МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Севастопольский государственный университет»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ
по дисциплине «Платформа 1С»
для студентов очной и заочной форм обучения
по направлениям
09.03.02 «Информационные системы и технологии»
09.03.03 «Прикладная информатика»

Севастополь
2021

УДК

Методические указания для выполнения лабораторных работ по дисциплине «Платформа 1С» для студентов направлений направлениям 09.03.02 «Информационные системы и технологии», 09.03.03 «Прикладная информатика» очной и заочной форм обучения / Сост.: (Вписать составителей) – Севастополь: СевГУ, 2021. – (Количество страниц) с.

Целью методических указаний является оказание помощи студентам в выполнении лабораторных работ по дисциплине. Приводятся постановка задачи для выполнения работы, варианты заданий, описаны этапы выполнения работы с приведенными примерами.

Методические указания рассмотрены и утверждены на заседании базовой кафедры (кафедру добавить) (протокол № __ от __ сентября 2021 г)

Рецензент:

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Технологическая платформа «1С:Предприятие» обеспечивает поддержку работы прикладных решений с различными операционными системами и СУБД, в том числе в среде открытого программного обеспечения и на мобильных устройствах. Открытость решений, возможность их динамичного развития, высокая функциональность и гибкость, возможность применения программ единой системы как в небольших предприятиях, так и в крупных организациях федерального масштаба обеспечивают высокую популярность решений на платформе «1С:Предприятия»: их использование широко распространено в России, Казахстане, Беларуси, они успешно применяются организациями многих других стран. Это создаёт большой рынок для программистов, владеющих языком и объектной моделью «1С:Предприятия».

Платформа «1С:Предприятие» является не универсальным, а специализированным, предметно-ориентированным средством разработки, предназначенным на решение задач автоматизации бизнеса. Одно из основных преимуществ этой предметно-ориентированной среды разработки – построение системы на основе технологической модели работы приложения, метаданных и прикладной модели работы приложения, что позволяет существенно упросить и ускорить разработку.

ЛАБОРАТОРНАЯ РАБОТА №3. ПРИКЛАДНЫЕ ОБЪЕКТЫ "РЕГИСТРЫ СВЕДЕНИЙ"

Цель работы: научиться создавать регистры сведений разных видов, получить навыки работы с данными регистров сведений с помощью объектной модели, научиться создавать с помощью конструктора движения документа. Научиться программно обрабатывать события изменения реквизитов объектов.

Используемое программное обеспечение: учебная версия платформы «1С:Предприятие».

1. Табличные данные. Основные положения

Задача любой системы – хранение и оперативное отображение информации пользователю. С помощью полученных данных, как правило, принимаются управленческие решения.

Предположим, что есть 1000 разных документов, и каждый из документов изменяет количество определенного товара на складе. Чтобы получить информацию о текущем количестве на складе, необходимо перебрать все документы: какие-то увеличивают количество товара, какие-то уменьшают, какие-то могут и увеличивать, и уменьшать. Такая система очень ресурсоёмкая.

Для упрощения данного процесса, разработчики создали особые объекты конфигурации – регистры.

Сведения из регистров используются для формирования отчетов. Классическая схема использования регистров в «1С: Предприятие» выглядит следующим образом:

Документы → Регистры → Отчеты

1) документами регистрируются события, приводящие к изменению значений показателей;
2) сами значения показателей хранятся в регистрах;
3) посредством отчетов пользователи получают информацию о состоянии показателей и проводят ее анализ.

В отличие от объектных данных, необъектные (табличные) данные не имеют собственной ценности и полностью описываются значениями своих полей. Необъектные данные представляют собой записи, которые хранятся в базе данных. Для записей не поддерживаются внутренние уникальные идентификаторы. Каждая из этих записей полностью описывается значениями своих полей. Для системы эти записи не обладают какой-либо значимостью, кроме того, что в их полях содержатся некоторые значения. Удалив некоторую запись и создав новую с точно такими же значениями полей, получим то же самое состояние базы данных, которое было до удаления записи.

Второе важное отличие заключается в том, что, изменив значения полей записи, мы получаем другую запись, в то время как изменение значений полей объекта не влечет за собой появление нового объекта.

Большинство необъектных сущностей конфигурации имеют общий порядок работы с данными. Из всей совокупности необъектных сущностей выделяются только константы: для каждой константы в базе данных хранится одно значение.

К необъектным данным относятся данные:

- регистров сведений;
- регистров накопления;
- регистров расчета;
- перерасчетов;
- регистров бухгалтерии;

- последовательностей;
- константы.

Регистр сведений 1С 8 — объект метаданных, предназначенный для хранения справочной информации в разрезе определенных разработчиком измерений.

Удобным средством для реализации задач хранения информации необъектных данных, развернутой в некоторых разрезах, является использование регистров сведений.

Объект конфигурации Регистр сведений предназначен для описания структуры хранения данных в разрезе нескольких измерений. На основе объекта конфигурации Регистр сведений платформа создает в базе данных таблицу, в которой может храниться произвольная информация, «привязанная» к набору измерений.

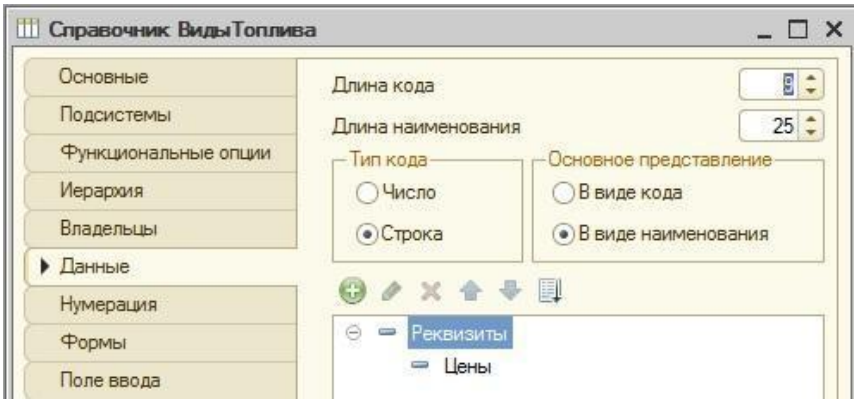
Принципиальное отличие регистра сведений от регистра накопления заключается в том, что каждое движение регистра сведений устанавливает новое значение ресурса, в то время как движение регистра накопления изменяет существующее значение ресурса. По этой причине регистр сведений может хранить любые данные.

Следующей важной особенностью регистра сведений является его способность хранить данные с привязкой ко времени. Благодаря этому регистр сведений может хранить не только актуальные значения данных, но и историю их изменения во времени - периодический регистр сведений.

Регистры сведений, информация которых изменяется во времени, называются Периодическими, а иначе эти регистры называют Непериодическими.

2. Непериодический и независимый регистры сведений

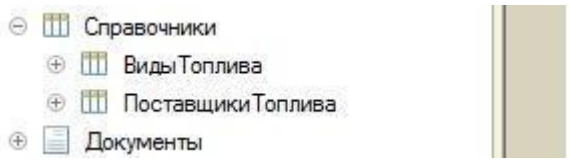
Основное предназначение регистров сведений в том, что в них должны храниться показатели аналитики. Например, у нас есть задача хранить виды топлива (АИ-92, АИ-95 и т.д.), но также и цену на этот вид топлива. Как это удобное всего организовать. Однозначно сами виды топлива необходимо хранить в каком-то справочнике. Так его и назовем – вид топлива. Но где же хранить цену на этот вид топлива? Самое первое решение в реквизите справочника.



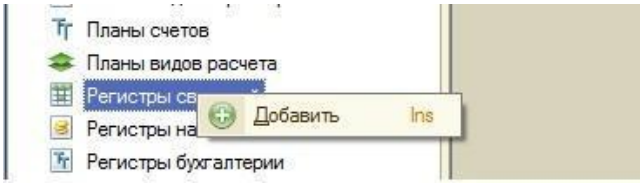
Очевидно, такое решение имеет место, если цена у нас никогда не изменяется. Но в жизни такое редко случается, поэтому если мы так сделаем, то возникнет необходимость каждый раз изменять элемент справочника при изменении цены. В принципе, почему бы и нет. Но, если мы еще добавим новый разрез цены – поставщик топлива: у одного и того же вида топлива может быть разная цена для разных поставщиков, то хранение цены в реквизите справочника станет принципиально не возможной: мы не будем знать, к какому поставщику относиться эта цена.

Для решения этих задач служит специальный объект конфигурации — регистр сведений. В этом регистре сведений можно создавать записи, в которых будет указано, что для такого-то вида топлива, для такого-то поставщика устанавливается такая-то цена.

Сейчас мы и решим эту маленькую прикладную задачу: в нашей конфигурации есть два справочника «Виды топлива» и «Поставщики топлива»

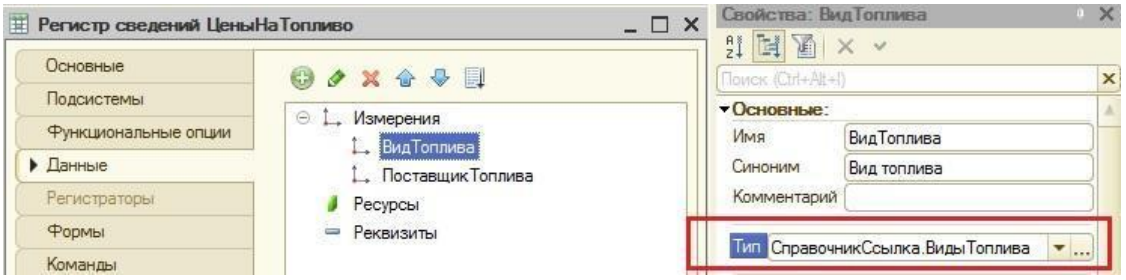


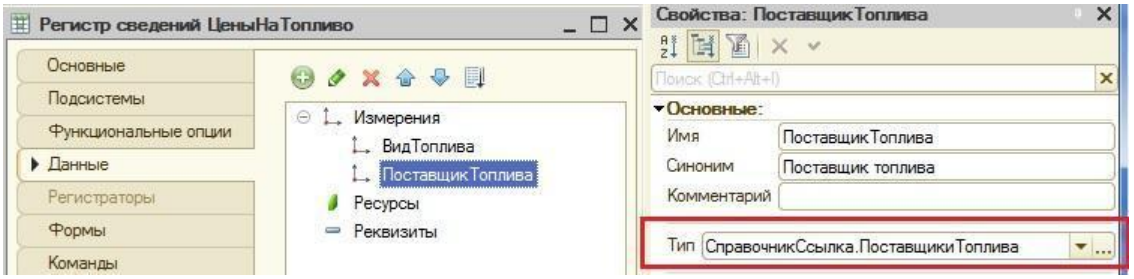
Необходимо организовать возможность хранения цены для каждого вида топлива с учетом поставщиков. Для этого в конфигураторе 1С создадим новый регистр сведений «ЦеныНаТопливо».



На закладке «*Основные*» установим имя и синоним. Все остальное оставим как есть.

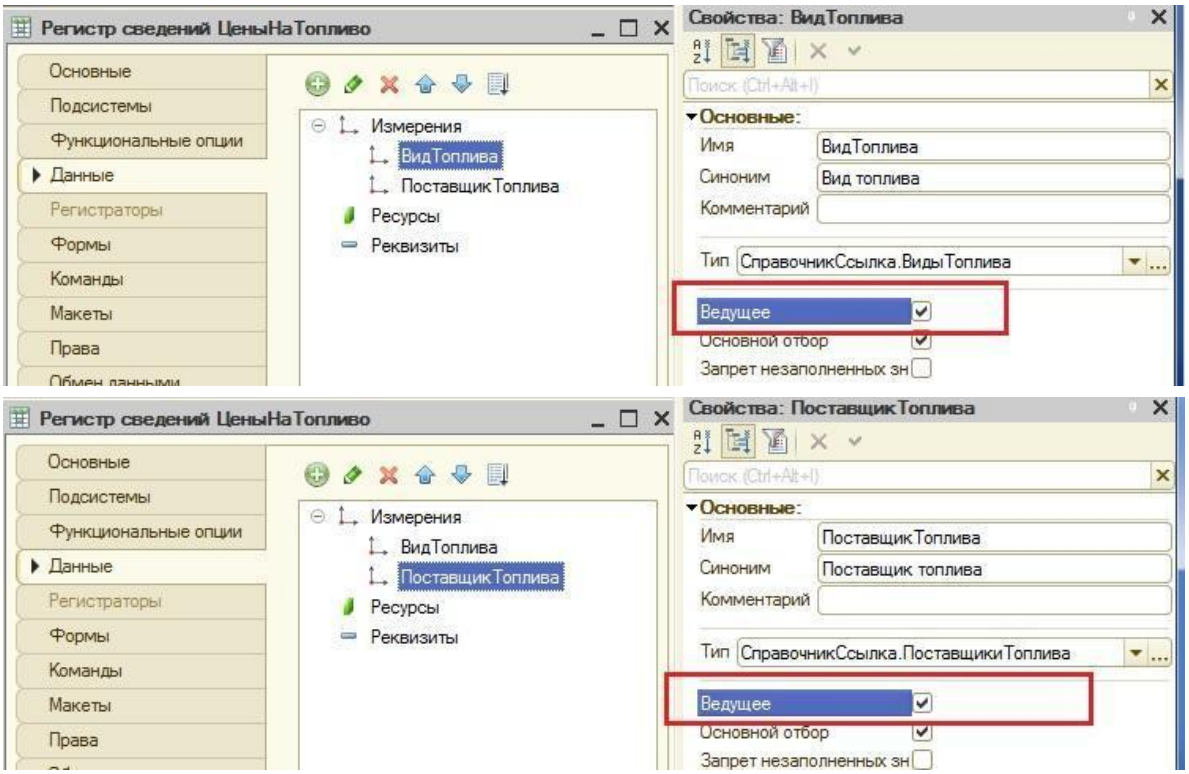
Теперь на закладке данные создадим два измерения – «ВидТоплива» и «ПоставщикТоплива», типы которых ссылки на соответствующие справочники.





