

Учебный курс

Подготовка к аттестации 1С:Специалист по платформе 1С:Предприятие 8.3

Общие, универсальные механизмы,
которые используются для решения
задач всех разделов

Главы 1 – 10

Версия 1.1.1

Оглавление

Введение	5
1. Какие настройки проведения документов рекомендуется использовать на аттестации	9
Свойство «Проведение»	10
Свойство «Оперативное проведение»	11
Свойство «Удаление движений»	14
Свойство «Запись движений при проведении»	16
Значения свойств документов, рекомендуемые для использования на экзамене	18
2. Формирование движений документа по регистрам	21
3. Что такое момент времени и для чего он используется	26
4. Как быстро получить движения по регистрам из управляемой формы – без единой строчки кода	30
Введение	30
Как перейти к движениям из формы документа?	31
Как перейти к движениям из формы списка?	32
5. Как обеспечить возможность запуска обычных форм в режиме управляемого приложения (используем Консоль запросов из каркасной конфигурации)	35
Как найти ошибку в тексте запроса	41
Итоги	44
6. Как реализовать поступление товаров в компанию	45
Обработка проведения для документа «Приходная накладная»	48
7. Старая и новая методики контроля остатков – их преимущества, недостатки и рекомендации, какую методику выбрать в решении	52
Старая методика контроля остатков	52
Новая методика контроля остатков	53
8. Как реализовать контроль остатков по новой методике	55
Обработка проведения для документа «Приходная накладная»	58
Обработка проведения для документа «Расходная накладная»	58
Установка маркера Записи у регистра (п.1)	60
Инициализация менеджера временных таблиц (п.2)	61
Запрос (п.3)	61
Загрузка таблицы значений в коллекцию движений документа (п.4)	64

Управляемая блокировка данных регистра (п.5)	64
Запись коллекции движений (п.6)	64
Запрос на получение отрицательных остатков (п.7)	65
Определение момента времени для получения остатков (п.8)	67
Если запрос не пустой, появились отрицательные остатки (п.9)	67
Выдача сообщений пользователю о наличии отрицательных остатков (п.10)	67
Есть ошибки, выходим из процедуры (п.11)	68
9. Как выполнить контроль остатков по старой методике	69
Обработка проведения для документа «Приходная накладная»	72
Обработка проведения для документа «Расходная накладная»	72
Подготовка наборов записей регистра (п.1)	75
Восстановление для свойства «Записывать» набора движений значения «Истина» (п.2)	76
Управляемая блокировка данных регистра (п. 3)	77
Запрос (п. 4)	77
Определение момента времени для получения остатков (п.5)	81
Цикл по номенклатуре документа (п.6)	82
Контроль остатков (п.7)	82
Выдача сообщений пользователю о наличии отрицательных остатков (п.8)	82
Расчет суммы для списания (п.9)	82
Формирование движения (п.10)	83
10. Что такое актуальные остатки и возможные проблемы при их использовании	84
Что такое актуальные остатки	84
Таблица движений	84
Таблица итогов	85
Управление итогами	86
Как получить актуальные остатки	87
Преимущества использования актуальных остатков	88
Недостатки использования актуальных остатков	88
Ввод документов будущей датой	88
Добавление данных в регистр вручную	89
Проведение документа программно	90
Подведение итогов	92

Введение

Данный курс предназначен для подготовки слушателей, решивших сдавать экзамен на сертификат 1С:Специалист по платформе «1С:Предприятие 8.3».

Чтобы конкретнее представить поставленную перед собой цель, вначале кратко познакомимся с самой процедурой сдачи экзамена (подробное описание доступно для скачивания на сайте фирмы 1С: <http://1c.ru/spec/questions.htm>).

В экзаменационном билете задачи скомпонованы так, чтобы проверить знания и умения программирования и конфигурирования в рамках разделов:

- оперативного учета и управления
- бухгалтерского учета
- периодических расчетов
- бизнес-процессов и задач
- управляемых форм.

В случае отсутствия решения по любому из разделов (кроме бизнес-процессов и управляемых форм) экзамен считается несданным.

По условию сертификации требуется использовать каркасную конфигурацию.

Каркасная конфигурация – это некоторая изначальная конфигурация, которая заполнена объектами метаданных с определенными настройками и данными.

Объекты, которые уже присутствуют в каркасной конфигурации, позволяют экономить время. Выделим некоторые из них:

- Обработка *Консоль запросов*, которая предназначена для отладки и просмотра результатов выполнения запросов в режиме «1С:Предприятие»
- Обработка заполнения календарей (графиков работы), которая потребуется для решения расчетных задач
- Большое количество созданных элементов справочников, которые существенно экономят время при решении каких-либо задач.

Однако нельзя с уверенностью гарантировать, что на экзамене будет предложена та самая конфигурация, которую можно скачать с сайта: <http://1c.ru/spec/questions.htm>. Поэтому, используя уже имеющийся в конфигурации объект, необходимо проверить, соответствуют ли его настройки

условиям, поставленным в задаче: нужно ли дополнять структуру объекта, удовлетворяют ли условиям задачи установленные типы данных.

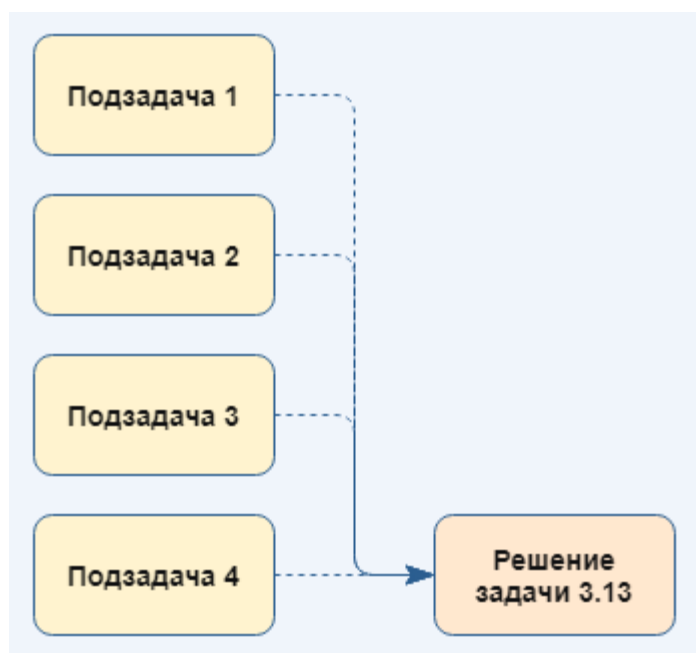
Рассмотрение примеров курса рассчитано на использование каркасной конфигурации, чтобы в процессе подготовки слушатель получил возможность сразу познакомиться с содержимым этой конфигурации и на экзамене не тратил время на дублирование объектов метаданных.

Данный курс рассчитан на слушателей, которые изучили материалы в объеме курса «[1С:Программист - Быстрый старт в профессию!](#)».

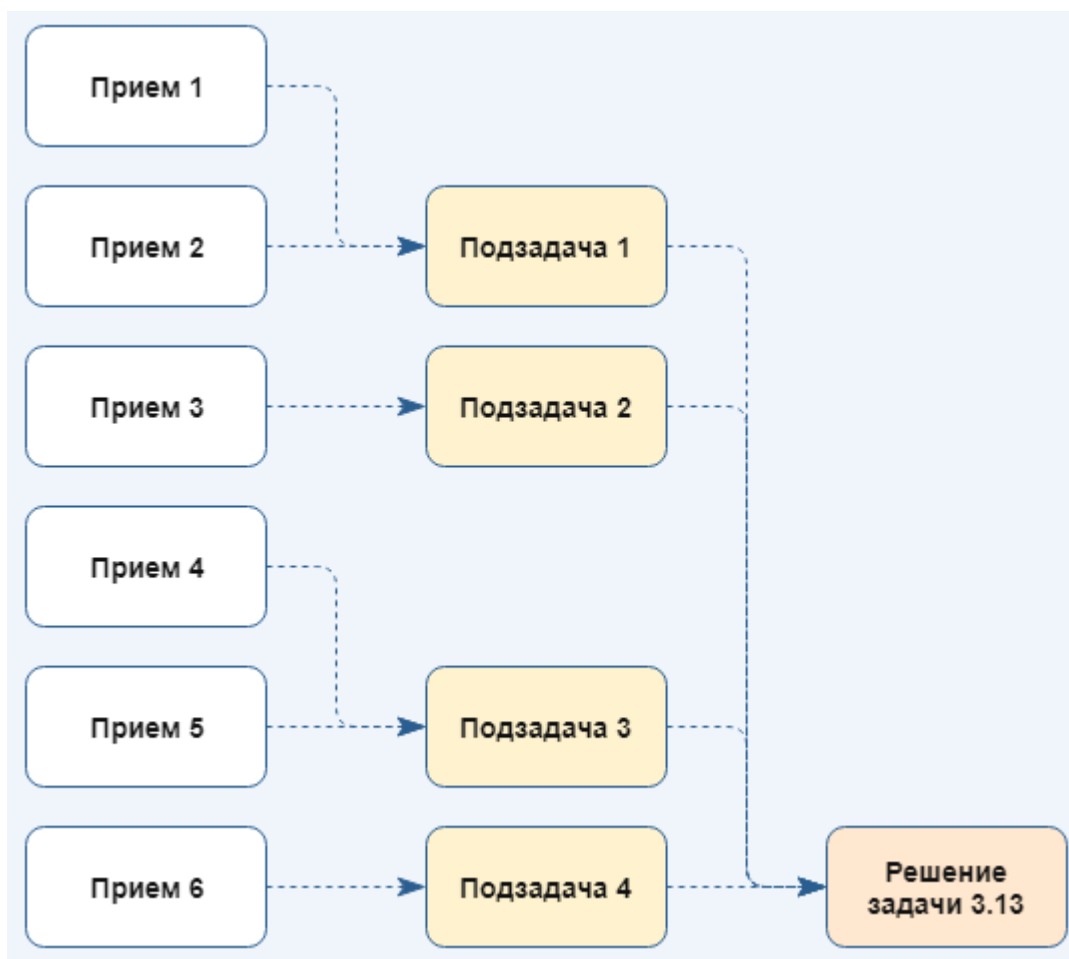
В курсе не рассматривается решение каждой экзаменационной задачи от начала до конца.

Рассмотрение каждой задачи от начала до конца потребовало бы многократного дублирования по сути одного и того же материала, что значительно увеличивает срок изучения материалов курса. Еще один недостаток такого подхода – привязанность к конкретным задачам и конкретным решениям. При изменении формулировки задачи у экзаменуемого могут возникнуть проблемы с ее решением.

В процессе подготовки данного курса решение всех задач, предлагаемых на экзамене, было разбито на самостоятельные этапы (подзадачи). Для наглядности на схеме показан один из примеров разбиения решения задачи:



Кроме того, для решения каждой подзадачи может использоваться один или несколько приемов:



Так был получен набор решений небольших подзадач, как некий конструктор, из которого можно собрать решение любой задачи. При этом решение подзадач предполагает многовариантность.

Таким образом, решение любой аттестационной задачи можно собрать из решений соответствующих подзадач. После изучения решений всех подзадач решение аттестационной задачи сводится к тому, чтобы на основании предложенных условий определить, из каких этапов она состоит.

Для удобства использования полученных решений подзадач была составлена специальная схема разделов курса, из которой становится понятным, как эти подзадачи взаимосвязаны (схема представлена на сайте и доступна для скачивания).

Например, в рамках первого раздела по общим и универсальным механизмам рассматривается решение подзадач:

- Контроль остатков номенклатуры по «новой» методике при проведении документов реализации с формированием движений по регистру оперативного учета
- Контроль остатков номенклатуры по «старой» методике при проведении документов реализации с формированием движений по регистру бухгалтерского учета
- Создание и настройка объектов конфигурации для хранения видов характеристик и их значений

- Настройки объектов метаданных для обеспечения использования дополнительных характеристик в отчетах и на управляемых формах.

Первый раздел курса является общим и предназначен для того, чтобы на его материалы можно было ссылаться при рассмотрении задач и подзадач из других разделов. Это общие механизмы, которые используются при решении задач оперативного, бухгалтерского учета и расчетных задач.

После изучения данного раздела слушатель должен уметь:

- Правильно использовать настройки проведения документов
- Реализовывать разные методики контроля отрицательных остатков при оперативном проведении документов, грамотно выбирать «старую» или «новую» методику в соответствии с условием поставленной задачи
- Использовать механизмы блокировок
- Реализовывать возможности хранения произвольных свойств с помощью специального объекта конфигурации *План видов характеристик*, а также хранение значений этих свойств в регистре сведений или в табличной части характеризуемого объекта
- Настраивать объекты метаданных для использования ими видов характеристик, определенных в *Плане видов характеристик*.

1. Какие настройки проведения документов рекомендуется использовать на аттестации

Для документов, отражающих разные хозяйственные операции, могут использоваться разные настройки проведения, обусловленные спецификой самого документа. Кроме того, вопросы проведения документов являются общими и при разборе задач из других разделов специально не рассматриваются. Это обязательные знания, необходимые для решения задач во всех разделах.

Данная тема выделена в отдельный блок, поскольку ее изучение позволит избежать некоторых ошибок, которые на аттестации считаются критичными.

В платформе «1С:Предприятие» **проведение** – это специальный термин, который можно применить только к объекту метаданных *Документ*. Элементы справочников или, например, планов видов характеристик нельзя провести, их можно записать в базу, пометить на удаление, удалить из базы. Эти объекты предназначены для хранения информации, не привязанной ко времени. А вот документы можно проводить. Это связано с тем, что именно документы используются для отражения хозяйственных операций в учетной системе.

Провести документ – это значит отразить данные документа в учете, то есть сформировать движения по регистрам. В регистрах хранится «переработанная», подготовленная информация с такой детализацией и в таких разрезах, чтобы ее удобно было использовать для вывода в отчеты. Именно на основании данных из регистров чаще всего формируются отчеты. Это связано в том числе и с вопросами производительности, потому что регистры специально оптимизированы на уровне платформы для хранения и быстрого получения больших объемов данных.

В конфигураторе свойства, относящиеся к настройкам проведения документа, расположены в окне редактирования документа на закладке *Движения*.

В верхней части закладки *Движения* необходимо указать значения свойств, определяющих характер его проведения:

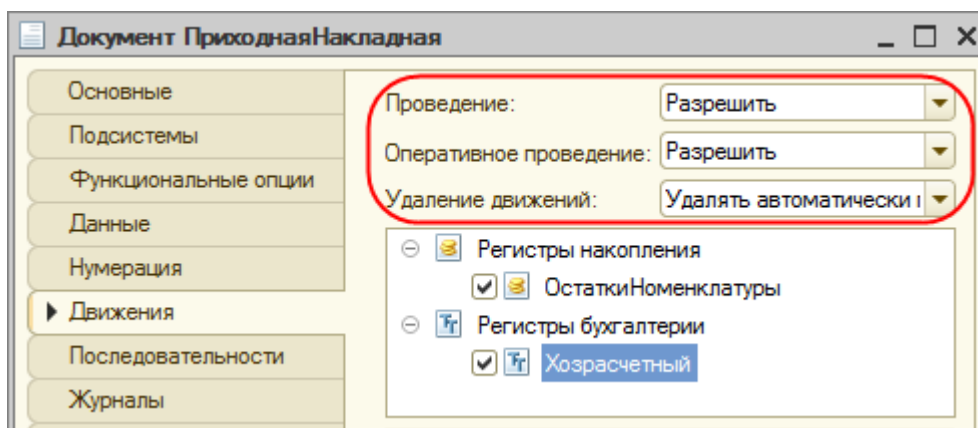


Рисунок 1 – Свойства документа, определяющие характер проведения

Далее подробно разберем, за что отвечает каждое свойство.

Свойство «Проведение»

Свойство документа *Проведение* может принимать одно из двух значений:

- «Разрешить»
- «Запретить».

Если свойство принимает значение «Разрешить», такой документ может проводиться, то есть формировать движения по регистрам, отражать данные документа в учете.

Если свойство принимает значение «Запретить», то такой документ проводиться не может. Не проводятся документы, которые не оказывают влияния на состояние учета в системе, или если с помощью документа пользователь непосредственно записывает движения в регистр. Например, в типовых конфигурациях можно встретить документ «Корректировка записей регистров», позволяющий вручную создавать движения по регистрам сведений и регистрам накопления. В документе «Бухгалтерская операция» пользователь выполняет ручное интерактивное редактирование движений по регистру бухгалтерии без выполнения операции проведения. Особенности реализации такого документа рассмотрены в разделе по бухучету.

Поскольку только документ может выступать регистратором для регистров, именно документы (а не, например, справочники) используются для непосредственной записи в регистры. Особенность документов, для которых запрещено проведение, заключается в том, что сами по себе они не хранят данные для формирования движений. Данные помещаются сразу в регистры.

Если разрешить проведение такого документа, то при отмене проведения созданные движения будут удалены. Т.к. в документе нет данных, по которым эти движения можно было бы сформировать

повторно, придется заново вручную создавать все движения по регистрам. Это лишняя и ненужная работа, поэтому если документ предназначен для интерактивного редактирования движений по регистрам, то проведение такого документа нужно запретить.

На аттестации указание в настройках документов типа «Бухгалтерская операция» разрешения на проведение является ошибкой.

Если свойство *Проведение* документа установлено в значение «Запретить», то свойства *Оперативное проведение* и *Удаление движений* становятся недоступными для редактирования:

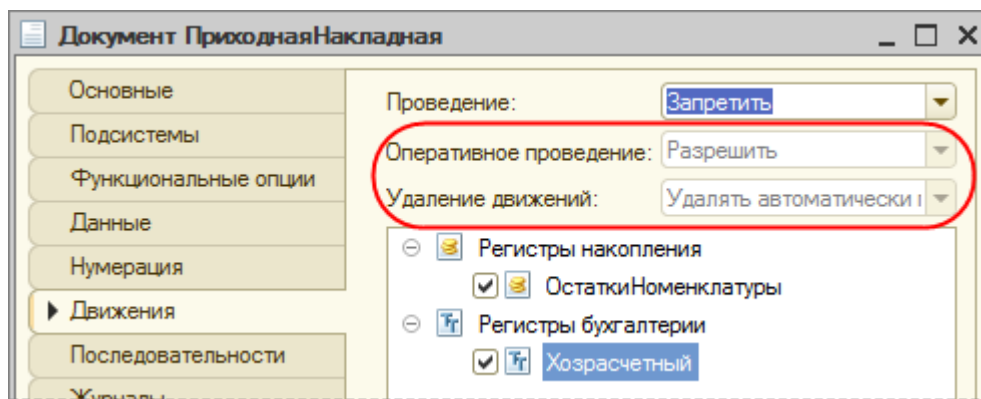


Рисунок 2 – Свойства, недоступные для документа с запрещенным проведением

Свойство «Оперативное проведение»

Свойство документа *Оперативное проведение* может принимать одно из двух значений:

- «Разрешить»
- «Запретить».

Оперативным называется проведение документа, происходящее в реальном времени, то есть регистрация хозяйственной операции здесь и сейчас.

Таким образом, можно определить один из двух сценариев проведения документа:

- **Оперативное проведение** – проведение в реальном времени. В этом случае можно выполнить дополнительные проверки (хватает ли товара на складе, не превышает ли задолженность покупателя допустимый порог и т.д.)
- **Неоперативное проведение** – отражение в учете уже свершившейся хозяйственной операции. В этом случае дополнительные проверки можно не проводить, поскольку необходимо только зафиксировать в базе событие, которое уже реально произошло. Однако при большом количестве пользователей бывают случаи преднамеренного или непреднамеренного

искажения информации. Поэтому клиент может пожелать, чтобы такие проверки выполнялись и при неоперативном проведении. При неоперативном проведении документов основные проверки могут проводиться и в типовых конфигурациях.

С технической точки зрения оперативное проведение имеет следующие особенности:

- Нельзя оперативно провести документы будущей датой
- Используется **механизм оперативной отметки времени**:
 - Документам с текущей датой присваивается текущее время
 - Если два документа проводятся одновременно, каждый будет иметь свое время (система разнесет документы по разным секундам).

В процедуре *ОбработкаПроведения* имеется параметр *РежимПроведения*. Значение этого параметра и определяет, как будет проводиться документ – оперативно или неоперативно. Используя этот параметр, можно реализовать различную логику проведения документа.

Процедура *ОбработкаПроведения*(Отказ, *РежимПроведения*)

Если *РежимПроведения* = *РежимПроведенияДокумента.Оперативный* Тогда

//Оперативное проведение

Иначе

//Неоперативное проведение

КонецЕсли;

КонецПроцедуры

В зависимости от режима проведения документа можно по-разному реализовать запрос на получение остатков по регистрам накопления (например, остатков товаров на складах).

Для документов, проводимых оперативно, в запросе можно не указывать дату, на которую требуется получить остатки. При этом будут получены актуальные остатки.

Запрос = Новый *Запрос*;

Запрос.Текст =

"ВЫБРАТЬ

| ОстаткиНоменклатурыОстатки.Номенклатура,

| ОстаткиНоменклатурыОстатки.КоличествоОстаток

| ИЗ

| РегистрНакопления.ОстаткиНоменклатуры.Остатки(, Номенклатура В (&СписокНоменклатуры))

КАК ОстаткиНоменклатурыОстатки";

Это важно с точки зрения производительности. Дело в том, что система всегда хранит в базе текущие (актуальные, самые последние) итоги на дату 01.11.3999. Если в запросе дата не указана, то получение

остатков происходит именно на эту дату. Поэтому такой запрос позволяет получить данные самым быстрым способом.

Если в запросе будет указана дата получения остатков, данные из базы будут получены более сложным способом:

Запрос = Новый Запрос;

Запрос.Текст =

```
"ВЫБРАТЬ
|      ОстаткиНоменклатурыОстатки.Номенклатура,
|      ОстаткиНоменклатурыОстатки.КоличествоОстаток
|ИЗ
|      РегистрНакопления.ОстаткиНоменклатуры.Остатки(&МоментВремени, Номенклатура В
(&СписокНоменклатуры)) КАК ОстаткиНоменклатурыОстатки";
```

В этом случае система получит остатки расчетным путем: сначала выберет данные из таблицы итогов, а затем скорректирует их при помощи данных из таблицы движений регистра. Это более сложный и, соответственно, менее производительный способ.

Если для документа разрешено оперативное проведение, а дата документа не совпадает с текущей датой (в систему вводятся данные «задним числом»), проведение будет происходить в неоперативном режиме.

Возникает вопрос: когда нет смысла использовать оперативное проведение документов?

- Если не требуется, чтобы проводимые документы последовательно располагались на временной оси, можно запретить оперативное проведение документа. В этом случае система не будет формировать для таких документов оперативные отметки времени.
- Если логика проведения по регистрам не отличается для оперативного и неоперативного режима, можно запретить оперативное проведение документа. Например, для формирования движений не требуется получать данные из регистров, используются только данные самого документа. Значит, не нужно будет реализовывать два различных алгоритма для оперативного (без указания даты, на которую получают остатки по регистру накопления) и неоперативного (с указанием даты, на которую получают остатки по регистру накопления) проведения. Чаще всего для таких документов не важно время документа, важна только дата – день, когда в базе должна быть зафиксирована операция. Следовательно, можно запретить оперативное проведение.
- Если планируется проводить документы будущей датой, то необходимо свойство документа *Оперативное проведение* установить в значение «Запретить».

Свойство «Удаление движений»

Свойство документа «Удаление движений» может принимать одно из трех значений:

- «Удалять автоматически при отмене проведения»
- «Удалять автоматически»
- «Не удалять автоматически».

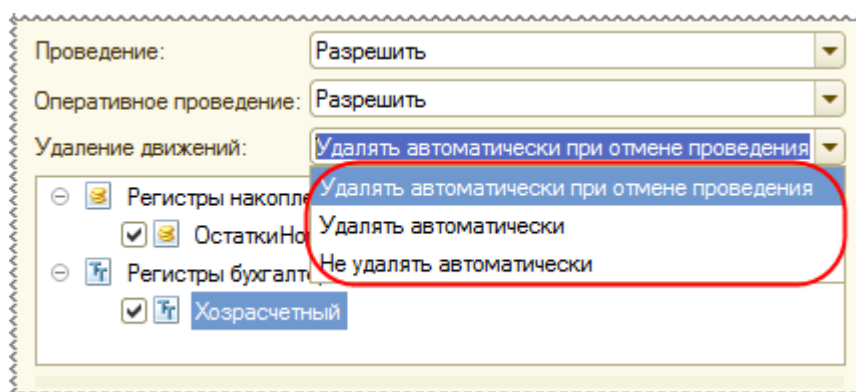


Рисунок 3 – Варианты значений свойства «Удаление движений»

Если используется значение «Удалять автоматически при отмене проведения», все движения документа по регистрам удаляются платформой только при удалении и отмене проведения документа. При перепроведении документа движения не удаляются, а перезаписываются.

Если используется значение «Удалять автоматически», платформа самостоятельно очищает движения по регистрам в следующих случаях:

- При повторном проведении уже проведенного документа (перепроведение)
- При пометке документа на удаление
- При отмене проведения документа.

Режим «Удалять автоматически» может вызывать проблемы при параллельной работе большого количества пользователей.

Если свойство документа *Удаление движений* установлено в значение «Удалять автоматически», при перепроведении документа существующие движения будут удалены. С этой целью платформа автоматически записывает в регистры пустые наборы записей. Запись начинается еще до выполнения события *ОбработкаПроведения* и тем самым сразу порождает открытие транзакции.

При удалении движений СУБД накладывает исключительную блокировку, которая действует в течение всей транзакции. Если проведение документа занимает достаточно продолжительное время,

такая блокировка может стать причиной долгого ожидания на блокировках других транзакций, выполняющихся параллельно.

Схематично время действия блокировки при режиме «Удалять автоматически» изображено на рисунке:

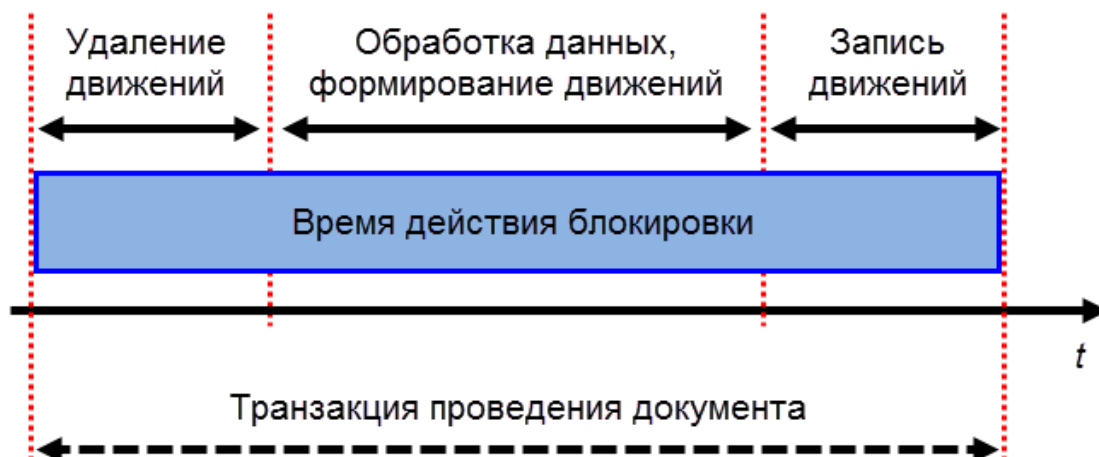


Рисунок 4 – Время действия блокировки при режиме «Удалять автоматически»

Если же свойство документа *Удаление движений* установлено в значение «Удалять автоматически при отмене проведения», то при перепроведении документа движения не будут удаляться. Значит, в таком случае нет необходимости в блокировке в самом начале транзакции. СУБД установит блокировку уже при записи движений, т.е. позже, ближе к концу транзакции.

