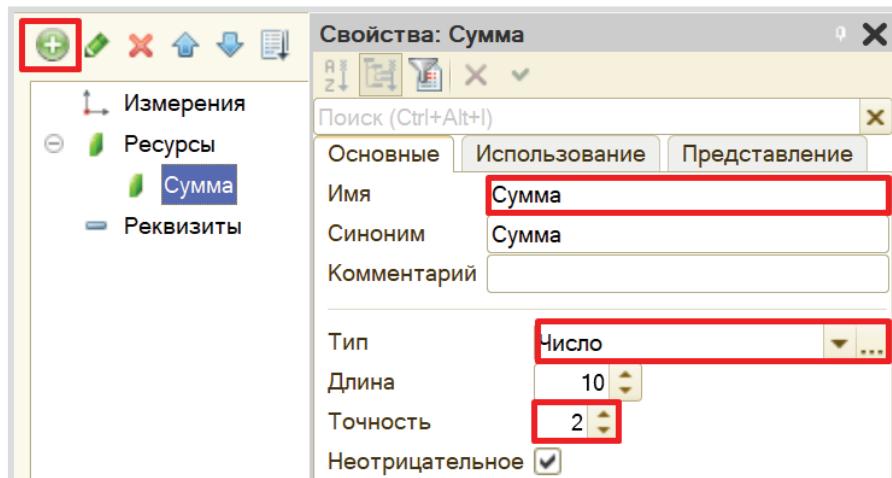
Основные	Имя: <input type="text" value="Касса"/>
Подсистемы	Синоним: <input type="text" value="Касса"/>
Функциональные опции	Комментарий: <input type="text"/>
Данные	
Регистраторы	
Формы	
Команды	

Для формирования структуры регистра переходим на вкладку «Данные».



Структура *регистра накопления* отличается от структуры документа.

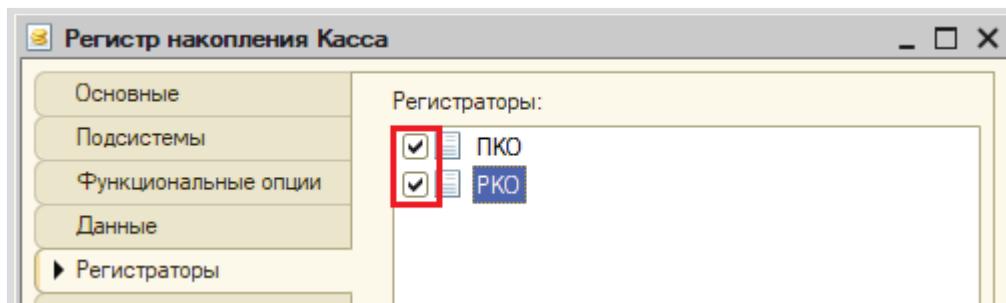
Добавим ресурс. Чтобы понять, что использовать в качестве ресурса, необходимо задать вопрос: «Что мы хотим накапливать/считать в данном регистре?». Мы хотим считать сумму. Следовательно, сумма и будет являться ресурсом. Тип данного реквизита – «Число».



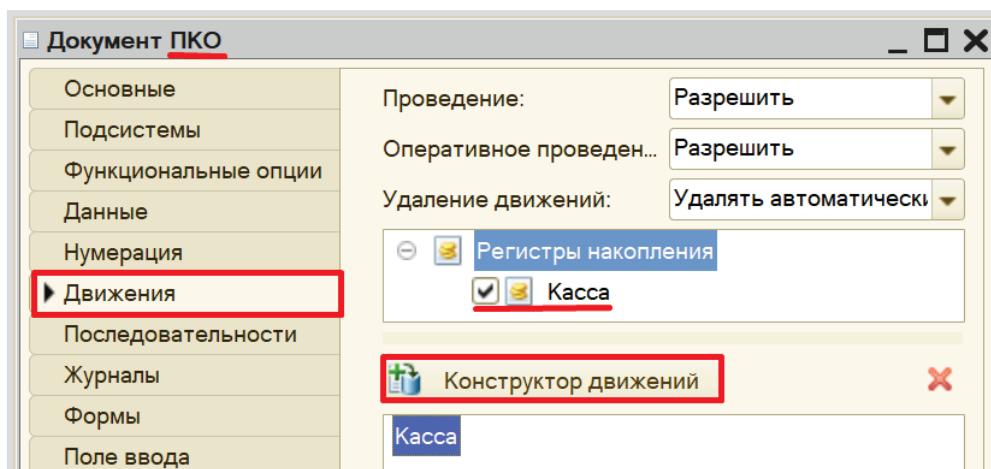
Чтобы *регистр накопления* заработал, нужно сделать следующее:

1. Определить источники данных, которые должны попадать в регистр (определить документы-регистраторы).
2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.

Для определения документов-регистраторов переходим на вкладку «Регистраторы». Здесь нужно выбрать объекты, которые будут передавать данные в регистр. В нашем случае, это документы «ПКО» и «РКО».



Далее требуется описать алгоритмы передачи данных для каждого документа. Откройте окно редактирования документа «ПКО» на вкладке «Движения». Воспользуйтесь *конструктором движений*.

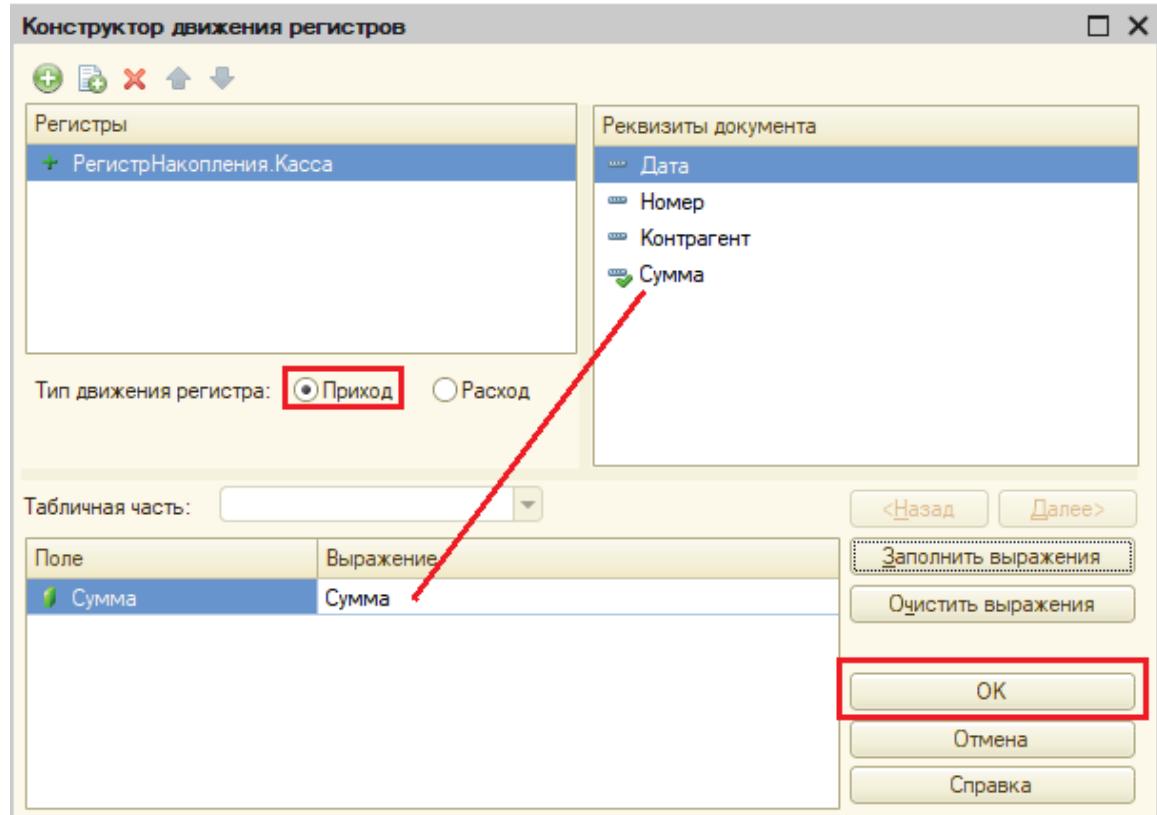


Окно *конструктора движений* состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (поскольку один документ может делать движения сразу в нескольких разных регистрах);
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части нужно выбрать ее в соответствующем поле;
- В нижней части окна описаны реквизиты *регистра накопления*. Нужно заполнить поле «Выражение» реквизитами документа.

Поскольку поступление денежных средств должно увеличивать сумму денег в кассе, то тип движения регистра необходимо выбрать «Приход». Регистр будет обозначаться знаком «+» (плюс).

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.



При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа сформирует движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

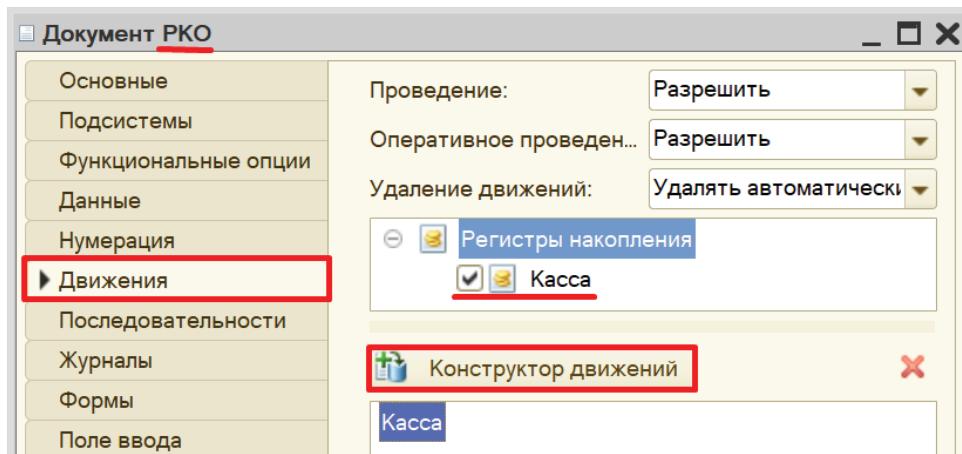
```
Документ ПКО: Модуль объекта
Процедура ОбработкаПроведения(Отказ, Режим)
//{{_КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
// Данный фрагмент построен конструктором.
// При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!

// регистр Касса Приход
Движение.Касса.Записывать = Истина;
Движение = Движение.Касса.Добавить();
Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
Движение.Период = Дата;
Движение.Сумма = Сумма;

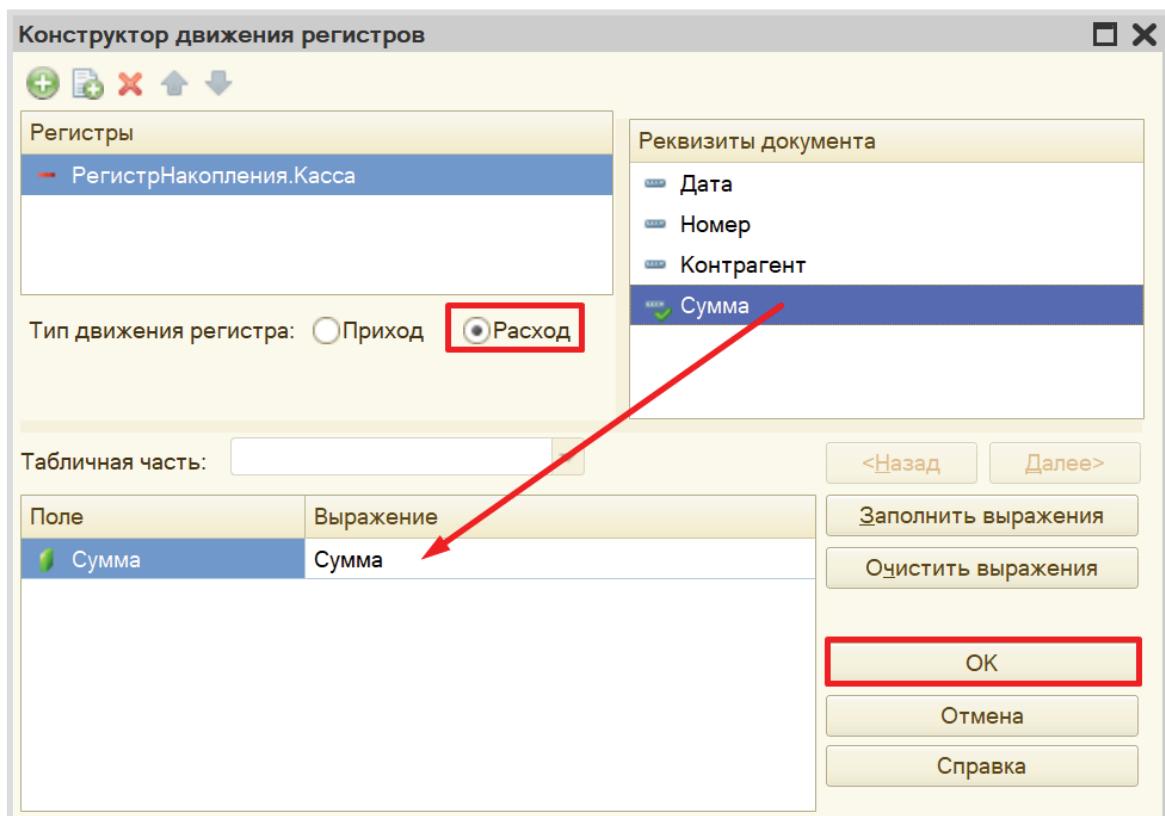
//}}}_КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Аналогичные действия нужно совершить и для документа «РКО».

Откройте окно редактирования на вкладке «Движения» и нажмите на кнопку «Конструктор движений».



Окно конструктора движений заполняется аналогично, с одной лишь разницей: документ будет иметь тип «Расход», поскольку будет уменьшать количество денежных средств в кассе.



□ Документ РКО: Модуль объекта

```

    Процедура ОбработкаПроведения(Отказ, Режим)
        //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
        //  Данный фрагмент построен конструктором.
        //  При повторном использовании конструктора, внесенные
        //  изменения не будут утеряны.

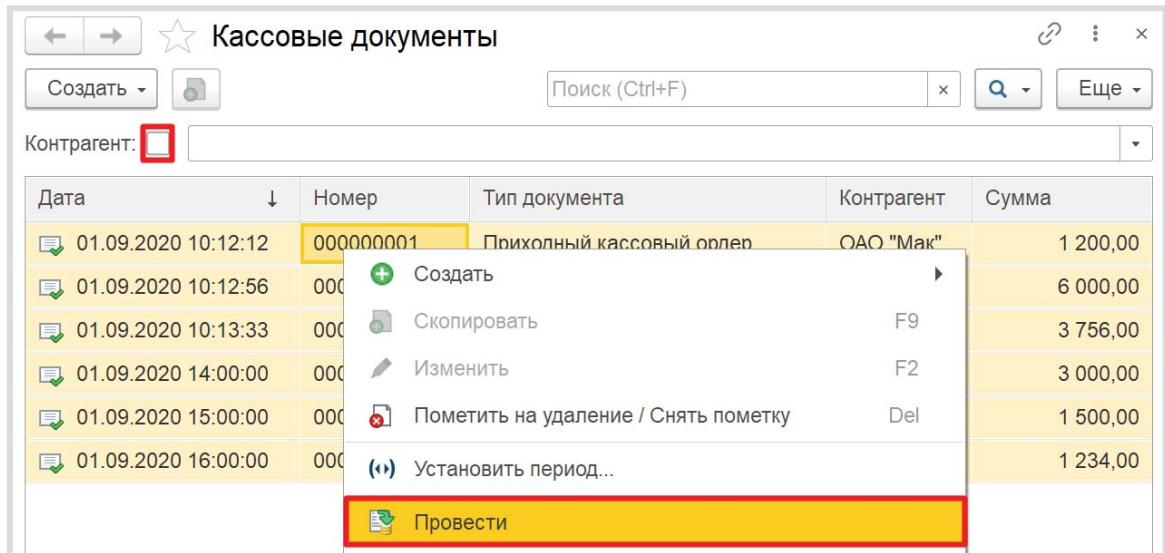
        // регистр Касса Расход
        Движение.Касса.Записывать = Истина;
        Движение = Движение.Касса.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Сумма = Сумма;

    //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    КонецПроцедуры

```

Откройте конфигурацию в режиме «1С:Предприятие».

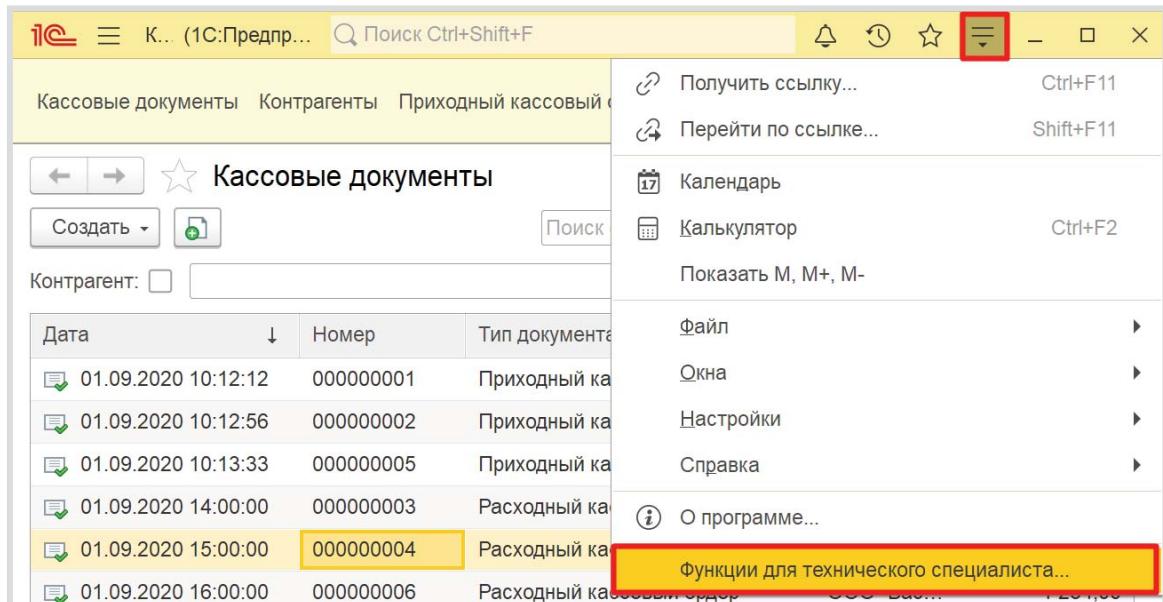
В первую очередь, необходимо перепровести (провести заново) все созданные нами ранее документы. Это легко можно сделать в журнале документов. Убедитесь, что галочка для отбора документов по контрагенту снята и перед вами список всех созданных документов. Выделите все документы с помощью комбинации клавиш **CTRL+A**. Затем щелкните по документам правой кнопкой мыши и выберите пункт контекстного меню «Провести».



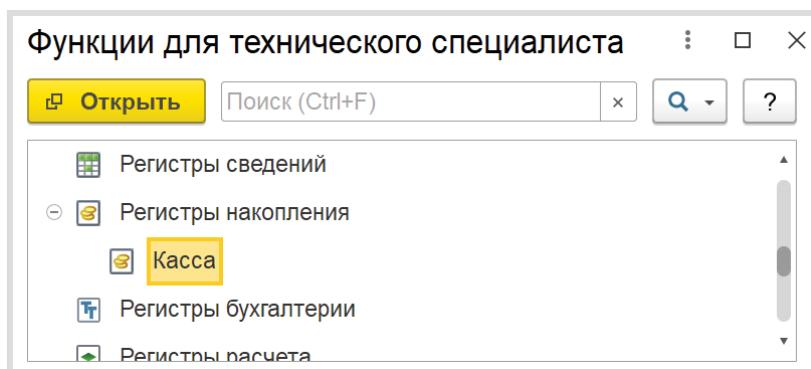
При проведении документов производятся движения (передача данных) в *регистр накопления*.

Обратите внимание, что на главной странице системы не создала кнопку открытия регистра накопления. Это связано с тем, что все расчеты в регистрах накопления происходят в фоновом режиме, «за кадром», то есть пользователю о них знать не нужно вовсе. Поэтому по умолчанию регистры накопления настраивают так, чтобы пользователи не имели к ним доступа.

Но мы, будучи разработчиками, можем обратиться к любому объекту конфигурации. Для этого воспользуемся функциями для технического специалиста.



Среди перечня всех созданных нами объектов найдем *регистр накопления «Касса»* и откроем его.



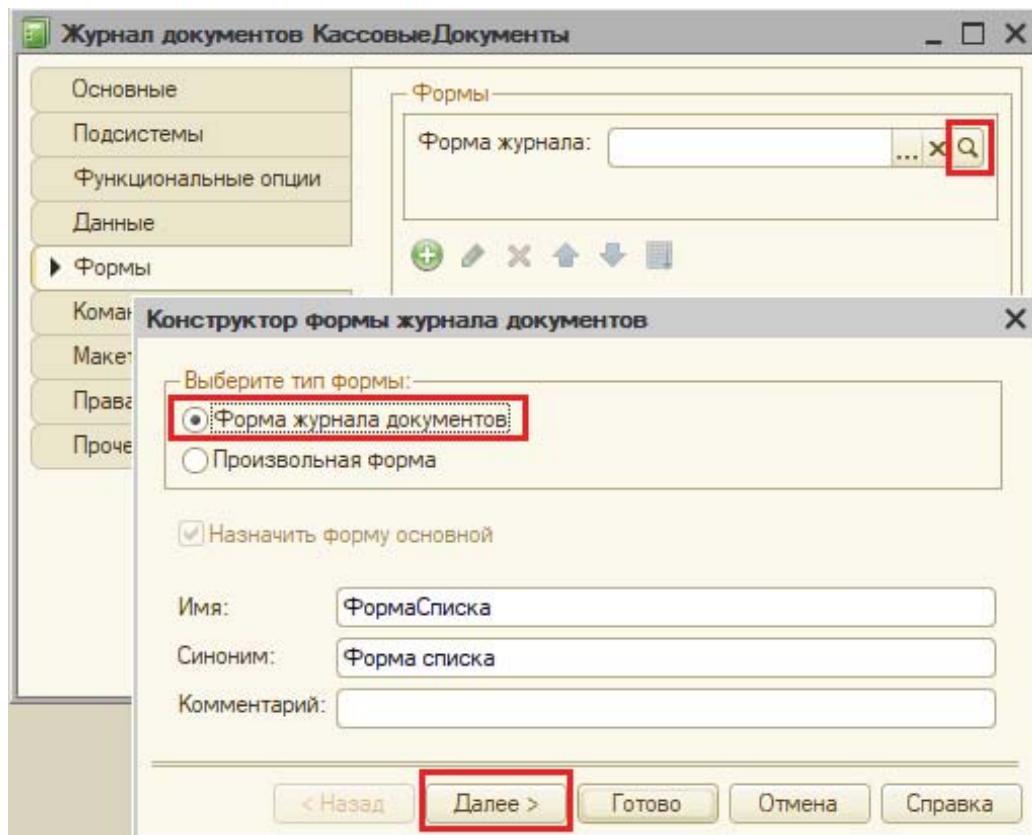
The screenshot shows a table titled 'Касса' (Cashier) with columns: Период (Period), Регистратор (Registrar), Номер строки (Line Number), and Сумма (Amount). The table lists various cash register entries, including income and expenditure orders, with their respective dates and amounts.

Период	Регистратор	Номер строки	Сумма
+ 01.09.2020 10:12:12	Приходный кассовый ордер 00000001 от 01.09.2020 10:12:12	1	1 200,00
+ 01.09.2020 10:12:56	Приходный кассовый ордер 00000002 от 01.09.2020 10:12:56	1	6 000,00
+ 01.09.2020 10:13:33	Приходный кассовый ордер 00000005 от 01.09.2020 10:13:33	1	3 756,00
- 01.09.2020 14:00:00	Расходный кассовый ордер 00000003 от 01.09.2020 14:00:00	1	3 000,00
- 01.09.2020 15:00:00	Расходный кассовый ордер 00000004 от 01.09.2020 15:00:00	1	1 500,00
- 01.09.2020 16:00:00	Расходный кассовый ордер 00000006 от 01.09.2020 16:00:00	1	1 234,00

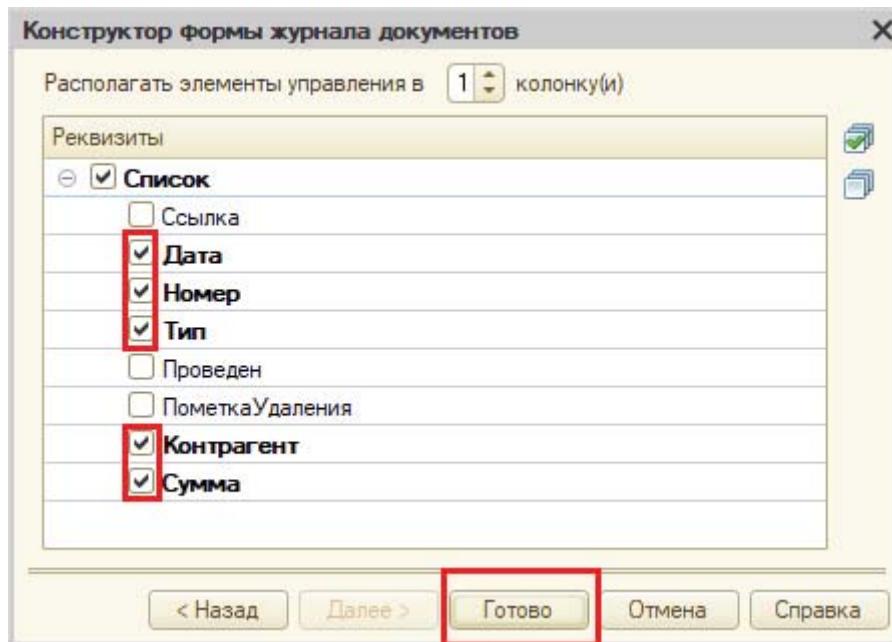
Таким образом, мы можем убедиться, что *регистр накопления* является некой сводной таблицей, в которую попадают данные по некоторым алгоритмам. В дальнейшем из такой таблицы будет проще собирать какие-либо данные, чем открывать каждый документ по отдельности.

Осталось лишь вывести остаток денег в кассе на форму журнала документов.

Откройте вкладку «Формы» окна редактирования журнала документов и добавьте новую форму журнала документов.



Выделите следующие реквизиты для отображения на форме:

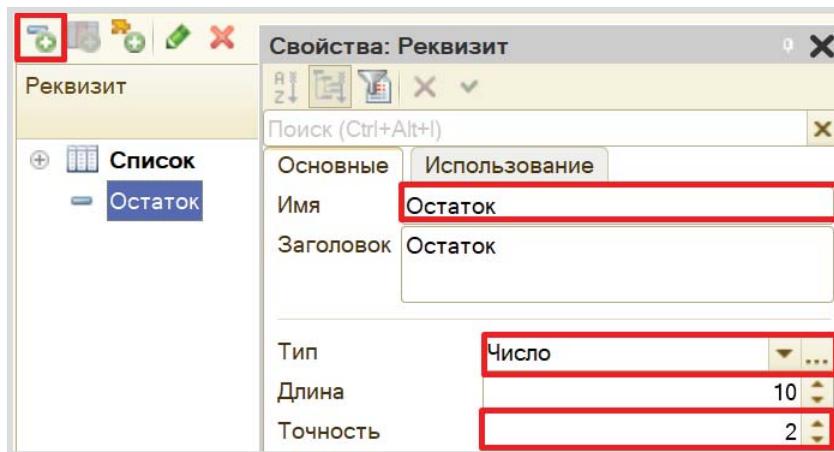


Конструктор формы состоит из трех областей, каждая из которых отвечает за ту или иную функциональность формы:

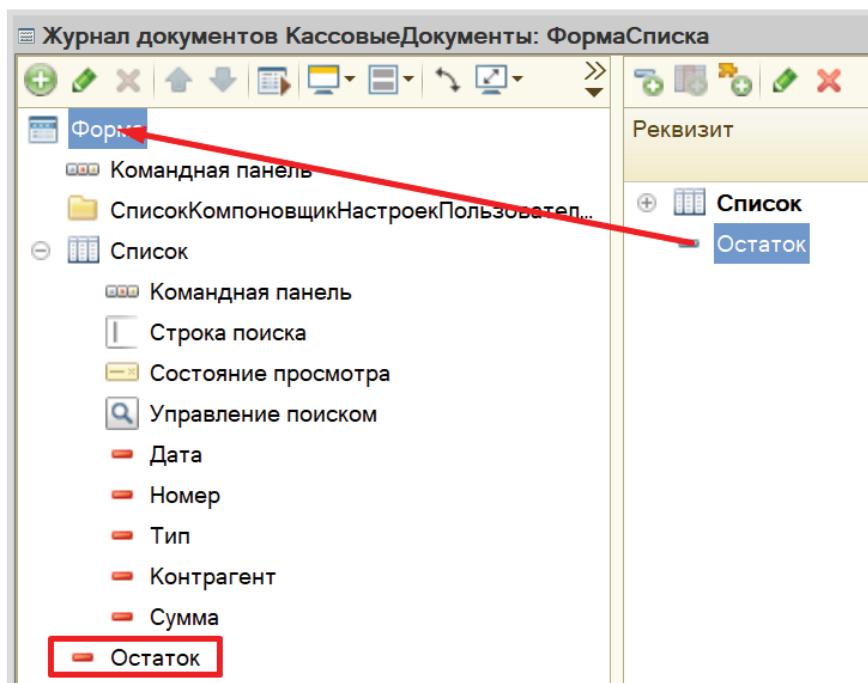
- Снизу находится область предпросмотра формы. Она позволяет лишь приблизительно понять, как будет отображаться данная форма для пользователя, поскольку может быть изменена с учетом множества различных факторов. Это своеобразная «иллюзия» того, что увидит пользователь.
- В правой верхней области находятся данные, которые мы вообще можем использовать в каком-либо виде на этой форме. Они разделены по вкладкам «Реквизиты», «Команды» и «Параметры».
- В левой верхней области *конструктора форм* описывается, какие именно данные будут изображены на форме и в каком именно виде. Здесь – две вкладки: «Элементы» и «Командный интерфейс». На вкладке «Элементы» настраивается внешний вид и расположение реквизитов на форме. Вкладка «Командный интерфейс» определяет положение команд (кнопок) на форме.

Теперь перейдем непосредственно к работе с остатком.

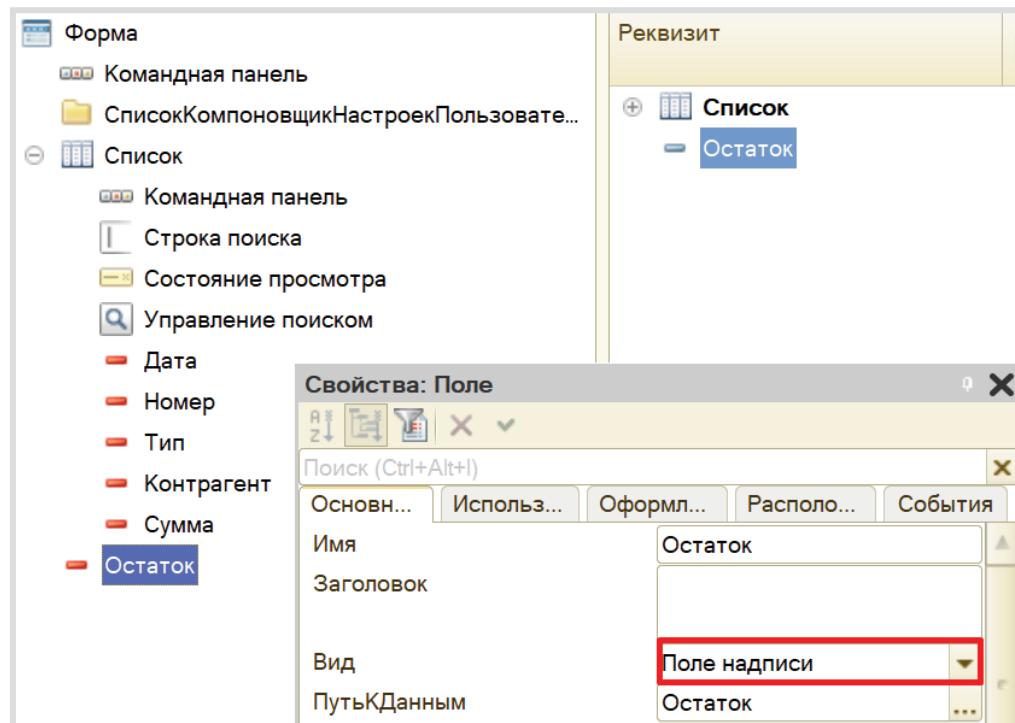
Остаток будем получать из *регистра накопления «Касса»* и выводить его на форму. Чтобы выводить на форму какую-либо информацию, необходимо добавить новый реквизит «Остаток».



Чтобы реквизит был виден на форме, удерживая левую кнопку мыши, перенесите его на элемент «Форма», чтобы он расположился под таблицей.

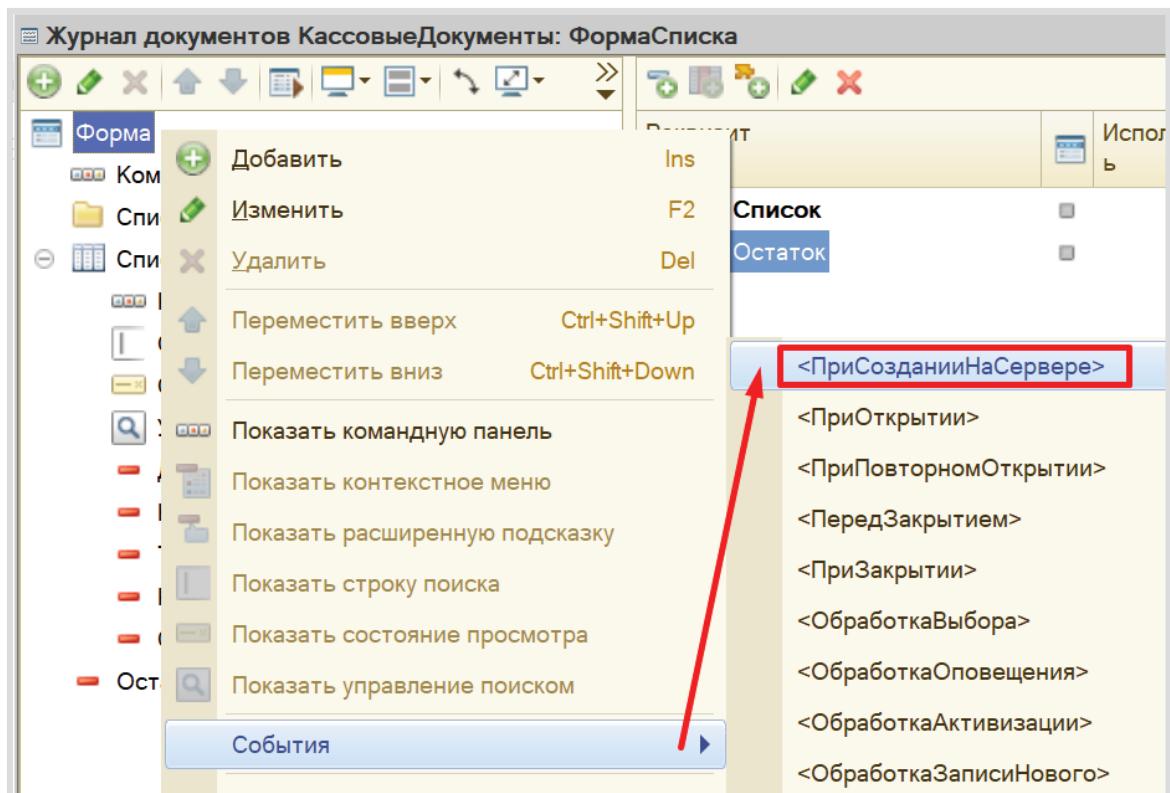


Теперь необходимо изменить вид данного элемента. Открываем его свойства и меняем вид с «Поле ввода» на «Поле надписи».

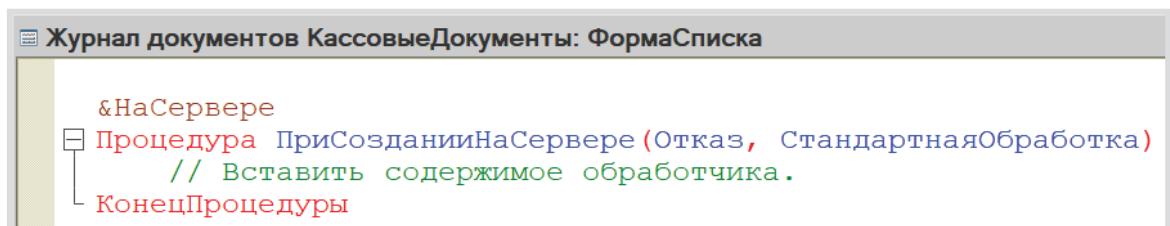


Данное поле не может заполниться само по себе. Оно будет заполняться по происшествии в системе какого-либо события. В нашем случае, необходимо, чтобы остаток денежных средств заполнялся всегда при открытии данной формы.

Создадим такое событие. Щелкнем на элемент «Форма» правой кнопкой мыши и в контекстном меню выберем пункт «События». Среди предложенных событий выберем «ПриСозданииНаСервере».

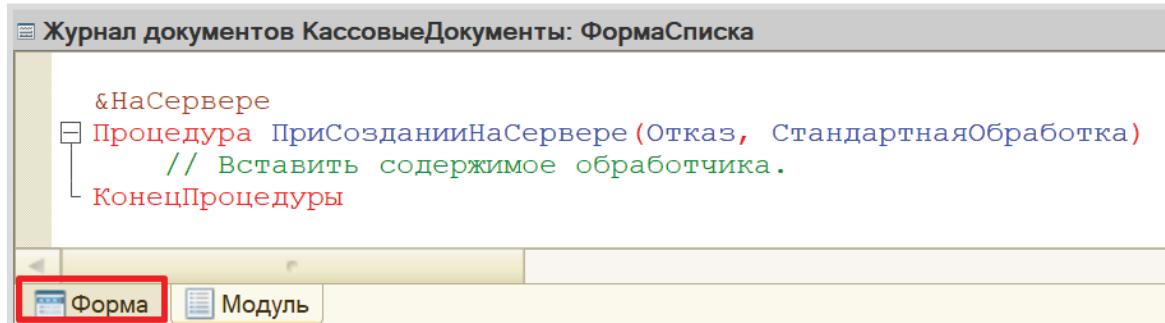


Откроется модуль формы, в котором будет представлен шаблон процедуры.

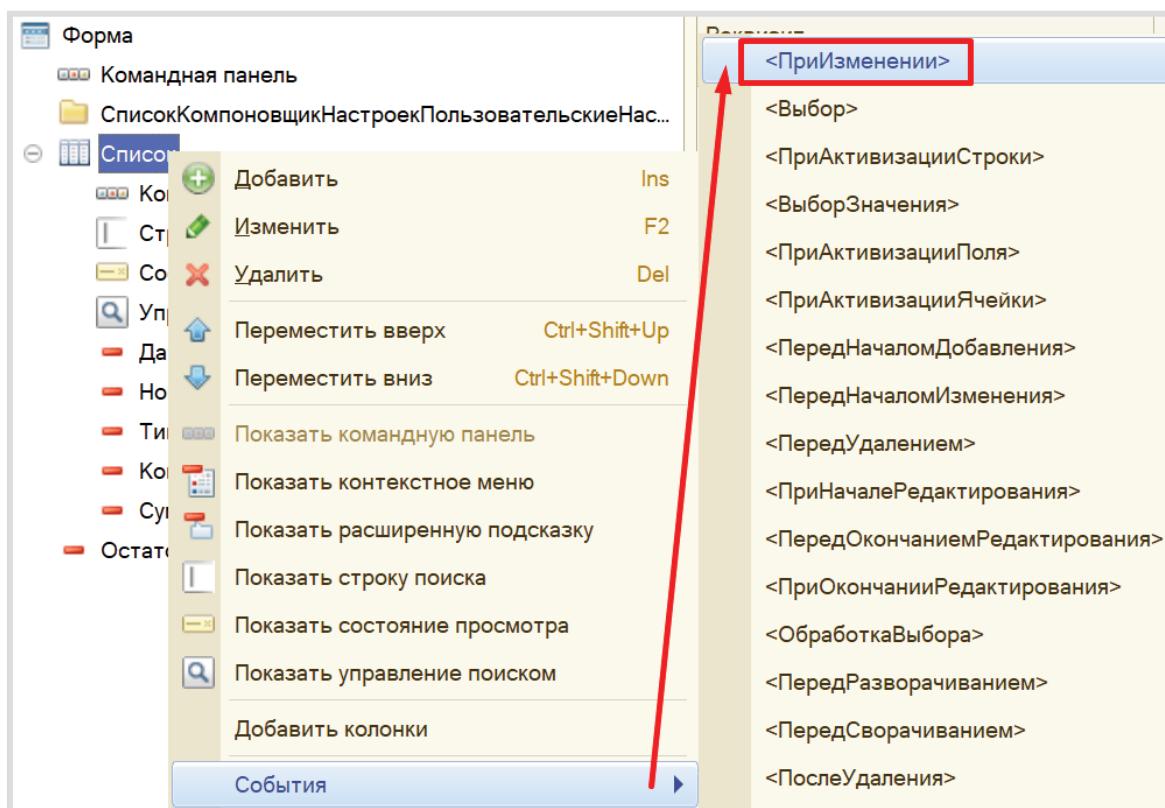


Алгоритм, описанный внутри данной процедуры, выполнится единожды при создании новой формы. Но нам нужно делать пересчет остатка каждый раз, когда мы добавляем новый документ в Журнал документов.

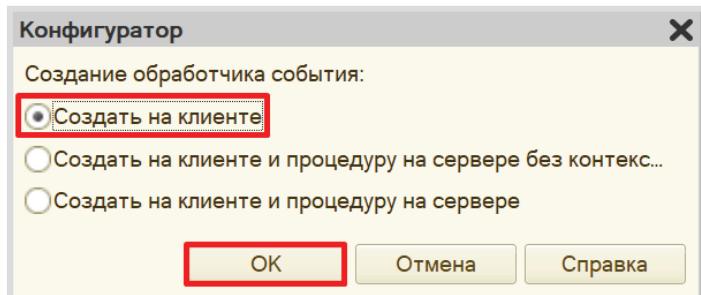
Возвращаемся на форму с помощью вкладки в нижней части *конструктора форм*.



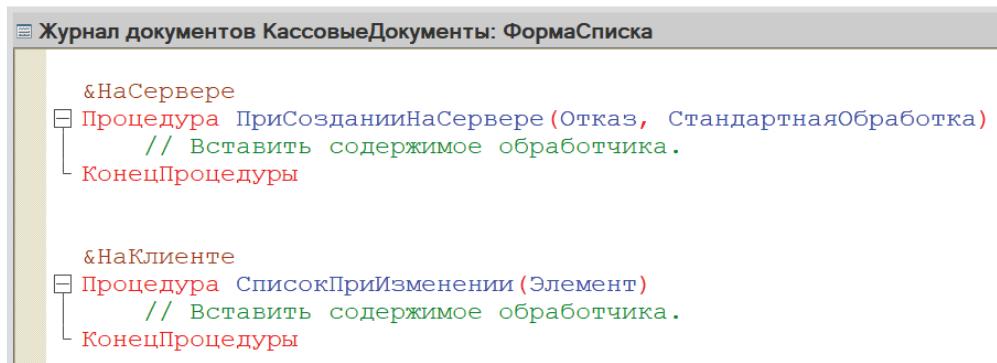
На этот раз следует описать событие не для работы формы в целом, а для ее элемента «Список». Аналогично вызываем контекстное меню, выбираем в меню «События» → «ПриИзменении».



Для решения поставленной задачи обращаться к серверу нет нужды, поэтому выберем обработчик события на стороне клиента.

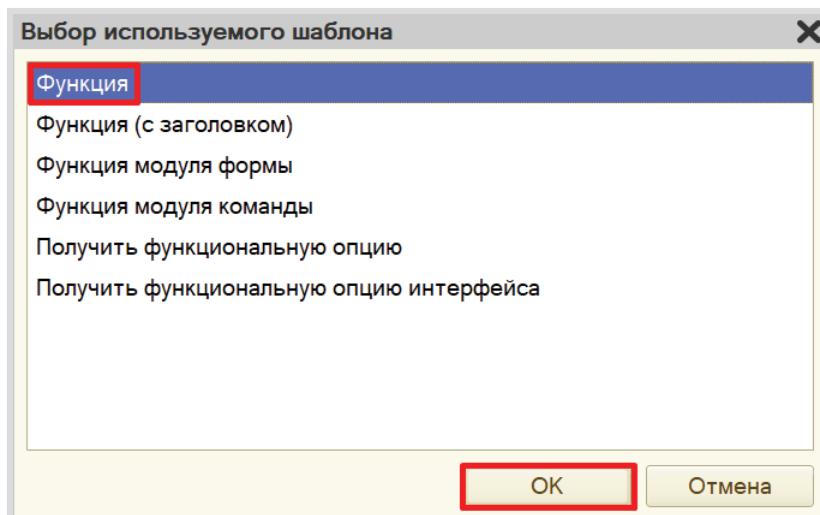


Теперь модуль формы выглядит следующим образом:



Обе эти процедуры будут делать одинаковые действия, а именно – вызывать функцию, которая будет рассчитывать остаток денежных средств в кассе и передавать данные в реквизит «Остаток».

Создадим собственную функцию. Для этого спуститесь в самый низ модуля, ниже описанных процедур, наберите «Функ» и нажмите комбинацию клавиш Ctrl+Q. Система предложим вам на выбор несколько вариантов создания шаблона. Нам нужна самая обычная функция.



На следующем этапе необходимо дать функции имя. Назовем ее «ПолучитьОстаток».



Система сформирует шаблон для написания функции. Далее нужно добавить директиву компиляции, то есть дать функции понять, на чьей стороне она будет обрабатываться – на стороне клиента или сервера. Скопируйте ее у процедуры «ПриСозданииНаСервере» и поместите перед функцией. В результате модуль должен выглядеть следующим образом:

```
Журнал документов КассовыеДокументы: ФормаСписка

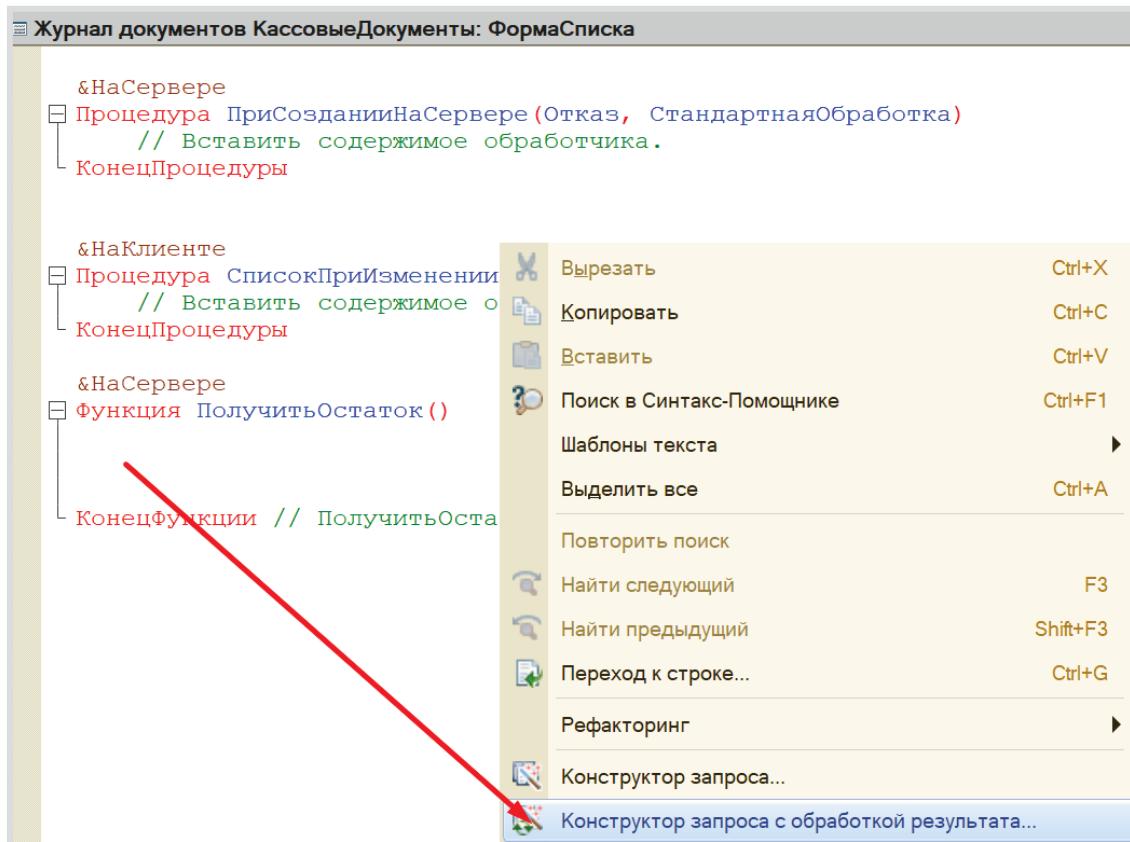
    &НаСервере
    ┌── Процедура ПриСозданииНаСервере (Отказ, СтандартнаяОбработка)
        // Вставить содержимое обработчика.
        КонецПроцедуры

    &НаКлиенте
    ┌── Процедура СписокПриИзменении (Элемент)
        // Вставить содержимое обработчика.
        КонецПроцедуры

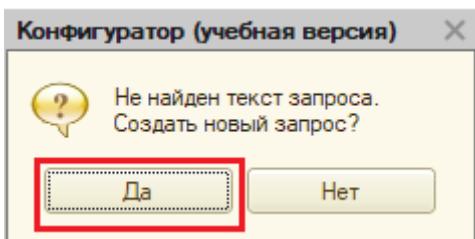
    &НаСервере
    ┌── Функция ПолучитьОстаток ()
        КонецФункции // ПолучитьОстаток ()
```

Для начала следует разобраться с функцией «ПолучитьОстаток». Эта процедура должна обратиться к данным *регистра накопления* и получить из него актуальный остаток в кассе.

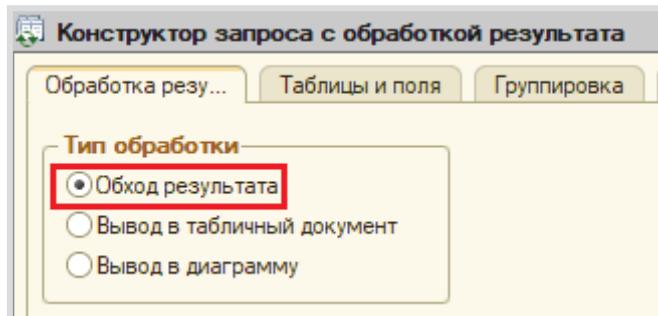
Опишем получение остатков с помощью *конструктора запроса с обработкой результата*. Такой запрос позволит не только получить данные из базы данных, но и обработать их результат. Для этого необходимо установить курсор внутри функции, а затем вызвать контекстное меню правой кнопкой мыши и выбрать пункт «Конструктор запроса с обработкой результата...».



Система выдаст предупреждение о том, что запрос найден не был, и предложит создать свой. Соглашаемся.



На вкладке «Обработка результата» установите тип обработки в значение «Обход результата».

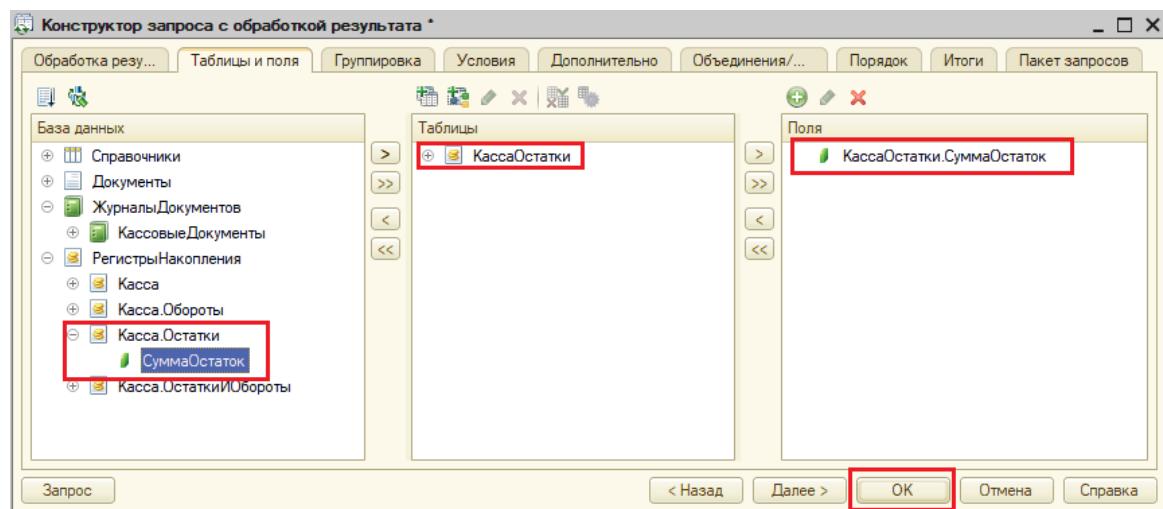


Далее переходим на вкладку «Таблицы и поля». Данное окно имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Необходимо выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать не из *регистра накоплений* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Эта виртуальная таблица позволит получить уже просуммированные значения по всем документам.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.



В результате работы с конструктором получен готовый программный модуль, извлекающий сумму остатка денег в кассе.

```
&НаСервере
Функция ПолучитьОстаток()

    //{{КОНСТРУКТОР_ЗАПРОСА_С_ОВРАБОТКОЙ_РЕЗУЛЬТАТА
//  данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
Запрос = Новый Запрос;
Запрос.Текст =
    "ВЫБРАТЬ
    | КассаОстатки.СуммаОстаток КАК СуммаОстаток
    |ИЗ
    |  РегистрНакопления.Касса.Остатки КАК КассаОстатки";

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
    // Вставить обработку выборки ВыборкаДетальныеЗаписи
КонецЦикла;

//}}КОНСТРУКТОР_ЗАПРОСА_С_ОВРАБОТКОЙ_РЕЗУЛЬТАТА

КонецФункции // ПолучитьОстаток()
```

Запрос будет либо возвращать одно значение, либо ничего не возвращать.

Немного изменим логику, которую создал конструктор, и вместо цикла опишем следующее условие:

```
&НаСервере
Функция ПолучитьОстаток()

    //{{КОНСТРУКТОР_ЗАПРОСА_С_ОВРАБОТКОЙ_РЕЗУЛЬТАТА
//  данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
Запрос = Новый Запрос;
Запрос.Текст =
    "ВЫБРАТЬ
    | КассаОстатки.СуммаОстаток КАК СуммаОстаток
    |ИЗ
    |  РегистрНакопления.Касса.Остатки КАК КассаОстатки";

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда
    Возврат ВыборкаДетальныеЗаписи.СуммаОстаток
Иначе
    Возврат 0
КонецЕсли;

//}}КОНСТРУКТОР_ЗАПРОСА_С_ОВРАБОТКОЙ_РЕЗУЛЬТАТА

КонецФункции // ПолучитьОстаток()
```

Обратите внимание на служебное слово «Возврат». Оно будет возвращать остаток денежных средств в кассе. Если же остаток в кассе будет иметь значение «0», тогда запрос ничего не вернет, а мы с помощью условия заставим функцию вернуть значение «0».

Таким образом, мы сформировали функцию, которая возвращает остаток денег в кассе.

Функция «ПолучитьОстаток» возвращает значение, которое необходимо передать в созданный ранее реквизит «Остаток». Значение остатка должно быть получено в обеих процедурах – «ПриСозданииНаСервере» и «СписокПриИзменении».

```
Журнал документов КассовыеДокументы: ФормаСписка

&НаСервере
└ Процедура ПриСозданииНаСервере (Отказ, СтандартнаяОбработка)
    Остаток = ПолучитьОстаток () ;
    КонецПроцедуры

&НаКлиенте
└ Процедура СписокПриИзменении (Элемент)
    Остаток = ПолучитьОстаток () ;
    КонецПроцедуры
```

Внимание!

Обязательно проверьте модуль на наличие синтаксических ошибок!

Для этого нажмите на кнопку проверки модуля и исправляйте ошибки до тех пор, пока в окне «Служебные сообщения» не появится надпись «Синтаксических ошибок не обнаружено».

Если ошибок нет, то пора приступить к проверке результатов. Запустите конфигурацию в режиме «1С:Предприятие» и откройте *Журнал документов*.

Дата	Номер	Тип документа	Контрагент	Сумма
01.01.2020 12:00:00	000000001	Приходный кассовый ордер	ОАО "Мак"	5 000,00
02.02.2020 15:00:00	000000001	Расходный кассовый ордер	ОАО "Мак"	1 500,00
01.01.2020 13:00:00	000000002	Приходный кассовый ордер	ООО "Василёк"	6 000,00
01.02.2020 14:00:00	000000002	Расходный кассовый ордер	ООО "Василёк"	3 000,00
Остаток:				6 500,00

Осталось лишь решить задачу, связанную с нумерацией.

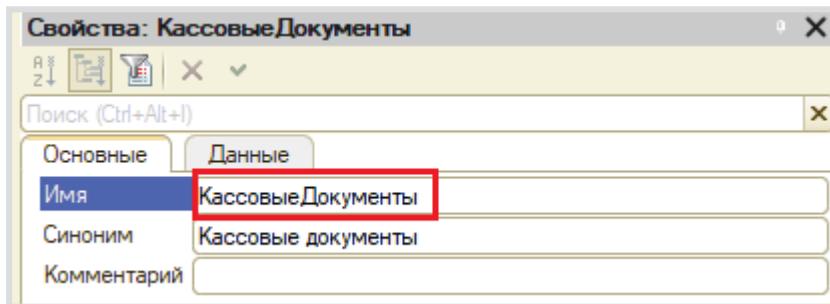
Обратите внимание, что в Журнале документов нумерация считается отдельно для каждого типа документов.

Сделаем нумерацию сквозной, то есть общей для двух видов документов.

Информация

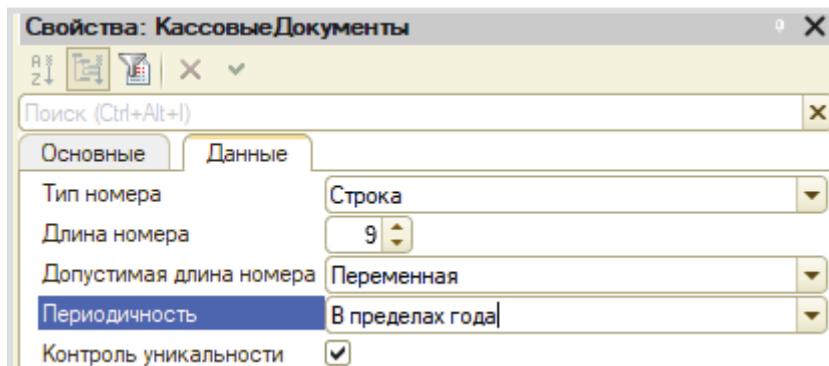
Для того чтобы документы разного вида имели сквозную нумерацию, в системе предусмотрен объект «Нумератор». Прочитать подробнее про нумераторы можно здесь: <https://v8.1c.ru/platforma/numerator/>.

Добавьте нумератор «КассовыеДокументы».



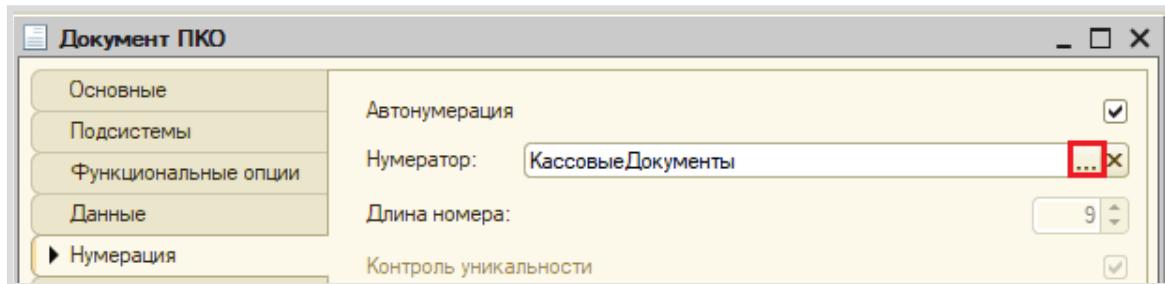
Сделаем так, чтобы нумерация документов сбрасывалась каждый год: таким образом, пользователь не сможет создать большее количество документов, чем мы предусмотрели.

Перейдем на вкладку «Данные» и настроим нумератор следующим образом:



Теперь нужно присвоить нумератор всем видам документов, которые должны иметь сквозную нумерацию. В нашем случае – это документы «ПКО» и «РКО».

Откройте окно редактирования документа «ПКО» на вкладке «Нумерация». В качестве нумератора выберите созданный нумератор «КассовыеДокументы».



Аналогично добавьте нумератор в документ «РКО».

Откройте систему в режиме «1С:Предприятие» и проверьте правильность выполнения работы. Для этого придется удалить созданные ранее документы и создать новые в различном порядке. Следите за полем остатка!

Обратите внимание, что система не контролирует отрицательные остатки!

Дата	Номер ↓	Тип документа	Контрагент	Сумма
01.01.2020 12:00:00	000000001	Приходный кассовый ордер	ОАО "Мак"	1 200,00
02.01.2020 12:00:00	000000002	Расходный кассовый ордер	ООО "Василёк"	3 000,00
03.01.2020 0:00:00	000000003	Расходный кассовый ордер	ОАО "Мак"	1 002,35
04.01.2020 0:00:00	000000004	Приходный кассовый ордер	ООО "Василёк"	10 000,00

Остаток: 7 197,65

Поставленная задача решена.

Лабораторная работа № 7

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ, РЕГИСТРИРУЮЩЕЙ ИЗМЕНЕНИЕ КУРСОВ ВАЛЮТ

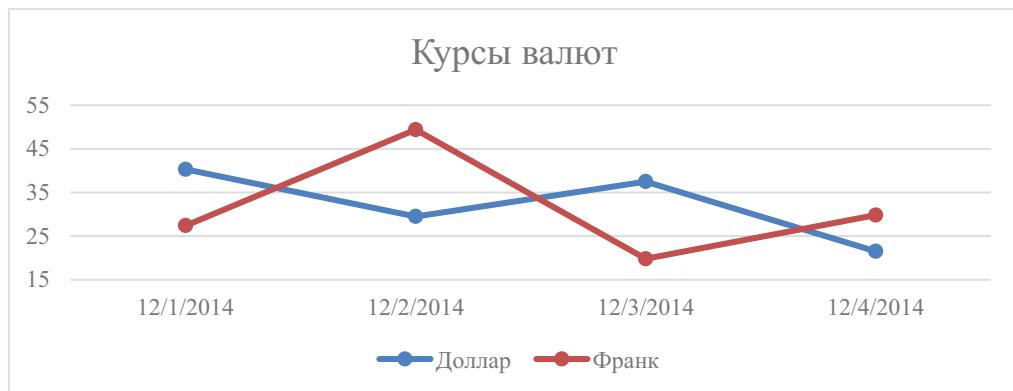
Сложность: *

Теги: справочник, регистр сведений, схема компоновки данных, дополнения

ЗАДАНИЕ

Заказчик просит разработать информационную систему, регистрирующую изменение курсов валют.

В результате выполнения лабораторной работы должен получиться график курса валют*:



* 2014-й год на графике использован умышленно, так как в этом году курсы валют изменились с наибольшей амплитудой.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

Какие именно валюты необходимы заказчику для работы с информационной системой?

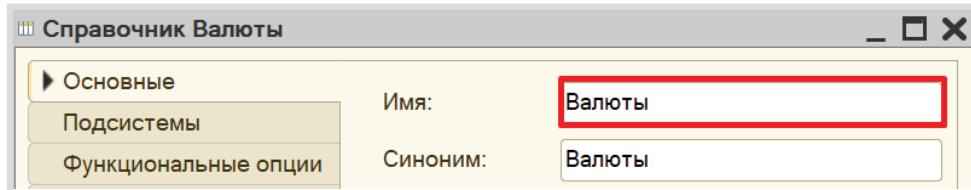
С точки зрения разработки нас эта информация не интересует, мы должны лишь создать место в информационной системе для хранения информации обо всех интересующих заказчика валютах. Для этого нужно воспользоваться объектом конфигурации *справочник*.

Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Элементы справочника могут быть добавлены пользователем или разработчиком. Данный справочник будет содержать объекты аналитического учета, то есть список валют.

Создадим справочник «Валюты».

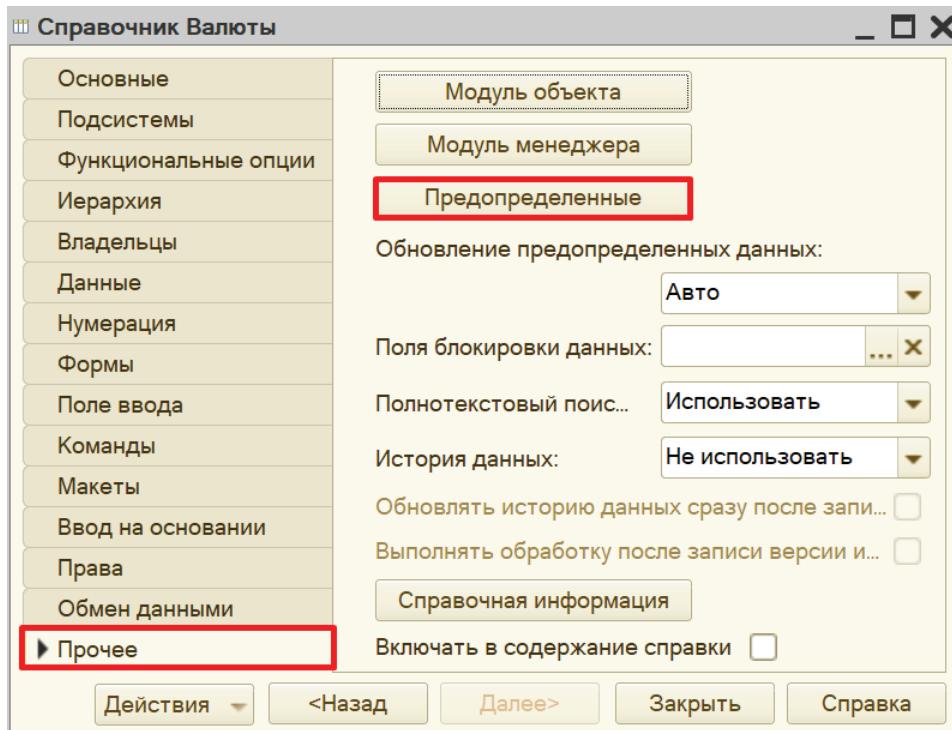


В окне редактирования справочника на вкладке «Прочие» можно создать предопределенные элементы.

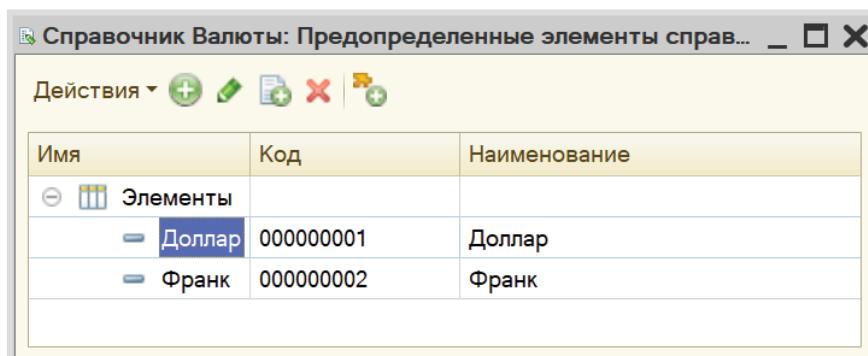
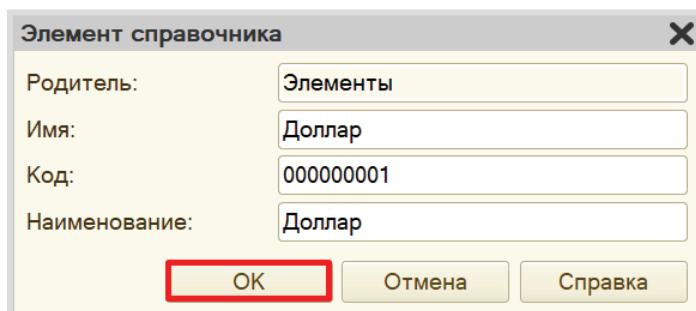
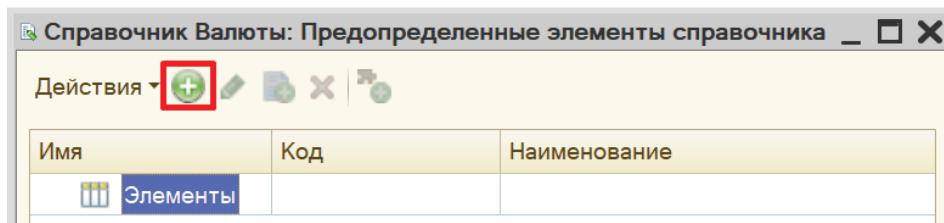
Определение

Предопределенные элементы – это такие элементы, которые создает разработчик в конфигураторе для удобства работы пользователя.

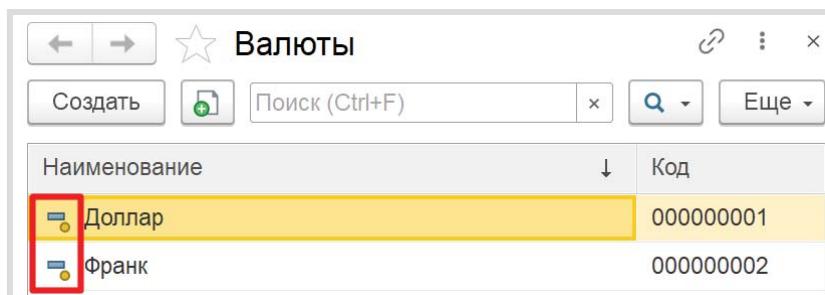
Созданный таким образом элемент будет доступен пользователю с первого запуска программы. Например, можно создать предопределенный элемент «Россия» в справочнике «Страны мира».



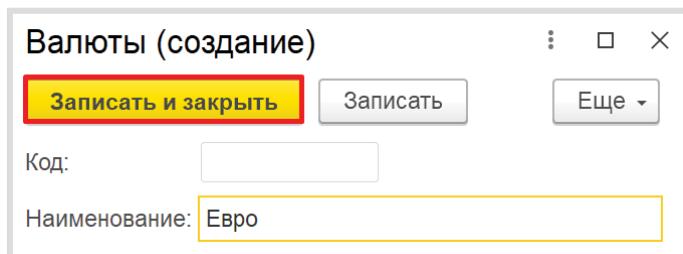
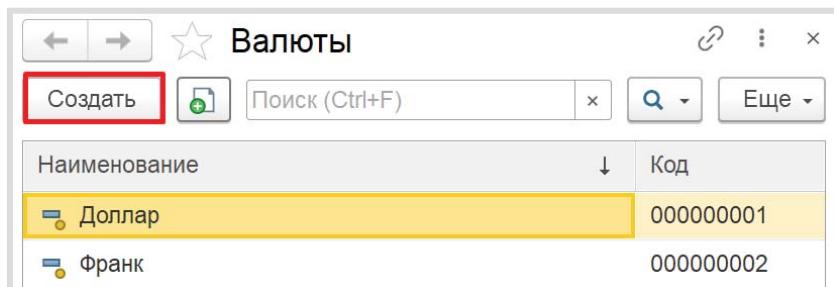
Создадим два новых предопределенных элемента: «Доллар» и «Франк».



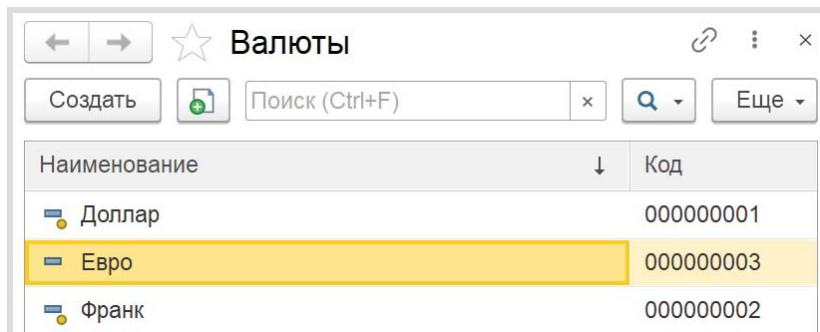
Запустим режим пользователя и убедимся, что предопределенные элементы справочника были созданы. Они обозначаются пиктограммой с желтым кружочком.



В случае необходимости пользователь может добавить новую валюту в справочник.



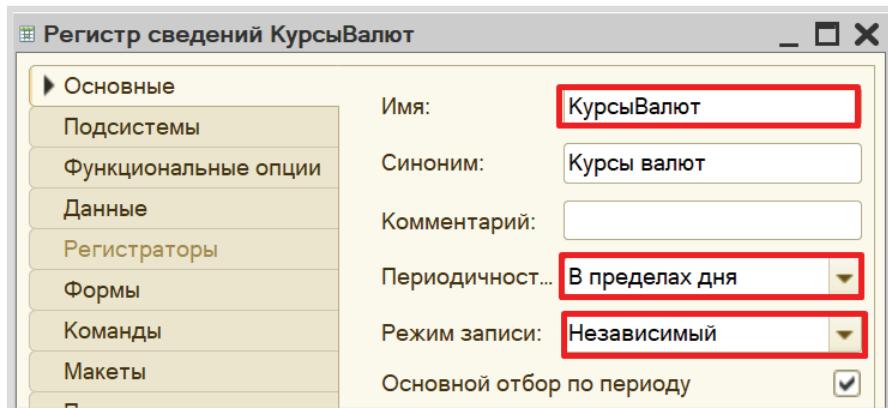
Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



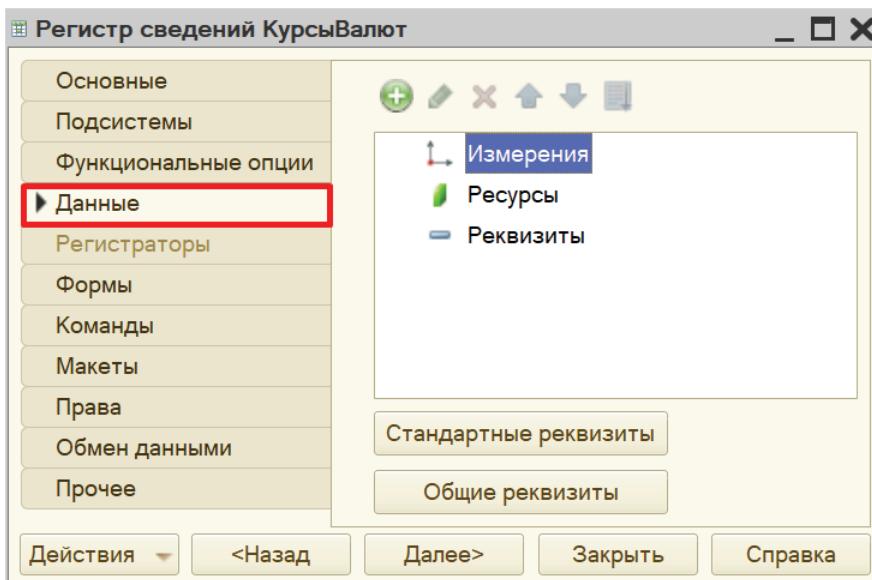
Значение курса валют меняется ежедневно. Для отслеживания этой динамики будем использовать механизм *регистра сведений*.

Регистр сведений позволяет сохранять информацию об изменении каких-либо показателей с течением времени (подробнее о *registraх сведений* можно прочитать здесь: <https://v8.1c.ru/platforma/registr-svedeniy/>).

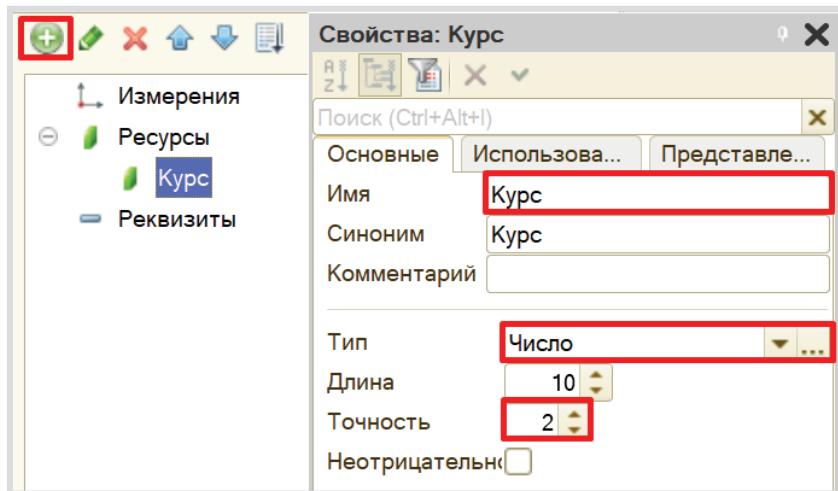
Добавим новый *регистр сведений* «КурсыВалют». Периодичность регистра установим в значение «В пределах дня». Это значит, что добавлять данные в регистр можно будет только один раз в день. Регистр будет независимым – это означает, что в него записи могут быть добавлены напрямую, без специального документа-регистратора.



Переходим к описанию структуры *регистра накопления*. Для этого откроем вкладку «Данные».



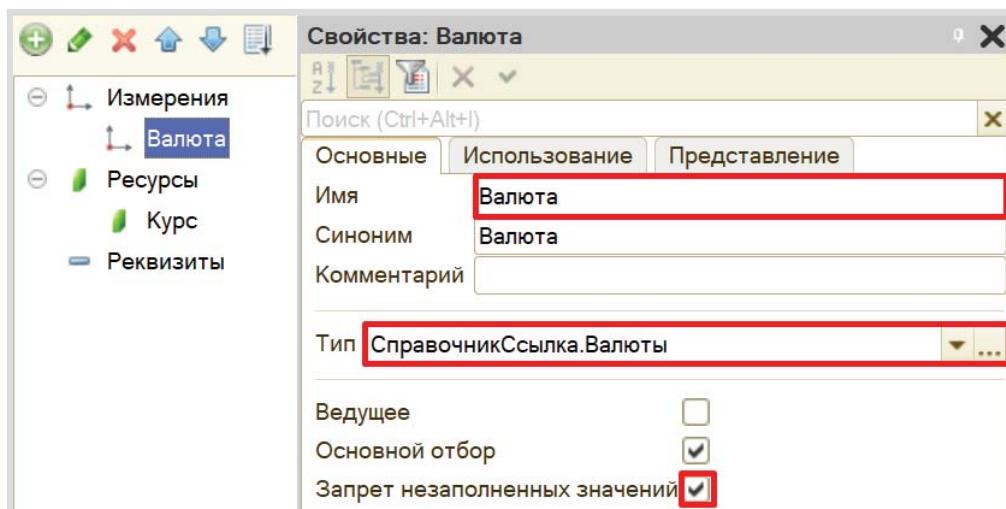
Заполнение данного окна всегда проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, следует задать вопрос: «Что нам нужно хранить в данном регистре?». Нам нужно хранить данные о курсах (валют). Следовательно, курс и будет являться ресурсом. Тип данного реквизита – «Число».



Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим учитывать курс? Мы хотим хранить данные о курсах (чего?) валют. Значит, в качестве измерения следует добавить реквизит «Валюта». Тип данного реквизита – «СправочникСсылка.Валюты». В этом поле будут храниться ссылки на элементы справочника «Валюты».

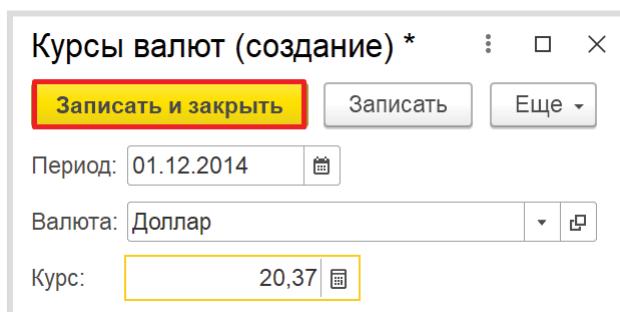
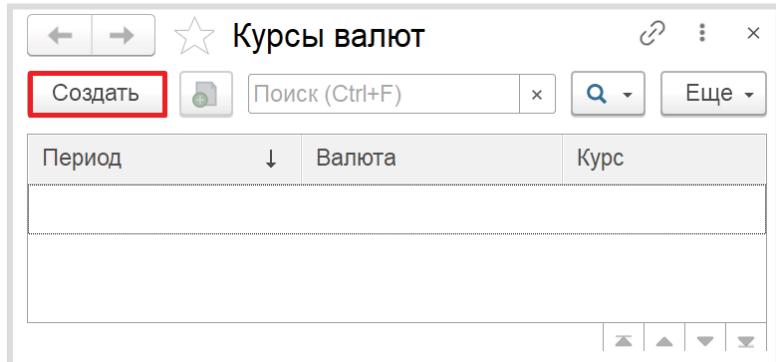
Внимание!

Обязательно поставьте галочку у свойства «Запрет незаполненных значений» – так пользователь не сможет сделать запись в регистр сведений без указания валюты.



Механизм хранения данных реализован.

Откроем программу в режиме «1С:Предприятия» и курсы валют за несколько дней.



Период	Валюта	Курс
01.12.2014	Доллар	20,37
01.12.2014	Франк	29,05
03.12.2014	Доллар	21,68
03.12.2014	Франк	24,82
05.12.2014	Доллар	23,52
05.12.2014	Франк	17,01
07.12.2014	Доллар	18,07
07.12.2014	Франк	29,49
09.12.2014	Доллар	25,17
09.12.2014	Франк	17,47

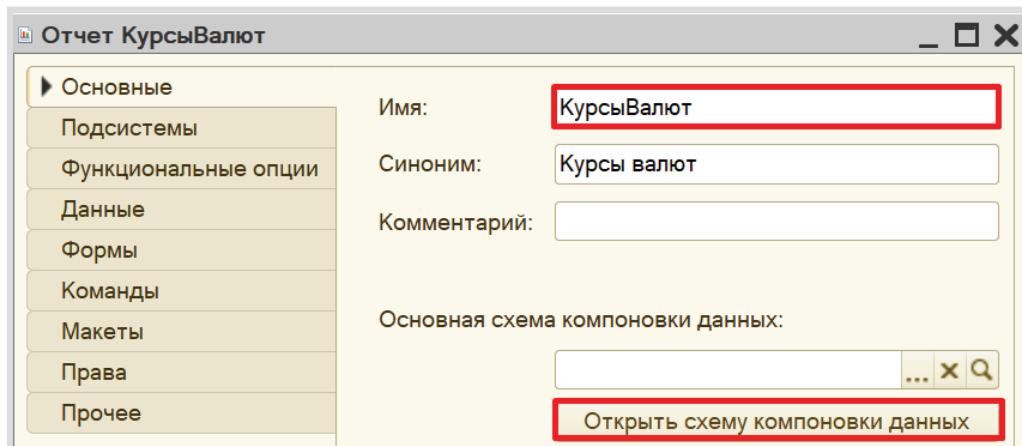
Таким образом, мы создали возможность хранения динамики курсов различных валют по дням.

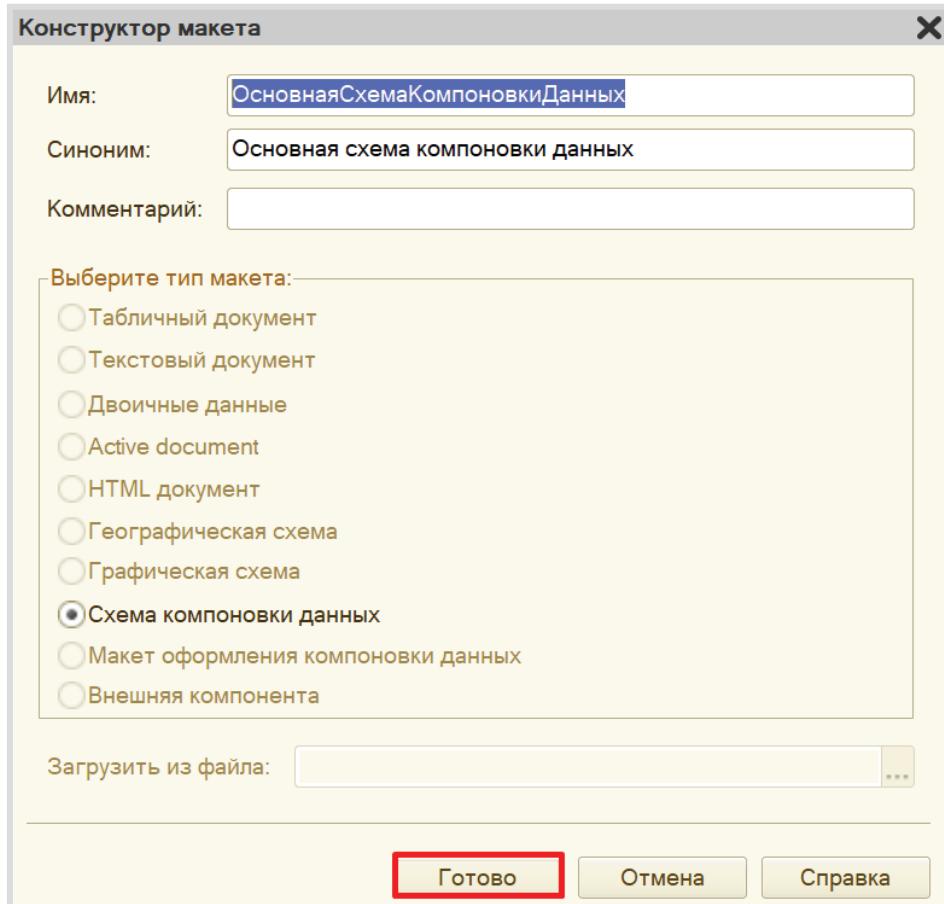
Чтобы отобразить эту динамику, следует использовать объект конфигурации *отчет*.

Определение

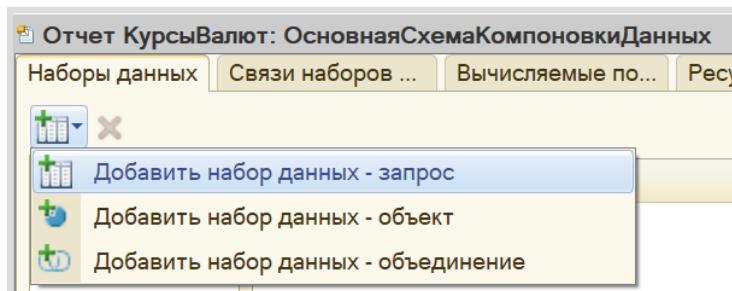
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Создадим новый отчет «КурсыВалют». Для наполнения отчета воспользуемся конструктором схемы компоновки данных.

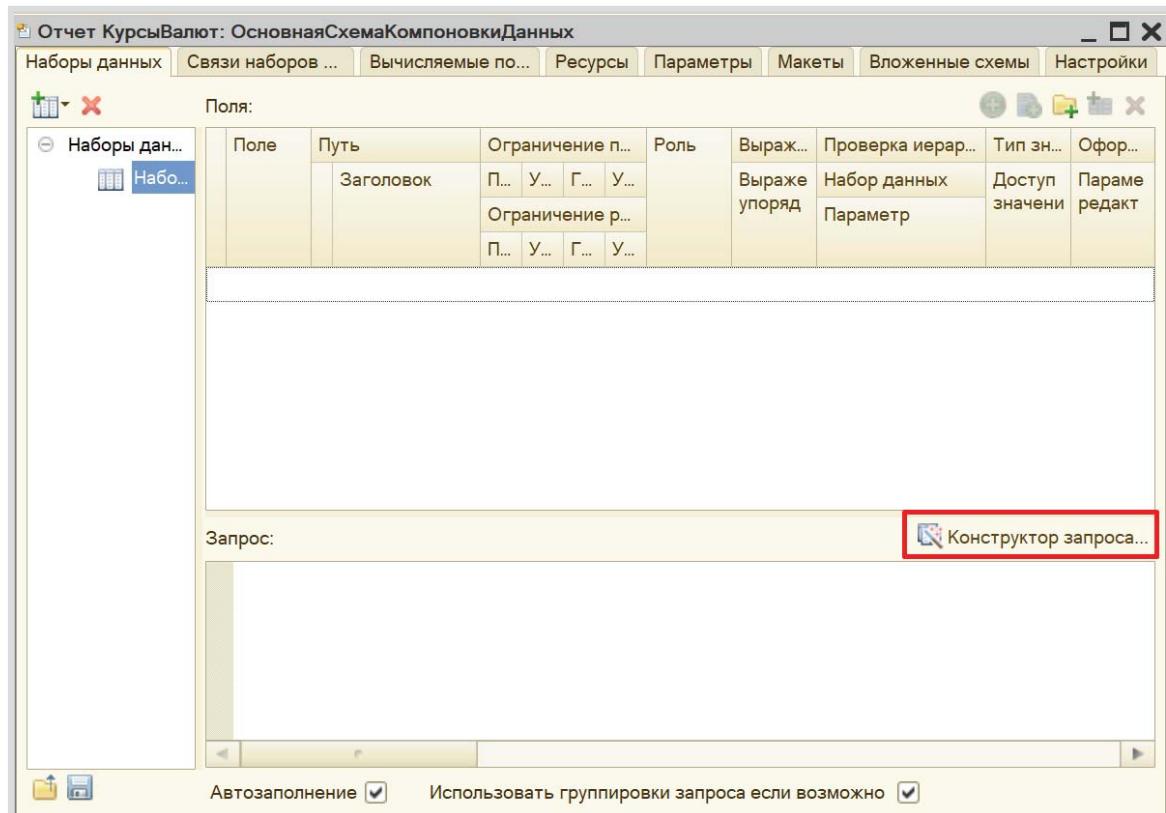




Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Воспользуемся конструктором запроса.



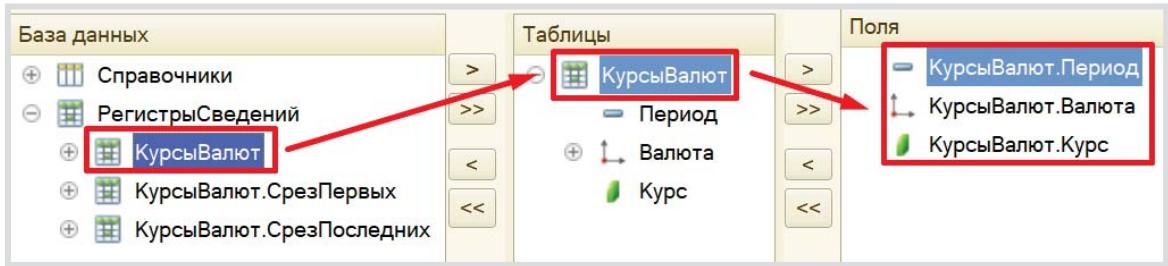
Открывшееся окошко имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных;
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета;
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

В качестве источника будем использовать таблицу *регистра сведений «КурсыВалют»*, которую только что самостоятельно заполнили в режиме пользователя.

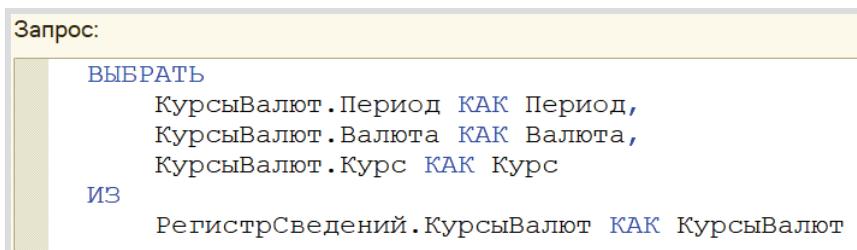
Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:

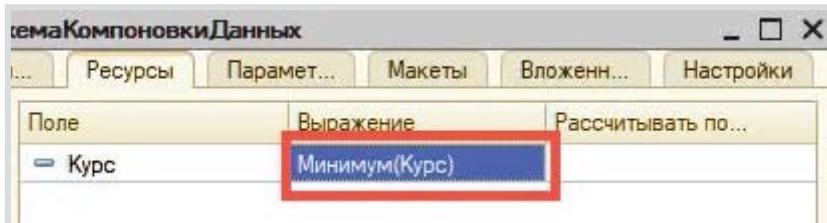


По завершении работы с *конструктором запроса* нажмите кнопку «OK».

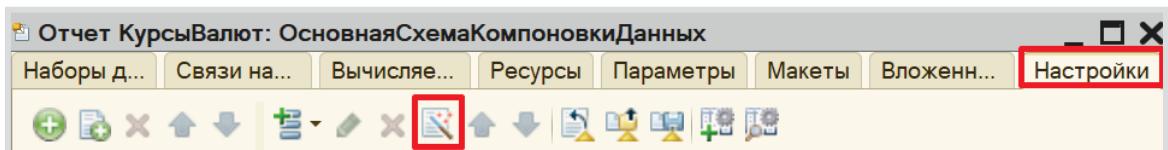
В результате должен получиться запрос к базе данных:



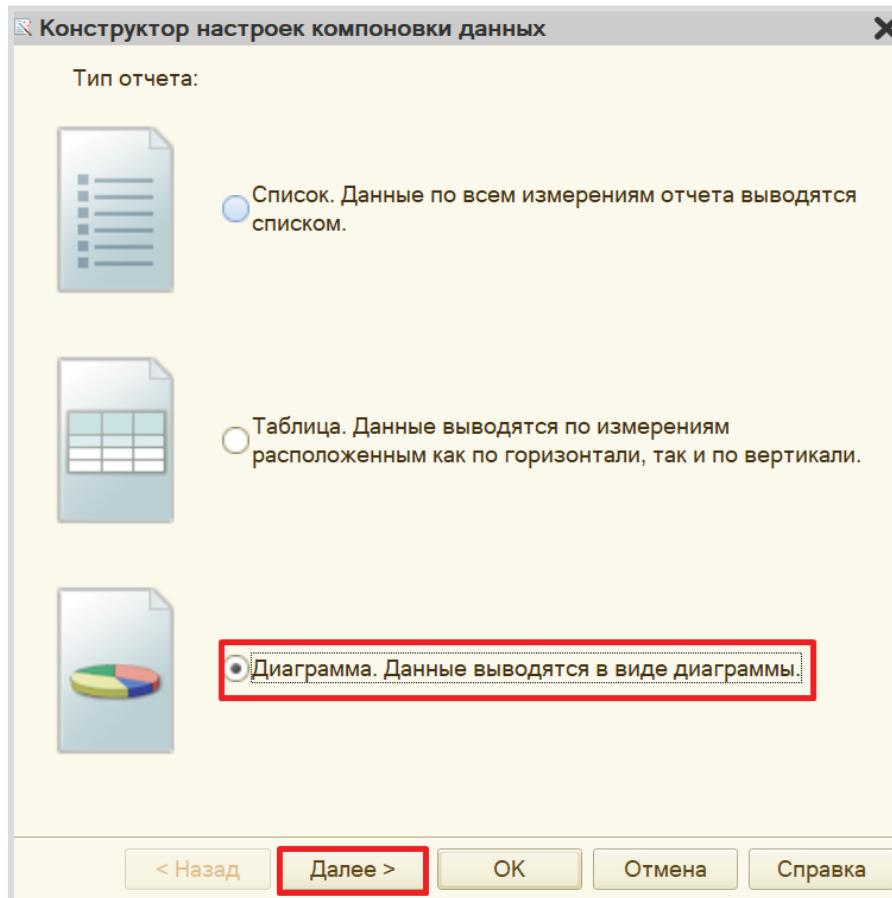
На вкладке «Ресурсы» определим реквизит «Курс» в качестве ресурса. Вместо выражения *по умолчанию* установим значение с функцией МИНИМУМ «Минимум(Курс)». Это нужно для корректной работы диаграммы.



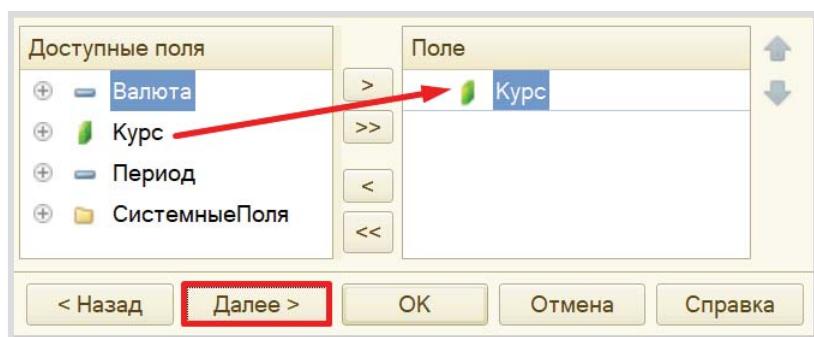
Теперь переходим к настройке внешнего вида отчета. Воспользуемся *конструктором настроек отчета*.



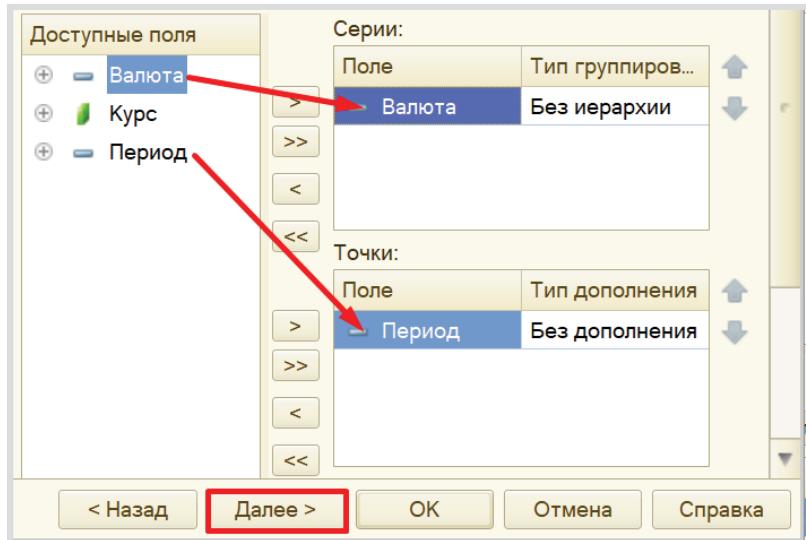
С помощью конструктора настроек отчета выберем вариант отчета в виде диаграммы.



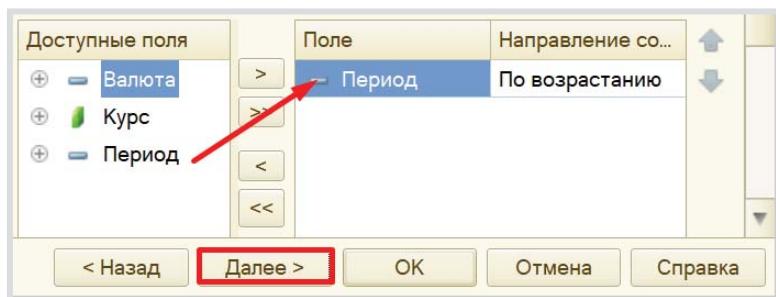
Выберем ресурсы, которые будут отображаться в диаграмме. Ресурс у нас всего один – «Курс».



Переходим к следующему этапу настройки диаграммы. Точки и серии – это оси X и Y графика соответственно. Пусть ось Y отображает стоимость валюты на конкретную дату. Даты же пусть будут расположены по оси X.



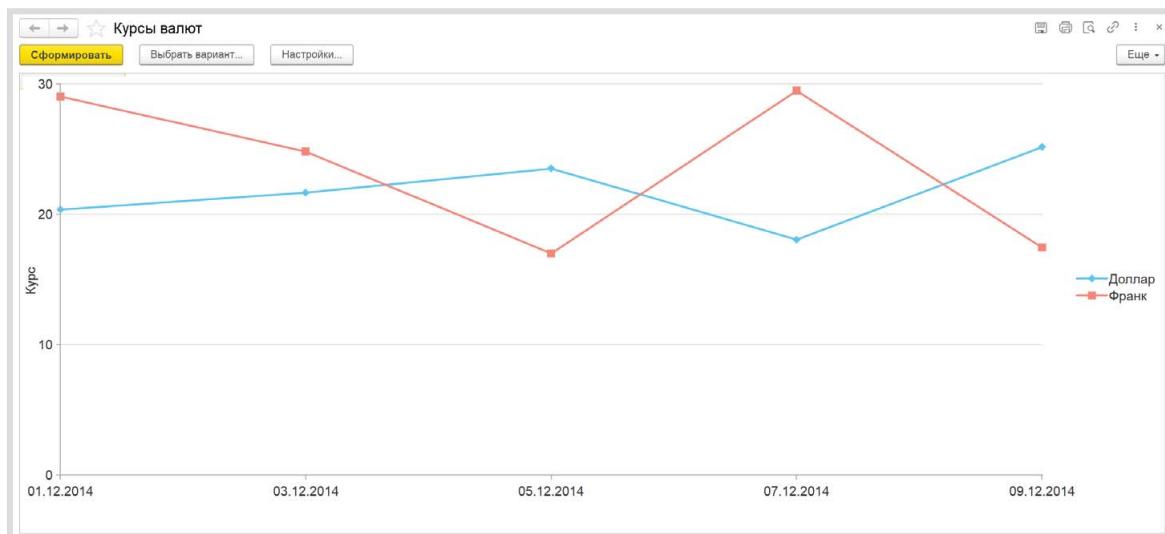
Также нужно отметить поля для упорядочивания. Упорядочим график по периоду, чтобы он строился по возрастанию даты.



На последнем шаге выбираем тип диаграммы – *график*.

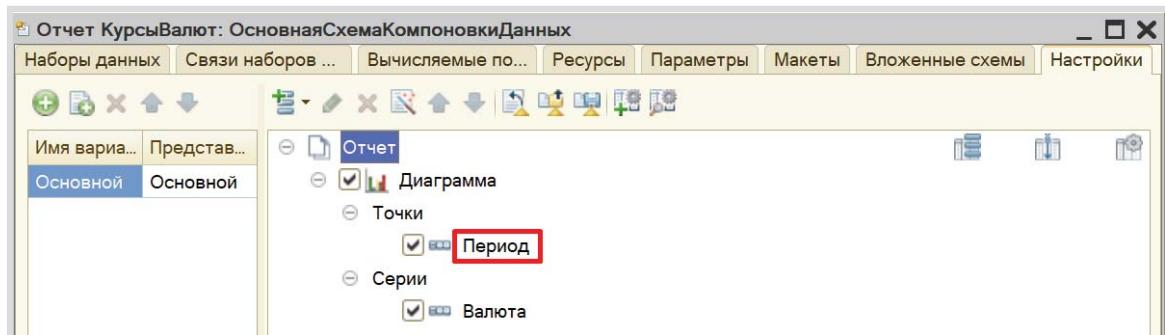


Посмотрим на то, как выглядит график в режиме «1С:Предприятие».

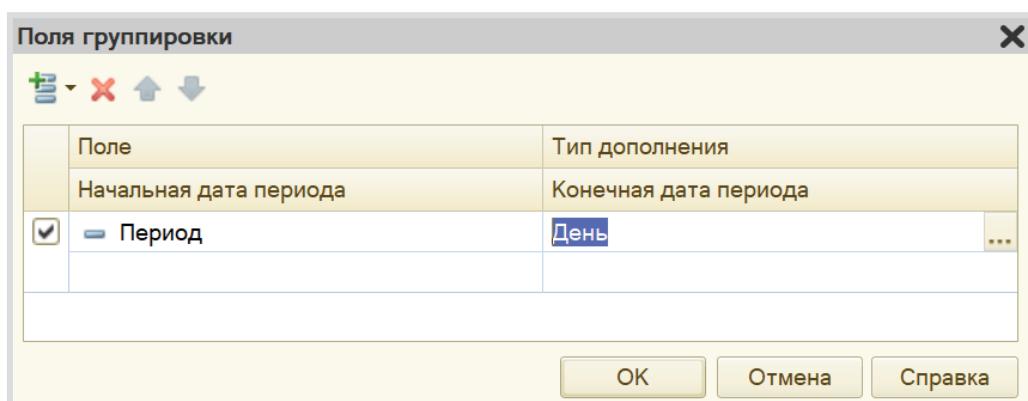


Обратите внимание, что данные курсов валют были введены через день, то есть данные на 2, 4, 6 и 8 декабря в регистре отсутствуют. График просто соединяет две близлежащие точки, но он не отображает ежедневную картину изменения курсов валют. Разумеется, мы можем добавить отсутствующие данные в регистр сведений и построить отчет заново. Но мы сделаем иначе и позволим отчету в пропущенных датах выставлять некоторое среднее значение курса для каждой валюты.

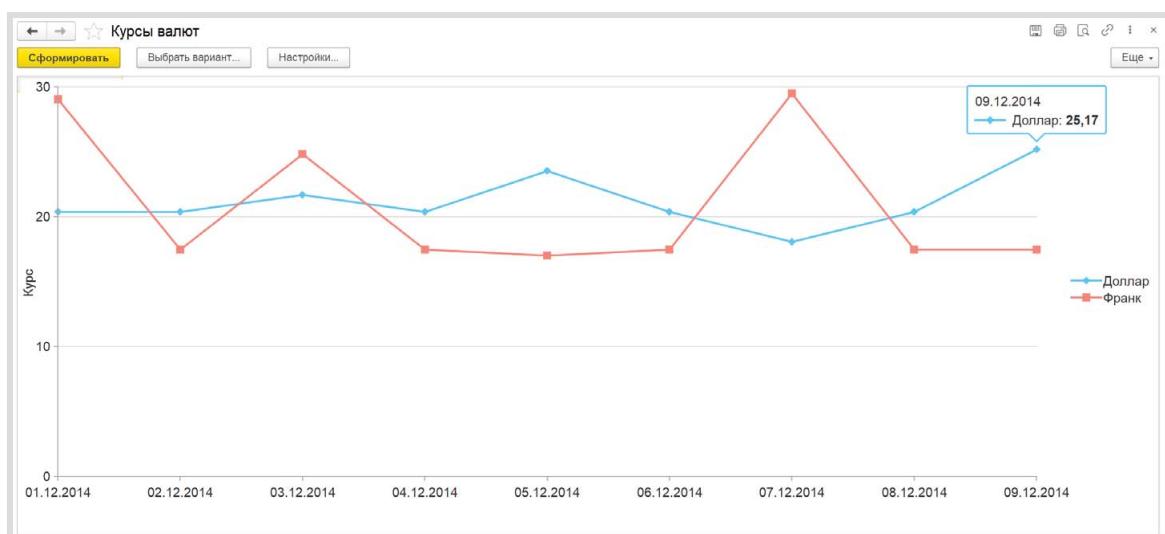
Возвращаемся в конфигуратор, к настройкам отчета «КурсыВалют». Двойным щелчком мыши откроем настройки точек по периоду.



Добавим дополнение, то есть расчет промежуточных, не вошедших в запрос, точек. В качестве варианта заполнения укажем «День».



Получившийся отчет выглядит следующим образом:



Поставленная задача решена.

Лабораторная работа № 8

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ, РЕГИСТРИРУЮЩЕЙ ИЗМЕНЕНИЕ ЦЕН КУПЛИ И ПРОДАЖИ ВАЛЮТ

Сложность: *

Теги: справочник, регистр сведений, схема
компоновки данных

ЗАДАНИЕ

Заказчик просит разработать информационную систему, регистрирующую изменение цен купли и продажи валют. Нужно составить отчет, в котором будет формироваться график изменения цен купли-продажи различных валют.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать, смотрите в Лабораторной работе № 2 (стр. 17).

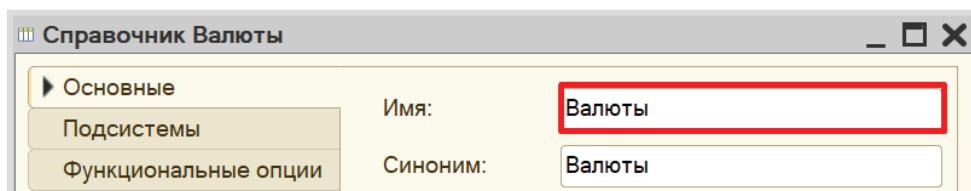
Выполнение

Согласно условию, нам необходимо где-то хранить информацию о валютах. Для этого будем использовать объект конфигурации *справочник*.

Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>). Данный справочник будет содержать объекты аналитического учета, то есть список валют.

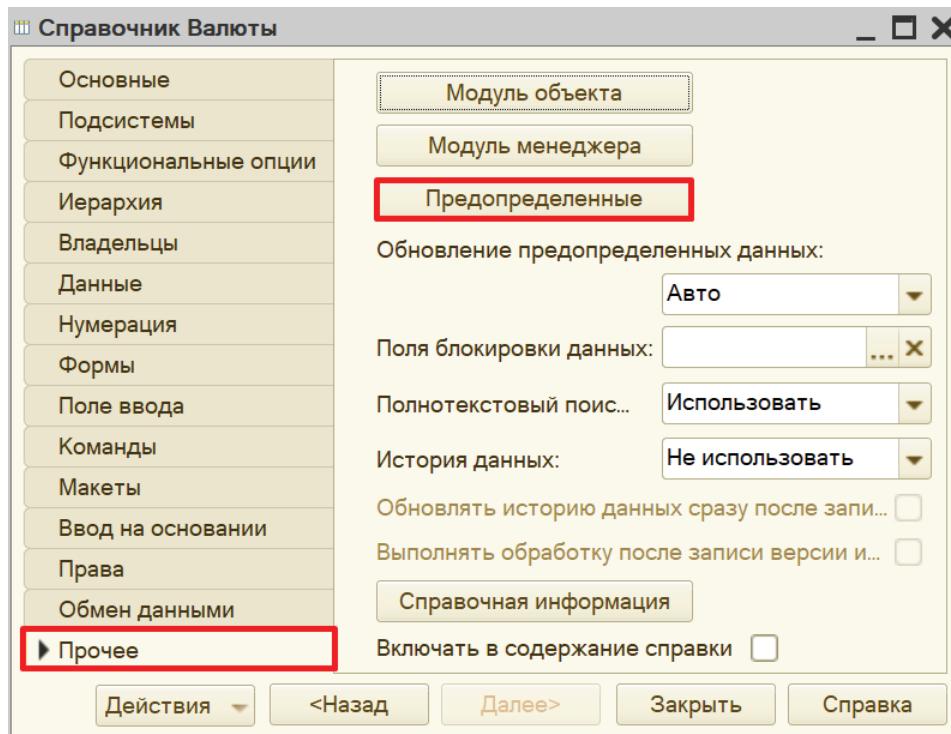
Создадим справочник «Валюты».



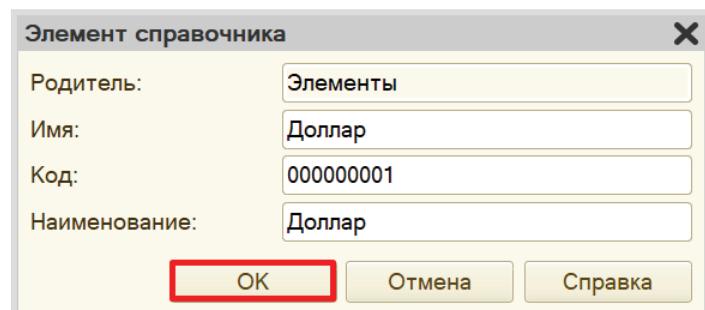
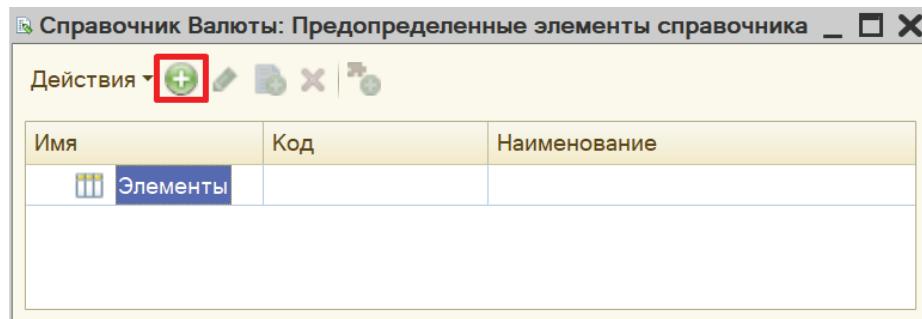
В окне редактирования справочника на вкладке «Прочие» можно создать предопределенные элементы.

Определение

Предопределенные элементы – это такие элементы, которые создает разработчик в конфигураторе для удобства работы пользователя. Созданный таким образом элемент будет доступен пользователю с первого запуска программы. Например, можно создать предопределенный элемент «Россия» в справочнике «Страны мира».



Создадим два новых предопределенных элемента: «Доллар» и «Франк».



Справочник Валюты: Предопределенные элементы справочника		
Действия		
Имя	Код	Наименование
Элементы		
Доллар	000000001	Доллар
Франк	000000002	Франк

Любая валюта на рынке может быть куплена или продана. То есть вид сделки подразумевает выбор из этих двух вариантов.

Для решения задачи хранения информации, которая представляет собой фиксированный набор альтернатив, нам понадобится новый объект, который называется *перечисление* (подробнее про перечисления можно прочитать здесь: <https://v8.1c.ru/platforma/perechisleniya/>).

Добавим новое перечисление «ВидСделки».

Перечисление ВидСделки

Основные	Имя: ВидСделки
Подсистемы	Синоним: Вид сделки
Функциональные опции	

Значения перечисления (заготовленный список выбора) заполним на вкладке «Данные».

Значения перечисления:

+ Добавить	Свойства: Покупка
	Поиск (Ctrl+Alt+I)
Значения	Основные
Покупка	Имя: Покупка
	Синоним: Покупка
	Комментарий

Значения перечисления:

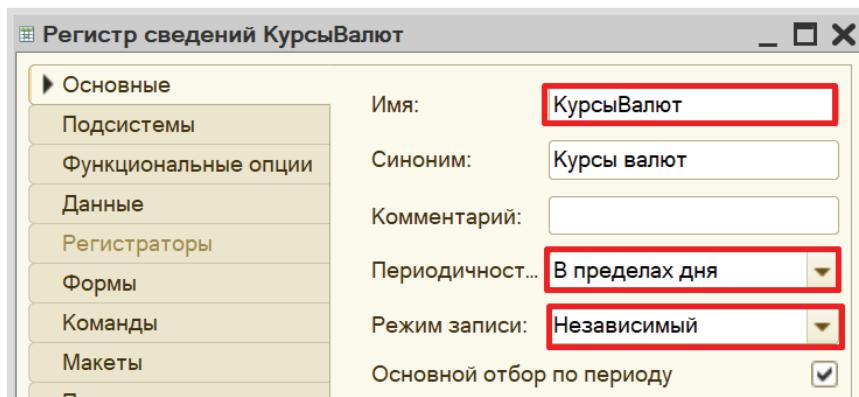
+ Добавить	Свойства: Продажа
	Поиск (Ctrl+Alt+I)
Значения	Основные
Покупка	Имя: Продажа
Продажа	Синоним: Продажа
	Комментарий

Для хранения информации о курсах валют мы будем использовать механизм *регистра сведений*.

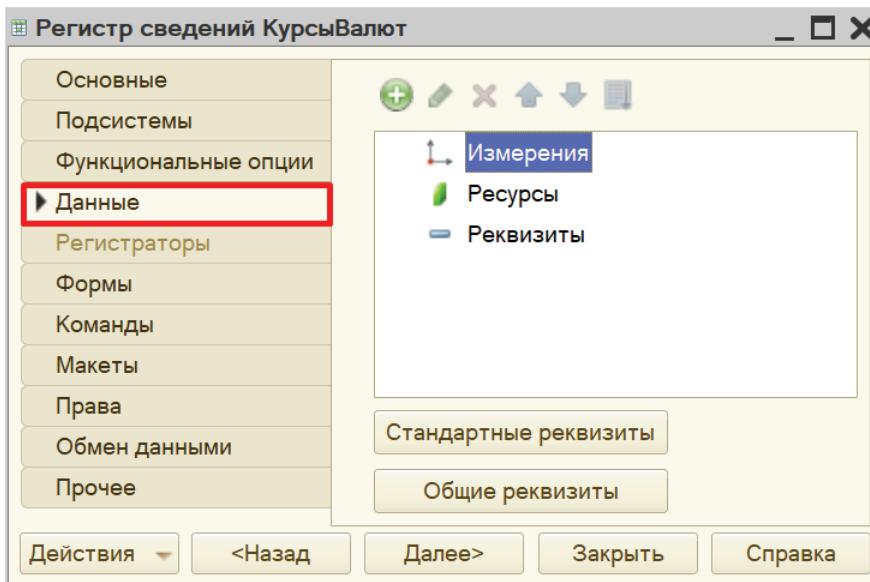
Регистр сведений позволяет сохранять информацию об изменении каких-либо показателей с течением времени (подробнее о *registraх сведений* можно прочитать здесь: <https://v8.1c.ru/platforma/registr-svedeniy/>).

В нашем случае, курсы валют меняются каждый день, следовательно, *регистр сведений* предназначен для хранения значений курсов валют по дням.

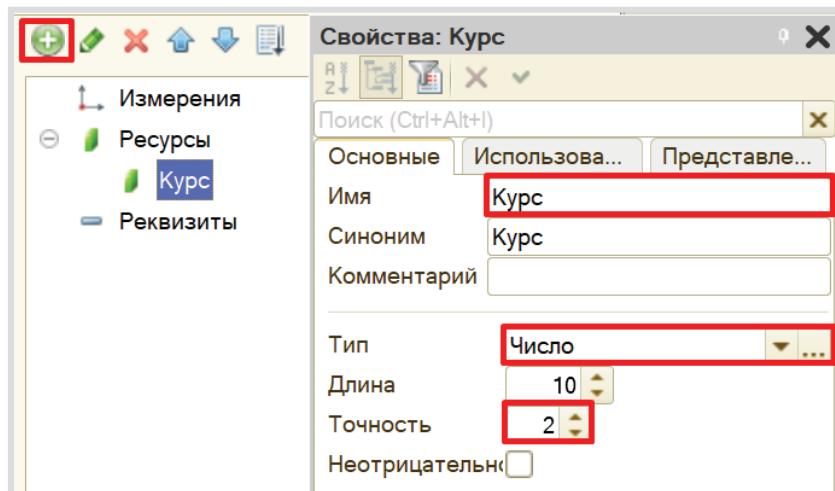
Добавим новый *регистр сведений* «КурсыВалют». Периодичность регистра установим в значение «В пределах дня». Это значит, что добавлять данные в регистр можно будет только один раз в день. Регистр будет независимым – это означает, что в него записи могут быть добавлены напрямую, без специального документа-регистратора.



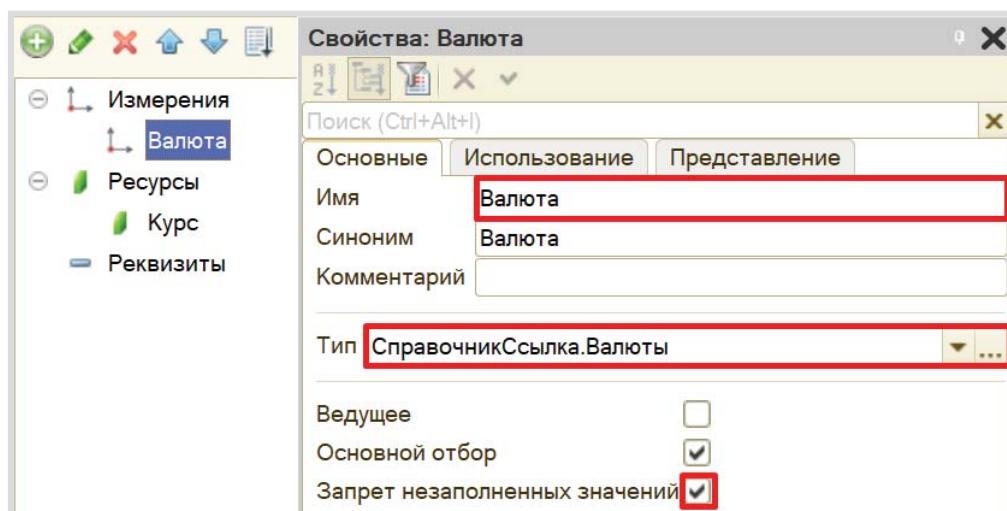
Переходим к описанию структуры *регистра накопления*. Для этого откроем вкладку «Данные».



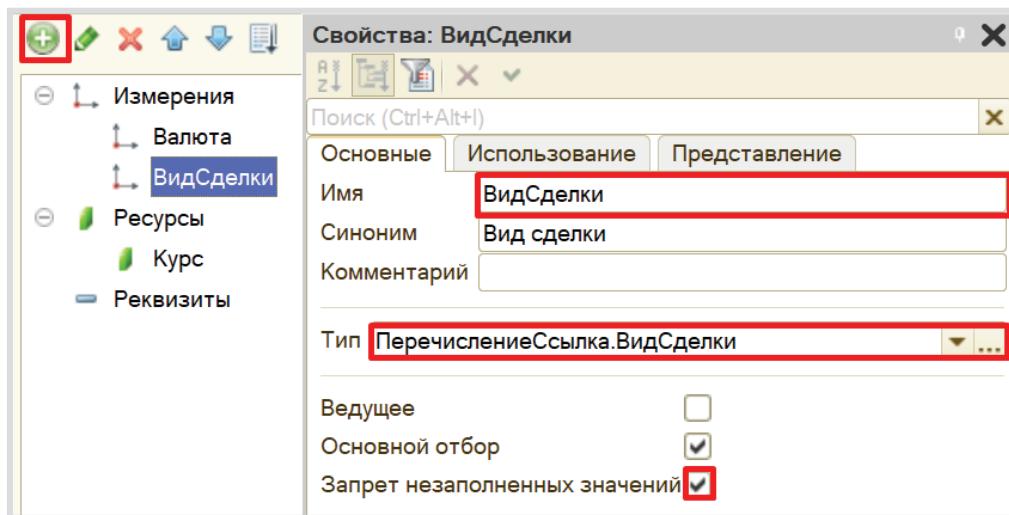
Заполнение данного окна всегда проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, следует задать вопрос: «Что нам нужно хранить в данном регистре?». Нам нужно хранить данные о курсах. Следовательно, курс и будет являться ресурсом. Тип данного реквизита – «Число».



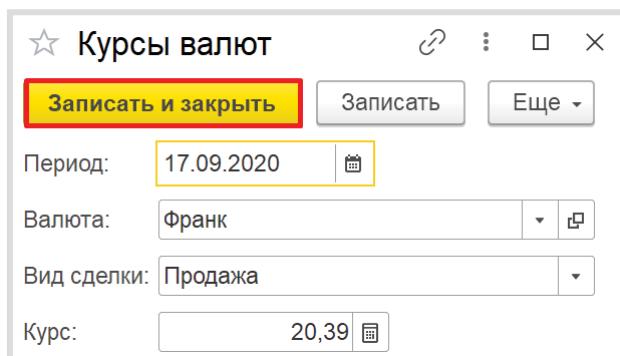
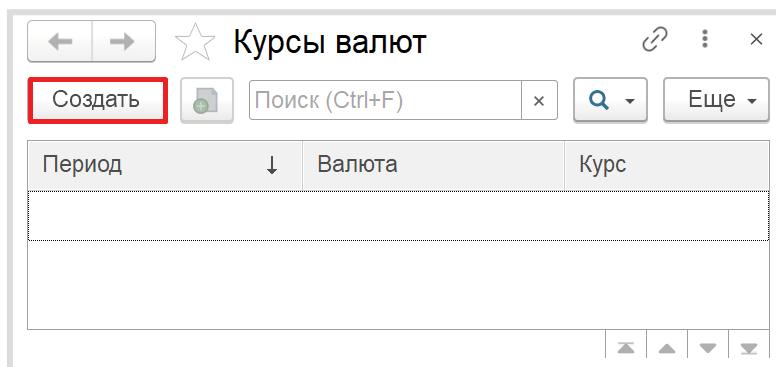
Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим учитывать курс? Мы хотим хранить курсы (чего?) валют. Значит, в качестве измерения необходимо добавить реквизит «Валюта». Тип данного реквизита – «СправочникСсылка.Валюты». В этом поле будут храниться ссылки на элементы справочника «Валюты».



Если бы нам нужно было просто хранить курсы валют, то мы оставили бы структуру именно такой. Но наша цель – хранить курс продажи или покупки валюты. Поэтому нужно добавить еще одно измерение – «ВидСделки».



Механизм хранения данных реализован. Откроем программу в режиме «1С:Предприятия» и введем несколько курсов валют.



Введем данные за несколько дней о стоимости покупки и продажи валюты.

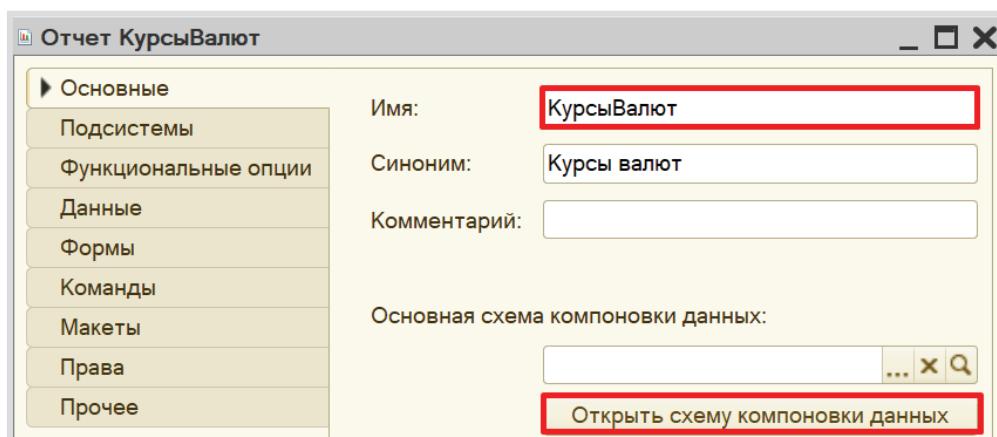
Период	↓	Валюта	Вид сделки	Курс
15.09.2020		Франк	Покупка	8,26
15.09.2020		Франк	Продажа	4,37
16.09.2020		Доллар	Покупка	6,26
16.09.2020		Доллар	Продажа	10,26
16.09.2020		Франк	Покупка	15,54
16.09.2020		Франк	Продажа	13,34
17.09.2020		Доллар	Покупка	17,33
17.09.2020		Доллар	Продажа	12,76
17.09.2020		Франк	Покупка	23,63
17.09.2020		Франк	Продажа	20,39

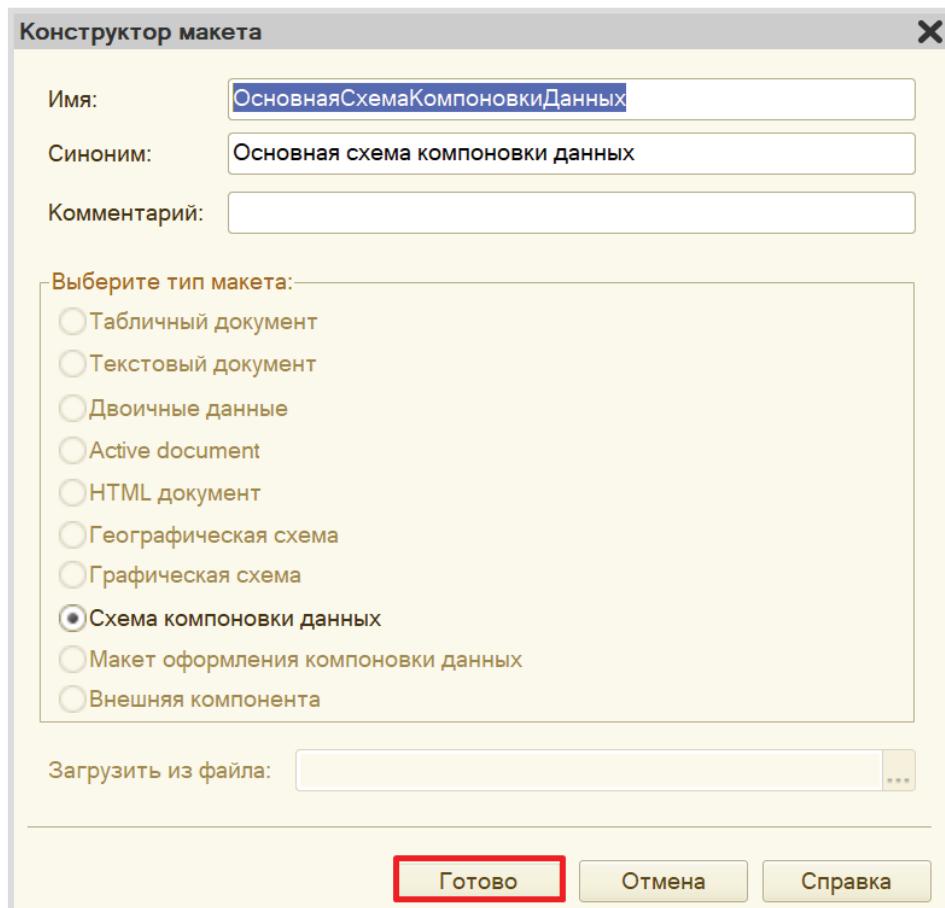
Чтобы отобразить изменение курсов с течением времени необходимо использовать объект конфигурации *отчет*.

Определение

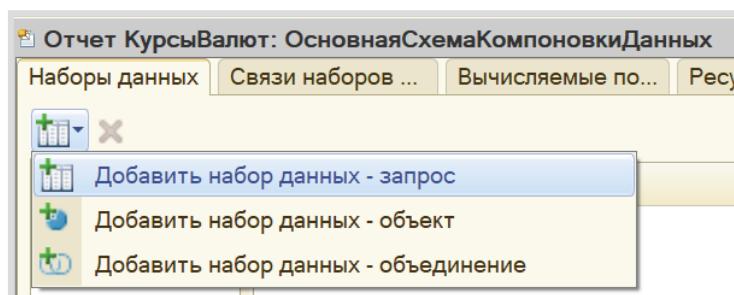
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

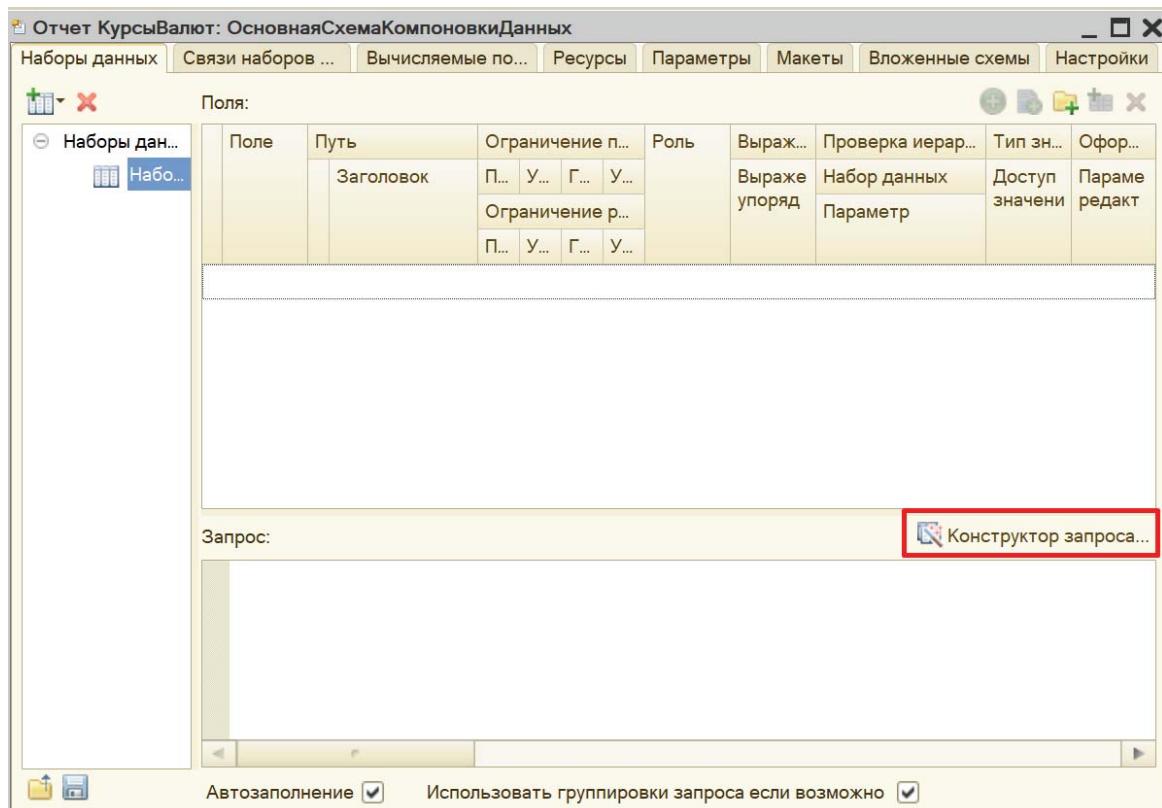
Создадим новый отчет «КурсыВалют». Для наполнения отчета воспользуемся конструктором схемы компоновки данных.





В открывшемся окне будем создавать запрос к базе данных. Для этого воспользуемся конструктором запроса.





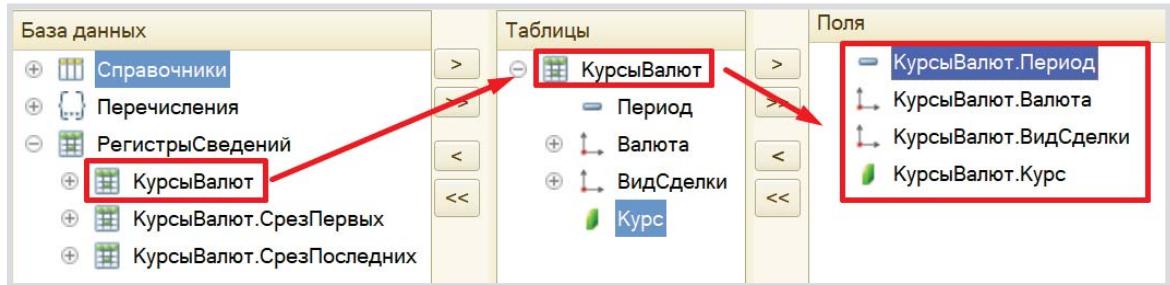
Открывшееся окошко имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных;
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета;
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

В качестве источника будем использовать таблицу *регистра сведений «КурсыВалют»*, которую только что самостоятельно заполнили в режиме пользователя.

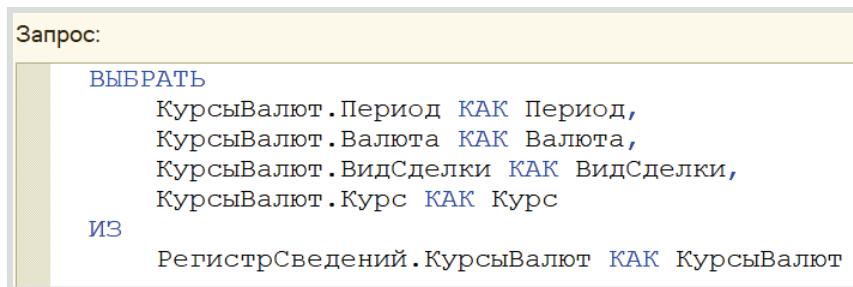
Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:

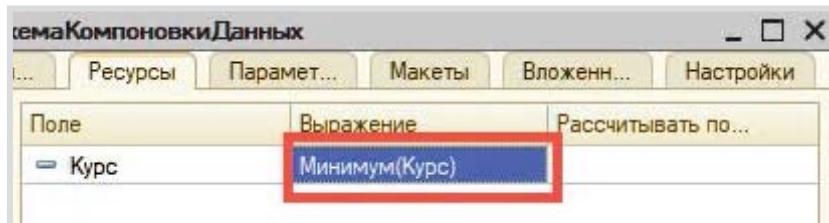


По завершении работы с конструктором нажимаем «OK».

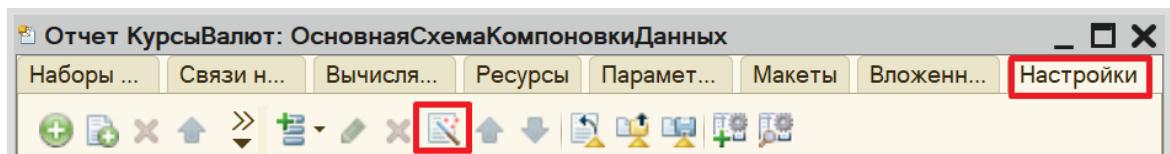
С помощью конструктора должен сформироваться следующий запрос к базе данных:



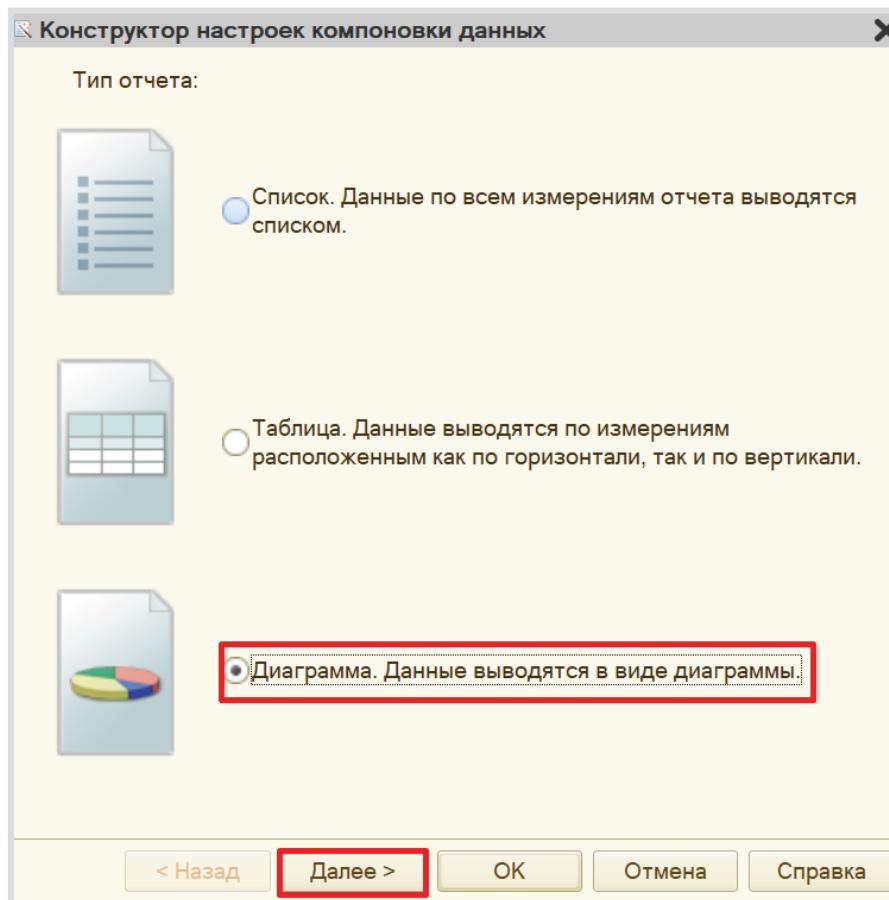
На вкладке «Ресурсы» определим реквизит «Курс» в качестве ресурса. Вместо выражения *по умолчанию* установим значение с функцией МИНИМУМ «Минимум(Курс)».



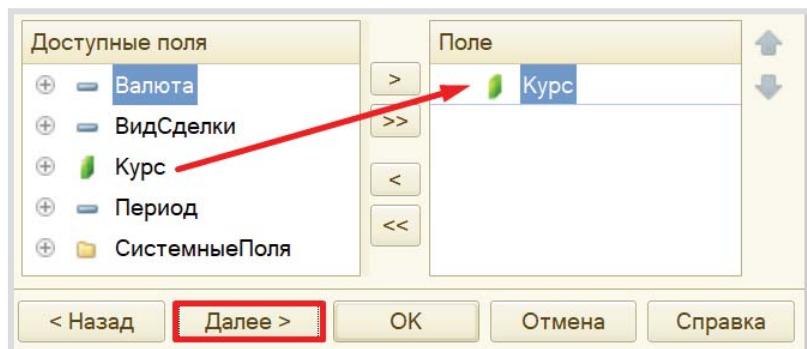
Теперь переходим к настройке внешнего вида отчета. Воспользуемся *конструктором настроек отчета*.



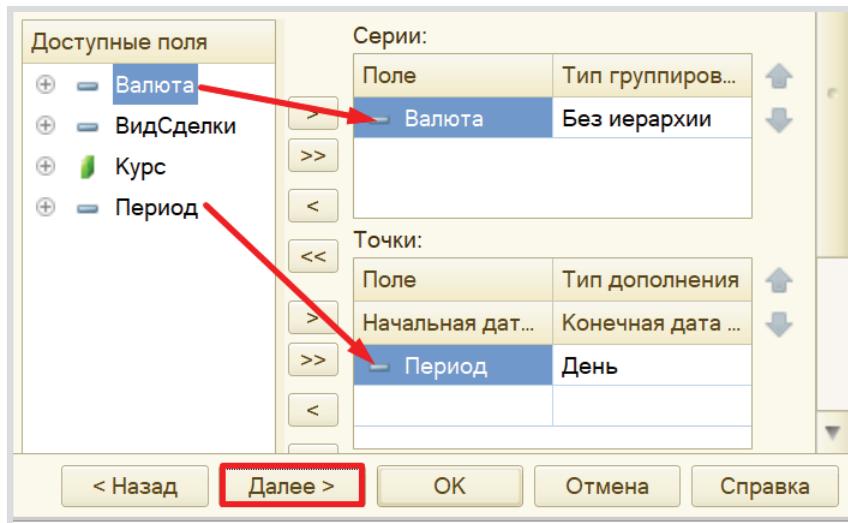
С помощью конструктора настроек отчета выберем вариант отчета в виде диаграммы.



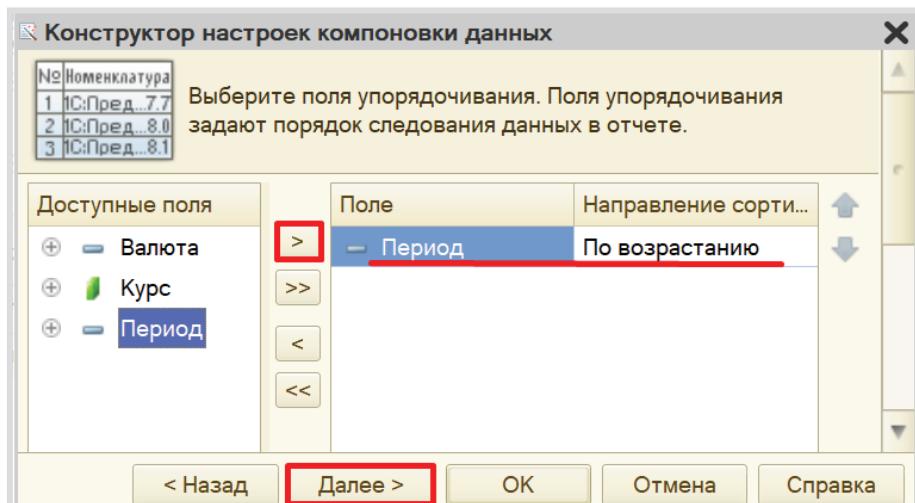
Укажем поля, которые будут отображаться в отчете.



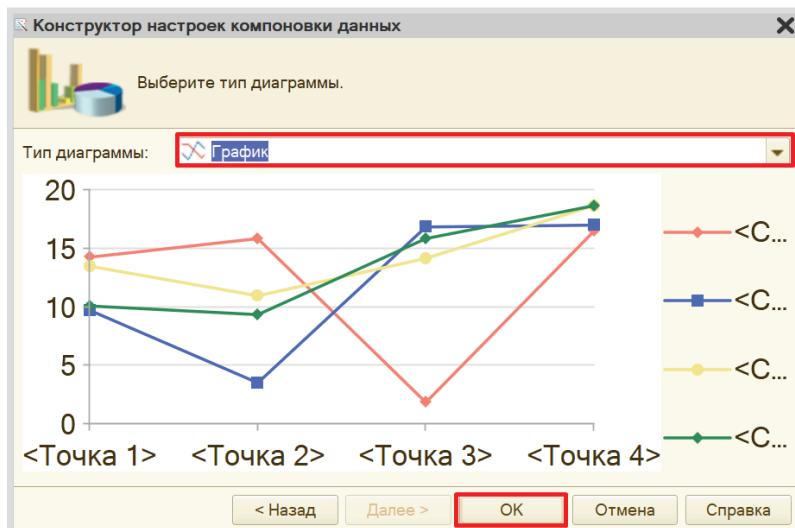
Переходим к следующему этапу настройки диаграммы. Точки и серии – это оси X и Y графика соответственно. Пусть ось Y отображает стоимость валюты на конкретную дату. Даты же пусть будут расположены по оси X.



Далее следует выбрать поля для упорядочивания. Упорядочим график по возрастанию периода.



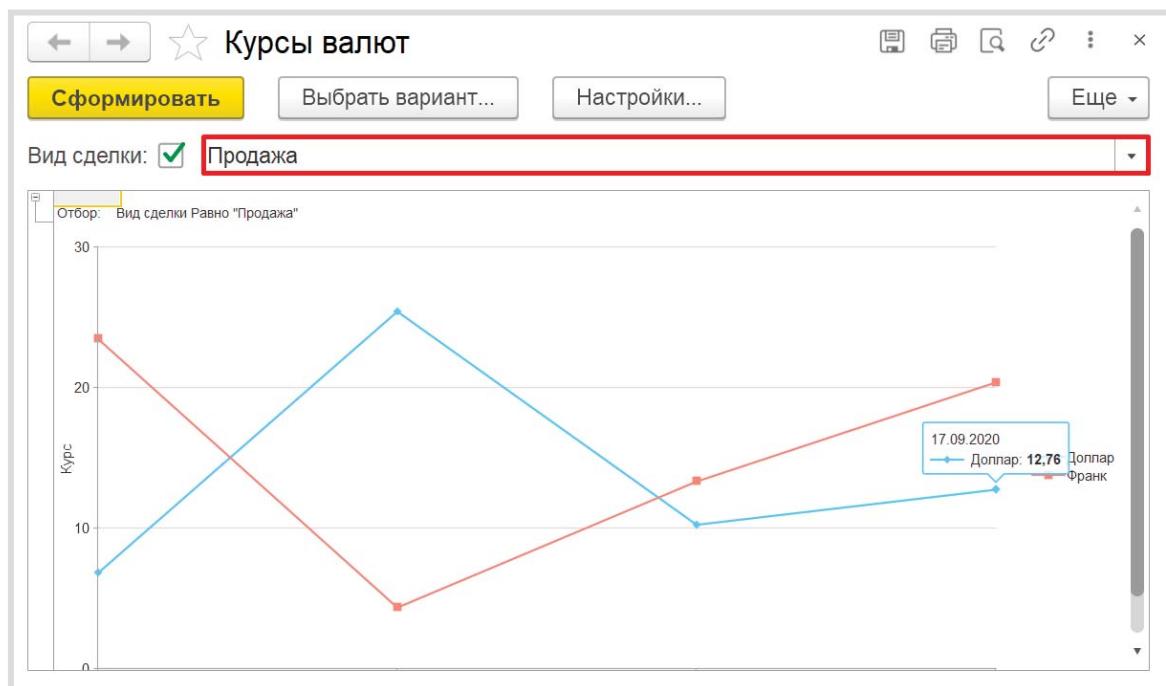
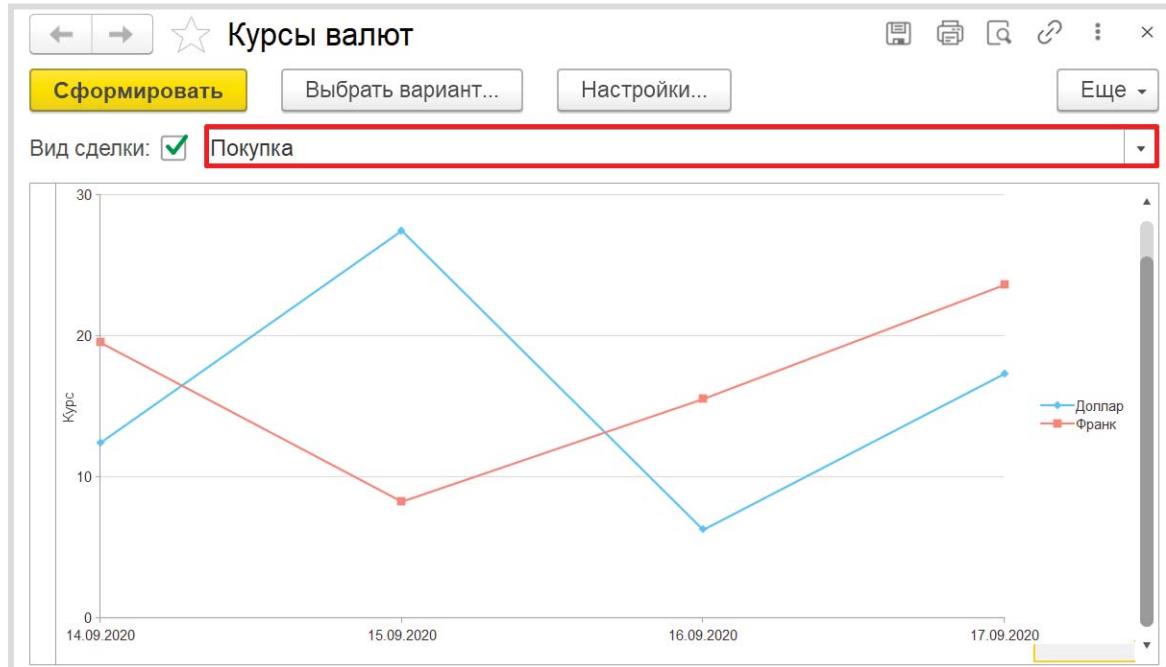
На последнем шаге выбираем тип диаграммы – *график*.



Теперь осталось лишь добавить вид сделки в пользовательские настройки, чтобы пользователь мог выбирать, какой график он хочет видеть. Это необходимо сделать на той же вкладке «Настройки», в нижней части окна.

Откроем вкладку «Отбор» и добавим отбор по виду сделки.

Посмотрим на то, как выглядят графики в режиме «1С:Предприятие».



Поставленная задача решена.

Лабораторная работа № 9

**СОЗДАТЬ НЕБОЛЬШУЮ
ИНФОРМАЦИОННУЮ СИСТЕМУ
ДЛЯ РЕГИСТРАЦИИ ПРОДАЖ
В СТУДЕНЧЕСКОМ КИОСКЕ**

Сложность: *

Теги: документ, регистр накопления, справочник,
обработка проведения, набор записей

ЗАДАНИЕ

Заказчик просит создать небольшую информационную систему для регистрации продаж в студенческом киоске.

1. В киоске продают канцелярские принадлежности, литературу и булочки. Нужно отобразить только факт продажи товаров в киоске.
2. В результате выполнения лабораторной работы должен получиться отчет вида:

Товар	Продано
Пончик	3
...	...

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать, смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

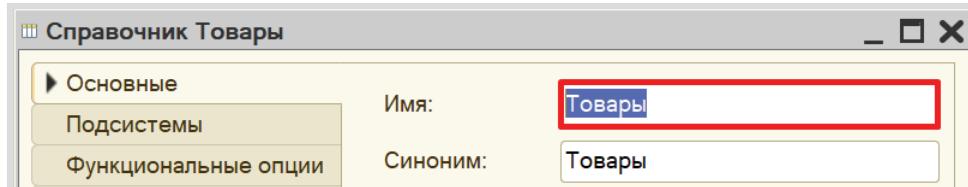
Проанализируем задачу и примем решение, что нам нужен *справочник*.

Определение

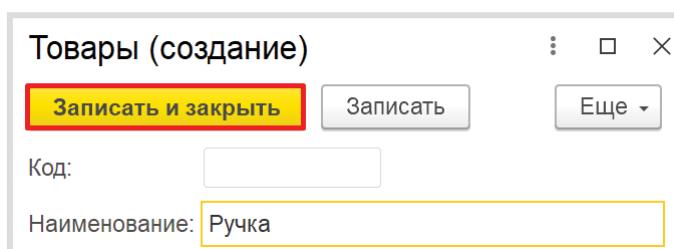
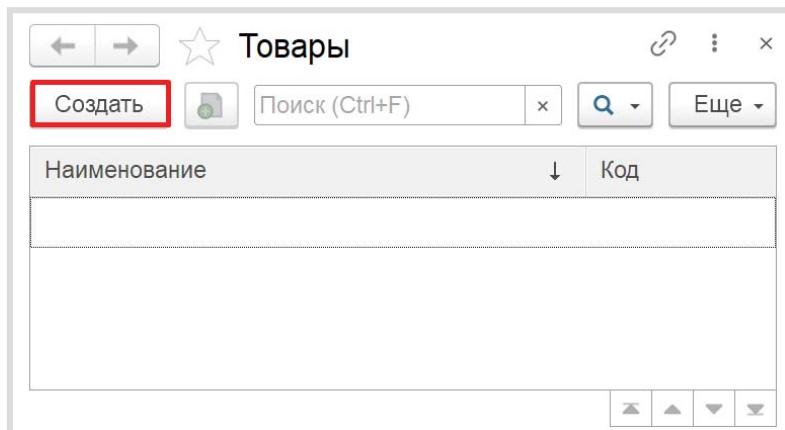
Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

В данном справочнике будет храниться перечень товаров, продаваемых в киоске.

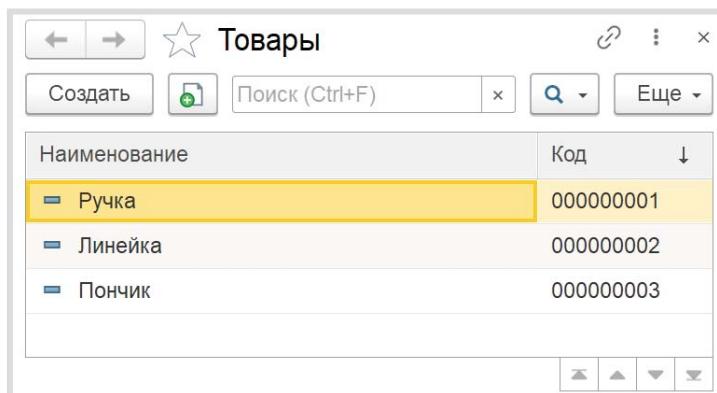
Создадим новый справочник «Товары».



Откроем справочник в режиме «1С:Предприятие» и добавим несколько товаров, например, «Ручка», «Линейка», «Пончик».



Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



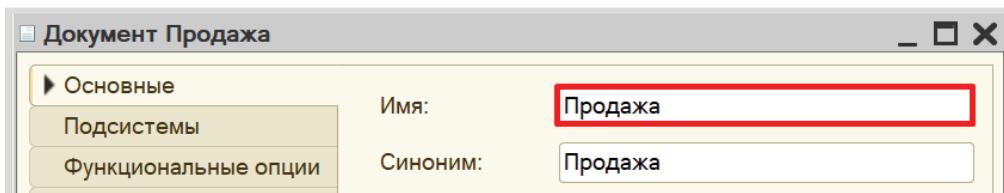
Создав справочник, мы организовали хранение объектов аналитики. Далее следует ответить на вопрос: «Как мы будем регистрировать продажи?».

Для данной цели подойдет объект конфигурации *документ*.

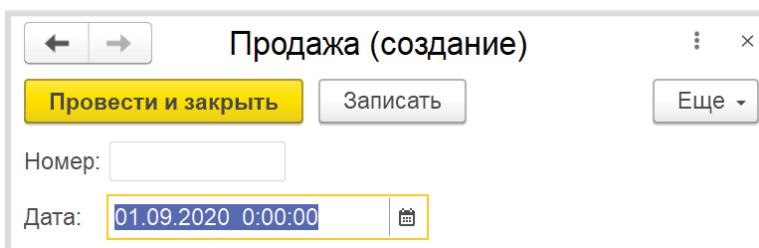
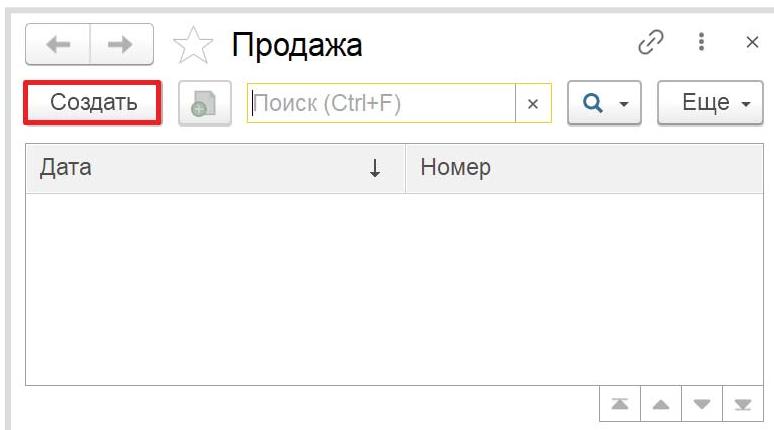
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим новый документ «Продажа».



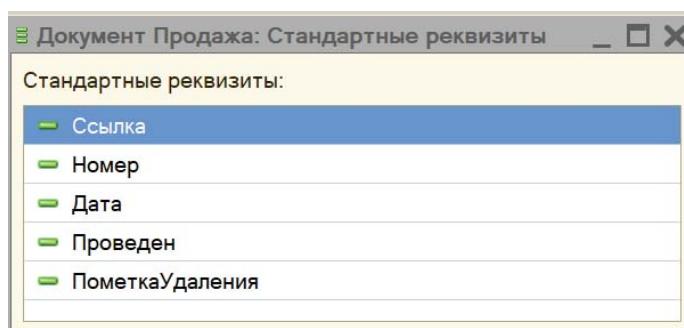
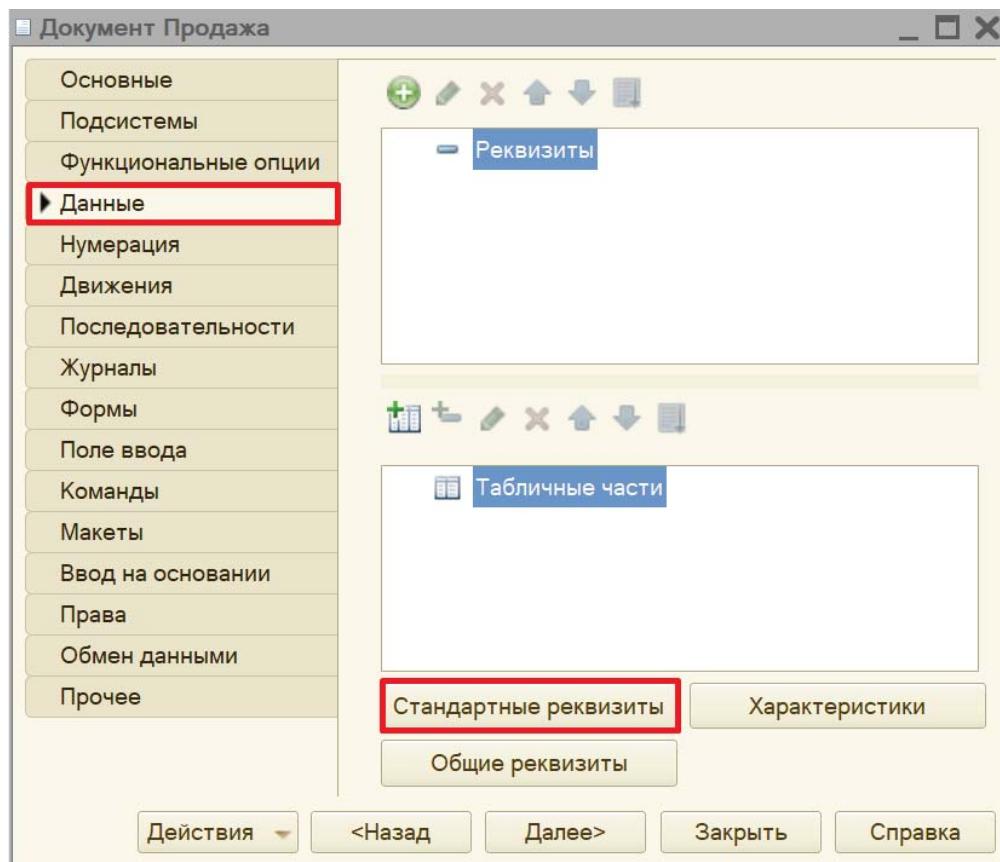
Вот так данный документ будет выглядеть в режиме «1С:Предприятие»:



Любой документ может находиться в одном из двух состояний: *подготовленный к совершению* или *совершенный*:

- чтобы подготовить документ для использования в будущем, необходимо его записать;
- чтобы отметить документ как совершенный – провести.

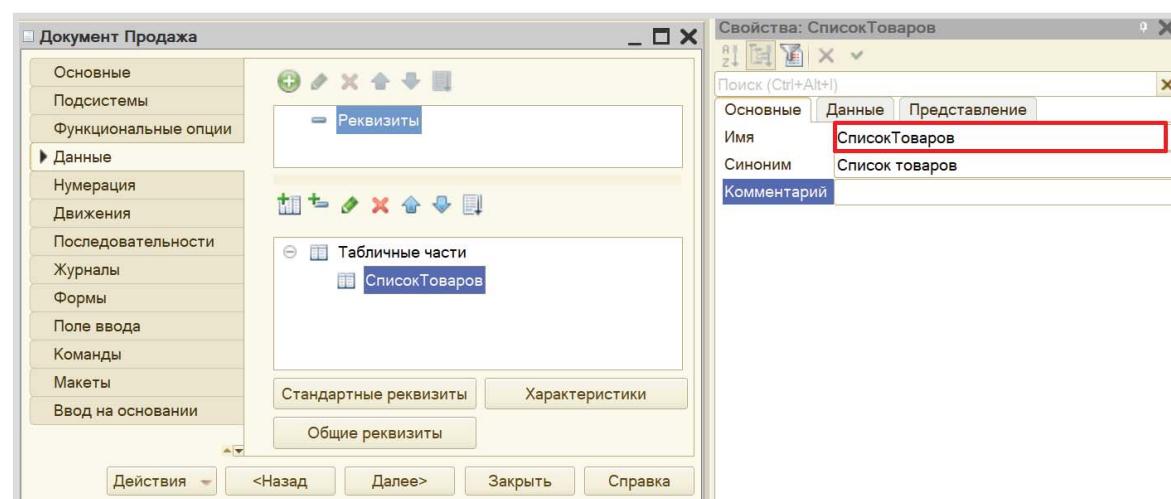
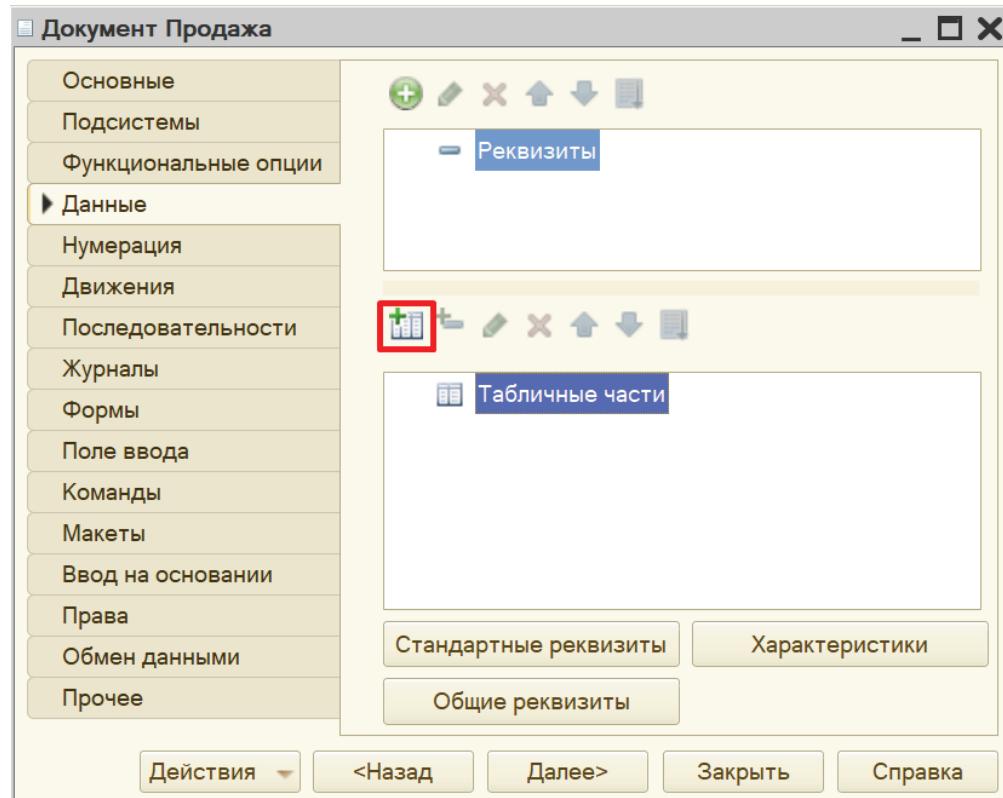
Поля «Номер» и «Дата» система создала автоматически при добавлении нового документа. Это стандартные реквизиты документа. Ознакомиться с перечнем всех стандартных реквизитов можно на вкладке «Данные» окна редактирования документа.



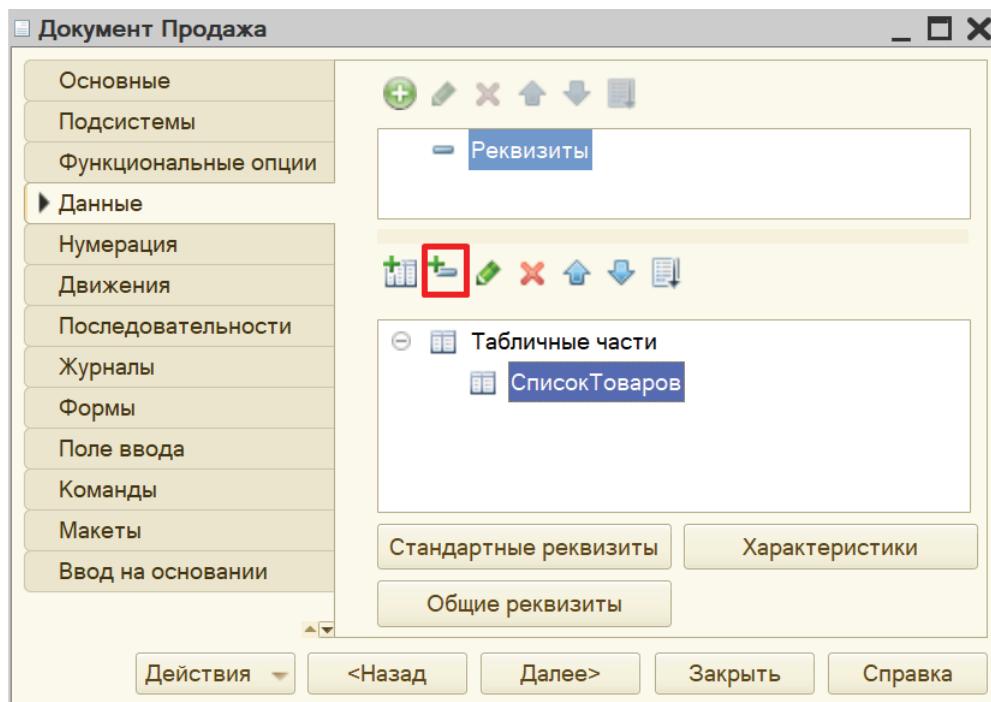
- Стоит обратить внимание на ключевой стандартный реквизит «Дата». Он будет хранить время регистрации продажи. Дополнительно создавать реквизит «Дата» нам не нужно, система уже позаботилась об этом.

Далее следует определиться со структурой документа. Мы хотим, чтобы в одном документе можно было регистрировать продажу сразу нескольких товаров. В этом нам поможет табличная часть документа.

Создадим табличную часть. Для этого воспользуемся кнопкой «Добавить табличную часть».

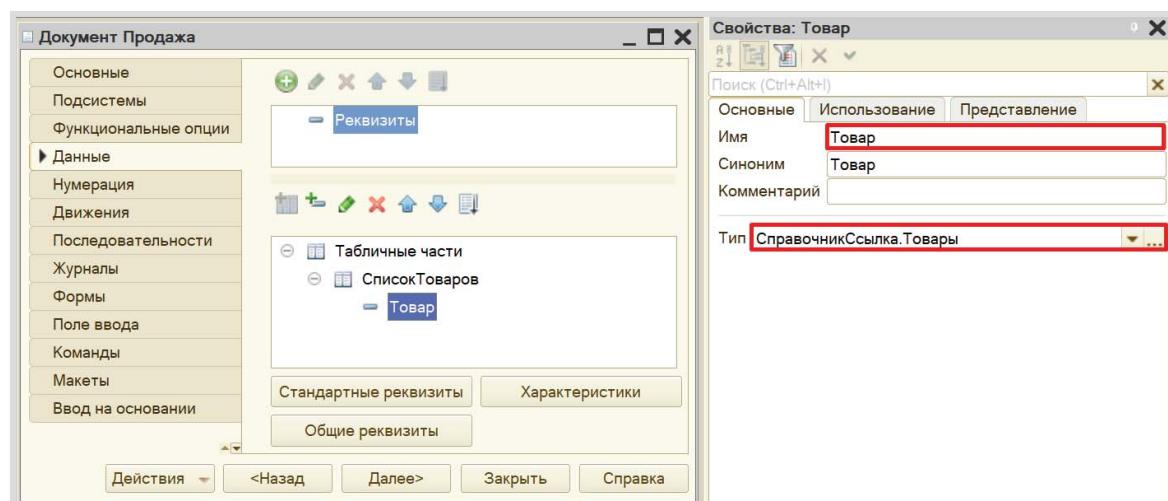


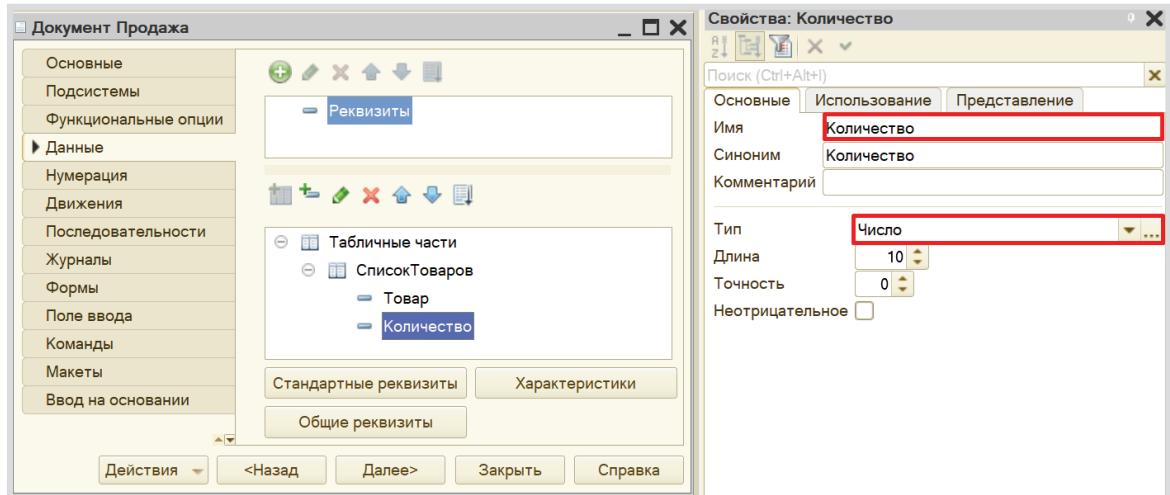
Далее нужно добавить реквизиты табличной части – колонки таблицы. Для поставленной задачи создайте колонки «Товар» и «Количество». Для добавления реквизита воспользуйтесь кнопкой «Добавить реквизит».



Для каждого из реквизитов нужно определить *имя* и *тип данных*:

- реквизит «Товар» должен иметь тип «СправочникСсылка.Товары», чтобы хранить ссылку на элемент справочника «Товары»;
- реквизит «Количество» должен быть числовым.





Проверим работу системы. Создадим документ и добавим в него несколько товаров.

Продажа (создание)

Номер:

Дата: 01.09.2020 0:00:00

Добавить Поиск (Ctrl+F) Еще

N	Товар	Количество

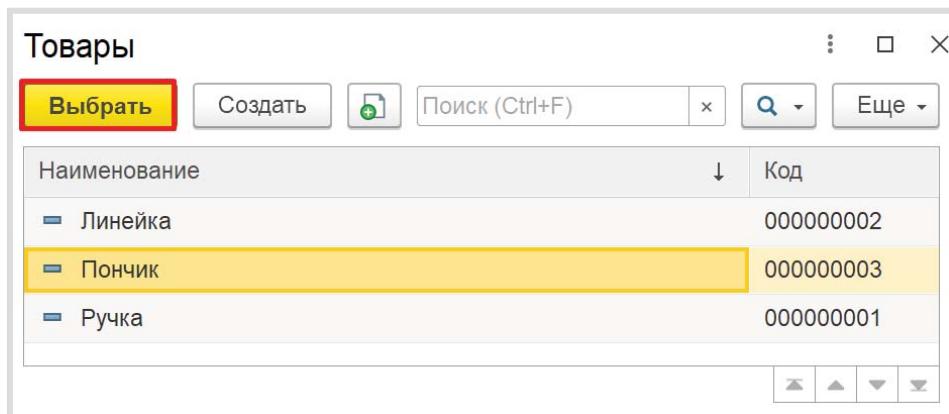
Продажа (создание) *

Номер:

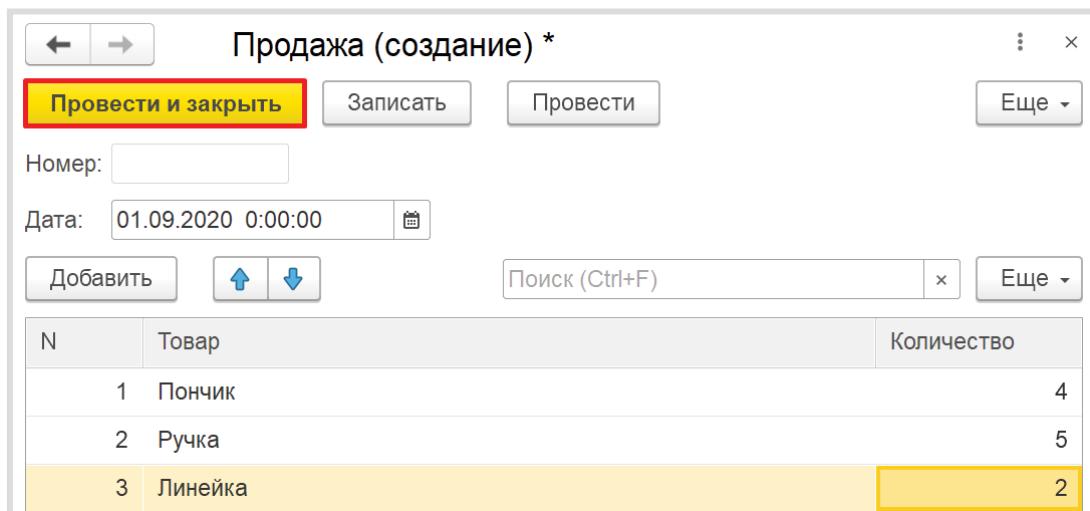
Дата: 01.09.2020 0:00:00

Добавить Поиск (Ctrl+F) Еще

N	Товар	Количество
1	Ручка	
	Линейка	
Показать все		<input type="button"/>



Аналогично нужно добавить еще несколько товаров, а также указать их количество.



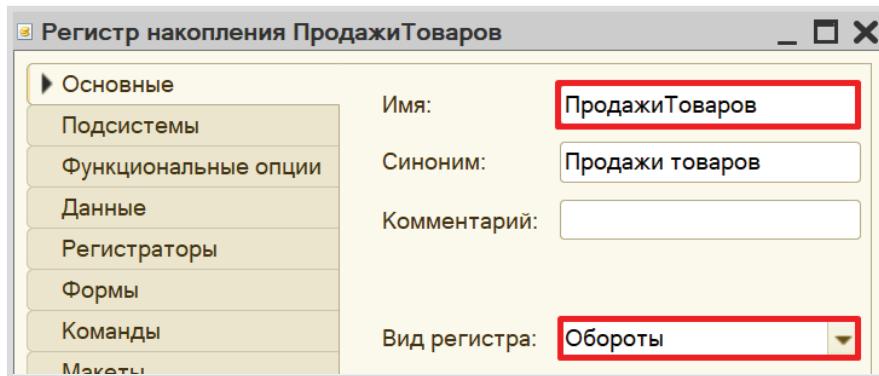
Можно ли теперь на основе таких документов построить отчет по продажам? Можно, но для этого придется прибегнуть к грубому перебору всех существующих документов. Данный вариант является неправильным, потому что, если таких документов окажется очень много, система будет требовать большого количества ресурсов и времени.

Для решения данной проблемы и ускорения процесса извлечения данных создадим еще один объект – *регистр накопления*.

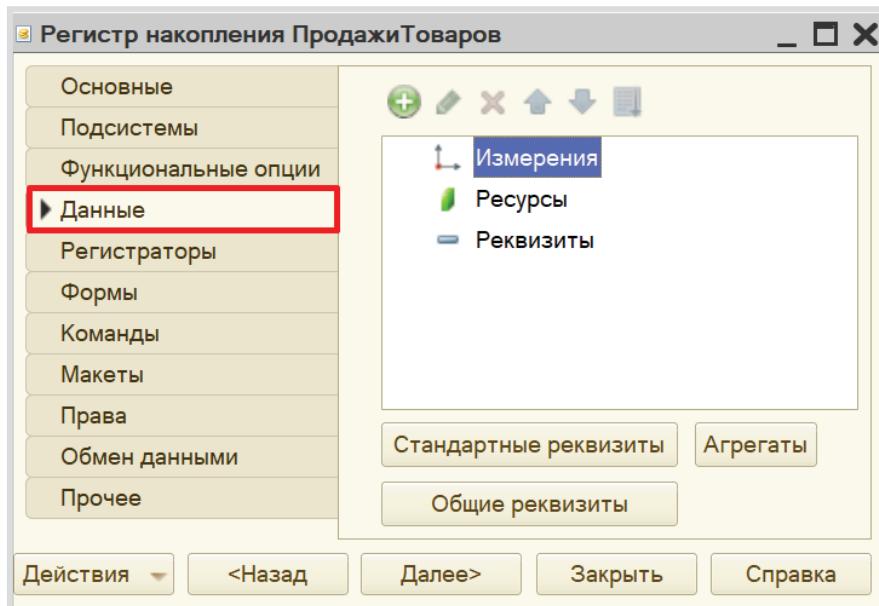
Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

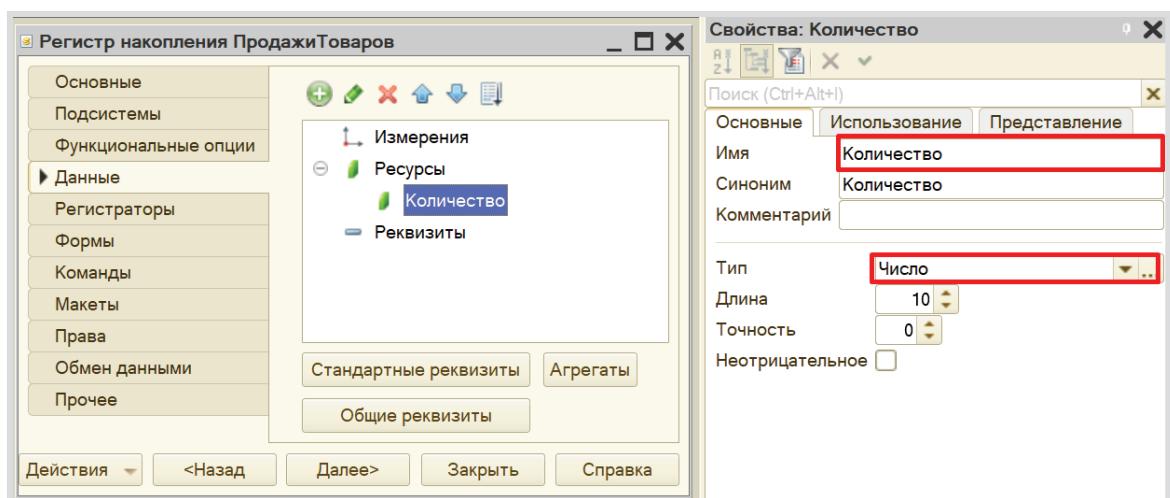
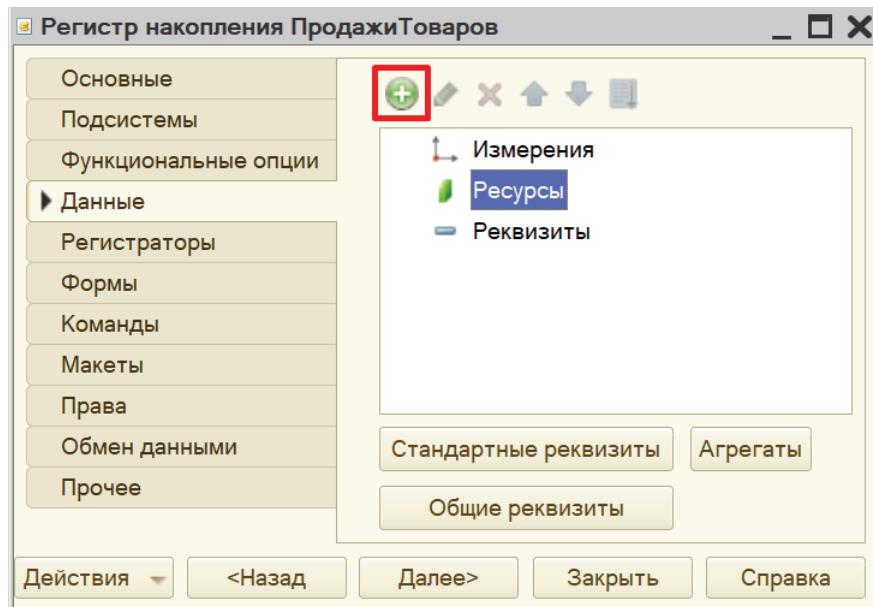
Данный регистр должен быть оборотным, чтобы накапливать данные о продаже товаров.



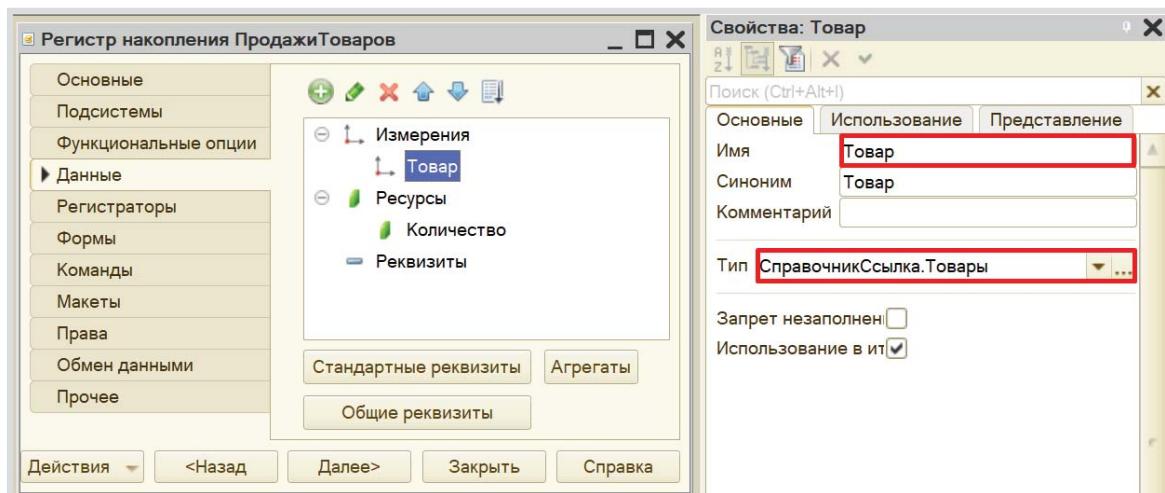
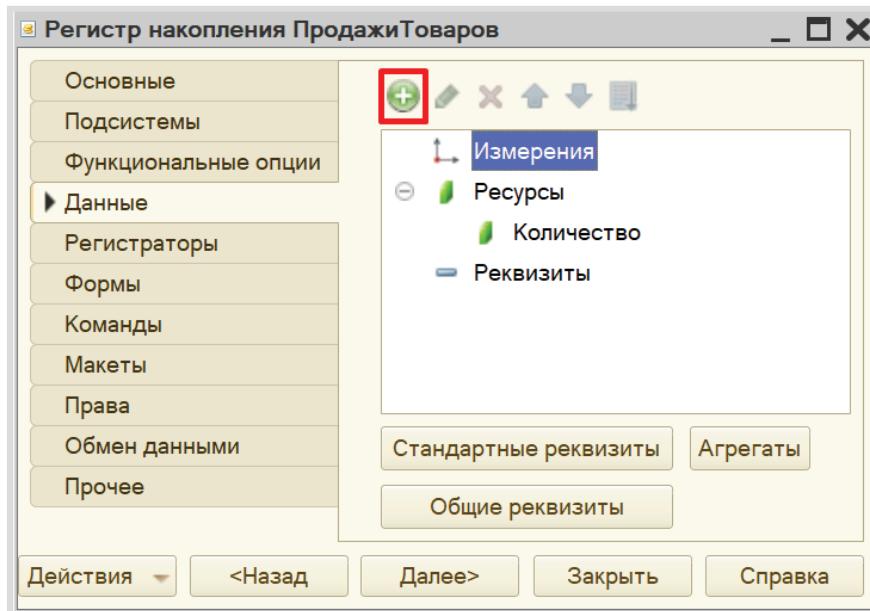
Переходим к описанию структуры *регистра накопления*. Для этого переходим на вкладку «Данные».



Заполнение данного окна всегда проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, необходимо задать вопрос: «Что нам нужно считать?». Нам нужно считать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число».



Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим считать количество. Мы хотим считать количество (чего?) товаров. Значит, в качестве измерения следует добавить реквизит «Товар». Тип данного реквизита – «СправочникСсылка.Товары».

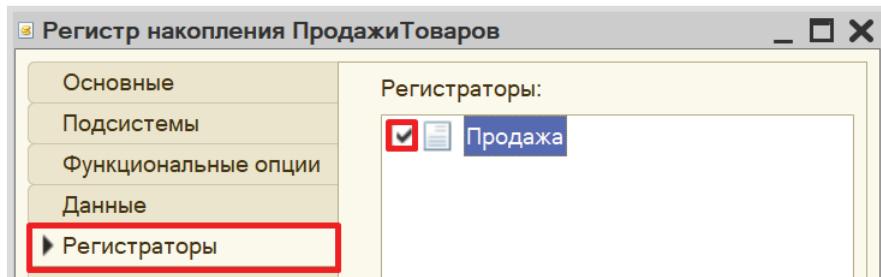


Чтобы *регистр накопления* заработал, нужно сделать следующее:

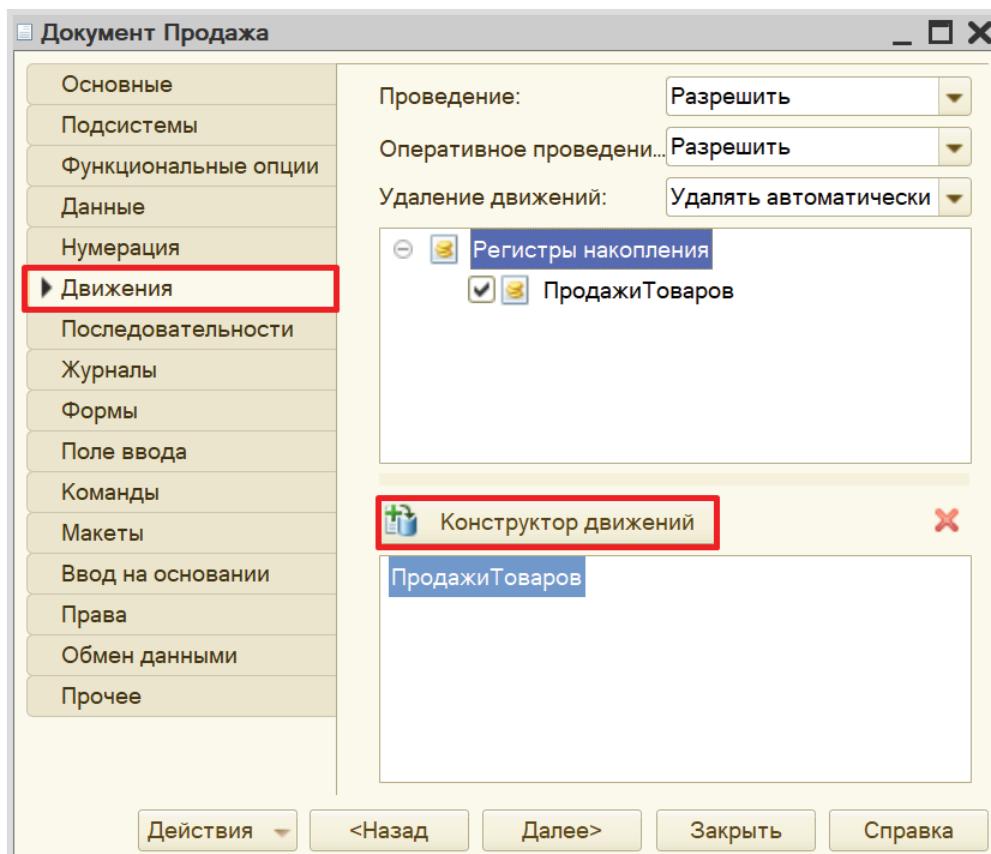
1. Определить источники данных, которые должны попадать в регистр (определить документы-регистраторы).
2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.

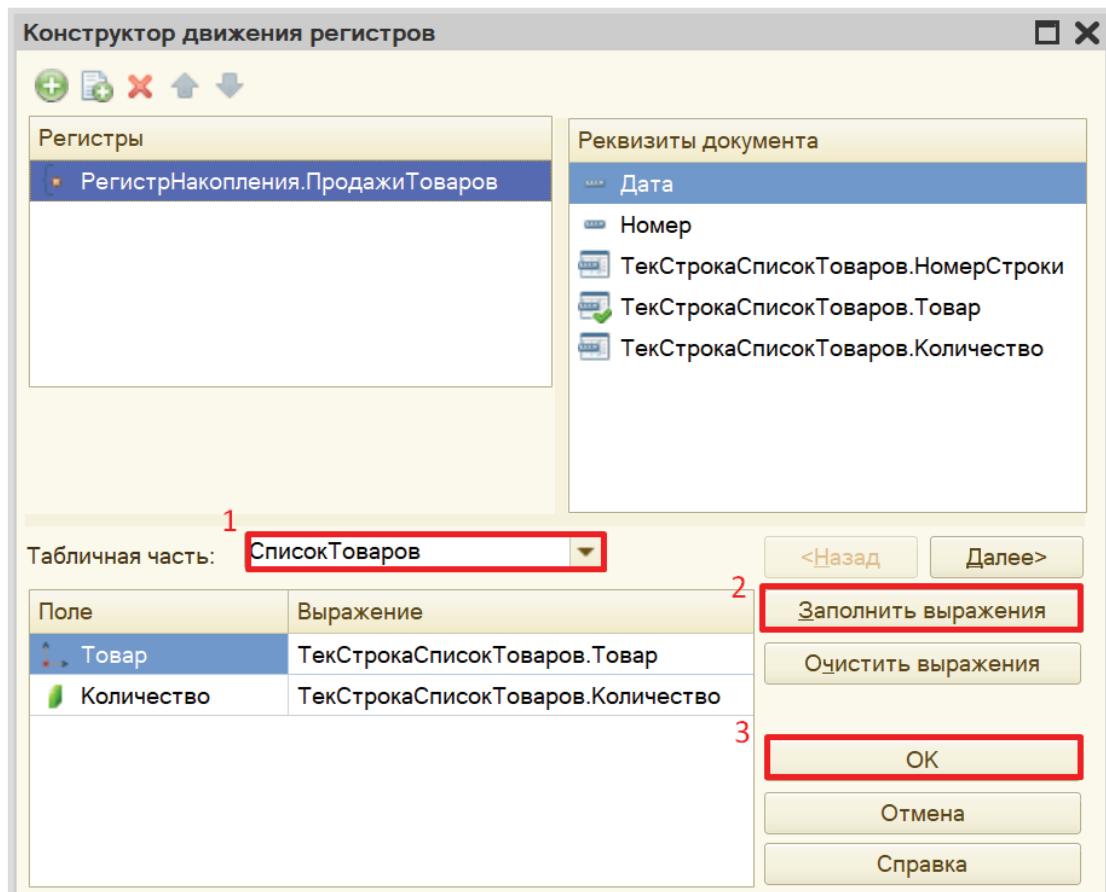
В нашем случае, данные в *регистр накопления* должны попадать из документа «Продажа», следовательно, данный документ и будет являться регистратором.

Для определения регистратора *регистра накопления* необходимо выбрать нужный документ на вкладке «Регистраторы».



Чтобы определить правила передачи данных из регистратора в *регистр накопления*, откроем вкладку «Движения» окна редактирования документа «Продажа» и воспользуемся конструктором движений.





Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты *регистра накопления*. Нужно заполнить поле «Выражение» реквизитами документа.

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.

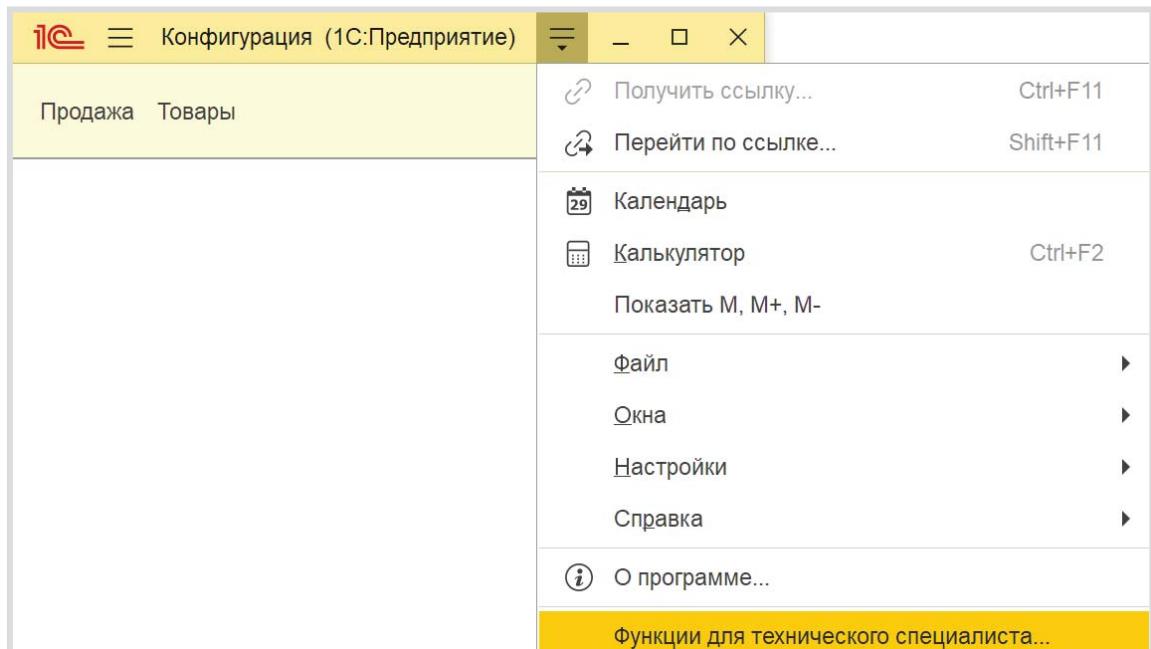
При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