У обоих измерений поставим флаг – «*Ведущее*». Это значит, что если мы удалим элемент справочника, который указан в какой-то записи регистра сведений, то эта запись удалиться автоматически. Так же есть одно интерфейсное следствие этого флага: если флаг установлен, то в форме элемента справочника мы сможем посмотреть на записи этого регистра для этого элемента.

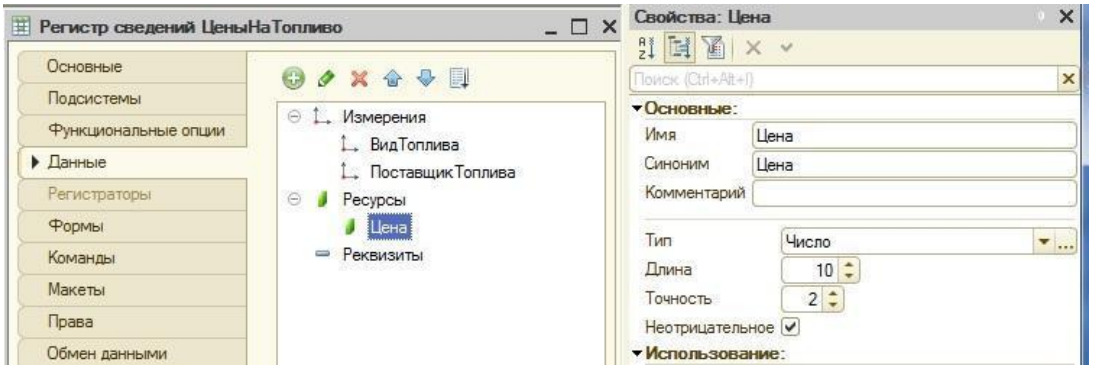
Установим для каждого измерения этот флаг.



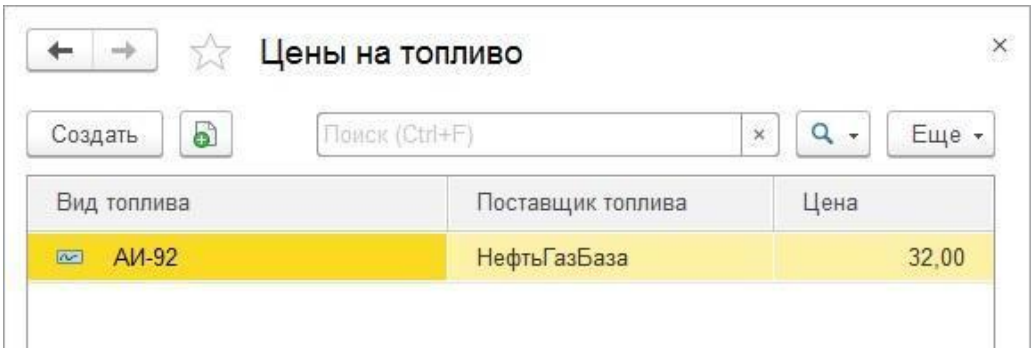
У нас непериодический регистр, и в нем два измерения «*ВидТоплива*» и «*ПоставщикТоплива*» это значит, что мы не сможем создать две записи с одинаковыми значениями полей «*ВидТоплива*» и «*ПоставщикТоплива*». Программа выдаст ошибку. Что и разумно – не может быть две разных цены на один и тот же вид топлива у одного и того же поставщика. А если может, то это значит, что необходимо добавить еще один разрез (например, база поставщика).

Кроме измерений у регистра сведений существуют «*Ресурсы*» и «*Реквизиты*». «*Ресурс*» должен хранить основную информацию регистра сведений, т.е. те данные, ради которых он создан, а «*Реквизит*» содержит дополнительную второстепенную информацию о записи.

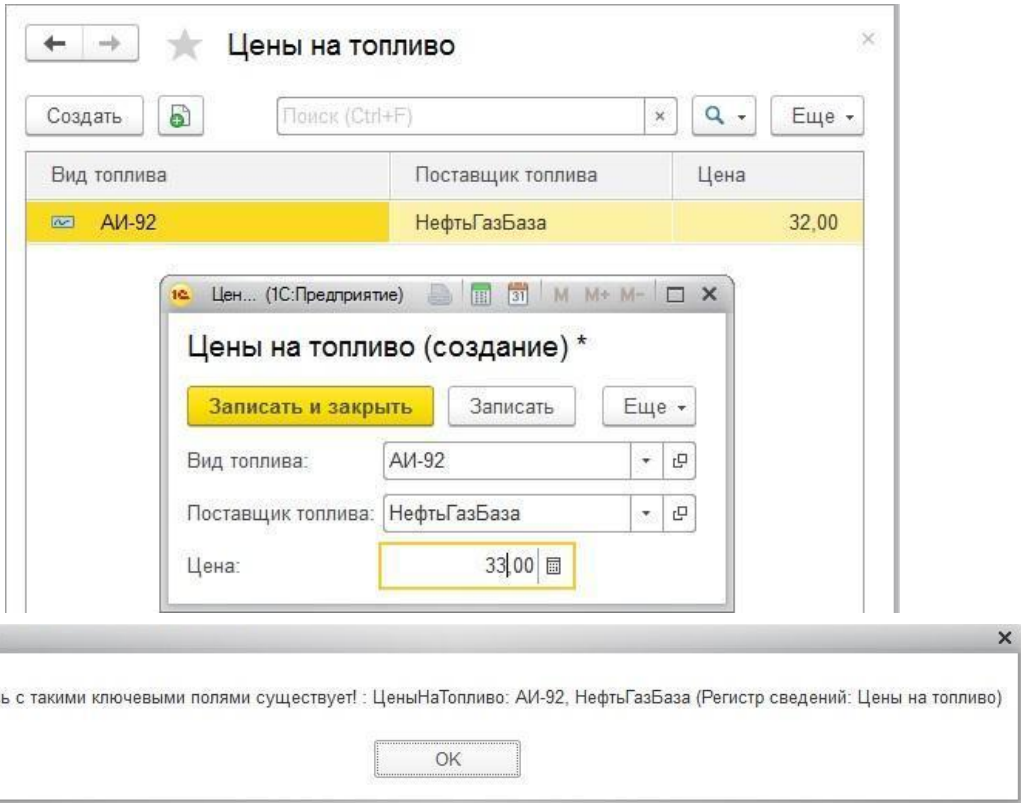
Мы создадим ресурс – «*Цена*» (тип число 10,2).



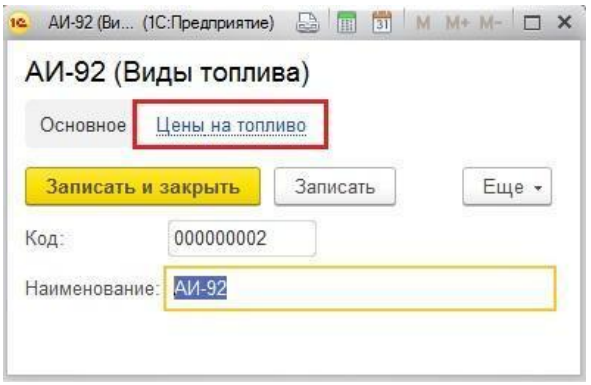
И всё. Сохраним конфигурацию и откроем этот регистр сведений и заведем какую-нибудь запись.



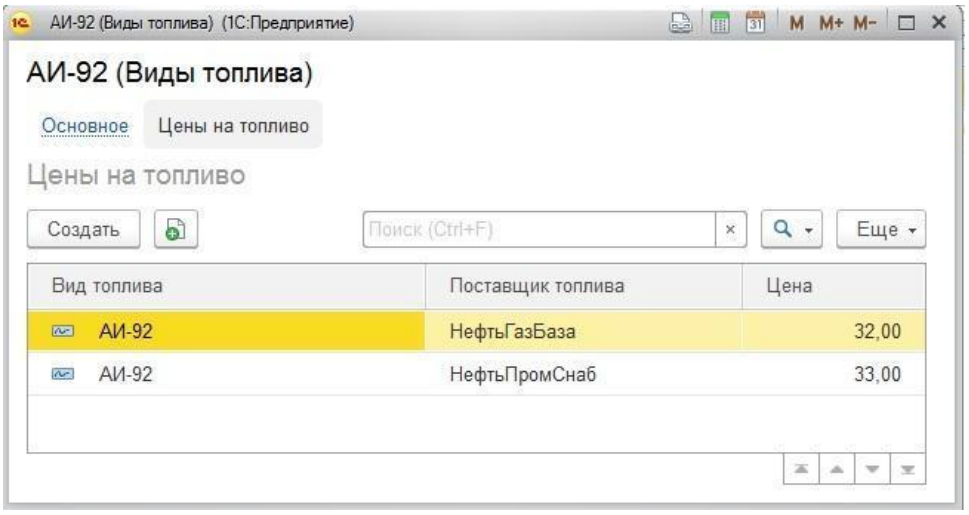
Если мы сейчас попробуем создать запись с точно таким же набором измерений, то возникнет ошибка: «Запись с такими ключевыми полями существует».



И последний момент: поскольку мы у измерения *ВидТоплива* установили флаг «Ведущее», то у элемента справочника *ВидыТоплива* появилась команда на открытие регистра сведений «Цена на топливо»



Если мы в управляемом приложении 1С перейдем по этой команде, то увидим все цены для нашего вида топлива.

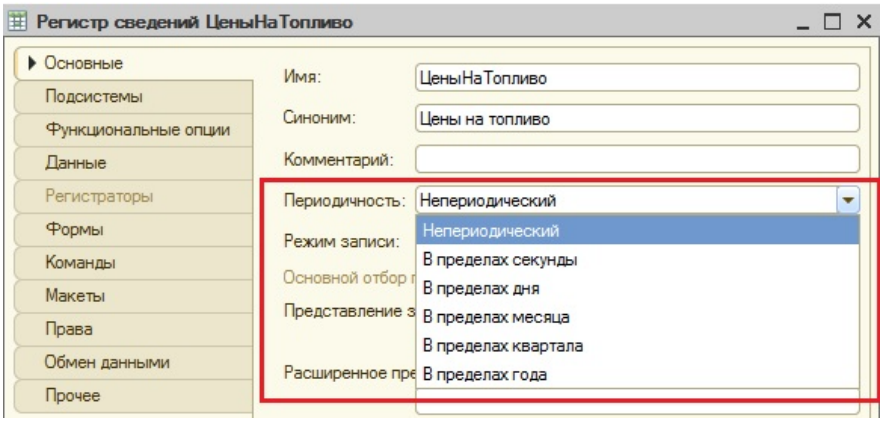


3. Периодические регистры сведений

В предыдущем примере наши показатели были статичны, т.е. была цена топлива и все, то в случае периодических регистров сведений эти показатели могут изменяться во времени. Например, цена топлива на 01.03.2018 может быть 33 р, а на 02.03.2018 – 34. Для такого учета мы можем использовать и обычный (непериодический) регистр сведений: просто наша запись с ценой для нужного вида топлива будет каждый раз переписываться для новой цены. Но, если у нас стоит цель знать историю цен и использовать цены старых периодов, то такое решение нам не подходит.