Схематично время действия блокировки при режиме «Удалять автоматически при отмене проведения» изображено на рисунке:

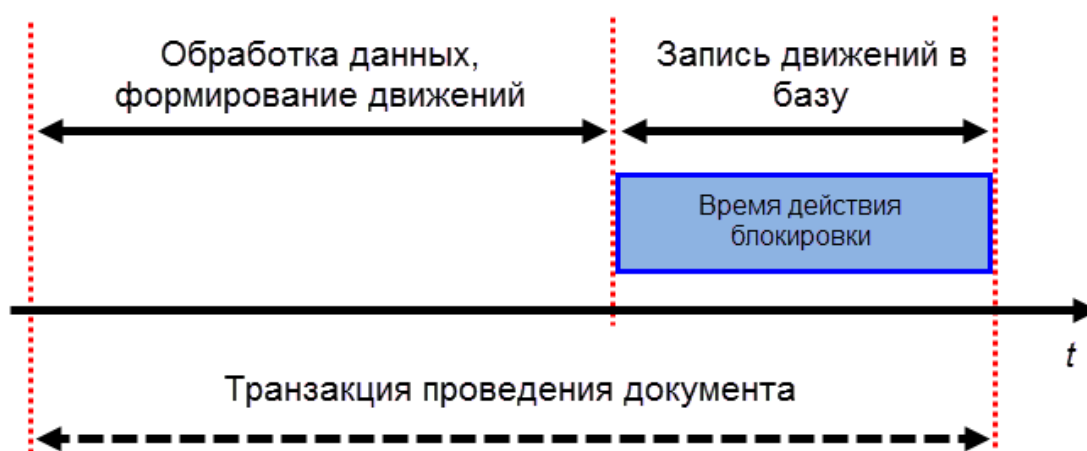


Рисунок 5 – Время действия блокировки при режиме «Удалять автоматически при отмене проведения»

Чем меньше время действия блокировки, тем меньше возможность возникновения проблем при параллельной работе пользователей.

Значение «Удалять автоматически при отмене проведения» впервые появилось в платформе 8.2. Добавлено оно было для оптимизации записи данных в базу и повышения производительности. Но для обеспечения совместимости с более старыми конфигурациями вариант «Удалять автоматически» был оставлен в платформе. Таким образом, разработчик может самостоятельно принимать решение, как именно должен вести себя каждый разрабатываемый документ.

Если для документа свойство *Удаление движений* принимает значение «Не удалять автоматически», то платформа не будет автоматически удалять движения документа по регистрам при перепроведении, пометке документа на удаление или при отмене проведения документа. При этом может возникнуть ситуация, когда у непроведенного документа имеются движения по регистрам.

Разработчику необходимо самостоятельно отслеживать все ситуации, где требуется очищать движения, и реализовывать соответствующий программный код. Например, можно в процедуре *ОбработкаУдаленияПроведения* описать логику, какие регистры следует очищать при отмене проведения, а движения каких регистров необходимо оставить даже у непроведенного документа.

Рассмотрим на практике, какие значения для свойства *Удаление движений* используются во флагманской конфигурации фирмы «1С» – «1С:ERP Управление предприятием 2.4» (релиз 2.4.1.172).

Всего в конфигурации 458 документов. Вариант «Удалять автоматически» используется только для 13 документов. Причем среди них такие специфические вспомогательные документы, как «ВыгрузкаРегламентированныхОтчетов», «ТранспортноеСообщение», «РегламентированныйОтчет» или «ДокументРасчетовСКонтрагентом». Для 233 документов используется режим «Не удалять автоматически», а для 212 – «Удалять автоматически при отмене проведения».

Таким образом, для большинства документов в 1С:ERP либо используется оптимизированный режим записи движений, либо разработчик самостоятельно отслеживает все ситуации, где необходима очистка наборов записей.

Свойство «Запись движений при проведении»

Еще одно важное свойство объекта метаданных *Документ*, которое не вынесено на закладку *Движения* окна редактирования объекта конфигурации. Эту настройку можно увидеть в палитре свойств, открываемой при помощи контекстного меню или сочетанием клавиш *Alt+Enter*:

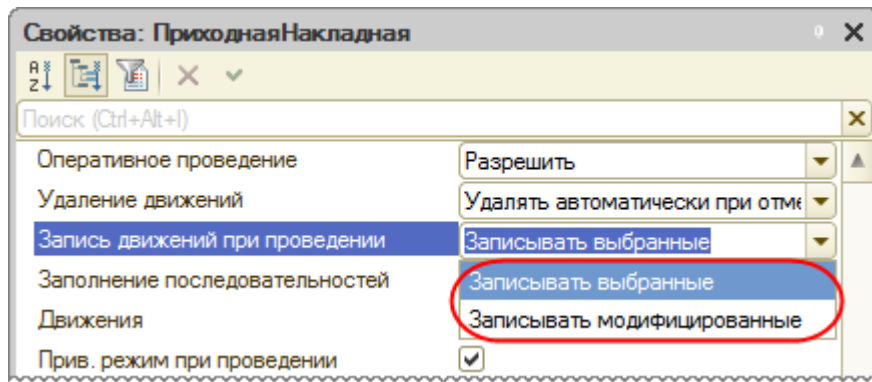


Рисунок 6 – Варианты значений свойства «Запись движений при проведении»

Свойство *Запись движений при проведении* может принимать два значения:

- «Записывать выбранные»
- «Записывать модифицированные».

Рассмотрим подробнее, как будет выполняться запись движений при разных значениях этого свойства.

1. Если для свойства установлено значение «Записывать выбранные», то разработчик в процедуре *ОбработкаПроведения* должен явно указать, какие движения следует записать в базу. Для этого нужно набору записей установить свойство *Записывать* в значение «Истина»:

```
Движения.ОстаткиНоменклатуры.Записывать = Истина;
```

Когда система выполнит весь программный код, расположенный в процедуре *ОбработкаПроведения*, те наборы записей, у которых свойство *Записывать* имеет значение «Истина», будут автоматически записаны в базу.

Можно не дожидаться окончания процедуры *ОбработкаПроведения*, записав уже сформированный набор записей в базу.

```
Движения.ОстаткиНоменклатуры.Записать();
```

После этого признак необходимости записи (свойство *Записывать*) автоматически будет установлен в значение «Ложь». То есть при окончании проведения документа уже записанные наборы записей не будут повторно писаться в базу. Это позволит уменьшить время проведения документа, сократить число блокировок, повысить параллельность работы пользователей, разумеется, если разработчик позаботился об оптимизации кода.

2. Если свойство *Запись движений при проведении* имеет значение «Записывать модифицированные», то после завершения процедуры *ОбработкаПроведения* платформа запишет в базу только модифицированные (измененные) наборы записей.

Проверить, был ли изменен набор записей, можно при помощи метода *Модифицированность*:

`Движения.ОстаткиНоменклатуры.Модифицированность();`

Когда система приступает к исполнению программного кода процедуры *ОбработкаПроведения*, признак модифицированности у всех движений установлен в значение «Ложь».

При программной работе с движениями признак модифицированности изменяется. Например, при очистке набора движений или при добавлении записи в набор движений модифицированность набора устанавливается в значение «Истина»:

`Движения.ОстаткиНоменклатуры.Очистить();`

`Движение = Движения.ОстаткиНоменклатуры.Добавить();`

После записи движений в базу модифицированность набора снова устанавливается в значение «Ложь»:

`Движения.ОстаткиНоменклатуры.Записать();`

Таким образом, свойство *Запись движений при проведении* позволяет разработчику управлять тем, какие движения будут записаны в базу при проведении документа. Если свойство принимает значение «Записывать выбранные», то необходимо явно указать, какие движения следует записать в базу. Если свойство принимает значение «Записывать модифицированные», то в базу будут записаны только измененные наборы записей.

Значения свойств документов, рекомендуемые для использования на экзамене

Чтобы на экзамене для создаваемых документов установить корректные значения рассмотренных выше свойств, требуется внимательно читать условие задачи.

Если в условии задачи сказано, что пользователь вводит данные непосредственно в движения документа по регистрам, то в настройках такого документа нужно установить для свойства *Проведение* значение «Запретить». Это означает, что такой документ не будет проводиться, для него не нужно реализовывать процедуру *ОбработкаПроведения* в модуле объекта. В качестве примера такого документа можно привести документ «Бухгалтерская операция». Нюансы создания такого документа рассматриваются в разделе, посвященном решению бухгалтерских задач.

В остальных случаях, если документ должен оказывать влияние на состояние учета в системе, то для свойства *Проведение* устанавливается значение «Разрешить».

Если в условии задачи сказано, что документы могут проводиться будущей датой, то свойство документа *Оперативное проведение* нужно установить в значение «Запретить». Также, если для проверки разработанного кода планируется проводить документы будущей датой, следует запретить оперативное проведение документа.

Если в условии задачи сказано, что документ должен проводиться, то в общем случае рекомендуется установить для свойства *Удаление движений* значение «Удалять автоматически при отмене проведения». Это связано с вопросами производительности. При перепроведении такого документа движения удаляться не будут, что уменьшит время действия блокировки при записи движений. Это оптимальный вариант в случае параллельной работы пользователей в базе.

Крайне не рекомендуется при решении аттестационных задач для свойства *Удаление движений* не использовать значение «Не удалять автоматически». Этот способ требует написания дополнительного программного кода для ручного удаления движений. Придется иметь в виду, что после отмены проведения такой документ может иметь движения по регистрам. Этот факт нужно будет учитывать при создании отчетов, он может оказывать влияние на логику проведения других документов.

В общем случае на экзамене не нужно создавать себе лишние сложности, которые требуют траты времени на написание дополнительного программного кода или на поиск потенциально возможных ошибок. Поэтому, если использование в настройках документа значения «Не удалять автоматически» для свойства *Удаление движений* логически не оправдано, то использовать его не следует.

Для свойства *Запись движений при проведении* рекомендуется использовать вариант «Записывать выбранные». В этом случае разработчик самостоятельно решает, какие регистры будут записаны в базу при завершении транзакции проведения. Для этого свойство *Записывать* набора записей необходимо установить в значение *Истина*. Это важно с точки зрения производительности. Если внутри процедуры *ОбработкаПроведения* движения по определенному регистру были записаны, то признак *Записывать* набора записей сбрасывается обратно в значение *Ложь*. Такой регистр не будет еще раз записываться в базу после того, как завершится транзакция проведения. Это позволит уменьшить время транзакции, избавиться от лишних блокировок, а следовательно, снизить вероятность возникновения проблем при параллельной работе большого количества пользователей.

Оформим рекомендуемые значения свойств в виде краткой таблицы:

Задача	Свойство	Значение свойства
Документ «Бухгалтерская операция» Документ предназначен для интерактивного редактирования движений по регистрам	Проведение	Запретить
Все остальные документы	Проведение	Разрешить

Документ может проводиться будущей датой	Оперативное проведение	Запретить
Документ должен проводиться	Удаление движений	Удалять автоматически при отмене проведения
Документ должен проводиться	Запись движений при проведении	Записывать выбранные

Рисунок 7 – Рекомендуемые к использованию на экзамене значения свойств документов

2. Формирование движений документа по регистрам

В верхней таблице на закладке *Движения* галочками необходимо отметить те регистры, для которых документ может являться регистратором.

Поскольку количество регистров в современных конфигурациях может быть очень большим, разработчику зачастую непросто работать со всем списком доступных регистров, часть из которых отмечена галочками. Полезно иметь отдельный перечень регистров, по которым документ может выполнять движения. Этот перечень отображается в нижней таблице на закладке *Движения*:

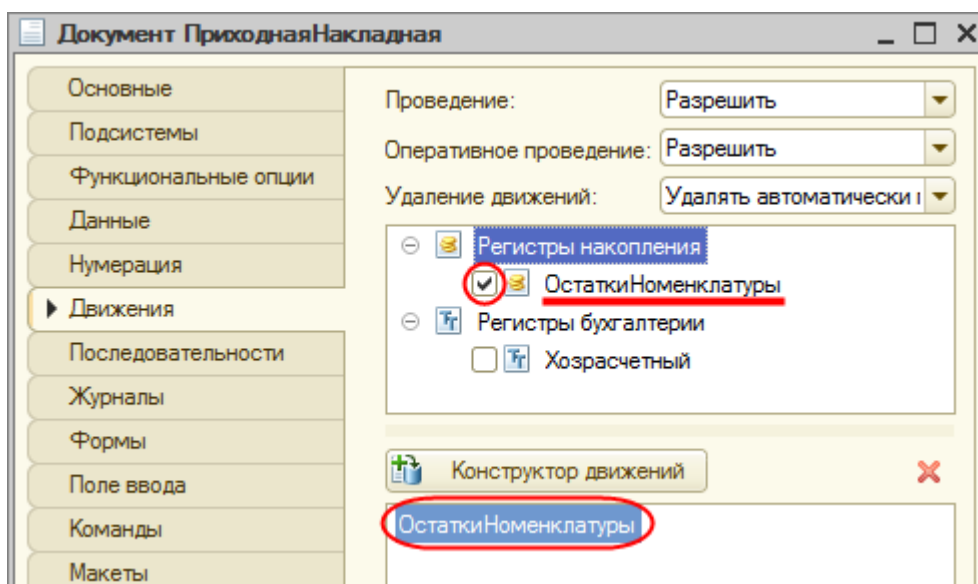


Рисунок 1 – Список регистров, по которым документ выполняет движения

Логика формирования движений документа по регистрам описывается в модуле объекта документа. Для этого предназначена процедура *ОбработкаПроведения*. Текст этой процедуры можно написать вручную, а можно для ускорения разработки сформировать заготовку этой процедуры при помощи конструктора движений, который открывается при нажатии кнопки *Конструктор движений*:

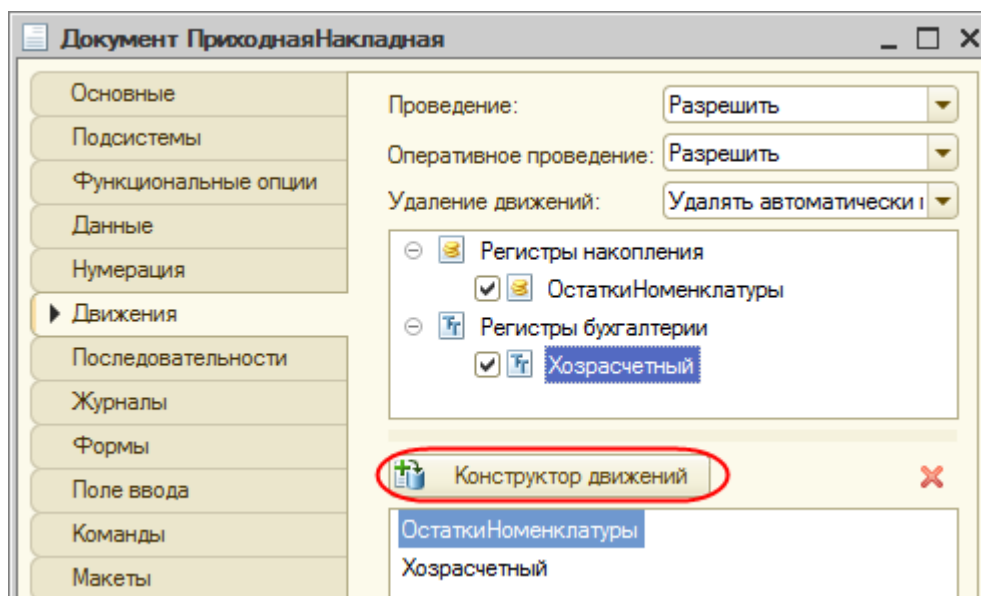


Рисунок 2 – Конструктор движений

В открывшемся окне конструктора в левой верхней таблице *Регистры* можно определить перечень регистров, по которым конструктор будет формировать движения:

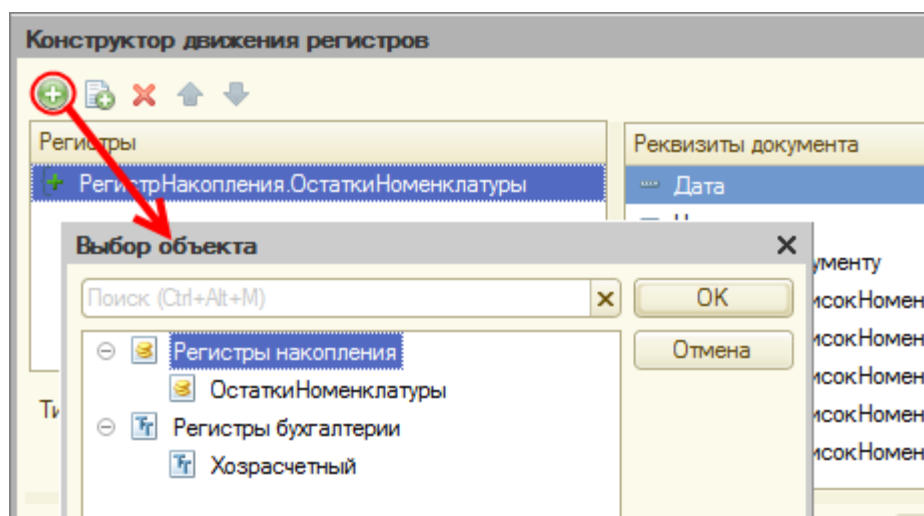


Рисунок 3 – Добавление регистров для формирования движений

Поскольку документ может формировать движения по различным регистрам, в таблице *Регистры* может быть несколько строк. Чтобы настроить логику формирования движений по конкретному регистру, необходимо выделить строку с тем регистром, для которого выполняется настройка.

Для регистров накопления остатков необходимо указать тип движения: *Приход* или *Расход*. Если движения должны формироваться по данным табличной части документа, то необходимо выбрать соответствующую табличную часть.

Для каждого выбранного регистра необходимо настроить, каким образом при проведении будут заполняться его поля. Выражения для полей можно указывать вручную. Для ускорения разработки можно воспользоваться кнопкой *Заполнить выражения*. Тогда система на основании совпадений типов хранимых данных, имен полей документа и регистра заполнит выражения автоматически (в случае неоднозначности выражение останется пустым, но реквизиты, подходящие для заполнения, будут помечены галочкой):

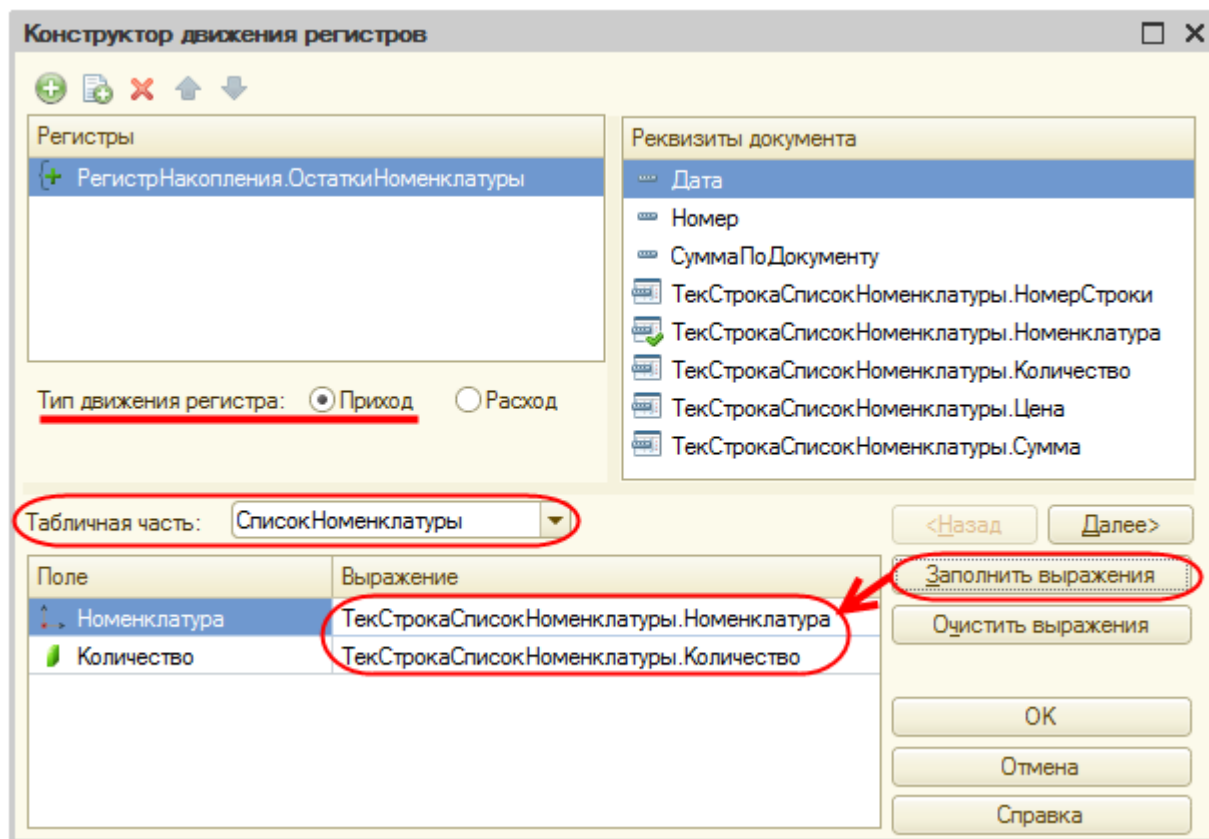


Рисунок 4 – Заполнение выражений для полей

После нажатия кнопки *OK* конструктор движений заполнит процедуру *ОбработкаПроведения* в модуле объекта и откроет этот модуль:

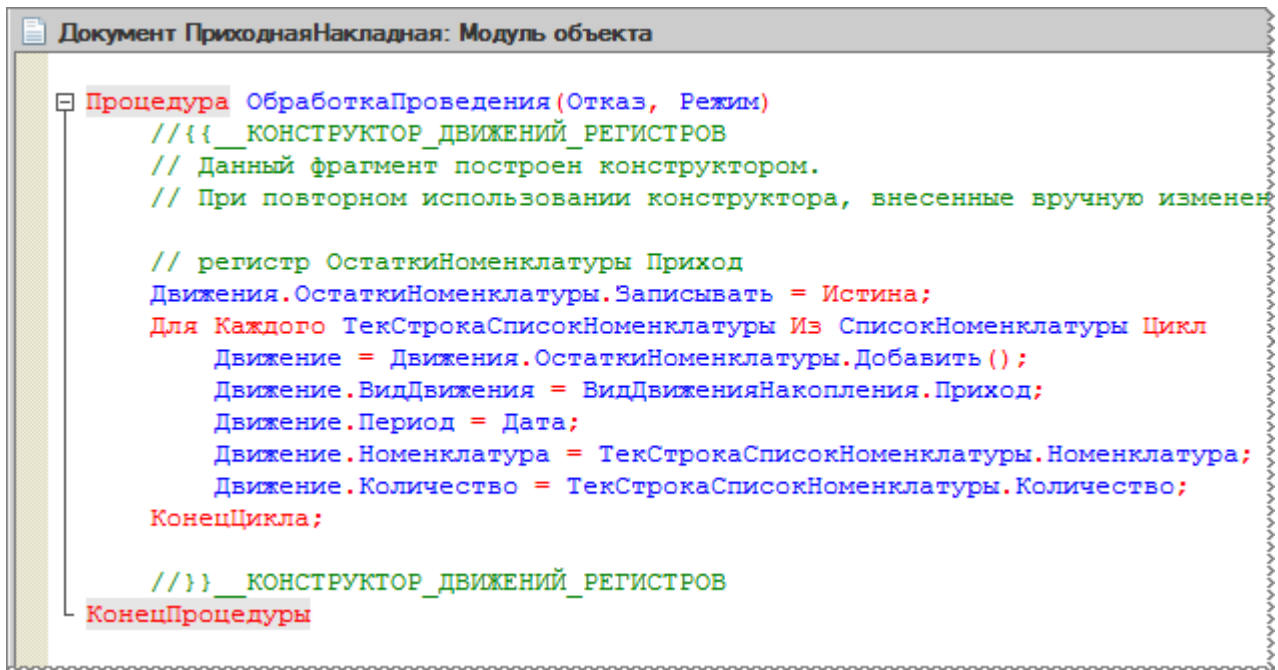


Рисунок 5 – Процедура, сформированная конструктором движений

Конечно, такая процедура – это только заготовка обработчика, а не полноценный код. При необходимости этот фрагмент программного кода можно доработать вручную, описать логику формирования движений документа.

На экзамене для приходных документов можно использовать обработку проведения, автоматически созданную конструктором. Для проведения приходных документов использование сложных алгоритмов на экзамене, как правило, не требуется. В процедурах проведения расходных документов требуется реализация сложной логики, и в таких случаях кода, сформированного конструктором, не достаточно.

Рассмотрим для примера документ «ПриходнаяНакладная». Откроем окно редактирования документа на закладке *Движения*:

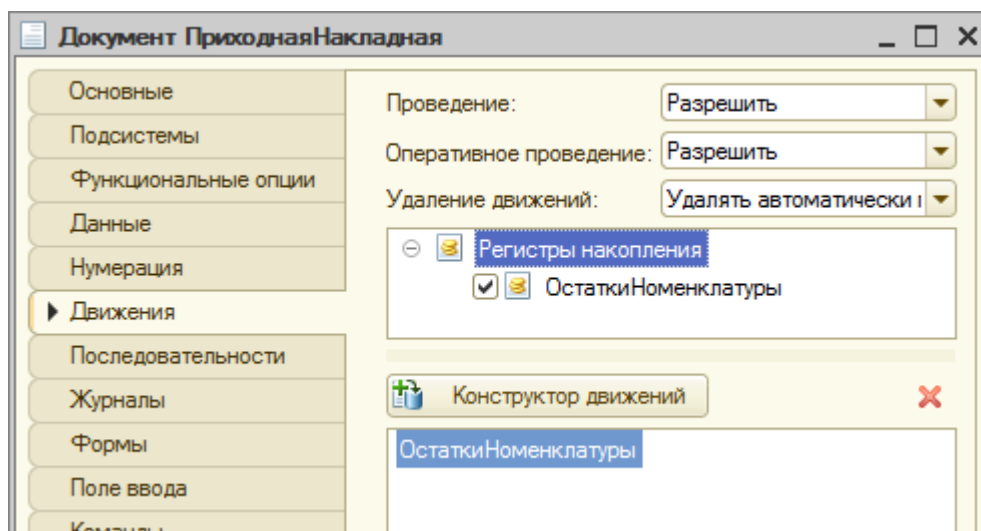


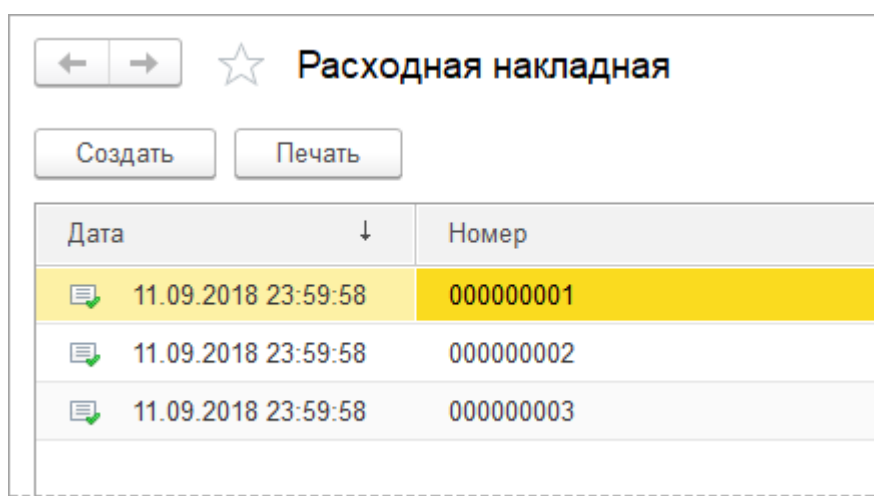
Рисунок 6 – Закладка «Движения»

Для этого документа процедура *ОбработкаПроведения* была сформирована автоматически. Документ может изменять состояние учета, то есть увеличивать остатки товаров на складах. При проведении документа формируются записи (движения прихода) в регистре накопления «ОстаткиНоменклатуры». Тем самым вносятся изменения в учетные данные. Остатки на складах увеличиваются. Происходит отражение хозяйственной операции в учете. Поскольку документ выполняет движения по регистру, данные этого документа попадут в отчеты, формируемые на основании данных этого регистра.

