Министерство науки и высшего образования Российской Федерации

Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

С.Н. Широбокова

РАЗРАБОТКА КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ НА ПЛАТФОРМЕ «1С:ПРЕДПРИЯТИЕ»

Учебное пособие к практическим занятиям

Новочеркасск
ЮРГПУ(НПИ)
2021

УДК 004.42 (075.8)
ББК 32.973-018.1я73
Ш64

Широбокова С.Н.

Ш64 Разработка корпоративных информационных систем на платформе «1С:Предприятие»: учебное пособие к практическим занятиям / С.Н. Широбокова; Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова. – Новочеркасск: ЮРГПУ(НПИ), 2021.– 96с.

Учебное пособие содержит материалы к практическим занятиям по дисциплине "Разработка корпоративных информационных систем на платформе «1С:Предприятие»".

Предназначено для студентов вузов, обучающихся по направлению подготовки «Прикладная информатика» (магистратура). Пособие может быть полезно студентам других компьютерных направлений подготовки, изучающим основы разработки корпоративных информационных систем на платформе «1С:Предприятие».

УДК 004.42 (075.8)
ББК 32.973-018.1я73

Содержание

ВВЕДЕНИЕ

Учебное пособие предназначено для методического обеспечения практических занятий по дисциплине "Разработка корпоративных информационных систем на платформе «1С:Предприятие»".

Практические занятия ориентированы на получение практических навыков конфигурирования и программирования в среде «1С:Предприятие» при создании информационно-аналитических учетных систем для различных предметных областей. В ходе практических занятий обучающиеся получают углубленные знания и практические навыки разработки учетных систем на платформе «1С:Предприятие» для заданной предметной области, использования Конфигуратора «1С:Предприятие» для создания структуры прикладных объектов и разработка командного интерфейса приложения (подсистемы, роли), программирования различных интерфейсных задач в формах и работы с асинхронными вызовами в системе «1С:Предприятие», конфигурирования и программирования прикладных объектов для организация решения задач оперативного учета, синтетического и аналитического бухгалтерского учета, решения расчетных задач в приложениях на платформе«1С:Предприятие», разработки сложных отчетов с помощью системы компоновки данных.

Для проведения практических занятий обязательно требуется использование лабораторий с компьютерной техникой (рабочими местами обучающихся, оснащенными компьютерами с установленной учебной версией платформы «1С:Предприятие»).

Тема 1. Системы на основе аналитики больших данных. Проектирование информационной системы на платформе «1С:Предприятие» для заданной предметной области. Использование Конфигуратора «1С:Предприятие» для создания структуры прикладных объектов. Разработка командного интерфейса приложения (подсистемы, роли)

В рамках практических занятий будет проектироваться информационно-аналитическая система для торговой организации на платформе «1С:Предприятие».

Все объекты конфигурации, которые существуют в системе «1С:Предприятие», образуют несколько основных видов (справочник, документы, константы и т.п.). Каждый вид объектов конфигурации представляет собой те «строительные элементы», из которых будет создаваться конфигурация. Разбивку объектов по видам можно увидеть в дереве конфигурации (они находятся на первом уровне) [1].

Структура конфигурации является моделью предметной области. Создание конфигурации выполняется при помощи конфигуратора. Созданная конфигурация используется системой «1С:Предприятие» для реализации программного окружения, пригодного для выполнения необходимых учетных задач. Для целей создания нашей конфигурации нам потребуются следующие основные типы объектов [2]:

Перечисления. Списки значений, задаваемых на этапе конфигурирования. Используются в системе для описания постоянных наборов значений, не изменяемых в процессе работы конфигурации. В отличие от справочников, значения перечислений задаются на этапе конфигурирования и не могут быть изменены на этапе исполнения.

Документы. Служат для ввода информации о совершаемых операциях в системе. Система поддерживает режим автоматической нумерации документов, при котором она самостоятельно может генерировать номер для нового документа. Кроме этого система позволяет осуществлять контроль уникальности номеров документов, не разрешая создавать документы с одинаковыми номерами.

Справочники. Это списки однородных элементов данных. Используются для хранения нормативно-справочной информации. Они позволяют хранить в информационной базе данные, имеющие одинаковую структуру и списочный характер. Обычно справочниками являются списки материалов, товаров, организаций, сотрудников, валют и т.п.

Журналы документов. Служат для отображения списков документов различного вида. Журналы документов предназначены для просмотра документов различных видов. Каждый вид документа может быть показан в нескольких журналах. Журнал не добавляет новые данные в системы, а является средством для отображения в едином списке документов нескольких видов.

Регистры сведений. Служат для хранения информации, состав которой развернут по определенной комбинации значений и, при необходимости,

развернут по времени. Например, с помощью регистра сведений можно хранить курсы валют.

Регистры накопления. Служат для накопления информации в разрезе измерений с возможностью получения остатков или оборотов числовых величин (или остатков и оборотов).

Регистры расчетов. Служат для накопления информации о периодических расчетах (например, заработной планы или арендных платежах).

Регистры бухгалтерии. Используются для отражения в бухгалтерском учете информации о хозяйственных операциях.

Отчеты. Средство получения выходной информации.

Планы счетов. Совокупность синтетических счетов. Один из основных понятий бухгалтерского учета. Предназначен для группировки информации о хозяйственной деятельности предприятия.

Планы видов характеристик. Предназначены для описания множеств однотипных объектов аналитического учета.

Планы видов расчета. Предназначены для описания множеств однотипных объектов и механизмов расчета. Используются совместно с регистрами расчета для решения расчетных задач (например, заработной платы).

Создадим минимально необходимые вспомогательные объекты конфигурации для реализации на последующих занятиях оперативных, бухгалтерских и расчетных задач.

1) Создание необходимых перечислений.

«ВидыНоменклатуры». Создадим перечисление «ВидыНоменклатуры» со следующими значениями: «Товар», «Услуга», «Материал».

«Пол». Создадим перечисление «Пол» со следующими значениями: «Мужской» и «Женский».

2) Создание справочников.

«Контрагенты». Создадим справочник «Контрагенты» со следующим реквизитом: ПолноеНаименование, тип Строка, длина 200.

«Договоры». Создадим справочник «Договоры» без дополнительных реквизитов и табличных частей, подчиненный справочнику «Контрагенты».

«Склады». Создадим справочник «Склады» без дополнительных реквизитов и табличных частей.

«Организации». Справочник «Организации» будет содержать один дополнительный реквизит «УчетПоПодразделениям», тип Булево.

«Подразделения». Справочник «Подразделения» создадим без дополнительных реквизитов и табличных частей, подчинен справочнику «Организации».

«ЕдиницыИзмерения». Справочник «ЕдиницыИзмерения» создадим также без дополнительных реквизитов и табличных частей.

«Номенклатура». Справочник «Номенклатура» в простейшем варианте будет содержать реквизиты:

«ЦенаПокупки» и «ЦенаПродажи», тип Число, длина 15, точность 2;

«ВидНоменклатуры», тип ПеречислениеСсылка.ВидыНоменклатуры;

«ОснЕдиницаИзмерения», тип СправочникСсылка.ЕдиницыИзмерения.

«ФизическиеЛица». Справочник «ФизическиеЛица» имеет следующие реквизиты:

«Фамилия», тип Строка, длина 25;

«Имя», тип Строка, длина 25;

«Отчество», тип Строка, длина 25;

«ДатаРождения»: тип Дата, состав «Дата»;

«Пол»: тип ПеречислениеСсылка «Пол»;

«ИмяФайла», тип строка, длина 150.

Справочник имеет табличную часть «ТрудоваяДеятельность». В нее входят следующие реквизиты:

«Организация», тип Строка, длина 40;

«Должность», тип Строка, длина 40;

«НачалоРаботы», тип Дата, состав Дата;

«ОкончаниеРаботы», тип Дата, состав Дата.

«Должности». Справочник «Должности» не имеет дополнительных реквизитов.

2) Создание документов.

Создадим документ «**ПоступлениеТоваров**» со следующей структурой. Реквизиты шапки:

«Контрагент», тип СправочникСсылка.Контрагенты;

«Договор»: тип СправочникСсылка.Договоры;

«Склад», тип СправочникСсылка.Склады;

«СуммаДокумента», тип Число, длина 15, точность 2.

Документ будет иметь табличную часть «Товары». В нее добавим следующие реквизиты табличной части:

«Номенклатура», тип СправочникСсылка.Номенклатура;

«Количество», тип Число, длина 15, точность 3;

«Цена», тип Число, длина 15, точность 2;

«Сумма», тип Число, длина 15, точность 2.

Создадим документ «**ПродажаТоваров**» со следующей структурой. Реквизиты шапки:

««Контрагент», тип СправочникСсылка.Контрагенты;

«Договор»: тип СправочникСсылка.Договоры;

«Склад», тип СправочникСсылка.Склады;

«СуммаДокумента», тип Число, длина 15, точность 2.

Документ будет иметь табличную часть «Товары». В нее добавим следующие реквизиты табличной части:

«Номенклатура», тип СправочникСсылка.Номенклатура;

«Количество», тип Число, длина 15, точность 3;

«Цена», тип Число, длина 15, точность 2;

«Сумма», тип Число, длина 15, точность 2.

3) **Создание журнала документов** «СкладскиеДокументы». Для объединения в одном хронологическом порядке документов поступления и

продажи товаров создадим журнал документов «СкладскиеДокументы». В нем создадим три графы:

«Контрагент»: в графе в свойстве ссылки выберем из обоих документов реквизит «Контрагент»;

«Склад»: в графе в свойстве ссылки выберем из обоих документов реквизит «Склад»;

«СуммаДокумента: в графе в свойстве ссылки выберем из обоих документов реквизит «СуммаДокумента».

4) **Создание подсистем**.

При запуске системы открывается основное окно программы [1]. В нем пользователю видна вся структура прикладного решения. Основное разделение функциональности представляется в виде панели разделов (пример представлен на рис. 1.1).

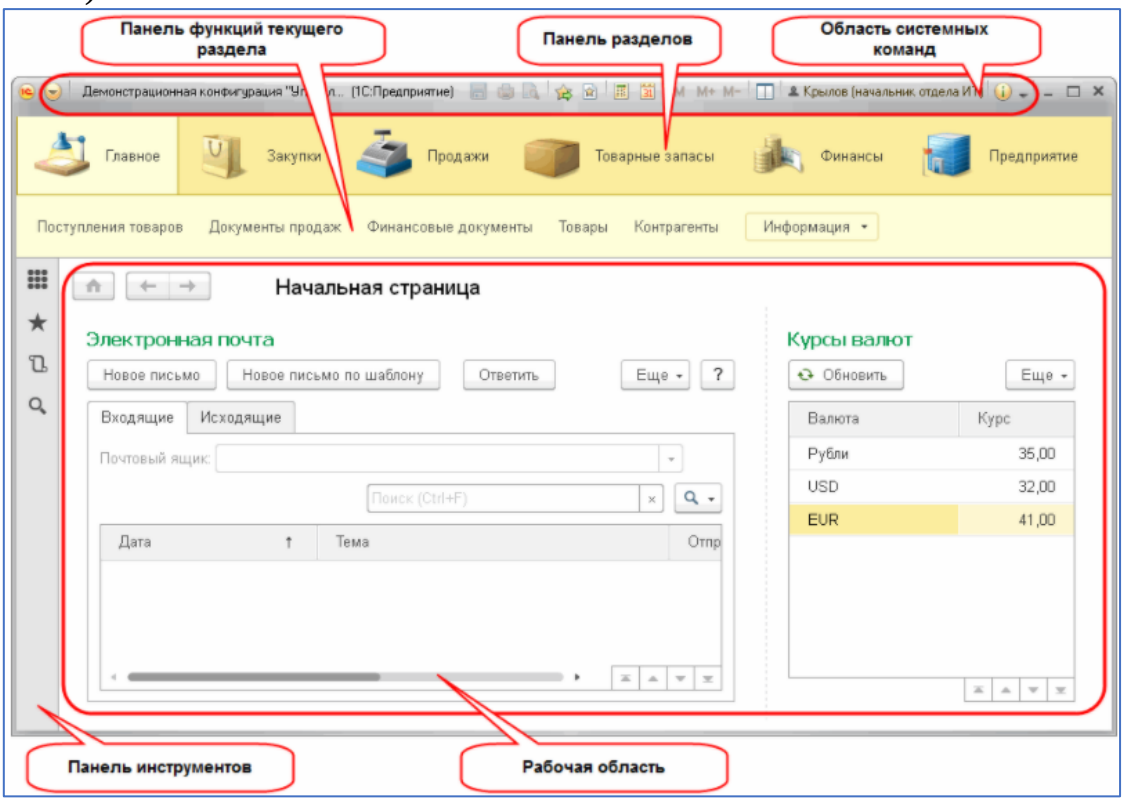


Рис. 1.1. Основное окно программы

В нашей конфигурации создадим для начала подсистему «НСИ» куда отнесем общие справочники; «Закупки», «Продажи», куда отнесем объекты, используемые для выполнения соответствующих бизнес-процессов. В дальнейшем интерфейс будет пополняться новыми подсистемами.

5) **Система прав доступа.**

Система прав доступа позволяет описывать наборы прав, соответствующие должностям пользователей или виду деятельности. Структура прав определяется конкретным прикладным решением. В системе «1С:Предприятие» для реализации ограничения прав доступа предназначены специальные объекты конфигурации — Роли.

Все права, поддерживаемые системой «1С:Предприятие 8», можно разделить на две большие группы: *основные* и *интерактивные* [1].

Основные права описывают действия, выполняемые над элементами данных системы или над всей системой в целом, и проверяются всегда, независимо от способа обращения к данным.

Интерактивные права описывают действия, которые могут быть выполнены пользователем интерактивно. Соответственно проверяются они только при выполнении интерактивных операций стандартными способами, причем в клиент-серверном варианте все проверки прав (кроме интерактивных) выполняются на сервере.

Основные и интерактивные права взаимосвязаны. Например, существует основное право Удаление, которому соответствуют два интерактивных права: Интерактивное удаление и Интерактивное удаление помеченных. Если пользователю запрещено Удаление, то и все интерактивные «удаления» также будут запрещены для него. В то же время, если пользователю разрешено Интерактивное удаление помеченных, это значит, что Удаление ему также разрешается.

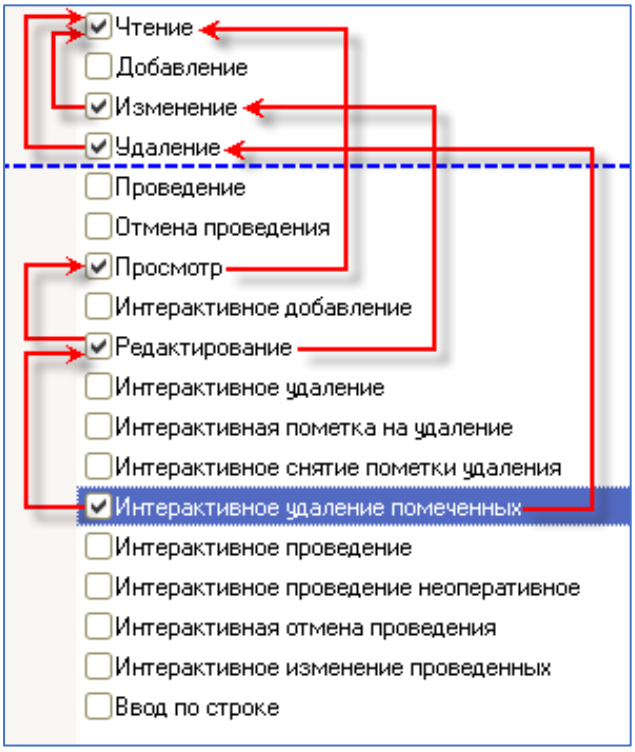


Рис. 1.2. Взаимосвязь основных и интерактивных прав на примере права «интерактивное удаление помеченных» [1]

Кроме того, основные права могут зависеть друг от друга. В результате образуются довольно сложные цепочки взаимосвязей, которые отслеживаются системой автоматически: как только разработчик снимает разрешение на какое-либо право, система сама снимает разрешения на все права, которые зависят от этого права. И наоборот, при установке какого-либо права разработчиком, система сама устанавливает все права, от которых это право зависит.

Например, для того, чтобы пользователь имел право Интерактивное удаление помеченных, ему необходимо обладать интерактивными правом Редактирование. Это право, в свою очередь, требует наличия интерактивного права Просмотр (рис. 1.2). Право Интерактивное удаление помеченных требует наличия основного права Удаление. Интерактивное право Редактирование требует наличия основного права Изменение. Интерактивное право Просмотр требует наличия основного права Чтение. Кроме этого основные права Изменение и Удаление требуют наличия основного права Чтение.

В общем случае права можно задавать:

- на всю конфигурацию в целом,
- объекты,
- реквизиты объектов,
- табличные части,
- реквизиты табличных частей,
- стандартные реквизиты.

Итак, роль определяет набор прав пользователя, которые он имеет. Для каждого из объектов (справочники, документы) разработчик устанавливает свой набор прав — чтение/запись/добавление/изменение и т.д.

Откроем в конфигурации ветку Общие. В ней есть вид объектов метаданных Роли (рис. 1.3), в которой следует добавлять наборы прав для разных категорий пользователей.

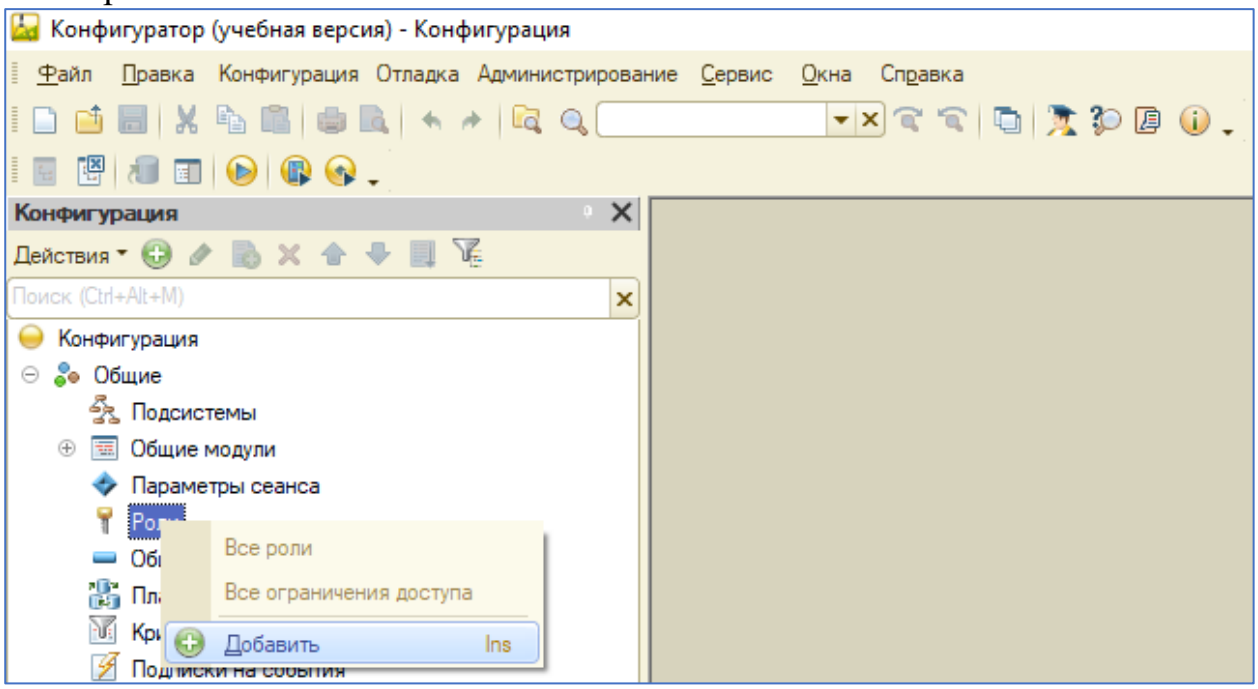


Рис. 1.3. Добавление ролей

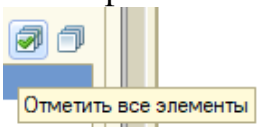
Создадим два набора прав: Администратор и Менеджер. Администратор будет обладать полными правами работы с информационной базой. У сотрудника будут отключены некоторые административные права по работе с информационной базой.

Добавим в ветке Роли новый объект и дадим имя новой роли «Администратор». Откроется окно редактора прав доступа следующего вида –

рис. 1.3. У объекта Роль есть две закладки — *Права* и *Шаблоны ограничений*. Права — основная закладка, Шаблоны — вкладка для настройки прав на уровне записи в 1С. В лабораторной работе мы рассмотрим только первую основную закладку Права. На закладке «Права» в левой части окна редактирования прав выводится дерево объектов конфигурации по всем подсистемам. В правой - список прав по выбранному объекту конфигурации в дереве конфигурации. Если для действия установлен флажок, то оно разрешено.

В нижней части окна есть три флага, Их назначение в следующем [1]:

- **Устанавливать права для новых объектов** — если флаг установлен у роли, на новые объекты метаданных будут автоматически установлены разрешающие права. В учебных целях, чтобы не забывать установить права на новые объекты, установим этот флаг. В реальных конфигурациях к установке этого флага надо подходить с осторожностью, поскольку для разных объектов, входящих в определенную ветку могут требоваться разные права доступа.
- **Устанавливать права для реквизитов и табличных частей по умолчанию** — флаг, при установке которого реквизиты и табличные части будут наследовать права владельца(справочника, документа и т.д.).
- **Независимые права подчиненных объектов** — если флаг установлен, то система при определении права на объект конфигурации учтёт права на родительский объект.

Установим курсор на корень конфигурации и увидим настройки как на рис. 1.4. Это настройки прав в целом на всю конфигурацию. Для роли Администратор отметим все флаги. Это можно сделать не только вручную, то и с помощью кнопки 

.

Краткая характеристика прав в целом на всю конфигурацию [1]:

- **Администрирование** – администрирование информационной базы (требуется наличие права «Администрирование данных»).
- **Администрирование данных** – право на административные действия над данными.
- **Обновление конфигурации базы данных** – право на обновление конфигурации базы данных.
- **Монопольный режим** – использование монопольного режима.
- **Активные пользователи** – просмотр списка активных пользователей.
- **Журнал регистрации** – журнал регистрации.
- **Тонкий клиент** – право запуска тонкого клиента.
- **Толстый клиент** – право роли запуска толстого клиента.
- **Веб клиент** – право запуска веб-клиента.
- **Внешнее соединение** – право запуска внешнего соединения.
- **Automation** – право на использование automation.