В этом случае необходимо использовать свойство Периодичность регистра сведений. Периодичность регистра сведений можно определить одним из следующих значений:

- в пределах секунды,
- в пределах дня,
- в пределах месяца,
- в пределах квартала,
- в пределах года;

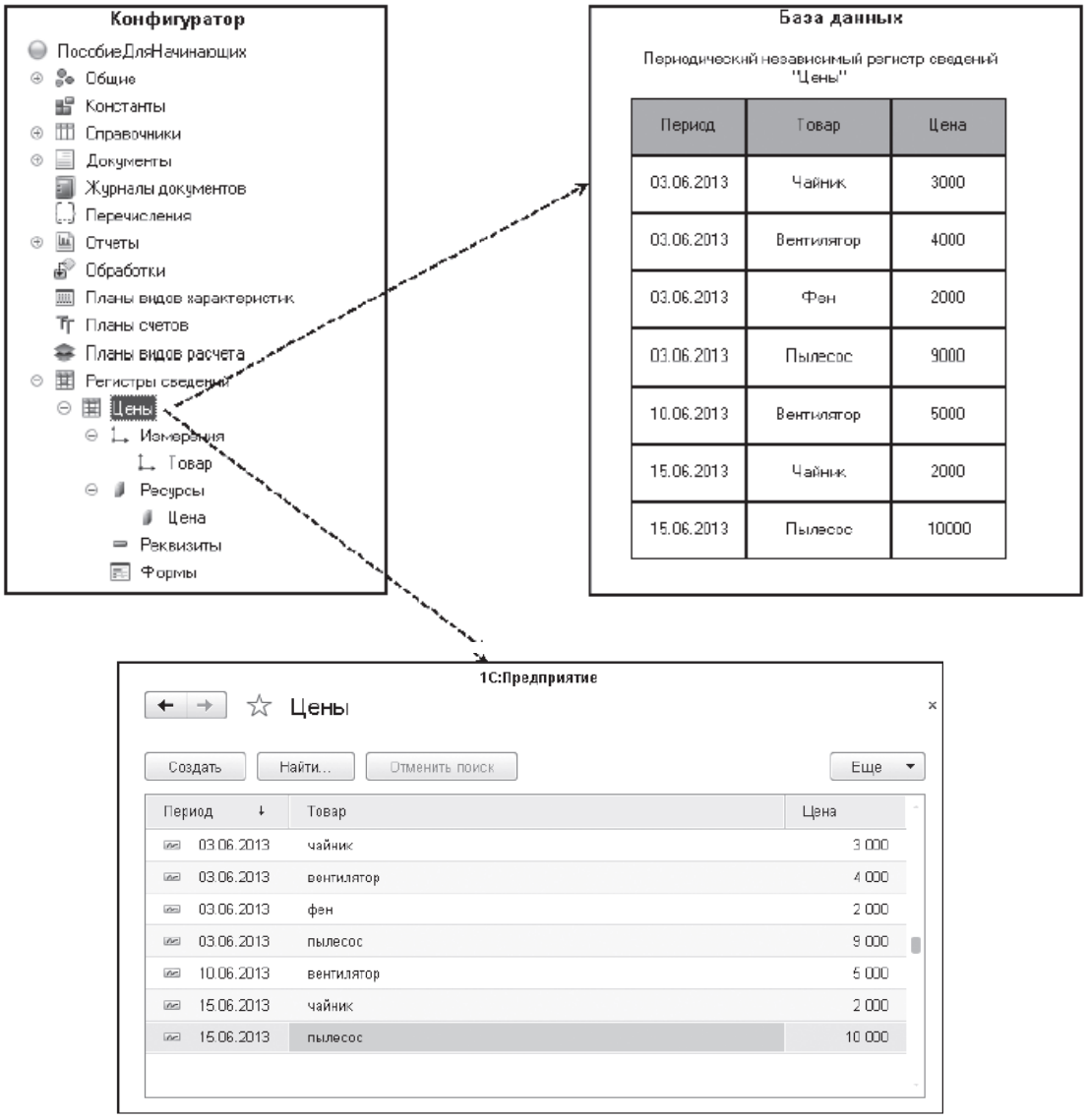


В Регистре сведений можно хранить не просто статические данные, но и историю изменений этих данных.

Периодичность нужна для выбора информации из регистра на определенный период времени. Если указать периодичность, записи в регистр будут производиться с периодом, когда была сделана запись. Например, периодический регистр сведений Цены сможет не

только хранить информацию о том, какова цена на определенную номенклатуру сейчас (или была в прошлом), но и о том, как она будет изменяться в будущем (если эти изменения будут заранее планироваться).

Периодичность в регистрах сведений нужна для информации, которая изменяется в течении времени, например: курсы валют, цены номенклатуры, скидки и наценки номенклатуры и т.д.



Периодический регистр сведений всегда содержит служебное поле Период, добавляемое системой автоматически. Оно имеет тип Дата и служит для указания факта принадлежности записи к какому-либо периоду. При записи данных в регистр платформа всегда приводит значение этого поля к началу того периода, в который он попадает.

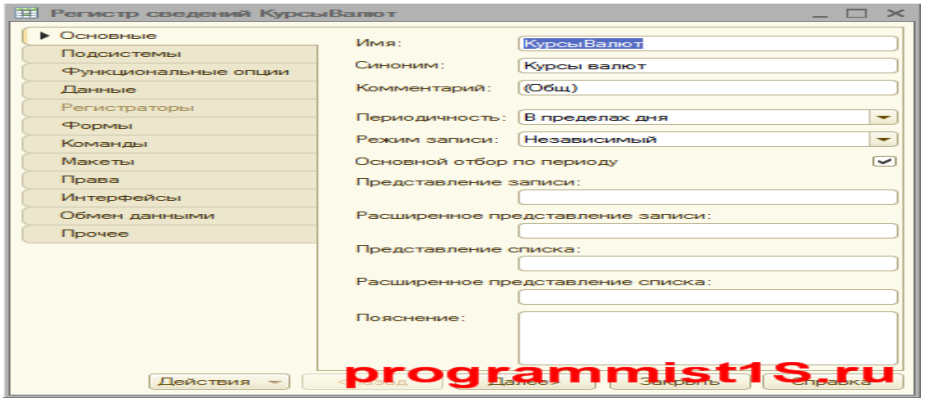
Например, если в регистр сведений с периодичностью в пределах месяца записать данные, в которых период указан как 08.04.2021, то регистр сохранит эти данные со значением периода, равным 01.04.2021.

Как и для других регистров, система контролирует уникальность записей для регистра сведений. Однако если для прочих регистров уникальным идентификатором записи является регистратор и номер строки, то для регистра сведений применяется другой принцип формирования ключевого значения.

Ключом записи, однозначно идентифицирующим запись, является в данном случае совокупность значений измерений регистра и периода (в случае если регистр сведений периодический). Регистр сведений не может содержать несколько записей с одинаковыми ключами. Если запись не уникальна – система 1С выдаст сообщение «Запись с такими ключевыми полями существует!» и не даст произвести запись в базу данных.

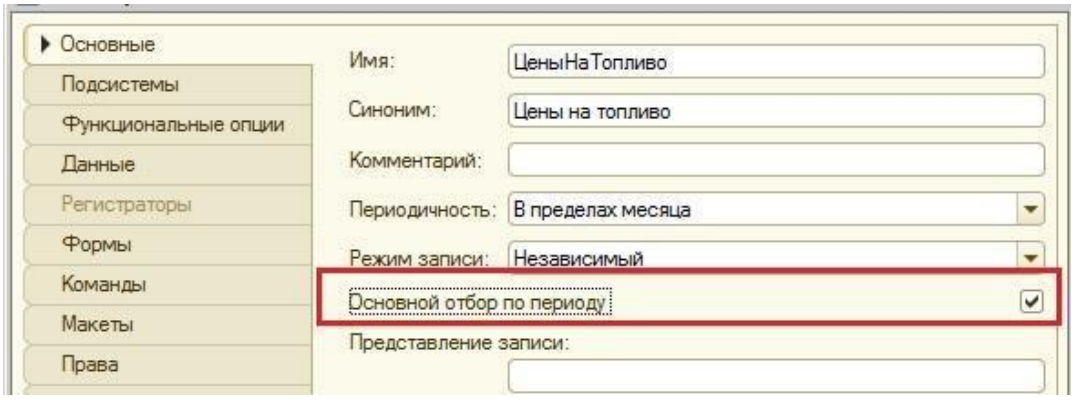
Например, для периодического регистра сведений с измерением Товар и ресурсом Цена ключом записи будет набор значений полей Период и Товар. Регистр сведений не может содержать несколько записей с одинаковыми ключами.

Еще один пример использования регистра сведений – хранение информации о курсе валют в разрезе валюты и периода.



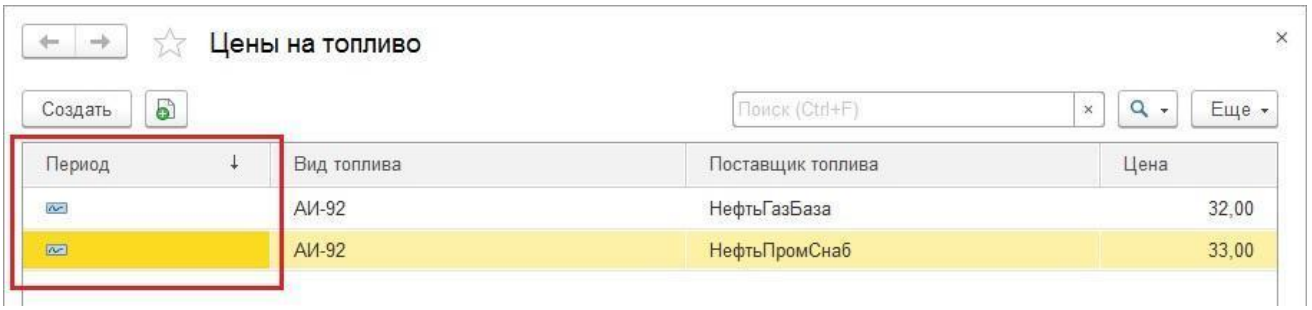
Одна из главных особенностей периодического регистра сведений – возможность получать готовые значения «Среза первых» и «Среза последних». Эта информация позволяет очень быстро получить из базы данных информацию о последнем (первом) установленном значении на определенную дату.

После того, как вы установили периодичность регистра сведений стал доступен флаг «Основной отбор по периоду».



Этот флаг необходим, в случае работы распределенной информационной базы, это значит, что при обмене данными отбор будет вестись, в том числе, по периоду.

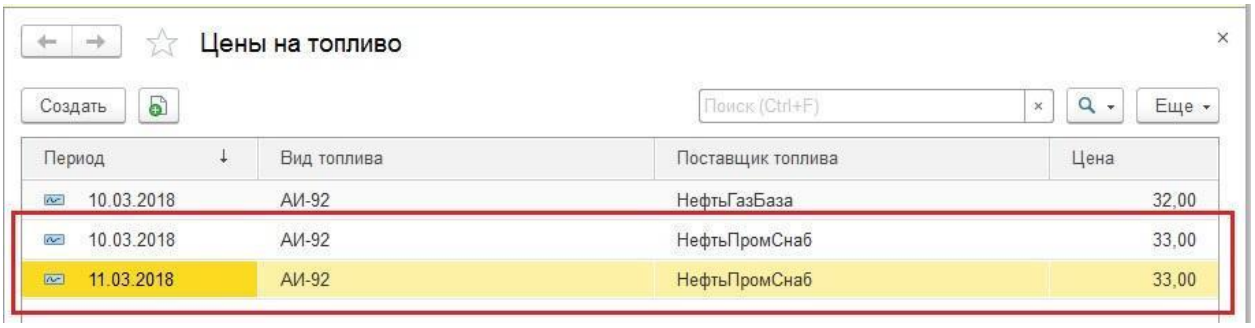
После того, как мы в конфигураторе 1С у регистра сведений добавили периодичность, в управляемом приложении 1С у наших записей появилось новое поле *период*.



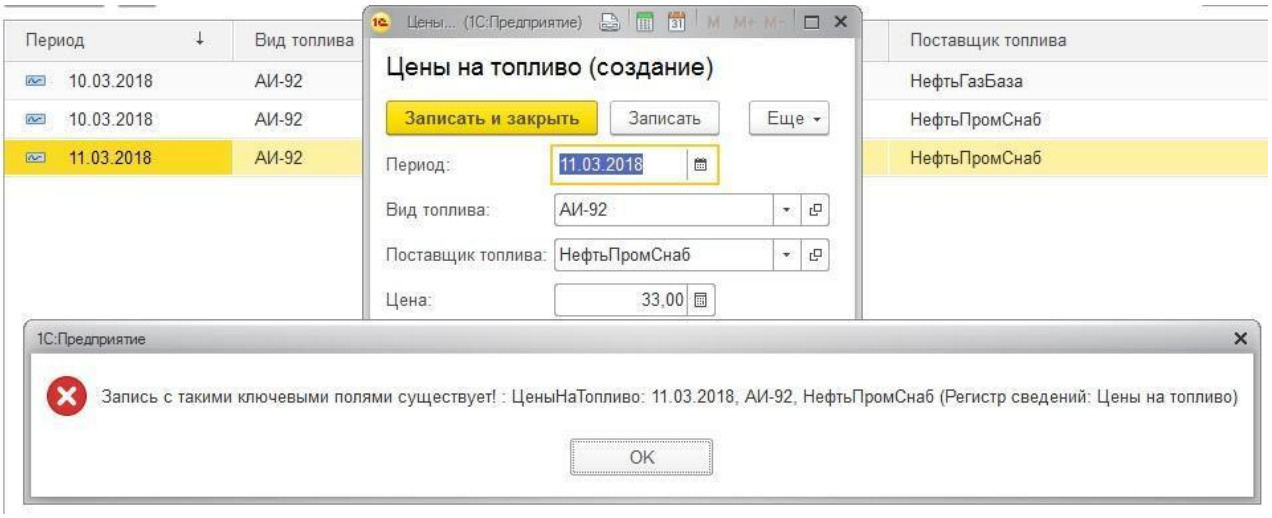
В этом поле необходимо указывать дату, на которую приходится соответствующая запись. Причем эта дата будет кратной той периодичности, которую мы установили в свойстве основной отбор. Например, если у нас периодичность месяц, то нельзя в поле Период установить дату 25.03.2018, платформа сама округлит её до 01.03.2018, а если будет периодичность год, то платформа округлит дату до начала года 01.01.2018.