Если в пользовательском режиме документ только сохранен в базу без проведения, то это еще только «черновик», он не изменяет состояние учета, не изменяет остатки товаров на складах. Такой документ не выполняет движений по регистру «ОстаткиНоменклатуры», соответственно, данные этого документа не попадут в отчеты, формируемые на основании данных регистра «ОстаткиНоменклатуры».

3. Что такое момент времени и для чего он используется

Для документа реквизит *Дата* включает в себя дату и время с точностью до секунды. При этом в случае неоперативного проведения документов возможна ситуация, когда для двух и более документов установлена одинаковая дата и одинаковое время:



Дата	Номер
11.09.2018 23:59:58	000000001
11.09.2018 23:59:58	000000002
11.09.2018 23:59:58	000000003

Рисунок 1 – Список документов с одинаковой датой и временем

Чтобы данные документы разместить в хронологическом порядке, платформа использует объект *Момент времени*:



Рисунок 2 – Расположение документов на оси времени

Момент времени создается платформой автоматически, и изменить его в пределах одной секунды невозможно. Изменить хронологию документов можно, только переместив дату одного из документов на следующую секунду.

Момент времени представляет собой поле *Дата+Ссылка*:

Запрос: Документ.РасходнаяНакладная (Записей в результате: 3)	
МоментВремени	
11.09.2018 23:59:58; Расходная накладная 000000001 от 11.09.2018 23:59:58	
11.09.2018 23:59:58; Расходная накладная 000000002 от 11.09.2018 23:59:58	
11.09.2018 23:59:58; Расходная накладная 000000003 от 11.09.2018 23:59:58	

Рисунок 3 – Представление момента времени

Момент времени используется для решения следующих задач:

- Отображение списков документов в хронологическом порядке
- Получение остатков из регистров на момент времени документа
- Сортировка документов в запросе в хронологическом порядке.

Принципиален вопрос, как получать остатки из регистров накопления: на *Дату* документа или на *Момент времени*?

Рассмотрим пример.

В информационную базу введены три документа «Расходная накладная» с *Датой* документа 11.09.2018 23:59:58. Получим остатки по регистрам на *Дату* проведения документа «Расход 2»:



Рисунок 4 – Получение остатков на дату документа

Как видно из схемы, при получении остатков на *Дату* документа не учитываются движения документов, введенных одной датой и временем. Данная ошибка может быть причиной отрицательных остатков.

Получим остатки по регистрам на *Момент времени* документа «Расход 2»:



Рисунок 5 – Получение остатков на момент времени документа

Как видно из схемы, при получении остатков на *Момент времени* документа учитываются движения документа «Расход 1» и остатки рассчитываются корректно. По этой причине необходимо получать остатки по регистрам на *Момент времени* документа, а не на *Дату* документа.

МоментВремени – это момент непосредственно перед позицией документа. В ряде случаев используется не *Момент времени*, а объект *Граница* – момент непосредственно перед или после позиции документа, в зависимости от параметра *ВидГраницы*:



Рисунок 6 – Получение остатков на границу включая движения документа

Объект *Граница* используется, например, при решении задачи контроля остатков по «новой» методике при проведении документа «Расходная накладная». Данная задача рассматривается в рамках очередного дня занятий.

Важно! На экзамене получение остатков по регистрам на Дату документа является грубой ошибкой.

Рассмотрим примеры кода.

Получение *МоментВремени*:

`МоментПолученияОстатков = МоментВремени();`

Получение *Границы*, включая движения документа:

`МоментПолученияОстатков = Новый Граница(МоментВремени(), ВидГраницы.Включая);`

4. Как быстро получить движения по регистрам из управляемой формы – без единой строчки кода

Введение

При решении многих аттестационных задач требуется реализовать движения документа по различным регистрам: регистрам сведений, накопления, бухгалтерии или по регистрам расчета. Особенности таких задач подробно рассматриваются в разделах, посвященных оперативному учету, расчету зарплаты и бухгалтерскому учету. В разделе по управляемым формам может встретиться следующая задача, связанная с движениями документа по регистрам:

В документе реализации товаров, в форме документа и в форме списка обеспечить возможность просматривать движения, которые документ выполнил по регистрам.

Эта задача очень простая, и приемы, используемые при ее решении, позволят на экзамене сэкономить время для отладки программного кода, формирующего движения по регистрам. Это очень важно, поскольку времени на аттестации выделяется не много, и необходимо его расходовать осознанно, только на самые важные действия.

Процесс создания движений по регистрам выглядит обычно следующим образом: написание программного кода, запуск приложения в пользовательском режиме, проведение тестовых документов, работа в отладчике, просмотр значений переменных, анализ сформированных движений по различным регистрам. И так несколько раз до получения корректного кода.

Чтобы увидеть движения по регистрам, можно из меню *Все функции* перейти к нужному регистру, отобразить записи по одному документу-регистратору, проверить корректность этих движений. Также можно вынести форму списка регистра в командный интерфейс для быстрого доступа.

Однако удобнее реализовать возможность перехода к движениям по регистрам прямо из формы документа. Это позволит ускорить отладку, поскольку не нужно будет тратить время на перемещение по командному интерфейсу, на работу с меню *Все функции*, на открытие каждого регистра, на установку отбора по регистратору, чтобы проверить, правильно ли написан код, сформировались ли требуемые движения по регистрам. Поэтому необходимо уметь реализовывать переход к движениям по регистрам прямо из формы.

Рассмотрим два варианта форм:

- Форма документа
- Форма списка.

Как перейти к движениям из формы документа?

Создадим управляемую форму документа. В конструкторе управляемой формы в левом верхнем углу перейдем на закладку *Командный интерфейс*. В панели навигации для пункта *Перейти* установим видимость движений по всем регистрам:

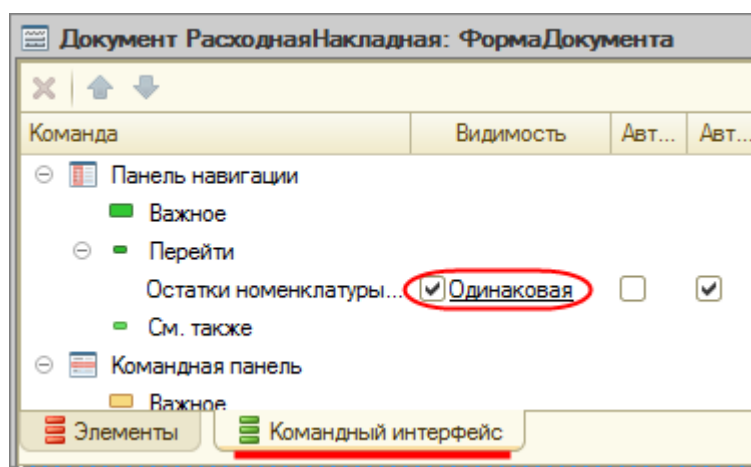


Рисунок 1 – Настройка панели навигации

По умолчанию видимость движений по регистрам выключена. Чтобы ее включить, нужно установить галочку в колонке *Видимость*.

В пользовательском режиме теперь будут отображаться гиперссылки для перехода к связанным регистрам:

Расходная накладная (создание) *

Основное **Остатки номенклатуры**

Провести и закрыть Записать Провести Еще

Номер:

Дата: 17.09.2018 0:00:00

Сумма по документу:

Добавить Еще

	Номенклатура	Количество	Цена	Сумма
1	Паркер "Golg"	5	3 000,00	15 000,00

Рисунок 2 – Ссылки в панели навигации

Как перейти к движениям из формы списка?

Создадим для документа управляемую форму списка. В форме списка в левом верхнем углу на закладке Элементы создадим в Командной панели группу-подменю *ГруппаПерейти*:

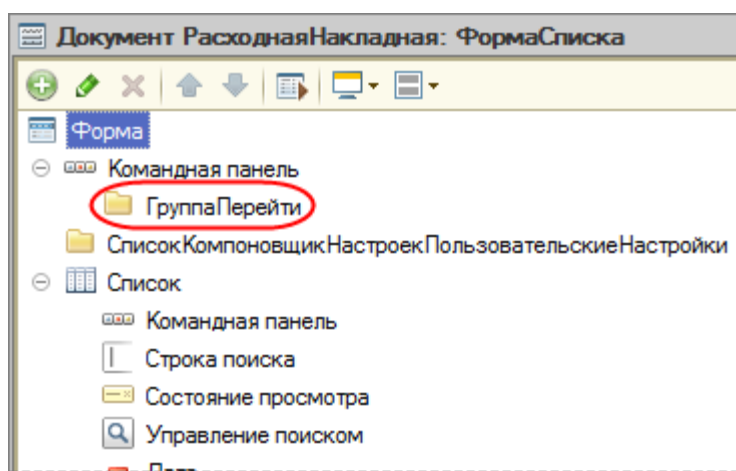


Рисунок 3 – Добавление группы на форму

В этом подменю будут располагаться команды для просмотра движений документа по регистрам.

В конструкторе управляемой формы в правом верхнем углу перейдем на закладку *Команды – Глобальные команды*. В табличной части *Параметризуемые* отображается список регистров, по которым документ может выполнять движения:

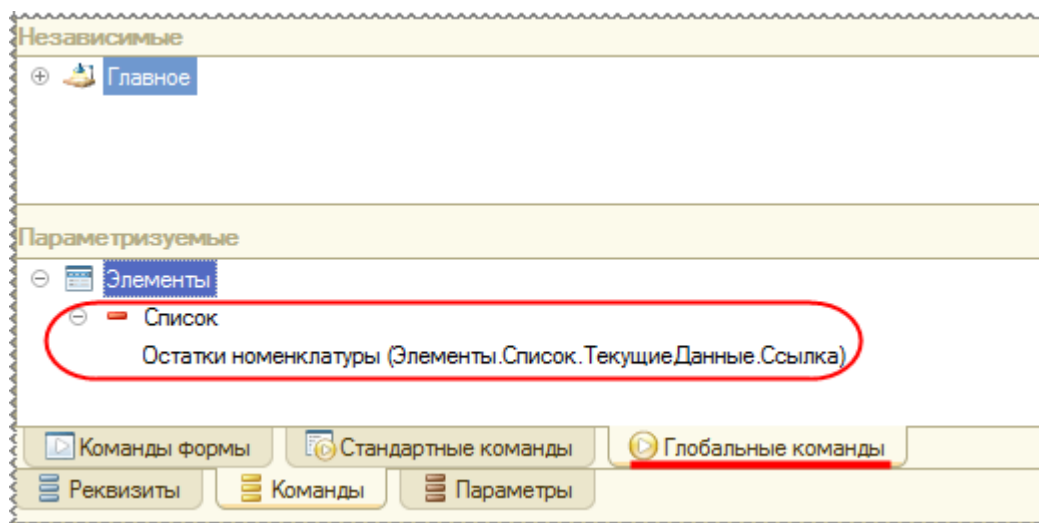


Рисунок 4 – Глобальные команды

Из этого списка мышью перетаскиваем команды на открытие форм записей регистров в подменю *ГруппаПерейти* на форме:

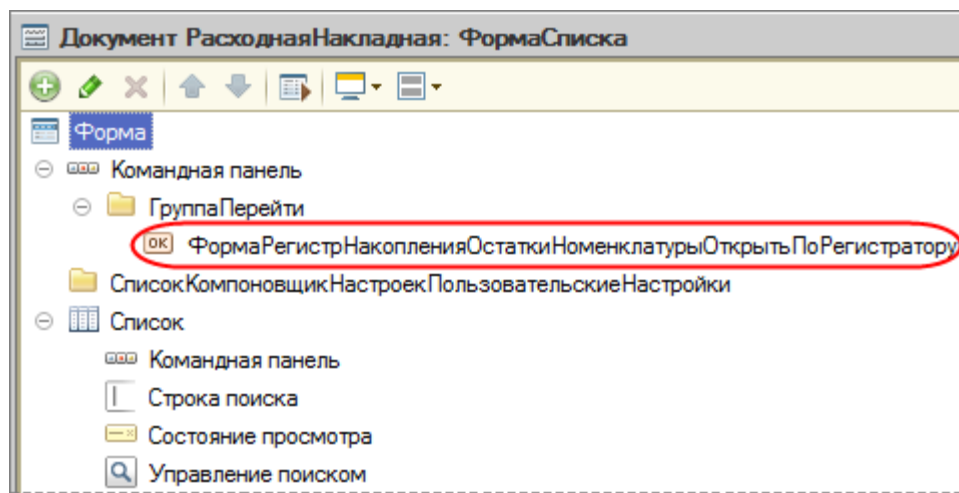


Рисунок 5 – Размещение команды на форме

В пользовательском режиме форма списка с такими настройками выглядит следующим образом:

Дата	
17.09.2018 16:29:16	000000001

Рисунок 6 – Переход к движениям по регистру из формы списка

Итак, в данной задаче рассмотрено, как в форме документа и в форме списка обеспечить возможность просматривать движения, которые документ выполнил по регистрам. Важно, что все настройки были выполнены без написания программного кода, а только интерактивными действиями. Это поможет сэкономить время на аттестации.

5. Как обеспечить возможность запуска обычных форм в режиме управляемого приложения (используем Консоль запросов из каркасной конфигурации)

В состав каркасной конфигурации входит обработка «Консоль запросов». Пользоваться ею на экзамене не обязательно, но она может пригодиться в процессе решения задач. Например, для следующих целей:

- Написать и отладить запрос, который будет использоваться при решении задачи
- Написать проверочные запросы, которые помогут найти причину ошибки в случае возникновения каких-либо сложностей или ошибок в процессе решения
- Написать при необходимости запрос для контроля наличия остатков по регистру. На аттестации очень важно, чтобы учетная схема решения задачи позволяла вывести в ноль все ресурсы регистра остатков. За невыполнение этого требования снимают от 0,5 до 3 баллов
- Проверить с помощью запроса состояние данных в информационной базе, прежде чем приступить к разработке отчетов.

«Консоль запросов» запускается в каркасной конфигурации в пользовательском режиме из меню **Сервис**:

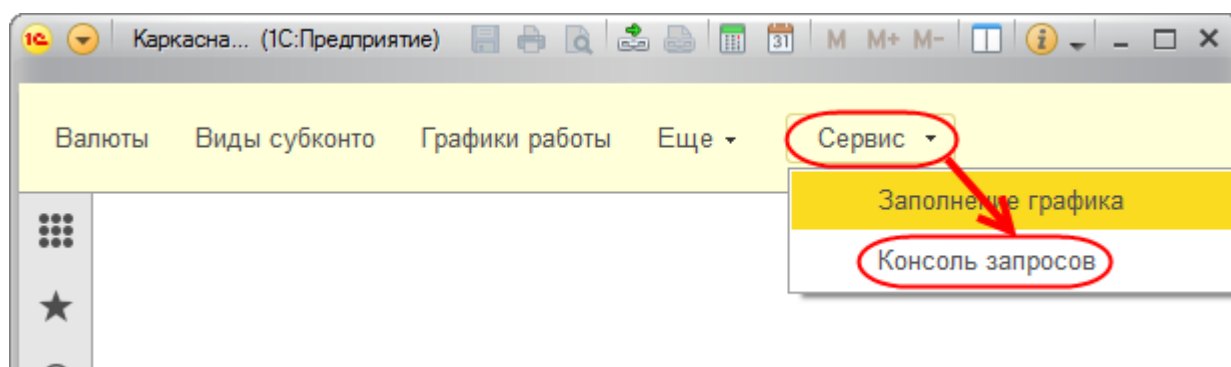


Рисунок 1 – Запуск обработки «Консоль запросов»

При этом система сообщает об ошибке:

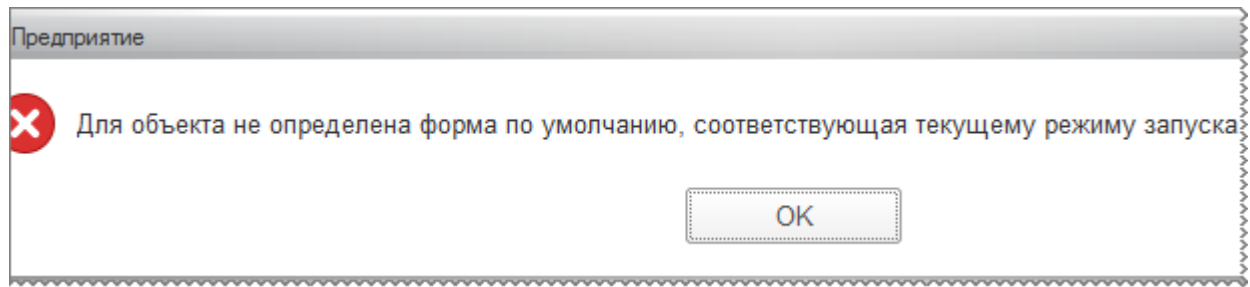


Рисунок 2 – Сообщение об ошибке при попытке запуска «Консоли запросов»

Из сообщения следует, что имеется проблема с формой. Проверим в конфигураторе, какие формы предусмотрены для обработки «Консоль запросов»:

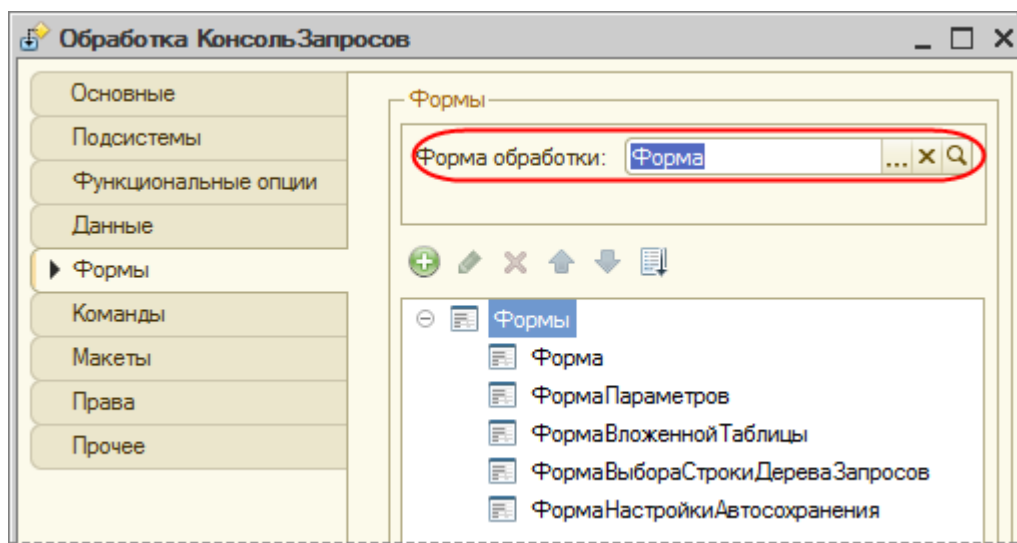


Рисунок 3 – Закладка «Формы» обработки «Консоль запросов»

Основная форма обработки определена. Откроем ее. Данная форма открывается в редакторе обычных форм:

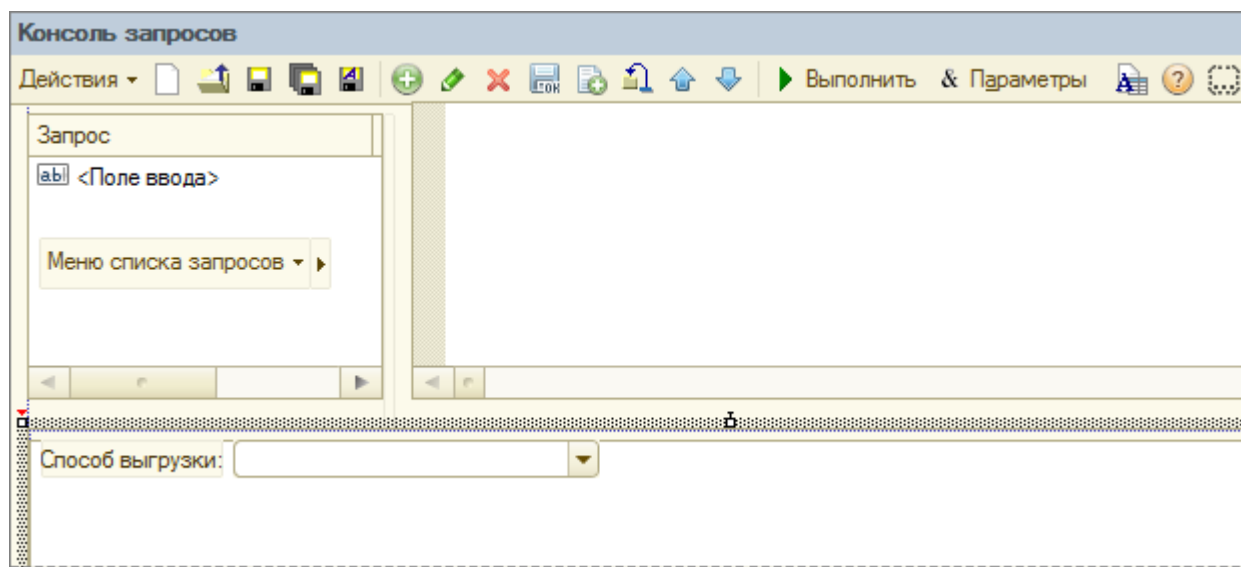


Рисунок 4 – Основная форма обработки «Консоль запросов»

Таким образом, причина сообщения об ошибке понятна: обработка «Консоль запросов» разработана на обычных формах, а попытка запуска осуществлялась в режиме управляемого приложения.

«Консоль запросов» включили в состав каркасной конфигурации в таком варианте не по ошибке, а преднамеренно, с целью проверки, что аттестуемый знает, как обеспечить запуск обычных форм в режиме управляемого приложения. На очной аттестации экзаменатор может попросить запустить «Консоль запросов» в качестве дополнительного вопроса. За неверный ответ на дополнительный вопрос снимают от 0,5 до 1 балла.

Умение разрабатывать обычные формы на аттестации не проверяется. Достаточно лишь уметь запускать их в режиме управляемого приложения. Но если необходима дополнительная информация по обычным формам, то найти ее можно в курсе «[Профессиональная разработка интерфейсов и форм в 1С:Предприятие 8.3](#)».

Разберем, как запускать обычные формы в режиме управляемого приложения.

Сначала в конфигураторе в меню *Сервис* выберем пункт *Параметры*:

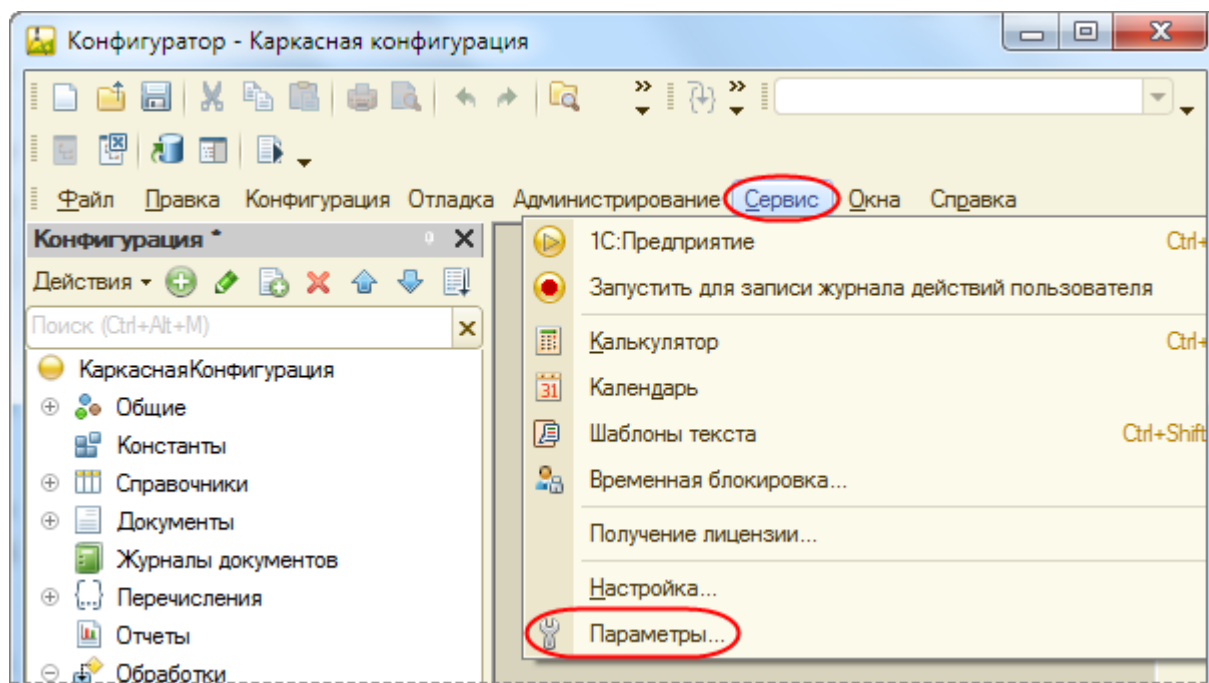


Рисунок 5 – Вызов окна настройки параметров

Откроется окно настройки параметров. На закладке *Общие* для параметра *Редактирование конфигурации для режимов запуска* выберем значение «Управляемое приложение и обычное приложение»:

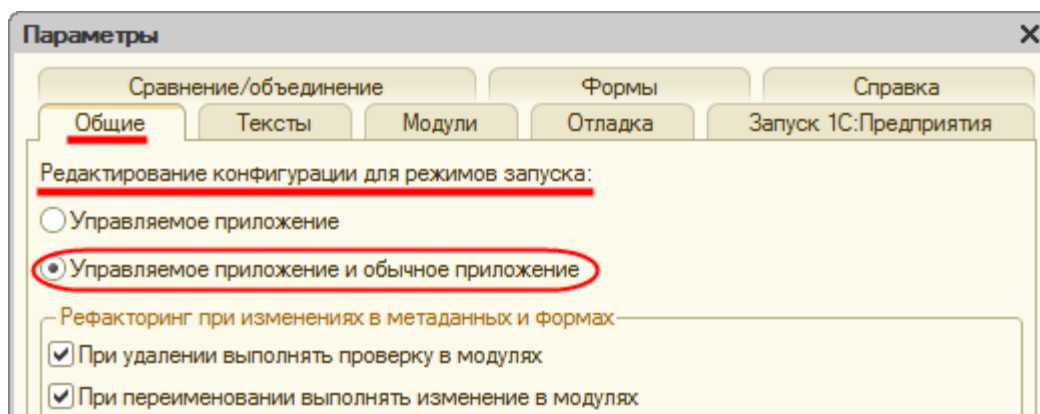


Рисунок 6 – Установка режима редактирования конфигурации для обеспечения возможности запуска обычных форм

После этого в свойствах конфигурации необходимо установить флаг «Использовать обычные формы в управляемом приложении»:

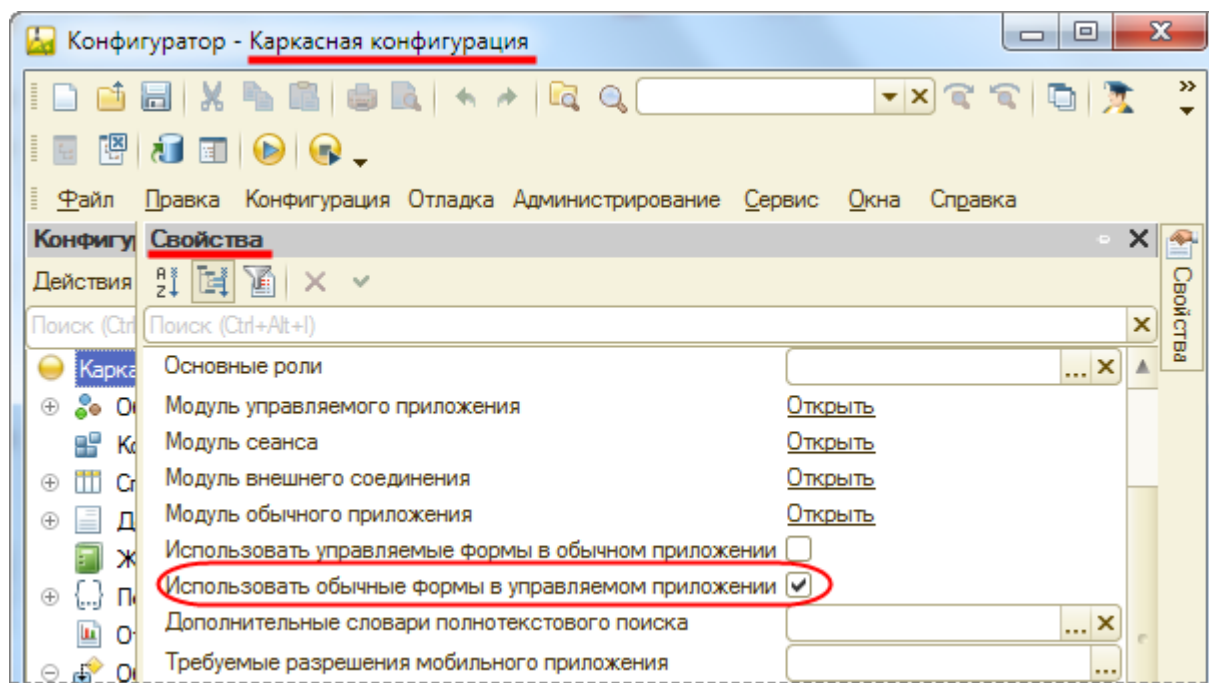


Рисунок 7 – Установка флага «Использовать обычные формы в управляемом приложении»

Однако после перезапуска конфигурации в режиме «1С:Предприятие» очередная попытка запустить обработку «Консоль запросов» вновь завершается сообщением об ошибке:

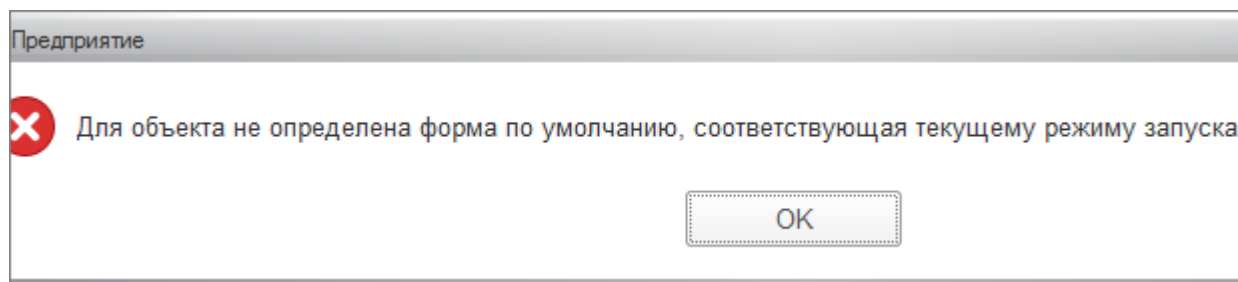


Рисунок 8 – Сообщение об ошибке при попытке запуска «Консоли запросов»

Дело в том, что приложение по умолчанию запускается в режиме тонкого клиента, а с обычными формами возможно работать только в толстом клиенте, так как модуль обычной формы компилируется в контексте толстого клиента.