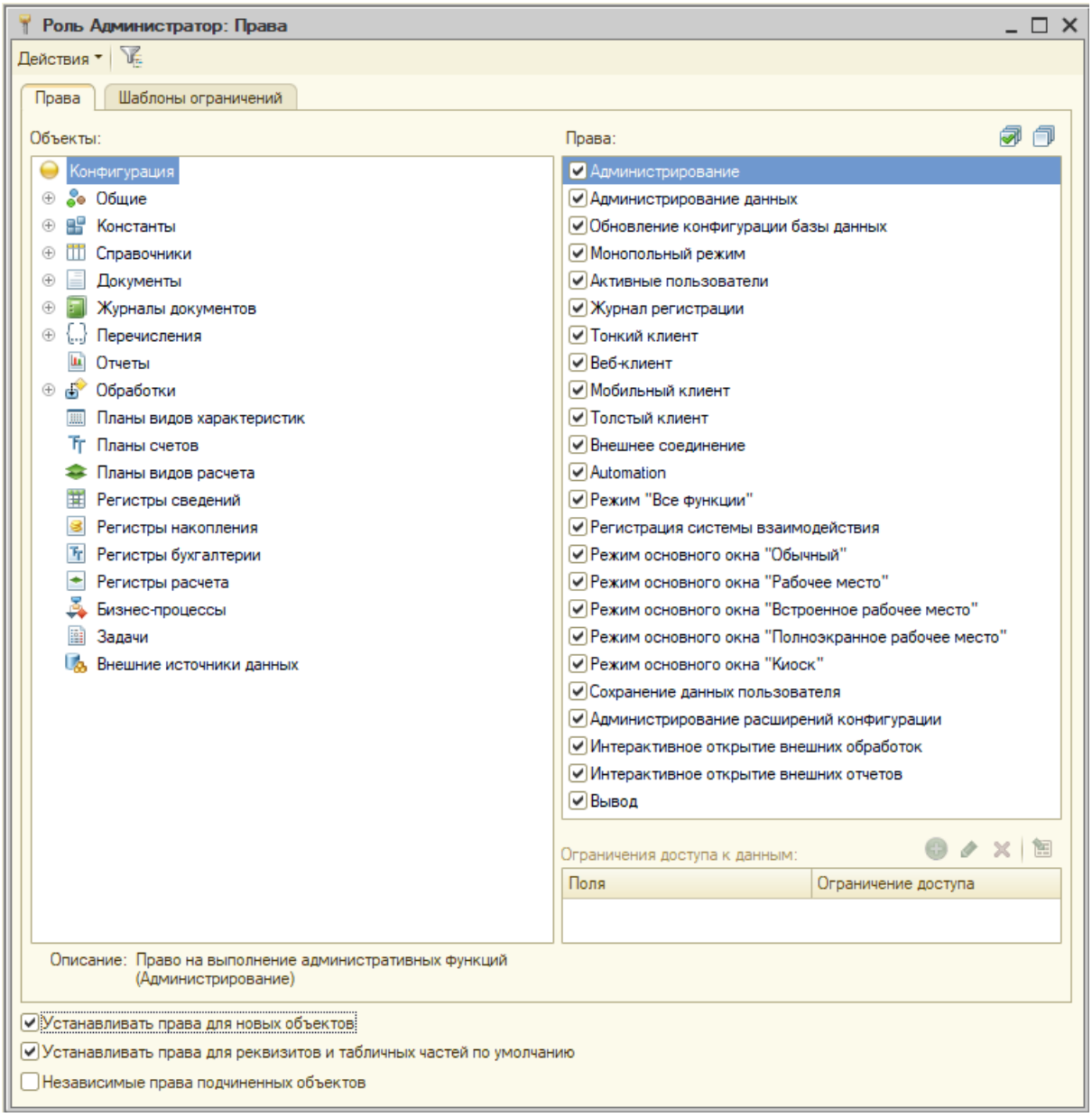


Рис. 1.4. Окно настройки прав для роли

- **Режим «Все функции»** – доступ к пункту меню «Все функции» в режиме управляемого приложения.
- **Сохранение данных пользователя** – разрешение или запрет на сохранение данных пользователя (настроек, избранного, истории). Особенно актуально для 1С управляемых форм.
- **Интерактивное открытие внешних обработок** – открытие внешних обработок.
- **Интерактивное открытие внешних отчетов** – открытие внешних отчетов.
- **Вывод** – вывод на печать, запись и копирование в буфер обмена.

Для остальных основных объектов (справочники, константы, документы, регистры…), набор прав у роли достаточно стандартен:

- **Чтение** — чтение (программное)
- **Добавление** — добавление (программное)

- **Изменение** — изменение (программное)
- **Удаление** — удаление (программное)
- **Просмотр** — просмотр
- **Интерактивное добавление** — интерактивное добавление
- **Редактирование** — редактирование
- **Интерактивная пометка удаления** — интерактивная пометка на удаление
- **Интерактивное снятие пометки удаления** — снятие пометки на удаление
- **Интерактивное удаление помеченных**— удаление помеченных объектов
- **Ввод по строке** — использование режима ввода по строке
- **Интерактивное удаление** — непосредственное удаление (shift +del)

Права только для **документов**:

- **Интерактивное проведение** — проведение
- **Отмена проведения** — отмена проведения документов
- **Интерактивное проведение неоперативное** — проведение (стандартными командами форм) документа в неоперативном режиме
- **Интерактивная отмена проведения** — интерактивная отмена проведения
- **Интерактивное изменение проведенных** — редактирование проведенного документа. Если право у роли не установлено, то пользователь не может удалить проведенный документ, установить пометку удаления, перепровести или сделать непроведенным. Форма такого документа открывается в режиме просмотра.

Только для регистров накопления и бухгалтерии:

- **УправлениеИтогами** — управление итогами регистра бухгалтерии и регистра накопления (установка периода, по который рассчитаны итоги, и пересчет итогов)

Только для обработок и отчетов:

- **Использование** — использование.

При создании новой роли все права устанавливаются конфигуратором в следующее состояние:

- для объектов права не установлены;
- для реквизитов (включая стандартные) и табличных частей (включая стандартные) права установлены.

В окне редактора прав роли расположены три табличных поля. В первом (слева) производится выбор нужного объекта конфигурации. В первой колонке второго табличного поля выводится список прав по выбранному объекту. Соседняя колонка предназначена для указания использования каждого права для роли.

В третьем табличном поле редактируются условия доступа к данным на уровне отдельных полей и записей.

Для роли Администратор для каждой ветви объектов установите все права.

Создадим вторую роль Менеджер. Общие права для этой роли оставим по умолчанию следующими (рис. 1.5):

Разрешим все права для роли Менеджер для всех видов объектов метаданных

После окончания редактирования обеих ролей сохранение конфигурацию перед вводом пользователей.

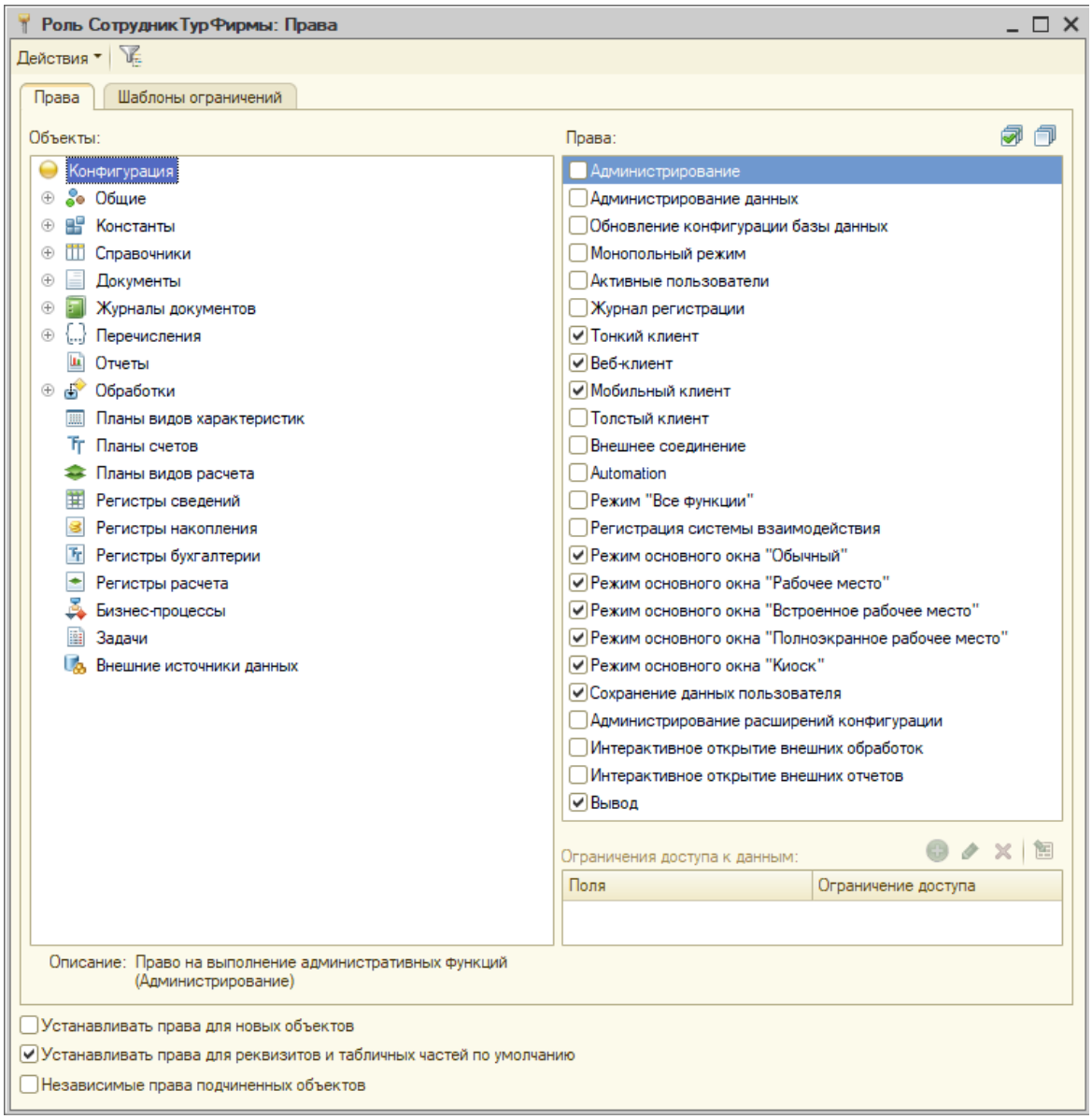


Рис. 1.5. Настройка общих прав для роли Менеджер

6) Список пользователей

Список пользователей — это один из инструментов администрирования. Система «1С:Предприятие» позволяет вести список пользователей, которым разрешена работа с системой. Этот список не является частью прикладного решения, а создается отдельно в конкретной организации, в которой используется система. Для ввода списка пользователей используется пункт меню «Администрирование» -> «Пользователи».

Тема 2. Программирование различных интерфейсных задач в формах

В рамках данной темы будут рассмотрены отдельные элементы конфигурирования и программирования интерфейсных задач в управляемых формах системы «1С:Предприятие», которые могут быть полезны при разработке пользовательского интерфейса.

1) Создание многостраничной формы. Редактор форм позволяет добавлять в форму специальные элементы, которые помогают придать форме собственный узнаваемый стиль, сделать доступ к данным простым и понятным, а также разместить большой объем информации на ограниченной площади.

Для этого редактор позволяет добавить в форму несколько элементов Группа – Страницы, каждая из которых может содержать несколько элементов Группа – Страница.

Создадим форму элемента для справочника «ФизическиеЛица». В ней создадим элемент типа Страницы, содержащий две страницы с заголовками «Основные» и «ТрудоваяДеятельность». На первой странице разместим все единичные реквизиты, а на вторую страницу вынесем табличную часть «ТрудоваяДеятельность».

2) Работа с картинками.

Платформа системы «1С:Предприятие» предоставляет возможность хранения некоторой вспомогательной информации в базе данных посредством полей типа *ХранилищеЗначения* [1]. С прикладной точки зрения в полях типа *ХранилищеЗначения* можно хранить, например, фотографии сотрудников и т.п.

Реализуем в нашей конфигурации возможность хранения фотографий сотрудников. Следует отметить, что возможность использования полей типа *ХранилищеЗначения* в составе объектов (например, справочников, документов), активно использующихся при реализации основной бизнес-логики, необходимо относиться аккуратно. Ведь данные объектов считываются целиком при обращении к ним. Поэтому, например, вопрос хранения тех же фотографий лучше решать не в составе справочника *Сотрудники*, а в отдельном подчиненном справочнике или регистре сведений. Тогда будет сохранена возможность идентификации соответствия изображений сотрудникам, к которым они относятся. И в то же время при использовании объектов справочника *Сотрудники* для решения других задач выполняемые операции не будут замедляться из-за необходимости считывания из базы данных больших объемов информации.

Ранее мы создали справочник «ФизическиеЛица». Самым простым вариантом реализации было бы добавить в него реквизит типа «ХранилищеЗнаения». Однако, прислушаемся к вышеуказанным рекомендациям и вынесем хранение фотографий в регистр сведений.

Создадим новый регистр сведений «ФотоФизическихЛиц». Измерение – «ФизическоеЛицо», тип СправочникСсылка.ФизическиеЛица.Ресурс – «Фото», тип ХранилищеЗначения.

В форме элемента справочника «ФизическиеЛица» создадим дополнительную группу типа «Страница» с именем «Фото» (рис. 2.1). Добавим для отображения фото реквизит формы «ОтображениеФото» и вынесем его на созданную страницу. В свойствах поля укажем тип «Поле картинки».

Создадим две команды формы «ЗагрузитьФото» и «УдалитьФото», для который далее реализуем функциональность загрузки фотографии физического лица из внешнего файла и возможность сохранения в информационной базе, а также возможность удаления загруженной ранее фотографии из базы.

Команды вытащим их на форму, расположив кнопки под созданным ранее полем отображения картинки. Чтобы кнопки команд можно было разместить горизонтально, объединим их в группу без отображения, у которой в свойствах укажем группировку «Горизонтальная всегда».

Для реализации функциональности команд, а также отображения фотографии в поле картинки, сохранения загруженной фотографии в базе в регистре сведений и извлечения данных из регистра сведений, получения навигационной ссылки загруженной ранее фотографии в модуле формы необходимо реализовать следующий код.

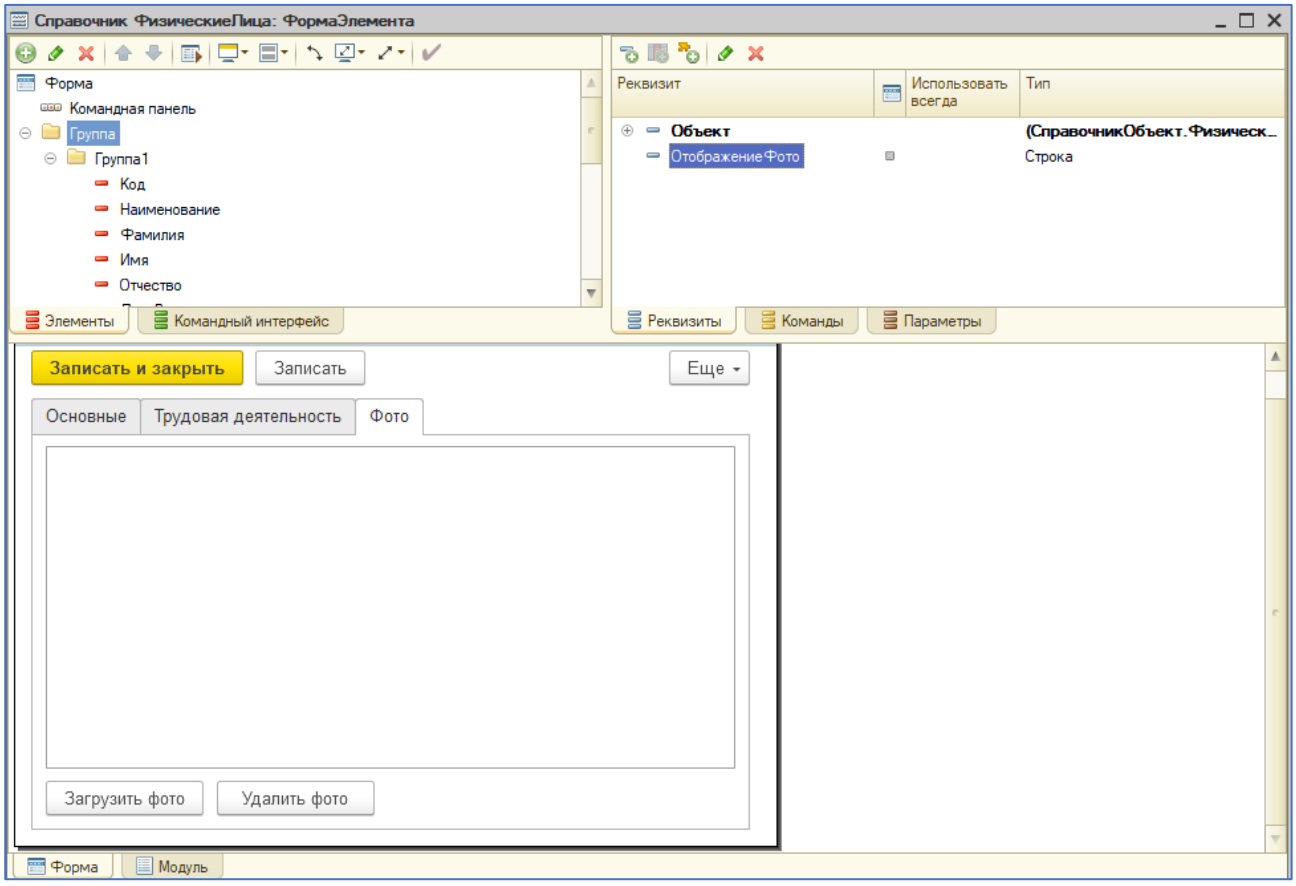


Рис. 2.1. Создание на форме элемента реквизита формы для отображения фотографий

```
&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
    ОтображениеФото="";
    ЗнКлюча=Новый Структура("ФизическоеЛицо",Объект.Ссылка);
```

```
    КлючНаЗапись =
РегистрыСведений.ФотоФизическихЛиц.СоздатьКлючЗаписи(ЗнКлюча);
      ОтображениеФото = ПолучитьНавигационнуюССылку(КлючНаЗапись,
"Фото", 0);
КонецПроцедуры

&НаСервере
Процедура ПередЗаписьюНаСервере(Отказ, ТекущийОбъект,
ПараметрыЗаписи)
      МенеджерЗаписи =
РегистрыСведений.ФотоФизическихЛиц.СоздатьМенеджерЗаписи();
    МенеджерЗаписи.ФизическоеЛицо = Объект.Ссылка;
      Если ЭтоАдресВременногоХранилища(ОтображениеФото) Тогда

      ДвоичныеДанные=ПолучитьИзВременногоХранилища(ОтображениеФото);
         МенеджерЗаписи.Фото= Новый ХранилищеЗначения(ДвоичныеДанные);

      ИначеЕсли ОтображениеФото = "" Тогда
         МенеджерЗаписи.Фото = Новый ХранилищеЗначения(Неопределено);
      КонецЕсли;
       МенеджерЗаписи.Записать();
 КонецПроцедуры

 &НаКлиенте
Процедура УдалитьФото(Команда)
      ОтображениеФото="";
КонецПроцедуры

&НаКлиенте
Процедура ЗагрузитьФото(Команда)
      Оповещение = Новый
ОписаниеОповещения("ОбработатьВыборФайлаФото", ЭтотОбъект);
      НачатьПомещениеФайлаНаСервер(Оповещение, , , , ,
УникальныйИдентификатор)
КонецПроцедуры

&НаКлиенте
Процедура ОбработатьВыборФайлаФото(Результат,
ДополнительныеПараметры) Экспорт
      Если Результат=Неопределено Тогда
         Возврат;
      КонецЕсли;
      ОтображениеФото = Результат.Адрес;
КонецПроцедуры
```

3) Связи параметров выбора. Связи параметров выбора – данное свойство позволяет указать список реквизитов, которые будут поставлять значения, используемые при выборе значения реквизита, при открытии формы выбора, при отображении списка быстрого выбора и при выполнении ввода по строке.

В качестве примера можно привести выбор договора с контрагентом. Вначале осуществляется выбор контрагента, а затем выполняется выбор договора только из списка договоров выбранного контрагента. Причем отбор автоматически изменяется при смене контрагента.

Для того чтобы ограничить выбор, реквизиту в свойстве Связи параметров выбора устанавливается соответствие имени реквизита, по которому будет выполняться фильтрация выбираемых значений, и реквизита, из которого будет браться значение фильтрации.

Выполним эти настройки для взаимосвязанных подчинением реквизитов «Контрагент» и «Договор» в документе «ПоступлениеТоваров» (рис. 2.2).

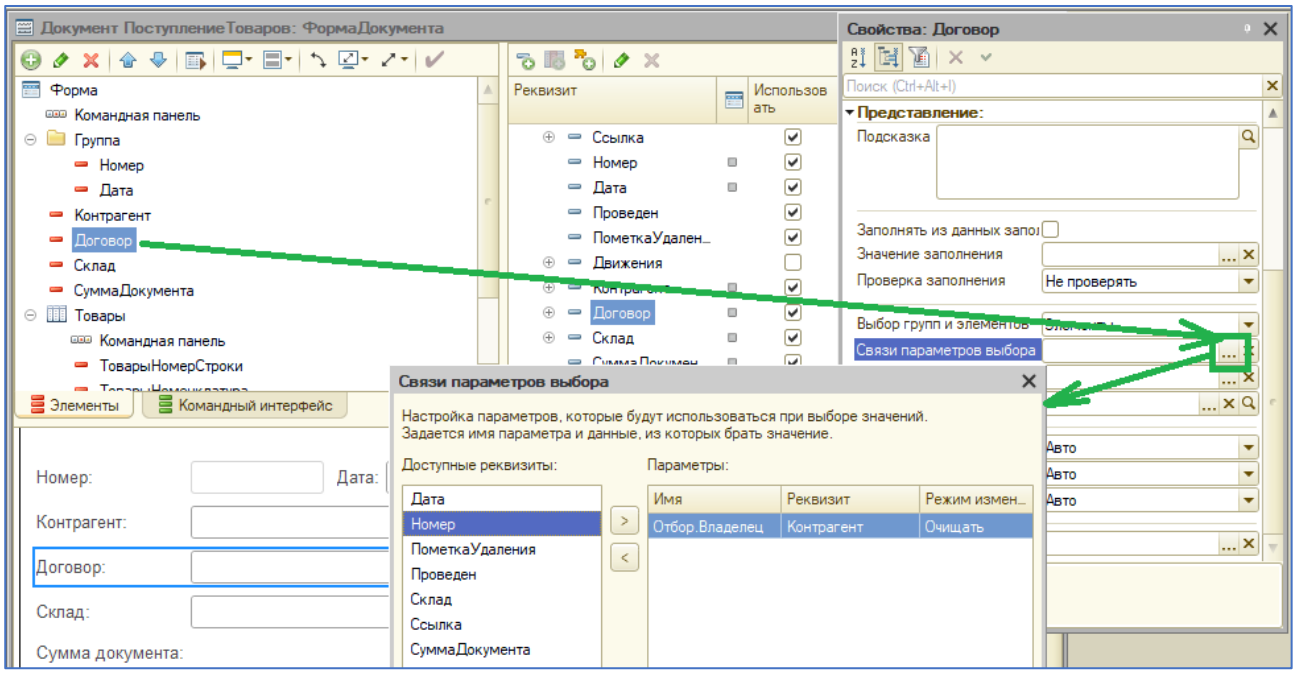


Рис. 2.2. Настройка связей параметров выбора

Значения, указанные в свойстве «Связи параметров выбора», будут переданы в открываемую форму через структуру Параметры. При этом значение колонки Имя будет соответствовать ключу элемента структуры, а значение реквизита, указанное в колонке Реквизит, – значению элемента структуры. Если в колонке Имя указано значение вида Отбор.Владелец, то будет создан параметр формы Отбор (типа Структура). В этой структуре будет создан элемент с ключом Код и значением, полученным из реквизита, указанного в колонке Реквизит (в нашем примере - Поставщик).

Также в окне редактирования связей параметров выбора можно задать режим очистки поля при изменении полей связи. Если значение свойства Режим изменения связанного значения равно Очищать, то поле будет очищено при интерактивном изменении значения связи (изменением считается также

повторный выбор значения, ранее находившегося в поле) до наступления события ПриИзменении. В противном случае (значение свойства равно Не изменять) поле не будет очищено. Очистка происходит вне зависимости от реального изменения значения в элементе связи и выполняется до вызова обработчика события ПриИзменении.