Мы у нашего регистра сведений «Цены номенклатуры» установим периодичность «В пределах дня».

И если раньше комбинация по двум одинаковым измерениям вызывала ошибку, то теперь мы сможем это осуществить с разной периодичностью.



Но, в тоже время период, по сути, становиться измерением, поэтому нельзя сделать запись с одним и тем же набором полей и периодом. Возникнет ошибка. Поле *Период* это, по сути, то же измерение.



Использование периодических регистров сведений позволяет хранить историю данных в любых разрезах: например, мы можем узнать какая была цена конкретного вида топлива конкретного поставщика в любой день (если есть соответствующая запись).

Мы рассмотрели независимые регистры сведений, но также можно создавать подчиненные регистры сведений: записи в которых создают документы.

4. Подчиненные регистры сведений

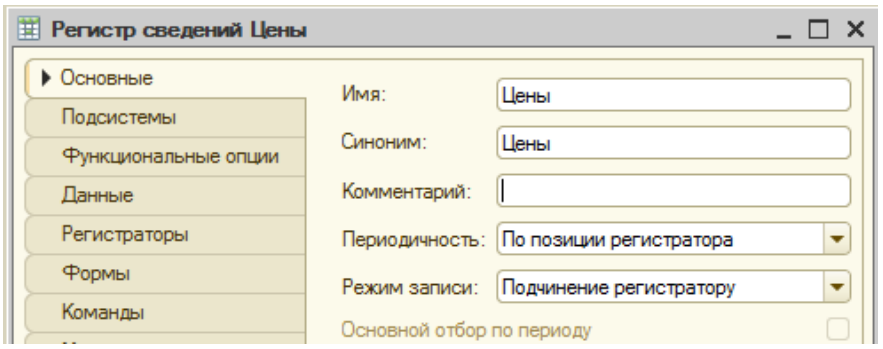
Подчинение записей регистратору. Использование режима записи Подчинение регистратору может потребоваться в случае, когда логика работы прикладного решения требует того, чтобы изменения, выполняемые в регистре сведений, были жестко связаны с документами, фиксирующими факты хозяйственной деятельности.

Например, изменение отпускных цен компании может производиться только определенным кругом лиц, и каждое такое изменение должно сопровождаться оформлением соответствующего документа. То есть данные регистра сведений должны четко «помнить», в связи с каким именно документом они там оказались. Для этого достаточно установить значение Подчинение регистратору свойству Режим записи регистра сведений как объекта конфигурации.

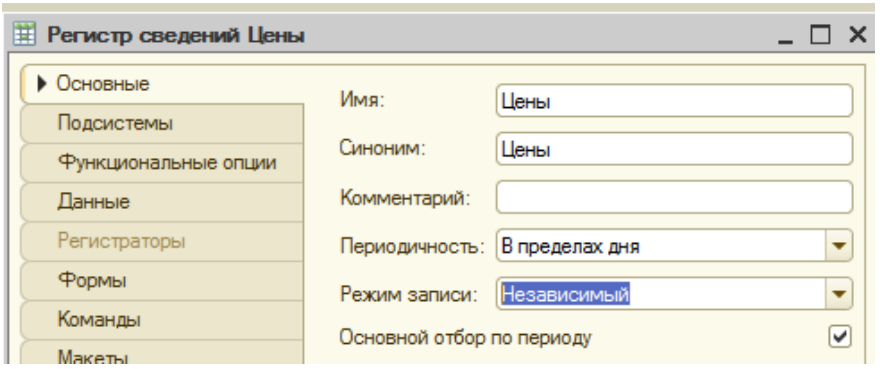
С точки зрения хранения данных в регистре установка значения Подчинение регистратору свойства Режим записи приводит к включению в состав регистра полей Регистратор, НомерСтроки и Активность.

Далее требуется указать, какие именно документы могут быть регистраторами записей регистра. Это можно сделать как при редактировании самого регистра, как объекта конфигурации, так и при редактировании документа-регистратора как объекта конфигурации. Благодаря этому облегчается труд разработчика, которому необходимо сформировать или модифицировать набор (наборы) записей регистра (регистров), подчиненных некоему документу.

Поле Регистратор обеспечивает привязку к документу, но не всегда входит в разряд ключевых полей. Для того чтобы поле Регистратор стало ключевым, необходимо для такого регистра сведений (с режимом записи Подчинение регистратору) установить для свойства Периодичность значение По позиции регистратора.

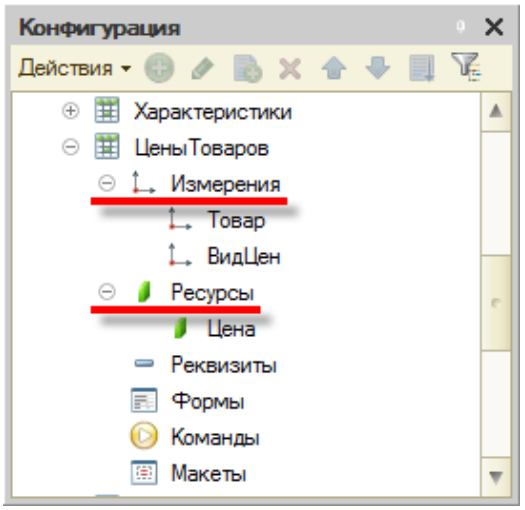


Если продолжать сравнение с регистром накопления, то можно сказать, что регистр сведений предоставляет больше свободы в редактировании хранимых данных. Наряду с возможностью использования в режиме подчинения регистратору (когда записи регистра сведений «привязаны» к документу-регистратору) регистр сведений может применяться и в независимом режиме, в котором пользователю предоставляется полная свобода интерактивной работы с данными регистра. Регистр сведений, не использующий подчинение регистратору, называют независимым регистром сведений.



Структура регистра сведений. В состав регистра сведений как объекта конфигурации включаются:

- измерения,
- ресурсы,
- реквизиты.



Ресурсы хранят данные регистра, то есть ту информацию, ради хранения которой регистр, собственно, создавался и с которой работает функциональность регистра.

Измерения используются для обеспечения разрезов хранения этой информации.

Для ситуаций, когда необходимо кроме данных хранить дополнительную информацию собственно о записях регистра, возможно использование реквизитов.

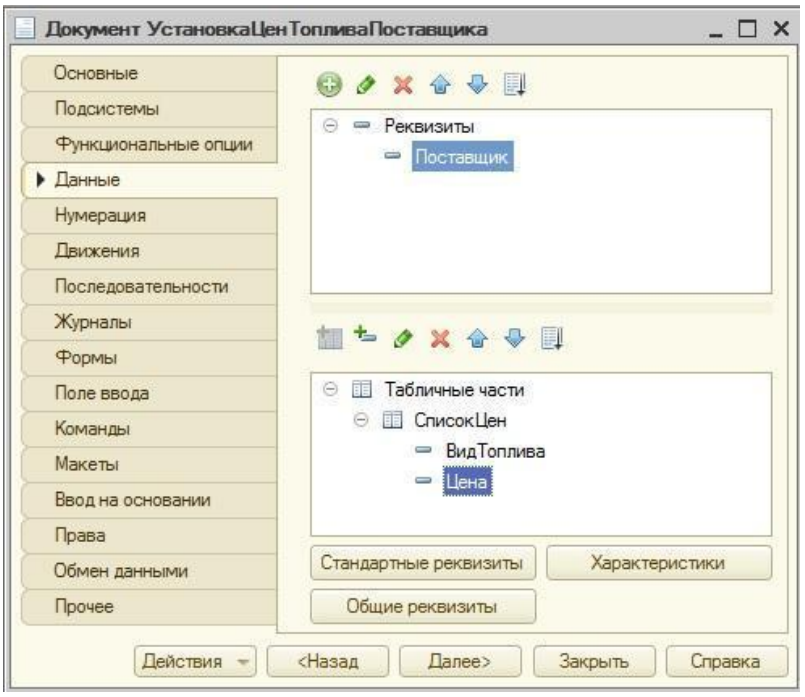
Как было рассмотрено выше, регистры сведений могут иметь режим записи Подчинение регистратору, вследствие чего в состав таблиц регистра добавляется поле Регистратор. И регистр сведений может быть периодическим, вследствие чего в состав его таблиц добавляется поле Период.

Данные каждого регистра сведений хранятся в отдельной таблице базы данных.

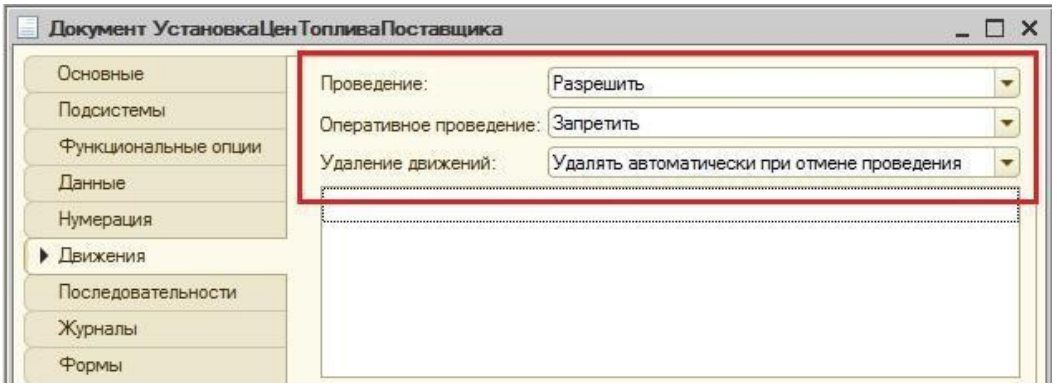
Каждая такая таблица имеет следующий состав колонок:

- Период – дата записи. Определяет положение данной записи на временной оси. Это поле существует только для периодических регистров;
- Регистратор – содержит ссылку на документ, которому подчинена данная запись. Это поле существует только для регистров с режимом записи Подчинение регистратору;
- НомерСтроки – уникальный номер данной записи в наборе записей регистра, подчиненных документу, указанному в поле Регистратор. Это поле существует только для регистров с режимом записи Подчинение регистратору;
- Активность – имеет тип Булево. Содержит признак влияния записи на получение информации из регистра. Записи, для которых значение данного свойства установлено в Ложь, не будут учитываться при получении «первых» или «последних» записей регистра, а также при получении сведений на определенный момент времени. Это поле существует только для регистров с режимом записи Подчинение регистратору;
- Ключ – короткий ключ записи регистра. Поле присутствует у непериодических регистров, имеющих хотя бы одно измерение;
    - <Измерение> – содержит значение измерения. Количество таких полей равно количеству измерений, определенных для регистра как объекта конфигурации;
    - <Ресурс> – значение ресурса. Количество таких полей равно количеству ресурсов, определенных для регистра как объекта конфигурации;
    - <Реквизит> – значение реквизита. Количество таких полей равно количеству реквизитов, определенных для регистра как объекта конфигурации.

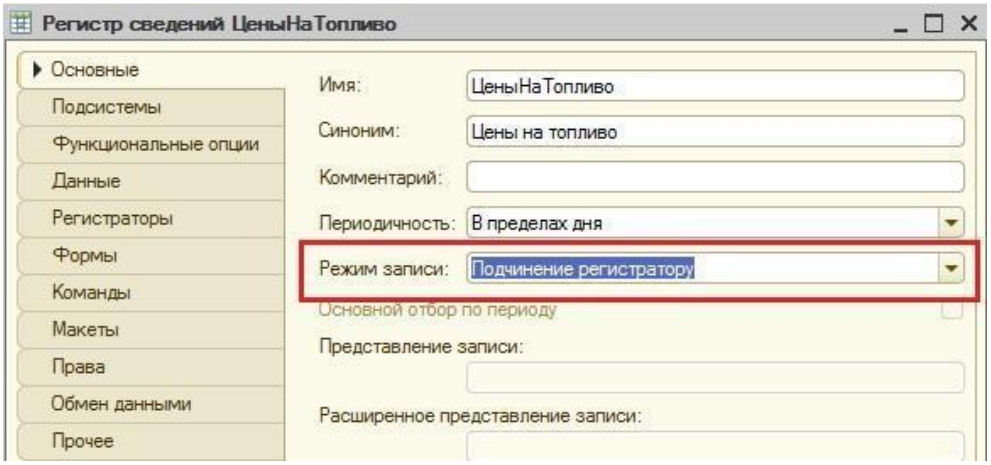
В предыдущем примере, мы создали регистр сведений «Цены на топливо», и сделали его периодическим. Теперь сделаем его подчиненным. Для этого создадим в конфигураторе 1С документ «Установка цен топлива поставщика», у которого будет один реквизит «Поставщик» с типом *СправочникСсылка.ПоставщикТоплива*, а так же табличная часть «Список цен» с реквизитами: «Вид топлива» с типом *СправочникСсылка.ВидыТоплива* и цена с типом Число (10,2), как у ресурса регистра сведений.