Поэтому запустим управляемое приложение в режиме толстого клиента:

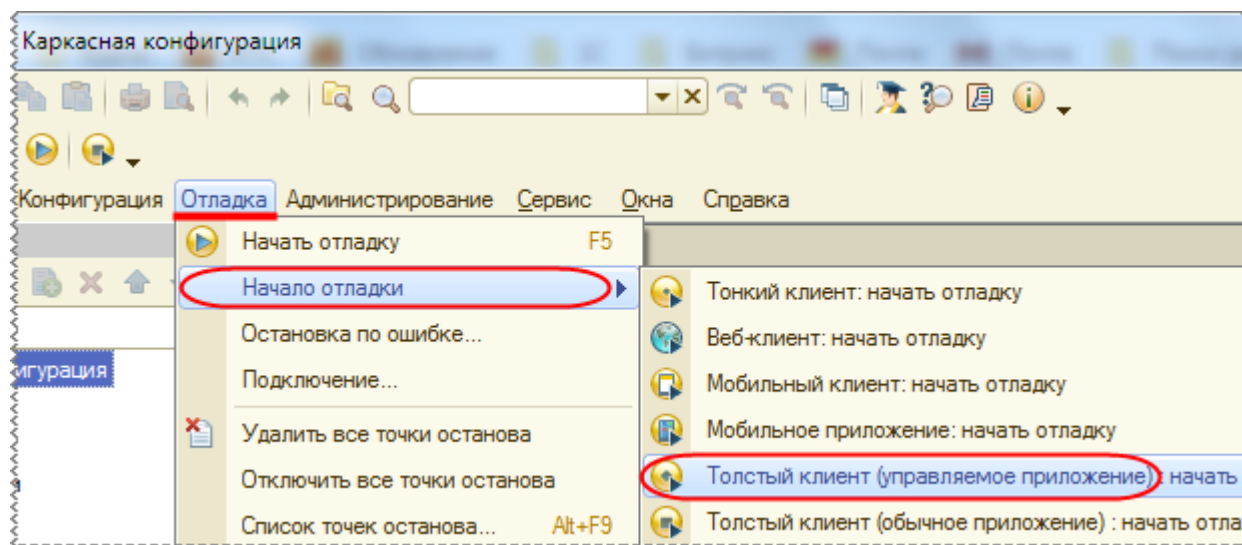


Рисунок 9 – Запуск «1С:Предприятия» в режиме Толстый клиент (управляемое приложение)

Если конфигурация редактируется в режиме «Управляемое приложение и обычное приложение», то доступен вариант запуска «Толстый клиент (обычное приложение)». Можно было бы выбрать и этот режим, чтобы убедиться, что конфигурация успешно запустится. «Консоль запросов» откроется. Но на аттестации так делать не следует, потому что предусмотрен вариант запуска в режиме управляемого приложения. Экзаменаторы проверяют знание именно этого варианта запуска. На аттестации вообще не следует использовать запуск в режиме обычного приложения. Решение должно работать именно в режиме управляемого приложения.

Для удобства в настройках установим режим запуска по умолчанию *Толстый клиент (управляемое приложение)*. Для этого в Конфигураторе в меню Сервис выберем пункт Параметры и в открывшейся форме на закладке Запуск 1С:Предприятия выберем вариант Толстый клиент (управляемое приложение):

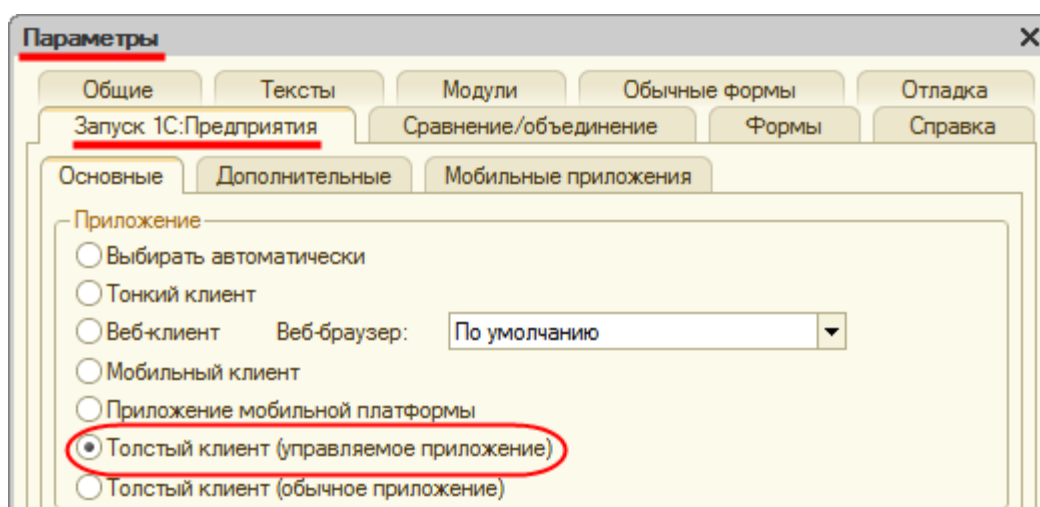


Рисунок 10 – Установка запуска по умолчанию в режиме «Толстый клиент (управляемое приложение)»

Снова запустим конфигурацию в режиме «1С:Предприятие» и благополучно откроем «Консоль запросов»:

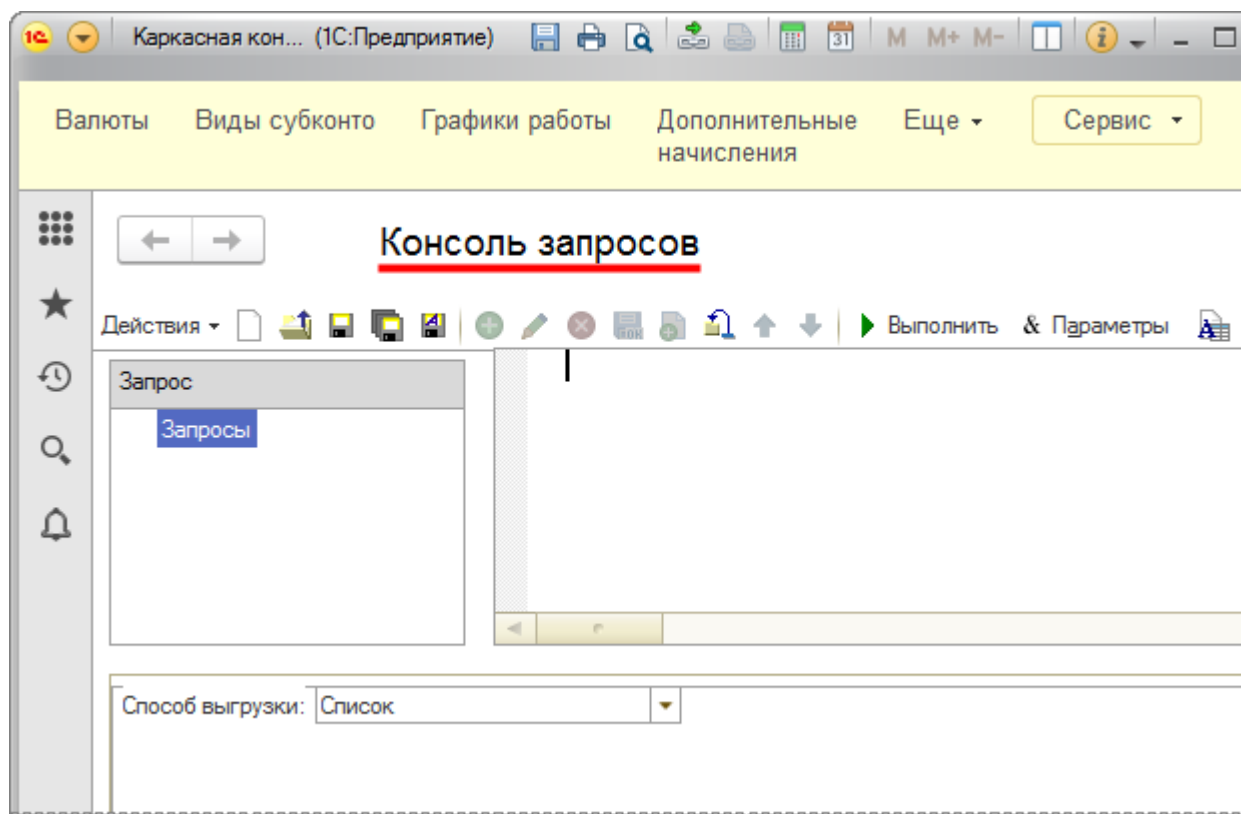


Рисунок 11 – Форма обработки «Консоль запросов»

Как найти ошибку в тексте запроса

Остался последний нюанс: если в тексте запроса будет допущена ошибка, то по нажатию на кнопку *Выполнить* увидим следующее сообщение об ошибке:

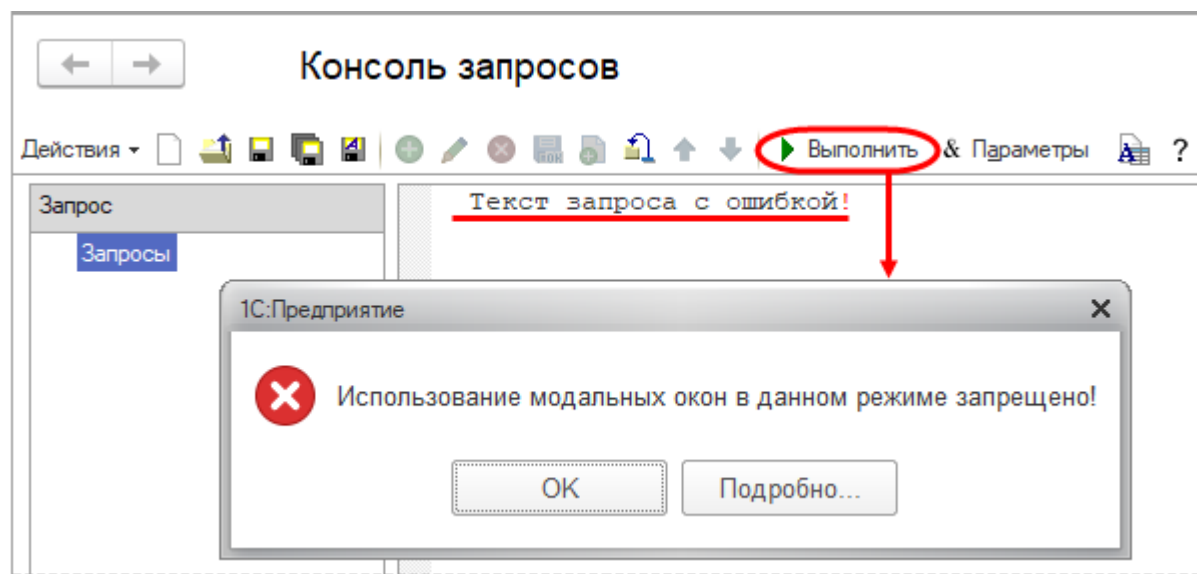


Рисунок 11 – Сообщение при попытке выполнить запрос с ошибкой

И даже если в окне с ошибкой нажать кнопку *Подробнее*, выведенное сообщение никак не поможет понять, где в тексте запроса ошибка.

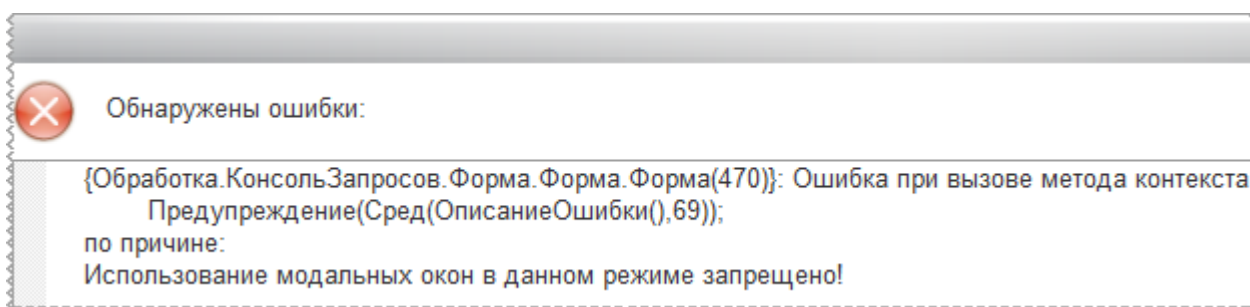


Рисунок 12 – Подробности сообщение об ошибке

Дело в том, что по умолчанию для конфигурации установлен режим использования модальности в значение «Не использовать». Чтобы все-таки увидеть текст ошибки, нужно в свойствах конфигурации изменить режим использования модальности – разрешить использование модальных окон.

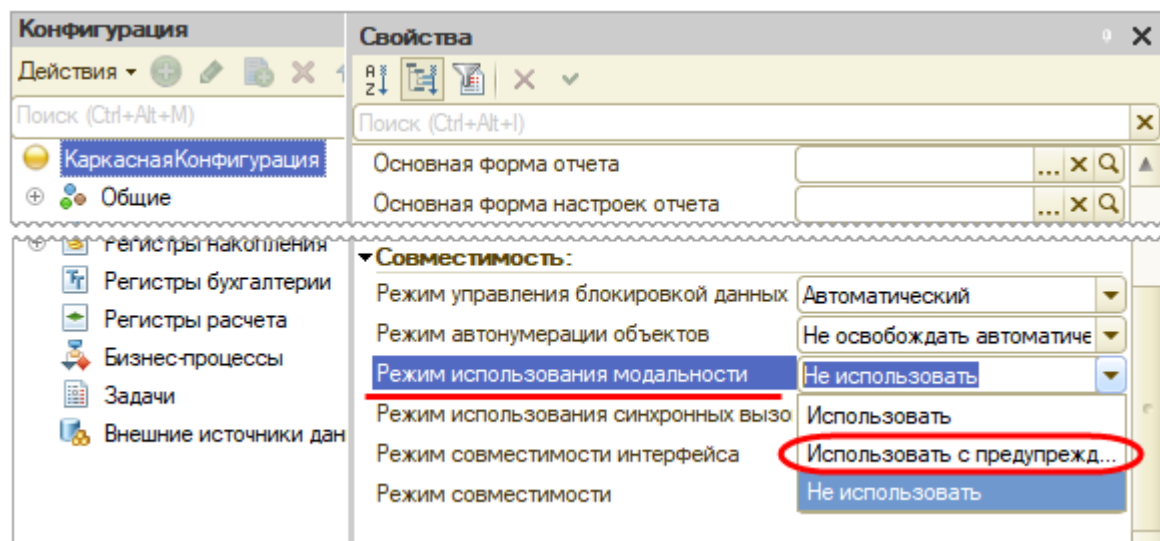


Рисунок 13 – Изменение режима модальности конфигурации

Установим режим использования модальности «Использовать с предупреждением». Запускаем конфигурацию в режиме «1С:Предприятие», открываем «Консоль запросов» и снова пытаемся выполнить запрос с ошибкой.

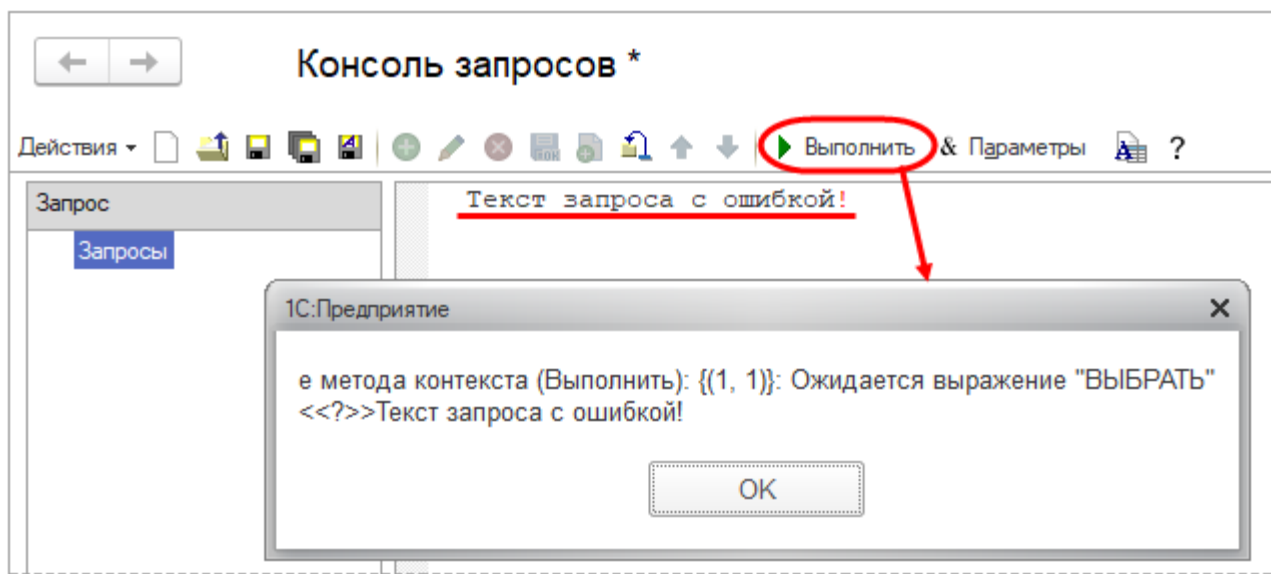


Рисунок 14 – Сообщение об ошибке в тексте запроса

Теперь получено информативное сообщение об ошибке в запросе. Что и требовалось.

Без необходимости изменять режим модальности не нужно, и после того, как ошибка в тексте запроса найдена, лучше вернуть режим использования модальности в «не использовать», так как в требованиях к экзамену есть указание, что решение должно одинаково работать как в тонком, так и в Веб-клиенте. А для обеспечения работы в Веб-клиенте нужно использовать режим работы без использования модальности.

Подробнее вопросы модальности рассматриваются в курсе «[Профессиональная разработка интерфейсов и форм в 1С:Предприятие 8.3](#)».

Итоги

Для запуска обычных форм в режиме управляемого приложения нужно выполнить следующие настройки:

- В меню *Сервис* выбрать пункт *Параметры* и в открывшейся форме на закладке *Общие* для параметра *Редактирование конфигурации для режима запуска* установить значение «Управляемое приложение и обычное приложение»
- В свойствах конфигурации установить флаг *Использовать обычные формы в управляемом приложении*.

Запустить конфигурацию в режиме *Толстый клиент (управляемое приложение)*. Можно в этом режиме настроить запуск по умолчанию, если в меню *Сервис* выбрать пункт *Параметры* и в открывшейся форме на закладке *Запуск 1С:Предприятие* выбрать вариант *Толстый клиент (управляемое приложение)*.

6. Как реализовать поступление товаров в компанию

В каждой экзаменационной задаче, которая требует автоматизировать учет номенклатуры, необходимо реализовать поступление товаров в компанию. Такая подзадача встречается в следующих задачах сборника для подготовки к экзамену «1С:Специалист по платформе»: № 1.1, 1.2, 2.9, 2.14 и другие.

Рассмотрим, как быстро реализовать данный функционал.

Постановка задачи:

Компания занимается оптовой торговлей. Поступление товаров отражается документом «Приходная накладная».

Для решения поставленной задачи в конфигурации должны быть созданы объекты:

- Справочник «Номенклатура»
- Документ «Приходная накладная»
- Регистр накопления «Остатки номенклатуры».

Примечание. При решении других задач из сборника для подготовки к экзамену «1С: Специалист по платформе» могут использоваться разные показатели. Например: себестоимость товаров, взаиморасчеты с контрагентами. В этом случае необходимо использовать другие регистры накопления.

Так как документ «Приходная накладная» только увеличивает остатки товаров, то контроль остатков при проведении этого документа не требуется.

Справочник «Номенклатура» в каркасной конфигурации уже присутствует. Для решения задачи новые реквизиты добавлять не нужно.

Для отражения хозяйственных операций по условию задачи требуется документ «Приходная накладная». Данный документ также уже присутствует в каркасной конфигурации. И в нем уже есть необходимые для решения задачи реквизиты табличной части *Список номенклатуры*:

- Номенклатура (СправочникСсылка.Номенклатура)
- Количество (Число 10, 0):

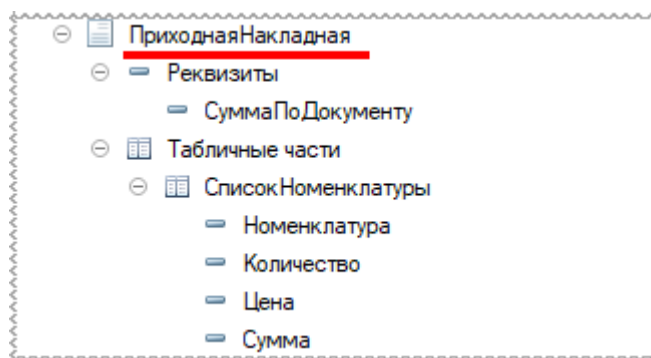


Рисунок 1 – Документ «Приходная накладная» в дереве конфигурации

Для экономии времени на решение задачи создавать форму документа не будем, так как в условии задачи нет прямого указания создавать формы. Для реализации алгоритма решения создание формы также не требуется.