```
Процедура ОбработкаПроведения(Отказ, Режим)
//{{ __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
//  Данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
//
//  регистр ПродажиТоваров
Движения.ПродажиТоваров.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ПродажиТоваров.Добавить();
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;

//}} } __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Проверим работу *регистра накопления*. В первую очередь, необходимо перепровести (проводи заново) созданный нами ранее документ, чтобы он сформировал движение в *регистре накопления*.

Чтобы посмотреть на движения в *регистре накопления* воспользуемся функциями для технического специалиста:



Найдем наш регистр, откроем его и посмотрим на движения.

The screenshot shows the 'Функции для технического специалиста' (Functions for Technical Specialist) window. In the left pane, under 'Регистры накопления' (Accumulation Registers), the 'Продажи товаров' (Sales) register is selected and highlighted with a yellow box. The main pane displays the 'Продажи товаров' (Sales) table with the following data:

Период	Регистратор	Номер ...	Товар	Количество
• 28.08.2020 18:27:52	Продажа 000000001 от 28.08.2020 ...	1	Линейка	3
• 28.08.2020 18:27:52	Продажа 000000001 от 28.08.2020 ...	2	Ручка	6

Таким образом, *регистр накопления* является своеобразной итоговой таблицей, куда заносятся данные из документа «Продажа».

Теперь можно строить отчет.

Определение

Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: [https://v8.1c.ru/platforma/отчет/](https://v8.1c.ru/platforma/otchet/)).

Создадим новый отчет «ОтчетПоПродажам». Для наполнения отчета воспользуемся конструктором схемы компоновки данных.

Отчет ОтчетПоПродажам

Основные

Подсистемы

Функциональные опции

Данные

Формы

Команды

Макеты

Права

Прочее

Имя: ОтчетПоПродажам

Синоним: Отчет по продажам

Комментарий:

Основная схема компоновки данных:

Открыть схему компоновки данных

Конструктор макета

Имя: ОсновнаяСхемаКомпоновкиДанных

Синоним: Основная схема компоновки данных

Комментарий:

Выберите тип макета:

- Табличный документ
- Текстовый документ
- Двоичные данные
- Active document
- HTML документ
- Географическая схема
- Графическая схема
- Схема компоновки данных
- Макет оформления компоновки данных
- Внешняя компонента

Загрузить из файла:

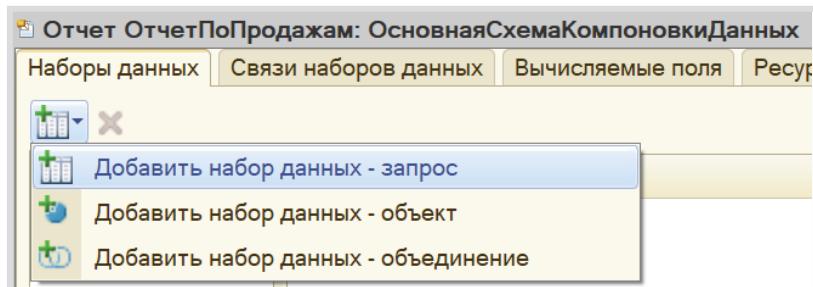
Готово

Отмена

Справка

Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными.

Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Воспользуемся конструктором запроса.

Поля:									
Наборы данных НаборД...	Поле	Путь	Ограничение п...	Роль	Выраже...	Проверка иерархии:	Тип зна...		
								Автозаголовок	P... Y... G... U...
								Ограничение р...	P... Y... G... U...

Запрос: [Конструктор запроса...]

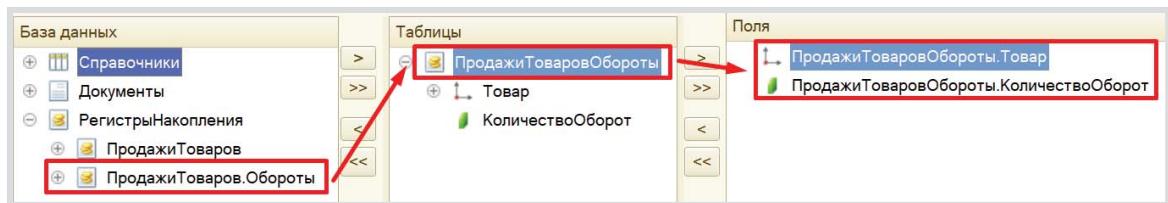
Открывшееся окно имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

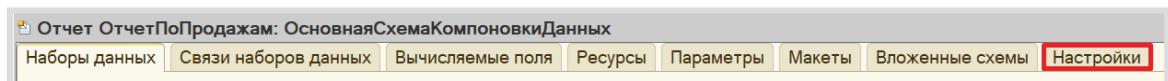
Данные будем брать не из *регистра накоплений* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Эта виртуальная таблица позволит получить уже просуммированные значения по всем документам.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

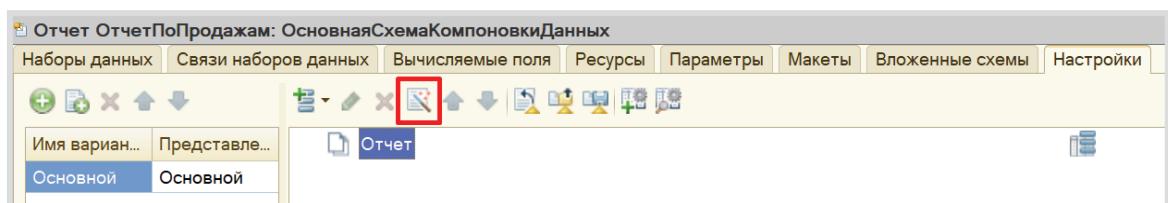
В результате данное окно должно быть заполнено следующим образом:



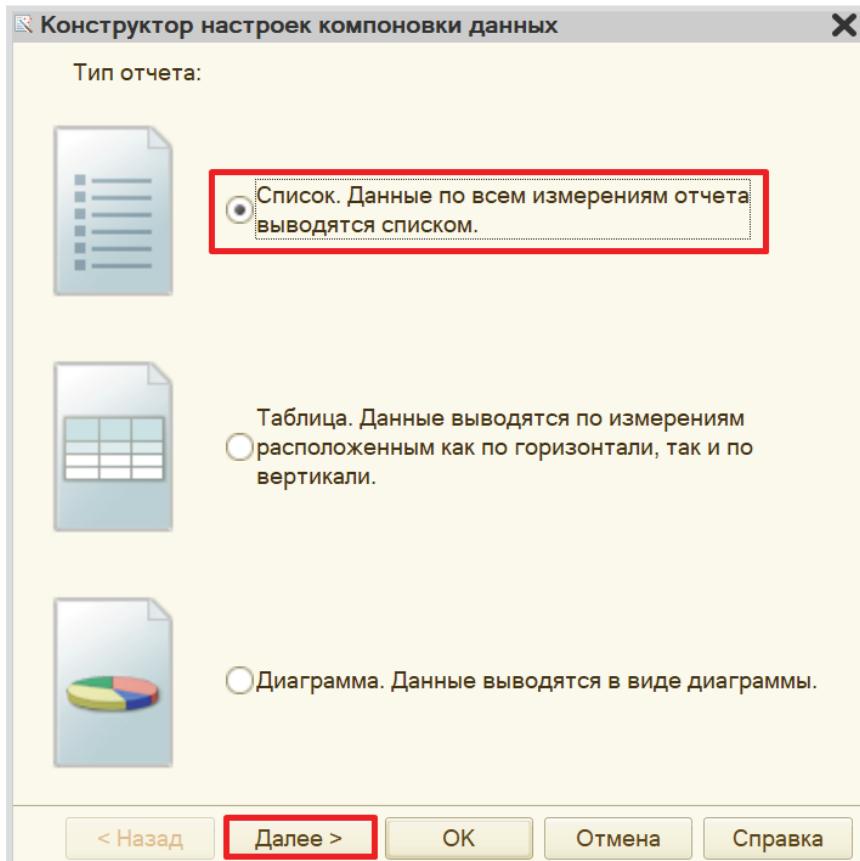
Запрос к базе данных сформирован. Теперь необходимо настроить внешний вид отчета. Для этого нужно перейти на вкладку «Настройки».



Для настройки внешнего вида отчета воспользуемся конструктором настроек отчета.

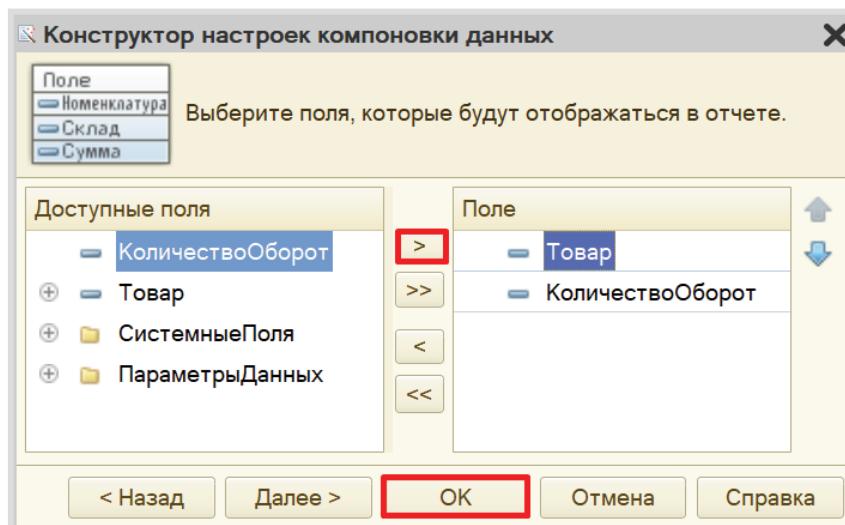


Наш отчет будет иметь форму списка.



Далее нужно выбрать поля, которые будут отображены в отчете.

Обратите внимание на порядок полей в правом столбце, именно в таком порядке они будут отображены в отчете.



Чтобы удостовериться, что отчет работает корректно, можно создать, наполнить и провести еще несколько документов «Продажа», а затем посмотреть на результаты в отчете. Если вы все сделали верно, то товары, указанные в документах, должны быть просуммированы.

← → ★ Отчет по продажам	
Сформировать	
Выбрать вариант...	
Настройки...	
Товар	Количество Оборот
Пончик	12
Ручка	15
Линейка	6

Поставленная задача решена.

Лабораторная работа № 10

**РАЗРАБОТКА КОНФИГУРАЦИИ
ДЛЯ УЧЕТА РАБОТЫ СТУДЕНТОВ
НА ЗАНЯТИЯХ**

Сложность: *

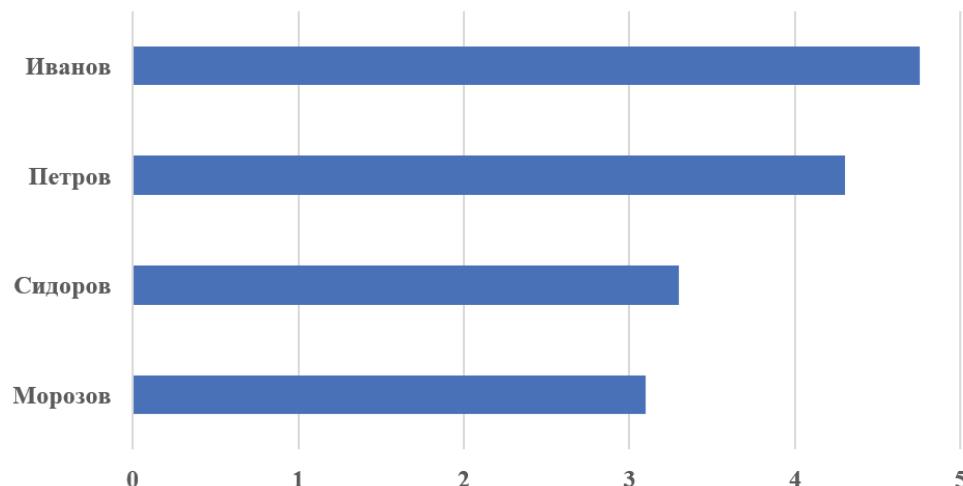
Теги: справочник, документ, регистр накопления,
схема компоновки данных, условное оформление

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета работы студентов на занятиях. Учет ведется в разрезе дисциплин.

1. В системе необходимо зарегистрировать *Занятия студентов*. В конце занятия пользователь в шапке документа указывает название дисциплины, а в табличной части – какие студенты какой балл получили.
2. Нужно построить *Отчет по текущей успеваемости студентов*.

Форма отчета:



Отчет строится по среднему арифметическому баллу студента по указанной дисциплине.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

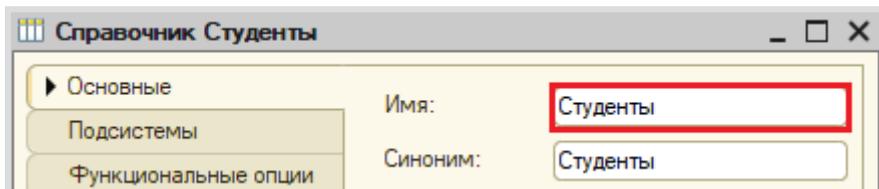
«Заказчик просит разработать конфигурацию для учета работы студентов на занятиях. Учет ведется в разрезе дисциплин».

Из условия следует, что необходимо хранить информацию о студентах и посещаемых ими дисциплинах. Для решения этой задачи нам понадобятся *справочники*.

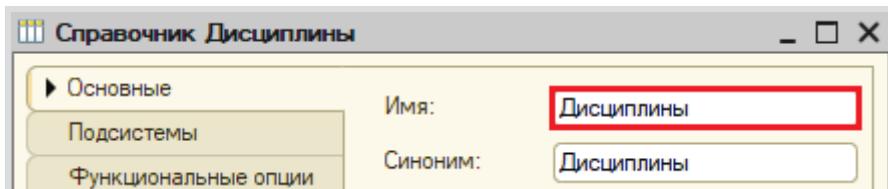
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

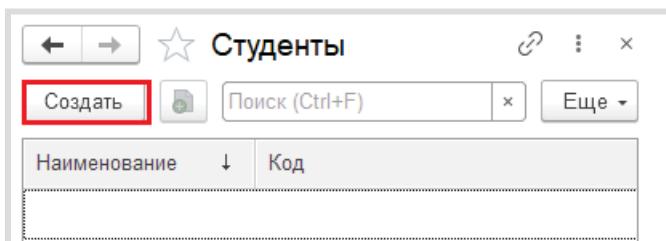
Создадим справочник «Студенты».



Создадим справочник «Дисциплины».



Откроем программу в режиме «1С:Предприятие» и добавим в каждый справочник несколько элементов.



Наименование	Код
Любимов	000000002
Малиновский	000000003
Марковский	000000004
Федоренко	000000001

Аналогично добавьте несколько элементов в справочник «Дисциплины».

Наименование	Код
Высшая математика	000000001
Операционные системы	000000002

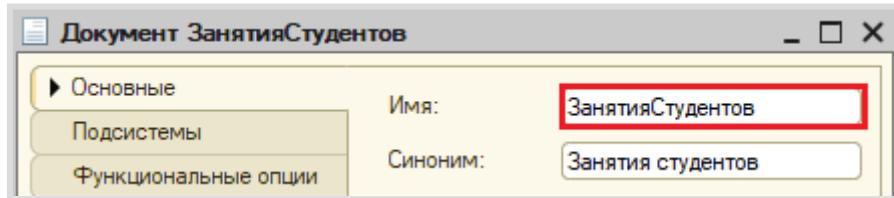
«В системе необходимо зарегистрировать Занятия студентов».

Для регистрации занятий студентов следует воспользоваться объектом конфигурации *документ*.

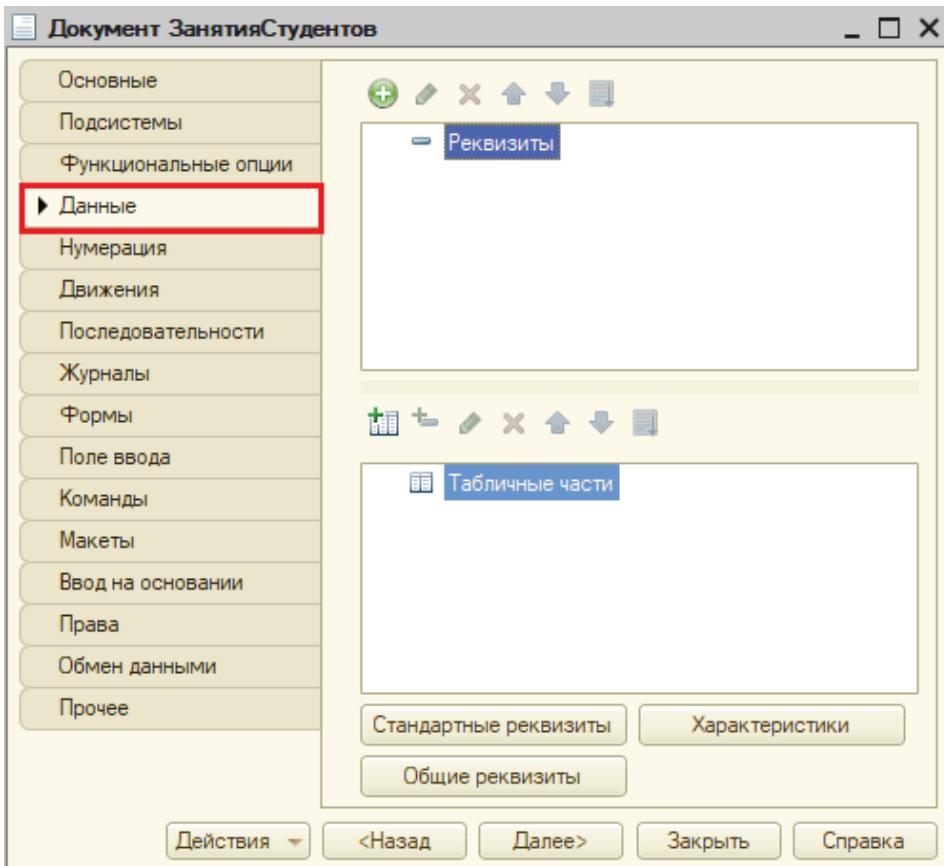
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим новый документ «ЗанятияСтудентов».



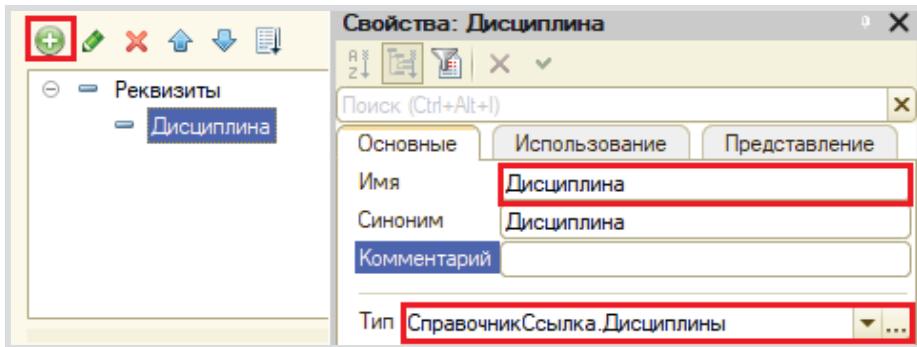
Для настройки структуры документа переходим на вкладку «Данные».



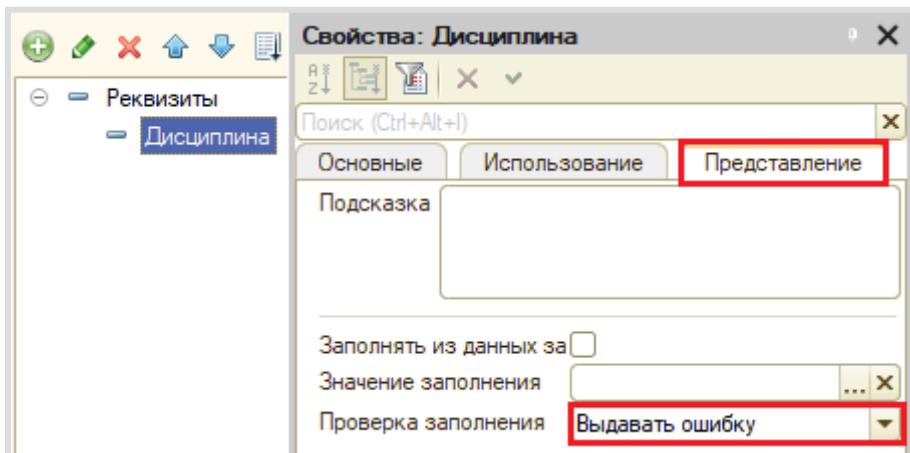
«В конце занятия пользователь в шапке документа указывает название дисциплины, а в табличной части – какие студенты какой балл получили».

Под шапкой документа подразумеваются данные, хранящиеся в верхней части документа до табличной части.

Добавим реквизит «Дисциплина».

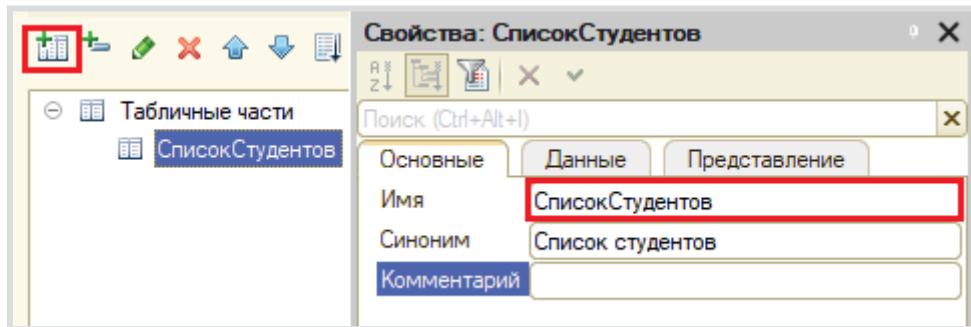


Сделаем реквизит обязательным для заполнения.

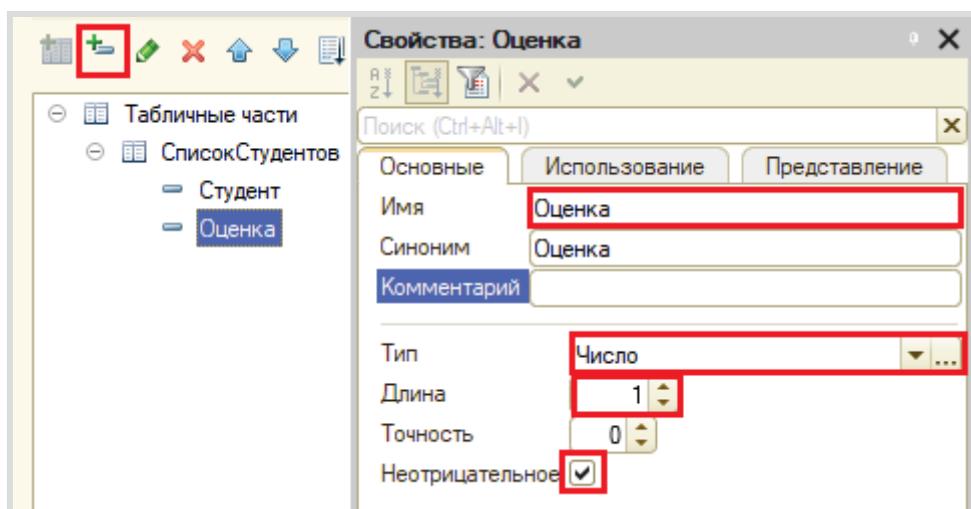
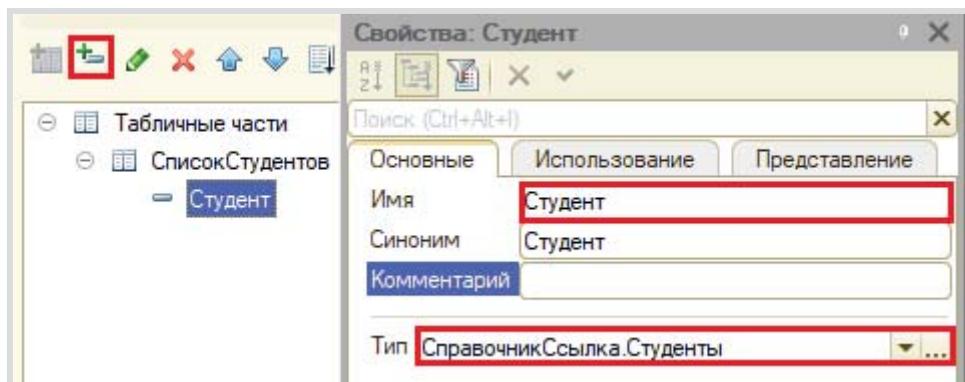


С помощью такой настройки пользователь не сможет сохранить документ, пока не заполнит поле «Дисциплина».

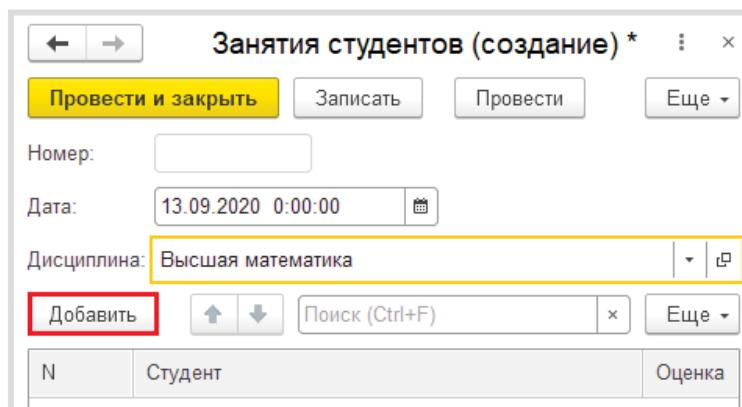
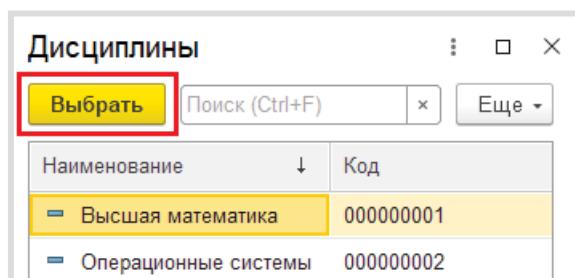
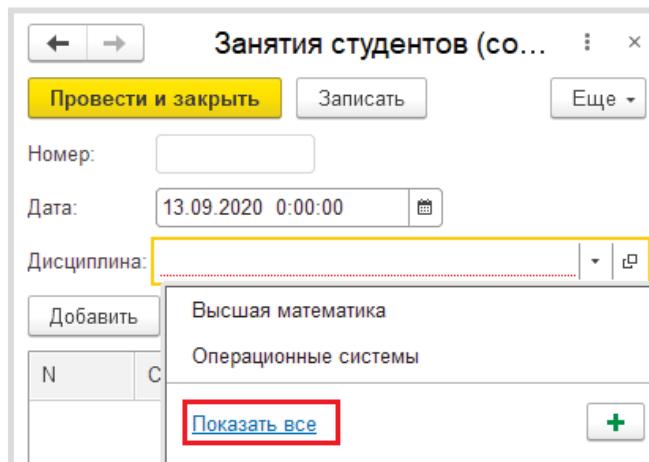
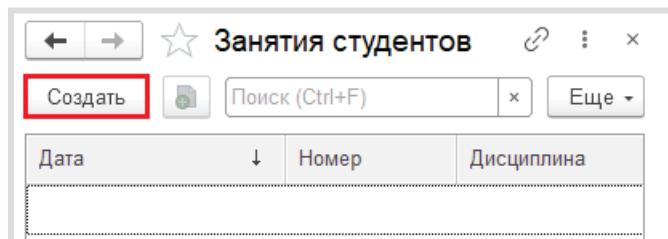
Далее, исходя из условия, нам понадобится добавить табличную часть.



Теперь добавим два реквизита табличной части: «Студент» (тип – СправочникСсылка.Студенты) и «Оценка» (тип – «Число»).



Запустим режим «1С:Предприятие» и попробуем создать несколько документов.



Занятия студентов (создание) *

Провести и закрыть Записать Провести Еще ▾

Номер:

Дата: 13.09.2020 0:00:00

Дисциплина: Высшая математика

Добавить Поиск (Ctrl+F) Еще ▾

N	Студент	Оценка
1	Любимов	4
2	Марковский	3
3	Малиновский	3
4	Федоренко	2

Занятия студентов

Создать Поиск (Ctrl+F) Еще ▾

Дата	↓	Номер	Дисциплина
09.09.2020 12:00:00		000000005	Высшая математика
10.09.2020 12:00:00		000000001	Операционные системы
11.09.2020 12:00:00		000000002	Операционные системы
12.09.2020 12:00:00		000000003	Операционные системы
13.09.2020 20:05:51		000000004	Высшая математика

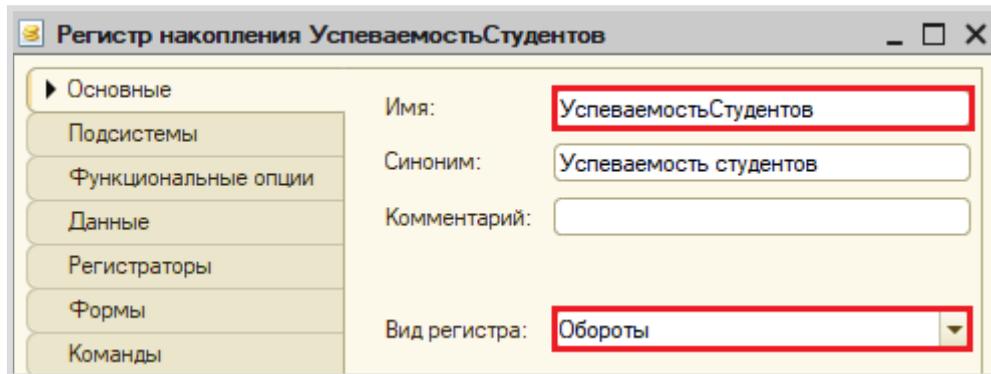
Можно ли теперь на основе таких документов построить отчет об успеваемости? Можно, но для этого придется прибегнуть к грубому перебору всех существующих документов. Данный вариант является неправильным, потому что, если таких документов окажется очень много, система будет требовать большого количества ресурсов и времени.

Для решения данной проблемы и ускорения процесса извлечения данных создадим еще один объект – *регистр накопления*.

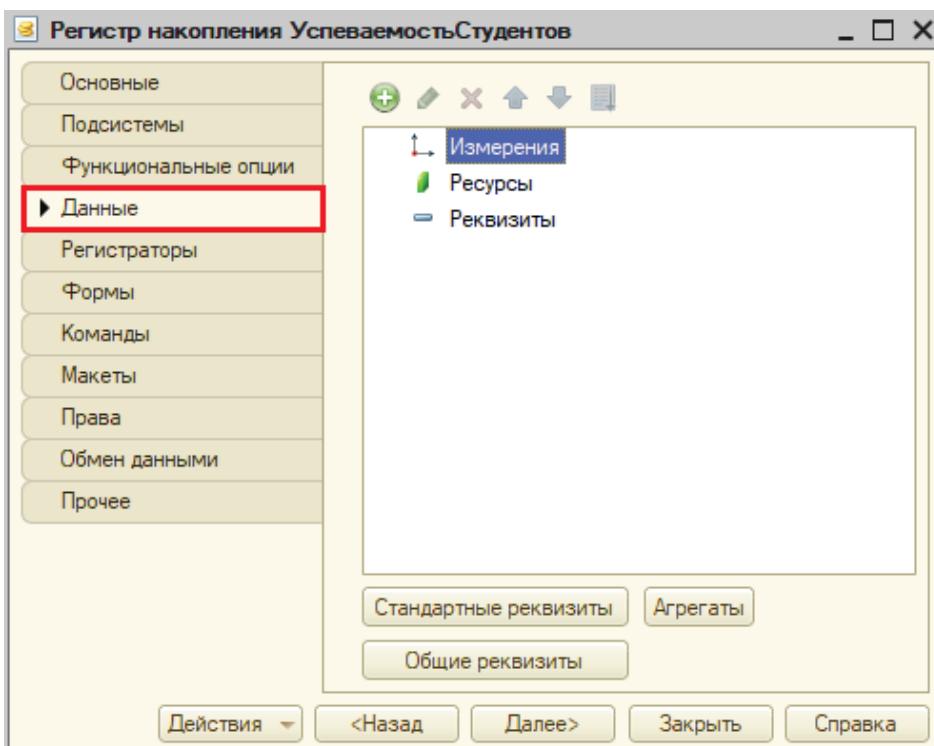
Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Добавим новый *регистр накопления* «УспеваемостьСтудентов». Вид данного регистра – «Обороты».

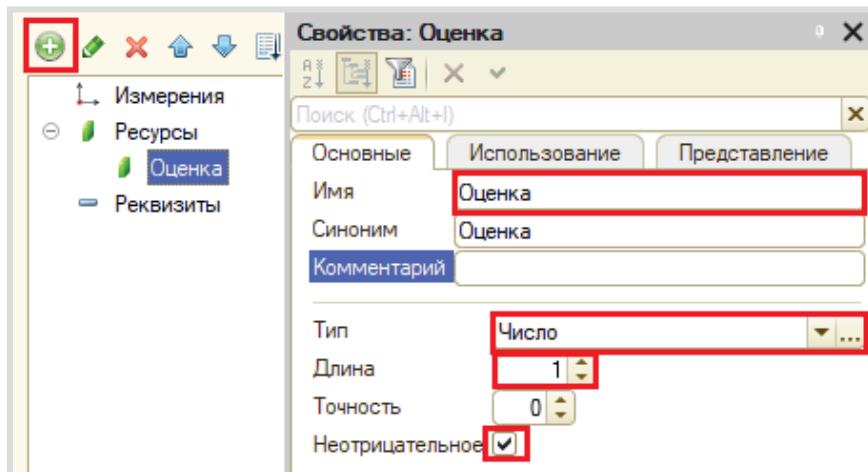


Как и в случае с документами, для формирования структуры переходим на вкладку «Данные».

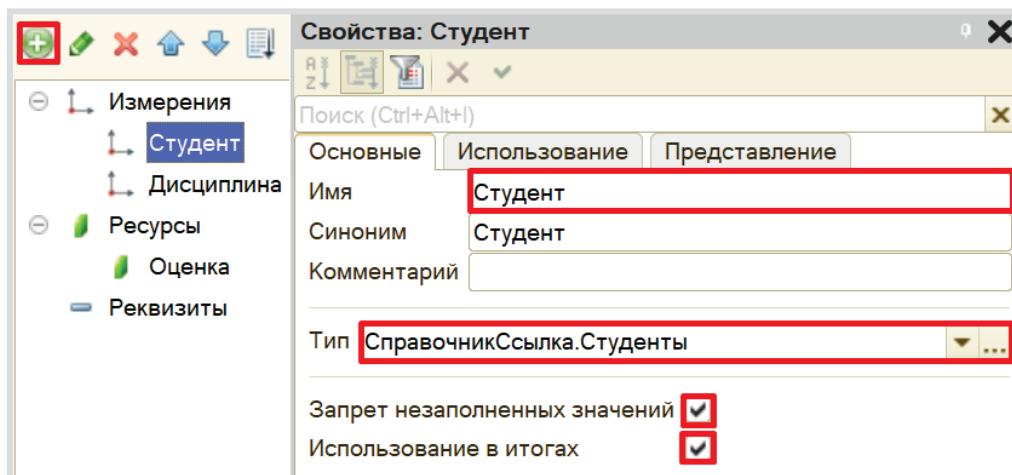


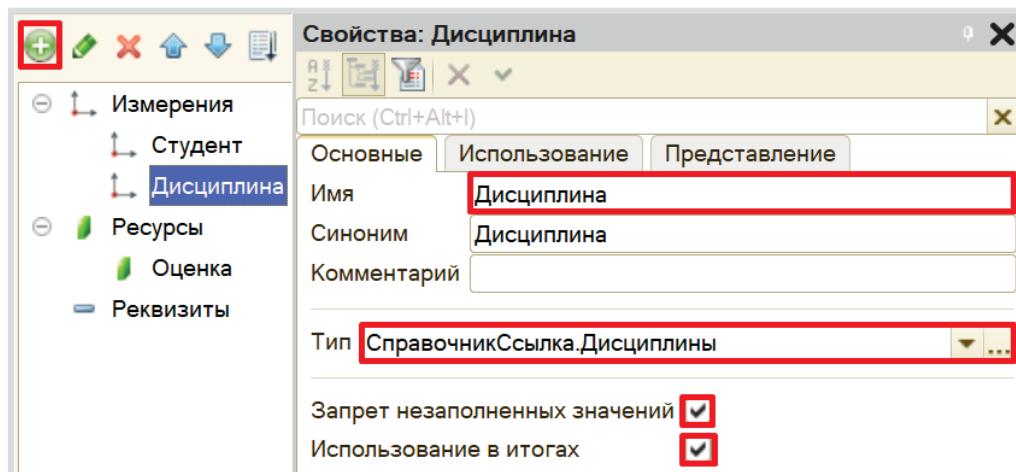
Структура регистра накопления отличается от структуры документа.

Заполнение данного окна проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, следует задать вопрос: «Что мы хотим накапливать/считывать в данном регистре?». Мы хотим считать оценки. Следовательно, оценка и будет являться ресурсом. Тип данного реквизита – «Число».



Чтобы разобраться с измерением, необходимо понять, в разрезе чего мы хотим считать оценки. Мы хотим считать оценки (кого?) студентов в разрезе (чего?) дисциплин. Значит, в качестве измерений нужно добавить реквизиты «Студент» (тип – «СправочникСсылка.Студенты») и «Дисциплина» (тип – «СправочникСсылка.Дисциплины»).

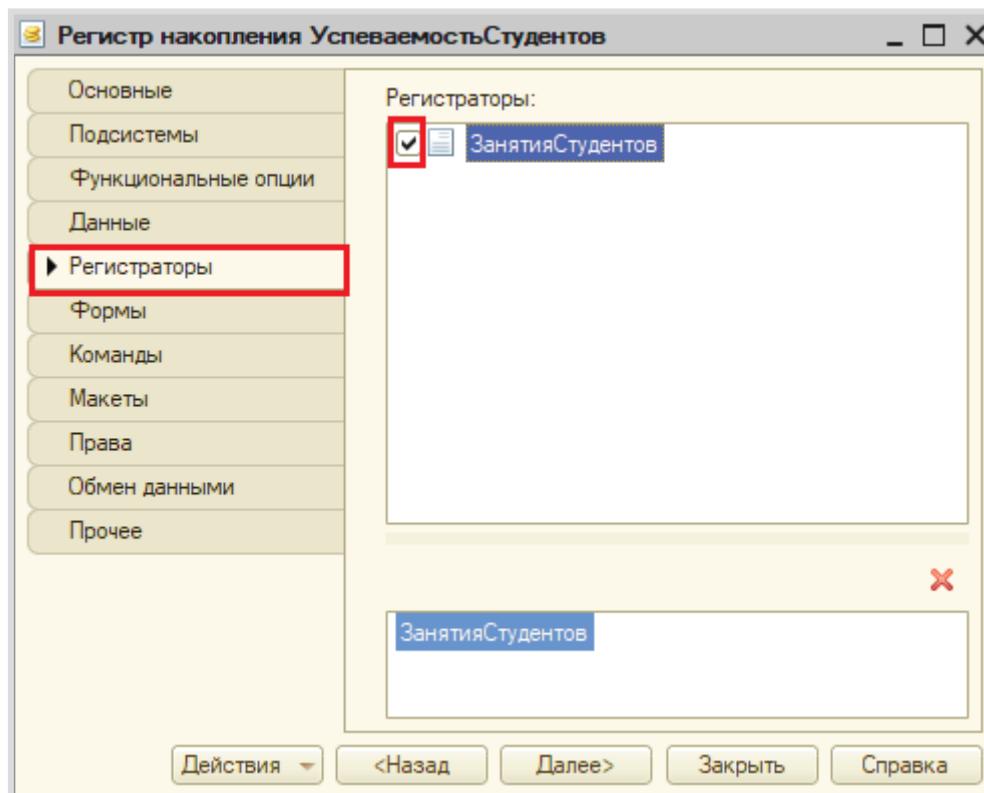




Чтобы *регистр накопления* заработал, необходимо сделать следующее:

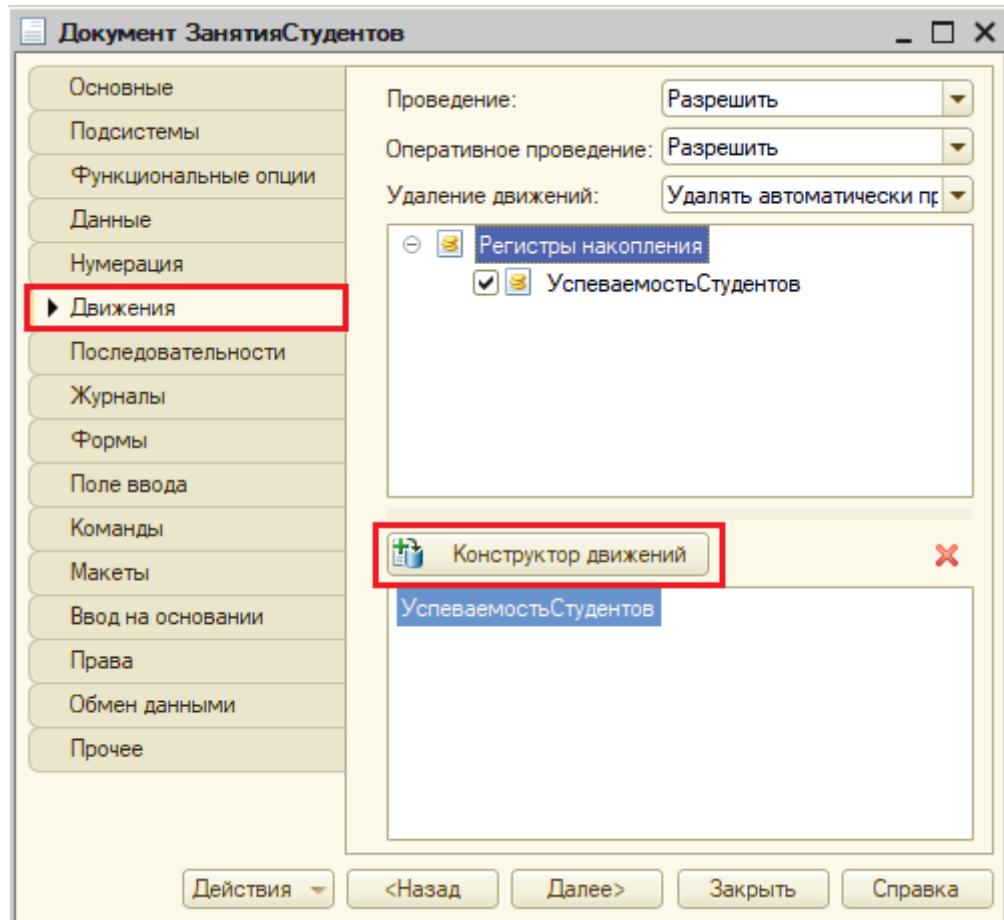
1. Определить источники данных регистра (определить документы-регистраторы).
2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.

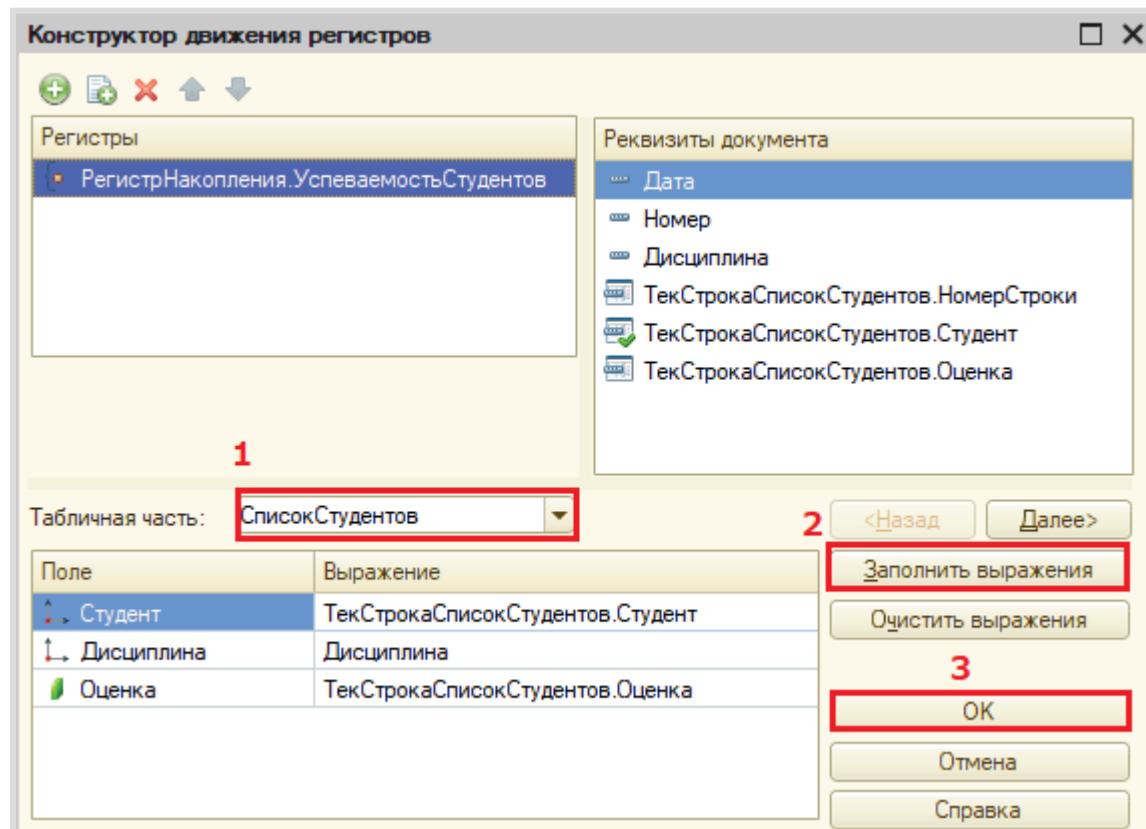
В нашем случае на оценку студента будет влиять единственный документ. Определим его в качестве документа-регистратора на вкладке «Регистраторы».



Далее для нашего документа необходимо описать процедуру копирования данных в *регистр накопления*.

Откроем окно редактирования данного документа на вкладке «Движения». Воспользуемся конструктором движений.

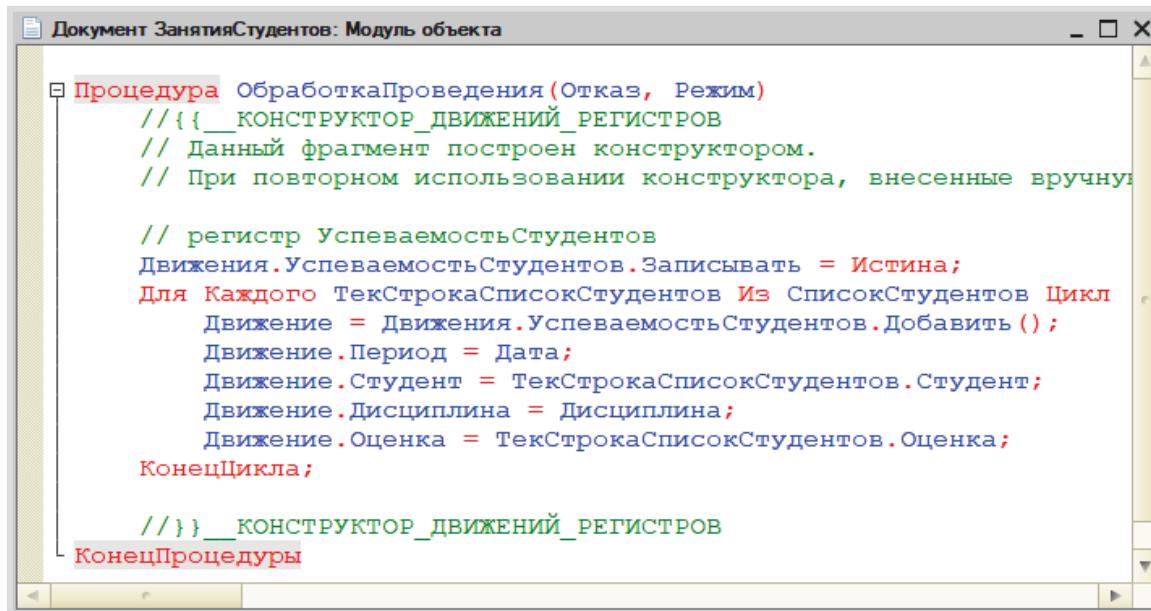




Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты *регистра накопления*. Нужно заполнить поле «Выражение» реквизитами документа.

