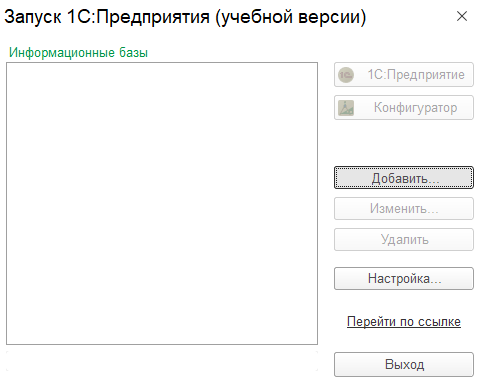
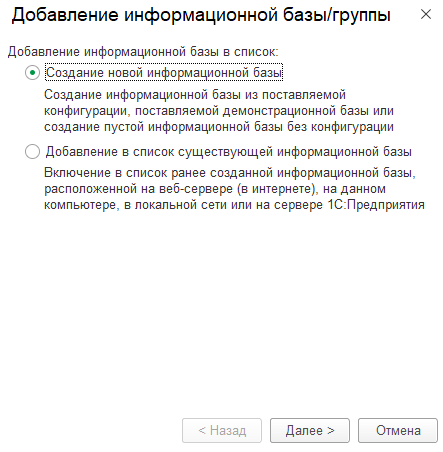
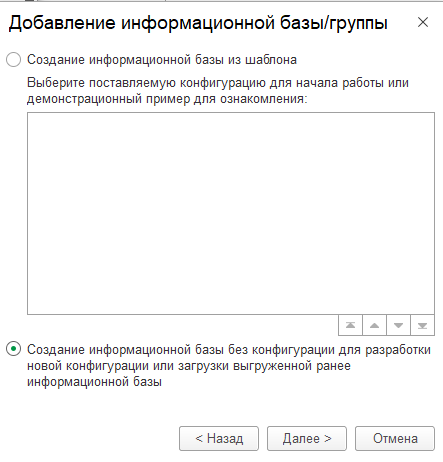
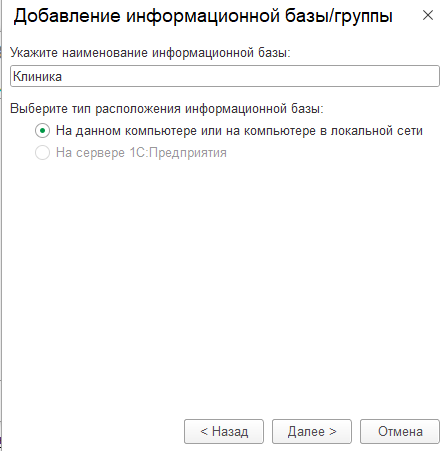
**Создание новой информационной базы**

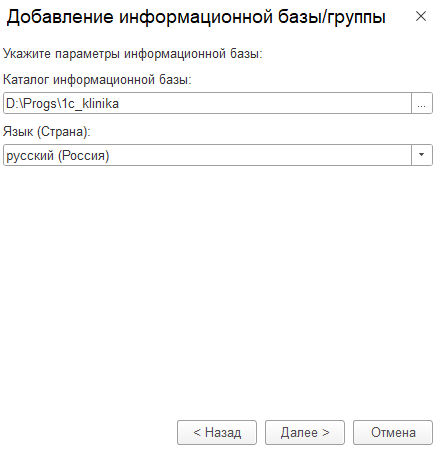
Выбирать пункты согласно скриншотам



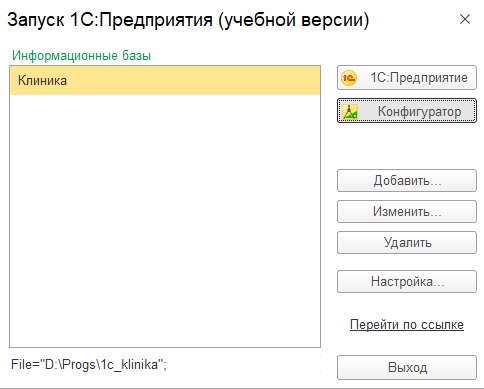




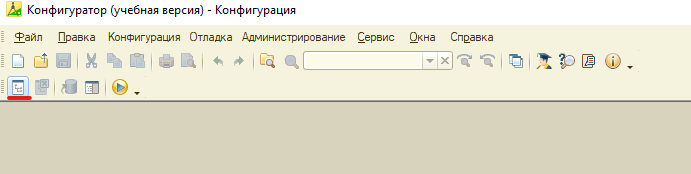






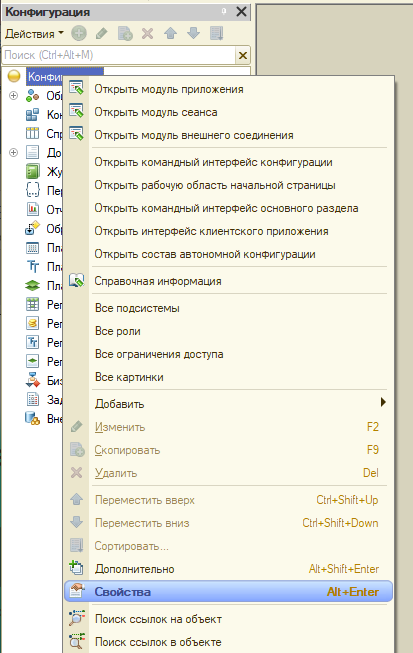


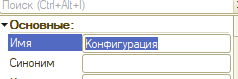
**Настройка свойств конфигурации**

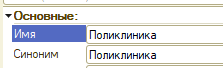




ПКМ по «Конфигурация» -> «Свойства»

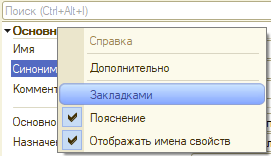


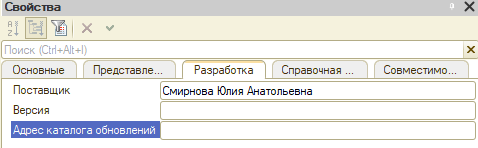
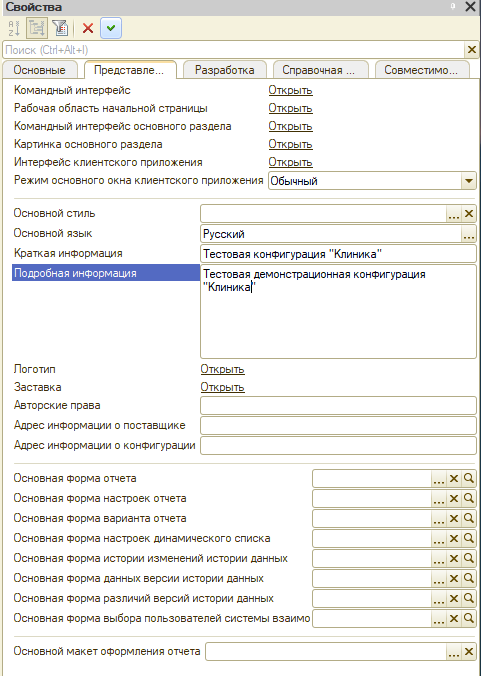




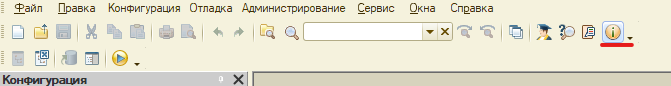
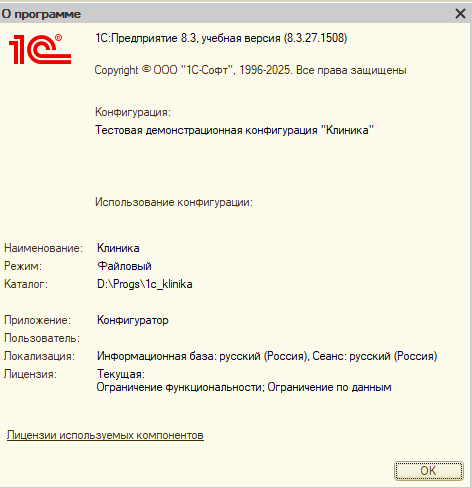
Имя в 1С – Наименование объекта используемое в коде

Синоним в 1С – Отображаемое наименование объекта

ПКМ на «Основные» -> «Закладками»  
  
Более удобный вариант отображения свойств

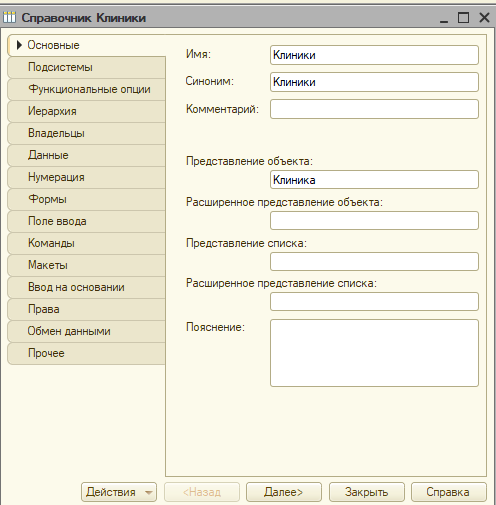
Указать разработчика  
  
  


Сохранить изменения, или сочетание клавиш ctrl+s  


После первого запуска информация о конфигурации изменится  
  
  
  


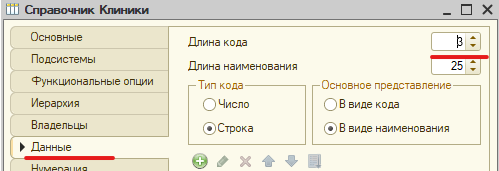
**Создание справочников**

Справочники — это прикладные объекты конфигурации. Они позволяют хранить в информационной базе данные, имеющие одинаковую структуру и списочный характер.



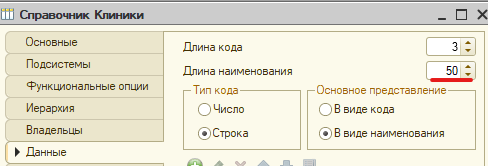
Представление объекта – объект в единичной форме свойства которого описываются в справочнике

Перейти на вкладку данные, уменьшить длину кода до 3



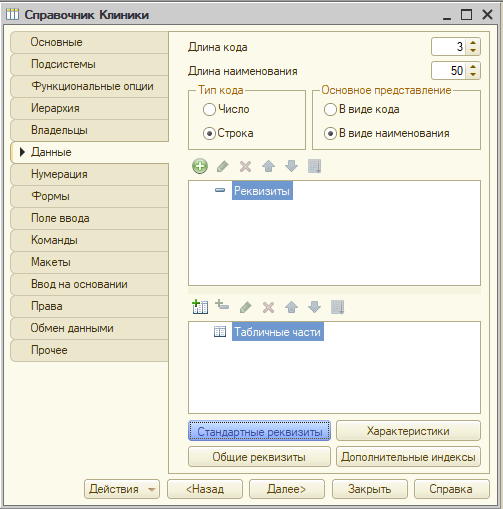
Длина кода – максимальная длина идентификаторов объектов справочников.  
В данном случае у нас может быть 999 клиник, т.е. длина кода 3 говорит о том, что нашим идентификатором объекта может быть максимум трехзначное число.

Используемая длина кода должна быть избыточной, т.к. слишком большая длина приводит к использованию лишних ресурсов, т.к. число 000000001 весит на порядок больше чем 001, в случае использования большого количества идентификаторов для большого количества справочников, эффект становится ощутимым



Длина наименования – Максимальная длина наименования объекта описываемого справочником

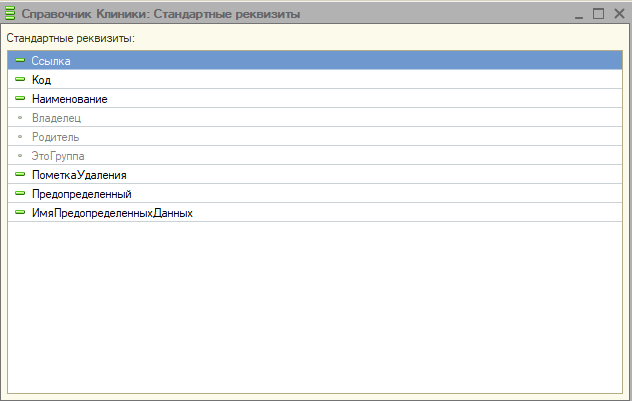
Рассмотрим свойства указанные для каждого объекта по умолчанию



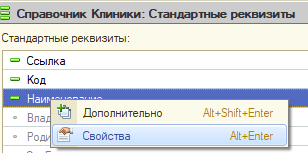
Код – идентификатор

Наименование – название

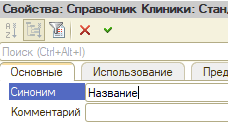
Остальные свойства будут рассмотрены позже

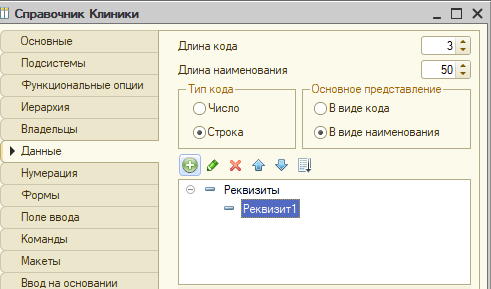


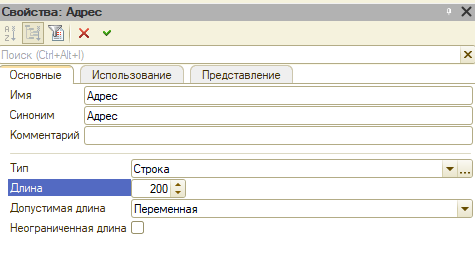
ПКМ «Наименование» -> «Свойства»

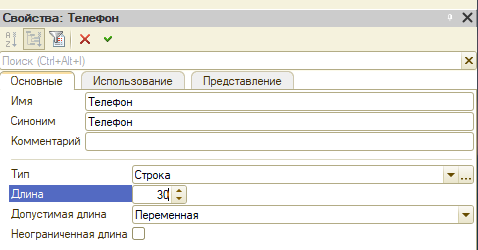


Укажем более приятное отображение свойства наименования объекта в системе

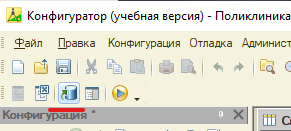
  
P.s. Теперь при запуске конфигурации и взаимодействии с справочником «Клиники» в системе мы будем видеть «Название», а не «Наименование»

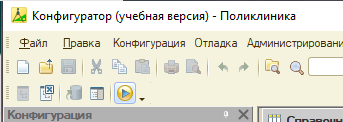
Добавим реквизит объекта  
  
Реквизит – свойство объекта

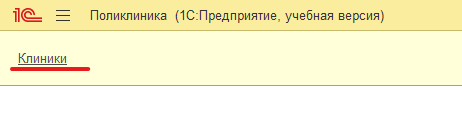


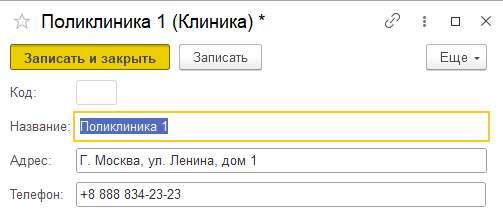


Обновить конфигурацию БД, или F7 для сохранения изменений



Начать отладку, или F5  






Данные успешно записались



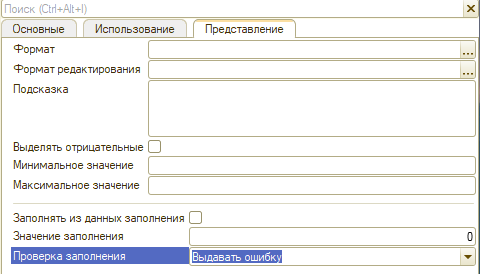
Выйти из режима отладки

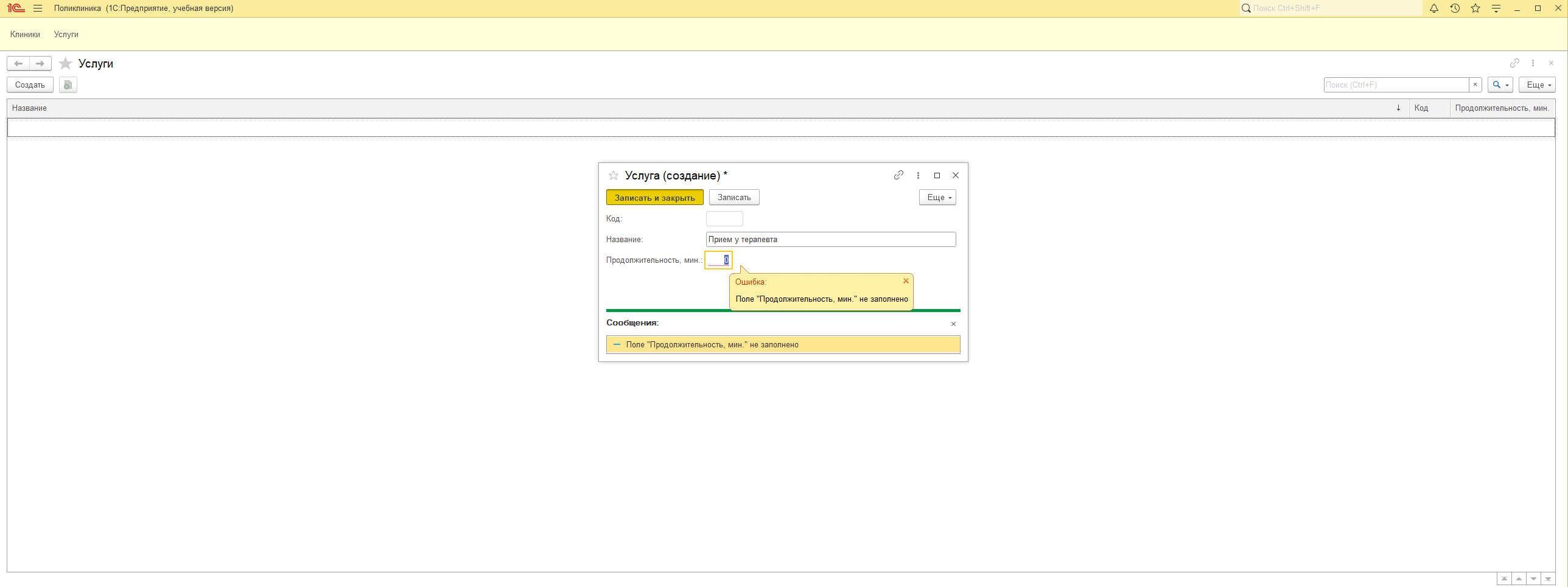
Создадим справочник для услуг

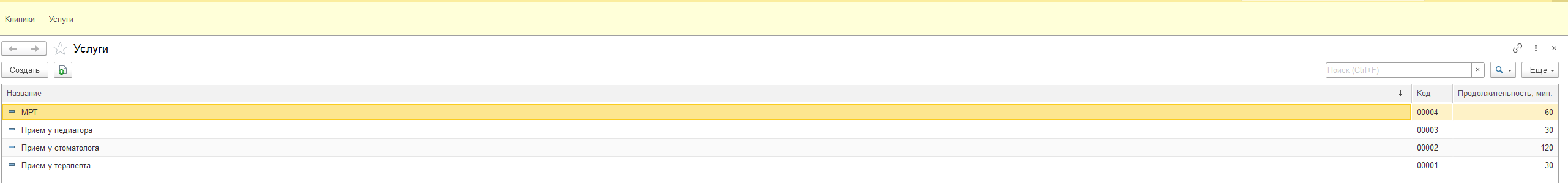
Создадим свойство «Продолжительность»



В синониме укажем что она задается в минутах, тип данных число, максимум трехзначное, целое (точность 0), неотрицательное

Сделаем свойство обязательным для указания  


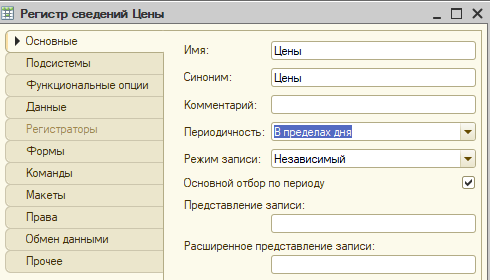
Запустим отладку, при попытке указать услугу без продолжительности, мы получим ошибку  


Заполним каталог услуг  


**Объект конфигурации «Регистр сведений»**

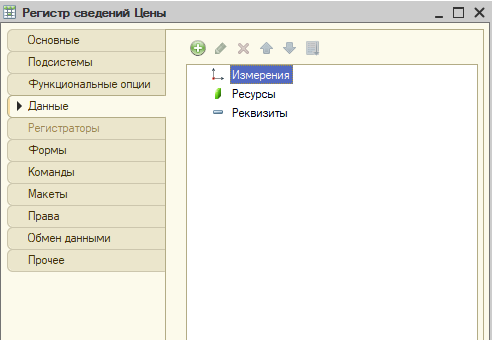
В случае, когда мы хотим фиксировать в наших объектах периодически изменяемые данные и нам необходимо отслеживать\хранить историю изменения значений, необходимо использовать объект конфигурации «Регистр сведений».

Наиболее очевидный пример использования данного объекта – фиксация изменения курса валют

Создадим регистр сведений для хранения цен  


Для включения функции хранения истории изменения объекта необходимо указать периодичность.

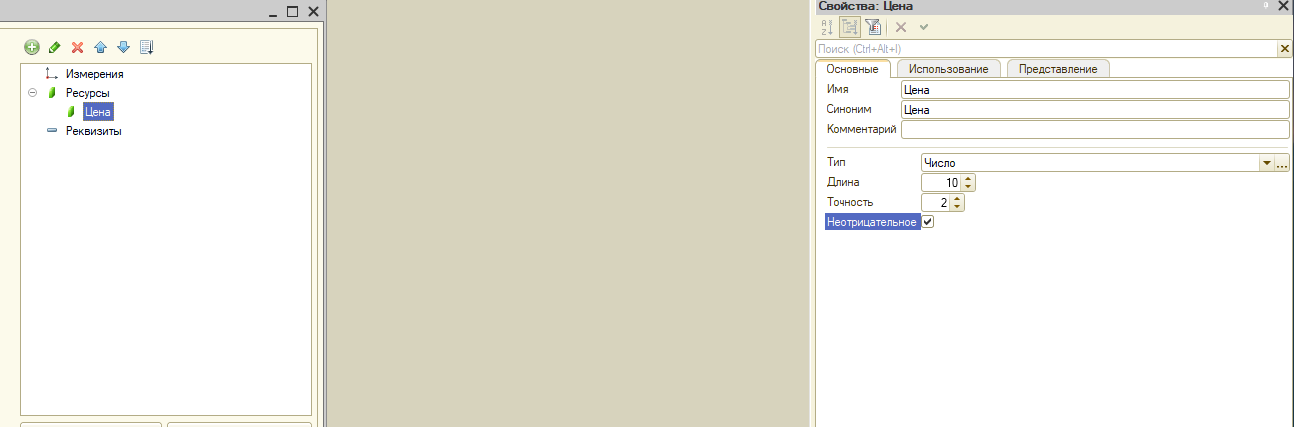
Периодичность «В пределах дня» означает что значение не может изменяться чаще чем раз в день



Измерения – в рамках какого объекта происходит изменение

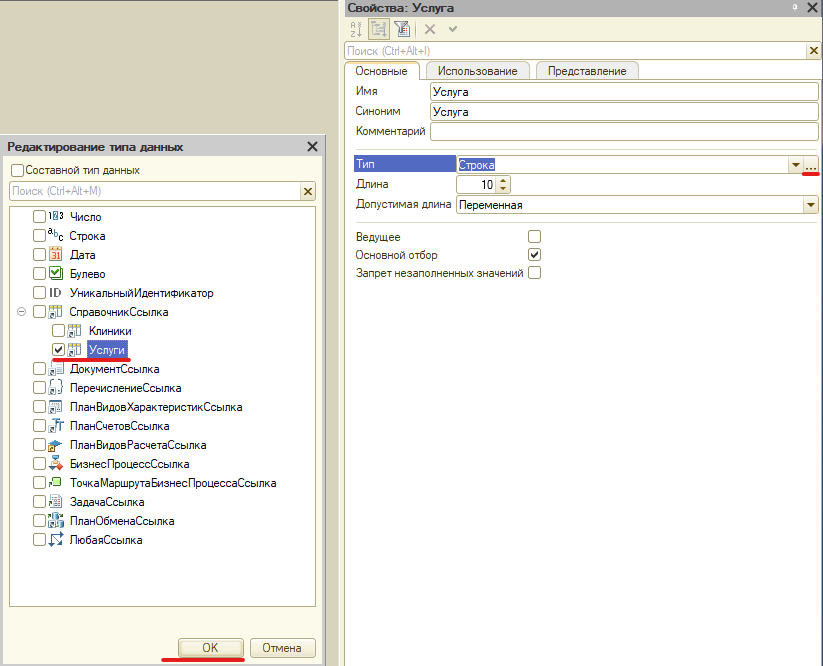
Ресурсы – изменяемый объект

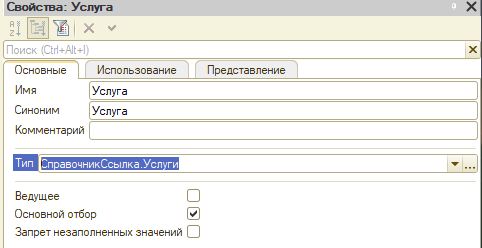
Добавим ресурс «Цена»



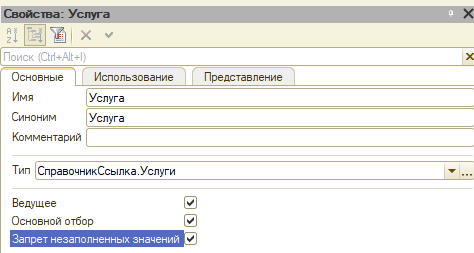
Добавим измерение «Услуга»

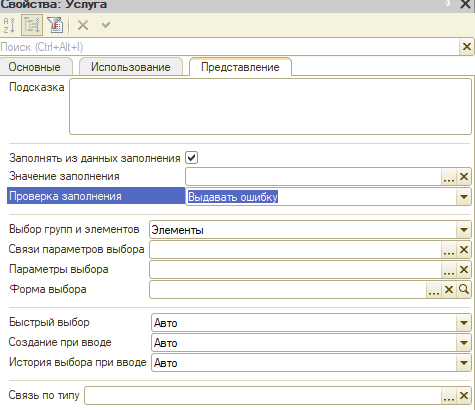
Т.к. изменение цены должно фиксироваться не в рамках абстрактной внесенной руками в регистр услуги, а в рамках услуг указанных в справочнике, необходимо указать как тип данных наш справочник описывающий услуги





Необходимо также указать свойство «Ведущее» чтобы запись о услуге в регистре была связана с записью о услуге в справочнике, т.е. таким образом все изменения произведенные в справочнике отобразятся в регистре. Также включим запрет незаполненных значений.





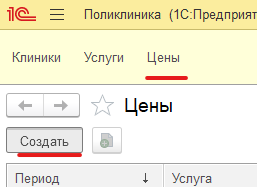
Свойства «Запрет незаполненных значений» и «Проверка заполнения» в системе 1С не исключают, а дополняют друг друга.

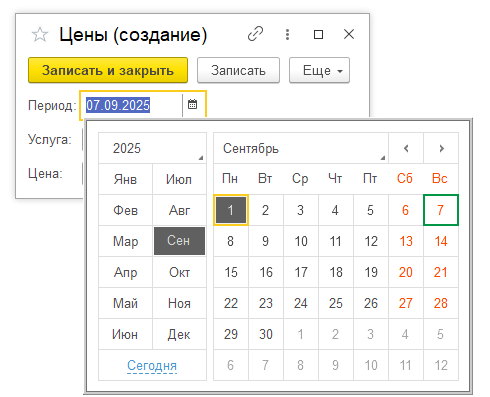
Свойство «Проверка заполнения» проверяется только в интерактивном режиме работы. Если оно установлено в значение «Выдавать ошибку» и поле не заполнено, пользователь получает сообщение об ошибке, и документ не проводится. Если документ формируется или проводится программно, это свойство не анализируется, и проверка не срабатывает.

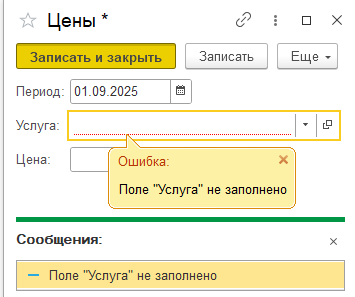
Свойство «Запрет незаполненных значений» для измерения регистра проверяется при каждой записи данных в регистр, независимо от того, программно она выполняется или интерактивно. Если значение этого свойства равно «Истина» и измерение не заполнено, при записи данных в регистр возникает исключение, и запись не производится.