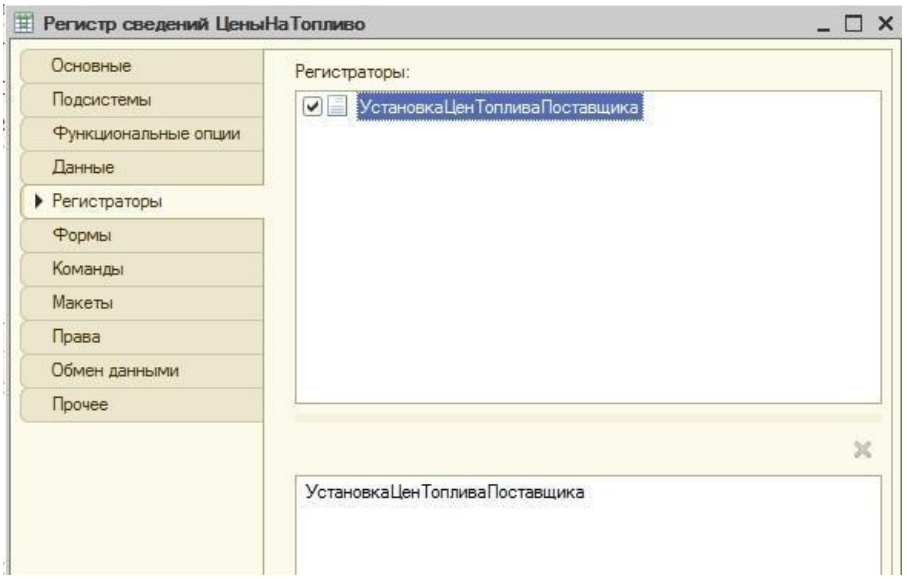
На закладке «Движения» конструктора документов свойство *Проведение* установим разрешить, а *Оперативное проведение* – запретить (цены можно устанавливать и задним числом).



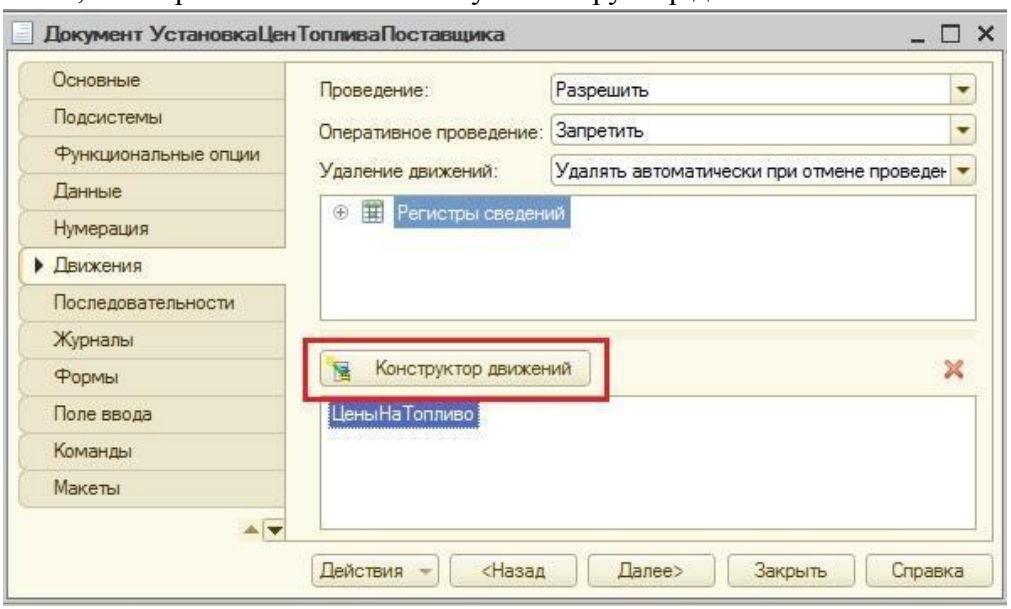
У регистра сведений режим записи установим «Подчинение регистратору», а периодичность оставим «В пределах дня».



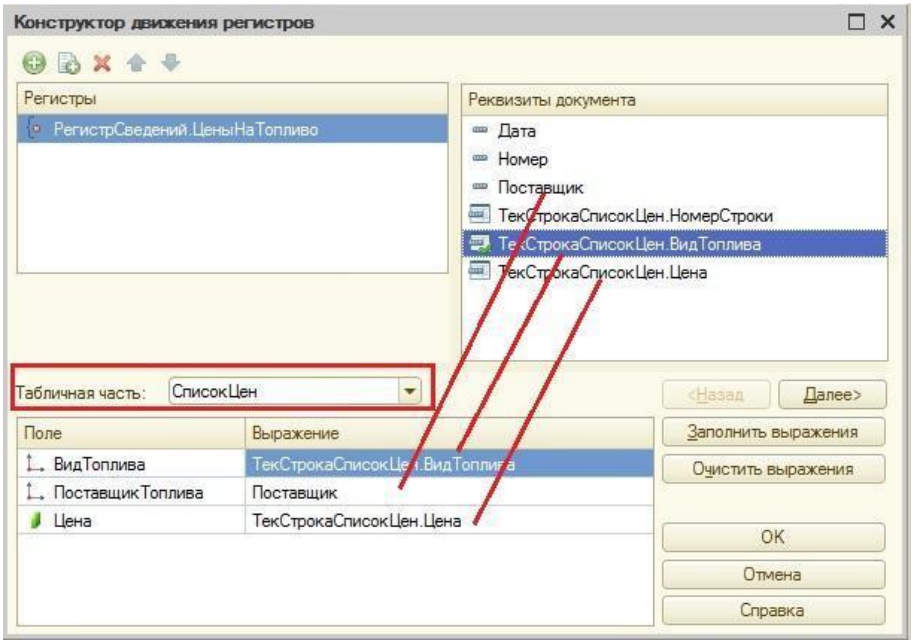
На закладке «Регистраторы» регистра сведений установим документ «Установка цен топлива поставщика»



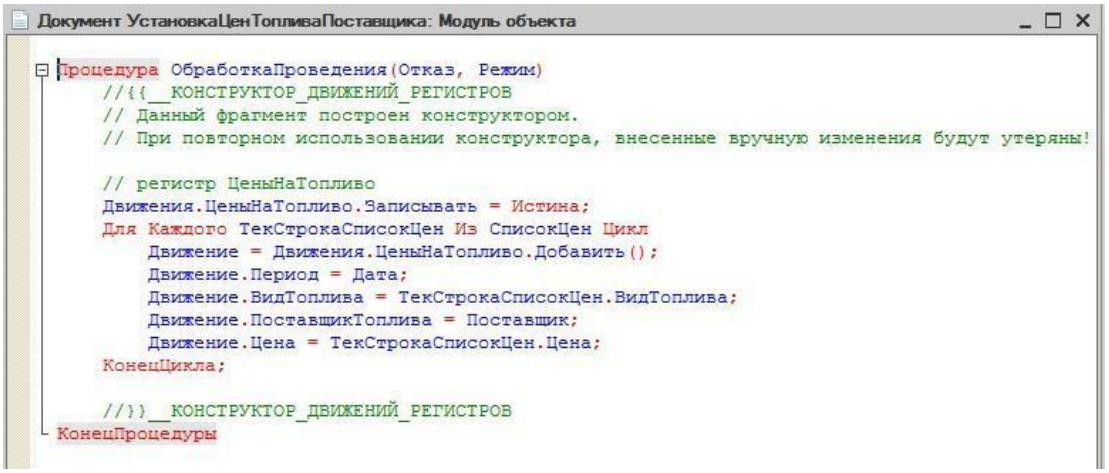
Осталось «прописать» движения этого документа по регистру сведений. Мы это сделаем при помощи конструктора. Для этого перейдем в редактор документа на закладку «Движения», в котором нажмем на кнопку «Конструктор движения».



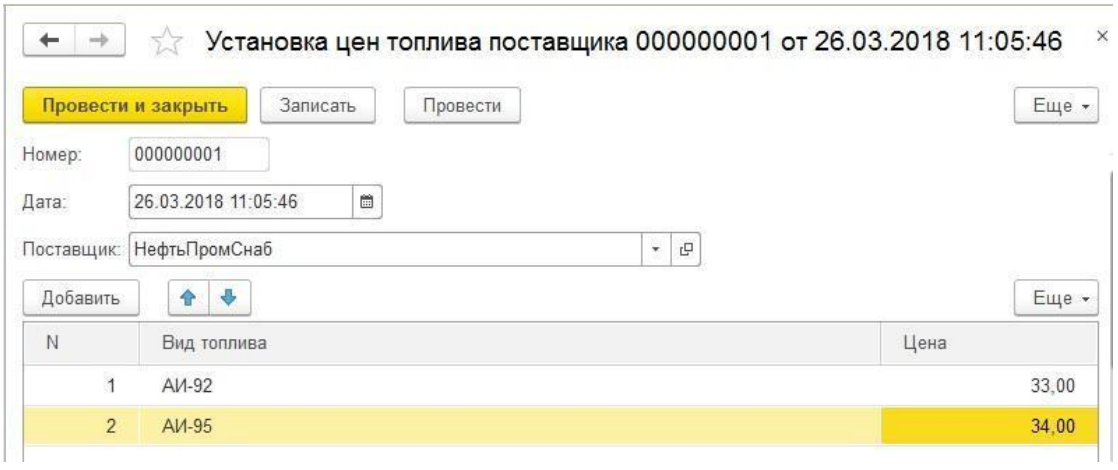
В открывшемся конструкторе движения регистров выберем табличную часть «Список цен» и свяжем реквизиты табличной части и документа с полями регистра сведений.



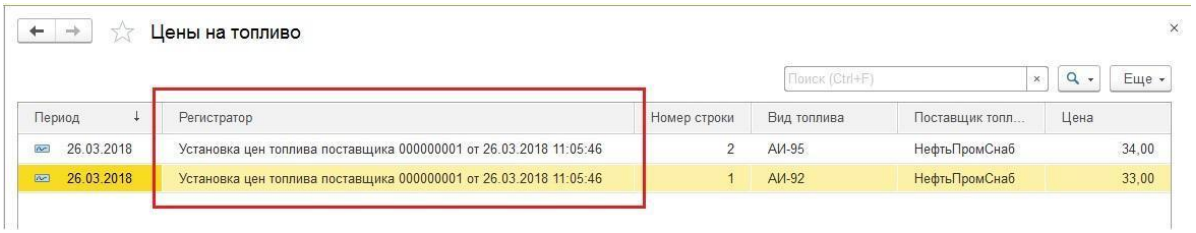
После этого в модуле документа «Установка цен топлива поставщика» должен появиться код, который будет осуществлять запись движений по регистру сведений при проведении документа.



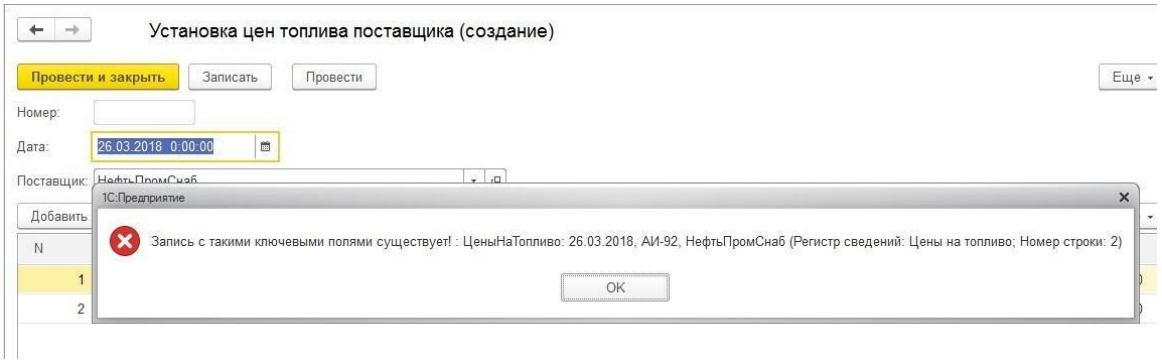
Запустим управляемое приложение 1С: Предприятия, и, если мы создадим документ и запишем в него какие-то данные.



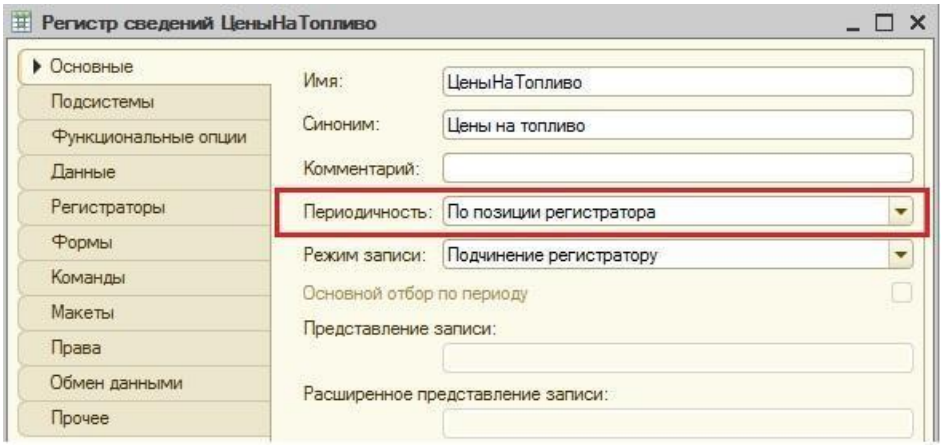
То после его проведения, будут созданы записи в регистре сведений. Причем у регистра сведений появится новое поле *Регистратор*.