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.



```

Документ ЗанятияСтудентов: Модуль объекта

Процедура ОбработкаПроведения (Отказ, Режим)
    // {{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    //  Данный фрагмент построен конструктором.
    //  При повторном использовании конструктора, внесенные вручную
    //  изменения не будут потеряны.

    // регистр УспеваемостьСтудентов
    Движения.УспеваемостьСтудентов.Записывать = Истина;
    Для Каждого ТекСтрока СписокСтудентов Из СписокСтудентов Цикл
        Движение = Движения.УспеваемостьСтудентов.Добавить ();
        Движение.Период = Дата;
        Движение.Студент = ТекСтрока СписокСтудентов.Студент;
        Движение.Дисциплина = Дисциплина;
        Движение.Оценка = ТекСтрока СписокСтудентов.Оценка;
    КонецЦикла;

    // }} } __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

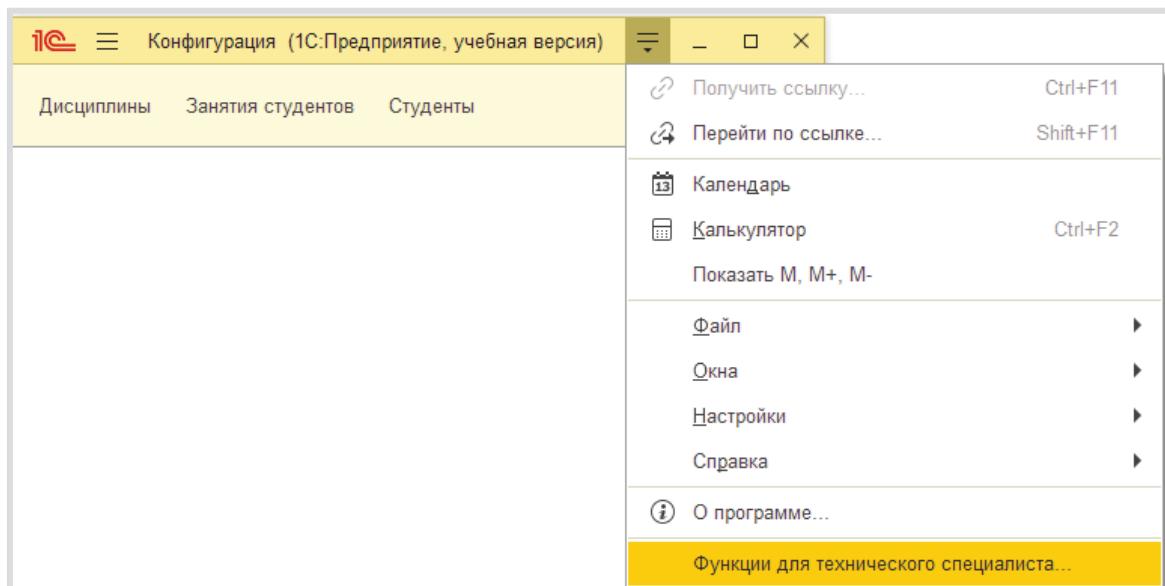
При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

Откроем систему в режиме «1С:Предприятие» и проверим работу *регистра накопления*.

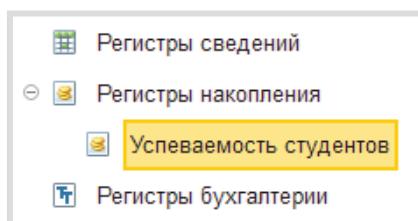
В первую очередь, необходимо перепровести (провести заново) созданный документ «Занятия студентов». Без проведения документов данные не будут скопированы в *регистр накопления*.

Обратите внимание, что на главной странице системы не создала кнопку открытия регистра накопления. Это связано с тем, что все расчеты в регистрах накопления происходят в фоновом режиме, «за кадром», то есть пользователю о них знать не нужно вовсе. Поэтому по умолчанию регистры накопления настраивают так, чтобы пользователи не имели к ним доступа.

Но мы, будучи разработчиками, можем обратиться к любому объекту конфигурации. Для этого воспользуемся функциями для технического специалиста.



В открывшемся списке найдем созданный нами *регистр накопления* и откроем его.



Период	↓	Регистратор	Номер строки	Студент	Дисциплина	Оценка
• 11.09.2020 12:00:00		Занятия студентов ...	4	Федоренко	Операционные си...	3
• 12.09.2020 12:00:00		Занятия студентов ...	1	Любимов	Операционные си...	5
• 12.09.2020 12:00:00		Занятия студентов ...	2	Марковский	Операционные си...	4
• 12.09.2020 12:00:00		Занятия студентов ...	3	Малиновский	Операционные си...	3
• 12.09.2020 12:00:00		Занятия студентов ...	4	Федоренко	Операционные си...	2
• 13.09.2020 20:05:51		Занятия студентов ...	1	Любимов	Высшая математ...	4
• 13.09.2020 20:05:51		Занятия студентов ...	2	Марковский	Высшая математ...	3
• 13.09.2020 20:05:51		Занятия студентов ...	3	Малиновский	Высшая математ...	3
• 13.09.2020 20:05:51		Занятия студентов ...	4	Федоренко	Высшая математ...	2

Таким образом, *регистр накопления* является некоторой итоговой таблицей. Сюда заносятся данные из документов-регистраторов по определенным правилам.

Мы реализовали движение информации об оценках студентов для последующего расчета среднего балла.

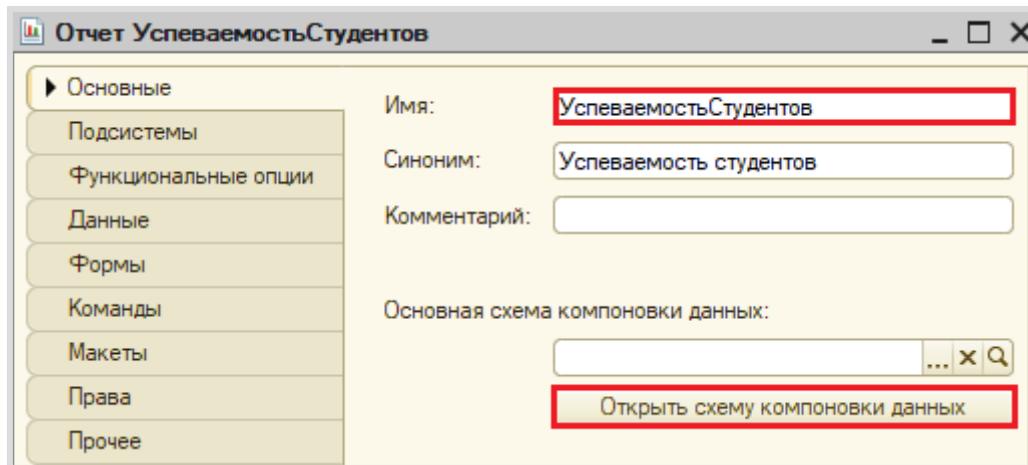
«Необходимо построить Отчет по текущей успеваемости студентов».

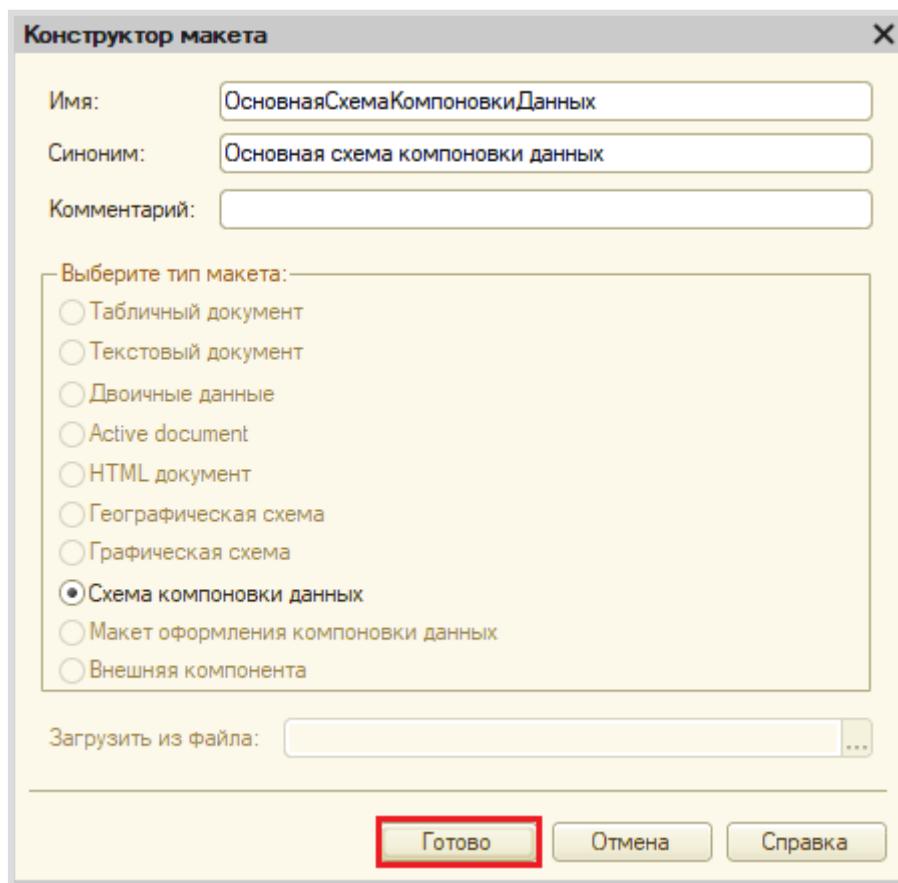
Построим отчет. Для этого воспользуемся соответствующим объектом конфигурации.

Определение

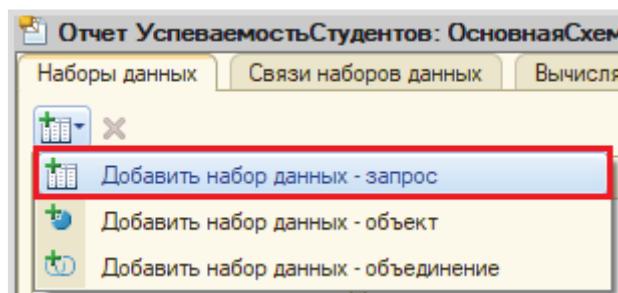
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Добавим отчет «УспеваемостьСтудентов». Воспользуемся схемой компоновки данных.

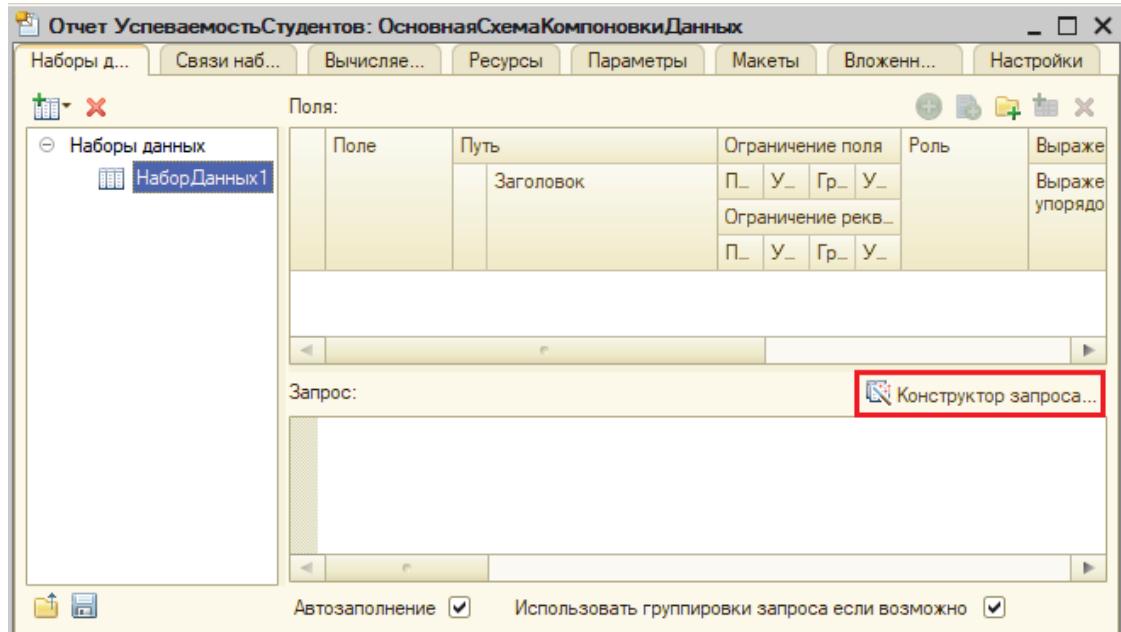




Добавим новый запрос к базе данных.



Для формирования запроса воспользуемся конструктором запроса.



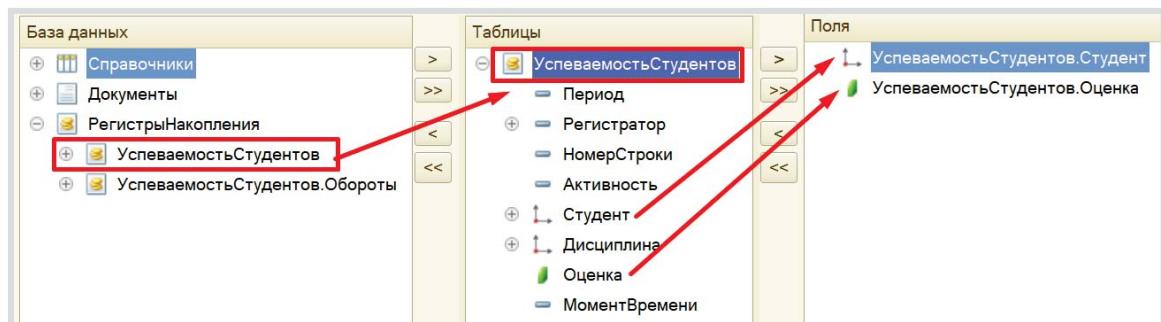
Открывается *конструктор запроса*. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать из *регистра накоплений* напрямую, чтобы иметь возможность рассчитывать средний балл.

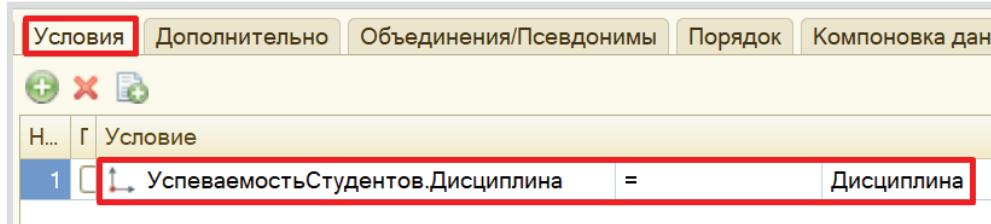
Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



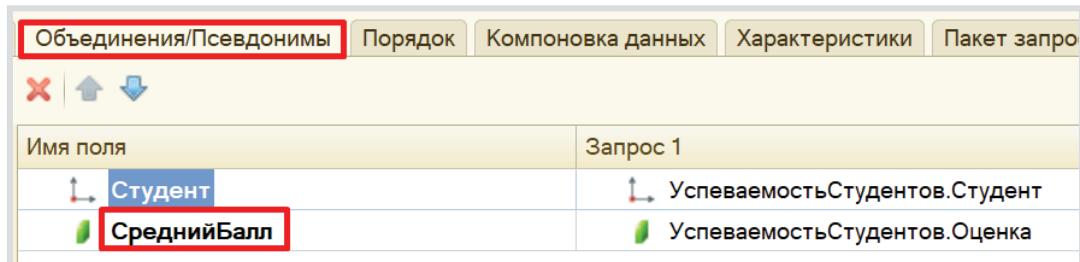
Чтобы иметь возможность получать различные отчеты в зависимости от выбранной дисциплины – перейдем на вкладку «Условия».

Перетащите измерение «Дисциплина» в правую область открывшегося окна и убедитесь, что условие выглядит так же, как на картинке, при необходимости исправьте вручную.

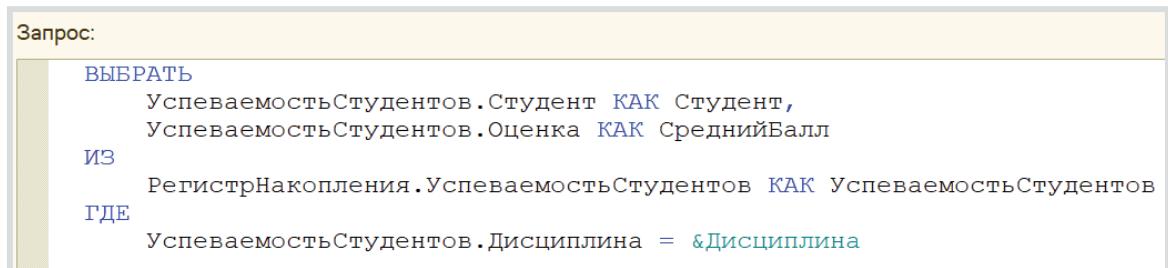


Чтобы отчет получился красивым – установим псевдоним для поля «Оценка» и завершим составление запроса. Для этого следует перейти на вкладку «Объединения и псевдонимы» и изменить имя поля с «Оценка» на «СреднийБалл». Для этого дважды щелкните по имени, должна появиться возможность для редактирования имени.

После изменения псевдонима данное окно должно быть заполнено следующим образом:



Нажмите на кнопку «OK». Система должна сформировать следующий запрос:



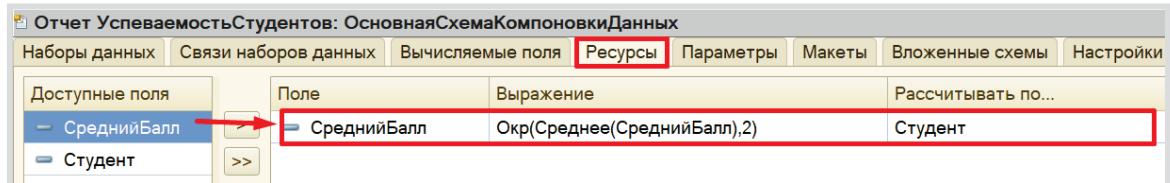
Следующим этапом будет расчет среднего балла для студента.

Для этого перейдем на вкладку «Ресурсы» и установим поле «СреднийБалл» в качестве ресурса. Данное поле должно высчитываться по следующему выражению:

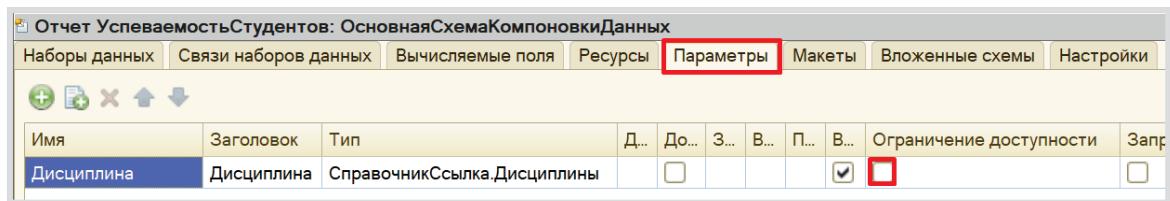
$\text{Окр}(\text{Среднее}(\text{СреднийБалл}), 2)$

С помощью метода « $\text{Окр}(*, 2)$ » мы сможем округлить полученное выражение до сотых.

Кроме того, следует указать, что ресурс должен рассчитываться по студенту.

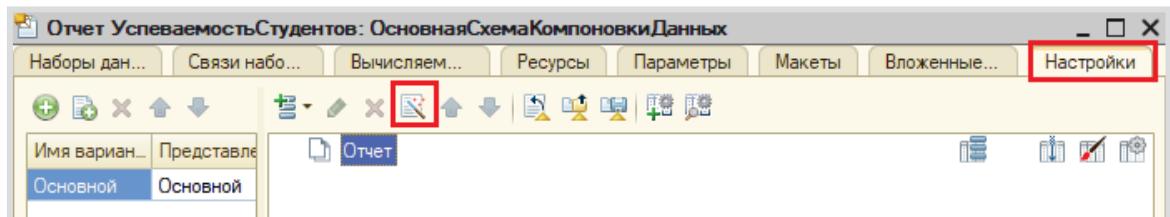


Далее на вкладке «Параметры» нам нужно отключить (снять галочку) ограничение доступности выбора дисциплины в отчете.

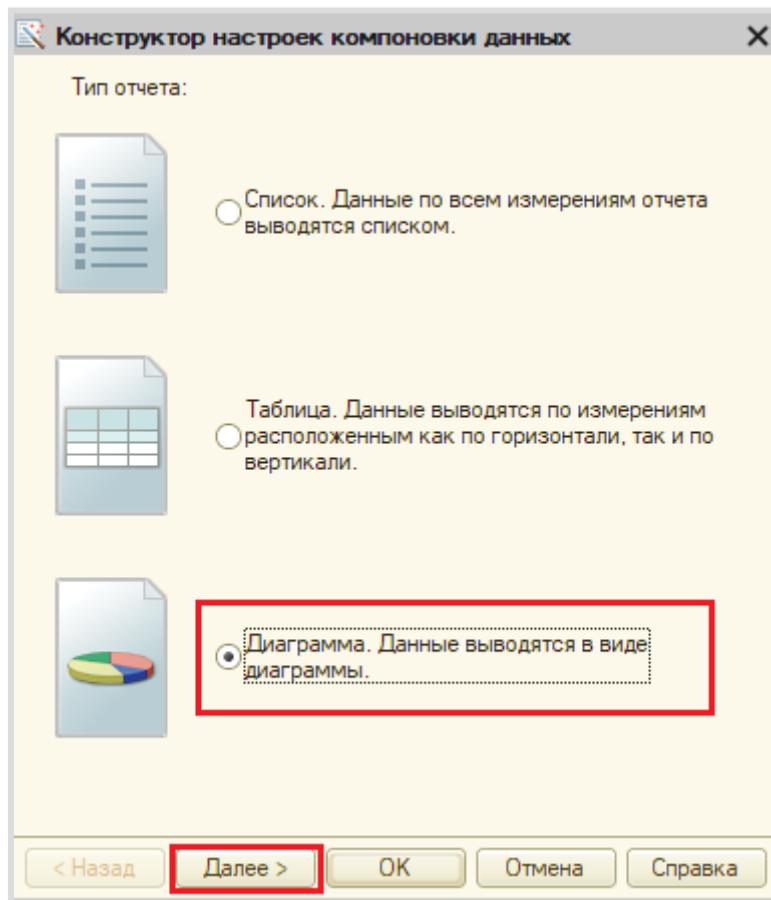


Теперь переходим на вкладку «Настройки» для оформления внешнего вида отчета.

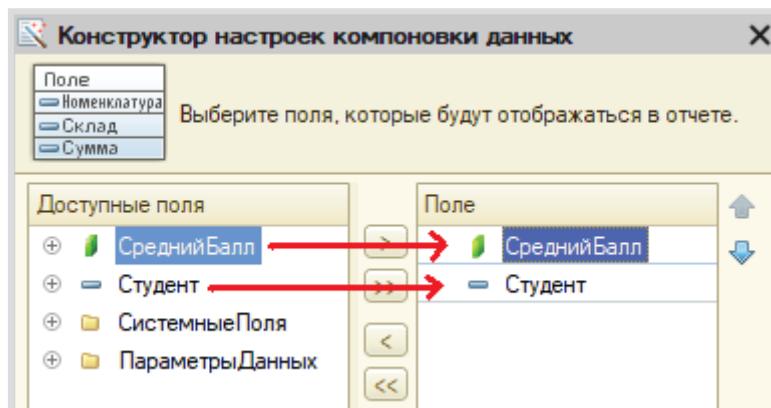
Воспользуемся *конструктором настроек отчета*.



Построим отчет в виде горизонтальной диаграммы.

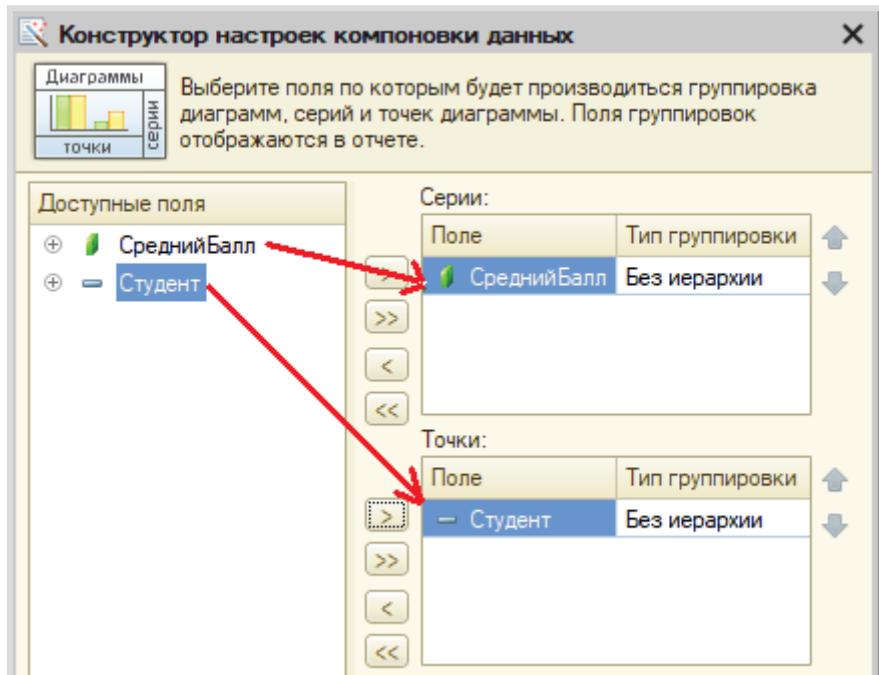


На данном этапе нужно выбрать поля, которые будут отображаться в отчете.



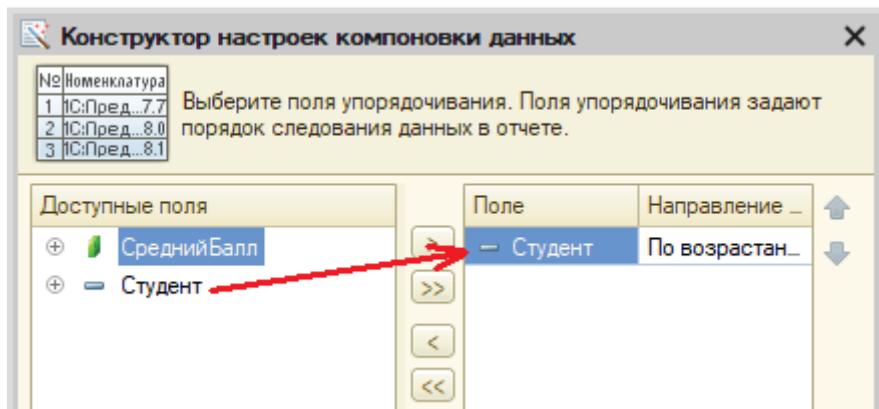
Нажмите на кнопку «Далее».

Теперь нужно определить оси X и Y нашей будущей диаграммы: ось X – это точки, Y – серии. Пусть по оси Y будет указан средний балл, а по оси X – студенты.



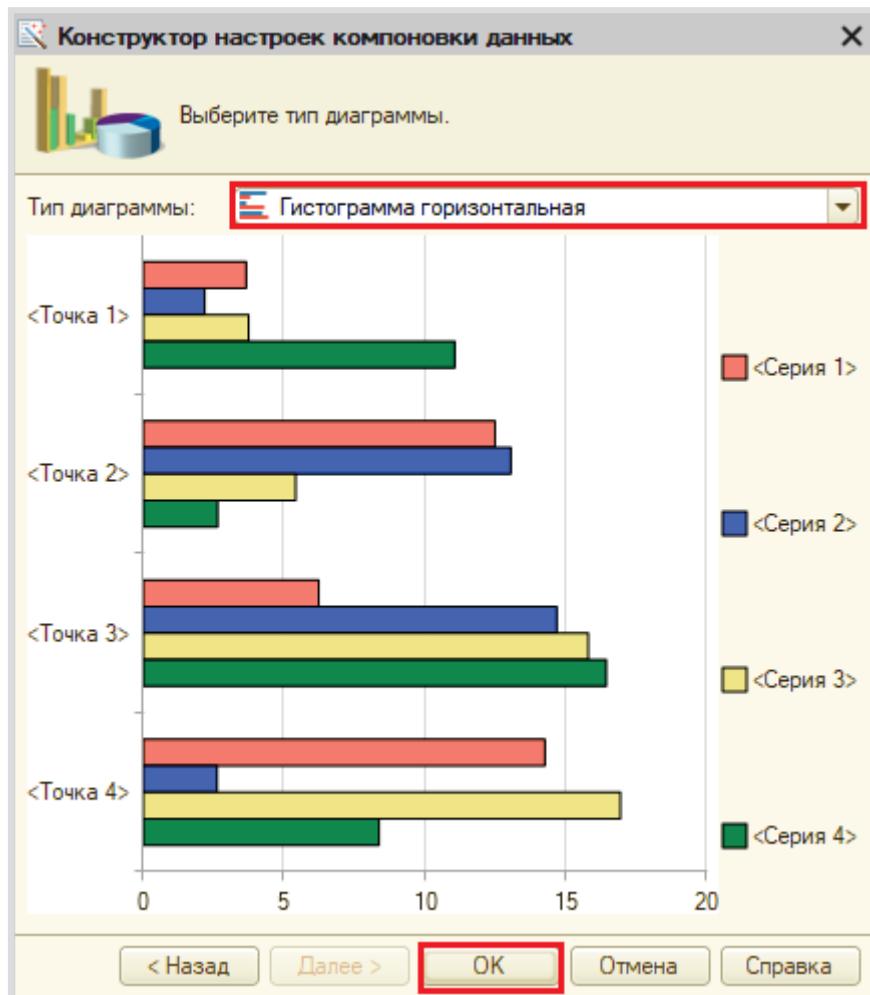
Нажмите на кнопку «Далее».

Чтобы сделать список студентов в отчете по алфавиту – добавим упорядочивание по полю «Студент».



Нажмите на кнопку «Далее».

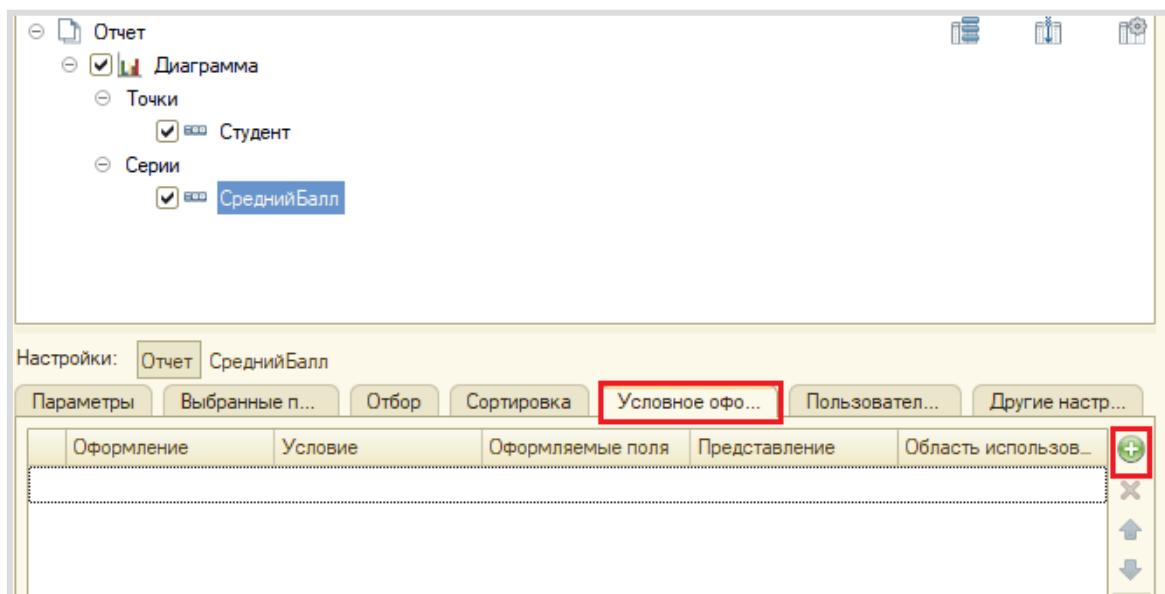
Среди приложенных вариантов диаграмм нужно выбрать вариант «Гистограмма горизонтальная».



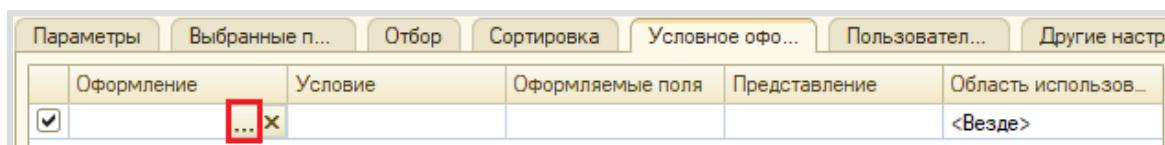
Чтобы у пользователя была возможность выбирать дисциплину, по которой он хочет построить отчет, необходимо включить параметр «Дисциплина» в пользовательские настройки.

Параметр	Значение
Дисциплина	СреднийБалл

Используем условное оформление для того, чтобы сделать отчет более понятным для пользователя. Для этого следует открыть вкладку «Условное оформление». Здесь можно задать оформление, которое будет применено к отчету или его части, когда происходит определенное событие. Добавьте новое условное оформление.



Сначала нужно выбрать оформление. Мы будем выделять различными цветами состояние успеваемости студентов.



В открывшемся окне нас интересует свойство «Цвет в диаграмме». Установим флагок и выберем значение цвета.

Параметр	Значение
<input type="checkbox"/> Цвет фона	[0, 0, 0]
<input type="checkbox"/> Цвет текста	[0, 0, 0]
<input checked="" type="checkbox"/> Цвет в диаграмме	[0, 0, 0]
<input type="checkbox"/> Цвет границы	[0, 0, 0]

Чтобы различать успеваемость студентов будем использовать четыре цвета:

- Зеленый – для отличников;
- Оранжевый – для хорошистов;
- Желтый – для троичников;
- Красный – для двоечников.

Выбор цвета

Из с... Web Wind...

Фон формы
Текст формы
Фон кнопки
Текст кнопки
Фон редактирования
Текст редактирования
Фон выделения редактирования
Текст выделения редактирования
Альтернативный фон редактиров...
Фон подсказки
Текст подсказки
Особый текст
Отрицательное число
Рамка
Фон шапки отчета

Красный: 51
Зеленый: 153
Синий: 102
339966

Взять с экрана Запомнить цвет

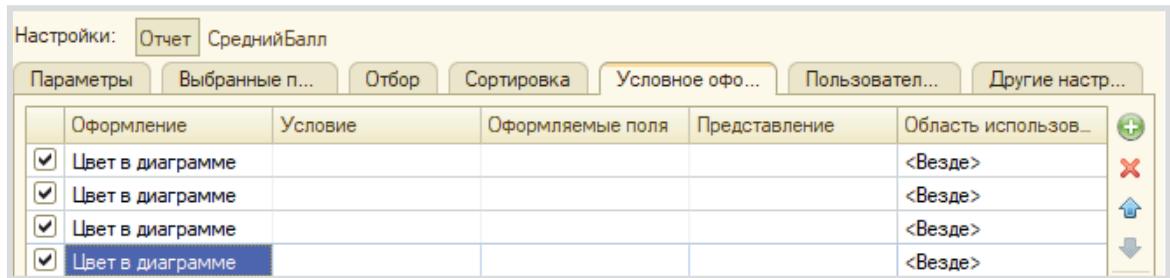
OK Отмена Справка

Редактирование параметров

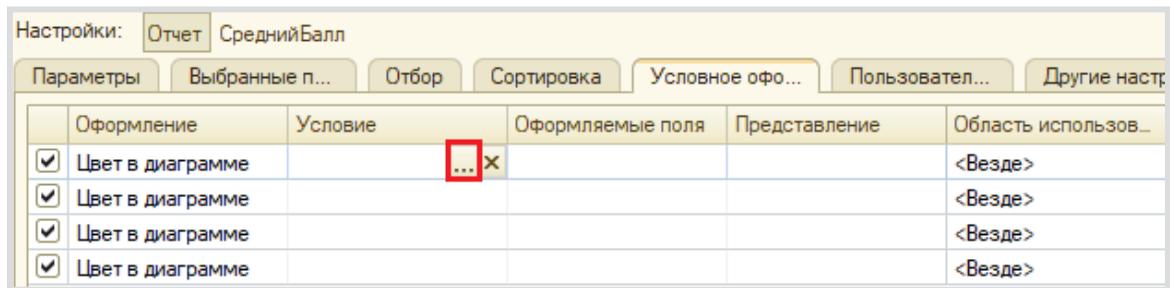
Параметр	Значение
Цвет фона	[0, 0, 0]
Цвет текста	[0, 0, 0]
<input checked="" type="checkbox"/> Цвет в диаграмме	[51, 153, 102]
Цвет границы	[0, 0, 0]
+ Стиль границы	Нет линии
Шрифт	Шрифт диалогов и меню
Отступ	
Автоотступ	
Горизонтальное положение	Прижать влево
Вертикальное положение	Прижать вверх
Размещение	Забивать
Ориентация текста	
Формат	
Выделять отрицательные	Ложь
Отметка незаполненного	Ложь
Минимальная ширина	

OK Отмена Справка

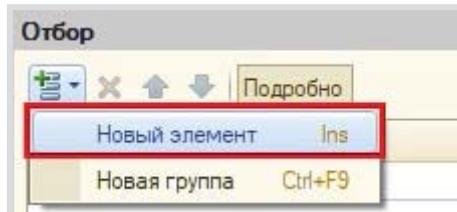
Мы добавили оформление для отличников (RGB: 51, 153, 102 – зеленый цвет). Аналогичным образом добавьте еще три оформления: для хорошистов, троекников и двоечников.



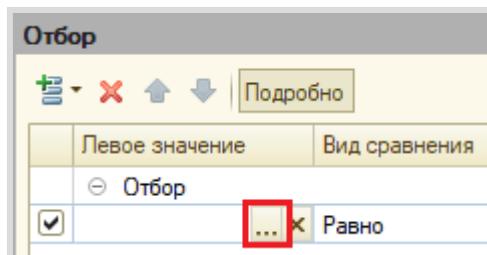
Заключительным этапом будет установка для каждого оформления условия, поскольку в зависимости от разного среднего балла шкала должна окрашиваться разными цветами. Начнем с первого оформления для отличников.



Нужно настроить отбор по среднему баллу. Добавим новый отбор.



В качестве левого сравниваемого значения выберем поле «СреднийБалл».



Вид сравнения установим в значение «больше или равно».

Левое значение	Вид сравнения	Правое значение
⊕ Отбор	Больше или равно	
✓ СреднийБалл	<div style="border: 1px solid #ccc; padding: 5px; width: fit-content;"> Больше или равно ▼ </div> <div style="background-color: #e0f2e0; border: 1px solid #ccc; padding: 5px; margin-top: 5px;"> Равно Не равно Меньше Меньше или равно Больше Больше или равно </div>	

В качестве правого значения выберем значение 4,5.

Левое значение	Вид сравнения	Правое значение
⊕ Отбор		
✓ СреднийБалл	Больше или равно	4,5

Чтобы в дальнейшем понимать, для кого сформировано условие – установим представление «Отличник».

Левое значение	Вид сравнения	Правое значение	Представление
⊕ Отбор			
✓ СреднийБалл	Больше или равно	4,5	Отличник

После чего нажмем на кнопку «OK».

Таким образом, мы настроили условие для первого оформления.

Параметры	Выбранные п...	Отбор	Сортировка	Условное офор...	Пользовател...	Другие настр...
	Оформление	Условие	Оформляемые поля	Представление	Область использо...	
✓ Цвет в диаграмме	Отличник	...			<Везде>	
✓ Цвет в диаграмме					<Везде>	
✓ Цвет в диаграмме					<Везде>	
✓ Цвет в диаграмме					<Везде>	

Далее заполните остальные условия по аналогии.

Хорошист:

	Левое значение	Вид сравнения	Правое значение	Представление
<input type="checkbox"/> Отбор				
<input checked="" type="checkbox"/>	СреднийБалл	Больше или равно	3,5	Хорошист
<input checked="" type="checkbox"/>	СреднийБалл	Меньше	4,5	Хорошист

Троичник:

	Левое значение	Вид сравнения	Правое значение	Представление
<input type="checkbox"/> Отбор				
<input checked="" type="checkbox"/>	СреднийБалл	Больше или равно	2,5	Троичник
<input checked="" type="checkbox"/>	СреднийБалл	Меньше	3,5	Троичник

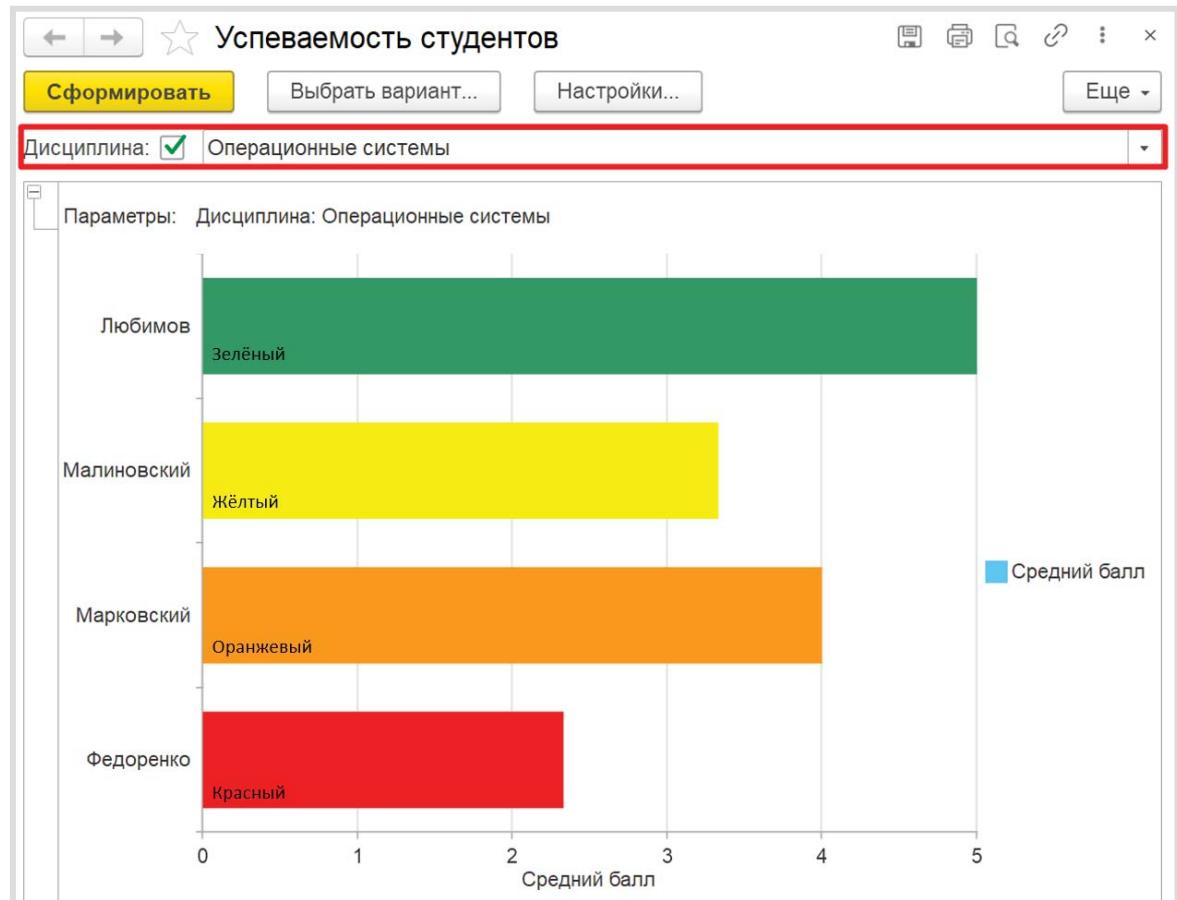
Двоичник:

	Левое значение	Вид сравнения	Правое значение	Представление
<input type="checkbox"/> Отбор				
<input checked="" type="checkbox"/>	СреднийБалл	Меньше	2,5	Двоичник

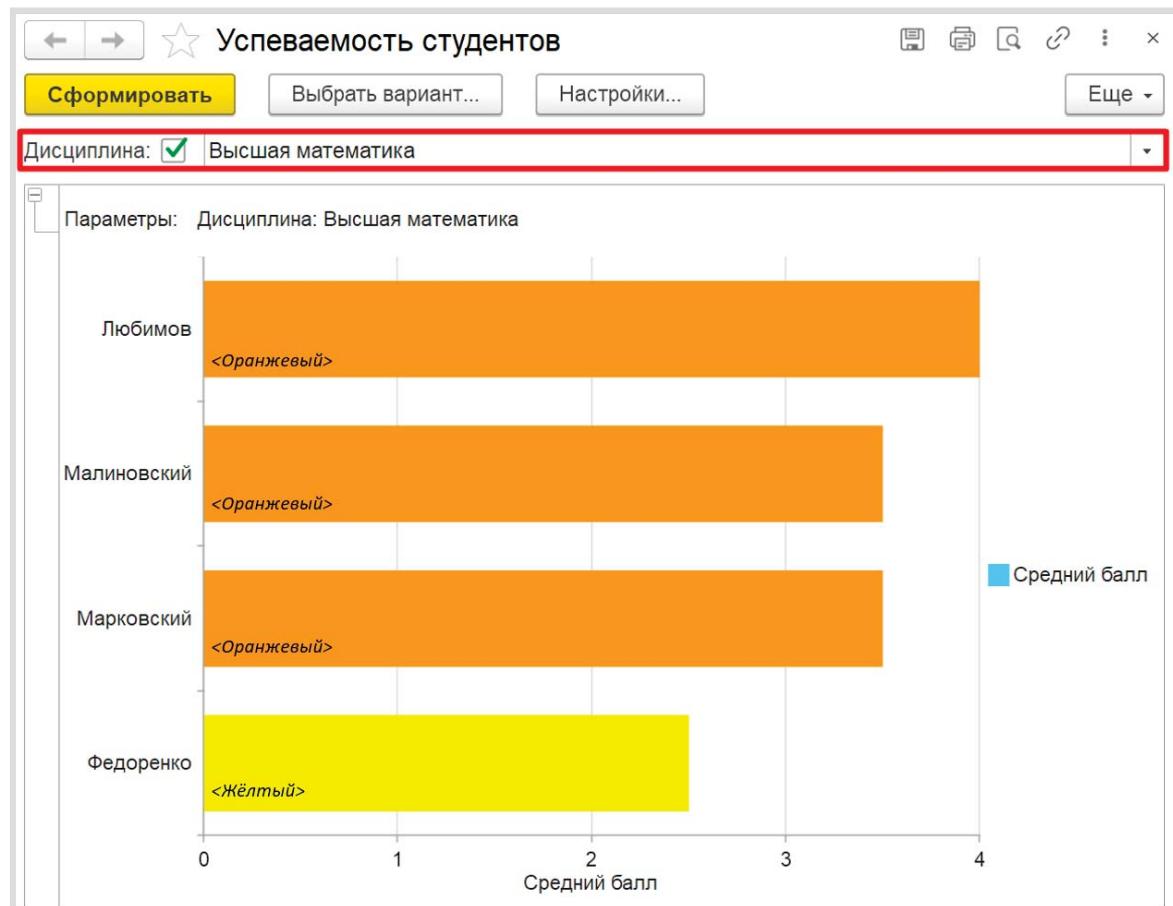
Результат должен получиться следующим:

Параметры	Выбранные п...	Отбор	Сортировка	Условное офор...	Пользовател...	Другие настр...
	Оформление	Условие	Оформляемые ...	Представление	Область использо...	
<input checked="" type="checkbox"/>	Цвет в диаграмме	Отличник			<Везде>	
<input checked="" type="checkbox"/>	Цвет в диаграмме	Хорошист И Хорошист			<Везде>	
<input checked="" type="checkbox"/>	Цвет в диаграмме	Троичник И Троичник			<Везде>	
<input checked="" type="checkbox"/>	Цвет в диаграмме	Двоичник			<Везде>	

Отчет готов. Запустим систему в режиме «1С:Предприятие».



В зависимости от выбранной дисциплины будет меняться отчет.



Поставленная задача решена.

Лабораторная работа № 11

**АВТОМАТИЗИРОВАТЬ
СИСТЕМУ ПУНКТА ПРОКАТА
ЭЛЕКТРОСАМОКАТОВ
В УЧЕБНОМ ЗАВЕДЕНИИ**

Сложность: *

Теги: документы, события документов, регистры накопления остатков, запросы к виртуальным таблицам

ЗАДАЧА

Заказчик просит автоматизировать систему пункта проката электросамокатов в учебном заведении.

Нужно фиксировать в информационной системе, какой студент забрал или вернул самокат. Выдача и возврат должны быть фиксироваться отдельно, причем количество самокатов учитывать нет необходимости, поскольку каждый студент может арендовать (и, соответственно, сдать) не более одного самоката.

В результате выполнения лабораторной работы должен получиться отчет вида:

АРЕНДАТОР
Иванов Иван Иванович
Петров Петр Петрович

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

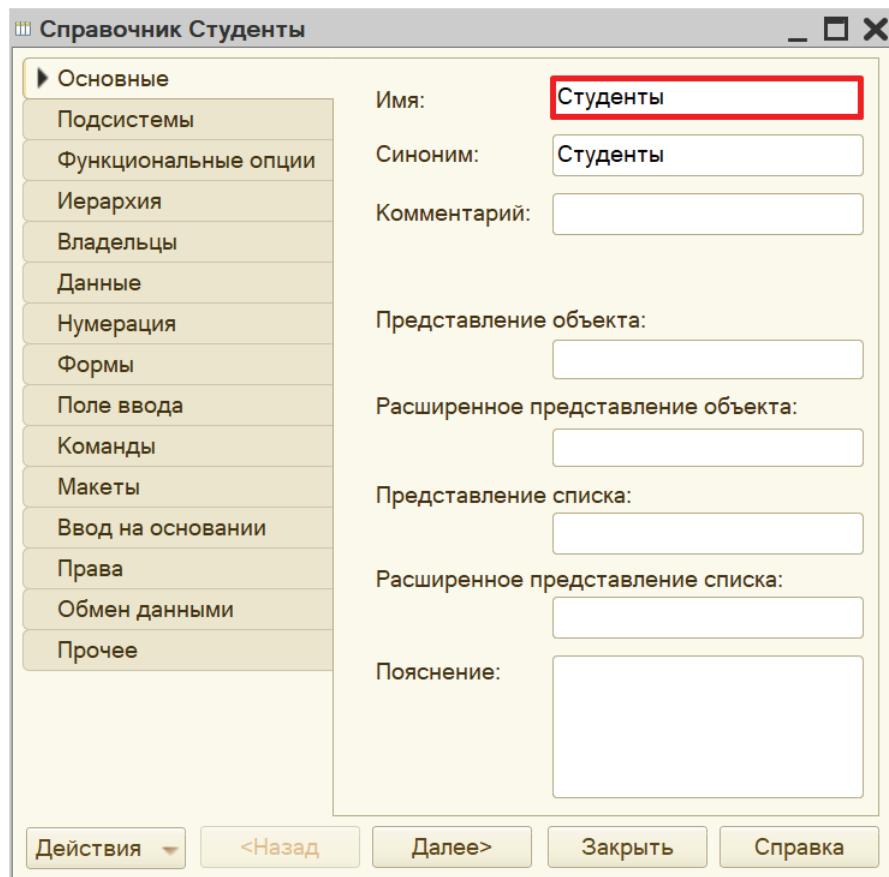
Выполнение

Начнем выполнение лабораторной работы с создания справочников.

Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

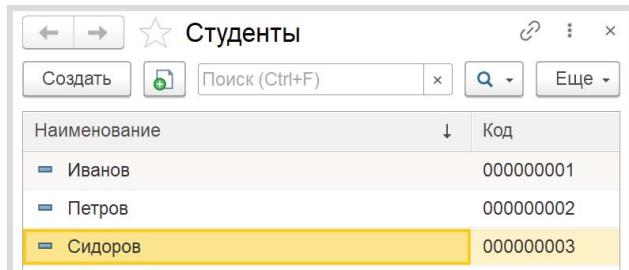
Создадим справочник, в котором будет храниться список студентов, которые могут арендовать электросамокат, и назовем его «Студенты».



Для решения данной задачи другие объекты аналитики нам не понадобятся.

Запустим программу в режиме «1С:Предприятие» и добавим несколько студентов.

Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



Таким образом, было организовано хранение в системе списка студентов.

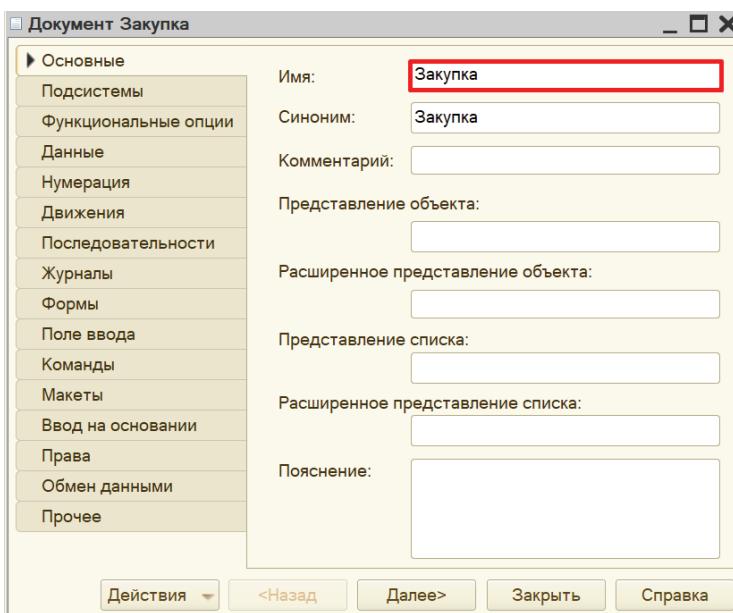
Далее нужно реализовать закупку электросамокатов для точки проката.

Для данной цели используем объект конфигурации *документ*.

Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

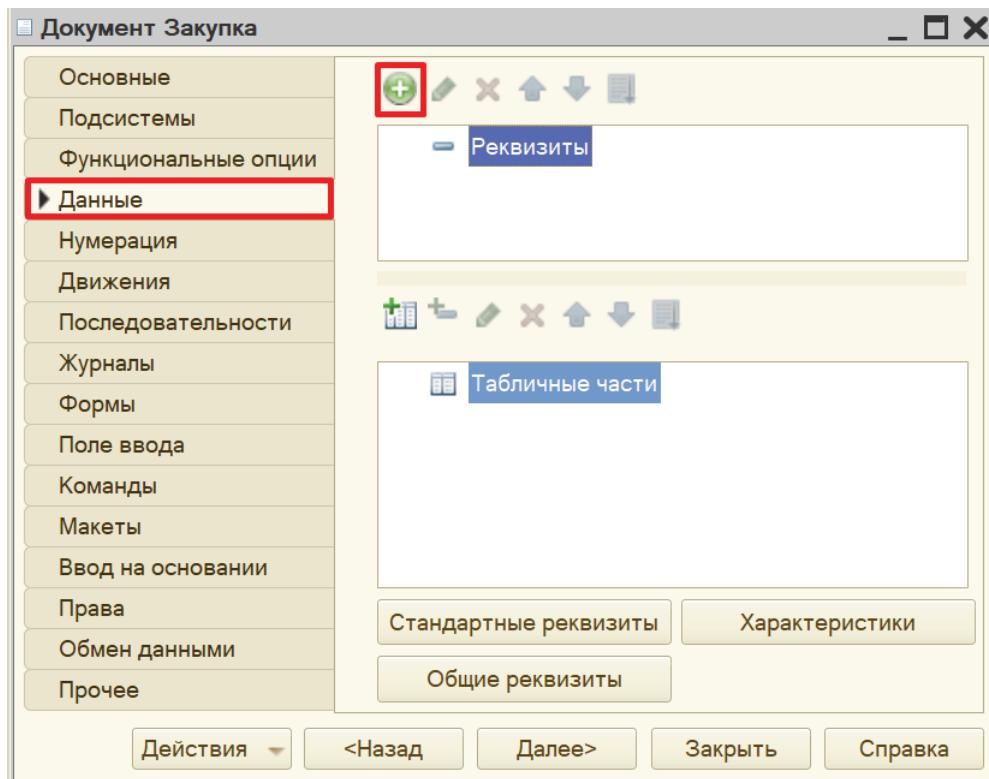
Создадим документ «Закупка».

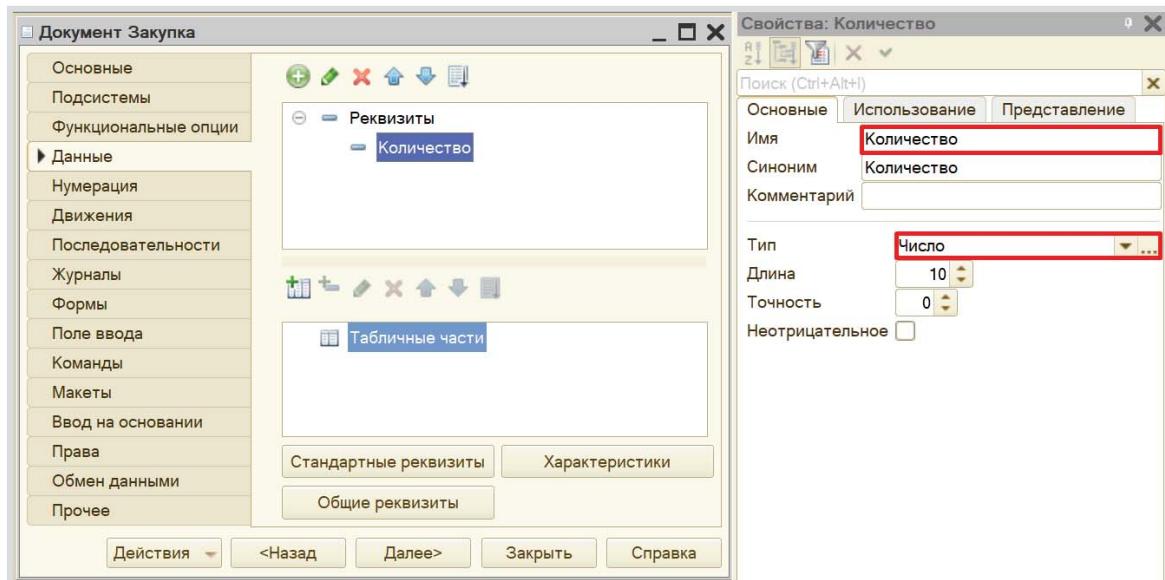


Для формирования структуры документа переходим на вкладку «Данные».

Все самокаты в нашем прокате совершенно одинаковы, поэтому хранить данные о самих самокатах нам не нужно. Важно лишь учитывать, сколько таких самокатов было закуплено для проката.

Для этого добавим новый реквизит документа «Количество» с помощью кнопки «Добавить». Тип данных – «Число».





Откроем программу в режиме «1С:Предприятие» и зафиксируем закупку нескольких электросамокатов.

Легко заметить, что система сгенерировала для документа другие стандартные реквизиты: «Номер» и «Дата». Оба поля заполняются автоматически, дата может быть изменена.

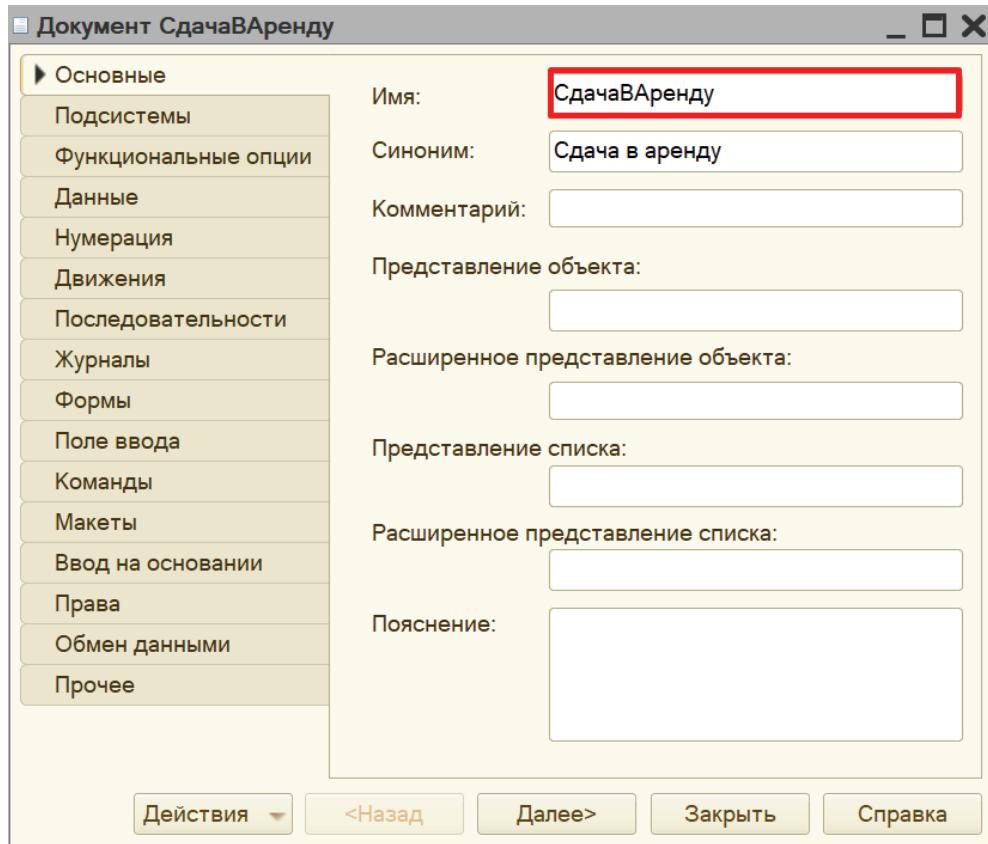
Любой документ может находиться в одном из двух состояний: *подготовленный к свершению* или *совершенный*:

- чтобы подготовить документ для использования в будущем, нужно его записать;
- чтобы отметить документ как совершенный – провести.

Таким образом, документ для регистрации закупки электросамокатов был успешно реализован.

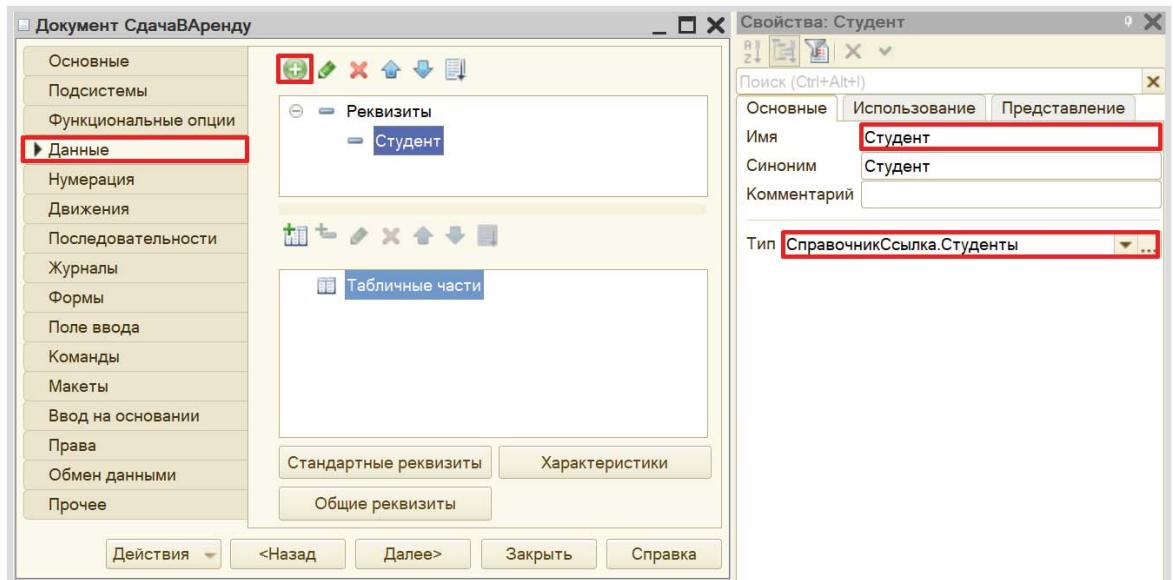
Следующий шаг – это регистрация операции по выдаче электросамокатов в аренду.

Создадим документ «Сдача В Аренду».



Переходим на вкладку «Данные» для формирования структуры документа.

В данном документе нужно фиксировать дату и время, а также ФИО студента, арендовавшего самокат. Поле «Дата» будет создано системой автоматически, а вот поле для ввода ФИО студента отсутствует. Добавим реквизит «Студент» с типом «Справочник Ссылка.Студенты». Данный реквизит будет содержать ссылку на объект справочника «Студенты».



Проверим работоспособность документа в режиме «1С:Предприятие».

Сдача в аренду

Создать Поиск (Ctrl+F) Еще

Дата	Номер	Студент
------	-------	---------

Сдача в аренду (создание)

Провести и закрыть Записать Провести Еще

Номер:

Дата: 04.09.2020 0:00:00

Студент:

Ведите строку для поиска
Нажмите [Показать все](#) для выбора
Нажмите [+ \(создать\)](#) для добавления

Студенты

Наименование	Код
Иванов	000000001
Петров	000000002
Сидоров	000000003

Сдача в аренду (создание) *

Номер:	<input type="text"/>
Дата:	04.09.2020 0:00:00 <input type="button" value=""/>
Студент:	Иванов <input type="button" value=""/>

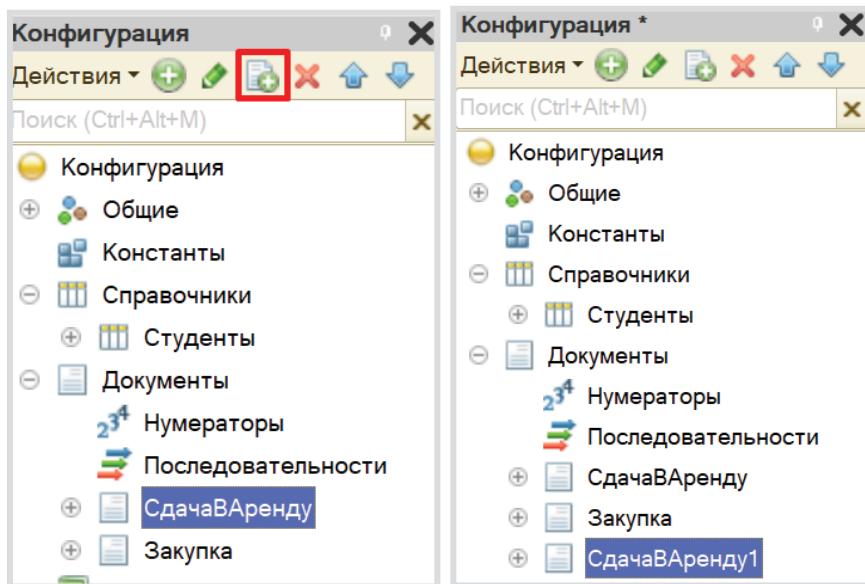
Сдача в аренду

Дата	Номер	Студент
04.09.2020 19:06:16	000000001	Иванов
04.09.2020 19:06:20	000000002	Сидоров
04.09.2020 19:17:39	000000003	Петров

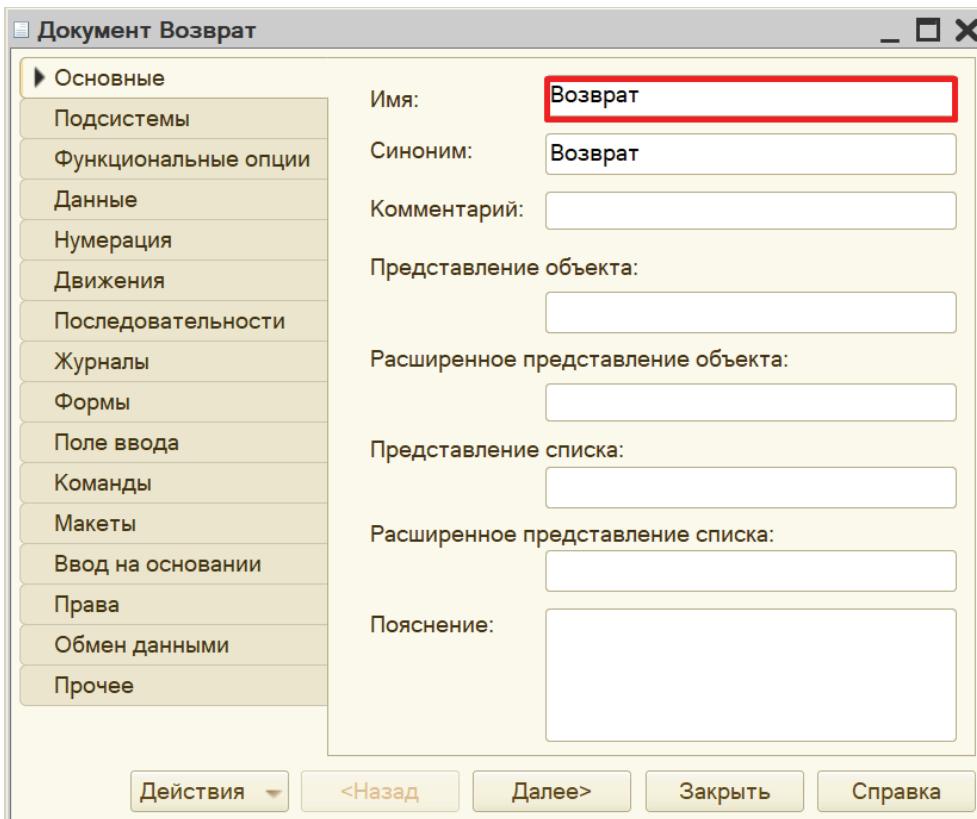
Теперь информационная система способна регистрировать дату и ФИО студента, арендовавшего самокат.

Регистрация возврата самокатов будет выглядеть аналогично регистрации сдачи в аренду.

Создадим новый документ «Возврат», копируя документ «Сдача В Аренду». Для этого выберем нужный документ в окне конфигурации и нажмем на кнопку «Добавить копированием».



Откроем окно редактирования документа «Сдача В Аренду1» и изменим название документа на «Возврат».



Чтобы убедиться в том, что все было сделано правильно, перейдите на вкладку «Данные» и проверьте, что у данного документа существует реквизит «Студент».

Созданная нами информационная система может хранить информацию о закупке электросамокатов, факте сдачи самоката в аренду конкретному студенту, а также факт возврата самоката студентом.

Предположим, что с помощью документа «Закупка» мы зарегистрировали в системе 12 электросамокатов. Если Иванов возьмет один из самокатов в аренду, то в прокате должно остаться 11. По возвращении самоката в прокате снова должно быть 12 самокатов.

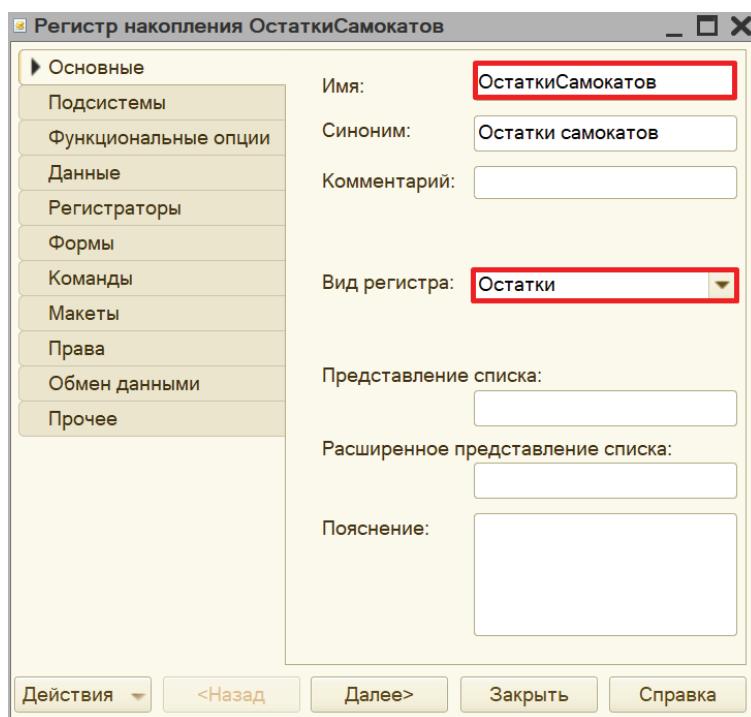
Но на деле все введенные нами документы никак между собой не связаны. Нам следует использовать такой объект, который сможет связать между собой данные документы, а также будет накапливать итоговые значения для расчета остатков. Такой объект называется *регистром накопления*.

Определение

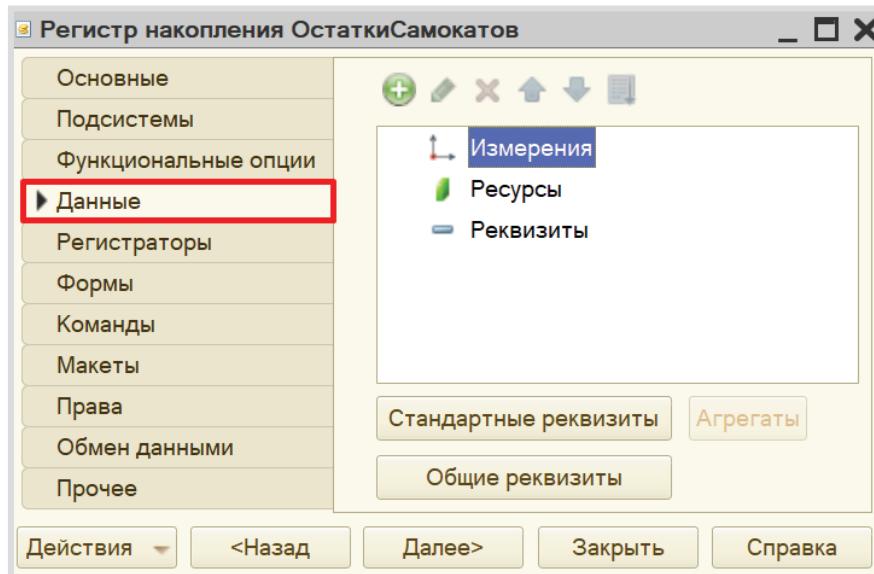
Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Добавим новый *регистр накопления* «ОстаткиСамокатов» вида «Остатки».

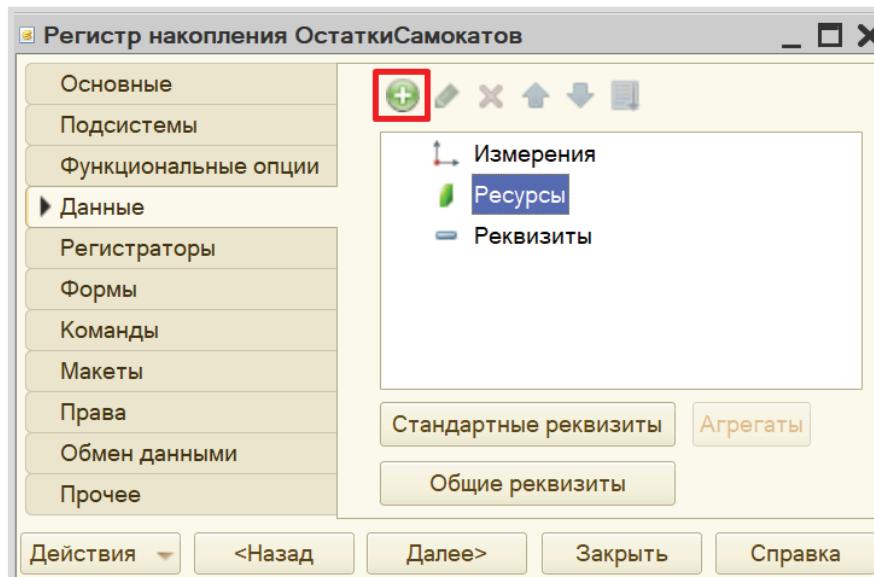
Вид регистра «Остатки» позволяет настроить данный регистр таким образом, что какие-то объекты будут вносить в него данные, а какие-то, наоборот, вычитать из него. Таким образом и получается хранение остатков.

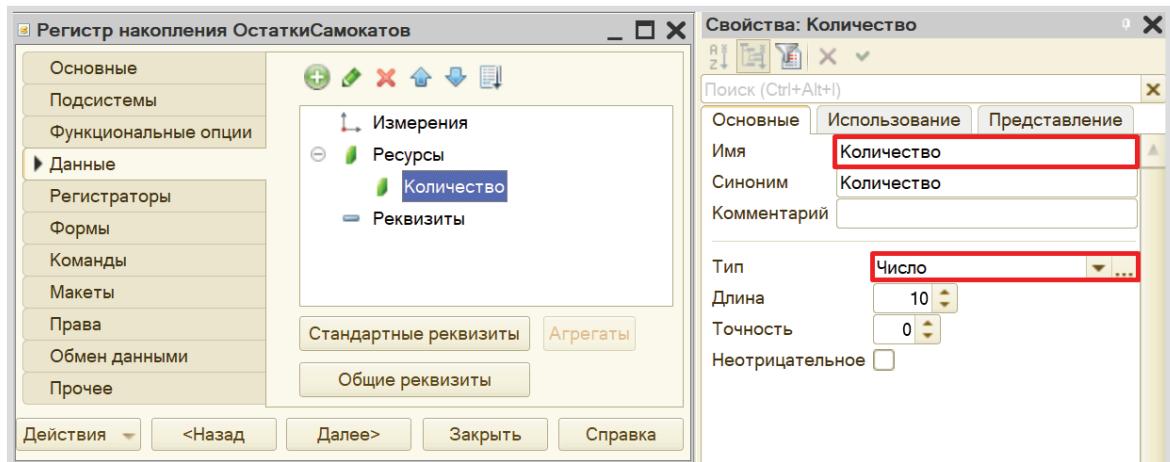


Открываем вкладку «Данные» для формирования структуры *регистра накопления*.

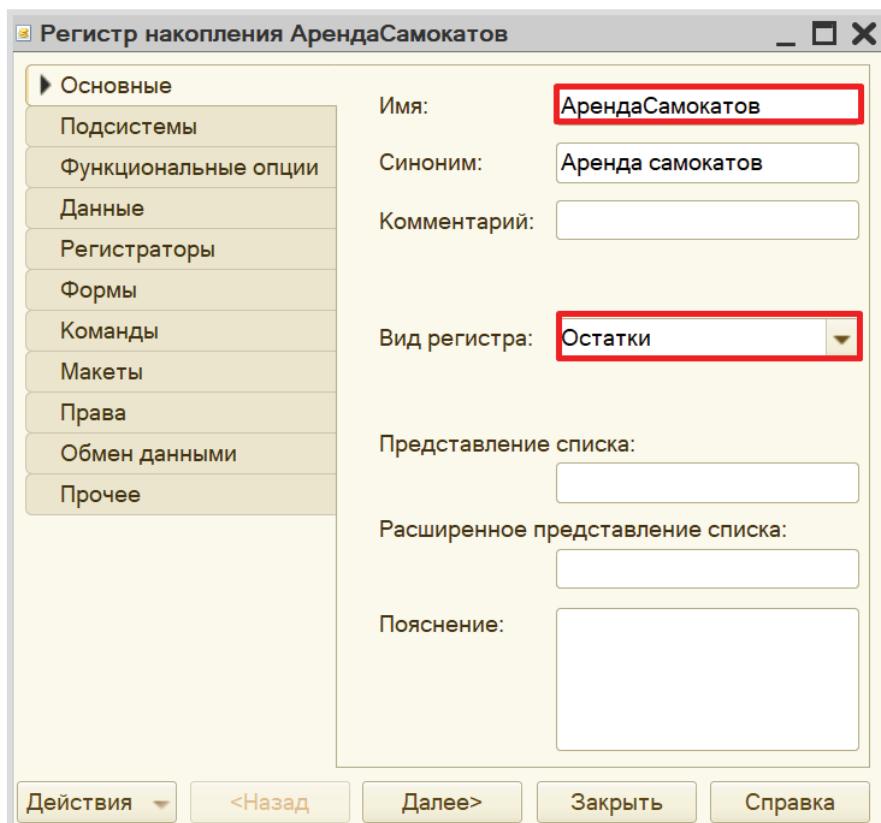


Добавим ресурс. Что мы хотим считать с помощью данного регистра? Мы хотим считать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число».

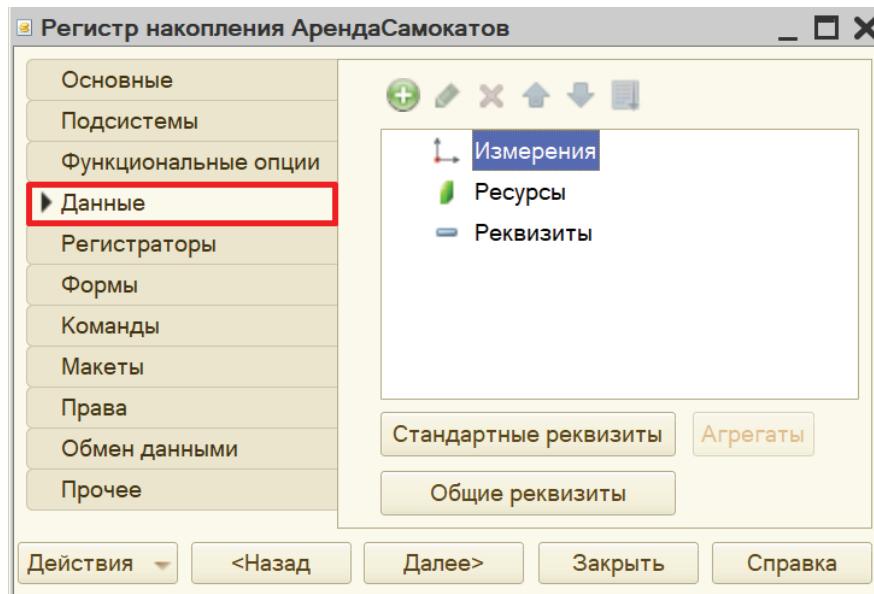




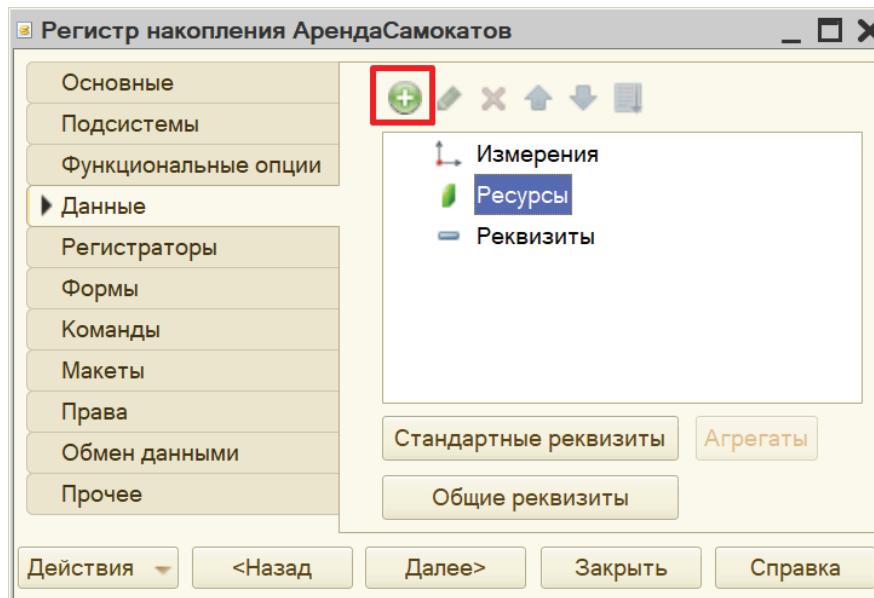
Необходимо создать еще один *регистр накопления* «АрендаСамокатов» вида «Остатки». Он уже будет учитывать, кто взял или вернул самокат.

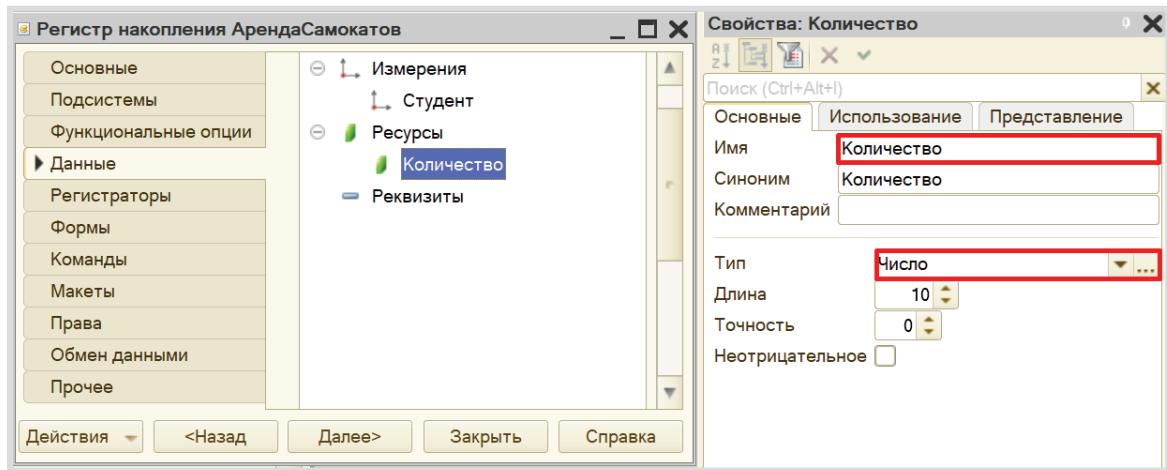


Открываем вкладку «Данные» для формирования структуры *регистра накопления*.

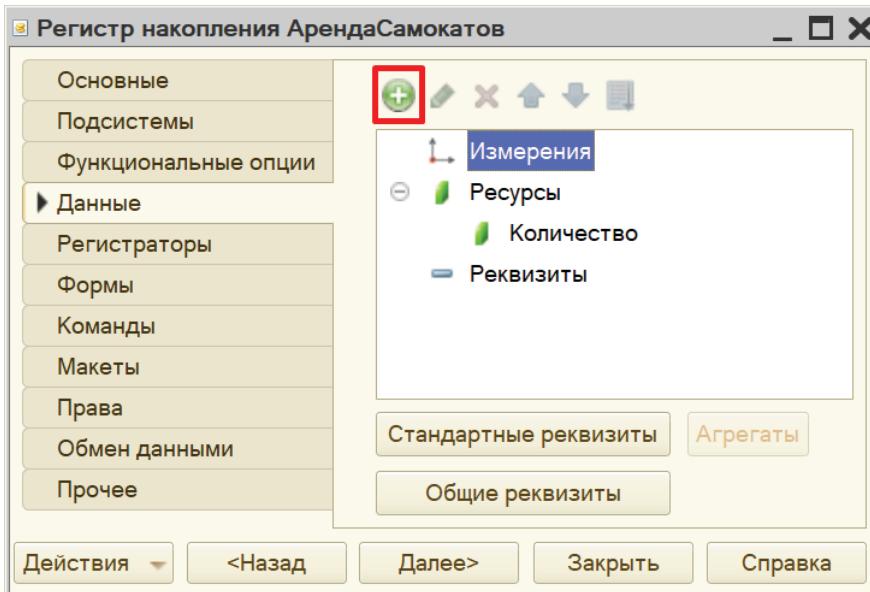


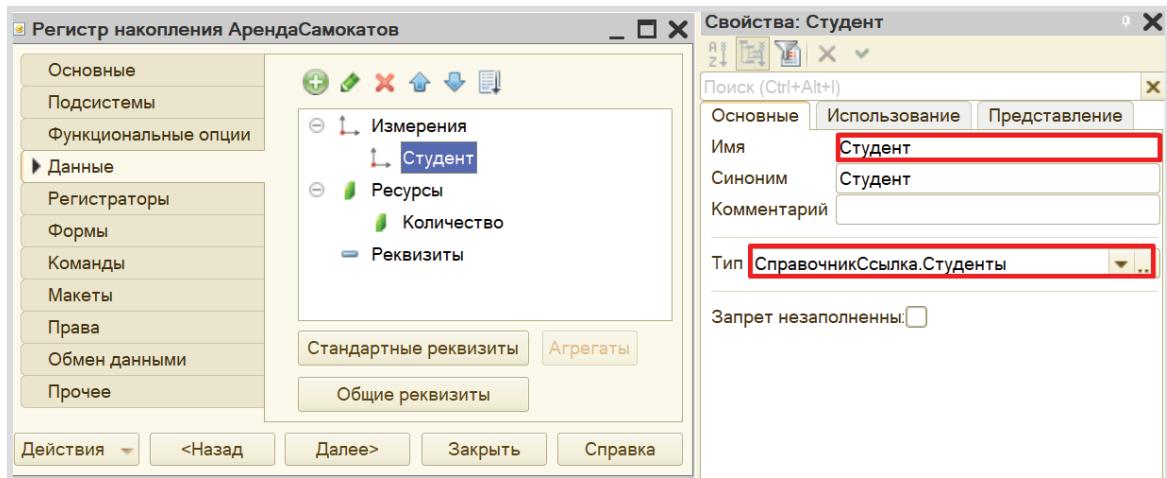
В данном *регистре накопления* (по аналогии с предыдущим) в качестве ресурса выступает реквизит «Количество», тип – «Число».





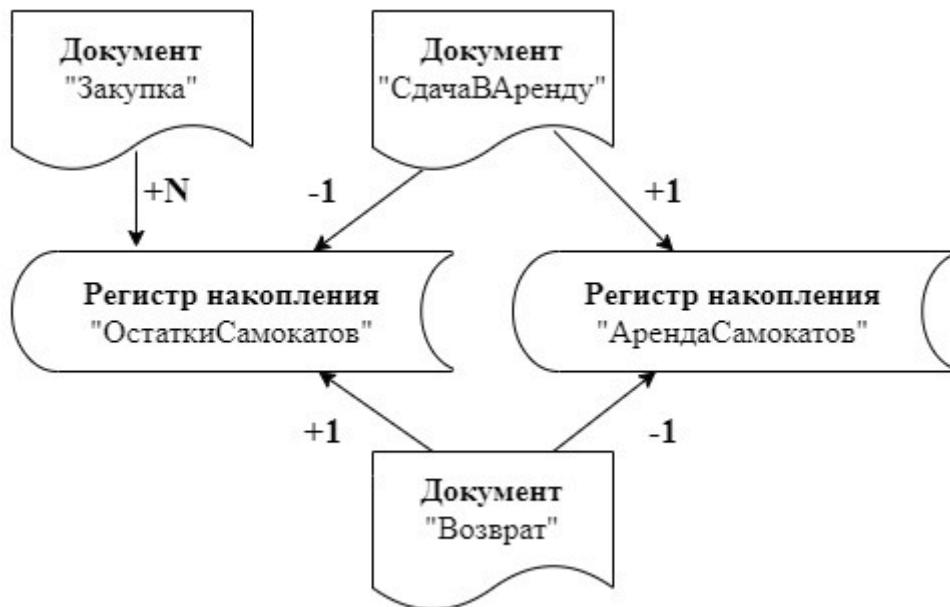
Чтобы разобраться с измерением, следует понять, в разрезе чего мы хотим считать количество. Мы хотим учитывать, какому студенту сдали самокат в аренду. Значит, в качестве измерения нужно добавить реквизит «Студент». Тип данного реквизита – «СправочникСсылка.Студенты».





Обратите внимание, что типы данных этих регистров совпадают с типами данных регистров, которые были добавлены в документы.

Взаимосвязь созданных документов и *регистров накопления* должна выглядеть следующим образом:



При проведении документа «Закупка» в *регистр накопления «Остатки Самокатов»* будет передано то количество самокатов, которое будет указано в документе.

При проведении документа «Сдача В Аренду» будет производиться вычет одного самоката из *регистра накопления* «ОстаткиСамокатов», но при этом студент, взявший самокат, будет внесен в регистр «Аренда Самокатов» в качестве арендатора.

Документ «Возврат» будет делать все наоборот: добавлять единицу в «ОстаткиСамокатов» и убирать студента из списка должников из реквизита «Аренда Самокатов».

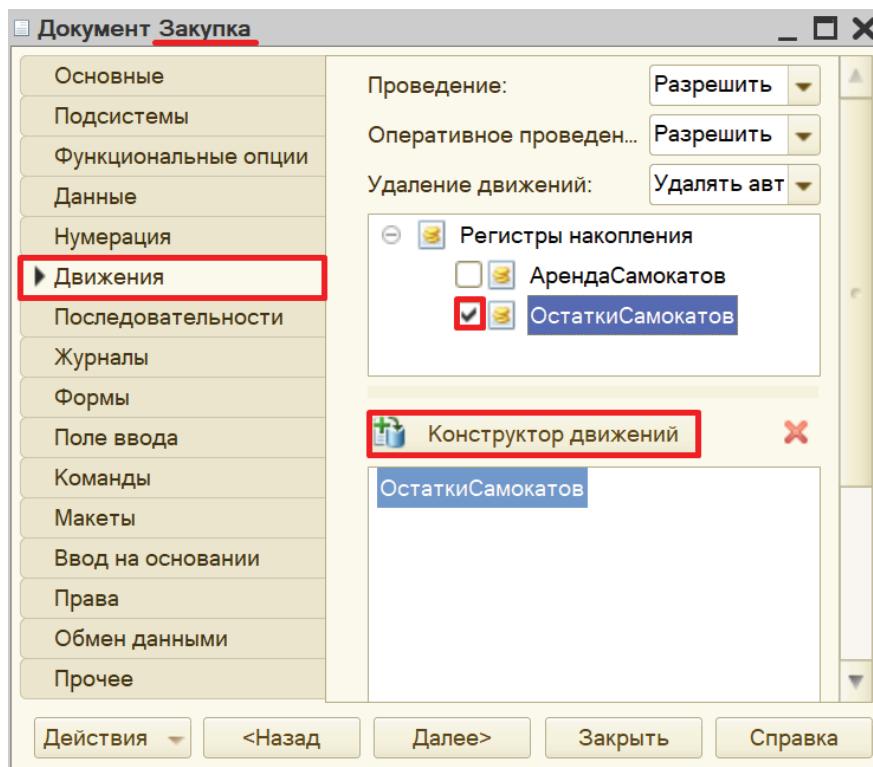
Таким образом, количество электросамокатов в *регистре накопления* «ОстаткиСамокатов» должно соответствовать текущему количеству самокатов на складе. А регистр накопления «Аренда Самокатов» должен содержать список должников.

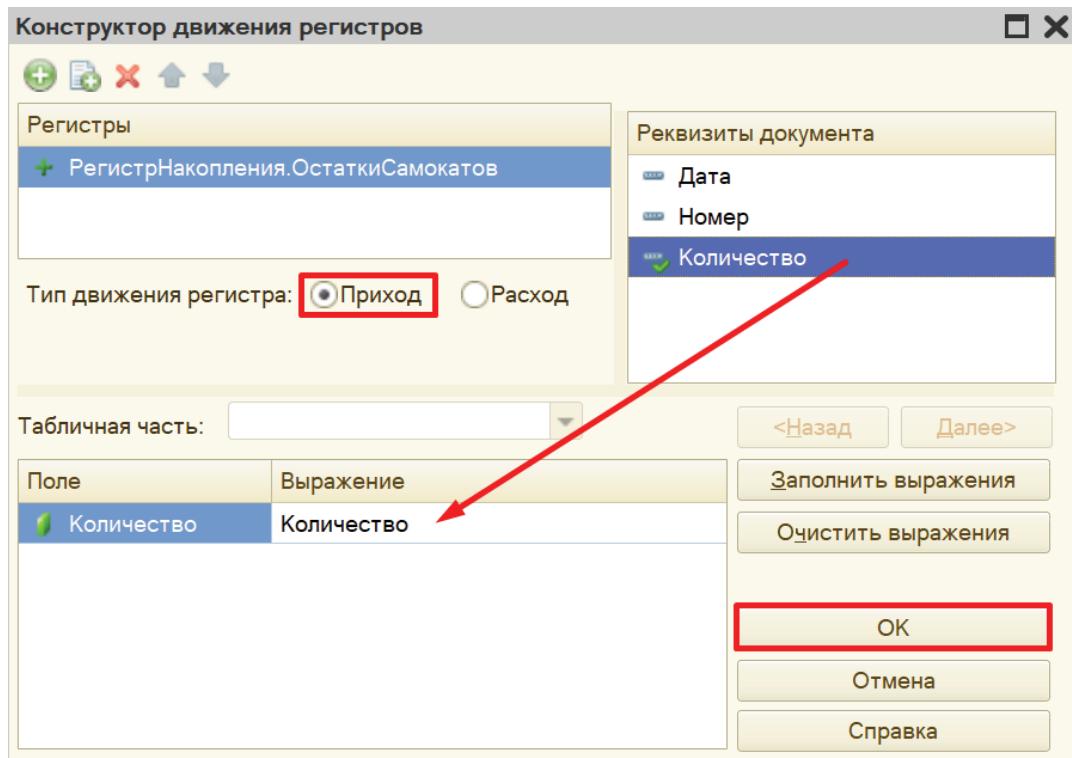
Чтобы *регистры накопления* заработали, нужно сделать следующее:

1. Определить, какие данные будут попадать в каждый регистр (определить документы-регистраторы).

2. Описать, каким образом данные из каждого документа-регистратора должны попадать в регистры.

Начнем с документа «Закупка». Откроем окно редактирования данного документа и перейдем на вкладку «Движения». Отметим *регистр накопления* «ОстаткиСамокатов», поскольку данные о количестве закупленных самокатов необходимо передавать только в этот регистр. Воспользуемся *конструктором движений*.





Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (поскольку один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора.
- В нижней части окна описаны реквизиты *регистра накопления*. Необходимо заполнить поле «Выражение» реквизитами документа.

Данный документ фиксирует количество закупленных электросамокатов. Следовательно, он должен в *регистр накопления* приходить со знаком «+» (плюс). Для этого нужно выбрать тип движения регистра «Приход».

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.

При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

Документ Закупка: Модуль объекта

```

Процедура ОбработкаПроведения (Отказ, Режим)
    //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    //  Данный фрагмент построен конструктором.
    //  При повторном использовании конструктора, внесенные
    //  изменения не будут утеряны.

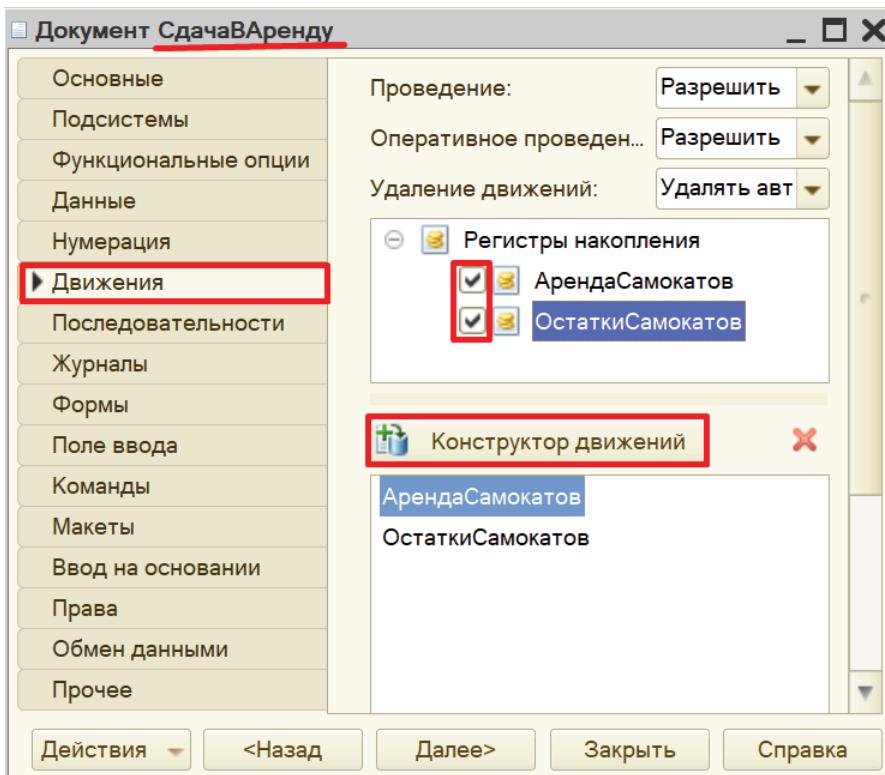
    // регистр ОстаткиСамокатов Приход
    Движение.ОстаткиСамокатов.Записывать = Истина;
    Движение = Движение.ОстаткиСамокатов.Добавить ();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Количество = Количество;

    //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

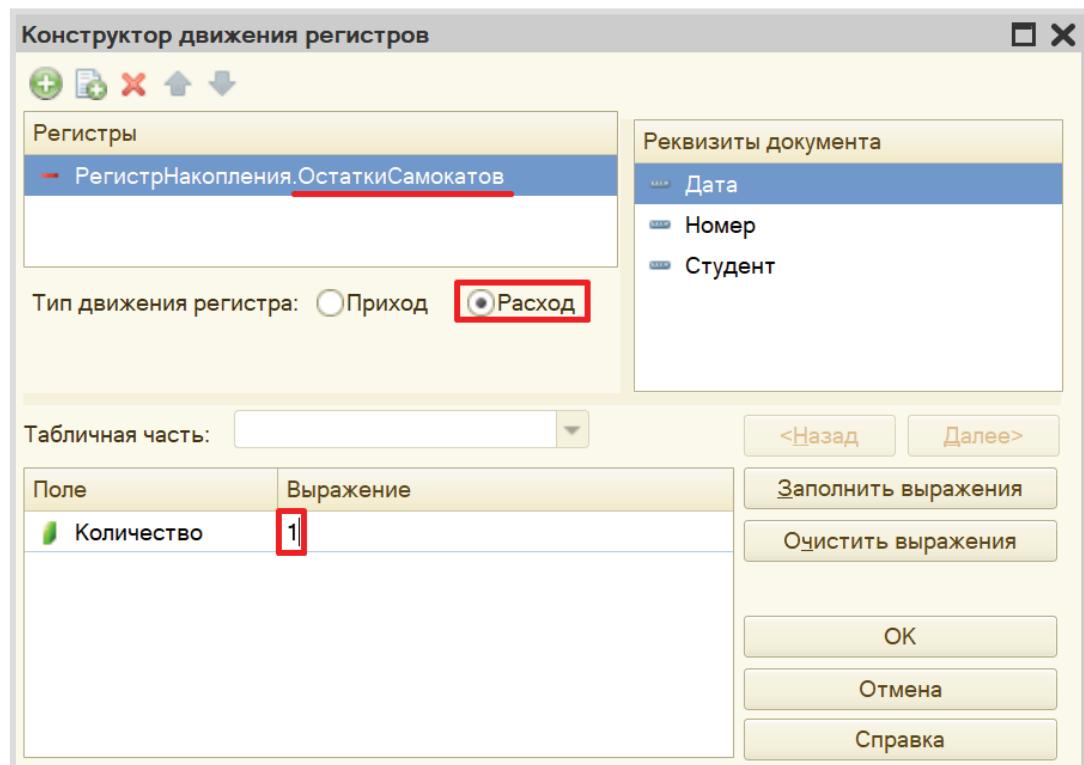
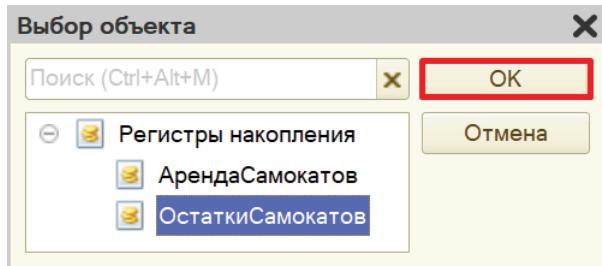
```

Программный код может быть отредактирован вручную, если возникли проблемы при работе с конструктором.

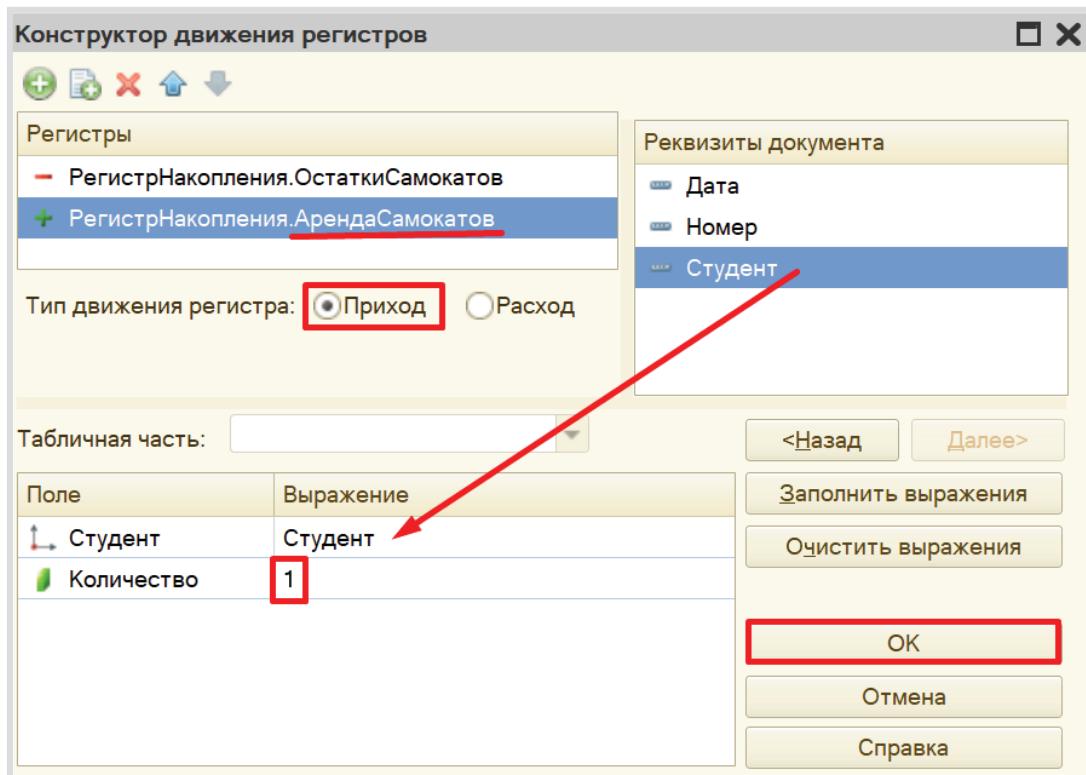
Переходим к настройке документа «СдачаВАренду». Данный документ должен делать движения в оба регистра накоплений.



Сначала опишем действия документа при работе с регистром «ОстаткиСамокатов».



Теперь опишем взаимодействие с другим *регистром накопления*. Для этого нужно нажать на кнопку «Добавить»  и выбрать *регистр накопления* «АрендаСамокатов».



```

Документ СдачаВАренду: Модуль объекта

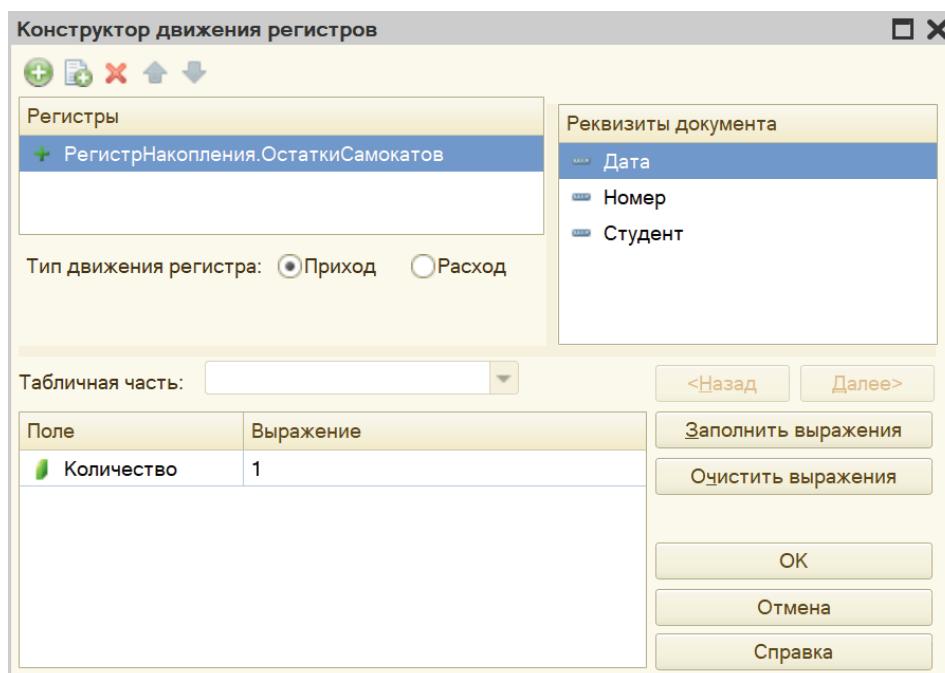
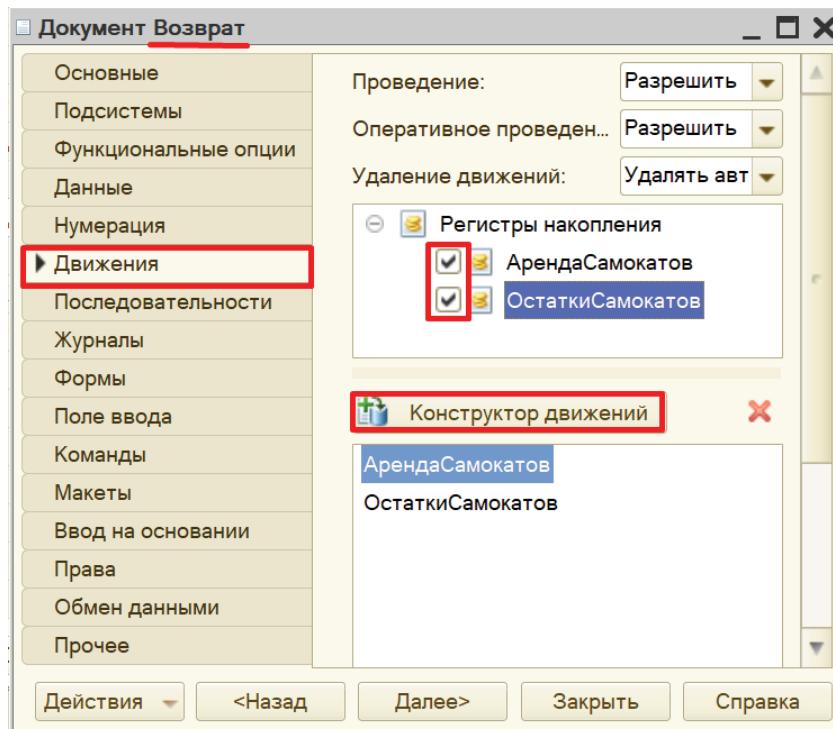
Процедура ОбработкаПроведения(Отказ, Режим)
    //__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    // Данный фрагмент построен конструктором.
    // При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
    // регистр ОстаткиСамокатов Расход
    Движения.ОстаткиСамокатов.Записывать = Истина;
    Движение = Движения.ОстаткиСамокатов.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Количество = 1;

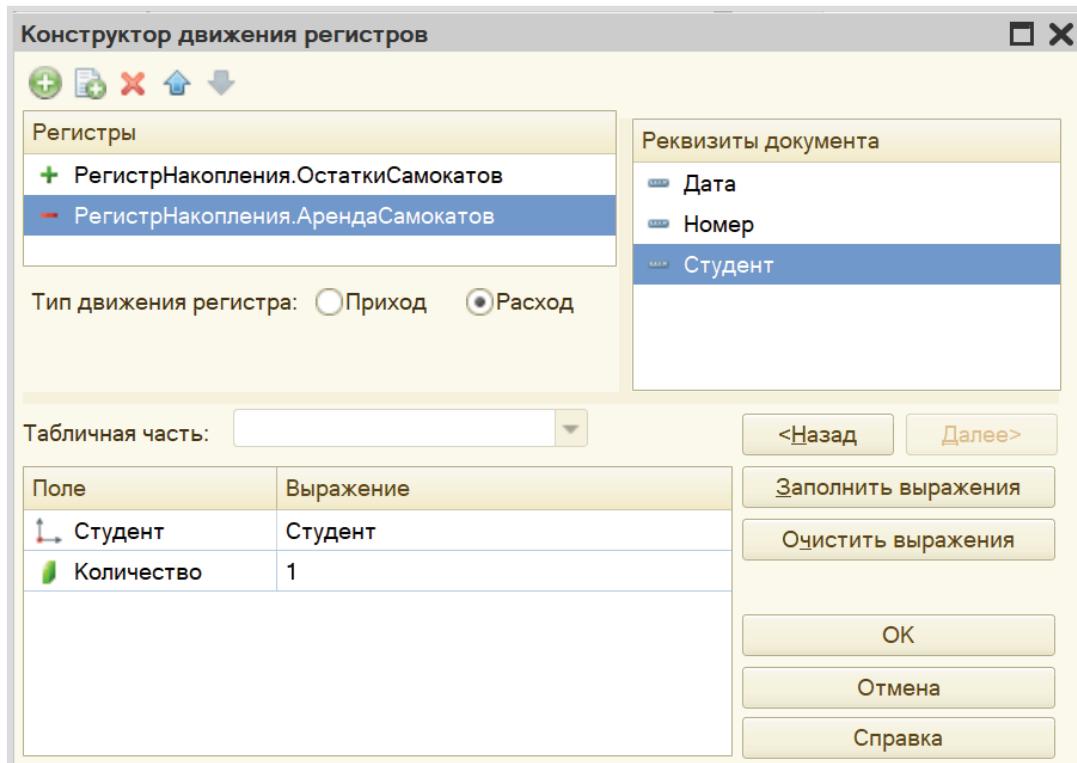
    // регистр АрендаСамокатов Приход
    Движения.АрендаСамокатов.Записывать = Истина;
    Движение = Движения.АрендаСамокатов.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Студент = Студент;
    Движение.Количество = 1;

    //__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

Переходим к настройке последнего документа – «Возврат». Данный документ настраивается аналогично документу «Сдача В Аренду».





Документ Возврат: Модуль объекта

```

    Процедура ОбработкаПроведения(Отказ, Режим)
        // __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
        // Данный фрагмент построен конструктором.
        // При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!

        // регистр ОстаткиСамокатов Приход
        Движения.ОстаткиСамокатов.Записывать = Истина;
        Движение = Движения.ОстаткиСамокатов.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
        Движение.Период = Дата;
        Движение.Количество = 1;

        // регистр АрендаСамокатов Расход
        Движения.АрендаСамокатов.Записывать = Истина;
        Движение = Движения.АрендаСамокатов.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Студент = Студент;
        Движение.Количество = 1;

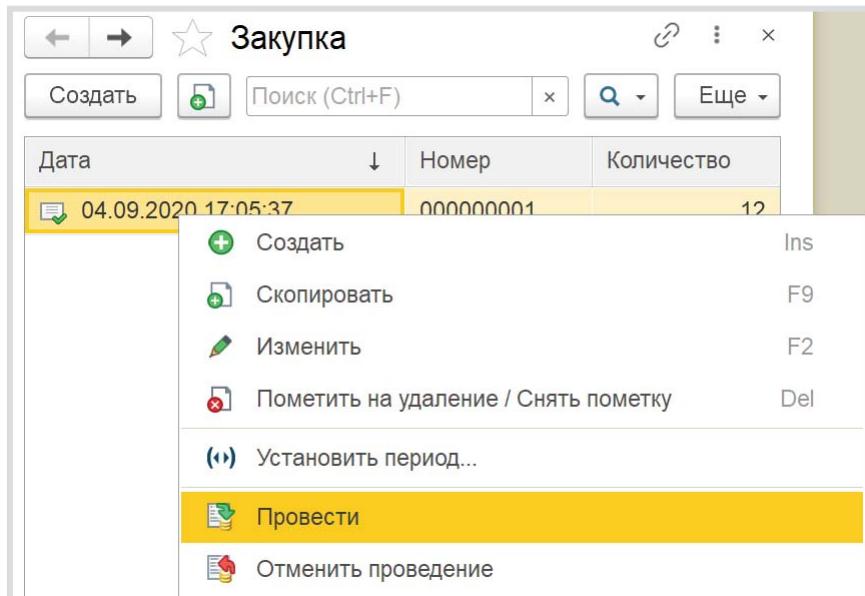
    __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    КонецПроцедуры

```

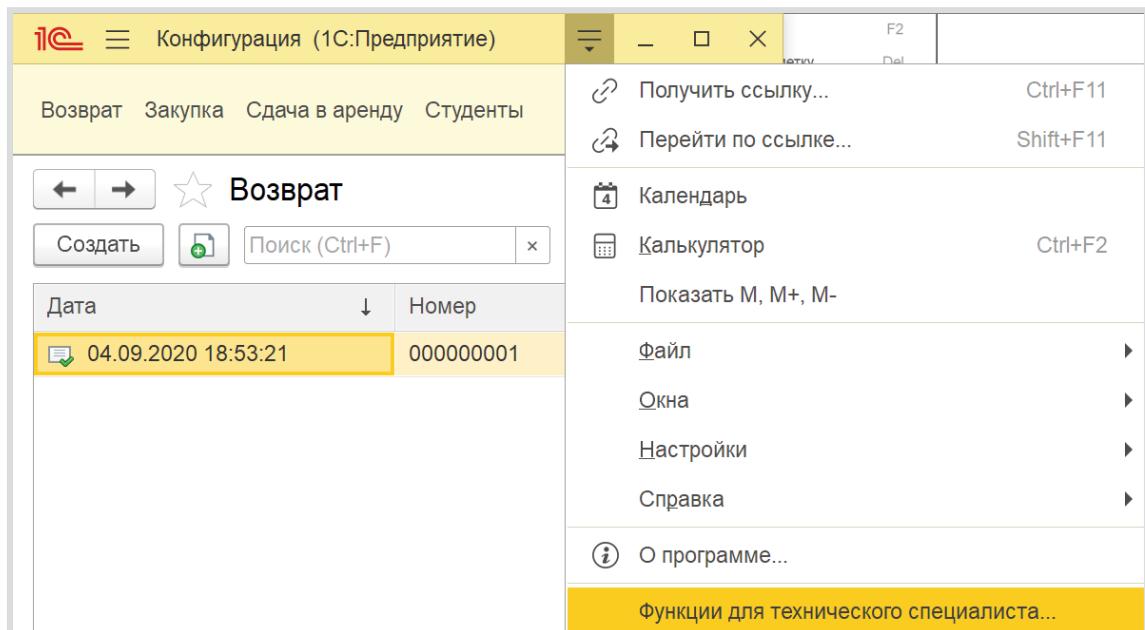
Готово! Теперь каждый из этих документов копирует данные в *регистры накопления*, прибавляя или вычитая количество электросамокатов.

Проверим работу регистра в режиме «1С:Предприятие».

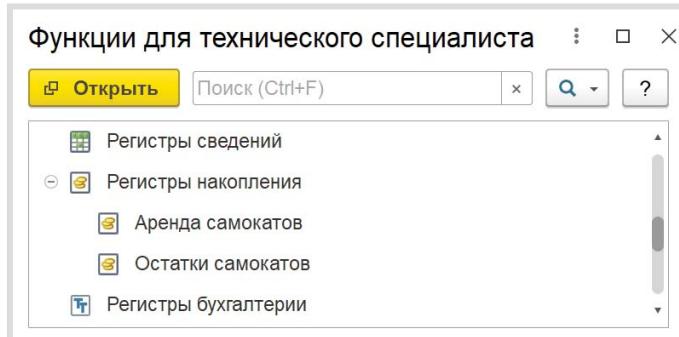
Для начала нужно перепровести (провести заново) все созданные ранее документы «Закупка» и «Сдача в аренду», чтобы данные из них попали в *регистр накопления*. Кроме того, добавим новый документ «Возврат».



Чтобы заглянуть в содержимое *регистра накопления*, следует прибегнуть к функциям технического специалиста.



Найдем наши регистры, откроем их и посмотрим на движения.



Остатки самокатов

Период	Регистратор	Номер строки	Количество
+ 04.09.2020 19:06:10	Закупка 00000001 от 04.09.2020 19:06:10	1	12
- 04.09.2020 19:06:16	Сдача в аренду 00000001 от 04.09.2020 19:06:16	1	1
- 04.09.2020 19:06:20	Сдача в аренду 00000002 от 04.09.2020 19:06:20	1	1
+ 04.09.2020 19:06:23	Возврат 00000001 от 04.09.2020 19:06:23	1	1
- 04.09.2020 19:17:39	Сдача в аренду 00000003 от 04.09.2020 19:17:39	1	1

Аренда самокатов

Период	Регистратор	Номер строки	Студент	Количество
+ 04.09.2020 19:06:16	Сдача в аренду 00000001 от 04.09.2020 19...	1	Иванов	1
+ 04.09.2020 19:06:20	Сдача в аренду 00000002 от 04.09.2020 19...	1	Сидоров	1
- 04.09.2020 19:06:23	Возврат 00000001 от 04.09.2020 19:06:23	1	Иванов	1
+ 04.09.2020 19:17:39	Сдача в аренду 00000003 от 04.09.2020 19...	1	Петров	1

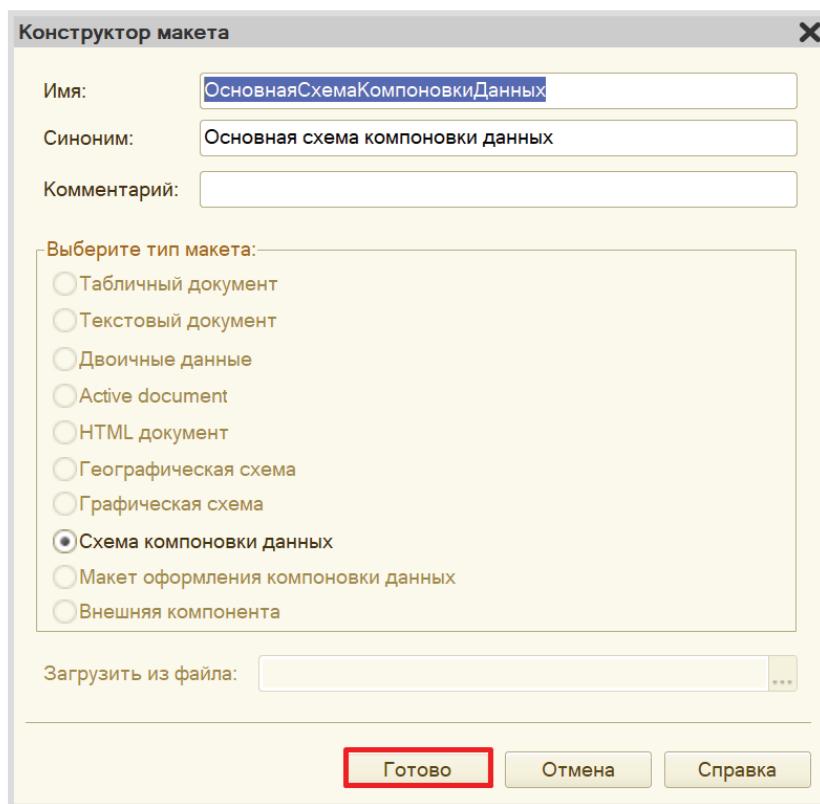
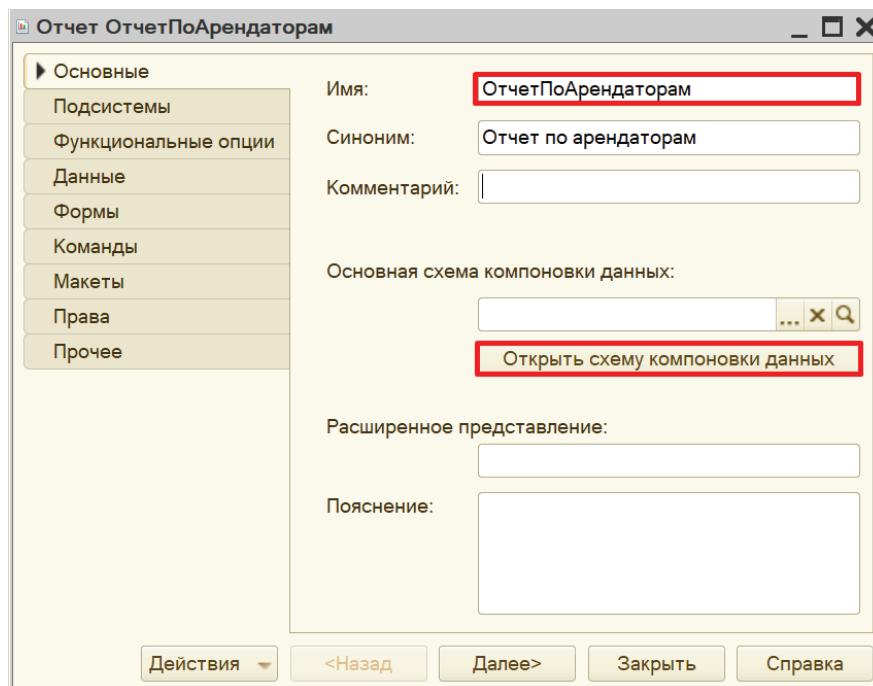
В *регистры накопления* действительно добавляются строки, соответствующие проведенным документам.

Для решения задачи осталось построить отчет, который будет отображать список студентов, которые еще не вернули самокаты.

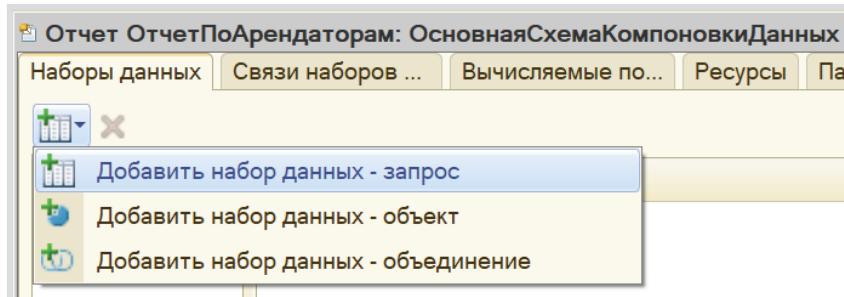
Определение

Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: [https://v8.1c.ru/platforma/отчет/](https://v8.1c.ru/platforma/otchet/)).

Создадим новый отчет «ОтчетПоАрендаторам». Для наполнения отчета воспользуемся *конструктором схемы компоновки данных*.



Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Воспользуемся *конструктором запроса*.

The screenshot shows the 'Query Constructor' window. The main area displays a table with columns: Поле (Field), Путь (Path), Ограничение п... (Restriction p...), Роль (Role), Выраже... (Expression), Проверка иерар... (Hierarchical check), Тип зн... (Type), and Офор... (Formatting). A row is selected in the 'Наборы данных' (Data sets) section under 'Поля' (Fields).

Below the table is a text input field labeled 'Запрос:' (Query:) containing a single character '|'. To the right of this field is a button labeled 'Конструктор запроса...' (Query constructor...), which is highlighted with a red rectangle.

At the bottom of the window, there are two checkboxes: 'Автозаполнение' (Autocompletion) and 'Использовать группировки запроса если возможно' (Use query grouping if possible).

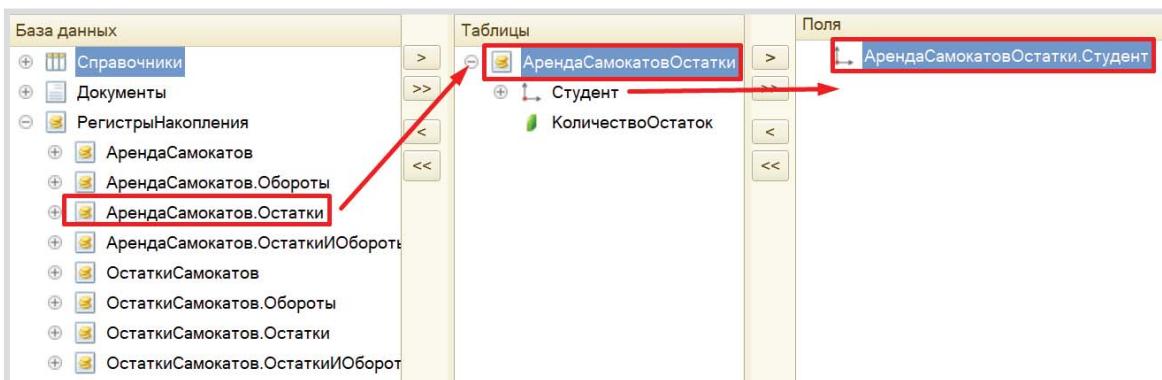
Открывшееся окно имеет три части:

- ❑ Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- ❑ Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- ❑ Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

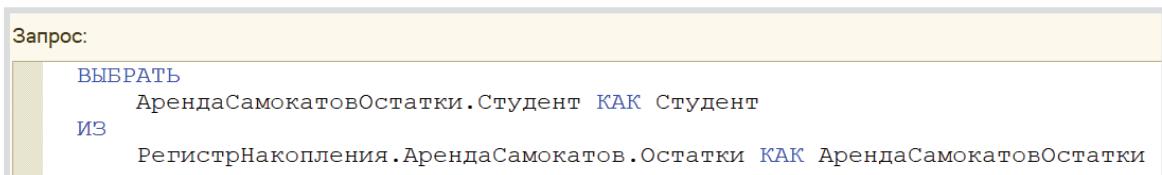
Данные будем брать не из *регистра накоплений* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица позволит получить уже просуммированные значения по всем документам.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

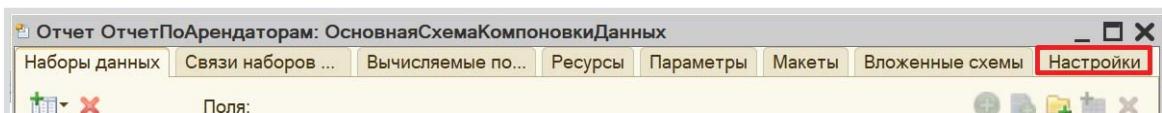
В результате данное окно должно быть заполнено следующим образом:



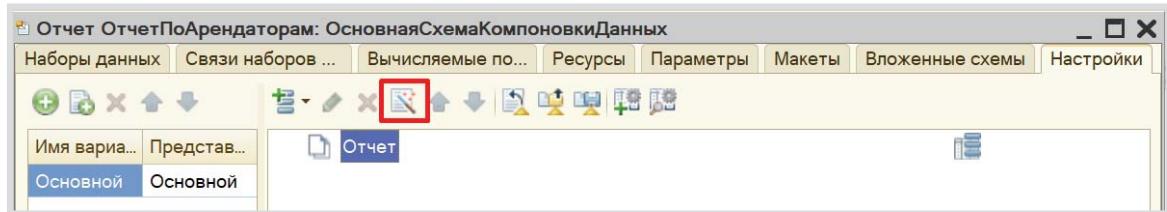
Запрос к базе данных должен выглядеть следующим образом:



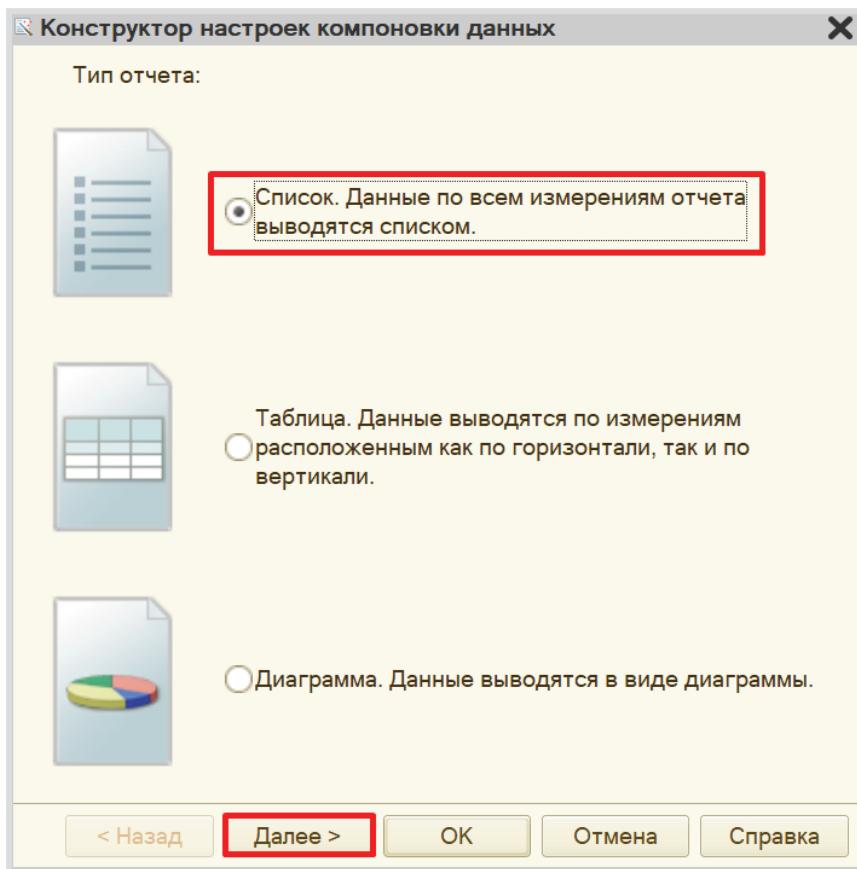
Запрос к базе данных сформирован. Теперь нужно настроить внешний вид отчета. Для этого следует перейти на вкладку «Настройки».



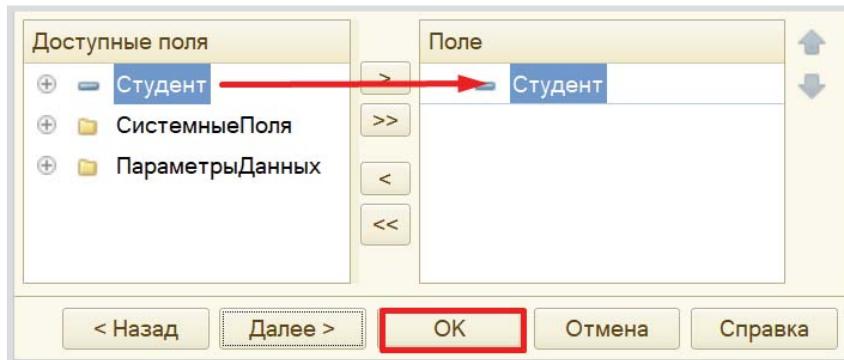
Для настройки внешнего вида отчета воспользуемся *конструктором настроек отчета*.



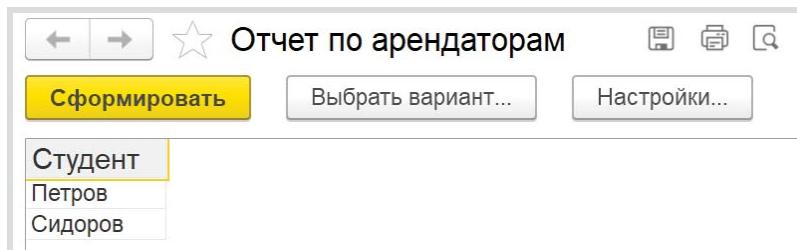
Наш отчет будет иметь форму списка.



Далее нужно выбрать поля, которые будут отображены в отчете.



Осталось лишь убедиться, что отчет работает верно. Для этого посмотрим на отчет в режиме «1С:Предприятие».



Таким образом, отчет выводит список арендаторов, не вернувших самокат в пункт проката.

Стоит отметить, что получившаяся информационная система не способна отслеживать остатки и работает «на честном слове». Это значит, что система не способна отслеживать, когда количество самокатов на складе станет отрицательным или тот факт, что один студент решит арендовать два самоката (хотя мы ограничили это условием задачи).

Поставленная задача решена.

Лабораторная работа № 12

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ БИБЛИОТЕКИ

Сложность: *

Теги: справочник, документ, ввод на основании,
регистр накопления, схема компоновки данных

ЗАДАНИЕ

Заказчик просит разработать информационную систему для библиотеки. Необходимо вести учет читателей библиотеки и книг.

1. В системе нужно регистрировать выдачу книг. Следует фиксировать читателя и перечень взятых им книг.
2. Кроме того, нужно регистрировать возврат книг в библиотеку. Причем возврат книг должен формироваться на основании выдачи книг.
3. Также необходимо формировать отчет, в котором будут выводиться должники и список взятых ими книг. В отчете нужно реализовать возможность производить отбор по читателю и по книге.

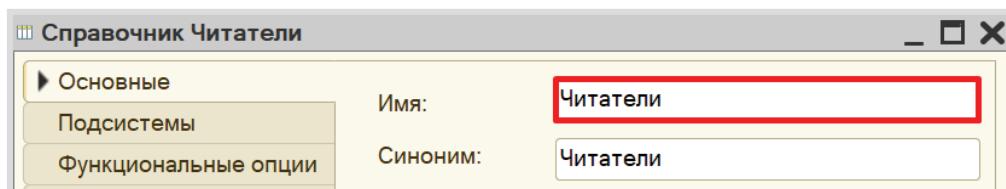
Выполнение

Чтобы вести учет читателей библиотеки и книг, в «1С:Предприятии» имеется объект конфигурации *справочник*.

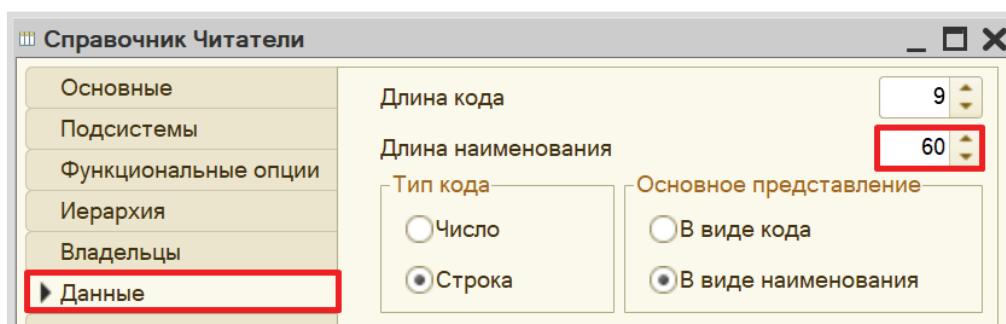
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

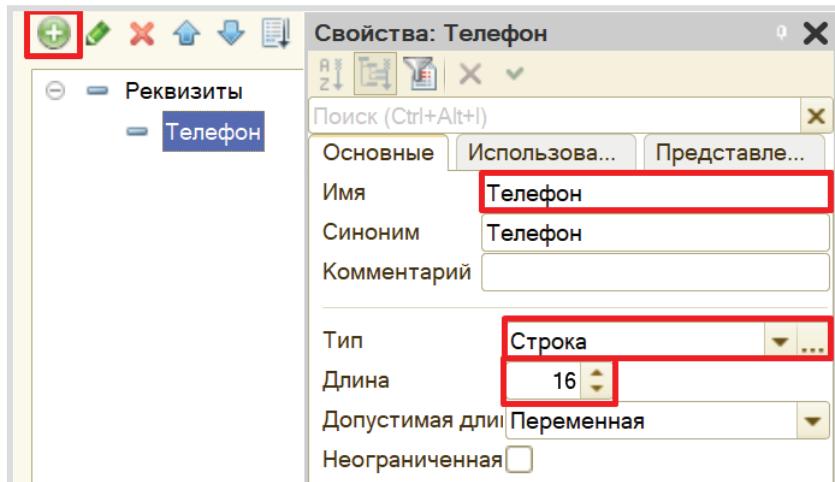
Добавим справочник «Читатели».



Перейдем на вкладку «Данные» и увеличим количество символов наименования до 60.



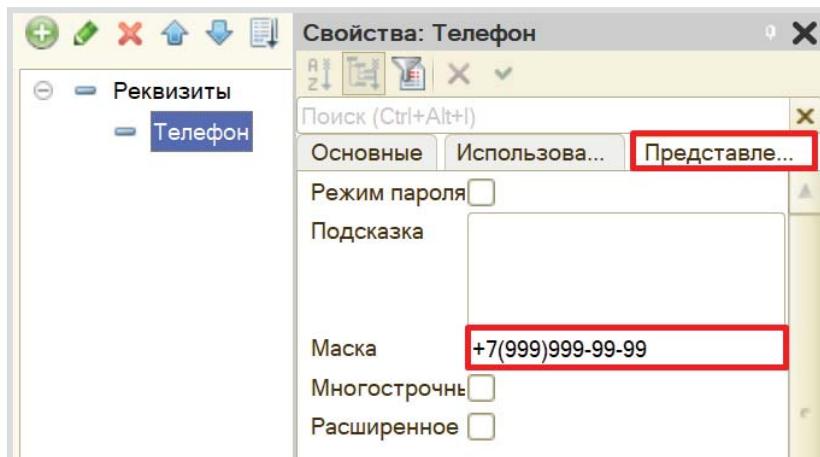
На этой вкладке можно добавить реквизиты для хранения любой информации о читателе. Например, добавим реквизит «Телефон».



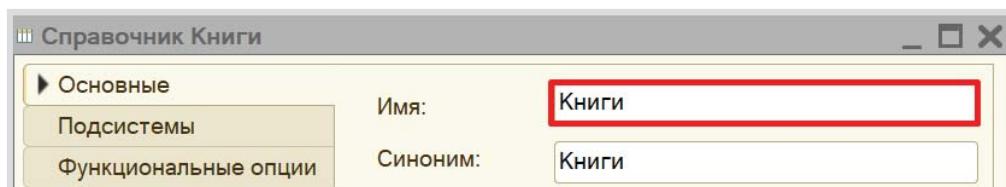
Зададим маску для данного реквизита.

Определение

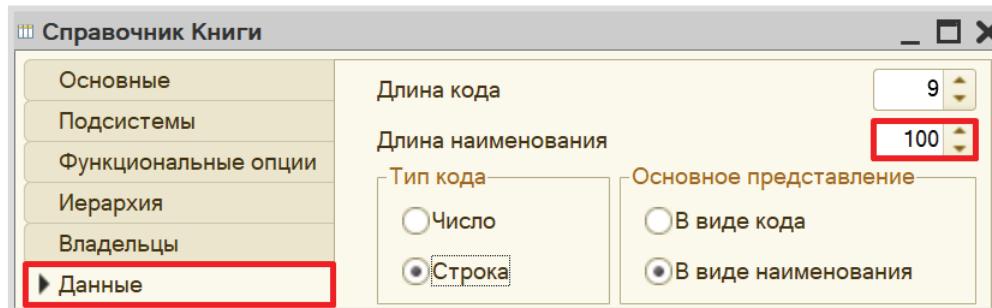
Маска – это шаблон для ввода информации в данный реквизит.



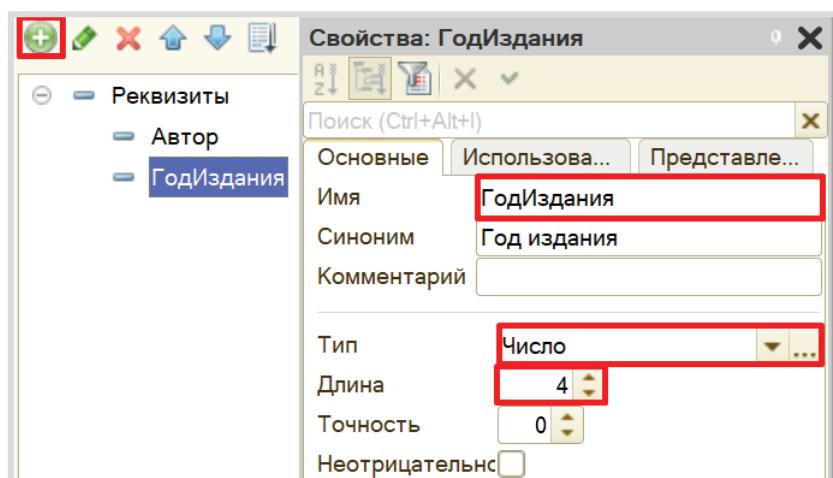
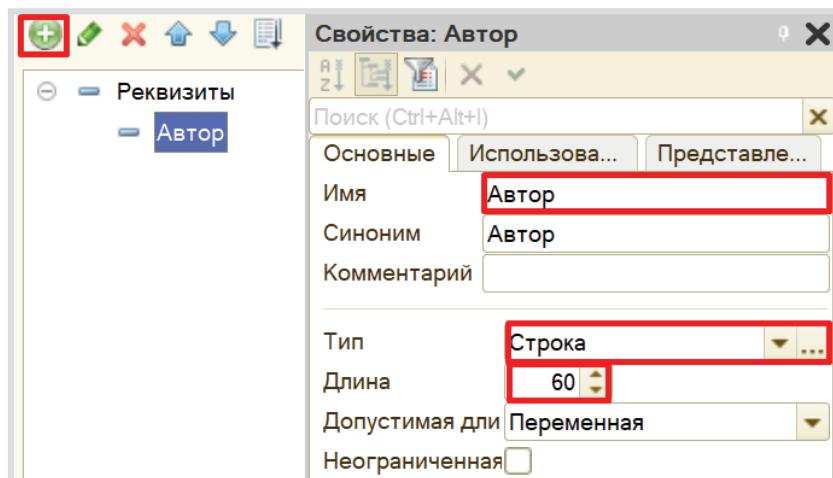
Создадим еще один справочник «Книги».



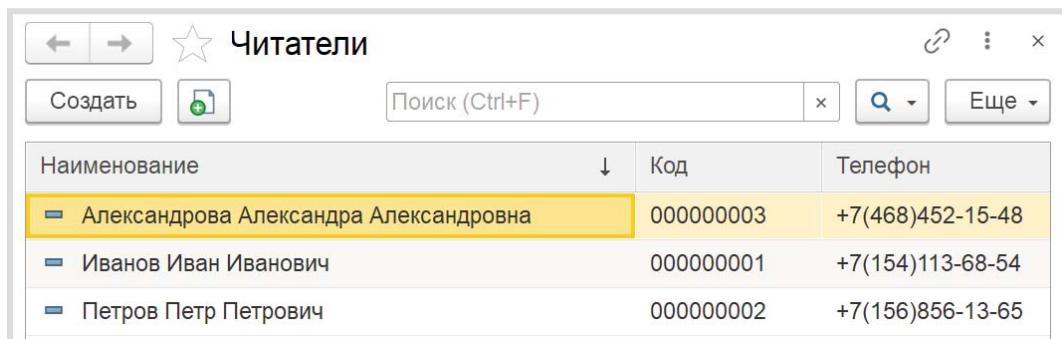
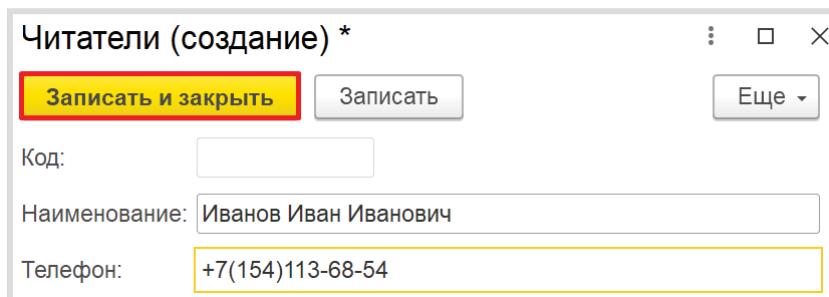
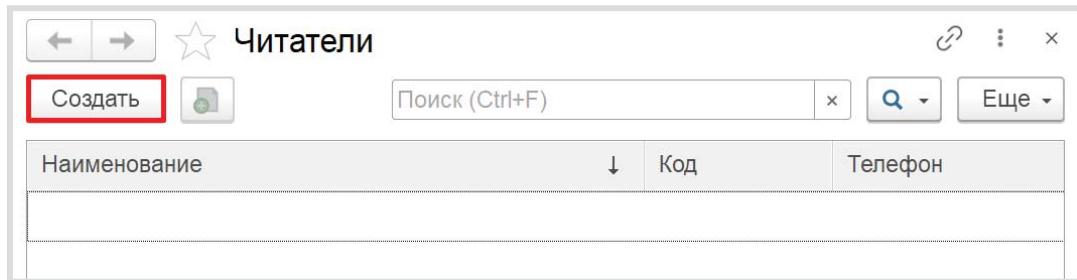
Аналогично перейдем на вкладку «Данные» и увеличим длину наименования до 100.



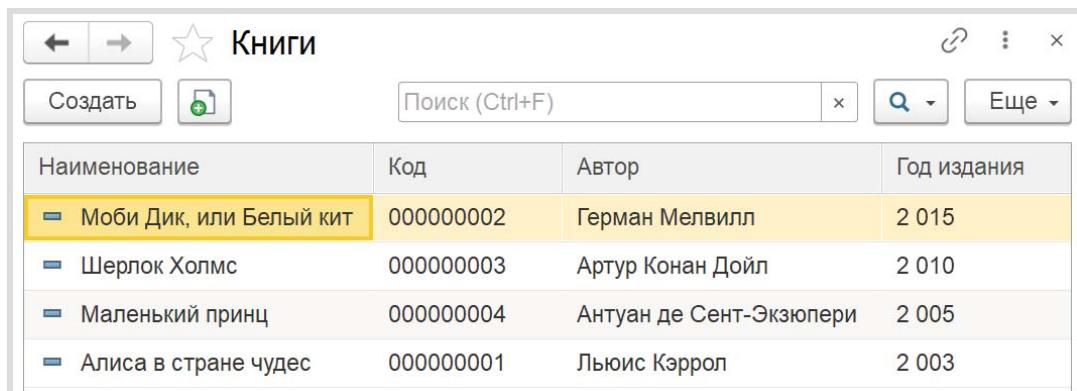
Добавим реквизиты «Автор» и «ГодИздания».



Теперь можно перейти в режим «1С:Предприятие» и добавить элементы в созданные справочники.



Аналогично добавим несколько элементов в справочник «Книги».



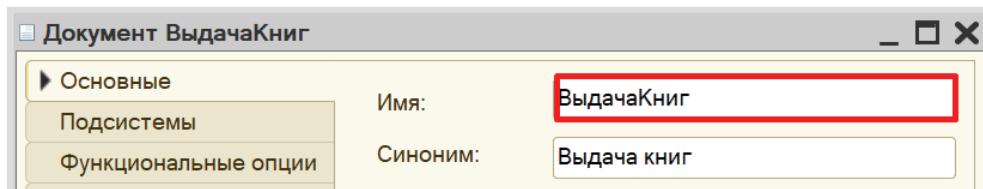
Таким образом, мы организовали хранение объектов аналитики.

Переходим к следующему пункту задачи: нам необходимо зарегистрировать выдачу некоторого перечня книг конкретному читателю. Для регистрации факта выдачи книг будем использовать объект конфигурации *документ*.

Определение

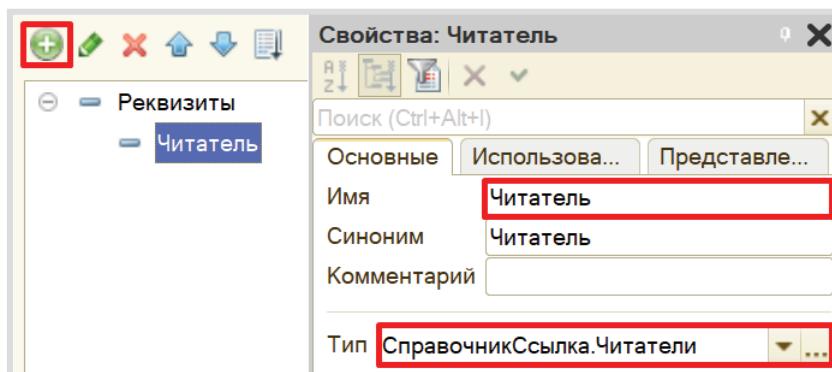
Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим новый документ «ВыдачаКниг».



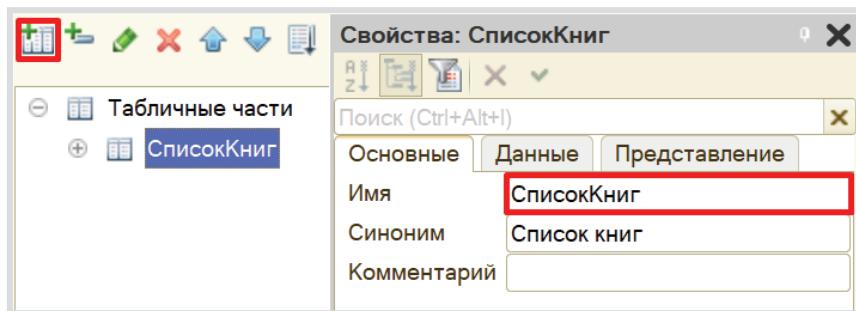
Перейдем на вкладку «Данные» для формирования структуры документа.

Добавим новый реквизит «Читатель» с типом «СправочникСсылка.Читатели». Таким образом, пользователь сможет заполнить данный реквизит (поле) только элементом из справочника «Читатели».