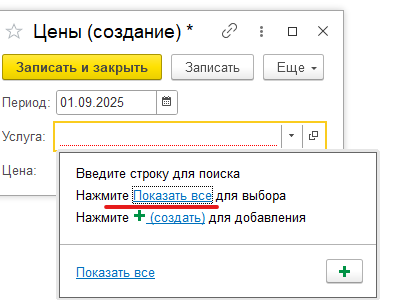
Таким образом, свойство «Запрет незаполненных значений» обеспечивает проверку при записи данных, а «Проверка заполнения» — при интерактивном вводе.

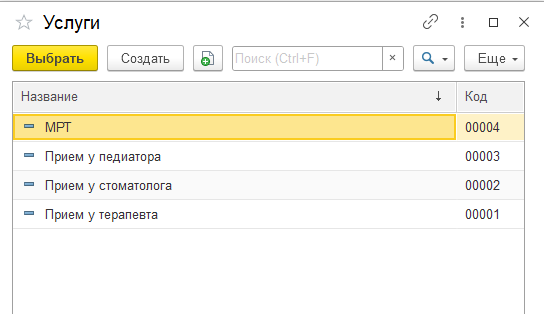
Запустим отладку, чтобы посмотреть изменения

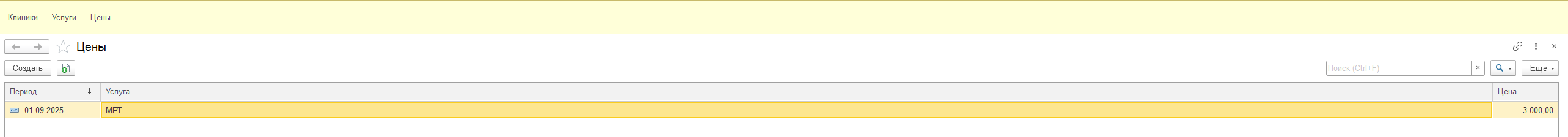












Заполним цену для каждой услуги

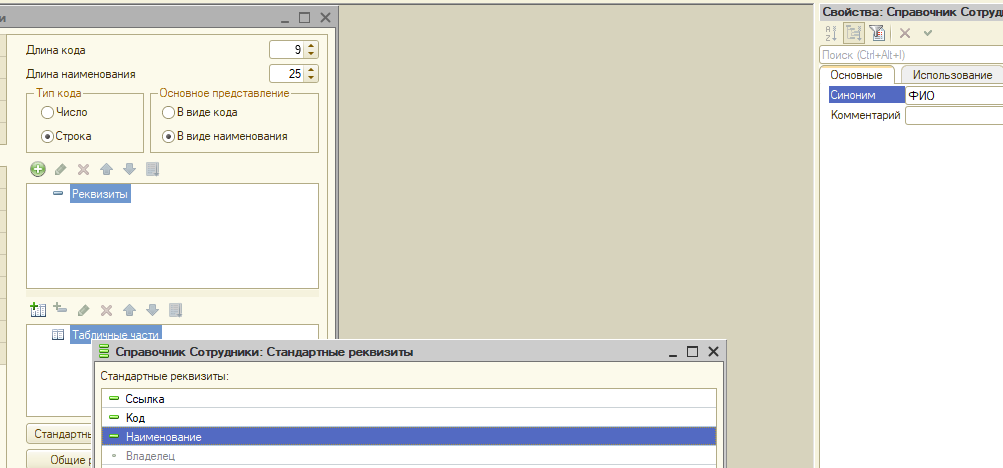


**Продолжение заполнения справочников**

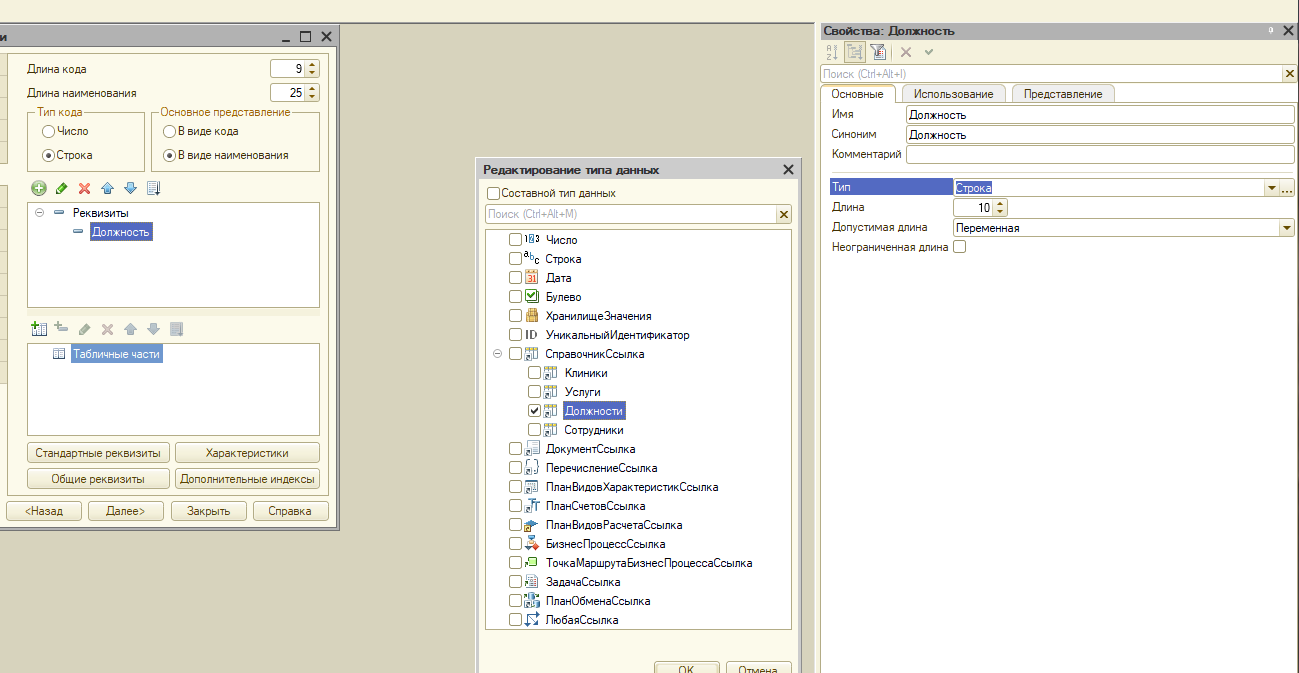
Создадим справочник для хранения сотрудников, но сперва создадим справочник для фиксации должностей сотрудников, во избежание разнонаписания.

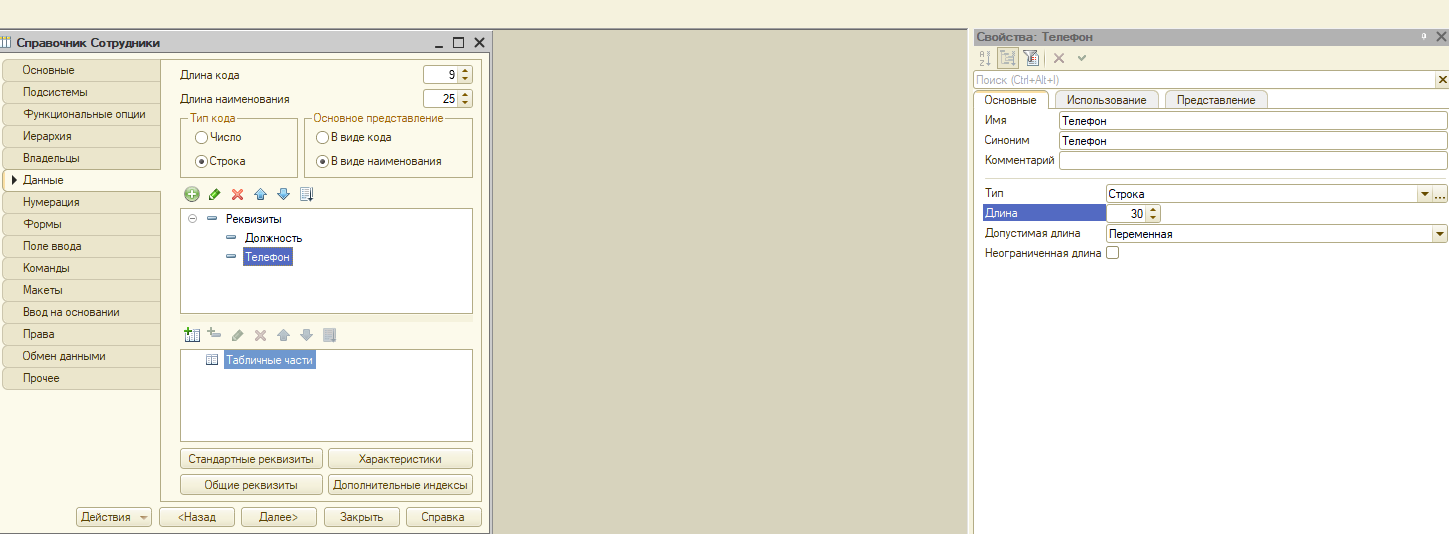
Для справочника «Должности» нам достаточно стандартных атрибутов

В качестве наименования для элемента справочника «Сотрудники» будем передавать ФИО

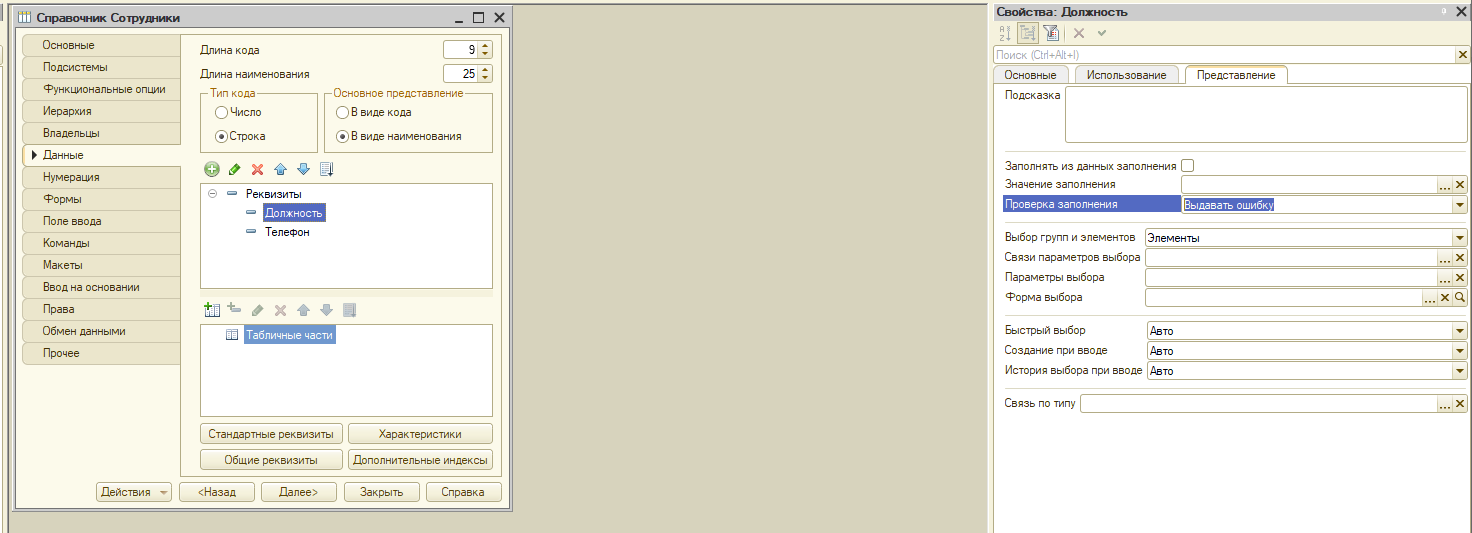


Должность будет браться из справочника



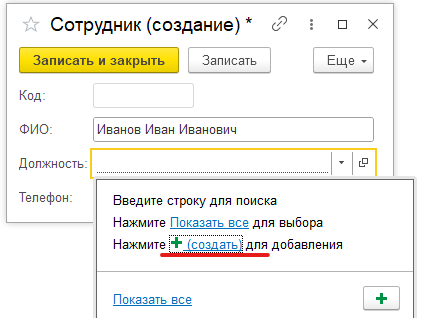


Должность и телефон будут обязательными атрибутами

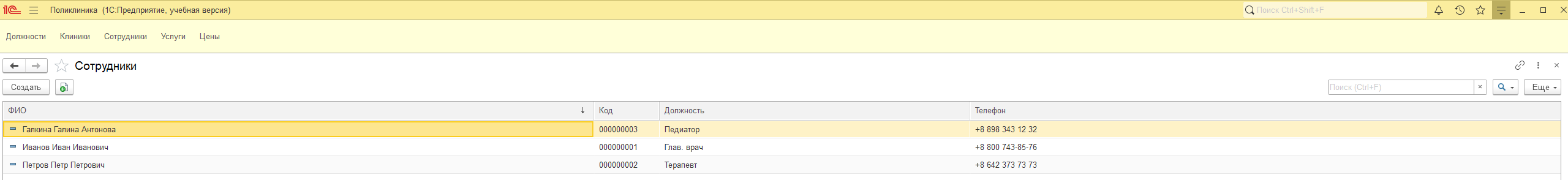


Запустим отладку, заведем сотрудников

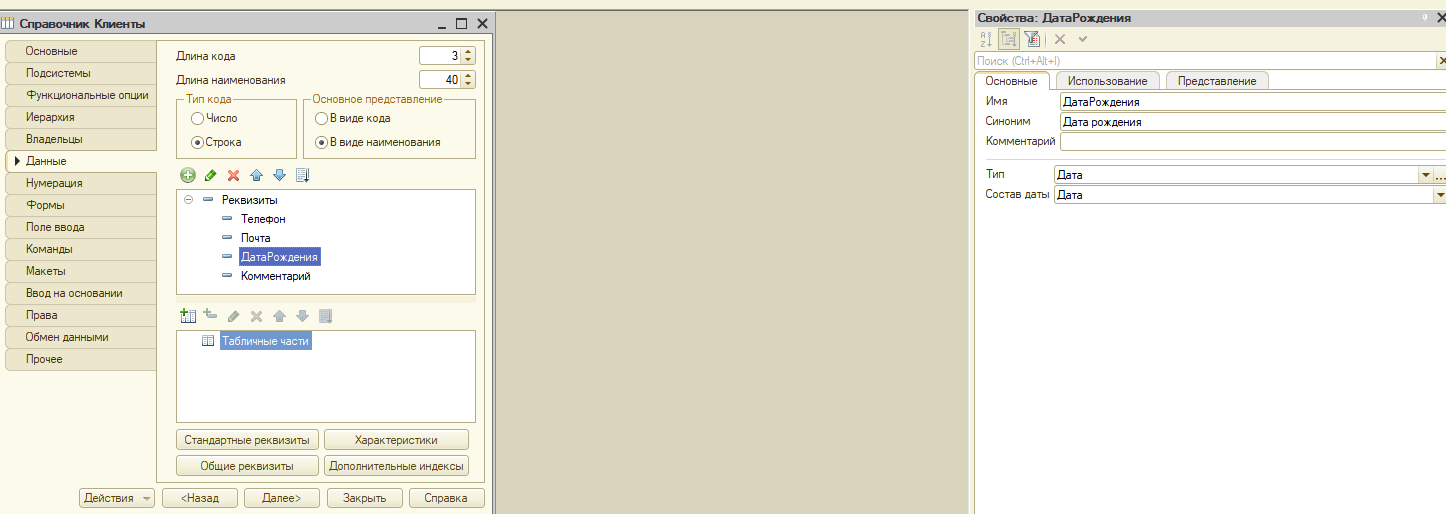
Заполнять справочник должностей возможно в момент заполнения справочника сотрудников



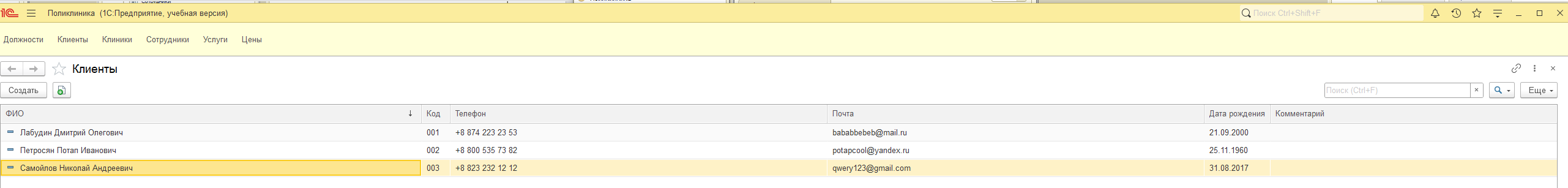
Заполним справочник



Создадим и заполним справочник клиентов



Все атрибуты кроме комментария обязательны



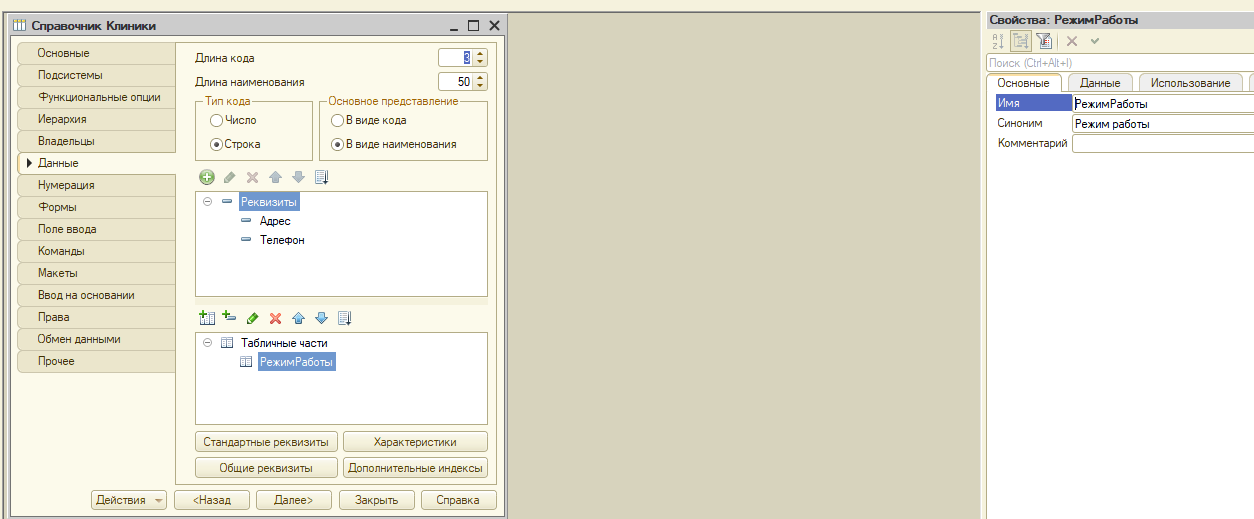
**Табличные части и перечисления**

Табличная часть – элемент справочника предназначенный для хранения информации в виде таблицы

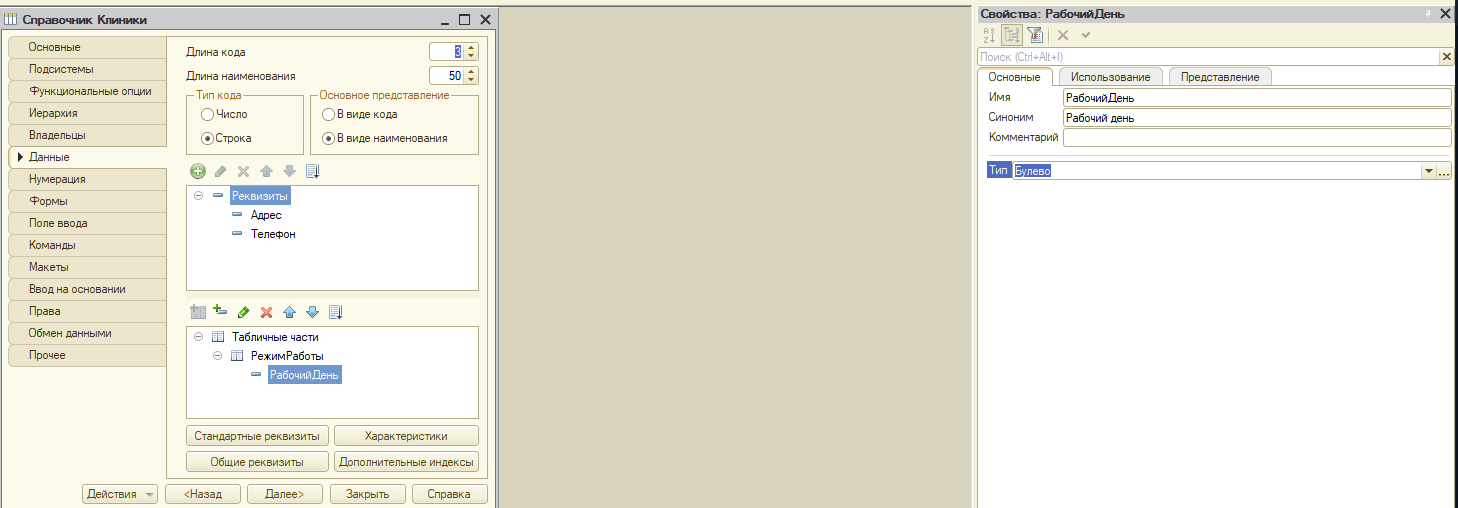
Перечисления – объект для хранения неизменных массивов объектов

В нашем случае табличная часть будет использоваться для хранения расписания клиники.

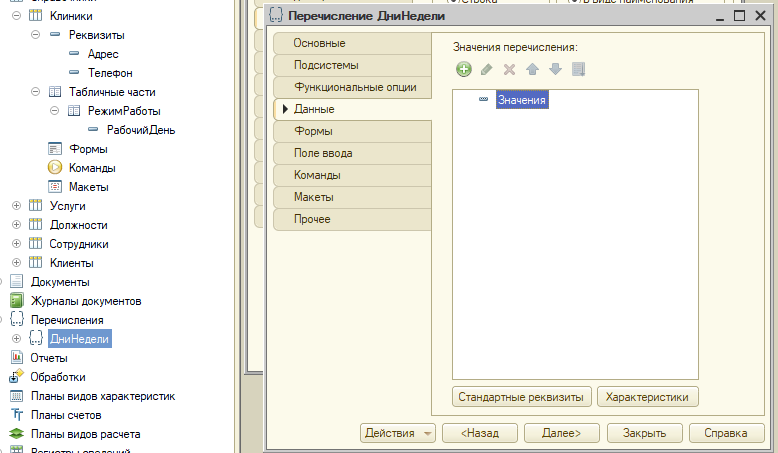
Создание табличной части



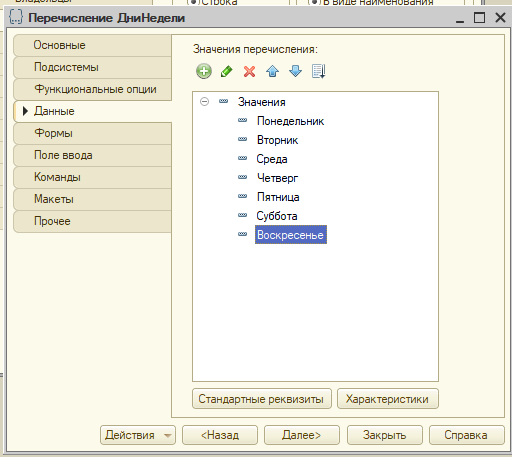
Добавим свойство указания на то является ли день недели рабочим



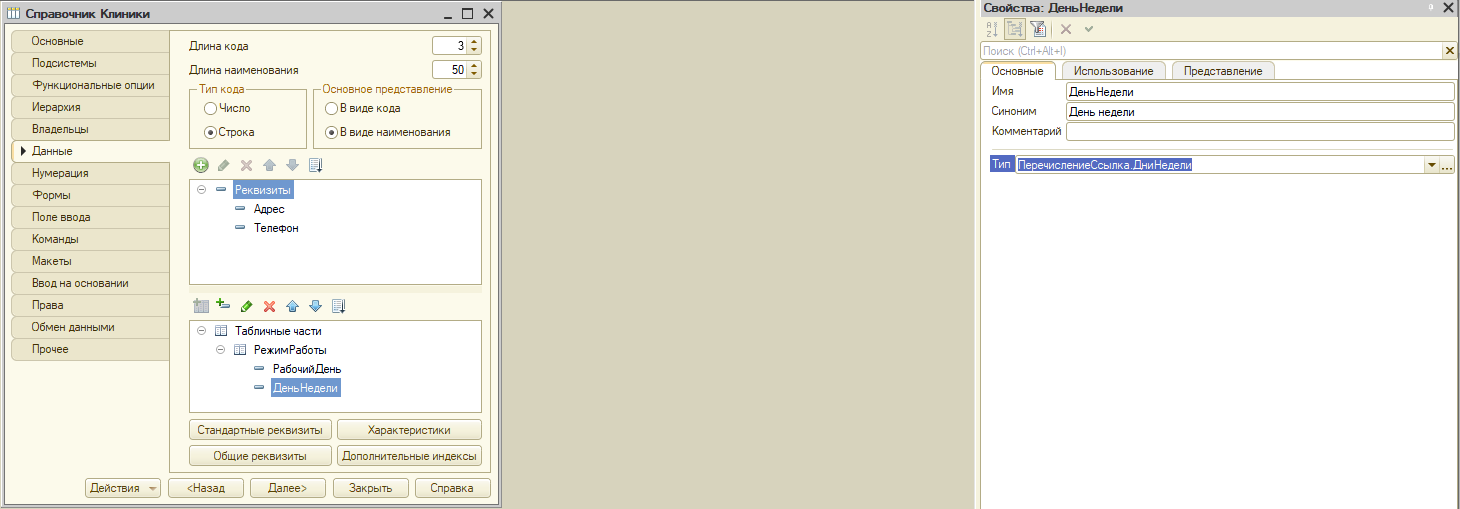
Создадим перечисление «ДниНедели»



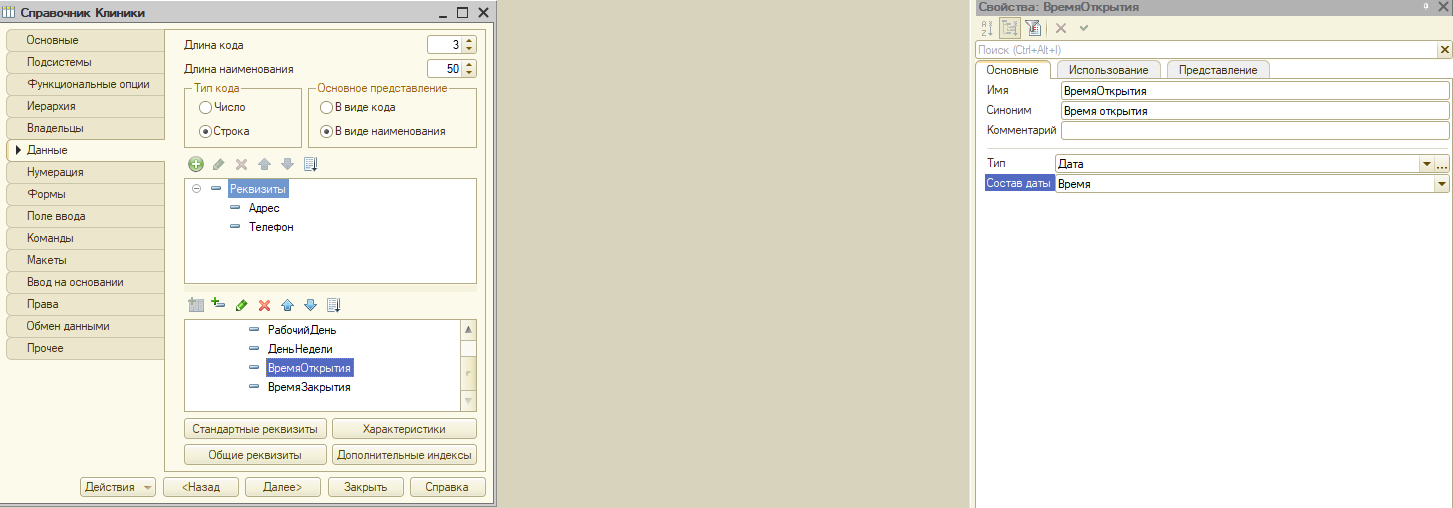
Заполним значения



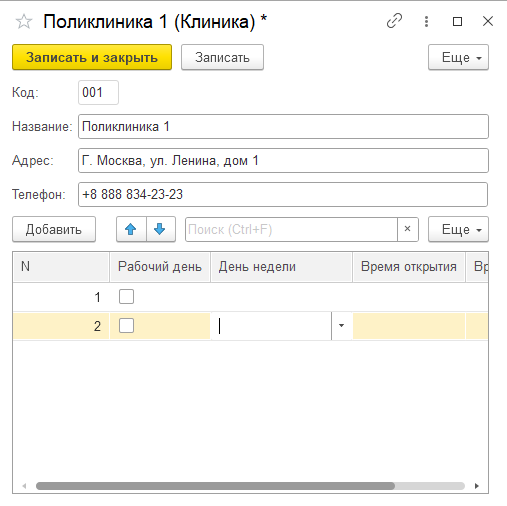
Добавим выбор дня недели в табличную часть