Для хранения остатков будем использовать уже имеющийся в каркасной конфигурации регистр накопления «Остатки номенклатуры». Вид данного регистра – *Остатки*:

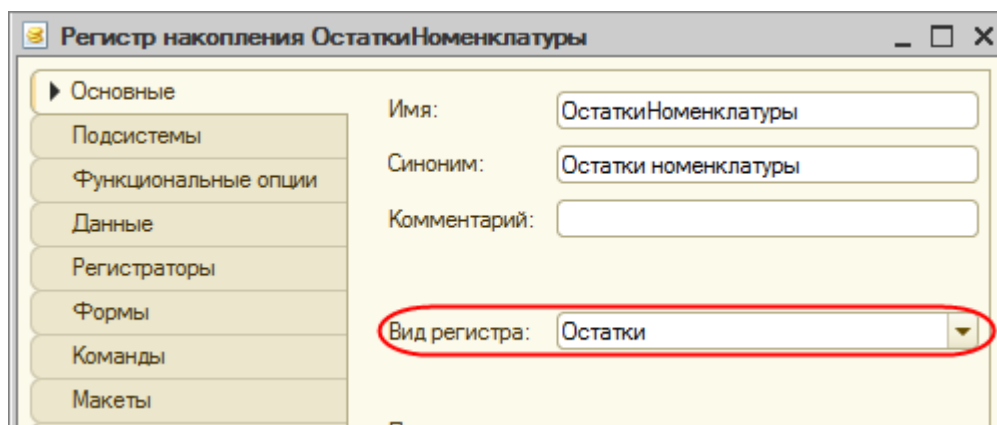


Рисунок 2 – Закладка Основные РН «Остатки номенклатуры»

Измерения регистра:

- *Номенклатура (СправочникСсылка.Номенклатура).*

Ресурсы:

- *Количество (Число 10, 0).*

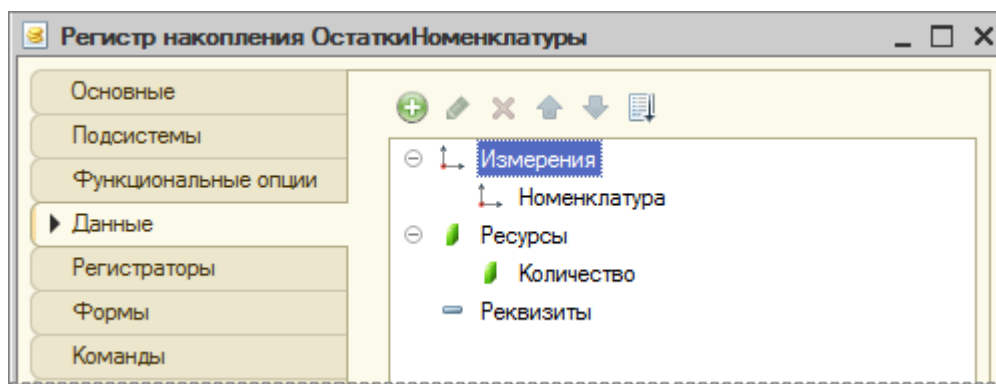


Рисунок 3 – Закладка *Данные* РН «Остатки номенклатуры»

Структура регистра полностью подходит для решения поставленной задачи. Единственное, что нужно изменить в структуре регистра – это включить свойство «Запрет незаполненных значений» у измерения «Номенклатура». Включение данного свойства необходимо, так как движения с пустой номенклатурой будут считаться учетной ошибкой:

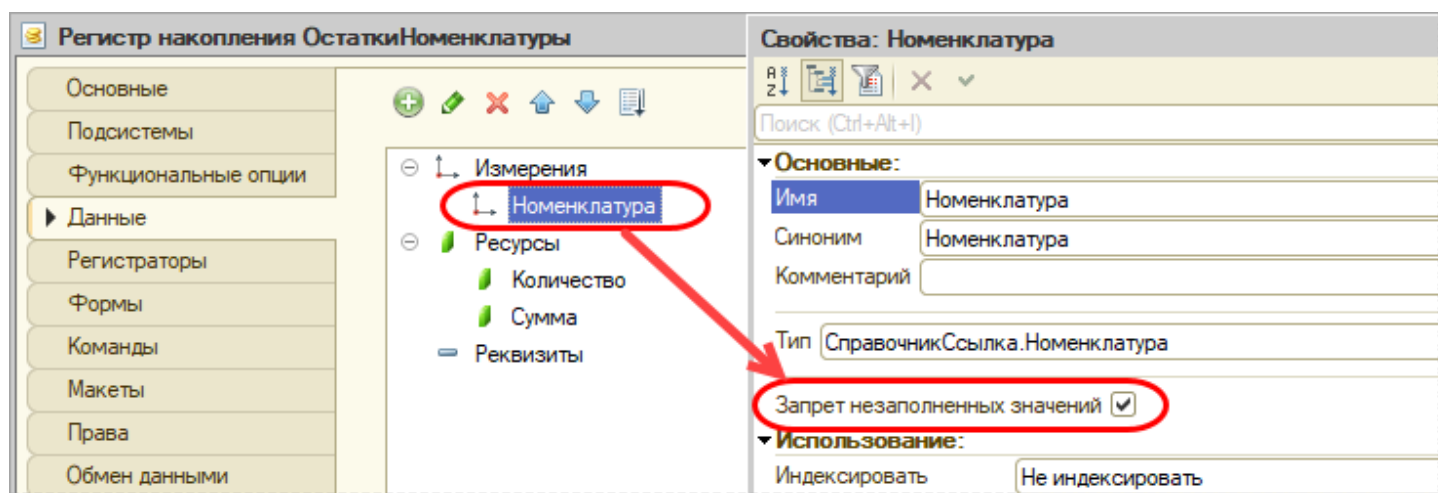


Рисунок 4 – Установка свойства «Запрет незаполненных значений»

Установим в качестве регистраторов документы:

- «Приходная накладная»
- «Расходная накладная»:

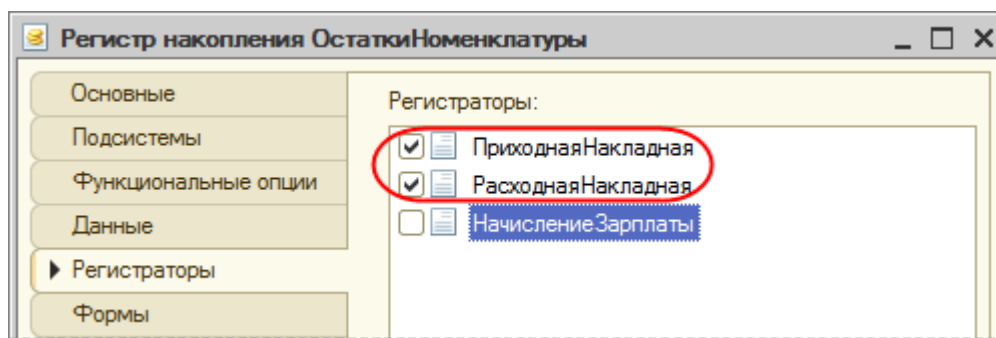


Рисунок 5 – Закладка *Регистраторы* РН «Остатки номенклатуры»

Обработка проведения для документа «Приходная накладная»

Для того чтобы документ «Приходная накладная» не проводился с незаполненной табличной частью «Список номенклатуры», настроим для нее свойство *Проверка заполнения*, установив значение «Выдавать ошибку»:

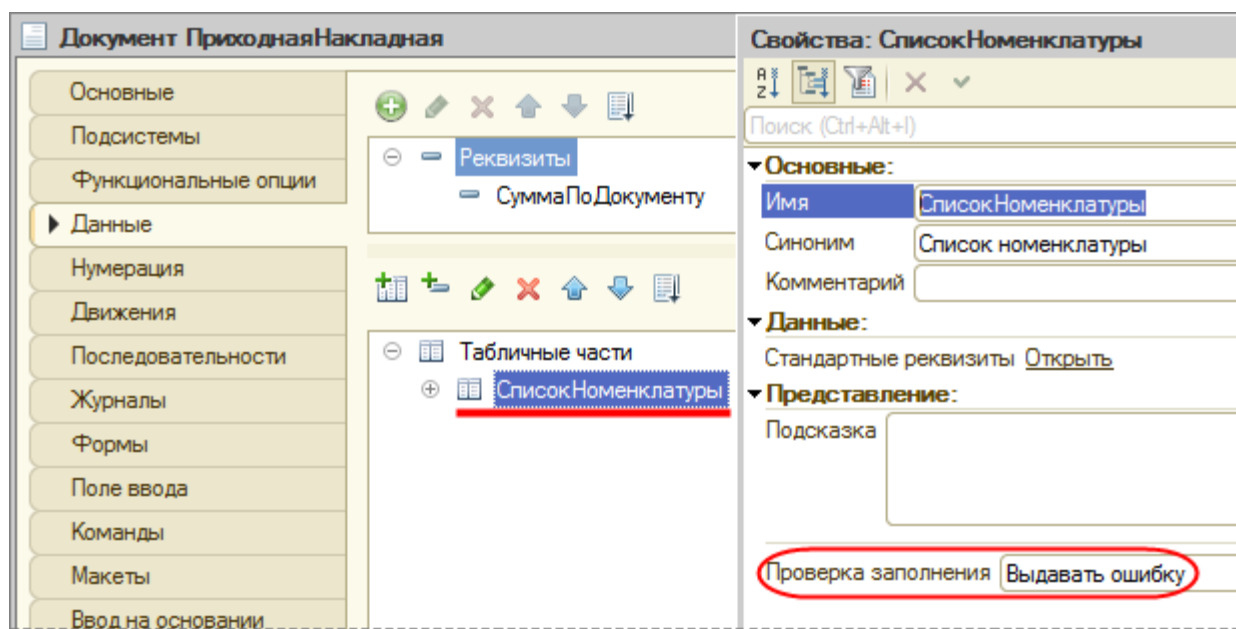


Рисунок 6 – Установка свойства «Проверка заполнения»

Процедуру *ОбработкаПроведения* можно создать двумя способами:

1. Полностью автоматически с помощью конструктора движений

2. Создать процедуру вручную. В процедуре с помощью запроса выбрать данные из табличной части *СписокНоменклатуры* и загрузить результат запроса в набор движений *ОстаткиНоменклатуры*.

Так как время на экзамене ограничено воспользуемся первым способом.

С помощью конструктора движений сформируем движения документа «Приходная накладная» по регистру накопления «Остатки номенклатуры» с видом *Приход*. Конструктор движений вызывается с помощью соответствующей кнопки на закладке *Движения* в окне редактирования объекта конфигурации:

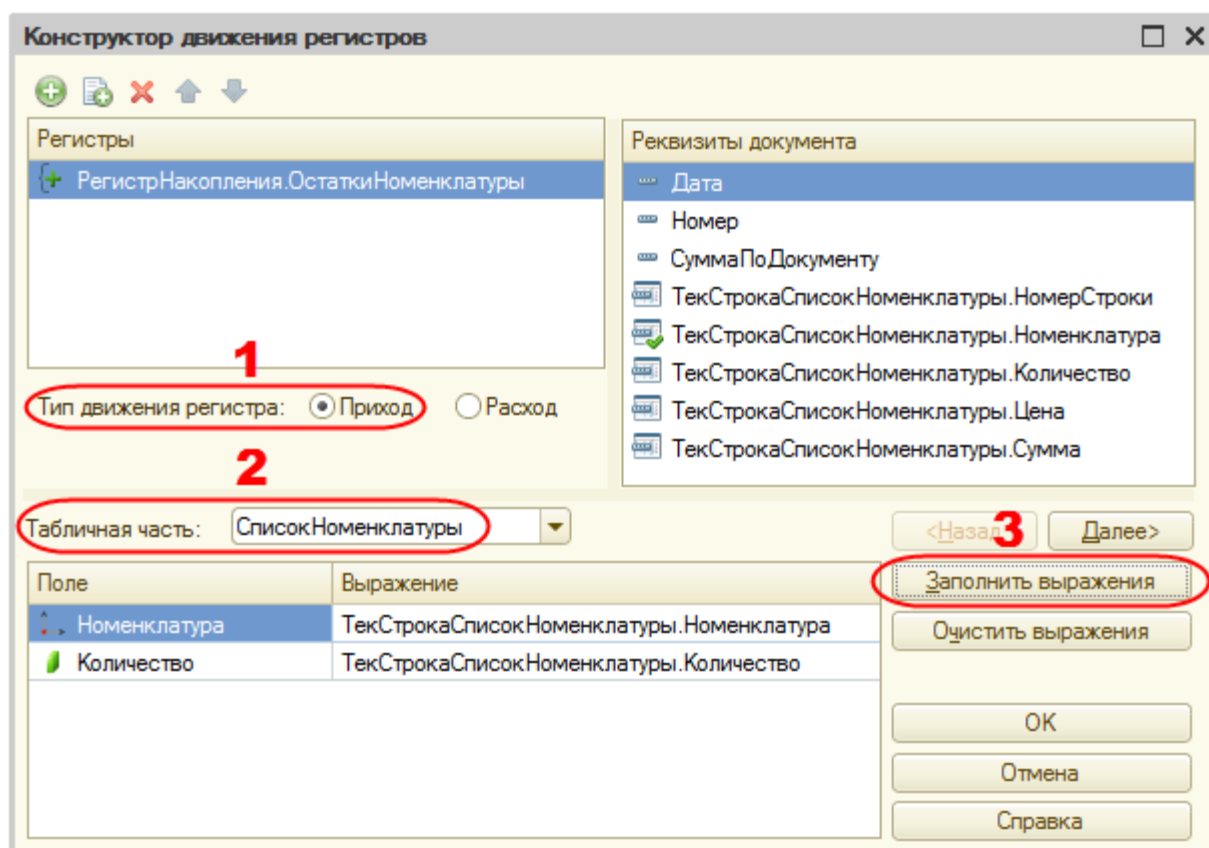


Рисунок 7 – Формирование движений конструктором

В модуле объекта автоматически была создана процедура *ОбработкаПроведения* со следующим кодом:

Процедура *ОбработкаПроведения(Отказ, Режим)*

///{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ

// Данный фрагмент построен конструктором.

// При повторном использовании конструктора, внесенные вручную изменения будут утеряны!!!

// регистр ОстаткиНоменклатуры Приход

Движения.ОстаткиНоменклатуры.Записывать = Истина;

Для Каждого ТекСтрокаТовары Из Товары Цикл

Движение = Движения.ОстаткиНоменклатуры.Добавить();


```

Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
Движение.Период = Дата;
Движение.Номенклатура = ТекСтрокаСписокНоменклатуры.Номенклатура;
Движение.Количество = ТекСтрокаСписокНоменклатуры.Количество;
КонецЦикла;

```

```

//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ

```

КонецПроцедуры

В комментариях конструктор движений регистров предупреждает, что в случае повторного его использования внесенные вручную изменения будут утеряны.

Запустим конфигурацию в режиме «1С:Предприятие» и проверим решение.

Сформируем документ «Приходная накладная»:

N	Номенклатура	Количество	Цена	Сумма
1	Big (капиллярная)	10	100,00	1 000,00

Рисунок 8 – Добавление документа «Приходная накладная»

Откроем регистр накопления «Остатки номенклатуры» и посмотрим сформированные движения.

Период	Регистратор	Номер стр	Номенклатура	Количество
28.09.2018 13:18:58	Приходная накладная 00...	1	Big (капиллярная)	10

Рисунок 9 – Движения документа «Приходная накладная»

Движения сформировались верно, задача решена.

7. Старая и новая методики контроля остатков – их преимущества, недостатки и рекомендации, какую методику выбрать в решении

Решение экзаменационных задач из разных разделов предполагает создание документов и реализацию алгоритмов их проведения. В алгоритмах проведения многих документов, связанных с оперативным и бухгалтерским учетом, требуется выполнение дополнительных проверок (хватает ли товара на складе, не превышает ли задолженность покупателя допустимый порог и т.д.).

Типичным примером документов, для которых осуществляется контроль остатков, является «Расходная накладная». В данном разделе рассмотрим методики контроля отрицательных остатков и их программную реализацию. Эти методики используются при решении задач оперативного и бухгалтерского учета. Поэтому при рассмотрении такого рода задач в соответствующих разделах курса подробно останавливаться на этих методиках уже не будем.

С появлением версии платформы «1С:Предприятие 8.2» существуют две методики контроля отрицательных остатков с условными названиями: «старая» и «новая».

«Старая методика» – это подход к контролю остатков, который использовался еще со времен «1С:Предприятие 8.0»: остатки контролируются до записи движений в регистры.

В «новой методике» контроль происходит после записи движений, то есть постфактум.

Рекомендуется всегда использовать «новую» методику контроля остатков, так как она более эффективна. «Старую» методику нужно использовать только тогда, когда «новую» методику технически выполнить невозможно.

Обе методики применяются в решении аттестационных и практических задач. При этом важно понимать какую из методик необходимо использовать для решения конкретной задачи.

Рассмотрим каждую методику более подробно.

Старая методика контроля остатков

Основным отличием «старой» методики от «новой» является последовательность действий по контролю остатков и формированию движений. По «старой» методике сначала получают данные об

остатках и осуществляется контроль их достаточности, а потом формируются движения. По «новой» методике сначала формируются движения, а потом контролируется полученный результат на наличие отрицательных остатков.

«Старая» методика контроля отрицательных остатков применяется, когда в самом документе данных для формирования движений недостаточно. Для формирования движений по регистру требуется обращение к другим таблицам базы данных. Поэтому использовать «новую» методику невозможно. Так, например, для формирования движений по регистру бухгалтерии требуются предварительно получить данные о себестоимости списываемых товаров.

Алгоритм можно разделить на 3 этапа:

1. Получение остатков из регистра на момент времени документа
2. Сравнение полученных остатков со значениями количества и суммы из документа
3. Формирование и запись движений по регистру в случае неотрицательного результата сравнения.

Недостатки:

- Использование соединения таблиц в запросе на получение остатков по позициям, которые присутствуют в документе, что усложняет конструкцию запроса
- Необходимость предварительного наложения блокировок для изменения необходимых данных регистра при получении актуальных остатков, что приводит к снижению параллельности работы пользователей.

Новая методика контроля остатков

Новая методика контроля отрицательных остатков применяется тогда, когда для формирования движений по контролируемому регистру достаточно данных из документа.

При использовании данной методики контроля отрицательных остатков алгоритм можно разделить на 2 этапа:

1. Формирование и запись движений
2. Получение и проверка остатков из регистра после записи движений. Если появились отрицательные остатки, то осуществляется возврат в точку программы, из которой было обращение к процедуре, то есть транзакция зафиксирована не будет и произойдет откат. Таким образом, документ не пройдет.

Преимущества:

- Отсутствие соединения таблиц в запросе на получение остатков

- Сокращение времени блокировки таблиц.

Как видно из описания, «новая» методика полностью противоположна «старой» методике контроля отрицательных остатков. При этом «новая» методика более оптимальна с точки зрения производительности и масштабируемости. Поэтому общая рекомендация – при малейшей возможности использовать «новую» методику контроля отрицательных остатков.

8. Как реализовать контроль остатков по новой методике

Постановка задачи:

Компания занимается оптовой торговлей товаров. Поступление товаров отражается документом «Приходная накладная», продажа – «Расходная накладная». Учет товаров в разрезе складов не ведется. При проведении расходной накладной при нехватке товара необходимо выдавать соответствующее предупреждение с указанием количества нехватки и не позволять проводить документ.

Для решения поставленной задачи в конфигурации должны быть созданы объекты:

- справочник «Номенклатура»
- документ «Приходная накладная»
- документ «Расходная накладная»
- регистр накопления «Остатки номенклатуры».

Так как документ «Приходная накладная» только увеличивает остатки товаров, то контроль остатков при проведении этого документа не требуется.

В документе «Расходная накладная» присутствуют все необходимые данные для формирования движений по регистру «Остатки номенклатуры» (данные по себестоимости не требуются исходя из условий задачи). Поэтому будем использовать «новую» методику контроля отрицательных остатков.

В решении будет использован справочник «Номенклатура» из каркасной конфигурации.

Для отражения хозяйственных операций требуются документы «Приходная накладная» и «Расходная накладная» (по тексту задачи). Данные документы также уже присутствуют в каркасной конфигурации. В этих документах уже есть необходимые для решения задачи реквизиты табличной части *Список номенклатуры*:

- *Номенклатура (СправочникСсылка.Номенклатура)*
- *Количество (Число 10, 0):*

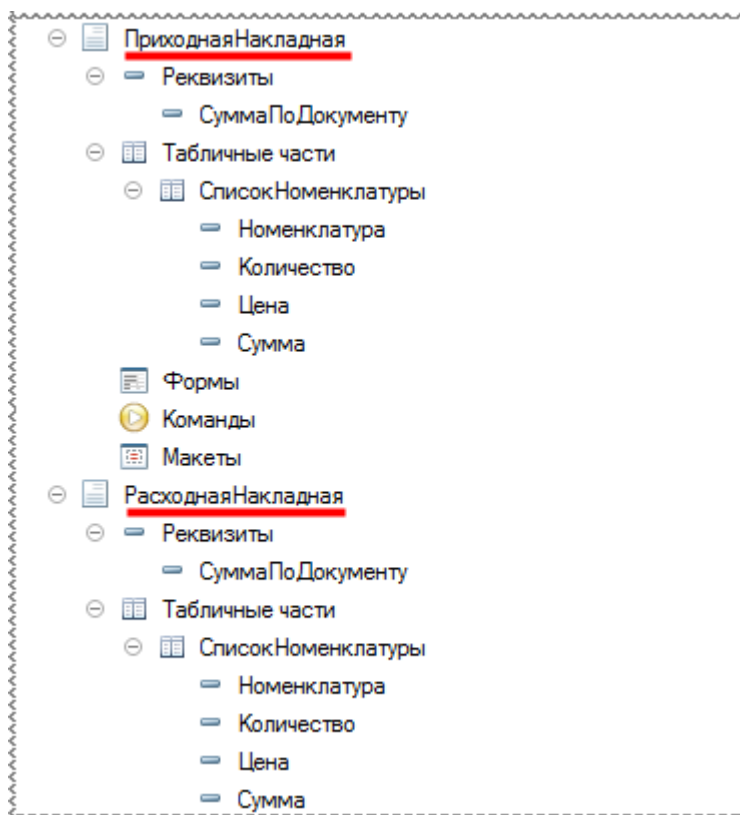


Рисунок 1 – Необходимые документы в дереве конфигурации

Для экономии времени на решение задачи создавать формы документов не будем.

Для хранения остатков будем использовать уже имеющийся в каркасной конфигурации регистр накопления «Остатки номенклатуры». Вид данного регистра – *Остатки*:

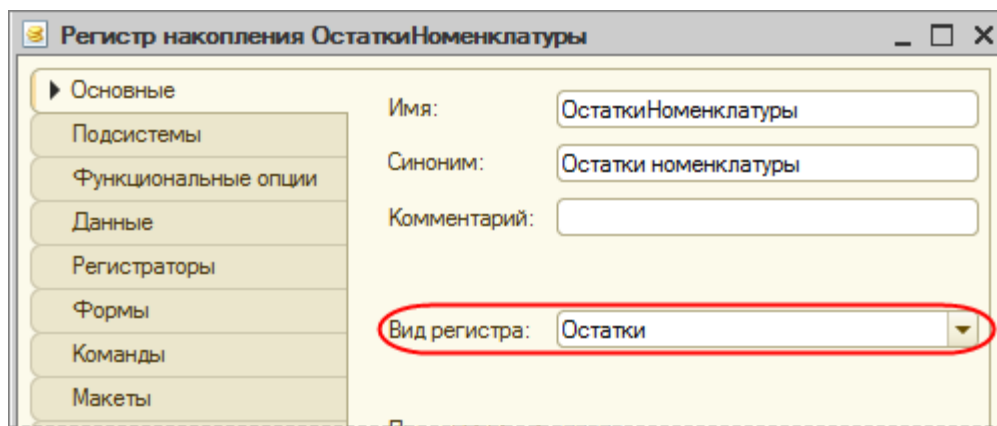


Рисунок 2 – Закладка Основные РН «Остатки номенклатуры»

Измерения регистра:

- Номенклатура (СправочникСсылка.Номенклатура).

Ресурсы:

- *Количество (Число 10, 0):*

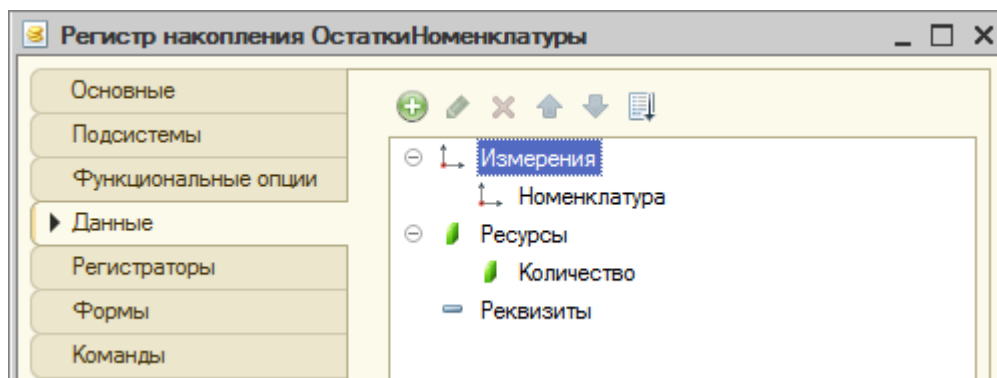


Рисунок 3 – Закладка *Данные* РН «Остатки номенклатуры»

Структура регистра полностью подходит для решения поставленной задачи. Единственное, что нужно изменить в структуре регистра, – это включить свойство «Запрет незаполненных значений» у измерения «Номенклатура». **Включение данного свойства необходимо, так как движения с пустой номенклатурой будут считаться учетной ошибкой.**

Регистраторы:

- «Приходная накладная»
- «Расходная накладная»:

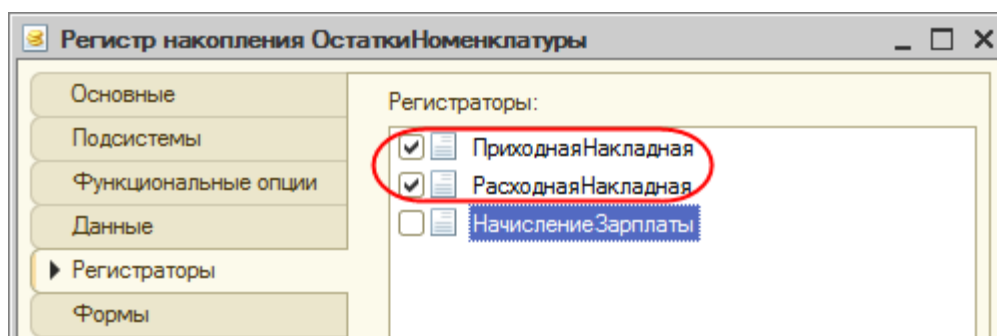


Рисунок 4 – Закладка *Регистраторы* РН «Остатки номенклатуры»

Обработка проведения для документа «Приходная накладная»

С помощью конструктора движений необходимо сформировать движения документа «Приходная накладная» по регистру накопления «Остатки номенклатуры» с видом *Приход*. Как это сделать, подробно рассмотрено в главе «6. Как реализовать поступление товаров в компанию».

Обработка проведения для документа «Расходная накладная»

Для документа «Расходная накладная» процедуру *ОбработкаПроведения* в модуле объекта создадим и заполним вручную:

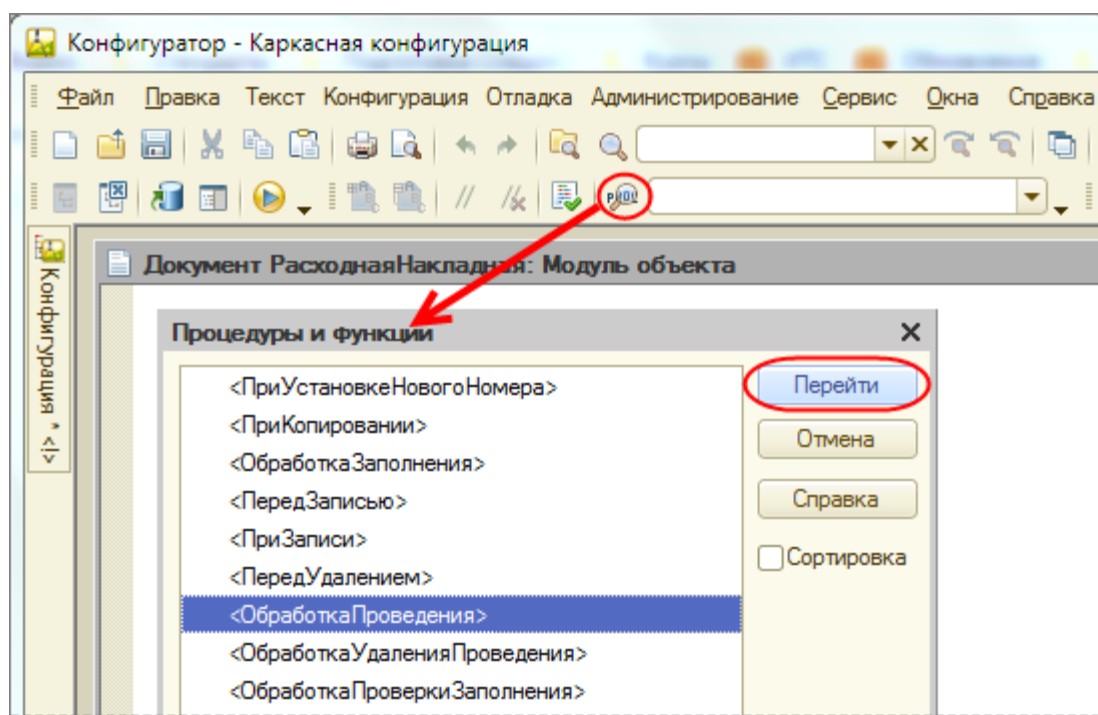


Рисунок 5 – Создание процедуры «ОбработкаПроведения»

Код обработки проведения будет выглядеть следующим образом:

Процедура *ОбработкаПроведения*(Отказ, РежимПроведения)

// 1. Установка маркера Записи у регистра

Движения.ОстаткиНоменклатуры.Записывать = Истина;

// 2. Инициализация менеджера временных таблиц

Запрос = Новый Запрос;

Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

// 3. Запрос, в котором группируются данные табличной части и подготавливается таблица
// значений для загрузки в коллекцию движений по регистру

Запрос.Текст =

"ВЫБРАТЬ

| РасходнаяНакладнаяСписокНоменклатуры.Номенклатура КАК Номенклатура,
| СУММА(РасходнаяНакладнаяСписокНоменклатуры.Количество) КАК Количество,
| &Период,
| &ВидДвижения

| ПОМЕСТИТЬ ТЧСписокНоменклатуры

| ИЗ

| Документ.РасходнаяНакладная.СписокНоменклатуры КАК

РасходнаяНакладнаяСписокНоменклатуры

| ГДЕ

| РасходнаяНакладнаяСписокНоменклатуры.Ссылка = &Ссылка

|

| СГРУППИРОВАТЬ ПО

| РасходнаяНакладнаяСписокНоменклатуры.Номенклатура

|

| ИНДЕКСИРОВАТЬ ПО

| Номенклатура

|;

|

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

| ВЫБРАТЬ

| ТЧСписокНоменклатуры.ВидДвижения,

| ТЧСписокНоменклатуры.Количество,

| ТЧСписокНоменклатуры.Номенклатура,

| ТЧСписокНоменклатуры.Период

| ИЗ

| ТЧСписокНоменклатуры КАК ТЧСписокНоменклатуры";

Запрос.УстановитьПараметр("Ссылка", Ссылка);

Запрос.УстановитьПараметр("ВидДвижения", ВидДвиженияНакопления.Расход);

Запрос.УстановитьПараметр("Период", Дата);

РезультатЗапроса = Запрос.Выполнить();

// 4. Загрузка таблицы значений в коллекцию движений

Движения.ОстаткиНоменклатуры.Загрузить(РезультатЗапроса.Выгрузить());

// 5. Управляемая блокировка данных регистра

Движения.ОстаткиНоменклатуры.БлокироватьДляИзменения = Истина;

// 6. Запись коллекции движений

Движения.Записать();

// 7. Запрос на получение отрицательных остатков

Запрос.Текст =

"ВЫБРАТЬ

```

|      ОстаткиНоменклатурыОстатки.Номенклатура КАК Номенклатура,
|      ОстаткиНоменклатурыОстатки.Номенклатура.Представление КАК
НоменклатураПредставление,
|      ОстаткиНоменклатурыОстатки.КоличествоОстаток КАК КоличествоОстаток
|ИЗ
|      РегистрНакопления.ОстаткиНоменклатуры.Остатки(
|          &ДатаОстатков,
|          Номенклатура В
|          (ВЫБРАТЬ
|              ТЧСписокНоменклатуры.Номенклатура
|              ИЗ
|              ТЧСписокНоменклатуры КАК ТЧСписокНоменклатуры)) КАК
ОстаткиНоменклатурыОстатки
|ГДЕ
|      ОстаткиНоменклатурыОстатки.КоличествоОстаток < 0";

// 8. Определение момента времени для получения остатков
Запрос.УстановитьПараметр("ДатаОстатков", Новый Граница(МоментВремени(),
ВидГраницы.Включая));

Результат = Запрос.Выполнить();

// 9. Если запрос не пустой, появились отрицательные остатки
Если НЕ Результат.Пустой() Тогда
    Отказ = Истина;

// 10. Выдача сообщений пользователю о наличии отрицательных остатков
Выборка = Результат.Выбрать();
Пока Выборка.Следующий() Цикл
    Сообщение = Новый СообщениеПользователю;
    Сообщение.Текст = "Недостаточно товара " +
Выборка.НоменклатураПредставление + ", не хватает: " + (-Выборка.КоличествоОстаток);
    Сообщение.Сообщить();
КонецЦикла;
КонецЕсли;

// 11. Есть ошибки, выходим из процедуры
Если Отказ Тогда
    Возврат;
КонецЕсли;

```

КонецПроцедуры

Рассмотрим ключевые точки алгоритма.