Разумеется, мы можем добавить несколько реквизитов и хранить в каждом из них отдельную книгу. Но количество взятых читателем книг может быть (теоретически) неограниченно. Следовательно, создание отдельного реквизита нам не поможет.

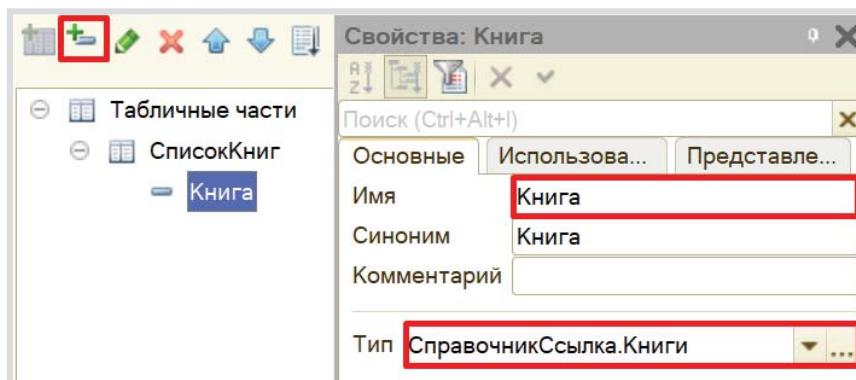
Добавим табличную часть «СписокКниг». В табличную часть можно добавить неограниченное количество строк.



На данный момент табличная часть совершенно пуста. Нужно добавить в нее колонки для заполнения оператором.

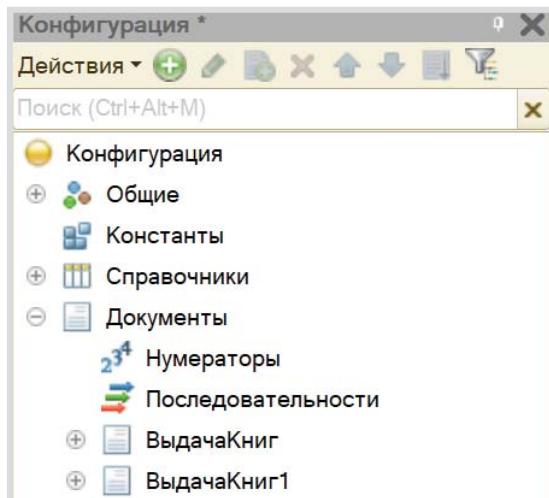
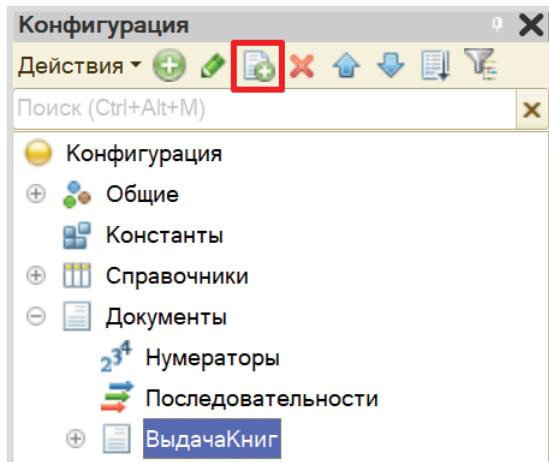
Поскольку нас интересует перечень взятых читателем книг, то именно реквизит «Книги» нужно добавить в данную табличную часть.

По аналогии с реквизитом «Читатель» дадим возможность оператору заполнять строки только элементами из справочника «Книги», поэтому установим тип реквизита «СправочникСсылка.Книги».

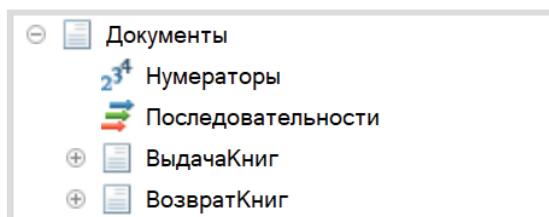


Помимо выдачи книг данная информационная система должна регистрировать возврат читателем книг в библиотеку. Здесь все должно быть аналогично: оператор должен отметить читателя, а также книги, которые читатель вернул в библиотеку. Структура документов совершенно одинакова, следовательно, мы можем не тратить время на создание точно такого же документа, а создать новый документ копированием существующего.

Для этого следует выделить нужный документ в окне конфигурации и нажать на кнопку «ДобавитьКопированием». Должен появиться документ с аналогичной структурой, но слегка отличающимся названием (система автоматически следит за уникальностью имен объектов конфигурации).



Переименуем документ в «ВозвратКниг».



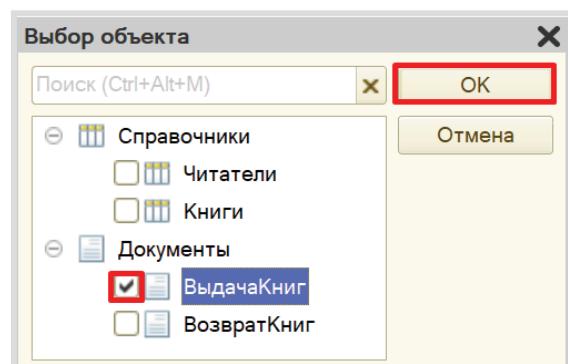
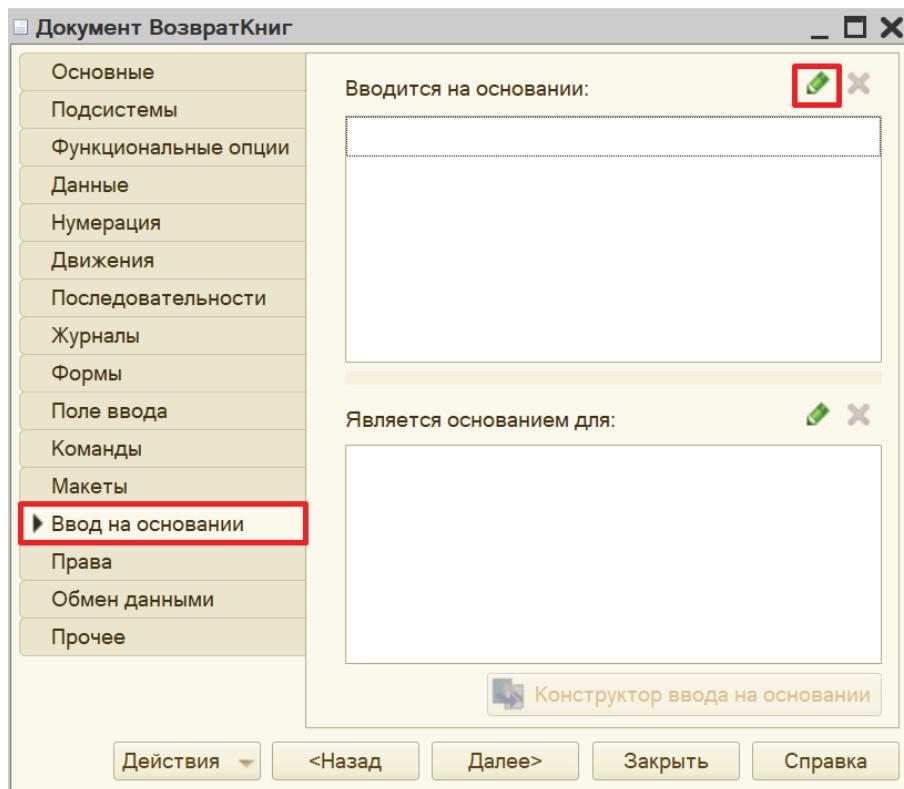
Вы можете убедиться в том, что новый документ обладает идентичной структурой. Для этого перейдите на вкладку «Данные» и сравните его с документом «ВыдачаКниг». Обратите внимание на типы реквизитов, они тоже должны быть одинаковыми.

Как нам ускорить процесс работы оператора? Можно заранее заполнить все поля документа «ВозвратКниг» на основе документа «ВыдачаКниг». Предположим, что некоторый читатель взял две книги, оператор заполнил документ «ВыдачаКниг», в который внес читателя и арендованные

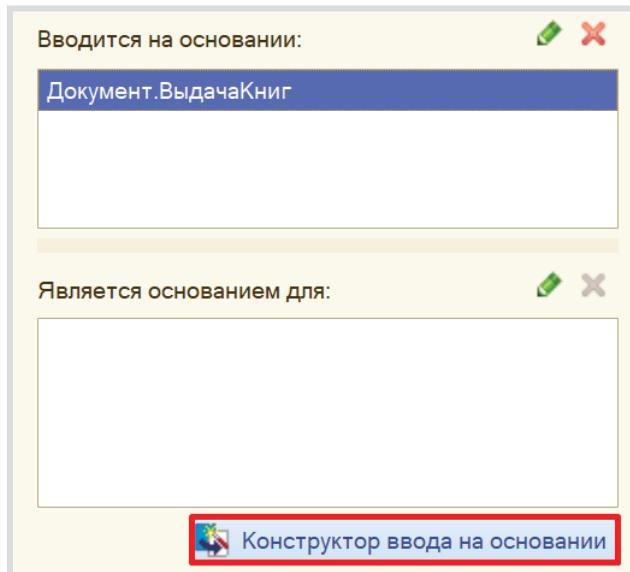
книги. Затем, когда читатель вернул книги в библиотеку, оператор сможет найти тот документ «ВыдачаКниг» и на основании него создать документ «ВозвратКниг» с уже заполненными полями.

Для выполнения данной задачи будем использовать *конструктор ввода на основании*. Более подробно про *конструктор ввода на основании* можно прочитать здесь: <https://v8.1c.ru/platforma/konstruktor-vvoda-na-osnovanii/>.

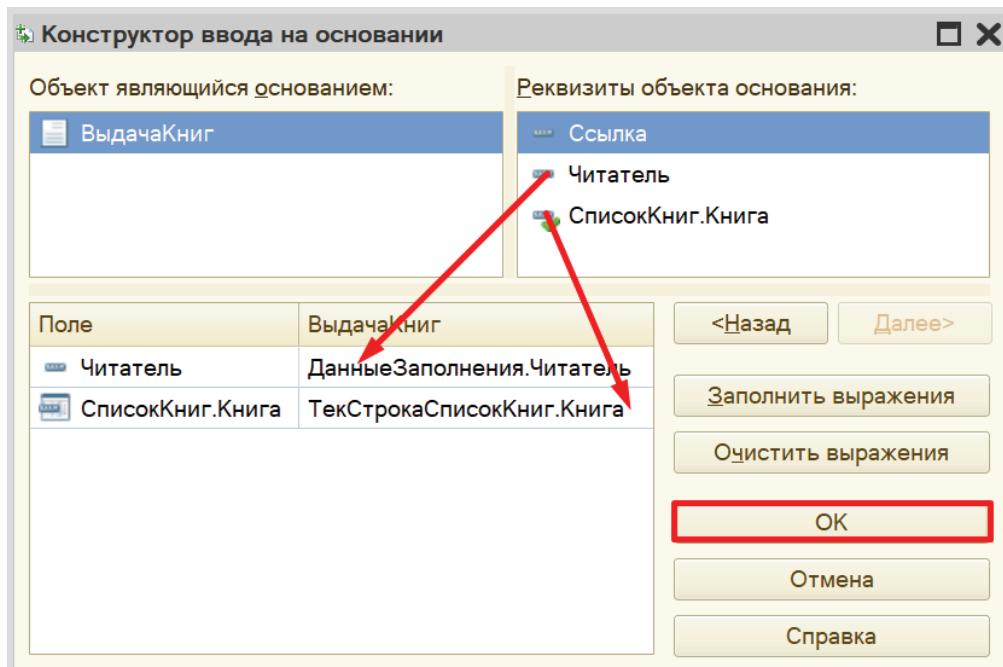
Откройте окно редактирования документа «ВозвратКниг» и перейдите на вкладку «Ввод на основании». Нажмите на кнопку с зеленым карандашом для выбора документа, на основании которого будет заполняться документ «ВозвратКниг».



Далее нужно воспользоваться конструктором ввода на основании.



В открывшемся окне следует указать, какими данными из документа «ВыдачаКниг» нужно заполнить поля документа «ВозвратКниг».



При нажатии на кнопку «OK» формируется программный код. Данный код описывает, какие данные из документа-основания нужно перенести в реквизиты документа «ВозвратКниг».

```

Документ ВозвратКниг: Модуль объекта

Процедура ОбработкаЗаполнения (ДанныеЗаполнения, СтандартнаяОбработка)
    //{{_КОНСТРУКТОР_ВВОД_НА_ОСНОВАНИИ
    // Данный фрагмент построен конструктором.
    // При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
    Если ТипЗнач(ДанныеЗаполнения) = Тип("ДокументСсылка.ВыдачаКниг") Тогда
        // Заполнение шапки
        Читатель = ДанныеЗаполнения.Читатель;
        Для Каждого ТекСтрокаСписокКниг Из ДанныеЗаполнения.СписокКниг Цикл
            НоваяСтрока = СписокКниг.Добавить();
            НоваяСтрока.Книга = ТекСтрокаСписокКниг.Книга;
        КонецЦикла;
    КонецЕсли;
    //}}_КОНСТРУКТОР_ВВОД_НА_ОСНОВАНИИ
КонецПроцедуры

```

Теперь можно перейти в режим «1С:Предприятие» и проверить работоспособность созданных документов.

Для начала заполним документ «ВыдачаКниг».

Наименование	Код
Александрова Александра Александровна	000000003
Иванов Иван Иванович	000000001
Петров Петр Петрович	000000002

Выдача книг (создание) *

Провести и закрыть Записать Провести Еще ▾

Номер:

Дата: 16.09.2020 0:00:00

Читатель: Петров Петр Петрович

Добавить Поиск (Ctrl+F) Еще ▾

N	Книга
1	Маленький принц
2	Шерлок Холмс
3	Алиса в стране чудес

Таким образом, была реализована возможность фиксировать выдачу книг читателям.

На основании данного документа можно сформировать документ «Возврат книг».

Выдача книг

Создать Создать на основании ▾ Поиск (Ctrl+F) Еще ▾

Дата	Возврат книг	Читатель
16.09.2020 11:44:43	000000001	Петров Петр Петрович

Откроется форма документа «Выдача книг» с уже заполненными данными. Мы можем отредактировать список книг, удалив несколько из списка.

Возврат книг (создание) *

Провести и закрыть Записать Провести Еще ▾

Номер:

Дата: 16.09.2020 0:00:00

Читатель: Петров Петр Петрович

Добавить Поиск (Ctrl+F) Еще ▾

N	Книга
1	Маленький принц
2	Алиса в стране чудес

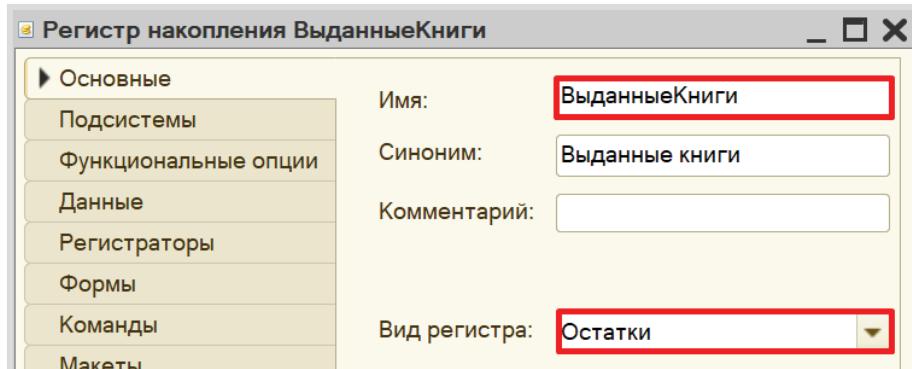
Таким образом, мы реализовали возможность регистрации возврата книг.

Далее нас интересует возможность вести учет книг и читателей, которым они выданы. Для этого воспользуемся *регистром накопления*.

Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

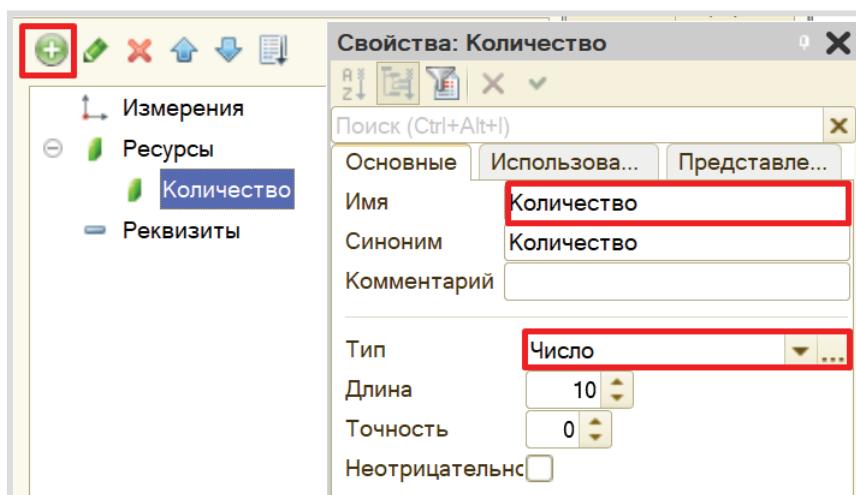
Добавим новый *регистр накопления* «ВыданнЫеКниги» вида «Остатки».



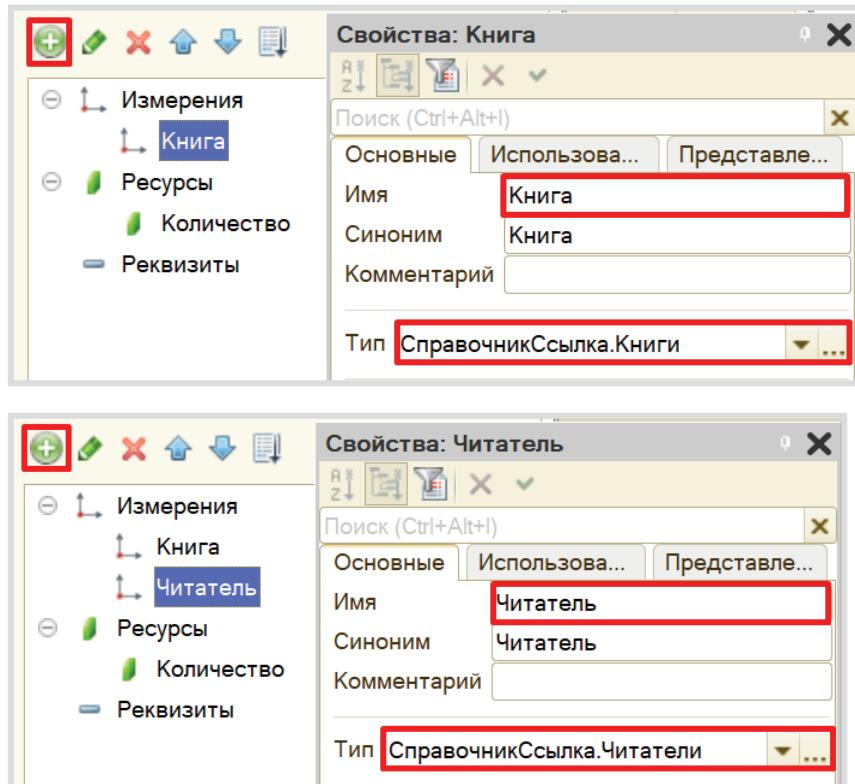
Для формирования структуры регистра перейдем на вкладку «Данные».

Структура *регистра накопления* отличается от структуры документа.

Заполнение данного окна проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, следует задать вопрос: «Что мы хотим накапливать/считать в данном регистре?». Мы хотим считать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число».



Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим считать количество. Мы хотим считать количество (чего?) книг в разрезе (чего?) читателей. Значит, в качестве измерения следует добавить реквизиты «Книга» (тип – СправочникСсылка.Книги) и «Читатель» (тип – СправочникСсылка.Читатели).

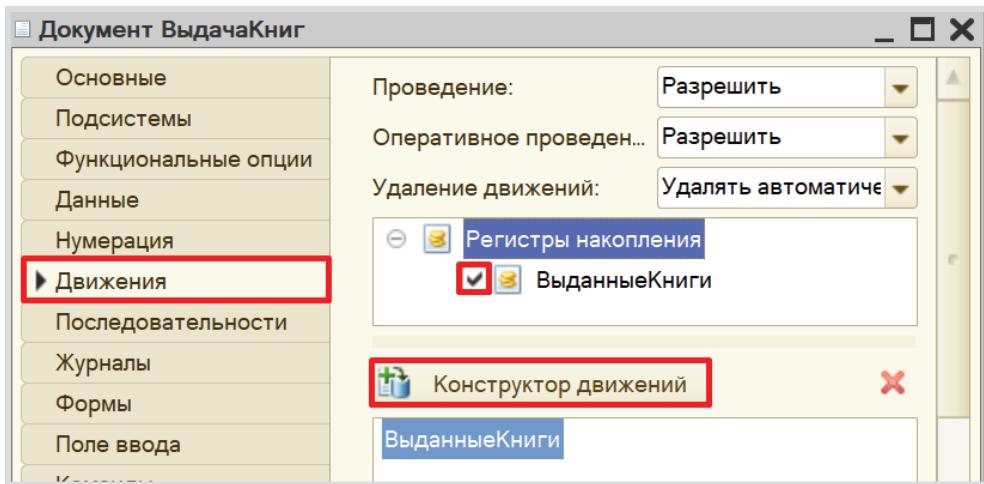


Чтобы *регистр накопления* заработал, нужно сделать следующее:

1. Определить источники данных, которые должны попадать в регистр (определить документы-регистраторы).
2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.

В данный регистр будут попадать данные из обоих созданных документов, поэтому оба документа будут являться регистраторами для регистра.

Начнем с документа «ВыдачаКниг» – откроем окно редактирования данного документа на вкладке «Движения». Отметим, что документ будет делать движения в *регистр накопления* и воспользуемся *конструктором движений*.

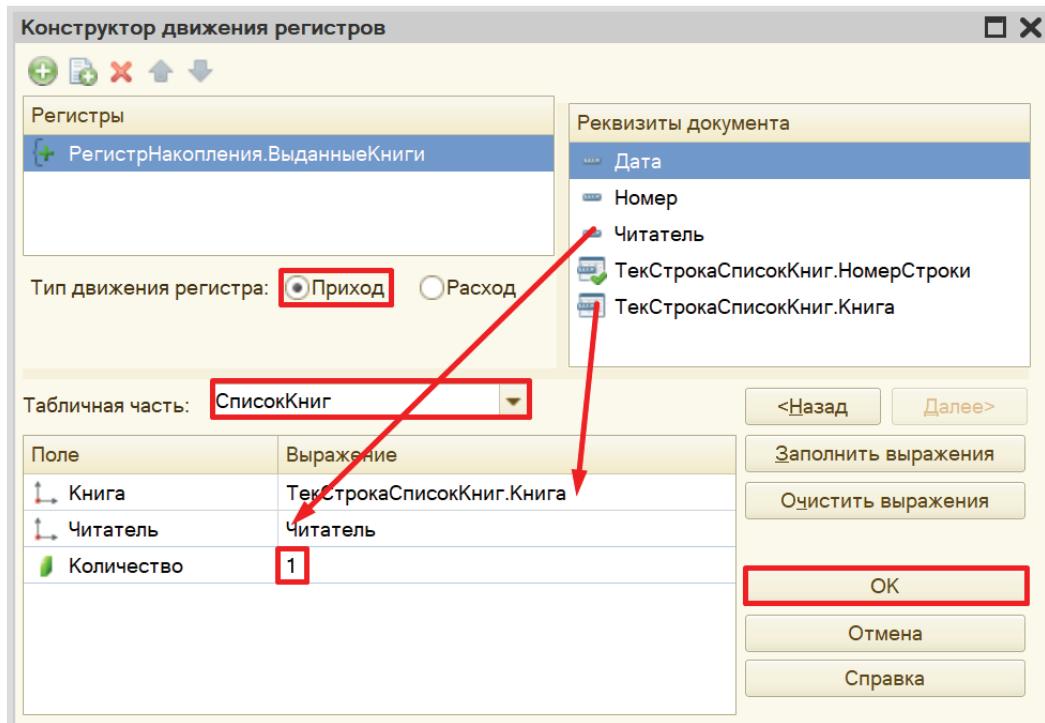


Окно *конструктора движений* состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты *регистра накопления*. Нужно заполнить поле «Выражение» реквизитами документа.

Поскольку выдача книг должна увеличивать количество читателей-должников, то тип движения регистра следует выбрать «Приход». Регистр будет обозначаться знаком «+» (плюс).

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.



При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

```
Документ ВыдачаКниг: Модуль объекта

Процедура ОбработкаПроведения(Отказ, Режим)
//{{ __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
//  Данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные в

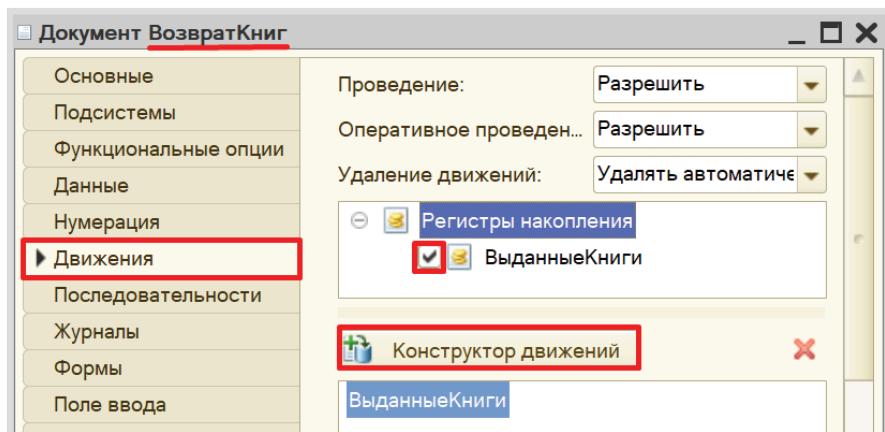
//  регистр ВыданыеКниги Приход
Движения.ВыданыеКниги.Записывать = Истина;
Для Каждого ТекСтроКаСписокКниг Из СписокКниг Цикл
  Движение = Движения.ВыданыеКниги.Добавить();
  Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
  Движение.Период = Дата;
  Движение.Книга = ТекСтроКаСписокКниг.Книга;
  Движение.Читатель = Читатель;
  Движение.Количество = 1;
КонецЦикла;

//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

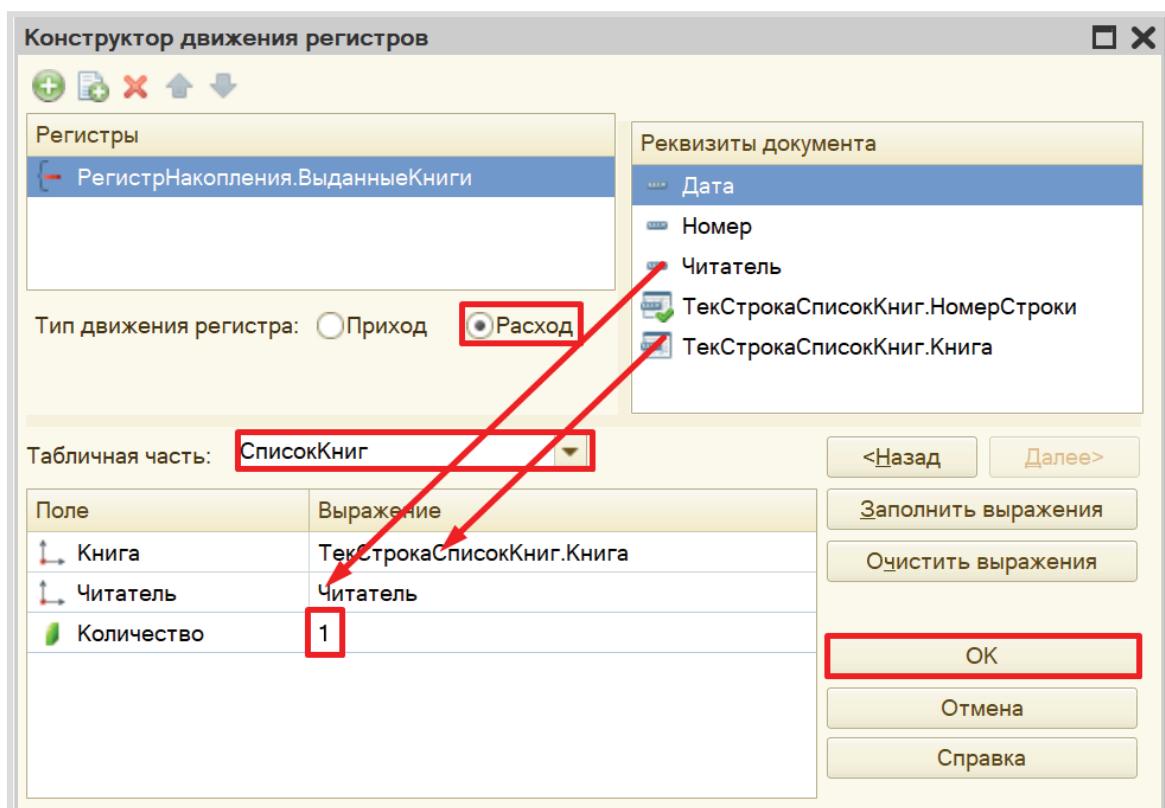
Значение в *регистре накопления* увеличивается на 1 (единицу) за каждую выданную читателю книгу. Документ «ВозвратКниг» должен делать все с точностью до наоборот: значение в *регистре накопления* должно уменьшаться на 1 (единицу) за каждую возвращенную в библиотеку книгу.

Реализуем это.

Откроем окно редактирования документа «ВозвратКниг» на вкладке «Данные». Отметим, что данные из документа будут двигаться в *регистр накопления* и воспользуемся *конструктором движений*.



Заполним окно *конструктора движений* точно так же, как и для документа «ВыдачаКниг». Единственная разница будет лишь в том, что документ будет совершать движение со знаком «-» (минус), следовательно, тип движения нужно выбрать «Расход».



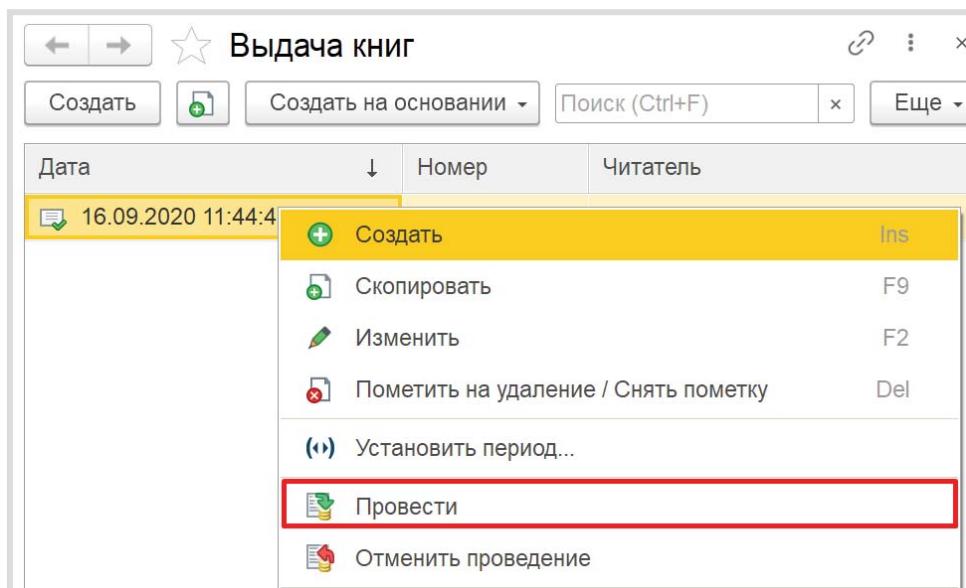
В результате имеем программный код, который расположился прямо под процедурой «ОбработкаЗаполнения», который был получен в результате работы с *конструктором ввода на основании*.

```
□ Документ ВозвратКниг: Модуль объекта

□ Процедура ОбработкаЗаполнения (ДанныеЗаполнения, СтандартнаяОбработка) ...
□ Процедура ОбработкаПроведения (Отказ, Режим)
    // { __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    //  Данный фрагмент построен конструктором.
    //  При повторном использовании конструктора, внесенные вручную изменения
    //  регистр ВыданныеКниги Расход
    Движения.ВыданныеКниги.Записывать = Истина;
    Для Каждого ТекСтрокааСписокКниг Из СписокКниг Цикл
        Движение = Движения.ВыданныеКниги.Добавить ();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Книга = ТекСтрокааСписокКниг.Книга;
        Движение.Читатель = Читатель;
        Движение.Количество = 1;
    КонецЦикла;
    // } __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

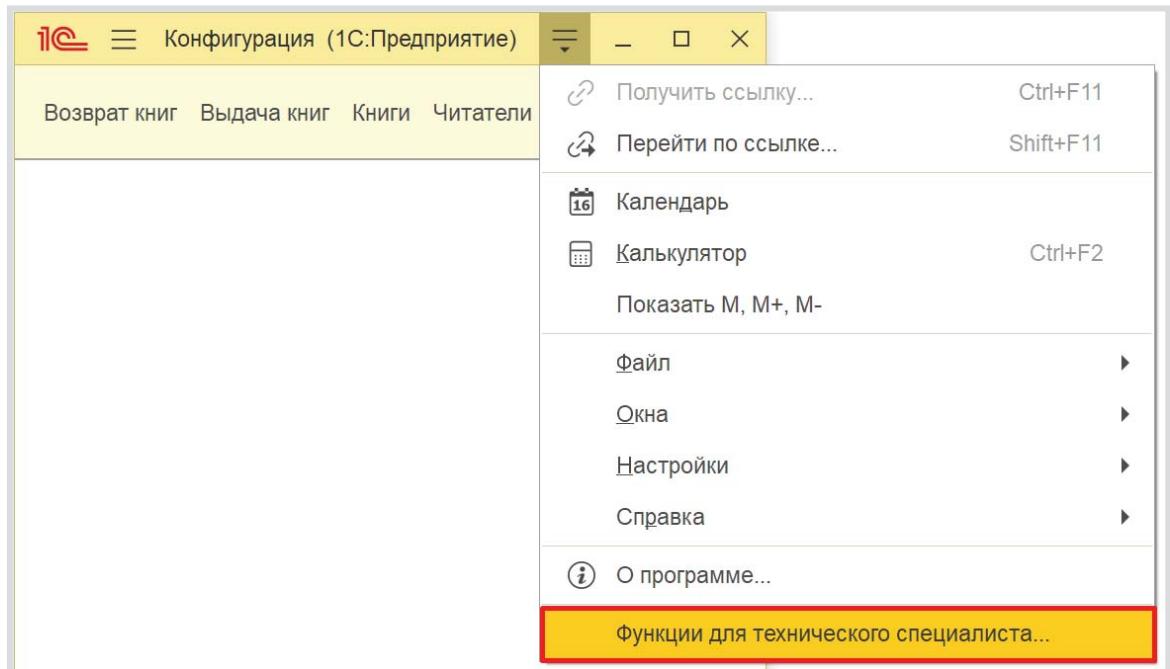
Проверим работоспособность *регистра накопления* в режиме «1С:Предприятие».

В первую очередь, нужно перепровести (провести заново) существующие документы выдачи и возврата книг, иначе данные никак не попадут в *регистр накопления*.

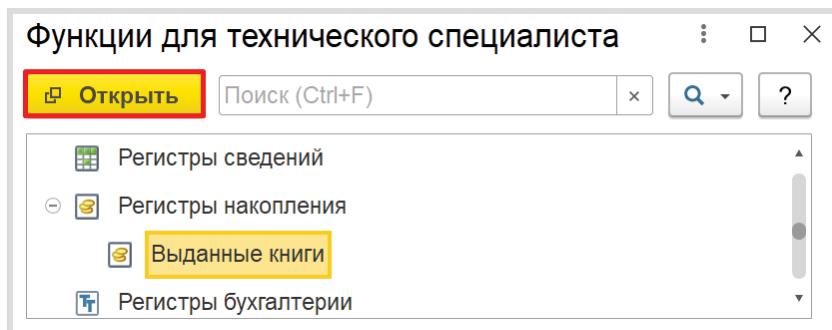


Обратите внимание, что на главной странице системы не создала кнопку открытия регистра накопления. Это связано с тем, что все расчеты в регистрах накопления происходят в фоновом режиме, «за кадром», то есть пользователю о них знать не нужно вовсе. Поэтому по умолчанию регистры накопления настраивают так, чтобы пользователи не имели к ним доступа.

Но мы, будучи разработчиками, можем обратиться к любому объекту конфигурации. Для этого воспользуемся функциями для технического специалиста.



Найдем в списке созданный *регистр накопления* «ВыданыеКниги» и изучим его содержимое.



Период	Регистратор	Номер строки	Книга	Читатель	Количество
+ 16.09.2020...	Выдача книг 000000001...	1	Маленький принц	Петров Петр Петрович	1
+ 16.09.2020...	Выдача книг 000000001...	2	Шерлок Холмс	Петров Петр Петрович	1
+ 16.09.2020...	Выдача книг 000000001...	3	Алиса в стране чудес	Петров Петр Петрович	1
- 16.09.2020...	Возврат книг 000000001...	1	Маленький принц	Петров Петр Петрович	1
- 16.09.2020...	Возврат книг 000000001...	2	Алиса в стране чудес	Петров Петр Петрович	1

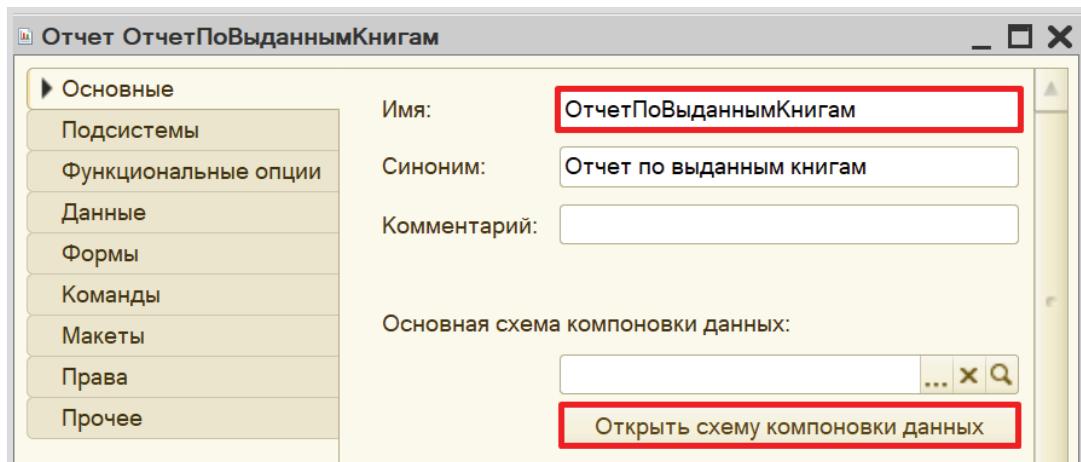
Видно, что в регистр попадают строки с информацией о каждой выданной или возвращенной книге.

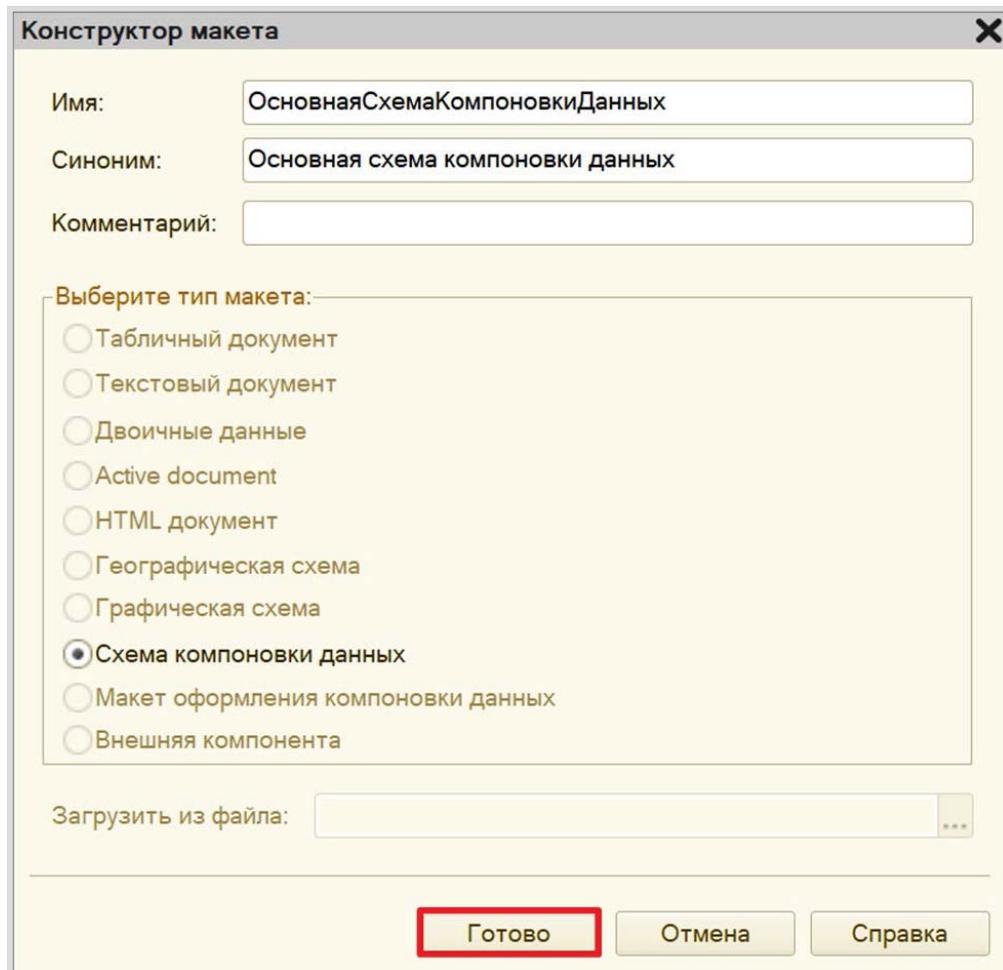
Последний шаг – построить отчет.

Определение

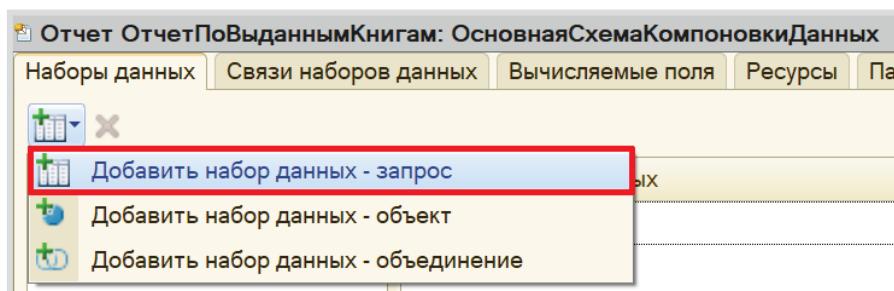
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Добавим новый отчет «ОтчетПоВыданнымКнигам» и откроем *схему компоновки данных*.

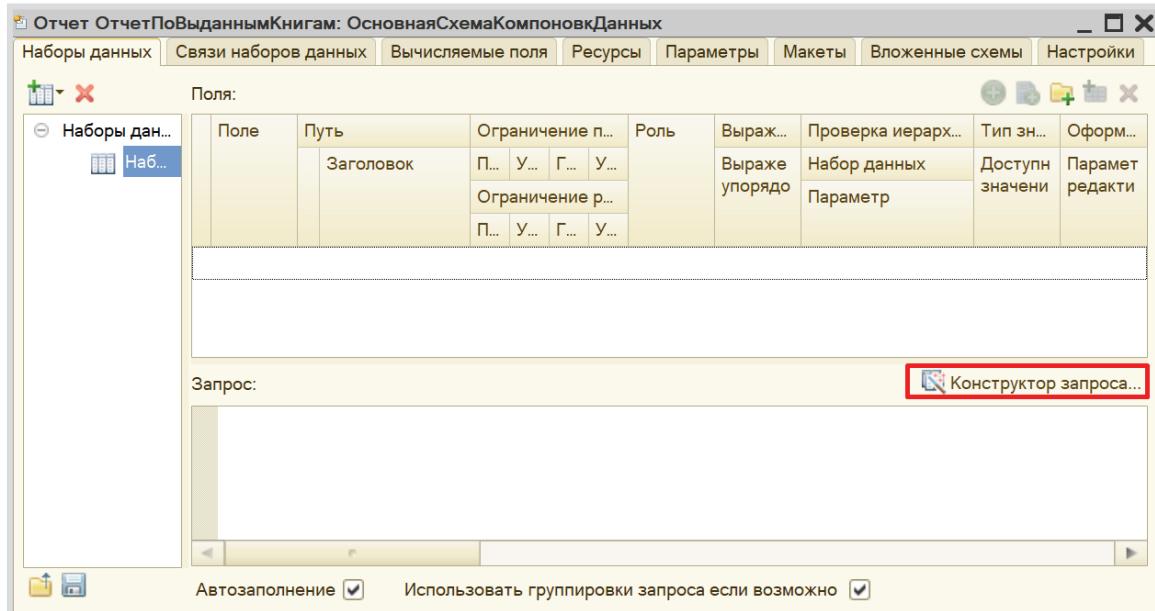




Добавим запрос к базе данных.



Воспользуемся конструктором запроса.



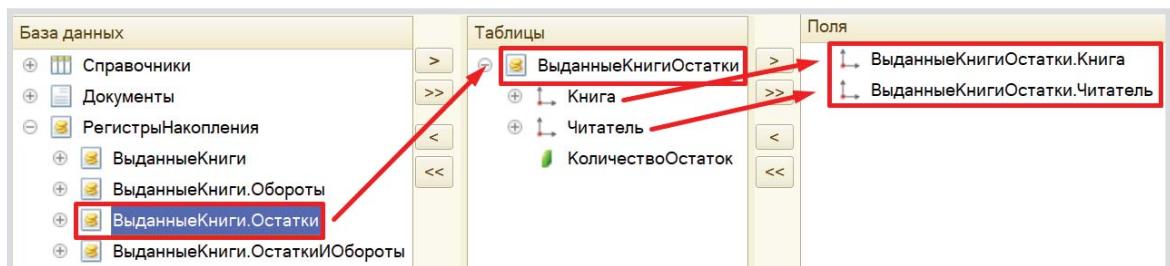
Открывается конструктор запроса. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

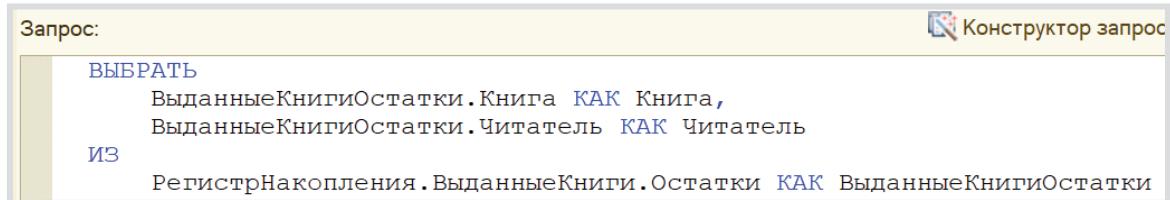
Данные будем брать не из *регистра накоплений* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица позволит получить уже просуммированные значения по всем документам.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

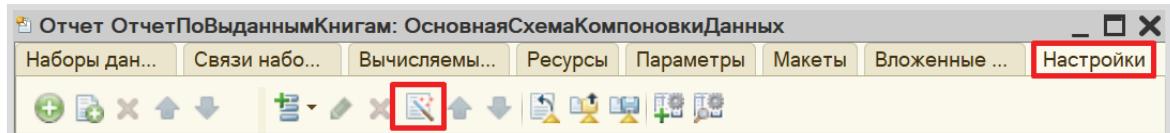
В результате данное окно должно быть заполнено следующим образом:



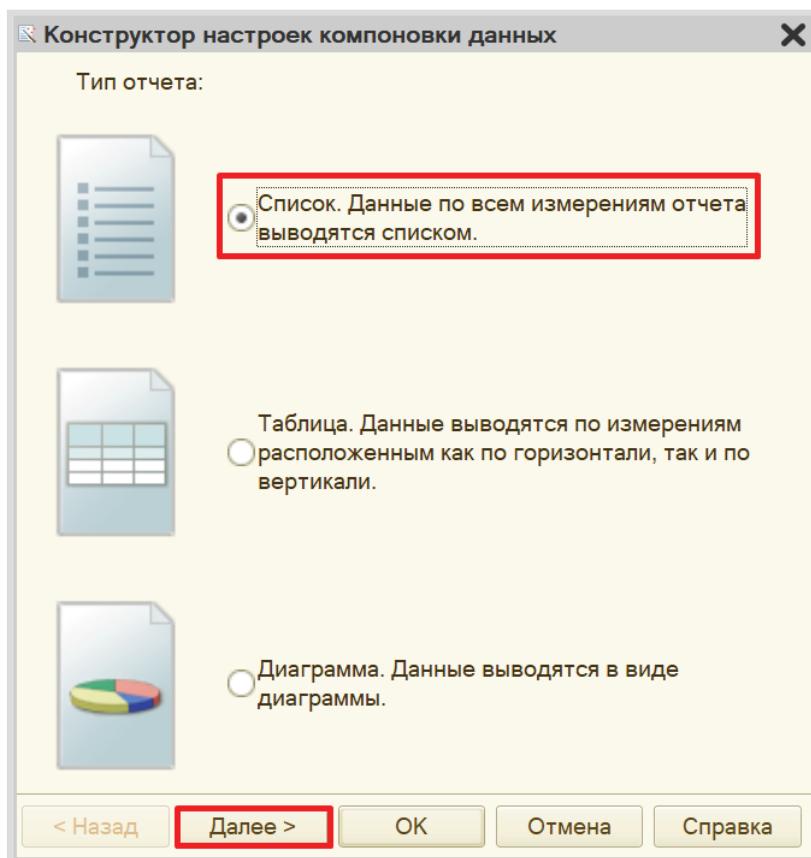
Нажмите на кнопку «OK». Получившийся запрос должен выглядеть следующим образом:



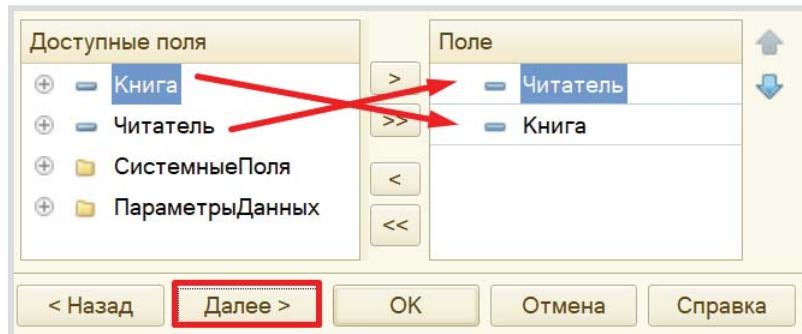
Переходим на вкладку «Настройки» для оформления внешнего вида отчета. Воспользуемся конструктором настроек отчета.



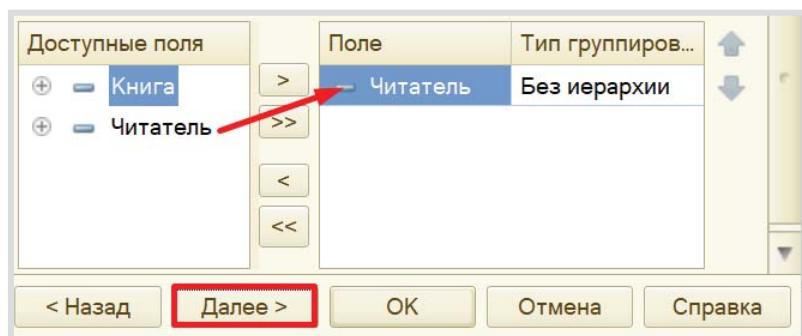
Отчет будем строить в виде списка.



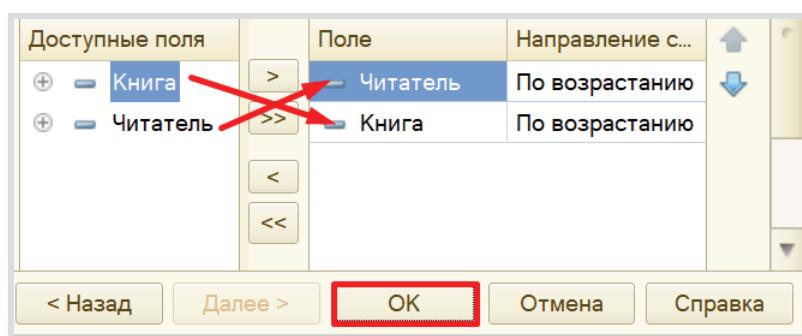
Выберем поля для отображения в отчете. Обязательно расставьте реквизиты в том порядке, в котором они должны быть в отчете. Для перемещения реквизитов воспользуйтесь стрелочками.



На следующем окне нужно выбрать группировку. Нас интересует группировка *по читателям*.



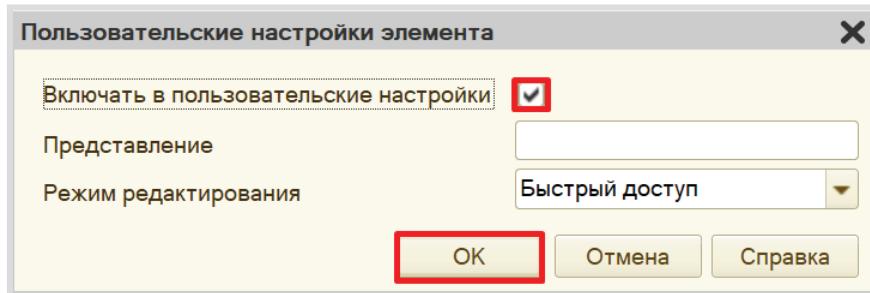
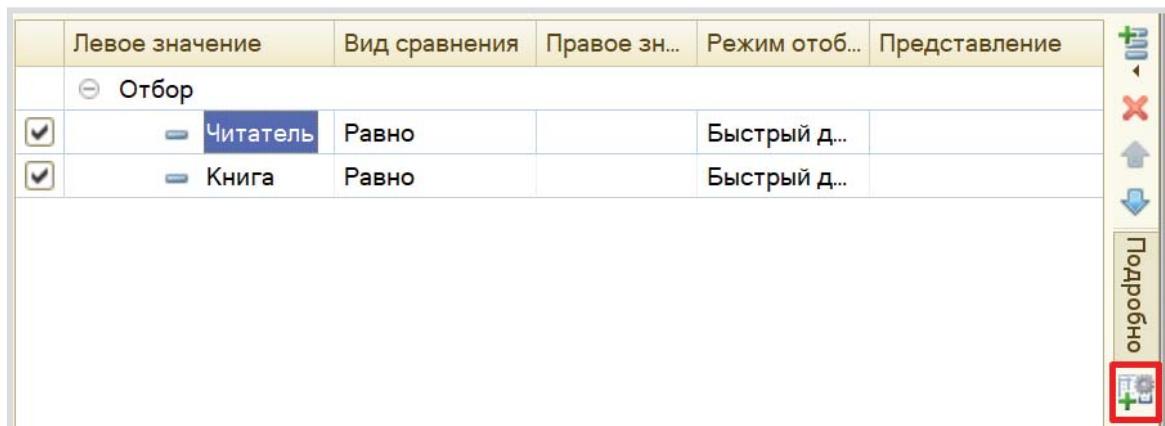
Ну, и на последнем окне необходимо установить упорядочивание. Сначала пусть происходит упорядочивание по читателям (имеется в виду упорядочивание по алфавиту), а затем – по книгам. Воспользуйтесь стрелочками для изменения порядка реквизитов.



Осталось лишь добавить пользователю возможность осуществлять отбор по некоторым полям. На той же вкладке «Настройки» найдите вкладку «Отбор» в нижней части окна и включите отбор по книге и читателю.



Теперь каждое из этих полей нужно сделать видимым для пользователя. Изменим свойства элемента пользовательских настроек.



Точно так же измените свойства пользовательских настроек и для второго реквизита отбора.

Теперь можно посмотреть на результат работы отчета в режиме «1С:Предприятие».

Для более наглядного вида отчета добавьте еще несколько документов о выдаче и возврате книг в библиотеку.

Отчет дает наглядное представление о том, кто из читателей какие книги взял.

The screenshot shows a software window titled "Отчет по выданным книгам". The interface includes a toolbar with back, forward, search, and other standard icons. Below the toolbar are four buttons: "Сформировать" (highlighted in yellow), "Выбрать вариант...", "Настройки...", and "Еще...". There are two dropdown menus: "Читатель:" and "Книга:". The main content area displays a hierarchical list of borrowed books:

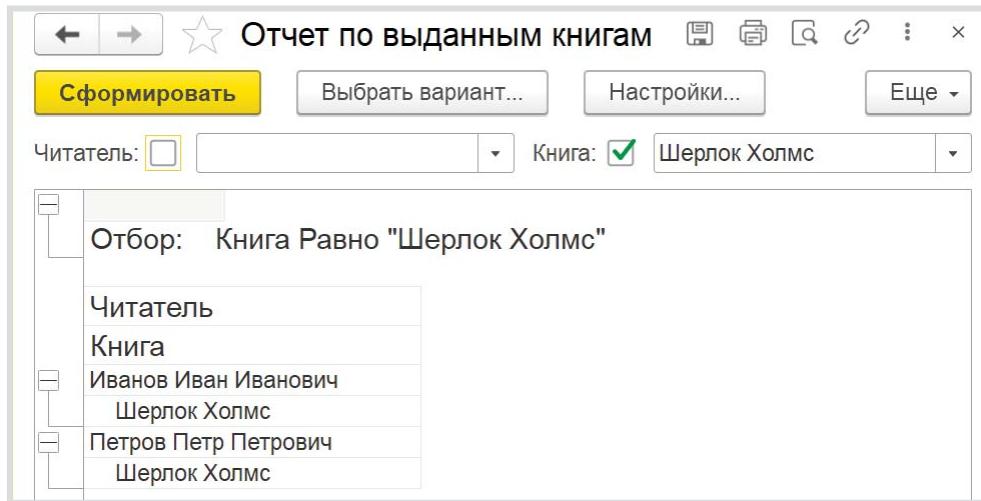
- Читатель**
- Книга**
- Александрова Александра Александровна
 - Алиса в стране чудес
 - Маленький принц
- Иванов Иван Иванович
 - Алиса в стране чудес
 - Моби Дик, или Белый кит
 - Шерлок Холмс
- Петров Петр Петрович
 - Шерлок Холмс

Теперь проверим работоспособность отбора. Сделаем отбор по читателю.

The screenshot shows the same software window after filtering by the reader "Иванов Иван Иванович". The "Читатель:" dropdown now has a checked checkbox next to "Иванов Иван Иванович". The main content area displays a hierarchical list of borrowed books, which is identical to the one in the previous screenshot but only shows the books borrowed by the selected reader:

- Отбор: Читатель Равно "Иванов Иван Иванович"**
- Читатель**
- Книга**
- Иванов Иван Иванович
 - Алиса в стране чудес
 - Моби Дик, или Белый кит
 - Шерлок Холмс

Ну, и попробуем сделать отбор по книге.



Поставленная задача решена.

Лабораторная работа № 13

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ НЕБОЛЬШОГО ТОРГОВОГО ПАВИЛЬОНА

Сложность: **

Теги: справочник, документ, форма, команда,
критерий отбора, схема компоновки данных

ЗАДАНИЕ

Заказчик просит разработать информационную систему для небольшого торгового павильона.

1. Необходимо зарегистрировать заказ товаров. В момент телефонного звонка оператор вводит в систему следующие данные:

- ФИО заказчика;
- номер телефона;
- перечень заказанных товаров.

2. Нужно регистрировать выдачу товаров. Оператор выделяет один или несколько заказов из списка и нажимает на кнопку «Создать выдачу заказов». Должна открыться новая форма с перечнем выделенных заказов, которую оператор может отредактировать и сохранить.

3. Также необходимо сформировать отчет, который выведет список всех документов, в которых был отмечен тот или иной товар, выбранный пользователем.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

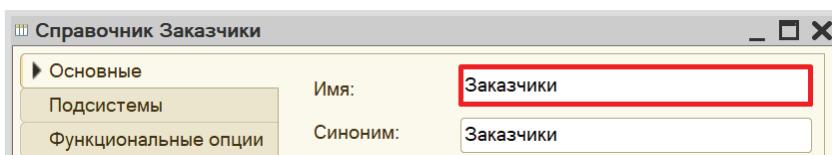
Выполнение

Исходя из условия, нам необходимо хранить информацию о товарах и заказчиках. Для этой цели в «1С:Предприятии» существует объект конфигурации *справочник*.

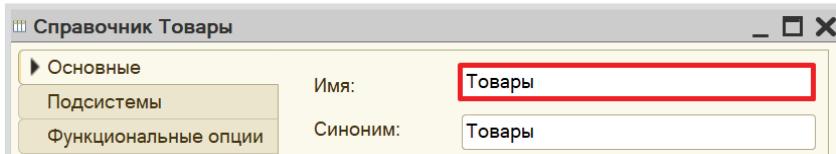
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

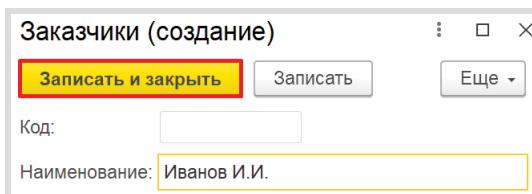
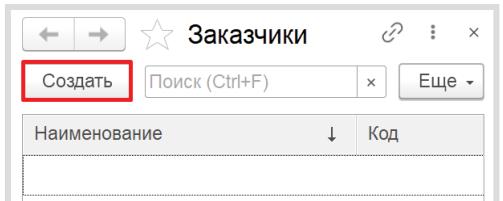
Добавим справочник «Заказчики».



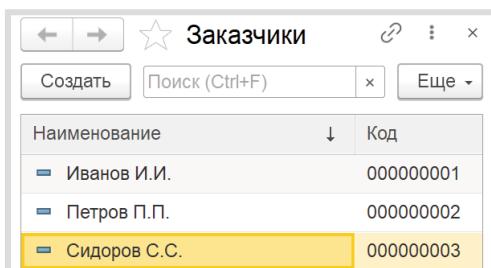
Добавим справочник «Товары».



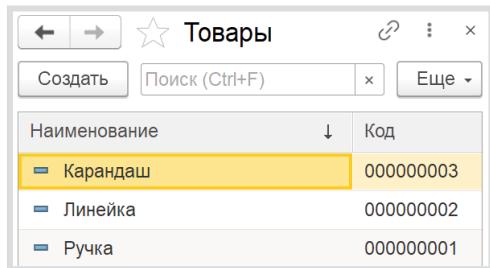
Запустим информационную базу в режиме «1С:Предприятие» и добавим несколько элементов в каждый справочник.



Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



Аналогично добавим несколько элементов в справочник «Товары».



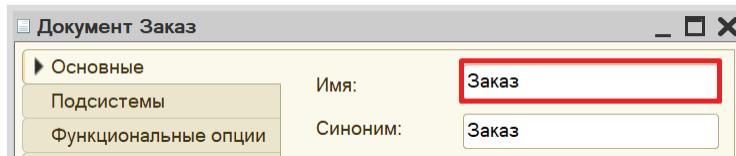
Таким образом, мы организовали хранение информации о заказчиках и товарах.

Далее нам нужно реализовать в системе формирование заказов. Для этого воспользуемся объектом конфигурации *документ*.

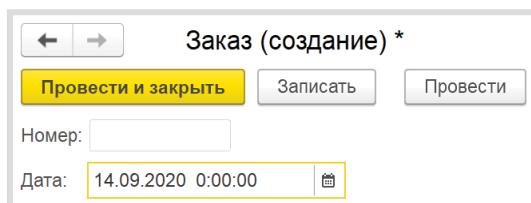
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим документ «Заказ».



Если запустить режим «1С:Предприятие», то форма документа будет выглядеть следующим образом:

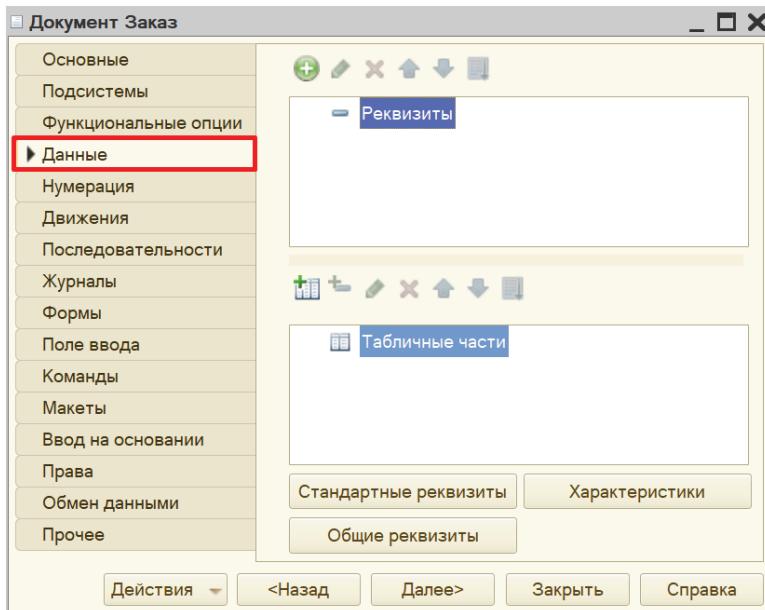


Легко заметить, что система сгенерировала для документа другие стандартные реквизиты: «Номер» и «Дата». Оба поля заполняются автоматически, дата может быть изменена.

Любой документ может находиться в одном из двух состояний: *подготовленный к свершению* или *совершенный*:

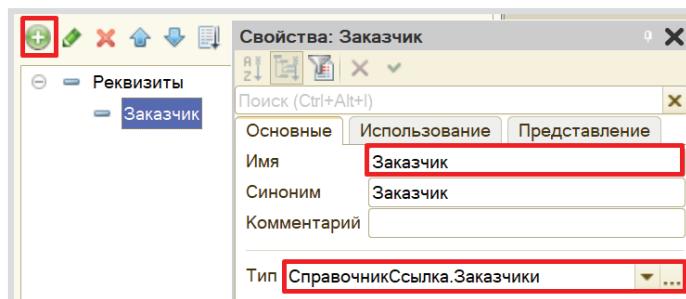
- Чтобы подготовить документ для использования в будущем, следует его записать;
- Чтобы отметить документ как совершенный – провести.

Полей документа «Номер» и «Дата» для решения поставленной задачи недостаточно. Заказчику необходимо фиксировать множество различных данных при формировании заказа. Для настройки структуры документа переходим на вкладку «Данные».



В документе обязательно нужно фиксировать имя заказчика.

Для этого нужно добавить реквизит «Заказчик» с типом «СправочникСсылка.Заказчики». Таким образом, пользователь сможет выбирать только элементы из справочника «Заказчики».

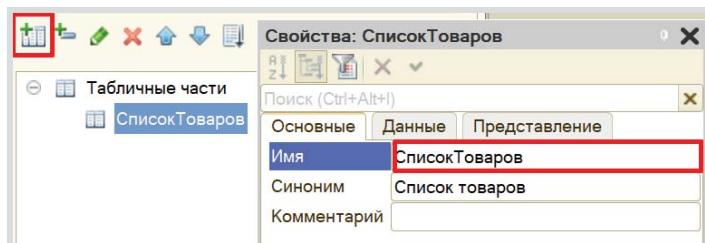


Кроме того, в документе нужно хранить перечень товаров, которые заказчик хочет заказать.

Конечно, мы можем создать несколько отдельных реквизитов. Но может получиться такая ситуация, когда заказчик решит заказать большее количество товаров, чем добавлено

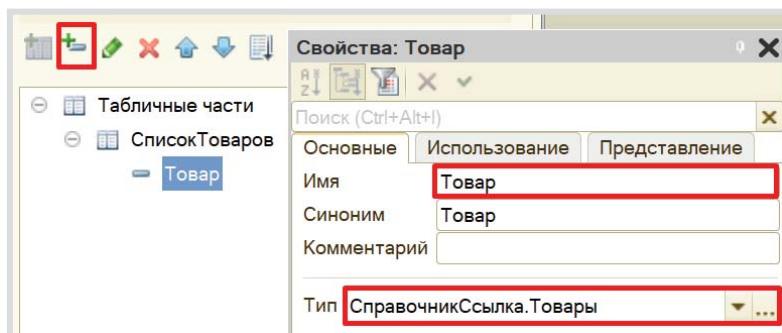
реквизитов. Идея с добавлением реквизитов нам не подходит. Поэтому перечень товаров логично разместить в табличной части справочника.

Добавим новую табличную часть «СписокТоваров».

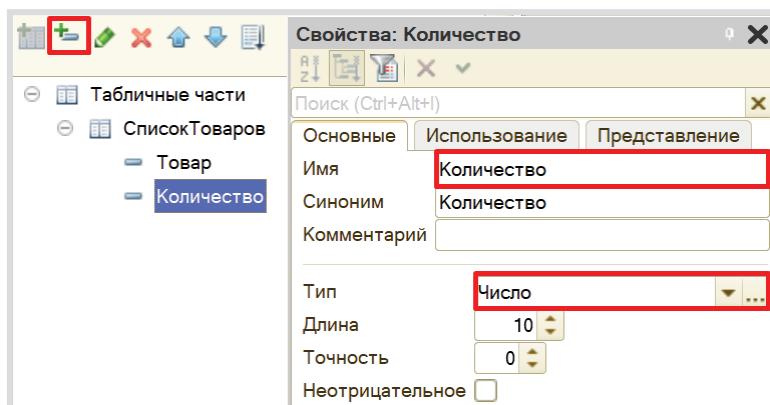


Если сейчас посмотреть на табличную часть документа в режиме «1С:Предприятие», то сама табличная часть будет совершенно пуста, в нее нельзя внести какие-либо данные. Необходимо добавить колонки для табличной части. Для этого нужно воспользоваться реквизитами табличной части.

Добавим реквизит табличной части «Товар» с типом «СправочникСсылка.Товары».



Добавим также количество для каждого выбранного в заказ товара. Аналогично создайте новый реквизит табличной части «Количество» с типом «Число».

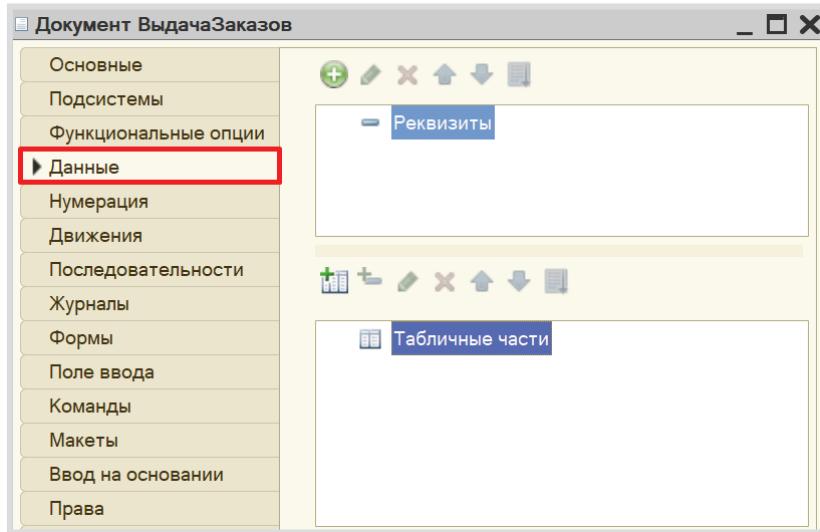


Перейдем в режим «1С:Предприятие» и зафиксируем несколько заказов.

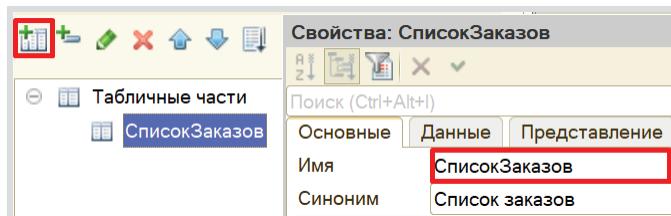
Далее следует приступить к созданию возможности фиксировать выдачу заказов.

Добавим еще один документ – «ВыдачаЗаказов».

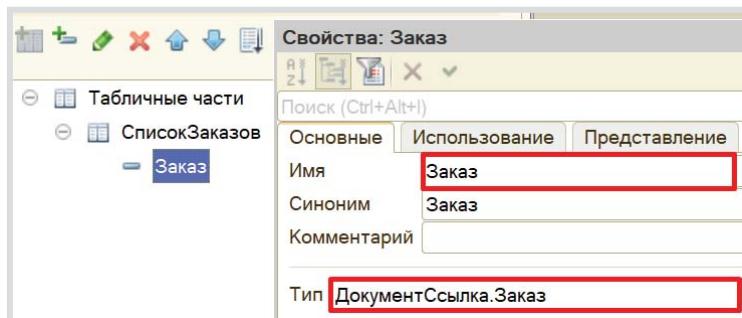
Для настройки структуры документа переходим на вкладку «Данные».



В документе нужно хранить перечень заказов, которые нужно выдать. Добавим табличную часть «СписокЗаказов».

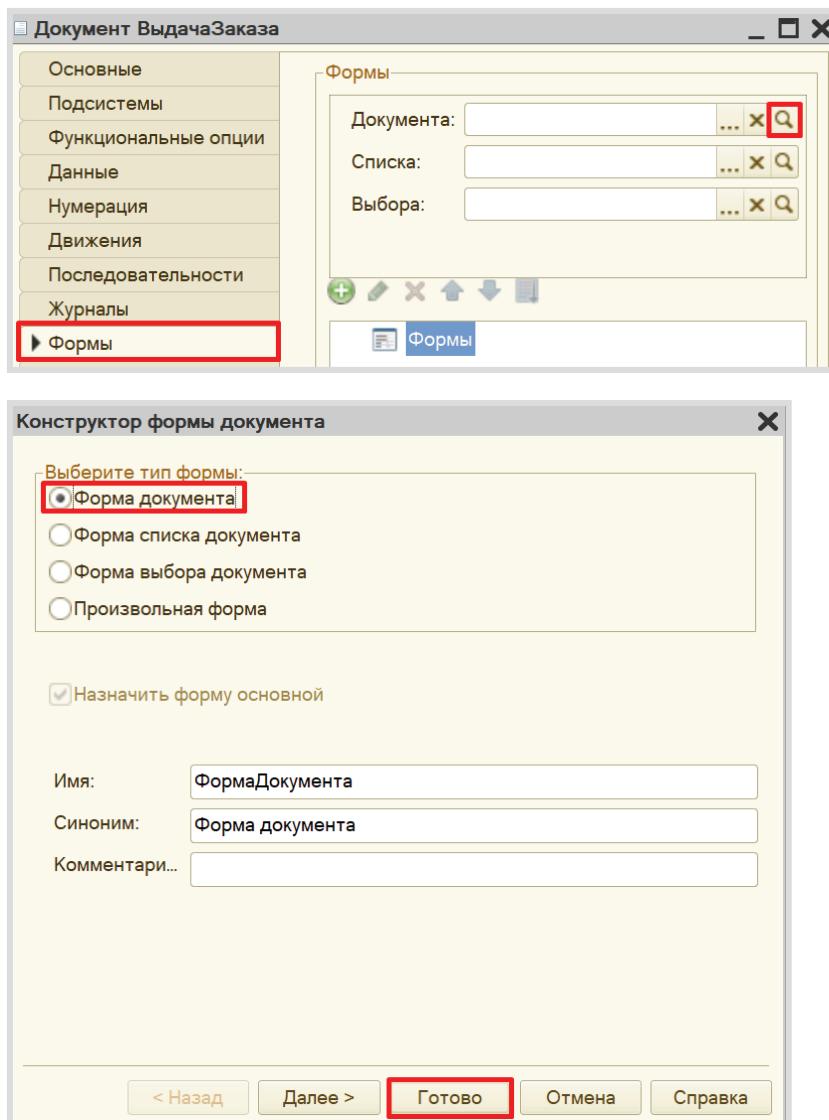


В табличной части следует добавить реквизит «Заказ» с типом «ДокументСсылка.Заказы».



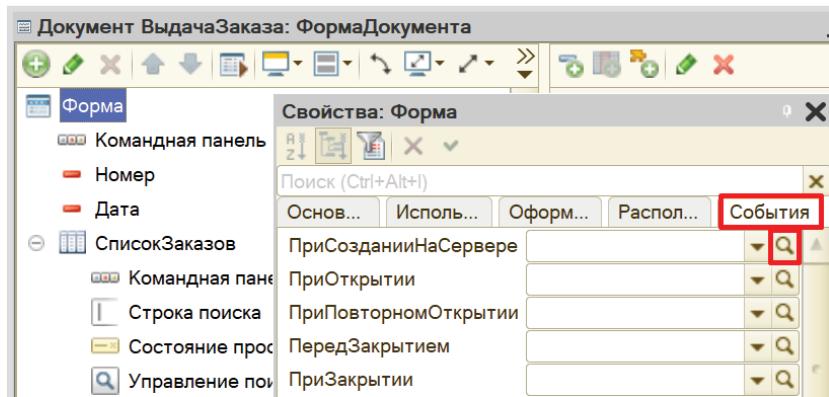
Теперь нужно создать форму документа. Это необходимо сделать для того, чтобы изменить стандартное поведение формы, описав свой алгоритм действий.

Переходим на вкладку «Формы» и создадим новую форму документа.



Открывается конструктор формы.

Здесь можно изменить расположение элементов, их отображение, размер и т. д. Наша задача заключается в том, чтобы изменить стандартное поведение данной формы. Для этого в списке элементов находим форму, а в палитре свойств – событие «ПриСозданииНаСервере» на вкладке «События».



Откроется модуль формы с шаблоном процедуры. То, что будет написано в данной процедуре будет выполняться платформой вместо ее стандартных настроек при создании нового документа на сервере.

Нужно написать следующее:

```
Документ ВыдачаЗаказа: ФормаДокумента
```

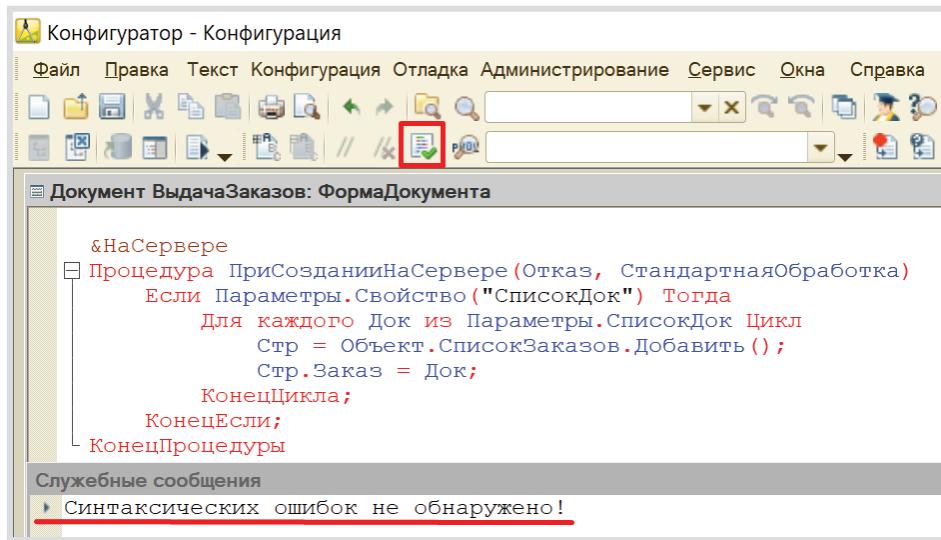
```
&НаСервере
Процедура ПриСозданииНаСервере (Отказ, СтандартнаяОбработка)
    Если Параметры.Свойство ("СписокДок") Тогда
        Для каждого Док из Параметры.СписокДок Цикл
            Стр = Объект.СписокЗаказов.Добавить ();
            Стр.Заказ = Док;
        КонецЦикла;
    КонецЕсли;
КонецПроцедуры
```

При создании нового документа будет отслеживаться свойство «СписокДок». Если такое свойство присутствует, тогда данная процедура заполнит табличную часть документа заказами.

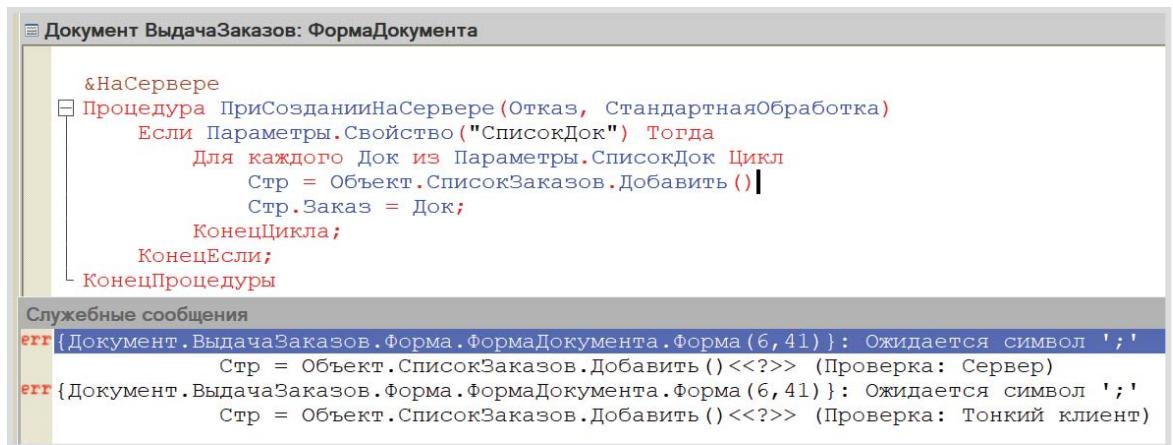
Внимание!

Обязательно проверьте модуль на наличие синтаксических ошибок.

Для этого нажмите на кнопку «Проверка модуля». Должно открыться окно «Служебные сообщения». Если синтаксических ошибок не обнаружено, то в данном окне появится надпись «Синтаксических ошибок не обнаружено».



Если же появится ошибка, то нужно внимательно ее изучить и щелкнуть по ней дважды правой кнопкой мыши. Курсор автоматически переместится на строку, в которой была обнаружена ошибка. Исправьте ошибку.



Исправляйте ошибки до тех пор, пока не появится сообщение об отсутствии ошибок.

```

Документ ВыдачаЗаказов: ФормаДокумента

&НаСервере
Процедура ПриСозданииНаСервере (Отказ, СтандартнаяОбработка)
    Если Параметры.Свойство ("СписокДок") Тогда
        Для каждого Док из Параметры.СписокДок Цикл
            Стр = Объект.СписокЗаказов.Добавить ()  
■
            Стр.Заказ = Док;
        КонецЦикла;
    КонецЕсли;
КонецПроцедуры

Служебные сообщения
err {Документ.ВыдачаЗаказов.Форма.ФормаДокумента.Форма (6, 41)}: Ожидается символ ';'
Стр = Объект.СписокЗаказов.Добавить ()<<?>> (Проверка: Сервер)
err {Документ.ВыдачаЗаказов.Форма.ФормаДокумента.Форма (6, 41)}: Ожидается символ ';'
Стр = Объект.СписокЗаказов.Добавить ()<<?>> (Проверка: Тонкий клиент)
▶ Синтаксических ошибок не обнаружено!

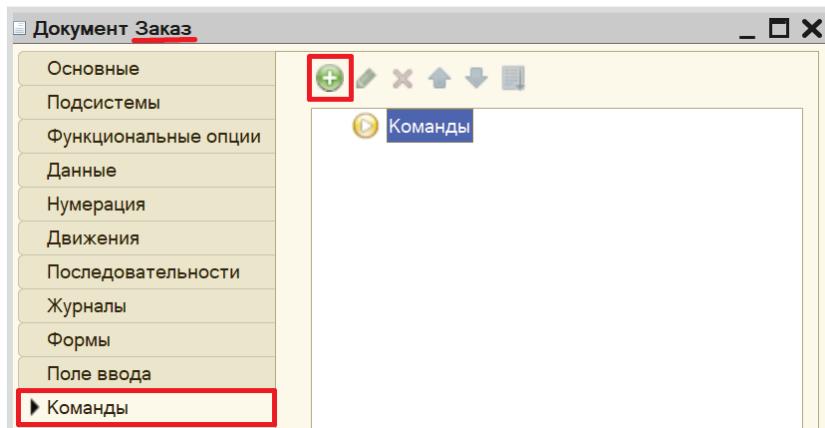
```

На этом работа с документом «ВыдачаЗаказов» завершена.

Откроем окно редактирования документа «Заказ».

Нам нужно, чтобы оператор смог выбрать один или несколько заказов из списка заказов, нажать на кнопку «Создать выдачу заказов». Система должна сформировать документ «ВыдачаЗаказов» с перечнем всех выбранных оператором заказов. Чтобы создать такую кнопку и запустить формирование нового документа необходимо создать команду.

Перейдем на вкладку «Команды» и добавим новую команду.

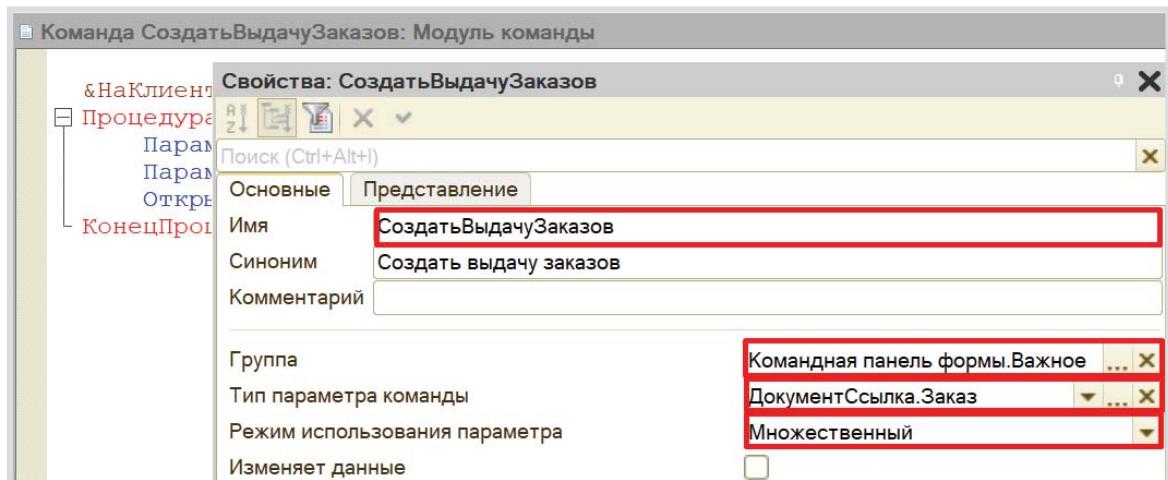


Должен открыться модуль команды. В первую очередь, дадим команде имя «СоздатьВыдачуЗаказов» в палитре свойств. Название модуля команды должно измениться.

Установите свойство «Группа» в значение «Командная панель формы.Важное». Таким образом, мы определили место, где созданная команда будет отображена на форме.

Установите свойство «Тип параметра команды» в значение «ДокументСсылка.Заказ». Команда становится параметризируемой, и будет доступна везде, где будет использоваться параметр команды, то есть наш документ «Заказ».

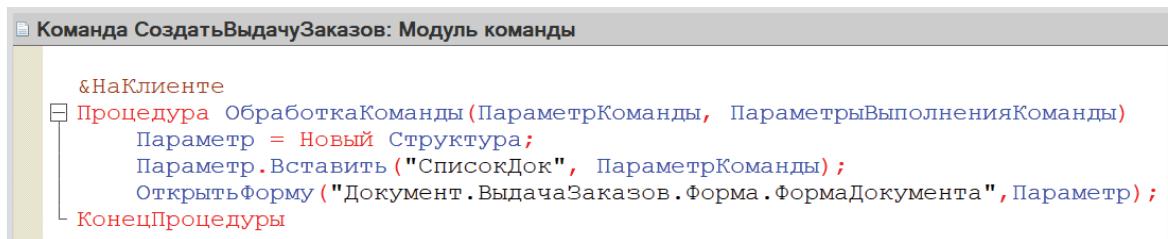
Установите свойство «Режим использования параметра» в значение «Множественный». Это значит, что оператор сможет выбрать сразу несколько документов и запустить данную команду.



Теперь, когда все настройки установлены, переключимся на модуль команды. Нам нужно открыть форму документа «ВыдачаЗаказов» и передать туда параметр.

Определение

Параметр команды — это набор документов, которые выделит пользователь.

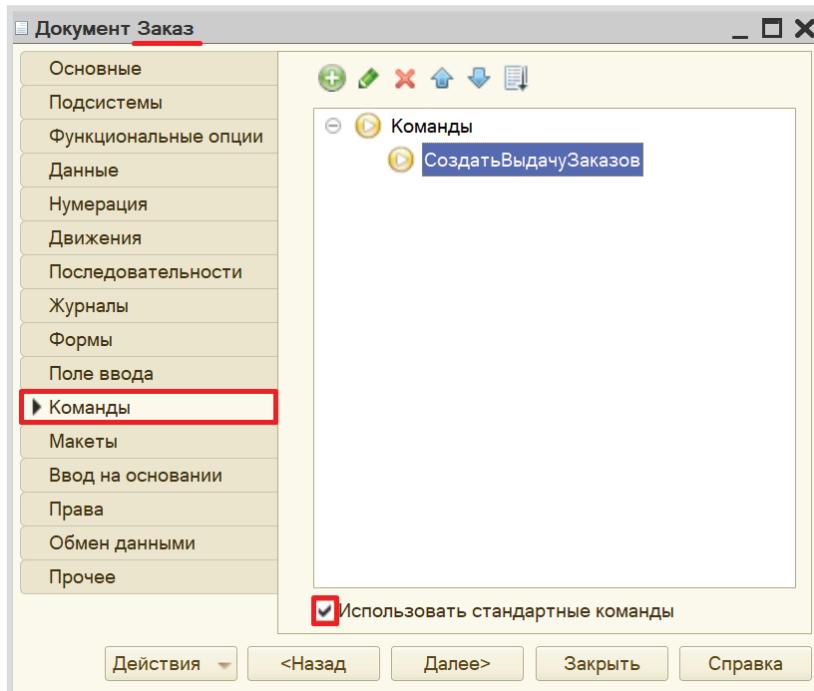


Данная команда будет делать следующее: оператор выделяет один или несколько заказов из списка и нажимает на кнопку «Создать выдачу заказов». Откроется форма создания нового документа «ВыдачаЗаказов» с заполненной табличной частью. Причем в табличной части будут находиться выделенные оператором заказы.

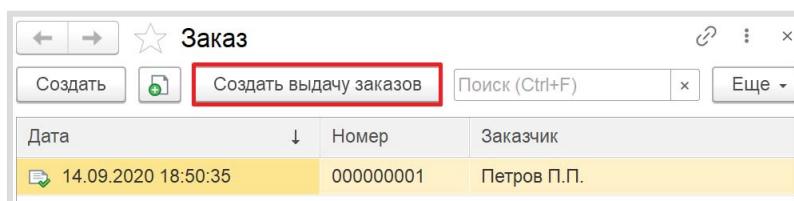
Внимание!

Не забудьте проверить модуль на наличие синтаксических ошибок!

Обязательно вернитесь к окну редактирования документа «Заказ» на вкладку «Команды» и установите галочку «Использовать стандартные команды».



Проверим работоспособность команды в режиме «1С:Предприятие». Попробуем создать выдачу заказа.



Открывается форма документа «Выдача Заказа», в табличной части которого находится выбранный заказ.

Обратите внимание, что форма документа заказа может быть открыта прямо из табличной части.
Это поможет оператору при выдаче быстро проверить содержимое заказа.

После редактирования форму можно провести и закрыть.

Последнее требование нашего заказчика – поиск того или иного товара среди всех документов «Заказ».

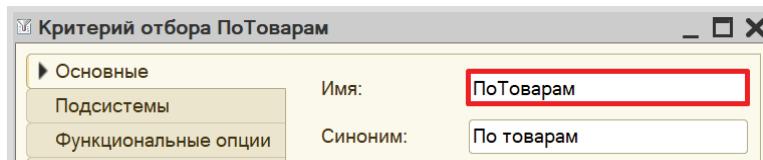
Для выполнения данной задачи будем использовать *отчет*. В отчете пользователь сможет выбрать интересующий его товар и получить список всех документов, содержащих данный товар в табличной части. Документы должны быть доступны для быстрого просмотра.

Для начала нужно создать критерий отбора. Он предназначен для описания некоторого правила выборки информации из объектных данных различных типов.

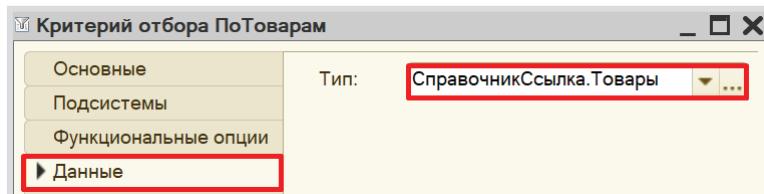
Добавим новый критерий отбора «По Товарам».

ПОДСКАЗКА

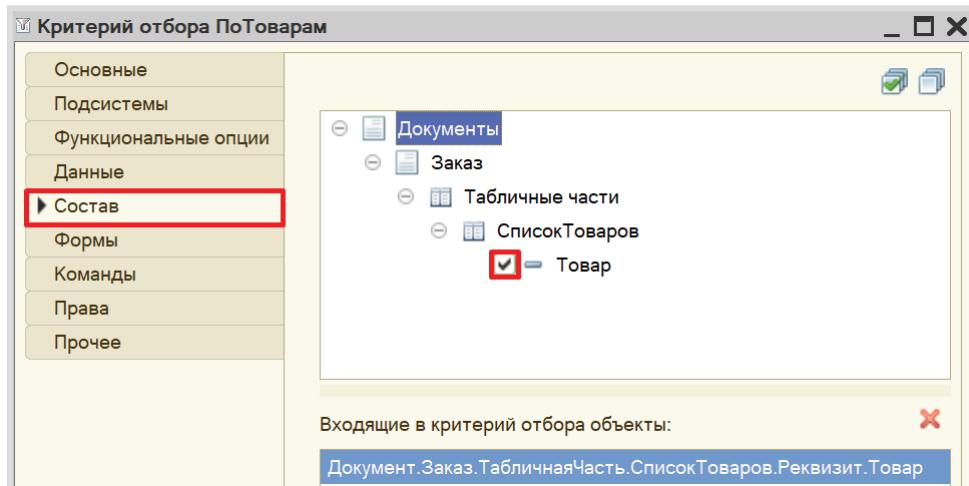
Критерии отбора находятся во вкладке «Общие» дерева конфигурации.



Перейдем на вкладку «Данные» и укажем тип значения для данного отбора. Очевидно, если мы хотим делать отбор по элементам справочника «Товары», то именно такого типа данные нам и нужны в качестве критерия отбора.



Определим состав критерия отбора на вкладке «Состав». Таким образом, мы определим, где именно система будет искать тот или иной товар в табличной части документа «Заказ».

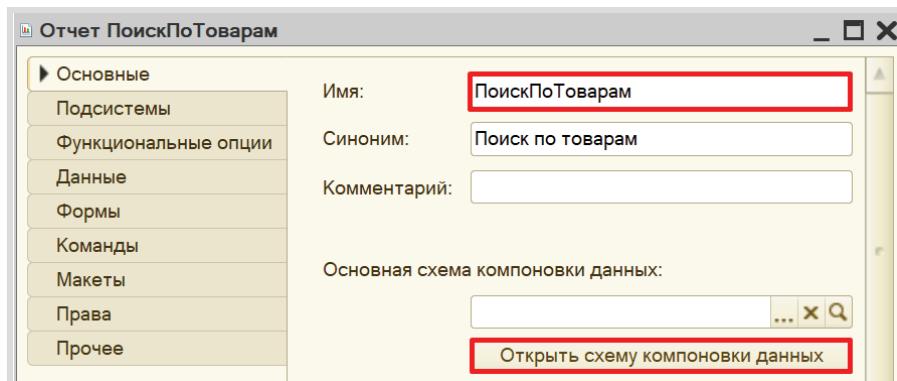


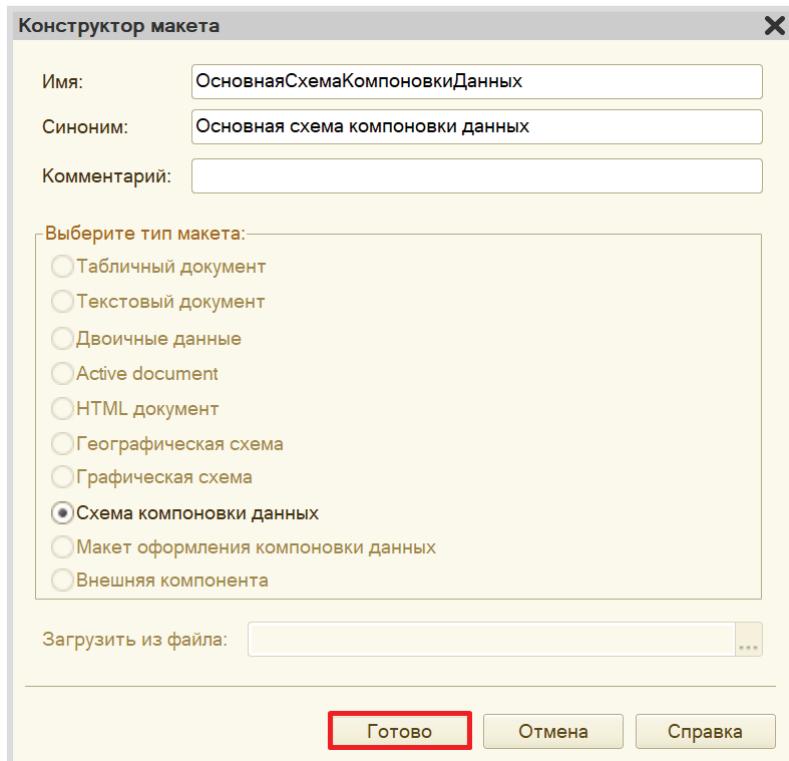
Теперь, когда критерий отбора создан, можно создать отчет.

Определение

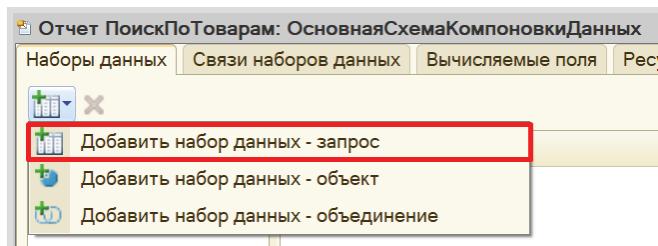
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: [https://v8.1c.ru/platorma/отчет/](https://v8.1c.ru/platorma/otchet/)).

Добавим новый отчет «ПоискПоТоварам». Откроем схему компоновки данных.

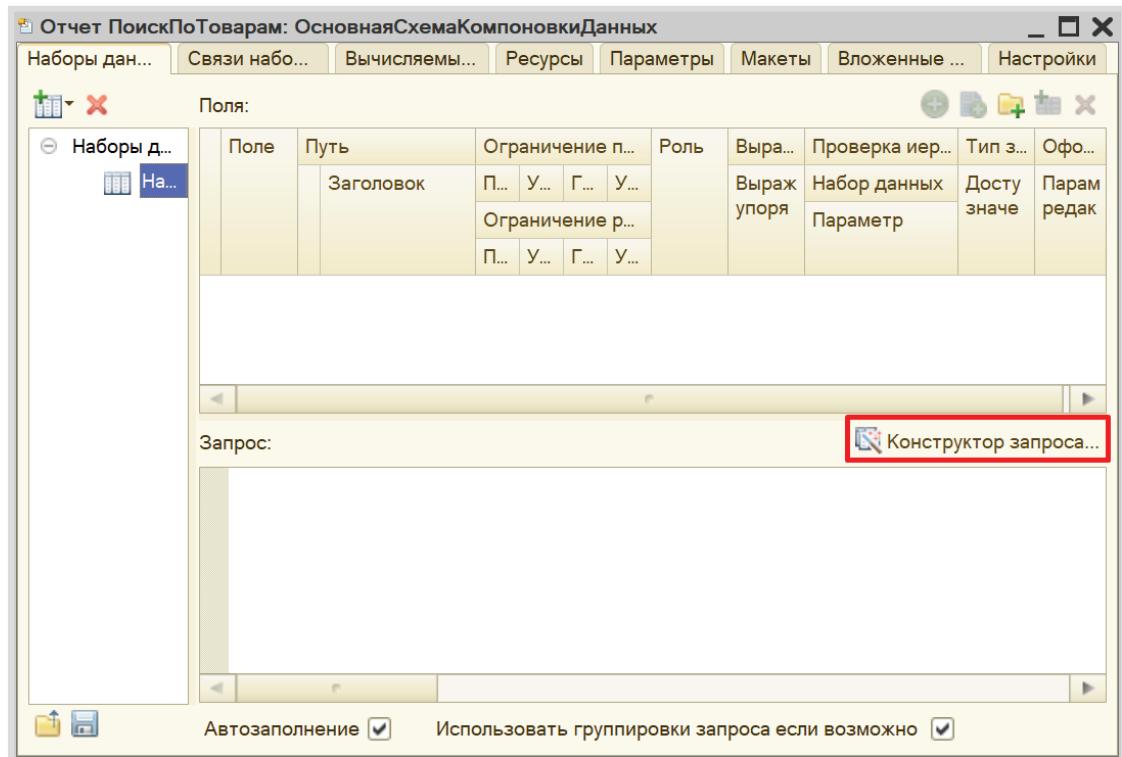




Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Для составления запроса воспользуемся *конструктором запроса*.

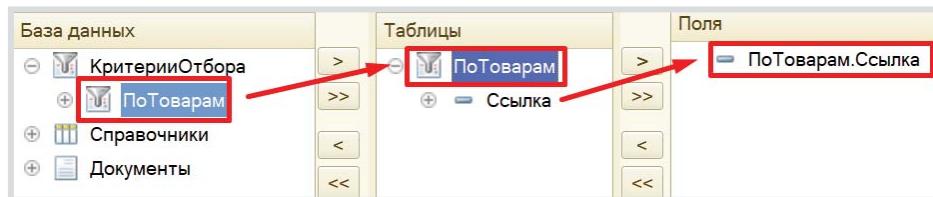


Открывается *конструктор запроса*. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

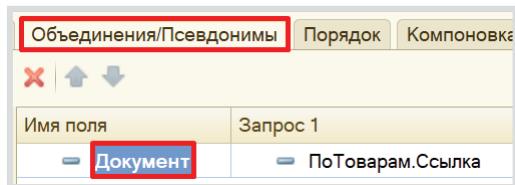
Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:

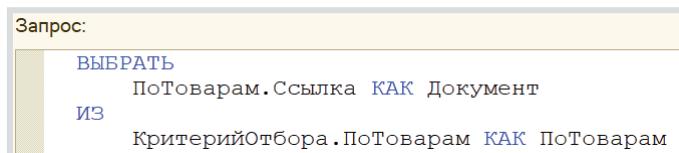


Переходим на вкладку «Объединения/Псевдонимы». Здесь мы можем изменить название реквизита, чтобы пользователю было понятнее, что именно он будет наблюдать в отчете.

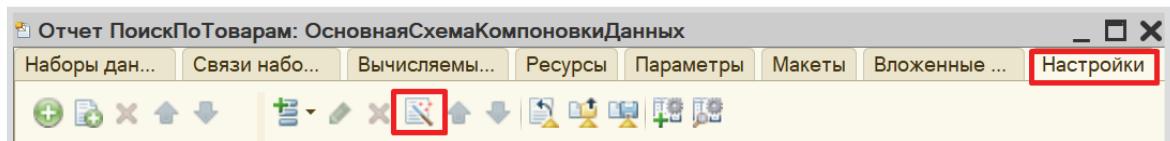
Изменим имя поля «Ссылка» на «Документ».



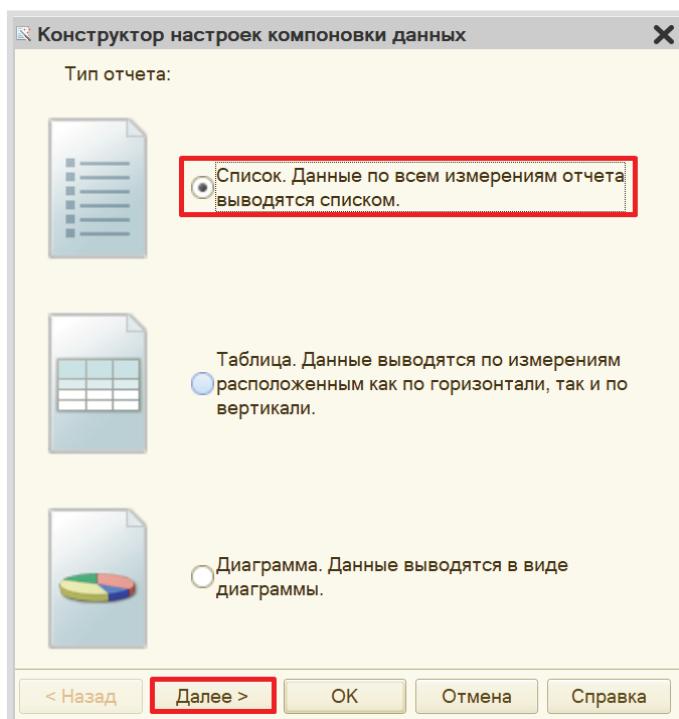
Закрываем конструктор запроса, нажав на кнопку «OK». Получившийся запрос должен выглядеть следующим образом:



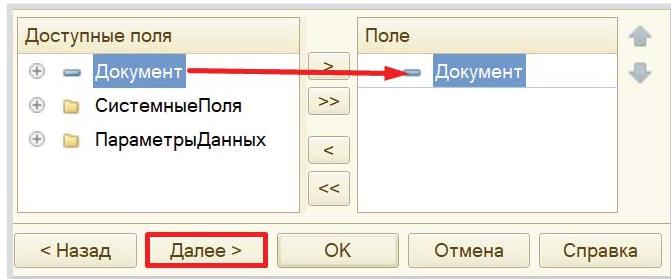
Переходим на вкладку «Настройки» для формирования внешнего вида отчета. Воспользуемся конструктором настроек отчета.



Создадим отчет в виде списка.



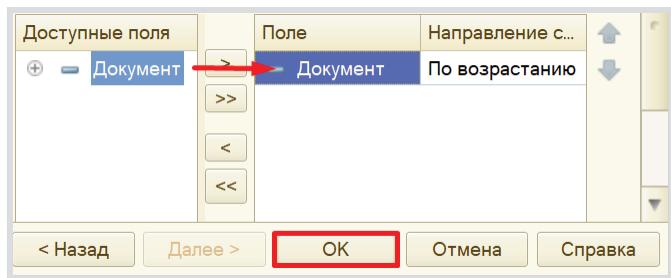
Вынесем реквизит «Документ» для отображения в отчете.



Группировку устанавливать не будем. Переходим к следующему пункту.

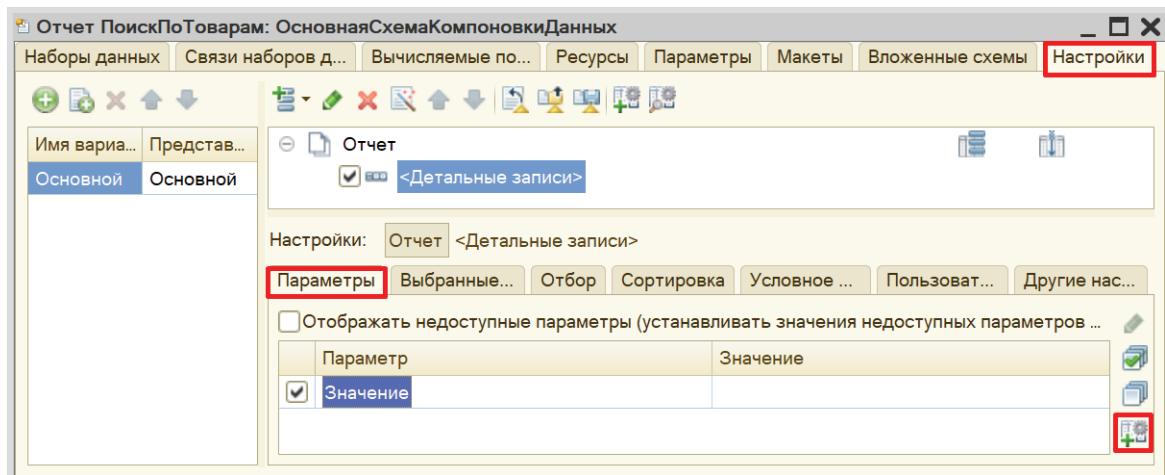


Установим сортировку документов по возрастанию.

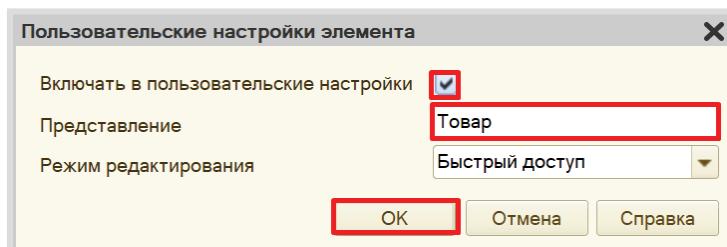


Далее в нижней части экрана вкладки «Настройки» найдем вкладку «Параметры» и вызовем «Свойства элемента пользовательских настроек» для параметра «Значение».

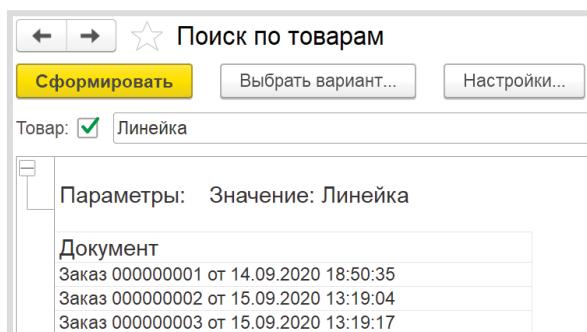
Данный параметр система формирует автоматически при работе с критериями отбора.



В открывшемся окне проставим галочку, чтобы данный параметр стал доступен пользователю для изменения. Также дадим этому параметру псевдоним «Товар».



Теперь можно перейти в режим «1С:Предприятие» и проверить работоспособность отчета.



Вы можете убедиться в правильности работы отчета. Двойным щелчком откройте любой из документов и найдите в перечне товаров «Линейку».

Поставленная задача решена.

Лабораторная работа № 14

**РАЗРАБОТКА КОНФИГУРАЦИИ
ДЛЯ УЧЕТА ПРОДАЖ ТОВАРОВ
С СОПУТСТВУЮЩИМИ
УСЛУГАМИ ПОКУПАТЕЛЯМ**

Сложность: **

Теги: справочник, документ, функциональная опция, схема компоновки данных

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета продаж товаров с сопутствующими услугами покупателям. Необходимо предусмотреть опциональную возможность использования различных валют.

При многовалютном учете пользователь системы при оформлении продажи должен обязательно указать валюту. Итоговая стоимость заказа должна формироваться автоматически.

Следует построить *Отчет по продажам* с возможностью выбора нужной валюты.

Форма отчета:

Валюта: Рубль		Валюта: Доллар	
Контрагент	Сумма	Контрагент	Сумма
ООО "Мак"	43 500	ООО "Василёк"	175
ООО "Василёк"	11 000	Итого	175
Итого	54 500		

Отчет выводит информацию по выбранной валюте, а также подводит общий итог.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

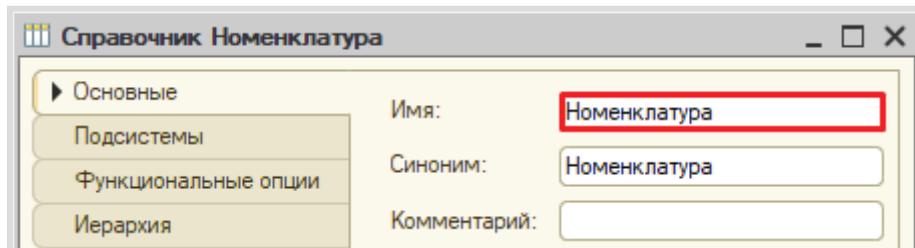
Выполнение

Из условия следует, что необходимо хранить информацию о покупателях, товарах и услугах, а также валютах. Для решения этой задачи нам понадобятся *справочники*.

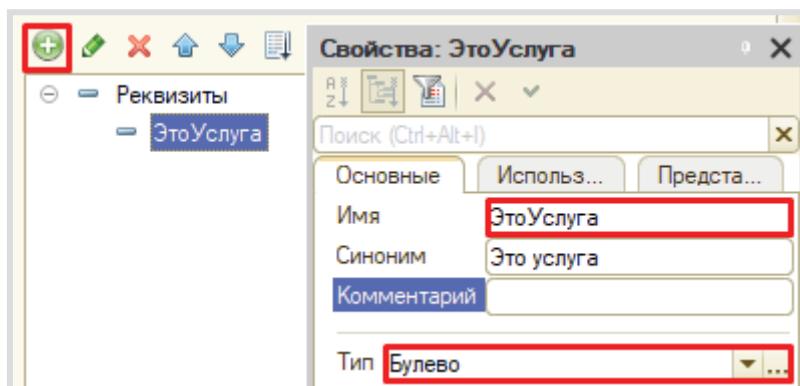
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Создадим справочник «Номенклатура».



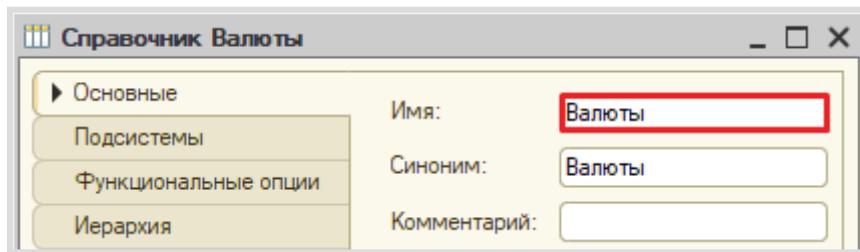
Перейдем на вкладку «Данные» и добавим реквизит «ЭтоУслуга», тип – «Булево». Данный реквизит необходим для того, чтобы отличать товары от услуг в справочнике.



Создадим справочник «Контрагенты».



Создадим справочник «Валюты».

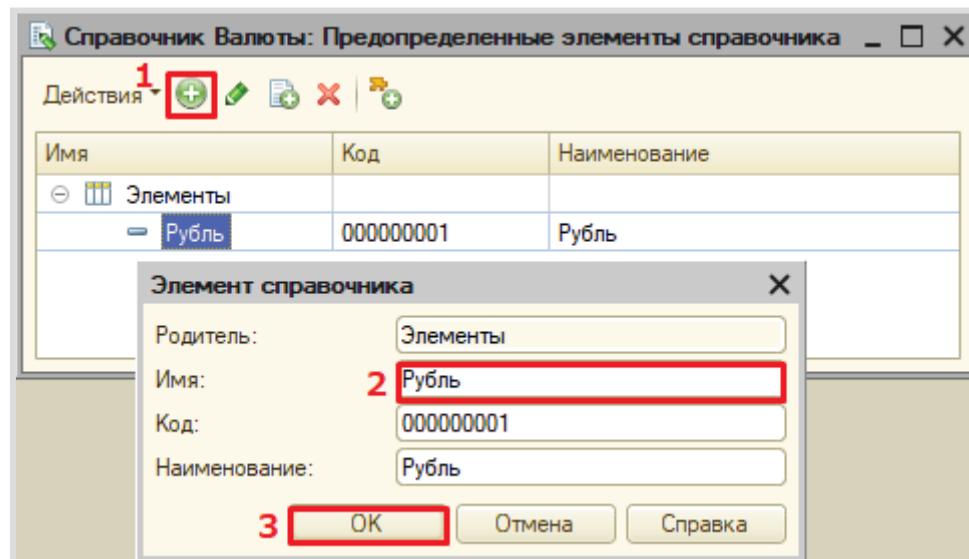
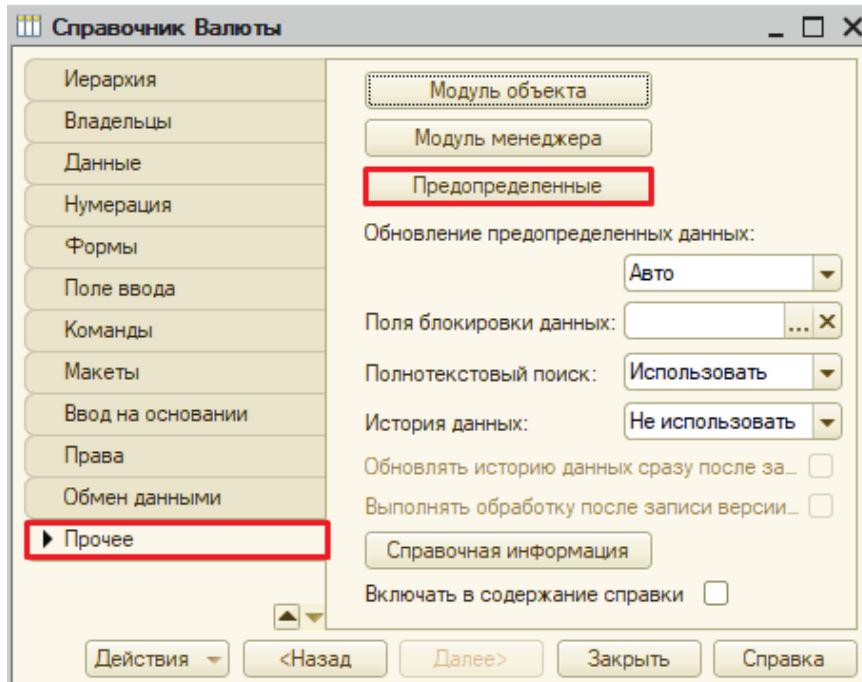


Обязательной валютой в данной информационной системе является рубль. Создадим предопределенный элемент.

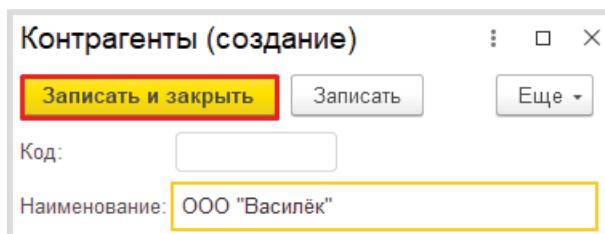
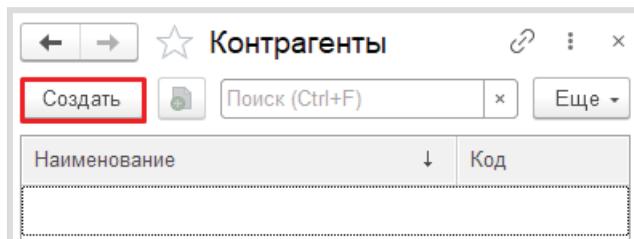
Определение

Предопределенные элементы – это такие элементы, которые создает разработчик в конфигураторе для удобства работы пользователя.

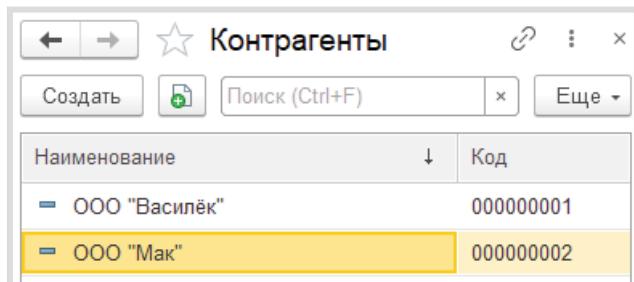
Созданный таким образом элемент будет доступен пользователю с первого запуска программы. Для создания такого элемента перейдем на вкладку «Прочее» и откроем список предопределенных элементов справочника.



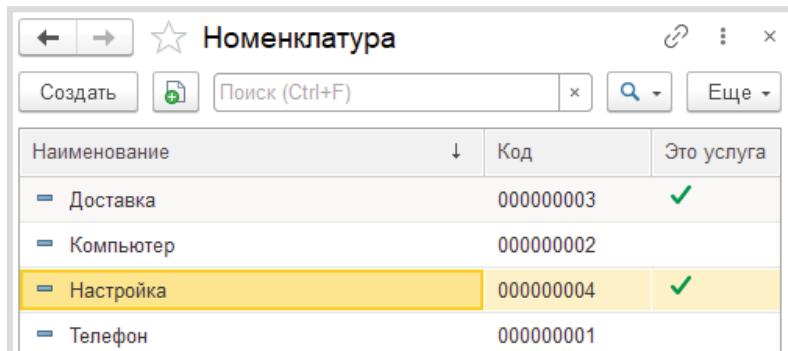
Откроем программу в режиме «1С:Предприятие» и добавим в каждый справочник несколько элементов.

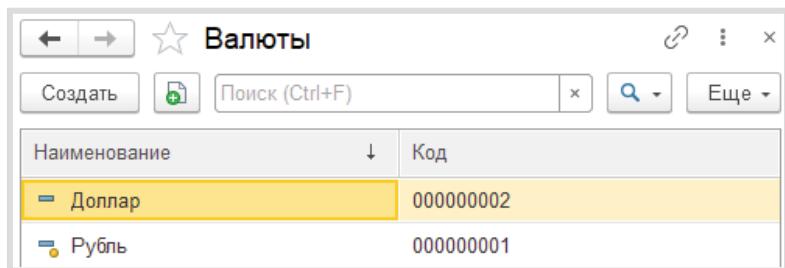


Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



Аналогично добавим элементы в справочники «Номенклатура» и «Валюты».





Заметим, что предопределенные элементы в справочнике отмечены желтым маркером.

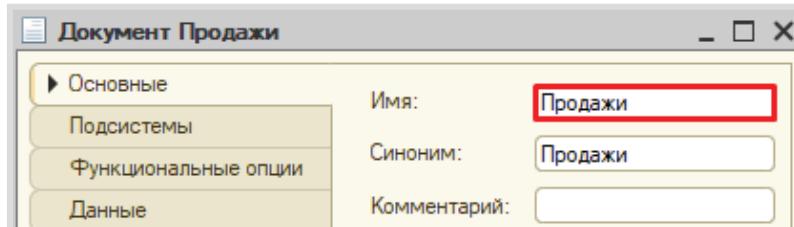
«Заказчик просит разработать конфигурацию для учета продаж товаров с сопутствующими услугами покупателям».

Для регистрации продаж следует воспользоваться объектом конфигурации *документ*.

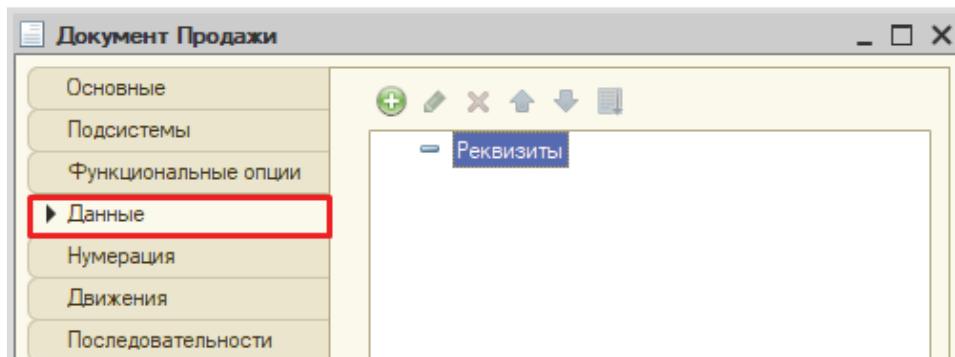
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty>).

Добавим новый документ «Продажи».



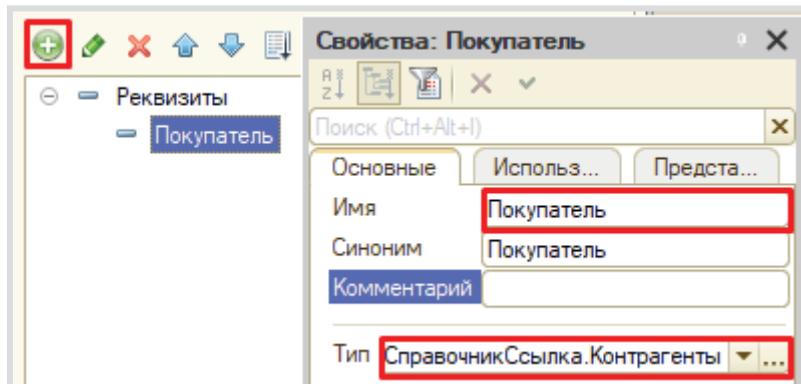
Для настройки структуры документа переходим на вкладку «Данные»



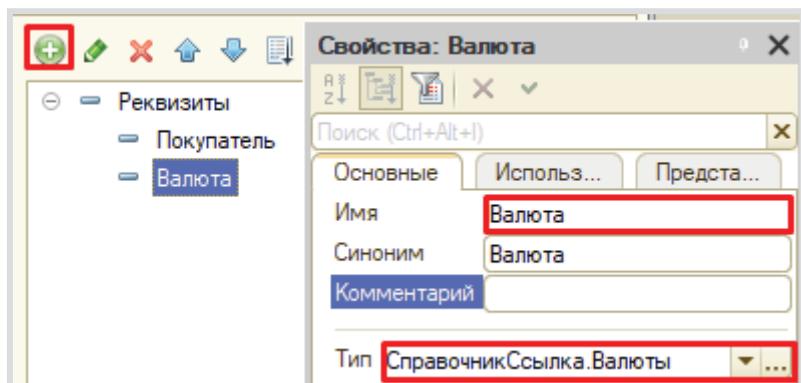
«При многовалютном учете пользователь системы при оформлении продажи должен обязательно указать валюту. Итоговая стоимость заказа должна формироваться автоматически».

Из условия следует, что при продаже обязательно нужно указывать покупателя и валюту. Также будем хранить и итоговую стоимость.

Добавим реквизит «Покупатель», тип – «СправочникСсылка.Контрагенты».

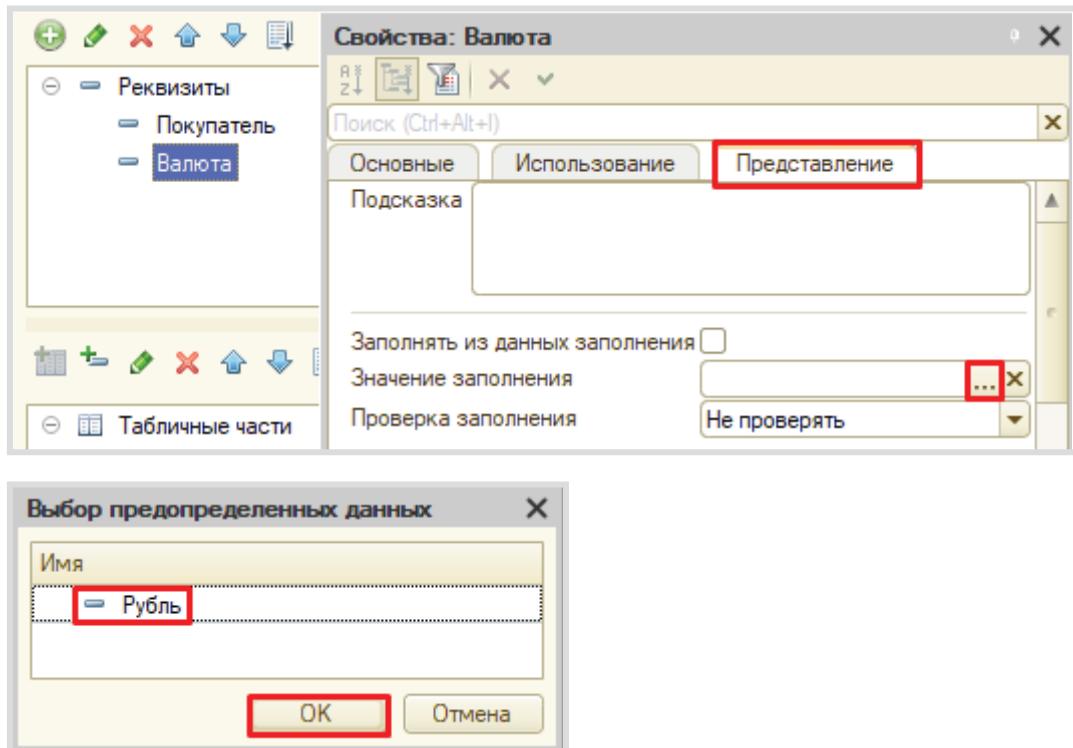


Далее добавим реквизит «Валюта».



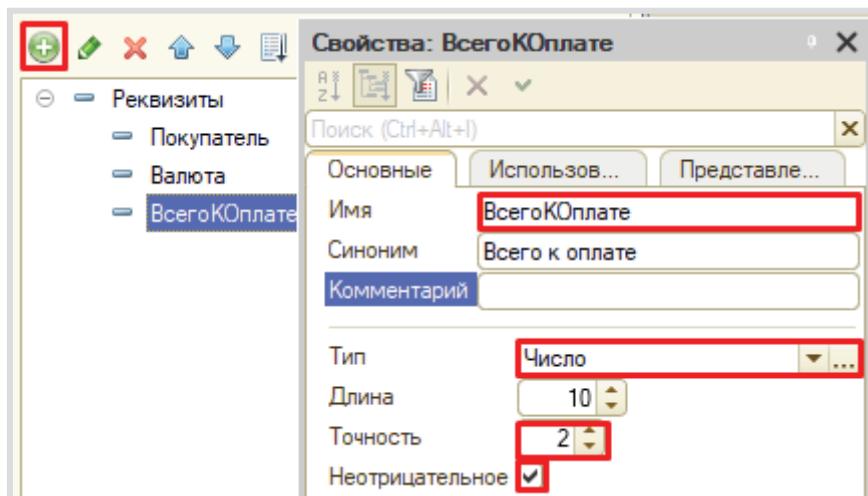
Пользователь может не использовать возможность ведения учета в различных валютах. В этом случае поле «Валюта» у него будет отсутствовать (этот функционал мы реализуем далее), но системе необходимо понимать, к чему привязана итоговая стоимость: к рублям или, например, долларам.

Поэтому установим значения заполнения по умолчанию на вкладке «Представление».



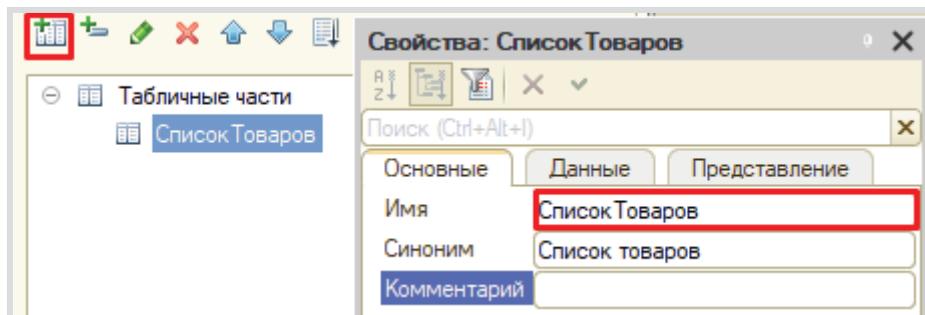
Таким образом, у пользователя по умолчанию будет указана валюта «Рубль», причем пользователь об этом может даже и не знать.

Последним реквизитом в шапку документа добавим «ВсегоКОплате».

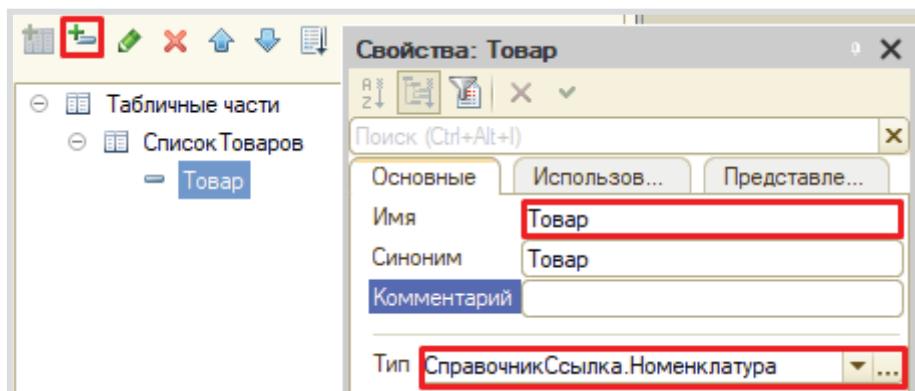


Чтобы регистрировать продажу товаров и услуг в одном документе, необходимо добавить две табличные части.