Добавим время открытия и закрытия с типом данных дата и указанием только времени



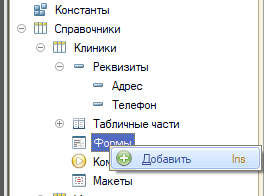
По итогу мы получили таблицу, в которую необходимо самостоятельно заводить каждый день недели для клиники, а также неудобным образом выбирать свойство рабочего дня, с ненужным указанием номера строки

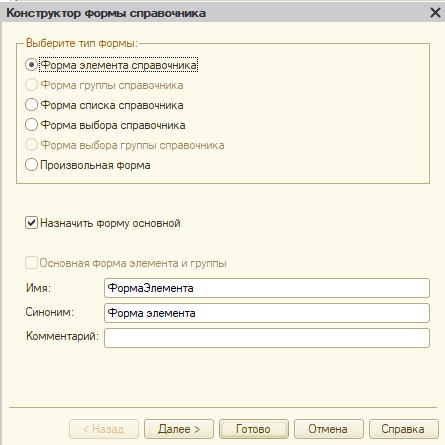


Устраним описанные выше недочеты, а также реализуем автоматическое создание таблицы с заполнением дней недели при создании новой клиники

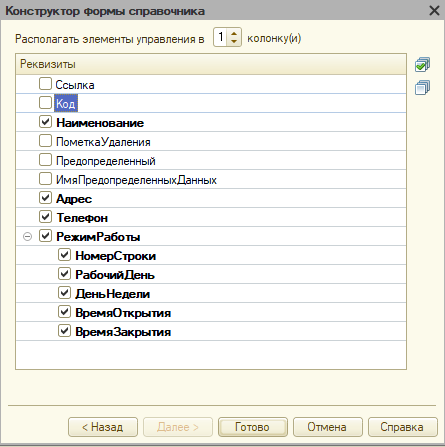
**Формы**

Форма – объект справочника, в рамках которого происходит редактирование отображения содержимого справочника. На прошлом скриншоте представлена форма, собранная по умолчанию, которая не отображается как отдельный объект

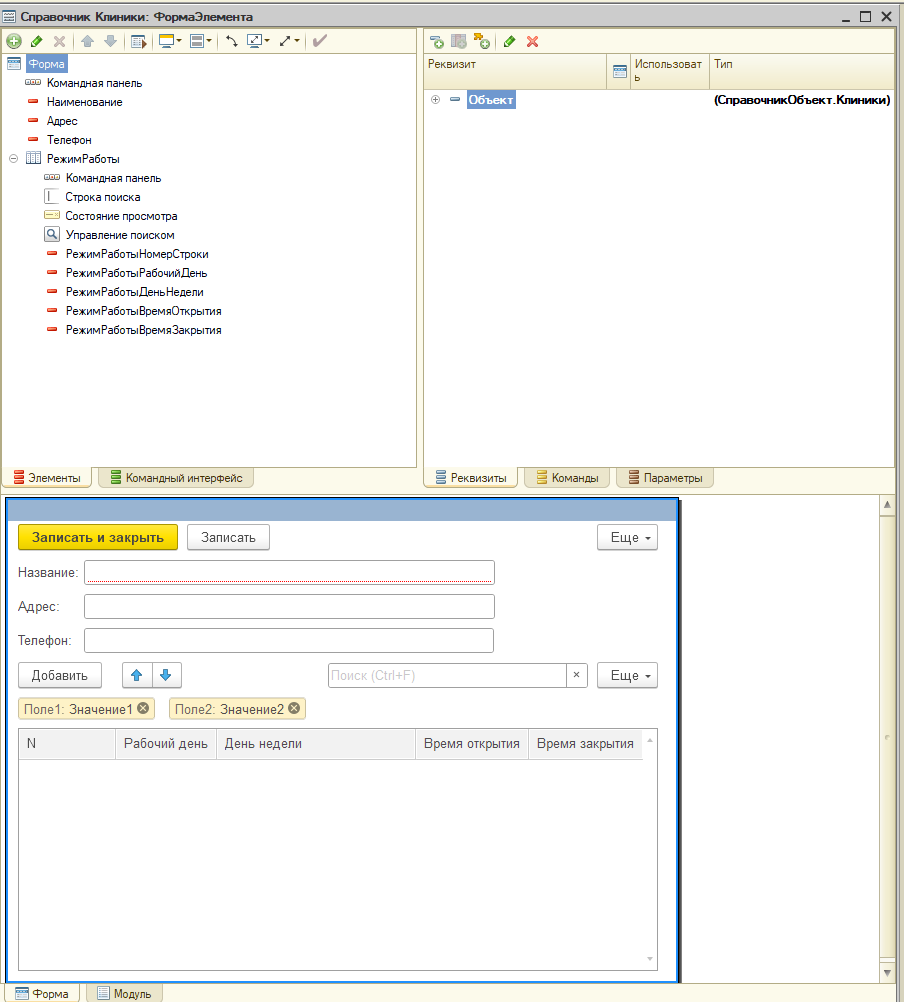




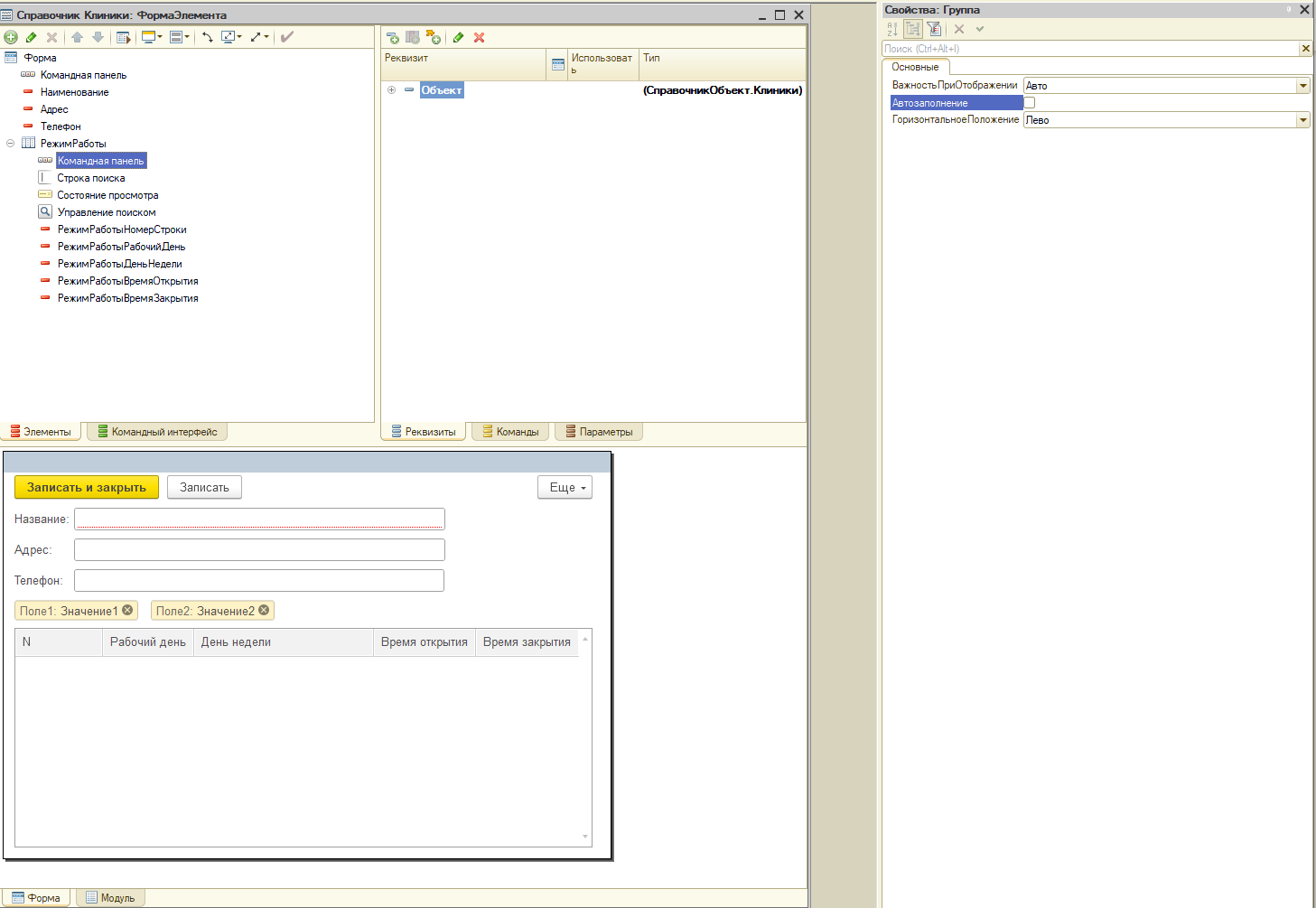
Скроем отображение кода



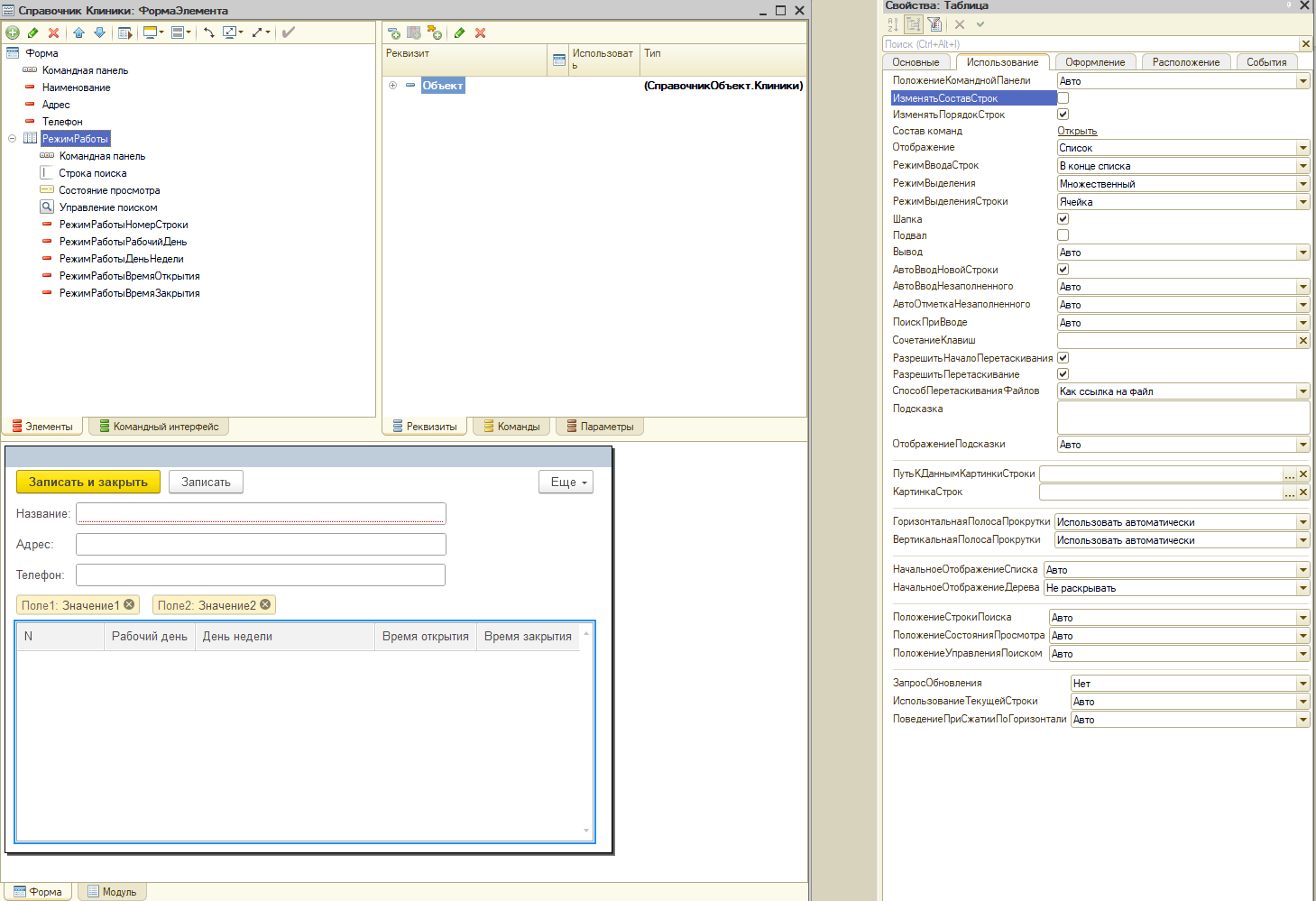
Перед нами открылся конструктор отображения информации о записи справочника «Клиники»



Уберем автозаполнение у элемента «Командная панель» для удаления возможности добавления новых строк отдельной кнопкой, для которых будет реализовано автоматическое заполнение

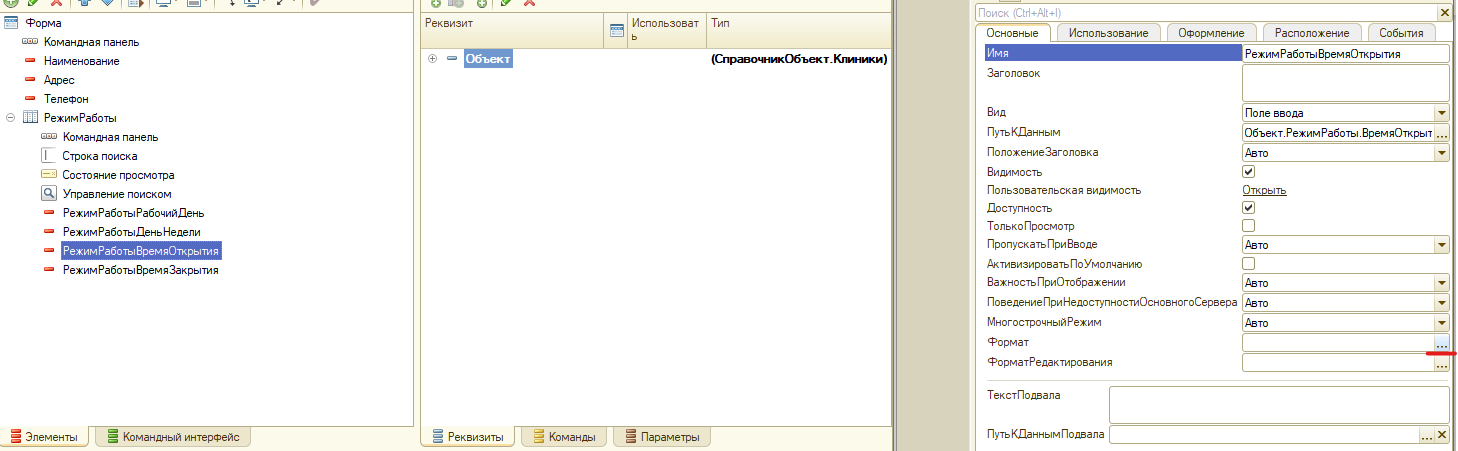


Перейдя в свойства табличной формы и выключив свойство «ИзменятьСоставСтрок», мы исключим возможность удаления, добавления, копирования строк в табличной форме

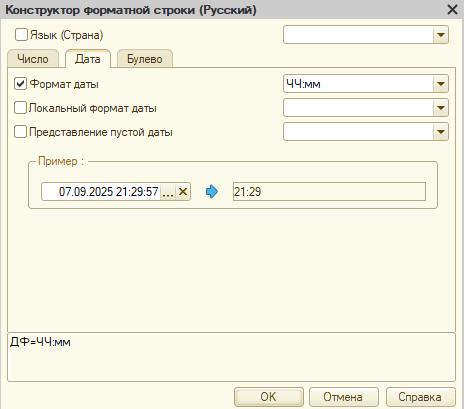


Удалим отображение номера строки

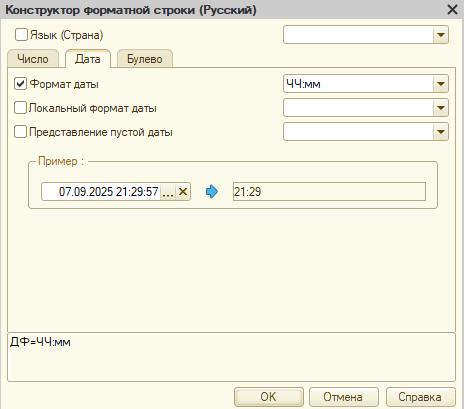
Укажем точность для времени открытия и закрытия клиники, по умолчанию мы можем указывать её вплоть до миллисекунд



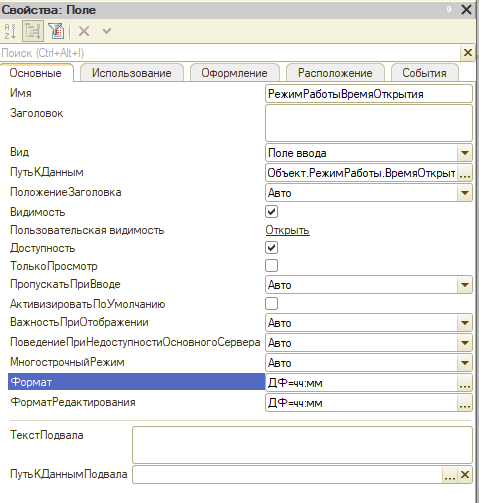
На вкладке «Дата» включим поле «Формат даты» и укажем значение ЧЧ:мм



Для проверки работы формулы нажмем “Enter” находясь в текстовом поле



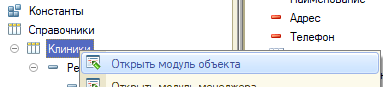
Формат редактирования автоматически проставился таким же как формат отображения



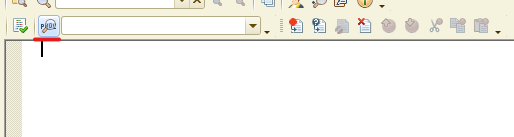
Скопируем формат на время закрытия

Рассмотрим один из способов автоматического заполнения табличных форм справочника при создании нового элемента справочника

Перейдем в редактор кода для справочника



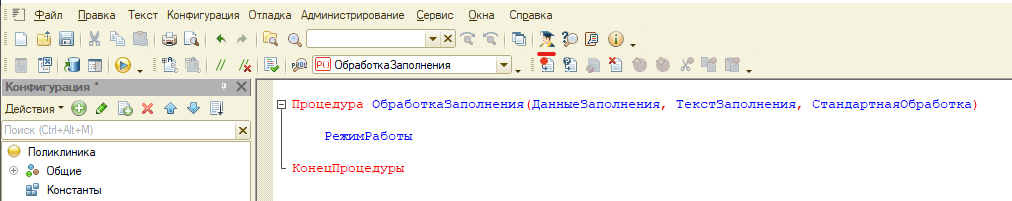
Рассмотрим список процедур и функций доступных для справочника

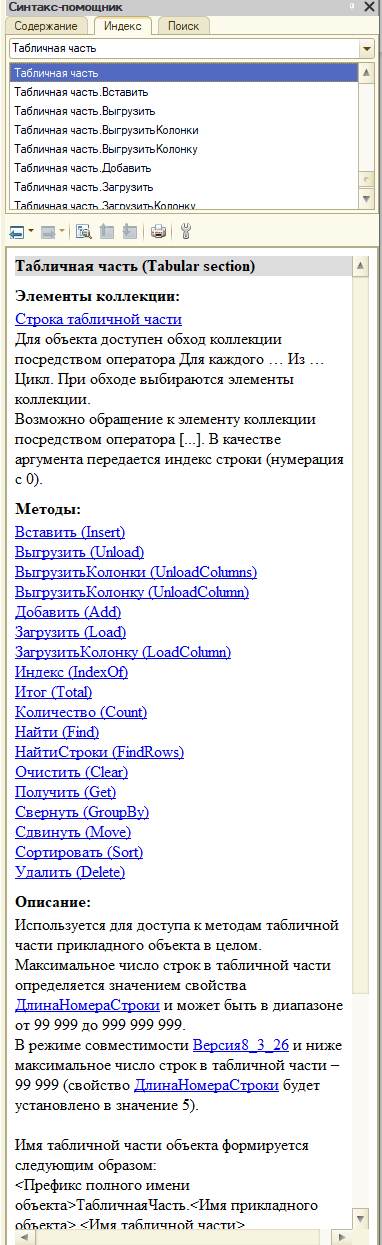


Перейдем в процедуру «Обработка заполнения» для заполнения табличной части в момент создания новой клиники

Т.к. у нас открыт редактор кода справочника «Клиники», мы можем обращаться ко всем объектам внутри него напрямую.

Также возможно открыть синтаксис-помощник для просмотра всех типов взаимодействия с различными типами объектов



Например, для объекта «ТабличнаяЧасть» доступно следующее описание

Т.к. наше перечисление дней недели находится в отдельном объекте, к нему необходимо обратиться отдельно. Мы можем обращаться к объектам ориентируясь по информации о объектах (метаданных) расположенных в дереве слева.

Информация о днях недели расположена по следующему пути:

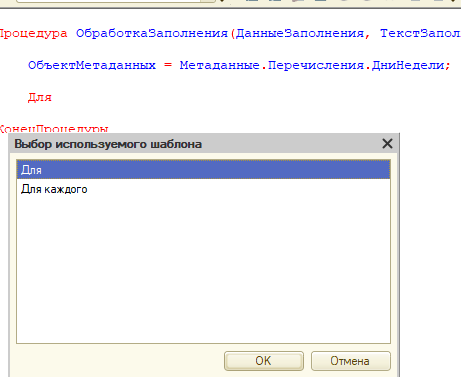


Как видно из кода, объект «Метаданные» находится в самом верху нашего древа объектов, после чего идет обращение к типу объектов, а уже после непосредственно к объекту.

Для использования данных и во избежание постоянного повторения длинной строки, их необходимо записать в переменную.

Далее необходимо обработать каждый день недели хранящийся в «ДниНедели» циклом.

В 1С встроена возможность автоматической подстановки синтаксиса. Для этого необходимо начать писать начало конструкции цикла «Для» и нажать сочетание клавиш ctrl+q



Далее необходимо выбрать «Для каждого» и трижды нажать «ОК»

Между «для каждого» и «из» необходимо указать название переменной, которое будет хранить в себе перебираемые значения массива дней недели.

Между «Из» и «Цикл» необходимо указать перебираемый массив объектов.

В нашем случае это будет «ОбъектМетаданных.ЗначенияПеречисления»

Разберем подробнее эту строку:

«ОбъектМетаданных» содержит в себе путь к перечислению дней недели Метаданные.Перечисления.ДниНедели;

В свою очередь все дни недели хранятся не непосредственно в перечислении, а во вкладке значения, поэтому для обращения к значениям перечисления необходимо после пути к перечислению обратиться к объекту ЗначенияПеречисления;



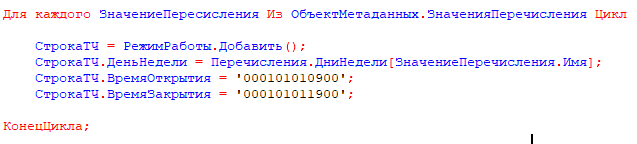
В коде выше описано автоматическое добавление дней недели. Разберемся подробнее:

СтрокаТЧ – переменная хранящая информацию о добавляемой строке, в данном случае в неё записан метод «Добавить()» примененный к табличной части «РежимРаботы», т.е. добавление информации в новую строку по названию колонки;

СтрокаТЧ.ДеньНедели – в развернутом виде РежимРаботы.Добавить().ДеньНедели вызывает добавление переданного в неё значения в колонку ДеньНедели;

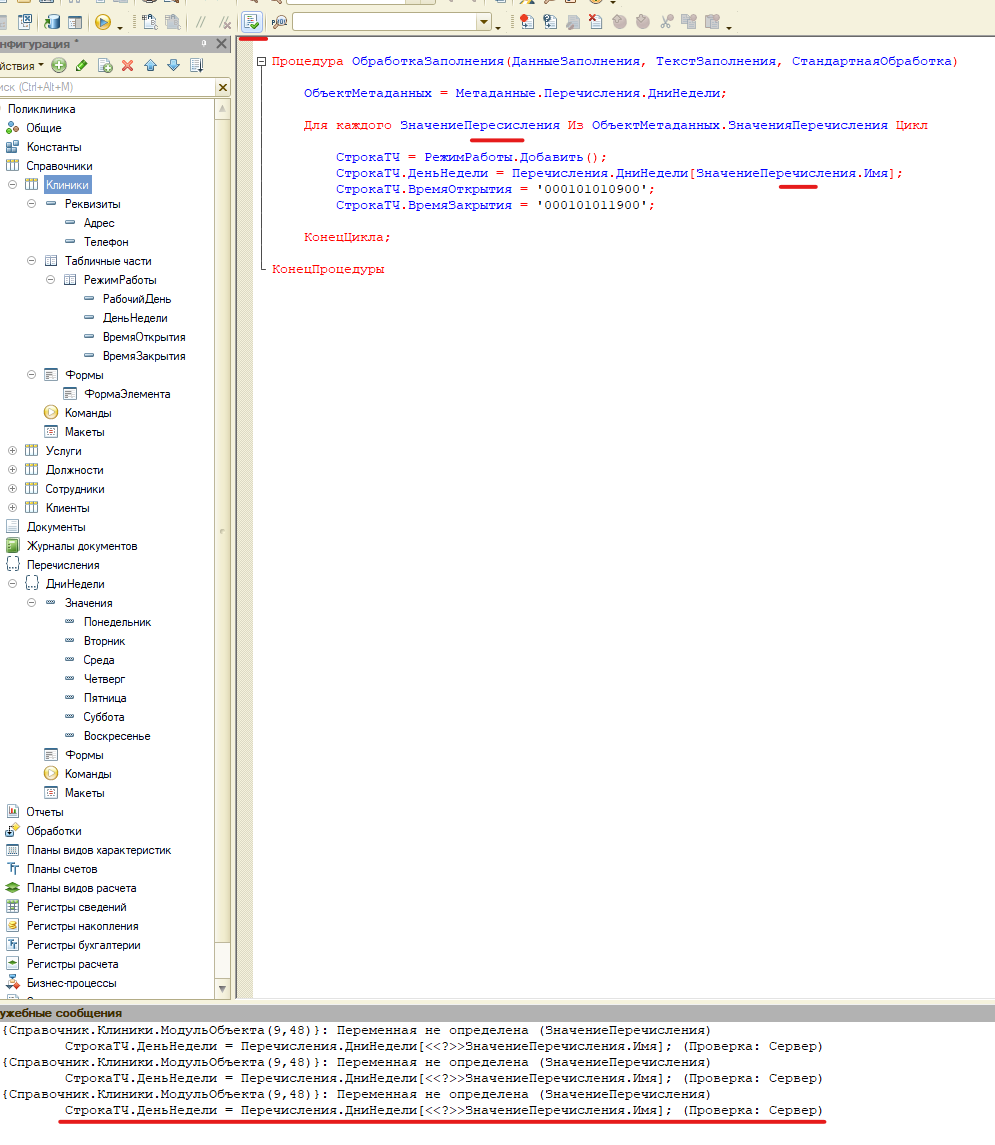
Перечисления.ДниНедели[ЗначениеПеречисления.Имя] – Т.к. нам необходимо записать в таблицу день недели как объект, взятый из перечисления, а не название дня недели, нам необходимо обращаться непосредственно к объекту, а не к информации о нем(метаданным). В данном случае мы обращаемся к массиву дней недели из одноименного перечисления (для обращения к элементу массива используются квадратные скобки) и находим объект хранящий каждый день недели по наименованию объекта полученному из метаднных.