4) Настройка группировки в динамическом списке.

Динамический список – это объект построения интерфейса, который предназначен для создания списка объектов или записей [1]. В формах списка динамический список является основным реквизитом.

Настройку будем производить для формы списка журнала «СкладскиеДокументы» (рис. 2.3).

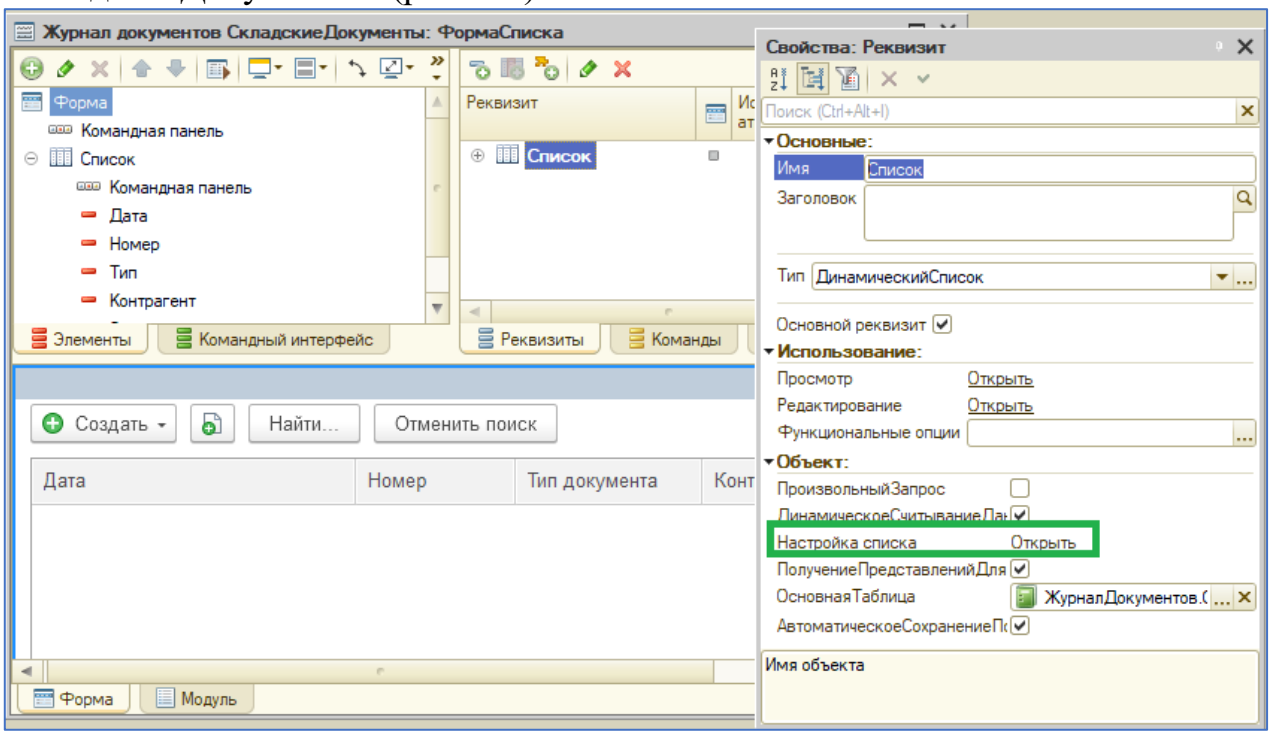


Рис. 2.3. Настройка динамического списка

В открывшемся окне на закладке «Настройки» есть несколько вложенных закладок, в том числе закладка «Группировка» (рис. 2.4). В качестве группировочного поля укажем, например, поле Дата.

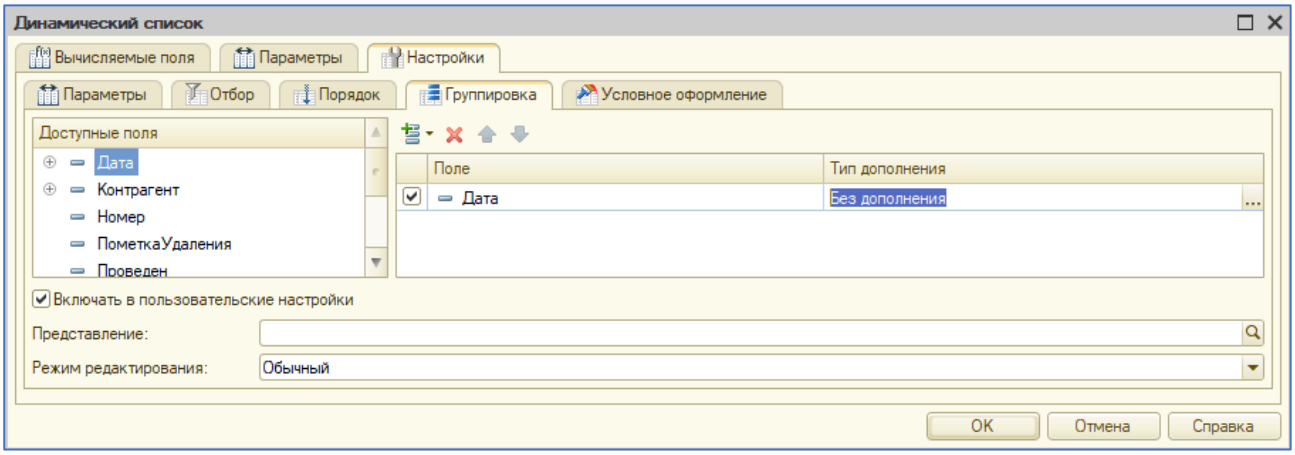
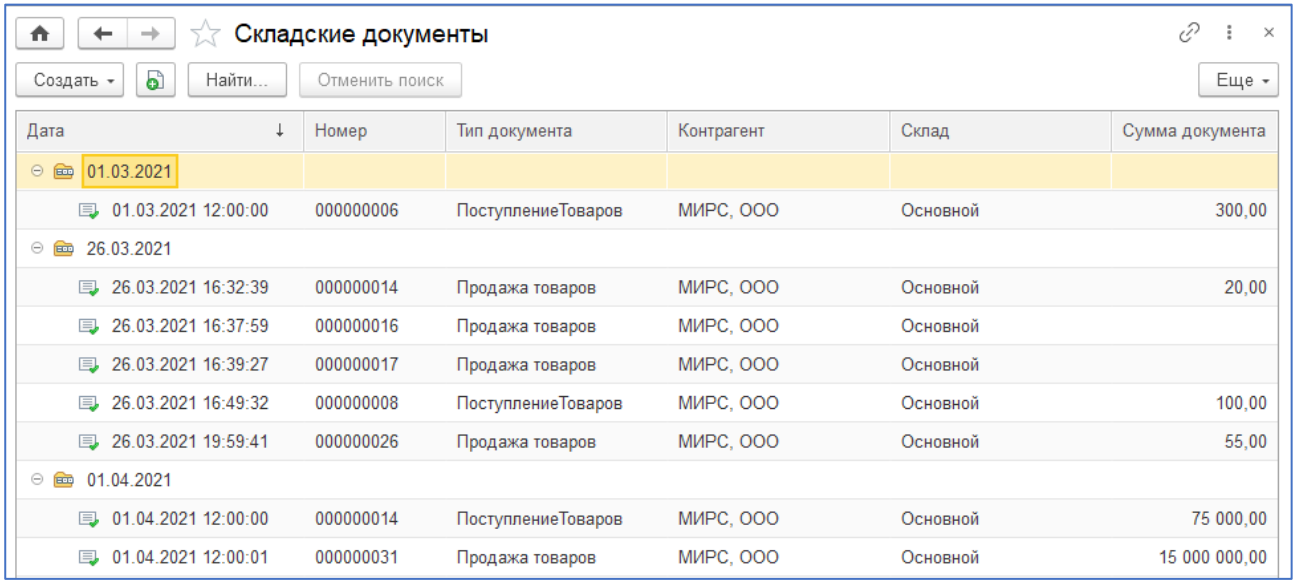


Рис. 2.4. Настройка группировки в динамическом списке

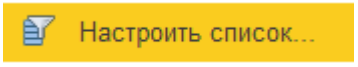
В результате в режиме исполнения динамический список будет содержать строки с объектами, сгруппированные по дате. Таким структурированным списком пользователю будет удобнее пользоваться.



Дата	Номер	Тип документа	Контрагент	Склад	Сумма документа
01.03.2021					
01.03.2021 12:00:00	000000006	ПоступлениеТоваров	МИРС, ООО	Основной	300,00
26.03.2021					
26.03.2021 16:32:39	000000014	Продажа товаров	МИРС, ООО	Основной	20,00
26.03.2021 16:37:59	000000016	Продажа товаров	МИРС, ООО	Основной	
26.03.2021 16:39:27	000000017	Продажа товаров	МИРС, ООО	Основной	
26.03.2021 16:49:32	000000008	ПоступлениеТоваров	МИРС, ООО	Основной	100,00
26.03.2021 19:59:41	000000026	Продажа товаров	МИРС, ООО	Основной	55,00
01.04.2021					
01.04.2021 12:00:00	000000014	ПоступлениеТоваров	МИРС, ООО	Основной	75 000,00
01.04.2021 12:00:01	000000031	Продажа товаров	МИРС, ООО	Основной	15 000 000,00

Рис. 2.5 Результат группировки динамического списка

5) Реализация отборов в динамическом списке программно.

Для реализации возможности фильтрации содержимого динамического списка на форме списка можно пользоваться отборами. При желании пользователь может их использовать напрямую – через кнопку «Еще…» и выбор пункта 

. В открывшемся окне на закладке «Отбор» пользователь может настроить необходимую ему фильтрацию.

Однако, такая настройка требует от пользователя определенных умений и не очень удобна для быстрой фильтрации.

Реализуем возможность установки значений некоторых, наиболее употребительных фильтров прямо на форме путем установки флагом и выбора значений фильтрации. Само наложение отбора будет производиться программно по нажатию на кнопку «Фильтровать».

Создадим дополнительные реквизиты, команду и элементы формы. Добавим на форму реквизиты формы «ФлагУчитыватьКонтрагента» и «ФлагУчитыватьСклад» типа Булево. Также добавим два вспомогательных реквизита формы «ВыбранныйКонтрагент» (тип СправочникСсылка.Контрагенты) и «ВыбранныйСклад» (тип СправочникСсылка. Склады»). На форму вынесем их, объединим в группу без отображения с горизонтальной группировкой и расположив ее над списком. Во всех четырех добавленных в эту группу элементов формы надо установить свойство «ПоложениеЗаголовка» в значение «Нет», чтобы не отображались лишние заголовки.

Создадим также команду формы «Фильтровать» и также вынесем ее на форму, добавим в группу.

В итоге внешний вид формы приобретет вид как на рис. 2.6.

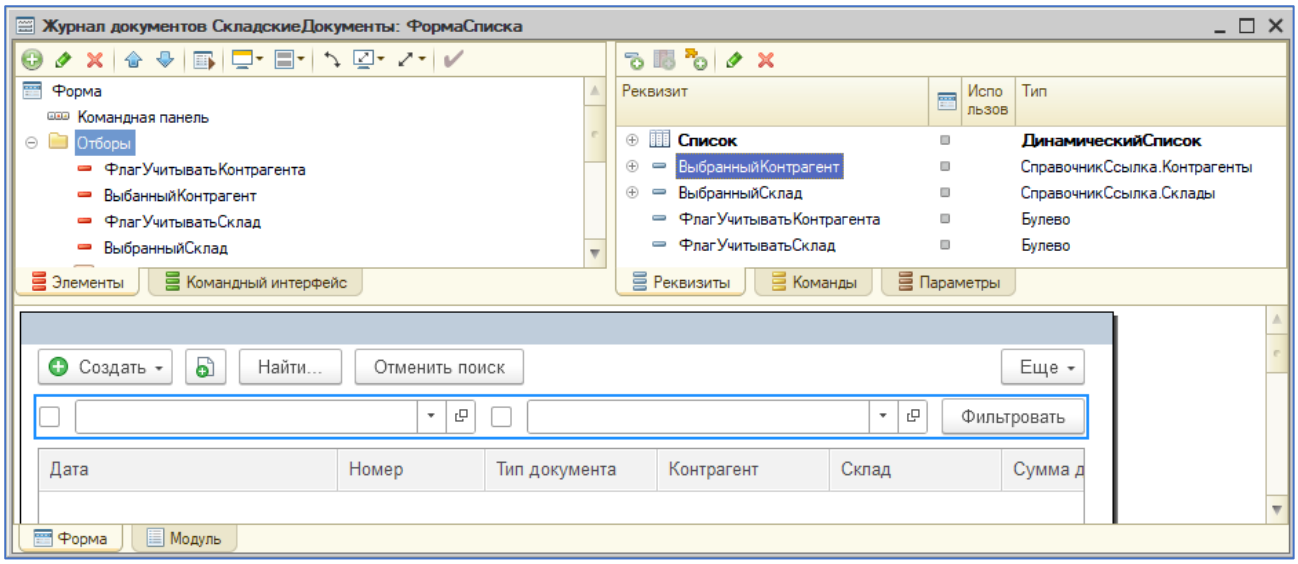


Рис. 2.6. Добавленные реквизиты и команда формы

Далее необходимо реализовать обработчик нажатия кнопки «Фильтровать». В модуле формы реализуем следующий код:

```
&НаКлиенте
Процедура Фильтровать(Команда)
    Для Каждого ЭлементОтбора Из Список.Отбор.Элементы Цикл
        Если ЭлементОтбора.Представление = "Программный отбор" Тогда
            Список.Отбор.Элементы.Удалить(ЭлементОтбора);
        КонецЕсли;
    КонецЦикла;
  ГруппаОтбора =
Список.Отбор.Элементы.Добавить(Тип("ГруппаЭлементовОтбораКомпоновки
Данных"));
  ГруппаОтбора.Использование = Истина;
  ГруппаОтбора.ТипГруппы =
ТипГруппыЭлементовОтбораКомпоновкиДанных.ГруппаИ;
    ГруппаОтбора.Представление = "Программный отбор";
    Если ФлагУчитыватьКонтрагента = Истина Тогда
        Отбор =
ГруппаОтбора.Элементы.Добавить(Тип("ЭлементОтбораКомпоновкиДанных"));
        Отбор.ВидСравнения = ВидСравненияКомпоновкиДанных.Равно;
        Отбор.Использование = Истина;
        Отбор.ЛевоеЗначение = Новый ПолеКомпоновкиДанных("Контрагент");
        Отбор.ПравоеЗначение = ВыбранныйКонтрагент;
    КонецЕсли;
    Если ФлагУчитыватьСклад = Истина Тогда
        Отбор =
ГруппаОтбора.Элементы.Добавить(Тип("ЭлементОтбораКомпоновкиДанных"));
        Отбор.ВидСравнения = ВидСравненияКомпоновкиДанных.Равно;
        Отбор.Использование = Истина;
        Отбор.ЛевоеЗначение = Новый ПолеКомпоновкиДанных("Склад");
```

```
        Отбор.ПравоеЗначение = ВыбранныйСклад;
    КонецЕсли;
КонецПроцедуры
```

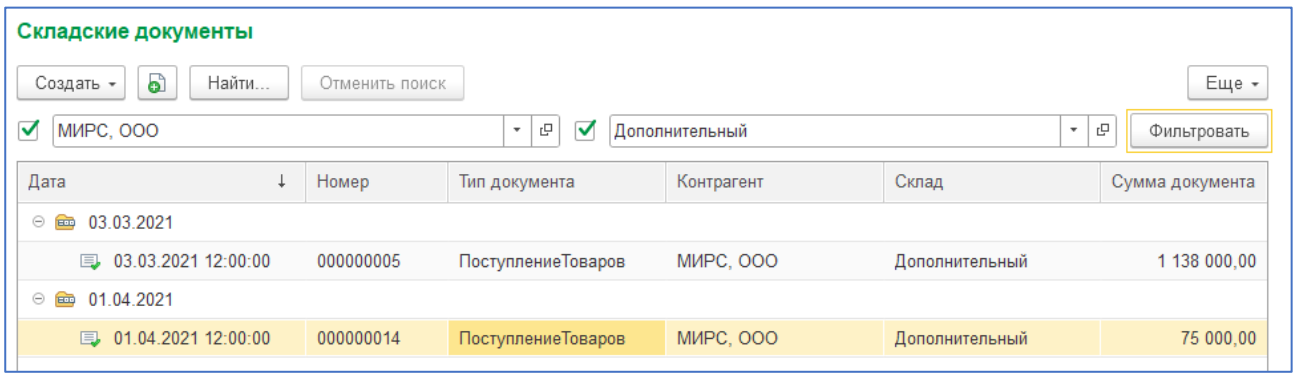


Рис. 2.7. Фильтрация списка по значениям полей «Контрагент» и «Склад»

Результат использования данной функциональности в режиме исполнения представлен на рис. 2.7.

6) Поле переключателя.

На формах может быть создан интересный элемент, делающий использование приложения более удобным, это – поле переключателя.

В справочнике «ФизическиеЛица» есть реквизит «Пол» типа ПеречислениеСсылка.Пол. Также в справочнике «Номенклатура» есть реквизит «ВидНоменклатуры» типа ПеречислениеСсылка.ВидыНоменклатуры. Это поля на формах можно красиво представить полями переключателями.

Установим на форме элемента справочника «Номенклатура» для поля «ВвдНоменклатуры» в свойстве Вид значение «Поле переключателя». Далее в свойстве Вид переключателя можно выбрать одно из значений: Авто, Тумблер, Переключатель. Установим для поля «ВидНоменклатуры» значение «Тумблер».

Помимо этого необходимо также указать список выбора для этого поля (рис. 2.8).

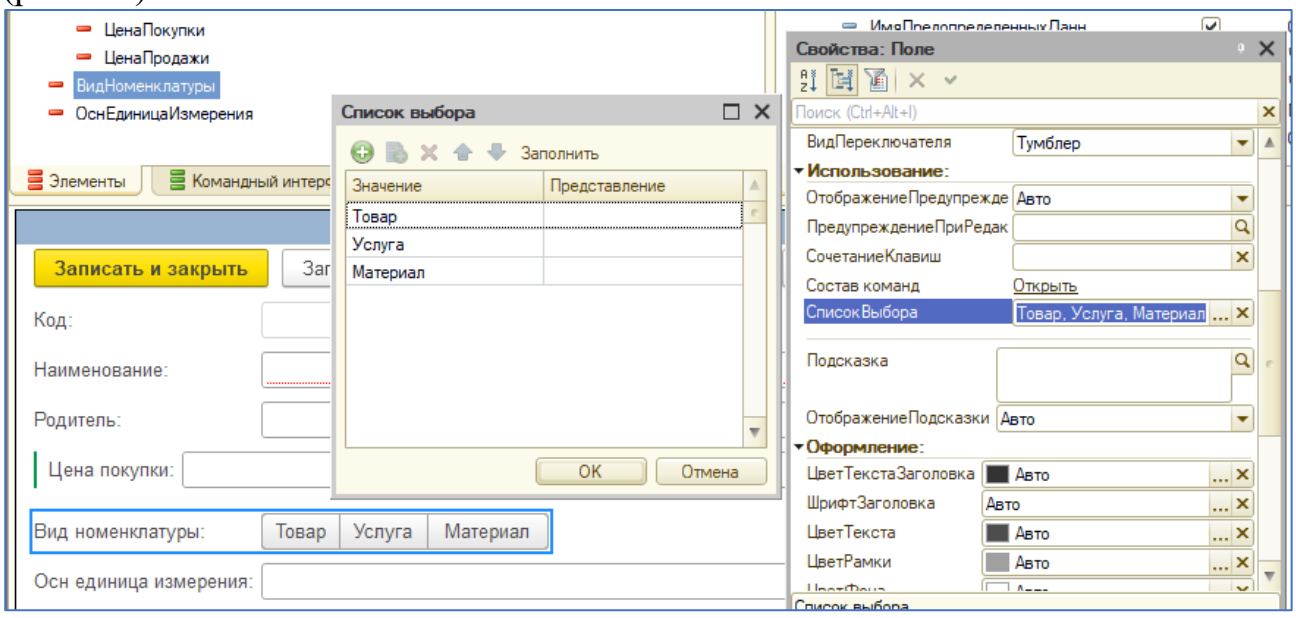


Рис. 2.8. Настройка списка выбора поля переключателя

На форме элемента справочника «ФизическиеЛица» для поля «Пол» выполним аналогичные настройки, но укажем вид переключателя «Переключатель». Полученный внешний вид поля представлен на рис. 2.9.

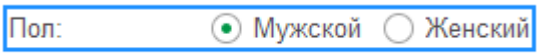


Рис. 2.9. Поле переключателя для реквизита «Пол»

7) Настройка проверки заполнения поля. В ряде алгоритмов важно, чтобы реквизиты не оставались незаполненными при записи объекта. Например, если эти реквизиты используются при проведении документа. В этом случае в конфигураторе для реквизита можно указать свойстве «Проверка заполнения» значение «Выдавать ошибку». Тогда при записи объекта автоматически будет контролироваться заполненность реквизита и в случае его незаполненности выдаваться пользователю сообщение об ошибке и выполнятся отказ в записи.

8) Расчет суммы документа перед записью. В документах «ПоступлениеТоваров» и «ПродажаТоваров» есть реквизиты «СуммаДокумента». Целесообразно на форме их сделать не полями ввода, а полями надписи, чтобы пользователь не мог вручную вводить в них значения. А также следует предусмотреть автоматический расчет суммы документа путем суммирования значений реквизита «Сумма» в табличной части.

Этот расчет можно сделать в обработчике события «Перед записью», разместив в нем следующий код:

```
Процедура ПередЗаписью(Отказ, РежимЗаписи, РежимПроведения)
        СуммаДокумента = Товары.Итог("Сумма");
КонецПроцедуры
```

9) Подстановки и расчеты полей в табличной части документа.

В табличной части документов «ПоступлениеТоваров» и «ПродажаТоваров» есть поля «Цена» и «Сумма» заполнение которых можно сделать автоматически. Рассмотрим реализацию на примере документа «ПоступлениеТоваров». Для второго документа эта функциональность может быть реализована аналогичным образом.

При выборе значения в поле «Номенклатура» из соответствующего справочника целесообразно реализовать автоматическое заполнение поля «Цена», которое подставить из реквизита «ЦенаПродажи» справочника «Номенклатура».

При вводе значения в поле «Количество» целесообразно выполнить вычисление поля Сумма как произведение количества на цену.

Обработчики событий при изменении полей, которые надо реализовать в модуле формы документа «ПоступлениеТоваров», имеют следующий код:

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

```
    Объект.СуммаДокумента=Объект.Товары.Итог("Сумма");
КонецПроцедуры

&НаСервереБезКонтекста
Функция ПолучитьЦенуНоменклатуры(Номенклатура)
    Возврат Номенклатура.ЦенаПокупки;
КонецФункции

&НаКлиенте
Процедура НоменклатураПриИзменении(Элемент)
    Стр = Элементы.Товары.ТекущиеДанные;
    Стр.Цена=ПолучитьЦенуНоменклатуры(Стр.Номенклатура);
    КоличествоПриИзменении(Элемент);
КонецПроцедуры
```

10) Управление видимостью элементов формы.

В некоторых случаях необходимо управлять видимостью полей на форме и скрывать при некоторых условиях часть полей.

Например, реализуем управление видимостью полей в форме документа ПродажаТоваров.