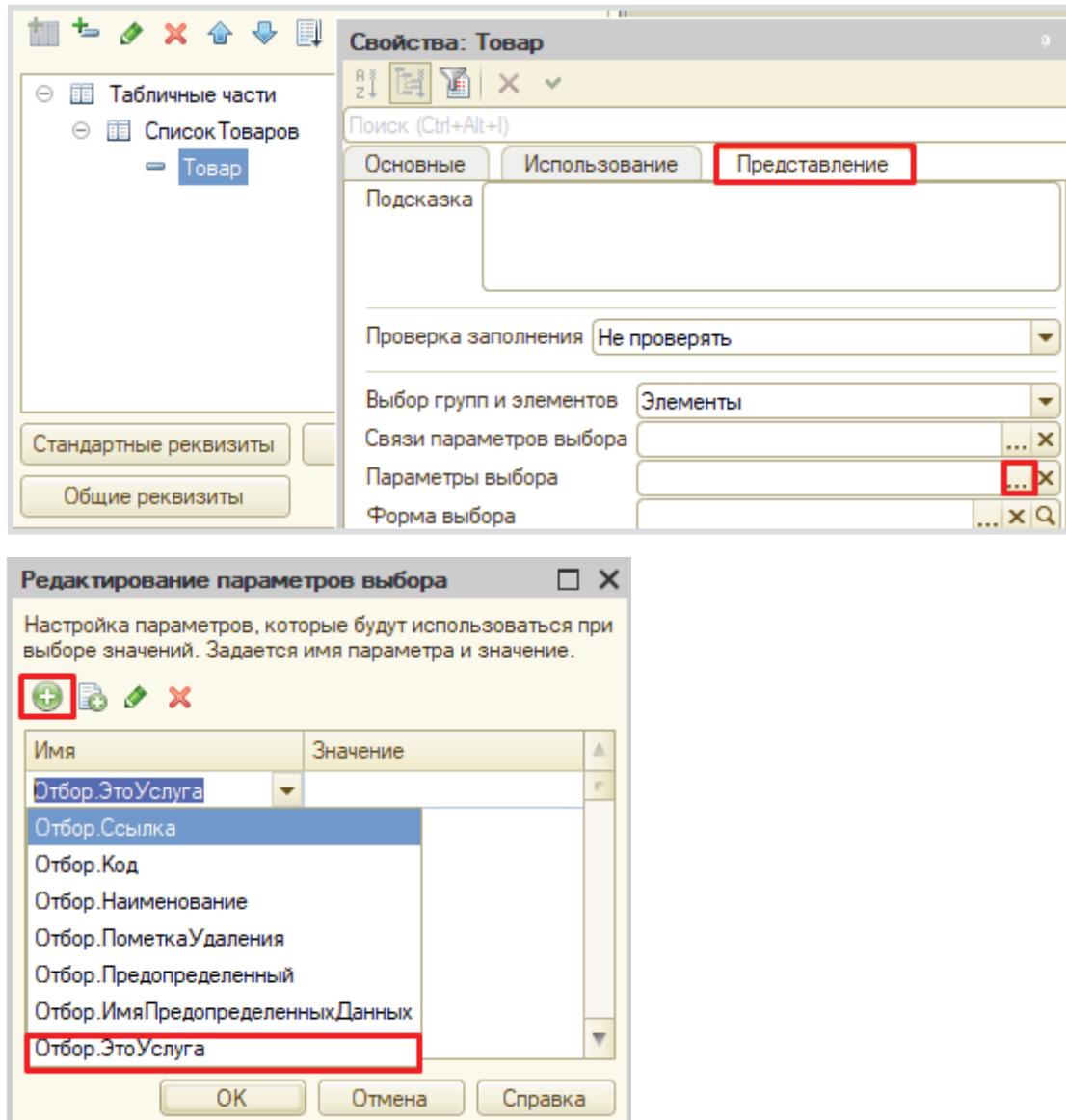
Сначала добавим табличную часть «СписокТоваров».



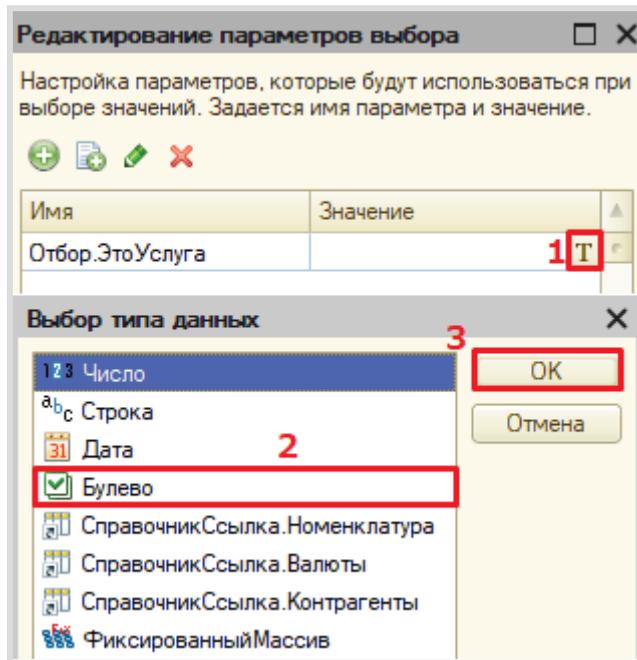
Добавим реквизит табличной части (колонку) «Товар».



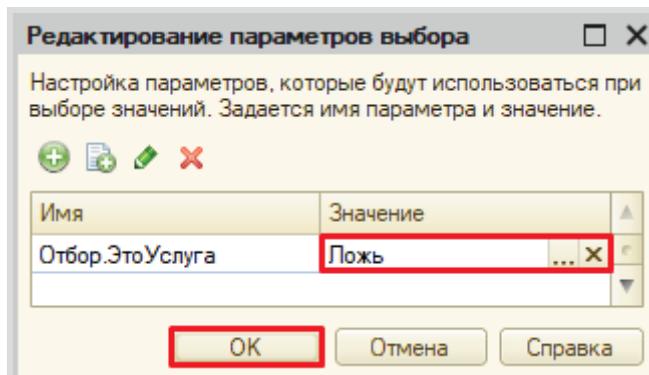
Поскольку мы ссылаемся на общий справочник номенклатура, то нам нужно на вкладке «Представление» настроить отбор только по тем элементам, у которых значение реквизита «ЭтоУслуга» установлено в положение «Ложь». Таким образом, пользователь будет в списке выбора видеть только товары.



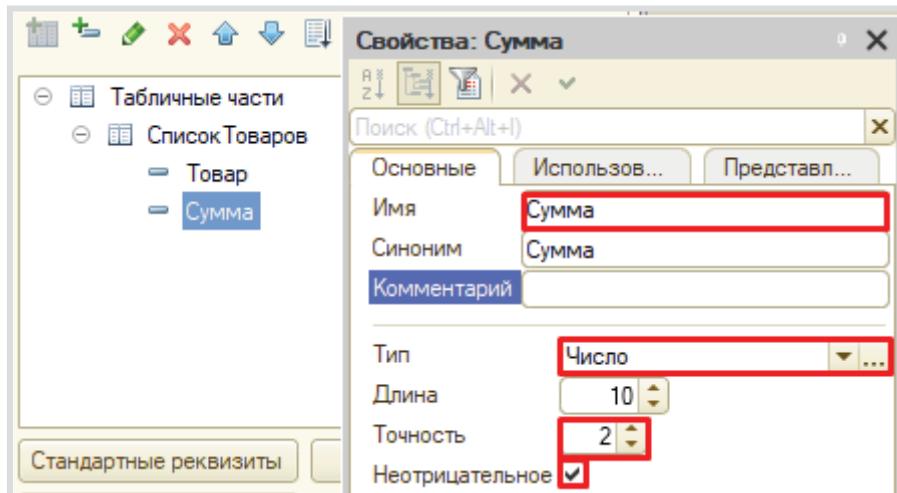
Поскольку в справочнике «Номенклатура» мы указывали для этого реквизита тип «Булево», то в значении параметра нужно указать именно его.



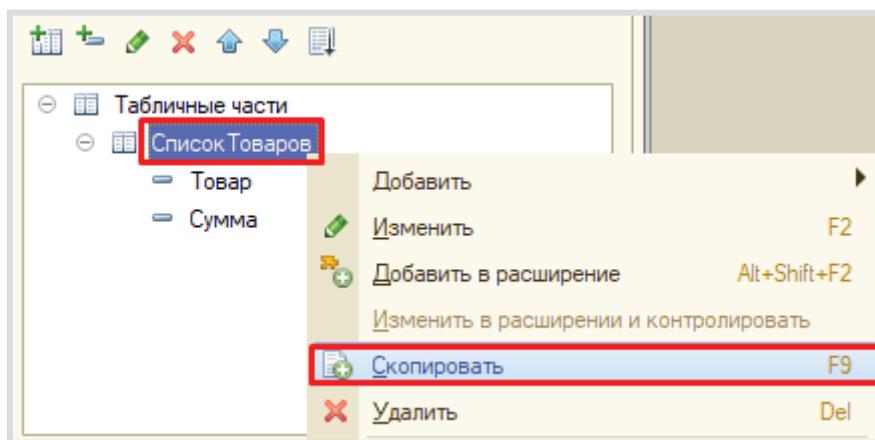
Для товаров значение «ЭтоУслуга» будет установлено в положении «Ложь».



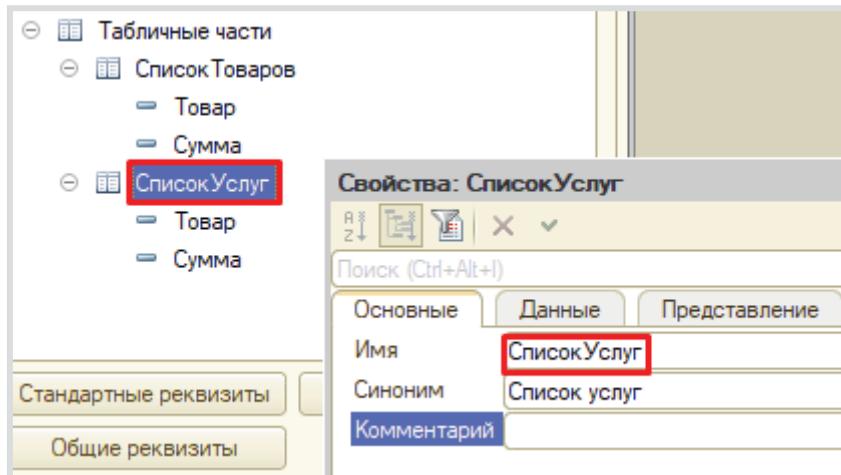
Далее добавим реквизит табличной части «Сумма».



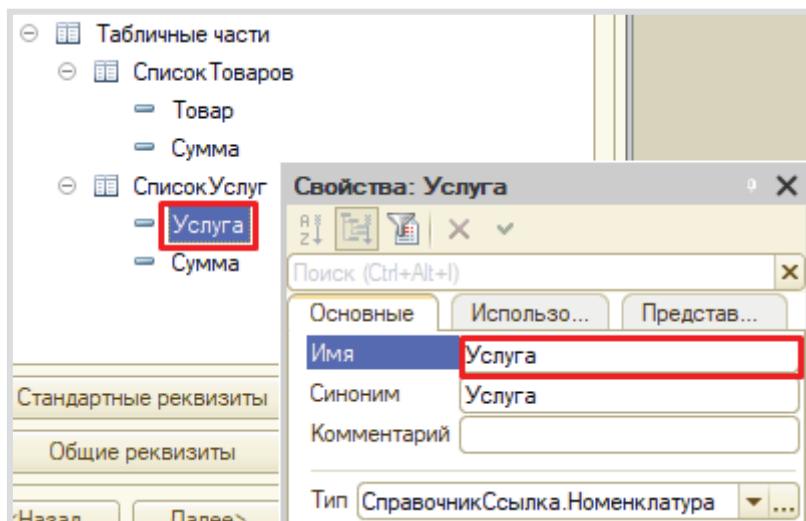
Табличная часть «СписокУслуг» структурно будет совпадать с уже созданной табличной частью «СписокТоваров». Для увеличения скорости разработки скопируем существующую табличную часть и несколько изменим ее.



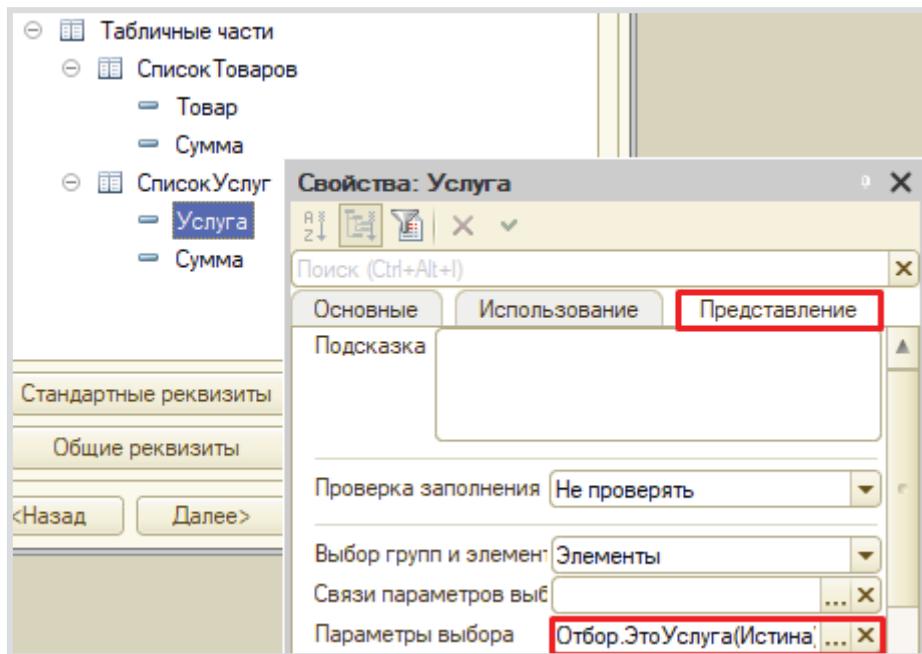
Во-первых, изменим свойство «Имя» у скопированной табличной части. Назовите табличную часть «СписокУслуг».



Во-вторых, изменим свойство «Имя» у реквизита табличной части «Товар». Поскольку данная табличная часть будет хранить перечень услуг, то и реквизиту нужно дать имя «Услуга».

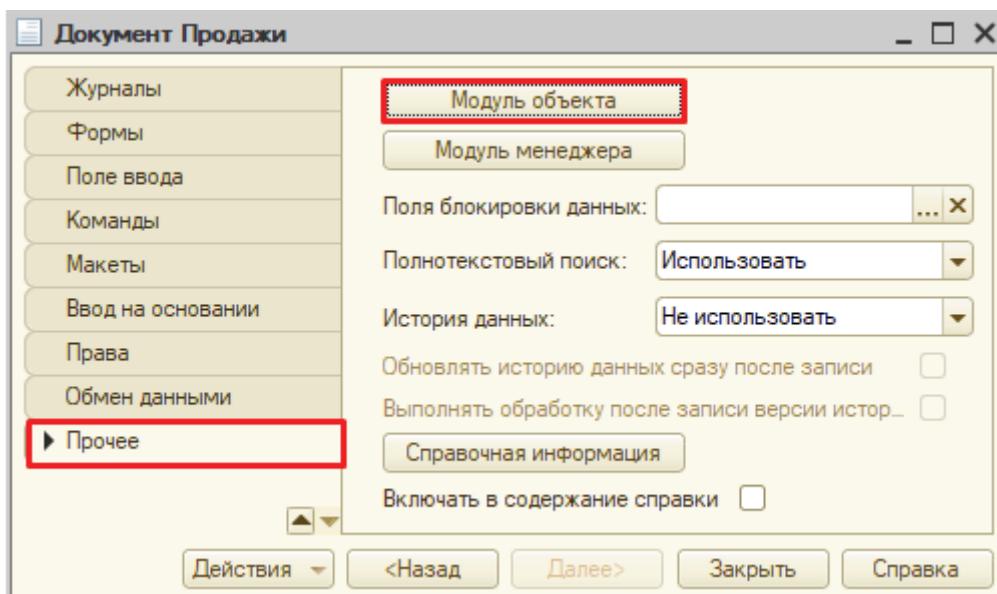


В-третьих, на вкладке «Представление» изменим параметры выбора: для услуг значение типа «Булево» будет в положении «Истина».

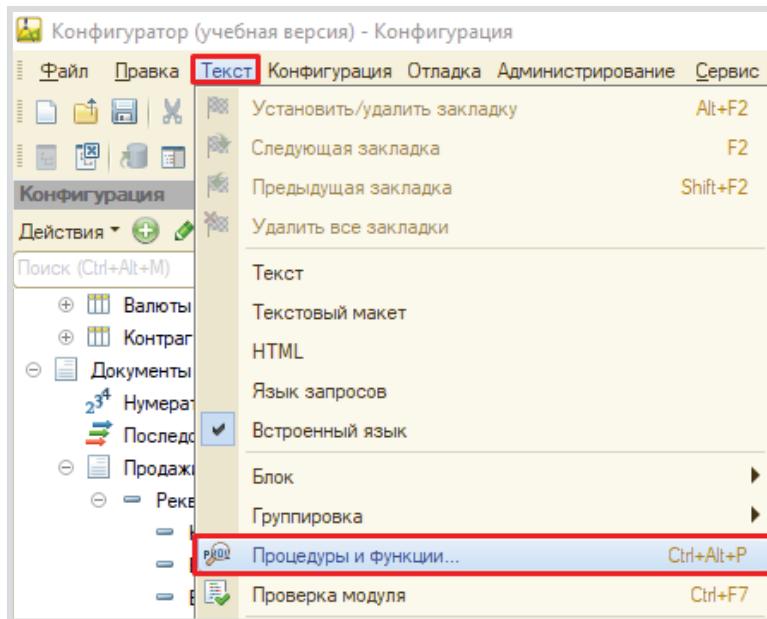


Чтобы итоговая стоимость формировалась автоматически, нужно описать событие в модуле объекта документа «Продажи».

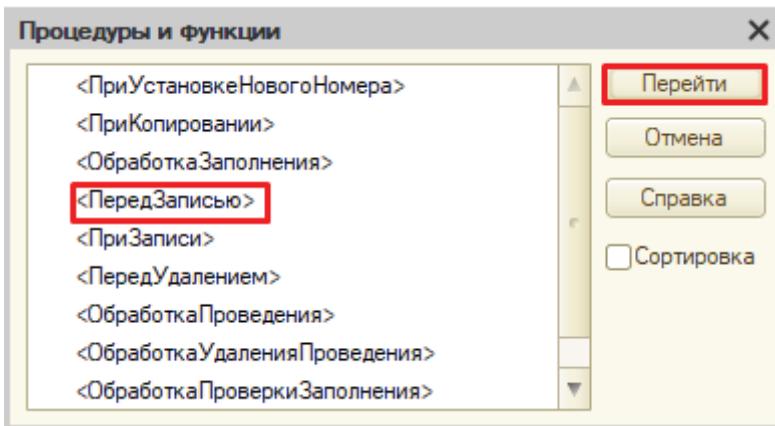
Для этого перейдем на вкладку «Прочее» и откроем «Модуль объекта».



Перед нами откроется модуль *объекта*, в котором нужно определить событие, при наступлении которого будет происходить описываемый алгоритм. Нам понадобится системное событие, вызовем его через главное меню «Текст» → «Процедуры и функции».



В открывшемся окне выберем «ПередЗаписью».



В модуле объекта сформируется обработчик события.

Далее нам нужно описать подсчет итоговой суммы как итог по товарам и услугам.

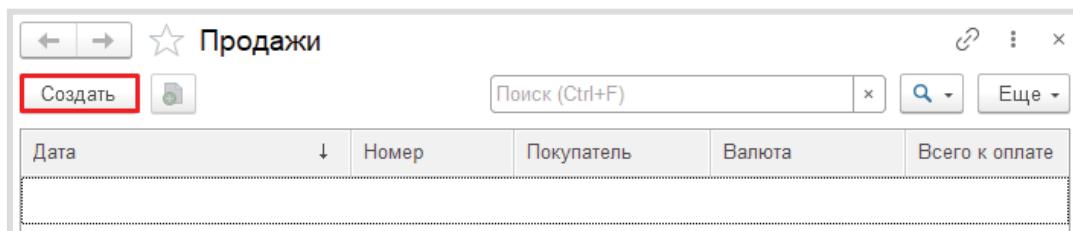
```

Процедура ПередЗаписью(Отказ, РежимЗаписи, РежимПроведения)
    ВсегоКОплате = СписокТоваров.Итог("Сумма") +
        СписокУслуг.Итог("Сумма");
КонецПроцедуры

```

ВсегоКОплате = СписокТоваров.Итог("Сумма") + СписокУслуг.Итог("Сумма");

Запустим режим «1С:Предприятие» и создадим несколько документов.



Заполним поля шапки документа, проигнорировав поле «Всего к оплате» (оно заполнится автоматически при записи).

Обратим внимание, что поле «Валюта» заполнено по умолчанию.

Затем заполним таблицы «Список товаров» и «Список услуг».

Список товаров		Список услуг	
<input type="button" value="Добавить"/> <input type="button" value="↑"/> <input type="button" value="↓"/>		Поиск (Ctrl+F) <input type="text"/> <input type="button" value="x"/> Еще <input type="button" value="▼"/>	
N	Товар	Сумма	
1	Компьютер	55 000,00	

Список товаров		Список услуг	
<input type="button" value="Добавить"/> <input type="button" value="↑"/> <input type="button" value="↓"/>		Поиск (Ctrl+F) <input type="text"/> <input type="button" value="x"/> Еще <input type="button" value="▼"/>	
N	Услуга	Сумма	
1	Доставка	3 000,00	

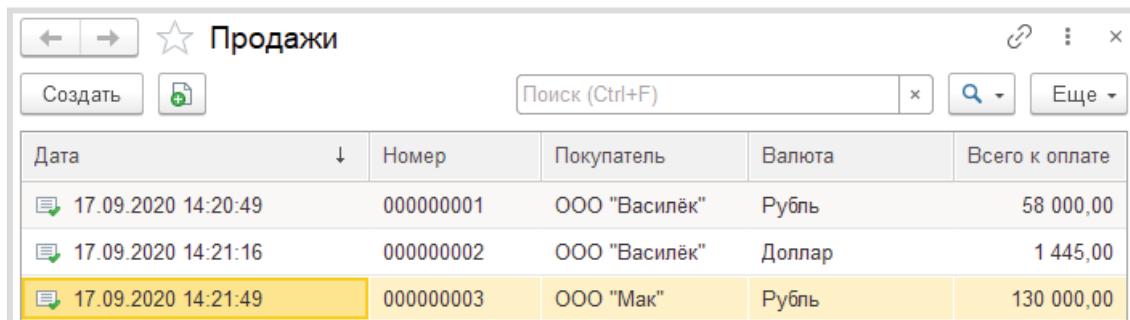
Внимание!

В список товаров можно добавлять только товары, а в список услуг – только услуги.

В момент записи или проведения документа поле «Всего к оплате» заполнится автоматически.

← → ⭐ Продажи 000000001 от 17.09.2020 14:19:39		🔗 ⋮ ×	
<input type="button" value="Провести и закрыть"/> <input type="button" value="Записать"/> <input style="border: 2px solid red; background-color: white; color: red; padding: 2px 10px; margin-left: 10px;" type="button" value="Провести"/>		Еще <input type="button" value="▼"/>	
Номер:	<input type="text" value="000000001"/>	Дата:	<input type="text" value="17.09.2020 14:19:39"/> <input type="button" value="CALENDAR"/>
Покупатель:	<input type="text" value="ООО " василёк""=""/> <input type="button" value="▼"/> <input type="button" value="却是"/>		
Валюта:	<input type="text" value="Рубль"/> <input type="button" value="▼"/> <input type="button" value="却是"/>		
Всего к оплате:	<input type="text" value="58 000,00"/> <input type="button" value="CALENDAR"/>		
<input type="button" value="Список товаров"/> <input type="button" value="Список услуг"/>			

Создайте еще несколько документов на разных покупателей и разные валюты.

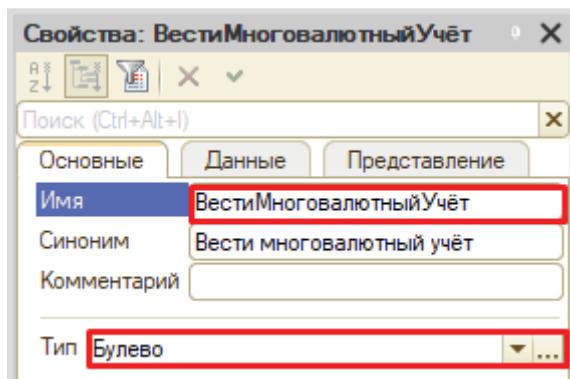


Дата	Номер	Покупатель	Валюта	Всего к оплате
17.09.2020 14:20:49	000000001	ООО "Василёк"	Рубль	58 000,00
17.09.2020 14:21:16	000000002	ООО "Василёк"	Доллар	1 445,00
17.09.2020 14:21:49	000000003	ООО "Мак"	Рубль	130 000,00

Сейчас мы по умолчанию установили валюту «Рубль», оставив пользователю возможность выбора другой валюты. Но возможность использовать разные валюты нам нужно сделать опциональной. Для этого воспользуемся общим механизмом *функциональные опции*. Более подробно про функциональные опции можно прочитать здесь: <https://v8.1c.ru/platforma/funktionalnaya-opciya/>.

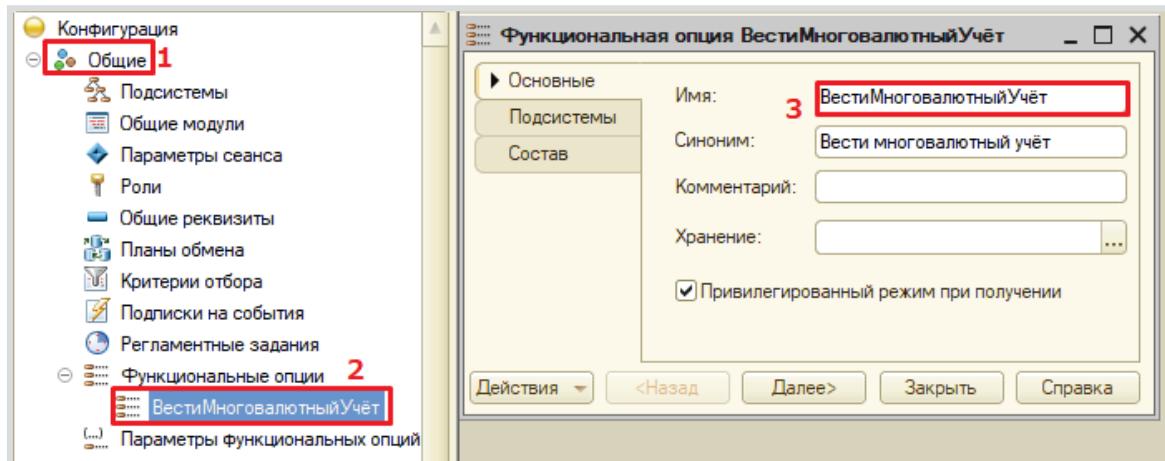
У пользователя должен быть физический способ включения и отключения многовалютного учета. Реализуем это с помощью обычной константы с типом «Булево».

Добавим константу «ВестиМноговалютныйУчёт».



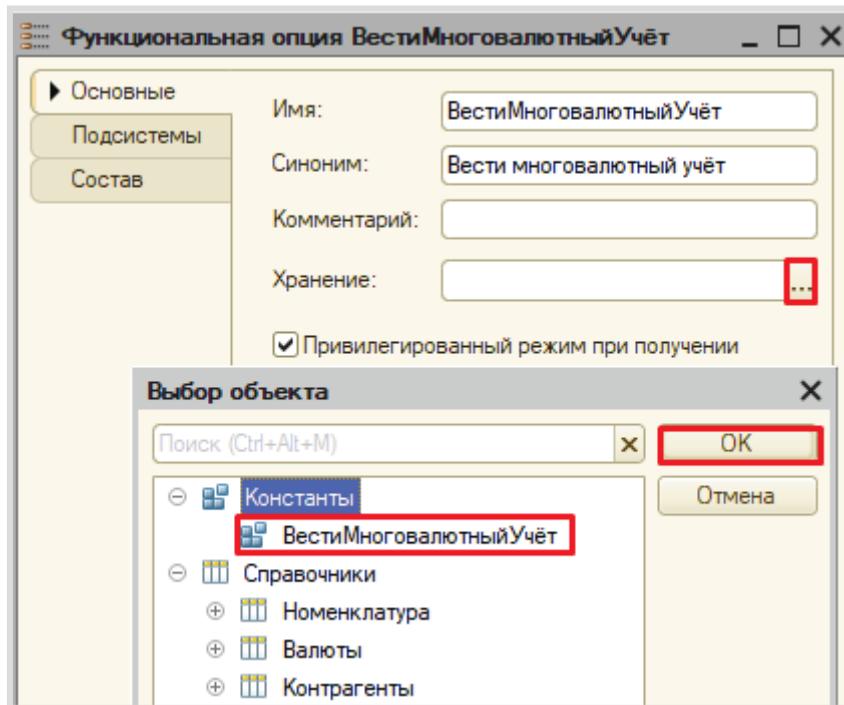
Теперь приступим к созданию и настройке функциональной опции.

В ветке «Общие» дерева конфигурации найдем раздел «Функциональные опции» и добавим опцию с именем «ВестиМноговалютныйУчет».



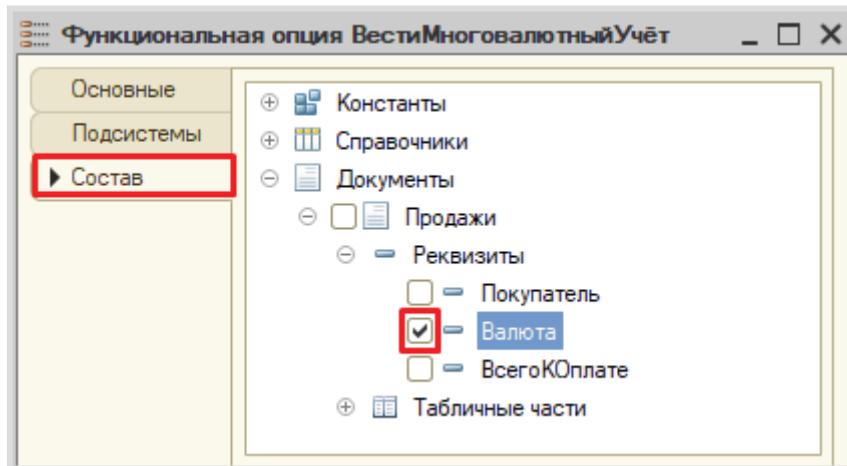
Теперь перейдем к ее настройке.

Первым делом нужно определить, где она будет храниться. Укажем в качестве места хранения заранее заготовленную константу с аналогичным именем.



Затем нужно выбрать, на что будет влиять наша функциональная опция. Она должна «включать» и «выключать» поле «Валюта» в документе «Продажи».

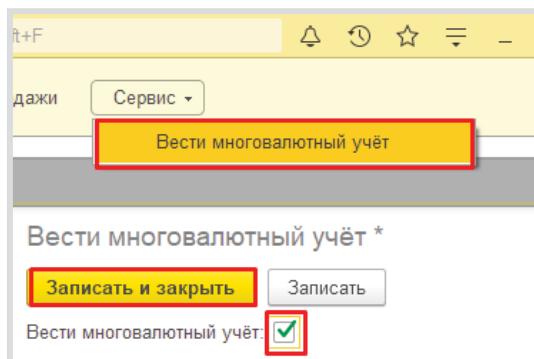
Укажем необходимый реквизит документа на вкладке «Состав».



Запустим режим «1С:Предприятие» и проверим работоспособность.

Продажи				
Дата	Номер	Покупатель	Всего к оплате	
17.09.2020 14:20:49	000000001	ООО "Василёк"	58 000,00	
17.09.2020 14:21:16	000000002	ООО "Василёк"	1 445,00	
17.09.2020 14:21:49	000000003	ООО "Мак"	130 000,00	

Открыв список документов, мы обнаружим, что отсутствует колонка «Валюта». Это связано с тем, что значение типа «Булево» по умолчанию – «Ложь». Чтобы вновь иметь возможность выбирать валюту, необходимо у константы перевести значение в положение «Истина» и перезапустить режим «1С:Предприятие».



При повторном открытии списка всех документов после проделанных действий мы обнаружим, что колонка «Валюта» снова отображается.

Дата	Номер	Покупатель	Валюта	Всего к оплате
17.09.2020 14:20:49	000000001	ООО "Василёк"	Рубль	58 000,00
17.09.2020 14:21:16	000000002	ООО "Василёк"	Доллар	1 445,00
17.09.2020 14:21:49	000000003	ООО "Мак"	Рубль	130 000,00

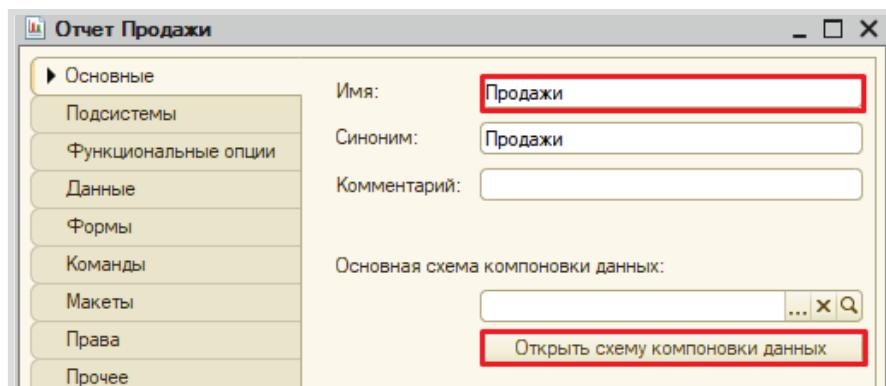
«Следует построить Отчет по продажам с возможностью выбора нужной валюты».

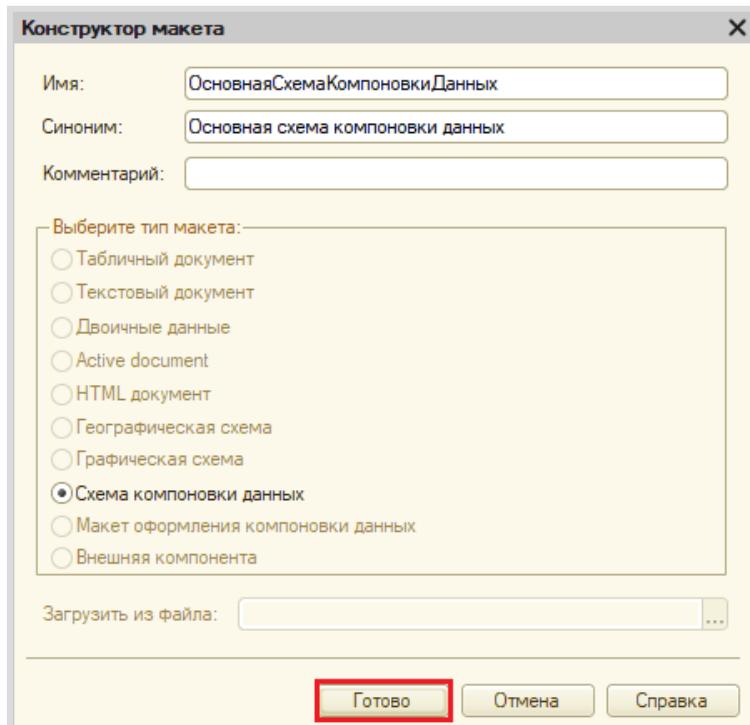
Построим отчет. Для этого воспользуемся соответствующим объектом конфигурации.

Определение

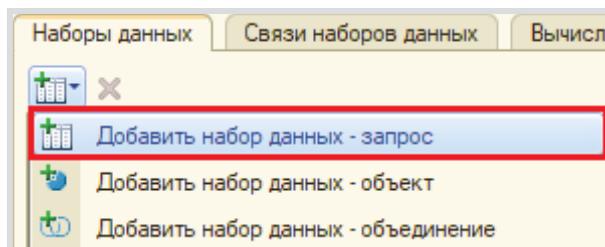
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Добавим отчет «Продажи». Для наполнения отчета воспользуемся *схемой компоновки данных*.

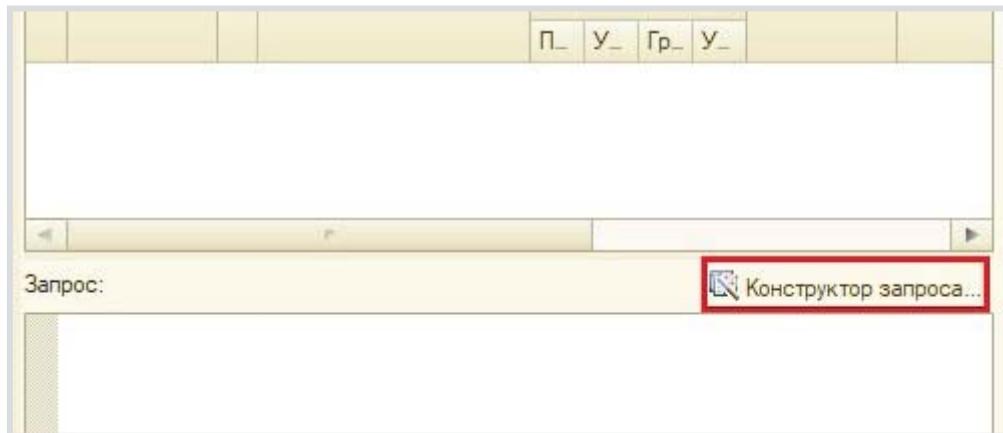




Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Для формирования запроса воспользуемся конструктором запроса.



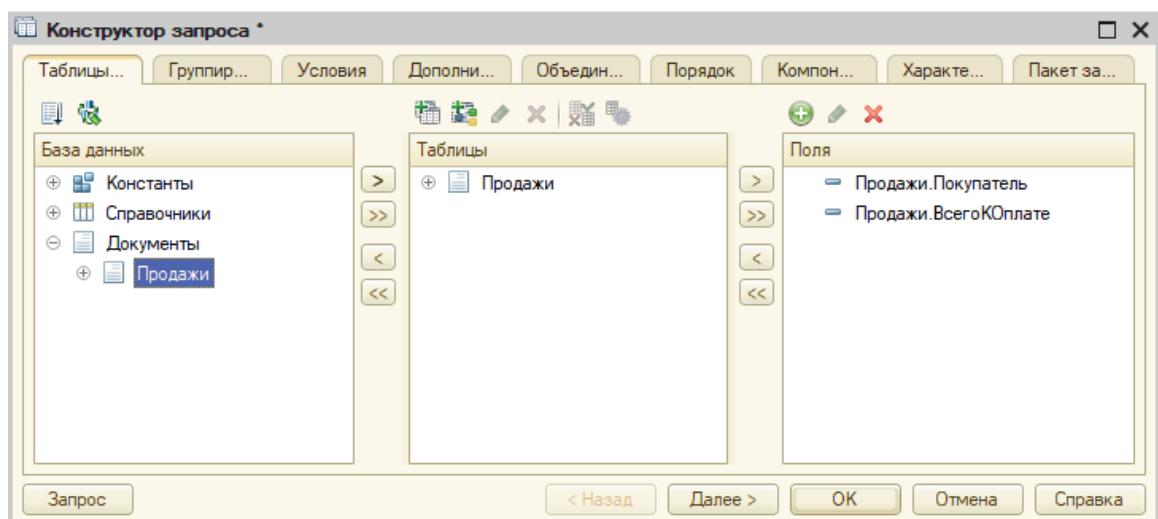
Открывается конструктор запроса. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать из *регистра накоплений* напрямую, чтобы иметь возможность рассчитывать средний балл.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

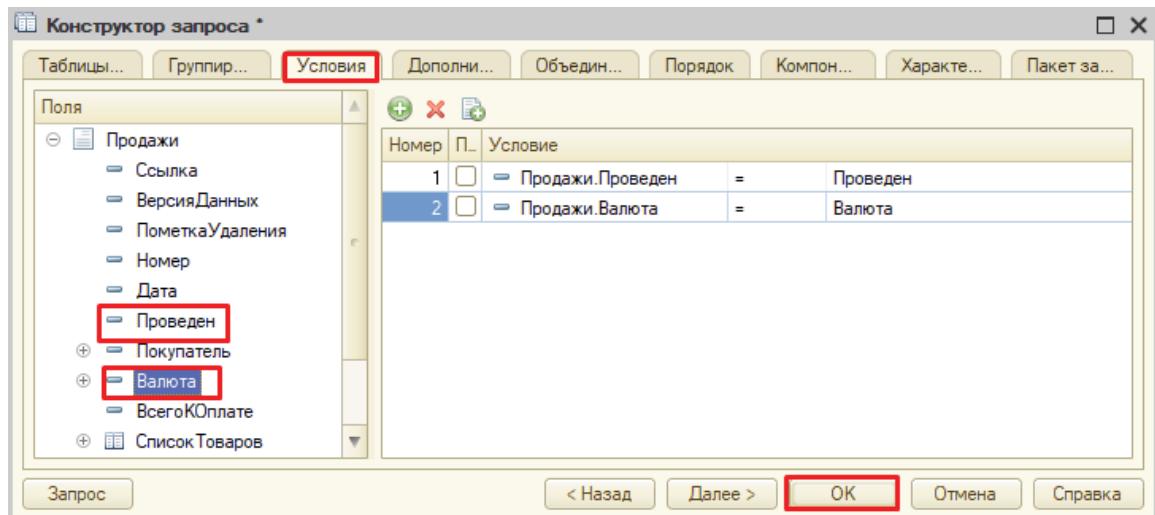
В результате данное окно должно быть заполнено следующим образом:



Перейдем на вкладку «Условия».

Наложим условие на реквизит «Проведен», чтобы в отчет попали только совершенные (проведенные) посещения экскурсий.

Наложим условие на поле «Валюта». Это нужно для того, чтобы дать пользователю возможность выбрать интересующую его валюту и построить на ее основании график.



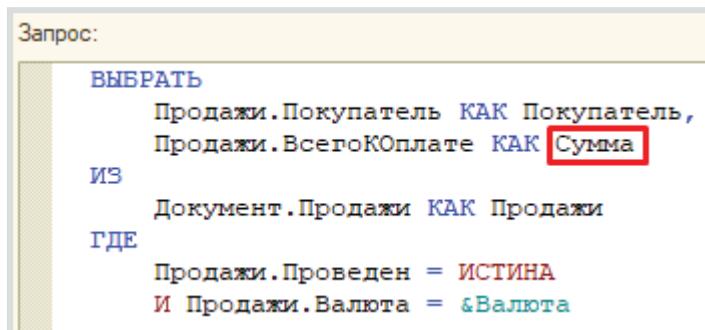
У нас сформируется запрос на встроенном языке запросов 1С. Поскольку пометка проведения имеет только два значения («Истина» и «Ложь»), то нам нужно скорректировать текст нашего запроса.

```

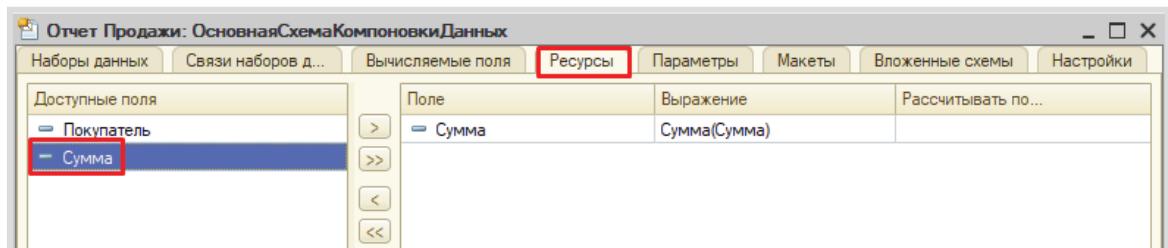
Запрос:
ВЫБРАТЬ
    Продажи.Покупатель КАК Покупатель,
    Продажи.ВсегоКОплате КАК ВсегоКОплате
ИЗ
    Документ.Продажи КАК Продажи
ГДЕ
    Продажи.Проведен = ИСТИНА
    И Продажи.Валюта = &Валюта

```

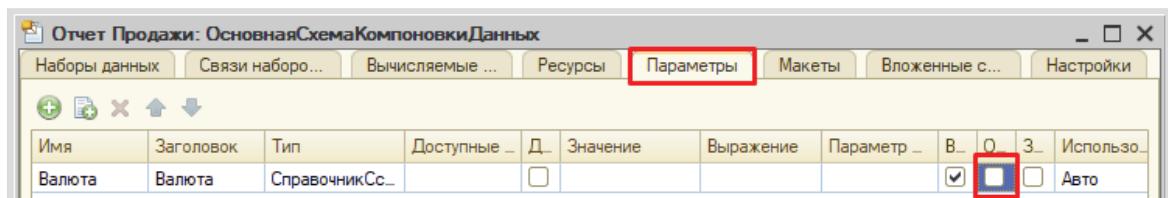
Для красоты зададим синоним полю «ВсегоКОплате». Сделать это можно с помощью конструктора запроса либо прямо в окне запроса.



Чтобы в отчете происходил итоговый подсчет по всем продажам, сделаем поле «Сумма» (синоним поля «ВсегоКОплате») ресурсом на соответствующей вкладке.

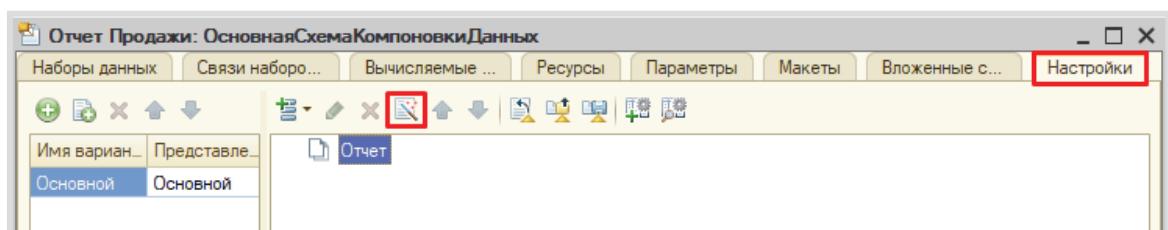


Далее нужно включить пользователю возможность выбирать нужную валюту. Для этого перейдем на вкладку «Параметры» и отключим ограничение доступности.

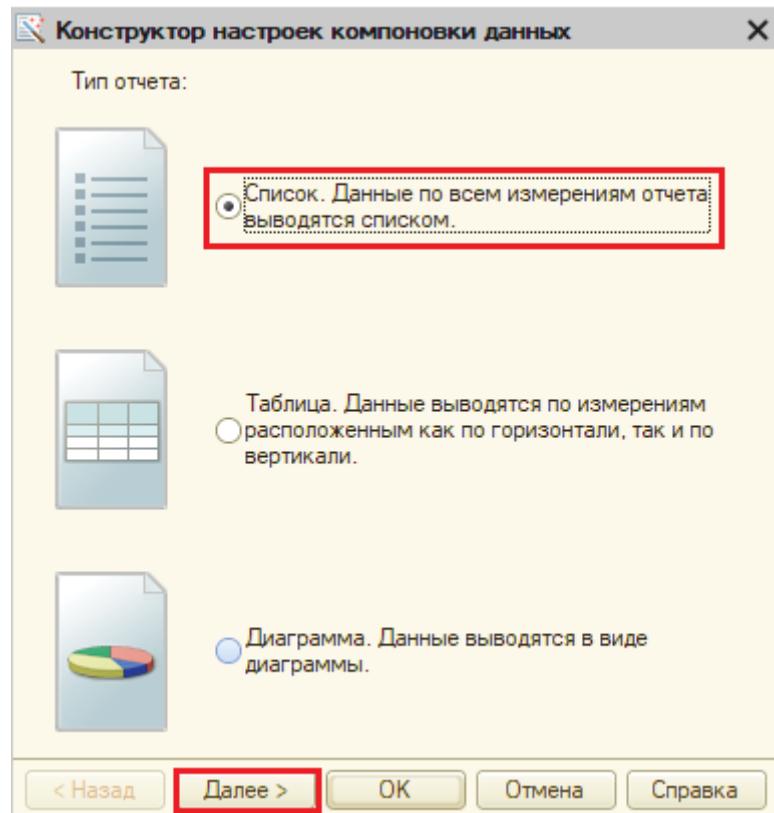


Теперь переходим на вкладку «Настройки» для оформления внешнего вида отчета.

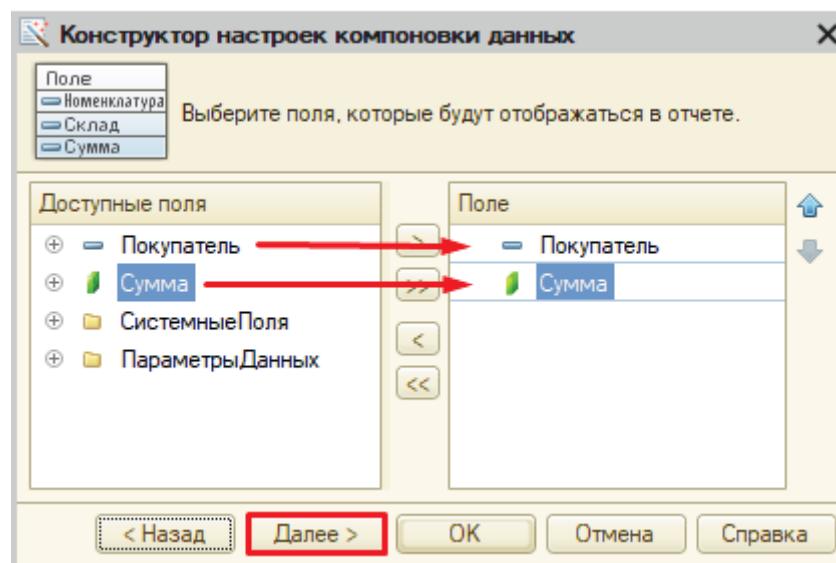
Воспользуемся конструктором настроек отчета.



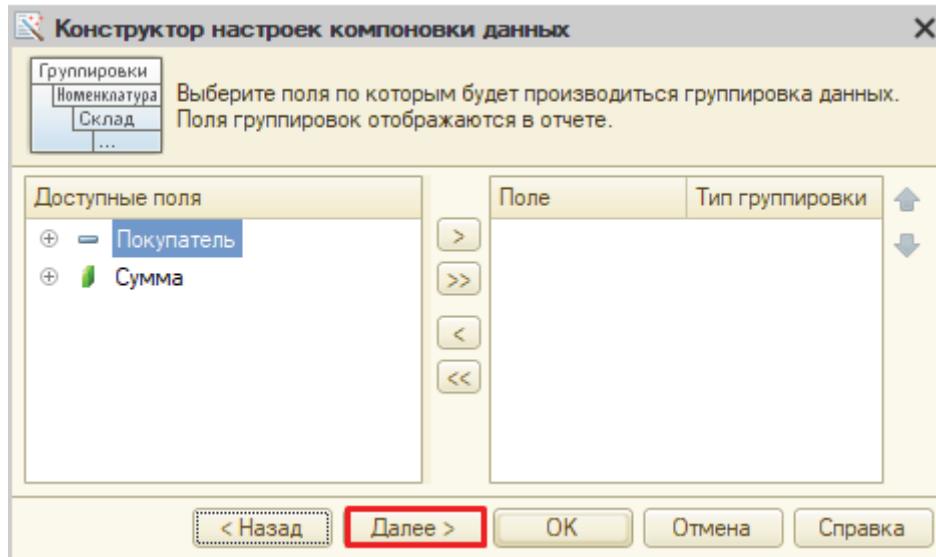
Построим отчет в виде списка.



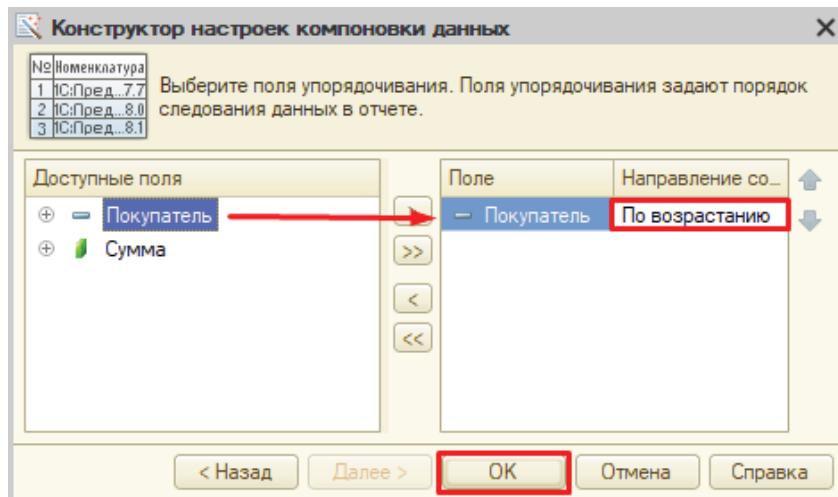
Выберем поля, которые будут отображаться в отчете.



Этап группировки мы пропустим.

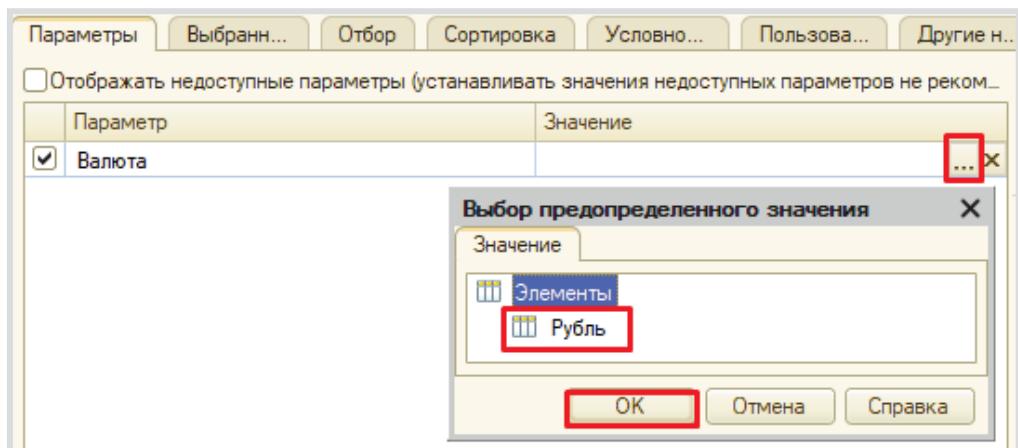
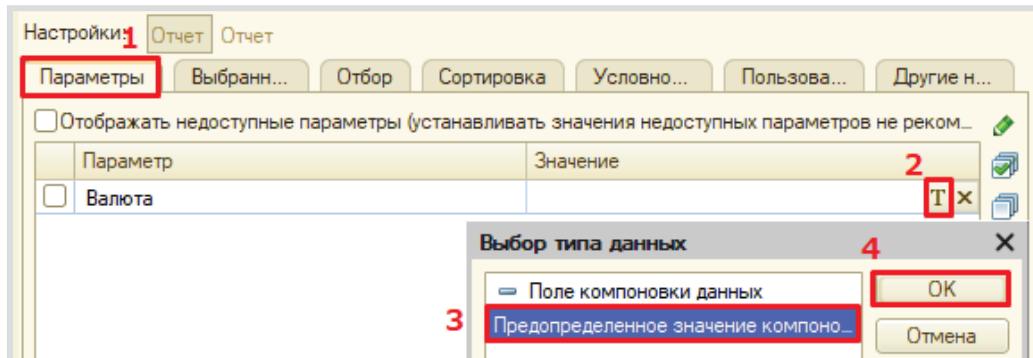


Отсортируем наш отчет по алфавитному порядку покупателей.

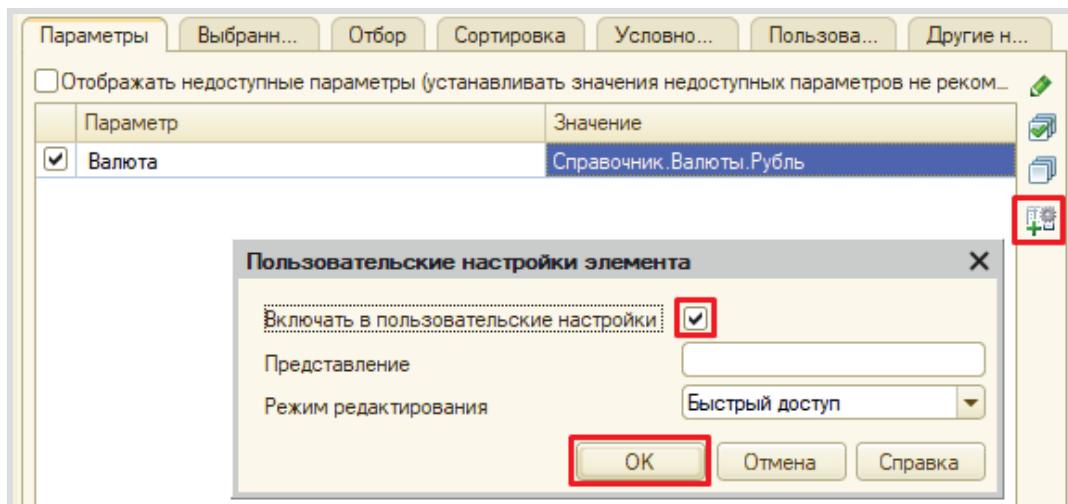


Заключительным этапом будет настройка параметра для отображения у пользователя.

Сначала установим значение по умолчанию.



Последнее, что осталось – включить параметр в пользовательские настройки.



Отчет готов.

Запустим систему в режиме «1С:Предприятие» и построим его, указав разные валюты.

Параметры: Валюта: Рубль	
Покупатель	Сумма
ООО "Василёк"	58 000,00
ООО "Мак"	130 000,00
Итого	188 000,00

Параметры: Валюта: Доллар	
Покупатель	Сумма
ООО "Василёк"	1 445,00
Итого	1 445,00

Поставленная задача решена.

Лабораторная работа № 15

РАЗРАБОТКА КОНФИГУРАЦИИ ДЛЯ УЧЕТА ДОХОДОВ ОТ ПРОДАЖ ТОВАРОВ

Сложность: **

Теги: справочник, документ, функциональная опция,
схема компоновки данных, регистр накопления,
регистр сведений

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета продаж товаров с сопутствующими услугами покупателям. Необходимо предусмотреть опциональную возможность использования различных валют.

При многовалютном учете пользователь системы при оформлении продажи должен обязательно указать валюту. Итоговая стоимость заказа должна формироваться автоматически.

В системе нужно реализовать хранилище суммы доходов в рублях по номенклатурным позициям.

Доходом считается сумма продажи в рублевом выражении.

Необходимо построить «Отчет по продажам» с упорядочиванием по сумме доходов.

Форма отчета:

Номенклатура	Сумма
Компьютер	239 000,00
Телефон	45 000,00
Доставка	34 000,00
Итого	318 000,00

Отчет выводит информацию по выбранной валюте, а также подводит общий итог.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать, смотрите в Лабораторной работе № 2 (стр. 17).

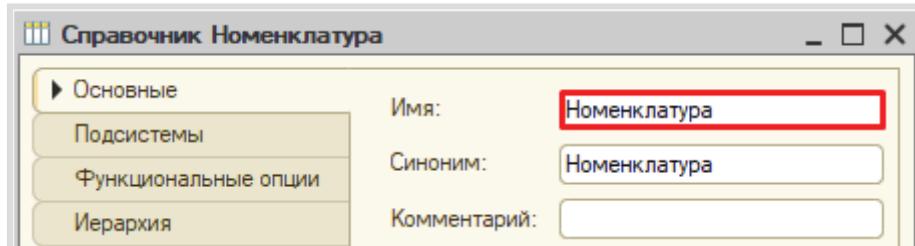
Выполнение

Из условия следует, что нужно хранить информацию о покупателях, товарах и услугах, а также валютах. Для решения этой задачи нам понадобятся *справочники*.

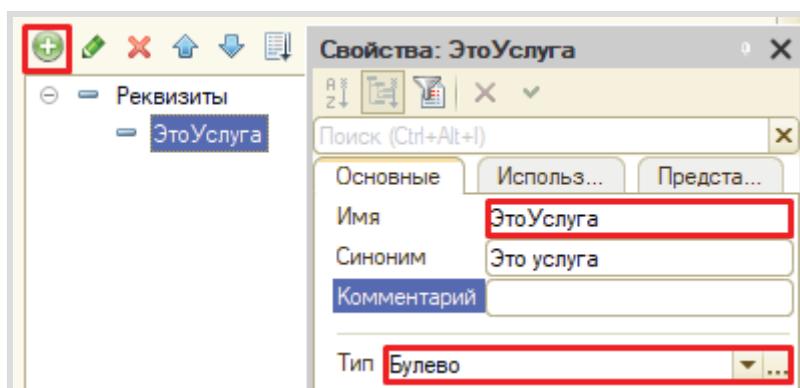
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

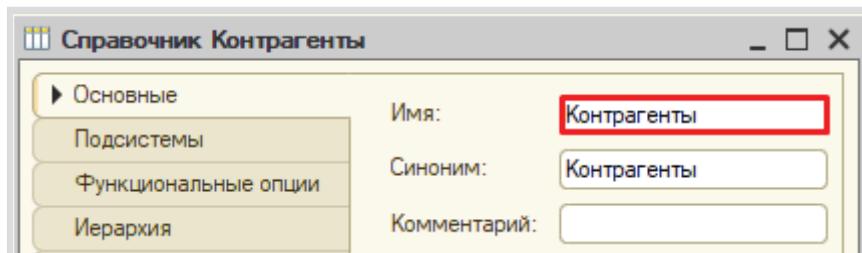
Создадим справочник «Номенклатура».



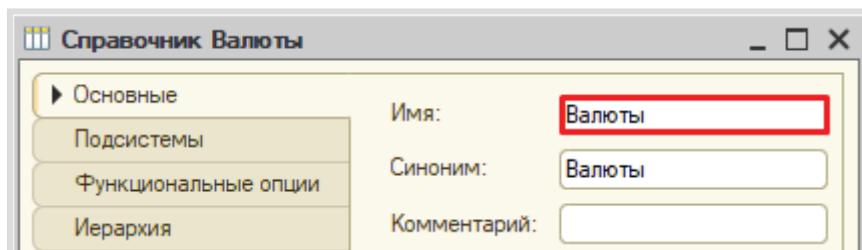
Перейдем на вкладку «Данные» и добавим реквизит «ЭтоУслуга», тип – «Булево». Данный реквизит необходим для того, чтобы отличать товары от услуг в справочнике.



Создадим справочник «Контрагенты».



Создадим справочник «Валюты».

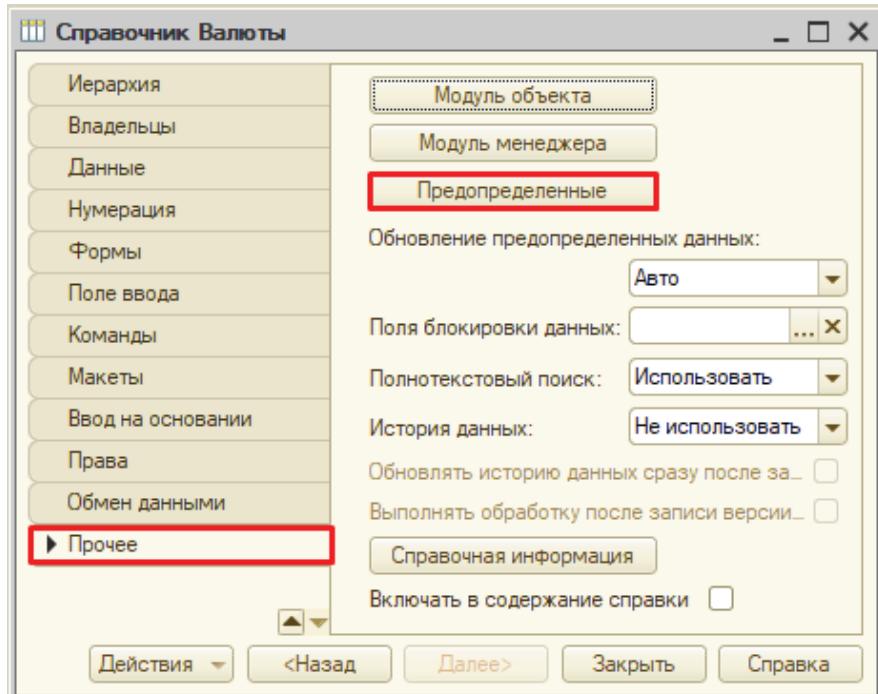


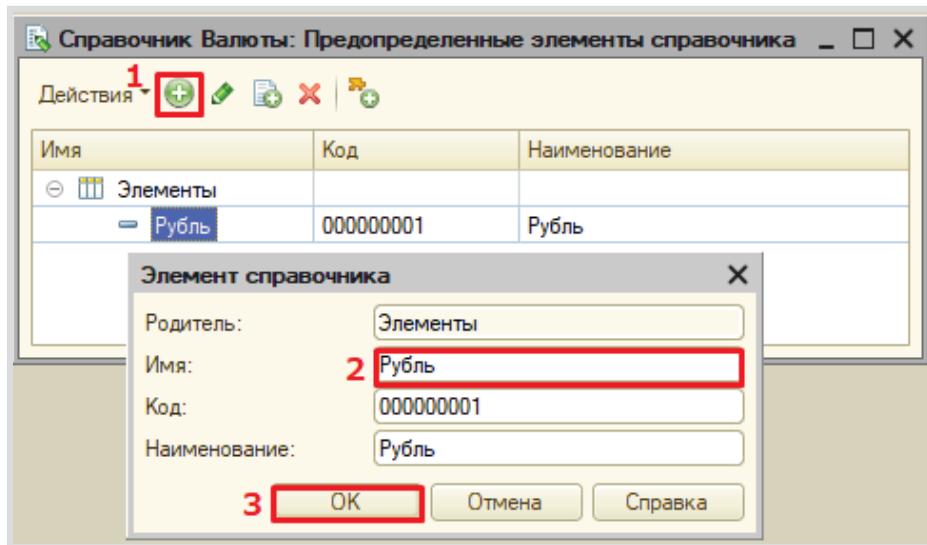
Обязательной валютой в данной информационной системе является рубль. Создадим предопределенный элемент.

Определение

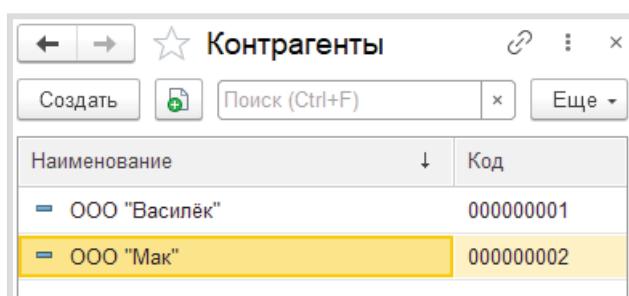
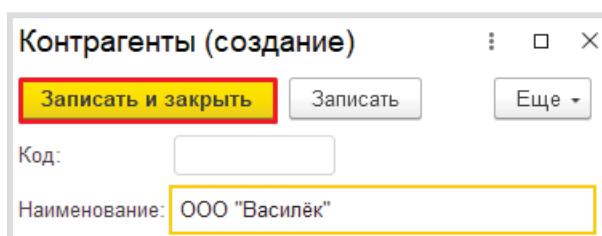
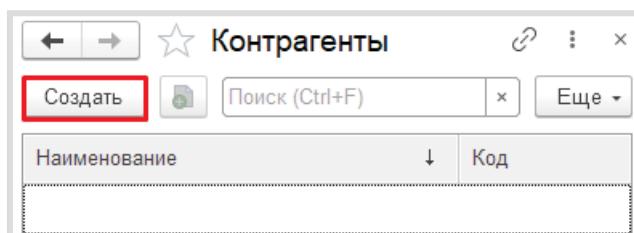
Предопределенные элементы – это такие элементы, которые создает разработчик в конфигураторе для удобства работы пользователя.

Созданный таким образом элемент будет доступен пользователю с первого запуска программы. Для создания такого элемента перейдем на вкладку «Прочее» и откроем список предопределенных элементов справочника.

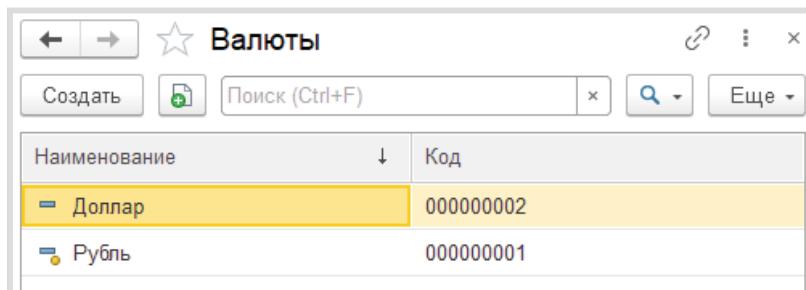
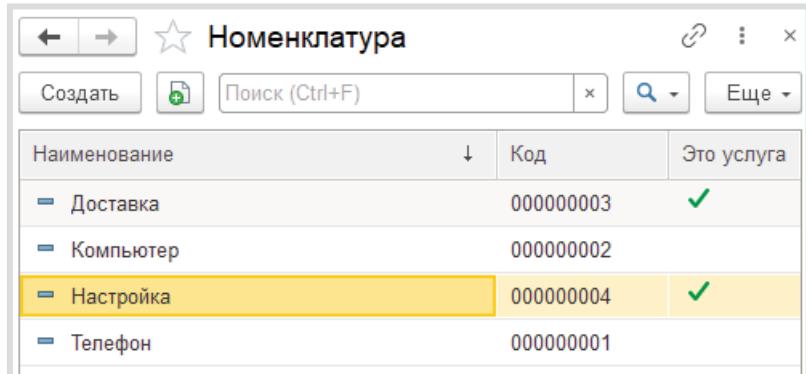




Откроем программу в режиме «1С:Предприятие» и добавим в каждый справочник несколько элементов.



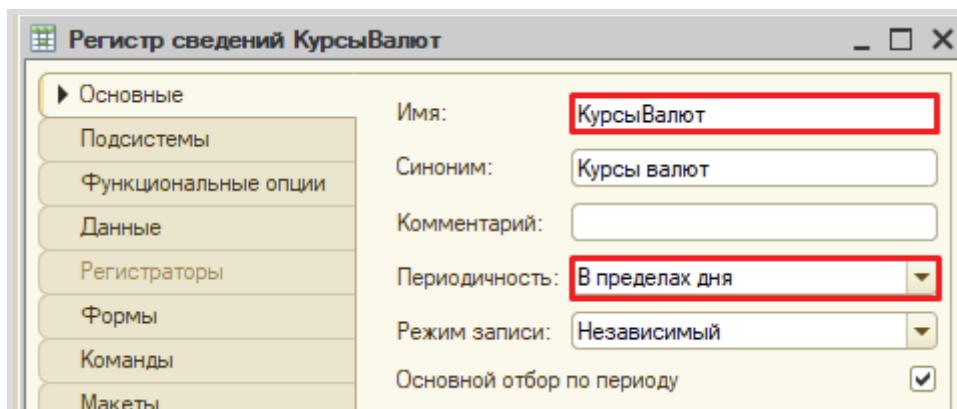
Аналогично добавьте элементы в справочники «Номенклатура» и «Валюты».



Заметим, что предопределенные элементы в справочнике отмечены желтым маркером.

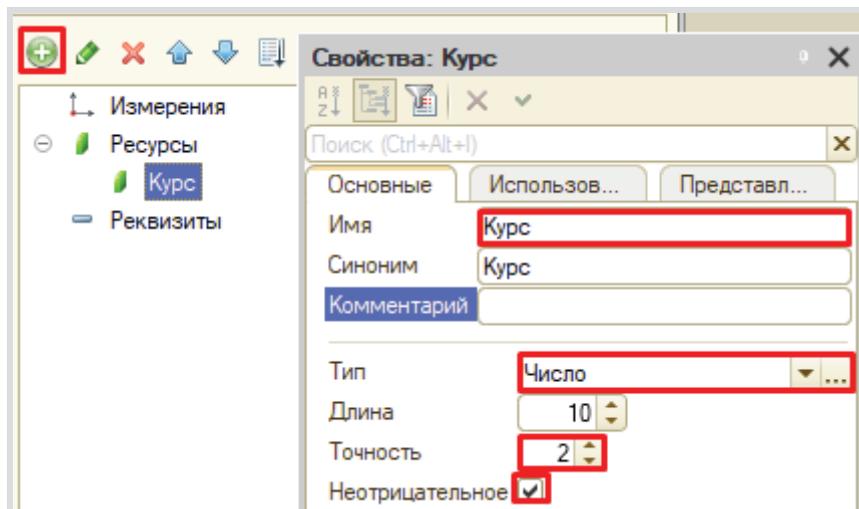
Чтобы хранить курсы валют, которые меняются со временем, нам понадобится специально настроенный справочник. Такой прикладной объект называется *регистр сведений* (подробнее про *регистры сведений* можно прочитать здесь: <https://v8.1c.ru/platforma/registr-svedeniy/>).

Создадим *регистр сведений* и назовем его «КурсыВалют». Поскольку курс валют может меняться раз в день, то установим периодичность в пределах дня.

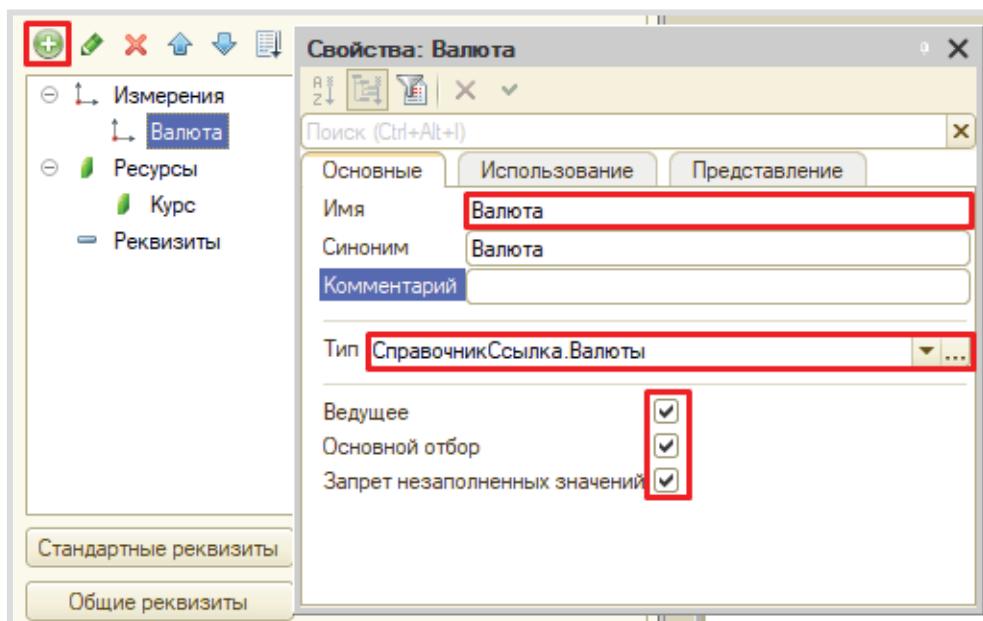


На вкладке «Данные» определим структуру нашего регистра.

Мы должны задаться вопросом: «Что мы хотим хранить?». Мы хотим хранить курс. То, что мы хотим хранить, – это и есть ресурс. Добавим ресурс «Курс».



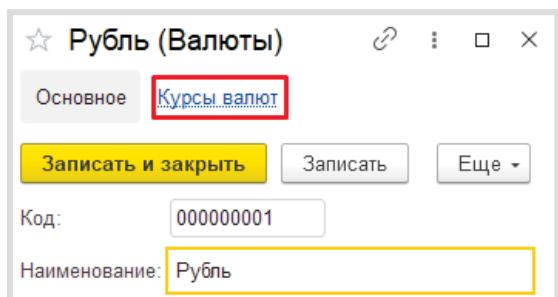
После того, как мы определили, что хотим хранить, нужно понять: в разрезе чего мы хотим это хранить? Курсы мы хотим хранить в разрезе валют. Значит, валюта – измерение.



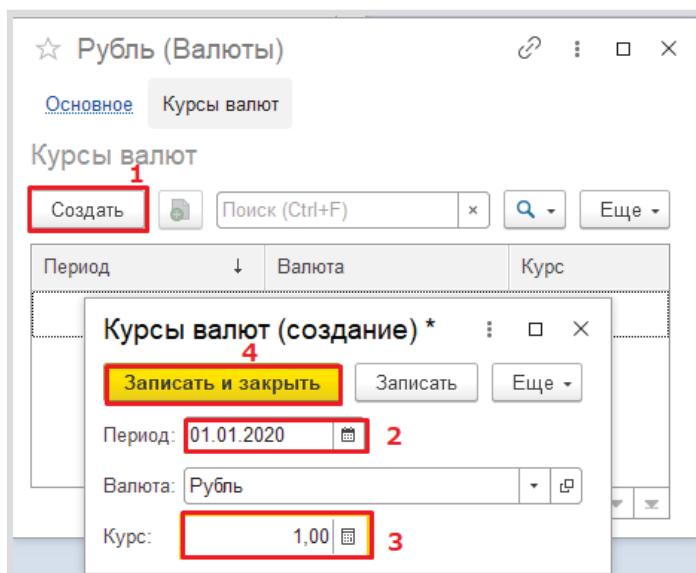
Свойство «Ведущее» позволит нам переходить к курсу валюты прямо из справочника соответствующего элемента.

Запустим режим «1С:Предприятие» и заполним курсы валют на разные даты.

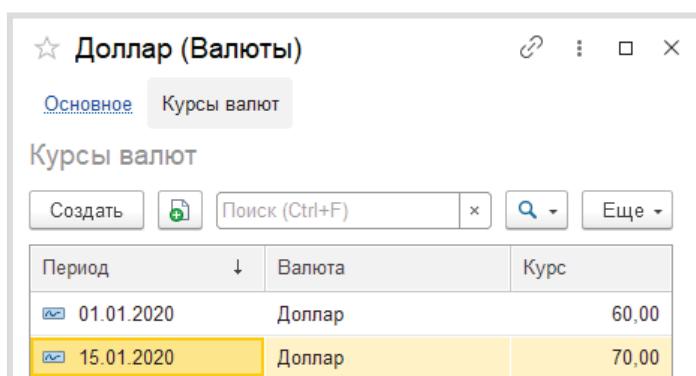
Откроем карточку элемента справочника «Валюты» и убедимся в работоспособности настройки свойства «Ведущее».



Обязательно укажем курс для рубля. Поскольку рубль к рублю идет в отношении 1:1, то укажем курс рубля – «1». Указывать будем на начало года.



Добавьте курсы на разные даты, в том числе и для других валют.

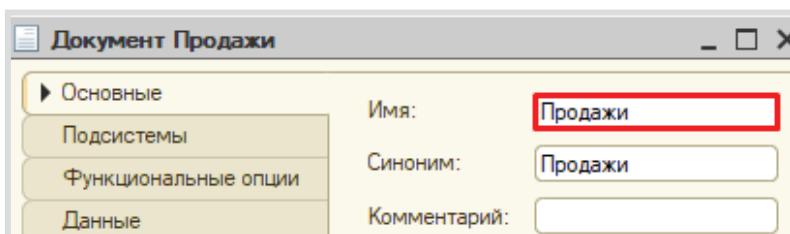


Для регистрации продаж следует воспользоваться объектом конфигурации *документ*.

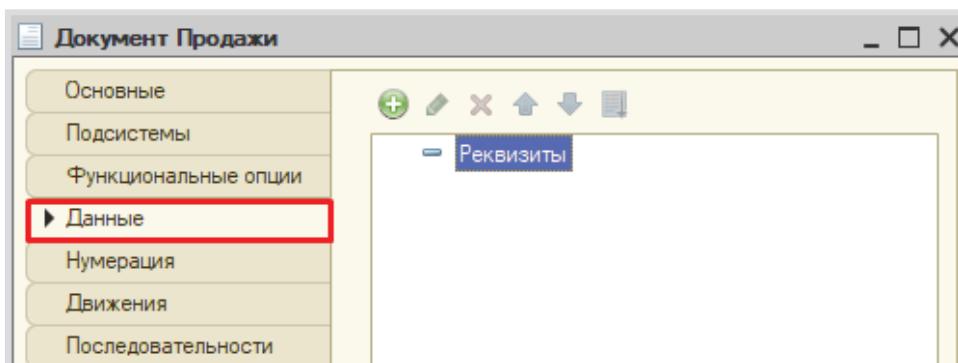
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим новый документ «Продажи».

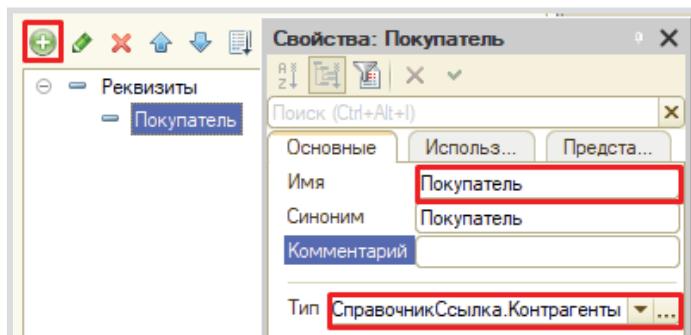


Для настройки структуры документа переходим на вкладку «Данные».

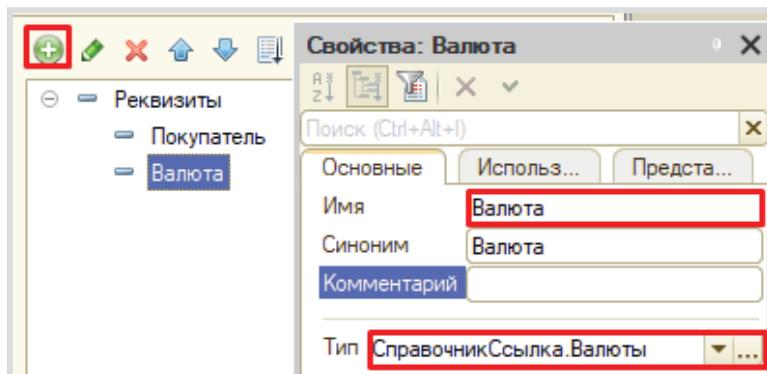


Из условия следует, что при продаже нужно указывать покупателя и валюту. Также будем хранить и итоговую стоимость.

Добавим реквизит «Покупатель».

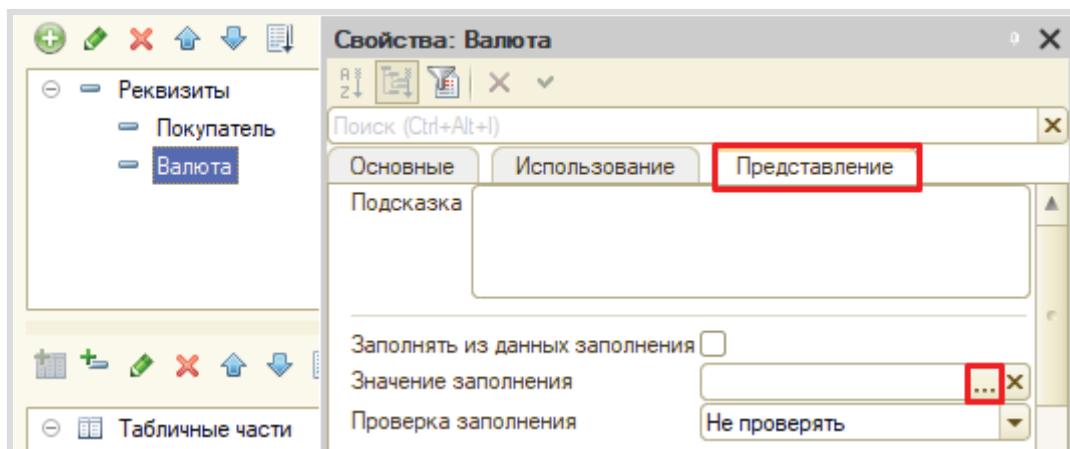


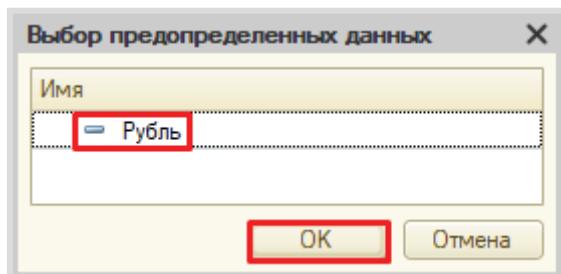
Далее добавим реквизит «Валюта».



Пользователь может не использовать возможность ведения учета в различных валютах. В этом случае поле «Валюта» у него будет отсутствовать (этот функционал мы реализуем далее), но системе необходимо понимать, к чему привязана итоговая стоимость: к рублям или, например, долларам.

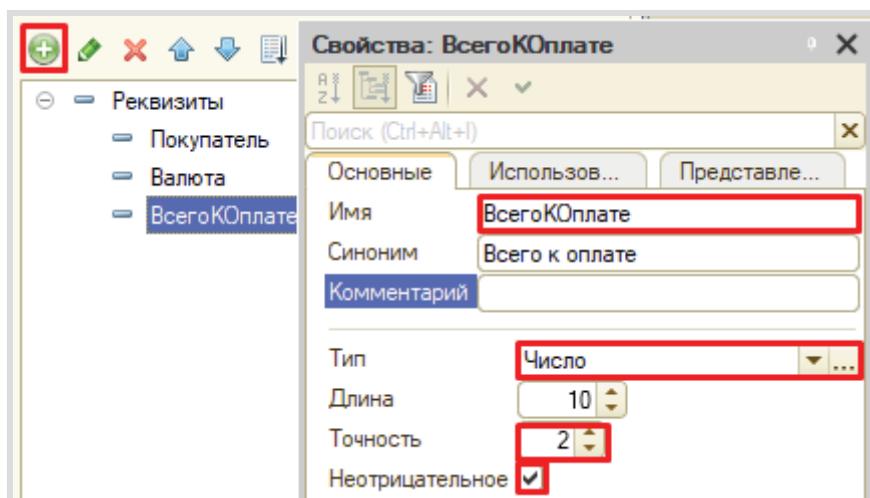
Поэтому установим значения заполнения по умолчанию на вкладке «Представление».





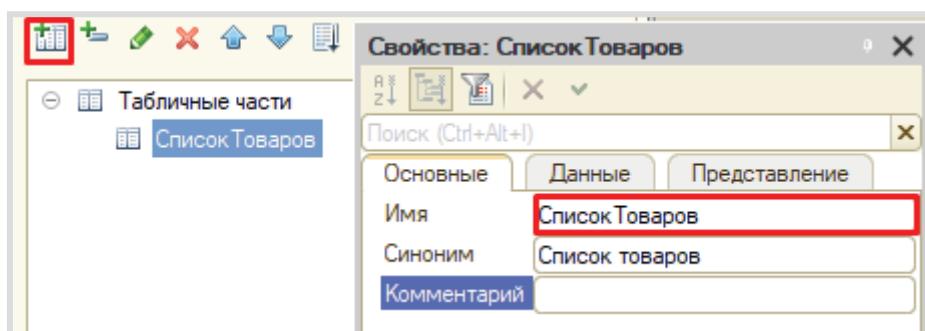
Таким образом, у пользователя по умолчанию будет указана валюта «Рубль», причем пользователь об этом может даже не знать.

Последним реквизитом в шапку документа добавим «ВсегоКОплате».

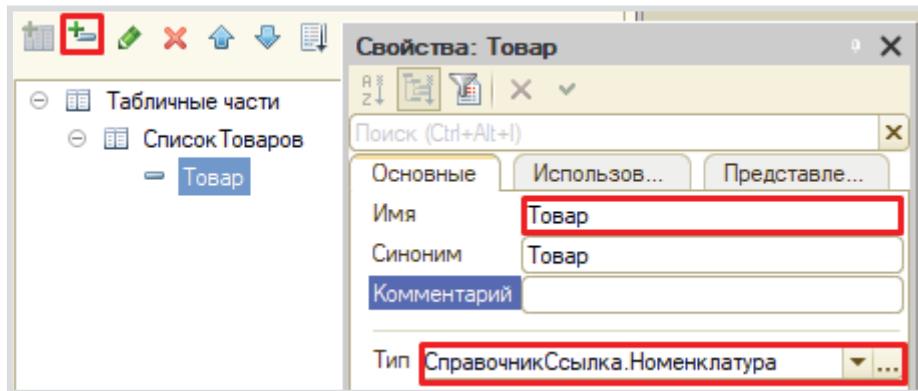


Чтобы зарегистрировать продажу товаров и услуг в одном документе, необходимо добавить две табличные части.

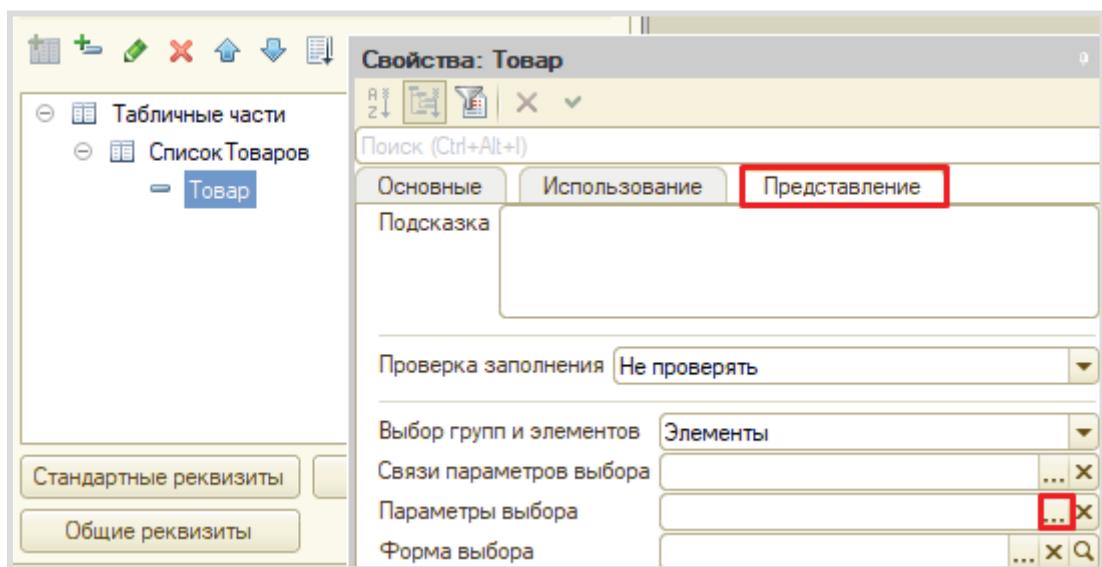
Сначала добавим табличную часть «СписокТоваров».

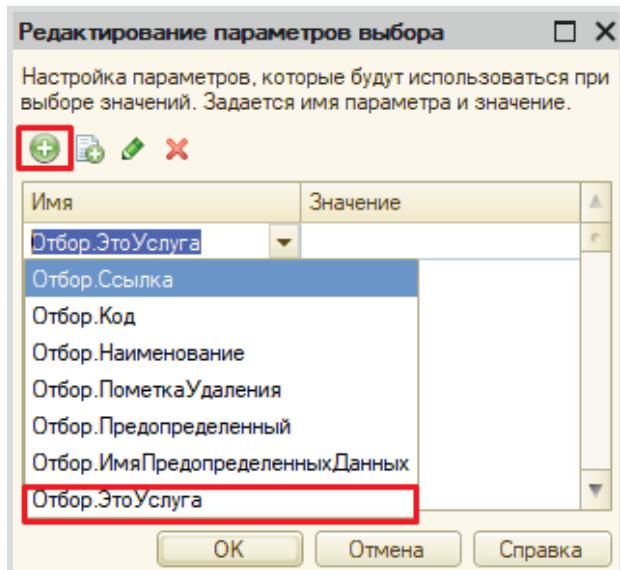


Добавим реквизит табличной части (колонку) «Товар».

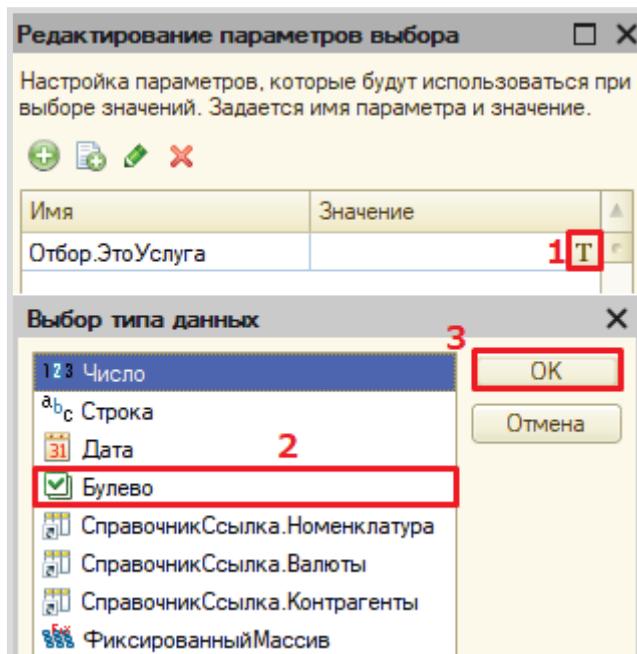


Поскольку мы ссылаемся на общий справочник «Номенклатура», то нам нужно на вкладке «Представление» настроить отбор только по тем элементам, у которых значение реквизита «ЭтоУслуга» установлено в положение «Ложь». Таким образом, пользователь будет в списке выбора видеть только товары.

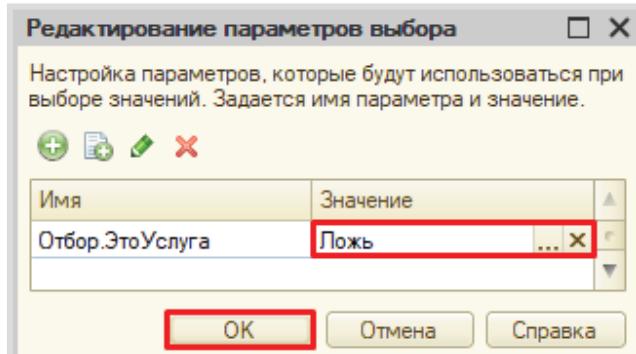




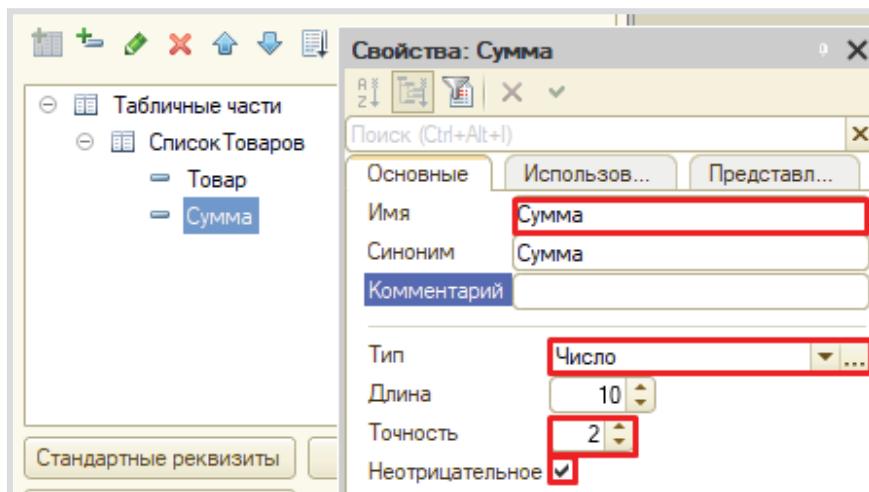
Поскольку в справочнике «Номенклатура» мы указывали для этого реквизита тип «Булево», то в значении параметра нужно указать именно его.



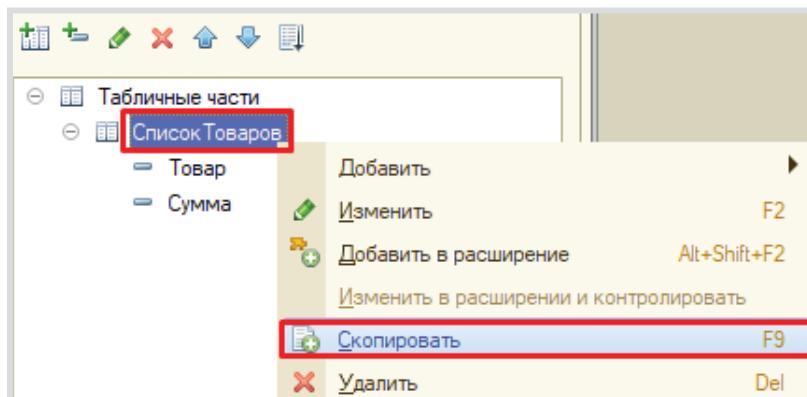
Для товаров значение «ЭтоУслуга» будет установлено в положении «Ложь».



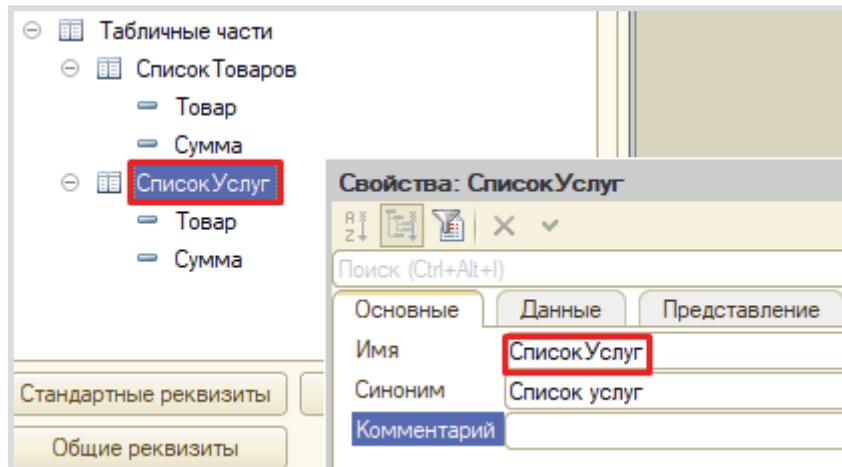
Далее добавим реквизит табличной части «Сумма».



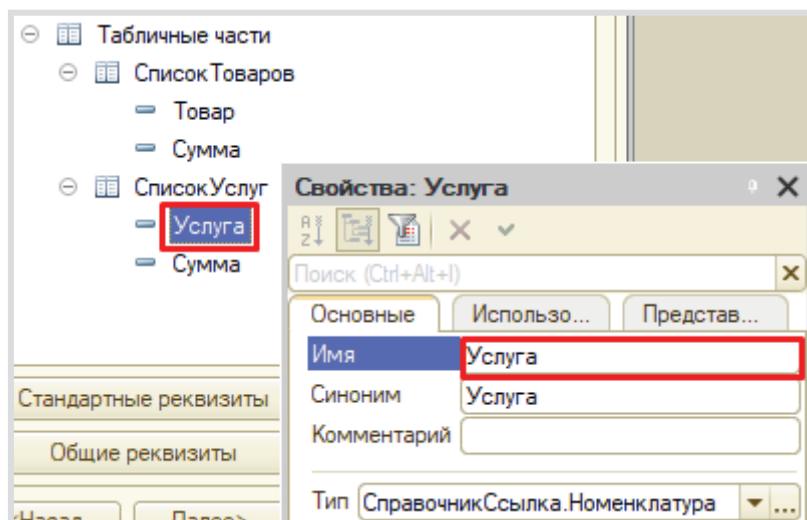
Табличная часть «СписокУслуг» структурно будет совпадать с уже созданной табличной частью «СписокТоваров». Для увеличения скорости разработки скопируем существующую табличную часть и несколько изменим ее.



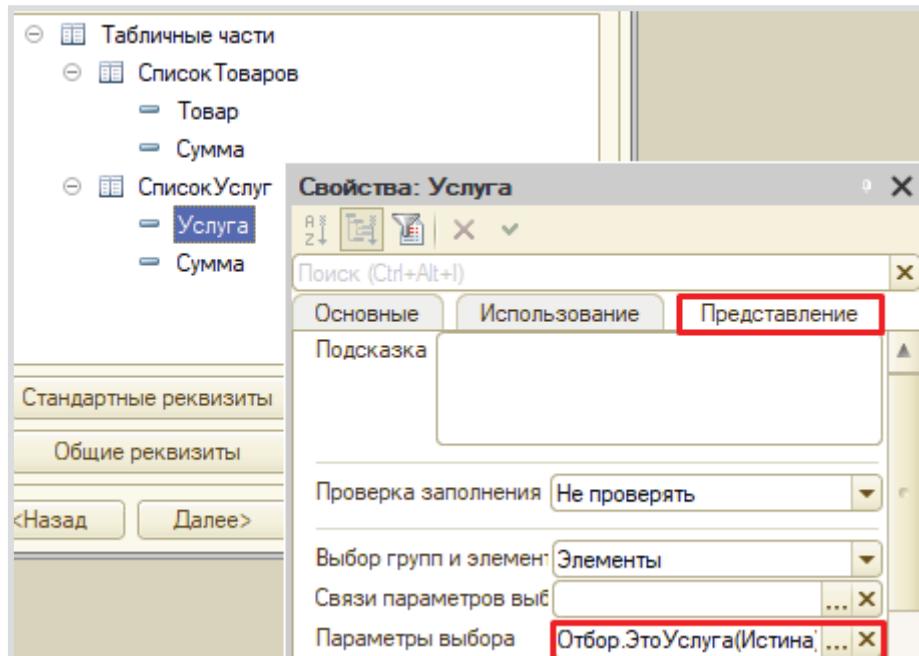
Во-первых, изменим свойство «Имя» у скопированной табличной части. Назовите табличную часть «СписокУслуг».



Во-вторых, изменим свойство «Имя» у реквизита табличной части «Товар». Поскольку данная табличная часть будет хранить перечень услуг, то и реквизиту нужно дать имя «Услуга».

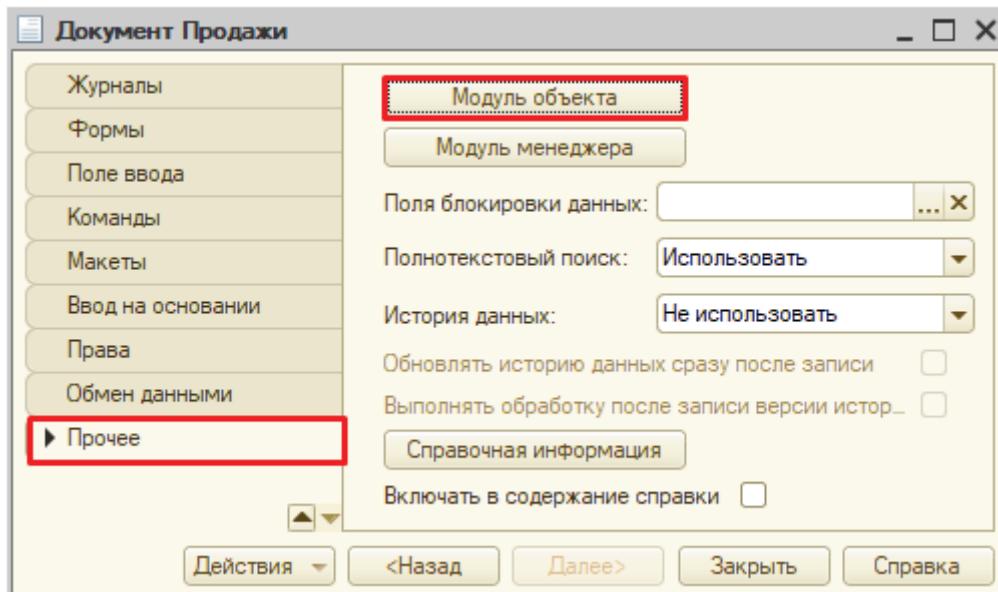


В-третьих, на вкладке «Представление» изменим параметры выбора: для услуг значение типа «Булево» будет в положении «Истина».



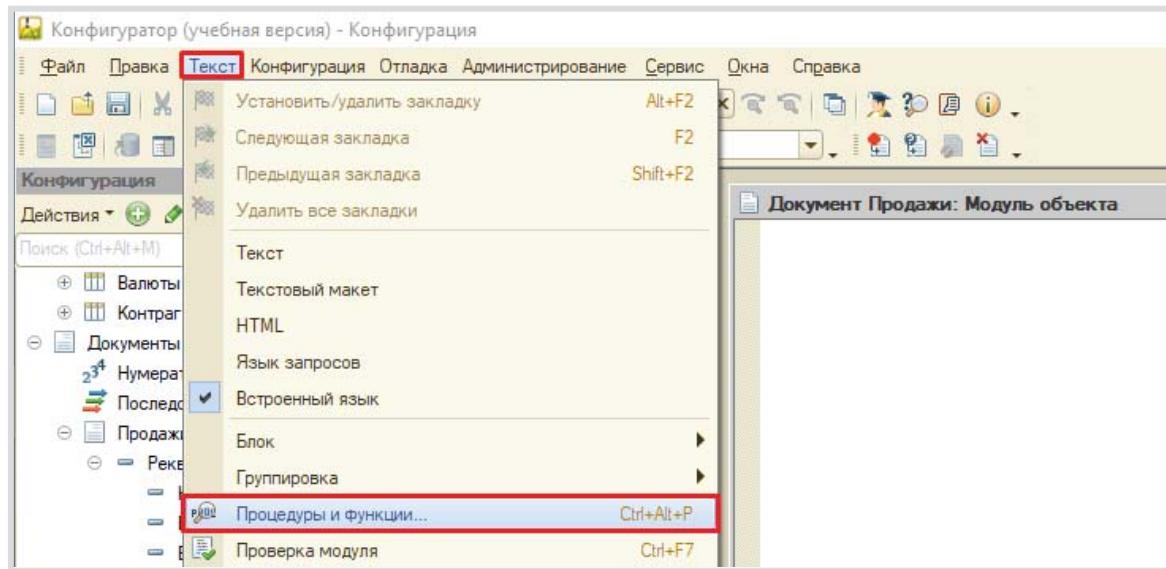
Чтобы итоговая стоимость формировалась автоматически, нам нужно описать событие в модуле документа «Продажи».

Для этого перейдем на вкладку «Прочее» и откроем «Модуль объекта».

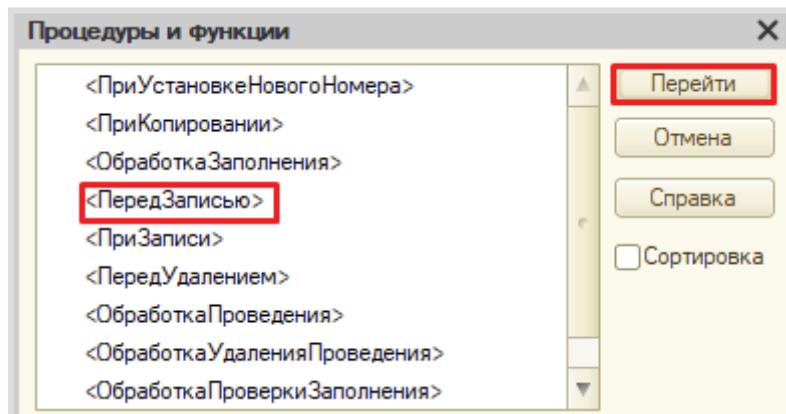


Перед нами откроется модуль объекта, в котором нужно определить событие, при наступлении которого будет происходить описываемый алгоритм.

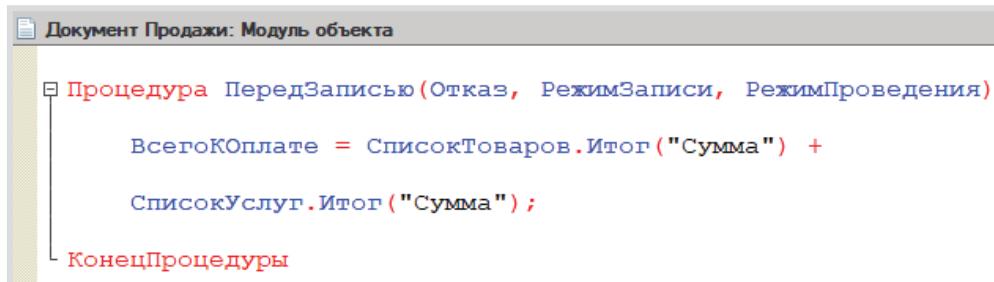
Нам понадобится системное событие, вызовем его через главное меню «Текст» → «Процедуры и функции».



В открывшемся окне выберем «ПередЗаписью».



В модуле объекта сформируется обработчик события. Далее нам нужно описать подсчет итоговой суммы как итог по товарам и услугам.



```

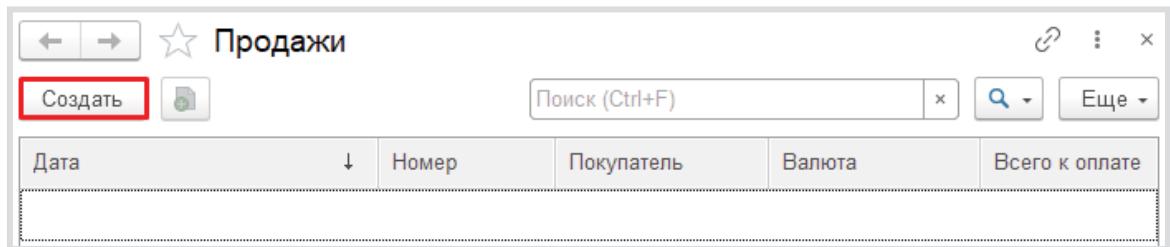
Документ Продажи: Модуль объекта

Процедура ПередЗаписью(Отказ, РежимЗаписи, РежимПроведения)
    ВсегоКОплате = СписокТоваров.Итог("Сумма") +
        СписокУслуг.Итог("Сумма");
    КонецПроцедуры

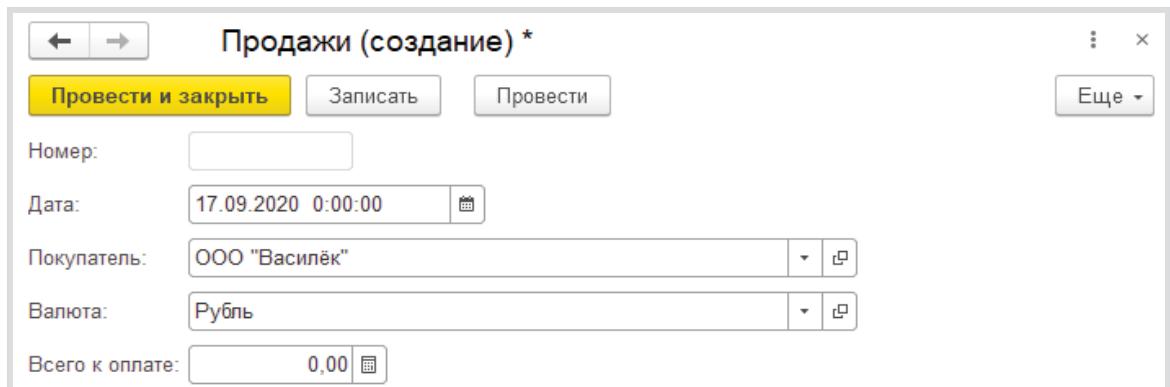
```

`ВсегоКОплате = СписокТоваров.Итог("Сумма") + СписокУслуг.Итог("Сумма");`

Запустим режим «1С:Предприятие» и создадим несколько документов.



Заполним поля шапки документа, проигнорировав поле «Всего к оплате» (оно заполнится автоматически при записи).



The screenshot shows the 'Продажи (создание)' (Sales creation) dialog. It includes buttons for 'Провести и закрыть' (Post and close), 'Запись' (Save), and 'Провести' (Post). The dialog fields are: 'Номер:' (Number) with an empty input field; 'Дата:' (Date) with a value of '17.09.2020 0:00:00' and a calendar icon; 'Покупатель:' (Buyer) with a value of 'ООО "Василёк"'; 'Валюта:' (Currency) with a value of 'Рубль'; and 'Всего к оплате:' (Total to pay) with a value of '0.00' and a currency icon. The 'Валюта' field has a dropdown arrow icon.

Обратим внимание, что поле «Валюта» уже было заполнено по умолчанию.

Затем заполним таблицы «Список товаров» и «Список услуг».

Список товаров		Список услуг	
Добавить		Поиск (Ctrl+F)	Еще ▾
N	Товар	Сумма	
1	Компьютер		55 000,00

Список товаров		Список услуг	
Добавить		Поиск (Ctrl+F)	Еще ▾
N	Услуга	Сумма	
1	Доставка		3 000,00

Внимание!

В список товаров можно добавлять только товары, а в список услуг – только услуги.

В момент записи или проведения документа поле «Всего к оплате» заполнится автоматически.

Продажи 000000001 от 17.09.2020 14:19:39

<input type="button" value="Провести и закрыть"/>	<input type="button" value="Записать"/>	<input style="border: 2px solid red;" type="button" value="Провести"/>	<input type="button" value="Еще ▾"/>
Номер:	000000001		
Дата:	17.09.2020 14:19:39		
Покупатель:	ООО "Василёк"		
Валюта:	Рубль		
Всего к оплате:	58 000,00	<input type="button" value=""/>	
<input type="button" value="Список товаров"/> <input type="button" value="Список услуг"/>			

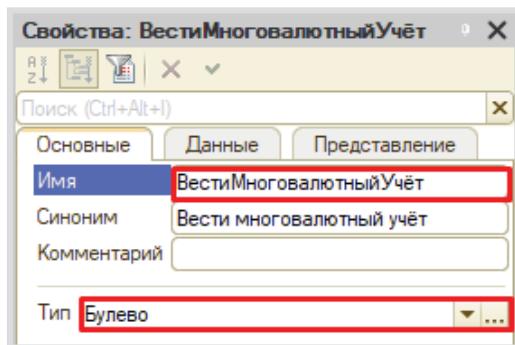
Создайте еще несколько документов на разных покупателей и разные валюты.

Дата	Номер	Покупатель	Валюта	Всего к оплате
17.09.2020 14:20:49	000000001	ООО "Василёк"	Рубль	58 000,00
17.09.2020 14:21:16	000000002	ООО "Василёк"	Доллар	1 445,00
17.09.2020 14:21:49	000000003	ООО "Мак"	Рубль	130 000,00

Сейчас мы по умолчанию установили валюту «Рубль», оставив пользователю возможность выбора другой валюты. Но возможность использовать разные валюты нам нужно сделать опциональной. Для этого воспользуемся общим механизмом *функциональные опции*. Более подробно про функциональные опции можно прочитать здесь: <https://v8.1c.ru/platforma/funktionalnaya-opciya/>.

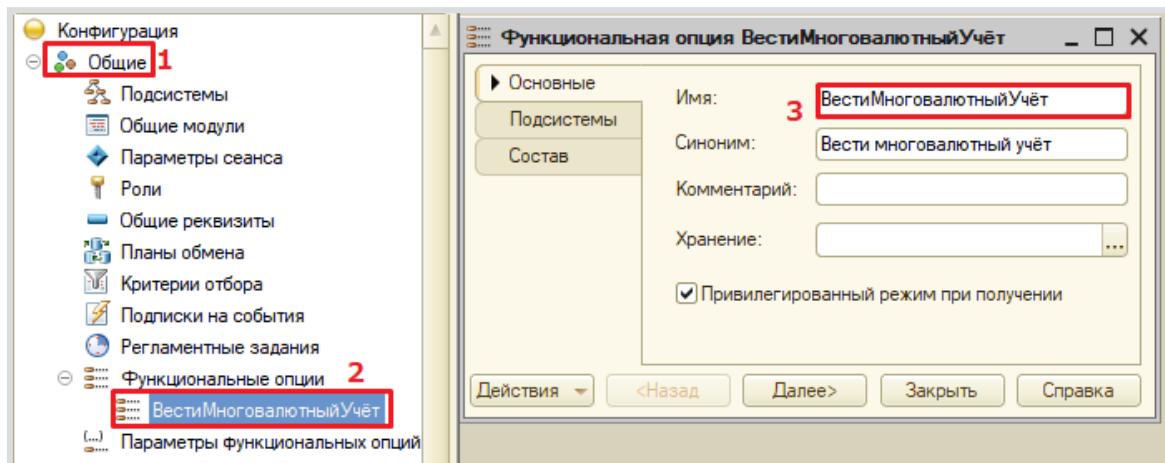
У пользователя должен быть физический способ включения и отключения многовалютного учета. Реализуем это с помощью обычной константы с типом «Булево».

Добавим константу «ВестиМноговалютныйУчет».



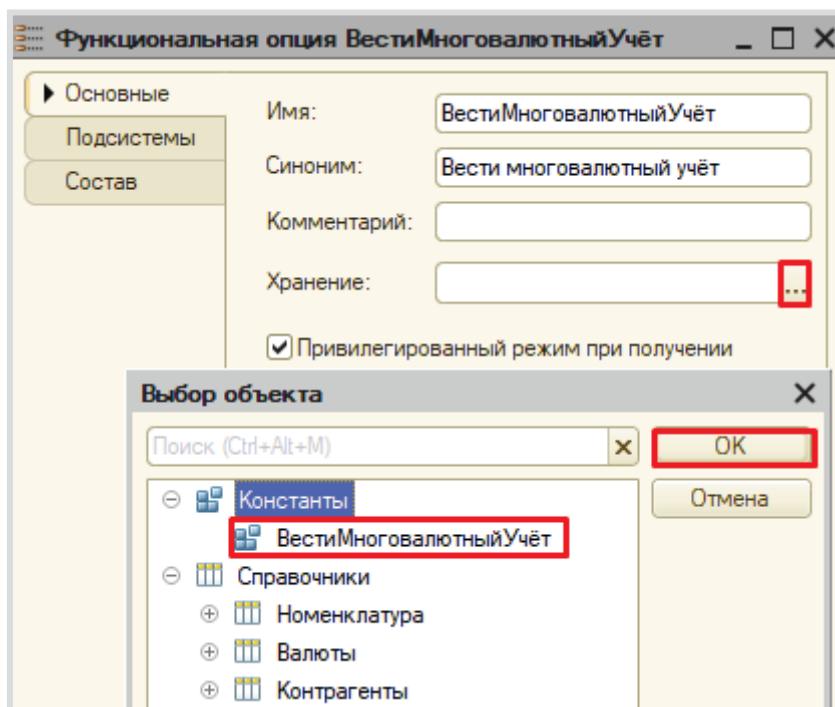
Теперь приступим к созданию и настройке функциональной опции.

В ветке «Общие» дерева конфигурации найдем раздел «Функциональные опции» и добавим опцию с именем «ВестиМноговалютныйУчет».



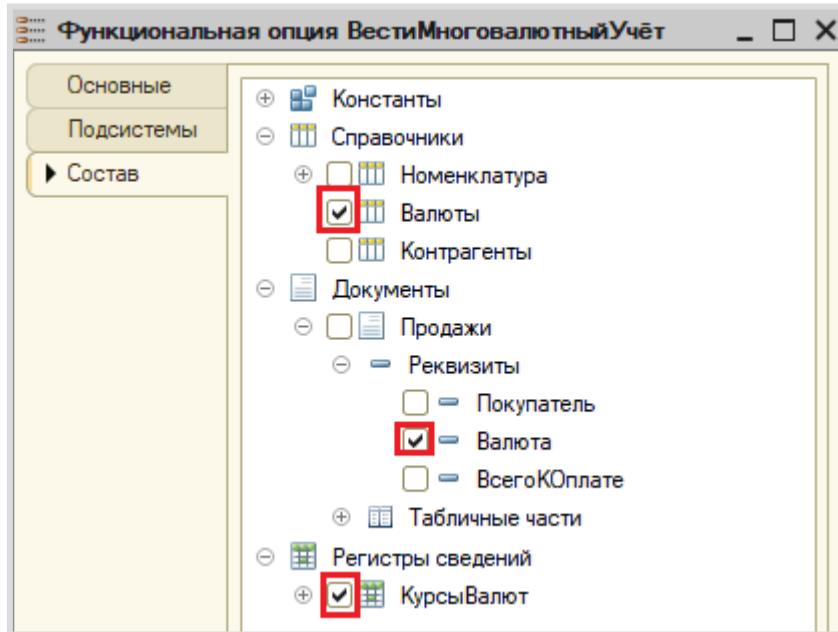
Теперь перейдем к ее настройке.

Первым делом нужно указать, где она будет храниться. Укажем в качестве места хранения заранее заготовленную константу с аналогичным именем.



Затем нужно указать, на что будет влиять наша функциональная опция. Она должна «включать» и «выключать» поле «Валюта» в документе «Продажи».

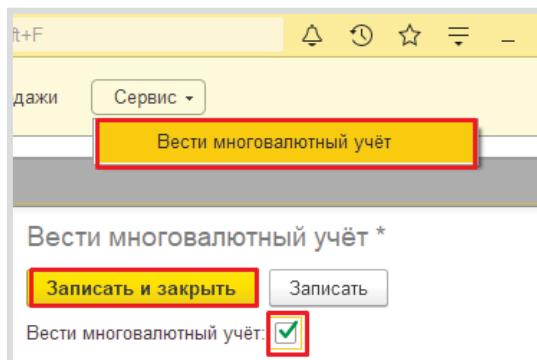
Укажем необходимые реквизиты на вкладке «Состав».



Запустим режим «1С:Предприятие» и проверим работоспособность.

Продажи				
Дата	Номер	Покупатель	Всего к оплате	
17.09.2020 14:20:49	000000001	ООО "Василёк"	58 000,00	
17.09.2020 14:21:16	000000002	ООО "Василёк"	1 445,00	
17.09.2020 14:21:49	000000003	ООО "Мак"	130 000,00	

Открыв список документов, мы обнаружим, что отсутствует колонка «Валюта». Это связано с тем, что значение типа «Булево» по умолчанию – «Ложь». Чтобы вновь иметь возможность выбирать валюты, следует у константы перевести значение в положение «Истина» и перезапустить режим «1С:Предприятие».



При повторном открытии списка всех документов после проделанных действий мы обнаружим, что колонка «Валюта» снова отображается.

Продажи					
Дата	Номер	Покупатель	Валюта	Всего к оплате	
17.09.2020 14:20:49	000000001	ООО "Василёк"	Рубль	58 000,00	
17.09.2020 14:21:16	000000002	ООО "Василёк"	Доллар	1 445,00	
17.09.2020 14:21:49	000000003	ООО "Мак"	Рубль	130 000,00	

«В системе нужно реализовать хранилище суммы доходов в рублях по номенклатурным позициям».

Для решения такого рода задачи нам потребуется *регистр накопления* вида «Обороты» (подробнее про регистры накоплений можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Добавим *регистр накопления* «Продажи».

Как и в случае с *регистром сведений*, на вкладке «Данные» определим структуру нашего регистра. Отвечая на вопросы: «что?» и «в разрезе чего?», укажем, что в качестве измерения будет выступать «Номенклатура», а в качестве ресурса – «Сумма».

The screenshot shows two 'Properties' dialog boxes side-by-side, both with tabs for 'Основные' (Main), 'Использование' (Usage), and 'Представление' (Presentation).

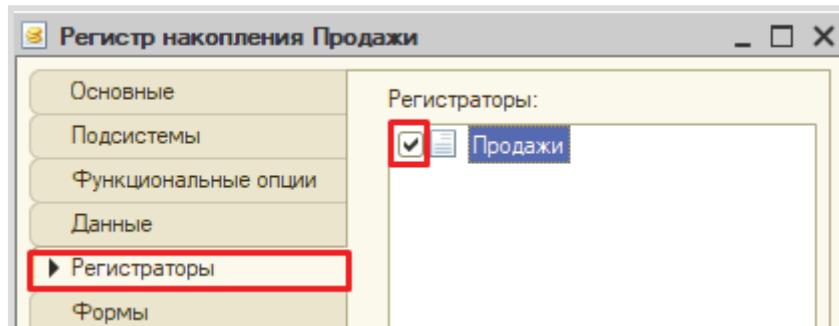
Properties: Номенклатура (Nomenclature)

- Основные (Main):**
 - Имя: Номенклатура
 - Синоним: Номенклатура
 - Комментарий:
- Использование (Usage):**
 - Тип: СправочникСсылка.Номенклатура
 - Запрет незаполненных значений:
 - Использование в итогах:

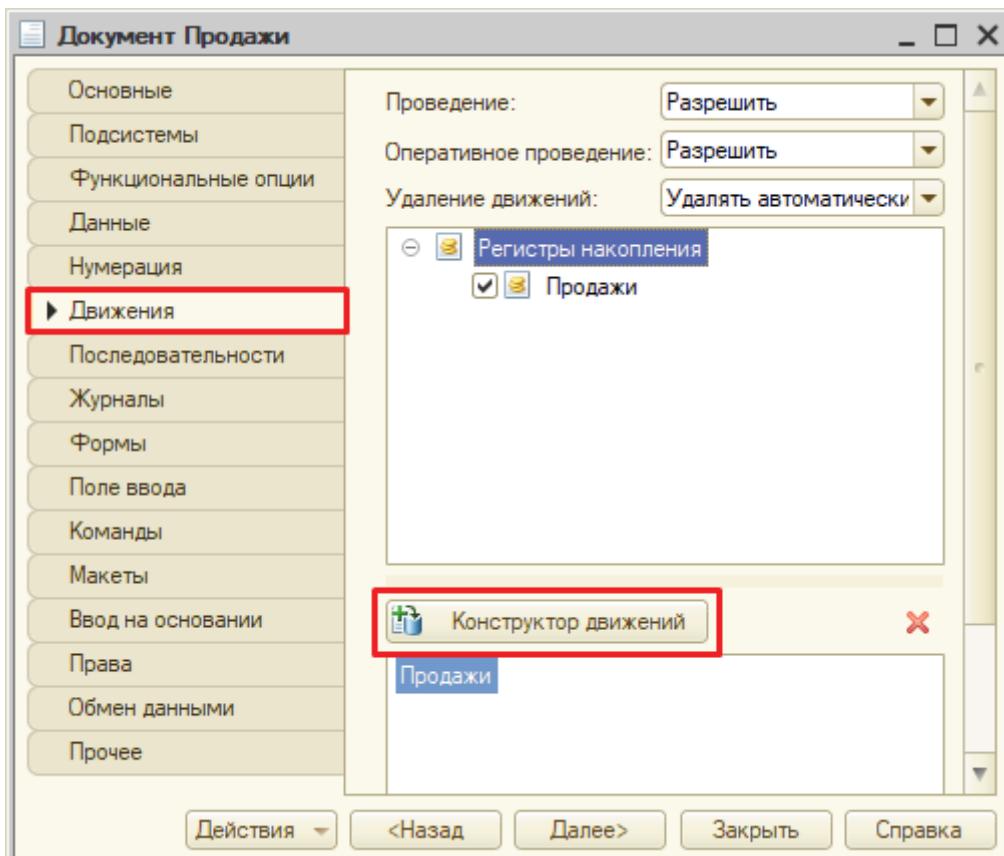
Properties: Сумма (Sum)

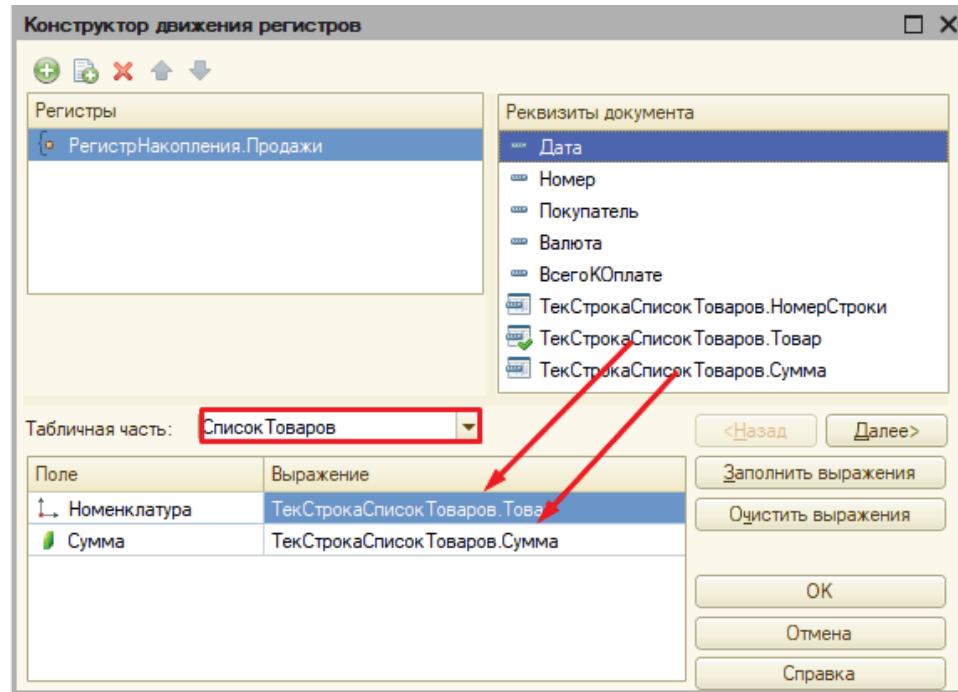
- Основные (Main):**
 - Имя: Сумма
 - Синоним: Сумма
 - Комментарий:
- Использование (Usage):**
 - Тип: Число
 - Длина: 10
 - Точность: 2
 - Неотрицательное:

Далее на вкладке «Регистраторы» следует указать документ-регистратор для нашего регистра.

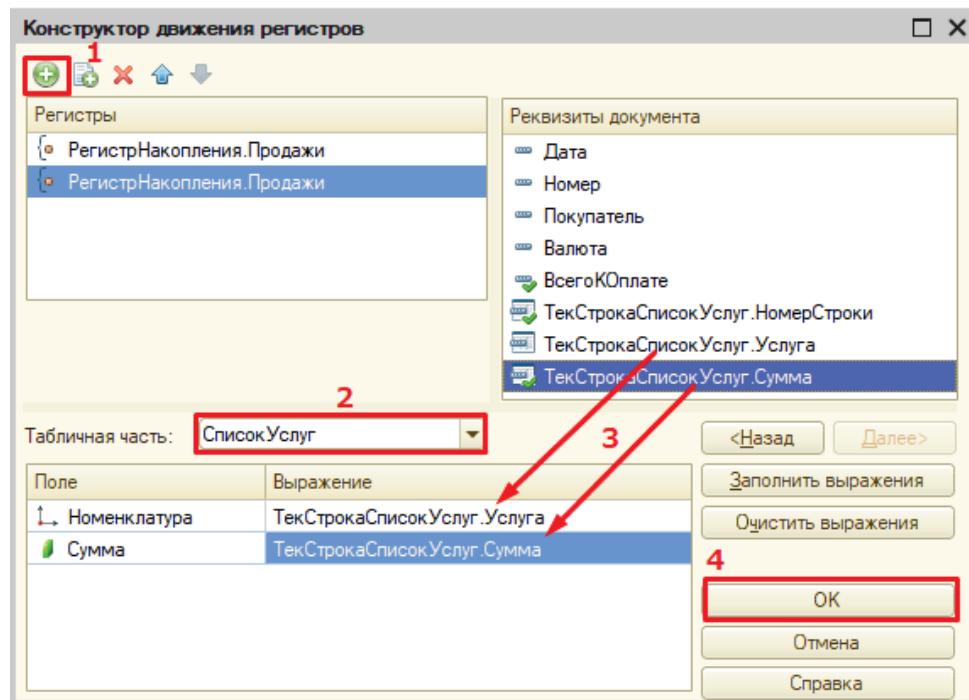


Но для совершения движений данных в регистр этого недостаточно. Необходимо описать правила с помощью конструктора движений документа «Продажи».





Но кроме табличной части с товарами есть и вторая часть – с услугами. Чтобы учитывать и ее – создадим еще одно движение в тот же регистр.



В модуле объекта (там же, где мы описывали итоговый подсчет) сформируется программный код, он будет совершать движения исключительно по числовым значениям реквизита «ВсегоКОплате», тем самым складывая рубли и другие валюты без разбора. Нам нужно это исправить.