Теперь добавим указание по умолчанию времени открытия и закрытия клиники

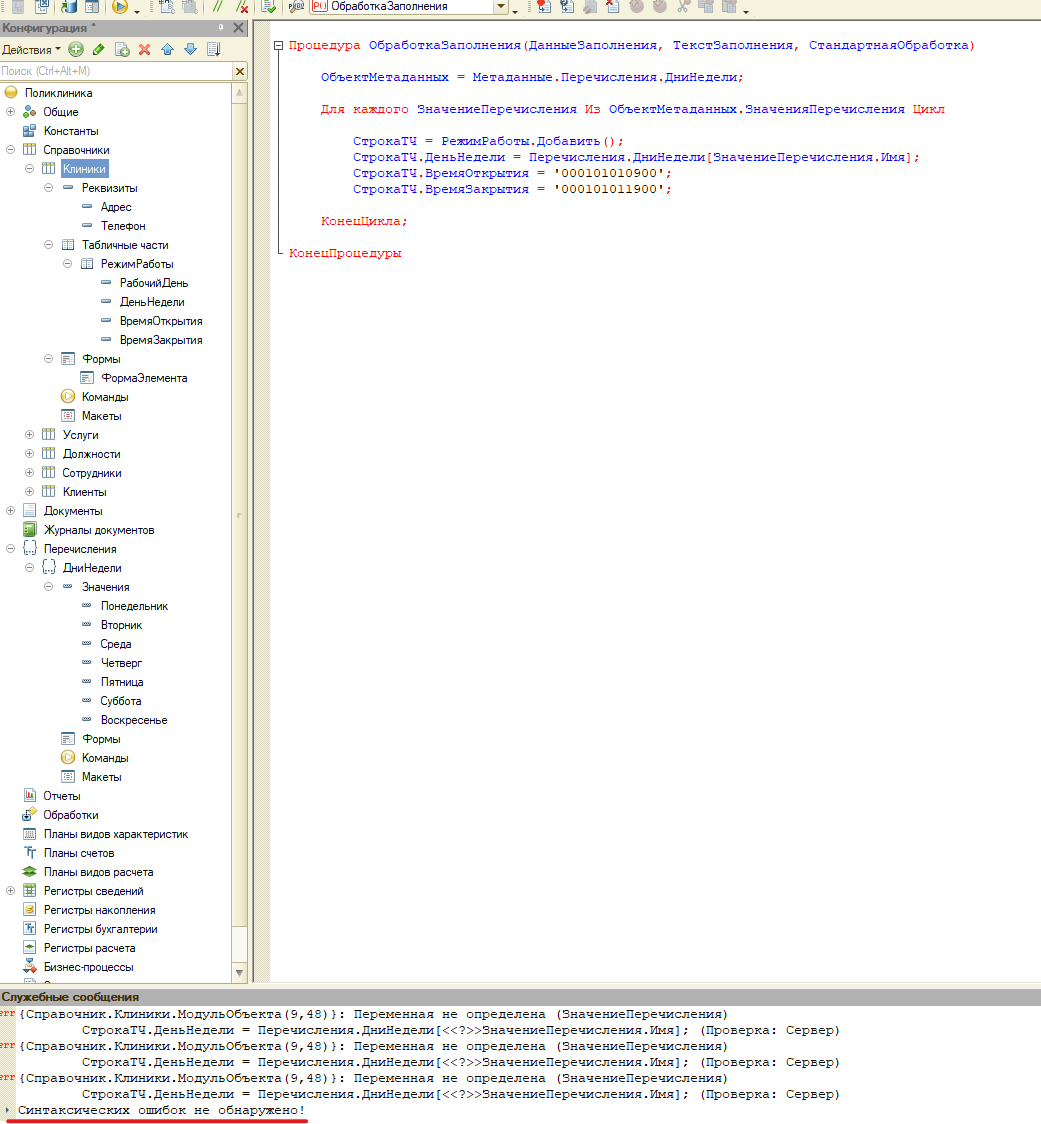


В описанном выше коде дата и время записаны в дискретном виде, где 0001 – первый год, 01 - первый месяц, 01 – первый день, 09\19 – девять утра\семь вечера, 00 – ноль минут

Выполнив проверку модуля, обнаружим ошибку в переменной



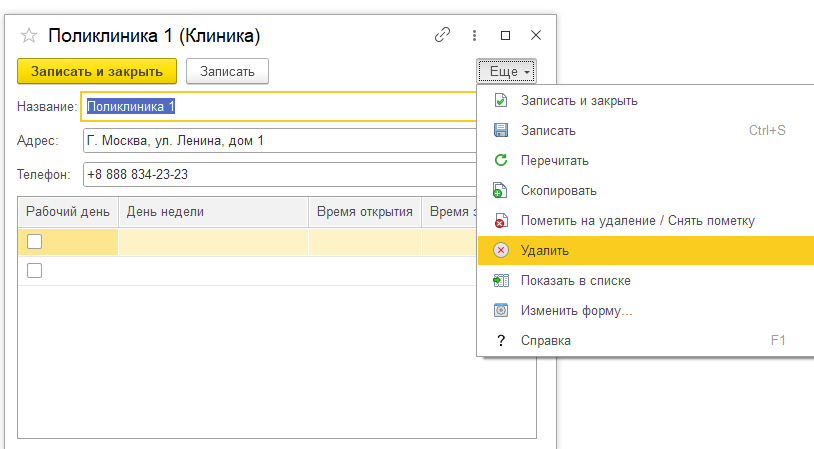
Исправим ошибку и выполним проверку повторно



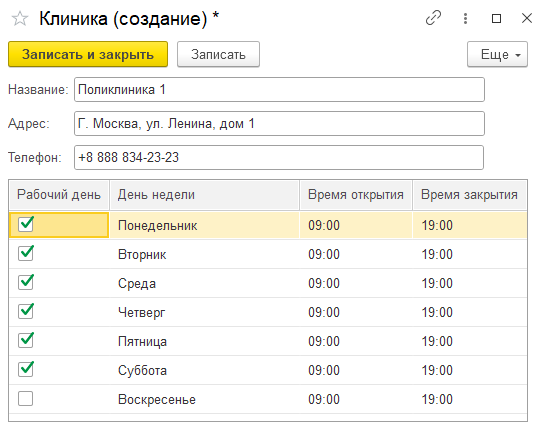
Запустим отладку повторно

Скопируем данные из имеющейся клиники

Удалим клинику



Создадим клинику заново

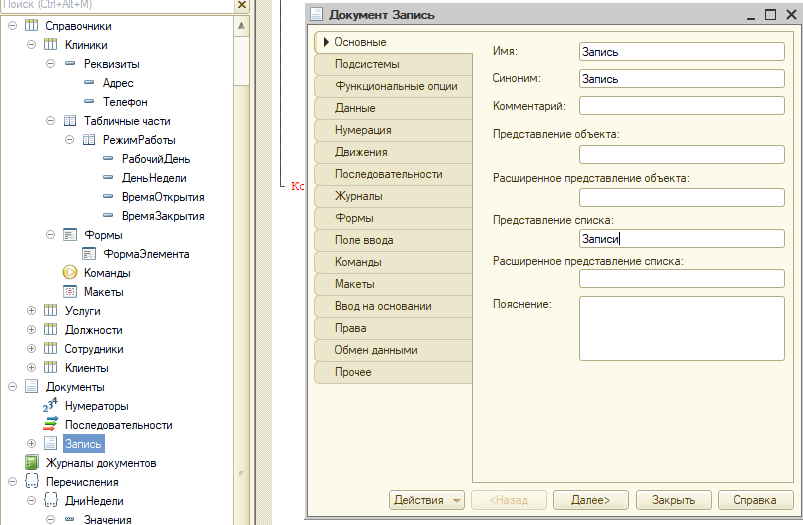


Как можно увидеть дни недели проставились автоматически, и мы не можем добавлять, удалять, копировать строки. Формат даты отображается корректно

**Документы**

Документ – основной объект конфигурации, отражающий информацию о определенном типе операций. В нашем случае это будет запись в клинику.

Создадим документ «Запись».



Представление списка – название списка представляющего множество операций определенного типа (в большинстве случаев множественное число названия).

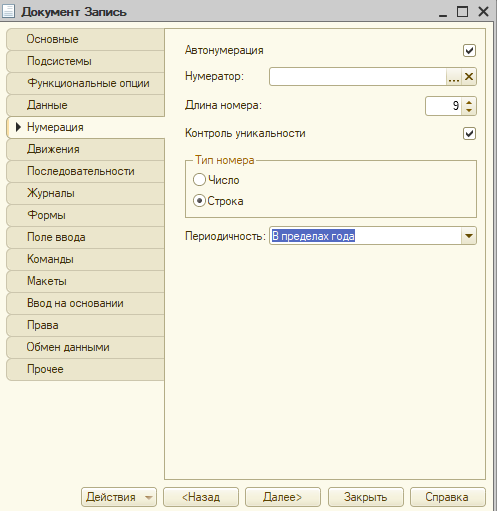
В отличие от справочников документ имеет два встроенных реквизита: Дата и номер. Дата по умолчанию указывается как текущая. Номер указывается согласно настройкам нумерации.

Оставим автонумерацию включенной. Длину номера оставим по умолчанию.

Числовой тип номера имеет вид: 1, 2, 3 и т.д.

Строковый вид номера имеет вид: 000000001, 000000002, 000000003 и т.д.

Периодичность – по прошествии какого периода времени счетчик сбрасывается.

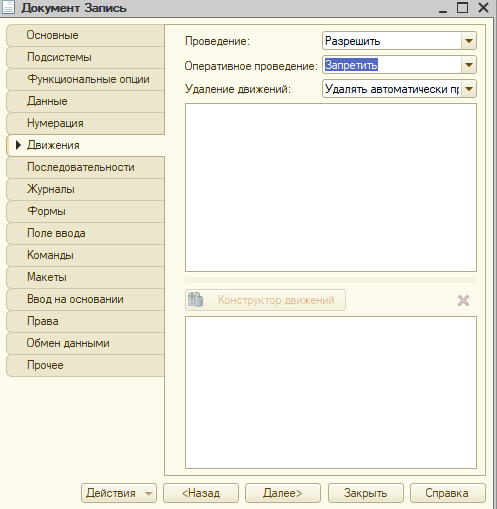


На данной вкладке описаны настройки проведения документа.

Проведение документа – отражение хозяйственной операции на соответствующих счетах бухгалтерского учета. Взятие документа в работу, после чего он становится частью программы, запускаются определенные предусмотренные процессы по обработке документа, а некоторые поля (например дата) становятся недоступными к изменению.

Мы не будем отключать проведение документа, т.к. основной функционал по обработке документа в таком случае станет недоступен.

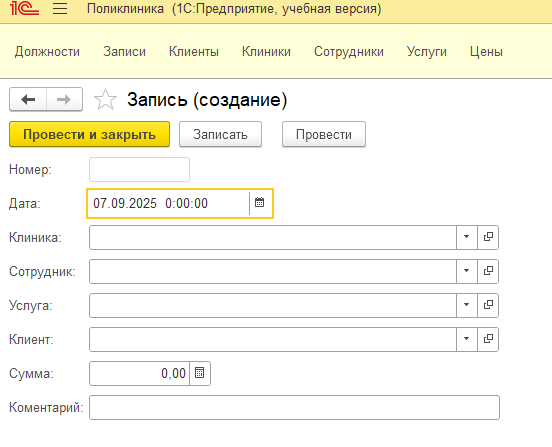
Мы запретим оперативно проведение, т.к. в таком случае будет недоступно формирование записей будущей датой

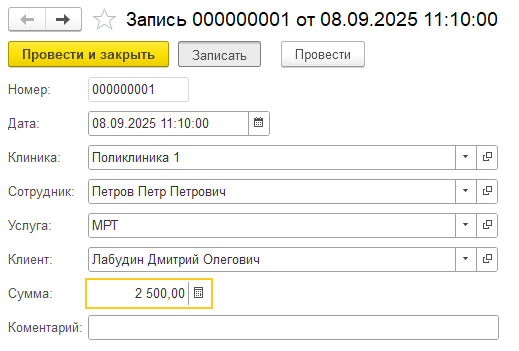


Перейдем на вкладку «Данные», укажем в реквизитах Клинику, Клиента, Сотрудника, Услугу в виде ссылок на одноименные элементы справочника. Также укажем как трехзначное неотрицательное целое число длительность в минутах, как десятизначное неотрицательное число с точностью 2 сумму (также укажем её в справочнике услуга, если не указали ранее) и строку комментарий длиной 200 символов.

На данный момент длительность и сумма не будут автоматически извлекаться из поля «Услуга», для этого необходимо реализовывать отдельный функционал программным кодом.

Запустим отладку. Посмотрим как выглядит наш документ.

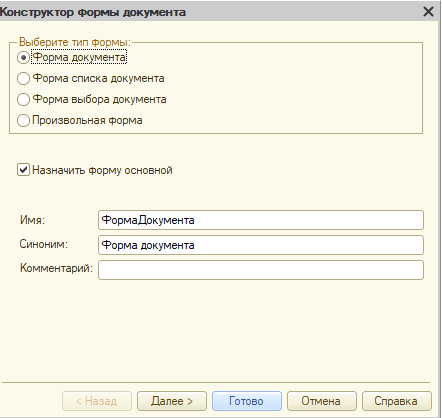




Выполним запись, без проведения документа.

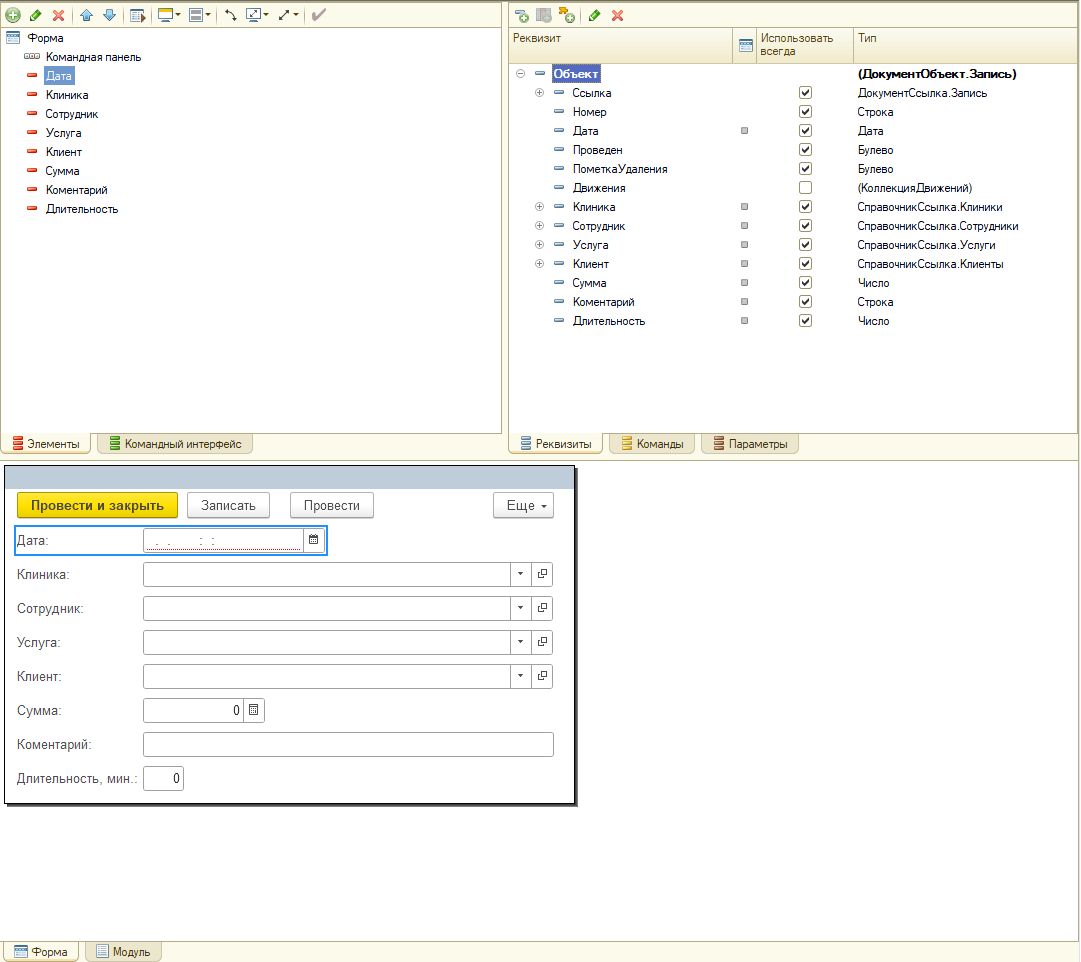
Перейдем к реализации через форму раздельной записи даты и времени, а также автоматическому заполнению суммы и длительности.

Создадим для документа «Запись» форму документа

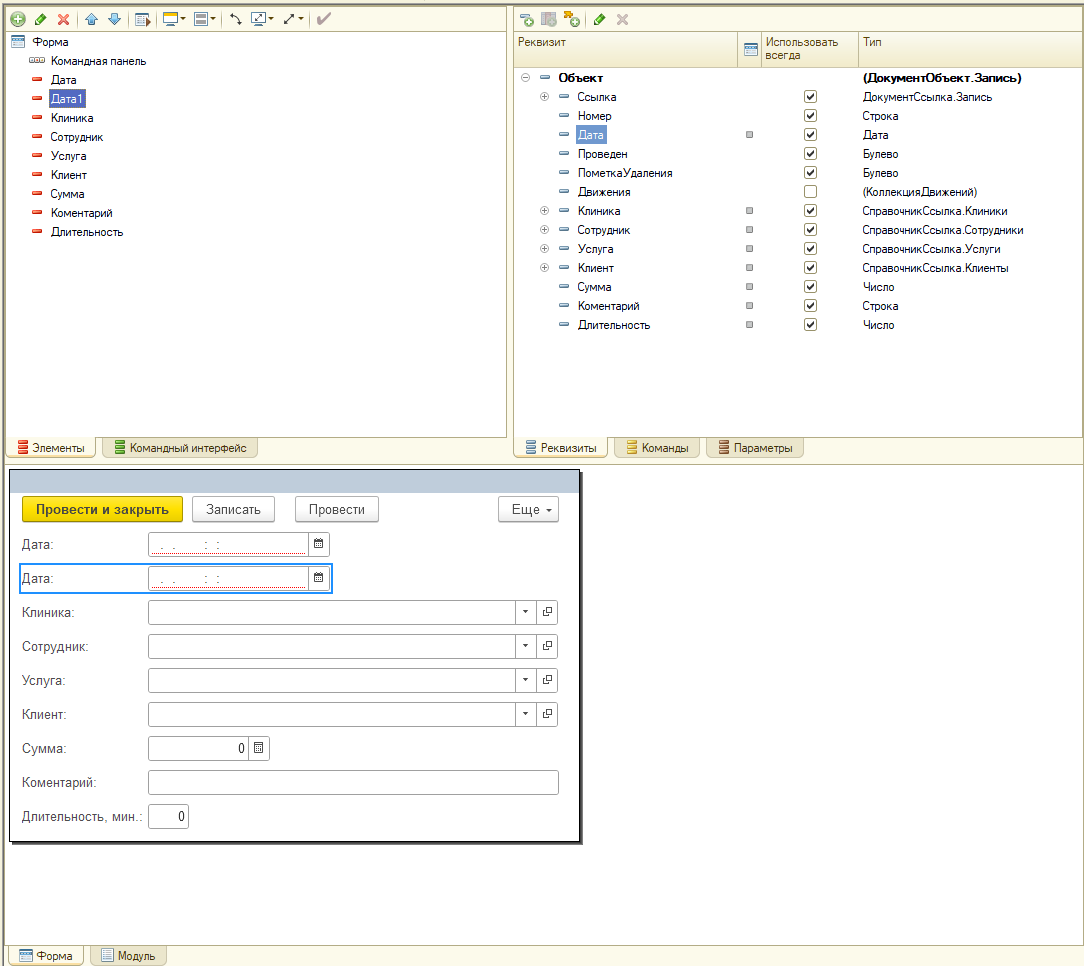


Удалим из формы номер.

Раскроем поле «Объект» в реквизитах документа

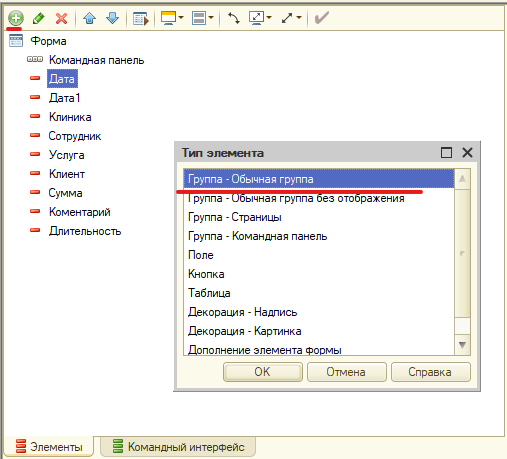


Перетянем влево поле «Дата»

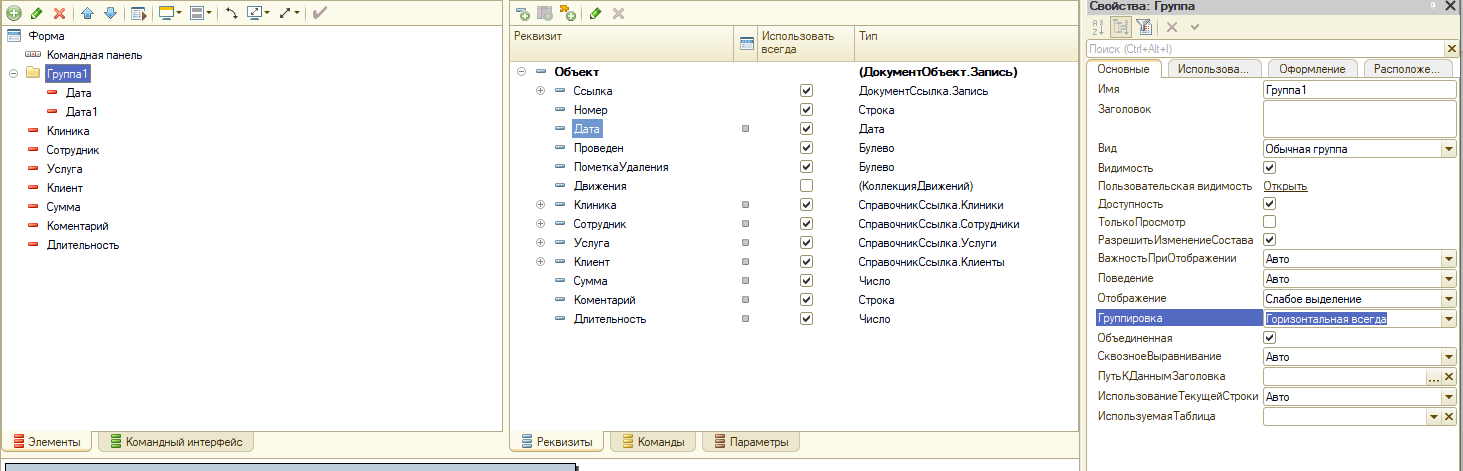


В элементах формы нажмем «Добавить» -> «Обычная группа»

Перетянем наши даты в группу

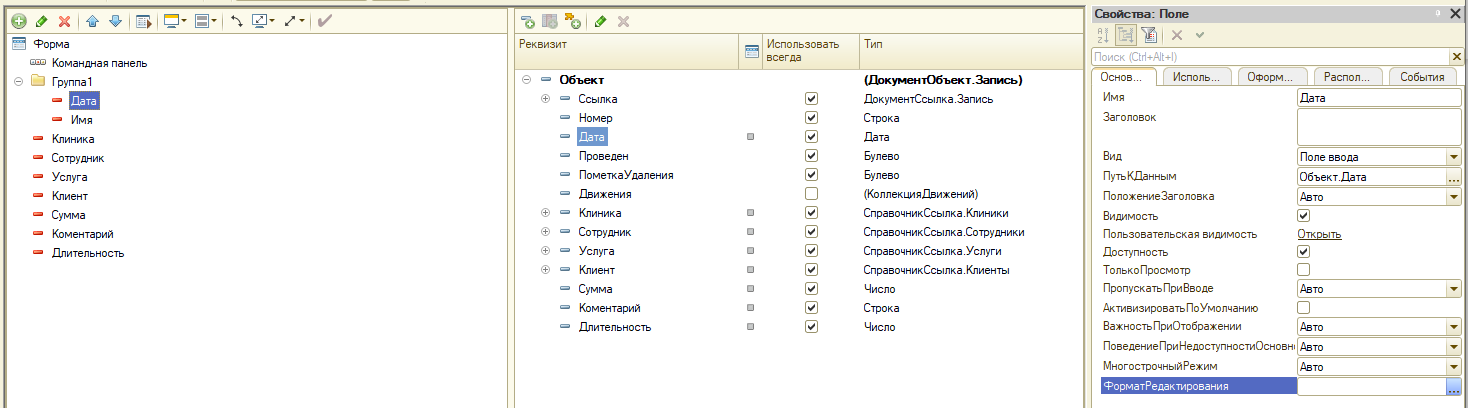


Для группы выберем тип группировки «Горизонтальная всегда»

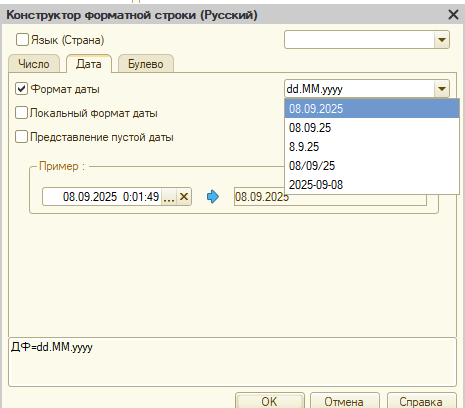


Для «Дата1» поменяем Имя и Заголовок на «Время»

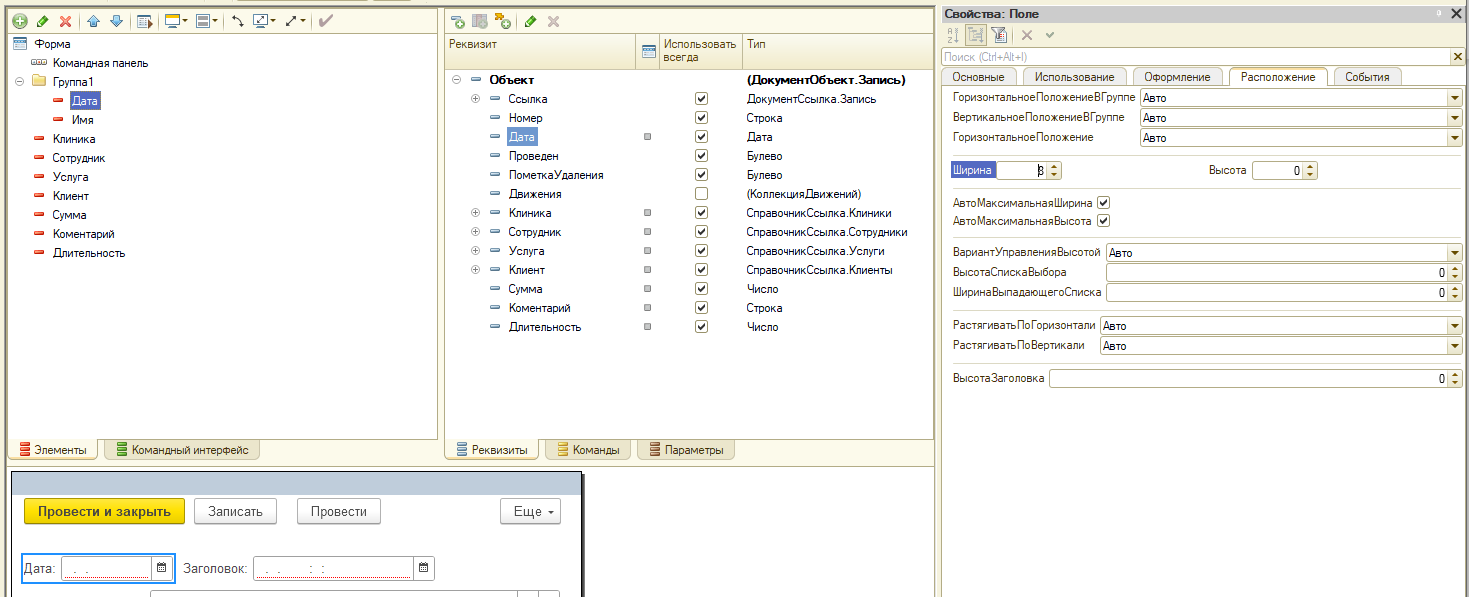
Перейдем формат редактирования Даты



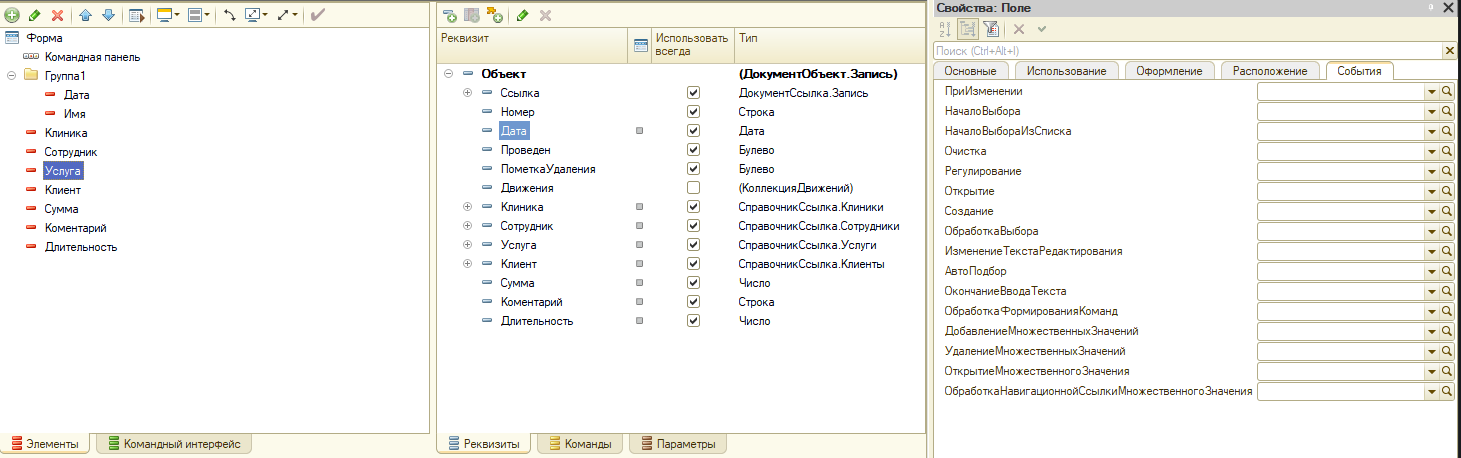
Выберем стандартный шаблон, нажмем «ОК»

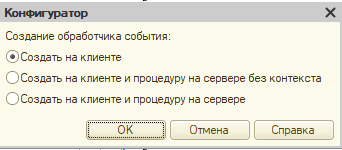
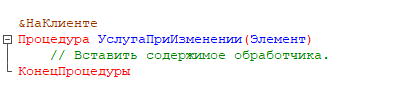


Перейдем на вкладку «Расположение» поля «Дата», укажем ширину 8



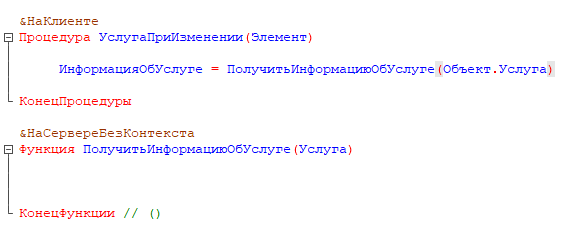
Укажем для времени формат ЧЧ.мм и ширину поля 4

На вкладке «События» свойств любого объекта мы можем настроить обработчики событий которые будут запускать код при различных событиях  
  
Например, если мы хотим чтобы при выборе Услуги у нас подтягивалась её стоимость и длительность, необходимо выбрать обработчик «ПриИзменении»

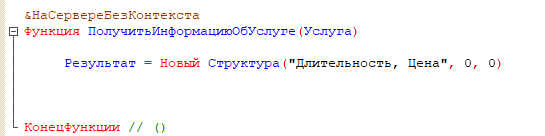
Нажмем на лупу, выберем  
  
На выводе получим следующий код  


Программный код может выполняться на сервере, или клиенте в зависимости от реализуемой задачи. Ранее написанный программный код (код модуля справочника «Клиника») выполняется только на сервере, а потому не требует указания расположения.