Дополним структуру документа тремя реквизитами: «Доставка», тип Булево; «АдресДоставки», тип Строка, 100; «СтоимостьДоставки». Первый реквизит при его установке в значение Истина будет обозначать что товары по заказу должны быть по адресу, указанному во втором реквизите «АдресДоставки». Стоимость по документу при этом должна быть увеличена на значение в поле «СтоимостьДоставки». Если флаг «Доставка» не установлен, то видимость полей «АдресДоставки» и «СтоимостьДоставки» надо установить равной Ложь.

Код, который реализует эту функциональность на форме, следующий:

```
&НаКлиенте
Процедура ДоставкаПриИзменении(Элемент)
    Если Объект.Доставка Тогда
        Элементы.РеквизитыДоставки.Видимость=Истина;
    Иначе
        Элементы.РеквизитыДоставки.Видимость=Ложь;
    КонецЕсли;
КонецПроцедуры
```

В режиме исполнения (рис. 2.10):

а)

б)

Рис. 2.10. Управление видимостью полей: а) поля скрыты; б) видимость включена

Тема 3. Работа с асинхронными вызовами в системе «1С:Предприятие». Фоновые задания по обработке данных

Одним из основных векторов развития платформы «1С:Предприятие» является возможность ее успешного использования в среде Интернет. Эта среда имеет ряд ограничений, которые отсутствуют в среде настольных приложений [1]:

– ограничения программ, используемых для работы в Интернете (браузеров);

– ограничения, связанные с качеством связи между клиентским приложением и сервером.

Приложения, предназначенные для работы через Интернет, должны быть построены с учетом этих ограничений, поэтому их модель отличается от настольных приложений.

Все интернет-приложения создаются с использованием ***асинхронной (событийно-управляемой) модели*** управления логикой, в отличие от принятой в настольных приложениях синхронной (последовательной) модели.

В асинхронной модели мы можем получить результат только обрабатывая события, которые возникают, когда пользователь делает свой выбор. При этом **на время ожидания выбора код приложения не останавливается** и оно продолжает работать.

Таким образом, получается, что такой код разбивается на две процедуры [1]:

Процедура, которая создает блокирующее окно, ожидающее действия пользователя,

Процедура - обработчик оповещения о том, что пользователь сделал свой выбор и мы можем использовать результат его действий для дальнейшей работы.

Для реализации асинхронной модели в первую очередь требуется возможность описания процедур-обработчиков оповещений, которые будут вызваны системой при завершении выбора пользователя.

Для этого в платформу был добавлен новый тип объектов – *ОписаниеОповещения* [1].

Этот объект имеет конструктор со следующими параметрами:

Новый ОписаниеОповещения(<ИмяПроцедуры>, <Модуль>, <ДополнительныеПараметры>, <ИмяПроцедурыОбработкиОшибки>, <МодульОбработкиОшибки>)

<ИмяПроцедуры> – Указывает имя процедуры-обработчика оповещения, которая будет выполнена после получения ответа пользователя,

<Модуль> – указывает, в каком модуле расположена эта процедура. Этот параметр может иметь следующие типы:

- УправляемаяФорма – процедура расположена в модуле управляемой формы,
- ОбщийМодуль – процедура расположена в общем неглобальном клиентском модуле,
- КомандаКомандногоИнтерфейса – процедура расположена в модуле команды.

Для получения значения модуля у вышеперечисленных объектов добавлено общее свойство *ЭтотОбъект*.

<ДополнительныеПараметры> – значение любого типа, которое будет передано в процедуру-обработчик оповещения при ее вызове.

При вызове указанной процедуры системой ей через параметры передается результат выбора пользователя и значение *ДополнительныеПараметры*.

Для перехода на асинхронную модель в платформу были добавлены методы, аналогичные модальным методам, но, в отличие от них, не блокирующие поток исполнения. Эти методы уже не являются элементами синхронной логики, поэтому могут беспрепятственно использоваться при разработке веб-приложений.

Группа этих методов отличается следующим [1]:

- Их имена начинаются со слова *Показать* либо *Начать*, например *ПоказатьВопрос()* вместо *Вопрос()*, *НачатьПомещениеФайла()* вместо *ПоместитьФайл()*, и так далее,
- Первым параметром принимают объект *ОписаниеОповещения*, указывающий на процедуру модуля, которая будет выполнена после того как пользователь сделает выбор в блокирующем окне,
- Не возвращают значения, вместо этого результат выбора пользователя будет передан в процедуру модуля, описанную объектом *ОписаниеОповещения*.

Рассмотрим подробнее использование механизма оповещения пользователю на практических примерах.

1) Метод ПоказатьОповещениеПользователя.

ПоказатьОповещениеПользователя(<Текст>, <ДействиеПриНажатии>, <Пояснение>, <Картинка>, <СтатусОповещенияПользователя>, <КлючУникальности>)

Этот метод показывает всплывающее окно оповещения, которое автоматически скрывается через 10 секунд. Одновременно может быть показано несколько оповещений. В окне может располагаться текст с пояснением, картинка. Нажатие на окно оповещения либо вызывает процедуру, определенную в описании оповещения, либо осуществляет переход по навигационной ссылке. После этого окно оповещения скрывается.

Параметры:

<Текст> (необязательный). Тип: *Строка*. Текст оповещения.

<ДействиеПриНажатии> (необязательный). Тип: *Строка*; *ОписаниеОповещения*.

Если тип *Строка*, то она содержит навигационную ссылку, по которой будет выполнен переход при нажатии на окно оповещения. После перехода окно оповещения будет закрыто. Если тип *ОписаниеОповещения*, то содержит описание процедуры, которая будет вызвана при нажатии на окно оповещения с параметрами: <*ДополнительныеПараметры*> – значение, которое было указано при создании объекта *ОписаниеОповещения*.

После вызова процедуры окно оповещения будет закрыто. При использовании этого параметра рекомендуется установить текст, содержащий указание на необходимость реакции пользователя на оповещение.

<Пояснение> (необязательный). Тип: *Строка*. Пояснение оповещения.

<Картинка> (необязательный). Тип: *Картинка*. Картинка, которая будет показана в оповещении.

<СтатусОповещенияПользователя> (необязательный). Тип: СтатусОповещенияПользователя. Параметр определяет важность оповещения пользователя. В режимах интерфейса, отличных от Такси, данный параметр игнорируется. Значение по умолчанию: Информация.

<КлючУникальности> (необязательный). Тип: *Строка*. Ключ уникальности оповещения. Если ключ задан и существует открытое оповещение с таким же ключом, то оно будет заменено на новое. Если параметр не указан или имеет тип *Неопределено*, то оповещение считается уникальным. Системные оповещения имеют статус Информация и в качестве значения ключа уникальности содержат навигационную ссылку объекта, редактируемого данной формой. В режимах интерфейса, отличных от Такси, этот параметр игнорируется. Значение по умолчанию: *Неопределено*.

Создадим обработку «*ДиалогиСПользователем*». В ней создадим команду «*ПокажемОповещение*» и вытащим кнопку на форму (рис. 3.1).

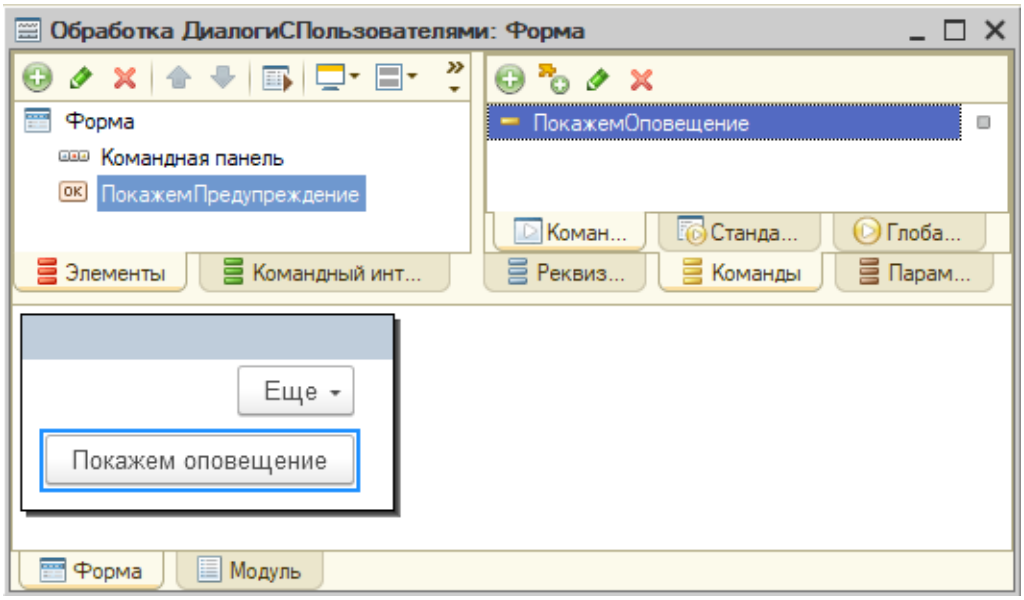


Рис. 3.1. Размещение кнопки на форме

В обработчике (на клиенте) этой команды пропишем следующий код (рис. 3.2):

```
&НаКлиенте
Процедура ПокажемОповещение(Команда)
    Оповещение= Новый ОписаниеОповещения("ПослеЗакрытияВопроса", ЭтотОбъект);
    ПоказатьОповещениеПользователя ("Данные устарели", Оповещение, "Для оповещения щелкните по этому сообщению",
                                    БиблиотекаКартинок.Обновить, СтатусОповещенияПользователя.Важное);
КонецПроцедуры

&НаКлиенте
Процедура солеЗакрытияВопроса(Параметры) Экспорт
     // какой то алгоритм, который что то делает, например,  обновляет данные
КонецПроцедуры
```

Рис. 3.2. Код обработчика команды, привязанной к кнопке

В режиме исполнения (рис. 3.3) получим следующее оповещение:

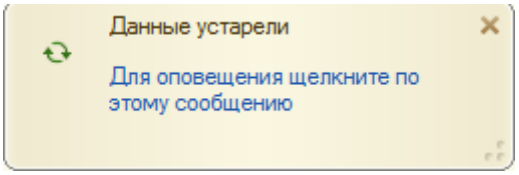


Рис. 3.3. Окно оповещения в режиме исполнения

2) ПоказатьПредупреждение

Если для конфигурации свойство *РежимИспользованияМодальности* установлено в *НеИспользовать*, следует использовать метод *ПоказатьПредупреждение*.

Пример: добавим на форму еще одну команду и кнопку «*ПокажемПредупреждениеПользователю*». В обработчике этой команды (на клиенте) напишем:

```
ПоказатьПредупреждение(,"Выберите документ:", 10, "ВНИМАНИЕ!");
```

Выводит на экран окно предупреждения (рис. 13.4), но не ожидает его закрытия [1].

ПоказатьПредупреждение(<ОписаниеОповещенияОЗавершении>, <ТекстПредупреждения>, <Таймаут>, <Заголовок>)

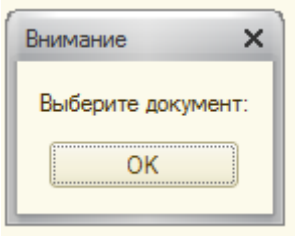


Рис. 3.4. Вывод на экран окна предупреждения

Параметр `<ОписаниеОповещенияОЗавершении>` (необязательный). Тип: *ОписаниеОповещения*. Содержит описание процедуры, которая будет вызвана после закрытия окна предупреждения со следующими параметрами: `<ДополнительныеПараметры>` – значение, которое было указано при создании объекта *ОписаниеОповещения*.

Если параметр не указан, то по завершении никакая процедура вызвана не будет [1].

```
Оп = Новый ОписаниеОповещения("ВыполнитьПослеЗакрытияВопроса",
ЭтотОбъект, Параметр);
```

3) ПоказатьВопрос

Метод выводит на экран окно вопроса, при этом не ожидается завершения ответа пользователя. Синтаксис:

ПоказатьВопрос(<ОписаниеОповещенияОЗавершении>, <ТекстВопроса>, <Кнопки>, <Таймаут>, <КнопкаПоУмолчанию>, <Заголовок>, <КнопкаТаймаута>)

Параметры:
<ОписаниеОповещенияОЗавершении> (обязательный). Тип: ОписаниеОповещения.
Содержит описание процедуры, которая будет вызвана после закрытия окна ввода значения со следующими параметрами [1]:

- *<РезультатВопроса>* - результат выбора пользователя: значение системного перечисления или значение, связанное с нажатой кнопкой. В случае закрытия диалога по истечении времени - значение *Таймаут*,
- *<ДополнительныеПараметры>* - значение, которое было указано при создании объекта *ОписаниеОповещения*.

<ТекстВопроса> (обязательный). Тип: *Строка*; *ФорматированнаяСтрока*. Текст задаваемого вопроса.
<Кнопки> (обязательный). Тип: *РежимДиалогаВопрос*; *СписокЗначений*. Задает состав и текст кнопок диалога, а также, связанные с кнопками значения. При использовании типа *СписокЗначений*:

- *Значение* – содержит значение, связанное с кнопкой. Это значение является возвращаемым значением при выборе кнопки. В качестве значения может использоваться значение перечисления *КодВозвратаДиалога*, а также другие значения;
- *Представление* – задает текст кнопки. Если представление не задано и в качестве значения используется значение перечисления *КодВозвратаДиалога*, то используется стандартное представление;
- *Картинка* – не используется (должна быть пустой);
- *Пометка* – не используется (должна быть пустой).

При использовании типа *СписокЗначений* список не должен быть пустым.
<Таймаут> (необязательный). Тип: *Число*. Интервал времени в секундах, в течение которого система будет ожидать ответа пользователя. По истечении интервала окно вопроса будет закрыто. Если параметр не указан, то время ожидания не ограничено.
Если параметр имеет отрицательное значение, будет сгенерировано исключение. Значение по умолчанию: 0.
<КнопкаПоУмолчанию> (необязательный). Тип: *Произвольный*. Определяет кнопку по умолчанию по типу кнопки или по связанному с ней значению.
<Заголовок> (необязательный). Тип: *Строка*. Содержит заголовок окна вопроса.
<КнопкаТаймаута> (необязательный). Тип: *Произвольный*. Определяет кнопку (по типу кнопки или по связанному с ней значению), на которой отображается количество секунд, оставшихся до истечения таймаута.
Возвращаемое значение: результат выбора пользователя будет передан в метод, описанный параметром *<ОписаниеОповещенияОЗавершении>*.

Пример использования: добавим еще одну команду в нашу обработку «ПокажемВопрос»:

```
&НаКлиенте
Процедура ПокажемВопрос(Команда)
    Режим = РежимДиалогаВопрос.ДаНет;
    Оповещение = Новый ОписаниеОповещения("ПослеЗакрытияВопроса", ЭтаФорма, Параметры);
    ПоказатьВопрос(Оповещение, "Продолжить выполнение операции? ",  Режим, 5);
    //...

КонецПроцедуры
&НаКлиенте
Процедура ПослеЗакрытияВопроса(Результат, Параметры) Экспорт
    Если Результат = КодВозвратаДиалога.Нет Тогда
        Сообщить("Выбрано НЕТ");
        Возврат;
    Иначе
        Сообщить("Выбрано ДА");
        //...
    КонецЕсли;
КонецПроцедуры
```

Рис. 3.5. Код обработчика команды «ПокажемВопрос»

4) Вывод произвольного текста в панель состояния

Метод Состояние() Выводит текст в панель состояния [1].

Состояние(<ТекстСообщения>, <Прогресс>, <Пояснение>, <Картинка>)

Параметры:

<ТекстСообщения> (необязательный). Тип: *Строка*. Строка, предназначенная для вывода в панель состояния. Если параметр не указан, возобновляется вывод системного текста в панель состояния.

<Прогресс> (необязательный). Тип: *Число*. Значение индикатора прогресса (от 1 до 100).

Если не задан, индикатор прогресса не отображается.

<Пояснение> (необязательный). Тип: *Строка*. Текст пояснения.

<Картинка> (необязательный). Тип: *Картинка*. Картинка.

```
&НаКлиенте
Процедура ПоказатьСостояние(Команда)
    max=150;
    Для a=1 по max Цикл
        ПроцентВыполнения=a/max*100;
        Состояние("Идет загрузка", ПроцентВыполнения,"...надо подождать..." , БиблиотекаКартинок.Задача);
    КонецЦикла;
КонецПроцедуры
```

Рис. 3.6. Код обработчика команды «ПоказатьСостояние»

На рис. 3.6 представлен код обработчика команды «ПоказатьСостояние», на рис. 3.7 показан вывод панели состояния в режиме исполнения.

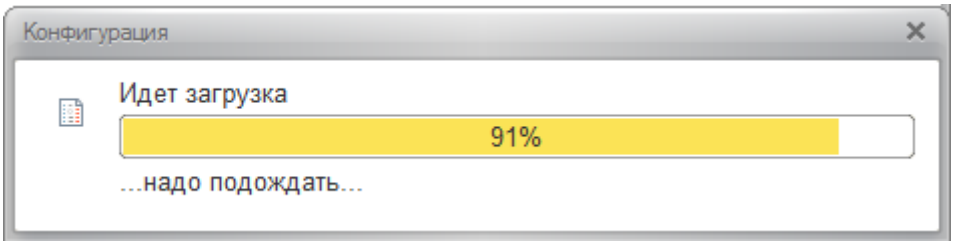


Рис. 3.7. Вывод панели состояния

5) **Фоновые задания по обработке данных.**

При разработке конфигураций следует избегать длительных вызовов из клиентского кода в серверный. Все длительные серверные вызовы, которые могут выполняться более 8 секунд в обычных сценариях работы пользователя, следует выполнять асинхронно, с помощью фонового задания [1].

К таким операциям относятся: формирование отчета, групповая обработка объектов, загрузка или выгрузка данных в другое приложение, заполнение больших табличных частей и т.п.

В противном случае такие вызовы могут привести к потере работоспособности приложения или затруднению работы с ним.

Кроме того, использование фоновых заданий может быть полезно при решении задач выполнения с определенной периодичностью некоторой обработки информации, которая может быть и не особенно длительно, но ее надо выполнять через заданный интервал времени постоянно, например, мониторинг (проверка) состояния, загрузка потоковых данных и т.п.

Реализуем просто учебный пример по использованию фонового задания в целях мониторинга.

Создадим перечисление «СтатусыСобытия»: «Запланировано», «Исполнено», «Просрочено», «Отменено».

Создадим документ «ПлановоеСобытие» с простейшей структурой: реквизиты «ПлановаяДатаСобытия», тип Дата и «СтатусСобытия», тип ПеречислениеСсылка.СтатусыСобытия.

Создадим форму списка для этого документа, на которой выведем реквизиты: Дата, Номер, ПлановаяДатаПоставки, СтатусПоставки. Для динамического списка настроим условное оформление (рис. 3.8).

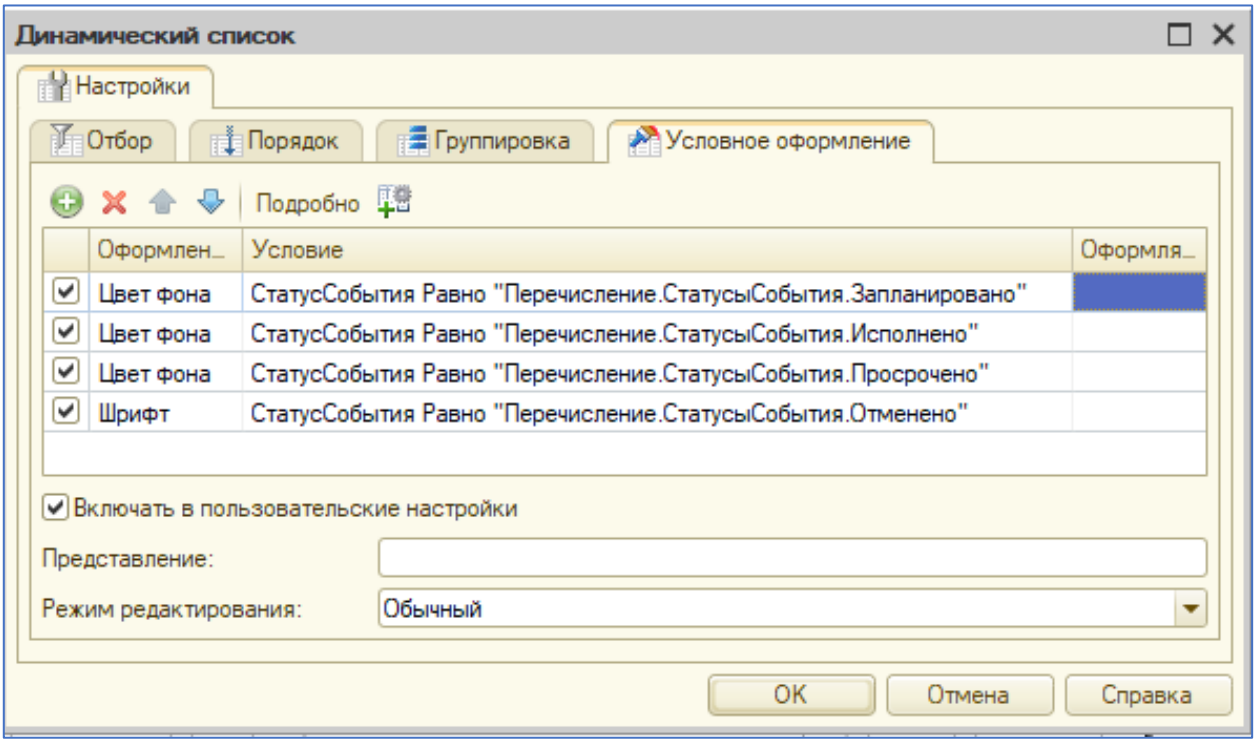


Рис. 3.8. Настройка условного оформления динамического списка

Чтобы открыть окно, представленное на рис. 3.8, необходимо в свойствах основного реквизита формы списка Список найти свойство «Настройка списка» и нажать гиперссылку Открыть.