Суммовые движения будут происходить как произведение суммы товара на его курс валюты, который мы будем получать с помощью функции – ее мы создадим позже.

```
□ Процедура ОбработкаПроведения(Отказ, Режим)

Движения.Продажи.Записывать = Истина;
для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.Продажи.Добавить();
    Движение.Период = Дата;
    Движение.Номенклатура = ТекСтрокаСписокТоваров.Товар;
    Движение.Сумма = ТекСтрокаСписокТоваров.Сумма * ПолучитьКурсВалюты(Валюта, Дата);
КонецЦикла;

Движения.Продажи.Записывать = Истина;
для Каждого ТекСтрокаСписокУслуг Из СписокУслуг Цикл
    Движение = Движения.Продажи.Добавить();
    Движение.Период = Дата;
    Движение.Номенклатура = ТекСтрокаСписокУслуг.Услуга;
    Движение.Сумма = ТекСтрокаСписокУслуг.Сумма * ПолучитьКурсВалюты(Валюта, Дата);
КонецЦикла;

КонецПроцедуры
```

* ПолучитьКурсВалюты(Валюта, Дата)

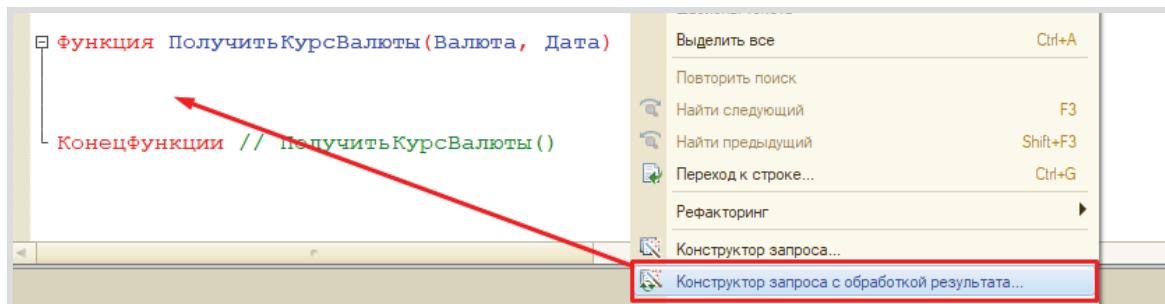
Следующим этапом будет написание функции, которая будет обращаться к *регистру сведений* и получать актуальный курс валют.

```
□ Функция ПолучитьКурсВалюты(Валюта, Дата)

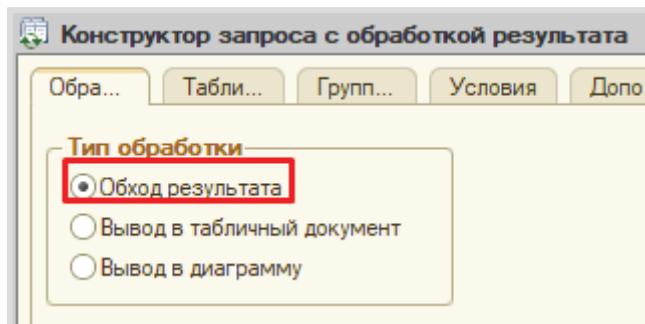
КонецФункции // ПолучитьКурсВалюты()
```

Информацию мы будем получать с помощью запроса к виртуальной таблице – нам нужен срез последних значений по указанным валюте и дате.

Установим курсор внутри функции и вызовем контекстное меню правой кнопкой мыши.



В открывшемся окне на вкладке «Обработка результата» следует выбрать тип обработки «Обход результата».



Переходим на вкладку «Таблицы и поля».

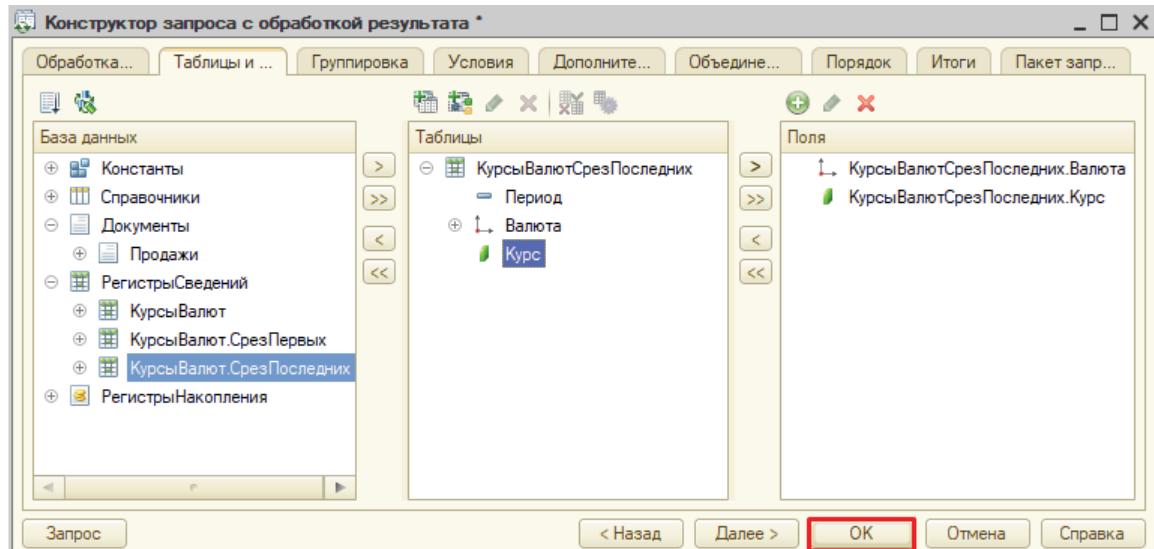
Открывается *конструктор запроса*. Эта вкладка имеет три части:

- ❑ Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- ❑ Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- ❑ Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать не из *регистра сведений* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица позволит получить последние введенные значения по каждой валюте.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



Далее нужно преобразовать полученный программный код.

```
Функция ПолучитьКурсВалюты(Валюта, Дата)

Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
|   КурсыВалютСрезПоследних.Валюта КАК Валюта,
|   КурсыВалютСрезПоследних.Курс КАК Курс
|ИЗ
|   РегистрСведений.КурсыВалют.СрезПоследних КАК КурсыВалютСрезПоследних";
РезультатЗапроса = Запрос.Выполнить();
ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
ВыборкаДетальныеЗаписи.Следующий();
КонецФункции // ПолучитьКурсВалюты()
```

Виртуальная таблица позволяет использовать встроенные методы для работы с параметрами. Такими параметрами у нас будут выступать *период* и *валюта*.

```
Запрос.Текст =
"ВЫБРАТЬ
|   КурсыВалютСрезПоследних.Валюта КАК Валюта,
|   КурсыВалютСрезПоследних.Курс КАК Курс
|ИЗ
|   РегистрСведений.КурсыВалют.СрезПоследних(&Период, Валюта = &Валюта) КАК КурсыВалютСрезПоследних";
```

Далее нужно установить значения этих параметров. Укажем их как передаваемые в функцию параметры «Валюта» и «Дата».

```
| РегистрСведений.КурсыВалют.СрезПоследних (&Период, Валюта = &Валюта) КАК КурсыВалютСрезПоследних;
| Запрос.УстановитьПараметр ("Валюта", Валюта);
| Запрос.УстановитьПараметр ("Период", НачалоДня (Дата));
РезультатЗапроса = Запрос.Выполнить();
```

Поскольку мы создаем функцию, то в самом конце нам нужно вернуть значение курса валюты.

```
ВыборкаДетальныеЗаписи.Следующий ();
Возврат ВыборкаДетальныеЗаписи.Курс;
КонецФункции // ПолучитьКурсВалюты()
```

Итоговый программный код функции «ПолучитьКурсВалюты» будет таким:

```
Функция ПолучитьКурсВалюты(Валюта, Дата)
Запрос = Новый Запрос;
Запрос.Текст =
    "ВЫБРАТЬ
    | КурсыВалютСрезПоследних.Валюта КАК Валюта,
    | КурсыВалютСрезПоследних.Курс КАК Курс
    | ИЗ
    | РегистрСведений.КурсыВалют.СрезПоследних (&Период, Валюта = &Валюта) КАК КурсыВалютСрезПоследних";
Запрос.УстановитьПараметр ("Валюта", Валюта);
Запрос.УстановитьПараметр ("Период", НачалоДня (Дата));
РезультатЗапроса = Запрос.Выполнить();
ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
Возврат ВыборкаДетальныеЗаписи.Курс;
КонецФункции // ПолучитьКурсВалюты()
```

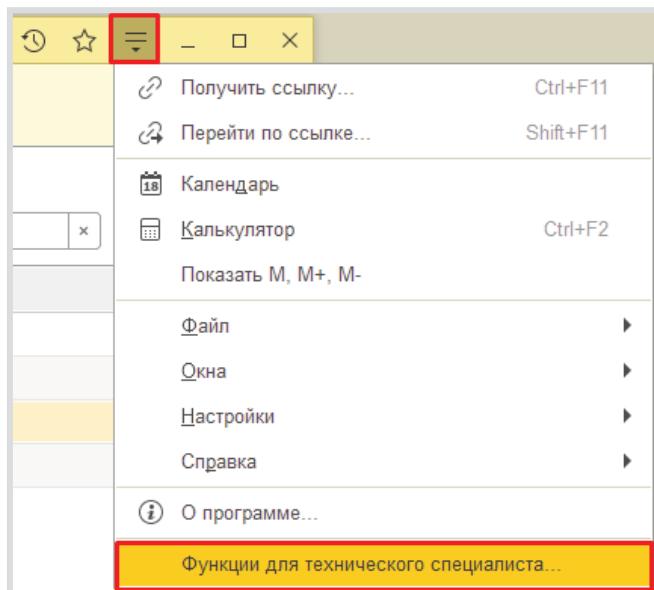
Чтобы движения были совершены в *регистр накопления*, необходимо перепровести (провести повторно) созданные документы.

Запустим режим «1С:Предприятие» и сделаем это.

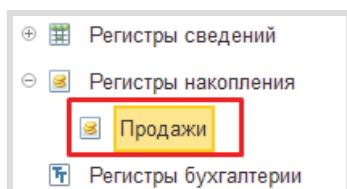
Дата	Номер	Покупатель	Валюта	Всего к оплате
01.01.2020 21:09:55	000000002	ООО "Василёк"	Доллар	1 000,00
01.01.2020 21:18:07	000000003	ООО "Мак"	Рубль	130 000,00
05.01.2020 21:18:08	000000001	ООО "Василёк"	Рубль	58 000,00
16.01.2020 21:18:09	000000004	ООО "Василёк"	Доллар	1 000,00

Обратите внимание, что на главной странице система не создала кнопку открытия регистра накопления. Это связано с тем, что все расчеты в регистрах накопления происходят в фоновом режиме, «за кадром», то есть пользователю о них знать не нужно вовсе. Поэтому по умолчанию регистры накопления настраивают так, чтобы пользователи не имели к ним доступа.

Но мы, будучи разработчиками, можем обратиться к любому объекту конфигурации. Для этого воспользуемся функциями для технического специалиста.



В общем перечне элементов следует найти *регистр накопления «Продажи»* и открыть его.



The screenshot shows a table with the following columns: Период, Регистратор, Номер строки, Номенклатура, and Сумма. The data is as follows:

Период	Регистратор	Номер строки	Номенклатура	Сумма
• 01.01.2020 21:09:55	Продажи 000000002 от 01.01.202...	1	Компьютер	48 000,00
• 01.01.2020 21:09:55	Продажи 000000002 от 01.01.202...	2	Доставка	12 000,00
• 01.01.2020 21:18:07	Продажи 000000003 от 01.01.202...	1	Компьютер	80 000,00
• 01.01.2020 21:18:07	Продажи 000000003 от 01.01.202...	2	Телефон	45 000,00
• 01.01.2020 21:18:07	Продажи 000000003 от 01.01.202...	3	Доставка	5 000,00
• 05.01.2020 21:18:08	Продажи 000000001 от 05.01.202...	1	Компьютер	55 000,00
• 05.01.2020 21:18:08	Продажи 000000001 от 05.01.202...	2	Доставка	3 000,00
• 16.01.2020 21:18:09	Продажи 000000004 от 16.01.202...	1	Компьютер	56 000,00
• 16.01.2020 21:18:09	Продажи 000000004 от 16.01.202...	2	Доставка	14 000,00

Если соотнести результаты, например, по движениям за 16 января, то заметим, что в регистр пришли значения с учетом указанного за 15 января курса, так как 16 числа не было изменений в курсе.

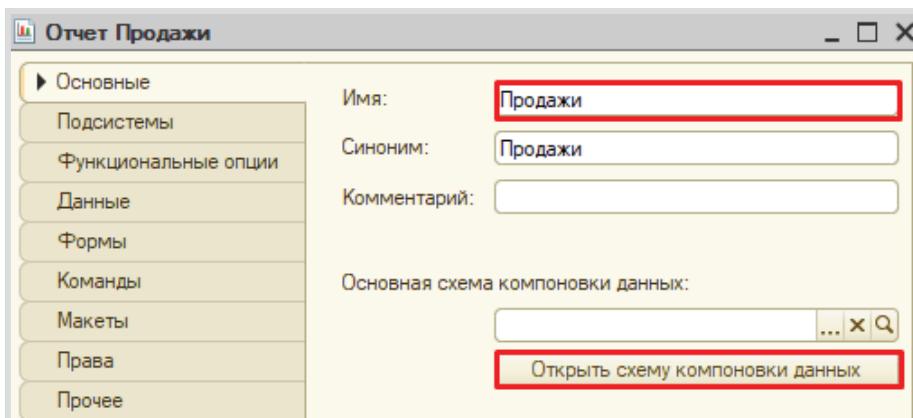
«Необходимо построить «Отчет по продажам» с упорядочиванием по сумме доходов».

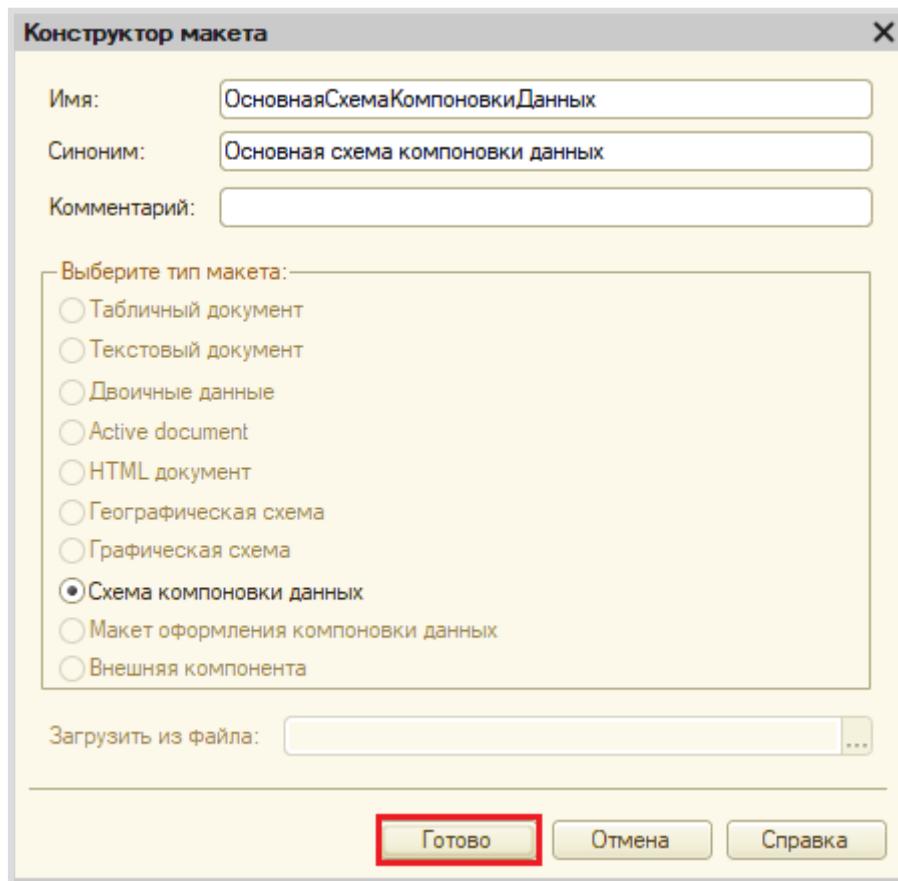
Построим отчет. Для этого воспользуемся соответствующим объектом конфигурации.

Определение

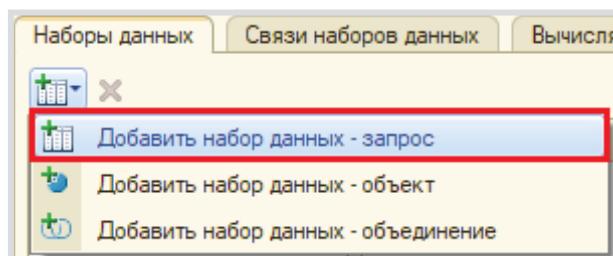
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Добавим отчет «Продажи». Воспользуемся *схемой компоновки данных*.

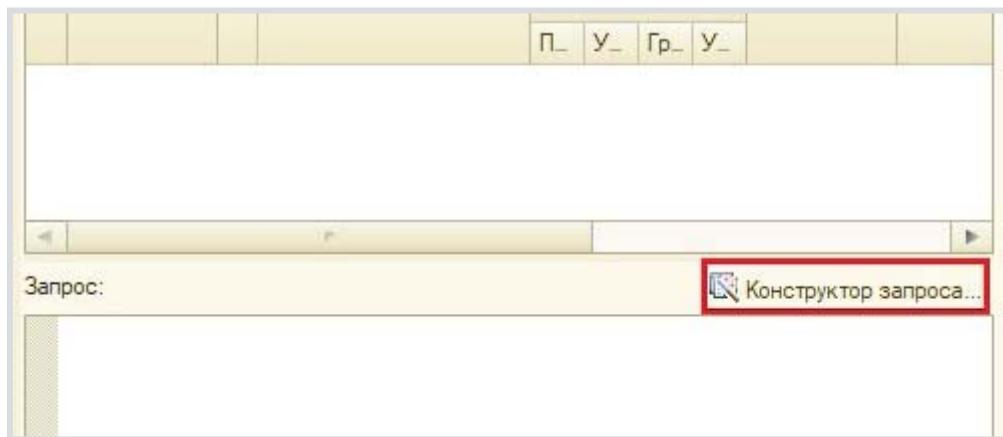




Добавим новый запрос к базе данных.



Для формирования запроса воспользуемся конструктором запроса.



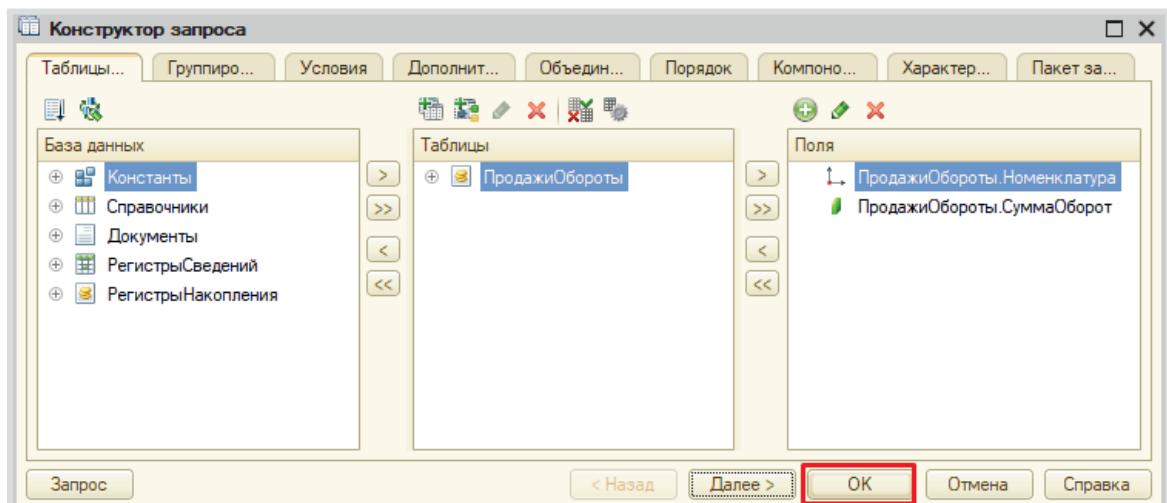
Открывается конструктор запроса. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать из *регистра накопления* напрямую, чтобы иметь возможность рассчитывать средний балл.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



У нас сформируется запрос на встроенном языке запросов 1С.

```

Запрос: Конструктор запроса...

ВЫБРАТЬ
    ПродажиОбороты.Номенклатура КАК Номенклатура,
    ПродажиОбороты.СуммаОборот КАК СуммаОборот
ИЗ
    РегистрНакопления.Продажи.Обороты КАК ПродажиОбороты

```

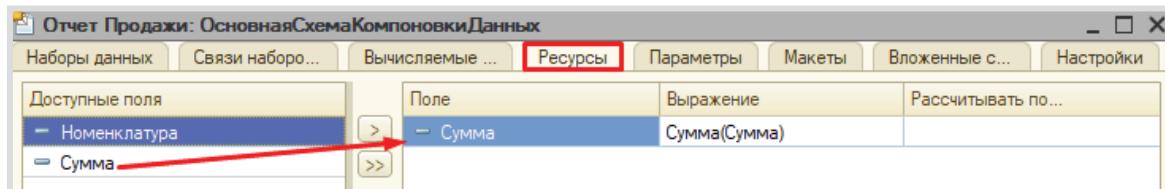
Для красоты зададим синоним полю «СуммаОборот». Сделать это можно с помощью **конструктора запроса** либо прямо в окне запроса.

```

ВЫБРАТЬ
    ПродажиОбороты.Номенклатура КАК Номенклатура,
    ПродажиОбороты.СуммаОборот КАК Сумма
ИЗ
    РегистрНакопления.Продажи.Обороты КАК ПродажиОбороты

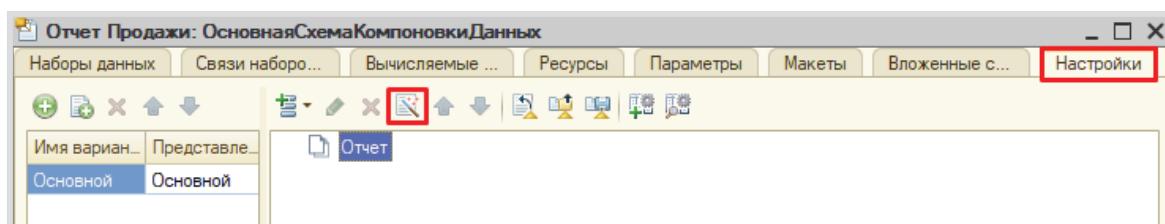
```

Далее мы реализуем итоговый подсчет всех сумм для нашего отчета, указав поле «Сумма» в качестве ресурса на соответствующей вкладке.

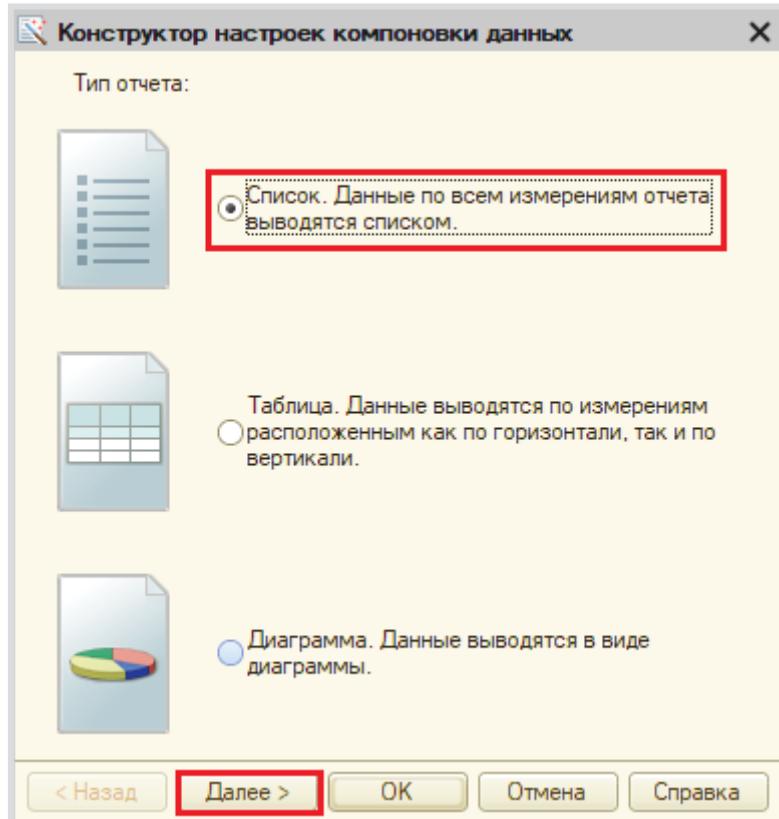


Теперь переходим на вкладку «Настройки» для оформления внешнего вида отчета.

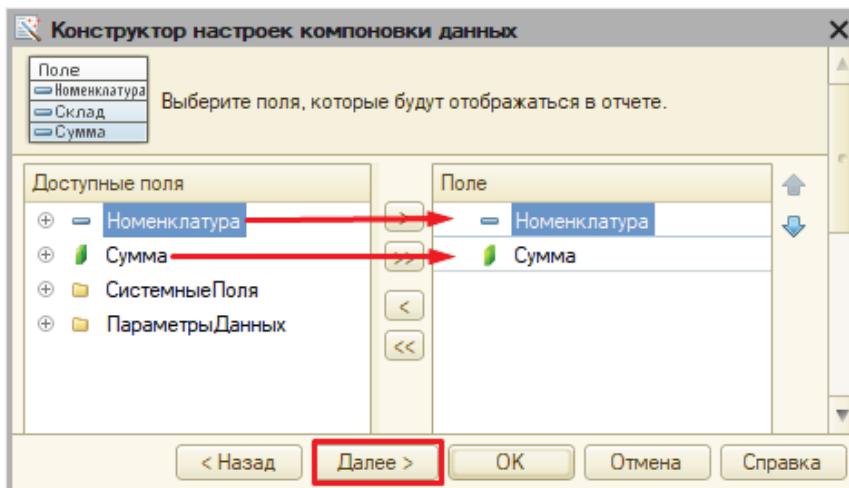
Воспользуемся *конструктором настроек отчета*.



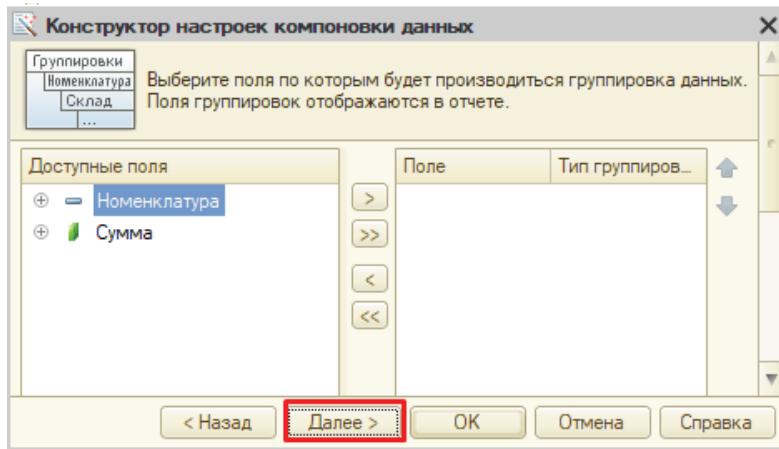
Построим отчет в виде списка.



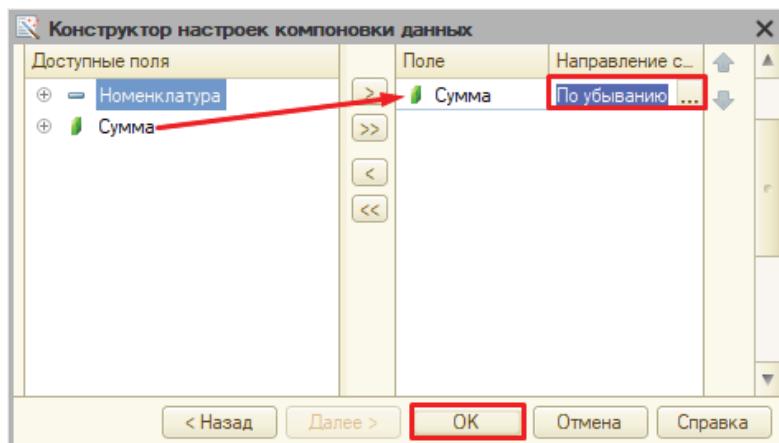
Выберем поля, которые будут отображаться в отчете.



Этап группировки пропустим.



Отсортируем отчет по возрастанию поля «Сумма».



Отчет готов. Запустим систему в режиме «1С:Предприятие» и построим отчет, указав разные валюты.

Продажи	
Сформировать	
Номенклатура	Сумма
Компьютер	239 000,00
Телефон	45 000,00
Доставка	34 000,00
Итого	318 000,00

Поставленная задача решена.

Лабораторная работа № 16

«ОТЛОВИТЬ» ПЕРВЫЙ ЗАПУСК ИНФОРМАЦИОННОЙ СИСТЕМЫ

Сложность: **

Теги: константы, форма, клиент, сервер,
программное открытие формы

ЗАДАНИЕ

После установки программы пользователю нужно заполнить некоторые параметры для дальнейшей работы. Без этого корректная работа программы невозможна.

Необходимо сделать так, чтобы при первом запуске системы открывалась форма с параметрами.

Признаком первого запуска будет служить константа со значением «ЛОЖЬ».

Для упрощения задачи форму следует открыть с простым текстом-декорацией «Обнаружен первый запуск программы!».

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

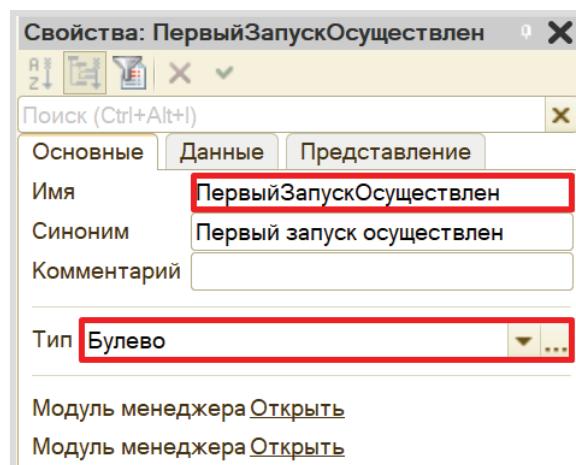
Выполнение

В первую очередь, нужно создать константу.

Информация

Константы нужны для хранения информации, которая практически никогда не меняется (а если меняется, то очень редко). Про константы подробнее можно прочитать здесь: <https://v8.1c.ru/platforma/konstanta/>.

Создадим новую константу «ПервыйЗапускОсуществлен». Тип значения – «Булево».



Напомним, что тип данных «Булево» подразумевает, что в данной константе могут находиться только значения «Истина» или «Ложь».

Данная константа должна имитировать первый запуск системы. Если константа содержит значение «Ложь» при запуске программы в режиме «1С:Предприятие», то следует оповестить пользователя о том, что это первый запуск системы. Если константа содержит значение «Истина», то при запуске программы окно с оповещением открываться не должно.

Для оповещения пользователя следует создать форму.

Определение

Форма – это окно, отображающее какую-либо информацию. Формы в 1С бывают различными: они могут принадлежать отдельным объектам конфигурации, например, справочникам. Также существуют общие формы, которые не привязаны к объектам конфигурации (подробнее про формы можно прочитать здесь: <https://v8.1c.ru/platforma/formy/>).

Для решения поставленной задачи нужно создать именно общую форму.

Создадим общую форму «УведомлениеОПервомЗапуске».

Подсказка

Объект конфигурации «Общие формы» находится во вкладке «Общие» окна дерева конфигурации.

При создании формы открывается *конструктор общих форм*. На первом шаге нужно указать название формы и ее тип. Тип создаваемой формы – произвольная. Обязательно поставьте галочку «Использовать стандартные команды».

Конструктор общих форм X

Выберите тип формы:

Произвольная форма

Форма констант

Форма отчета

Форма настроек отчета

Форма варианта отчета

Форма настроек динамического списка

Форма поиска

Форма истории изменений истории данных

Форма данных версии истории данных

Форма различий версий истории данных

Имя:

Синоним:

Комментарии...

Расширенное представление...

Пояснение:

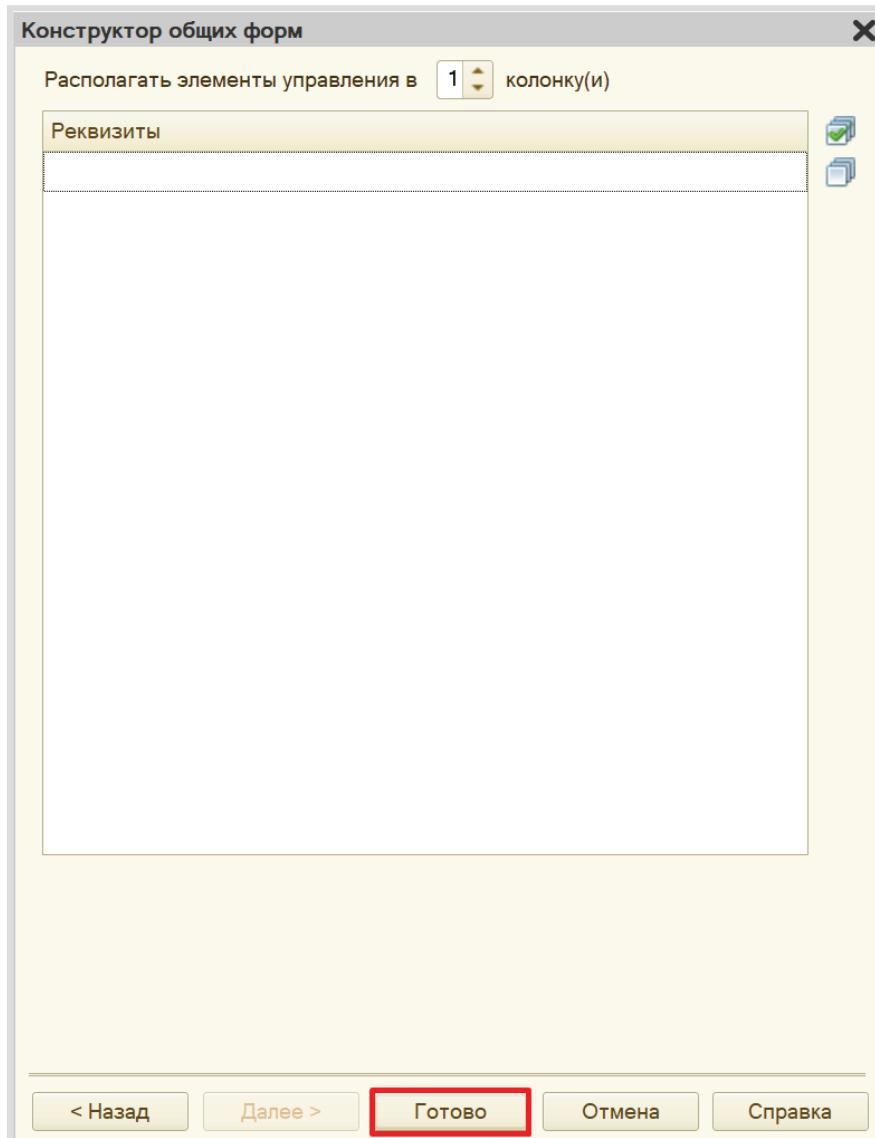
Использовать стандартные команды

Командная панель формы сверху

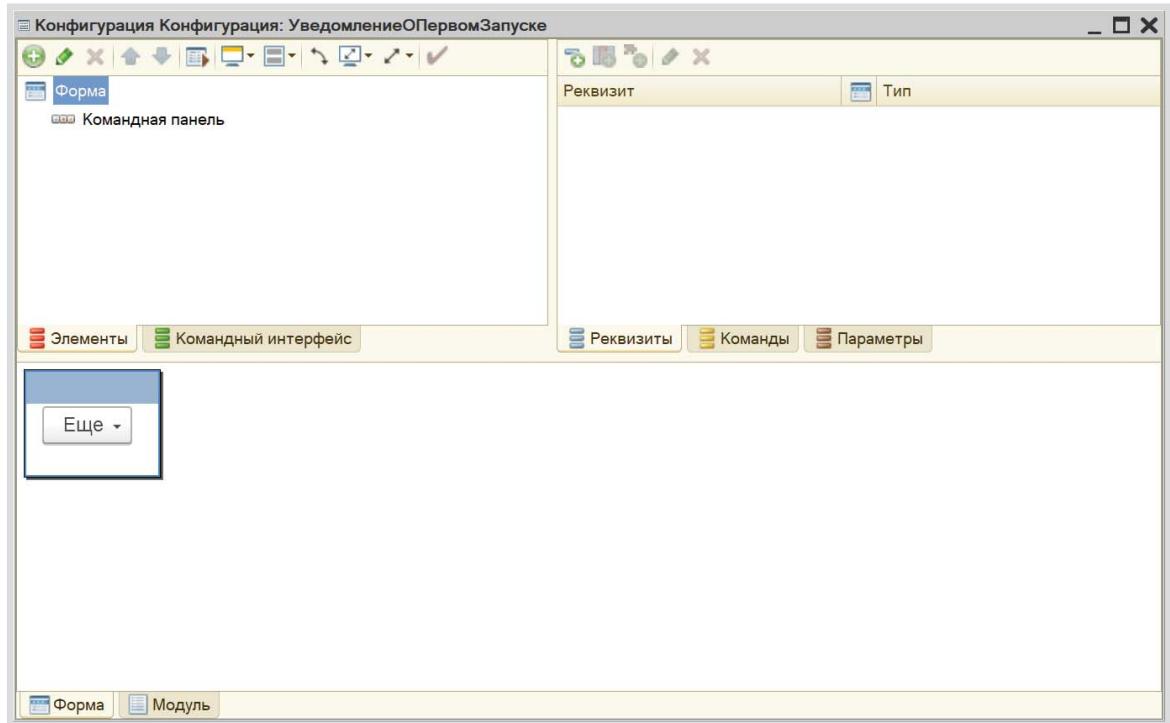
Командная панель формы снизу

[< Назад](#) Далее > [Готово](#) [Отмена](#) [Справка](#)

Так как у произвольной формы нет связанного объекта, то на втором шаге нам выбрать нечего, поэтому можно нажать кнопку «Готово».



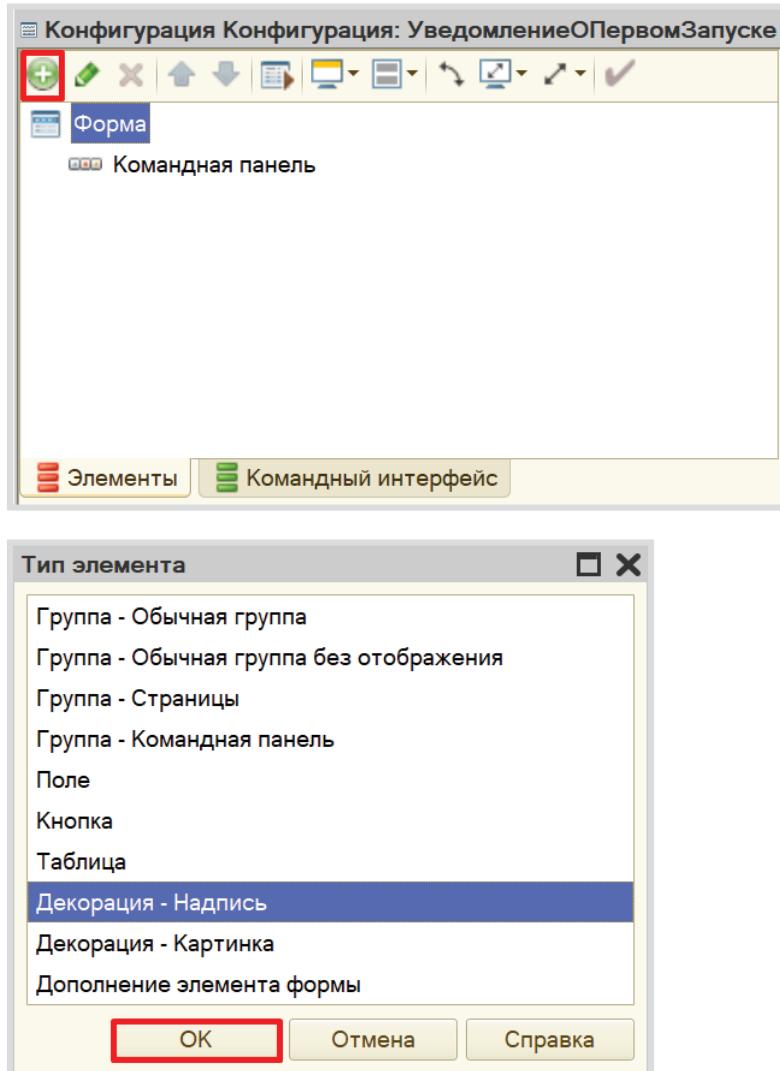
Открылся конструктор управляемой формы.



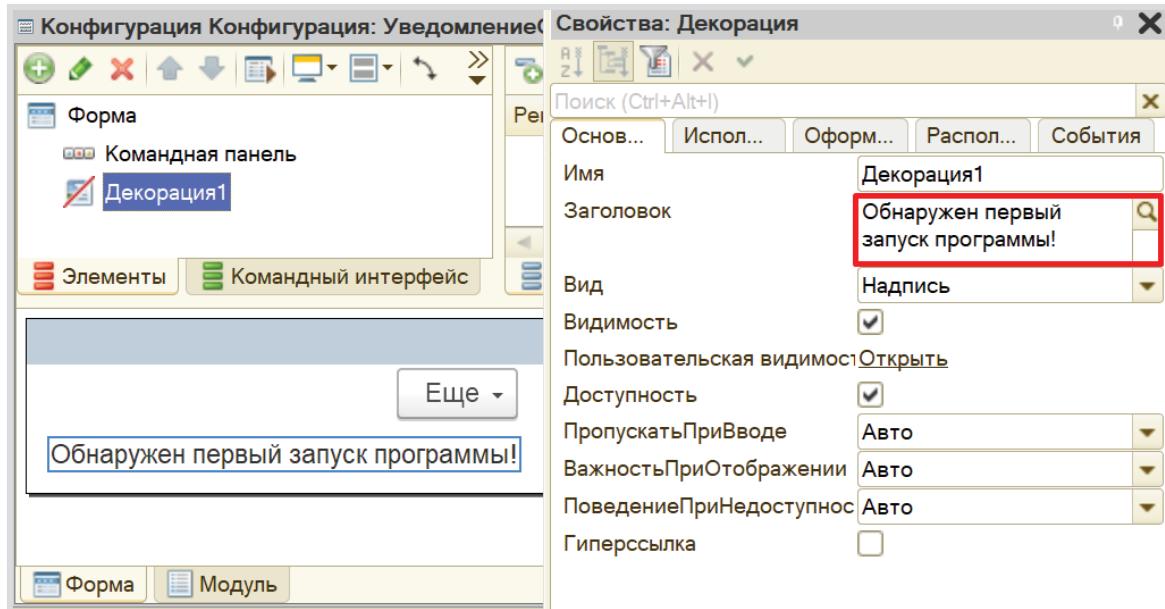
Конструктор формы состоит из трех областей, каждая из которых отвечает за ту или иную функциональность формы:

- Снизу находится область предпросмотра формы. Она позволяет лишь приблизительно понять, как будет отображаться данная форма для пользователя, поскольку может быть изменена с учетом множества различных факторов. Это своеобразная «иллюзия» того, что увидит пользователь.
- В правой верхней области находятся данные, которые мы вообще можем использовать в каком-либо виде на этой форме. Они разделены по вкладкам «Реквизиты», «Команды» и «Параметры».
- В левой верхней области *конструктора форм* описывается, какие именно данные будут изображены на форме, и то, как они будут выглядеть. Здесь – две вкладки: «Элементы» и «Командный интерфейс». На вкладке «Элементы» настраивается внешний вид и расположение реквизитов на форме. Вкладка «Командный интерфейс» определяет положение команд (кнопок) на форме.

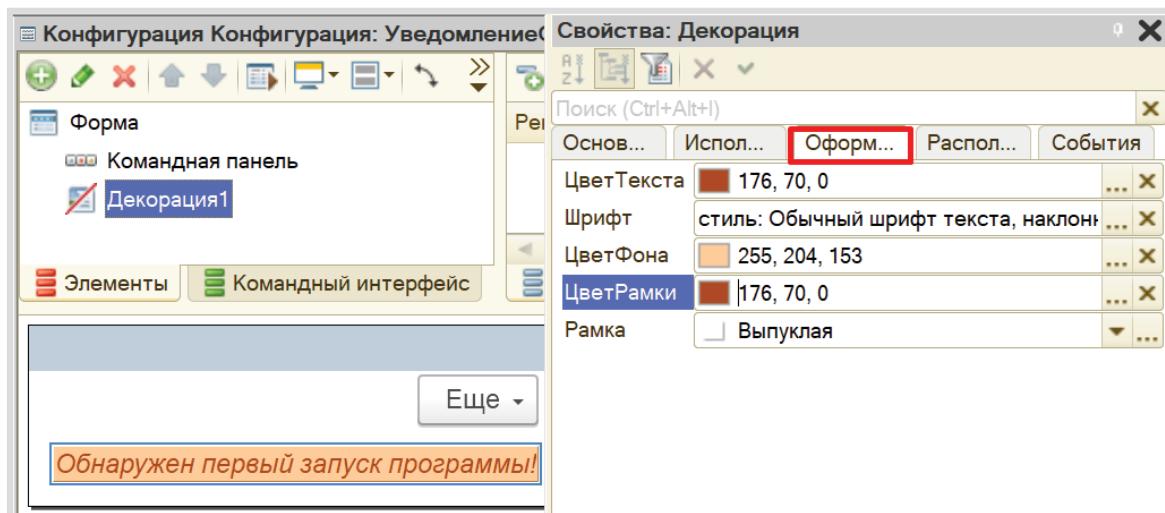
Добавим новый элемент формы: «Декорация – Надпись».



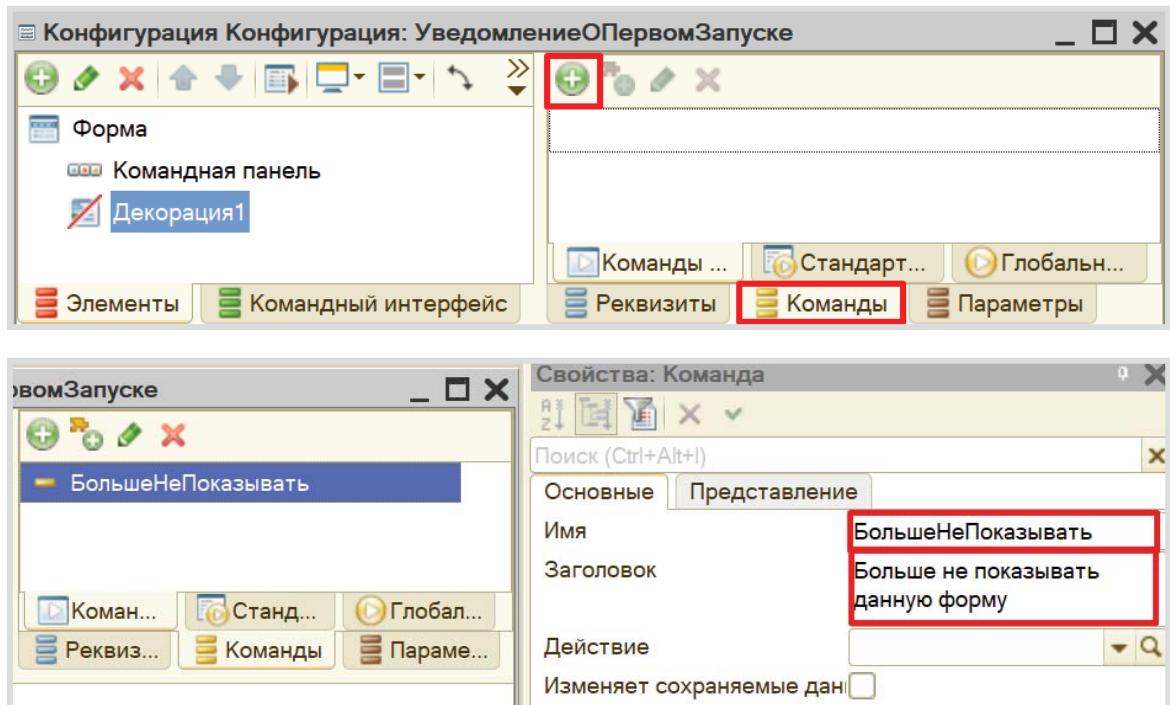
Среди элементов формы появится элемент «Декорация1», а также откроется палитра свойств данного элемента. Добавим заголовок: «Обнаружен первый запуск программы!» и посмотрим на результат в области предпросмотра.



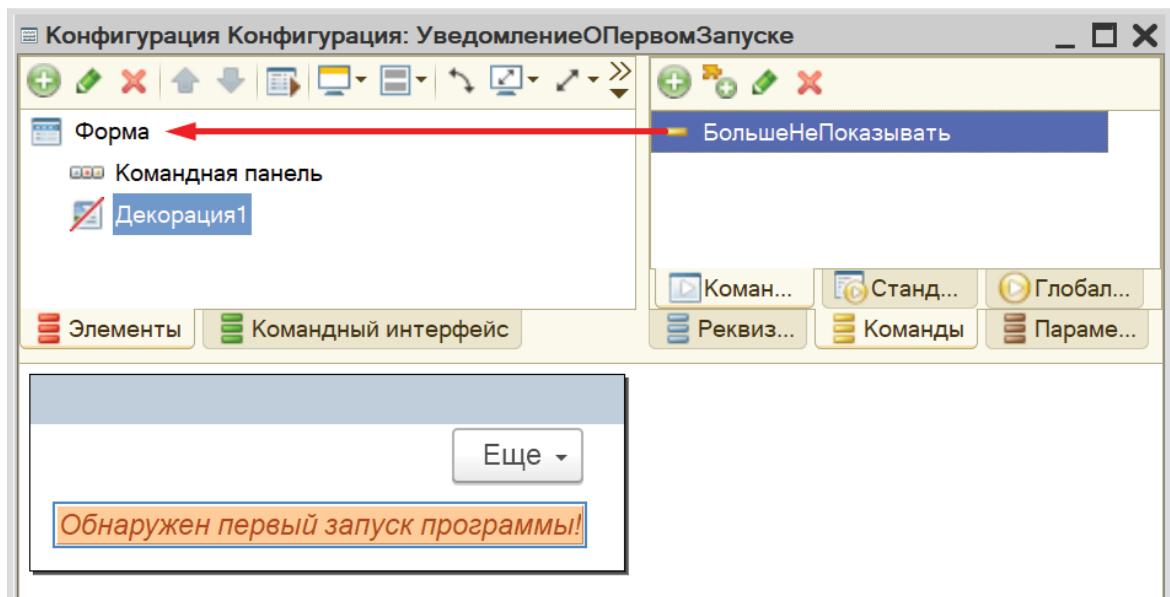
На вкладке «Оформление» можно настроить размер и цвет новой декорации.



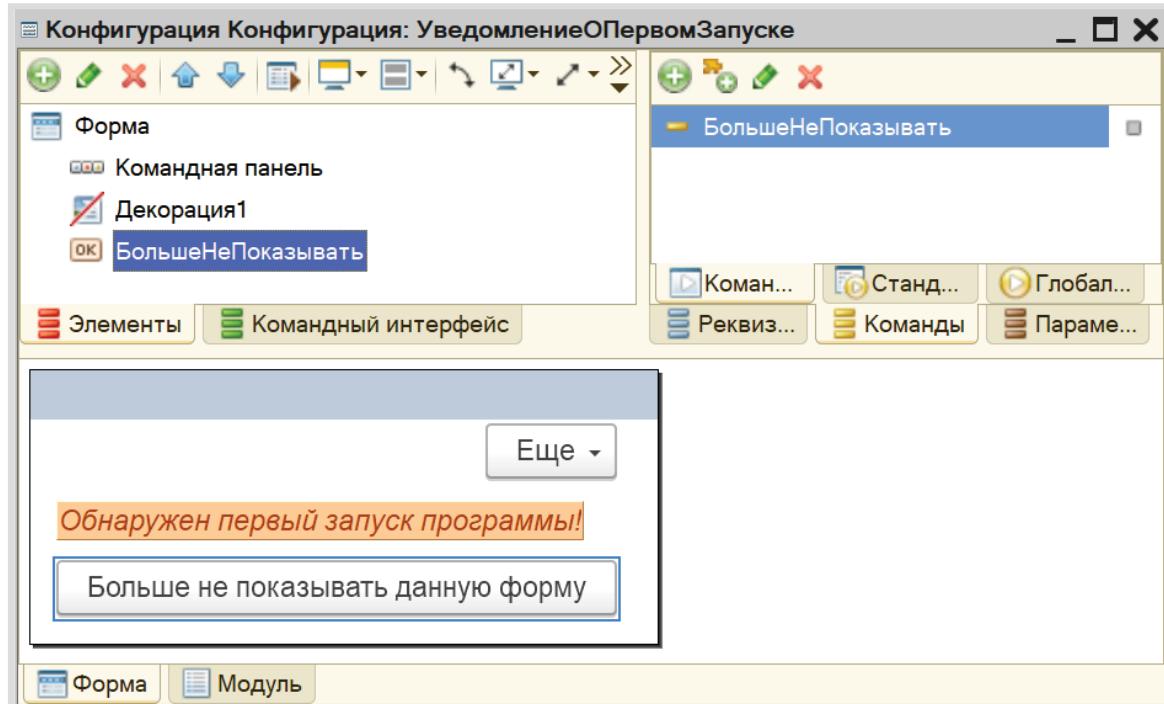
Данная форма будет выводиться пользователю, если константа содержит значение «Ложь». Но необходимо также добавить команду (кнопку формы), с помощью которой пользователь сможет указать системе, чтобы в дальнейшем эта форма не открывалась.



Перетащим команду на форму. Для этого следует «зажать» кнопку мыши на команде и перетащить ее на форму.



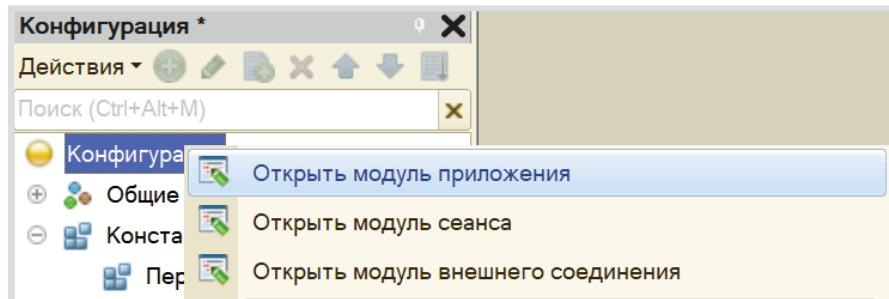
Команда автоматически встанет вслед за декорацией. Причем команда предстала перед нами как элемент формы «Кнопка».



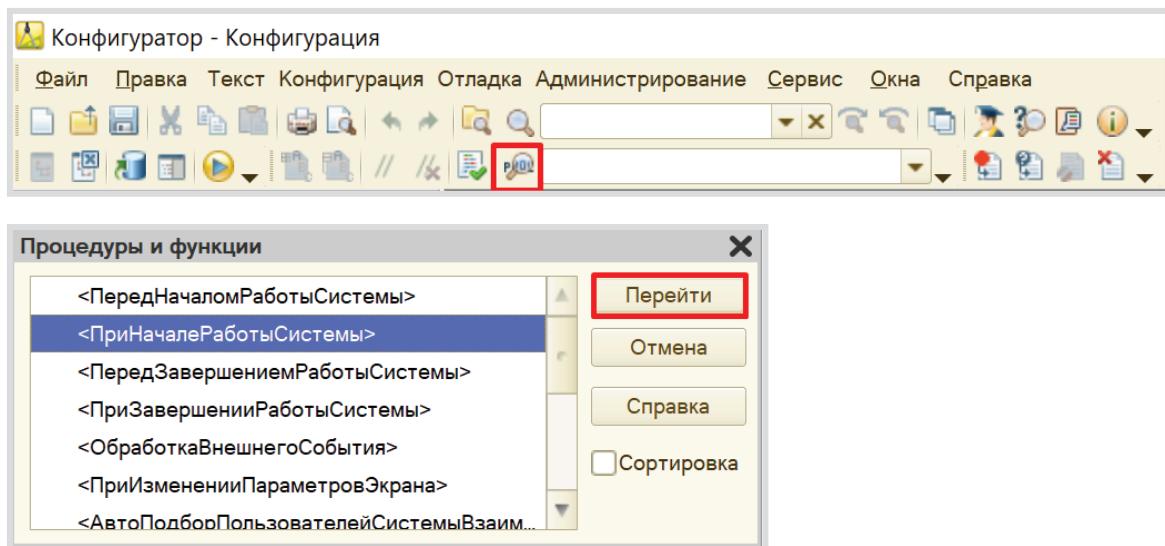
Если запустить систему в режиме «1С:Предприятие», то кнопка на форме будет, но она не будет совершать никаких действий, поскольку мы их еще не описали. Вернемся к этому чуть позже.

Вторая часть задачи заключается в том, чтобы «поймать» момент запуска системы, потому что именно в этот момент должна произойти проверка константы и открытие формы.

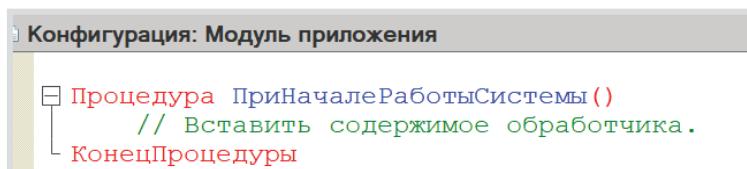
Алгоритмы, происходящие при запуске системы, описываются в модуле приложения. Найти этот модуль можно в контекстном меню корня конфигурации.



Открывается пустой модуль приложения. Для удобства разработчик может воспользоваться шаблонами некоторых процедур и функций. Откроем список процедур и выберем процедуру «*ПриНачалеРаботыСистемы*».



Система создала описание процедуры.



Наша задача состоит в том, чтобы прочитать значение константы и, если значение константы «Ложь», открыть форму.

Но получить данные константы, которая хранится в базе данных, напрямую из формы приложения мы никак не сможем. Это связано с тем, что модуль приложения выполняется на стороне клиента, а к базе данных получить доступ можно только со стороны сервера.



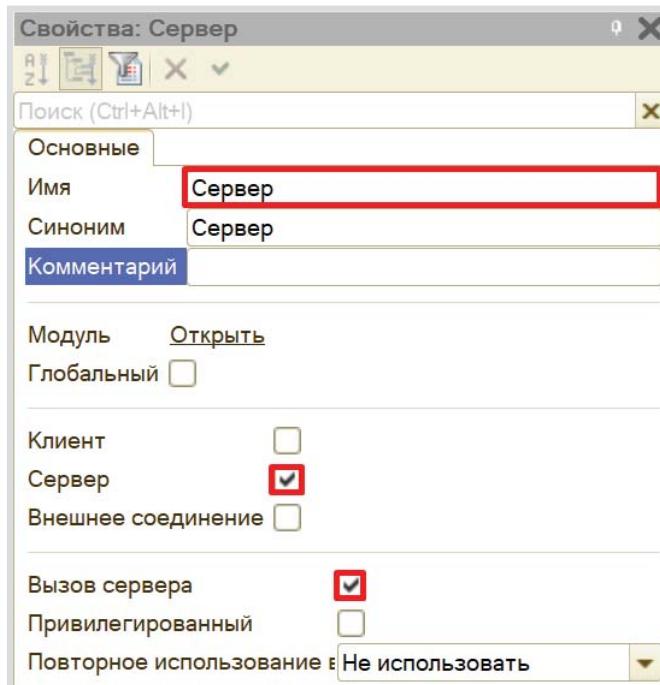
Чтобы разобраться, как устроено клиент-серверное взаимодействие, рассмотрим конкретный пример.

Предположим, пользователь хочет сформировать отчет.

1. Клиент нажимает на кнопку «Сформировать».
2. Запрос отправляется на Сервер 1С.
3. Сервер 1С отправляет запрос SQL-серверу.
4. SQL-сервер запрашивает данные из информационной базы и формирует таблицы.
5. Заполненные таблицы SQL-сервер отправляет Серверу 1С.
6. Сервер 1С рисует форму отчета и отправляет отчет клиенту в простом и понятном для него виде, например, таблицей или диаграммой.

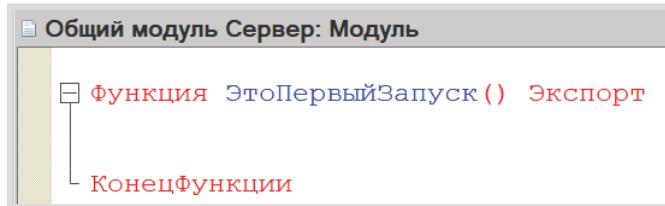
Таким образом, для того, чтобы получить любые данные из базы данных, нужно обратиться к серверу. Как это сделать?

Для получения контекста сервера необходимо создать общий модуль «Сервер». Обязательно нужно проставить галочки у свойств «Сервер» и «Вызов сервера».



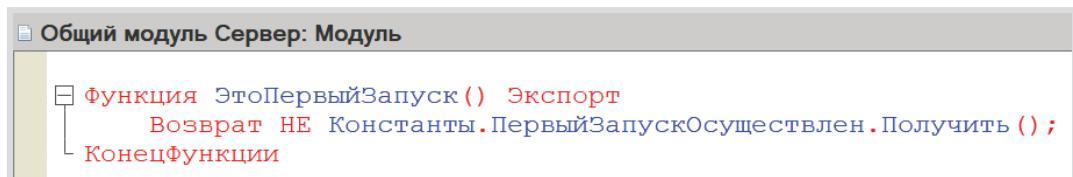
Включение свойства «Сервер» означает, что данный модуль будет выполняться на стороне сервера. Включение свойства «Вызов сервера» значит, что клиентские модули (например, созданная нами ранее форма) смогут обращаться к данному модулю и использовать его процедуры и функции.

Опишем в общем модуле функцию «ЭтоПервыйЗапуск».



Служебное слово «Экспорт», указанное после названия функции, делает ее видимой для вызова из других модулей системы.

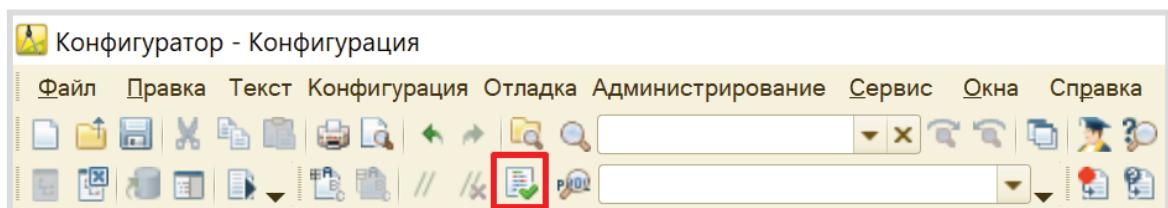
В самой функции опишем получение константы и возврат ее перевернутого значения. Если в константе хранится значение «Истина», то есть первый запуск уже был осуществлен, то функция вернет значение «Ложь», и наоборот.



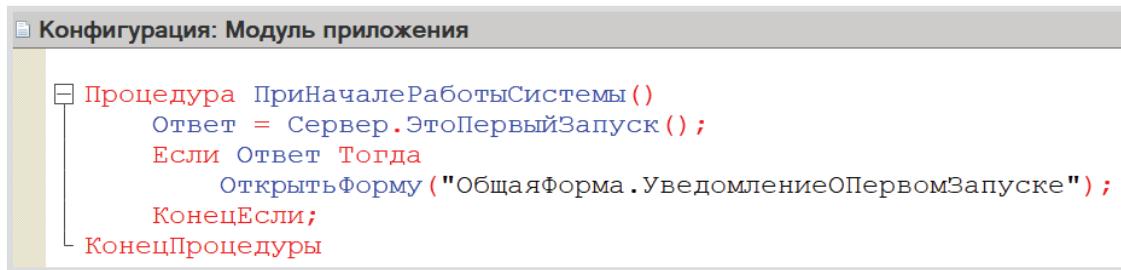
Внимание!

Обязательно проверьте модуль на наличие синтаксических ошибок.

Для этого нажмите на кнопку «Проверка модуля». Должно открыться окно «Служебные сообщения». Если синтаксических ошибок не обнаружено, то в данном окне появится надпись «Синтаксических ошибок не обнаружено».



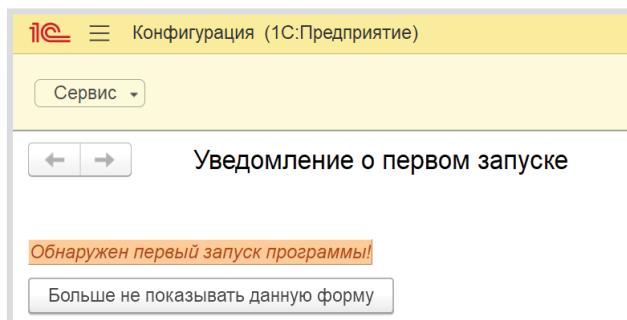
Поскольку общий модуль работает на стороне сервера, то он легко сможет получить значение константы. А служебное слово «Экспорт» позволит обратиться к этой функции откуда угодно. Значит, можно вызвать эту функцию из модуля приложения, где был подготовлен шаблон процедуры «ПриНачалеРаботыСистемы».



Переменной «Ответ» будет присвоено значение функции «ЭтоПервыйЗапуск», осуществляющейся на стороне сервера. Затем, если в переменной «Ответ» хранится значение «Истина», будет открыта созданная форма.

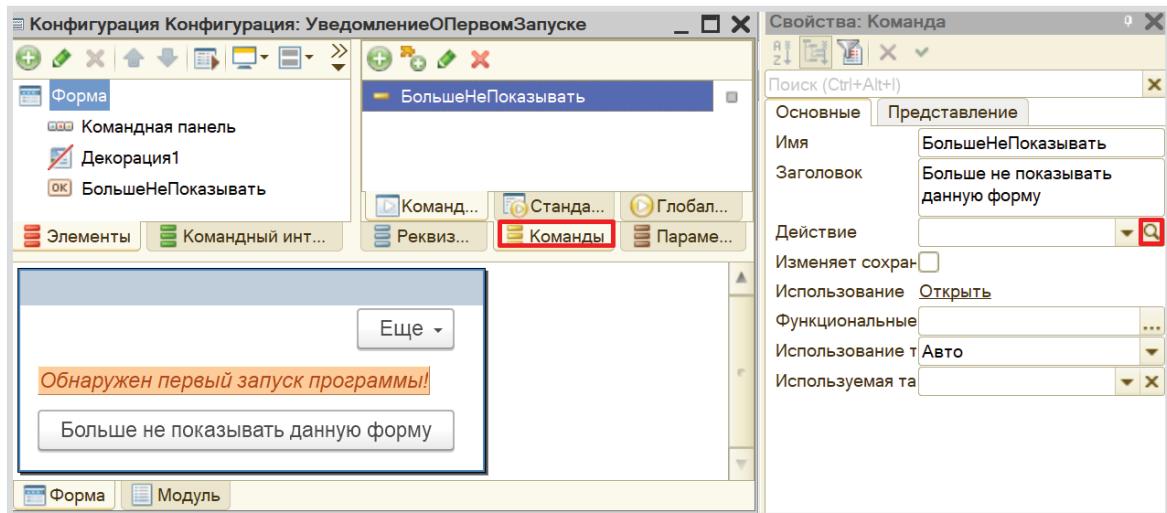
Не забудьте проверить модуль на наличие синтаксических ошибок.

Запустим систему в режиме «1С:Предприятие».



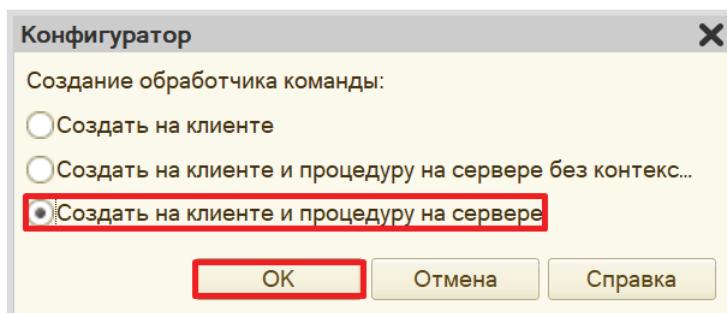
При запуске системы открылась созданная форма, уведомляющая пользователя о том, что был произведен первый запуск. Данная форма будет открываться при каждом запуске системы до тех пор, пока значение константы «ПервыйЗапускОсуществлен» не изменится на «Истина» (по умолчанию всегда «Ложь»).

Для того чтобы форма не открывалась повторно, опишем алгоритм работы кнопки, которая была добавлена на форму. Откроем редактор общей формы и в панели свойств команды создадим новое действие, которое будет выполнять данная команда.



Система задаст вопрос: «Где создать обработчик команды?».

Для того чтобы сделать правильный выбор, нужно разобраться, что мы хотим сделать? Мы хотим установить значение константы в значение «Истина». Для этого нужен доступ к базе данных, следовательно, нам необходим контекст сервера. Выберем третий вариант.



```

&НаСервере
Процедура БольшеНеПоказыватьНаСервере ()
    // Вставить содержимое обработчика.
КонецПроцедуры

&НаКлиенте
Процедура БольшеНеПоказывать (Команда)
    БольшеНоНееПоказыватьНаСервере ();
КонецПроцедуры

```

The screenshot shows the configuration interface with the title 'Конфигурация Конфигурация: УведомлениеОПервомЗапуске'. It displays generated VBA-like code for handling a command. The code defines two procedures: one for the server ('НаСервере') and one for the client ('НаКлиенте'). The server procedure contains a comment to insert the contents of the handler. The client procedure calls the server procedure and ends with a semicolon.

Система создала в модуле формы две процедуры: нижняя – обработчик события нажатия кнопки, верхняя – серверная процедура, которая вызывается из нижнего обработчика.

Опишем присвоение значения константе.

```

Конфигурация Конфигурация: УведомлениеОПервомЗапуске

&НаСервере
Процедура БольшеНеПоказыватьНаСервере ()
    Константы.ПервыйЗапускОсуществлен.Установить (Истина) ;
    КонецПроцедуры

&НаКлиенте
Процедура БольшеНеПоказывать (Команда)
    БольшеНеПоказыватьНаСервере () ;
    КонецПроцедуры

```

Осталось лишь закрыть форму. Форма всегда существует на клиенте, закрывать форму будем в контексте клиента.

```

Конфигурация Конфигурация: УведомлениеОПервомЗапуске

&НаСервере
Процедура БольшеНеПоказыватьНаСервере ()
    Константы.ПервыйЗапускОсуществлен.Установить (Истина) ;
    КонецПроцедуры

&НаКлиенте
Процедура БольшеНеПоказывать (Команда)
    БольшеНеПоказыватьНаСервере () ;
    ЭтаФорма.Закрыть () ;
    КонецПроцедуры

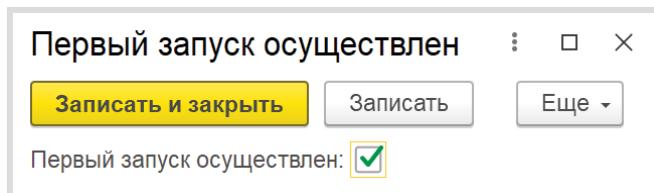
```

Проверьте модуль на наличие синтаксических ошибок.

Откроем систему в режиме «1С:Предприятие».

При запуске отрывается окно с уведомлением. Жмем на кнопку «Больше не показывать данную форму». Форма должна закрыться.

Проверим, изменилось ли значение константы.



Значение константы изменилось на «Истина».

При повторном запуске системы форма с уведомлением открываться не должна.

Поставленная задача решена.

Лабораторная работа № 17

РАЗРАБОТКА КОНФИГУРАЦИИ ДЛЯ УЧЕТА ТОВАРОВ. САМАЯ ПРОСТАЯ ЗАДАЧА

Сложность: **

Теги: справочник, документ, регистр накопления,
обработка проведения, запрос, схема компоновки
данных

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета товаров.

Многоскладской учет не ведется. Суммовой учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся.

В системе необходимо регистрировать закупку товара. При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили. Нужно предусмотреть учет до граммов.

В системе следует регистрировать продажу товара. При продаже товаров указывается, какие товары были проданы и в каком количестве.

Продать товар «в минус» нельзя, в момент продажи необходимо проверять остаток товара.

Нужно построить «Отчет» по остаткам товаров.

Форма отчета:

Остатки товаров на 31.01.2020

Товар	Остаток
Ложка	100.000
Вилка	148.000
Поварешка	2.000
Сахар	1.560

Отчет строится на конец дня, указанного пользователем. Особое внимание следует уделить последней секунде дня: документы, записанные на эту секунду, должны попадать в отчет.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

«Многоскладской учет не ведется. Суммовой учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся».

Из данной части условия можно сделать вывод, что никакой информации о складах, сумме, валютах, покупателях и поставщиках в информационной системе хранить не нужно.

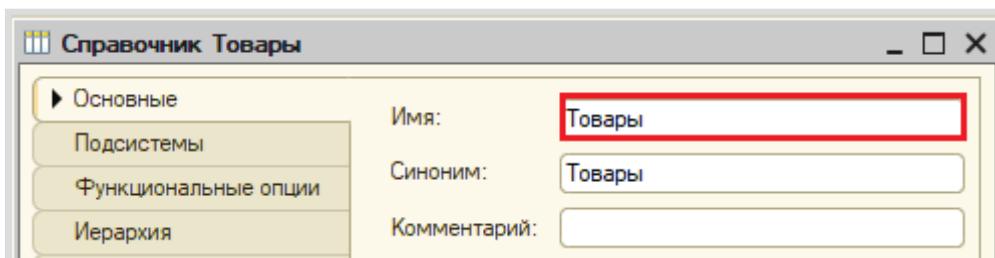
«Заказчик просит разработать конфигурацию для учета товаров».

Становится понятно, что в базе нужно хранить список товаров. Для этой задачи нам потребуется создать новый справочник.

Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Добавим новый справочник «Товары».



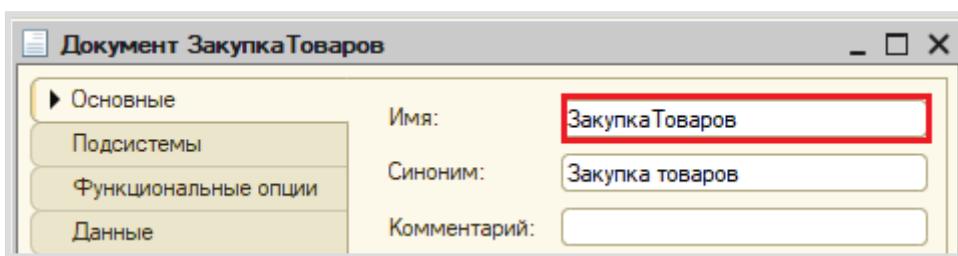
«В системе следует регистрировать поступление товара».

Для регистрации поступления товара в платформе «1С:Предприятие 8» существует специальный объект, который называется *документ*.

Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

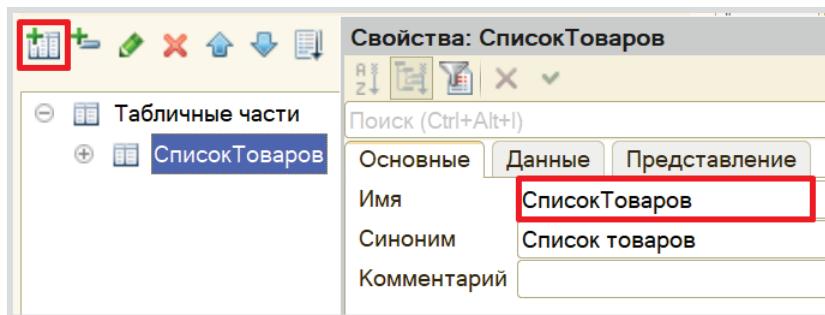
Создайте документ «ЗакупкаТоваров».



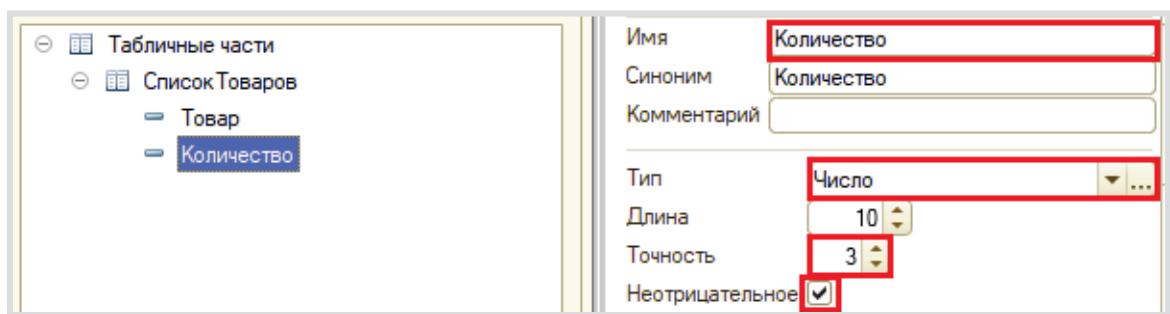
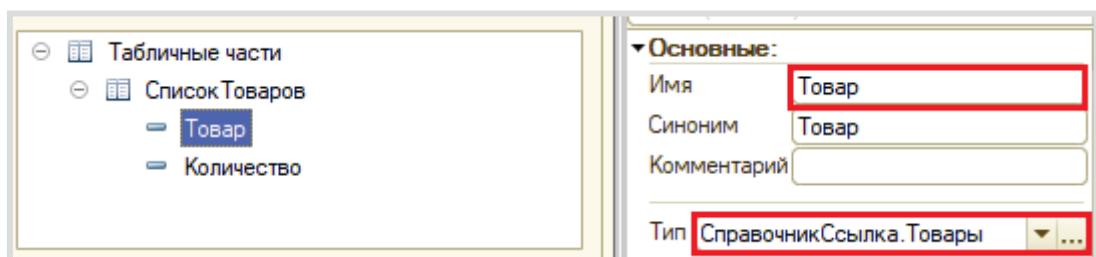
Для настройки структуры документа переходим на вкладку «Данные».

«При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили. Нужно предусмотреть учет до граммов».

Добавим табличную часть «СписокТоваров».



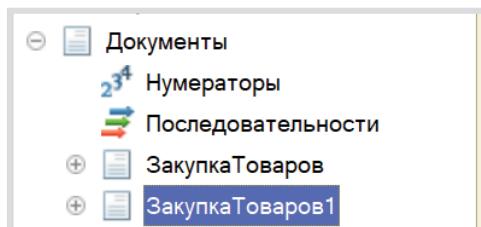
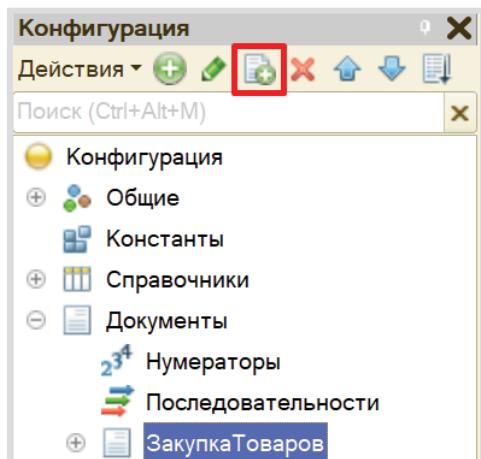
Далее добавим два реквизита табличной части (колонки таблицы): «Товар» (тип – СправочникСсылка.Товары) и «Количество» (тип – «Число»).



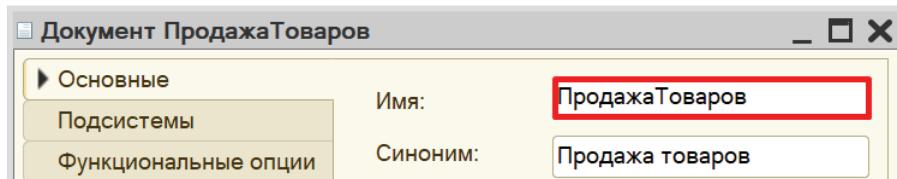
«В системе следует регистрировать продажу товара. При продаже товаров указывается, какие товары были проданы и в каком количестве».

Продажа товара осуществляется аналогично закупке, следовательно, документ по структуре будет точно таким же, как документ «ЗакупкаТовара».

Чтобы не тратить время на создание точно такого же документа, воспользуемся возможностью платформы создать новый документ копированием.



Получаем точную копию документа. Изменим имя документа на «ПродажаТоваров».



На вкладке «Данные» структура должна быть аналогична структуре документа «ПолучениеТовара»: иметь табличную часть с реквизитами «Товар» и «Количество».

Можно ли теперь на основе таких документов построить отчет по остаткам товаров? Можно, но для этого придется прибегнуть к грубому перебору всех существующих документов. Данный вариант является неправильным, потому что, если таких документов окажется очень много, система будет требовать большого количества ресурсов и времени.

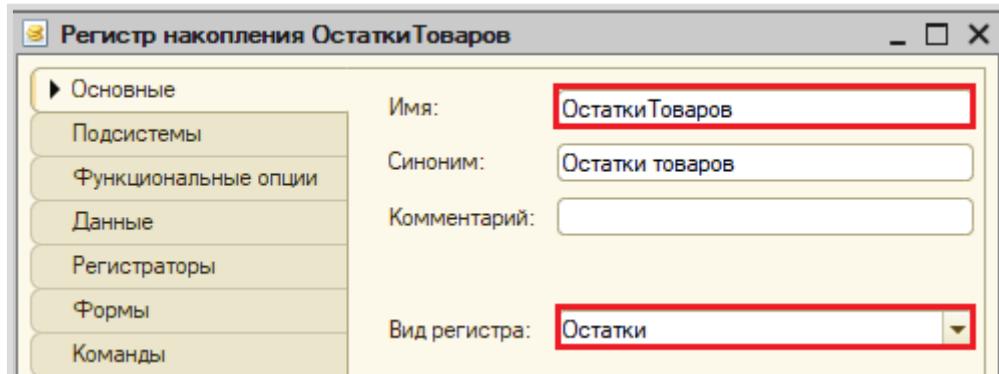
Для решения данной проблемы и ускорения процесса извлечения данных создадим еще один объект – *регистр накопления*.

Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Создайте регистр накопления «ОстаткиТоваров» вида «Остатки».

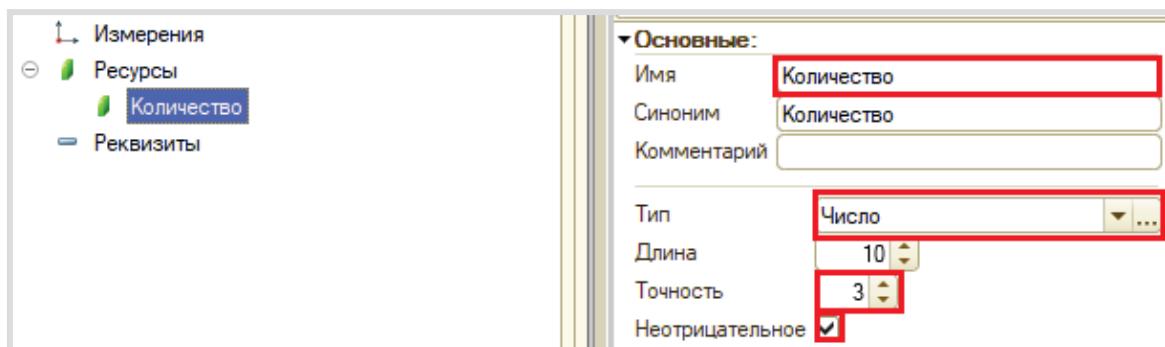
Вид регистра «Остатки» позволяет настроить данный регистр таким образом, что какие-то объекты будут вносить в него данные, а какие-то, наоборот, вычитать. Таким образом и получается хранение остатков.



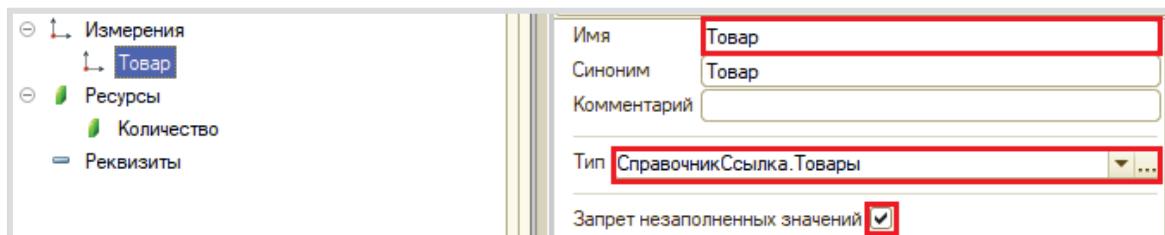
Для формирования структуры регистра переходим на вкладку «Данные».

Структура *регистра накопления* отличается от структуры документа.

Заполнение данного окна проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, нужно задать вопрос: «Что мы хотим накапливать/считывать в данном регистре?». Мы хотим считать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число».



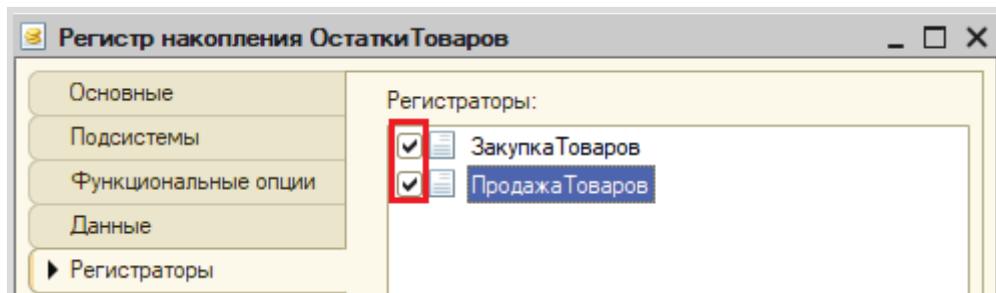
Чтобы разобраться с измерением, необходимо понять, в разрезе чего мы хотим считать количество. Мы хотим считать количество (чего?) товаров. Значит, в качестве измерения следует добавить реквизиты «Товар» (тип – «СправочникСсылка.Товары»).



Чтобы регистр накопления заработал, нужно сделать следующее:

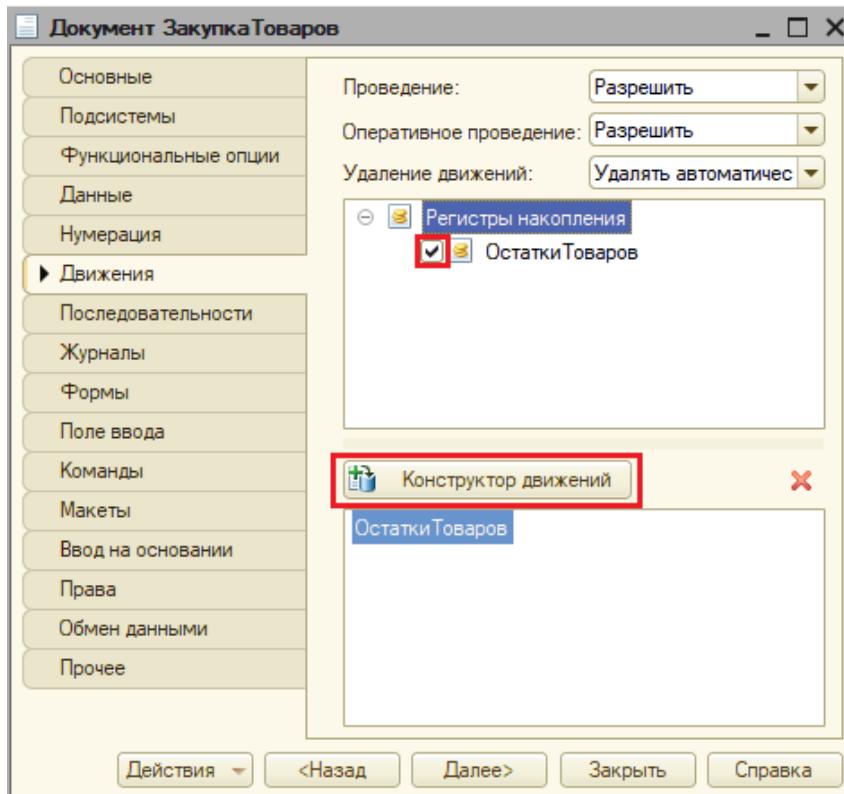
1. Определить источники данных, которые должны попадать в регистр (определить документы-регистраторы).
2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.

Укажем документы, из которых будут формироваться движения в созданный *регистр накопления*. Для этого перейдем на вкладку «Регистраторы» и отметим галочкой созданные нами документы.



Далее для каждого документа-регистратора требуется сформировать движения в созданный *регистр накопления*.

Откройте окно редактирования объекта «ЗакупкаТоваров» на вкладке «Движения». Убедитесь, что галочка у регистра активна и воспользуйтесь конструктором движений.

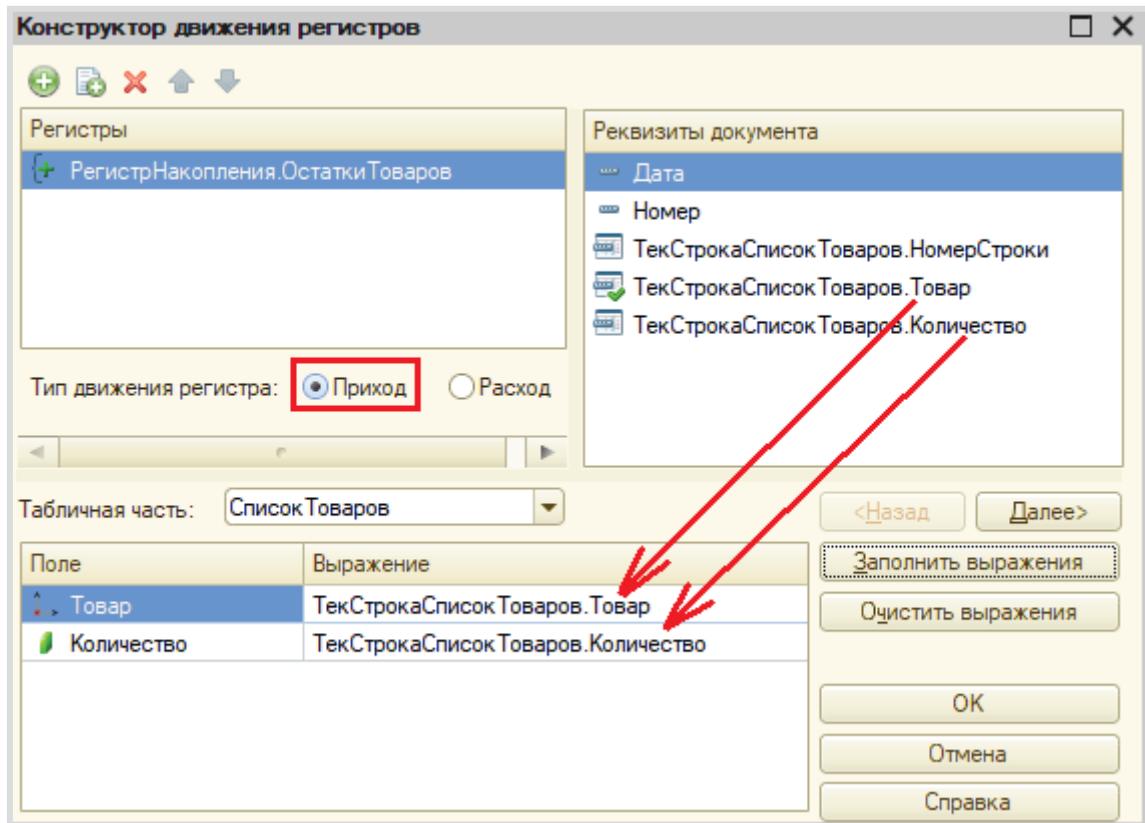


Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части, нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты *регистра накопления*. Нужно заполнить поле «Выражение» реквизитами документа.

Поскольку закупка товара должна увеличивать количество товаров на складе, то тип движения регистра необходимо выбрать «Приход». Регистр будет обозначаться знаком «+» (плюс).

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.

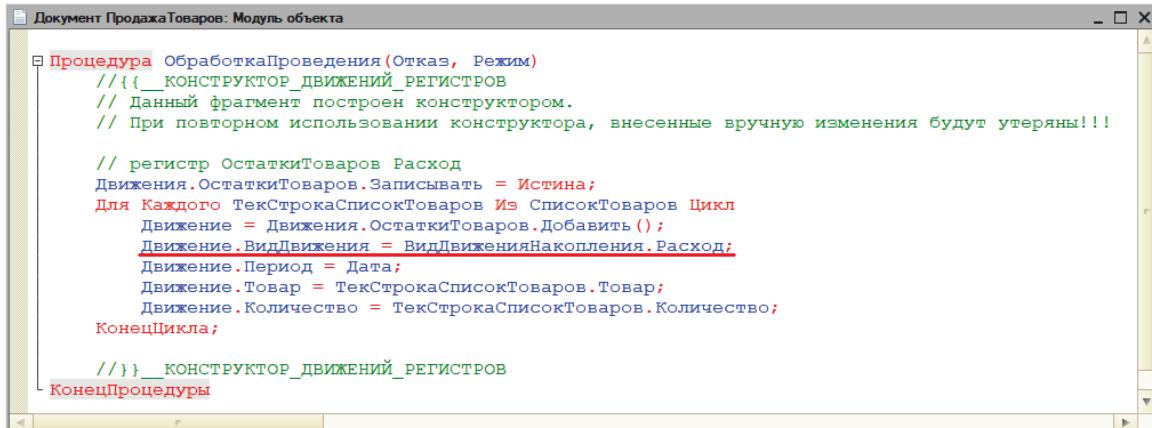


При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

```
Документ ЗакупкаТоваров: Модуль объекта
■ Процедура ОбработкаПроведения(Отказ, Режим)
    //({__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    // Данный фрагмент построен конструктором.
    // При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
    //
    // регистр ОстаткиТоваров Приход
    Движение.ОстаткиТоваров.Записывать = Истина;
    Для Каждого ТекСтроКаСписокТоваров Из СписокТоваров Цикл
        Движение = Движения.ОстаткиТоваров.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
        Движение.Период = Дата;
        Движение.Товар = ТекСтроКаСписокТоваров.Товар;
        Движение.Количество = ТекСтроКаСписокТоваров.Количество;
    КонецЦикла;
    //}
    __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

По аналогии сформируйте движения для второго документа «ПродажаТоваров», изменив вид движения на «Расход».

В результате в модуле объекта документа «ПродажаТоваров» вы получите аналогичный программный код, который отличается только видом движения.



```

Документ ПродажаТоваров: Модуль объекта

Процедура ОбработкаПроведения(Отказ, Режим)
    //__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
    // данный фрагмент построен конструктором.
    // При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!
    // регистр ОстаткиТоваров Расход
    Движение.ОстаткиТоваров.Записывать = Истина;
    Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
        Движение = Движения.ОстаткиТоваров.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Товар = ТекСтрокаСписокТоваров.Товар;
        Движение.Количество = ТекСтрокаСписокТоваров.Количество;
    КонецЦикла;
    //}
    КонецПроцедуры

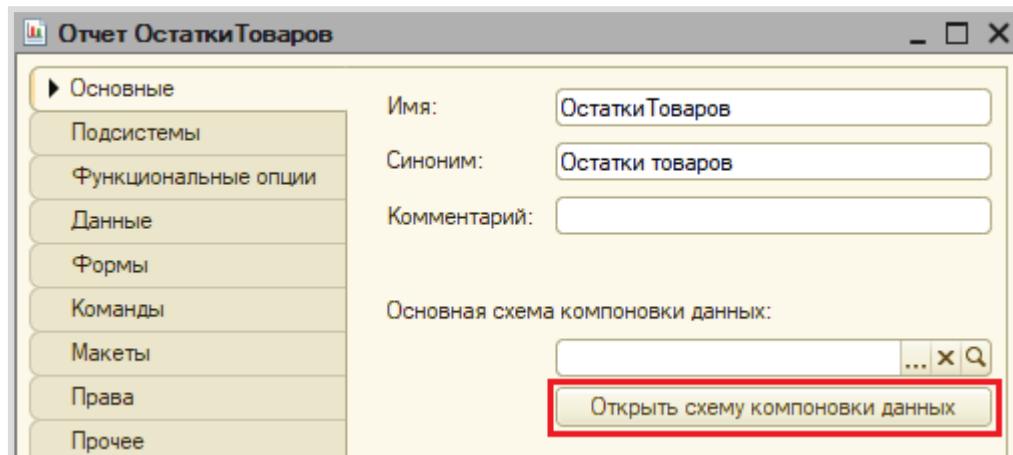
```

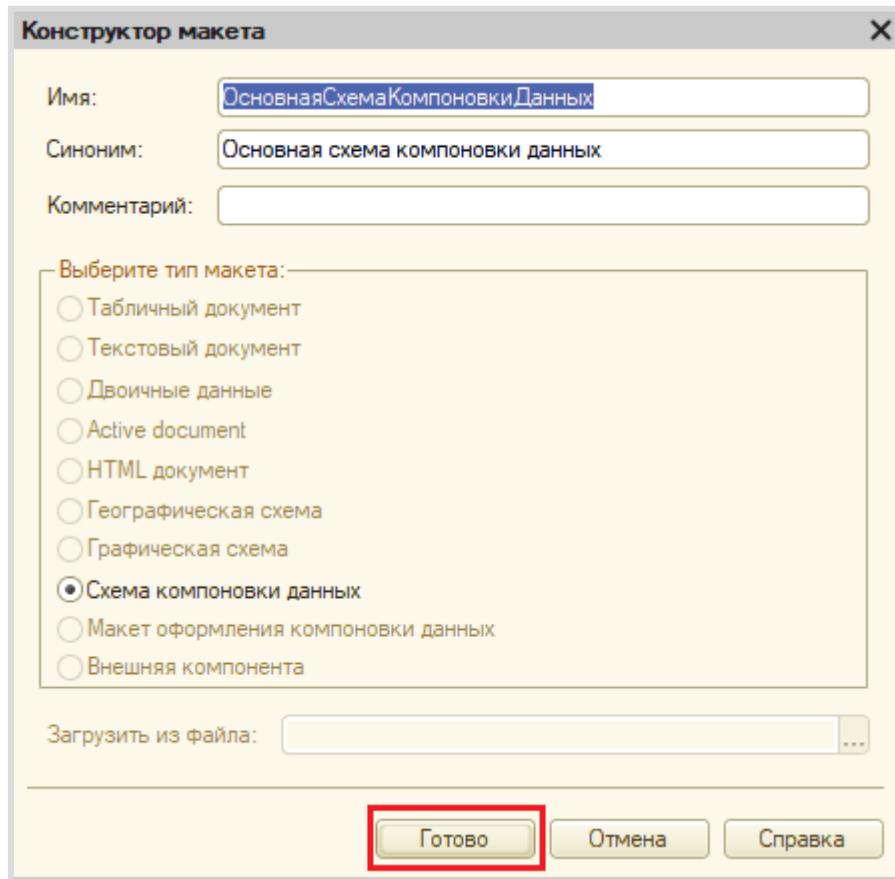
Для визуализации работы *регистра накопления* нужно создать отчет.

Определение

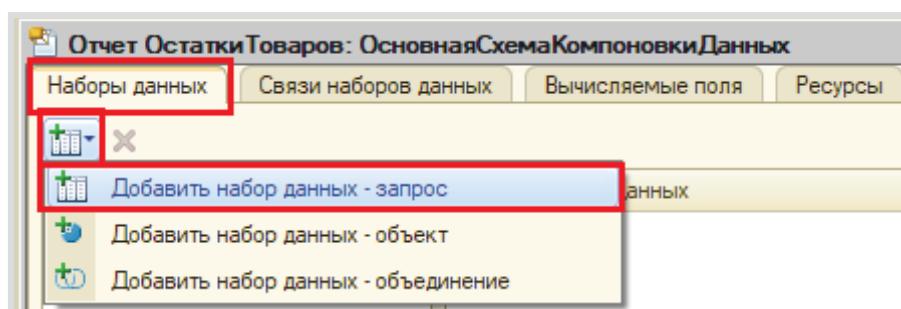
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Создадим новый отчет «ОстаткиТоваров». Для наполнения отчета воспользуемся конструктором схемы компоновки данных.

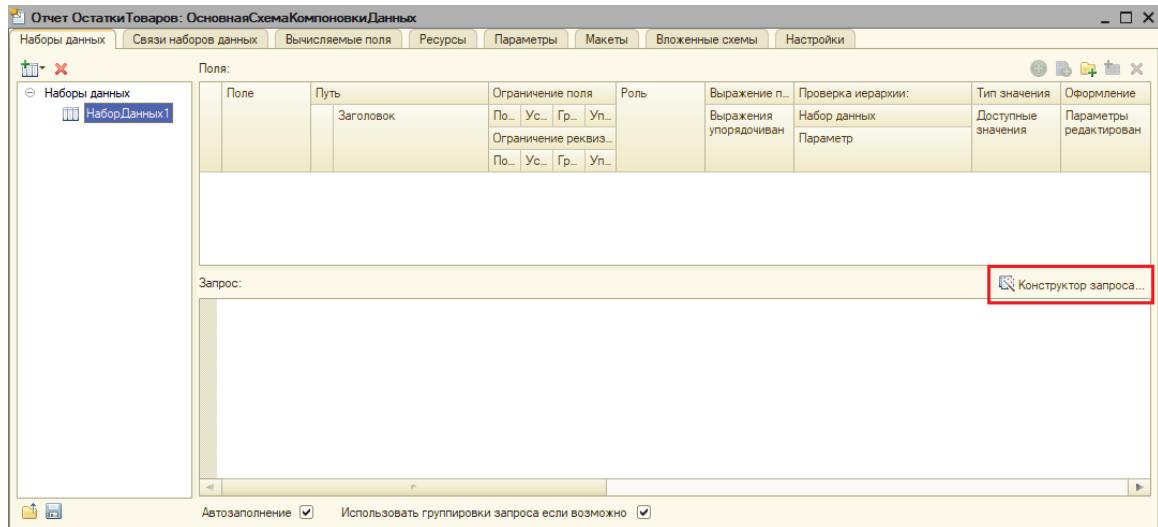




Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



При построении запросов используется собственный язык запросов 1С. Их можно писать вручную или воспользоваться *конструктором запросов*.

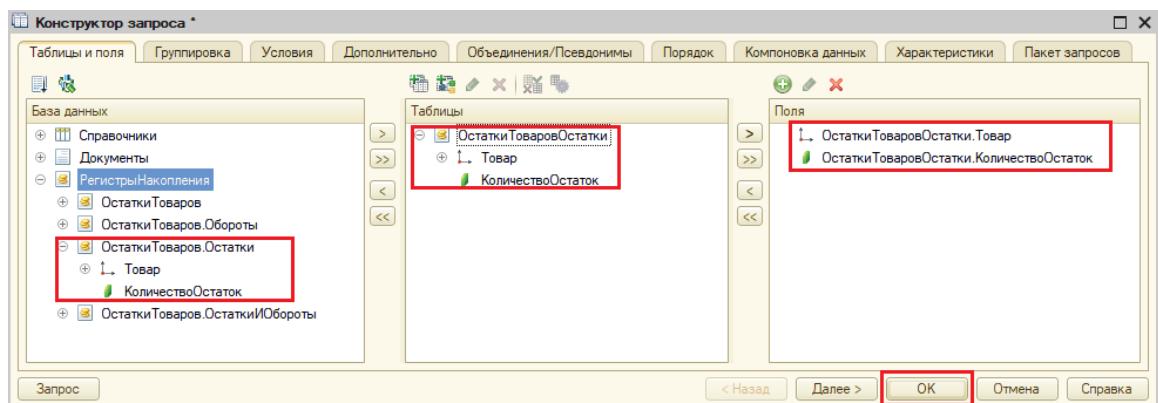


Открывается *конструктор запроса*. Эта вкладка имеет три части:

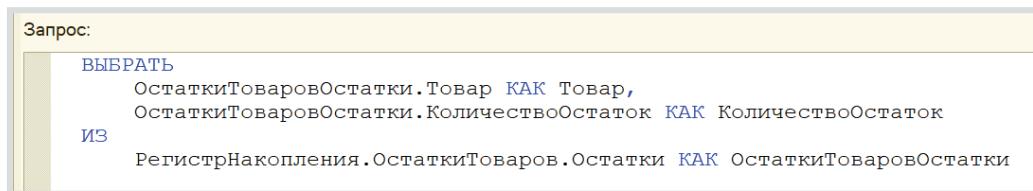
- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица позволит получить уже просуммированные значения по всем документам.

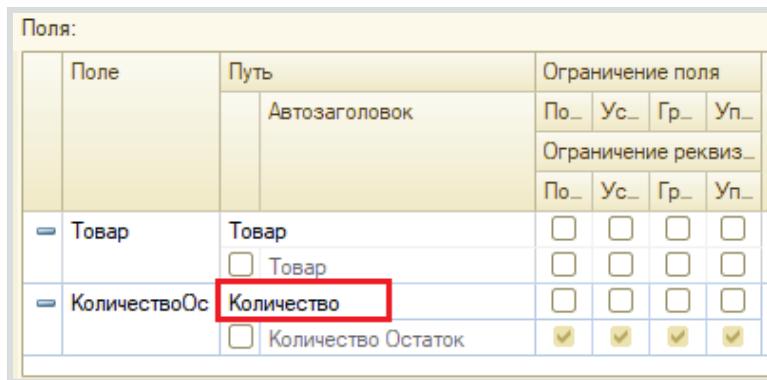
Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.



По завершении работы с конструктором нажмите на кнопку «OK». Система сформирует следующий запрос:

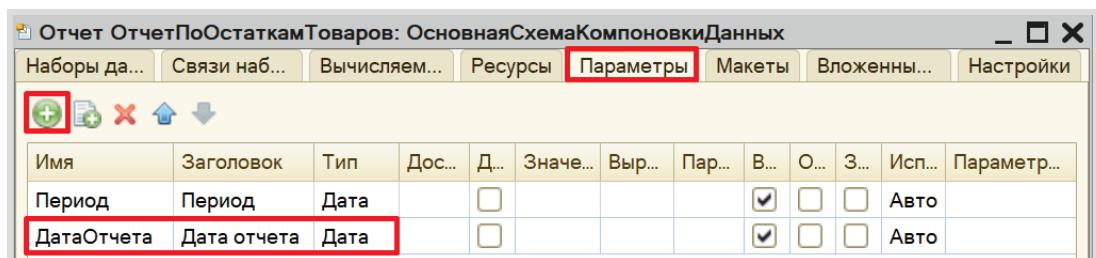


Измените заголовок у поля «КоличествоОстаток» на «Количество».



«Отчет строится на конец дня, указанного пользователем. Особое внимание следует уделить последней секунде дня: документы, записанные на эту секунду, должны попадать в отчет».

Из условия следует, что отчет должен включать документы, записанные на последнюю секунду дня. При использовании стандартных методов такие документы в отчет попадать не будут. Поэтому нужно добавить новый параметр «ДатаОтчета» на соответствующей вкладке.



Чтобы у пользователя была возможность выбирать только даты, без указания секунд, нужно настроить формат редактирования параметра «ДатаОтчета».

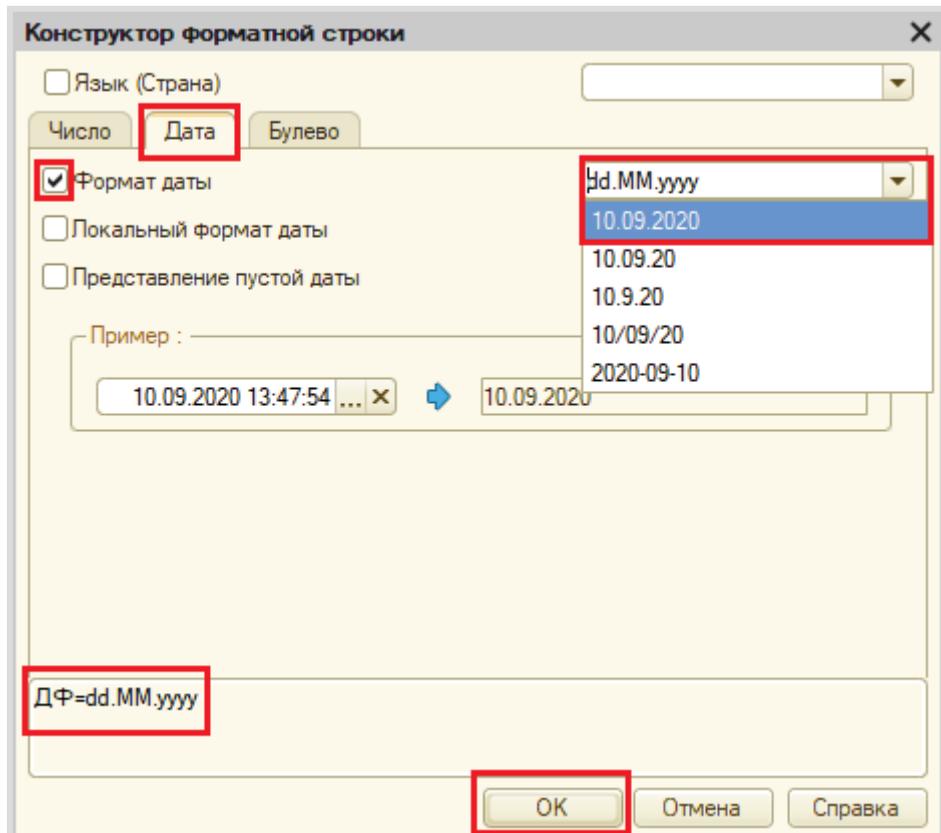
Отчет ОтчетПоОстаткамТоваров: ОсновнаяСхемаКомпоновкиДанных

Наборы данных	Связи наборов данных	Вычисляемые поля	Ресурсы	Параметры	Макеты	Вложенные схемы	Настройки
<input type="button" value="+"/> <input type="button" value="X"/> <input type="button" value="Up"/> <input type="button" value="Down"/>							
Имя	Заголовок	Тип	Д...	Д...	З...	Выражение	П... В... О... З... И... Параметры редактирования
Период	Период	Дата	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ДОБАВИТЬКДАТЕ(КОНЕЦПЕРИОДА(&ДатаОтчета, "ДЕНЬ"), "СЕКУНДА", 1)	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> А...
ДатаОтчета	Дата отчета	Дата	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input checked="" type="checkbox"/> <input type="checkbox"/> А... <input type="checkbox"/> Формат редактирования

Параметры редактирования

Параметр	Значение
<input type="checkbox"/> Маска	
<input type="checkbox"/> Связи параметров выбора	
<input type="checkbox"/> Параметры выбора	
<input type="checkbox"/> Связь по типу	
<input type="checkbox"/> Форма выбора	
<input checked="" type="checkbox"/> Формат редактирования	<input type="button" value="..."/>
<input type="checkbox"/> Быстрый выбор	Ложь
<input type="checkbox"/> Выбор групп и элементов	Группы и элементы

OK Отмена Справка



После нажатия кнопки «OK» нужно настроить стандартный параметр «Период» для корректного учета последней секунды дня:

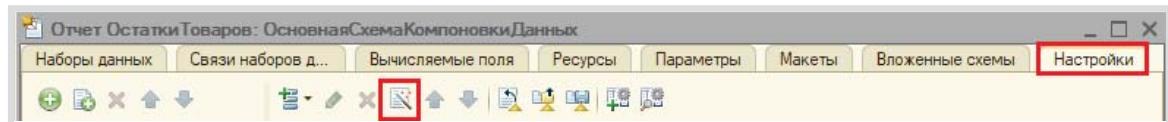
- во-первых, этот параметр должен быть недоступен пользователю, так как носит вычислительный характер;
- во-вторых, для корректного расчета требуется написать выражение для стандартного параметра «Период»:

ДОБАВИТЬКДАТЕ(КОНЕЦПЕРИОДА(&ДатаОтчета, "ДЕНЬ"),"СЕКУНДА",1)

Сначала берется последний момент дня, указанного в параметре «ДатаОтчета», а затем прибавляется еще одна секунда, чтобы учитывались даже документы, проведенные за эту последнюю секунду.

Отчет ОтчетПоОстаткамТоваров: ОсновнаяСхемаКомпоновкиДанных												
Наборы данных		Связи наборов данных		Вычисляемые поля		Ресурсы		Параметры		Макеты	Вложенные схемы	
Имя	Заголовок	Тип	Д...	Д...	З...	Выражение	П...	В...	О...	З...	И...	Параметры редактирования
Период	Период	Дата	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	ДобавитьКДАТЕ(КОНЕЦПЕРИОДА(&ДатаОтчета, "ДЕНЬ"),"СЕКУНДА",1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	A...
ДатаОтчета	Дата отчета	Дата	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Формат редактирования

Последняя подзадача, связанная с отчетом, – настроить его внешний вид. Переходим на вкладку «Настройки» и воспользуемся конструктором настроек отчета.



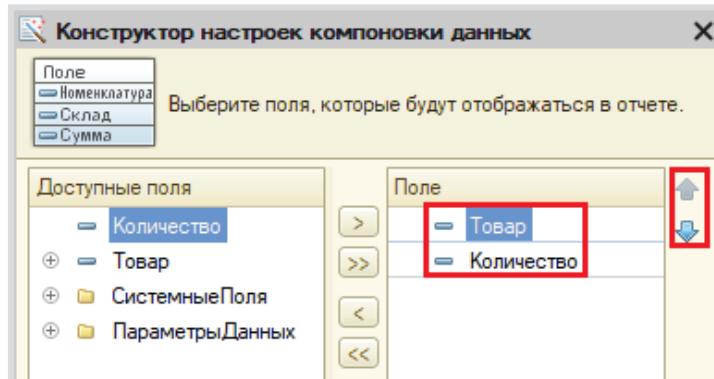
The screenshot shows the 'Report Properties' window with the 'Data Layout' tab selected. A red box highlights the 'Настройки' (Properties) button in the toolbar.

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.

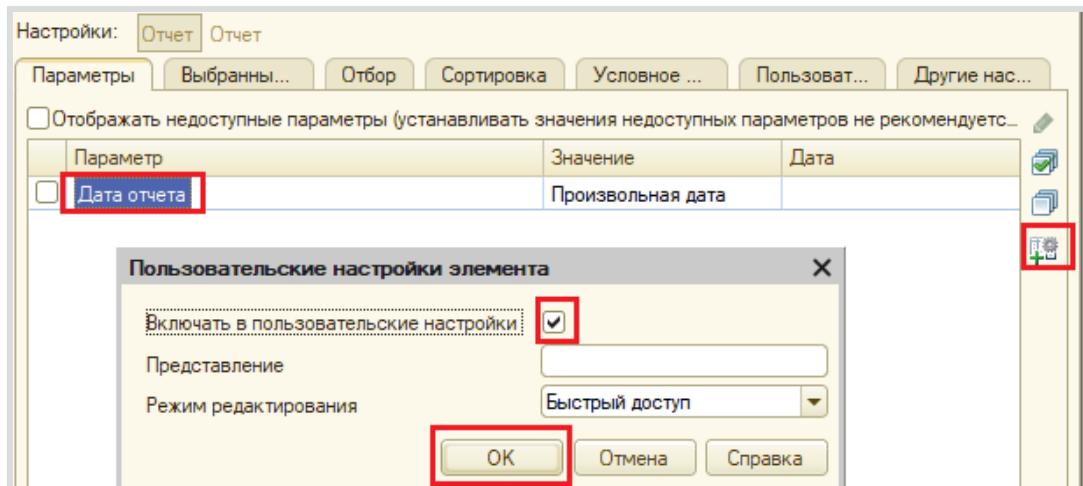
Кнопки на панели инструментов: < Назад, Далее >, OK, Отмена, Справка. The 'Далее >' button is highlighted with a red box.



The screenshot shows the 'Select Fields' dialog. On the left, under 'Available Fields', there are categories: Поле (Field), Номенклатура (Inventory), Склад (Warehouse), and Сумма (Sum). Under 'Available Fields' itself, there are items: Количество (Quantity), Товар (Item), Системные Поля (System Fields), and Параметры Данных (Data Parameters). On the right, under 'Selected Fields', there are items: Товар (Item) and Количество (Quantity). Arrows between the two lists indicate the selection process.

После выбора нужных полей нажмите на кнопку «OK».

Чтобы у пользователя была возможность выбирать требуемый день, вам нужно включить параметр «ДатаОтчета» в пользовательские настройки. Это можно сделать в нижней части вкладки «Настройки».



Проверьте работоспособность системы, заполнив тестовыми данными справочник, документы «ЗакупкаТоваров» и «ПродажаТоваров», а также построив отчет.

Чтобы в дальнейшем убедиться в корректности вывода информации о документах, проведенных в конце дня, создавайте документы за предыдущие числа. Это необходимо, поскольку созданные вами документы имеют оперативный режим проведения.

Для наглядности постройте отчет до проведения документа «ПродажаТоваров» и после.

Товар	Количество
Арбузы	23,456
Яблоки	10,500

Товар	Количество
Арбузы	-0,544
Яблоки	4,850

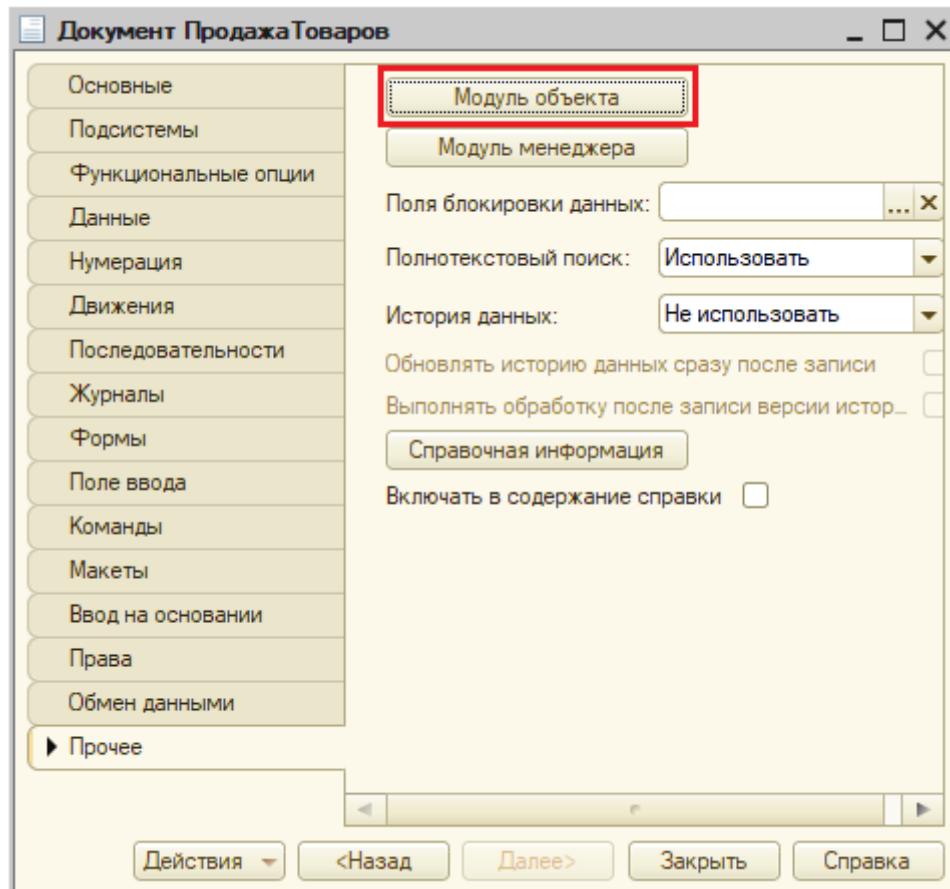
Если в документе «Продажа товаров» указать количество больше, чем имеется на складе, то будут получены *отрицательные остатки*.

Контроль отрицательных остатков является заключительным и самым важным этапом выполнения данной работы.

Проверять остатки товаров будем следующим образом:

1. Сделаем движение данных из документа в *регистр накопления*.
2. Проверим, появились ли в регистре остатки, значение которых меньше нуля (то есть отрицательные).
3. Если есть отрицательные остатки, то отменим сделанное движение в *регистр накопления* и выведем пользователю сообщение об ошибке.

Контроль отрицательных остатков должен происходить в момент проведения документа «ПродажаТоваров». Откроем модуль объекта данного документа.



Чтобы сделать движение данных из документа в *регистр накопления*, допишем после окончания цикла строку «Движения.Записать();». Метод записывает только те движения документа, у которых установлен флаг «Записывать», при этом флаг в итоге снимается, что не приводит к повторной записи движений по окончании транзакции проведения.

И главное, «Движения.Записать();» всегда записывают движения в том порядке, в котором таблицы указаны в дереве метаданных, что на порядок уменьшает шансы взаимных блокировок, ведь все транзакции в одинаковом порядке блокируют таблицы.

```

Процедура ОбработкаПроведения (Отказ, Режим)
  // регистр ОстаткиТоваров Расход
  Движения.ОстаткиТоваров.Записывать = Истина;
  Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваров.Добавить ();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
  КонецЦикла;

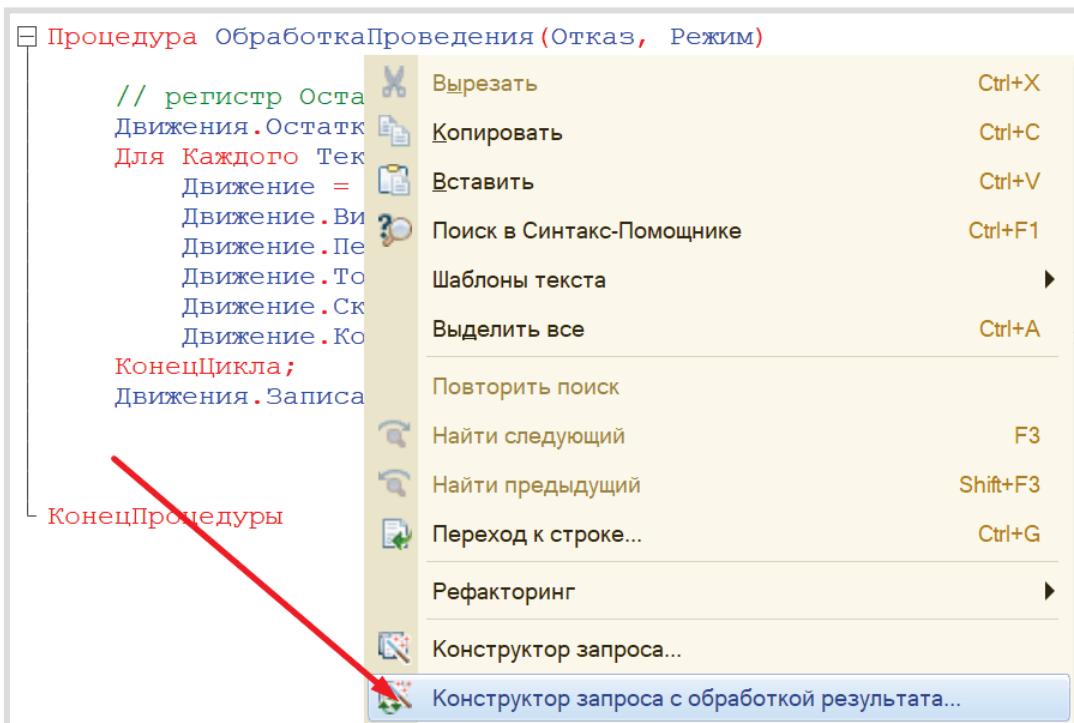
  Движения.Записать ();

КонецПроцедуры

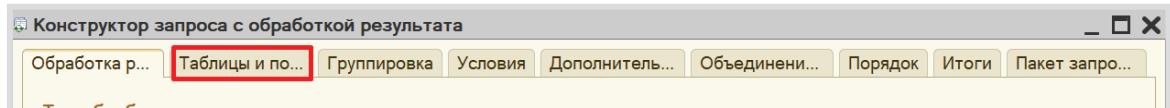
```

Теперь, когда движение было сделано, можно обратиться к данным *регистра накопления*.

Чтобы это сделать, воспользуемся *конструктором запроса с обработкой результата*. Этот конструктор можно открыть из контекстного меню, открывающегося щелчком правой кнопки мыши по области модуля. Данный конструктор обязательно должен быть вызван внутри процедуры «ОбработкаПроведения».



Соглашаемся с созданием нового запроса. Открывается окно *конструктора запроса с обработкой результата*. Переходим на вкладку «Таблицы и поля».



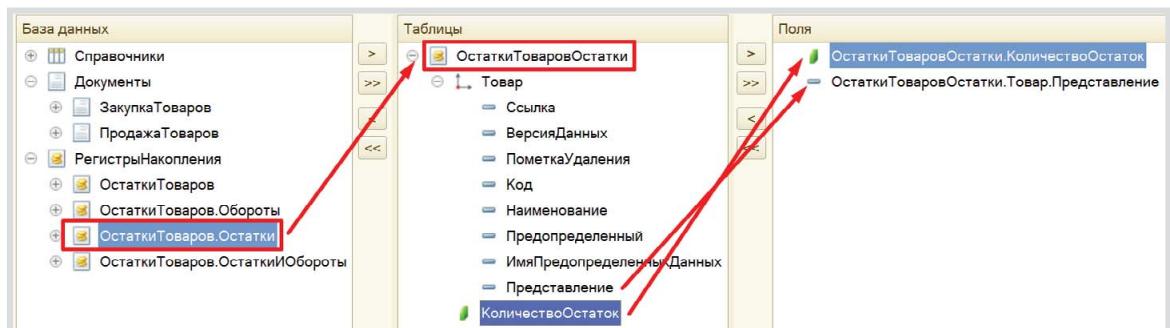
Открывшееся окно имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного запроса.
- Справа поля – это те значения (поля), которые мы хотим получить.

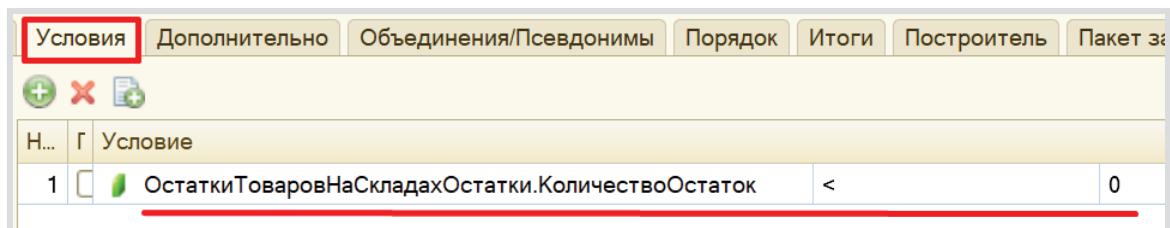
Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица способна обработать основную таблицу и самостоятельно посчитать остатки товаров.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



Переходим на вкладку «Условия» и добавим новое условие. Пусть в запрос попадут только данные с отрицательными остатками.

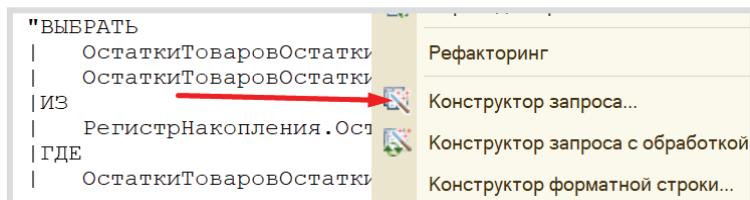


По окончании работы с данным конструктором нажимаем на кнопку «OK». Конструктор выдаст предупреждение об ошибке, которое нужно проигнорировать. Для корректной работы запроса следует удалить знак амперсанта (&) перед нулем в условии. Запрос должен выглядеть следующим образом:

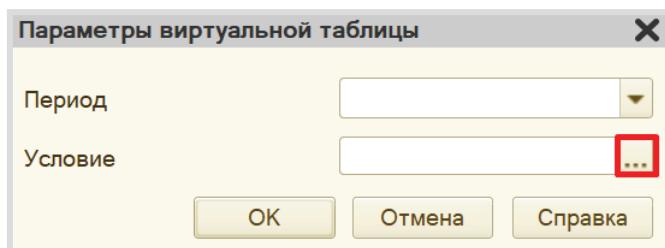
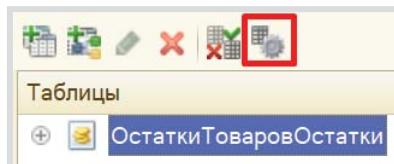
```
"ВЫБРАТЬ
| ОстаткиТоваровОстатки.КоличествоОстаток КАК КоличествоОстаток,
| ОстаткиТоваровОстатки.Товар.Представление КАК ТоварПредставление
| ИЗ
| РегистрНакопления.ОстаткиТоваров.Остатки КАК ОстаткиТоваровОстатки
| ГДЕ
| ОстаткиТоваровОстатки.КоличествоОстаток < 0"
```

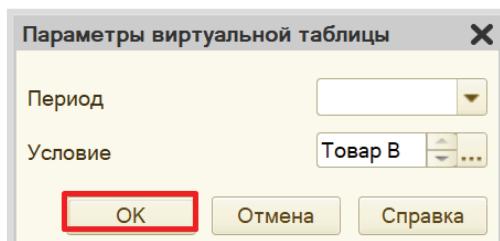
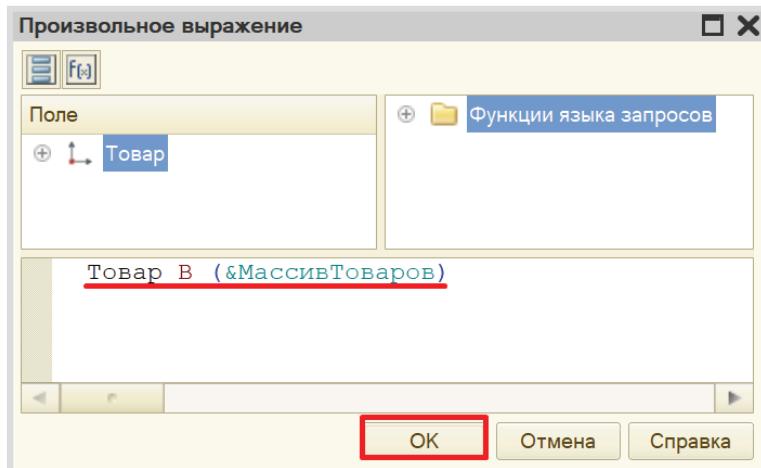
С помощью данного запроса из базы данных можно получить отрицательные остатки по всем товарам. Но нам нет необходимости получать такую большую выборку, нужно сузить запрос до перечня товаров, перечисленных в табличной части.

Откроем *конструктор запроса*. Для этого следует щелкнуть в любом месте самого запроса (черный текст в двойных кавычках) правой кнопкой мыши и вызвать *конструктор запроса*.



Далее нужно наложить условия на виртуальную таблицу *регистра накопления*.





Данное условие поможет ограничить запрос по тем товарам, которые находятся в табличной части документа.

Нажимаем на кнопку «OK». Текст запроса изменился:

```
| ГДЕ
|   ОстаткиТоваров.Остатки.Количество < 0" ;
Запрос.УстановитьПараметр ("Дата", Новый Граница ( МоментВремени (), ВидГраницы.Включая )) ;
Запрос.УстановитьПараметр ("МассивТоваров", СписокТоваров.ВыгрузитьКолонку ("Товар")) ;
```

Мы добавили параметры. Теперь запрос будет проводить поиск только по конкретному списку товаров. Осталось лишь указать эти товары сразу после текста запроса. Так же необходимо добавить параметр «Дата».