В обычных формулах весь код выполняется на клиенте. В нашем случае форма является управляемой, а потому часть кода будет выполняться на сервере (получение информации о услуге, т.к. информация хранится в БД), а часть на клиенте (взаимодействия непосредственно с формой)

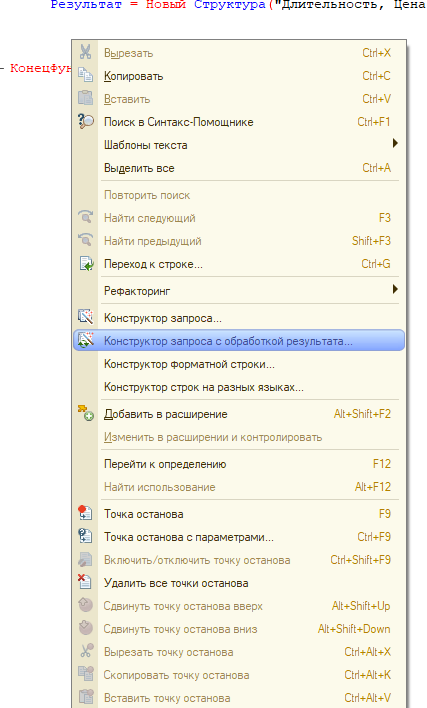
Опишем шаблон для получения информации о услуге из БД  


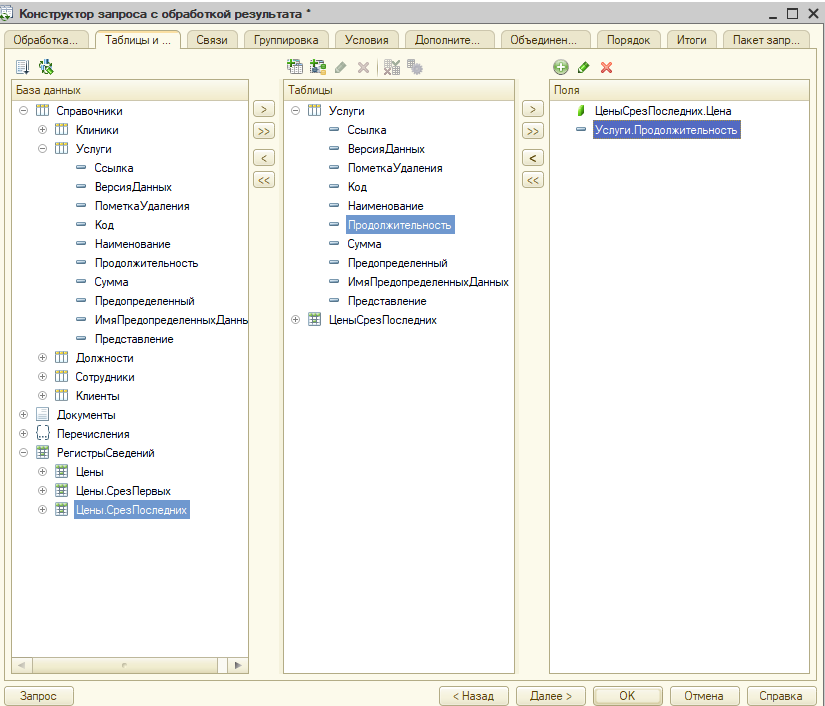
Где «Объект.Услуга» поле Услуга нашей формы



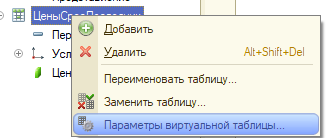
Объявим новую структуру, которую будет возвращать наша функция для дальнейшей обработки на клиенте

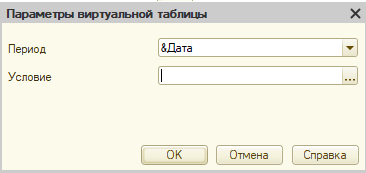
Кликнем ПКМ пустой строке -> “Конструктор запроса с обработкой результата” -> “Да”



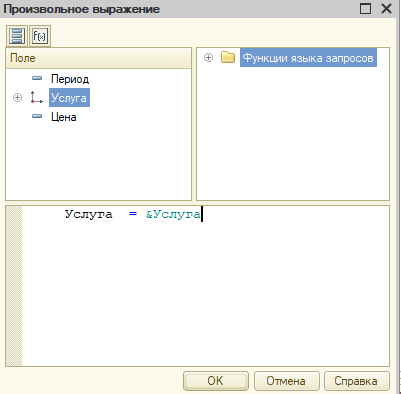
Выберем поля и таблицы согласно скриншоту ниже  


От услуги нам необходима только продолжительность, а из справочника нам необходима цена.  
Также Регистр сведений «ЦеныСрезПоследних» не был создан нами, он является виртуально таблицей, результатом вычислений в системе. Т.к. необходимы не просто все цены, необходимо настроить таблицу.

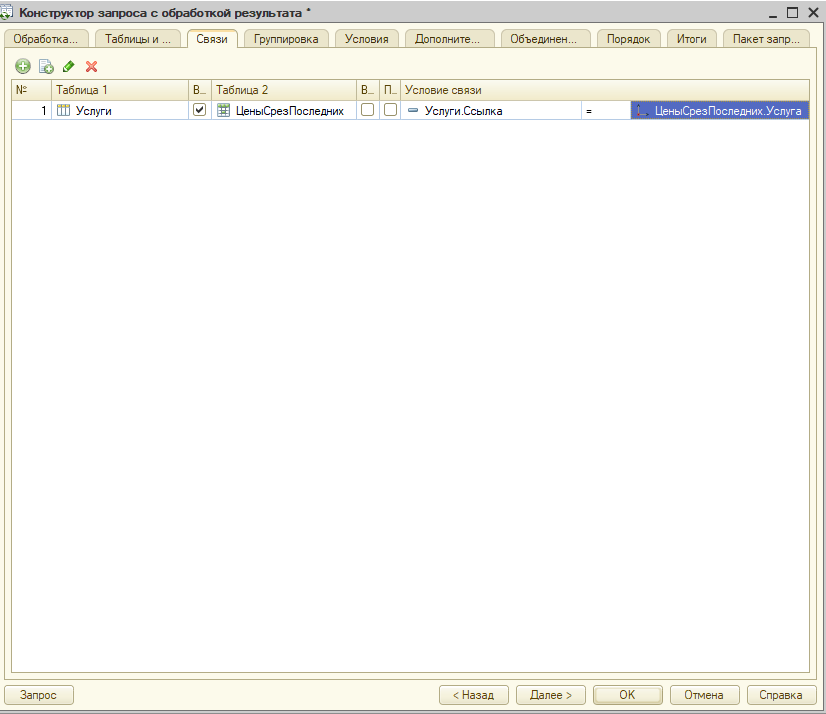


Укажем период, на момент которого мы будем получать цену. &Дата является параметром передаваемом в коде  


Нажмем на многоточие у условия. Перетащим поле услуга на полотно ниже и выполним поиск услуги по услуге передаваемой в качестве параметра.

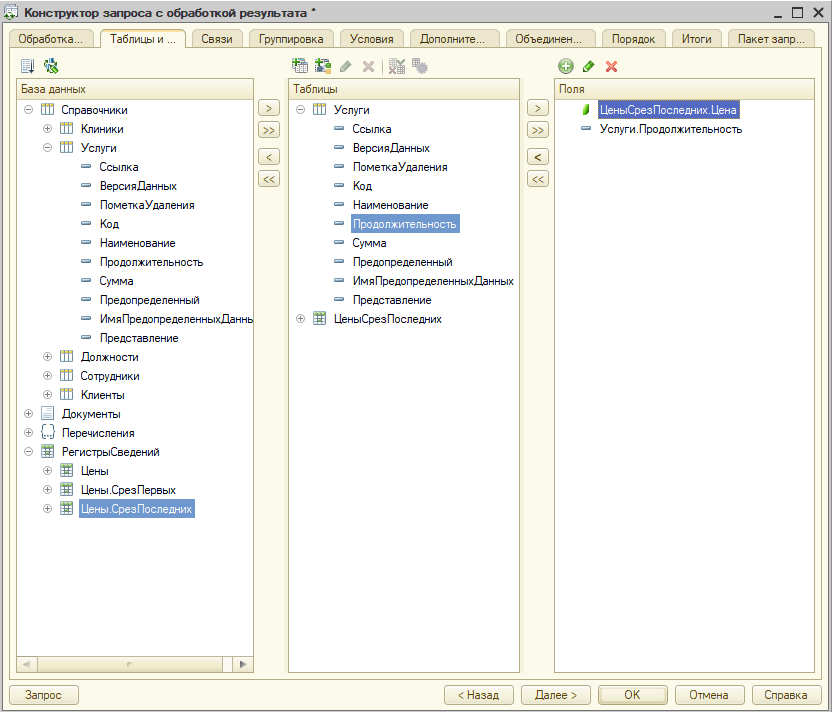


Перейдем на вкладку «Связи», удалим текущее условие

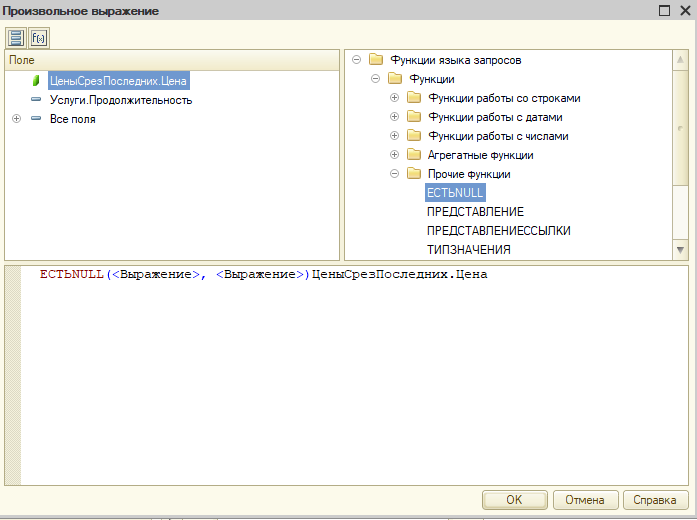
Создадим следующую связь  


Около таблички Услуга галочка означает то, что будут выведены все значения таблицы услуга в независимости от того найдутся ли согласно условию сопоставления в таблице Цены.

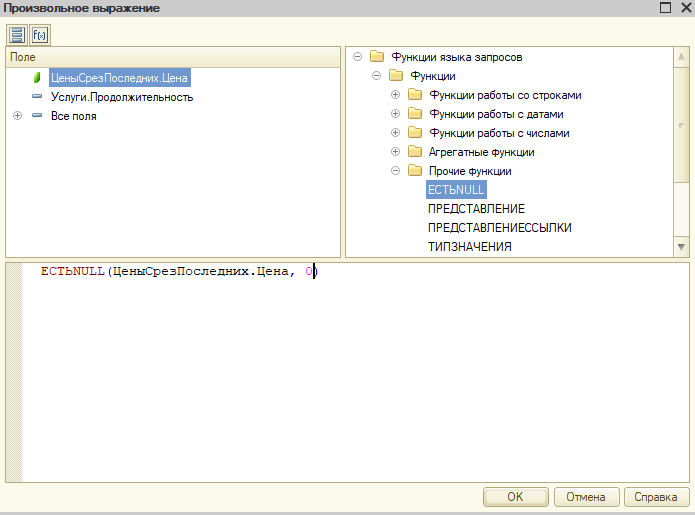
Кликнем дважды на поле «Цена», для открытия редактора формулы поля.



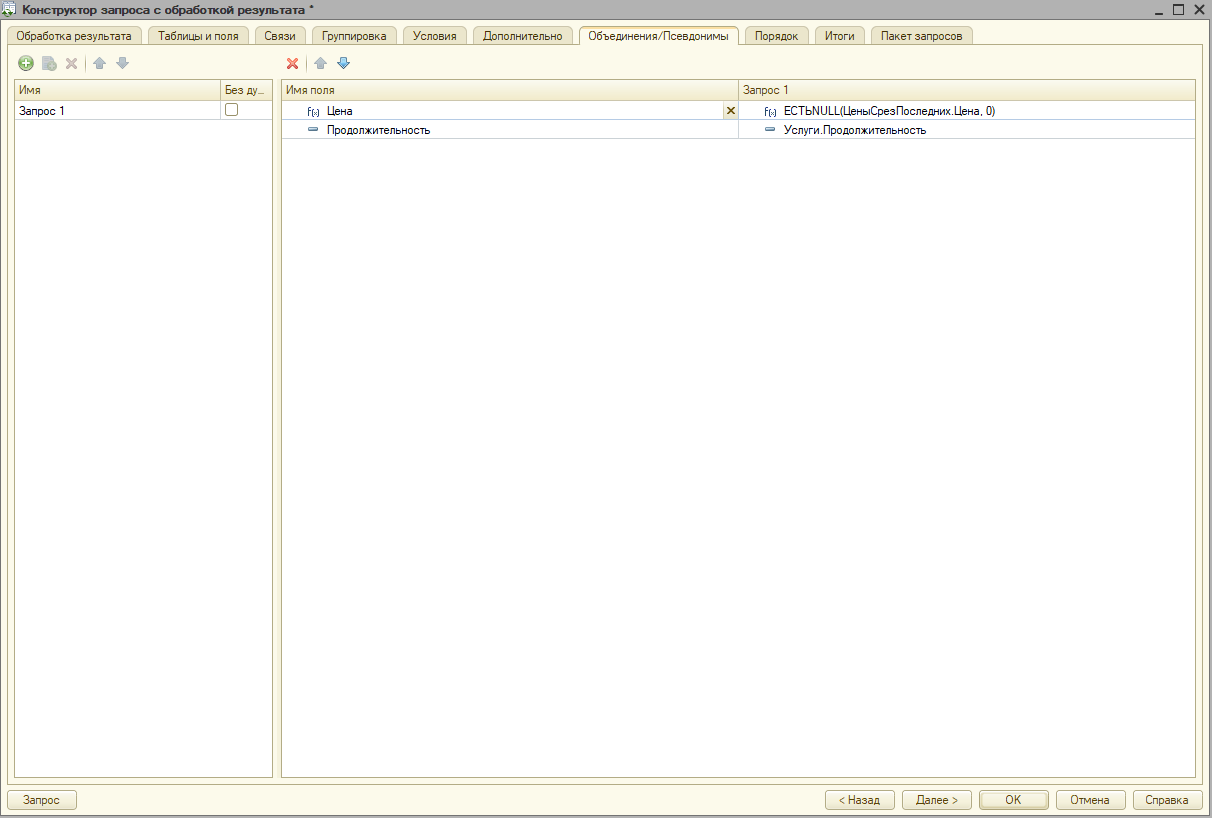
Найдем функцию ЕСТЬNULL

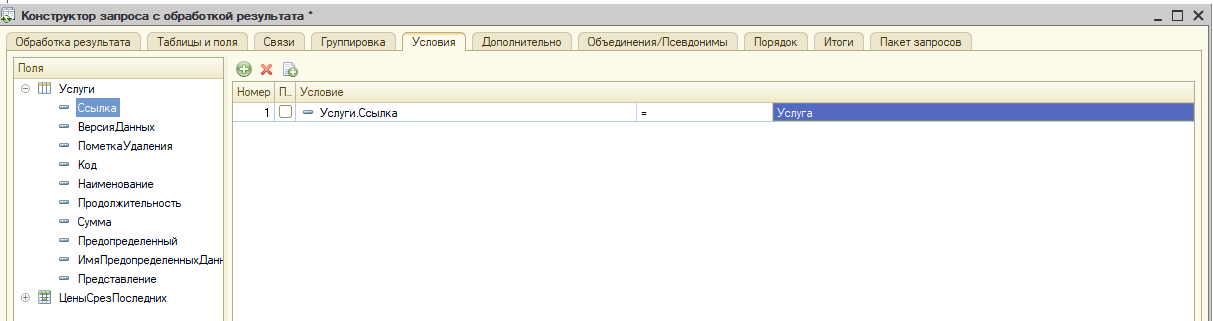


Подставим поле в функцию

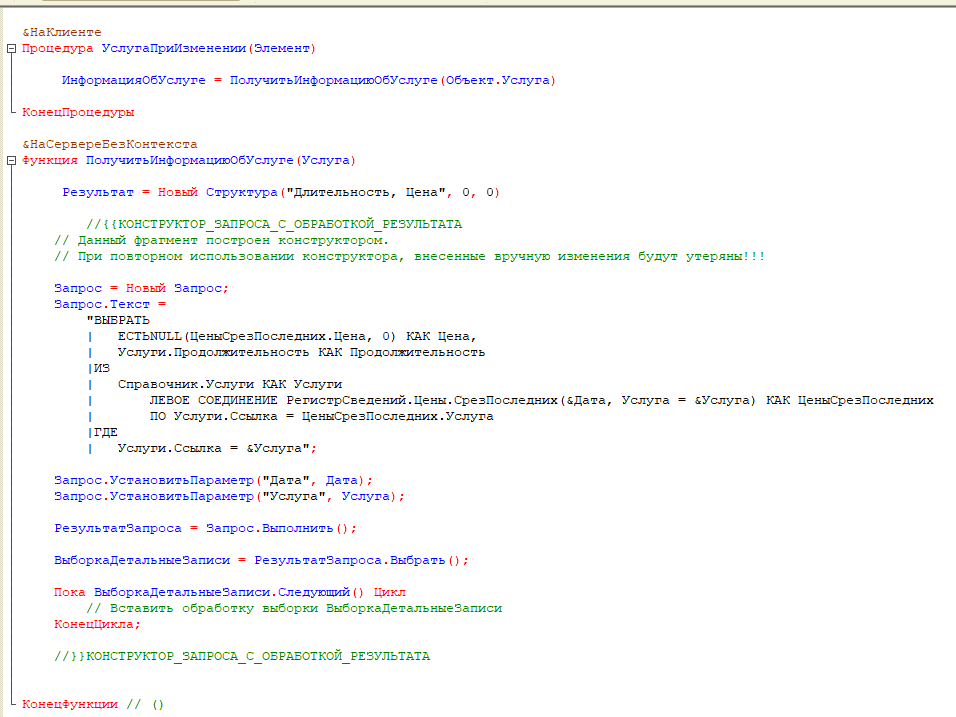


Теперь поле цена проходит обработку на пустоту. Если для услуги не будет указана цена, то вместо непонятного пользователю null значения мы получим 0.

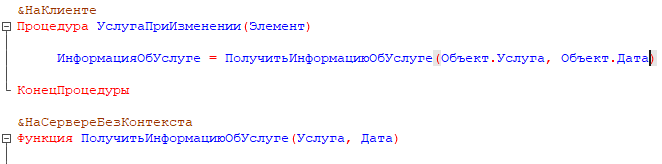
Пропишем названия для полей  


Также пропишем в условия поиска сопоставление услуги   


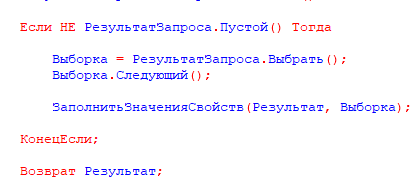
Нажимаем «ОК»

Получаем код запроса на местном диалекте SQL  


Одним из параметров поиска является Дата, но она никак не передается, добавим её передачу из формы в функцию

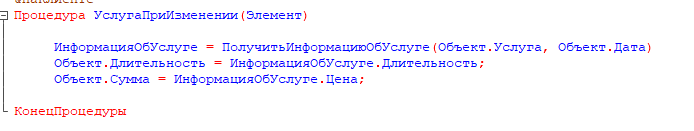


Т.к. наш запрос возвращает только одну услугу по названию услуги, нам не нужен цикл для обработки результата.

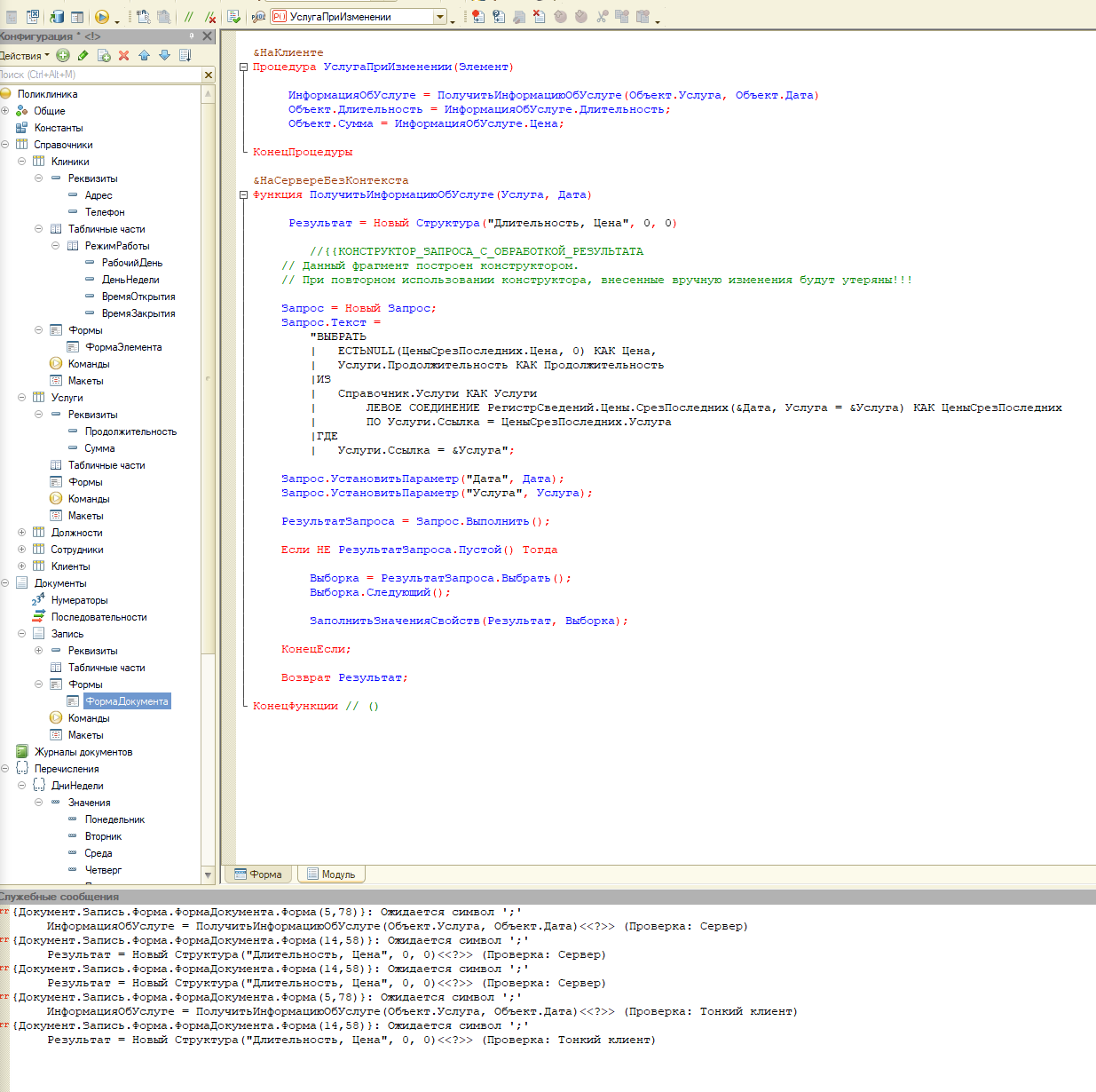


Добавим проверку на пустоту вывода запроса и заполним возвращаемую переменную

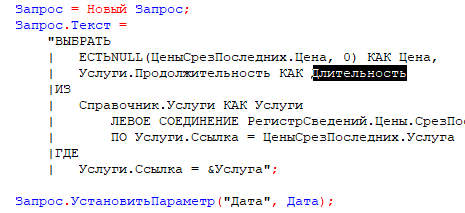
Добавим заполнение полей формы



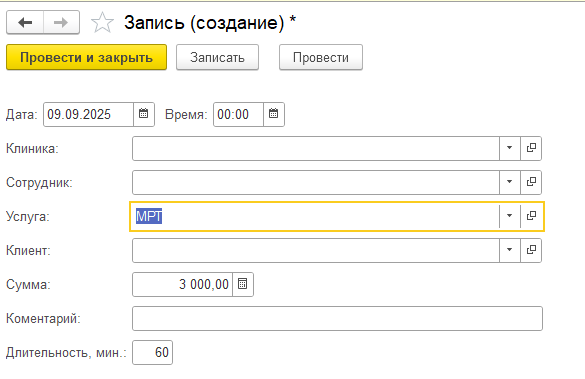
Выполним проверку на синтаксис и исправим ошибки

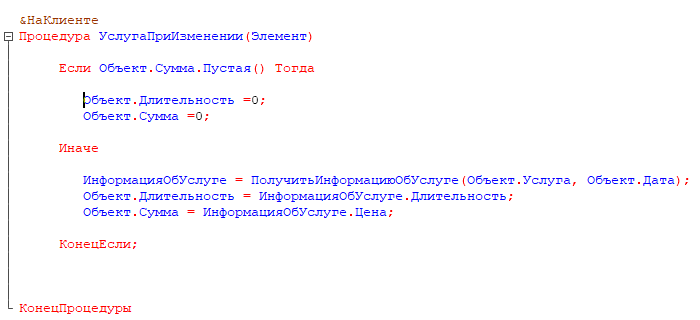


Запустим отладку. Обнаружим что у нас не заполняется продолжительность. Исправим сопоставление названий полей в запросе.



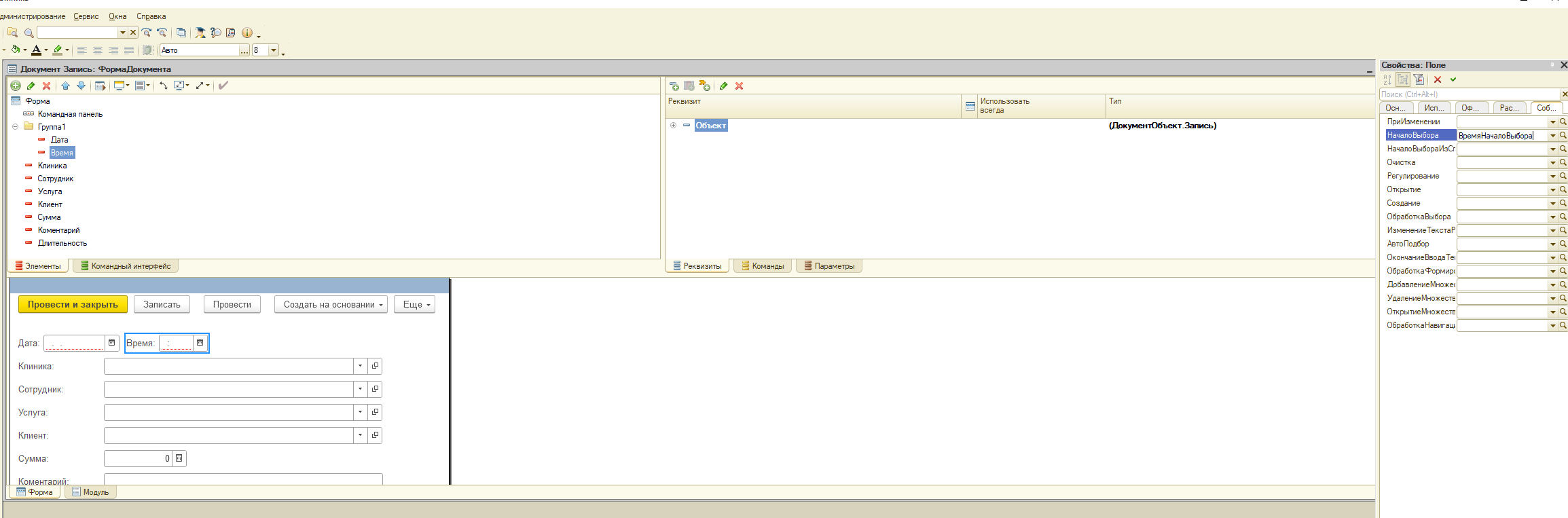
Повторно запустим отладку. Информация подтягивается успешно.