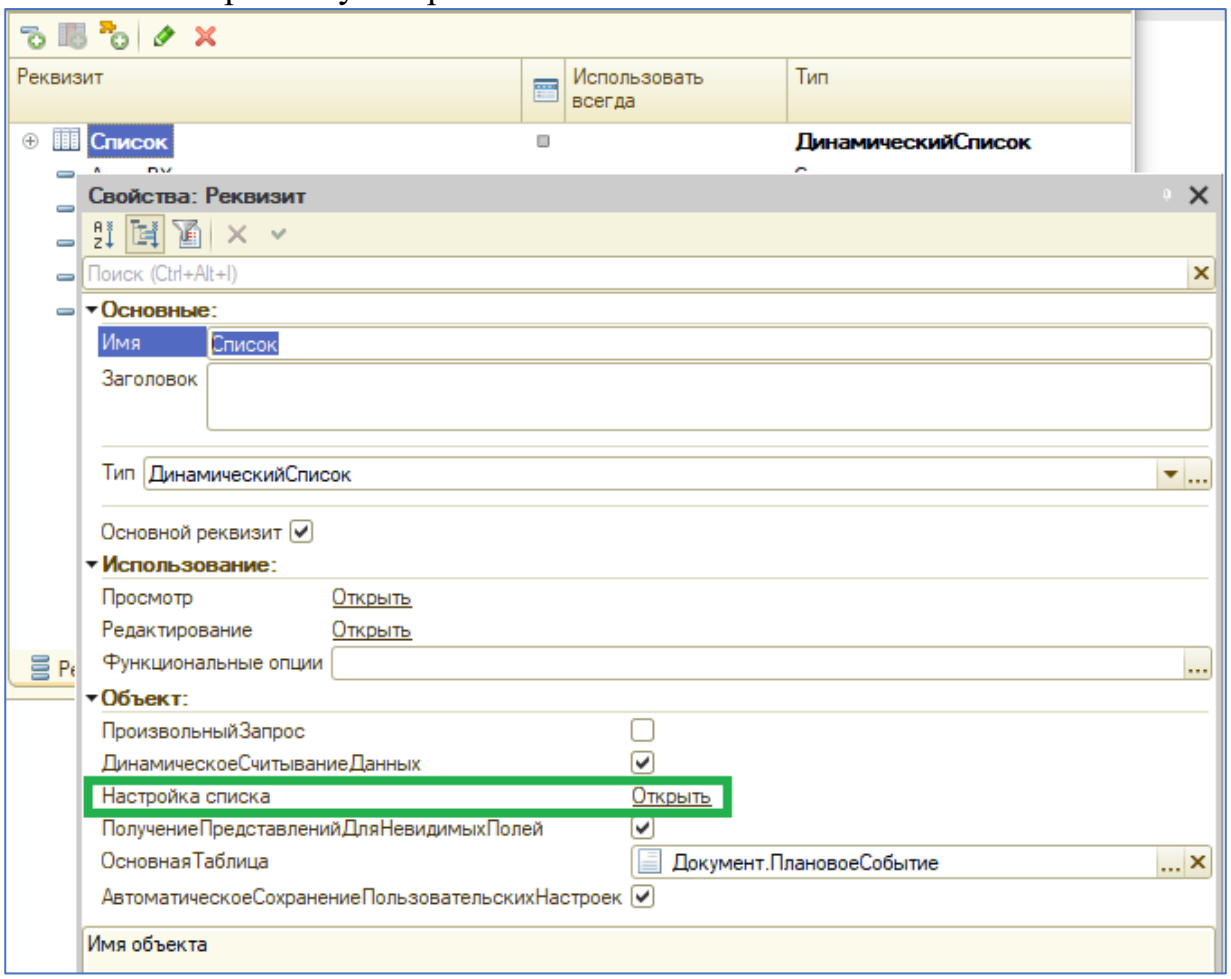


Рис. 3.10. Настойка динамического списка

Результат в пользовательском режиме будет выглядеть следующим образом (рис. 3.11).

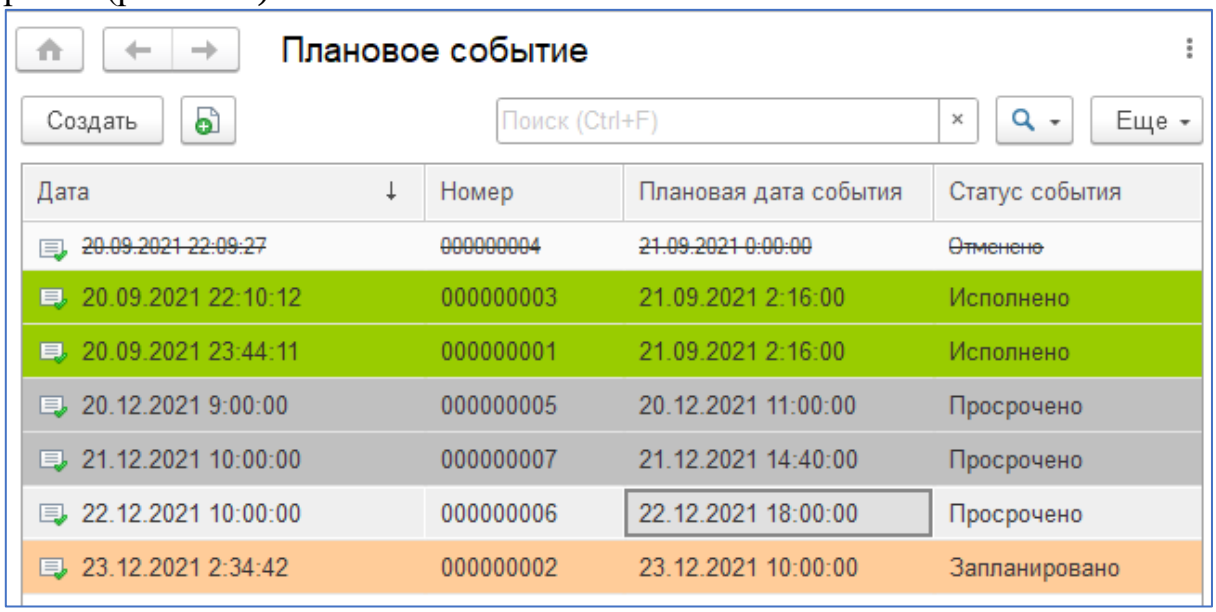


Рис. 3.11. Результат условного оформления списка в пользовательском режиме

В модуле формы реализуем следующие обработчики событий:

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)
   Интервал=Монитор.ПолучитьИнтервалЗапускаФоновыхЗаданий();
   Если Интервал<>0 Тогда
      Попытка
   ПодключитьОбработчикОжидания("ОбработатьОжидание", Интервал, Ложь);
      Исключение
      КонецПопытки;
   КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ОбработатьОжидание()
   ОбработатьОжиданиеНаСервере();
    Элементы.Список.Обновить();
КонецПроцедуры

&НаСервере
Процедура ОбработатьОжиданиеНаСервере()
   ФоновоеЗадание=ФоновыеЗадания.Выполнить("Монитор.ПроизвестиФонову
юПроверку");
КонецПроцедуры

&НаКлиенте
Процедура ПриЗакрытии(ЗавершениеРаботы)
   ОтключитьОбработчикОжидания("ОбработатьОжидание");
КонецПроцедуры
```

Для подключения выполнения фонового задания с определенной периодичностью реализуем задание интервала с помощью константы.

Создадим новую константу «ПериодичностьМониторинга» (тип Число, длина 7), в которой будем задавать интервал в секундах между вызовами фонового задания.

Обработчик фонового задания разместим в общем модуле с названием «Монитор». В этом общем модуле разместим следующий код:

```
Функция ПолучитьИнтервалЗапускаФоновыхЗаданий() Экспорт
      Возврат Константы.ПериодичностьМониторинга.Получить();
КонецФункции

Процедура ПроизвестиФоновуюПроверку() Экспорт
      Запрос = Новый Запрос;
      Запрос.Текст =
            "ВЫБРАТЬ
            |      ПлановоеСобытие.Ссылка КАК Ссылка,
```

```
                    |       ПлановоеСобытие.СтатусСобытия КАК
СтатусСобытия,
                    |       ПлановоеСобытие.ПлановаяДатаСобытия КАК
ПлановаяДатаПоставки
                    |ИЗ
                    |       Документ.ПлановоеСобытие КАК ПлановоеСобытие
                    |ГДЕ
                    |       ПлановоеСобытие.СтатусСобытия =
ЗНАЧЕНИЕ(Перечисление.СтатусыСобытия.Запланировано)
                    |       И ПлановоеСобытие.ПлановаяДатаСобытия <
&ТекДата";
            Запрос.УстановитьПараметр("ТекДата", ТекущаяДата());
            РезультатЗапроса = Запрос.Выполнить();
            ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();
            Пока ВыборкаДетальныеЗаписи.Следующий() Цикл

   ДокОбъект=ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

   ДокОбъект.СтатусСобытия=Перечисления.СтатусыСобытия.Просрочено;
                    ДокОбъект.Записать();
            КонецЦикла;
      КонецПроцедуры
```

Запустим конфигурацию в режиме исполнения и протестируем разработанный механизм мониторинга. Сначала зададим в константе интервал в секундах равный 10. Далее введем несколько событий на разные даты (как уже прошедшие, так и будущие со статусом «Запланировано». И подождем время 10 сек. В случае успешного запуска выполнения фонового задания произойдет запуск фоновой обработки, по результатам которой события в списке, дата и врем которых раньше, чем текущая дата, получат статус «Просрочено». Строки, соответствующие этим событиям, будут оформлены согласно установленному для статусов условному оформлению.

Итак, фоновое задание в системе «1С:Предприятие» – это объект встроенного языка. Он является частью механизма заданий. Фоновые задания предназначены для выполнения прикладных задач асинхронно. Они позволяют нам производить какие-либо вычисления в системе незаметно для пользователя, то есть в фоне. К примеру, в типовых конфигурациях, в то время как пользователь работает, происходит выполнение различного рода сервисных фоновых заданий. Об этом может свидетельствовать записи журнала регистрации, в котором фиксируется факт выполнения таких действий. Причем на работу пользователя это никак не влияет, он просто их не замечает.

В «1С:Предприетие» фоновое задание имеет некоторые ограничения. Поскольку оно выполняется на стороне сервера, то нет возможности интерактивной работы с пользователем. Например, нельзя вывести сообщение, ну и вообще какую-то информацию [1].

Тема 4. Организация решения задач оперативного учета в «1С:Предприятие». Проведение документов по регистрам остатков и оборотов

Многие задачи, которые решаются на платформе «1С:Предприятие», можно классифицировать как управленческие или учетные. Значения контролируемых показателей можно разделить на группы [2]:

Показатели остатка. Используются для учета явлений, которые прирастают, то убывают, но нам необходимо знать их состояние на последний (или на каждый) момент времени. Примеры таких показателей: «количество товара на складе», «количество студентов, имеющих академическую задолженность» и др. отличительным критерием этих показателей является то, что текущее состояние показателей остатка тесно связано с прошлым их состоянием.

Оборотные показатели. Оборотные показатели характеризуют движение за какой-то период, причем учитываются явления, растущие только в одну сторону. Информация выдается по совокупному обороту этого явления за указанный период. Для этих показателей критериями отнесения являются:

– наличие периода в определении;

– независимость его состояния за текущий период от состояния за прошлый период.

Показатели состояния. Это класс показателей, характеризующих состояния чего-то. С помощью таких показателей удобно отражать разовые явления, устанавливающие некие показатели в определённое состояние. Критерием отнесения к этому виду показателей является следующий: устанавливаемое состояние в принципе не зависит от прошлого состояния, но будет действовать не только в рамках какого-то жесткого периода, а или вечно, или до следующе смены состояния. Смена состояния моет производиться в произвольный момент. Примеры таких показателей: курсы валют, котировки акций, статус заказа и др.

Задача любой учетной или управленческой системы мгновенно выдавать информацию по состоянию того или иного показателя. Регистры – это объекты, которые предназначены для хранения и практически мгновенной выдачи значений тех или иных показателей в произвольных разрезах.

Регистры накопления с двумя видами (остатков и оборотов) служат для хранения показателей остатков и оборотов и формирования текущих итогов. Хранение показателей третьего вида обеспечивают регистры сведений [2].

В рамках практического занятия создадим конфигурацию для автоматизации торгового предприятия, позволяющую вести учет торговых операций (покупки, продажи товара, контроля остатка товара на складе и взаиморасчетом с контрагентами).

Работа с регистрами накопления связана с несколькими понятиями:

– измерения;

– ресурсы;

– реквизиты;

– регистратор;
– период записи;
– период рассчитанных итогов;
– агрегаты.

Структура регистра накопления включает набор измерений, ресурсов, могут присутствовать реквизиты.

Измерения – это объекты, в разрезе которых ведется учет. ***Ресурсы*** – показатели и характеристики, которые требуется учитывать. ***Реквизиты*** служат для хранения вспомогательной информации о конкретных записях в регистре.

При проектировании структуры регистров следует учитывать следующие аспекты [2]:

– измерениями регистров могут быть объекты ссылочных или примитивных типов данных. Рекомендуется использовать именно ссылочные;
– количество измерений определяет размерность регистра и, следовательно, от него сильно зависит физический размер таблиц регистра;
– тип ресурсов регистра накопления всегда Число;
– реквизиты – это дополнительная информация о каждом движении (записи) регистра.

Регистр накопления физически хранится в двух таблицах: первая хранит записи движений (приращения); вторая – рассчитанные итоги по этим записям.

Основная функциональность регистра связана с оперативным предоставлением информации о состоянии показателей (ресурсов).

При получении итогов система всегда посчитает их правильно, но чем ближе к реальной дате граница периода рассчитанных данных в таблице итогов, тем быстрее система будет выдавать требуемые данные. При получении итогов помесячно система берет уже готовые итоги, а если требуется получить итоги на некруглую дату учитывает еще движения от ближайшей «круглой» даты. Управлять положением границы периода рассчитанных итогов можно программно и в пользовательском режиме (рис. 4.1).

Доступ в пользовательском режиме осуществляется через функции для технического специалиста. Далее надо выбрать в разделе «Стандартные» пункт «Управление итогами». Итогами можно управлять для всех регистров сразу или отдельно.

Можно также управлять установкой режима разделения итогов (рис. 4.2), т.е. разрешать или не разрешать системе при параллельной записи вместо ожидания предоставления возможности корректировки занятых другими сеансами итоговых записей. Эта возможность повышает параллельность работы в системе, но принципиально может приводить к существенному разрастанию таблицы итогов, поскольку сворачивание разделенных итогов выполняется только при пересчете итогов.

Чтобы этот режим активировать, необходимо сначала установить флаг для регистра «Разрешить разделение итогов», а далее уже в пользовательском режиме «Включить разделение итогов».

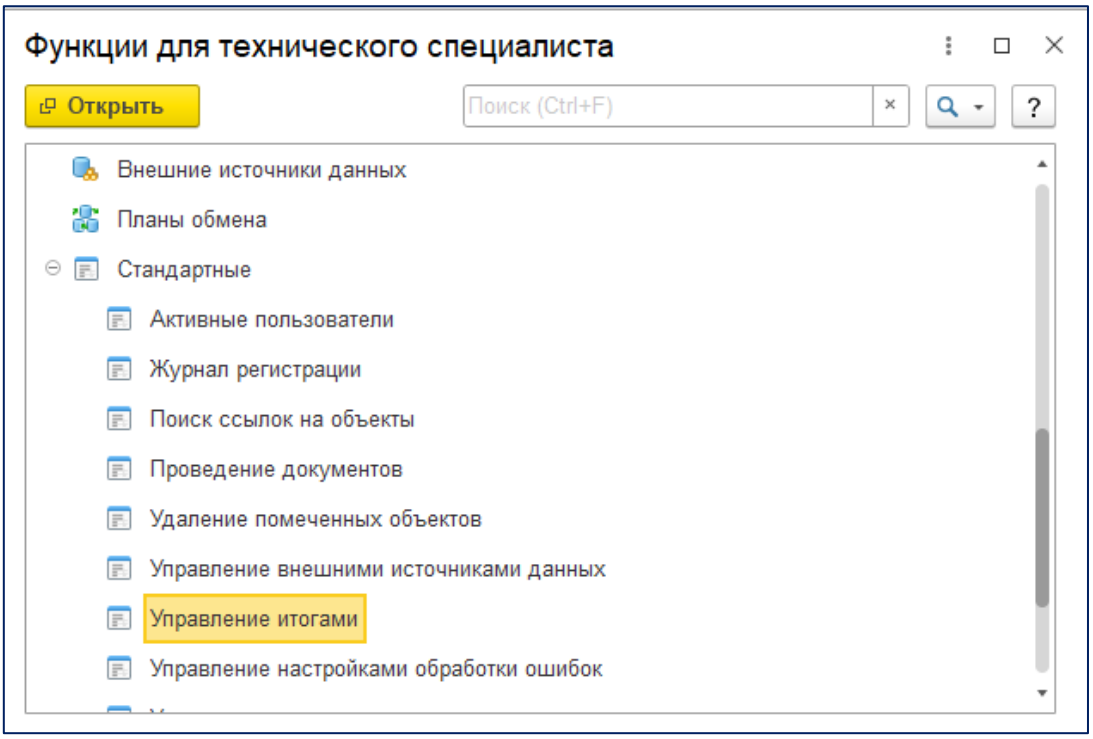


Рис. 4.1 Пункт «Управление итогами» в пользовательском режиме

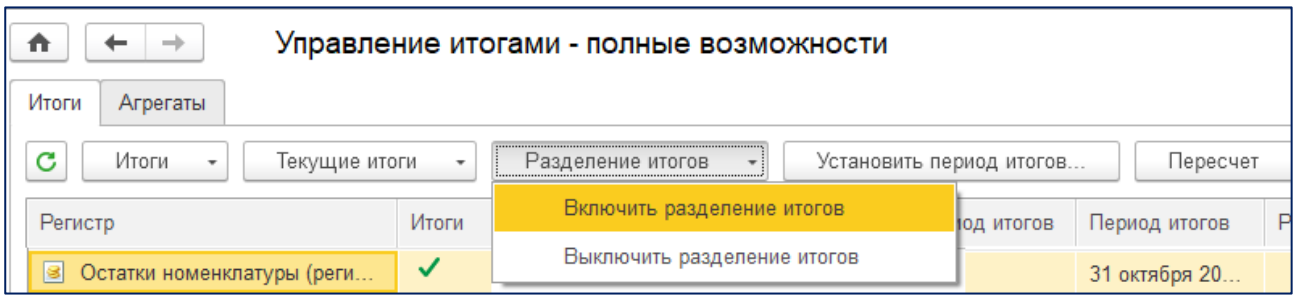


Рис. 4.2. Установка режима разделения итогов

1) Создадим регистр накопления «ОстаткиНоменклатуры». Вид регистра: остатки. Его структура следующая.

Измерения:

«Номенклатура» – тип СправочникСсылка. Номенклатура;

«Склад» – тип СправочникСсылка.Склады;

«Партия» – тип ДокументСсылка.ПоступлениеТоваров.

Ресурсы:

«Количество», тип Число, длина 15, точность 2.

«Сумма», тип Число, длина 15, точность 2.

Регистраторами для этого регистра назначим документы «ПоступлениеТоваров» и «ПродажаТоваров».

2) Создадим регистр накопления оборотов «Продажи». Вид регистра – обороты. Структура регистра включает:

Измерения:

«Контрагент» – тип СправочникСсылка.Контагенты;

«Номенклатура» – тип СправочникСсылка.Номенклатура.

Ресурсы:

«Количество» – тип Число, длина 10, точность 3.

«Сумма» – тип Число, длина 15, точность 2.

Регистратором для этого регистра выступает документ «ПродажаТоваров».

3) С помощью конструктора движений реализуем процедуру проведения документа «ПоступлениеТоваров» в модуле документа.

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)
    Движения.ОстаткиНоменклатуры.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
    // регистр ОстаткиНоменклатуры Приход
        Движение = Движения.ОстаткиНоменклатуры.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
        Движение.Период = Дата;
        Движение.Номенклатура = ТекСтрокаТовары.Номенклатура;
        Движение.Склад = Склад;
        Движение.Партия =Ссылка;
        Движение.Количество = ТекСтрокаТовары.Количество;
        Движение.Сумма = ТекСтрокаТовары.Сумма;

    КонецЦикла;
КонецПроцедуры
```

4) Реализуем также процедуру обработки проведения документа «ПродажаТоваров».

Перед этим создадим дополнительные объекты:

Перечисление «МетодыСписания» со значениями «FIFO» и «LIFO».

Регистр сведений «УчетнаяПолитика» со следующей структурой: не имеет измерений, есть один ресурс «МетодСписания» типа ПеречислениеСсылка.МетодыСписания. Режим записи – независимый, Периодичность – по позиции регистратора.

```
Процедура ОбработкаПроведения(Отказ, Режим)
    // регистр ОстаткиНоменклатуры Расход
    Движения.ОстаткиНоменклатуры.Записывать = Истина;
    // ОБЕСПЕЧЕНИЕ НЕИЗМЕННОСТИ ДАННЫХ МЕЖДУ РАСЧЕТОМ И ОКОНЧАНИЕМ ПРОВЕДЕНИЯ
    ТаблицаДляБлокирования=Товары.Выгрузить(, "Номенклатура");
    ТаблицаДляБлокирования.Колонки.Добавить("Склад");
    ТаблицаДляБлокирования.ЗаполнитьЗначения(Склад, "Склад");
    Блокировка = Новый БлокировкаДанных;
```

```
    ЭлементБлокировки =
Блокировка.Добавить("РегистрНакопления.ОстаткиНоменклатуры");
    ЭлементБлокировки.Режим=РежимБлокировкиДанных.Исключительный;
    ЭлементБлокировки.ИсточникДанных=ТаблицаДляБлокирования;
    ЭлементБлокировки.ИспользоватьИзИсточникаДанных("Номенклатура","Н
оменклатура");
    ЭлементБлокировки.ИспользоватьИзИсточникаДанных("Склад","Склад");
    Блокировка.Заблокировать();
    // УДАЛЕНИЕ СОБСТВЕННЫХ СТАРЫХ ДВИЖЕНИЙ ПО РЕГИСТРУ
ОСТАТКИ НОМЕНКЛАТУРЫ
    Движения.ОстаткиНоменклатуры.Записать();
    // ПОЛУЧЕНИЕ ДАННЫХ ТОЛЬКО ПО ТОВАРАМ
    Запрос = Новый Запрос;
    Запрос.Текст =
      "ВЫБРАТЬ
      |       ПродажаТоваровТовары.Номенклатура,
      |       СУММА(ПродажаТоваровТовары.Количество) КАК Количество,
      |       СУММА(ПродажаТоваровТовары.Сумма) КАК Сумма,
      |       ПродажаТоваровТовары.Ссылка.Склад
      |ПОМЕСТИТЬ ТабДок
      |ИЗ
      |       Документ.ПродажаТоваров.Товары КАК ПродажаТоваровТовары
      |ГДЕ
      |       ПродажаТоваровТовары.Ссылка = &Ссылка
      |       И ПродажаТоваровТовары.Номенклатура.ВидНоменклатуры <>
ЗНАЧЕНИЕ(Перечисление.ВидыТоваров.Услуга)
      |
      |СГРУППИРОВАТЬ ПО
      |       ПродажаТоваровТовары.Номенклатура,
      |       ПродажаТоваровТовары.Ссылка.Склад
      |;
      |
      |////////////////////////////////////////////////////////////////////////////
      |ВЫБРАТЬ
      |       ТабДок.Номенклатура КАК Номенклатура,
      |       ТабДок.Количество КАК Количество,
      |       ТабДок.Сумма КАК Сумма,
      |       ТабДок.Склад,
      |       ЕСТЬNULL(ОстаткиНоменклатурыОстатки.КоличествоОстаток, 0)
КАК КоличествоОстаток,
      |       ЕСТЬNULL(ОстаткиНоменклатурыОстатки.СуммаОстаток, 0) КАК
СуммаОстаток,
      |       ОстаткиНоменклатурыОстатки.Партия КАК Партия,
      |       ОстаткиНоменклатурыОстатки.МоментВремени КАК
МоментВремени
```

```
        |ИЗ
        |       ТабДок КАК ТабДок
        |               ЛЕВОЕ СОЕДИНЕНИЕ
РегистрНакопления.ОстаткиНоменклатуры.Остатки(
        |                       &Момент,
        |                       (Номенклатура, Склад) В
        |                               (ВЫБРАТЬ
        |                                       ТабДок.Номенклатура,
        |                                       ТабДок.Склад
        |                               ИЗ
        |                                       ТабДок КАК ТабДок)) КАК
ОстаткиНоменклатурыОстатки
        |               ПО ТабДок.Номенклатура =
ОстаткиНоменклатурыОстатки.Номенклатура
        |
        |УПОРЯДОЧИТЬ ПО
        |       Номенклатура,
        |       МоментВремени ВОЗР,
        |       Партия
        |ИТОГИ
        |       МАКСИМУМ(Количество),
        |       СУММА(КоличествоОстаток),
        |       СУММА(СуммаОстаток),
        |ПО
        |       Номенклатура";
    ТекущаяУчетнаяПолитика=РегистрыСведений.УчетнаяПолитика.Получить
Последнее(Дата);
    Если
ТекущаяУчетнаяПолитика.МетодСписания=Перечисления.МетодыСписанияПа
ртии.LIFO Тогда
        Запрос.Текст=СтрЗаменить(Запрос.Текст,"МоментВремени ВОЗР",
"МоментВРемени УБЫВ");
        КонецЕсли;
    Если Режим=РежимПроведенияДокумента.Оперативный Тогда

    Запрос.УстановитьПараметр("Момент", Неопределено);
Иначе
    Запрос.УстановитьПараметр("Момент", МоментВремени());
    КонецЕсли;
    Запрос.УстановитьПараметр("Ссылка", Ссылка);

    РезультатЗапроса = Запрос.Выполнить();

        ВыборкаНоменклатура                                             =
РезультатЗапроса.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
```

```
    Пока ВыборкаНоменклатура.Следующий() Цикл
        // Вставить обработку выборки ВыборкаНоменклатура
        Если
(ВыборкаНоменклатура.КоличествоОстаток)<ВыборкаНоменклатура.Количест
во Тогда
                Отказ = Истина;
                Сообщение = новый СообщениеПользователю;
                Нехватка = ВыборкаНоменклатура.Количество-
ВыборкаНоменклатура.КоличествоОстаток;
                Сообщение.Текст = "В документе №"+Номер+"от"+Дата+"не
хватает"+Нехватка+"единиц товара"+ВыборкаНоменклатура.Номенклатура;
                Сообщение.Сообщить();
        Иначе
                КоличествоНадоСписать= ВыборкаНоменклатура.Количество;
                ВыборкаДетальныеЗаписи=ВыборкаНоменклатура.Выбрать();
        Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
                Если
ВыборкаДетальныеЗаписи.КоличествоОстаток<=КоличествоНадоСписать
                        Тогда

    КоличествоКСписанию=ВыборкаДетальныеЗаписи.КоличествоОстаток;

    СтоимостьКСписанию=ВыборкаДетальныеЗаписи.СуммаОстаток;
                        КоличествоНадоСписать=КоличествоНадоСписать-
КоличествоКСписанию;
                Иначе
                        КоличествоКСписанию= КоличествоНадоСписать;
                        Если ВыборкаДетальныеЗаписи.КоличествоОстаток<>0 Тогда

    СтоимостьКСписанию=ВыборкаДетальныеЗаписи.СуммаОстаток/Выборка
ДетальныеЗаписи.КоличествоОстаток*КоличествоКСписанию;
                        Иначе
    СтоимостьКСписанию=ВыборкаДетальныеЗаписи.СуммаОстаток;
                        КонецЕсли;
                        КоличествоНадоСписать=0;
                         КонецЕсли;

        Движение = Движения.ОстаткиНоменклатуры.Добавить();
        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
        Движение.Период = Дата;
        Движение.Номенклатура = ВыборкаДетальныеЗаписи.Номенклатура;
        Движение.Склад=Склад;
        Движение.Склад = ВыборкаДетальныеЗаписи.Партия;
        Движение.Количество = КоличествоКСписанию;
```

```
        Движение.Сумма= СтоимостьКСписанию;
        Если КоличествоНадоСписать<=0 Тогда
            Прервать;
        КонецЕсли;
        КонецЦикла;
        КонецЕсли;
КонецЦикла;
КонецПроцедуры
```

В данной процедуре проведения осуществляется контроль отрицательных остатков, т.е. приложение не дает списать со склада больше товара, чем есть на складе.