```
Запрос.УстановитьПараметр ("Дата", Новый Граница ( МоментВремени (), ВидГраницы.Включая )) ;
Запрос.УстановитьПараметр ("МассивТоваров", СписокТоваров.ВыгрузитьКолонку ("Товар")) ;
```

Ну, и последний шаг – выдать сообщение пользователю, если запрос вернул отрицательные остатки. В первую очередь, добавим блок условия:

```
РезультатЗапроса = Запрос.Выполнить ();
Если НЕ РезультатЗапроса.Пустой () Тогда
    КонецЕсли;
```

Внутрь цикла можно попасть только в том случае, если запрос пришел не пустой, то есть если были найдены отрицательные остатки. В таком случае нужно отменить проведение документа и выдать пользователю сообщение:

```

Если НЕ РезультатЗапроса.Пустой() Тогда
    Отказ = Истина;
    ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        Сообщить ("Не хватает товара " + ВыборкаДетальныеЗаписи.ТоварПредставление +
                    ", в количестве " + (- ВыборкаДетальныеЗаписи.КоличествоОстаток));
    КонецЦикла;

КонецЕсли;

```

Код процедуры полностью должен выглядеть следующим образом:

```

Процедура ОбработкаПроведения(Отказ, Режим)

// регистр ОстаткиТоваров Расход
Движения.ОстаткиТоваров.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваров.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;
Движения.Записать();

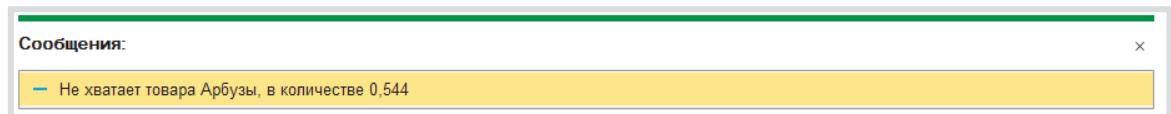
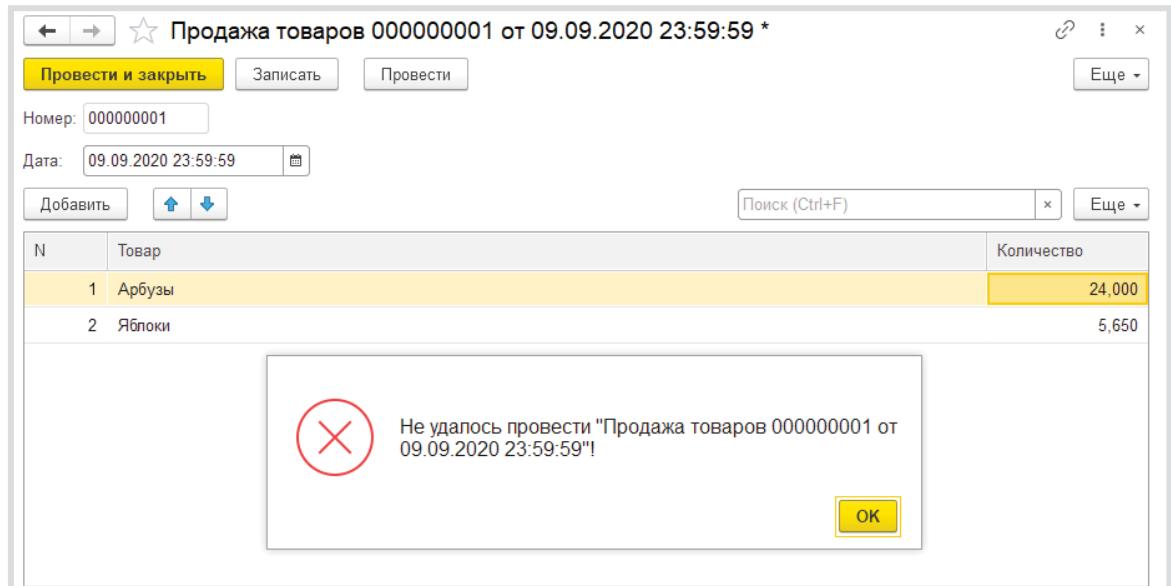
Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
| ОстаткиТоваровОстатки.Товар.Представление КАК ТоварПредставление,
| ОстаткиТоваровОстатки.КоличествоОстаток КАК КоличествоОстаток
|ИЗ
| РегистрНакопления.ОстаткиТоваров.Остатки(&Дата, Товар В (&МассивТоваров) ) КАК ОстаткиТоваровОстатки
|ГДЕ
| ОстаткиТоваровОстатки.КоличествоОстаток < 0";

Запрос.УстановитьПараметр("Дата", Новый Граница ( МоментВремени(), ВидГраницы.Включая));
Запрос.УстановитьПараметр("МассивТоваров", СписокТоваров.ВыгрузитьКолонку("Товар"));
РезультатЗапроса = Запрос.Выполнить();

Если НЕ РезультатЗапроса.Пустой() Тогда
    Отказ = Истина;
    ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        Сообщить ("Не хватает товара " + ВыборкаДетальныеЗаписи.ТоварПредставление +
                    ", в количестве " + (- ВыборкаДетальныеЗаписи.КоличествоОстаток));
    КонецЦикла;
КонецЕсли;
КонецПроцедуры

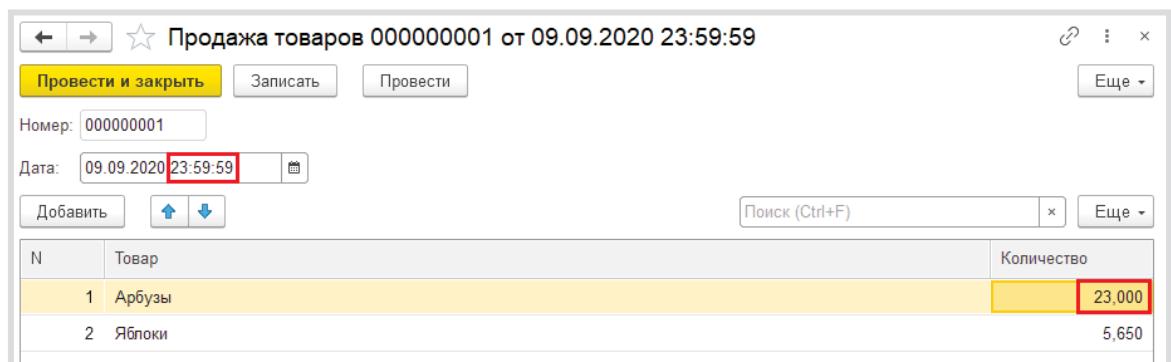
```

Теперь проверьте корректность работы системы на предмет наличия ошибок при попытке продать товара больше, чем его имеется в наличии.



Как видно из сообщения, товара не хватает ровно в том количестве, которое раньше выводилось в отчете со знаком «-» (минус).

Если изменить количество на меньшее, то механизм будет работать корректно.



Также отчет будет учитывать документы, проведенные в конце дня.

← → ⭐ Остатки товаров

Сформировать Выбрать вариант... Настройки...

Дата отчета: 09.09.2020

Параметры: Дата отчета: 09.09.2020

Товар	Количество
Арбузы	0,456
Яблоки	4,850

Поставленная задача решена.

Лабораторная работа № 18

**РАЗРАБОТКА КОНФИГУРАЦИИ
ДЛЯ УЧЕТА ТОВАРОВ.
ПРОДАЖА ТОВАРОВ
С ОДНОГО СКЛАДА**

Сложность: ***

Теги: справочник, документ, регистр накопления,
обработка проведения, запрос, схема компоновки
данных

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета товаров.

Суммовой учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся.

Учет товаров ведется в разрезе складов.

В системе необходимо регистрировать поступление товара. При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили в организацию. Нужно предусмотреть учет до граммов. В шапке документа выбираем склад, куда поступают товары.

В системе следует регистрировать продажу товара. При продаже товаров указывается, какие товары были проданы и в каком количестве, с какого склада производится списание. Склад выбирается в шапке документа.

Продать товар «в минус» нельзя, то есть в момент продажи нужно проверять остаток товара.

Необходимо построить «Отчет» по остаткам товаров следующего вида:

Остатки товаров на 31.01.2020

Товар/Склад	Юг	Север	Запад	Итого
Ложка	100.000	40.000		140.000
Вилка	45.000		80.000	125.000
Поварешка		12.000	1.000	13.000

Отчет строится на конец дня, указанного пользователем. Особое внимание следует уделить последней секунде дня: документы, записанные на эту секунду, должны попадать в отчет.

Выполнение

«Суммовой учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся».

Данная часть условия говорит нам о том, что никаких данных о суммах, валютах, покупателях и поставщиках в информационной системе хранить не нужно.

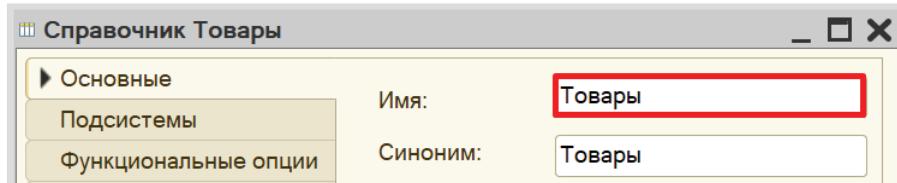
«Заказчик просит разработать конфигурацию для учета товаров. Учет товаров ведется в разрезе складов».

Появляется информация о тех объектах аналитики, которые нам понадобятся для решения поставленной задачи: товарах и складах. Для их хранения нам понадобятся *справочники*.

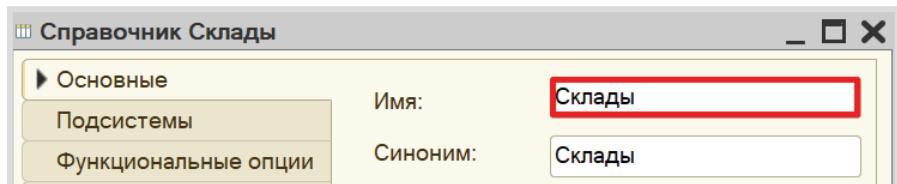
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

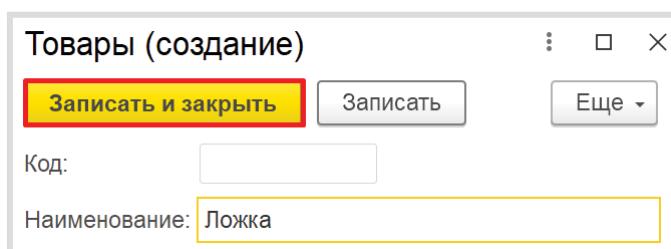
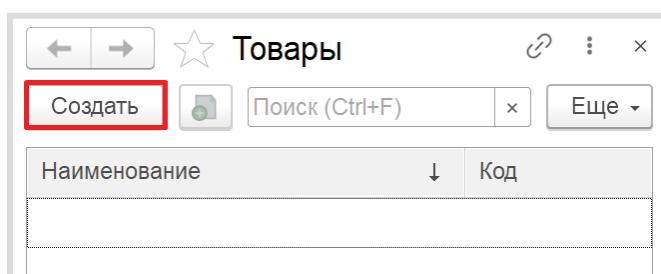
Добавим новый справочник «Товары».



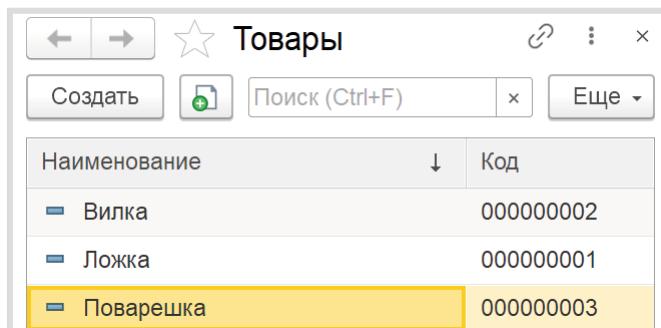
Добавим новый справочник «Склады».



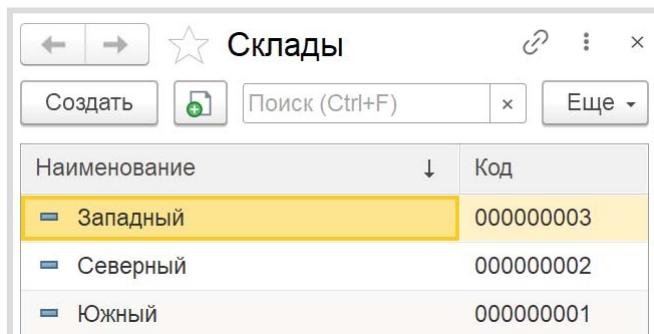
Откроем программу в режиме «1С:Предприятие» и добавим в каждый справочник несколько элементов.



Обратите внимание, что поля «Код» и «Наименование» система сгенерировала самостоятельно при добавлении нового справочника. Эти поля являются стандартными реквизитами. Стандартные реквизиты платформа создает автоматически, исходя из свойств конкретного объекта конфигурации. Поле «Код» заполнять не нужно, система сделает это автоматически. Поле «Наименование» является обязательным для заполнения.



Аналогично добавьте элементы в справочник «Склады».



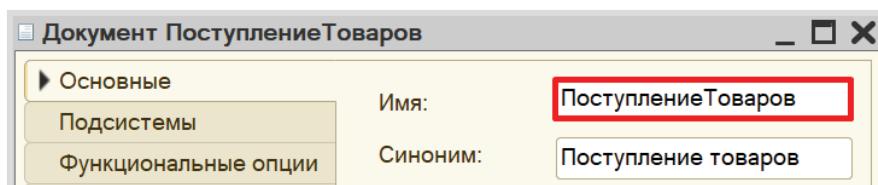
«В системе необходимо зарегистрировать поступление товара».

Для регистрации поступления товаров нужно воспользоваться объектом конфигурации *документ*.

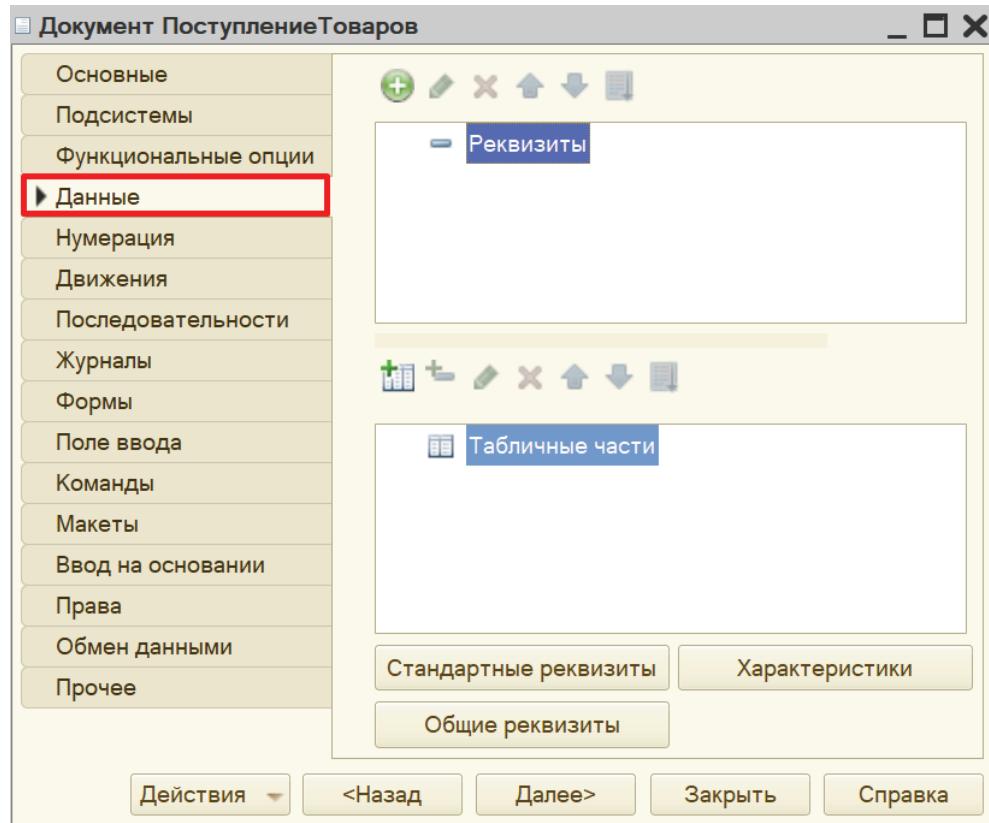
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty>).

Добавим новый документ «Поступление Товаров».

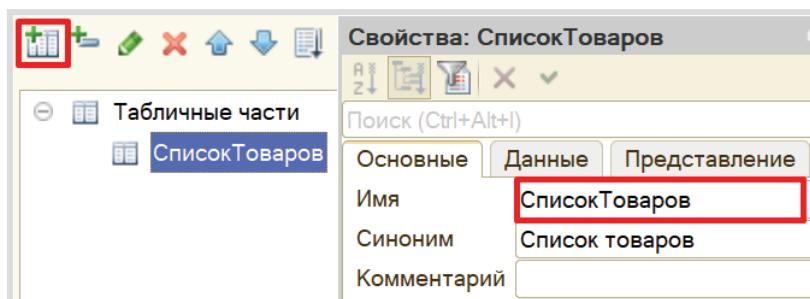


Для настройки структуры документа переходим на вкладку «Данные».

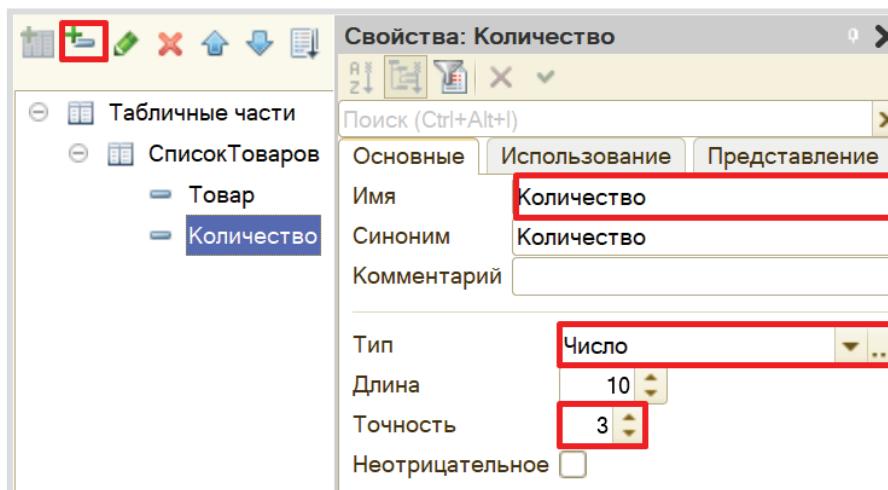
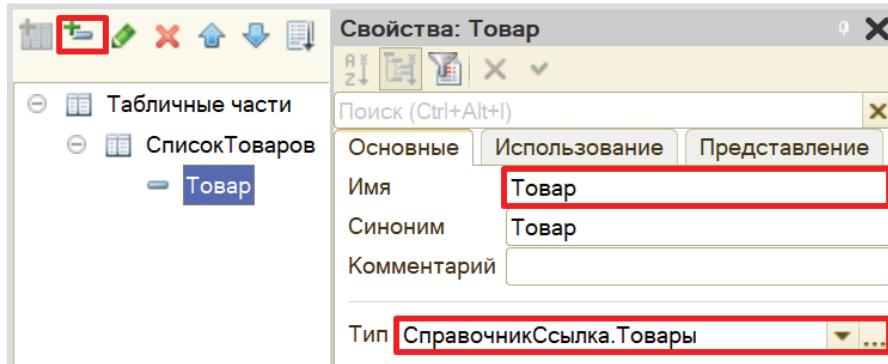


«При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили в организацию. Нужно предусмотреть учет до граммов».

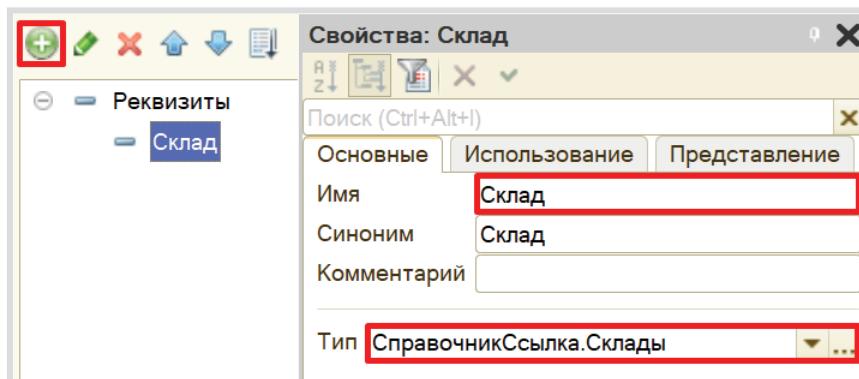
Добавим табличную часть.



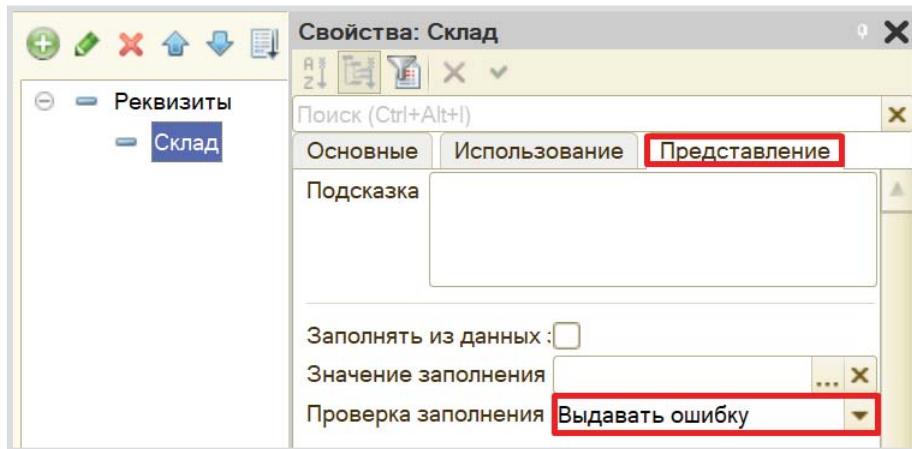
Теперь добавим два реквизита табличной части: «Товар» и «Количество».



«В шапке документа выбираем склад, куда поступают товары».

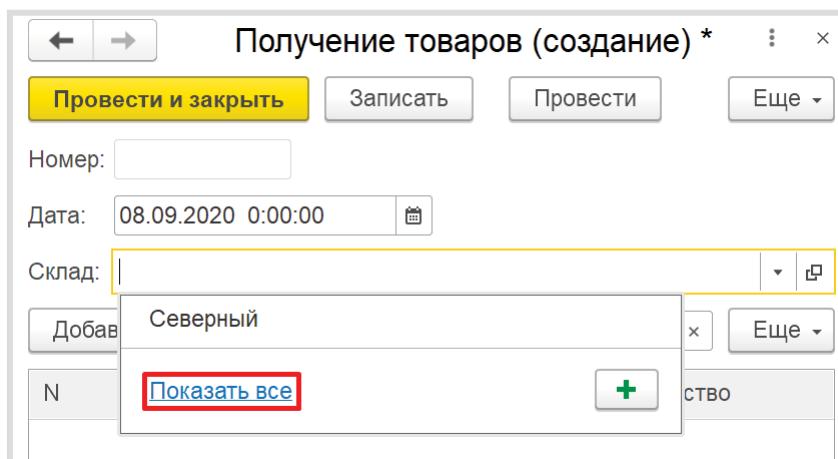
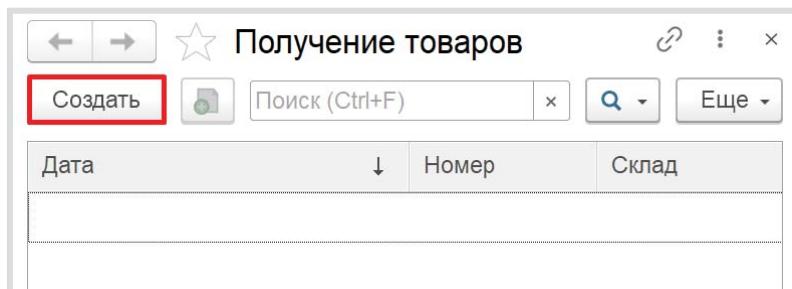


Сделаем данный реквизит обязательным для заполнения.



С помощью такой настройки пользователь не сможет сохранить документ, пока не заполнит поле «Склад».

Запустим режим «1С:Предприятие» и попробуем создать новый документ.



Склады

Наименование	Код
Западный	000000003
Северный	000000002
Южный	000000001

Получение товаров (создание) *

Номер:		
Дата: 08.09.2020 0:00:00		
Склад: Северный		
Добавить		
N	Товар	Количество
1	Ложка	10,000
2	Вилка	15,000
3	Поварешка	13,000

Получение товаров (создание) *

Номер:		
Дата: 08.09.2020 0:00:00		
Склад: Северный		
Добавить		
N	Товар	Количество
1	Ложка	10,000
2	Вилка	15,000
3	Поварешка	13,000

Любой документ может находиться в одном из двух состояний: *подготовленный к свершению* или *совершенный*:

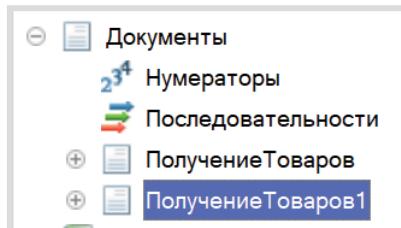
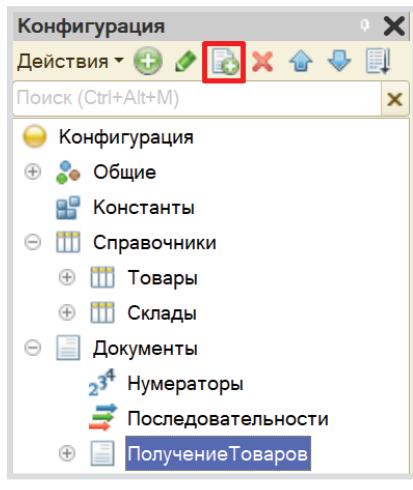
- чтобы подготовить документ для использования в будущем, необходимо его записать;
- чтобы отметить документ как совершенный – провести.

Теперь система способна регистрировать получение товаров на определенный склад.

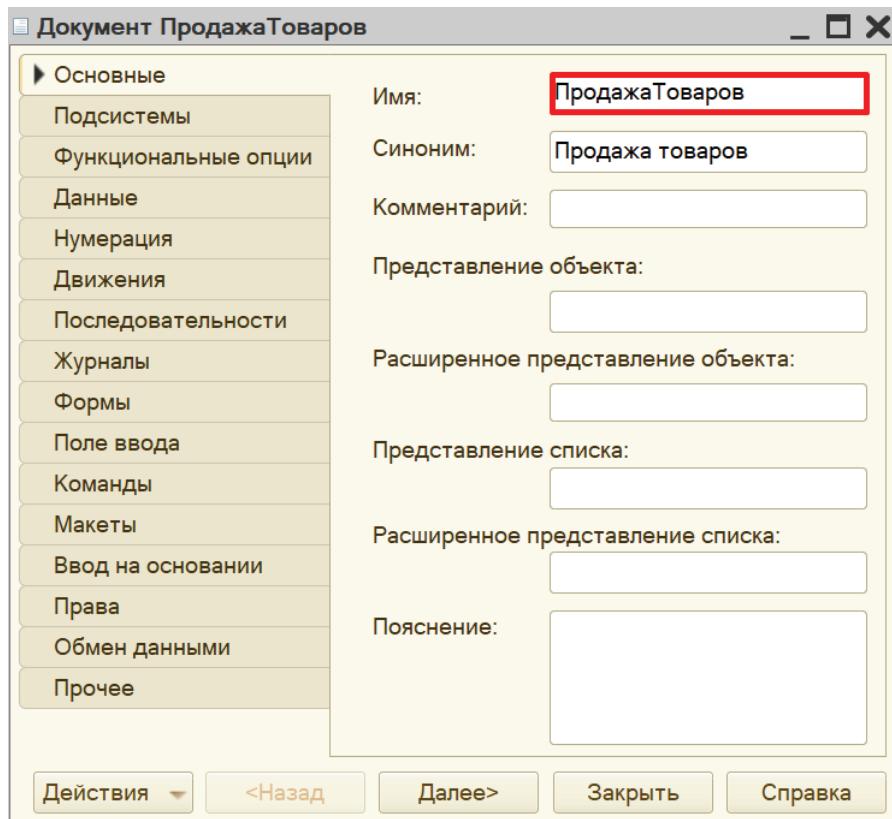
«В системе следует регистрировать продажу товара. При продаже товаров указывается, какие товары были проданы и в каком количестве, с какого склада производится списание. Склад выбирается в шапке документа».

Продажа товара осуществляется аналогично получению, следовательно, документ по структуре будет точно таким же, как документ «ПолучениеТоваров».

Чтобы не тратить время на создание точно такого же документа, воспользуемся возможностью платформы создать новый документ копированием.



Получаем точную копию документа. Изменим имя документа на «ПродажаТоваров».



На вкладке «Данные» структура должна быть аналогична структуре документа «ПолучениеТовара»: иметь реквизит шапки «Склад» и табличную часть с реквизитами «Товар» и «Количество».

«Продать товар "в минус" нельзя, то есть в момент продажи нужно проверять остаток товара».

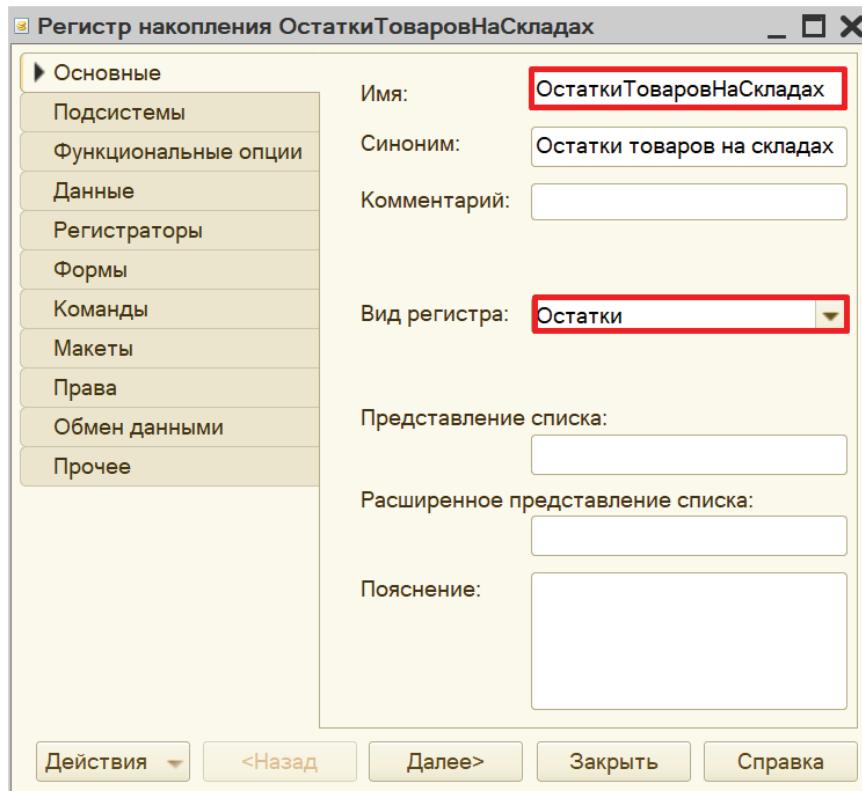
Если создать новый документ «ПродажаТовара» и попытаться продать со склада больше товаров, чем на складе имеется в данный момент, то система не сможет этого предотвратить, поскольку учет остатков никак не ведется.

Для начала следует каким-то образом вести подсчет остатков товаров на складах. Для этого нам потребуется *регистр накопления*.

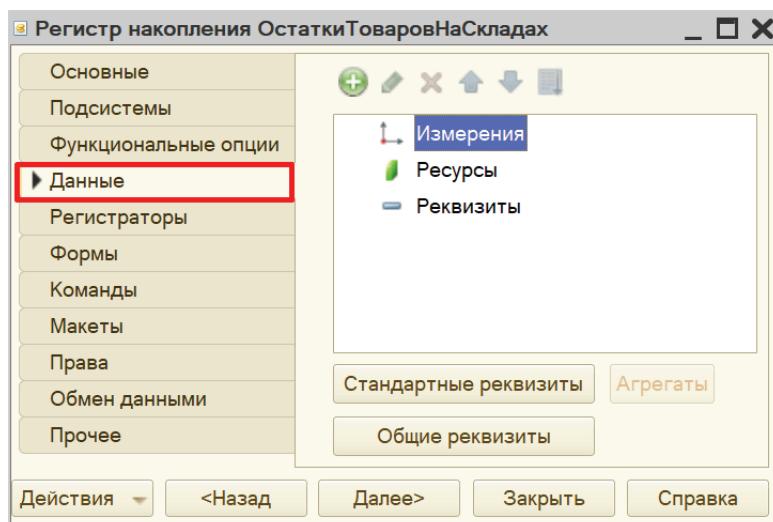
Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Добавим новый *регистр накопления* «ОстаткиТоваровНаСкладах». Вид данного регистра – «Остатки».

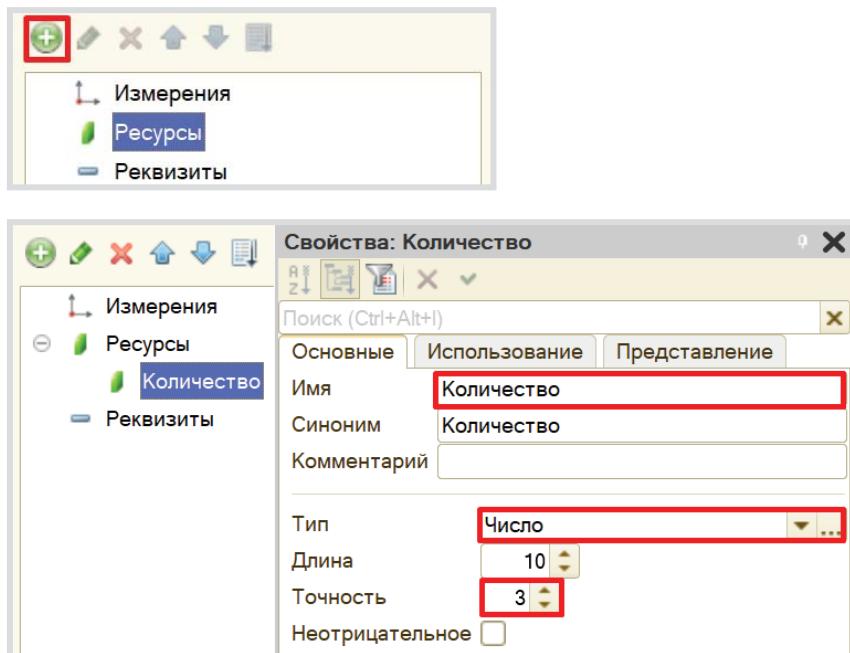


Как и в случае с документами, для формирования структуры переходим на вкладку «Данные».

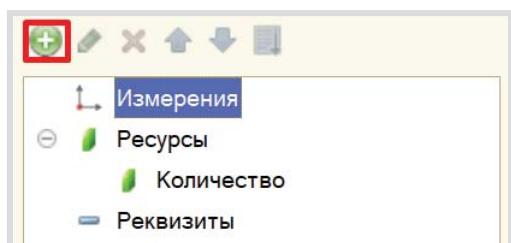


Структура регистра накопления отличается от структуры документа.

Заполнение данного окна проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, следует задать вопрос: «Что мы хотим накапливать/считывать в данном регистре?». Мы хотим считать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число». Точность – «3», поскольку в реквизит должно попадать количество с точностью до грамм.



Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим считать количество. Мы хотим считать количество (чего?) товаров в разрезе (чего?) складов. Значит, в качестве измерения следует добавить реквизиты «Товар» (тип – «СправочникСсылка.Товары») и «Склад» (тип – «СправочникСсылка.Склады»).

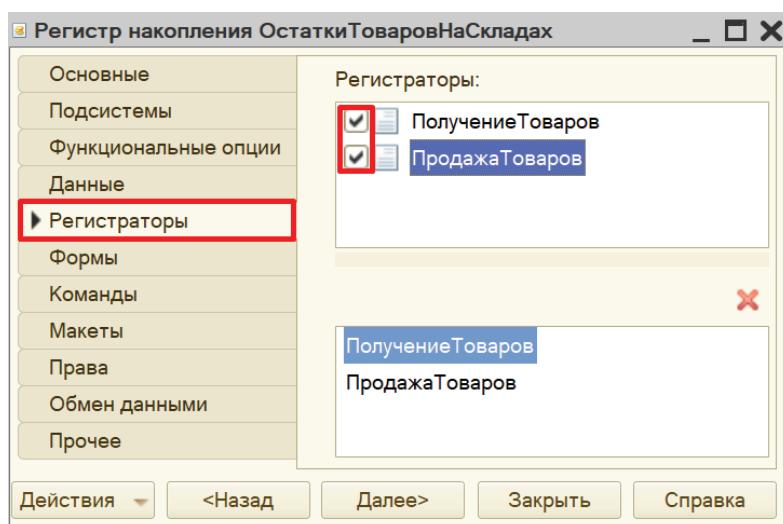


The first screenshot shows the 'Properties: Товар' dialog. The left sidebar lists 'Измерения' (Measurements) with 'Товар' selected, 'Ресурсы' (Resources), 'Количество' (Quantity), and 'Реквизиты' (Attributes). The main tab 'Основные' (Main) contains fields: 'Имя' (Name) set to 'Товар', 'Синоним' (Synonym) also set to 'Товар', and 'Комментарий' (Comment). The 'Тип' (Type) field is set to 'СправочникСсылка.Товары' (ReferenceLink.Products), which is highlighted with a red box.

The second screenshot shows the 'Properties: Склад' (Warehouse Properties) dialog. The left sidebar lists 'Измерения' with 'Склад' selected, 'Ресурсы', 'Количество', and 'Реквизиты'. The main tab 'Основные' contains fields: 'Имя' set to 'Склад', 'Синоним' also set to 'Склад', and 'Комментарий'. The 'Тип' field is set to 'СправочникСсылка.Склады' (ReferenceLink.Warehouses), which is highlighted with a red box.

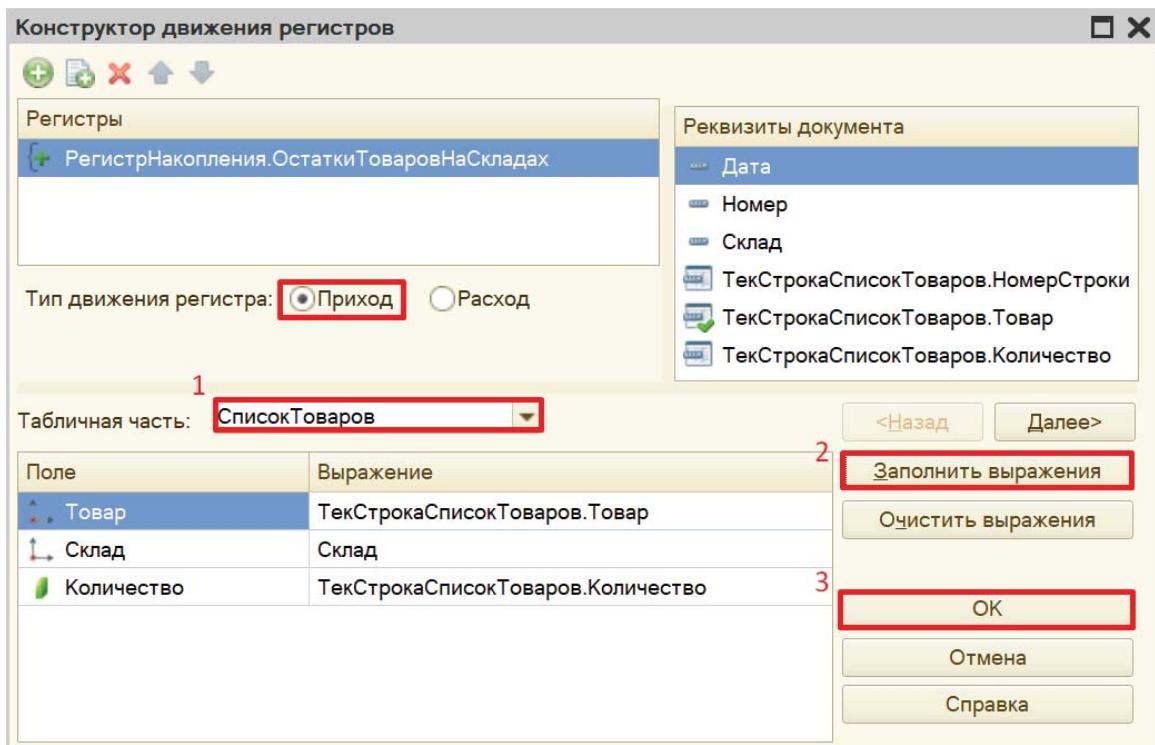
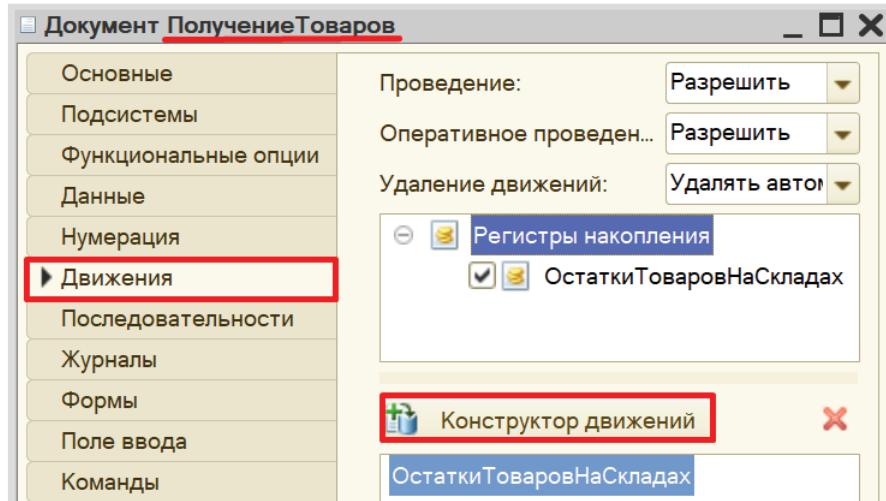
Чтобы *регистр накопления* заработал, нужно сделать следующее:

1. Определить источники данных регистра (определить документы-регистраторы).
 2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.
- В нашем случае на количество товаров будут влиять оба созданных документа. Определим их в качестве документов-регистраторов на вкладке «Регистраторы».



Далее для каждого из этих документов нужно описать процедуру копирования данных в *регистр накопления*.

Начнем с документа «Получение Товаров», откроем окно редактирования данного документа на вкладке «Движения». Воспользуемся *конструктором движений*.



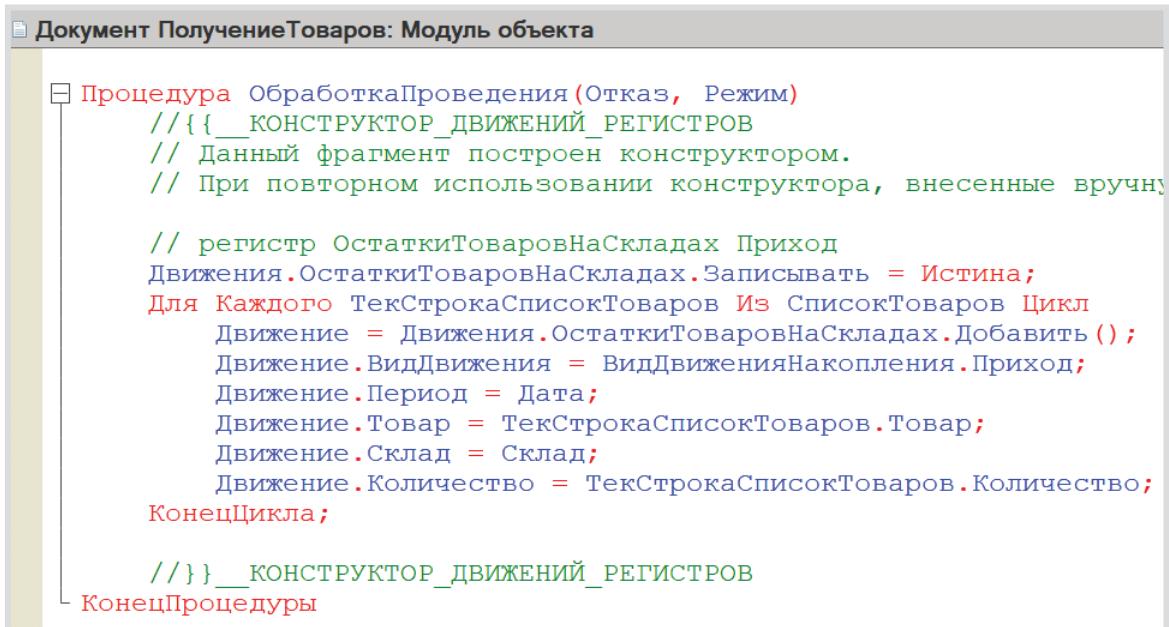
Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части, нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты регистра накопления. Нужно заполнить поле «Выражение» реквизитами документа.

Поскольку получение товара должно увеличивать количество товаров на складе, то тип движения регистра следует выбрать «Приход». Регистр будет обозначаться знаком «+» (плюс).

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.

При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в регистр накопления, то есть скопирует данные из документа в регистр накопления.



```

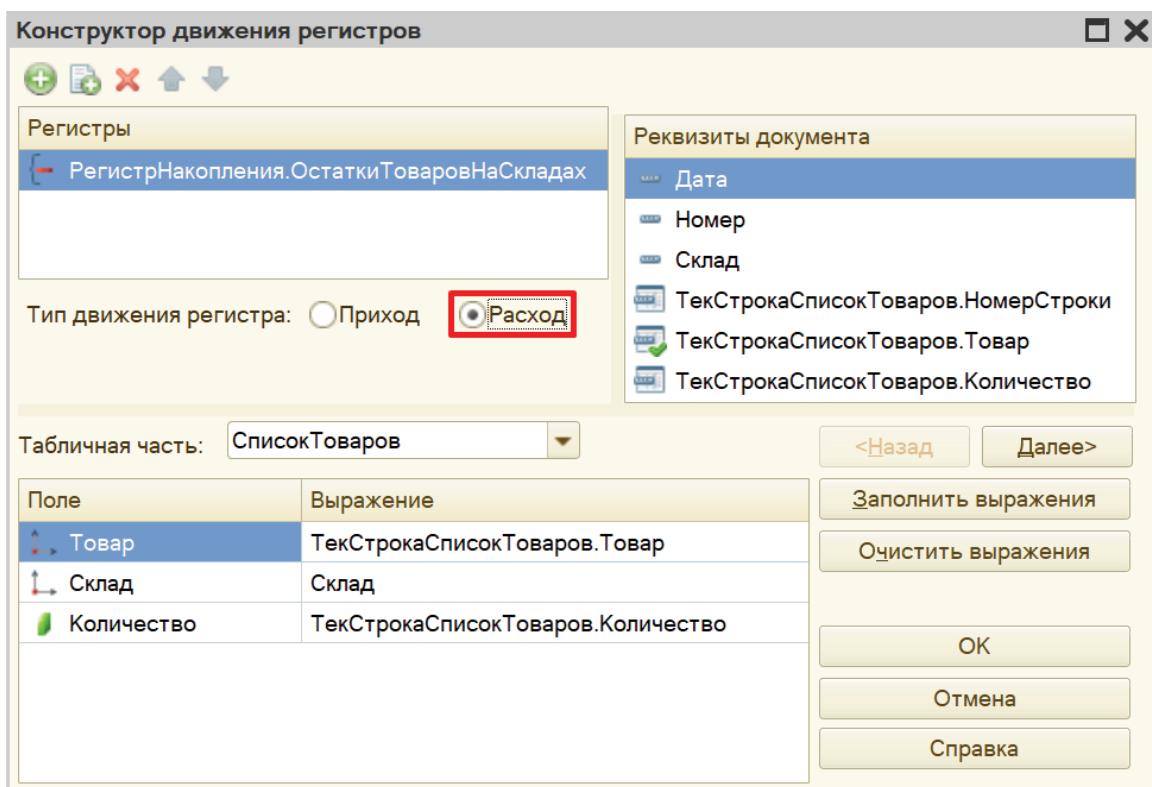
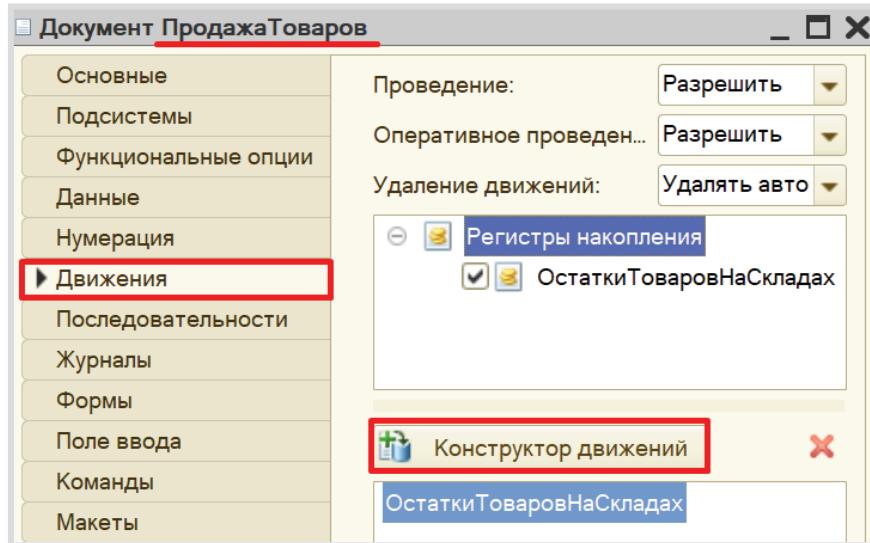
Документ ПолучениеТоваров: Модуль объекта

Процедура ОбработкаПроведения (Отказ, Режим)
// {{ _КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
//   Данный фрагмент построен конструктором.
//   При повторном использовании конструктора, внесенные вручную
//   регистр ОстаткиТоваровНаСкладах Приход
Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
  Движение = Движения.ОстаткиТоваровНаСкладах.Добавить ();
  Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
  Движение.Период = Дата;
  Движение.Товар = ТекСтрокаСписокТоваров.Товар;
  Движение.Склад = Склад;
  Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;

// }} _КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

Аналогичные действия нужно проделать с документом «Продажа Товаров».



Продажа товара должна уменьшать количество товаров на складе, значит, тип движения регистра следует выбрать «Расход». Регистр будет обозначаться знаком «-» (минус).

Документ ПродажаТоваров: Модуль объекта

```

Процедура ОбработкаПроведения(Отказ, Режим)
//{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
// Данный фрагмент построен конструктором.
// При повторном использовании конструктора, внесенные вручную

// регистр ОстаткиТоваровНаСкладах Расход
Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваровНаСкладах.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.Склад = Склад;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;

//}} __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

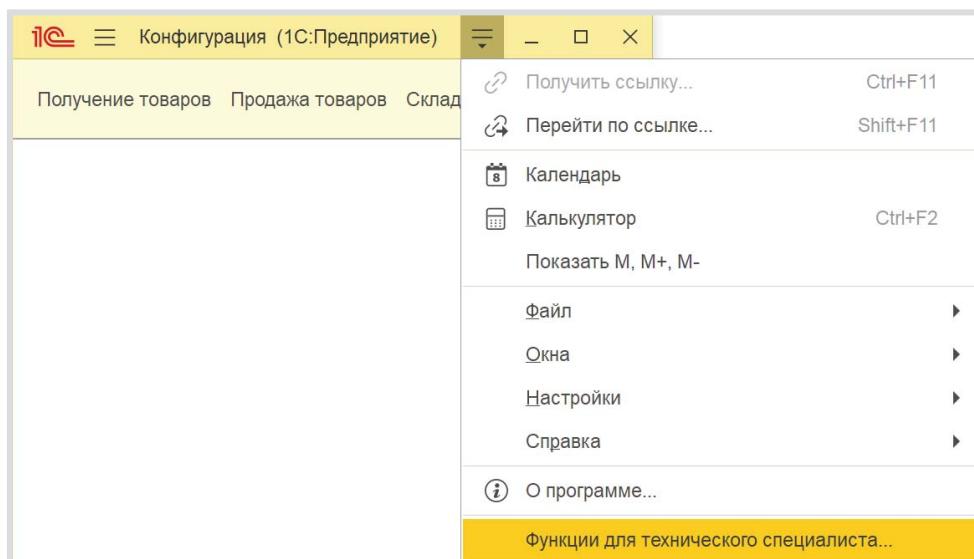
```

Откроем систему в режиме «1С:Предприятие» и проверим работу *регистра накопления*.

В первую очередь, необходимо перепровести (провести заново) созданный документ «Поступление товаров», а также создать и провести хотя бы один документ «Продажа товаров». Без проведения документов данные не будут скопированы в *регистр накопления*.

Обратите внимание, что на главной странице системы не создала кнопку открытия регистра накопления. Это связано с тем, что все расчеты в регистрах накопления происходят в фоновом режиме, «за кадром», то есть пользователю о них знать не нужно вовсе. Поэтому по умолчанию регистры накопления настраивают так, чтобы пользователи не имели к ним доступа.

Но мы, будучи разработчиками, можем обратиться к любому объекту конфигурации. Для этого воспользуемся функциями для технического специалиста.



В открывшемся списке найдем созданный нами *регистр накопления* и откроем его.

Функции для технического специалиста

Открыть Поиск (Ctrl+F) ?

- Регистры сведений
- Регистры накопления
 - Остатки товаров на складах
- Регистры бухгалтерии

Остатки товаров на складах

Поиск (Ctrl+F) ? Еще -

Период	↓	Регистратор	Номер строки	Товар	Склад
+ 08.09.2020 13:27:19		Получение товаров 000000001 ...	1	Ложка	Северный
+ 08.09.2020 13:27:19		Получение товаров 000000001 ...	2	Вилка	Северный
+ 08.09.2020 13:27:19		Получение товаров 000000001 ...	3	Поварешка	Северный
- 08.09.2020 13:27:33		Продажа товаров 000000001 от...	1	Поварешка	Северный
- 08.09.2020 13:27:33		Продажа товаров 000000001 от...	2	Вилка	Северный
- 08.09.2020 13:27:33		Продажа товаров 000000001 от...	3	Ложка	Северный

Таким образом, *регистр накопления* является некоторой итоговой таблицей. Сюда заносятся данные из документов-регистраторов по определенным правилам.

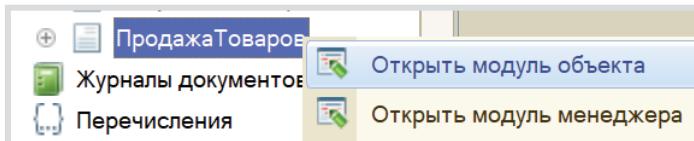
Таким образом, мы соединили между собой созданные ранее документы:

- документ «ПолучениеТоваров» увеличивает количество товаров на складах;
- документ «ПродажаТоваров», наоборот, уменьшает;
- Информация обо всех движениях товаров дублируется в *регистр накопления*.

«Продать товар "в минус" нельзя, то есть в момент продажи нужно проверять остаток товара».

К сожалению, регистра накопления недостаточно для того, чтобы вести учет отрицательных остатков. Нужно описать алгоритм работы документа «ПродажаТоваров».

Откроем модуль объекта документа «ПродажаТоваров» и дополним процедуру «ОбработкаПроведения».



Проверять остатки товаров будем следующим образом:

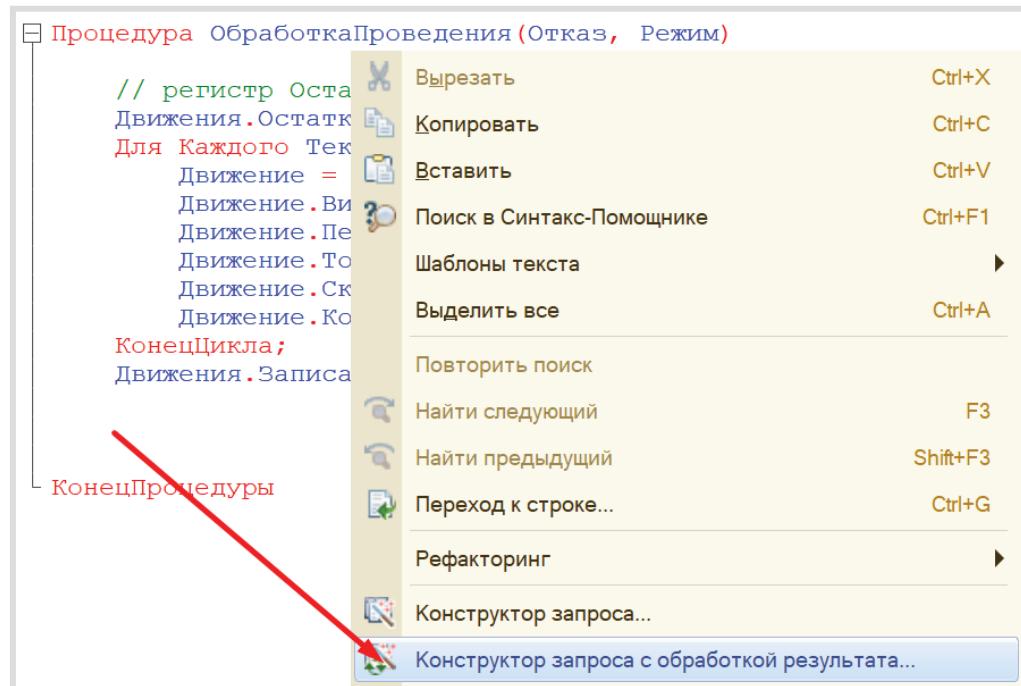
1. Сделаем движение данных из документа в *регистр накопления*.
2. Проверим, появились ли в регистре остатки, значение которых меньше нуля (то есть отрицательные).
3. Если есть отрицательные остатки, то отменим сделанное движение в *регистр накопления* и выведем пользователю сообщение об ошибке.

Чтобы сделать движение данных их документа в *регистр накопления*, допишем после окончания цикла строку «Движения.Записать();». Метод записывает только те движения документа, у которых установлен флаг «Записывать», при этом флаг в итоге снимается, что не приводит к повторной записи движений по окончании транзакции проведения. И главное, «Движения.Записать();» всегда записывают движения в том порядке, в котором таблицы указаны в дереве метаданных, что на порядок уменьшает шансы взаимных блокировок, ведь все транзакции в одинаковом порядке блокируют таблицы.

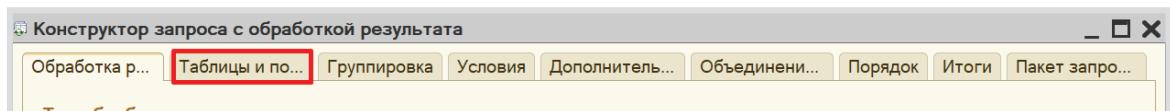
```
Процедура ОбработкаПроведения(Отказ, Режим)
    // регистр ОстаткиТоваровНаСкладах Расход
    Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
    Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
        Движение = Движения.ОстаткиТоваровНаСкладах.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Товар = ТекСтрокаСписокТоваров.Товар;
        Движение.Склад = Склад;
        Движение.Количество = ТекСтрокаСписокТоваров.Количество;
    КонецЦикла;
    Движения.Записать();
КонецПроцедуры
```

Теперь, когда движение было сделано, можно обратиться к данным *регистра накопления*.

Чтобы это сделать, воспользуемся конструктором запроса с обработкой результата. Этот конструктор можно открыть из контекстного меню щелчком правой кнопки мыши по области модуля. Данный конструктор обязательно должен быть вызван внутри процедуры «ОбработкаПроведения».



Соглашаемся с созданием нового запроса. Открывается окно *конструктора запроса с обработкой результата*. Переходим на вкладку «Таблицы и поля».



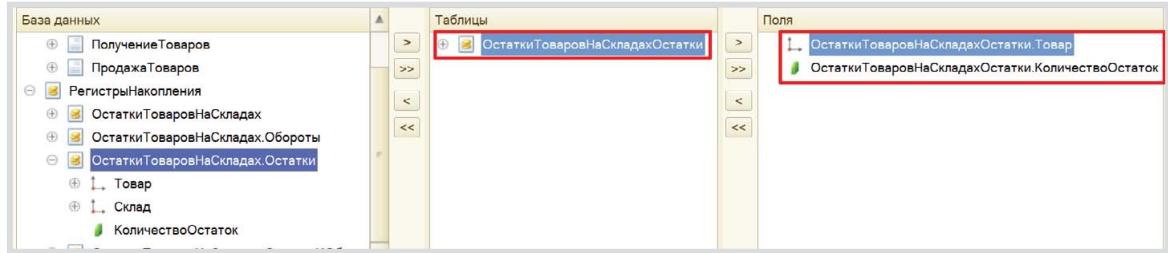
Открывшееся окно имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного запроса.
- Справа поля – это те значения (поля), которые мы хотим получить.

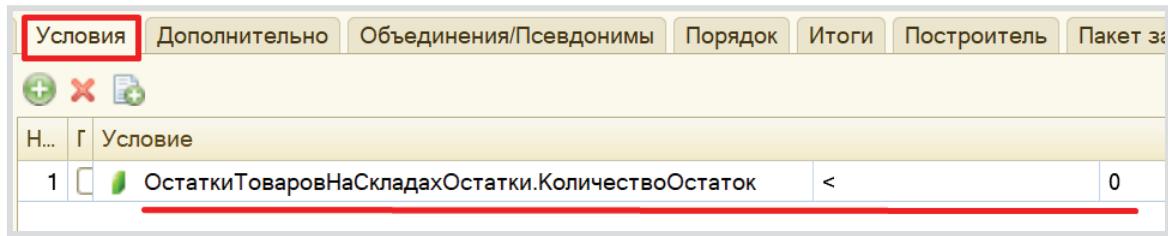
Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица способна обработать основную таблицу и самостоятельно посчитать остатки товаров.

Нам нужны поля «Товар» и «КоличествоОстаток» из таблицы «ОстаткиТоваровНаСкладахОстатки».

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.



Переходим на вкладку «Условие» и добавим новое условие. Пусть в запрос попадут только данные с отрицательными остатками.



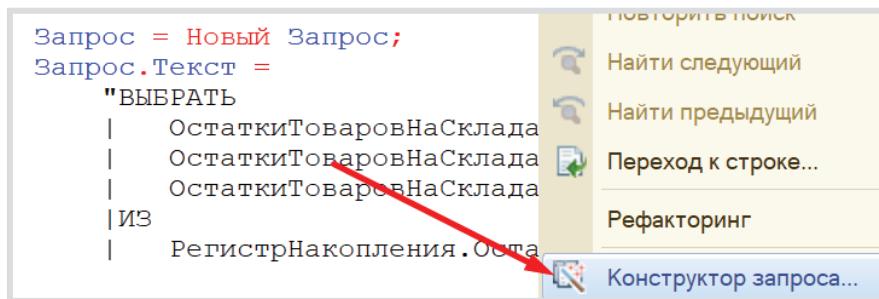
По окончании работы с данным конструктором нажимаем на кнопку «OK».

Конструктор выдаст предупреждение об ошибке, которое следует проигнорировать. Для корректной работы запроса нужно удалить знак амперсанта (&) перед нулем в условии. Запрос должен выглядеть следующим образом:

```
Запрос.Текст =
"ВЫБРАТЬ
| ОстаткиТоваровНаСкладахОстатки.Товар КАК Товар,
| ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток КАК КоличествоОстаток
| ИЗ
| РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки КАК ОстаткиТоваровНаСкладахОстатки
| ГДЕ
| ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток < 0";
```

С помощью данного запроса из базы данных можно получить отрицательные остатки по всем товарам со всех складов. Но нам нет необходимости получать такую большую выборку, нужно сузить запрос до склада, который указан в шапке документа и товаров, перечисленных в табличной части.

Откроем *конструктор запроса*. Для этого нужно щелкнуть в любом месте самого запроса (черный текст в двойных кавычках) правой кнопкой мыши и вызвать *конструктор запроса*.



Далее следует наложить условия на виртуальную таблицу *регистра накопления*.

Таблицы
+ ОстаткиТоваровНаСкладахОстатки

Параметры виртуальной таблицы

Период
Условие ...

OK Отмена Справка

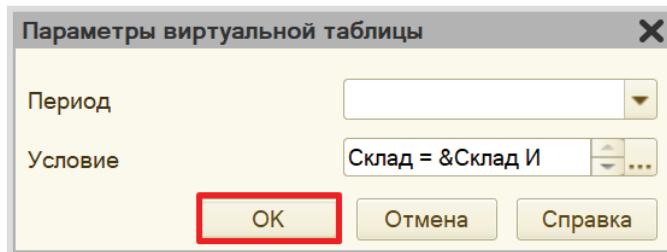
Произвольное выражение

Поле
+ Склад
+ Товар

+ Функции языка запросов

Склад = &Склад
И Товар В (&МассивТоваров)

OK Отмена Справка



Данное условие поможет ограничить запрос по складу и по тем товарам, которые находятся в табличной части документа.

Нажимаем на кнопку OK. Текст запроса изменился:

```
Запрос.Текст =
    "ВЫБРАТЬ
        | ОстаткиТоваровНаСкладахОстатки.Товар КАК Товар,
        | ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток КАК КоличествоОстаток
    | ИЗ
    |     РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки(
    |         ,
    |         Склад = &Склад
    |             И Товар В (&МассивТоваров) ) КАК ОстаткиТоваровНаСкладахОстатки
    | ГДЕ
    |     ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток < 0";
```

Мы добавили параметры. Теперь запрос будет проводить поиск только по конкретному складу и конкретному списку товаров. Осталось лишь указать этот склад и товары сразу после текста запроса.

```
Запрос.УстановитьПараметр ("Склад", Склад);
Запрос.УстановитьПараметр ("МассивТоваров", СписокТоваров.ВыгрузитьКолонку ("Товар"));
```

Ну, и последний шаг – выдать сообщение пользователю, если запрос вернул отрицательные остатки. В первую очередь, добавим блок условия:

```
РезультатЗапроса = Запрос.Выполнить ();
Если НЕ РезультатЗапроса.Пустой() Тогда
    КонецЕсли;
```

Внутрь цикла можно попасть только в том случае, если запрос пришел не пустой, то есть если были найдены отрицательные остатки. В таком случае нужно отменить проведение документа и выдать пользователю сообщение. Переместим весь оставшийся код внутрь данного цикла, а также сократим название переменной «ВыборкаДетальныеЗаписи» на «Выборка» для удобства.

```

РезультатЗапроса = Запрос.Выполнить ();
Если НЕ РезультатЗапроса.Пустой() Тогда
    Отказ = Истина;
    Выборка = РезультатЗапроса.Выбрать ();
    Пока Выборка.Следующий() Цикл
        Сообщить ("На складе " + Склад + " не хватает " +
                    Выборка.КоличествоОстаток*(-1) + " шт. товаров " + Выборка.Товар);
    КонецЦикла;
КонецЕсли;

```

Код процедуры полностью должен выглядеть следующим образом:

```

Процедура ОбработкаПроведения(Отказ, Режим)
    // регистр ОстаткиТоваровНаСкладах Расход
    Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
    Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
        Движение = Движения.ОстаткиТоваровНаСкладах.Добавить ();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Товар = ТекСтрокаСписокТоваров.Товар;
        Движение.Склад = Склад;
        Движение.Количество = ТекСтрокаСписокТоваров.Количество;
    КонецЦикла;
    Движения.Записать ();

    // контроль отрицательных остатков
    Запрос = Новый Запрос;
    Запрос.Текст =
        "ВЫБРАТЬ
        | ОстаткиТоваровНаСкладахОстатки.Товар КАК Товар,
        | ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток КАК КоличествоОстаток
    | ИЗ
    |     РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки(
    |
    |         ,
    |         Склад = &Склад
    |         И Товар В (&МассивТоваров) КАК ОстаткиТоваровНаСкладахОстатки
    | ГДЕ
    |     ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток < 0";

    Запрос.УстановитьПараметр("Склад", Склад);
    Запрос.УстановитьПараметр("МассивТоваров", СписокТоваров.ВыгрузитьКолонку("Товар"));

    РезультатЗапроса = Запрос.Выполнить ();
    Если НЕ РезультатЗапроса.Пустой() Тогда
        Отказ = Истина;
        Выборка = РезультатЗапроса.Выбрать ();
        Пока Выборка.Следующий() Цикл
            Сообщить ("На складе " + Склад + " не хватает " +
                        Выборка.КоличествоОстаток*(-1) + " шт. товаров " + Выборка.Товар);
        КонецЦикла;
    КонецЕсли;

КонецПроцедуры

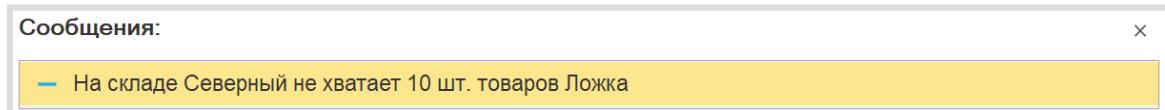
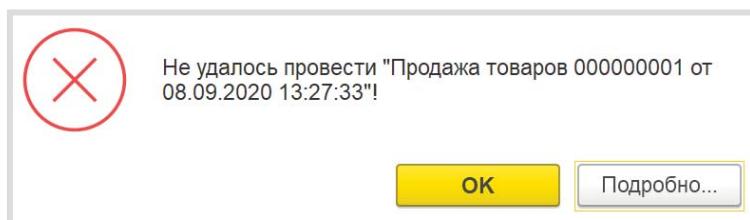
```

Переходим в режим «1С:Предприятие».

Добавим новый документ «Продажа товаров» так, чтобы хотя бы одного из товаров не хватало на складе.

N	Товар	Количество
1	Поварешка	5,000
2	Вилка	7,000
3	Ложка	20,000

Если все было сделано правильно, и вы пытаетесь продать товаров больше, чем имеется на складе, то система выдаст ошибку:



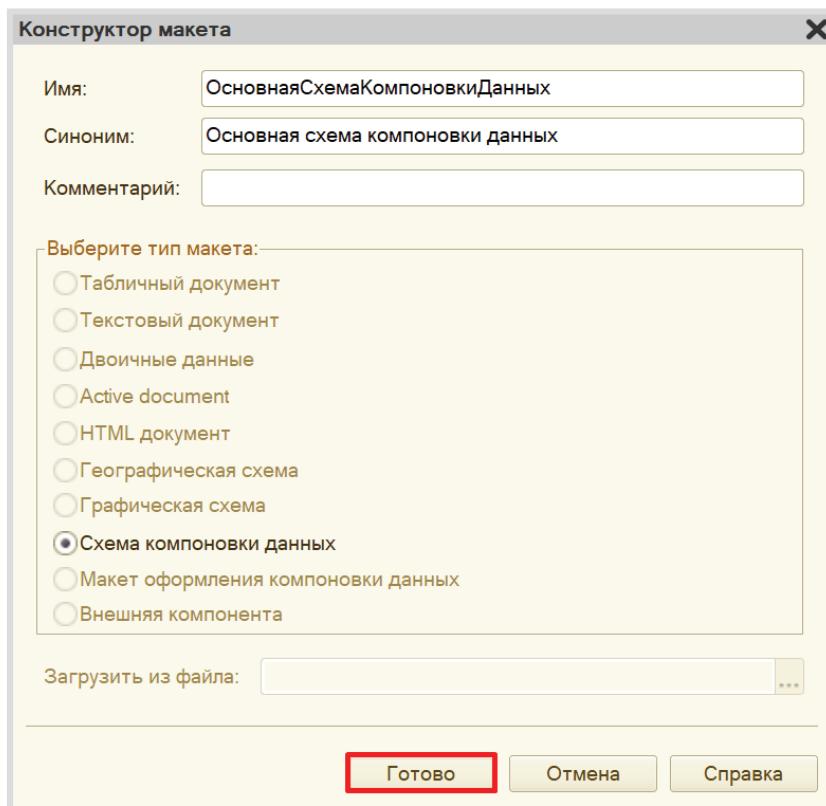
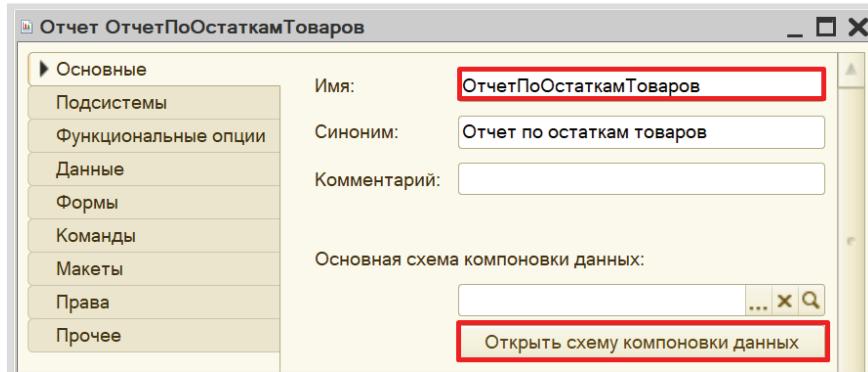
«Необходимо построить отчет по остаткам товаров».

Построим отчет. Для этого воспользуемся соответствующим объектом конфигурации.

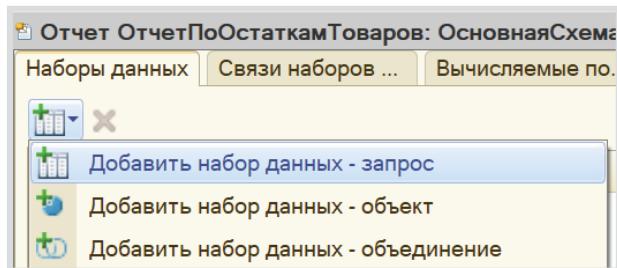
Определение

Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

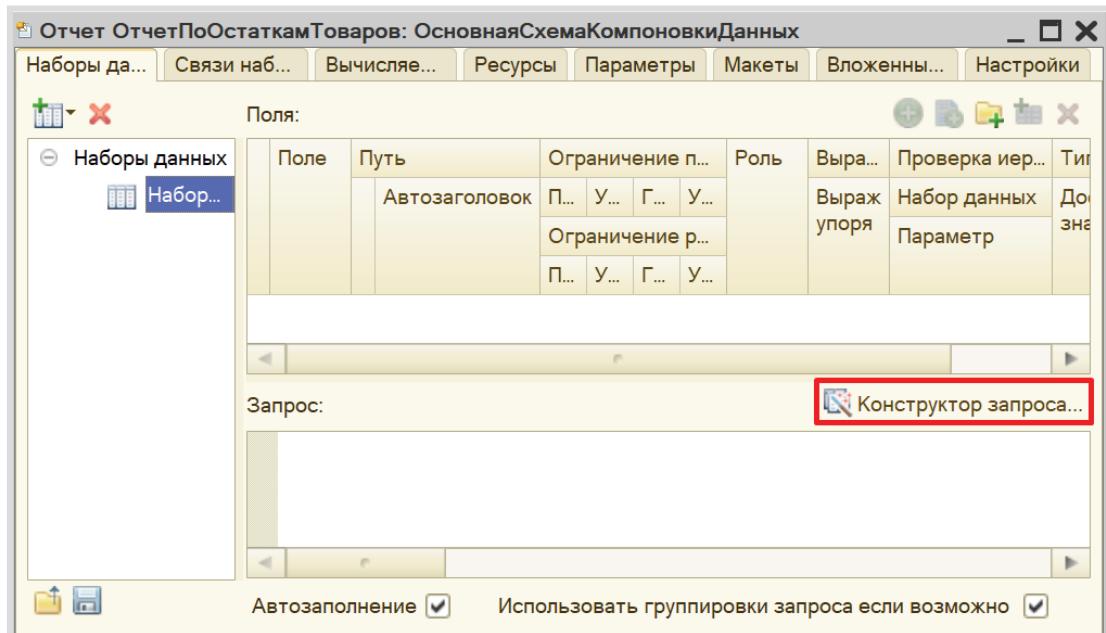
Добавим отчет «ОтчетПоОстаткамТоваров». Воспользуемся *схемой компоновки данных*.



Все созданные нами объекты конфигурации представляют собой *таблицы базы данных*. В режиме «1С:Предприятие» мы заполняем эти таблицы данными. Чтобы получить эти данные для отображения в отчете, нужно сформировать запрос к базе данных.



Для формирования запроса воспользуемся конструктором запроса.



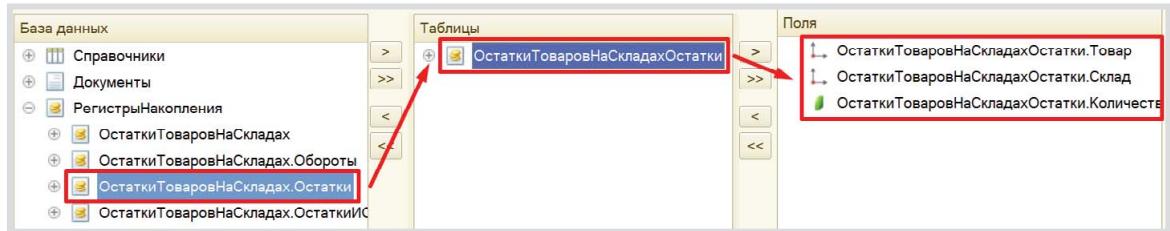
Открывается конструктор запроса. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица позволит получить уже просуммированные значения по всем документам.

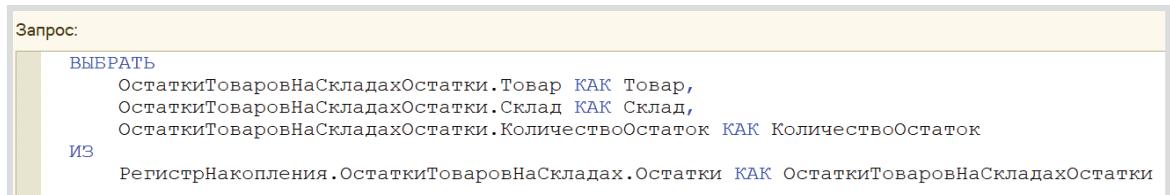
Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



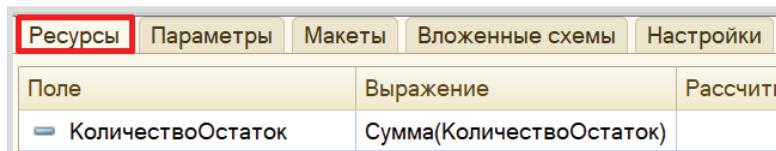
Закрываем конструктор запроса, нажав на кнопку «OK».

Получившийся запрос должен выглядеть так:



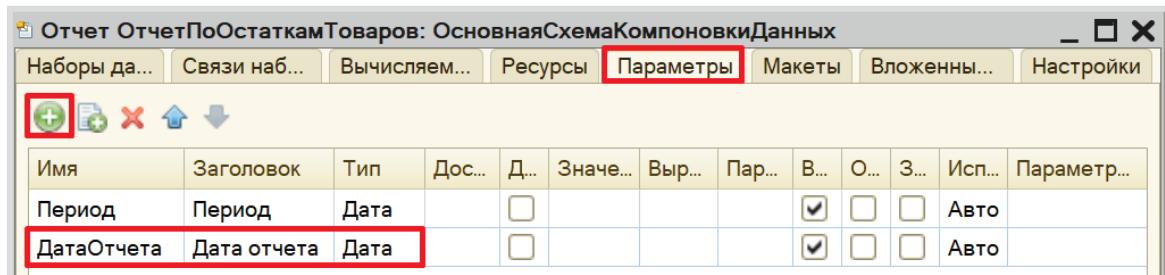
Теперь система понимает, какие данные ей нужны для формирования отчета.

Переходим на вкладку «Ресурсы» и устанавливаем реквизит «Количество.Остаток» в качестве ресурса: это позволит нам в отчете получать итоговые (просуммированные) значения.

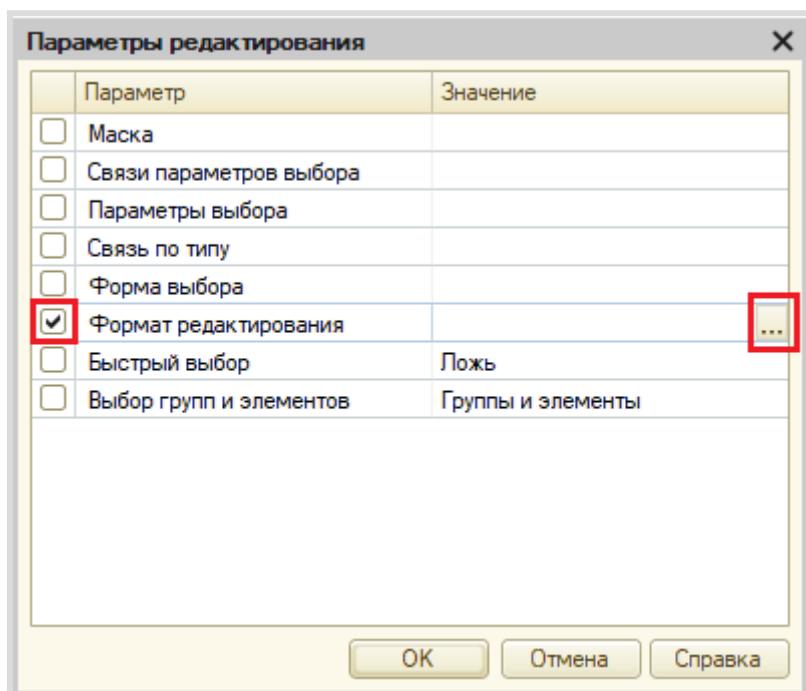
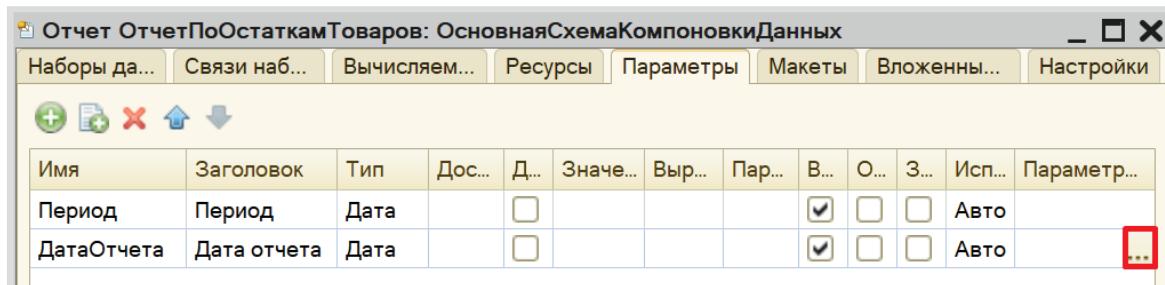


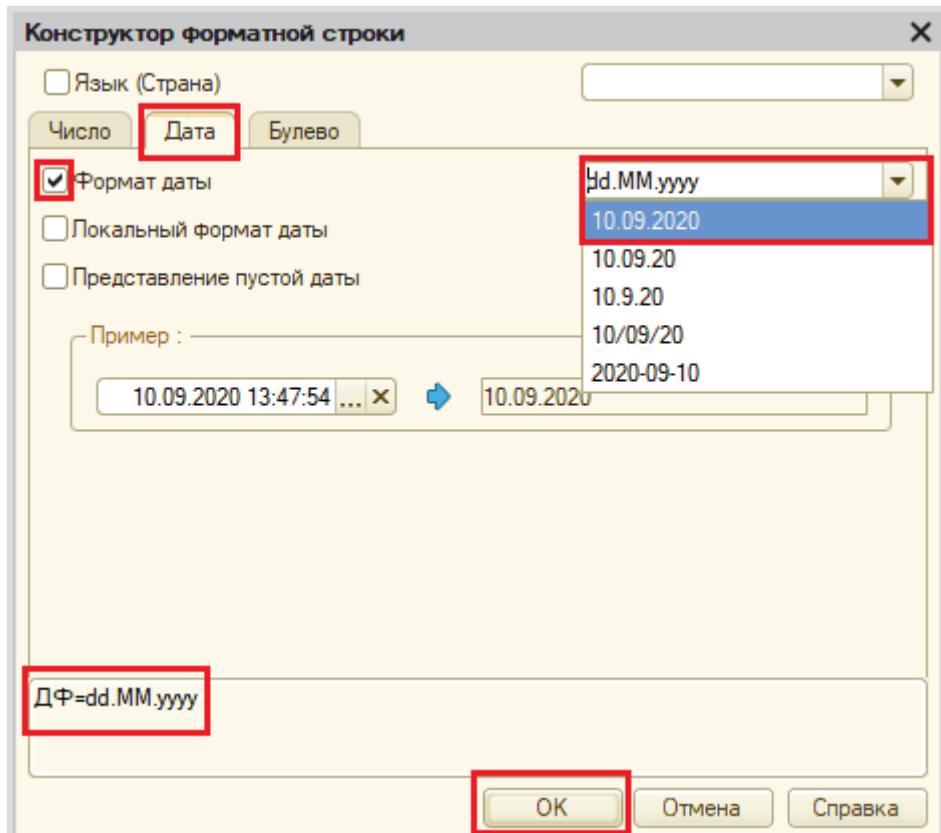
«Отчет строится на конец дня, указанного пользователем. Особое внимание следует уделить последней секунде дня: документы, записанные на эту секунду, должны попадать в отчет».

Из условия следует, что отчет должен включать документы, записанные на последнюю секунду дня. При использовании стандартных методов такие документы в отчет попадать не будут. Поэтому нужно добавить новый параметр «ДатаОтчета» на соответствующей вкладке.



Чтобы у пользователя была возможность выбирать только даты, без указания секунд, нужно настроить формат редактирования параметра «ДатаОтчета».





После нажатия кнопки «OK» нужно настроить стандартный параметр «Период» для корректного учета последней секунды дня:

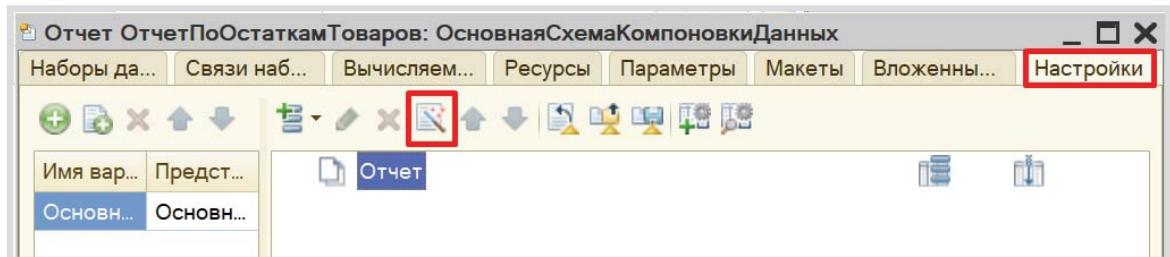
- во-первых, этот параметр должен быть недоступен пользователю, так как носит вычислительный характер;
- во-вторых, для корректного расчета требуется написать выражение для стандартного параметра «Период»:

ДОБАВИТЬКДАТЕ(КОНЕЦПЕРИОДА(&ДатаОтчета, "ДЕНЬ"),"СЕКУНДА",1)

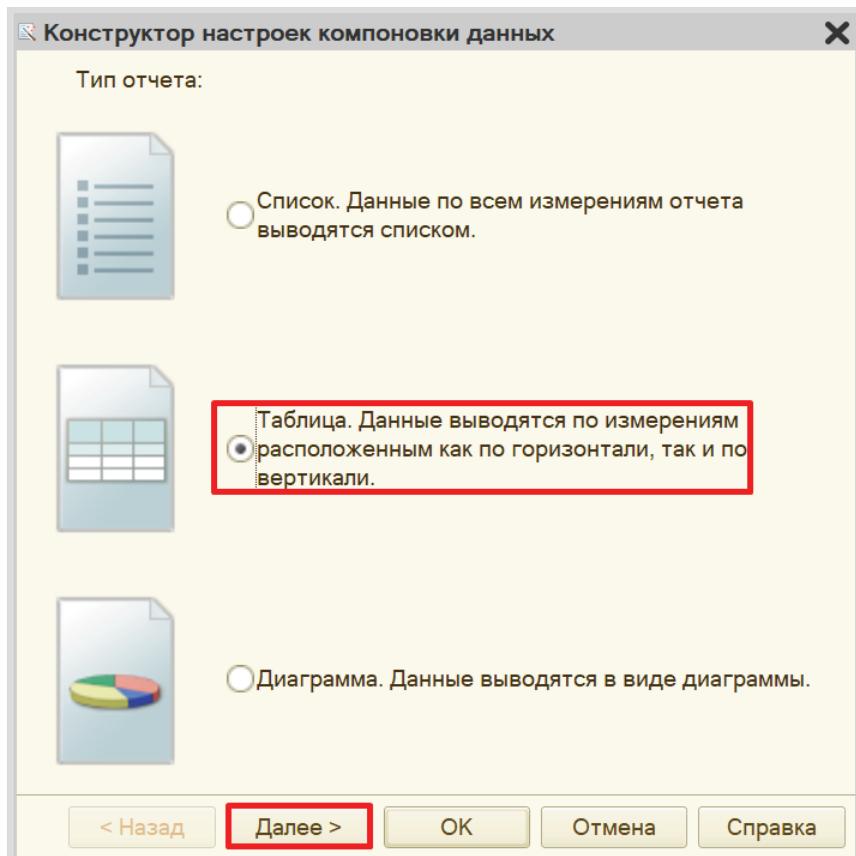
Сначала берется последний момент дня, указанного в параметре «ДатаОтчета», а затем прибавляется еще одна секунда, чтобы учитывались даже документы, проведенные за эту последнюю секунду.

Отчет ОтчетПоОстаткамТоваров: ОсновнаяСхемаКомпоновкиДанных												
Наборы данных		Связи наборов данных		Вычисляемые поля		Ресурсы		Параметры		Макеты	Вложенные схемы	
Имя	Заголовок	Тип	Д...	Д...	З...	Выражение	П...	В...	О...	З...	И...	Параметры редактирования
Период	Период	Дата	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	ДобавитьКДАТЕ(КОНЕЦПЕРИОДА(&ДатаОтчета, "ДЕНЬ"),"СЕКУНДА",1)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	A...
ДатаОтчета	Дата отчета	Дата	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Формат редактирования

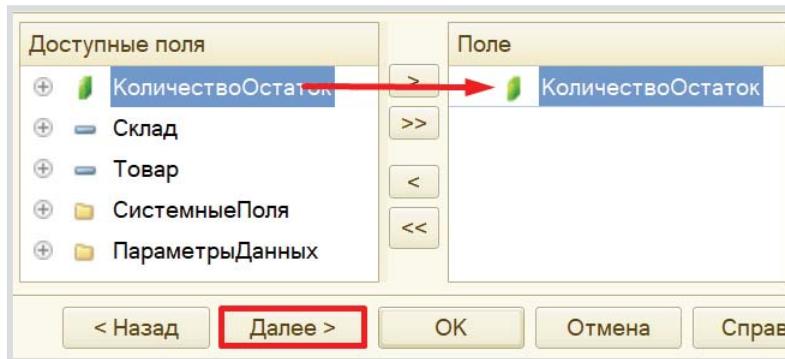
Теперь переходим на вкладку «Настройки» для оформления внешнего вида отчета. Воспользуемся конструктором настроек отчета.



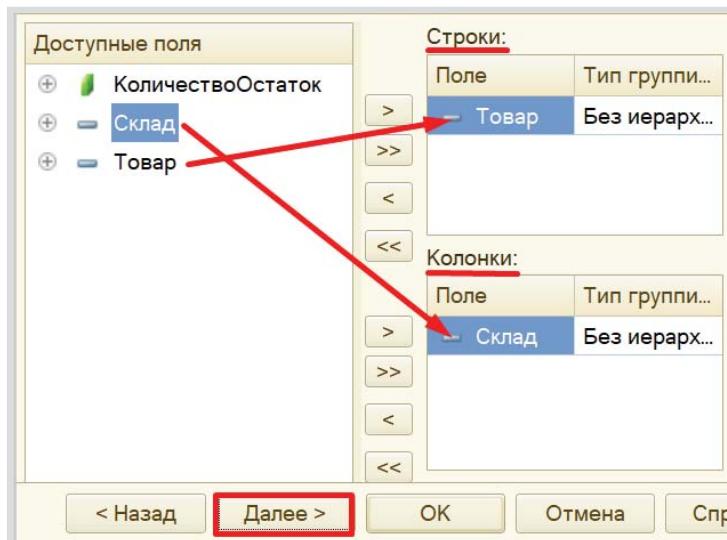
Построим отчет в виде таблицы.



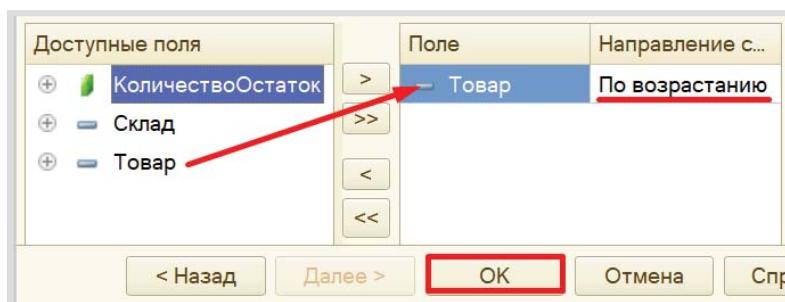
Далее следует выбрать ресурс, который будет отображен в таблице. В данном случае реквизит у нас всего один, его и выбираем.



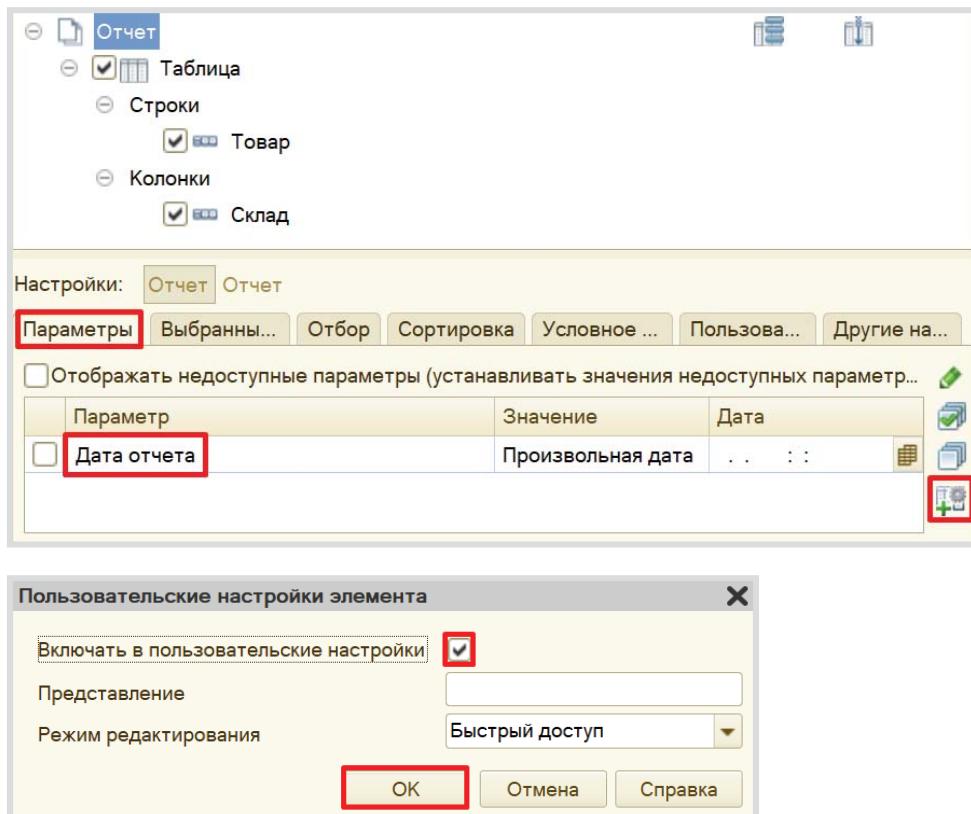
На следующем шаге необходимо определить те реквизиты, которыми будут заполняться колонки и строки таблицы.



Ну, и на последнем шаге нужно установить сортировку. Установим сортировку по товарам по возрастанию (то есть по алфавиту).

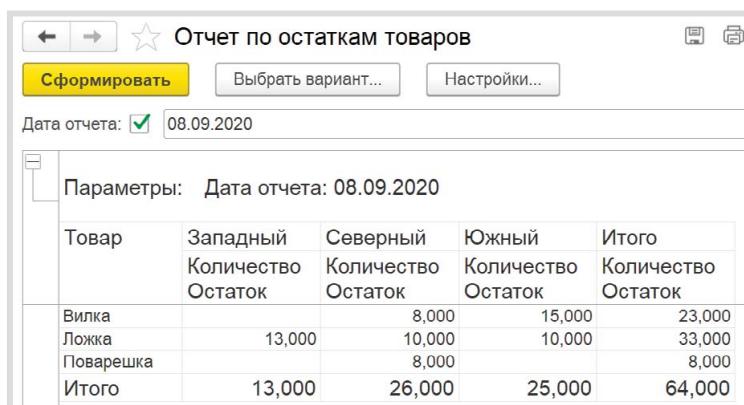


Чтобы у пользователя была возможность выбирать день, на который он хочет построить отчет, нужно включить параметр «ДатаОтчета» в пользовательские настройки.



Отчет готов. Запустим систему в режиме «1С:Предприятие».

Добавьте еще несколько документов по получению и продаже товаров, чтобы убедиться, что отчет работает корректно.



Поставленная задача решена.

Лабораторная работа № 19

**РАЗРАБОТКА КОНФИГУРАЦИИ
ДЛЯ УЧЕТА ТОВАРОВ.
ПРОДАЖА ТОВАРОВ
С РАЗНЫХ СКЛАДОВ**

Сложность: ***

Теги: справочник, документ, регистр накопления,
обработка проведения, запрос, схема компоновки
данных

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета товаров.

Суммовой учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся.

Учет товаров ведется в разрезе складов.

В системе необходимо регистрировать два вида операций: «Поступление товара» и «Продажа товара».

При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили в организацию. Нужно предусмотреть учет до граммов. В шапке документа выбирается склад, на который оформляется поступление.

При продаже товаров указывается, какие товары были проданы и в каком количестве, с какого склада. Склад, с которого списываются товары, выбирается для каждого товара в табличной части документа.

Продать товар «в минус» нельзя, в момент продажи следует проверять остаток товара.

Нужно построить «Отчет» по остаткам товаров следующего вида:

Остатки товаров на 31.01.2020

Товар/Склад	Юг	Север	Запад	Итого
Ложка	100.000	40.000		140.000
Вилка	45.000		80.000	125.000
Поварешка		12.000	1.000	13.000

Отчет строится на конец дня, указанного пользователем. Особое внимание следует уделить последней секунде дня: документы, записанные на эту секунду, должны попадать в отчет.

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть окно конфигурации.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

«Суммовой учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся».

Эта часть условия говорит нам о том, что никаких данных о суммах, валютах, покупателях и поставщиках в информационной системе хранить не нужно.

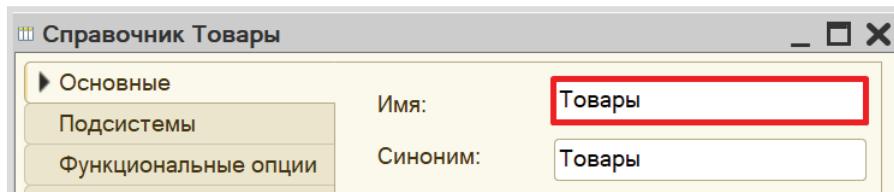
«Заказчик просит разработать конфигурацию для учета товаров. Учет товаров ведется в разрезе складов».

Появляется информация о тех объектах аналитики, которые нам понадобятся для решения поставленной задачи: товарах и складах. Для их реализации нам понадобятся *справочники*.

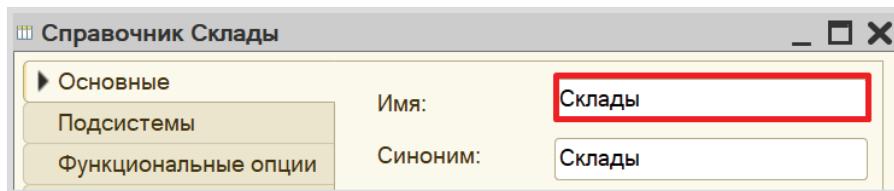
Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, перечень товаров или список сотрудников (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

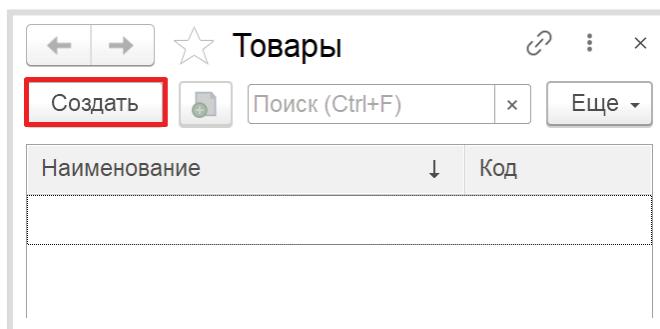
Добавим справочник «Товары».



Добавим справочник «Склады».



Откроем программу в режиме «1С:Предприятие» и добавим в каждый справочник несколько элементов.



Товары (создание)

Записать и закрыть Записать Еще ▾

Код:

Наименование: Ложка

Товары

← → ⌂ ⋮ ×

Создать Помощь (Ctrl+F) × Еще ▾

Наименование	↓	Код
— Вилка		000000002
— Ложка		000000001
— Поварешка		000000003

Аналогично добавьте элементы в справочник «Склады».

Склады

← → ⌂ ⋮ ×

Создать Помощь (Ctrl+F) × Еще ▾

Наименование	↓	Код
— Западный		000000003
— Северный		000000002
— Южный		000000001

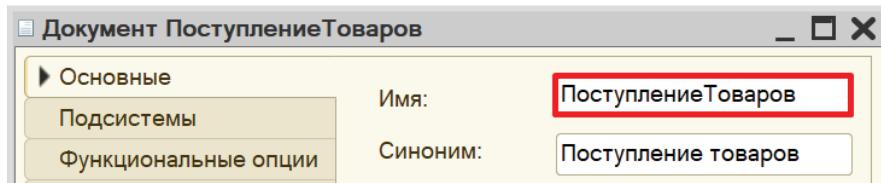
«В системе необходимо регистрировать поступление товара».

Для регистрации поступления товаров следует воспользоваться объектом конфигурации *документ*.

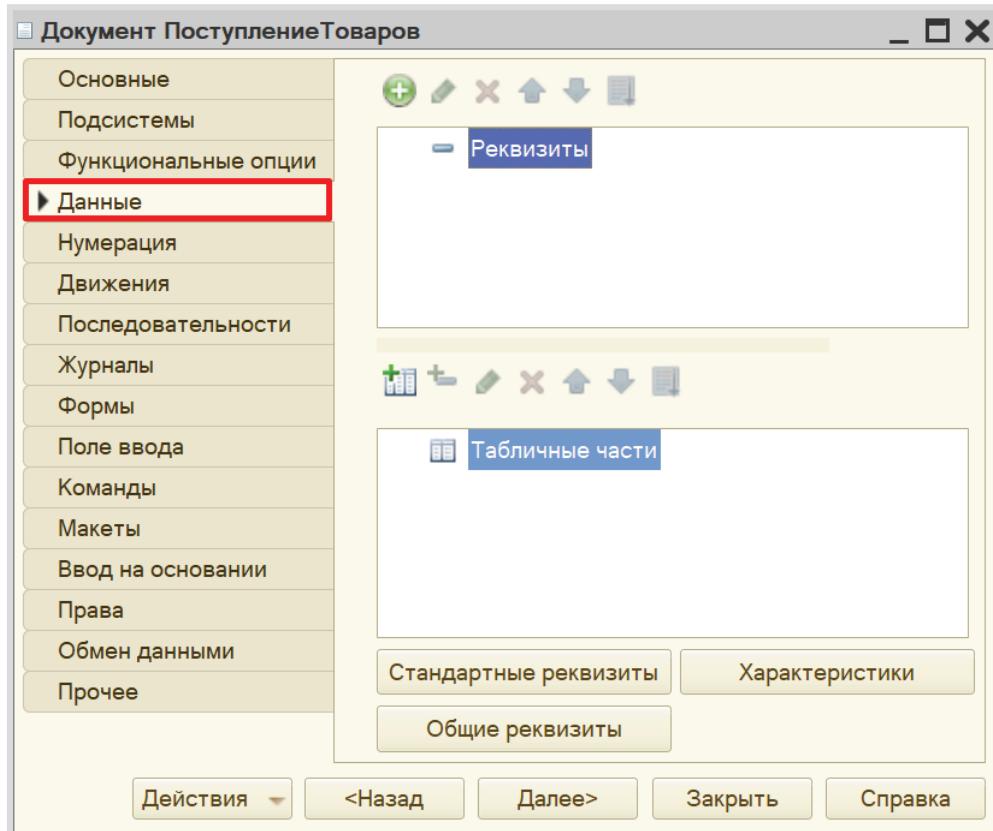
Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

Добавим новый документ «Поступление Товаров».

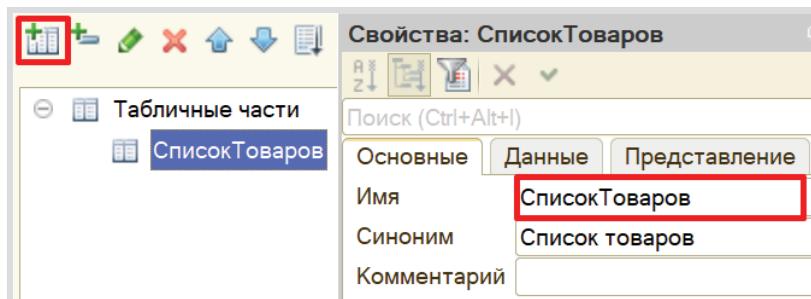


Для настройки структуры документа переходим на вкладку «Данные».

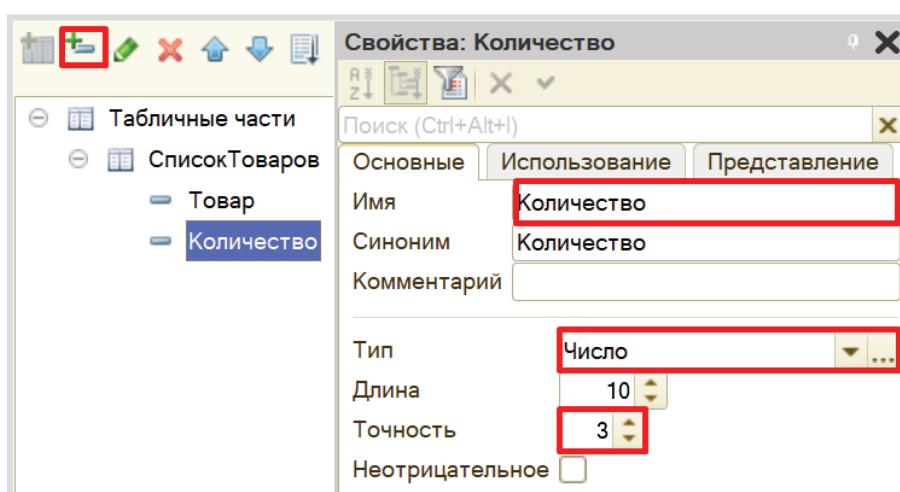
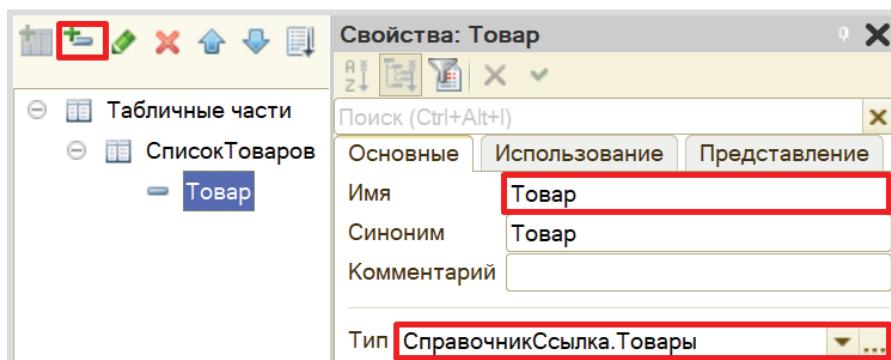


«При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили в организацию. Нужно предусмотреть учет до граммов».

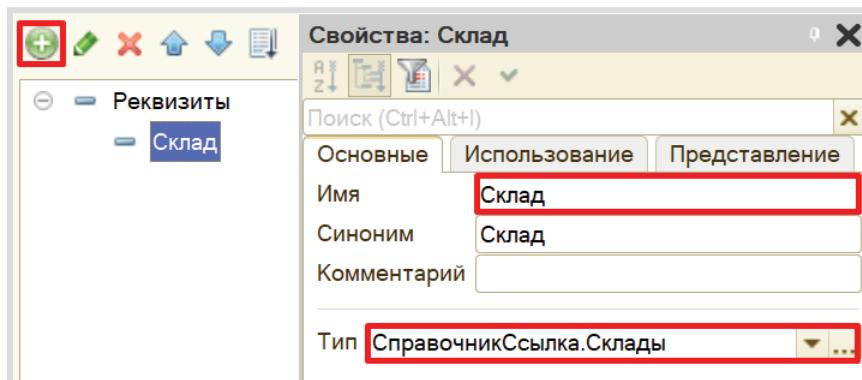
Добавим табличную часть.



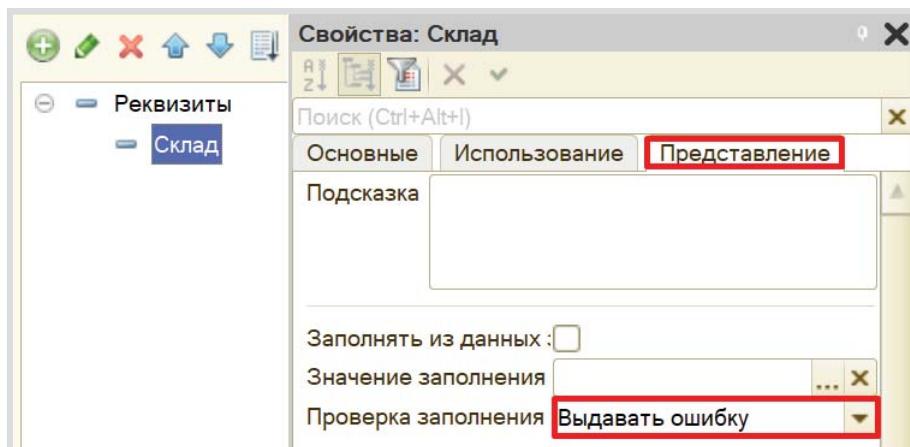
Теперь добавим два реквизита табличной части: «Товар» (тип – «СправочникСсылка.Товары») и «Количество» (тип – «Число»).



«В шапке документа выбираем склад, куда поступают товары».

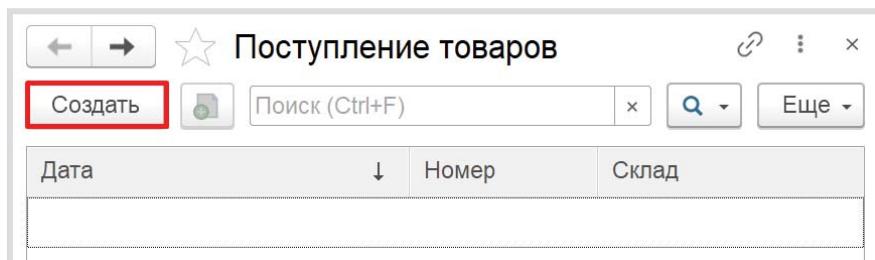


Сделаем данный реквизит обязательным для заполнения.



С помощью такой настройки пользователь не сможет сохранить документ, пока не заполнит поле «Склад».

Запустим режим «1С:Предприятие» и попробуем создать новый документ.



Поступление товаров (создание) *

← → Провести и закрыть Записать Провести Еще ▾

Номер:

Дата: 09.09.2020 0:00:00

Склад:

Добавить Введите строку для поиска
Нажмите [Показать все](#) для выбора
Нажмите [+](#) ([создать](#)) для добавления

[Показать все](#)

Склады

Выбрать Поиск (Ctrl+F) Еще ▾

Наименование	Код
Западный	000000003
Северный	000000002
Южный	000000001

Поступление товаров (создание) *

← → Провести и закрыть Записать Провести Еще ▾

Номер:

Дата: 09.09.2020 0:00:00

Склад: Северный

[Добавить](#) Поиск (Ctrl+F) Еще ▾

N	Товар	Количество

Поступление товаров (создание) *

Провести и закрыть **Записать** **Провести** **Еще** ▾

Номер:

Дата: 09.09.2020 0:00:00

Склад: Северный

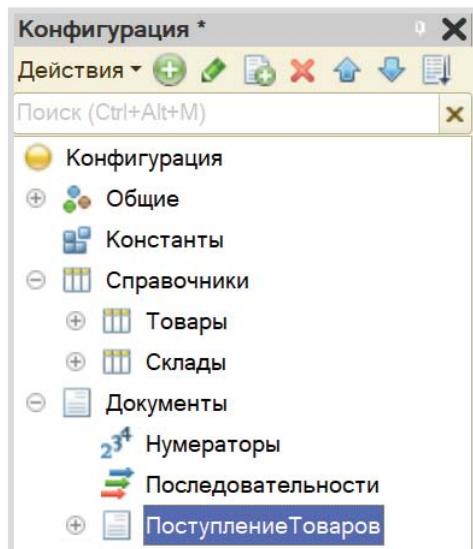
Добавить **Поиск (Ctrl+F)** **Еще** ▾

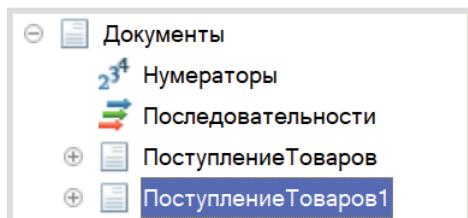
N	Товар	Количество
1	Ложка	10,000
2	Вилка	15,000
3	Поварешка	13,000

Теперь система способна регистрировать получение товаров на определенный склад.

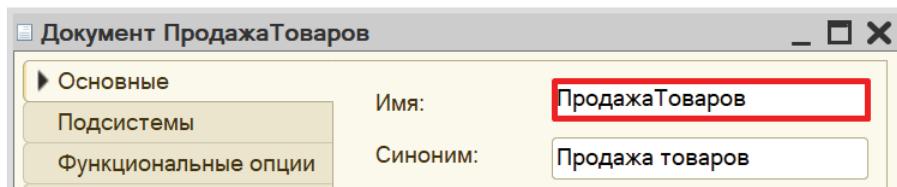
Продажа товара осуществляется аналогично получению, следовательно, документ по структуре будет точно таким же, как документ «Поступление Товаров».

Чтобы не тратить время на создание точно такого же документа, воспользуемся возможностью платформы создать новый документ копированием.





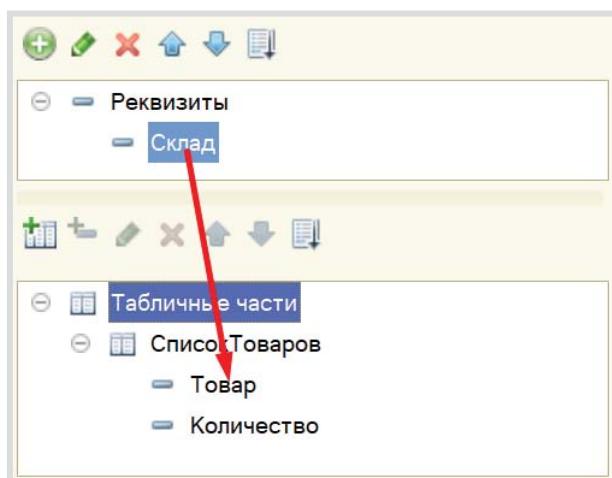
Получаем точную копию документа. Изменим имя документа на «Продажа Товаров».



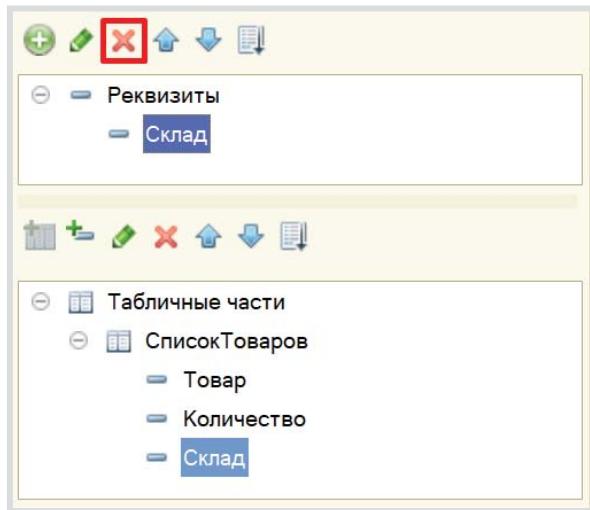
«При продаже товаров указывается, какие товары были проданы и в каком количестве, с какого склада. Склад, с которого списываются товары, выбирается для каждого товара в табличной части документа».

На вкладке «Данные» структура должна быть немного изменена.

Реквизит «Склад» следует скопировать в табличную часть. Перетаскивать реквизит нужно именно в список реквизитов табличной части.



Теперь реквизит «Склад» из шапки документа нужно удалить.



«Продать товар "в минус" нельзя, то есть в момент продажи следует проверять остаток товара».

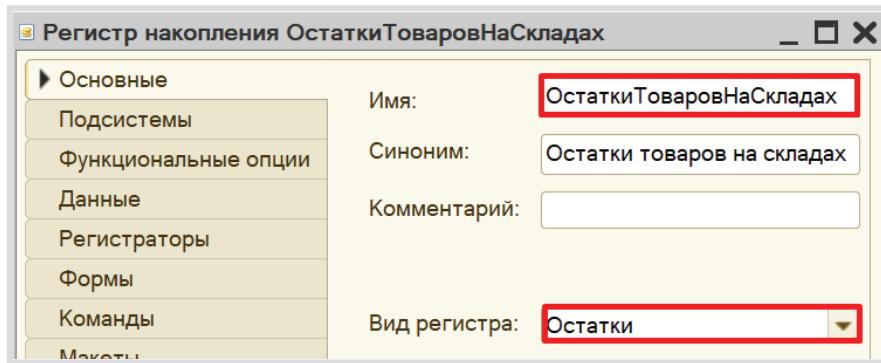
Если создать новый документ «ПродажаТовара» и попытаться продать со склада больше товаров, чем на складе имеется в данный момент, то система не сможет этого предотвратить, поскольку учет остатков никак не ведется.

Для начала нужно каким-то образом вести подсчет остатков товаров на складах. Для этого нам потребуется *регистр накопления*.

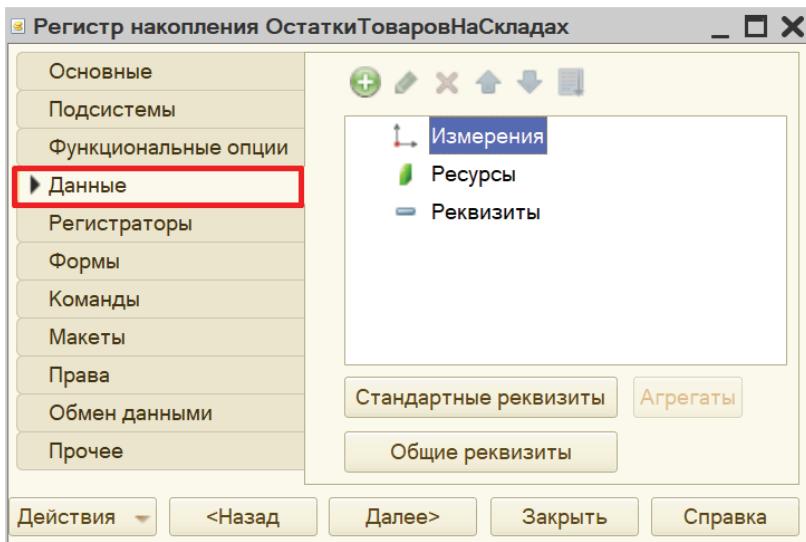
Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

Добавим новый *регистр накопления* «ОстаткиТоваровНаСкладах». Вид данного регистра – «Остатки».

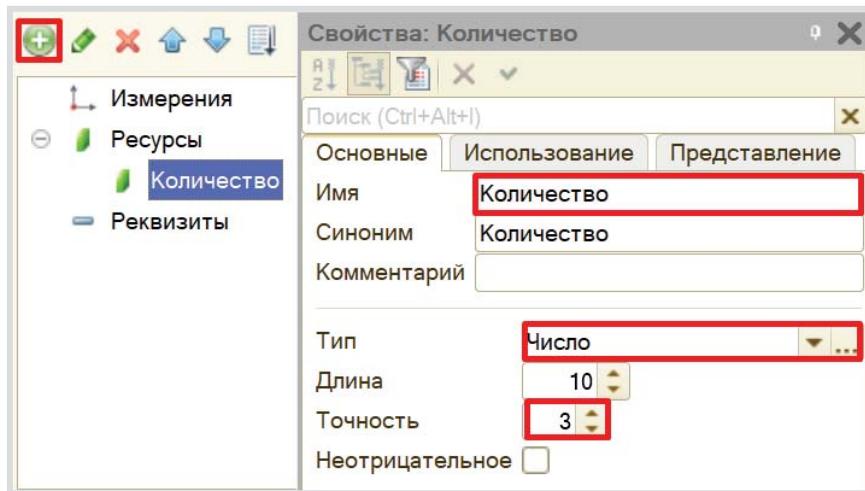


Как и в случае с документами, для формирования структуры переходим на вкладку «Данные».

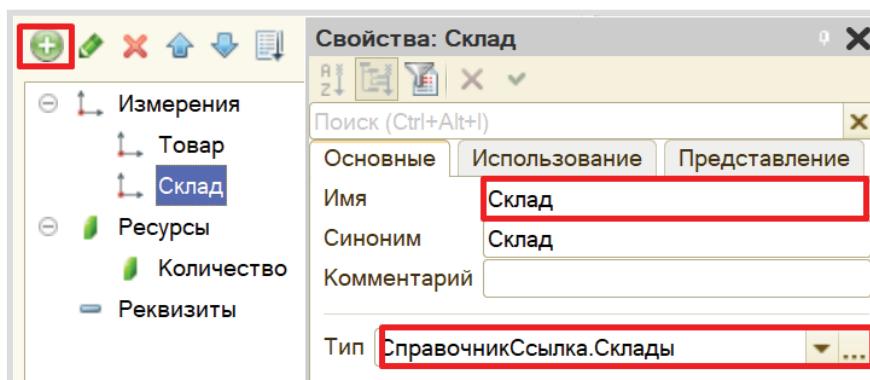
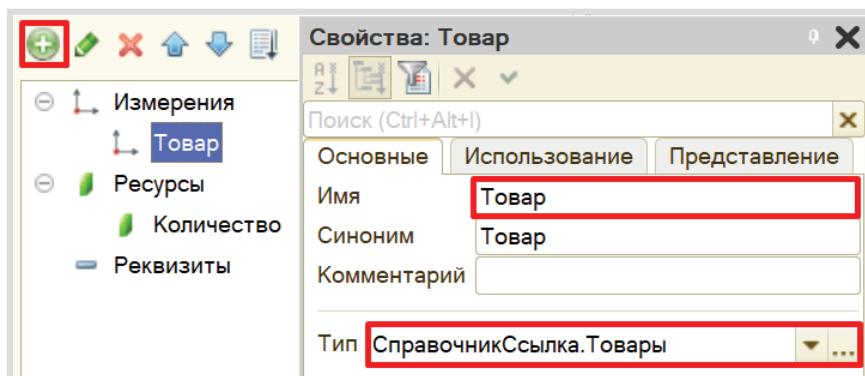


Структура *регистра накопления* отличается от структуры документа.

Заполнение данного окна проще всего начинать с добавления ресурса. Чтобы понять, что использовать в качестве ресурса, следует задать вопрос: «Что мы хотим накапливать/считывать в данном регистре?». Мы хотим считывать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число». Точность – «3», поскольку в реквизит должно попадать количество с точностью до грамм.



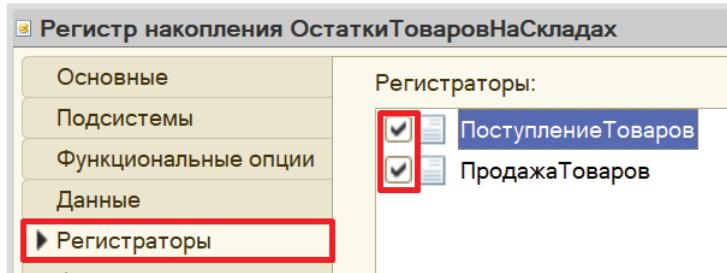
Чтобы разобраться с измерением, нужно понять, в разрезе чего мы хотим считать количество. Мы хотим считать количество (чего?) товаров в разрезе (чего?) складов. Значит, в качестве измерения необходимо добавить реквизиты «Товар» (тип – «СправочникСсылка.Товары») и «Склад» (тип – «СправочникСсылка.Склады»).



Чтобы регистр накопления заработал, следует сделать следующее:

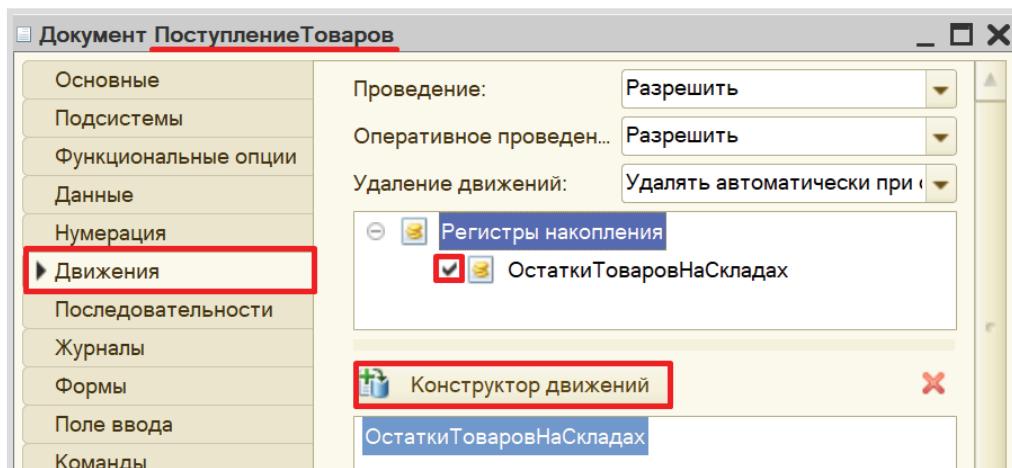
1. Определить источники данных регистра (определить документы-регистраторы).
2. Описать, каким образом данные из документа-регистратора должны попадать в регистр.

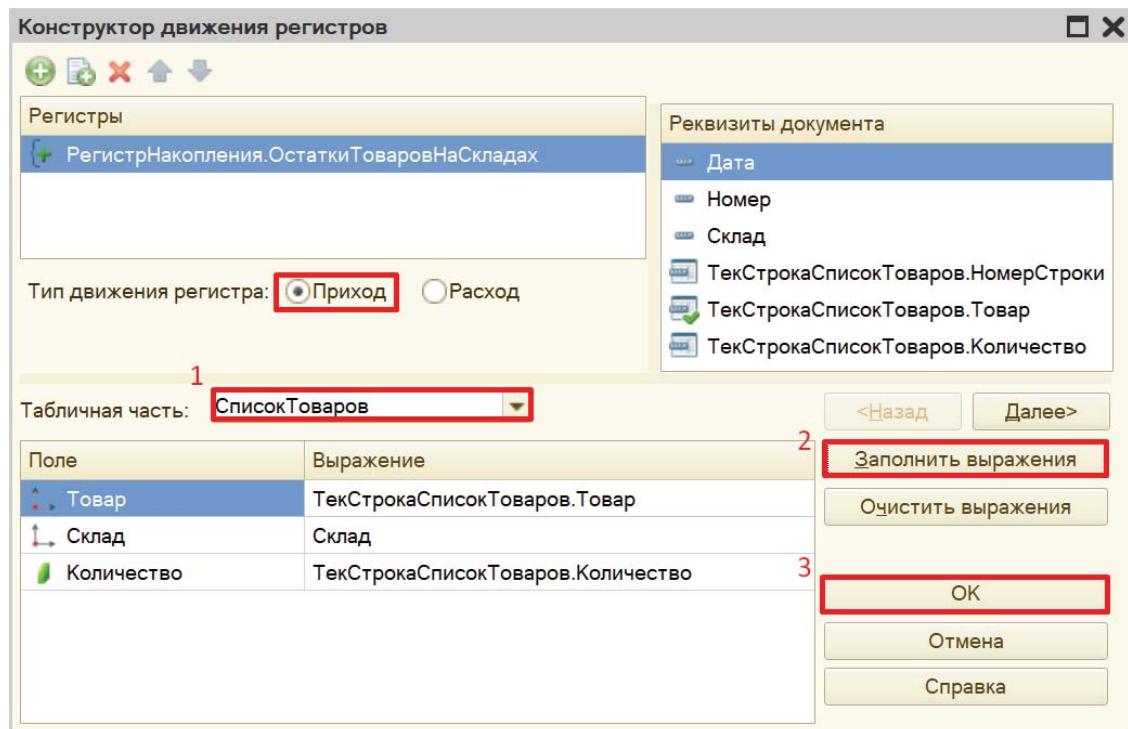
В нашем случае, на количество товаров будут влиять оба созданных документа. Определим их в качестве документов-регистраторов на вкладке «Регистраторы».



Далее для каждого из этих документов нужно описать процедуру копирования данных в регистр накопления.

Начнем с документа «Поступление Товаров», откроем окно редактирования данного документа на вкладке «Движения». Воспользуемся конструктором движений.





Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистрах).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части, нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты *регистра накопления*. Следует заполнить поле «Выражение» реквизитами документа.

Поскольку получение товара должно увеличивать количество товаров на складе, то тип движения регистра необходимо выбрать «Приход». Регистр будет обозначаться знаком «+» (плюс).

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.

При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.

Документ ПоступлениеТоваров: Модуль объекта

```

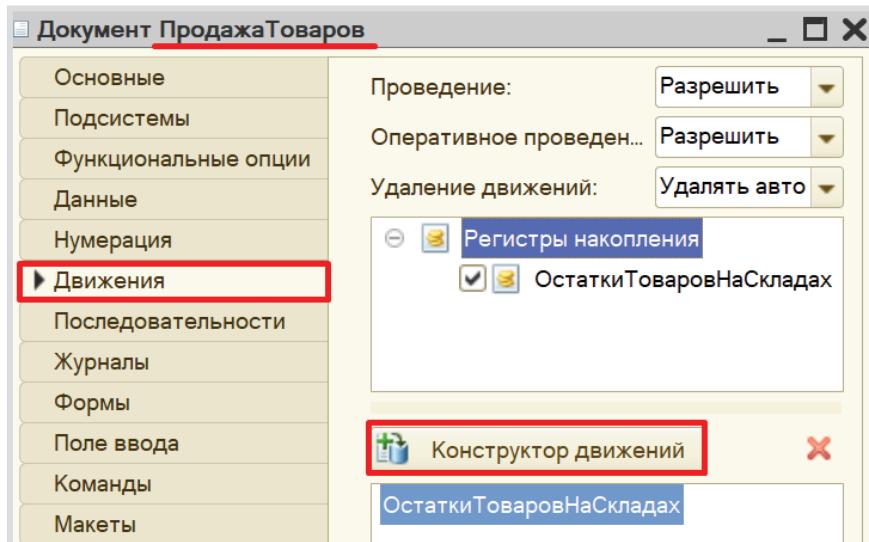
Процедура ОбработкаПроведения (Отказ, Режим)
// { __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
//  Данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную

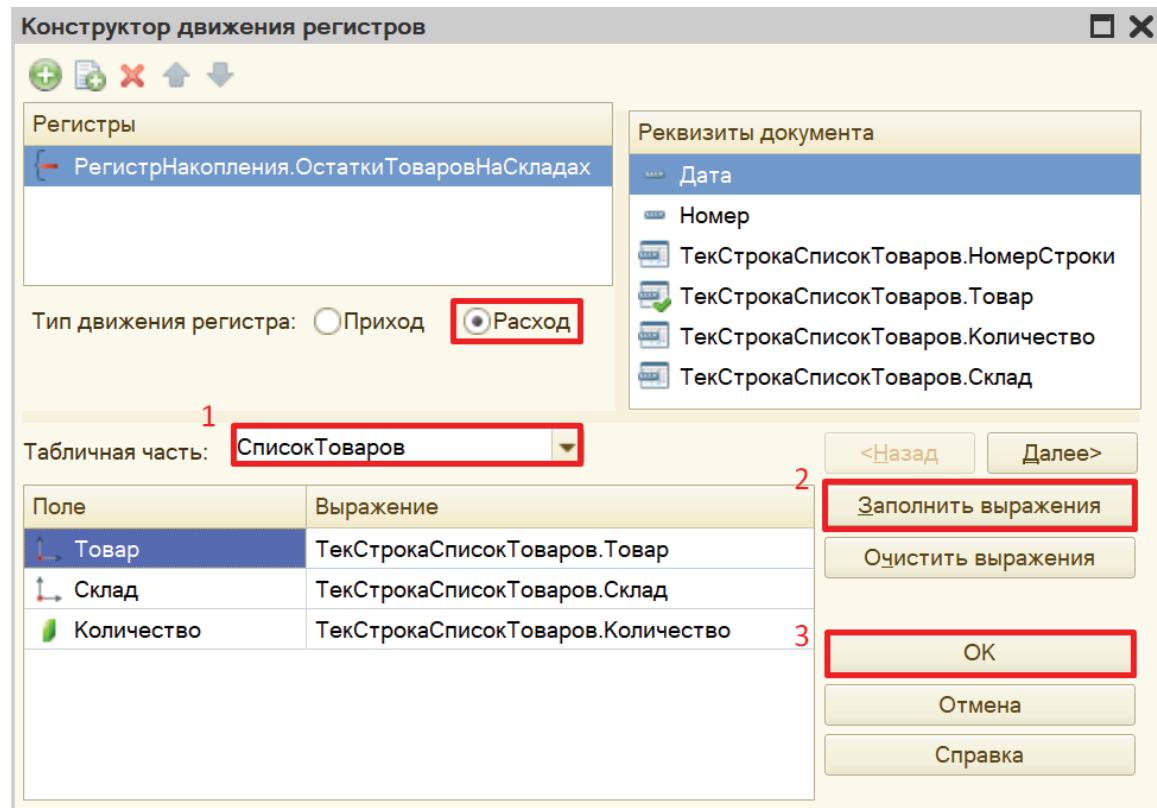
//  регистр ОстаткиТоваровНаСкладах Приход
Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваровНаСкладах.Добавить ();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.Склад = Склад;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;

// } __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

Аналогичные действия нужно проделать с документом «ПродажаТоваров».





Продажа товара должна уменьшать количество товаров на складе, значит, тип движения регистра следует выбрать «Расход». Регистр будет обозначаться знаком «-» (минус).