Поскольку у нас периодичность регистра сведений *День*, то при проведении нового документа с точно таким же набором данных и за тот же период, возникнет ошибка «Запись с такими ключевыми полями существует!».



Если по прикладной задаче стоит условие, что в один период можно создавать и проводить разные документы, то для избежание подобных ошибок нам необходимо изменить периодичность регистра сведений на *По позиции регистратора*.



После такой установки свойств регистра сведений можно создавать и проводить несколько документов с одинаковыми периодами и одинаковыми наборами данных.

Регистры, записываемые независимо, могут свободно редактироваться вручную или средствами встроенного языка. При этом если измерение такого регистра назначено как «ведущее» и значением измерения является ссылка на объект базы данных, то будет считаться, что запись регистра имеет смысл, только пока существует этот объект. Например, если назначить ведущим измерение «Конкурент», то считается, что запись имеет смысл только как информация по данному конкуренту. Соответственно, при удалении конкурента записи по нему будут удалены автоматически.

Если регистр записывается регистратором, то это значит, что записи будут жестко подчинены регистраторам – документам. Обычно это значит, что записи будут порождаться при проведении документов. Соответственно, при удалении документа

записи будут удаляться автоматически. В отличие от ведущих измерений, регистратор может быть только один.

5. Программная запись в регистр сведений 1С

Программный код для работы с регистром сведений. В программных модулях для общих действий над регистром сведений (поиск, выбор и создание записей регистра) служит объект РегистрСведенийМенеджер.<Имя регистра сведений>.

Для чтения, записи и удаления отдельных записей регистра сведений, не управляемого регистраторами, служит объект РегистрСведенийМенеджерЗаписи.<Имя регистра сведений>.

Для считывания и занесения набора записей в базу данных по определенному условию отбора служит объект РегистрСведенийНаборЗаписей.<Имя регистра сведений>.

Для динамического обхода записей регистра служит объект РегистрСведенийВыборка.<Имя регистра сведений>.

5.1.Запись в регистр сведений 1С

Строки в регистре с периодом и регистратором, содержащие информацию о ресурсах в разрезе измерений, называются записями.

Чтобы добавить запись в регистр используются или менеджер записей, или набор записей. Если у записей в регистре имеется общий ключ, то необходимо использовать НаборЗаписей. А для записи одной единственной записи, если в регистре все записи уникальны, необходимо использовать МенеджерЗаписи.

Пример записи при использовании объекта РегистрСведенийНаборЗаписей.

Использование менеджера записи:

```
НоваяЗапись = РегистрыСведений.КурсыВалют.СоздатьМенеджерЗаписи();
НоваяЗапись.Валюта =    Справочники.Валюты.НайтиПоНаименованию(«USD»);
НоваяЗапись.Период = Дата(31,12,2016);
НоваяЗапись.Курс = 100;
НоваяЗапись.Кратность = 1;
НоваяЗапись.Записать();
```

При использовании набора записей и метода «Записать» происходит запись в регистр сведений набора записей. При этом может происходить как просто добавление строк, так и замещение уже имеющихся строк в регистре. Для независимых регистров, без установки отборов, будет произведено удаление всех записей в регистре и замещение на добавляемые записи.

Если записать без отбора данные в подчиненны регистр, возникнет ошибка.

Пример записи с использованием набора записей в подчиненный регистратору регистр сведений ЦеныНоменклатуры:

```
НовыйНаборЗаписей =    РегистрыСведений.ЦеныНоменклатуры.СоздатьНаборЗаписей();
НовыйНаборЗаписей.Отбор.Регистратор.Установить(Ссылка);
НоваяЗаписьНабора = НовыйНаборЗаписей.Добавить();
НоваяЗаписьНабора.Период = Ссылка.Дата;
НоваяЗаписьНабора.Номенклатура = Ссылка.Номенклатура;
НоваяЗаписьНабора.Цена = Ссылка.Цена;
НовыйНаборЗаписей.Записать();
```

Пример записи через менеджер записи:

```
Запись = РегистрыСведений.КурсыВалют.СоздатьМенеджерЗаписи();
Запись.Период = Дата;
Запись.Курс = Курс;
Запись.Валюта = Валюта;
Запись.Записать();
```

5.2.Поиск и чтение в регистре сведений

При помощи менеджера можно также легко менять или удалять единичные записи. Найдём, изменим, а затем удалим созданную запись.

```
Выборка = РегистрыСведений.КурсВалют.Выбрать('20140101', '20140101');
Если Выборка.Следующий() Тогда
    Запись = Выборка.ПолучитьМенеджерЗаписи();
    // изменим
    Запись.Прочитать();
    Запись.Курс = 25;
    // запишем
    Запись.Записать(Истина);
    // и тут же удалим
    Запись.Удалить();
КонецЕсли;
```

5.3.Изменение и удаление записей

Чтобы удалить запись регистра сведений, например, все курсы валюты EUR, воспользуйтесь следующим кодом:

```
СтруктураОтбора = новый Структура("Валюта",    Справочники.Валюты.НайтиПоНаименованию("EUR"));
    Выборка = РегистрыСведений.КурсыВалют.Выбрать(,,СтруктураОтбора);
    Пока Выборка.Следующий() Цикл
        МенеджерЗаписи = Выборка.ПолучитьМенеджерЗаписи();
        Выборка.ПолучитьМенеджерЗаписи().Удалить();
    КонецЦикла;
```

Для быстрой и полной очистки регистра можно использовать следующий код:

```
НоваяЗапись = РегистрыСведений.ТестовыйРегистр.СоздатьНаборЗаписей();
НоваяЗапись.Записать();
```

Чтобы скорректировать и изменить регистр, а также быстро заполнить регистр данными, можно написать универсальную обработку.

5.4.Обращение к периодическим сведениям с помощью методов

Объект **РегистрСведенийМенеджер** позволяет обращаться к «итогам» регистра. Под «итогами» периодического регистра сведений понимаются первые или последние значения ресурсов по указанным измерениям. При этом применяются следующие методы:

Метод	Описание
Получить **(<Период>, <Отбор>)**	Возвращает в виде структуры значения ресурсов одной записи регистра, соответствующей указанным

	значениям **всех** (!) измерений регистра и периоду.
ПолучитьПоследнее **(<Конец периода>, <Отбор>)**	Этот метод возвращает актуальное значение ресурсов, действовавшее на заданную дату. Если он не находит запись в регистре по данной комбинации измерений точно на заданный период, то возвращается структура, содержащая значения ресурсов ближайшей **более поздней** записи.
ПолучитьПервое **(<Начало периода>, <Отбор>)**	Этот метод действует аналогично методу ПолучитьПоследнее, но если записи на данный момент не находится, то возвращается структура, содержащая значения ресурсов ближайшей **более ранней** записи.
СрезПоследних **(<Конец периода>, <Отбор>)**	Эти методы аналогичны методам ПолучитьПоследнее и ПолучитьПервое соответственно , но при их использовании, как правило, не указывается одно или несколько измерений. В результате возвращается не структура, как в предыдущих случаях, а **таблица значений**, заполненная данными найденных записей регистра сведений.
СрезПервых **(<Начало периода>, <Отбор>)**	

При вызове методов ПолучитьПервое, ПолучитьПоследнее, СрезПервых и СрезПоследних первый параметр может иметь тип "дата", МоментВремени или Граница, но его можно вообще не указывать, тогда будут найдены значения ресурсов из самой первой или последней записи регистра соответственно.

Продемонстрируем использование этих методов на примерах:

Пример 1. Получение курса валюты точно на заданную дату (если записи именно на эту дату нет, то в элементах структуры будут пустые значения).

```
ВалютаОтбор = Новый Структура ("Валюта", ВыбВалюта);
СтруктураКурсКратность = РегистрыСведений.КурсыВалют.Получить(ВыбДата,
ВалютаОтбор);
Если СтруктураКурсКратность.Курс = 0 Тогда
    Сообщить("Курс точно на эту дату не указан!");
Иначе
    Сообщить("Курс валюты:" + СокрЛП(СтруктураКурсКратность.Курс) + ", кратность: "
+ СокрЛП(СтруктураКурсКратность.Кратность));
КонецЕсли;
```

Пример 2. Получение актуального курса валюты на заданную дату (если записи именно на эту дату нет, то будет найдена ближайшая более поздняя запись)

```
ВалютаОтбор = Новый Структура("Валюта", ВыбВалюта);
СтруктураКурсКратность =
РегистрыСведений.КурсыВалют.ПолучитьПоследнее(ВыбДата, ВалютаОтбор);
Сообщить("Актуальный курс на заданную дату: " + СтруктураКурсКратность.Курс);
```

Пример 3. Получение актуальных курсов всех валют на заданную дату (отбор не указан, т.е. мы хотим получить сведения по всем значениям измерений).

```
тзДанные = РегистрыСведений.КурсыВалют.СрезПоследних(ВыбДата, );
Для Каждого Стр Из тзДанные Цикл
    Сообщить("Для валюты " + Строка(Стр.Валюта) + " курс на заданную дату: " +
Строка(Стр.Курс) + ", кратность: " + Строка(Стр.Кратность));
КонецЦикла;
```

Пример 4. Получение актуальных оптовых цен на товары (указан отбор по измерению "ТипЦен")

```
ОтборТипЦен = Новый Структура("ТипЦен", Перечисления.ТипыЦен.Оптовая);
тзДанные = РегистрыСведений.ЦеныКомпании.СрезПоследних(ВыбДата, ОтборТипЦен);
Для Каждого Стр Из тзДанные Цикл
    Сообщить("Для номенклатуры " + Строка(Стр.Номенклатура) + " оптовая цена: " +
Строка(Стр.Цена));
КонецЦикла;
```

6. Обработка «Проведение» документа.

Запись документа — это процесс переноса значений реквизитов в таблицы базы данных.

Запись бывает трех видов:

- Запись
- Проведение
- Отмена проведения

Проведение документа — это запись документа в режиме проведения. Как правило при проведении документа формируются движения по регистрам. Также при проведении будет вызываться обработчик **ОбработкаПроведения** из модуля объекта. У документа есть реквизит **Проведен**, который равен Истина, если документ проведен и Ложь, если не проведен.

Движения документа — это набор записей регистра (накопления, бухгалтерии, расчета, сведений), который подчинен регистратору (документу).

Состав движений документа настраивается на закладке **Движения**. Там же можно разрешить или запретить проведение документа. Если проведение запрещено, то у документа не будет кнопки «Провести», при записи не будет вызываться обработка проведения в модуле объекта.

Отмена проведения документов – при записи документа в режиме отмены проведения выполняется удаление движений по регистрам. До непосредственного удаления движений вызывается обработчик **ОбработкаУдаленияПроведения** в модуле объекта документа. В данном обработчике можно прописать свой алгоритм действий при отмене приведения документа.

На закладке Движения можно настроить будут ли автоматически удалять движения при отмене проведения или нет. За это отвечает свойство **УдалениеДвижений**:

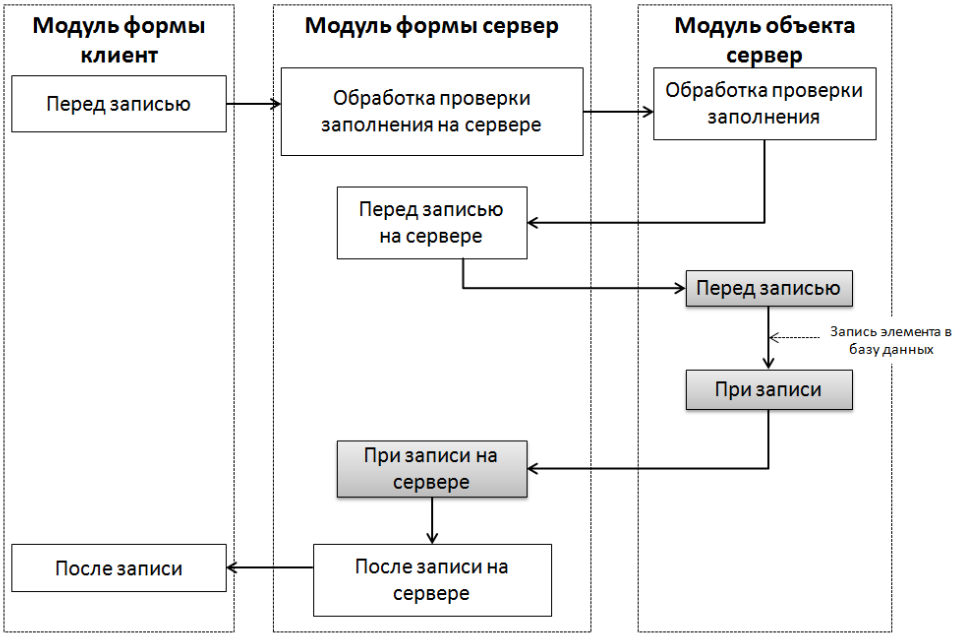
Свойство УдалениеДвижений может принимать три значения:

- **Удалять автоматически при отмене проведения** — движения документа будут автоматически удаляться только при отмене проведения документа. При перепроведении документа движения не удаляются, а перезаписываются.