Следует также обратить внимание на использование в тексте процедуры управляемой блокировки. Рассмотрим подробнее ее назначение.

Выполнение запроса по любому запросу производится на чтение данных. Поэтому мы не мешаем другим процессам также обращаться к регистру для чтения данных.

Однако, некоторые алгоритмы могут работать некорректно в такой ситуации. В частности, для документа «ПродажаТоваров» возможна ситуация конкурентного проведения нескольких документов с разным временем, затрачиваемым на проведение (например, документы с разным количеством позиций в списке: есть один большой документ и другой маленький). В такой ситуации важно, чтобы не случилась ситуация, при которой большой документ, получив запросом данные для расчета себестоимости, начинает их обрабатывать, а маленький документ в этом время уже списал эти товары, т.е. внес изменения в учет товаров на складе.

Поэтому важно обеспечить низменность данных в информационной базе от начала действия алгоритма до его полного завершения – наложить блокировку данных. При этом блокировку надо наложить избирательно, т.к. излишняя блокировка снижает параллельность работы пользователей и, как следствие, всю эффективность системы.

В приведенном алгоритме мы сначала создаем конструктором объект «БлокировкаДанных». В него в общем случае может входить несколько элементов, каждый для отдельного объекта данных. Указываем режим блокировки. Их существует два: «РежимБлокировкиДанных.Разделяемый» и «РежимБлокировкиДанных.Исключительный». Если выбрать первый вариант блокировки, то данные другим транзакциям будет нельзя изменять, но можно читать. При выборе вариант «РежимБлокировкиДанных.Исключительный» другие транзакции не смогут даже начать читать эти заблокированные данные, пока мы не закончим работать с ними. В нашем случае используется второй вариант блокировки.

С помощью метода УстановитьЗначение() мы может указать отбор по единичному значению, чтобы блокировались только нужные данные. Когда позиций (отбор по нескольким значениям) для блокировки несколько, то используется назначение источника данных нашему элементу блокировки. В

качестве значения передаем таблицу значений, которую мы получаем выгрузкой из табличной части документа.

5) Реализуем проведение документа «ПродажаТоваров» по второму регистру «Продажи». Для этого допишем в процедуре обработки проведения следующий код:

```
Движения.Продажи.Записывать=Истина;
Движение = Движения.Продажи.Добавить();
Движение.Период = Дата;
Движение.Номенкатура = ВыборкаДетальныеЗаписи.Номенклатура
Движение.Контрагент = Контрагент;
Движение.Сумма = ВыборкаДетальныеЗаписи.Сумма;
Движение.Количество = ВыборкаДетальныеЗаписи.Количество;
```

Данный фрагмент надо добавить после следующих строк по реализации движений по регистру «ОстаткиНоменклатуры»:

```
Движение.Количество = КоличествоКСписанию;
Движение.Сумма= СтоимостьКСписанию;
```

6) Реализуем отчет «МатериальнаяВедомость» (рис. 4.3). В качестве источника данных укажем запрос со следующим текстом:

```
ВЫБРАТЬ
   ОстаткиНоменклатурыОстаткиИОбороты.Номенклатура,
   ОстаткиНоменклатурыОстаткиИОбороты.Регистратор,
   ОстаткиНоменклатурыОстаткиИОбороты.ПериодДень,
   ОстаткиНоменклатурыОстаткиИОбороты.КоличествоНачальныйОстаток,
   ОстаткиНоменклатурыОстаткиИОбороты.КоличествоПриход,
   ОстаткиНоменклатурыОстаткиИОбороты.КоличествоРасход,
   ОстаткиНоменклатурыОстаткиИОбороты.КоличествоКонечныйОстаток,
   ОстаткиНоменклатурыОстаткиИОбороты.СуммаКонечныйОстаток
ИЗ
   РегистрНакопления.ОстаткиНоменклатуры.ОстаткиИОбороты(&ДатаНачала,
&ДатаКонца, Авто, , ) КАК ОстаткиНоменклатурыОстаткиИОбороты
```

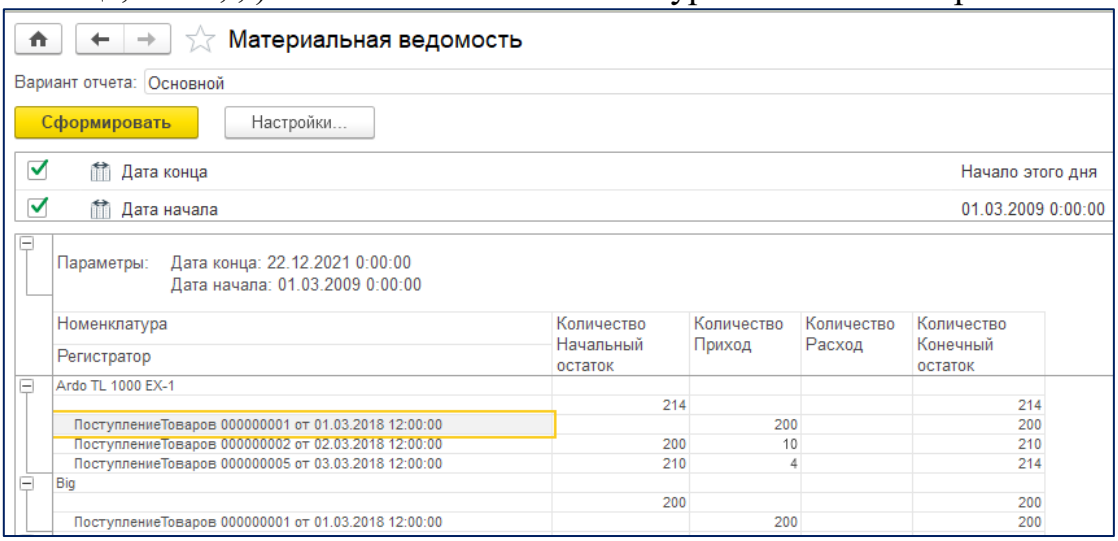


Рис. 4.3. Отчет «МатериальнаяВедомость»

В качестве ресурсов отчета выберем поля: «КоличествоНачальныйОстаток», «КоличествоКонечныйОстаток», «КоличествоПриход», «КоличествоРасход» и «СуммаКонечныйОстаток». В настройках создадим все вложенные группировки: «Номенклатура» и «регистратор». В качестве выбранных полей у группировки Номенклатура оставим поле Авто, а в группировке «Регистратор» в качестве выбранных полей укажем: «КоличествоНачальныйОстаток», «КоличествоКонечныйОстаток», «КоличествоПриход», «КоличествоРасход» и «СуммаКонечныйОстаток».

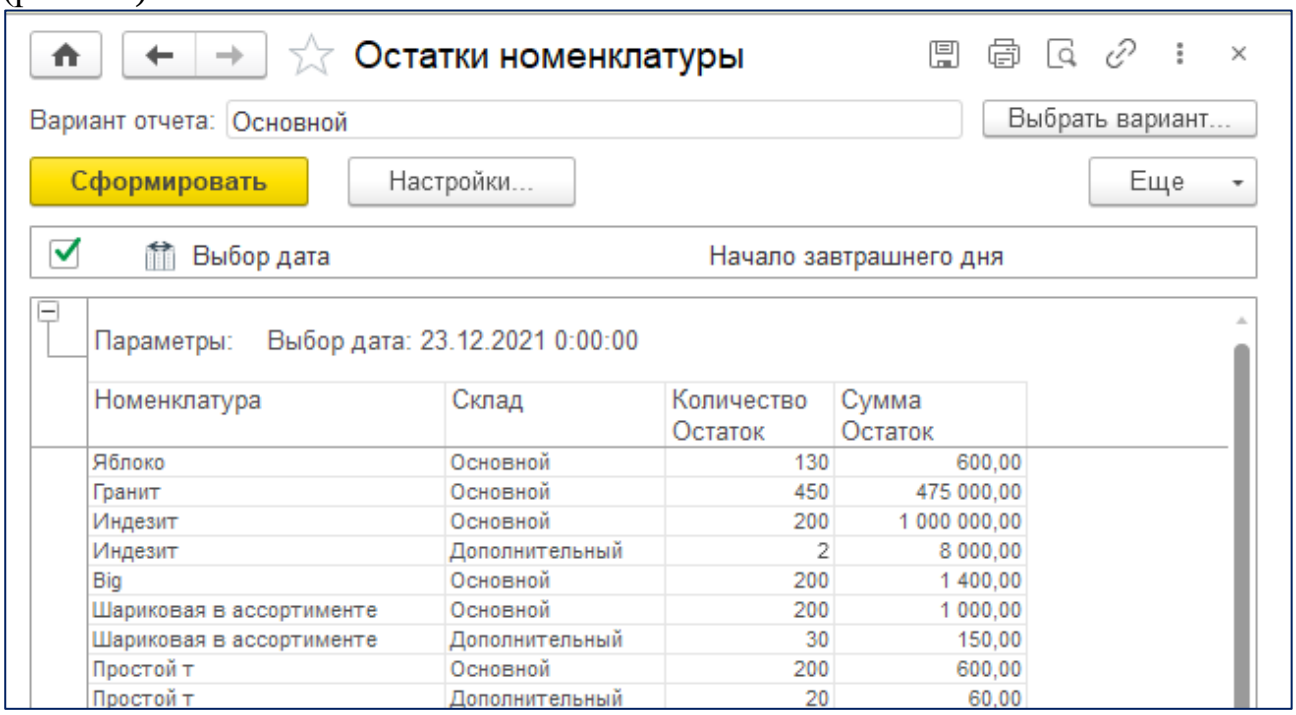
Запустим отчет в режиме исполнения и проверим его работоспособность.

7) Создадим новый отчет с именем «ОстаткиНоменклатуры». В качестве источника данных укажем запрос со следующим текстом:

```
ВЫБРАТЬ
    ОстаткиНоменклатурыОстатки.Номенклатура,
    ОстаткиНоменклатурыОстатки.Склад,
    ОстаткиНоменклатурыОстатки.КоличествоОстаток,
    ОстаткиНоменклатурыОстатки.СуммаОстаток
ИЗ
    РегистрНакопления.ОстаткиНоменклатуры.Остатки(&ВыборДата, )
                                КАК ОстаткиНоменклатурыОстатки
```

В качестве ресурсов отчета выберем поля: «КоличествоОстаток» и «СуммаОстаток». В настройках создадим одну группировку «ДетальныеЗаписи». В качестве выбранных полей укажем: «Номенклатура», «Склад», «КоличествоОстаток», «СуммаОстаток».

Запустим отчет в режиме исполнения и проверим его работоспособность (рис. 4.4).



Номенклатура	Склад	Количество Остаток	Сумма Остаток
Яблоко	Основной	130	600,00
Гранит	Основной	450	475 000,00
Индезит	Основной	200	1 000 000,00
Индезит	Дополнительный	2	8 000,00
Big	Основной	200	1 400,00
Шариковая в ассортименте	Основной	200	1 000,00
Шариковая в ассортименте	Дополнительный	30	150,00
Простой т	Основной	200	600,00
Простой т	Дополнительный	20	60,00

Рис. 4.4. Отчет «ОстаткиНоменклатуры»

Тема 5. Методы и инструментальные средства бизнес-аналитики для решения задач профессиональной деятельности. Организация синтетического бухгалтерского учета в «1С:Предприятие». Проведение документов по регистру бухгалтерии

Хозяйственная деятельность предприятия выступает предметом бухгалтерского учета. Его основная задача – управление финансово-хозяйственной деятельностью предприятия. Решение этой задачи требует использования специфических методов, в частности двойной записи на счетах, образующей замкнутую систему показателей. В приложениях на платформе «1С:Предприятие» реализация этих задач выполняется с использованием трех основных объектов: плана счетов, плана видов характеристик и регистра бухгалтерии. При этом план видов характеристик используется для описания объектов аналитического учета, в разрезе которых будет вестись учет на тех или иных счетах, и будет рассмотрен на последующих занятиях.

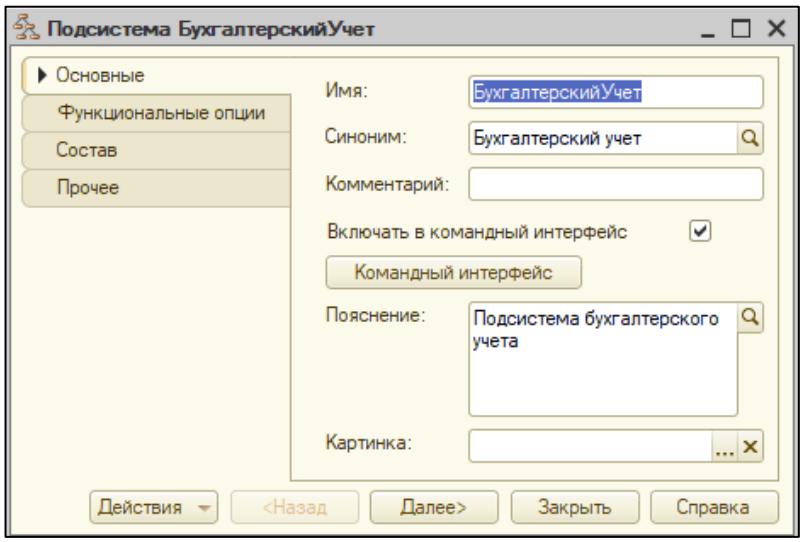


Рис. 5.1. Новая подсистема «БухгалтерскийУчет»

Синтетический учет – это обобщение данных о видах имущества, обязательств и хозяйственных операций по определенным экономическим признакам. Синтетический учет ведется на основных счетах бухучета, которые являются балансовыми [3].

Организуем ведение синтетического учета (основного вида бухгалтерского учета) для отражения хозяйственных операций на счетах бухгалтерского учета с использованием плана счетов и регистра бухгалтерии.

1) Создание подсистемы «БухгалтерскийУчет». Добавим в конфигурацию новую подсистему «БухгалтерскийУчет» (рис. 5.1), к которой будем относить все объекты, создаваемые в рамках решения задач бухучета.

2) Создание плана счетов «Бухгалтерский». Однородные бухгалтерские операции группируются на счетах бухгалтерского учета, которые являются основным разрезом бухгалтерского учета. Счета, входящие в один план счетов, образуют замкнутую систему показателей. Группировка хозяйственных операций по счетам позволяет формировать бухгалтерский баланс и другие формы сводной отчетности.

Создадим в конфигурации основной срез ведения бухучета – новый план счетов «Бухгалтерский». На закладке «Основные» зададим помимо имени все представления объекта, т.к. именно они будут видны пользователю при работе с планом счетов в пользовательском режиме. На закладке «Подсистемы» новый план счетов следует добавить в подсистему «БухгалтерскийУчет».

Настроим на закладке «Данные» его свойства (рис. 5.2). Упрощенно будем считать, что иерархия кодов счетов-субсчетов в нашем плане не будет превышать двух уровней. Поэтому зададим маску кода @@.@@, которая позволит создавать два уровня вложенности: два знака в коде счета и два знака в коде субсчета.

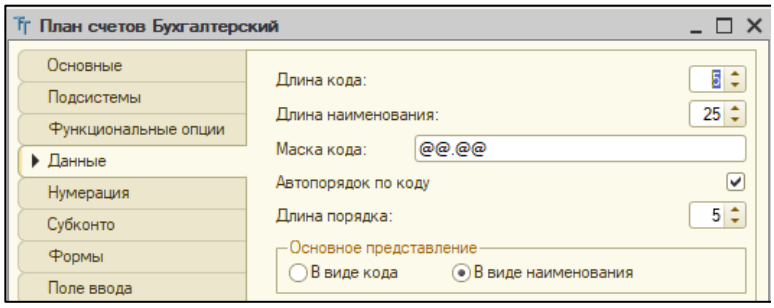


Рис. 5.2. Создание плана счетов и настройка его свойств

В описании метода «Маска» встроенного языка можно почитать перечень и описание назначения символов, которые можно использовать в маске. Специальный символ @, который был нами использован в маске, позволит не только вводить любые буквенно-цифровые символы или пробел в коде счета, но и влияет на его хранение в базе и отображение в диалоге ввода. Его использование также влияет на формирование поля «Порядок», которое нужно для правильной сортировки счетов.

В созданном плане счетов введем предопределенные счета (рис. 5.3).

План счетов Бухгалтерский: Предопределенные счета

Действия

Имя	Код	Наименование	Вид	Забалан…	Порядок
Счета					
Активы	1	Активы	Активный		1
Касса	1.1	Касса	Активный		1. 1
Покупатели	1.2	Покупатели	Активный		1. 2
Товары	1.3	Товары	Активный		1. 3
Обязательства	2	Обязательства	Пассивный		2
Сотрудники	2.1	Сотрудники	Пассивный		2. 1
Поставщики	2.2	Поставщики	Пассивный		2. 2
Капитал	3	Капитал	Пассивный		3
ТоварыНаХранении	6	Товары на хранении	Активный	✔	6

Рис. 5.3. Создание предопределенных счетов

Для каждого счета необходимо указать ряд свойств, представленных в табл. 5.2 [1].

Таблица 5.2

Основные свойства счета

Свойство	Описание
1	2
Имя	Вводится уникальное имя счета, которое позволит получать ссылку на счет. Есть только у предопределенных счетов. Пользователь не может изменить его в режиме исполнения.
Код	Код счета, который будет виден пользователю в поле выбора в случае, если основным представлением счета выбран Код. Пользователь может изменить в пользовательском режиме.
Наименование	Наименование счета, которое будет видно пользователю в поле выбора в случае, если основным представлением счета выбрано Наименование.
Вид	Счет может быть активным, пассивным или активно-пассивным. Это признак определяет тип сальдо на счете. У активных счетов сальдо только дебетовое, у пассивных – кредитовое, а у активно-пассивных зависит от оборотов и может быть как дебетовым, так и кредитовым.
Забалансовый	Если у счета указан этот признак, то счет исключается из правила двойной записи, т.е. счет не может корреспондировать в проводке с другими. Обычно такие счета используются для хранения данных об имуществе, которое нам не принадлежит.
Порядок	Это поле используется для сортировки. Оно заполняется автоматически при создании предопределенных счетов, если при описании маски кода использовались только знаки @ и знаки разделители, например, точка.

Рассмотрим подробнее, для чего используется поле Порядок. Полезность этого поля очевидна, если мы установили в качестве основного представления Наименование и хотим упорядочить по счету, а также если выбран Код в качестве основного представления, но он плохо подходит для сортировки. В табл. 5.2 приведен пример сортировки по коду счета.

Таблица 5.2

Пример сортировки плана счетов по коду

Правильный порядок счетов	**Счета, отсортированные по коду**
1.1	1.1
1.2	1.10
1.3	1.11
1.N	1.2
1.10	1.3
1.11	1.N

Как видно из табл. 5.2 сортировка по коду дает неправильный результат. Поэтому для сортировки используется другое поле – Порядок.

3) Создание регистра бухгалтерии «Проводки». По сути регистр бухгалтерии представляет собой регистр регистров, т.к. содержит в себе множество учетных регистров (счетов), которые могут иметь независимую аналитику, в разрезе которой будут отражаться учетные показатели.

Создадим для хранения бухгалтерских проводок и итогов новый регистр бухгалтерии «Проводки», который включим в подсистему «БухгалтерскийУчет». Для регистра бухгалтерии на закладке «Основные» настроим связь с планом счетов: в свойстве «План счетов» выберем ссылку на созданный ранее план счетов «Бухгалтерский». Включим созданный объект в подсистему «БухгалтерскийУчет».

Установим флаг «Корреспонденция», что будет означать включение поддержки корреспонденции в регистре (это даст возможность указывать в проводке дебет и кредит и видеть обороты между счетами – табл. 5.2). Принцип двойной записи – это способ ведения бухучёта, при котором каждое изменение состояния средств организации отражается, по крайней мере, на двух счетах, обеспечивая общий баланс. По Дт одного счета и по Кт другого счета.

Таблица 5.2

Простая бухгалтерская проводка (с поддержкой корреспонденции)

Счет дебета	Счет кредита	Сумма	Содержание
Касса	Капитал	150 000	Инвестиции

В отчетах можно будет анализировать движения, остатки по счету, обороты по счету и обороты между счетами.

Если поддержку корреспонденции отключить, то проводка будет выглядеть как в табл. 5.3.

Таблица 5.3

Сборная проводка (без поддержки корреспонденции)

Счет	Вид движения	Сумма	Содержание
Касса	Дебет	150 000	Вклад наличными
Капитал	Кредит	150 000	Формирование капитала

В случае отсутствия поддержки корреспонденции в общем случае количество дебетовых и кредитовых записей набора регистра может различаться. При баланс двойной записи для балансовых счетов, измерений и ресурсов будет контролироваться платформой. В отчетах можно будет анализировать движения, остатки по счету, обороты по счету, но анализ оборотов между счетами будет недоступен.