```
Документ ПродажаТоваров: Модуль объекта

Процедура ОбработкаПроведения (Отказ, Режим)
//{{ __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
//  Данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную.

//  регистр ОстаткиТоваровНаСкладах Расход
Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
Для Каждого ТекСтроКаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваровНаСкладах.Добавить ();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтроКаСписокТоваров.Товар;
    Движение.Склад = ТекСтроКаСписокТоваров.Склад;
    Движение.Количество = ТекСтроКаСписокТоваров.Количество;
КонецЦикла;

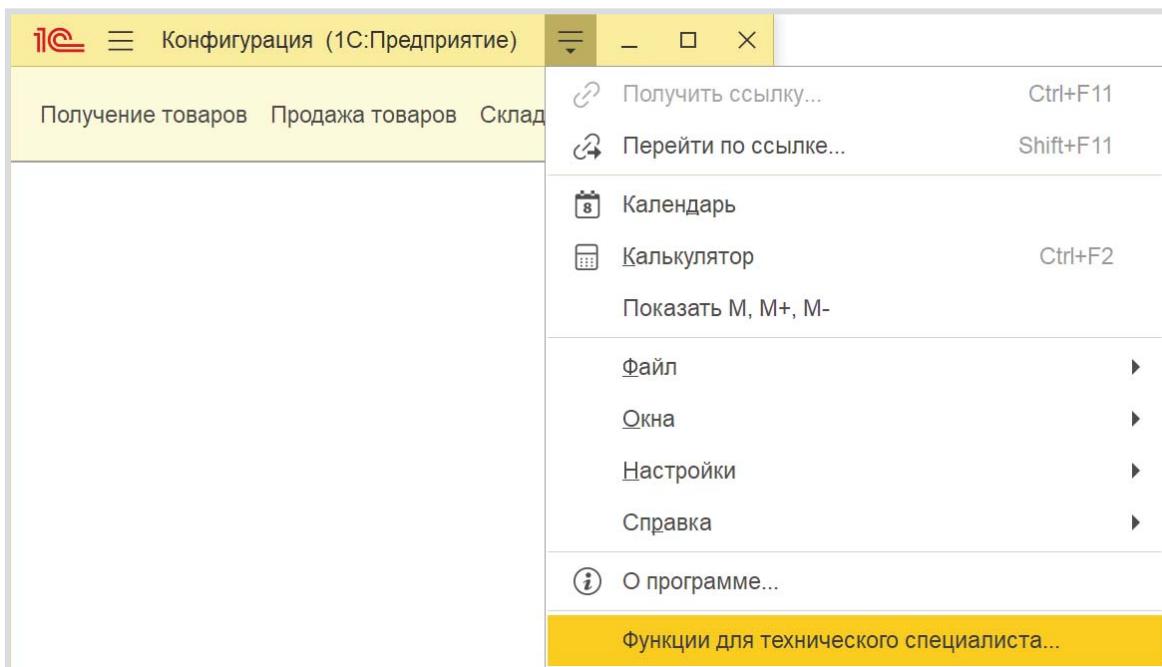
//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Откроем систему в режиме «1С:Предприятие» и проверим работу *регистра накопления*.

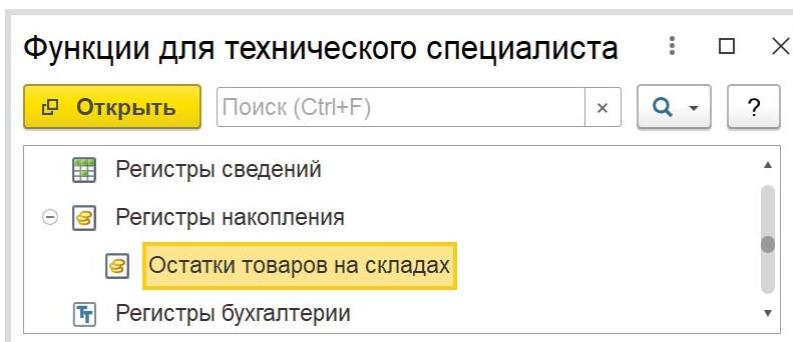
В первую очередь, необходимо перепровести (провести заново) созданный документ «Поступление товаров», а также создать и провести хотя бы один документ «Продажа товаров». Без проведения документов данные не будут скопированы в *регистр накопления*.

Обратите внимание, что на главной странице системы не создала кнопку открытия регистра накопления. Это связано с тем, что все расчеты в регистрах накопления происходят в фоновом режиме, «за кадром», то есть пользователю о них знать не нужно вовсе. Поэтому по умолчанию регистры накопления настраивают так, чтобы пользователи не имели к ним доступа.

Но мы, будучи разработчиками, можем обратиться к любому объекту конфигурации. Для этого воспользуемся функциями для технического специалиста.



В открывшемся списке найдем созданный нами *регистр накопления* и откроем его.



Период	↓	Регистратор	Номер строки	Товар	Склад	Количество
+ 08.09.2020 0:00:00		Поступление товаров...	1	Вилка	Северный	10,000
+ 08.09.2020 0:00:00		Поступление товаров...	2	Ложка	Северный	15,000
+ 08.09.2020 0:00:00		Поступление товаров...	3	Поварешка	Северный	17,000
- 08.09.2020 0:00:00		Продажа товаров 00...	1	Поварешка	Северный	5,000
- 08.09.2020 0:00:00		Продажа товаров 00...	2	Ложка	Южный	3,000
- 08.09.2020 0:00:00		Продажа товаров 00...	3	Вилка	Западный	7,000

Регистр накопления является некоторой итоговой таблицей. Сюда заносятся данные из документов-регистраторов по определенным правилам.

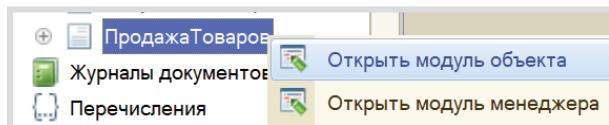
Таким образом, мы соединили между собой созданные ранее документы:

- документ «Поступление Товаров» увеличивает количество товаров на складах;
- документ «Продажа Товаров», наоборот, его уменьшает;
- информация обо всех движениях товаров дублируется в *регистр накопления*.

«Продать товар "в минус" нельзя, то есть в момент продажи следует проверять остаток товара».

К сожалению, *регистра накопления* недостаточно для того, чтобы вести учет отрицательных остатков. Необходимо описать алгоритм работы документа «Продажа Товаров».

Откроем модуль объекта документа «Продажа Товаров» и дополним процедуру «Обработка Проведения».



Проверять остатки товаров будем следующим образом:

1. Сделаем движение данных из документа в *регистр накопления*.
2. Проверим, появились ли в регистре остатки, значение которых меньше нуля (то есть отрицательные).
3. Если есть отрицательные остатки, то отменим сделанное движение в *регистр накопления* и выведем пользователю сообщение об ошибке.

Чтобы сделать движение данных из документа в *регистр накопления*, допишем после окончания цикла строки «Движения.Записать();». Метод записывает только те движения документа, у которых установлен флаг «Записывать», при этом флаг в итоге снимается, что не приводит к повторной записи движений по окончании транзакции проведения. И главное, «Движения.Записать();» всегда записывают движения в том порядке, в котором таблицы указаны в дереве метаданных, что на порядок уменьшает шансы взаимных блокировок, ведь все транзакции в одинаковом порядке блокируют таблицы.

Документ ПродажаТоваров: Модуль объекта

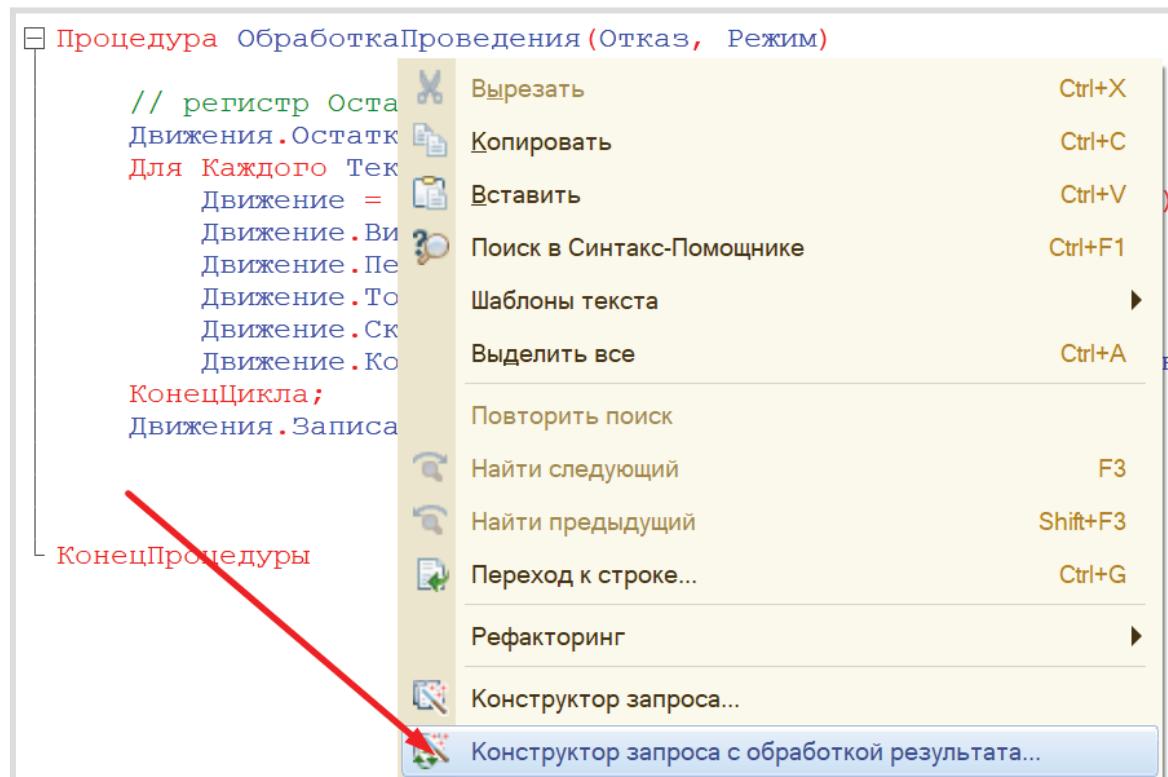
```

    Процедура ОбработкаПроведения(Отказ, Режим)
        // регистр ОстаткиТоваровНаСкладах Расход
        Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
        Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
            Движение = Движения.ОстаткиТоваровНаСкладах.Добавить();
            Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
            Движение.Период = Дата;
            Движение.Товар = ТекСтрокаСписокТоваров.Товар;
            Движение.Склад = ТекСтрокаСписокТоваров.Склад;
            Движение.Количество = ТекСтрокаСписокТоваров.Количество;
        КонецЦикла;
        Движения.Записать();
    КонецПроцедуры

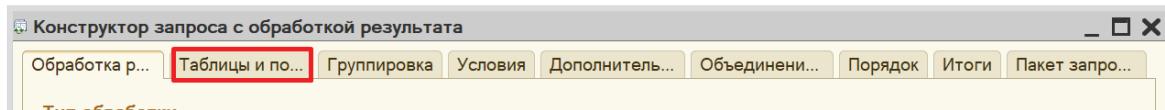
```

Теперь, когда движение было сделано, можно обратиться к данным *регистра накопления*.

Чтобы это сделать, воспользуемся *конструктором запроса с обработкой результата*. Этот конструктор можно открыть из контекстного меню щелчком правой кнопки мыши по области модуля. Данный конструктор обязательно должен быть вызван внутри процедуры «ОбработкаПроведения».



Соглашаемся с созданием нового запроса. Открывается окно *конструктора запроса с обработкой результата*. Переходим на вкладку «Таблицы и поля».



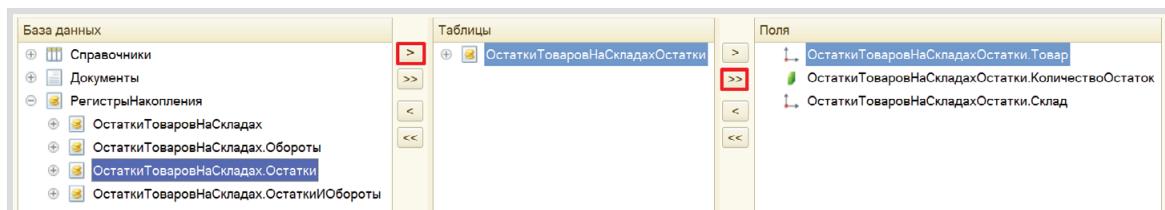
Открывшееся окно имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного запроса.
- Справа поля – это те значения (поля), которые мы хотим получить.

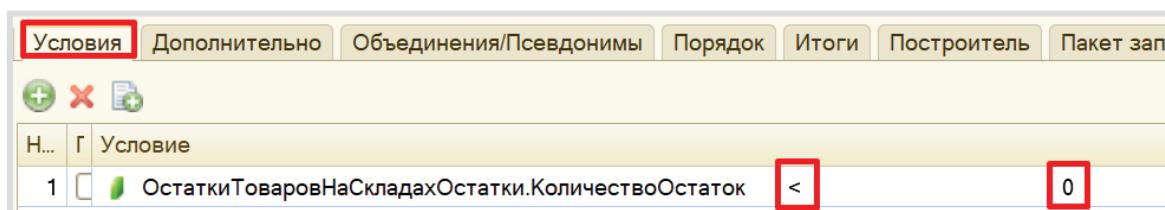
Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица способна обработать основную таблицу и самостоятельно посчитать остатки товаров.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши, либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



Переходим на вкладку «Условие» и добавим новое условие. Пусть в запрос попадут только данные с отрицательными остатками.



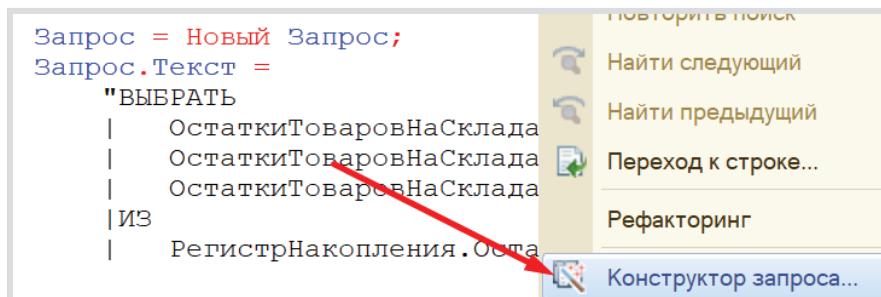
По окончании работы с данным конструктором нажимаем на кнопку «OK». Конструктор выдаст предупреждение об ошибке, которое следует проигнорировать. *Обратите внимание* на текст запроса. Необходимо удалить знак амперсанта (&) перед нулем в условии.

Запрос должен выглядеть так:

```
"ВЫБРАТЬ
| ОстаткиТоваровНаСкладахОстатки.Товар КАК Товар,
| ОстаткиТоваровНаСкладахОстатки.Склад КАК Склад,
| ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток КАК КоличествоОстаток
| ИЗ
| РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки КАК ОстаткиТоваровНаСкладахОстатки
| ГДЕ
| ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток < 0"
```

С помощью данного запроса из базы данных можно получить отрицательные остатки по всем товарам со всех складов. Но нам нет необходимости получать такую большую выборку, нужно сузить запрос до склада, который указан в шапке документа и товаров, перечисленных в табличной части.

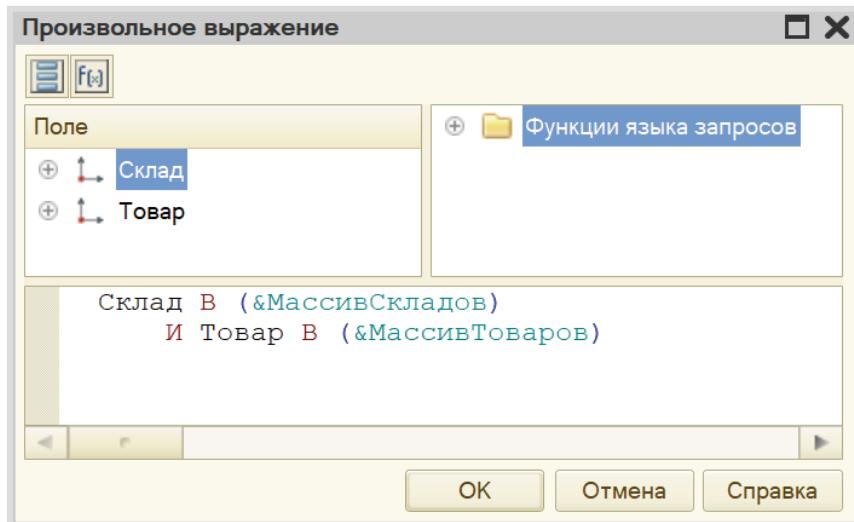
Откроем *конструктор запроса*. Для этого нужно щелкнуть в любом месте самого запроса (черный текст в двойных кавычках) правой кнопкой мыши и вызвать *конструктор запроса*.



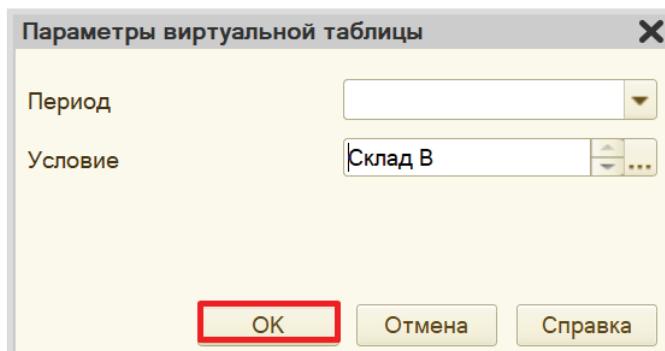
Следует наложить условия на виртуальную таблицу *регистра накопления*.

В открывшемся окне нужно написать следующее выражение:

Склад В (&МассивСкладов) И Товар В (&МассивТоваров)



Нажмите на кнопку «OK».



Данное условие поможет ограничить запрос по складам и товарам, которые указаны в табличной части документа.

Нажимаем на кнопку «OK». Текст запроса изменился:

```
Запрос.Текст =
"ВЫБРАТЬ
| ОстаткиТоваровНаСкладахОстатки.Товар КАК Товар,
| ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток КАК КоличествоОстаток,
| ОстаткиТоваровНаСкладахОстатки.Склад КАК Склад
| ИЗ
|   РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки(
|
|       Склад В (&МассивСкладов)
|       И Товар В (&МассивТоваров) ) КАК ОстаткиТоваровНаСкладахОстатки
| ГДЕ
|   ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток < 0";
"
```

Мы добавили параметры. Теперь запрос будет проводить поиск только по конкретному складу и конкретному списку товаров. Осталось лишь указать этот склад и эти товары сразу после текста запроса.

```
| ГДЕ
|   ОстаткиТоваровНаСкладах.Остатки.Количество.Остаток < 0";
Запрос.УстановитьПараметр ("МассивСкладов", СписокТоваров.ВыгрузитьКолонку ("Склад"));
Запрос.УстановитьПараметр ("МассивТоваров", СписокТоваров.ВыгрузитьКолонку ("Товар"));
```

Ну, и последний шаг – выдать сообщение пользователю, если запрос вернул отрицательные остатки. В первую очередь, добавим блок условия сразу после определения параметров.

```
РезультатЗапроса = Запрос.Выполнить ();
Если НЕ РезультатЗапроса.Пустой () Тогда
    КонецЕсли;
```

Внутрь цикла можно попасть только в том случае, если запрос пришел не пустой, то есть если были найдены отрицательные остатки. В таком случае нужно отменить проведение документа и выдать пользователю сообщение.

```
РезультатЗапроса = Запрос.Выполнить ();
Если НЕ РезультатЗапроса.Пустой () Тогда
    Отказ = Истина;
    Выборка = РезультатЗапроса.Выбрать ();
    Пока Выборка.Следующий () Цикл
        Сообщить ("На складе " + Выборка.Склад + " не хватает " +
                    Выборка.Количество.Остаток*(-1) + " шт. товаров " + Выборка.Товар);
    КонецЦикла;
КонецЕсли;
```

Код процедуры полностью должен выглядеть следующим образом:

```
Процедура ОбработкаПроведения (Отказ, Режим)
    // регистр ОстаткиТоваровНаСкладах Расход
    Движения.ОстаткиТоваровНаСкладах.Записывать = Истина;
    Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
        Движение = Движения.ОстаткиТоваровНаСкладах.Добавить ();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Товар = ТекСтрокаСписокТоваров.Товар;
        Движение.Склад = ТекСтрокаСписокТоваров.Склад;
        Движение.Количество = ТекСтрокаСписокТоваров.Количество;
    КонецЦикла;
    Движения.Записать ();
```

```

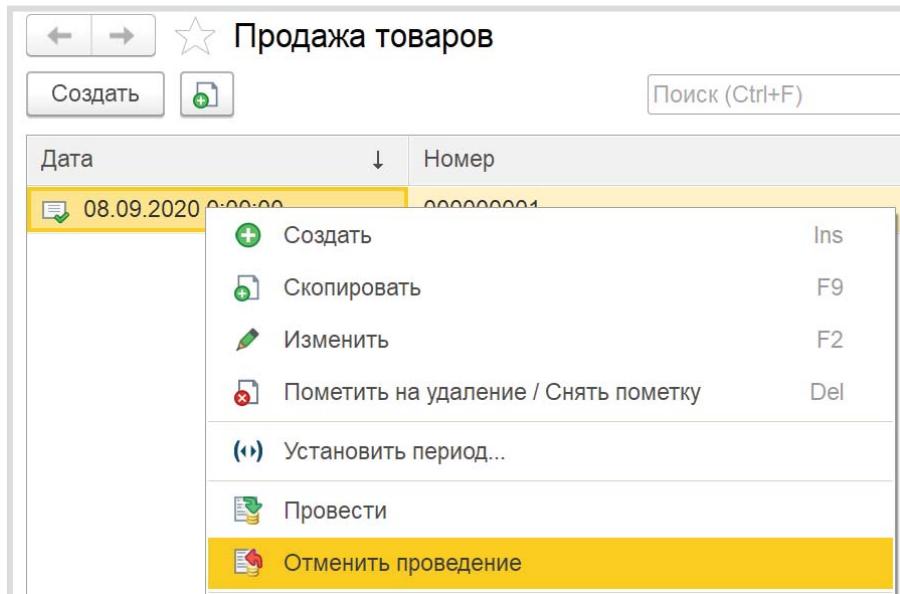
// контроль отрицательных остатков
Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
| ОстаткиТоваровНаСкладахОстатки.Товар КАК Товар,
| ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток КАК КоличествоОстаток,
| ОстаткиТоваровНаСкладахОстатки.Склад КАК Склад
| ИЗ
|     РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки (
|
|         , Склад В (&МассивСкладов)
|             И Товар В (&МассивТоваров) ) КАК ОстаткиТоваровНаСкладахОстатки
| ГДЕ
|     ОстаткиТоваровНаСкладахОстатки.КоличествоОстаток < 0";
Запрос.УстановитьПараметр ("МассивСкладов", СписокТоваров.ВыгрузитьКолонку ("Склад"));
Запрос.УстановитьПараметр ("МассивТоваров", СписокТоваров.ВыгрузитьКолонку ("Товар"));

РезультатЗапроса = Запрос.Выполнить ();
Если НЕ РезультатЗапроса.Пустой () Тогда
    Отказ = Истина;
    Выборка = РезультатЗапроса.Выбрать ();
    Пока Выборка.Следующий() Цикл
        Сообщить ("На складе " + Выборка.Склад + " не хватает " +
        Выборка.КоличествоОстаток*(-1) + " шт. товаров " + Выборка.Товар);
    КонецЦикла;
КонецЕсли;
КонецПроцедуры

```

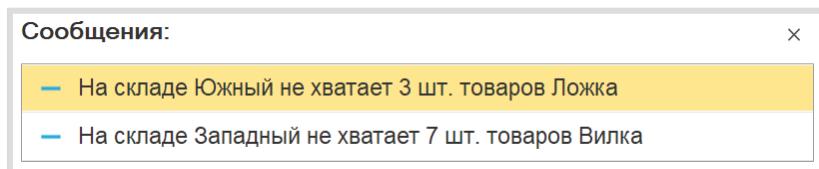
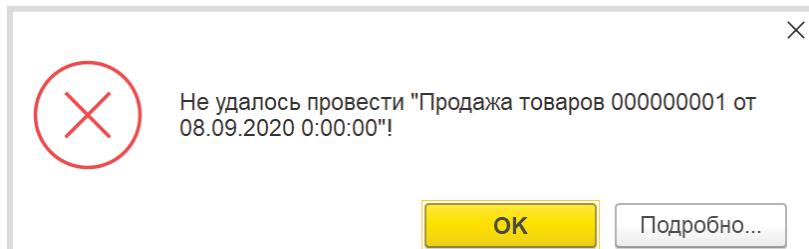
Переходим в режим «1С:Предприятие».

Если у вас уже имеются документы «ПродажаТоваров», отмените их проведение. Для этого нажмите по документу в списке правой кнопкой мыши и выберите «Отменить проведение». Сделайте это для всех документов.



Проведите документы заново. Добавьте новый документ «Продажа товаров» так, чтобы хотя бы одного из вида товаров не хватало на одном из складов.

Если все было сделано правильно, и вы пытаетесь продать товаров больше, чем имеется на складе, то система выдаст ошибку:



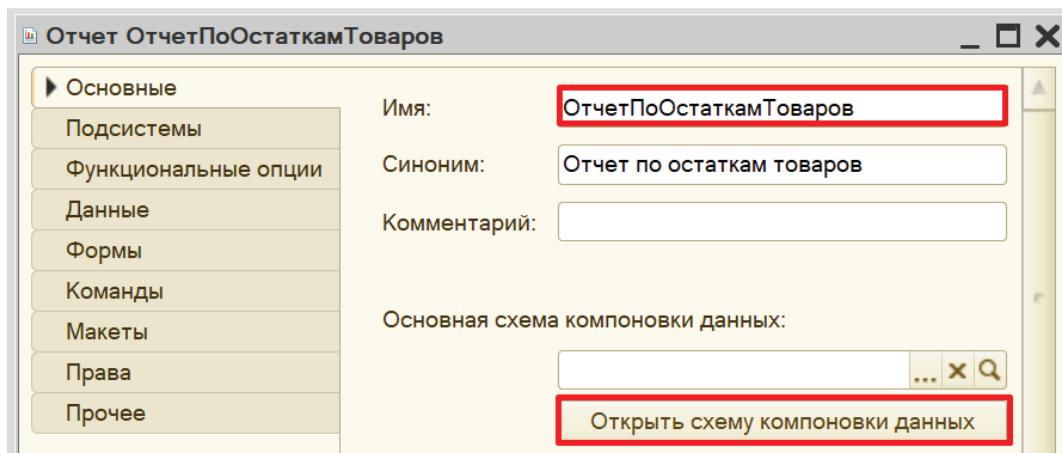
«Нужно построить отчет по остаткам товаров».

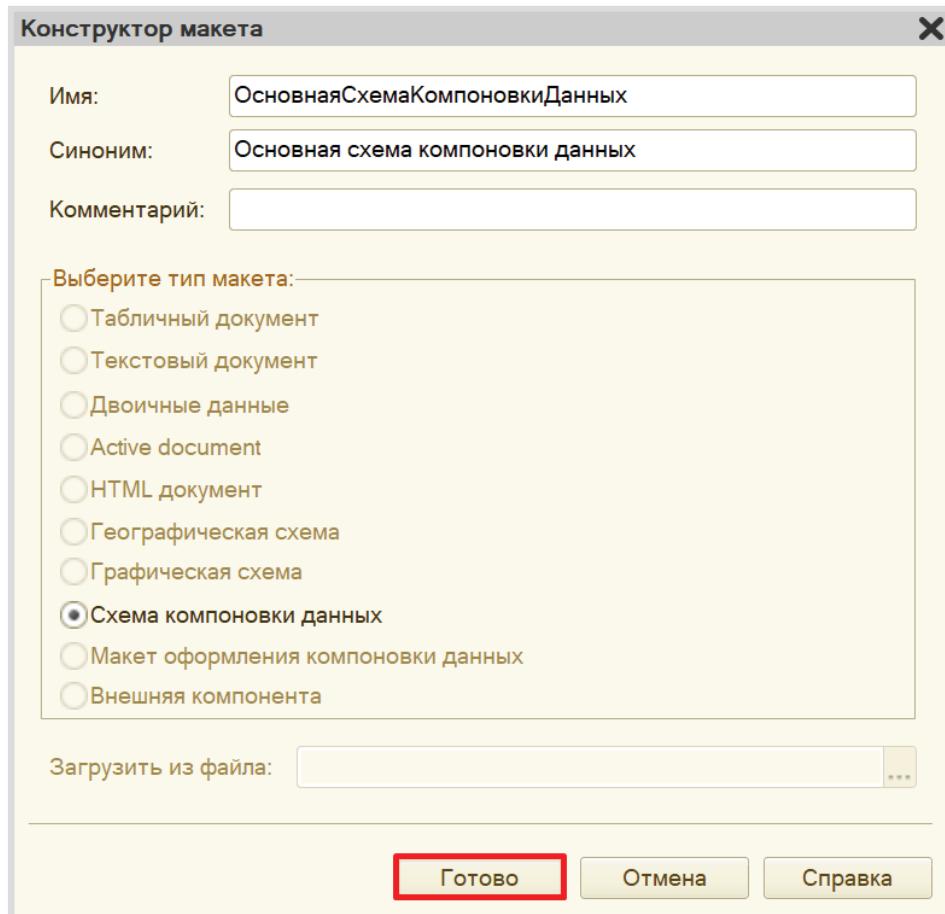
Построим отчет. Для этого воспользуемся соответствующим объектом конфигурации.

Определение

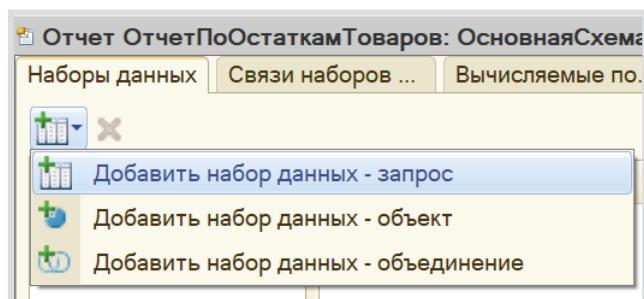
Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: <https://v8.1c.ru/platforma/otchet/>).

Добавим отчет «ОтчетПоОстаткамТоваров». Воспользуемся схемой компоновки данных.

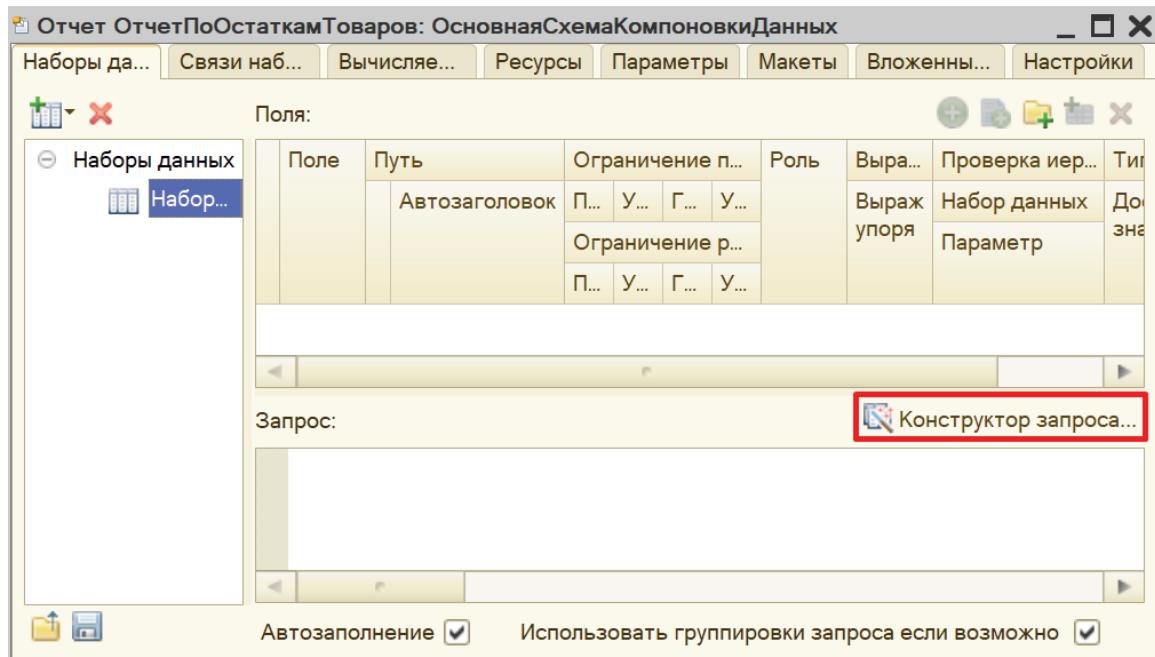




Добавим новый запрос к базе данных.



Для формирования запроса воспользуемся конструктором запроса.



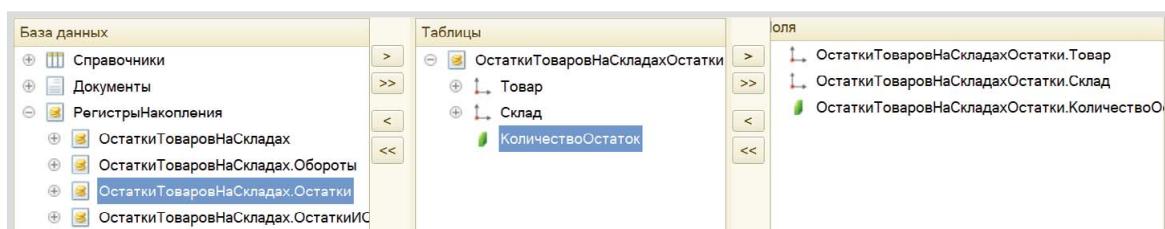
Открывается конструктор запроса. Эта вкладка имеет три части:

- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного отчета.
- Справа поля – это те значения (поля), которые мы хотим увидеть в отчете.

Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица позволит получить уже просуммированные значения по всем документам.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

В результате данное окно должно быть заполнено следующим образом:



Закрываем конструктор запроса, нажав на кнопку «OK». Сформировался запрос.

```

Запрос:
ВЫБРАТЬ
    ОстаткиТоваровНаСкладах.Остатки.Товар КАК Товар,
    ОстаткиТоваровНаСкладах.Остатки.Склад КАК Склад,
    ОстаткиТоваровНаСкладах.Остатки.Количество.Остаток КАК Количество.Остаток
ИЗ
    РегистрНакопления.ОстаткиТоваровНаСкладах.Остатки КАК ОстаткиТоваровНаСкладах.Остатки

```

Теперь система понимает, какие данные ей нужны для формирования отчета.

Переходим на вкладку «Ресурсы» и устанавливаем реквизит «Количество.Остаток» в качестве ресурса: это позволит нам в отчете получать итоговые (просуммированные) значения.

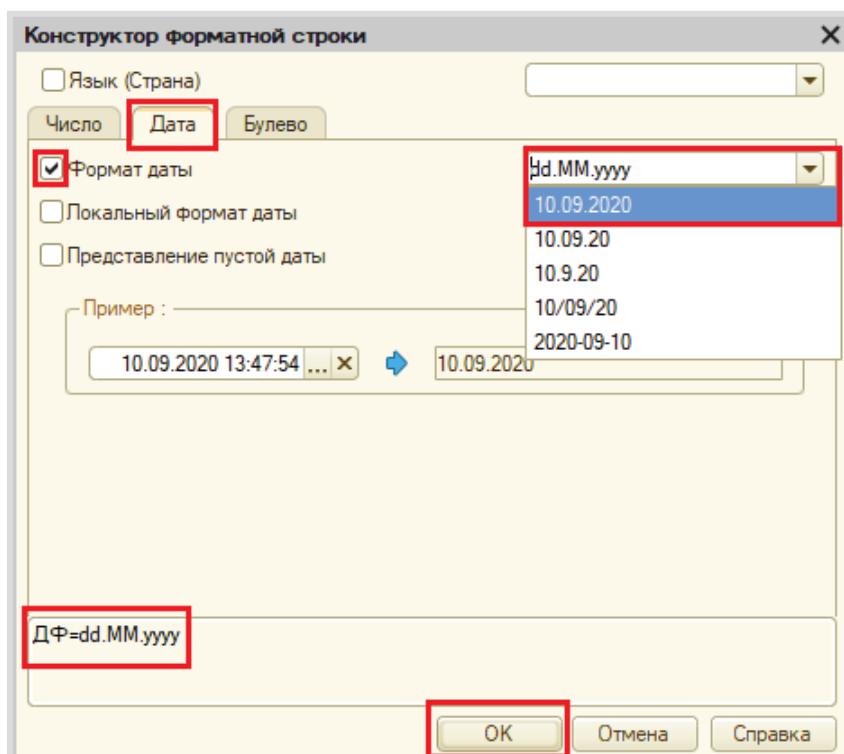
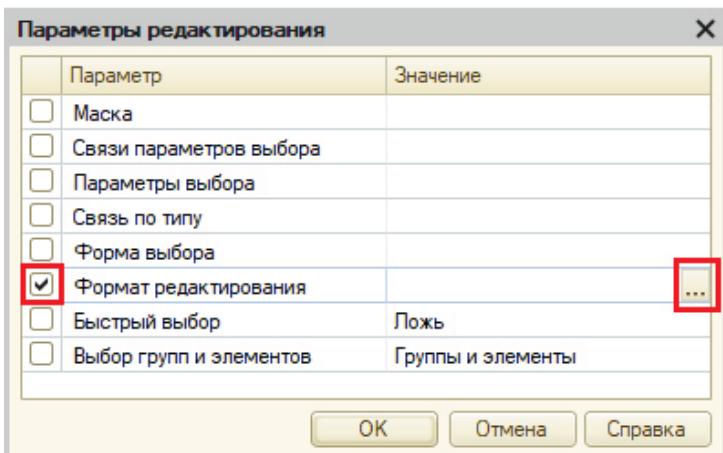
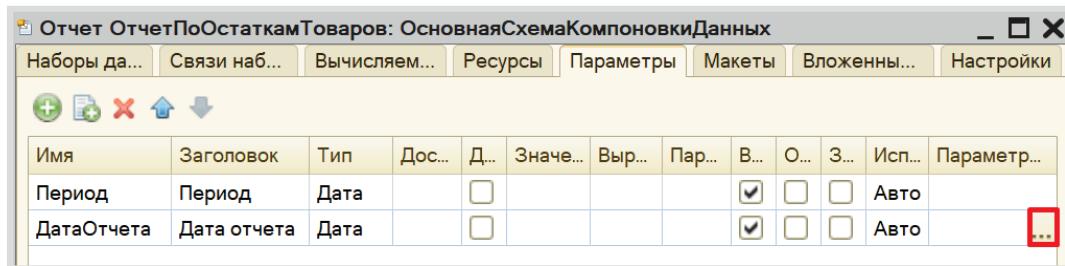
Ресурсы	Параметры	Макеты	Вложенные схемы	Настройки
Поле	Выражение	Рассчитывать		
Количество.Остаток	Сумма(Количество.Остаток)			

«Отчет строится на конец дня, указанного пользователем. Особое внимание следует уделить последней секунде дня: документы, записанные на эту секунду, должны попадать в отчет».

Из условия следует, что отчет должен включать документы, записанные на последнюю секунду дня. При использовании стандартных методов такие документы в отчет попадать не будут. Поэтому нужно добавить новый параметр «Дата.Отчета» на соответствующей вкладке.

Отчет ОтчетПоОстаткамТоваров: ОсновнаяСхема.Компоновка.Данных												
Наборы да...	Связи наб...	Вычисляем...	Ресурсы	Параметры	Макеты	Вложенные...	Настройки					
Имя	Заголовок	Тип	Дос...	Д...	Значе...	Выр...	Пар...	В...	О...	З...	Исп...	Параметр...
Период	Период	Дата		<input type="checkbox"/>				<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		Авто
Дата.Отчета	Дата отчета	Дата		<input type="checkbox"/>				<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		Авто

Чтобы у пользователя была возможность выбирать только даты, без указания секунд, нужно настроить формат редактирования параметра «Дата.Отчета».

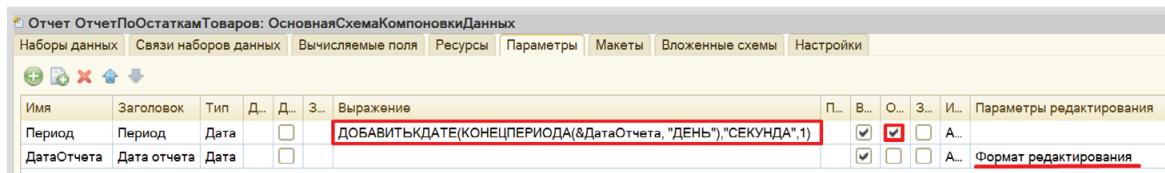


После нажатия кнопки «OK» нужно настроить стандартный параметр «Период» для корректного учета последней секунды дня:

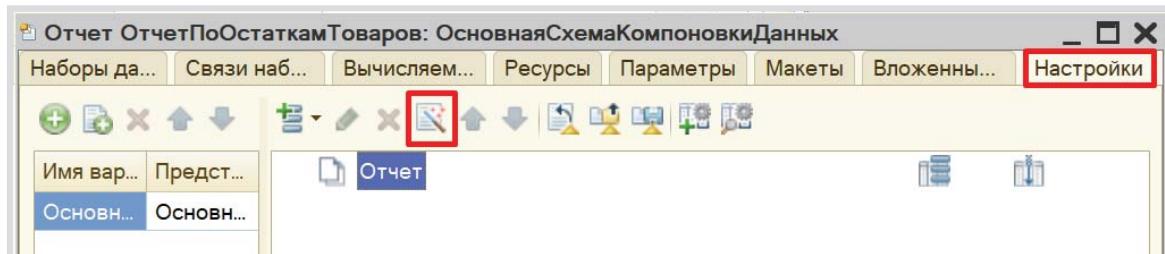
- во-первых, этот параметр должен быть недоступен пользователю, так как носит вычислительный характер;
- во-вторых, для корректного расчета требуется написать выражение для стандартного параметра «Период»:

ДОБАВИТЬКДАТЕ(КОНЕЦПЕРИОДА(&ДатаОтчета, "ДЕНЬ"),"СЕКУНДА",1)

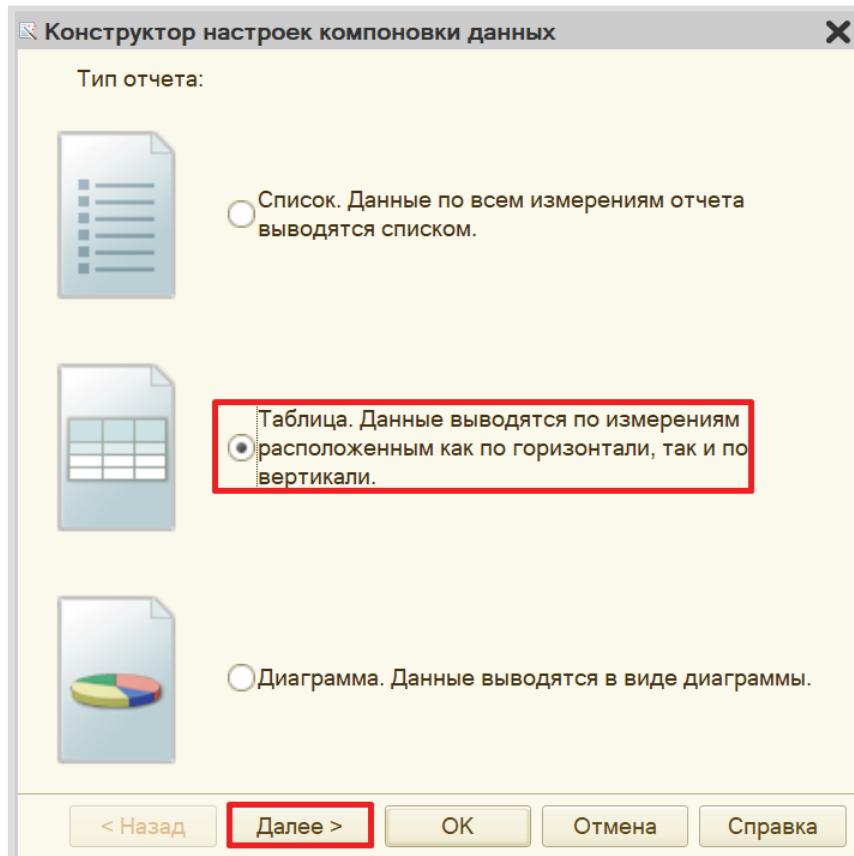
Сначала берется последний момент дня, указанного в параметре «ДатаОтчета», а затем прибавляется еще одна секунда, чтобы учитывались даже документы, проведенные за эту последнюю секунду.



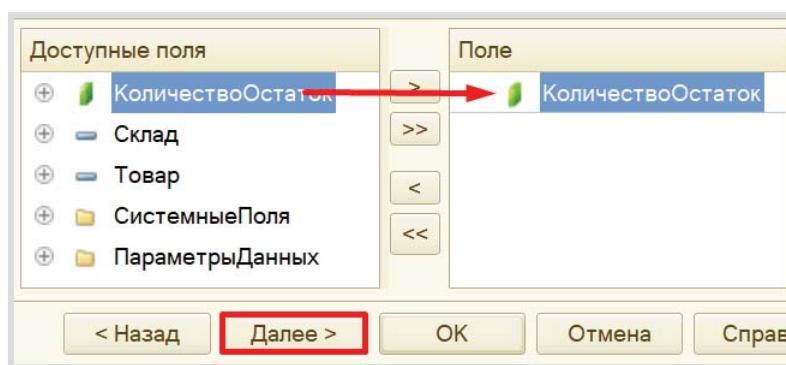
Теперь переходим на вкладку «Настройки» для оформления внешнего вида отчета. Воспользуемся конструктором настроек отчета.



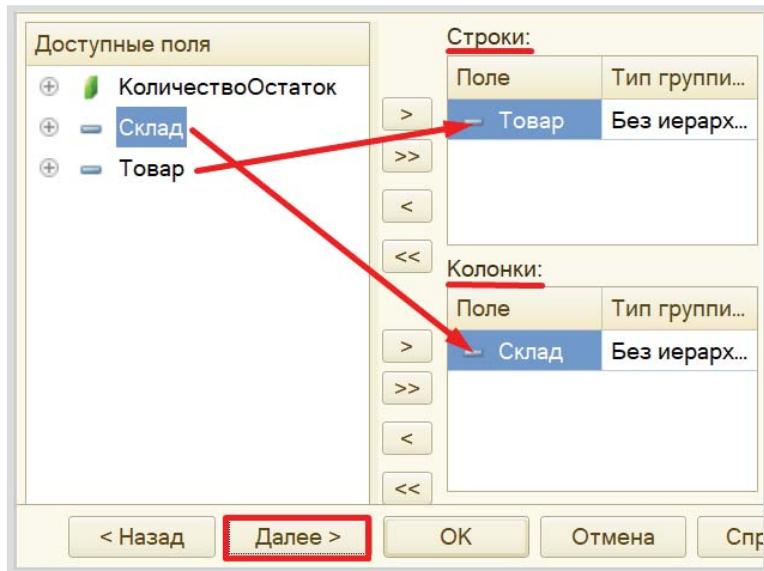
Построим отчет в виде таблицы.



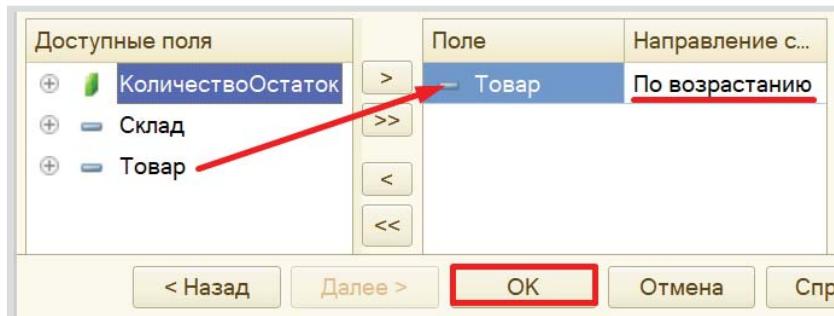
Далее нужно выбрать ресурс, который будет отображен в таблице. В данном случае реквизит у нас всего один, его и выбираем.



На следующем шаге следует определить те реквизиты, которыми будут заполняться колонки и строки таблицы.



Ну, и на последнем шаге нужно установить сортировку. Установим сортировку по товарам по возрастанию (то есть по алфавиту).



Чтобы у пользователя была возможность выбирать день, на который он хочет построить отчет, нужно включить параметр «ДатаОтчета» в пользовательские настройки.

The screenshot shows the configuration of a report. In the left pane, under 'Отчет' (Report), 'Таблица' (Table) is selected. Under 'Строки' (Rows), 'Товар' (Product) is checked. Under 'Колонки' (Columns), 'Склад' (Warehouse) is checked. Below this, the 'Настройки' (Settings) tab is selected, showing the 'Отчет' (Report) tab is active. A red box highlights the 'Параметры' (Parameters) button. The parameters table shows a row for 'Дата отчета' (Report date) set to 'Произвольная дата' (Any date). A red box highlights this row. To the right of the table are icons for copy, cut, paste, and add. A red box highlights the 'Добавить' (Add) icon.

Пользовательские настройки элемента

Включать в пользовательские настройки

Представление

Режим редактирования

OK **Отмена** **Справка**

Отчет готов. Запустим систему в режиме «1С:Предприятие».

Добавьте еще несколько отчетов по получению товаров и продаже товаров, чтобы убедиться, что отчет работает корректно.

The window title is 'Отчет по остаткам товаров'. It has buttons for 'Сформировать' (Formulate), 'Выбрать вариант...' (Select variant...), and 'Настройки...' (Settings...). A date filter 'Дата отчета:' is set to '08.09.2020'. The main area displays a table of product inventories:

Товар	Западный	Северный	Южный	Итого
	Количество	Количество	Количество	
Остаток	Остаток	Остаток	Остаток	
Вилка	6,000	10,000	20,000	36,000
Ложка	5,000	15,000	20,000	40,000
Поварешка	4,000	17,000	20,000	41,000
Итого	15,000	42,000	60,000	117,000

Поставленная задача решена.

Лабораторная работа № 20

**РАЗРАБОТКА КОНФИГУРАЦИИ
ДЛЯ УЧЕТА ТОВАРОВ.
КОНТРОЛЬ СРОКА ГОДНОСТИ
ТОВАРОВ**

Сложность: ***

Теги: справочник, документ, регистр накопления,
обработка проведения, запрос, макет отчета

ЗАДАНИЕ

Заказчик просит разработать конфигурацию для учета товаров.

Многоскладской учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся.

В системе необходимо регистрировать поступление товара. При поступлении товара пользователь в табличной части указывает, какие товары и в каком количестве поступили в организацию. Следует предусмотреть учет до граммов. При поступлении товаров указывается срок годности партии, для каждого товара – свой.

В системе нужно регистрировать продажу товара. При продаже товаров указывается, какие товары были проданы, в каком количестве, на какую сумму. При продаже товара необходимо, в первую очередь, списывать те товары, срок годности которых подходит к концу.

К примеру, если поставка молока «Буренка» поступила со сроками годности 30.01.2020 и 31.01.2020, то сначала списывается партия со сроком годности 30.01.2020.

Продать товар «в минус» нельзя, в момент продажи необходимо проверять остаток товара.

Важно помнить, что пользователь может вводить документы задним числом!

В результате выполнения лабораторной работы должен получиться отчет вида:

Остатки товаров на 15.01.2020

Товар	Срок годности	Количество
Молоко "Буренка"	28.01.2020	10.000
Молоко "Буренка"	30.01.2020	15.000
Молоко "Буренка"	31.01.2020	5.000

Подготовка

- Создать новую информационную базу.
- Открыть информационную базу в режиме «Конфигуратор».
- Открыть дерево метаданных.

Подробнее о том, как это сделать,смотрите в Лабораторной работе № 2 (стр. 17).

Выполнение

«Многоскладской учет не ведется. Взаиморасчеты с покупателями и поставщиками не ведутся».

Из данного условия делаем вывод, что в информационной системе не нужно хранить каких-либо данных о складах, покупателях и поставщиках.

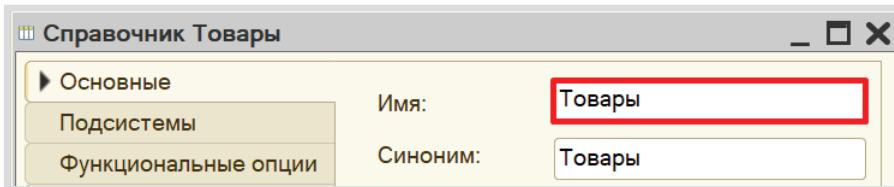
«Заказчик просит разработать конфигурацию для учета товаров».

Появляется необходимость в хранении объектов аналитики, а именно – товаров. Для хранения перечня товаров будем использовать *справочник*.

Определение

Справочник – это объект конфигурации, который хранит справочную информацию, например, список сотрудников, складов и т. д. (более подробно про справочники можно прочитать здесь: <https://v8.1c.ru/platforma/spravochniki/>).

Добавим справочник «Товары».



Таким образом, мы организовали хранение товаров в информационной системе.

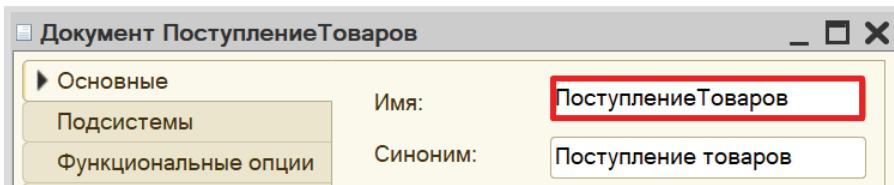
«В системе необходимо зарегистрировать поступление товара».

Для регистрации поступления товаров будем использовать *документ*.

Определение

Документ – это объект конфигурации, хранящий информацию о каких-либо событиях, произошедших в «жизни» предприятия. Например, с помощью документа можно зарегистрировать (то есть сохранить данные документа для последующей обработки) продажу товаров или начисление зарплаты (подробнее про документы можно прочитать здесь: <https://v8.1c.ru/platforma/dokumenty/>).

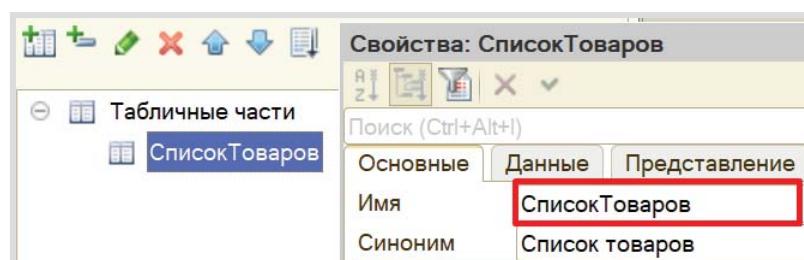
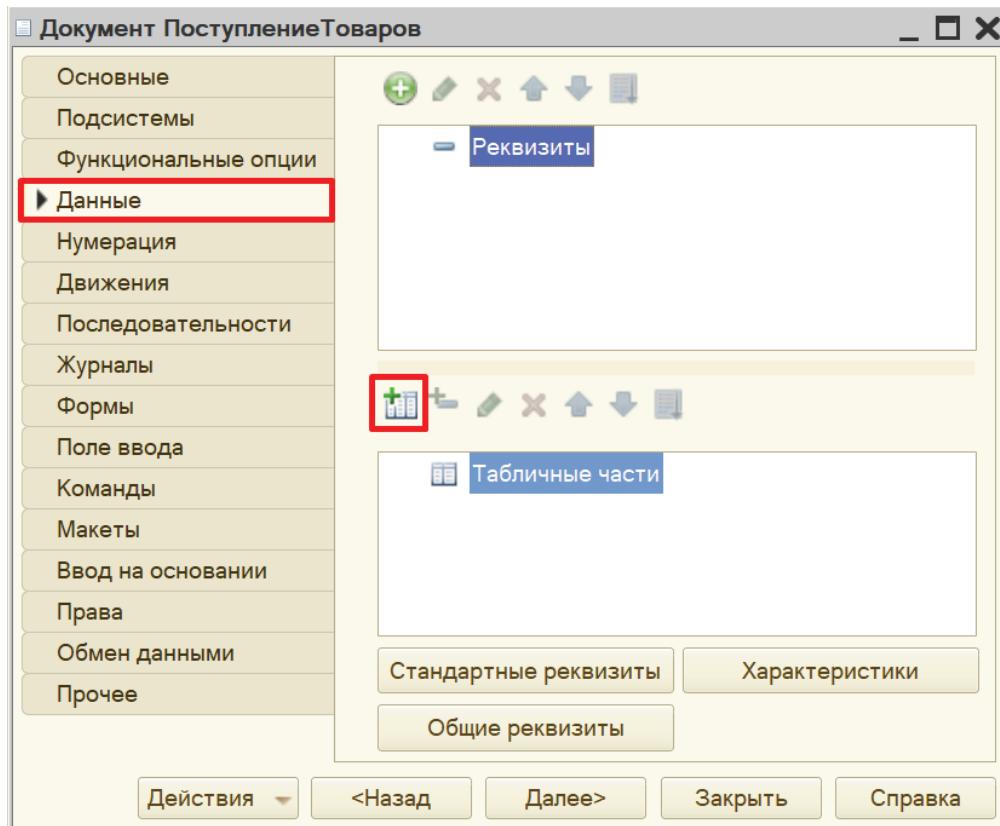
Добавим документ «Поступление Товаров».



Для настройки структуры документа следует перейти на вкладку «Данные».

«При поступлении товара пользователь в табличной части указывает...».

Добавим табличную часть документа.

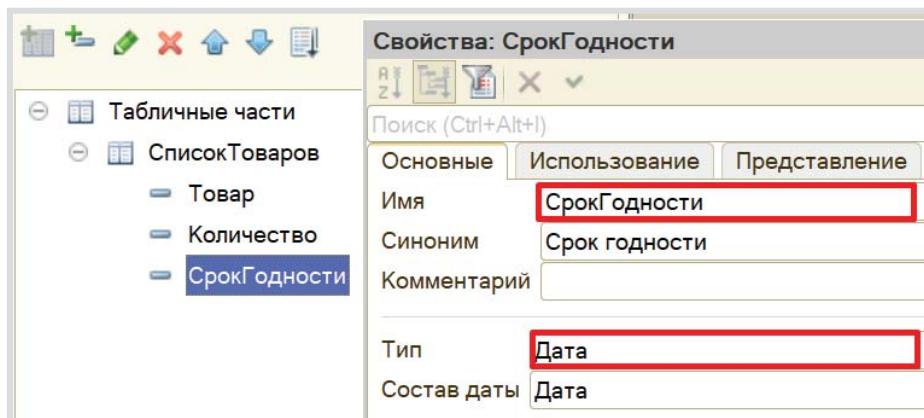


«...пользователь в табличной части указывает, какие товары и в каком количестве поступили в организацию. Следует предусмотреть учет до граммов. При поступлении товаров указывается срок годности партии, для каждого товара – свой».

Исходя из условия, табличная часть должна иметь три реквизита: «Товар», «Количество» и «СрокГодности». Добавим их.

The figure consists of three vertically stacked screenshots from the 1C:Enterprise software interface, illustrating the creation of document properties:

- Screenshot 1:** Shows the 'Свойства' (Properties) window for 'Товар' (Product). The 'Имя' (Name) field is set to 'Товар'. The 'Тип' (Type) field is set to 'СправочникСсылка.Товары' (DictionaryLink.Product). The 'Синоним' (Synonym) and 'Комментарий' (Comment) fields are also visible.
- Screenshot 2:** Shows the 'Свойства' (Properties) window for 'Количество' (Quantity). The 'Имя' (Name) field is set to 'Количество'. The 'Тип' (Type) field is set to 'Число' (Number). The 'Длина' (Length) field contains '10'. The 'Точность' (Precision) field contains '3'. The 'Неотрицательное' (Non-negative) checkbox is checked.
- Screenshot 3:** Shows the 'Свойства' (Properties) window for 'СписокТоваров' (List of Products). A new property 'Товар' (Product) has been added under it. This property has the same settings as the one in Screenshot 1.



Теперь в системе можно регистрировать поступление товаров.

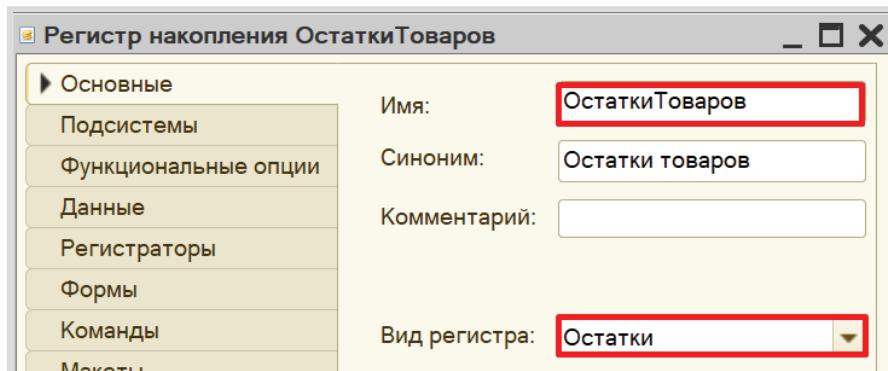
Далее нужно каким-то образом вести подсчет остатков товаров. Для этого нам потребуется *регистр накопления*.

Определение

Регистр накопления – это такая итоговая таблица, которая может автоматически считать какие-либо элементы, например, денежные средства, материалы (дополнительно про регистры накопления можно прочитать здесь: <https://v8.1c.ru/platforma/registr-nakopleniya/>).

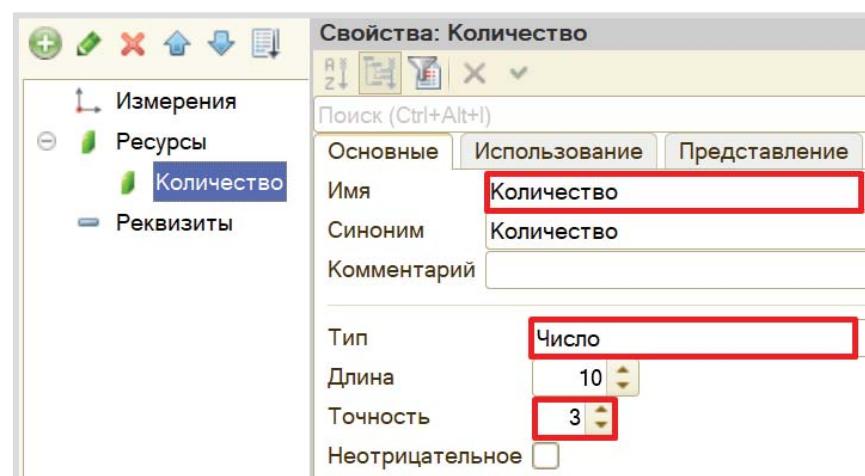
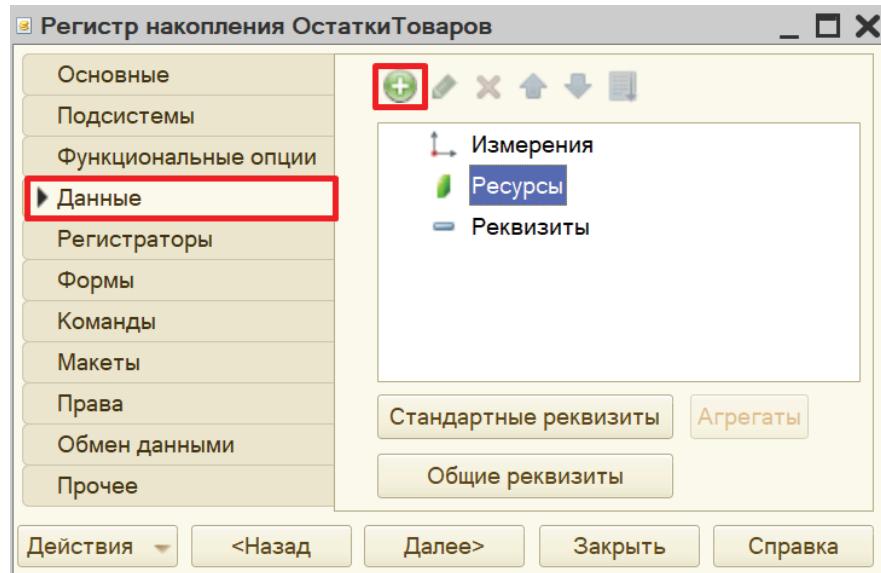
Добавим *регистр накопления* «ОстаткиТоваров» вида «Остатки».

Вид регистра «Остатки» позволяет настроить данный регистр таким образом, что какие-то объекты будут вносить в него данные, а какие-то, наоборот, вычитать. Таким образом и получается хранение остатков.



Открываем вкладку «Данные» для формирования структуры *регистра накопления*.

Добавим ресурс. Что мы хотим считать с помощью данного регистра? Мы хотим считать количество. Следовательно, количество и будет являться ресурсом. Тип данного реквизита – «Число».



Чтобы разобраться с измерением, следует понять, в разрезе чего мы хотим считать количество. Мы хотим учитывать, количество (чего?) товаров с (чем?) различным сроком годности. Значит, в качестве измерения нужно добавить два реквизита: «Товар» (тип – «СправочникСсылка.Товары») и «Срок годности» (тип – «Дата»).

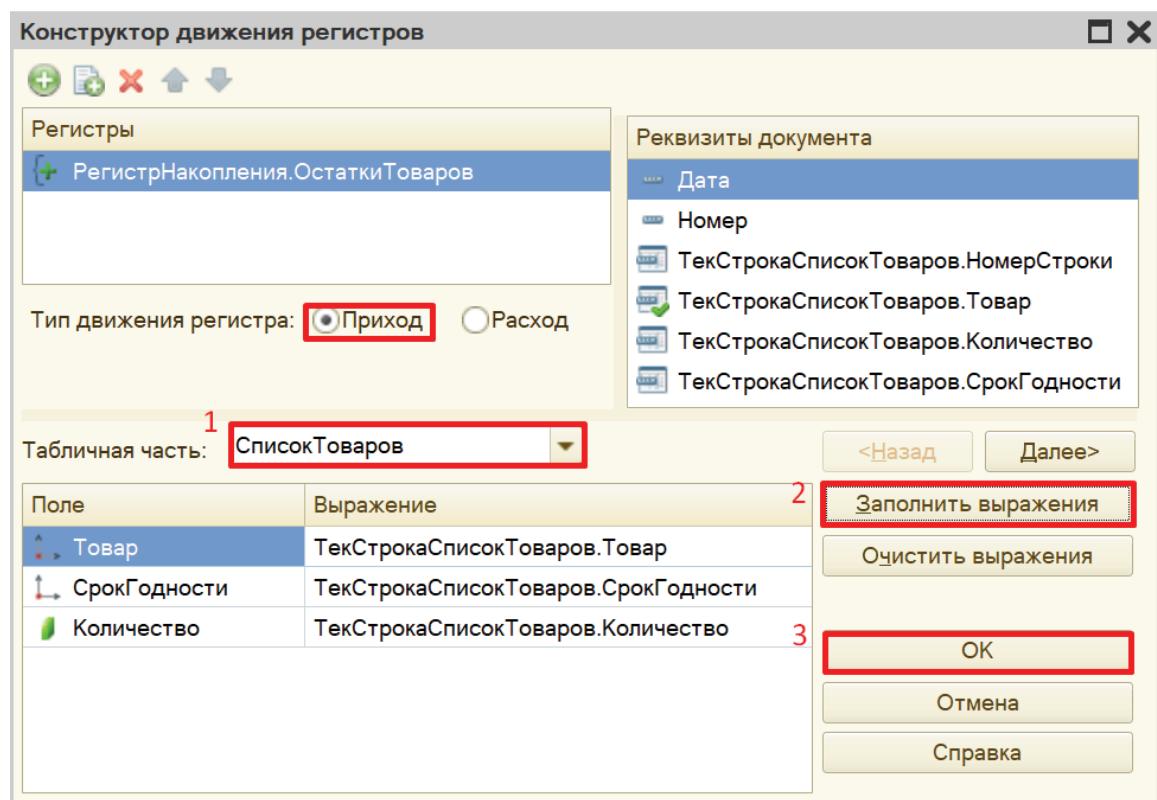
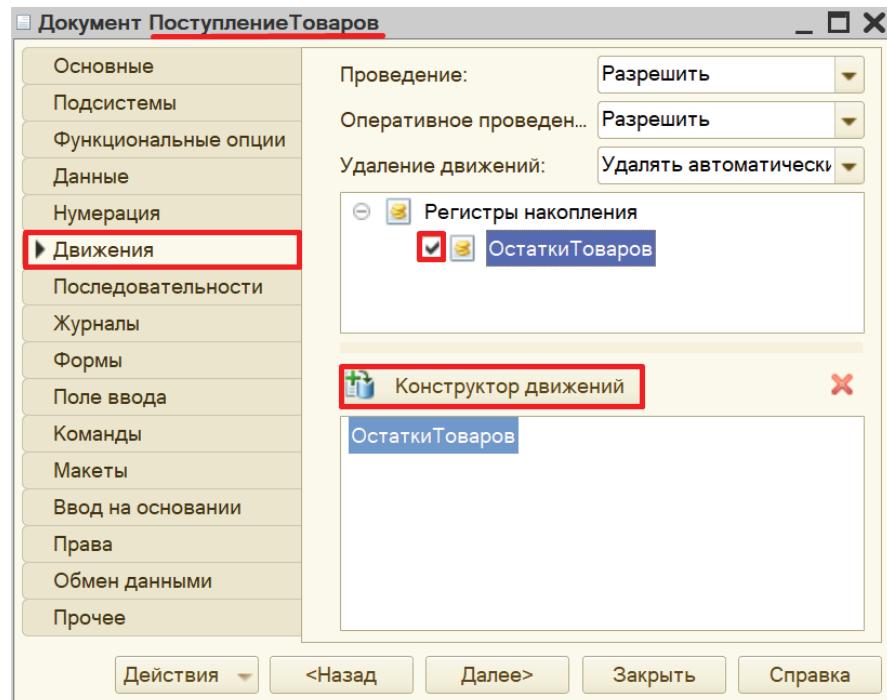
Свойства: Товар	
Измерения	Товар
Ресурсы	
Количество	
Реквизиты	
Тип СправочникСсылка.Товары	

Свойства: СрокГодности	
Измерения	СрокГодности
Ресурсы	
Количество	
Реквизиты	
Тип Дата	

Чтобы *регистр накопления* заработал, нужно сделать следующее:

1. Определить источники данных, которые будут попадать в регистр (определить документы-регистраторы).
2. Описать, каким образом данные из каждого документа-регистратора должны попадать в регистры.

Сейчас в нашей конфигурации всего один документ, он должен передавать значения в *регистр накопления*. Откроем окно редактирования документа «Поступление Товаров» на вкладке «Движения». Выберем регистр, в который будут передаваться данные документа, и воспользуемся *конструктором движений*.



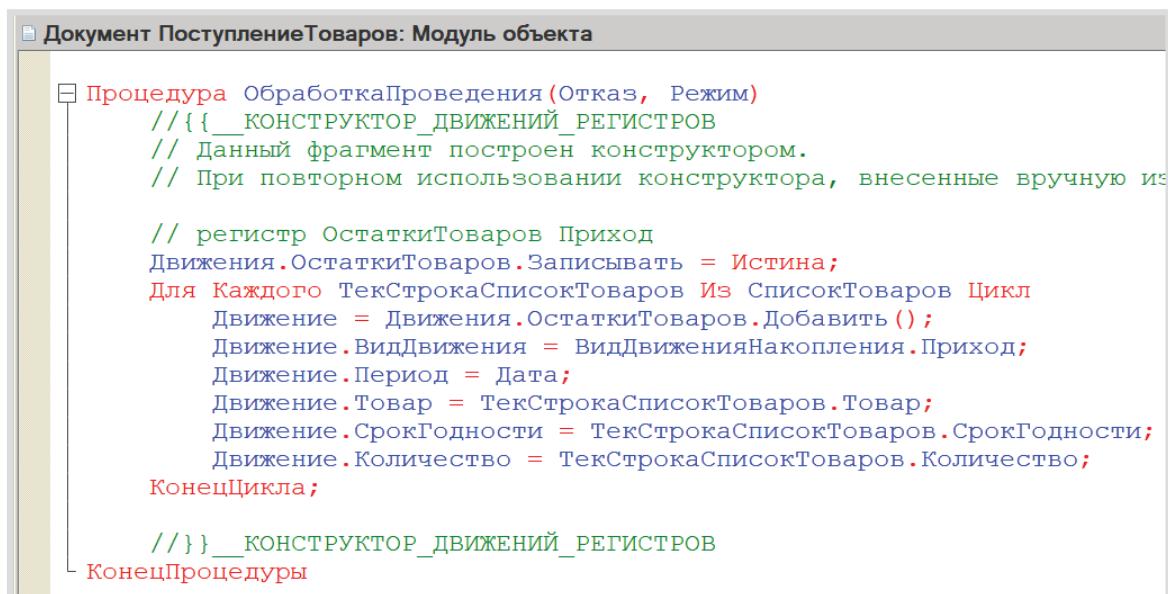
Окно конструктора движений состоит из нескольких областей:

- Левая верхняя область позволяет переключаться между регистрами (один документ может делать движения сразу в несколько разных регистров).
- Правая верхняя область описывает реквизиты документа-регистратора. Чтобы отобразить в данной области реквизиты табличной части нужно выбрать ее в соответствующем поле.
- В нижней части окна описаны реквизиты *регистра накопления*. Нужно заполнить поле «Выражение» реквизитами документа.

Поскольку получение товара должно увеличивать общее количество товаров, то тип движения регистра необходимо выбрать «Приход». Регистр будет обозначаться знаком «+» (плюс).

Если все было сделано правильно, имена и типы реквизитов совпадают, то при нажатии на кнопку «Заполнить выражения» реквизиты регистра заполняются автоматически. Если этого не произошло, то заполните поле «Выражение» вручную, путем выбора соответствующих реквизитов документа.

При нажатии на кнопку «OK» система сформирует программный код, который при успешном проведении документа произведет движения в *регистр накопления*, то есть скопирует данные из документа в *регистр накопления*.



```

Документ ПоступлениеТоваров: Модуль объекта

Процедура ОбработкаПроведения (Отказ, Режим)
//{{ Конструктор_движений_регистров
//  Данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную из

//  регистр ОстаткиТоваров Приход
Движения.ОстаткиТоваров.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваров.Добавить ();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.СрокГодности = ТекСтрокаСписокТоваров.СрокГодности;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;

//}}_ Конструктор_движений_регистров
КонецПроцедуры

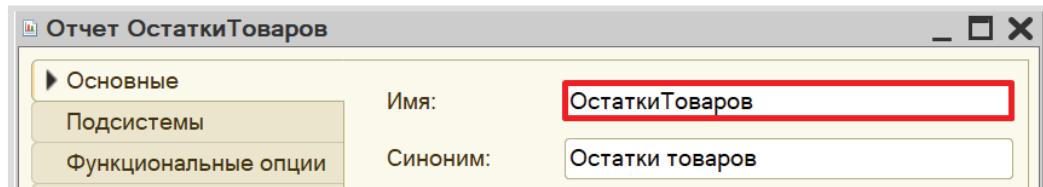
```

По данным, которые формирует *регистр накопления*, можно сформировать отчет.

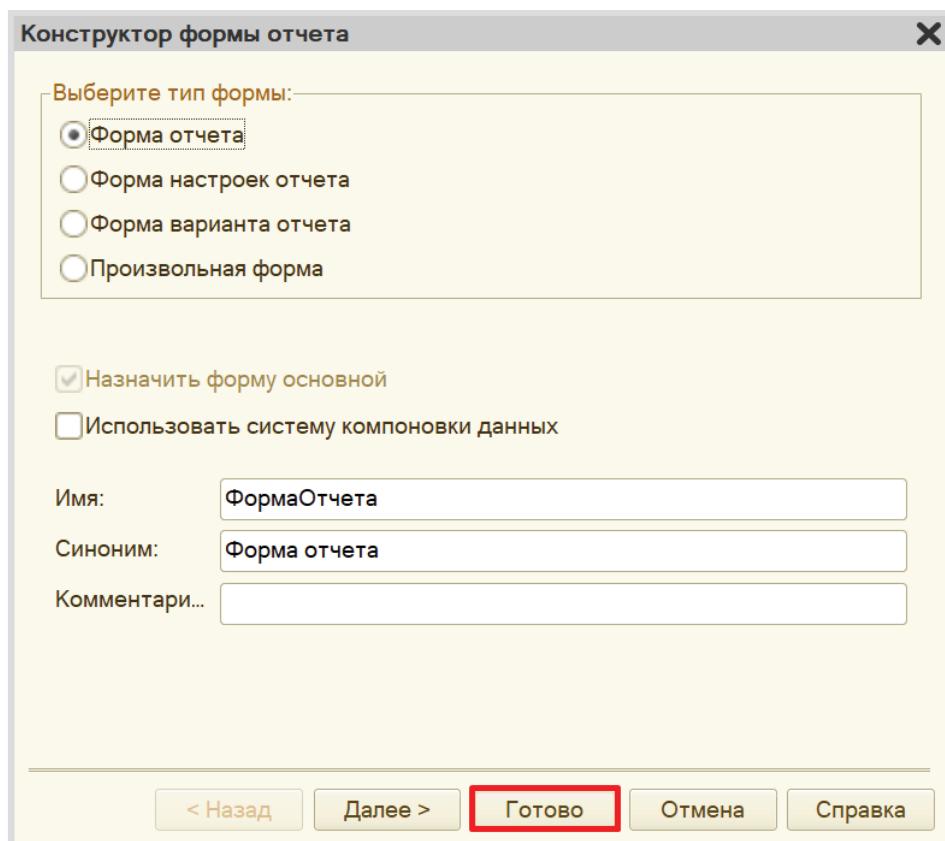
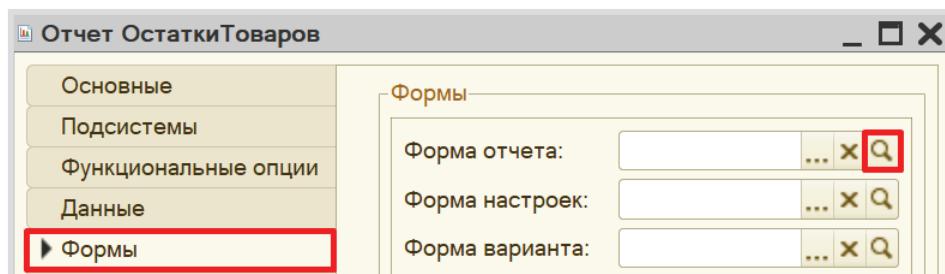
Определение

Отчет – это такой объект конфигурации, который формирует данные в понятном для пользователя виде, например, в виде таблицы или диаграммы (подробнее про отчеты можно прочитать здесь: [https://v8.1c.ru/platforma/отчет/](https://v8.1c.ru/platforma/otchet/)).

Добавим отчет «ОстаткиТоваров».



Создавать отчет будем с помощью формы. Переходим на вкладку «Формы» и создаем новую форму отчета.

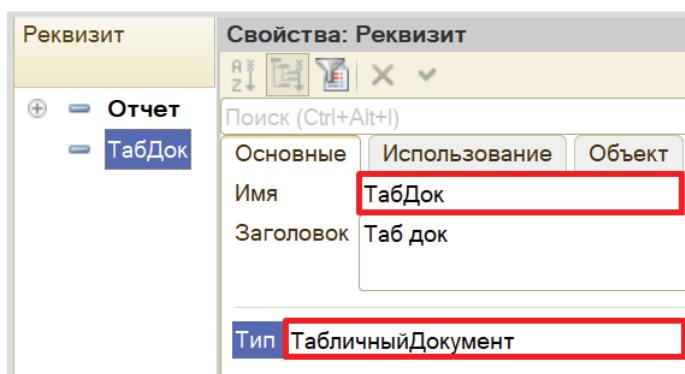
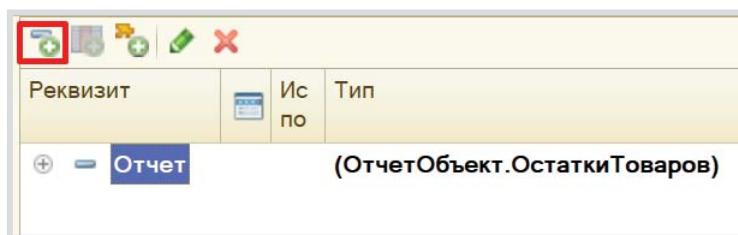


Открывается окно редактора форм.

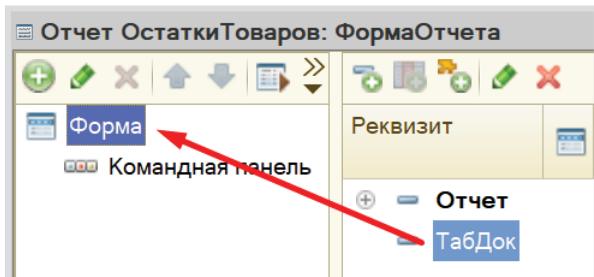
Конструктор формы состоит из трех областей, каждая из которых отвечает за ту или иную функциональность формы:

- Снизу находится область предпросмотра формы. Она позволяет лишь приблизительно понять, как будет отображаться данная форма для пользователя, поскольку она может быть изменена с учетом множества различных факторов. Это своеобразная «иллюзия» того, что увидит пользователь.
- В правой верхней области находятся данные, которые мы вообще можем использовать в каком-либо виде на этой форме. Они разделены по вкладкам «Реквизиты», «Команды» и «Параметры».
- В левой верхней области *конструктора формы* описывается, какие именно данные будут представлены на форме, а также их графический вид. Здесь – две вкладки «Элементы» и «Командный интерфейс». На вкладке «Элементы» настраивается внешний вид и расположение реквизитов на форме. Вкладка «Командный интерфейс» определяет положение команд (кнопок) на форме.

Добавим новый реквизит «ТабДок», тип – «Табличный документ».

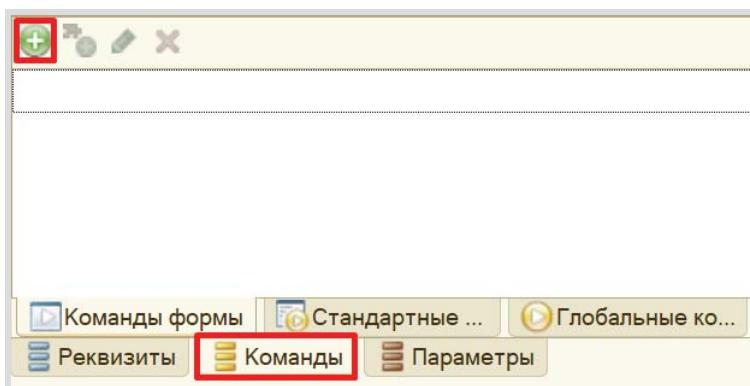


Перенесем созданный реквизит на форму.

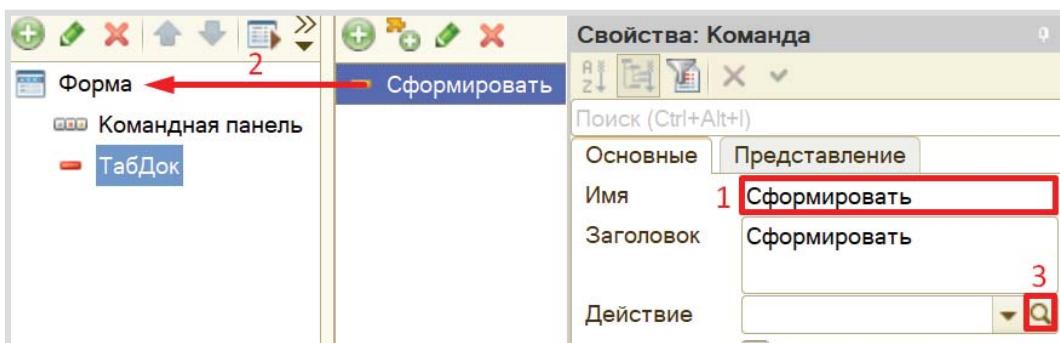


В области предпросмотра должна появиться заготовка табличного документа.

Далее добавим кнопку, при нажатии на которую будет формироваться отчет.

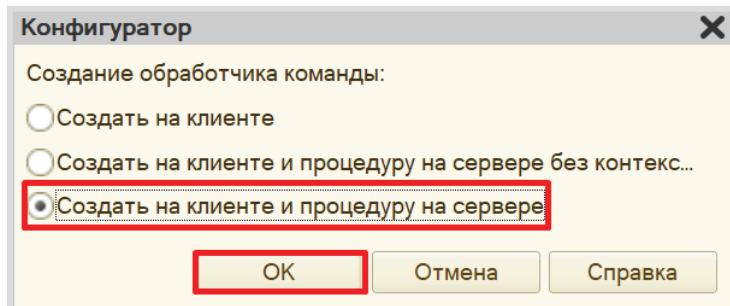


Дадим кнопке имя «Сформировать», перенесем ее на форму, а затем опишем ее действие.



Система спросит, где ей создать обработчик команды: на клиенте или на сервере?

Поскольку для формирования отчета нам понадобятся данные из базы данных, доступ к которым можно получить только со стороны сервера, то выберем вариант «Создать на клиенте и процедуру на сервере».



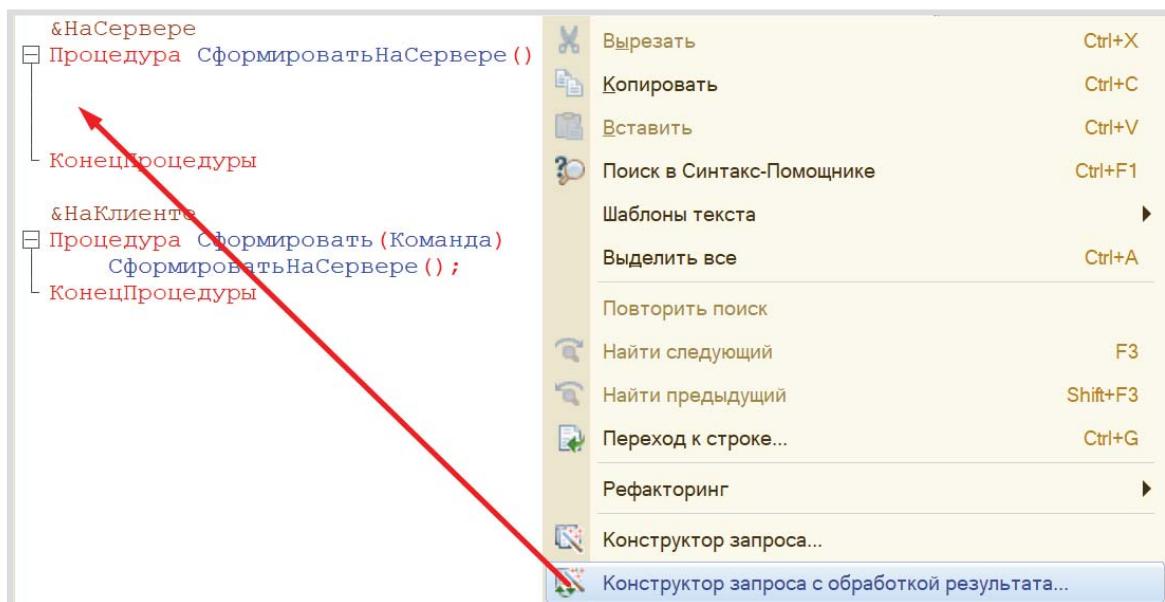
Система сформировала заготовки для двух процедур.

```
Отчет ОстаткиТоваров: ФормаОтчета

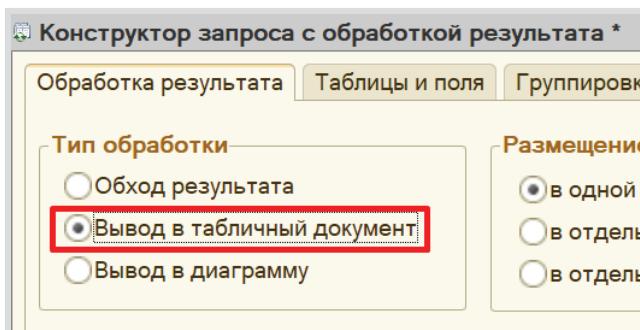
    &НаСервере
    ┌── Процедура СформироватьНаСервере ()
        // Вставить содержимое обработчика.
        КонецПроцедуры

    &НаКлиенте
    ┌── Процедура Сформировать (Команда)
        СформироватьНаСервере () ;
        КонецПроцедуры
```

Внутри процедуры «СформироватьНаСервере» откроем *конструктор запроса с обработкой результата*. Для этого нужно вызвать контекстное меню правой кнопкой мыши, щелкнув по пустой области внутри процедуры. Из контекстного меню выбрать необходимый конструктор. Создадим новый запрос.



Откроется окно *конструктора запроса*. На вкладке «Обработка результата» из предложенных вариантов выбора обработки необходимо выбрать вариант вывода в табличный документ.



Переходим на следующую вкладку «Таблицы и поля».

Открывшееся окно имеет три части:

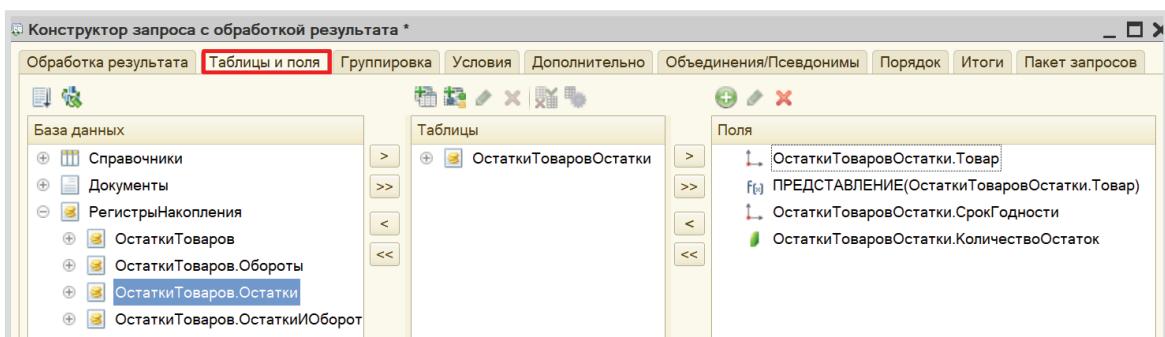
- Часть слева отображает все объекты конфигурации, имеющиеся в нашей базе данных. Нужно выбрать лишь те объекты, из которых мы хотим получать данные.
- Посередине находятся таблицы – это выбранные нами объекты, откуда мы хотим получать данные для конкретного запроса.
- Справа поля – это те значения (поля), которые мы хотим получить.

Данные будем брать не из *регистра накопления* напрямую, а из виртуальной таблицы, которую создает этот регистр автоматически. Данная виртуальная таблица способна обработать основную таблицу и самостоятельно посчитать остатки товаров.

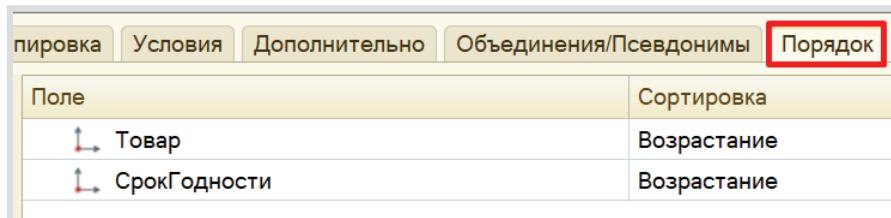
Нам следует вывести в отчет все реквизиты виртуальной таблицы остатков.

Чтобы перенести данные из одного окна в другое, просто перетащите нужные поля с помощью мыши либо воспользуйтесь стрелочками, расположенными между окнами.

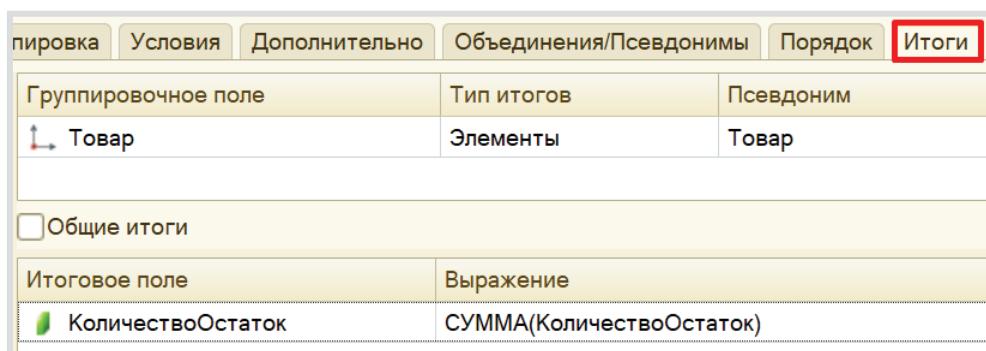
В результате данное окно должно быть заполнено следующим образом:



Далее переключаемся на вкладку «Порядок». Здесь можно упорядочить элементы так, как нам удобно. Упорядочим их сначала по наименованию, а затем – по сроку годности.



Далее переходим на вкладку «Итоги» и просуммируем все товары между собой. Для этого нужно заполнить поля следующим образом:



По завершении работы с *конструктором движений* нажимаем на кнопку «OK».

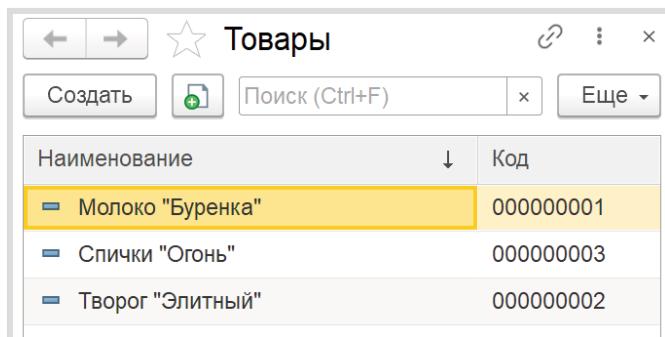
Система на основе введенных ранее настроек сформировала макет, по которому будет строиться отчет.

Отчет ОстаткиТоваров: Макет				
	1	2	3	4
Заголовок	1			
	2			
	3			
ШапкаТабл	4	Товар	СрокГодности	КоличествоОстаток
Товар	5	<ТоварПредставление>		<КоличествоОстаток>
Детали	6		<СрокГодности>	<КоличествоОстаток>
ПодвалТабл	7			
Подвал	8			

Внутри процедуры «СформироватьНаСервере» должен появиться код, который будет заполнять макет данными.

Теперь попробуем запустить систему в режиме «1С:Предприятие» и проверить, корректно ли работают созданные нами объекты.

Начнем с добавления новых товаров.



Теперь добавим несколько документов «Поступление товаров», причем для одних и тех же продуктов укажем разные сроки годности.

Поступление товаров (создание) *

Добавить		Провести и закрыть	Записать	Провести	Еще
Номер:					
Дата:	09.09.2020 0:00:00				
Добавить		Поиск (Ctrl+F)	Еще		
N	Товар	Количество	Срок годности		
1	Молоко "Буренка"	15,000	19.09.2020		
2	Творог "Элитный"	10,000	21.09.2020		

Поступление товаров (создание) *

Добавить		Провести и закрыть	Записать	Провести	Еще
Номер:					
Дата:	09.09.2020 0:00:00				
Добавить		Поиск (Ctrl+F)	Еще		
N	Товар	Количество	Срок годности		
1	Молоко "Буренка"	20,000	17.09.2020		
2	Творог "Элитный"	18,000	19.09.2020		

А теперь попробуем сформировать отчет.

Откроем его и нажмем на кнопку «Сформировать», которая находится под таблицей. Получаем отчет:

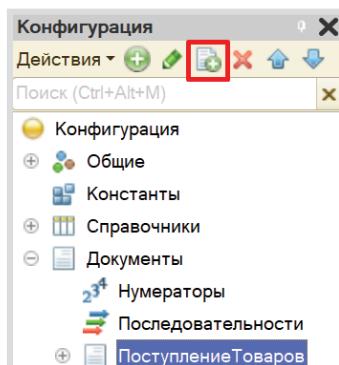
Товар	СрокГодности	Количество	Остаток
Молоко "Буренка"	17.09.2020 0:00:00	20,000	35,000
	19.09.2020 0:00:00	15,000	
Творог "Элитный"	19.09.2020 0:00:00	18,000	28,000
	21.09.2020 0:00:00	10,000	

В отчете мы можем увидеть общее количество тех или иных товаров с различными сроками годности (обозначено полужирным). Произведено упорядочивание товаров, в первую очередь, по алфавиту, а затем уже – по сроку годности.

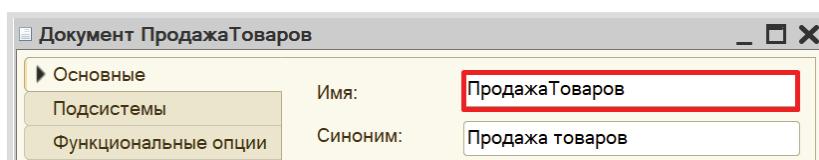
Данный отчет понадобится нам для реализации продажи товаров. Сначала мы можем посмотреть, сколько всего единиц того или иного товара у нас имеется, и только затем уже списывать товары из различных партий (с различным сроком годности).

«В системе нужно регистрировать продажу товара».

Добавим новый документ «ПродажаТоваров» путем копирования документа «ПоступлениеТоваров».

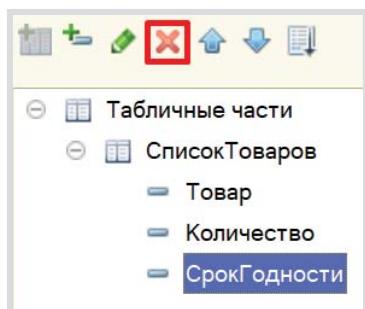


Должен появиться документ «ПоступлениеТоваров1». Переименуем его в «ПродажаТоваров» и перейдем на вкладку «Данные».

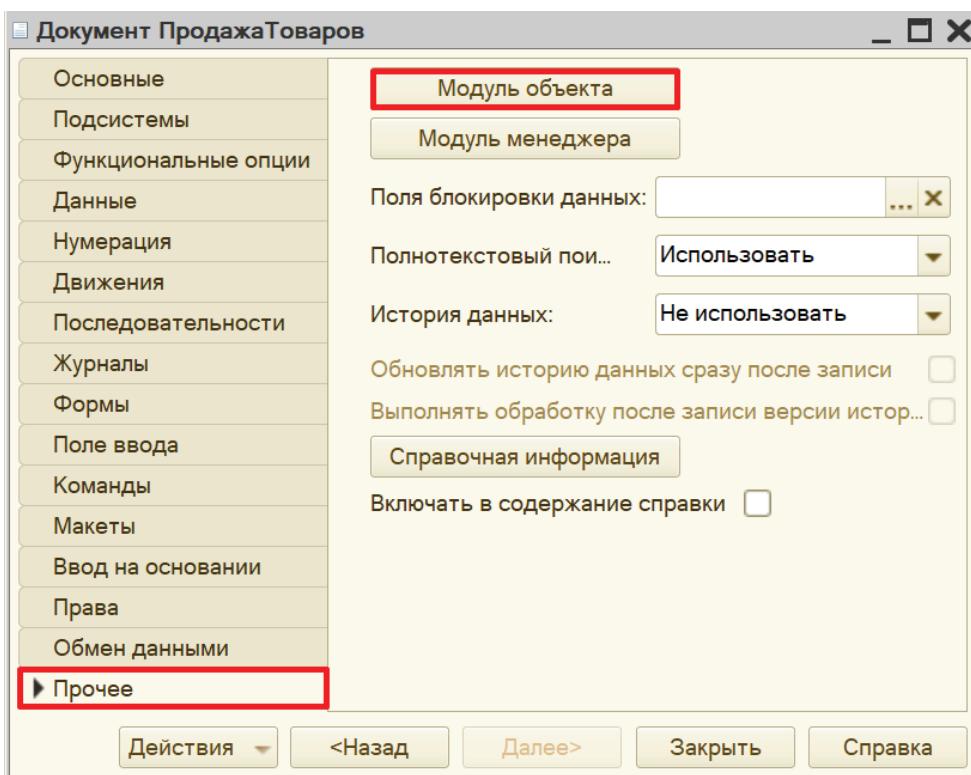


«При продаже товаров указывается, какие товары были проданы и в каком количестве».

Из данного условия делаем вывод, что в табличной части документа должно быть только два реквизита: «Товар» и «Количество». Значит, нужно удалить реквизит «СрокГодности» из табличной части.



Теперь следует описать логику проведения данного документа. Для этого нужно открыть модуль объекта, который находится на вкладке «Прочее».



Документ был скопирован полностью, в том числе была скопирована обработка проведения. Закомментируем код процедуры. Для этого нужно выделить нужные строки и нажать на кнопку с изображением двух обратных косых черт.

```

Документ ПродажаТоваров: Модуль объекта

Процедура ОбработкаПроведения (Отказ, Режим)
//{{ __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
//  Данный фрагмент построен конструктором.
//  При повторном использовании конструктора, внесенные вручную измене
//  регистр ОстаткиТоваров Приход
Движения.ОстаткиТоваров.Записывать = Истина;
Для Каждого ТекСтрокаСписокТоваров Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваров.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Товар = ТекСтрокаСписокТоваров.Товар;
    Движение.СрокГодности = ТекСтрокаСписокТоваров.СрокГодности;
    Движение.Количество = ТекСтрокаСписокТоваров.Количество;
КонецЦикла;

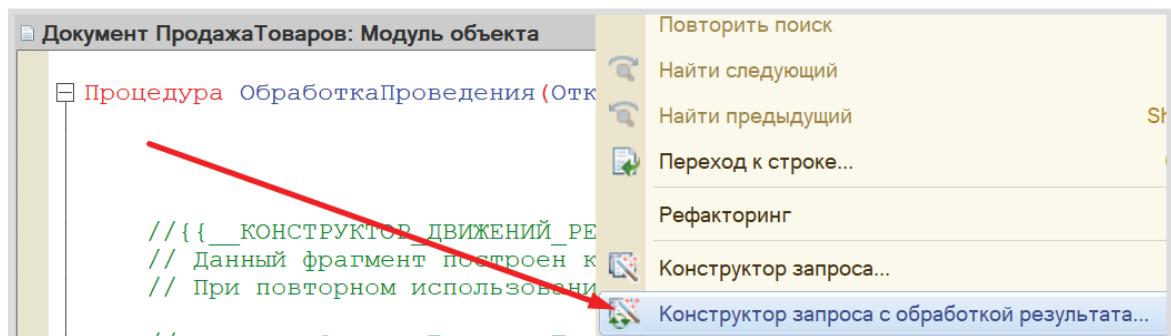
//}} __КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры

```

В документ «ПродажаТоваров» пользователь должен добавлять информацию о товаре и количестве, которое он хочет продать. В первую очередь, система должна проанализировать, достаточно ли каждого вида товаров для продажи. Если достаточно, тогда нужно списывать товар партиями по возрастанию срока годности.

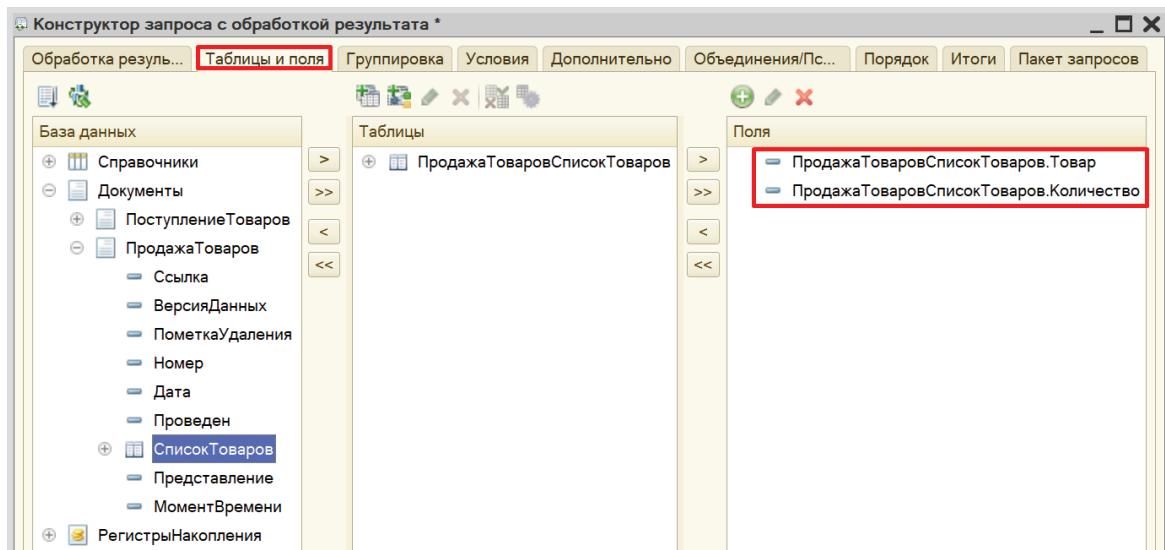
Для проведения данного анализа необходимо получить данные как из документа, так и из *регистра накопления*. Получим эти данные с помощью *конструктора запросов*.

Этот конструктор можно открыть из контекстного меню щелчком правой кнопки мыши по области модуля. Данный конструктор обязательно должен быть вызван внутри процедуры «ОбработкаПроведения».

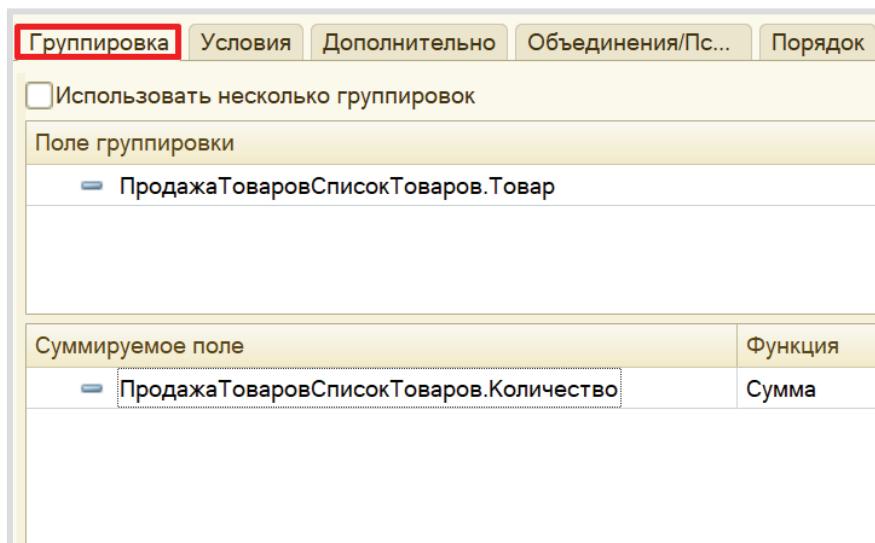


Соглашаемся с предложением системы создать новый запрос.

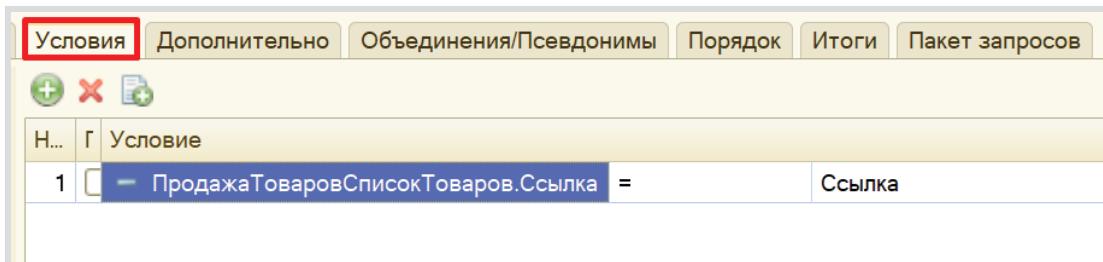
Открывается окно *конструктора запроса с обработкой результата*. Переходим на вкладку «Таблицы и поля». Здесь нам нужно выбрать поля табличной части документа.



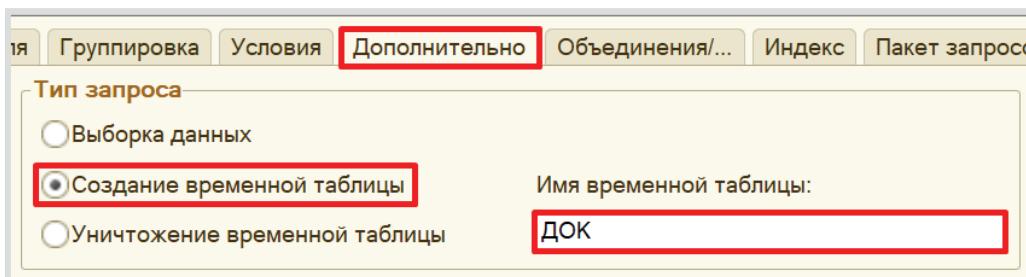
Переходим на вкладку «Группировка» и сгруппируем товары между собой: вдруг пользователь ввел несколько отдельных строк с одинаковым товаром?



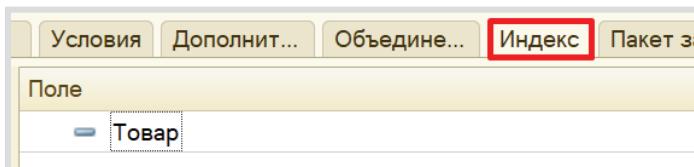
Далее переходим на вкладку «Условие» и поставим условие на ссылку. Таким образом, запрос будет обрабатывать табличную часть только того одного документа, чью ссылку мы передадим в запрос.



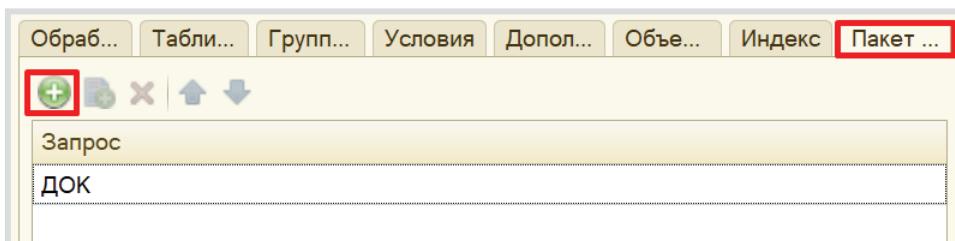
Теперь все полученные данные из документа «ПродажаТоваров» следует поместить во времененную таблицу. Для этого переходим на вкладку «Дополнительно», выбираем пункт «Создание временной таблицы» и зададим ей имя «ДОК».



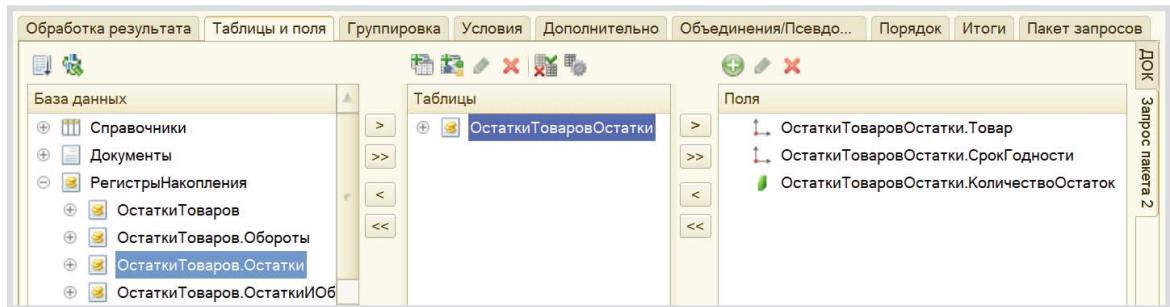
Переходим на вкладку «Индекс» и проиндексируем таблицу по полю «Товар», поскольку далее нужно будет соединить несколько таблиц.



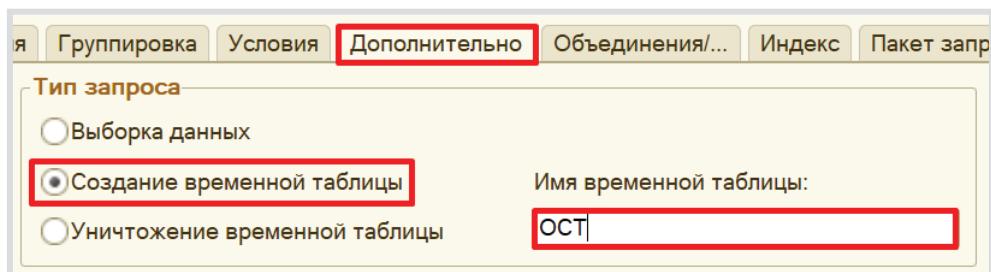
Переходим на вкладку «Пакет запросов» и добавляем новый запрос.



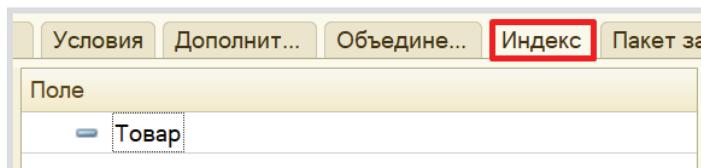
Снова открывается вкладка «Таблицы поля». Здесь нужно выбрать уже остатки из *регистра накопления*.



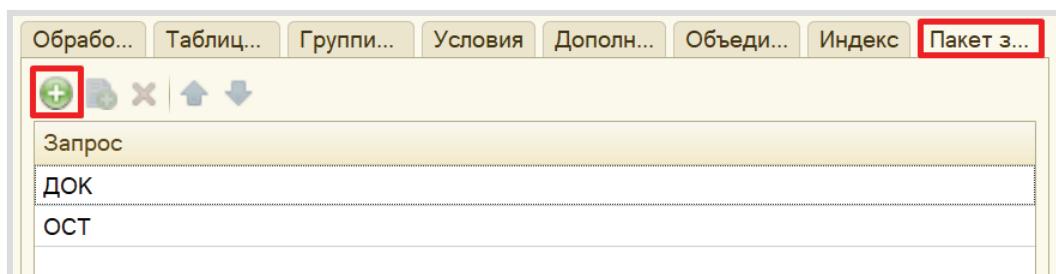
Полученные данные тоже положим во временную таблицу. Переходим на вкладку «Дополнительно» и создаем временную таблицу «ОСТ».



Переходим на вкладку «Индекс» и снова индексируем по товару.



Осталось лишь добавить последний запрос, который сможет объединить в себе данные из двух получившихся временных таблиц. Вкладка «Пакет запросов», кнопка «Добавить».



Необходимо выбрать следующие поля из двух таблиц:

Теперь нужно перейти на вкладку «Связь» и связать между собой две временные таблицы по полю «Товар».

Далее переходим на вкладку «Порядок» и упорядочиваем элементы в запросе по товару и сроку годности.

Далее переходим на вкладку «Итоги» и снова просуммируем все товары.

По окончании работы над итогами нажимаем на кнопку «OK».

В процедуре «ОбработкаПроведения» появился код, который был сформирован конструктором запроса.

Под текстом запроса можно увидеть цикл в цикле: внешний цикл перебирает товары, а внутренний – сроки годности этих товаров.

```

Пока ВыборкаТовар.Следующий() Цикл
    Если ВыборкаТовар.Количество > ВыборкаТовар.КоличествоОстаток Тогда
        Сообщить ("Не хватает товаров " + ВыборкаТовар.Товар + " " +
                    (ВыборкаТовар.Количество - ВыборкаТовар.КоличествоОстаток) + " шт.");
        Отказ = Истина;
    КонецЕсли;

    ВыборкаДетальныеЗаписи = ВыборкаТовар.Выбрать();

    Пока ВыборкаДетальныеЗаписи.Следующий() И ОсталосьСписать > 0 Цикл
        // Вставить обработку выборки ВыборкаДетальныеЗаписи
    КонецЦикла;
КонецЦикла;

```

Пример

Предположим, что некий клиент решил купить 50 пакетов молока. В первую очередь, нужно проверить, а есть ли у нас 50 пакетов на складе?

Именно такую проверку и нужно добавить. В случае нехватки товаров нужно вывести пользователю сообщение и отметить проведение.

```

Пока ВыборкаТовар.Следующий() Цикл
    // сколько хотят купить      //сколько осталось на складе
    Если ВыборкаТовар.Количество > ВыборкаТовар.КоличествоОстаток Тогда
        Сообщить ("Не хватает товаров " + ВыборкаТовар.Товар + " " +
                    (ВыборкаТовар.Количество - ВыборкаТовар.КоличествоОстаток) + " шт.");
        Отказ = Истина;
    КонецЕсли;
    ВыборкаДетальныеЗаписи = ВыборкаТовар.Выбрать();

    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        // Вставить обработку выборки ВыборкаДетальныеЗаписи
    КонецЦикла;
КонецЦикла;

```

Теперь будем перебирать партии.

Пример

Предположим, что в документе указано 50 пакетов молока. На складе в одной партии – 20 пакетов, а в другой – 40. Необходимо сначала списать 20 из первой партии, а затем – 30 из второй.

Добавим переменную «ОсталосьСписать» – она будет своеобразным счетчиком, который мы будем вести до нуля.

```

Пока ВыборкаТовар.Следующий() Цикл
    // сколько хотят купить //сколько осталось на складе
    Если ВыборкаТовар.Количество > ВыборкаТовар.КоличествоОстаток Тогда
        Сообщить ("Не хватает товаров " + ВыборкаТовар.Товар + " " +
                    (ВыборкаТовар.Количество - ВыборкаТовар.КоличествоОстаток) + " шт.");
        Отказ = Истина;
    КонецЕсли;
    ОсталосьСписать = ВыборкаТовар.Количество;
    ВыборкаДетальныеЗаписи = ВыборкаТовар.Выбрать();
    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        // вставить обработку выборки ВыборкаДетальныеЗаписи
    КонецЦикла;
КонецЦикла;

```

Теперь проверяем, сколько осталось товаров в конкретной партии и списывать это количество из переменной «ОсталосьСписать», после чего будем переходить к следующей партии.

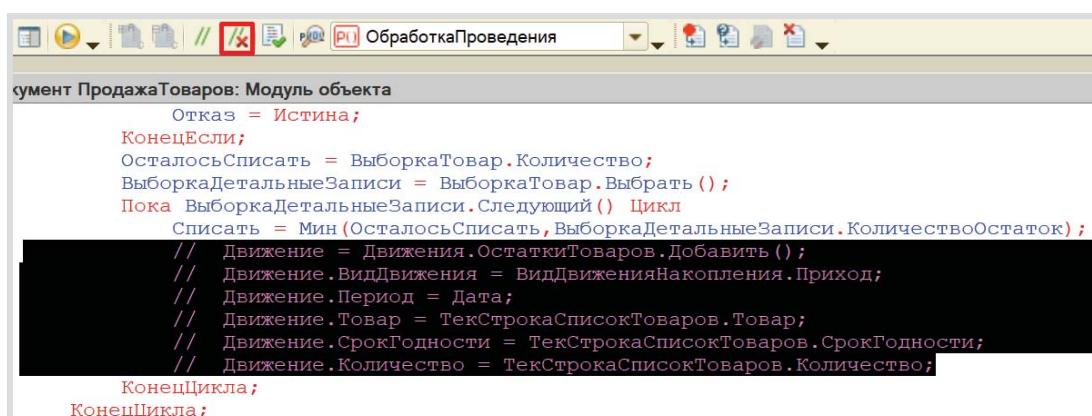
Списать из конкретной партии можно не больше, чем в ней есть, или меньше. Воспользуемся функцией «Мин», которая выбирает наименьшее из заданных в нее чисел.

```

Пока ВыборкаТовар.Следующий() Цикл
    // сколько хотят купить //сколько осталось на складе
    Если ВыборкаТовар.Количество > ВыборкаТовар.КоличествоОстаток Тогда
        Сообщить ("Не хватает товаров " + ВыборкаТовар.Товар + " " +
                    (ВыборкаТовар.Количество - ВыборкаТовар.КоличествоОстаток) + " шт.");
        Отказ = Истина;
    КонецЕсли;
    ОсталосьСписать = ВыборкаТовар.Количество;
    ВыборкаДетальныеЗаписи = ВыборкаТовар.Выбрать();
    Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        Списать = Мин(ОсталосьСписать, ВыборкаДетальныеЗаписи.КоличествоОстаток);
    КонецЦикла;
КонецЦикла;

```

Теперь в переменной «Списать» хранится то значение, которое мы можем легко списать и при этом не получить отрицательных остатков. Можно делать движение в *регистр накопления*. Скопируем ранее закомментированный текст, добавим в цикл и раскомментируем.



Изменим вид движения на «Расход», поскольку данный документ должен делать движения со знаком «-» (минус), то есть уменьшать значение в регистре. Также изменим «Товар», «Срок годности» и «Количество».

```

Пока ВыборкаДетальныеЗаписи.Следующий () Цикл
    Списать = Мин (ОсталосьСписать, ВыборкаДетальныеЗаписи.КоличествоОстаток) ;
    Движение = Движения.ОстаткиТоваров.Добавить () ;
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ВыборкаДетальныеЗаписи.Товар;
    Движение.СрокГодности = ВыборкаДетальныеЗаписи.СрокГодности;
    Движение.Количество = Списать;
КонецЦикла;

```

Далее нужно уменьшить количество товаров, которые еще нужно списать, и добавить дополнительное условие на переменную «ОсталосьСписать».

```

Пока ВыборкаДетальныеЗаписи.Следующий () И ОсталосьСписать > 0 Цикл
    Списать = Мин (ОсталосьСписать, ВыборкаДетальныеЗаписи.КоличествоОстаток) ;
    Движение = Движения.ОстаткиТоваров.Добавить () ;
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ВыборкаДетальныеЗаписи.Товар;
    Движение.СрокГодности = ВыборкаДетальныеЗаписи.СрокГодности;
    Движение.Количество = Списать;
    ОсталосьСписать = ОсталосьСписать - Списать;
КонецЦикла;

```

Установим маркер формирования движений в значение «Истина».

```

Пока ВыборкаДетальныеЗаписи.Следующий () И ОсталосьСписать > 0 Цикл
    Списать = Мин (ОсталосьСписать, ВыборкаДетальныеЗаписи.КоличествоОстаток) ;
    Движение = Движения.ОстаткиТоваров.Добавить () ;
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Товар = ВыборкаДетальныеЗаписи.Товар;
    Движение.СрокГодности = ВыборкаДетальныеЗаписи.СрокГодности;
    Движение.Количество = Списать;
    ОсталосьСписать = ОсталосьСписать - Списать;
КонецЦикла;
Движение.ОстаткиТоваров.Записывать = Истина;

```

Целиком получившаяся процедура должна выглядеть следующим образом:

```

Процедура ОбработкаПроведения(Отказ, Режим)
    Запрос = Новый Запрос;
    Запрос.Текст =
        "ВЫБРАТЬ
        | ПродажаТоваровСписокТоваров.Товар КАК Товар,
        | СУММА(ПродажаТоваровСписокТоваров.Количество) КАК Количество
        | ПОМЕСТИТЬ ДОК
        | ИЗ
        |     Документ.ПродажаТоваров.СписокТоваров КАК ПродажаТоваровСписокТоваров
        | ГДЕ
        |     ПродажаТоваровСписокТоваров.Ссылка = &Ссылка
        |
        | СГРУППИРОВАТЬ ПО
        |     ПродажаТоваровСписокТоваров.Товар
        |
        | ИНДЕКСИРОВАТЬ ПО
        |     Товар
        | ;
        |
        ///////////////////////////////////////////////////////////////////
        | ВЫБРАТЬ
        |     ОстаткиТоваровОстатки.Товар КАК Товар,
        |     ОстаткиТоваровОстатки.СрокГодности КАК СрокГодности,
        |     ОстаткиТоваровОстатки.КоличествоОстаток КАК КоличествоОстаток
        | ПОМЕСТИТЬ ОСТ
        | ИЗ
        |     РегистрНакопления.ОстаткиТоваров.Остатки КАК ОстаткиТоваровОстатки
        |
        | ИНДЕКСИРОВАТЬ ПО
        |     Товар
        | ;
        |
        ///////////////////////////////////////////////////////////////////
        | ВЫБРАТЬ
        |     ДОК.Товар КАК Товар,
        |     ДОК.Количество КАК Количество,
        |     ОСТ.СрокГодности КАК СрокГодности,
        |     ОСТ.КоличествоОстаток КАК КоличествоОстаток
        |
        | ИЗ
        |     ДОК КАК ДОК
        |         ЛЕВОЕ СОЕДИНЕНИЕ ОСТ КАК ОСТ
        |             ПО ДОК.Товар = ОСТ.Товар
        |
        | УПОРЯДОЧИТЬ ПО
        |     Товар,
        |     СрокГодности
        | ИТОГИ
        |     МИНИМУМ(Количество),
        |     СУММА(КоличествоОстаток)
        | ПО
        |     Товар";
    Запрос.УстановитьПараметр("Ссылка", Ссылка);
    РезультатЗапроса = Запрос.Выполнить();

```

```

ВыборкаТовар = РезультатЗапроса.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

Пока ВыборкаТовар.Следующий() Цикл
    // сколько хотят купить // сколько осталось на складе
    Если ВыборкаТовар.Количество > ВыборкаТовар.КоличествоОстаток Тогда
        Сообщить ("Не хватает товаров " + ВыборкаТовар.Товар + " " +
                    (ВыборкаТовар.Количество - ВыборкаТовар.КоличествоОстаток) + " шт.");
        Отказ = Истина;
    КонецЕсли;
    ОсталосьСписать = ВыборкаТовар.Количество;
    ВыборкаДетальныеЗаписи = ВыборкаТовар.Выбрать();
    Пока ВыборкаДетальныеЗаписи.Следующий() И ОсталосьСписать > 0 Цикл
        Списать = Мин(ОсталосьСписать, ВыборкаДетальныеЗаписи.КоличествоОстаток);
        Движение = Движения.ОстаткиТоваров.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Товар = ВыборкаДетальныеЗаписи.Товар;
        Движение.СрокГодности = ВыборкаДетальныеЗаписи.СрокГодности;
        Движение.Количество = Списать;
        ОсталосьСписать = ОсталосьСписать - Списать;
    КонецЦикла;

КонецЦикла;
Движение.ОстаткиТоваров.Записывать = Истина;

КонецПроцедуры

```

Проверим работоспособность разработанной информационной системы.

Откроем отчет по остаткам.

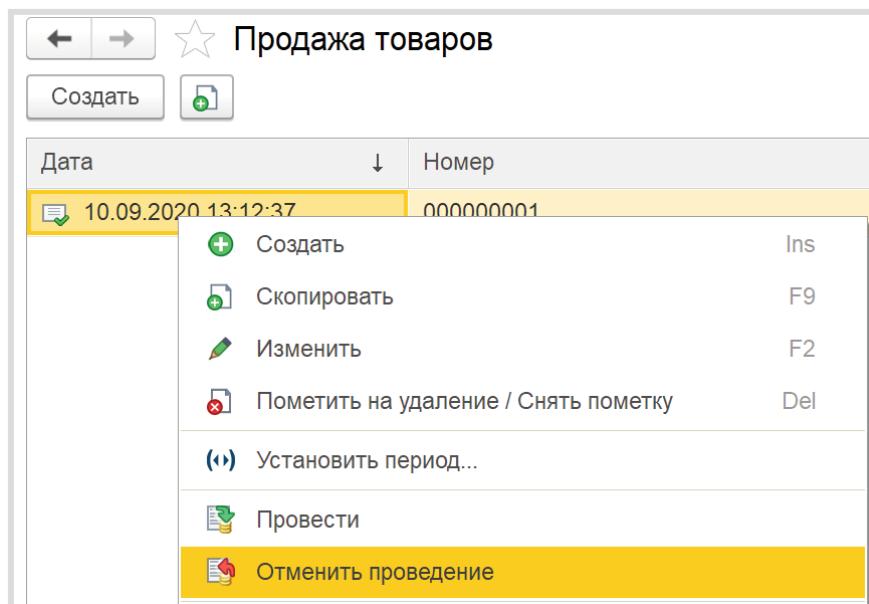
Товар	СрокГодности	КоличествоОстаток
Молоко "Буренка"		35,000
	17.09.2020	
	0:00:00	20,000
	19.09.2020	
	0:00:00	15,000
Творог "Элитный"		28,000
	19.09.2020	
	0:00:00	18,000
	21.09.2020	
	0:00:00	10,000

Всего имеется 35 единиц молока «Буренка». Попробуем продать 25 единиц молока.

Для этого добавим документ «Продажа товаров». Полученный документ необходимо провести.

N	Товар	Количество
1	Молоко "Буренка"	25,000

Если появились проблемы с проведением документа «Продажа товаров», тогда вернитесь к списку всех документов и отмените их проведение. Для этого выделите документ в списке, нажмите на него правой кнопкой мыши и выберите пункт меню «Отменить проведение». Сделайте это для всех документов. Затем проведите документы заново.



Вернемся к отчету и сформируем его.

Товар	Срок годности	Количество	Остаток
Молоко "Буренка"	19.09.2020 0:00:00	10,000	
Творог "Элитный"	19.09.2020 0:00:00	10,000	28,000
	21.09.2020 0:00:00	18,000	
		10,000	

Система списала 20 единиц молока из партии со сроком годности до 17 сентября, и еще 5 – из партии со сроком годности до 19 сентября.

Поставленная задача решена.

© ООО «1С-Паблишинг», 2021

© Оформление. ООО «1С-Паблишинг», 2021

Все права защищены.

Материалы предназначены для личного индивидуального использования приобретателем.

Запрещено тиражирование, распространение материалов, предоставление доступа по сети к материалам без письменного разрешения правообладателей.

Разрешено копирование фрагментов программного кода для использования в разрабатываемых прикладных решениях.

Фирма «1С»

123056, Москва, а/я 64, Селезневская ул., 21.

Тел.: (495) 737-92-57, факс: (495) 681-44-07.

1c@1c.ru, <http://www.1c.ru/>

Издательство ООО «1С-Паблишинг»

127434, Москва, Дмитровское ш., д. 9.

Тел.: (495) 681-02-21, факс: (495) 681-44-07.

publishing@1c.ru, <http://books.1c.ru>

Об опечатках просьба сообщать по адресу publishing@1c.ru.