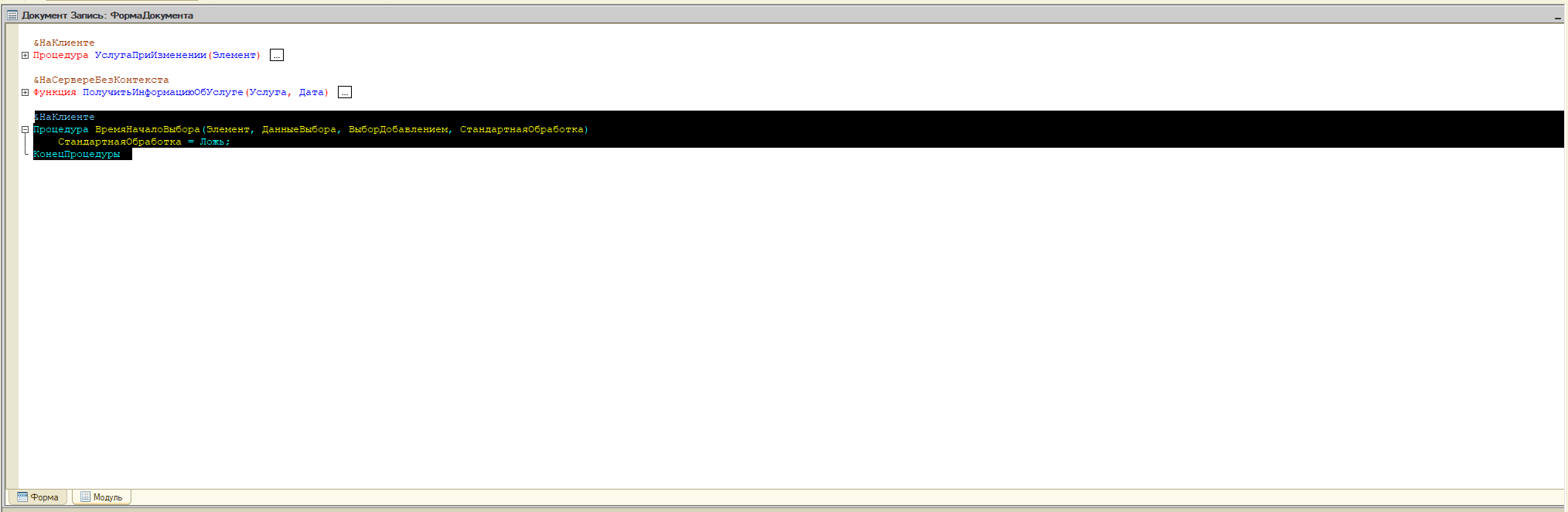


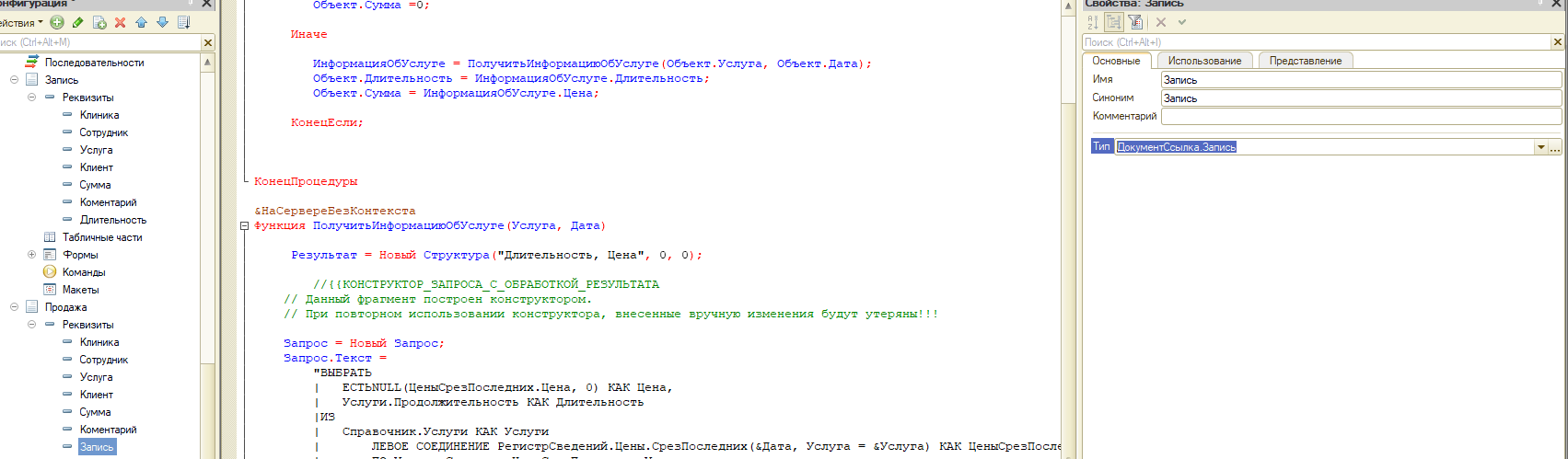
Добавим в код обработку обнуления значений на случай обнуления услуги.  


Отключим выбор времени через календарь

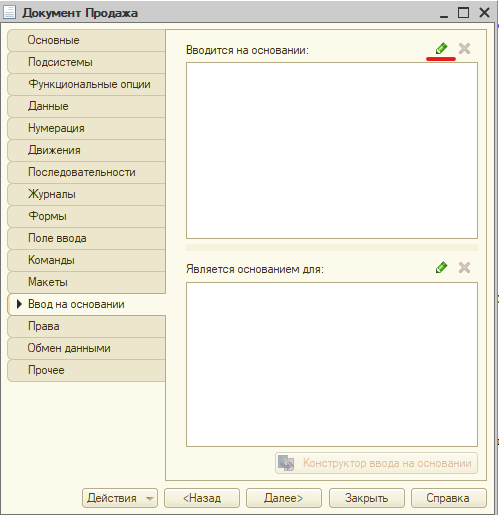






Создадим отдельный документ для фиксации предоставления услуги клиенту «Продажа». Скопируем все реквизиты из прошлого документа, кроме длительности. Добавим ссылку на запись, если услуга была оказана по предварительной записи.  


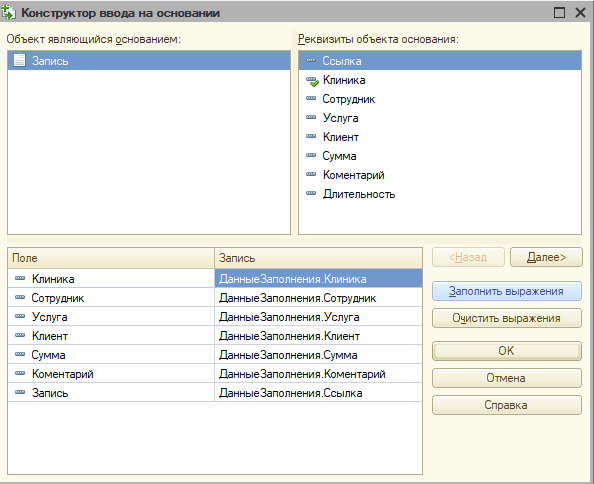
Для исключения рукописного введения информации о записи в продажу, реализуем функционал создания продажи на основании записи

Создадим ввод на основании для продаж.  


Откроем конструктор, опишем сопоставление полей.

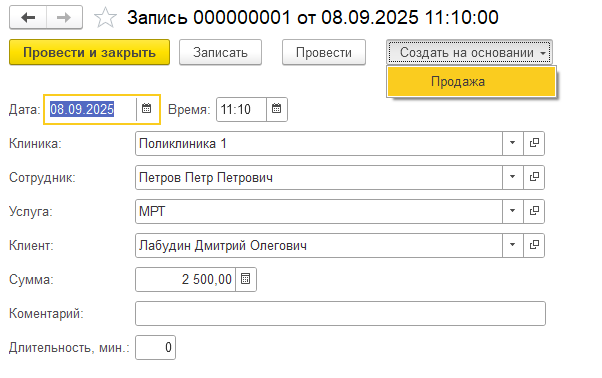


Т.к. поля одноименные, по нажатию на кнопку «заполнить» они заполняются автоматически.

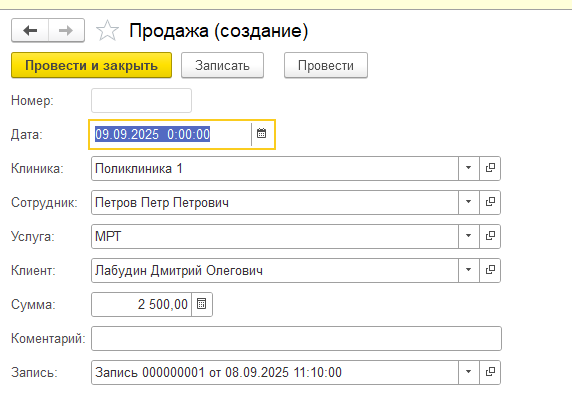


Нажимаем «ОК», запускаем демонстарцию.

В записи появилась кнопка «Создать на основании»



При нажатии у нас появляется уже заполненная продажа



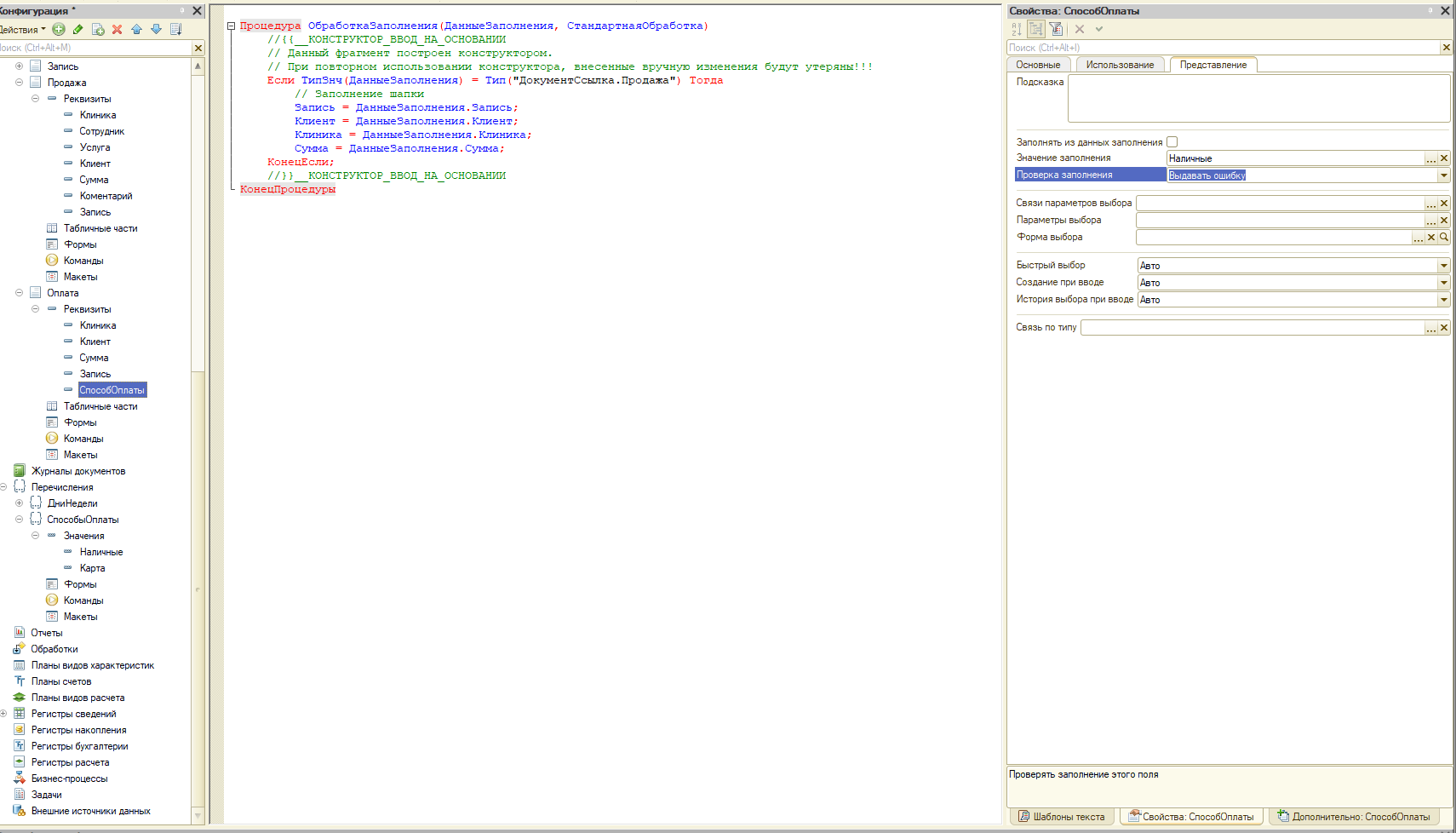
!!!В документе продажа необходимо запретить оперативно проведение и указать нумерацию периодом в течении года, как в записи

Создадим документ оплата с аналогичными настройками (нумерация, запрет оперативного проведения). Скопируем информацию о клинике, клиенте, сумме, записи.

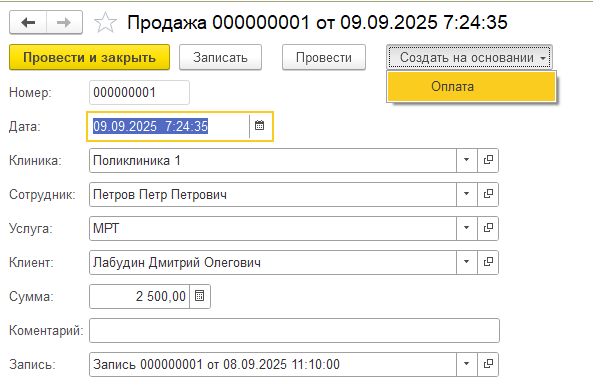
Создадим реквизит СпособОплаты. Для добавления выбора создадим перечисление с содержимым: наличные, карта.

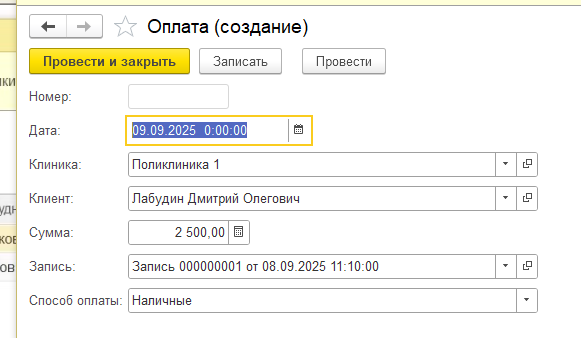
Создадим для документа оплаты ввод на основании продажи.

Укажем для способа оплаты значение «Наличные» по умолчанию



Запустим отладку.





Оплата успешно появилась.

Создадим документ для отражения зарплаты.

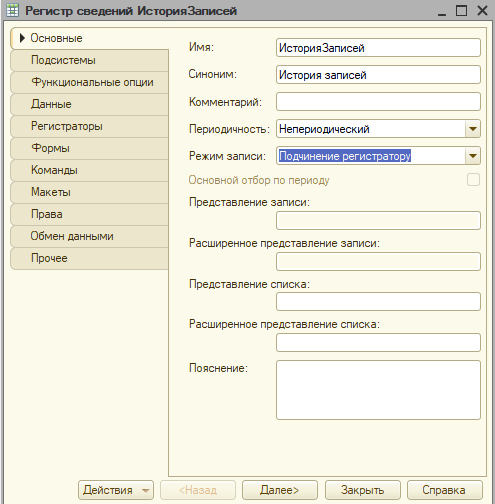
Скопируем клинику, добавим реквизит месяц. Добавим табличную часть с сотрудниками и суммой.

**Регистры**

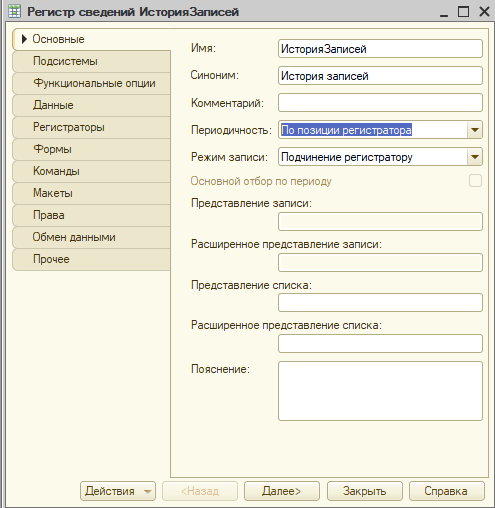
**Регистры –** таблицы, заполняемые автоматически при проведении документов для сбора аналитики. Один из регистров был использован ранее для записи изменений Цены.

Создадим регистр «ИсторияЗаписей».

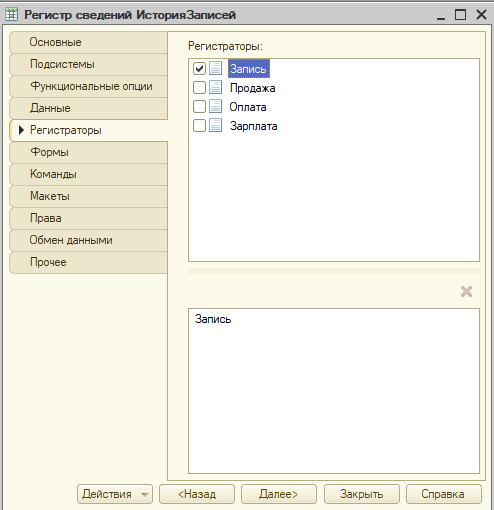
Указав данный режим записи, мы сделаем регистр, заполняемый данными документов при проведении.



Укажем регистру брать периодичность как у документа.



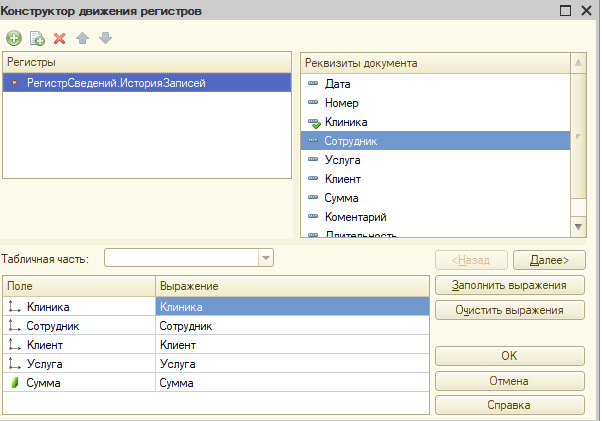
На вкладке регистраторы выбираем документ, из которого будут поступать данные.



Укажем измерения Клиника, Клиент, Сотрудник, Услуга и ресурс Сумма

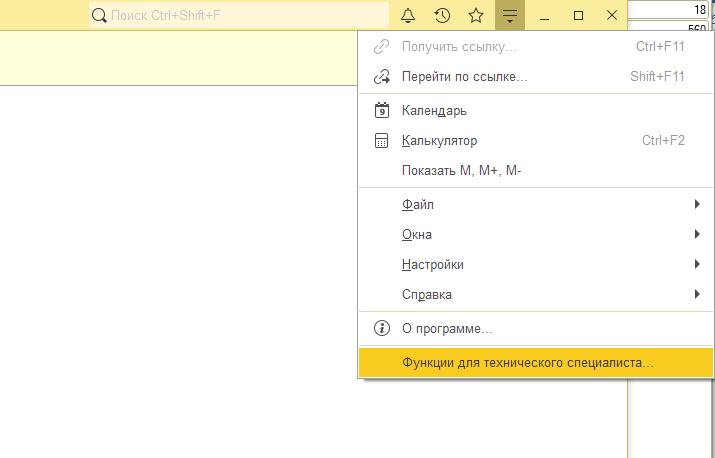
Каждое измерение сделаем ведущим и запретим оставлять незаполненным.

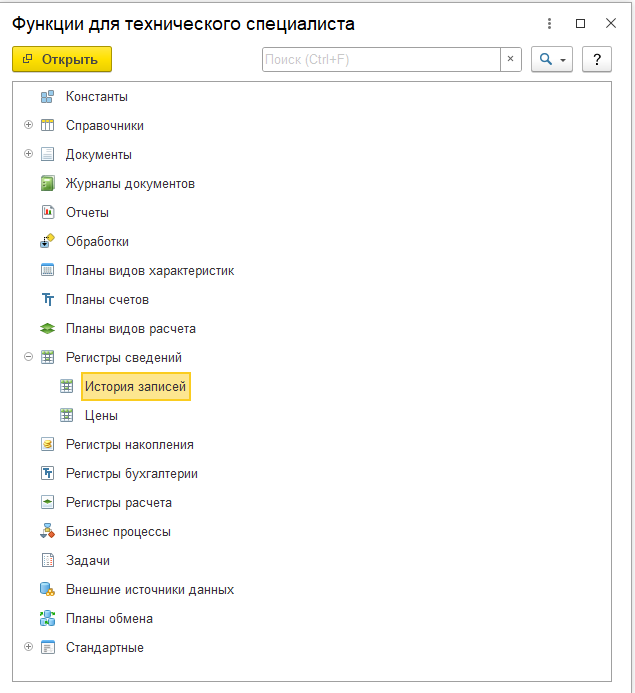
Перейдем в документ Запись, на вкладку движения, откроем конструктор движений для последующей настройки заполнения данных в регистр истории записей.

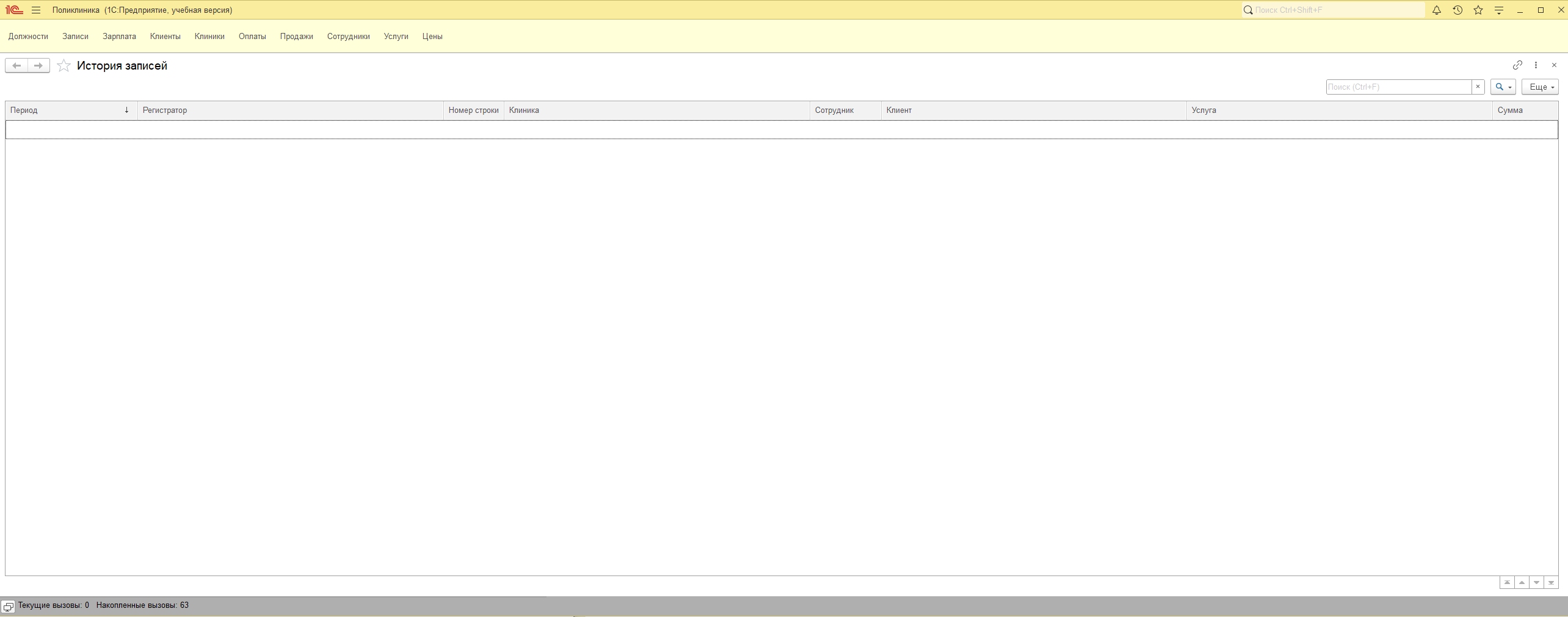
Нажмем «Заполнить выражения»  


Нажмем ОК, запустим отладку

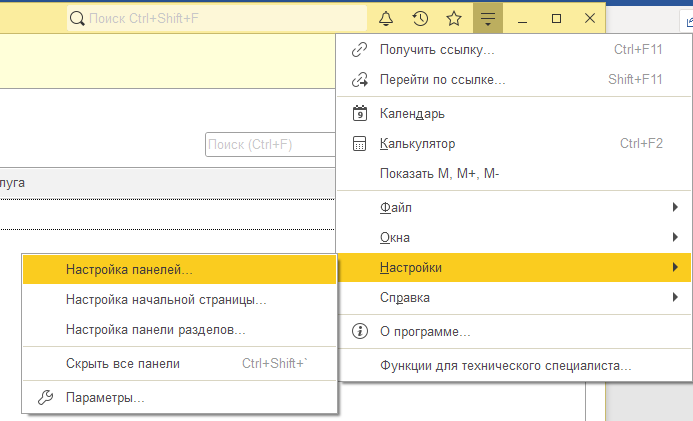
Поскольку регистр сведений подчинен регистратору, предполагается что он не будет просматриваться напрямую пользователем, а использоваться в аналитиках.