В регистре бухгалтерии на закладке «Данные» добавим новый ресурс «Сумма», для которого обозначим признак «Балансовый». Тип Число, длина 10, точность 2. Балансовый ресурс представлен в проводке одним полем, т.е. одна и та же сумма будет проходить и по дебету и по кредиту проводки. Небалансовый ресурс создает два поля отдельно для дебета и отдельно для кредита.

Для хранения дополнительной информации о проводке служат реквизиты регистра. Добавим реквизит «Содержание» для хранения информации о содержании проводки. Тип Строка, длина 30.

4) Настройка документа-регистратора «ПоступлениеТоваров». Любое движение в регистре бухгалтерии должно иметь ссылку на документ-регистратор. Движения одного документа называются операцией.

Сделаем ранее созданный документ «ПоступлениеТоваров» регистратором для регистра бухгалтерии. У документа «ПоступлениеТоваров» на закладке «Движения» должно быть свойство «Проведение» установлено в значение «Разрешить». Это позволит документу выполнять обработку проведения. Поскольку в бухгалтерском учете контроль оперативности, как правило, не используется, то свойство «Оперативное проведение» установим в значение «Запретить». Свойство «Удаление движений» установим в стандартное значение «Удалять автоматически при отмене проведения». Другие варианты («Удалять автоматически», «Не удалять автоматически») используются когда нужно обеспечить нестандартный способ проведения документа.

Отметим флаг, что документ «ПоступлениеТоваров» является регистратором (выполняет движения) в регистре бухгалтерии «Проводки».

5) Проведение документа «ПоступлениеТоваров» по регистру бухгалтерии. Для создания алгоритма проведения документа по регистру бухгалтерии воспользуемся конструктором движений. Для этого нажмем кнопку «Конструктор движений» на закладке «Движения» документа. Чтобы использовать в проводках реквизиты табличной части в поле «Табличная часть» выберем табличную часть «Товары» документа. Сформируем с помощью конструктора следующий код:

```
Процедура ОбработкаПроведения(Отказ, Режим)
    Движения.Проводки.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        // регистр Проводки
        Движение = Движения.Проводки.Добавить();
        Движение.СчетДт = ПланыСчетов.Бухгалтерский.Товары;
        Движение.СчетКт = ПланыСчетов.Бухгалтерский.Поставщики;
        Движение.Период = Дата;
        Движение.Сумма = ТекСтрокаТовары.Сумма;
        Движение.Содержание = «Поступление товаров»;
    КонецЦикла;
КонецПроцедуры
```

В этом коде для набора записей Движени.Проводки устанавливаем флаг «Записывать» в Истина для того, чтобы указать, что данный набор записей должен быть записан в базу. Далее в цикле перебираем строки табличной части «Товары» и заполняем движения в набор записей. При записи документа этот набор будет записан в регистр бухгалтерии.

6) Создание формы документа и настройка просмотра движений документа в командном интерфейсе. Создадим для документа «ПоступлениеТоваров» форму документа. Обеспечим видимость проводок этого документа для пользователя. На закладке «Командный интерфейс» в форме документа установим видимость команды «Журнал проводок» в группе команд «Перейти».

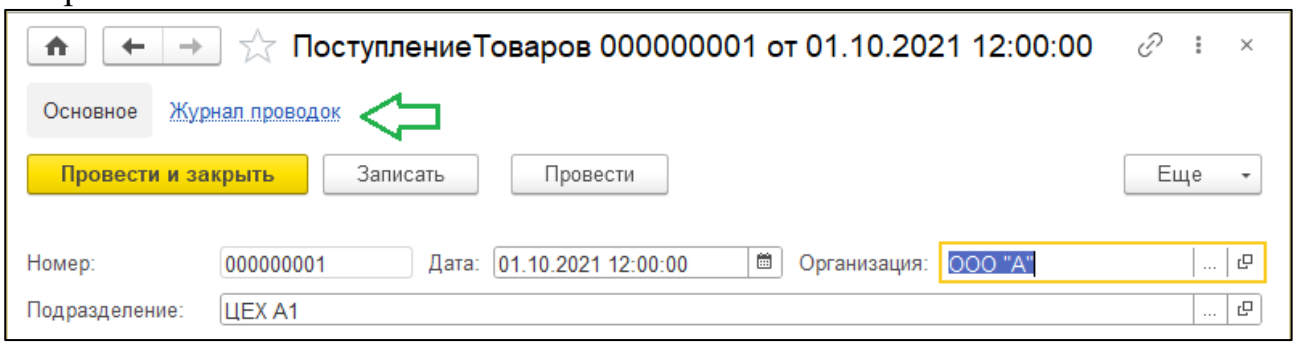


Рис. 5.4. Гиперссылка для перехода к проводкам для пользователя

В результате в пользовательском режиме у пользователя на форме на панели навигации (рис. 5.4) появится возможность открывать для просмотра проводки документа (рис. 5.5).

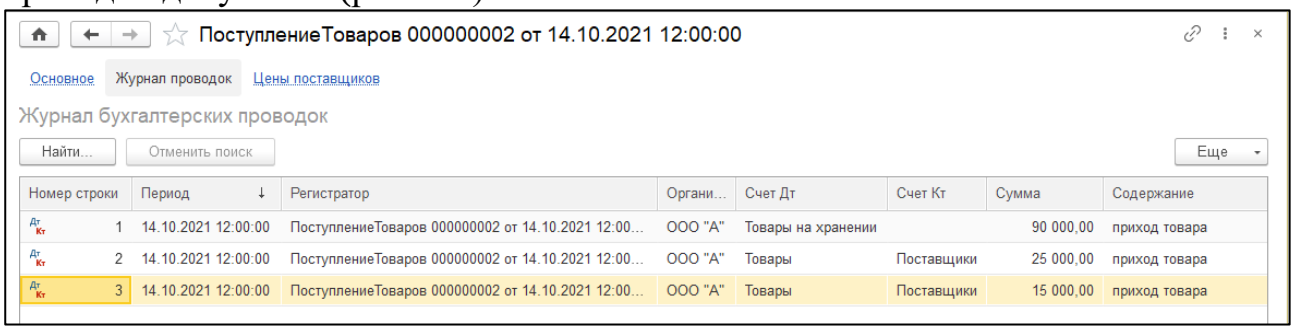


Рис. 5.5. Возможность просмотра проводок документа

7) Доработка документа: выбор счета учета в диалоге формы. В приведенном выше простейшем варианте кода процедуры обработки в качестве счета дебета был указан балансовый счет «Товары». Однако, кроме этого счета, на котором учитываются в нашем плане товары, которые принадлежат нам, есть еще счет «Товары на хранении», на котором учитываются ценности «за балансом», т.е. не принадлежащие нам.

Если предполагается оформлять поступление таких ценностей тем же документом, то встает задача обеспечить выбор счета в диалоге формы. То есть может быть ситуация, когда одним документом «ПоступлениеТоваров» могут приходоваться и собственные товары, и товары, принятые на ответственное хранение. Поэтому в табличную часть «Товары» добавим новый реквизит «СчетУчета» типа «ПланСчетовСсылка.Бухгалтерский». Новый реквизит надо вынести на созданную ранее форму документа.

Далее изменим алгоритм проведения. Откроем модуль документа с закладки «Прочее». Внесем в обработку проведения документа следующие изменения:

```
Процедура ОбработкаПроведения(Отказ, Режим)
    Движения.Проводки.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        // регистр Проводки
        Движение = Движения.Проводки.Добавить();
        Движение.СчетДт = ТекСтрокаТовары.СчетУчета;
        Движение.КоличествоДт=ТекСтрокаТовары.Количество;
        Если Не ТекСтрокаТовары.СчетУчета.Забалансовый Тогда
            Движение.СчетКт = ПланыСчетов.Бухгалтерский.Поставщики;
        КонецЕсли;
            Движение.Период = Дата;
            Движение.Организация = Организация;
            Движение.Сумма = ТекСтрокаТовары.Сумма;
            Движение.Содержание = "Поступление товара";
    КонецЦикла;
КонецПроцедуры
```

Счет дебета пользователь будет выбирать в диалоге формы. При проведении проверяется является ли выбранный пользователем счет балансовым. Если нет, то счет кредита не заполняется, поскольку забалансовый счет не может корреспондировать с балансовым. Правило двойной записи не контролируется для забалансовых счетов.

8) Запрет выбора счетов верхнего уровня при выборе счета. В нашем плане счетов есть счета, которые имеют субсчета (например, счет «Обязательства») и сами непосредственно не должны участвовать в проводках. Следовательно, надо запретить выбор таких счетов верхнего уровня в поле ввода «Счет учета» документа «ПоступлениеТоваров».

Для реализации этой задачи добавим в план счетов новый реквизит «ЗапретитьИспользоватьВПроводках», тип Булево. В режиме исполнения надо заполнить этот реквизит значением «Истина» для счетов, имеющих субсчета.

Далее реализуем с помощью параметров выбора возможность для пользователя видеть в форме документа «ПоступлениеТоваров» только счета, для которых реквизит «ЗапретитьИспользоватьВПроводках» имеет значение Ложь. Для этого в свойствах реквизита документа «СчетУчета» установим в свойстве «ПараметрыВыбора» следующие настройки (рис. 5.6).

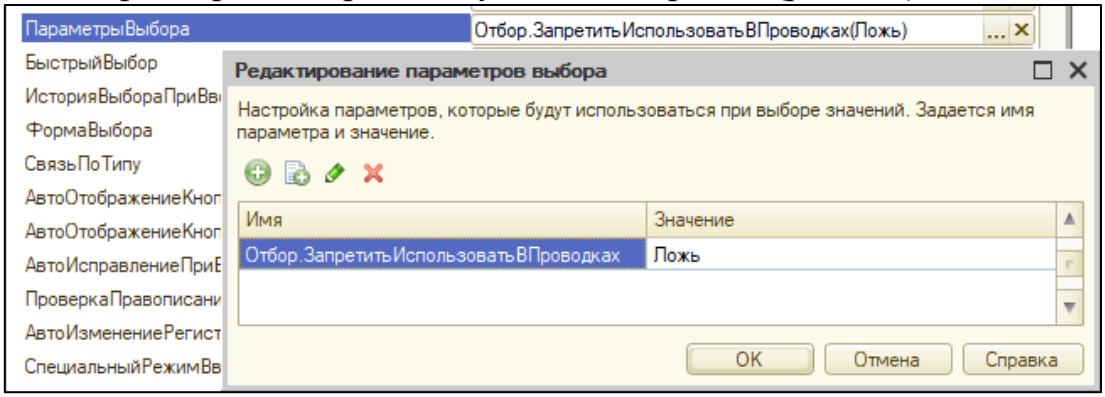


Рис. 5.6. Настройка параметров выбора для реквизита «СчетУчета»

9) Условное оформление в настройке списка плана счетов

Зададим для счетов верхнего уровня, которые недоступны для выбора в документах, условное оформление в форме списка плана счетов, чтобы они отличались от остальных. Создадим форму списка для плана счетов «Бухгалтерский». Откроем окно свойств динамического списка, зададим условное оформление (рис. 5.7). Для счетов, у которых реквизит ЗапретитьИспользоватьВПроводках = Истина, установим серый цвет фона для выделения строк.

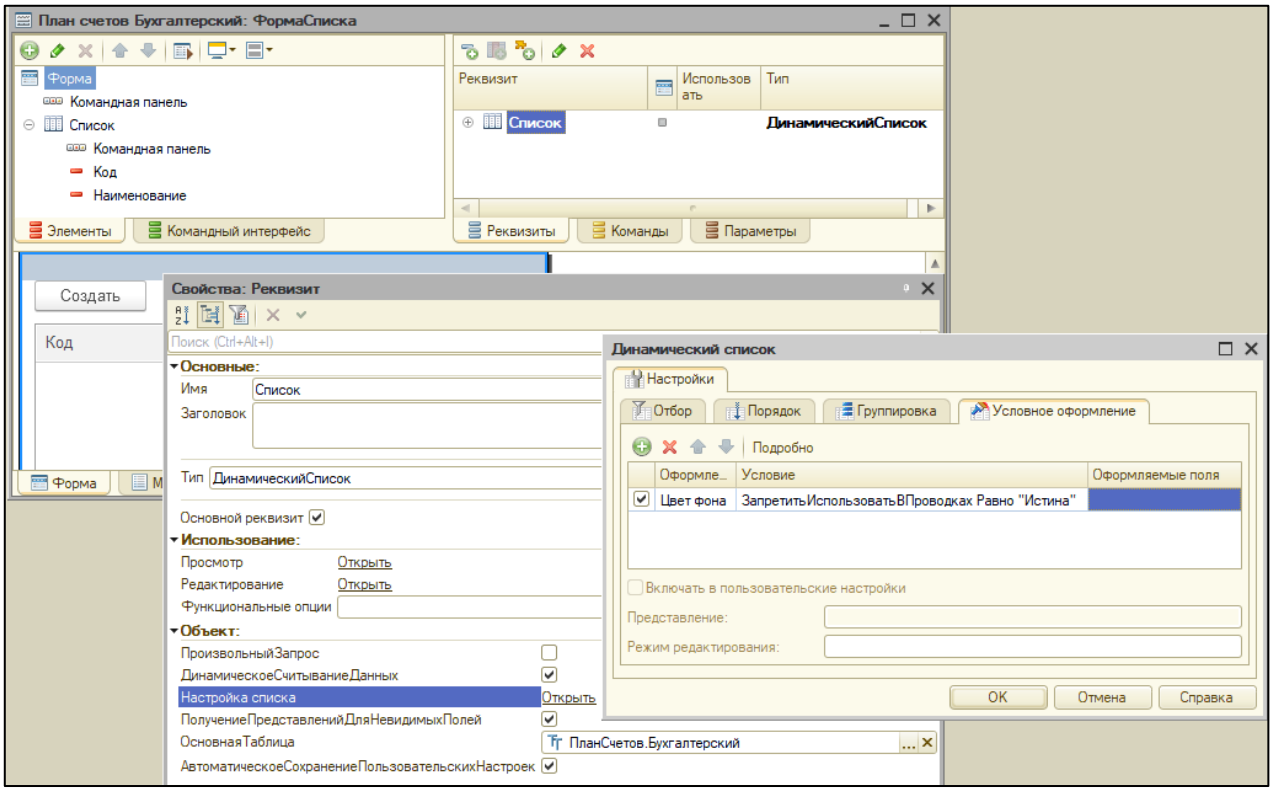


Рис. 5.7. Условное оформление

В режиме исполнения счета верхнего уровня, которые не должны использоваться в проводках, будут выделены серым цветом (рис. 5.8).

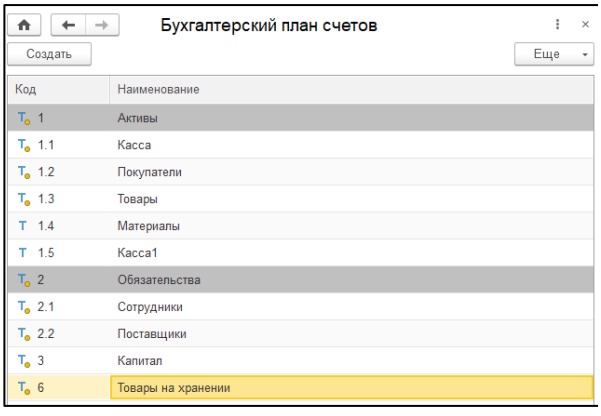


Рис. 5.8. Условное оформление списка счетов

10) Проведение документа «ПродажаТоваров» по регистру бухгалтерии. Сделаем ранее созданный документ «ПродажаТоваров» регистратором для регистра бухгалтерии. У документа «ПродажаТоваров» на закладке «Движения» должно быть свойство «Проведение» установлено в значение «Разрешить». Это позволит документу выполнять обработку проведения. Поскольку в бухгалтерском учете контроль

оперативности, как правило, не используется, то свойство «Оперативное проведение» установим в значение «Запретить». Свойство «Удаление движений» установим в стандартное значение «Удалять автоматически при отмене проведения». Отметим, что документ «ПродажаТоваров» является регистратором (выполняет движения) в регистре бухгалтерии «Проводки».

В модуле объекта необходимо сформировать процедуру проведения документа. Для каждой строки табличной части «Товары» формируются следующие проводки:

Дт «Капитал» (если в строке выбран балансовый счет учета «Товары») Кт «Товары» на «Стоимость» продаваемого товара;

Дт «Покупатель» Кт «Капитал» на сумму товара из реквизита «Сумма».

Для каждой строки табличной части «Услуга» формируется проводка:

Дт «Покупатели» Кт «Капитал» на сумму услуги из реквизита «Сумма».

Ниже приведен примерный код процедуры проведения:

```
Процедура ОбработкаПроведения(Отказ, Режим)
    Движения.Проводки.Записывать = Истина;
    // формируем проводки по таблично части Товары
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        // регистр Проводки
        // первая проводка
        Движение = Движения.Проводки.Добавить();
        Движение.СчетКт = ТекСтрокаТовары.СчетУчета;
        Если Не ТекСтрокаТовары.СчетУчета.Забалансовый Тогда
            Движение.СчетДт = ПланыСчетов.Бухгалтерский.Капитал;
        КонецЕсли;
            Движение.Период = Дата;
            Движение.Организация = Организация;
            Движение.Сумма = ТекСтрокаТовары.Стоимость;
            Движение.Содержание = "Реализация товаров";
        // вторая проводка
        Движение = Движения.Проводки.Добавить();
        Движение.СчетДт = ПланыСчетов.Бухгалтерский.Покупатели;
        Движение.СчетКт = ПланыСчетов.Бухгалтерский.Капитал;
        Движение.Период = Дата;
        Движение.Организация = Организация;
        Движение.Сумма = ТекСтрокаТовары.Сумма;
        Движение.Содержание = "Выручка за товары";
    КонецЦикла;

    // формируем проводки по таблично части Услуги
    Для Каждого ТекСтрокаУслуги Из Услуги Цикл
        // регистр Проводки
        // проводка по услугам
```

```
            Движение = Движения.Проводки.Добавить();
            Движение.СчетДт = ПланыСчетов.Бухгалтерский.Покупатели;
            Движение.СчетКт = ПланыСчетов.Бухгалтерский.Капитал;
            Движение.Период = Дата;
            Движение.Организация = Организация;
            Движение.Сумма = ТекСтрокаТовары.Сумма;
            Движение.Содержание = "Выручка за услуги";
    КонецЕсли;
КонецПроцедуры
```

11) Создание формы документа и настройка просмотра движений документа в командном интерфейсе. Создадим для документа «ПродажаТоваров» форму документа. Обеспечим видимость проводок этого документа для пользователя. На закладке «Командный интерфейс» в форме документа установим видимость команды «Журнал проводок» в группе команд «Перейти» (рис. 5.9). В результате в пользовательском режиме у пользователя на форме на панели навигации появится гиперссылка, дающая возможность пользователю открывать для просмотра проводки документа.

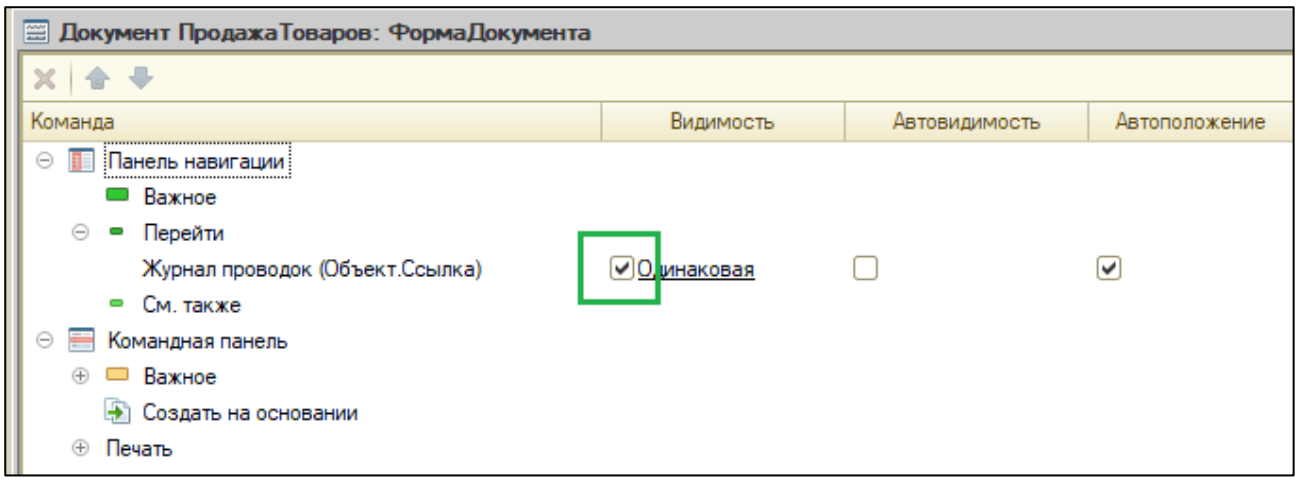


Рис. 5.9. Видимость команды «Журнал проводок» в группе команд «Перейти»

Таким образом, в рамках практических занятий по теме №5 закреплены на практике теоретические знания по механизмам платформы «1С:Предприятие» для реализации синтетического учета, составляющего основу бухгалтерского учета. Получены навыки правильного выбора и проектирования объектов платформы «1С:Предприятие» для решение задач бухгалтерского учета (план счетов и регистр бухгалтерии), конфигурирования, настройки их свойств, реализации движений с помощью документов-регистраторов в регистре бухгалтерии, использования конструктора движений, настройки интерфейса (видимости команд, условного оформления, параметров выбора) для более удобного отражения информации для пользователя.

Тема 6. Разработка сложных отчетов с помощью системы компоновки данных

Разработку сложных отчетов с помощью системы компоновки данных (СКД) рассмотрим на примере нескольких сводных отчетов, анализирующих данные синтетического учета.

Счета, имеющие субсчета, имеют некоторые особенности при работе с ними в отчетах [3].

Если в отчете нас интересуют остатки и обороты по счету, имеющему субсчета, например, счету «Активы» как по счету-группе, который объединяет субсчета активов нашего предприятия, можно воспользоваться следующей конструкцией языка запросов:

Счет В ИЕРАРХИИ(&Счет)

Если бы нас интересовали остатки и обороты по счету «Активы», основанные только на проводках, где он указан как один из корреспондирующих, то можно было бы воспользоваться следующим предложением языка запросов:

Счет = & Счет

1) Создание отчета «ШахматнаяВедомость». Этот отчет должен предоставить пользователю информацию о сводных оборотах между счетами в виде кросс-отчета: строки – дебетуемые в корреспонденциях счета, колонки – кредитуемые. На пересечении отражается оборот в дебет счета с кредита счета. Общий итог отчета отражается в нижнем правом углу.

Создадим в конфигурации новый отчет «ШахматнаяВедомость». Отнесем его к подсистеме «БухгалтерскийУчет». Создадим для отчета основную схему компоновки данных. В качестве набора данных будет выступать запрос следующего вида:

```
ВЫБРАТЬ
	ПроводкиОборотыДтКт.СчетДт,
	ПроводкиОборотыДтКт.СчетКт,
	ПроводкиОборотыДтКт.СуммаОборот,
	ПроводкиОборотыДтКт.Организация
ИЗ
	РегистрБухгалтерии.Проводки.ОборотыДтКт(
			,
			,
			,
			(НЕ СчетДт.Забалансовый)
				ИЛИ &ВыводитьЗабалансовыеСчета,
			,
			(НЕ СчетКт.Забалансовый)
				ИЛИ &ВыводитьЗабалансовыеСчета,
			,
			) КАК ПроводкиОборотыДтКт
```

Как видно из текста запроса, в качестве источника выбрана виртуальная таблица «ОборотыДтКт», в параметрах которой указан отбор по полям «СчетДт» и «СчетКт» для исключения из отчета забалансовых счетов. В секции ВЫБРАТЬ указаны поля «СчетДт», «СчетКт» и «СуммаОборот».