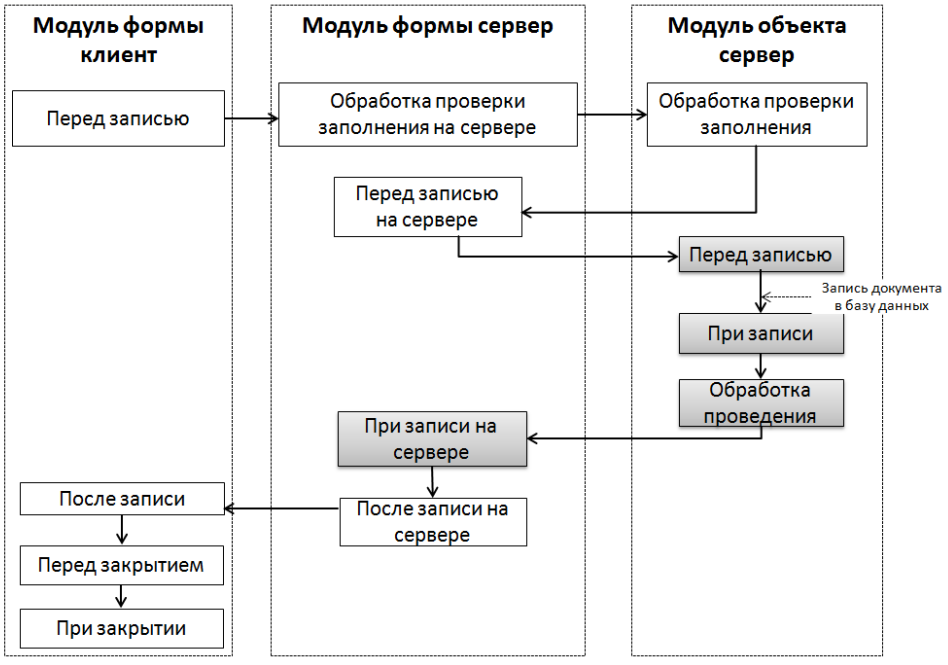
- **Удалять автоматически** — движения документа будут автоматически удаляться как при отмене проведения, так и при перепроведении документа.
- **Не удалять автоматически** — движения документа не будут удаляться автоматически.

При создании нового документа в конфигураторе автоматически устанавливается Удалять автоматически при отмене проведения.

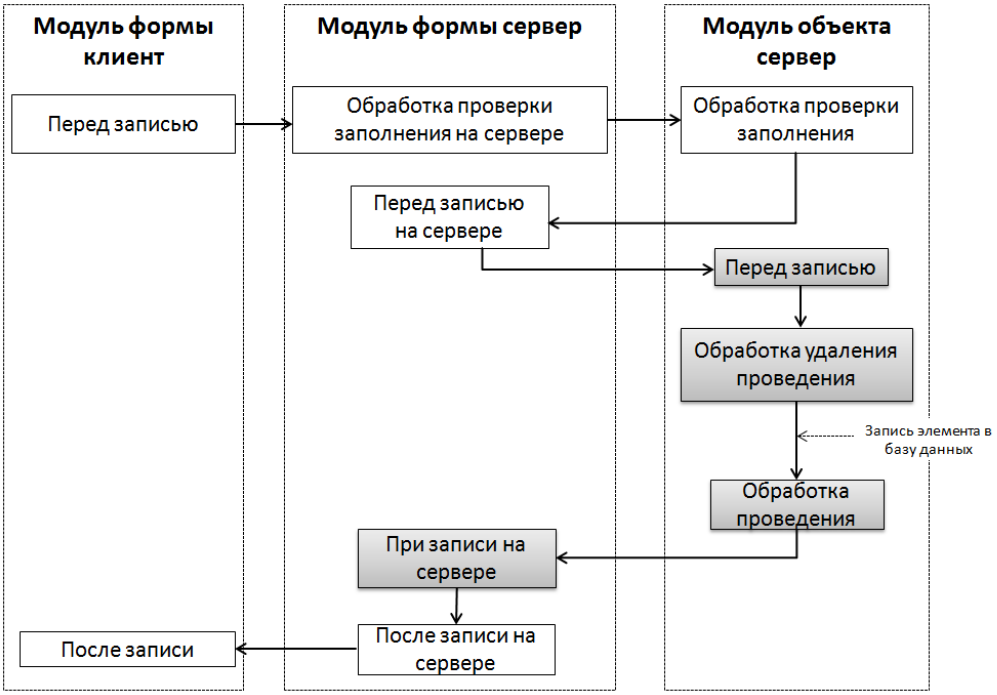
Последовательность событий при записи документа из формы документа. Заливкой выделены события, выполняющиеся в транзакции записи.



Последовательность событий при проведении документа из формы документа (провести и закрыть). Заливкой выделены события, выполняющиеся в транзакции записи.



Последовательность событий при отмене проведения документа из формы документа. Заливкой выделены события, выполняющиеся в транзакции записи.



7. Обработчики реквизитов

При работе с событиями на платформе «1С:Предприятие» следует различать два типа событий: события, связанные с формой и ее элементами, и все остальные. Разница заключается в том, что обработчики событий, связанных с формой и ее элементами, – назначаемые, а обработчики всех остальных событий – фиксированные.

Фиксированный обработчик события должен иметь имя, совпадающие с именем события. Только в этом случае он будет вызываться при возникновении соответствующего события.

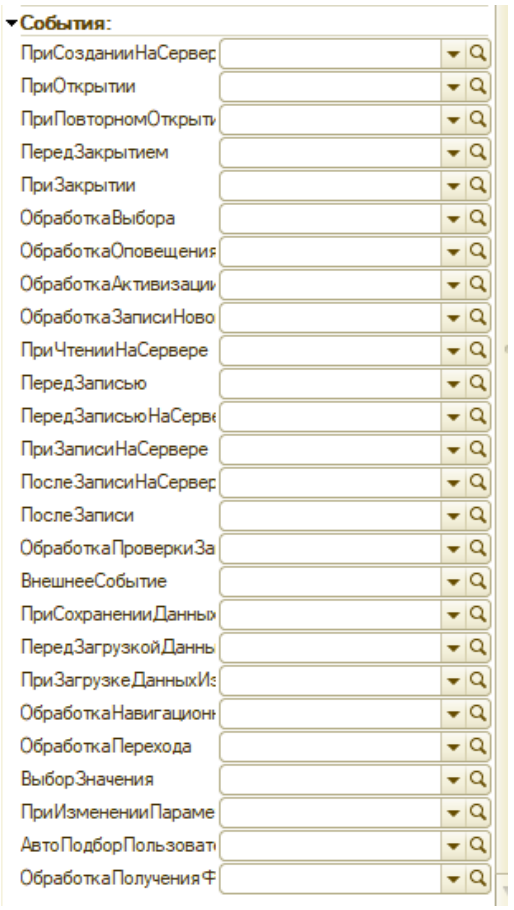
Назначаемый обработчик может иметь произвольное имя. Если имя процедуры совпадает с именем события формы или ее элемента, этого совсем недостаточно для вызова процедуры обработки события с таким именем. Требуется явное назначение процедуры обработчиком этого события в палитре свойств, в соответствующем событии.

Таким образом, любая процедура, расположенная в модуле формы, может быть назначена обработчиком любого события (или сразу нескольких событий) формы или ее элемента, расположенного в форме. Имя процедуры в этом случае не имеет значения. Важно лишь то, что она назначена обрабатывать какое-либо событие.

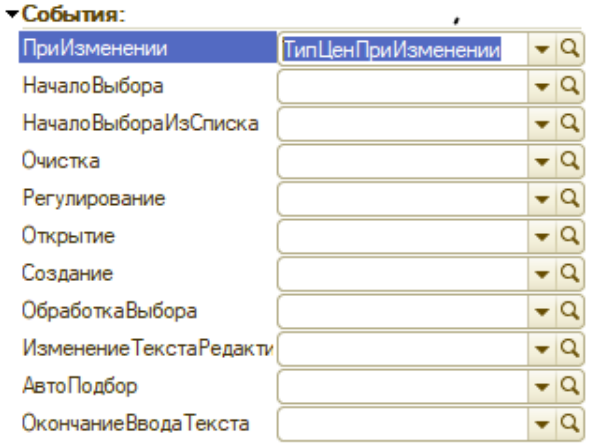
Назначение обработчика может выполняться интерактивно, при работе с формой в конфигураторе, или программно, используя методы формы и ее элементов – УстановитьДействие().

Если процедура-обработчик события относится к форме или элементу управления, то ее обязательно нужно указывать в палитре свойств для формы или элемента управления.

Ниже показана палитра свойств для формы элемента справочника «Номенклатура» с несколькими назначенными обработчиками событий:



Обратите внимание на важный момент, **имя процедуры-обработчика событий может не совпадать с именем события**. Для элементов управления чаще всего так и бывает, например, процедура "ТипЦенПриИзменении" обрабатывает событие "ПриИзменении" поля ввода для реквизита "ТипЦен", как показано на следующем рисунке:



Как правило, процедура-обработчик имеет тот же набор параметров, что и событие. Если у нее нет соответствующих параметров, то обработка события может получиться неполной. Поэтому рекомендуется создавать процедуры-обработчики конструктором через палитру свойств, нажимая кнопку с лупой или выбирая процедуру из выпадающего списка.

Есть еще одна интересная возможность: одна и та же процедура может "обслуживать" несколько событий формы или элементов управления, в том числе от разных источников. Элемент управления, который инициировал событие, передается в качестве первого параметра в эту процедуру-обработчик (параметр "Элемент"), и при

необходимости алгоритм может проанализировать, откуда пришло событие, и выполнить соответствующие действия.

Процедуры-обработчики событий, расположенные в модуле приложения, модуле внешнего соединения, модуле прикладного объекта должны называться точно так, как называются соответствующие события.

1. Процедуры-обработчики событий, расположенные в модуле приложения или модуле внешнего соединения, совпадают с именами событий:

- ПередНачаломРаботыСистемы
- ПриНачалеРаботыСистемы
- ПриЗавершенииРаботыСистемы
- ПередЗавершениемРаботыСистемы
- ОбработкаВнешнегоСобытия

2. Имена процедур-обработчиков событий, расположенных в модуле объекта, тоже строго соответствуют именам событий:

для модуля документа (события объекта типа "ДокументОбъект")

- ПередЗаписью
- ПриЗаписи
- ПриУдалении
- ПриКопировании
- ОбработкаЗаполнения (для обработки "ввода на основании")
- ОбработкаПроведения
- ОбработкаУдаленияПроведения
- ПриУстановкеНовогоНомера

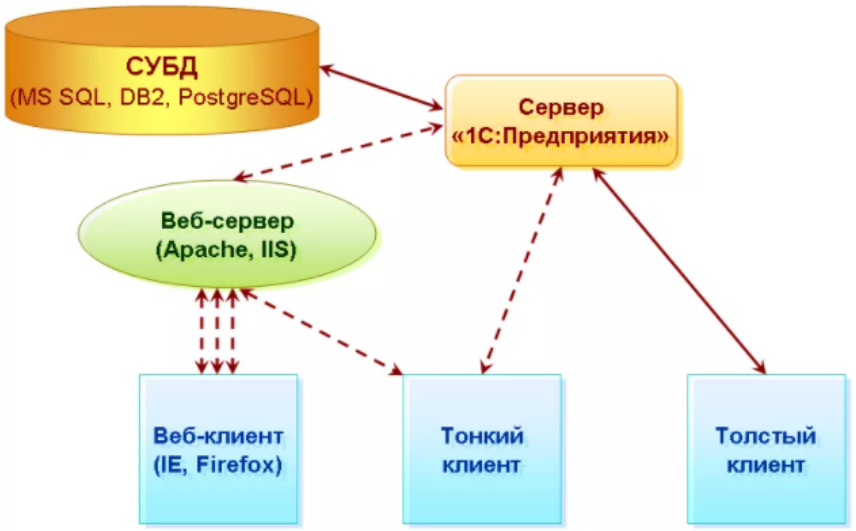
Аналогичные обработчики событий могут располагаться в модуле справочника и модулях других прикладных объектов.

3. Есть также модуль набора записей для всех видов регистров, который подобен модулям прикладных объектов. Модуль набора записей может содержать следующие процедуры-обработчики событий (имена процедур должны совпадать с именами событий):

- ПередЗаписью
- ПриЗаписи

8. Клиентская и серверная части приложения. Директивы компиляции

Серверу и клиенту доступны разные объекты, а также их свойства и методы встроенного языка. Сервер выполняет действия, связанные с доступом к данным БД и их обработкой, а клиент выполняет отображение этих данных и взаимодействие с пользователем. Таким образом, клиент-серверное взаимодействие определяет ряд правил при разработке кода. Структура кода определяется не логикой решаемой задачи, а логикой клиент-серверного взаимодействия. Клиентский код пишется как сценарий передачи управления с клиента на сервер и обратно, а не как последовательность действий к исполнению.