Раскроем функции для технического специалиста  


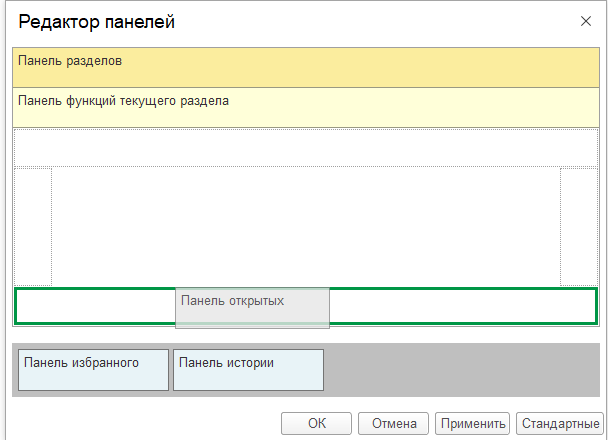


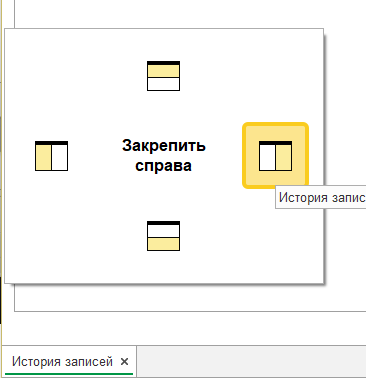


Перейдем в настройку панелей

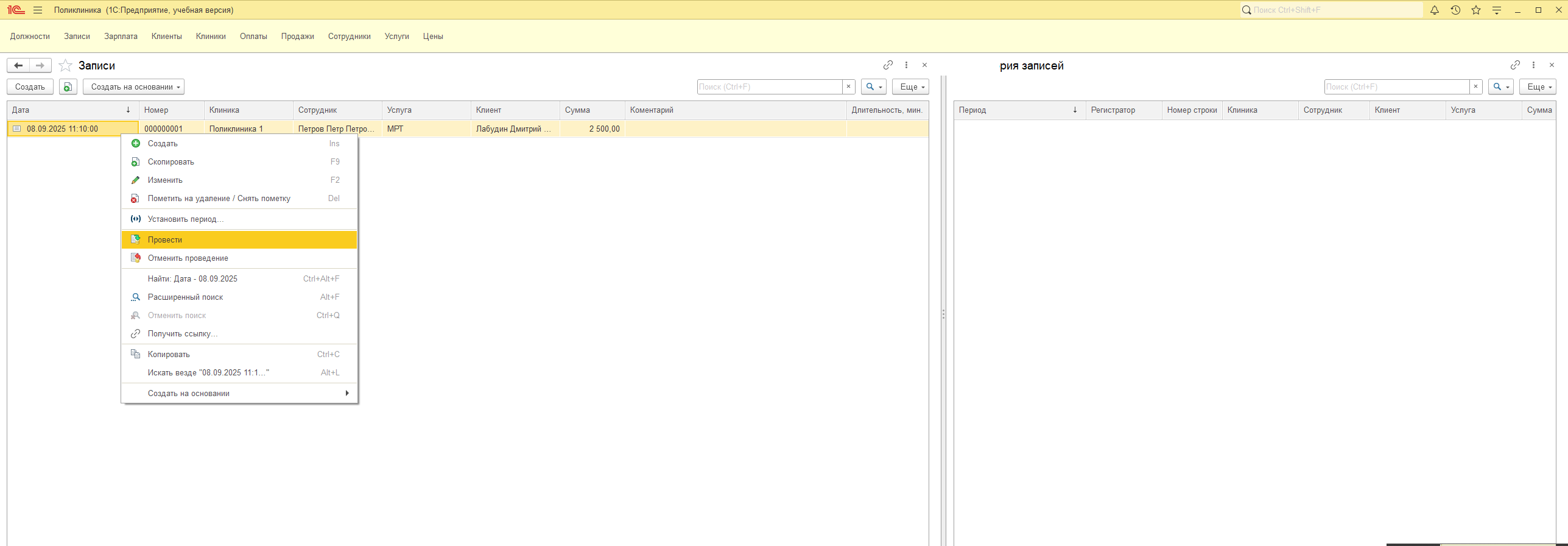


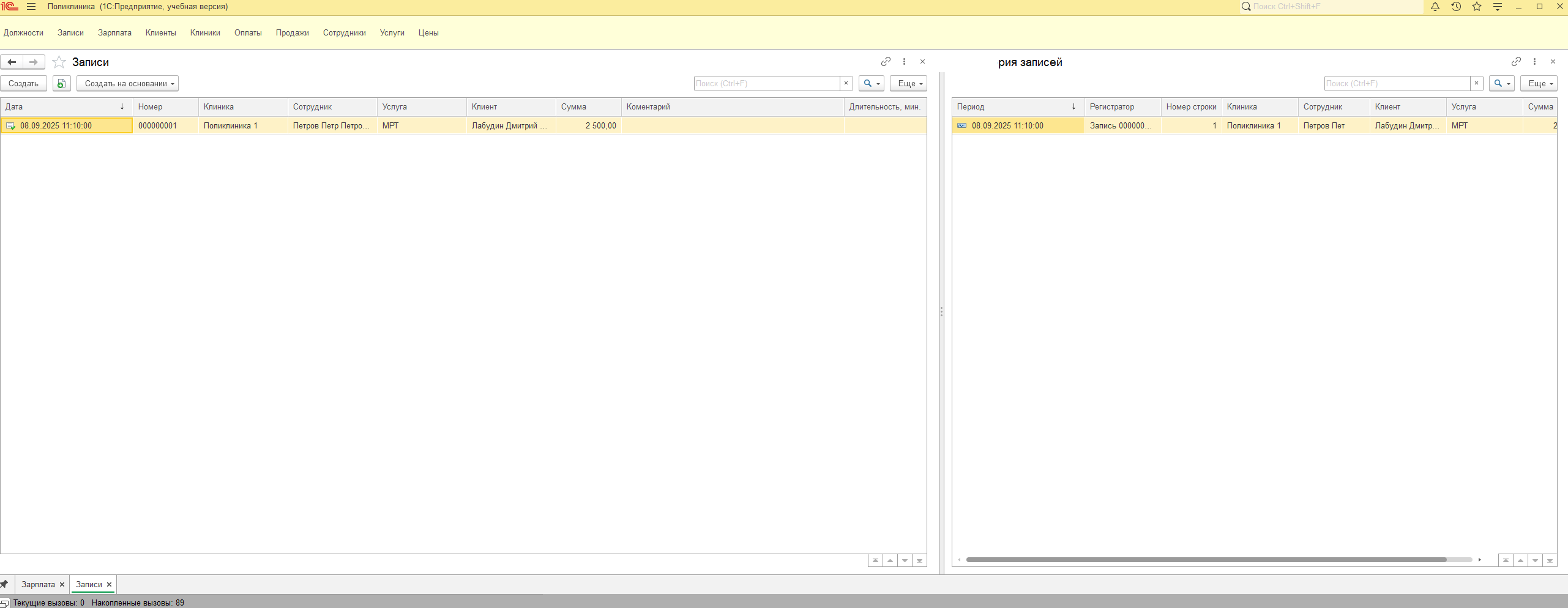
Перетянем панель открытых вниз.



Перетянем историю записей и прикрепим её справа  


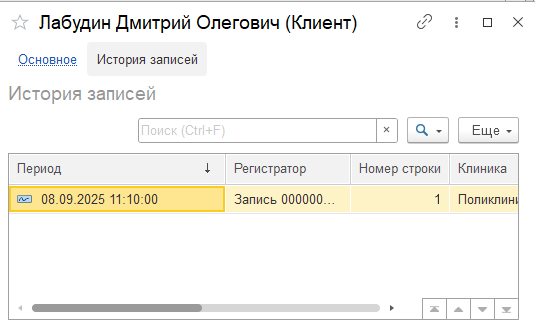
Выполним проведение записи





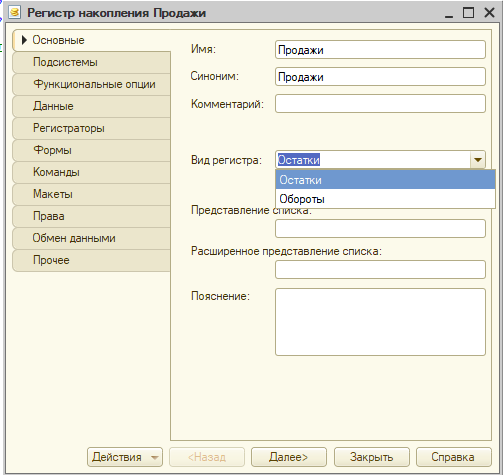
Благодаря регистру и проставлению ведущих измерений мы можем:

Например, перейти к клиенту и просмотреть его историю записей



Аналогично для каждого отдельного измерения

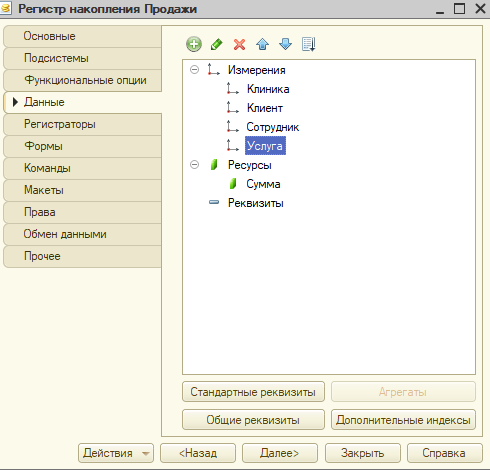
Создадим регистры накоплений для продаж, оплат, зарплат

Регистр накоплений бывает двух видов  


Остатки – ресурс может увеличиваться и уменьшаться

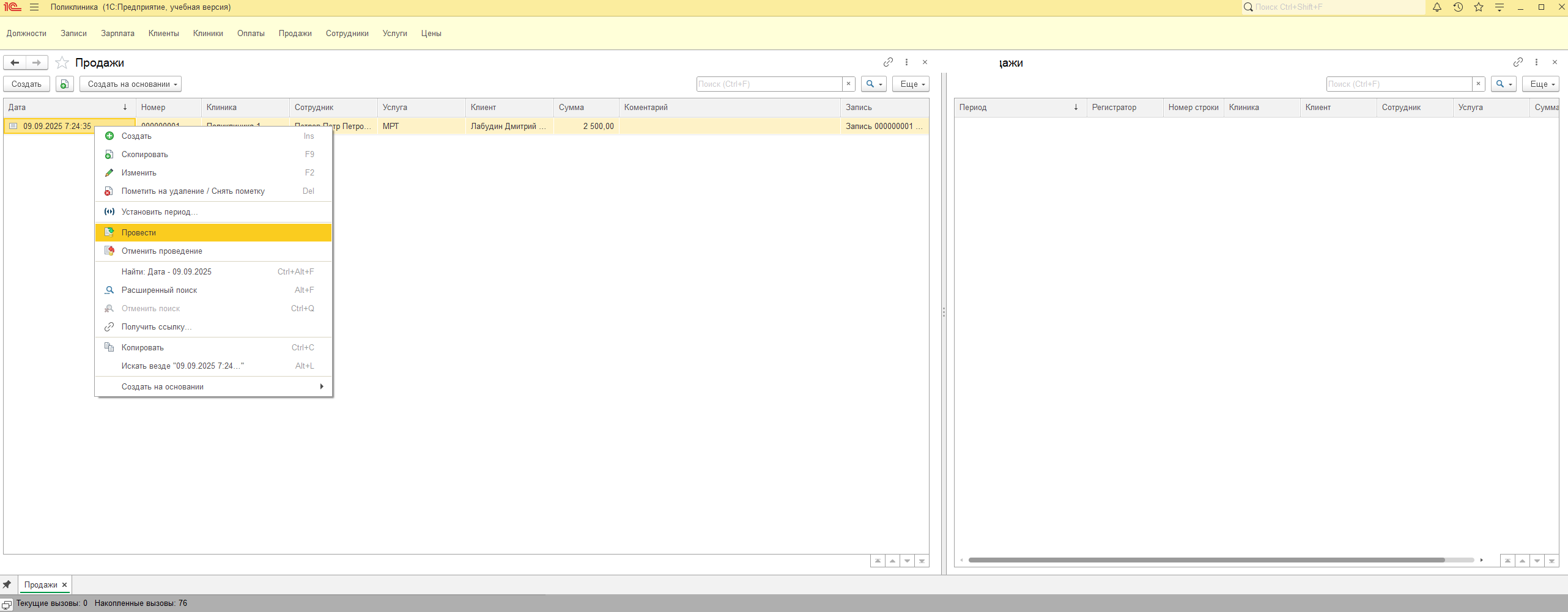
Обороты – ресурс может только поступать

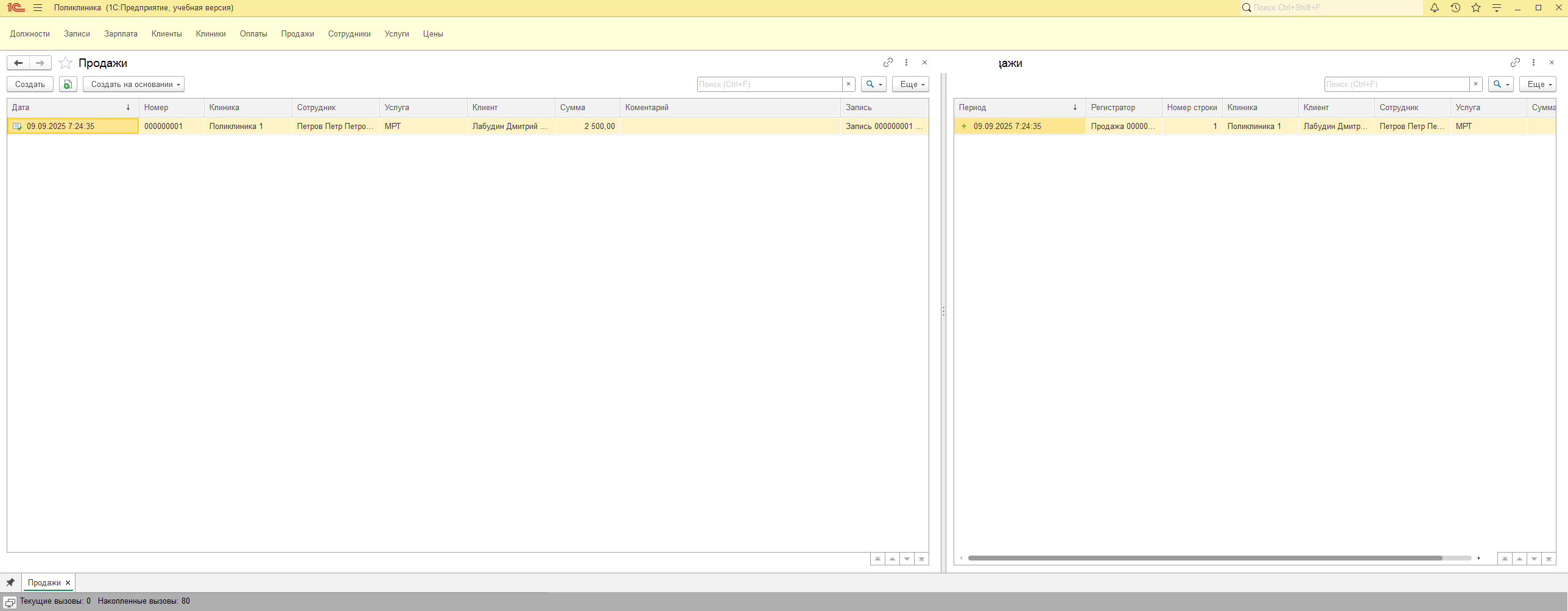
Для продаж будет использоваться регистр вида обороты

Заполним данные реквизита продаж  


Укажем документ продажи как регистратор

Перейдем в документ продаж -> движения -> конструктор движений

Запускаем отладку. Закрепим регистр продаж справа. Выполним проведение продажи.  




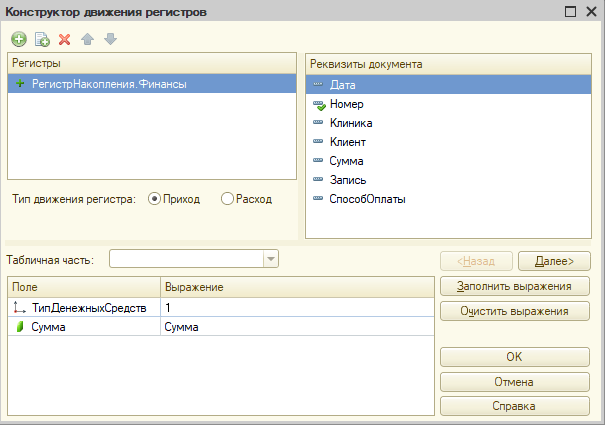
Создадим регистр «Финансы» с видом регистра Остатки.

Измерением «ТипДенежныхСредств» и ресурсом «Сумма». Регистратор – Оплата.

Создадим перечисление «ТипыДенежныхСредств» с значениями «Наличные» и «Безналичные».

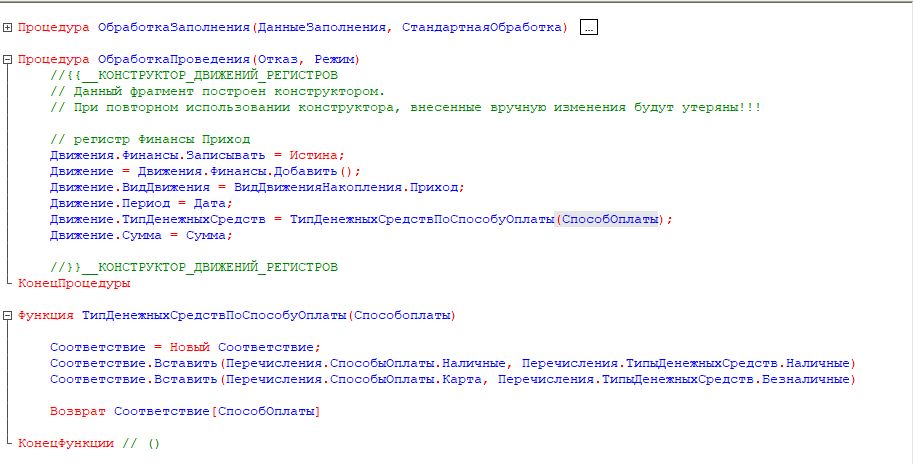
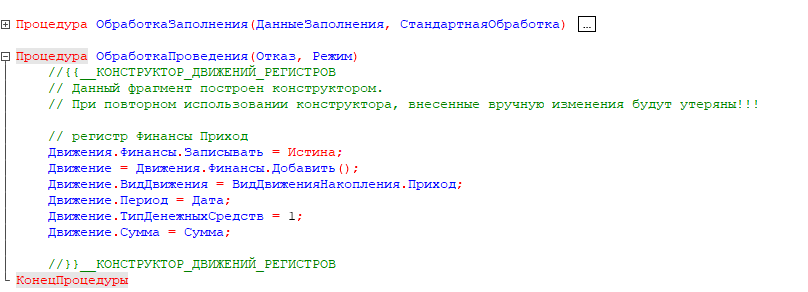
Перейдем в документ Оплата. Откроем конструктор движений.

Т.к. ТипДенежныхСредств отсутствует в документе проставим 1 для создания заглушки.



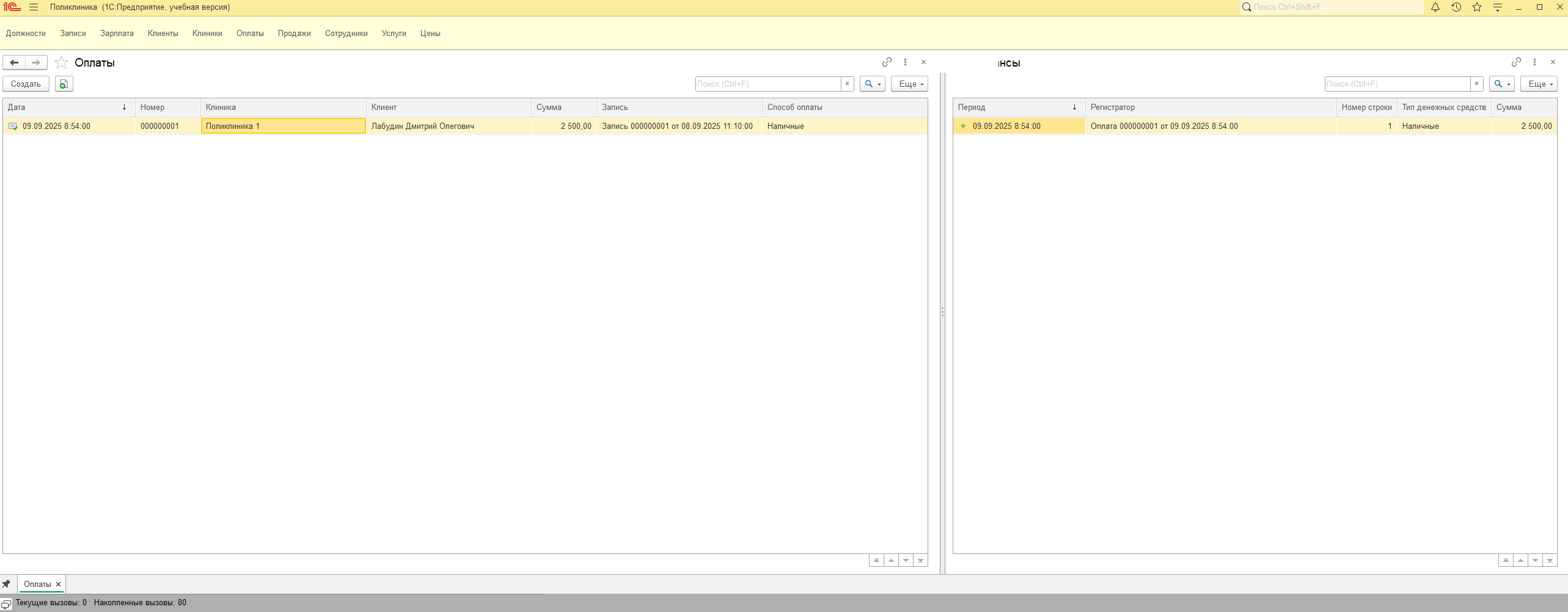
Обратим внимание на то, что т.к. данный регистр типа «Остатки» движение может быть как приходом, так и расходом. В данном случае это приход.

Реализуем сопоставление способа оплаты с типом денежных средств в коде движения

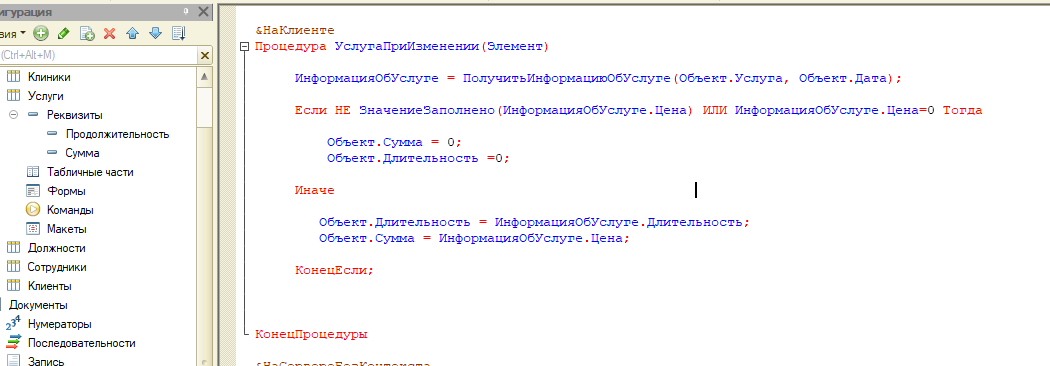


Где Соответствие специальная конструкция вида ключ-значение, т.е. возвращает значение в зависимости от переданного ключа. В функцию передается непосредственно способ оплаты(ключ)

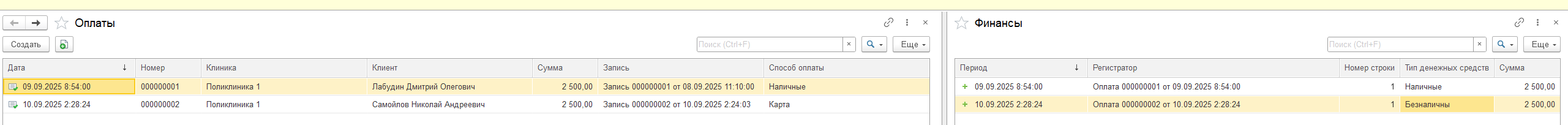
Запусти отладку, откроем регистр, выполним проведение.



Создадим и проведем запись -> продажу -> оплату по безналу -> проведение документов.

Перед созданием записи необходимо исправить логическую ошибку в форме документа. Корректный код.  


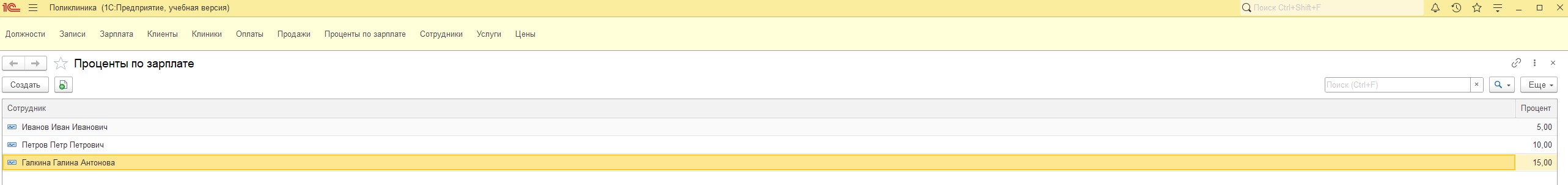
На выходе должно получиться две записи прихода в финансах



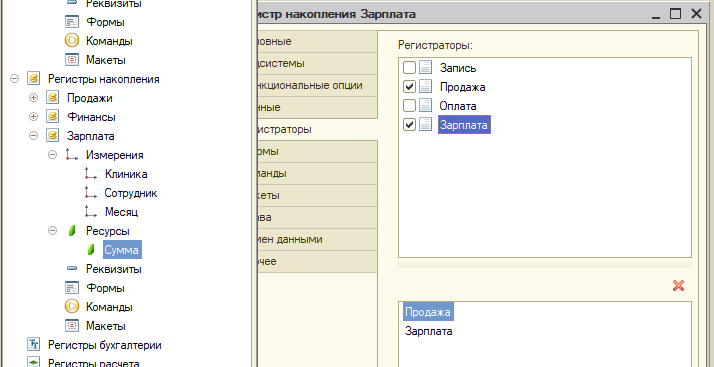
Увеличим длину строки типа оплаты до 15, т.к. надпись «Безналичные» не помещается

**Расчет зарплаты.**

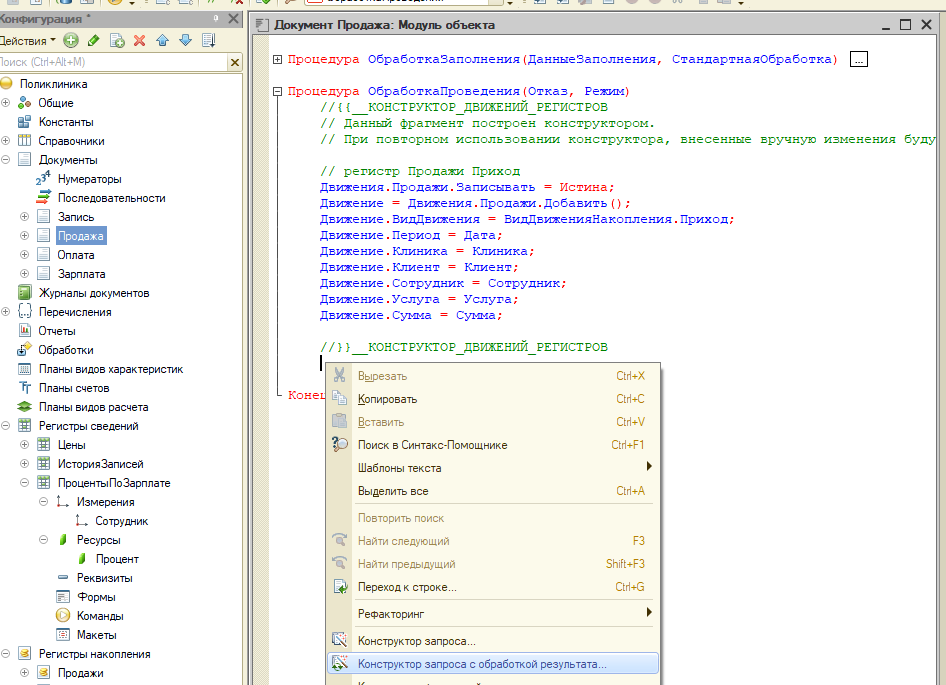
Предусмотрим расчёт зарплаты работникам. С целью демонстрации многообразия бизнес-процессов реализуем начисление работникам помимо основной зарплаты, фиксированный процент от продажи. Данный функционал может быть реализован разными способами. Для обеспечения максимальной возможности разграничения прав доступа за начисление процента будет отвечать Регистр сведений «ПроцентыПоЗарплате», регистр будет непериодическим, т.к. нам не интересна история, добавим измерение число (длина 5, точность 2), измерение сотрудник. Запусти отладку, назначим сотрудниками 5, 10, 15 процентов.