Установка маркера Записи у регистра (п.1)

Установим маркер *Записывать* в значение *Истина*, чтобы при использовании метода *Движения.Записать()* записались движения по регистру накопления «Остатки номенклатуры».

Важно. В конце обработки проведения система автоматически запишет движения, у которых установлен маркер *Записывать* в значение *Истина*. Поэтому не нужно в конце процедуры использовать метод *Движения.Записать()*, т.к. аналогичное действие выполняется платформой неявно.

Инициализация менеджера временных таблиц (п.2)

Менеджер будет необходим, чтобы созданная в запросе временная таблица была доступна и в следующих запросах. Таким образом, данные табличной части получаются один раз, сохраняются во временную таблицу и далее используются многократно.

Запрос (п.3)

Так как для проведения документа по регистру накопления «Остатки номенклатуры» все необходимые данные в документе присутствуют, реализуем «новую» методику контроля остатков.

Рассмотрим построение запроса с помощью конструктора запроса подробно.

В качестве источника данных выберем табличную часть *СписокНоменклатуры* документа «Расходная накладная». В список выбранных полей перенесем поля:

- *Номенклатура*
- *Количество*.

Чтобы в результате запроса получить сразу все данные, которые потребуются для последующей загрузки в коллекцию движений, в указанный список добавим параметры *Период* и *ВидДвижения*:

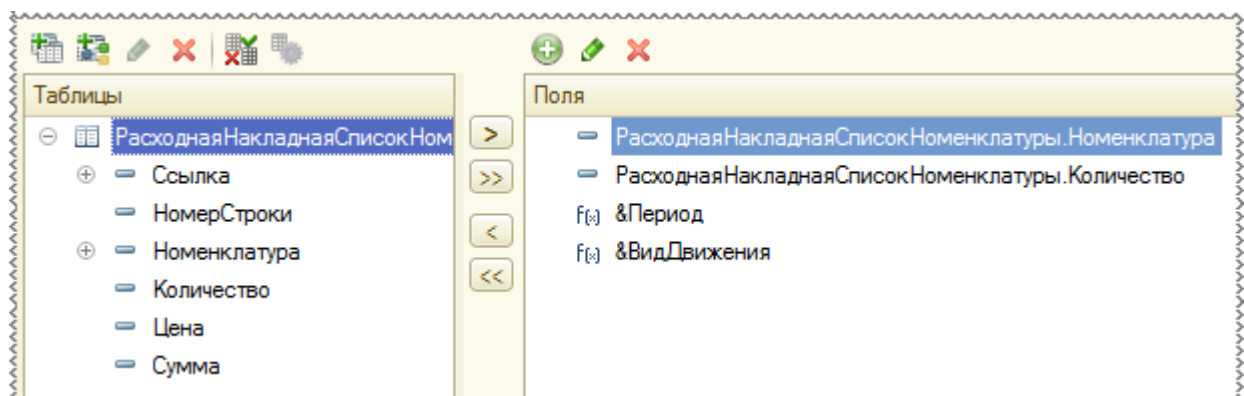


Рисунок 6 – Закладка *Таблицы* и *поля* конструктора запросов

Поскольку в табличной части *СписокНоменклатуры* может быть несколько строк с одинаковой номенклатурой, сгруппируем данные по полю *Номенклатура*, чтобы получить суммарное *Количество*:

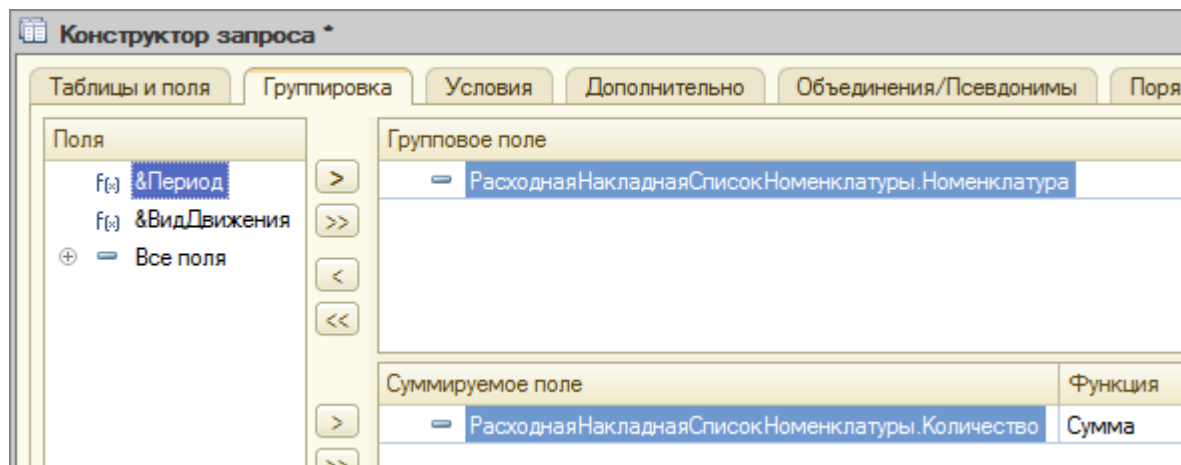


Рисунок 7 – Закладка *Группировка* конструктора запросов

Добавим условие отбора:

- Требуются данные табличной части *СписокНоменклатуры* только из текущего документа:

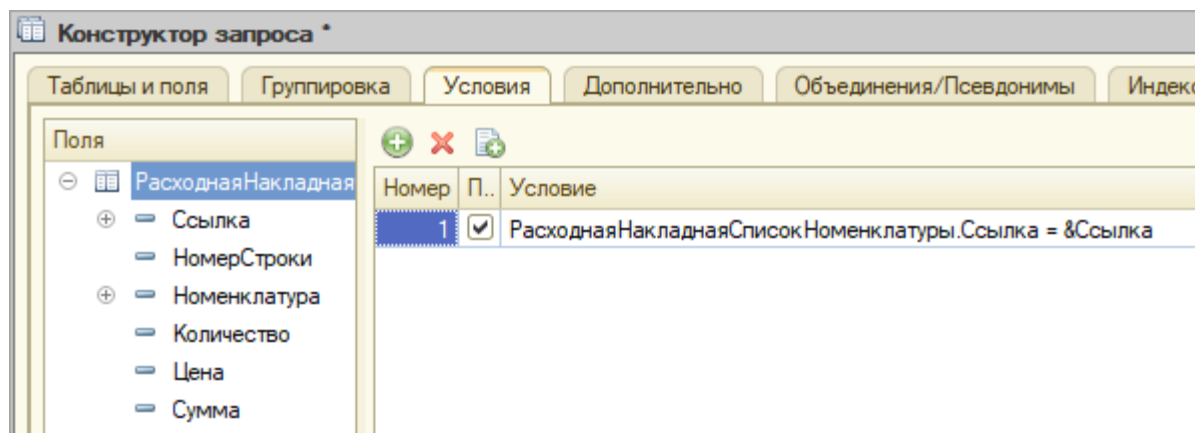


Рисунок 8 – Закладка *Условия* конструктора запросов

Поместим полученные данные во временную таблицу *ТЧСписокНоменклатуры*:

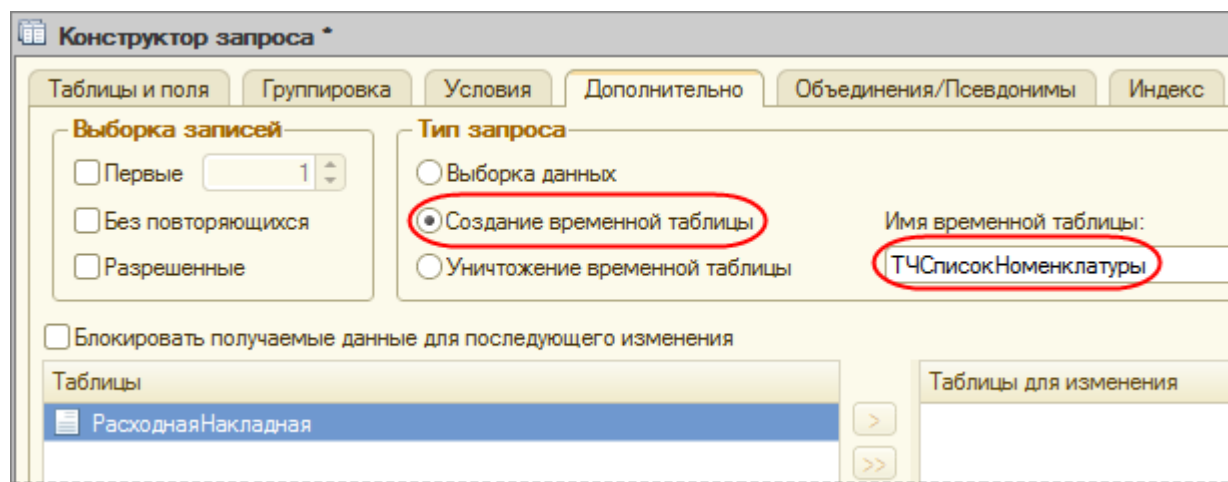


Рисунок 9 – Закладка *Дополнительно* конструктора запросов

В дальнейшем будем использовать временную таблицу *ТЧСписокНоменклатуры* в других запросах и выполнять соединение по полю Номенклатура. Поэтому для ускорения работы этих запросов добавим по данному полю индекс:

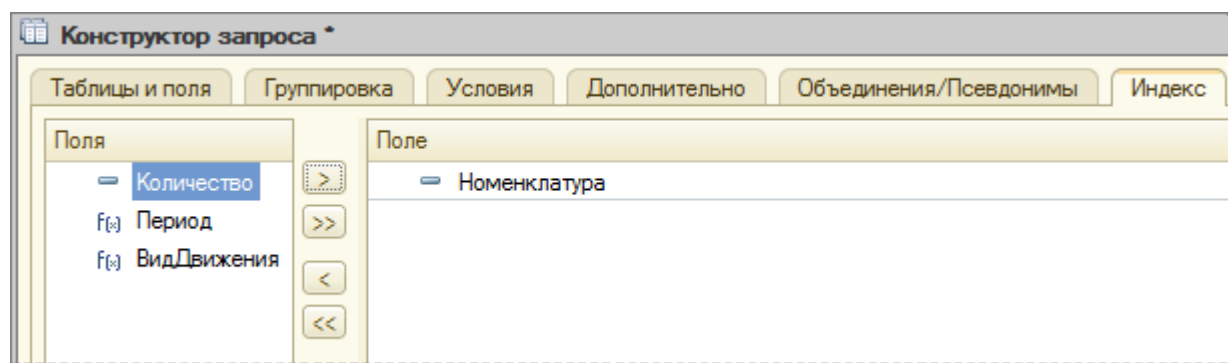


Рисунок 10 – Закладка *Индекс* конструктора запросов

Произведенные в конструкторе запроса настройки позволяют поместить результаты запроса во временную таблицу. Чтобы в результате выполнения запроса получить выборку, добавим в пакет еще один запрос, в котором выберем все поля из временной таблицы *ТЧСписокНоменклатуры*:

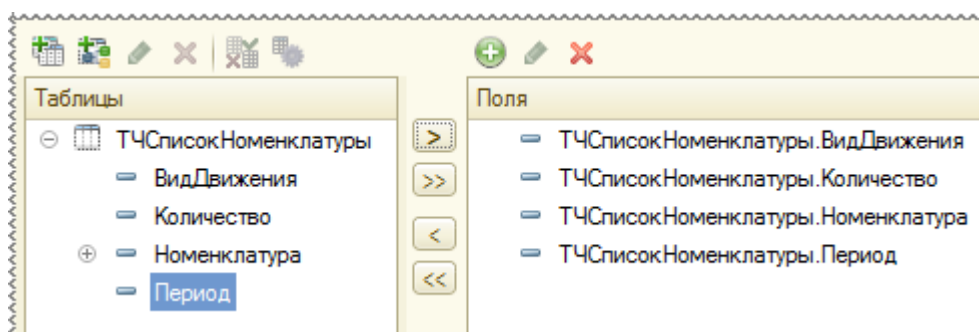


Рисунок 11 – Закладка *Таблицы и поля* конструктора запросов

Загрузка таблицы значений в коллекцию движений документа (п.4)

Преимущества использования метода *Загрузить()* у коллекции движений:

- Нет необходимости в начале процедуры делать предварительную очистку движений по регистру
- Загрузка таблицы значений в набор записей производительней, чем добавление движений в цикл. Данный подход используется в типовых конфигурациях.

Важно. Использование данного метода не является ошибкой на экзамене следующего типа:

Использование механизма соединения таблиц вместо того, чтобы задать значения параметров виртуальных таблиц	0,5 – 1,0
Выгрузка результата запроса в промежуточную таблицу (например, в таблицу значений) без необходимости	1,0
Дополнительный вопрос	0,5 - 1,0
Отсутствие возможности работы предлагаемого решения в режиме управляемого приложения	3,0

Рисунок 12 – Фрагмент из списка часто встречающихся ошибок

Управляемая блокировка данных регистра (п.5)

Установим признак необходимости блокировки данных регистра накопления «Остатки номенклатуры». Тема управляемых блокировок в рамках этого раздела будет рассмотрена несколько позже.

Запись коллекции движений (п.6)

Движения можно было записать, используя метод *Движения.ОстаткиНоменклатуры.Записать()*. Но более универсальным является метод *Движения.Записать()*, который записывает все движения с установленным маркером *Записывать* в значение *Истина*. После выполнения метода *Движения.Записать()* маркер *Записывать* у всех наборов сбросится в *Ложь*.

Запрос на получение отрицательных остатков (п.7)

Движения по регистру накопления «Остатки номенклатуры» сформированы и записаны. Проверим, появились ли после этого отрицательные остатки.

Для этого сформируем еще один запрос.

Рассмотрим построение запроса с помощью конструктора запроса подробно.

В качестве источника данных выберем виртуальную таблицу регистра накопления *ОстаткиНоменклатуры.Остатки*.

Остатки будем получать на дату, определенную в поле *Период* параметров виртуальной таблицы как *&ДатаОстатков*. Также в параметрах виртуальной таблицы определим условие отбора: выбранная номенклатура должна содержаться в табличной части *ТЧСписокНоменклатуры*:

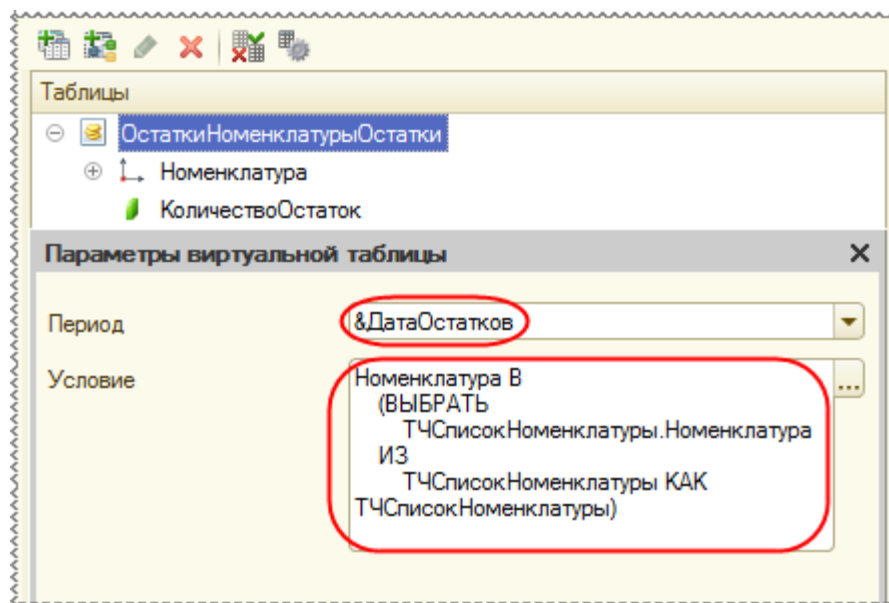


Рисунок 13 – Установка параметров виртуальной таблицы

В список выбранных полей перенесем поля:

- *Номенклатура*
- *Номенклатура.Представление* (для формирования текстовых сообщений)
- *КоличествоОстаток*:

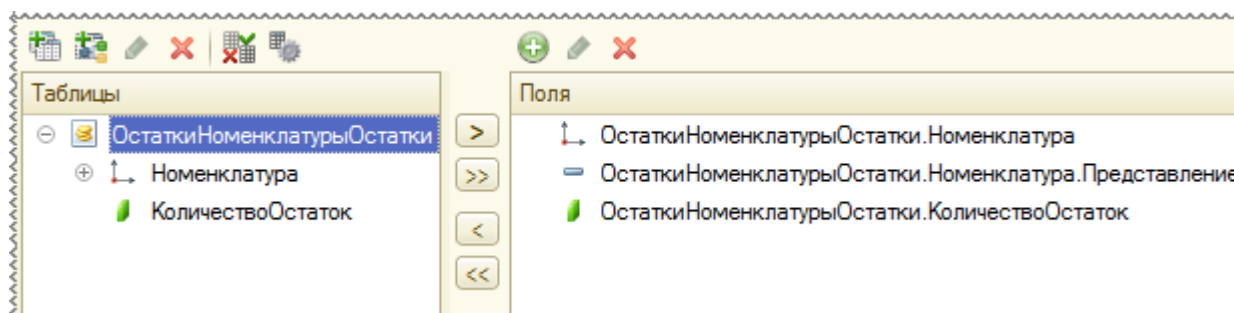


Рисунок 14 – Закладка *Таблицы и поля* конструктора запросов

Примечание: если использовать просто ссылку на элемент номенклатуры, а не его представление, то при выводе каждого очередного сообщения в цикле будет делаться запрос к информационной базе для получения представления. Это на экзамене является грубой ошибкой (получение данных базы в цикле – минус 2 балла):

Конфигурация должна устойчиво работать при наличии дублей строк (номенклатуры или сотрудников или т.п.) в документах.	0,5
Необходимо обеспечить корректное проведение документов при этом.	
Получение информации, хранящейся в информационной базе, (остатков, оборотов, данных базы, данных графика и т.п.) в цикле	2,0
Отсутствие значений параметров в виртуальной таблице или использование вместо них условия «ГДЕ»	2,0

Рисунок 15 – Фрагмент из списка часто встречающихся ошибок

При наличии в результатах запроса представления будет сразу выводиться готовый текст.

Важно. Добавим условие отбора по количеству остатков номенклатуры меньше нуля, то есть будем контролировать наличие отрицательных остатков товаров:

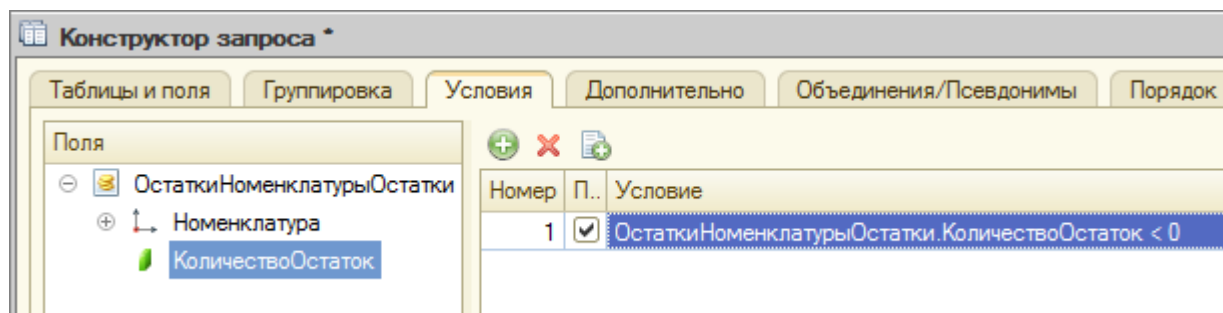


Рисунок 16 – Закладка *Условия* конструктора запросов

Определение момента времени для получения остатков (п.8)

Установим значение параметра запроса *ДатаОстатков*. Так как нужно получить остатки с учетом движений текущего документа, будем передавать границу, включающую момент времени документа.

МоментВремени и объект *Граница* подробно рассмотрены в главе «3. Что такое момент времени и для чего он используется».

Важно. На экзамене нельзя использовать актуальные остатки (в качестве параметра *ДатаОстатков* передается Неопределенно) и конструкцию:

```
МоментВремениОстатков = ?(РежимПроведения = РежимПроведенияДокумента.Оперативный, Неопределено,
Новый Граница(МоментВремени(), ВидГраницы.Включая));
```

```
Запрос.УстановитьПараметр("ДатаОстатков", МоментВремениОстатков)
```

На экзамене это считается ошибкой и из оценки вычитается 1 балл:

В задачах получения данных из информационной базы установка отборов по неиндексированным полям	0,5
Если при проведении документа используются каким-то образом данные, считываемые из регистров, обязательно требуется предусмотреть получение таких данных на момент проведения документа	1,0
Конфигурация должна устойчиво работать не только при движении вперед, но и назад.	1,0

Рисунок 17 – Фрагмент из списка часто встречающихся ошибок

Какие проблемы могут быть при использовании актуальных остатков, рассмотрим в отдельной главе.

Если запрос не пустой, появились отрицательные остатки (п.9)

Проверим, есть ли в результате запроса записи. Если записи есть, значит присутствуют отрицательные остатки. Присвоим переменной *Отказ* значение *Истина*.

Выдача сообщений пользователю о наличии отрицательных остатков (п.10)

В цикле обойдем все отрицательные остатки и выведем сообщения пользователю. В сообщениях будет передаваться информация, какой номенклатуры не хватает и в каком количестве.

Есть ошибки, выходим из процедуры (п.11)

При наличии отрицательных остатков выполним возврат. Смысла выполнять движения по другим регистрам нет.

Задача решена, контроль остатков выполняется по «новой» методике.

9. Как выполнить контроль остатков по старой методике

Постановка задачи:

*Компания занимается оптовой торговлей товаров. Поступление товаров отражается документом «Приходная накладная», продажа – «Расходная накладная». Учет товаров в разрезе складов не ведется. При проведении расходной накладной при нехватке товара необходимо выдавать соответствующее предупреждение с указанием количества нехватки и не позволять проводить документ. **При продаже себестоимость товара рассчитывается как средняя по всей компании в целом.***

Всего одно предложение о себестоимости товаров лишает возможности использовать «новую» методику контроля остатков. Дело в том, что в документе «Расходная накладная» данные о себестоимости товара отсутствуют. Эти данные должны быть получены путем обращения к базе данных для расчетов себестоимости списания. Поэтому в указанном примере следует использовать «старую» методику контроля остатков.

Для решения поставленной задачи в конфигурации должны быть созданы объекты:

- справочник «Номенклатура»
- документ «Приходная накладная»
- документ «Расходная накладная»
- регистр накопления «Остатки номенклатуры».

Так как документ «Приходная накладная» только увеличивает остатки товаров, то контроль остатков при проведении этого документа не требуется.

В решении будет использован справочник «Номенклатура» из каркасной конфигурации.

Для отражения хозяйственных операций требуются документы «Приходная накладная» и «Расходная накладная» (по тексту задачи). Данные документы также уже присутствуют в каркасной конфигурации. В этих документах уже присутствуют необходимые для решения задачи реквизиты табличной части *Список номенклатуры*:

- Номенклатура (СправочникСсылка.Номенклатура)
- Количество (Число 10, 0)
- Сумма (Число 12, 2):

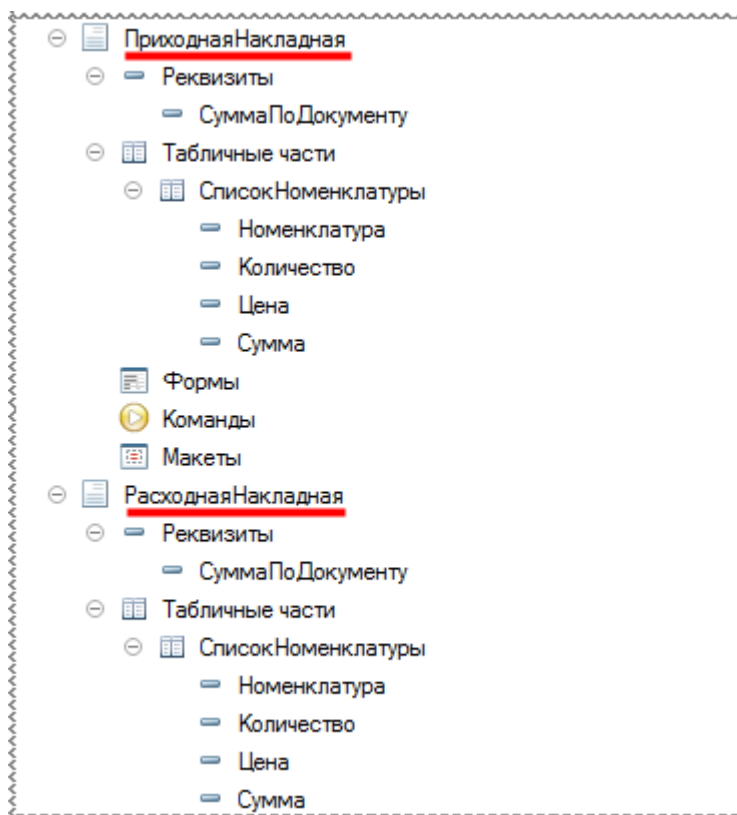


Рисунок 1 – Необходимые документы в дереве конфигурации

Для экономии времени на решение задачи создавать формы документов не будем.