Для организации такого сценария предназначены *директивы компиляции*:

- &НаСервере
- &НаКлиенте
- &НаСервереБезКонтекста
- &НаКлиентеНаСервереБезКонтекста

Код, реализующий бизнес-логику, должен быть отделен от кода, реализующего интерфейс. Нужно понимать, что форма существует одновременно и на клиенте, и на сервере, но каждый из них «видит» только свою часть со всеми вытекающими последствиями. По этой причине все процедуры и функции, создаваемые в модуле формы, должны иметь явное указание на то, в каком контексте они будут исполняться.

Если директива компиляции перед описанием процедуры или функции отсутствует, то считается, что данная процедура или функция исполняется на сервере *(&НаСервере*).

Процедуры или функции, исполняемые в контексте клиента, называют *клиентскими*, соответственно исполняемые в контексте серверной части – *серверными*. Из клиентских процедур и функций можно вызывать серверные, когда они будут выполнены, исполнение кода вернется на клиент. Следует отметить, что принудительно вернуть исполнение кода на клиент не получится, т.е. из серверных процедур и функций вызвать клиентские – нельзя.

&НаСервере. Директива указывает на то, что процедура или функция исполняется в контексте серверного приложения. Её используют для всех обработчиков серверных событий формы, а также для собственных процедур и функций, которые определит разработчик, чтобы передать выполнение кода на сервер. Такие процедуры и функции упрощенно называют серверными процедурами формы. Разработчик должен управлять частотой вызовов сервера. В идеале, нужно стремиться к тому, чтобы их количество было минимально, т.к. при каждом вызове система пересылает туда-сюда данные.

&НаКлиенте. Директива указывает на то, что процедура или функция исполняется в контексте клиентского приложения и ей будет доступен весь контекст формы – реквизиты, элементы и параметры формы. Но к элементам базы данных обратиться не получится.

&НаСервереБезКонтекста. Директива указывает на то, что код будет исполняться на сервере, при этом контекст формы будет недоступен. А значит получить доступ к реквизитам и элементам формы из процедуры или функции, выполняемой с данной директивой – не получится. Но, с другой стороны, вызов такой процедуры или функции существенно «легче» по объему передаваемых данных на сервер. Для тонкого клиента или веб-клиента это играет важную роль. Если реквизиты и элементы формы не потребуются для исполнения кода на сервере, то вызов процедуры или функции, скомпилированной с директивой *&НаСервереБезКонтекста* – это верный выбор.

&НаКлиентеНаСервереБезКонтекста. Данная директива определяет, что процедура или функция может исполняться как в контексте клиента, так и в контексте сервера. Когда такое может понадобиться при разработке? Скорее всего тогда, когда нужно выполнить одинаковые действия в обеих частях приложения. Чтобы не создавать две одинаковые процедуры с разными директивами – можно сделать одну, с директивой *&НаКлиентеНаСервереБезКонтекста.*

Директива	Данные формы	База данных
&НаКлиенте	+	–
&НаСервере	+	+
&НаСервереБезКонтекста	–	+
&НаКлиентеНаСервереБезКонтекста	–	–

Задание на лабораторную работу

Разработка конфигурации для регистрации непериодических и периодических данных. Задача 1 - организация необходимых справочников. Задача 2 - организация непериодического регистра сведений. Задачи 3 и 4 - организация периодических регистров сведений. В задаче 3 регистр независимый, а в задаче 4 регистр заполняется при проведении документа. При этом в документе может содержаться табличная часть для отражения информации сразу о нескольких событиях, что приведет к формированию нескольких записей регистра одним документом. Структура справочников, документа и регистров выбираются самостоятельно.

Вариант 1

Постановка: Заказчик просит разработать информационную систему для автоматизации работы автошколы. В частности, конфигурация должна позволять хранить:

1. Список учебных автомобилей, список преподавателей, список учебных групп.
2. Для каждого преподавателя содержать информацию о закрепленном за ним автомобиле.
3. Для каждой группы содержать информацию о стоимости обучения на начало квартала и длительности обучения в днях.
4. Информацию о часовой ставке преподавателя на момент изменения/установки ставки должен записывать документ УстановкаСтавки при своем проведении.

Вариант 2

Постановка: Заказчик просит разработать информационную систему для автоматизации работы университета. В частности, конфигурация должна позволять хранить:

1. Список предметов, список преподавателей, список учебных групп.
2. Для заданных предметов и групп содержать информацию о ведущем преподавателе.
3. Для каждой группы содержать информацию о стоимости годового обучения на начало каждого учебного года.
4. Информацию о часовой ставке преподавателя на момент изменения/установки ставки должен записывать документ УстановкаСтавкок при своем проведении.

Вариант 3

Постановка: Заказчик просит разработать информационную систему для автоматизации работы гостиницы. В частности, конфигурация должна позволять хранить:

1. Список гостиничных номеров, список горничных, список гостей.
2. Для каждого гостя содержать информацию о заказанном номере.
3. Для каждого номера содержать информацию о его стоимости на начало каждого месяца.
4. Информацию на каждый день о том, в каком номере какая горничная производит уборку, должен записывать документ НазначениеСотрудников при своем проведении.

Вариант 4

Постановка: Заказчик просит разработать информационную систему для автоматизации работы салона красоты. В частности, конфигурация должна позволять хранить:

1. Список специалистов, список услуг, список клиентов.

2. Для каждого специалиста и каждой услуги содержать информацию об ожидаемом времени выполнения услуги.
3. Для каждого специалиста на каждый день содержать информацию о времени начала и окончания работы.
4. Информацию на произвольную дату и время о том, какой клиент к какому специалисту записан, должен записывать документ ЗаписьНаПрием при своем проведении.

Вариант 5

Постановка: Заказчик просит разработать информационную систему для автоматизации работы туристического агентства. В частности, конфигурация должна позволять хранить:

1. Список туров, список сотрудников, список клиентов.
2. Для каждого тура содержать информацию о его длительности в днях.
3. Для каждого тура содержать информацию о его стоимости на начало каждого месяца.
4. Информацию на каждый день о том, какому клиенту какой тур оформлен и каким сотрудником, должен записывать документ ОформлениеТуров при своем проведении.

Вариант 6

Постановка: Заказчик просит разработать информационную систему для автоматизации работы школьной библиотеке. В частности, конфигурация должна позволять хранить:

1. Список книг, список издательств, список читателей.
2. Для каждой книги содержать информацию об авторе, издательстве, количестве страниц.
3. Для каждой книги на каждый день содержать информацию о том, какой читатель взял книгу.
4. Информацию на каждый день о том, какова остаточная стоимость каждой книги, должен записывать документ ОценкаСтоимостиКниг при своем проведении.

Вариант 7

Постановка: Заказчик просит разработать информационную систему для автоматизации работы системы обеспечения дошкольного питания. В частности, конфигурация должна позволять хранить:

1. Список блюд, список частей дня, список дошкольных групп.
2. Для каждого сочетания блюда и части дня содержать информацию о возможности использовании блюда в этой части дня.
3. Для каждого блюда содержать информацию о его стоимости на начало каждого дня.
4. Информацию на каждый день для каждой группы о том, в какой части дня какое блюдо и под каким номером готовится, должен записывать документ ПланПитания при своем проведении.

Вариант 8

Постановка: Заказчик просит разработать информационную систему для автоматизации работы торгового предприятия. В частности, конфигурация должна позволять хранить:

1. Список товаров, список сотрудников, список покупателей.

2. Для каждого покупателя содержать информацию о его юридическом адресе и директоре.
3. Для каждого товара содержать информацию о его цене продажи на начало каждого дня.
4. Информацию на каждый день о том, какому покупателю какой товар продан с указанием количества и стоимости этого количества товара, должен записывать документ РеализацияТоваров при своем проведении.

Вариант 9

Постановка: Заказчик просит разработать информационную систему для автоматизации работы транспортного предприятия. В частности, конфигурация должна позволять хранить:

1. Список транспортных средств, список услуг, список клиентов.
2. Для каждого транспортного средства содержать информацию о его стране производителе, и вместимости фургона в кубических метрах.
3. Для каждой услуги и транспортного средства содержать информацию о стоимости услуги на начало каждого месяца.
4. Информацию на каждый день о том, какому клиенту какая услуга оказана и какое транспортное средство использовалось, должен записывать документ ОказаниеУслуг при своем проведении.

Вариант 10

Постановка: Заказчик просит разработать информационную систему для автоматизации работы Интернет-провайдера. В частности, конфигурация должна позволять хранить:

1. Список тарифов, список услуг, список клиентов.
2. Для каждого сочетания тарифа и услуги содержать информацию о возможности такой комбинации.
3. Для каждого сочетания тарифа и услуги содержать информацию о его стоимости на начало каждого месяца.
4. Информацию на каждый день о том, какому клиенту какая услуга была заблокирована и на сколько дней, должен записывать документ БлокировкаУслуг при своем проведении.

Вариант 11

Постановка: Заказчик просит разработать информационную систему для автоматизации расчета оплаты труда на предприятии. В частности, конфигурация должна позволять хранить:

1. Список видов начислений (зарплата, премия и пр.), список сотрудников, список способов начисления (почасовая, окладная и пр.).
2. Для каждого сочетания сотрудника и вида начисления содержать информацию о способе начисления.
3. Для каждого сотрудника содержать информацию о его окладе или часовой ставке на начало каждого месяца.
4. Информацию на каждый месяц о том, какому сотруднику какой вид начисления применен и на какую сумму, должен записывать документ НачислениеЗарплаты при своем проведении.

Вариант 12

Постановка: Заказчик просит разработать информационную систему для автоматизации работы жилищно-коммунального предприятия. В частности, конфигурация должна позволять хранить:

1. Список сотрудников, список услуг, список домохозяйств.
2. Для каждого домохозяйства содержать информацию о его площади.
3. Для каждой услуги содержать информацию о ее стоимости на начало каждого месяца.
4. Информацию на каждый месяц о том, за каким домохозяйством для каждой услуги закреплен какой специалист, должен записывать документ ОказаниеУслуг при своем проведении.

Вариант 13

Постановка: Заказчик просит разработать информационную систему для автоматизации работы мобильного оператора. В частности, конфигурация должна позволять хранить:

1. Список тарифов, список услуг, список клиентов.
2. Для каждого клиента содержать информацию об основном тарифе.
3. Для каждого тарифа и услуги содержать информацию об их стоимости на начало каждого месяца.
4. Информацию на каждый месяц о том, какому клиенту какая услуга предоставлена и по какому тарифу, должен записывать документ ОказаниеУслуг при своем проведении.

Вариант 14

Постановка: Заказчик просит разработать информационную систему для автоматизации управления прайс-листами. В частности, конфигурация должна позволять хранить:

1. Список товаров, список прайс-листов, список клиентов.
2. Для каждого прайс-листа содержать информацию о количестве позиций в его составе и дате утверждения.
3. Для каждого прайс-листа и товара содержать информацию о цене товара на начало каждого дня.
4. Информацию на каждый день о том, какому клиенту по какому прайс-листу производилась отгрузка товара, должен записывать документ РеализацияТоваров при своем проведении.

Вариант 15

Постановка: Заказчик просит разработать информационную систему для автоматизации работы подсистемы товарооборота предприятия. В частности, конфигурация должна позволять хранить:

1. Список товаров, список сотрудников, список клиентов.
2. Для каждого сотрудника содержать информацию о начале его работы на предприятии и его паспортных данных.
3. Для каждого товара и клиента содержать информацию о стоимости товара для данного клиента на начало каждого месяца.
4. Информацию на каждый день о том, какому клиенту какой товар продан и каким сотрудником с указанием суммы продажи, должен записывать документ РеализацияТоваров при своем проведении.

Содержание отчета

1. Цель работы.
2. Описание варианта задания.
3. Пошаговое описание процесса выполнения варианта задания.
4. Выводы

Контрольные вопросы

1. Перечислите объекты конфигурации, которые относятся к табличным типам данных.
2. Что представляют собой табличные данные с точки зрения «1С:Предприятие»?
3. В чем отличие объектных данных от табличных?
5. Что такое движения регистра, и что такое регистратор?
6. Все ли табличные записи относятся к какому-либо регистратору?
7. Для чего используются наборы записей?
8. Опишите процесс добавления новой записи табличных данных к существующим.
9. Опишите процесс замещения существующей записи табличных данных.
10. Почему следует использовать регистры, хотя необходимая информация содержится в других объектах?
11. Для чего нужны измерения регистра, ресурсы и реквизиты?
12. Как создать движения документа с помощью конструктора движений?
13. Как средствами встроенного языка обойти табличную часть документа и обратиться к ее данным?
14. Как показать команды открытия списка регистра в интерфейсе конфигурации и в интерфейсе формы?
15. Для чего предназначен объект конфигурации «Регистр сведений»?
16. Какими особенностями обладает объект конфигурации «Регистр сведений»?
17. Какие поля определяют ключ уникальности регистра сведений?
18. Что такое периодический регистр сведений
19. Что такое независимый регистр сведений?
20. Как создать периодический регистр сведений?
21. Что такое ведущее измерение регистра?