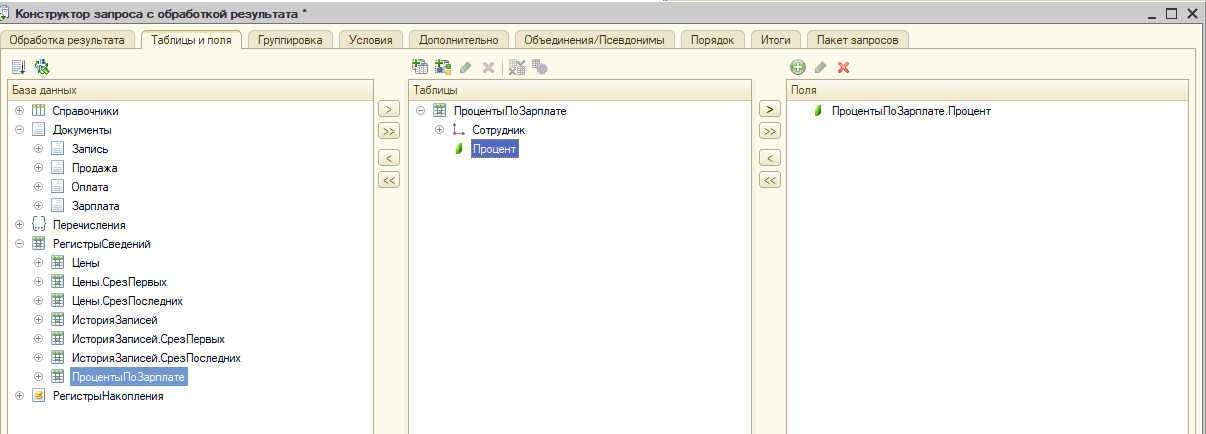


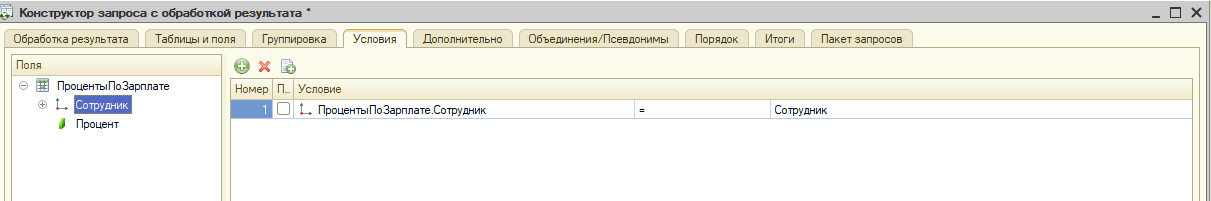
Добавим Регистр зарплата

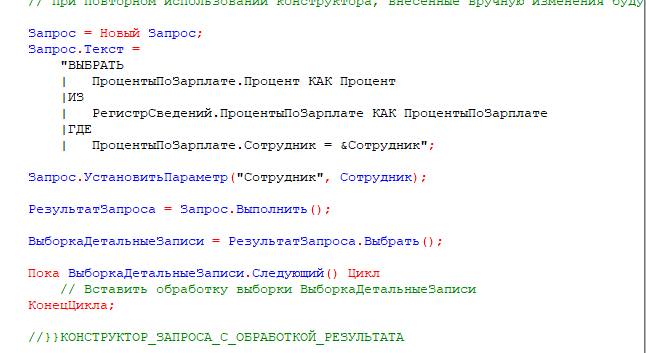


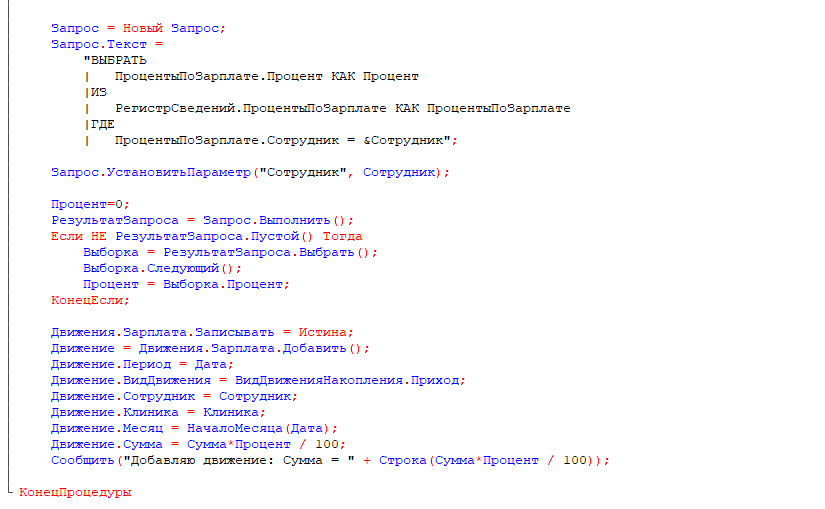
Где Продажа – приход, зарплата – расход



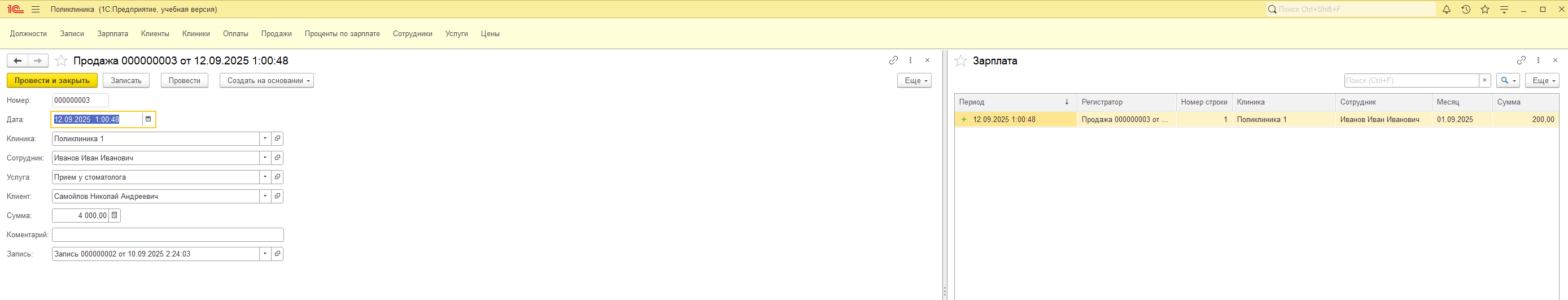








Строку «Сообщить» можно не использовать в работе. Указана как пример поиска проблем в ошибках кода.



Далее предлагается самостоятельно настроить движение «расход» из документа «Зарплата» в регистр накоплений «Зарплата», пока что без реализации учета накопления к оплате ежемесячного оклада.

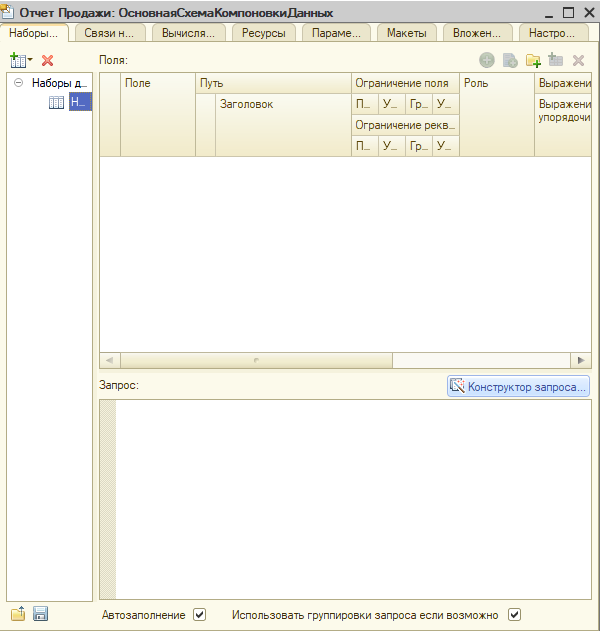
**Отчёты.**

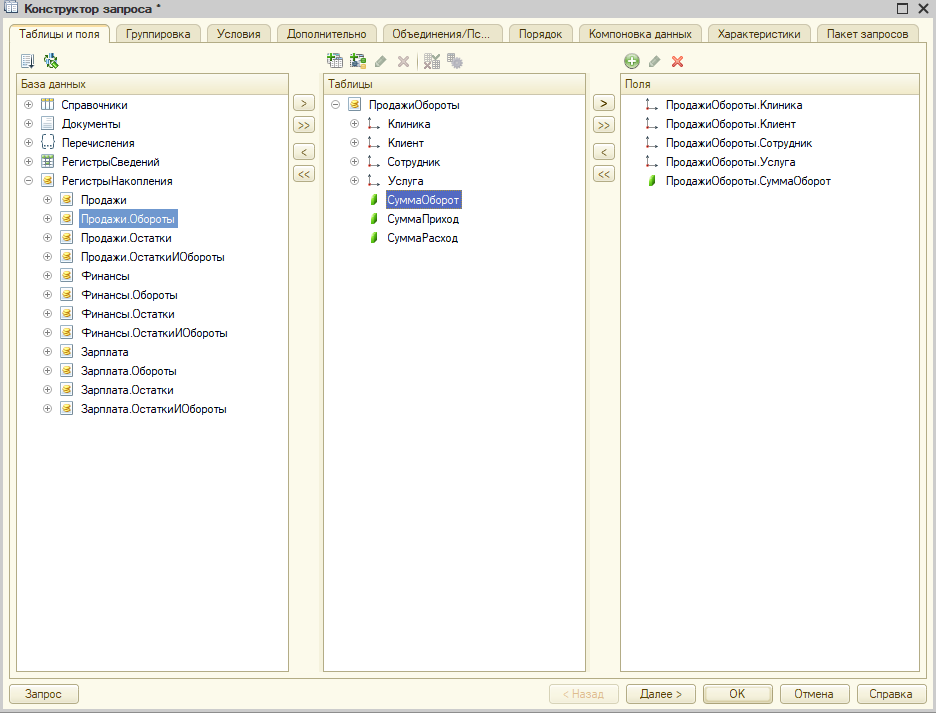
Рассмотрим на примере создание отчета по продажам.



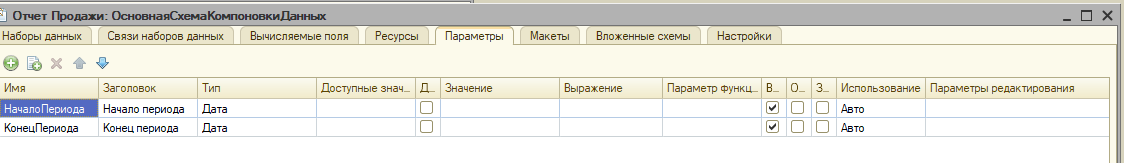


Набор данных – запрос

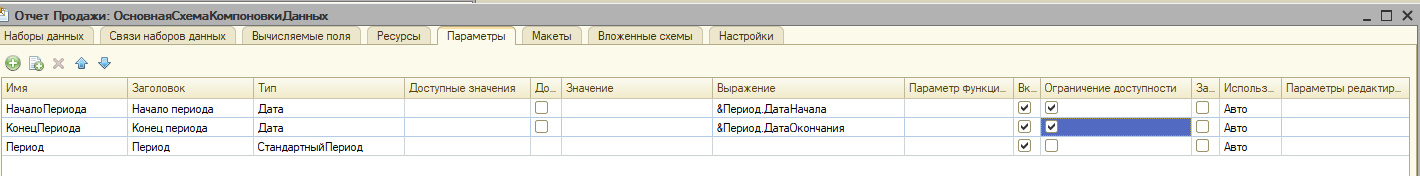


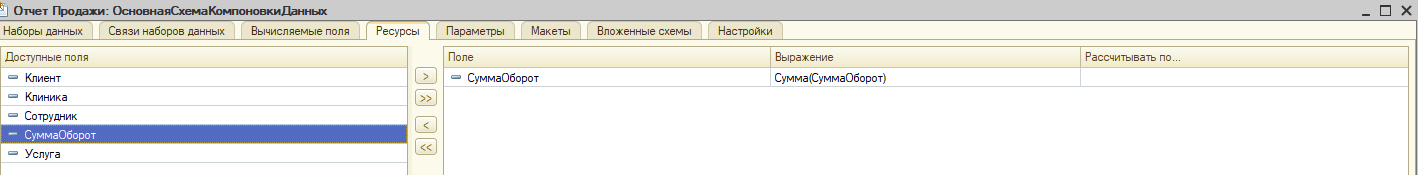


Параметры периода были добавлены автоматически

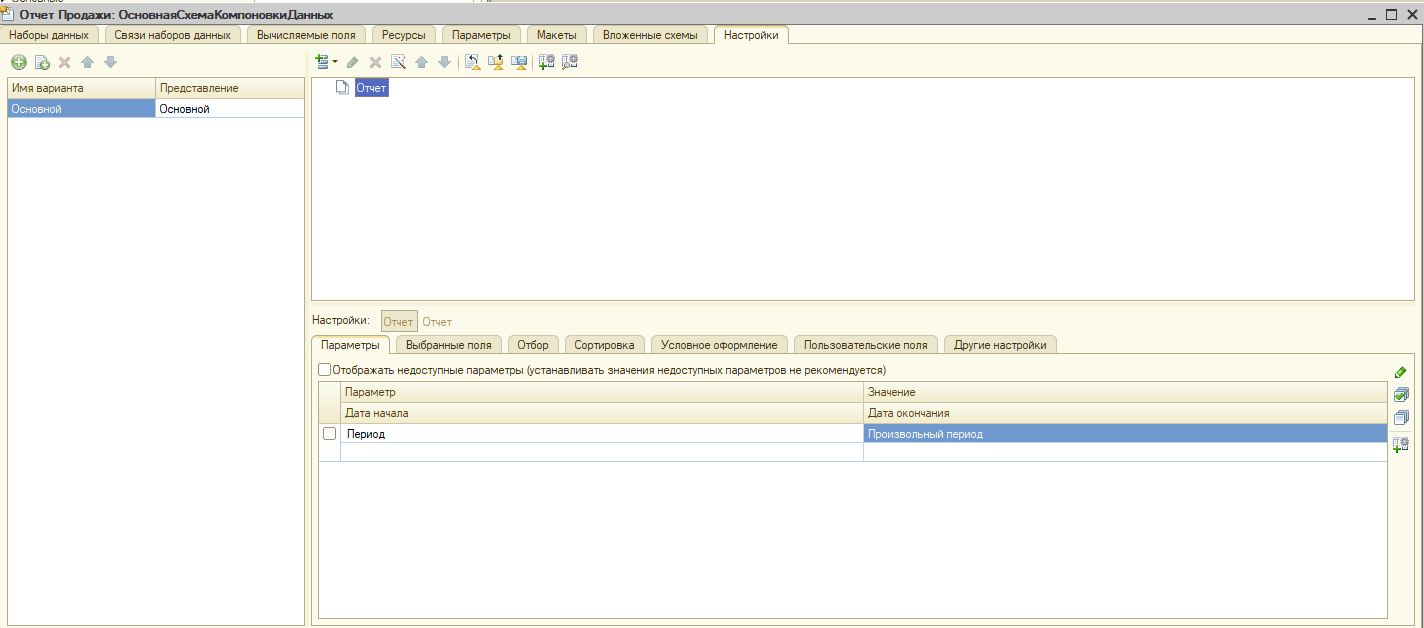


Реализуем указание диапазона через один параметр



Для каждого отчета настраивается ресурс – поле по которому считается итог, как правило числовое поле, но не всегда

Реализуем несколько вариантов отчета



Отчет может содержать как таблицы, так и диаграммы

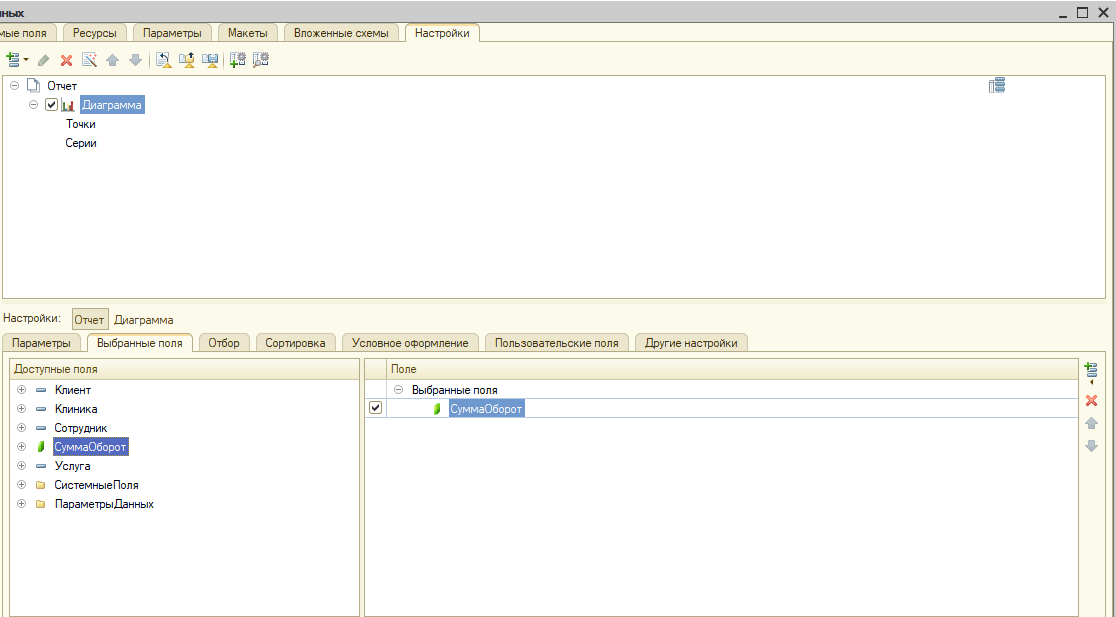
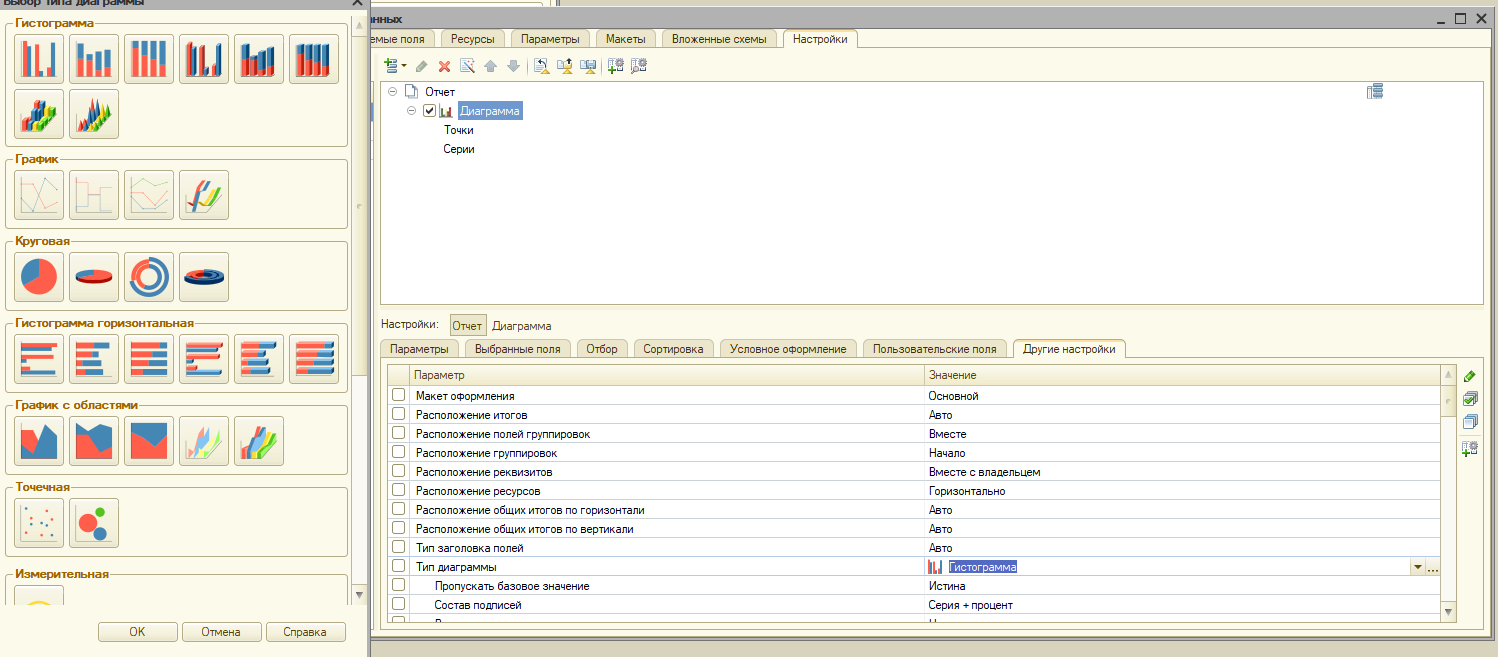
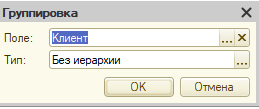
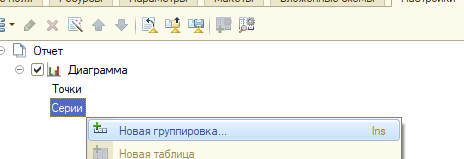


Диаграмма содержит точки, серии и значения.

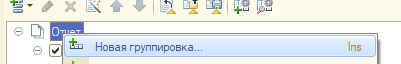
Значение наш ресурс (скрин. выше), точки и серии – аналитики в рамках которых будет отображаться наш ресурс.

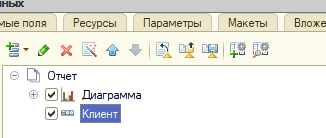


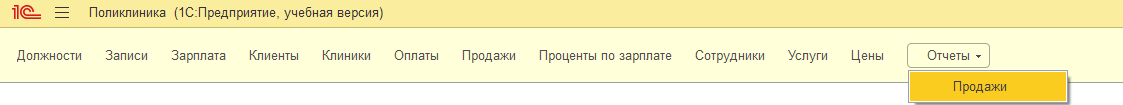
Выбор вида диаграммы. Например в круговой диаграмме серия является сегментом круга

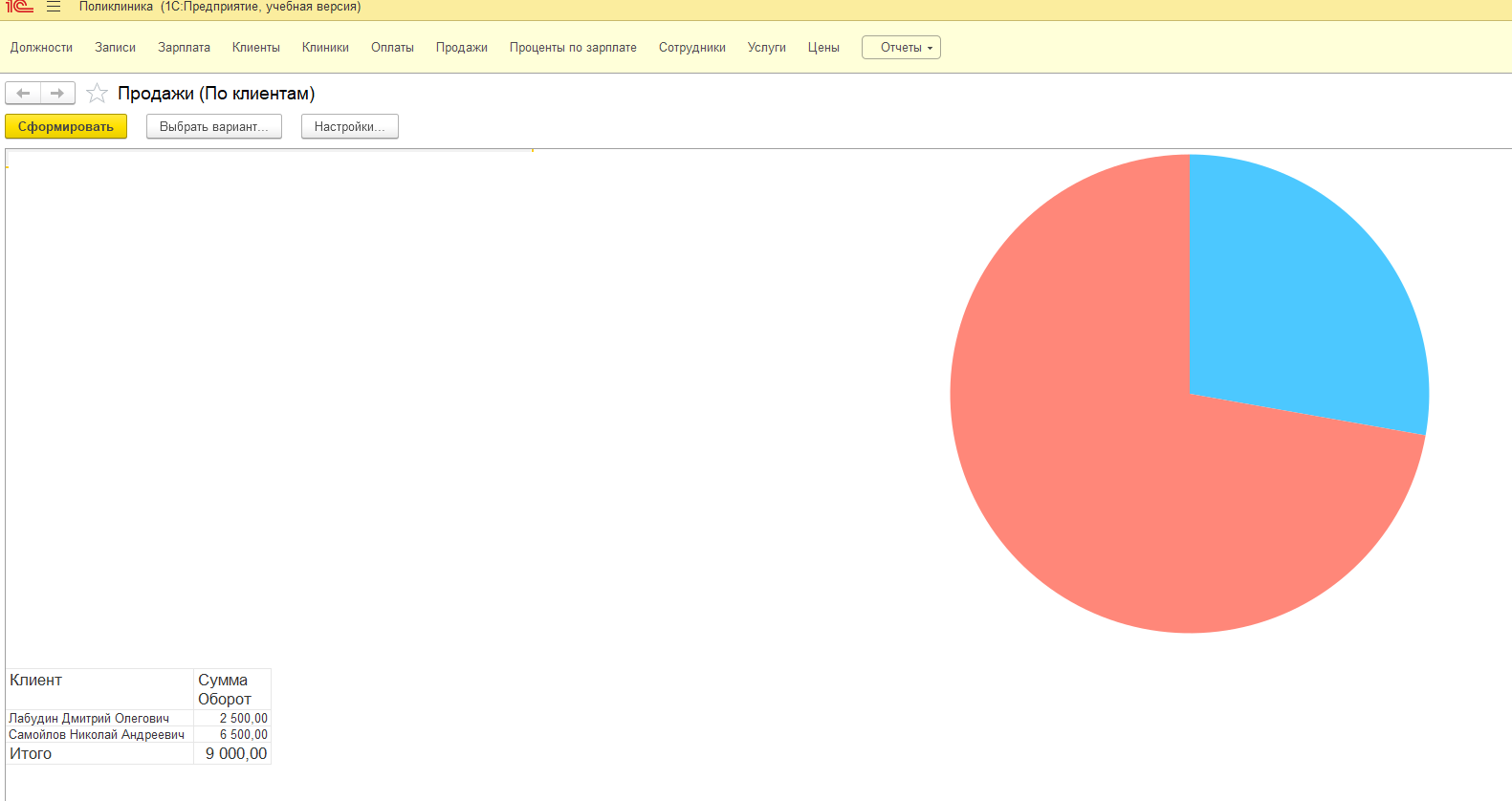


Для добавления таблицы создадим новую группировку по клиенту в корне

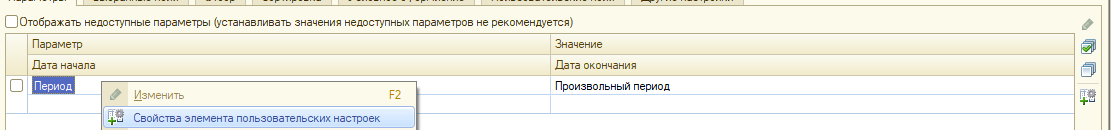


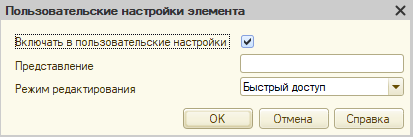




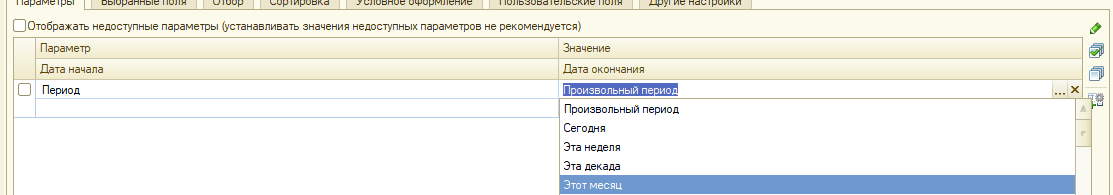


Вариацию диаграммы по сотрудникам и услуга предлагается настроить самостоятельно.

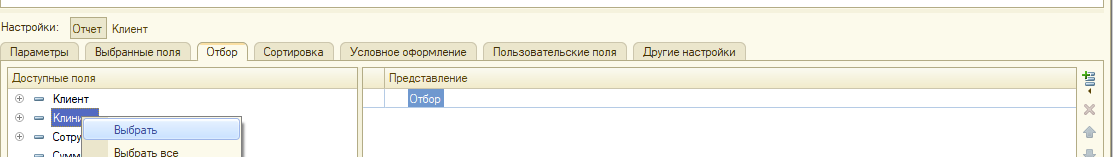


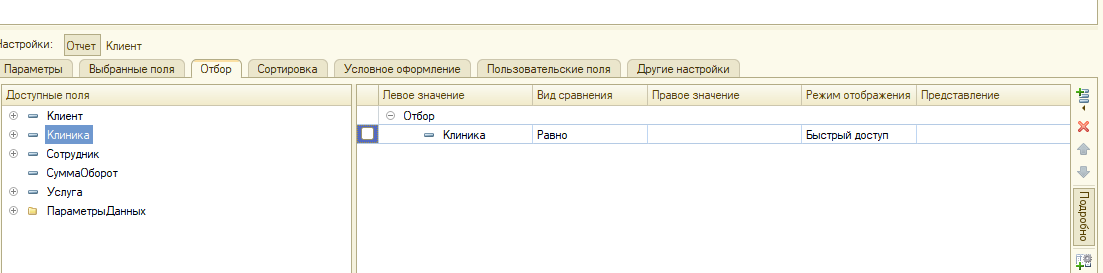


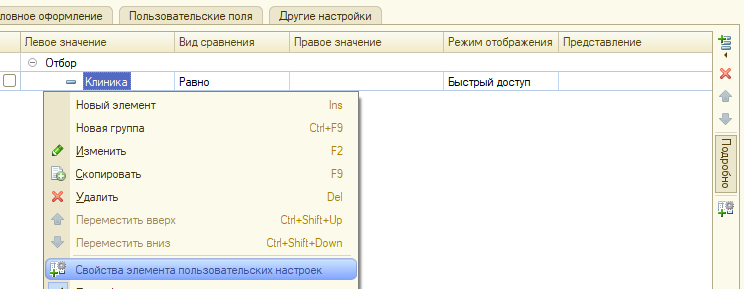
Т.к. наш период не является настройкой по-умолчанию, её необходимо включить, для каждого варианта



Указание периода по-умолчанию

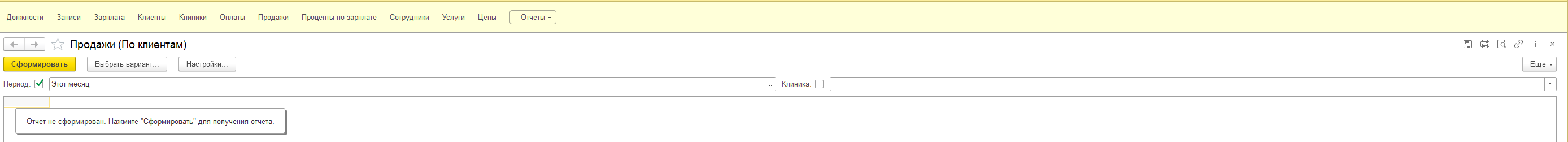








Добавление возможности фильтрации значений по одному из полей

Наши фильтры успешно отобразились  


Далее предлагается реализовать отчеты по оставшимся двум регистрам самостоятельно.

**Задание**

Самостоятельно разработать и реализовать по одной описанной в рамках курса сущности. Т.к. весь необходимый к реализации функционал создается аналогично описанию выше, просьба внимательно выполнять перепроверку своих действий и сверку с описанными методиками перед запросом о помощи.