Из условия задачи следует, что учет остатков и себестоимости товаров ведется по предприятию в целом. Нет дополнительного учета в разрезе складов, заказов и т.д., то есть аналитика учета остатков и себестоимости товаров совпадает. Поэтому для хранения остатков будем использовать уже имеющийся в каркасной конфигурации регистр накопления «Остатки номенклатуры». Вид данного регистра – *Остатки*:

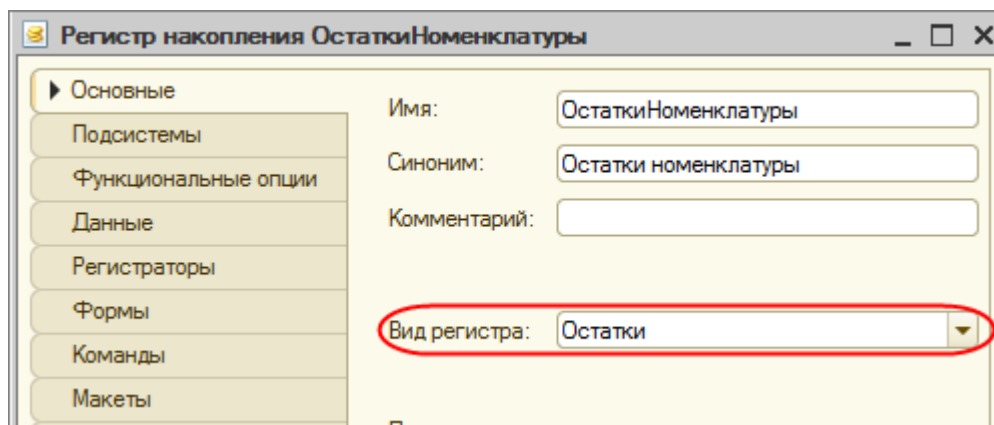


Рисунок 2 – Закладка Основные РН «Остатки номенклатуры»

Измерения регистра:

- Номенклатура (СправочникСсылка.Номенклатура).

Ресурсы:

- Количество (Число 10, 0).

Для решения учебной задачи добавим ресурс Сумма (Число 12, 2):

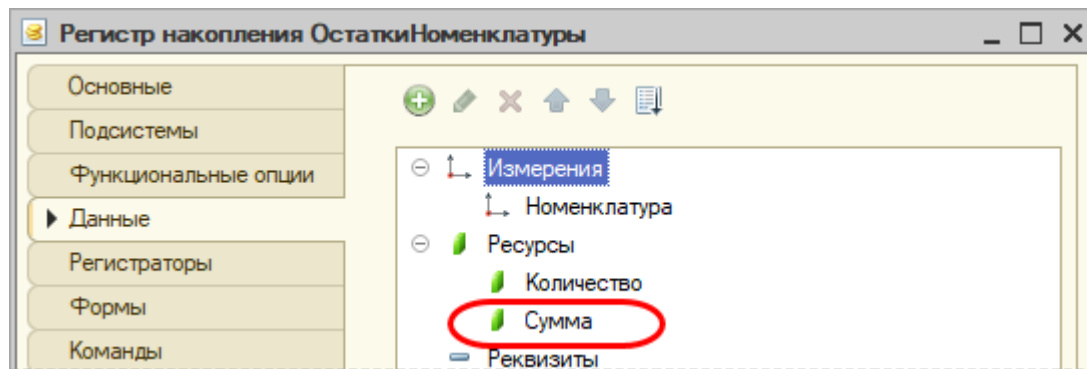


Рисунок 3 – Закладка *Данные* РН «Остатки номенклатуры»

У измерения «Номенклатура» включим свойство «Запрет незаполненных значений». Это необходимо, так как движения с пустой номенклатурой будут считаться учетной ошибкой.

Регистраторы:

- «Приходная накладная»
- «Расходная накладная»:

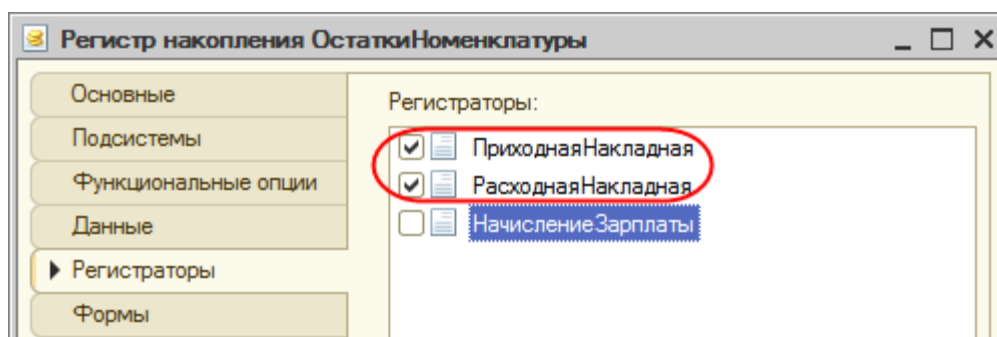


Рисунок 4 – Закладка *Регистраторы* РН «Остатки номенклатуры»

Обработка проведения для документа «Приходная накладная»

С помощью конструктора движений необходимо сформировать движения документа «Приходная накладная» по регистру накопления «Остатки номенклатуры» с видом *Приход*. Как это сделать, подробно рассмотрено в главе «**6. Как реализовать поступление товаров в компанию**».

Дополнительно для данной задачи необходимо заполнить ресурс Сумма:

Конструктор движения регистров

Регистры

- РегистрНакопления.ОстаткиНоменклатуры

Тип движения регистра: ☒ Приход ☐ Расход

Табличная часть: Список Номенклатуры

Поле	Выражение
Номенклатура	ТекСтрокаСписокНоменклатуры.Номенклатура
Количество	ТекСтрокаСписокНоменклатуры.Количество
Сумма	ТекСтрокаСписокНоменклатуры.Сумма

Реквизиты документа

- Дата
- Номер
- СуммаПоДокументу
- ТекСтрокаСписокНоменклатуры.НомерСтроки
- ТекСтрокаСписокНоменклатуры.Номенклатура
- ТекСтрокаСписокНоменклатуры.Количество
- ТекСтрокаСписокНоменклатуры.Цена

Назад Далее

Заполнить выражения

Очистить выражения

OK

Отмена

Справка

Рисунок 5 – Формирование движений конструктором

Обработка проведения для документа «Расходная накладная»

Для документа «Расходная накладная» процедуру *ОбработкаПроведения* в модуле объекта создадим и заполним вручную:

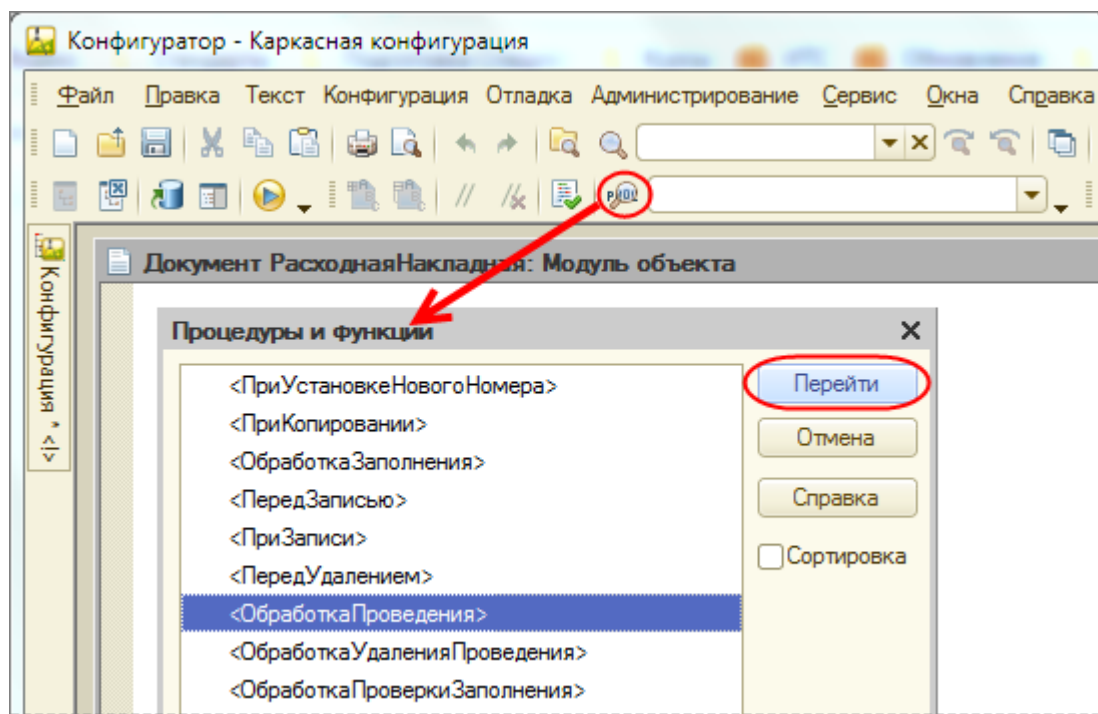


Рисунок 6 – Создание процедуры «ОбработкаПроведения»

Код обработки проведения будет выглядеть следующим образом:

Процедура ОбработкаПроведения(Отказ, РежимПроведения)

```
// 1. Подготовка наборов записей регистра
Движения.ОстаткиНоменклатуры.Записывать = Истина;
Движения.Записать();

// 2. Восстановление для свойства набора движений Записывать значения Истина
Движения.ОстаткиНоменклатуры.Записывать = Истина;

// 3. Управляемая блокировка данных регистра
Блокировка = Новый БлокировкаДанных;
Элемент = Блокировка.Добавить("РегистрНакопления.ОстаткиНоменклатуры");
Элемент.Режим = РежимБлокировкиДанных.Исключительный;
Элемент.ИсточникДанных = СписокНоменклатуры;
Элемент.ИспользоватьИзИсточникаДанных("Номенклатура", "Номенклатура");

Блокировка.Заблокировать();

// 4. Запрос, в котором группируются данные табличной части и получают остатки
Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
|     РасходнаяНакладнаяСписокНоменклатуры.Номенклатура КАК Номенклатура,
|     СУММА(РасходнаяНакладнаяСписокНоменклатуры.Количество) КАК Количество
|ПОМЕСТИТЬ ТЧСписокНоменклатуры
|ИЗ
```



```

|      Документ.РасходнаяНакладная.СписокНоменклатуры КАК
РасходнаяНакладнаяСписокНоменклатуры
|ГДЕ
|      РасходнаяНакладнаяСписокНоменклатуры.Ссылка = &Ссылка
|
|СГРУППИРОВАТЬ ПО
|      РасходнаяНакладнаяСписокНоменклатуры.Номенклатура
|
|ИНДЕКСИРОВАТЬ ПО
|      Номенклатура
|;
|
|////////////////////////////////////
|ВЫБРАТЬ
|      ТЧСписокНоменклатуры.Номенклатура КАК Номенклатура,
|      ТЧСписокНоменклатуры.Номенклатура.Представление КАК НоменклатураПредставление,
|      ТЧСписокНоменклатуры.Количество КАК Количество,
|      ЕСТЬNULL(ОстаткиНоменклатурыОстатки.КоличествоОстаток, 0) КАК КоличествоОстаток,
|      ОстаткиНоменклатурыОстатки.СуммаОстаток КАК СуммаОстаток
|ИЗ
|      ТЧСписокНоменклатуры КАК ТЧСписокНоменклатуры
|          ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ОстаткиНоменклатуры.Остатки(
|              &МоментВремени,
|              Номенклатура В
|                  (ВЫБРАТЬ
|                      ТЧСписокНоменклатуры.Номенклатура
|                      ИЗ
|                          ТЧСписокНоменклатуры КАК ТЧСписокНоменклатуры)) КАК
ОстаткиНоменклатурыОстатки
|          ПО ТЧСписокНоменклатуры.Номенклатура =
ОстаткиНоменклатурыОстатки.Номенклатура";

```

// 5. Определение момента времени для получения остатков

```

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

```

```

Результат = Запрос.Выполнить();

```

// 6. Цикл по номенклатуре документа

```

Выборка = Результат.Выбрать();

```

Пока Выборка.Следующий() Цикл

// 7. Контроль остатков

```

Нехватает = Выборка.Количество - Выборка.КоличествоОстаток;

```

Если Нехватает > 0 Тогда

// 8. Выдача сообщений пользователю о наличии отрицательных остатков

```

Сообщение = Новый СообщениеПользователю;

```

```

Сообщение.Текст = "Недостаточно товара " + Выборка.НоменклатураПредставление + ",

```

не хватает: " + Нехватает;

```

Сообщение.Сообщить();

```

```

        Отказ = Истина;
    КонечЕсли;

    Если Отказ Тогда
        Продолжить;
    КонечЕсли;

    // 9. Расчет суммы для списания
    Если Выборка.Количество = Выборка.КоличествоОстаток Тогда
        Себестоимость = Выборка.СуммаОстаток;
    Иначе
        Себестоимость = Выборка.Количество * Выборка.СуммаОстаток /
        Выборка.КоличествоОстаток;
    КонечЕсли;

    // 10. Формирование движения
    Движение = Движения.ОстаткиНоменклатуры.ДобавитьРасход();
    Движение.Период = Дата;
    Движение.Количество = Выборка.Количество;
    Движение.Номенклатура = Выборка.Номенклатура;
    Движение.Сумма = Себестоимость;

    КонечЦикла;

КонечПроцедуры

```

Подготовка наборов записей регистра (п.1)

Для документа «РасходнаяНакладная» установлен режим удаления движений – «Удалять автоматически при отмене проведения»:

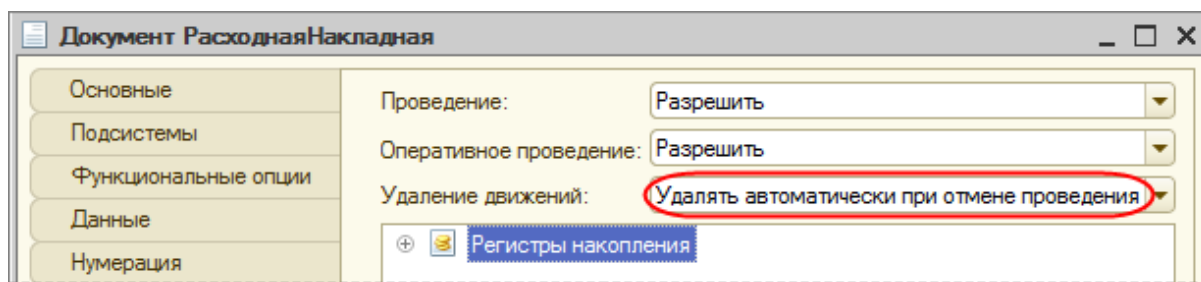


Рисунок 7 – Закладка *Движения* документа «Расходная накладная»

При такой настройке в случае перепроведения документа движения автоматически удаляться не будут, при обращении к регистру накопления «ОстаткиНоменклатуры» могут учитываться старые

движения самого документа. Это происходит, когда дата документа при перепроведении сдвигается вперед.

Для очистки старых движений документа используем принудительную запись пустого набора. В этом случае при чтении данных из регистра старые движения документа в расчет виртуальных таблиц не попадут.

Важно:

1. Использование режима удаления движений «Удалять автоматически» на экзамене считается ошибкой, так как в этом случае платформа записывает пустые наборы записей в регистры автоматически. Причем это происходит еще до начала выполнения процедуры *ОбработкаПроведения* и приводит к преждевременным блокировкам
2. Ситуация с учетом старых движений документа может возникнуть только в случае сдвига даты документа вперед. Поэтому при решении реальных задач очистку движений рекомендуется делать только в этом случае
3. Движения можно записать, используя метод *Движения.ОстаткиНоменклатуры.Записать()*. Но более универсальным является метод *Движения.Записать()*. Этот метод записывает сразу все наборы движений по регистрам, для которых свойство *Записывать* установлено в значение *Истина*. После выполнения метода *Движения.Записать()* для всех наборов движений свойство *Записывать* примет значение *Ложь*. При этом для наборов движений, которые нужно записать в конце обработки проведения, требуется заново установить значение *Истина* для свойства *Записывать*.

Восстановление для свойства «Записывать» набора движений значения «Истина» (п.2)

Установим для свойства *Записывать* набора движений по регистру накопления «ОстаткиНоменклатуры» значение *Истина*, чтобы сформированные движения записались в конце алгоритма проведения.

Важно. В конце обработки проведения система автоматически записывает наборы движений, для которых свойство *Записывать* установлено в значение *Истина*. В конце процедуры использовать метод *Движения.Записать()* не требуется.

Управляемая блокировка данных регистра (п. 3)

Установим управляемую блокировку на данные регистру накопления «ОстаткиНоменклатуры» по номенклатуре. Тема управляемых блокировок в рамках этого раздела будет рассмотрена несколько позже.

Запрос (п. 4)

Необходимые данные будем получать запросом.

Рассмотрим процесс создания пакета запросов с помощью конструктора запроса подробно.

В качестве источника данных выберем табличную часть *СписокНоменклатуры* документа «Расходная накладная». В список выбранных полей перенесем поля *Номенклатура* и *Количество*:

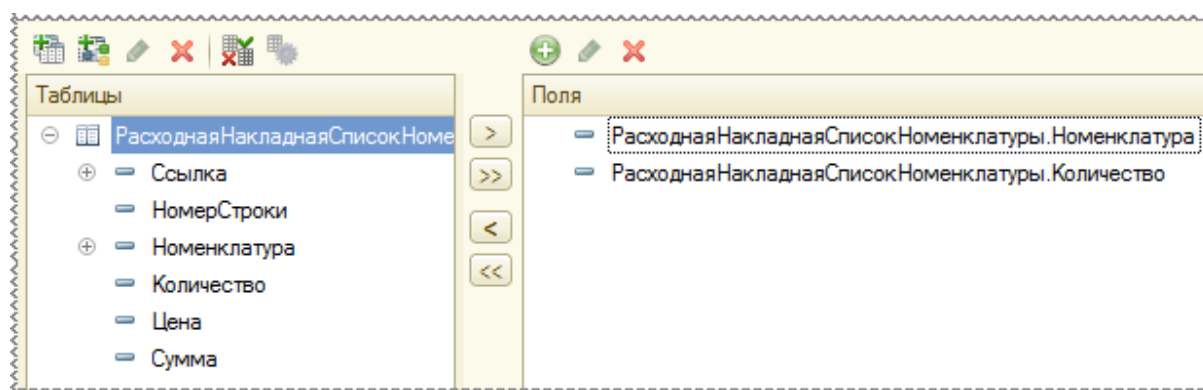


Рисунок 8 – Закладка Таблицы и поля конструктора запросов

Поскольку в табличной части *СписокНоменклатуры* может быть несколько строк с одинаковой номенклатурой, сгруппируем данные по полю *Номенклатура*, чтобы получить суммарное *Количество*:

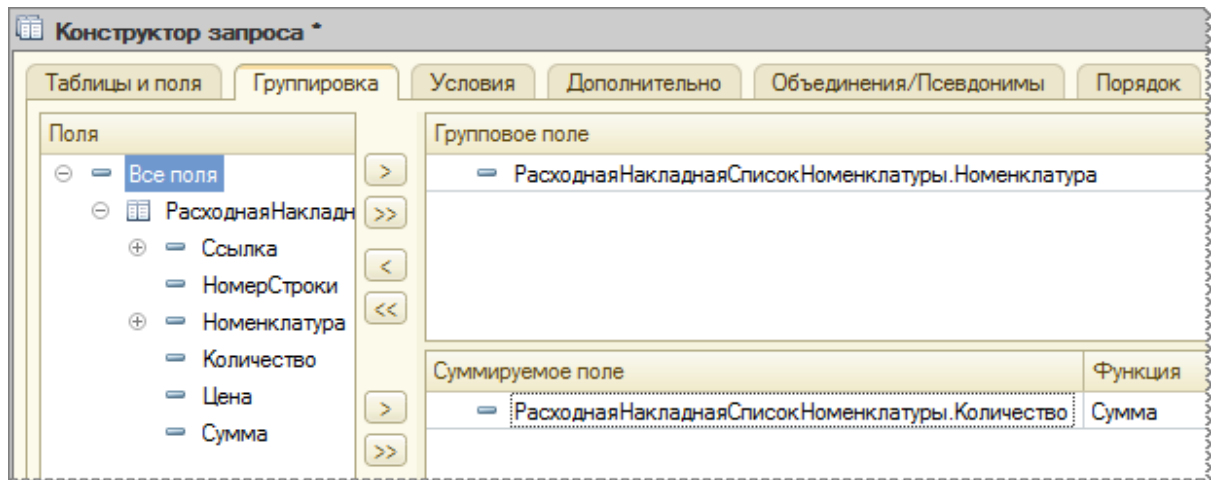


Рисунок 9 – Закладка Группировка конструктора запросов

Добавим условие отбора: требуются данные табличной части *СписокНоменклатуры* только из текущего документа:

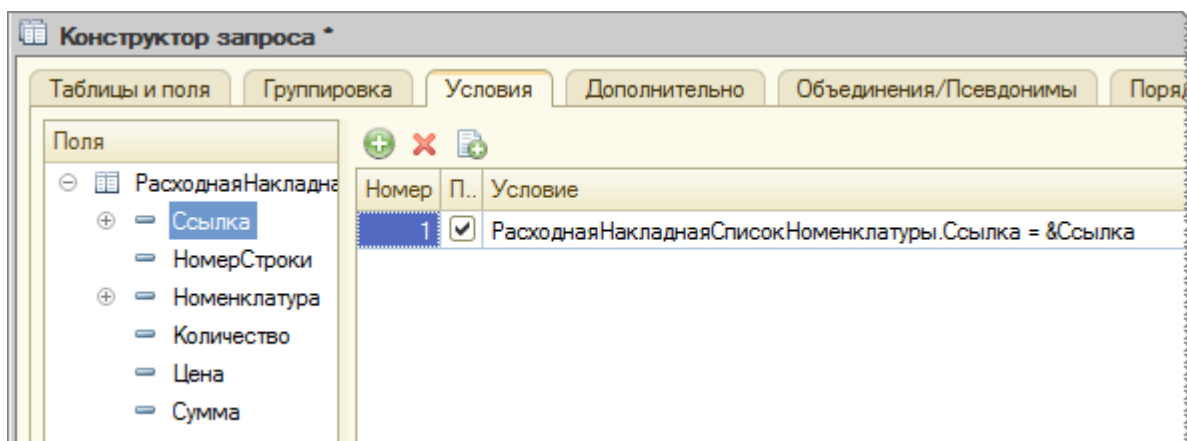


Рисунок 10 – Закладка Условия конструктора запросов

Поместим полученные данные во временную таблицу *ТЧСписокНоменклатуры*:

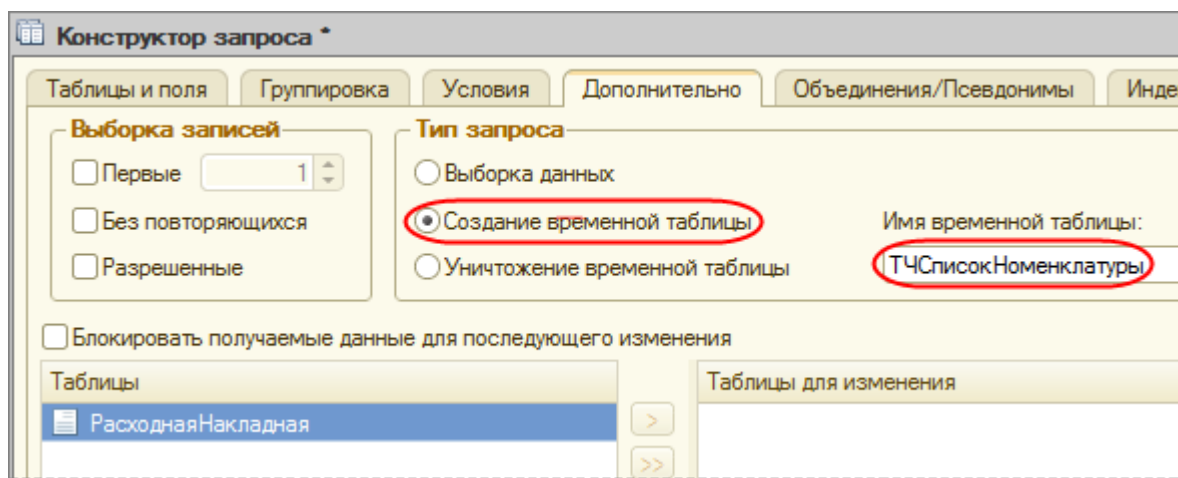


Рисунок 11 – Закладка Дополнительно конструктора запросов

В дальнейшем будем использовать временную таблицу *ТЧСписокНоменклатуры* в других запросах и выполнять соединение по полю Номенклатура. Поэтому для ускорения работы этих запросов добавим по данному полю индекс:

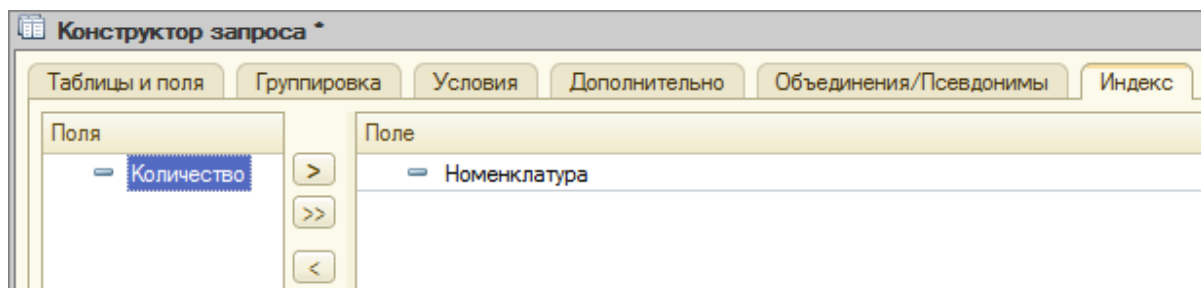


Рисунок 12 – Закладка Индекс конструктора запросов

Произведенные в конструкторе запроса настройки позволяют поместить результаты запроса во временную таблицу. Добавим в пакет еще один запрос, а в качестве источника данных для запроса выберем временную таблицу *ТЧСписокНоменклатуры* и виртуальную таблицу регистра накопления *ОстаткиНоменклатуры.Остатки*.

Остатки будем получать на дату, определенную в поле *Период* параметров виртуальной таблицы как *&МоментВремени*. Для обеспечения отбора в параметрах виртуальной таблицы *ОстаткиНоменклатуры.Остатки* также определим:

- Условие, обеспечивающее отбор записей, в которых *Номенклатура* содержится в табличной части *ТЧСписокНоменклатуры*:

Номенклатура В

(ВЫБРАТЬ

ТЧСписокНоменклатуры.Номенклатура

из

ТЧСписокНоменклатуры КАК ТЧСписокНоменклатуры)

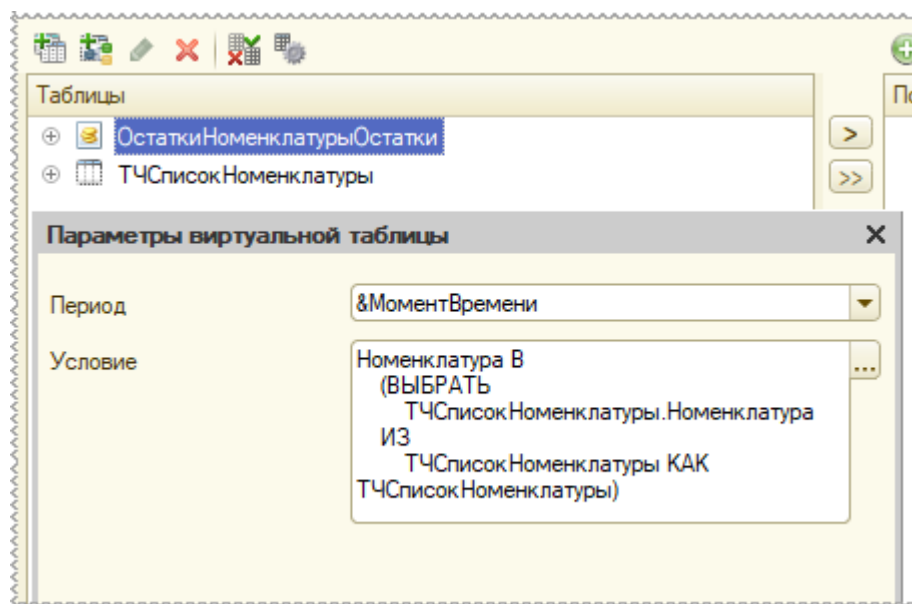


Рисунок 13 – Установка параметров виртуальной таблицы

В список выбранных полей перенесем поля:

- из временной таблицы ТЧСписокНоменклатуры:
 - Номенклатура
 - Номенклатура.Представление (для формирования текстовых сообщений)
 - Количество
- из виртуальной таблицы ОстаткиНоменклатуры.Остатки:
 - КоличествоОстаток (если данное поле не заполнено, количество равно нулю)
 - СуммаОстаток:

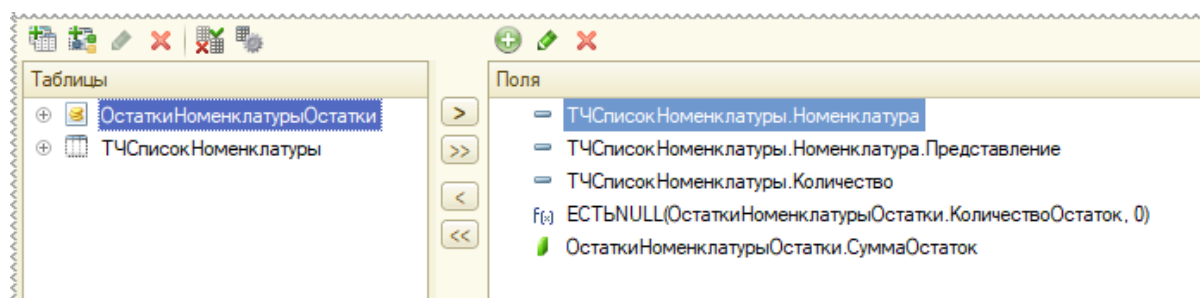


Рисунок 14 – Закладка Таблицы и поля конструктора запросов

Установим связь между таблицами: ЛЕВОЕ СОЕДИНЕНИЕ с таблицей ОстаткиНоменклатуры.Остатки по полю Номенклатура:

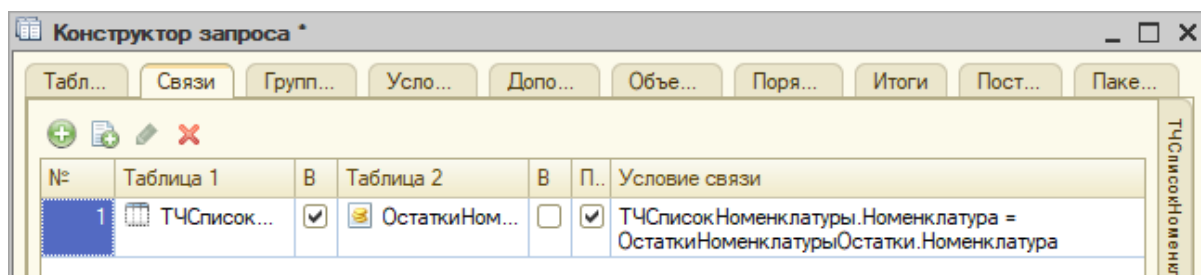


Рисунок 15 – Закладка *Связи* конструктора запросов

Определение момента времени для получения остатков (п.5)

Установим значение параметра запроса *МоментВремени*. Будем передавать момент времени документа.

Момент времени подробно рассмотрен в главе «3. Что такое момент времени и для чего он используется».

Важно: получение остатков на дату документа будет считаться грубейшей ошибкой на экзамене. Также на экзамене нельзя использовать актуальные остатки (в качестве параметра *МоментВремени* передается Неопределенно) и конструкцию:

```
МоментВремениОстатков = ?(РежимПроведения = РежимПроведенияДокумента.Оперативный, Неопределено, МоментВремени());
```

```
Запрос.УстановитьПараметр("МоментВремени", МоментВремениОстатков)
```

На экзамене это считается ошибкой и из оценки вычитается 1 балл:

В задачах получения данных из информационной базы установка отборов по неиндексированным полям	0,5
Если при проведении документа используются каким-то образом данные, считываемые из регистров, обязательно требуется предусмотреть получение таких данных на момент проведения документа	1,0
Конфигурация должна устойчиво работать не только при движении вперед, но и назад.	1,0

Рисунок 17 – Фрагмент из списка часто встречающихся ошибок

Какие проблемы могут быть при использовании актуальных остатков, рассмотрим в отдельной главе.

Цикл по номенклатуре документа (п.6)

В цикле проверяем достаточно ли номенклатуры для списания. Если достаточно, формируем движение, иначе устанавливаем *Отказ* в значение *Истина*.

Контроль остатков (п.7)

Вычисляем дефицит номенклатуры по простой формуле: *Количество из документа* – *Количество остаток*. Если получаем положительное число, то количество в документе превышает остаток.

Выдача сообщений пользователю о наличии отрицательных остатков (п.8)

При положительном дефиците выведем сообщение пользователю. В сообщении будет передаваться информация, какой номенклатуры не хватает и в каком количестве.

Расчет суммы для списания (п.9)

Если количество по документу равно остатку по регистру, то себестоимость не рассчитываем, а берем сумму из остатка. Иначе рассчитываем себестоимость по простейшей формуле согласно пропорции.

Важно. Проверку на равенство количества по документу и остатка по регистру на экзамене следует делать обязательно. За ее отсутствие проверяющий может снизить оценку на 0,5- 2,0 балла:

Отсутствие в решении проверок на правильное заполнение ресурсов регистра, приводящее, например, к появлению отрицательных остатков товаров на складе. Наличие отрицательных значений ресурсов регистра допустимо, только если об этом явно сказано в задании или следует из логики учетной схемы, не противоречащей ситуации, возникающей в реальной практике ведения учета	1,0 - 2,0
Использование неправильных или упрощенных алгоритмов при расчете значений ресурсов регистра. Например, при решении «проблемы копеек»	0,5 - 2,0
Построенная в решении учетная схема не обеспечивает правильного занесения данных в регистры. Например, необходимо списать 1000, а списывается 500	2,0

Рисунок 16 – Фрагмент из списка часто встречающихся ошибок

Формирование движения (п.10)

Себестоимость рассчитана, формируем движение по регистру накопления «ОстаткиНоменклатуры».

Важно. В конце процедуры не следует использовать метод `Движения.Записать()`. В этом нет необходимости, так как запись будет произведена платформой автоматически при завершении процедуры проведения документа.

Задача решена, контроль остатков выполняется по «старой» методике.

Подведем итоги: В нескольких квантах был рассмотрен контроль остатков по «старой» и «новой» методике. Рекомендуем всегда, если это технически возможно, использовать «новую» методику контроля остатков, так как она более эффективна. «Старую» методику нужно использовать только тогда, когда «новую» методику технически выполнить невозможно.

Обратите внимание, что оба варианта контроля остатков подразумевают выполнение запросов к БД в процедуре обработки проведения документов. Работа с запросами является неотъемлемой частью программирования в среде «1С:Предприятие» и обязательным навыком для экзаменуемого на специалиста по платформе. В рамках данного курса провести подробное изучение механизма запросов, к сожалению, нет возможности. Если необходима дополнительная информация по работе с запросами, то найти ее можно в курсе «[Разработка и оптимизация запросов в 1С:Предприятие 8.3](#)».

10. Что такое актуальные остатки и возможные проблемы при их использовании

В данной главе рассмотрим, что такое актуальные остатки и какие проблемы могут возникнуть при их использовании. Все эксперименты будем проводить на конфигурации, разработанной в главе «9. Как выполнить контроль остатков по старой методике».

Что такое актуальные остатки

Рассмотрим, что такое актуальные остатки на примере регистра накопления *ОстаткиНоменклатуры*.

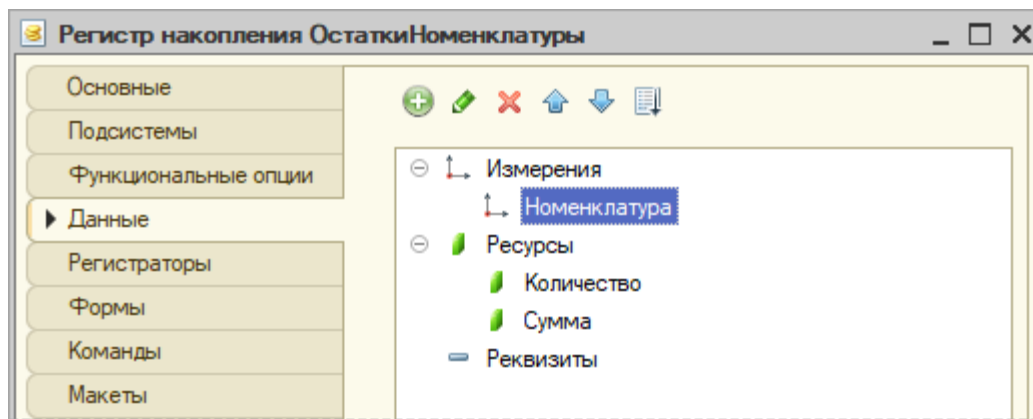


Рисунок 1 – Структура регистра накопления *ОстаткиНоменклатуры*

Регистр накопления остатков состоит из двух таблиц:

- Таблица движений
- Таблица итогов.

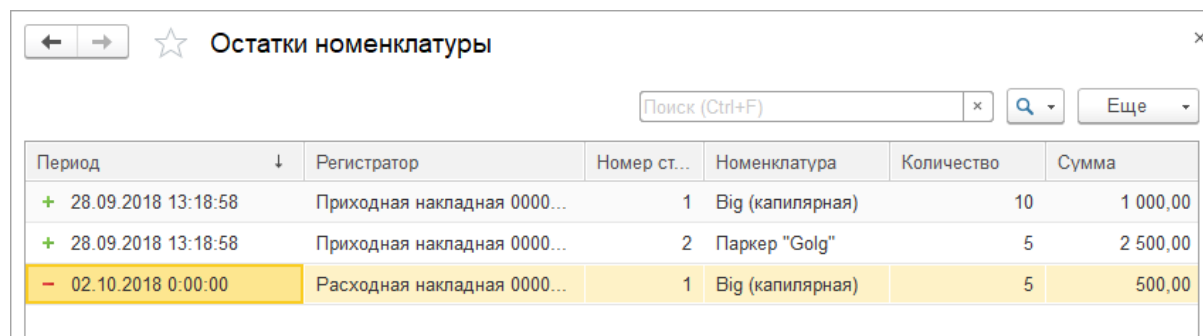
Таблица движений

В таблице движений хранятся записи, сформированные при проведении документов. Таблица движений состоит из следующих полей:

- Вид движения (Приход, Расход)

- Период
- Регистратор
- Номер строки
- Все измерения
- Все ресурсы
- Все реквизиты.

Чтобы увидеть, как выглядит таблица движений, достаточно открыть регистр накопления в пользовательском режиме.



Период	Регистратор	Номер ст...	Номенклатура	Количество	Сумма
+ 28.09.2018 13:18:58	Приходная накладная 0000...	1	Big (капиллярная)	10	1 000,00
+ 28.09.2018 13:18:58	Приходная накладная 0000...	2	Паркер "Golg"	5	2 500,00
- 02.10.2018 0:00:00	Расходная накладная 0000...	1	Big (капиллярная)	5	500,00

Рисунок 2 – Таблица движений регистра накопления *ОстаткиНоменклатуры*

Таблица итогов

В таблице итогов хранятся остатки в разрезе всех измерений с периодичностью месяц (на начало месяца) и остатки на крайнюю дату – 01.11.3999. И это не ошибка, остатки будут действительно на дату 01.11.3999, а не на 01.12.3999 или на 31.12.3999, как могло бы ожидать. Именно остатки на дату 01.11.3999 называются актуальными остатками (также их называют текущими остатками).

Для регистра накопления *ОстаткиНоменклатуры* таблица итогов будет выглядеть следующим образом:

Период	Номенклатура	Количество	Сумма
01.10.2018 00:00:00	Виг (капилярная)	10	1000
01.10.2018 00:00:00	Паркер "Golg"	5	2500
01.11.2018 00:00:00	Виг (капилярная)	5	500
01.11.2018 00:00:00	Паркер "Golg"	5	2500
01.12.2018 00:00:00	Виг (капилярная)	5	500
01.12.2018 00:00:00	Паркер "Golg"	5	2500
01.11.3999 00:00:00	Виг (капилярная)	5	500
01.11.3999 00:00:00	Паркер "Golg"	5	2500

Управление итогами

В пользовательском режиме 1С:Предприятие можно управлять итогами. Для этого нужно открыть специальный инструмент *Все функции – Стандартные – Управление итогами*:

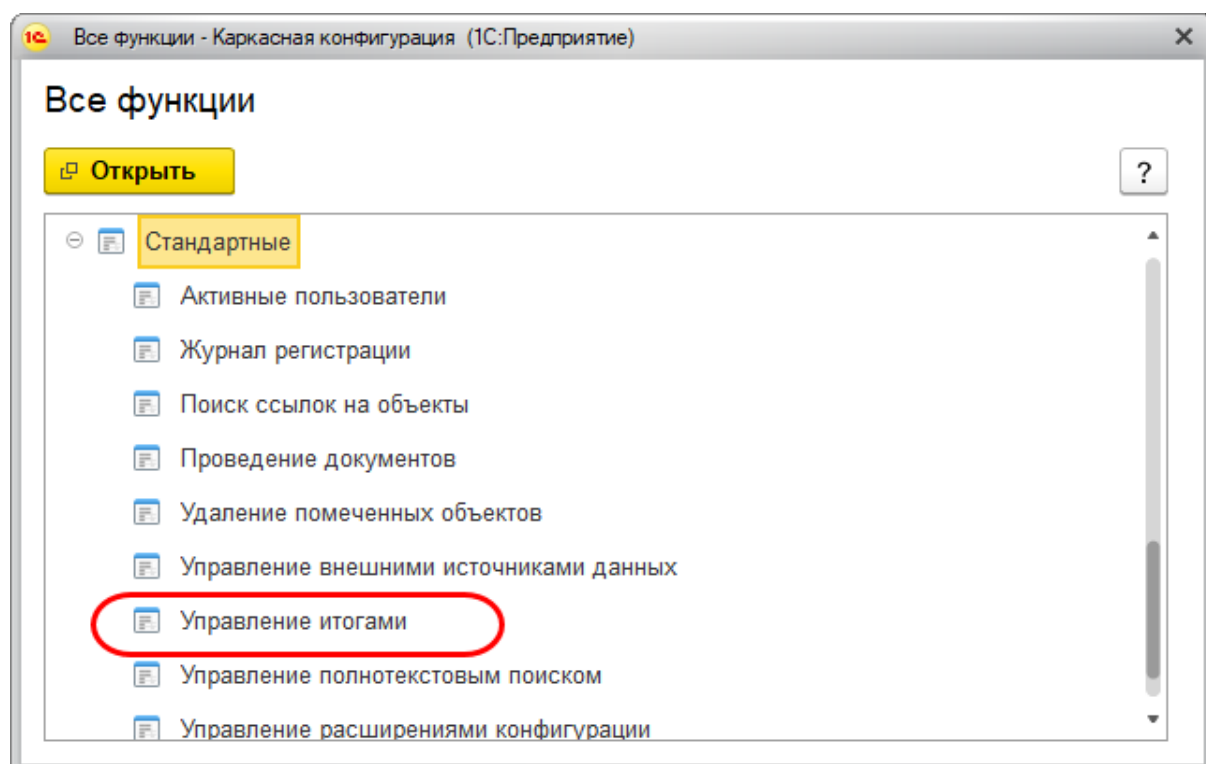


Рисунок 3 – Открытие окна *Управление итогами*

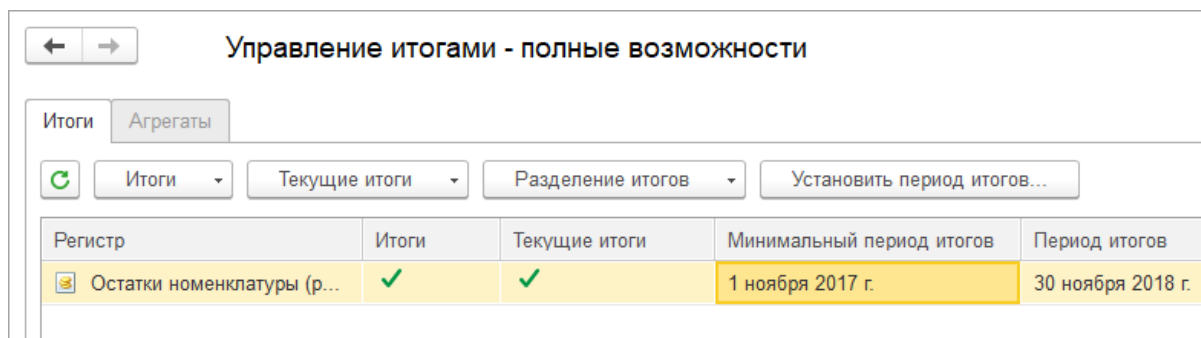


Рисунок 4 – Окно *Управление итогами: полные возможности*

В данном окне можно отключить:

- Использование итогов. При этом будут недоступны виртуальные таблицы регистра. При попытке к ним обратиться будет выдаваться ошибка.

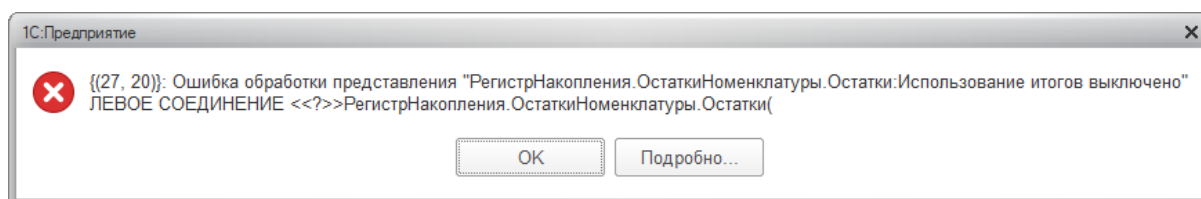


Рисунок 5 – Ошибка при обращении к виртуальным таблицам регистра

- Использование текущих итогов. При этом в таблице итогов удалятся записи на дату 01.11.3999.

Отключение итогов используется когда нужно провести много документов по регистру. Так как в этом случае при проведении документа не будут пересчитываться итоги, операция выполнится быстрее. После выполнения данной операции необходимо обратно включить итоги по регистру. При этом итоги автоматически пересчитаются..

Как получить актуальные остатки

Чтобы получить актуальные остатки достаточно в запросе к виртуальной таблице остатков регистра накопления передать в качестве параметра *Период* пустую дату или *Неопределенно*.

Пример:

Запрос = Новый Запрос;

Запрос.Текст = "ВЫБРАТЬ

| ОстаткиНоменклатурыОстатки.Номенклатура,

| ОстаткиНоменклатурыОстатки.КоличествоОстаток,

```
|      ОстаткиНоменклатурыОстатки.СуммаОстаток
|ИЗ
|      РегистрНакопления.ОстаткиНоменклатуры.Остатки(&Период , ) КАК
ОстаткиНоменклатурыОстатки";
Запрос.УстановитьПараметр("Период",Неопределено);
```

Преимущества использования актуальных остатков

Когда получаются остатки на произвольную дату в таблице итогов находится ближайшая дата, на которую рассчитаны остатки и добавляются движения из таблицы движений.

Пример:

При получении остатков на 03.10.2018 из таблицы итогов будут получены остатки на 01.10.2018 и вычтен расход от 02.10.2018 из таблицы движений.

При получение актуальных остатков идет обращение только к таблице итогов. Так как актуальные остатки уже рассчитаны, получение их происходит быстрее. Поэтому при решении практических задач часто используют следующую конструкцию:

```
МоментВремениОстатков = ?(РежимПроведения = РежимПроведенияДокумента.Оперативный, Неопределено,
МоментВремени());
Запрос.УстановитьПараметр("Период", МоментВремениОстатков)
```

Примечание. При оперативном проведении считается, что документ проводится в реальном времени поэтому можно использовать актуальные остатки. Оперативное проведение документов подробно рассмотрено в главе «1. Какие настройки проведения документов рекомендуется использовать на аттестации».

Недостатки использования актуальных остатков

При получении актуальных остатков мы можем получить недостоверные данные как минимум в трех случаях.

Ввод документов будущей датой

Платформа «1С:Предприятие» позволяет ввести документы будущей датой если у свойства документа *Оперативное проведение* установить значение *Запретить*.

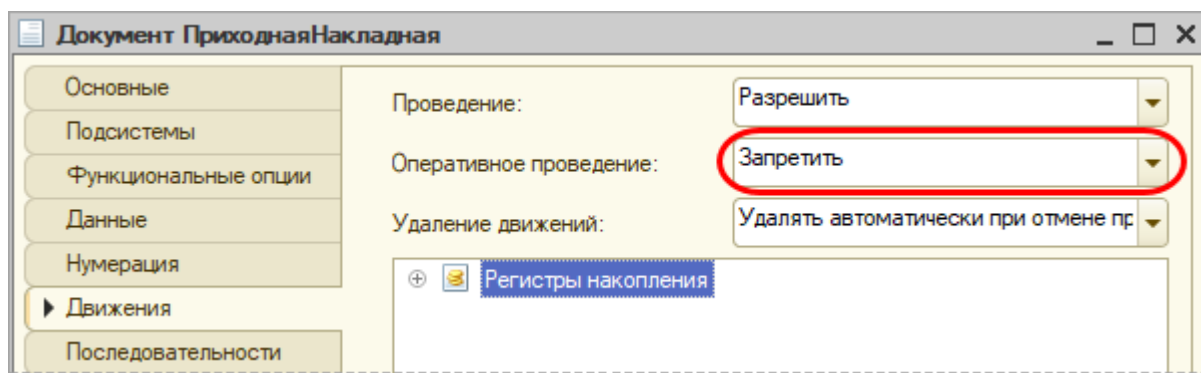


Рисунок 6 – Свойство *Оперативное проведение* документа *Приходная Накладная*

При такой настройке можно провести документ *Приходная накладная* будущей датой 31.12.2018.

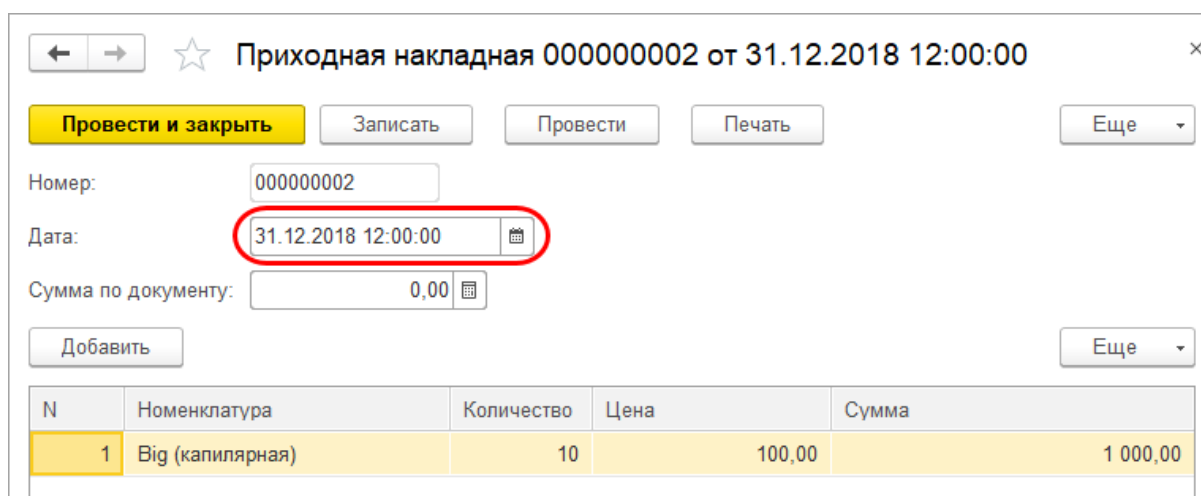


Рисунок 7 – Проведение документа *Приходная накладная* будущей датой

Добавление данных в регистр вручную

Во всех типовых конфигурациях есть служебный документ, позволяющий пользователю ввести данные в регистры вручную. Таким образом в данном документе можно указать движение будущим периодом.

Пример:

Документ «Корректировка регистров» в конфигурации 1С:ERP:

Период	N	Активность	Вид движения	Номенклатура	Характеристика
31.12.2018 0:00:00	1	Да	Приход	Big (капиллярная)	

Рисунок 8 – Корректировка регистров в конфигурации 1С:ERP

Проведение документа программно

Даже если у документа установлено свойство *Оперативное проведение* в значение *Разрешить*. Программно можно провести документ будущей датой.

Пример:

Достаточно написать простую обработку, в которой указывается документ и дата.

Рисунок 9 – Проведение документа *Приходная накладная* будущей датой с помощью обработки

По нажатию на кнопку *Провести будущей датой* в документе устанавливается дата, указанная в обработке и документ проводится.

Листинг обработки:

&НаСервере

Процедура ПровестиБудущейДатойНаСервере()

ДокументОбъект = Объект.ПриходнаяНакладная.ПолучитьОбъект();

ДокументОбъект.Дата = Объект.Дата;

ДокументОбъект.Записать(РежимЗаписиДокумента.Проведение);

КонецПроцедуры

&НаКлиенте

Процедура ПровестиБудущейДатой(Команда)

ПровестиБудущейДатойНаСервере();

КонецПроцедуры

В итоге выполнения обработки документ проводится будущей датой.

N	Номенклатура	Количество	Цена	Сумма
1	Big (капиллярная)	10	100,00	1 000,00

Рисунок 10 – Проведенный документ Приходная накладная будущей датой

Как результат описанных выше действий при оперативном проведении документа «Расходная накладная» от 03.12.2018 и получении актуальных остатков движение от 31.12.2018 учтется, что позволит отгрузить товар в минус.

Рисунок 11 – Документ *Расходная накладная*

Подведение итогов

Использование актуальных итогов позволяет ускорить проведение документов, но есть риск получить недостоверные данные при контроле остатков. Поэтому разработчик должен самостоятельно принимать решение, использовать их на реальных проектах или нет.

А на экзамене за использование актуальных остатков из оценки вычитается 1 балл. Так как это считается ошибкой.

В задачах получения данных из информационной базы установка отборов по неиндексированным полям	0,5
Если при проведении документа используются каким-то образом данные, считываемые из регистров, обязательно требуется предусмотреть получение таких данных на момент проведения документа	1,0
Конфигурация должна устойчиво работать не только при движении вперед, но и назад.	1,0

Рисунок 12 – Фрагмент из списка часто встречающихся ошибок