В списке доступных полей отчета, полученных в результате автозаполнения, отредактируем заголовок ресурса «СуммаОборот» – установим равным просто «Сумма».

На закладке «Ресурсы» укажем в качестве ресурса поле «СуммаОборот». На закладке Параметры укажем и опишем работу параметра Период (стандартный период) как на рис. 6.1.

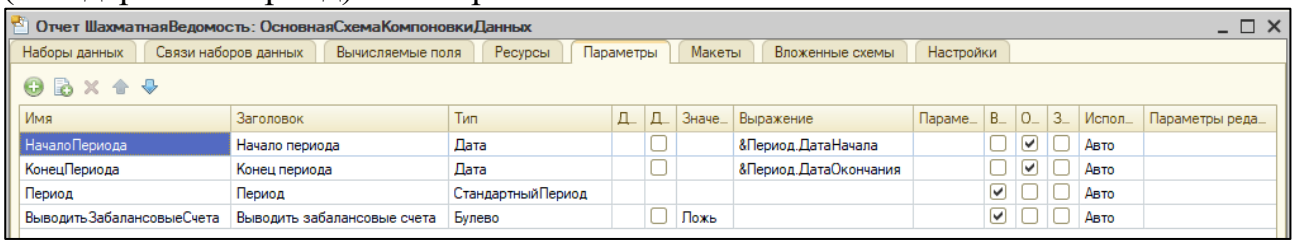


Рис. 6.1. Параметры отчета «ШахматнаяВедомость»

На закладке «Настройки» создадим новую таблицу (рис. 6.2), строками которой будут счета дт (с иерархией), а колонками счета кт (с иерархией).

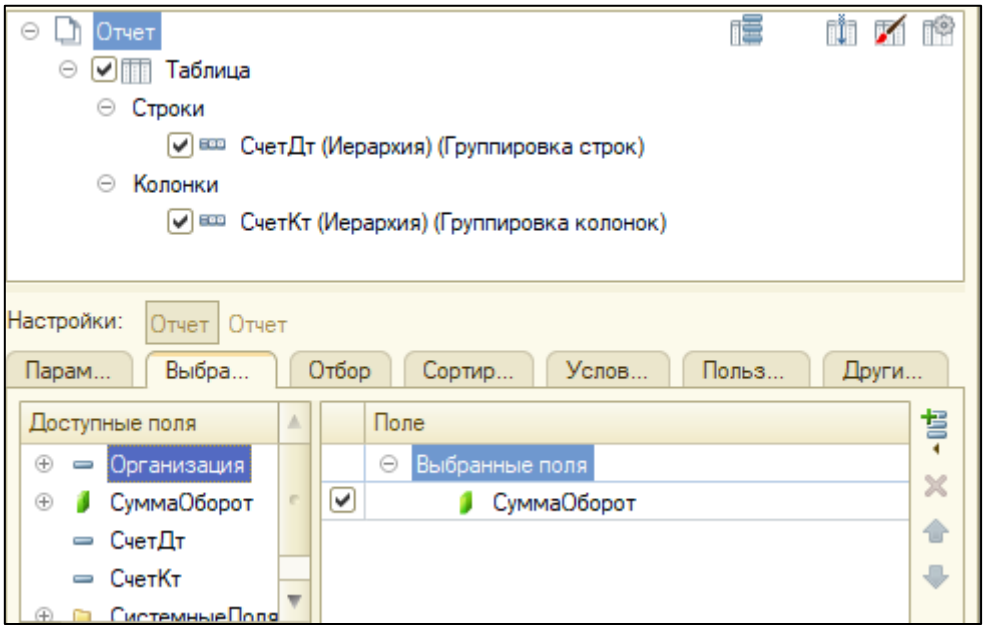


Рис. 6.2. Группировки отчета «ШахматнаяВедомость»

Как видно из рис. 6.2, для обеих группировок установлено имя. Для этого щелкнув правой кнопкой мыши по соответствующей группировке, выбираем пункт контекстного меню «Установить имя». Для группировки по счету дт даем имя «Группировка строк», для второй группировки «Группировка колонок».

Для более красивого внешнего вила отчета предлагается создать макет отчета. Переходим на закладку «Макеты». Создаем там области, как показано на рис. 6.3. Привязываем их к группировкам отчета. Например, для ячейки СчетКт в свойствах ячейки указываем Заполнение=Параметр; имя параметра «СчетКт», которое совпадает с именем поля запроса. В списке слева добавляем «Макет

группировки», имя группировки «Группировка колонок», тип макета «Залоговок». Нажав на кнопку с троеточием в колонке «Область», открываем окно для выбора диапазона ячеек – указываем R2C1, для чего в табличном документе выделяем ячейку, в которой записан параметр «СчетКт» и повторно нажимаем на кнопку с троеточием в окне выбора диапазона. Аналогичным образом создаем и остальные области макета. Отчет готов, можно запустить режим исполнения и проверить его работоспособность.

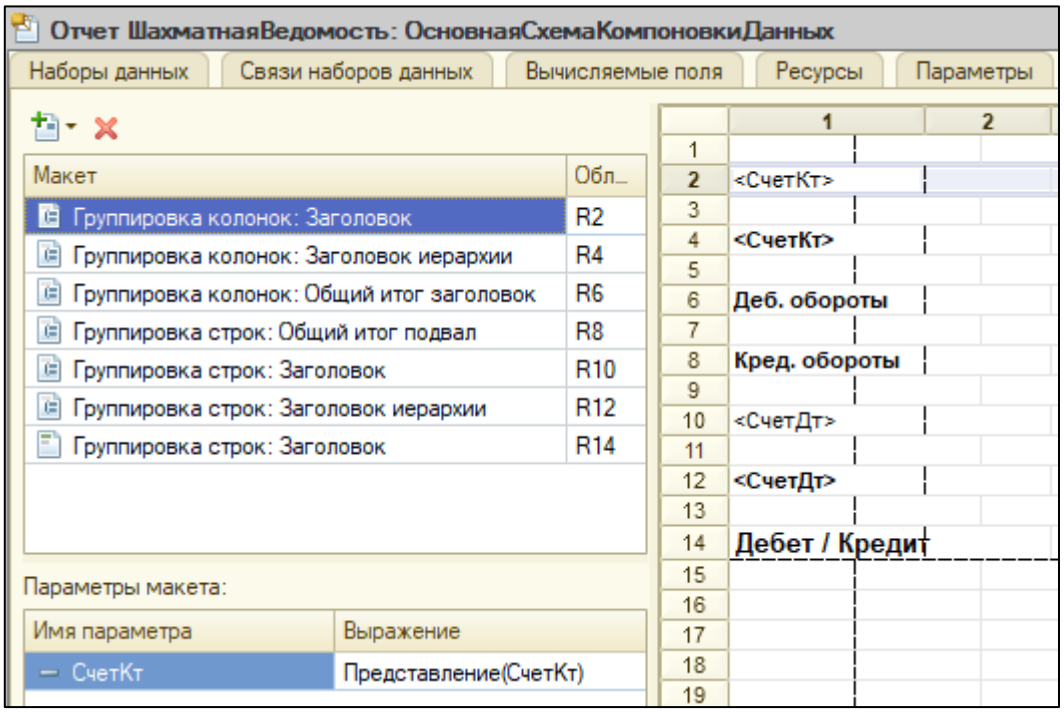


Рис 6.3. Макет для отчета «ШахматнаяВедомость»

На закладке «Настройки» дадим имя разработанному варианту отчета «Шахматка».

2) Вариант «СводныеПроводки» отчета «ШахматнаяВедомость». Добавим для отчета еще один вариант настроек – более компактный вид «СводныеПроводки». На закладке Настройки в списке вариантов отчетов в левом верхнем углу добавим еще один вариант. Назовем его «СводныеПроводки» (рис. 6.4).

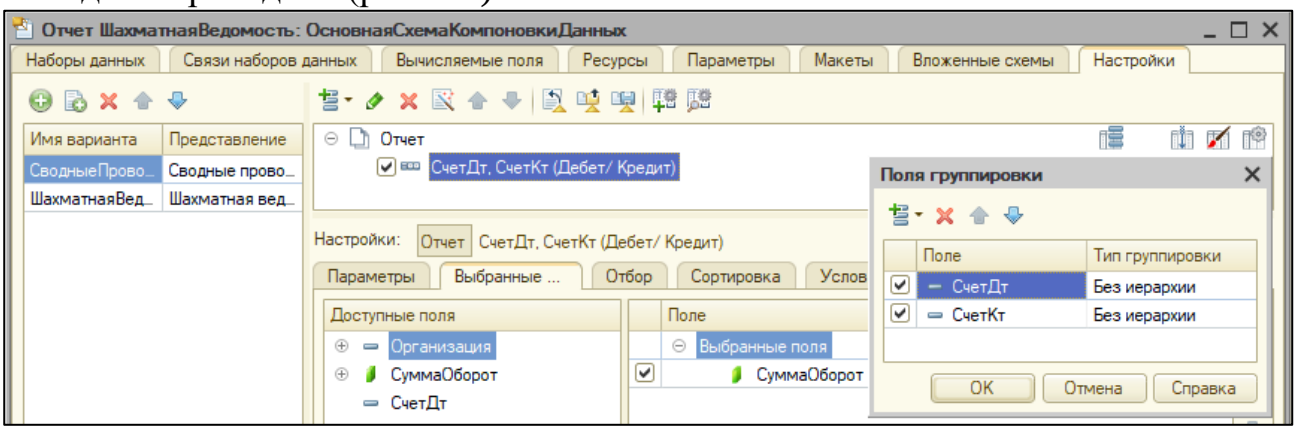


Рис. 6.4. Настройка варианта отчета «СводныеПроводки»

Настроим этот вариант отчета. На закладке «Настройки» добавим группировку «Счет дт, Счет кт». Чтобы ее создать сначала добавляем группировку «Счет дт», далее нажимаем «Изменить» и в открывшемся окне добавляем к выбранному полю группировки СчетДт еще и второе поле СчетКт.

На закладке «Параметры» варианта настроек для параметров сделаем следующие настройки по умолчанию: «ВыводитьЗабалансовыеСчета» = Ложь (флаг Использовать отмечен); «Период» = этот год (флаг Использовать отмечен).

В качестве выбранных полей выберем одно поле «СуммаОборот».

На закладке условное оформление можно сделать следующие настройки: Добавим новый элемент «Оформление – Выделять отрицательные = Истина». Зададим представление как «Отрицательное красным».

На закладке «Другие настройки» зададим макет оформления «Зеленый», в качестве заголовка отчета укажем «Сводные проводки».

Пользовательский интерфейс на закладке «Параметры»: по кнопке  отметим видимость пользователю обоих параметров в режиме быстрого доступа. Для условного оформления «Отрицательное красным» выполним такие же настройки видимости, только укажем режим редактирования «Обычный».

Второй вариант отчета готов. Можно запустить режим исполнения и проверить работоспособность варианта отчета «СводныеПроводки».

3) Создание отчета «ОборотноСальдоваяВедомость». Это отчет, который показывает пользователю в виде списка остатки на начало периода, обороты по дебету и по кредиту и остатки на конец периода по каждому счету, причем в иерархии и с общими итогами.

Создадим новый отчет «ОборотноСальдоваяВедомость». Отнесем его в подсистеме «БухгалтерскийУчет».

Набор данных для отчета сформирует запрос следующего вида:

```
ВЫБРАТЬ
    ПроводкиОстаткиИОбороты.Счет,
    ПроводкиОстаткиИОбороты.Организация,
    ПроводкиОстаткиИОбороты.СуммаНачальныйОстатокДт,
    ПроводкиОстаткиИОбороты.СуммаНачальныйОстатокКт,
    ПроводкиОстаткиИОбороты.СуммаОборотДт,
    ПроводкиОстаткиИОбороты.СуммаОборотКт,
    ПроводкиОстаткиИОбороты.СуммаКонечныйОстатокДт,
    ПроводкиОстаткиИОбороты.СуммаКонечныйОстатокКт

ИЗ
    РегистрБухгалтерии.Проводки.ОстаткиИОбороты( , , , ,
            НЕ Счет.Забалансовый
                ИЛИ &ВыводитьЗабалансовыеСчета,
        ,
    ) КАК ПроводкиОстаткиИОбороты
```

Как видно из текста запроса, в качестве источника данных выбрана виртуальная таблица регистра бухгалтерии «Остатки и обороты». Оставив флаг «Автозаполнение», получаем доступные поля. Немного отредактируем их.

Поле Счет: укажем выражение представления как Счет.Код. Это позволит пользователю в любом варианте отчета счет видеть, как строку с кодом, а не наименование, которое является основным представлением согласно нашим настройкам.

Для полей «СуммаНачальныйОстатокДт», «СуммаНачальныйОстатокКт», «СуммаКонечныйОстатокДт», «СуммаКонечныйОстатокКт» укажем в колонке Роль значение «Без роли». Такая настройка позволит рассчитать общие итоги без проверки вида счета, в противном случае не удастся получить общие итоги по остаткам на начало и конец периода.

Сгруппируем пол в папки, чтобы пользователю с ними было удобно работать. Используя кнопку  , создадим в списке доступных полей три папки «СальдоНаНачало», «ОборотыЗаПериод», «СальдоНаКонец». Далее изменим пути к полям ресурсов в колонке «Путь», привязывая их к созданным папкам:

СальдоНаНачало.СуммаНачальныйОстатокДт;

СальдоНаНачало.СуммаНачальныйОстатокКт;

ОборотыЗаПериод.СуммаОборотДт;

ОборотыЗаПериод.СуммаОборотКт;

СальдоНаКонец.СуммаКонечныйОстатокКт;

СальдоНаКонец.СуммаКонечныйОстатокДт.

Чтобы отчет выглядел опрятно, для всех полей ресурсов установим в оформлении минимальную и максимальную ширину колонки равную значению 12.

На закладке «Настройки» создадим вариант настройки «Основной». Добавим группировку по полю «Счет» (с иерархией).

На закладке «Параметры» варианта настроек для параметров сделаем следующие настройки по умолчанию: «ВыводитьЗабалансовыеСчета» = Ложь (флаг Использовать отмечен); «Период» = этот год (флаг Использовать отмечен). По кнопке  отметим видимость пользователю обоих параметров в режиме быстрого доступа.

На закладке условное оформление можно сделать следующие настройки: Добавим новый элемент «Оформление – Выделять отрицательные = Истина». Зададим представление как «Отрицательное красным». Дополнительно раскрасим счета верхнего уровня жирным шрифтом: добавим еще одно оформление для шрифта с условием «ЗапретитьИспользоватьВПроводках» = Истина и представлением «Выделять счета первого уровня». Оба оформления сделаем по кнопке  доступными пользователю в режиме редактирования «Обычный»

На закладке «Другие настройки» зададим макет оформления «Зеленый», в качестве заголовка отчета укажем «Оборотно-сальдовая ведомость».

В качестве выбранных полей выберем все три наши папки и наименование счета. Внутри каждой папки для полей назначим имена. Для этого щелкнем правой кнопкой мыши по полю и выберем из контекстного меню пункт «Установить заголовок». Назначим заголовки «Дебет» и «Кредит».

На закладке «Другие настройки» включим опцию «Расположение реквизитов = отдельно». Отчет готов. Можно запустить систему в режиме исполнения и проверить работоспособность созданного отчета (рис. 6.5).

Организация	СальдоНаНачало		ОборотыЗаПериод		СальдоНаКонец	
Подразделение	Дебет	Кредит	Дебит	Кредит	Дебет	Кредит
ООО "А"			348 200,00	15 514,55	332 685,45	
ЦЕХ А1			348 200,00	15 514,55	332 685,45	
Итого			348 200,00	15 514,55	332 685,45	

Рис. 6.5. Отчет «Оборотно-сальдовая ведомость»

4) Создание отчета «Анализ счета». Данный отчет должен показывать пользователю по выбранному счету остатки на начало периода, обороты с коррсчетами, обороты ща период и остатки на конец периода.

Этот отчет имеет особенности:

– для его формирования придется использовать более чем один источник;

– его реализация будет осуществлена с помощью двух наборов данных.

Создадим новый отчет «АнализСчета». Включим его в подсистему «БухгалтерскийУчет». В отчете создадим два набора данных «ОстаткиОбороты» и «КорОбороты».

Первый набор данных – запрос имеет следующий вид:

```
ВЫБРАТЬ
    ПроводкиОстаткиИОбороты.СуммаНачальныйОстатокДт,
    ПроводкиОстаткиИОбороты.СуммаНачальныйОстатокКт,
    ПроводкиОстаткиИОбороты.СуммаОборотДт КАК ВсегоСуммаОборотДт,
    ПроводкиОстаткиИОбороты.СуммаОборотКт КАК ВсегоСуммаОборотКт,
    ПроводкиОстаткиИОбороты.СуммаКонечныйОстатокДт,
    ПроводкиОстаткиИОбороты.СуммаКонечныйОстатокКт,
```

ПроводкиОстаткиИОбороты.Организация
ИЗ
РегистрБухгалтерии.Проводки.ОстаткиИОбороты(, , , , Счет В ИЕРАРХИИ (&Счет), ,) КАК ПроводкиОстаткиИОбороты

В качестве источника данных выступает виртуальная таблица «ОстаткиИОбороты» с отбором по счету в параметре виртуальной таблицы. Этим запросом мы извлекаем остатки на начало и на конце периода, обороты за период. Полям оборотов назначим псевдонимы «ВсегоСуммаОборотДт» и «ВсегоСуммаОборотКт».

Оставим поле Автозаполнение включенным по умолчанию и получим список доступных полей. В списке доступных полей для поля Счет установим флаг недоступности в колонке «ОграниченияПоля».

Второй набор данных «КорОбороты» – – запрос имеет следующий вид:

ВЫБРАТЬ
ПроводкиОбороты.КорСчет,
ПроводкиОбороты.СуммаОборотДт,
ПроводкиОбороты.СуммаОборотКт,
ПроводкиОбороты.Организация
ИЗ
РегистрБухгалтерии.Проводки.Обороты(, , , Счет В ИЕРАРХИИ (&Счет), , , ,) КАК ПроводкиОбороты

В качестве источника данных выступает виртуальная таблица Остатки с отбором по счету в параметре виртуальной таблицы. Этим запросом мы извлекаем обороты за период с корреспондирующими счетами.

Оставим поле Автозаполнение включенным по умолчанию и получим список доступных полей. В списке доступных полей также для поля Счет установим флаг недоступности в колонке «ОграниченияПоля».

Закладка «Ресурсы»: поля «СуммаНачальныйОстатокДт», «СУммаНачальныйОСтатокКт», «СуммаОборотДт», «СУммаОборотКт», «ВсегоСуммаОборотДт», «ВсегоСуммаОборотКт», «СуммаКонечныйОстатокДт», «СуммаКонечныйОстатокКт» добавляем в число ресурсов.

На закладке «Параметры», как и в прошлых отчетах, укажем и опишем работу параметра Период (стандартный период), а еще там добавится поле Счет, доступное для пользователя.

Настройки отчета на закладке «Настройки»: добавим там две группировки (рис. 6.6). Верхняя группировка «Детальные записи» (не указываем группированное поле) будет формировать строки с остатками на начало,

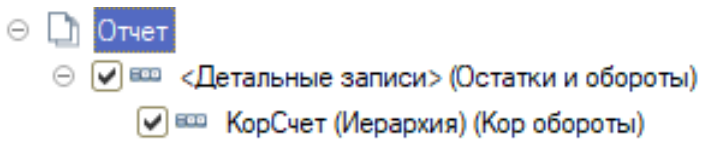


Рис. 6.6. Группировки отчета «Анализ счета»

остатками на конец периода и оборотам по счету в целом. Зададим этой верхней группировки имя «Остатки и обороты».

Вложенная группировка будет формировать отборы с корреспондирующими счетами. Ее назовем «Кор обороты» и в качестве группированного поля укажем поле «КорСчет».

Для отчета в целом указываем настройку параметров – доступность пользователю в быстром доступе. Поскольку для данного отчеты мы сформируем далее макет, то из списка выбранных полей удаляем все поля, включая автополя.

На закладке «Другие настройки» укажем макет оформления «Зеленый» и заголовок отчета «Анализ счета».

Далее настроим каждую группировку.

Нажмем кнопку <Детальные записи> (Остатки и обороты) на панели настроек и выполним настройку группировки «Остатки и обороты». На закладке «Другие настройки» укажем: расположение группировок = начало и конец.

Макет отчета представлен на рис. 6.7. Все области, представленные в нем, являются макетами группировки, кроме области «Итоги: Заголовок – это макет заголовка группировки.

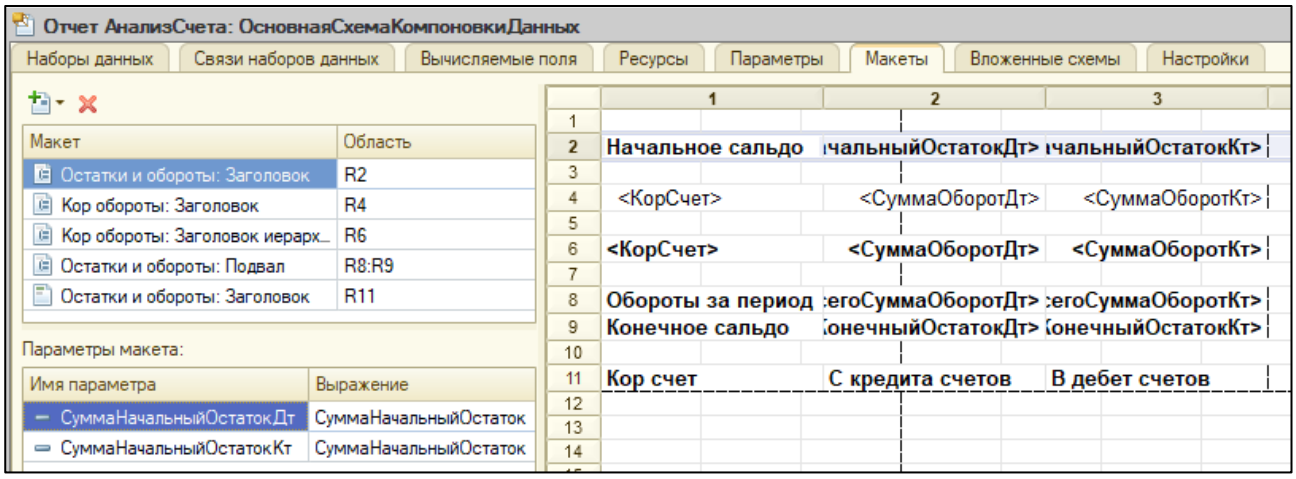


Рис. 6.7. Макет отчета «АнлизСчета»

В режиме исполнения можно проверить работоспособность разработанного отчета.

5) Создание отчета «ОборотыСчетаАвто». Назначением этого отчета является получение дебетового и кредитового оборота счета со всеми корсчетами с заданной периодичностью (по дням, месяцам, декадам и т.п.).

В этом отчете мы воспользуемся дополнительным параметром «Периодичность», который присутствует в виртуальных таблицах, содержащих обороты. С его помощью можно получать обороты с дополнительной группировкой по стандартным периодам. В случае, если в параметре «Периодичность» не задано значение или задано значение «Период», дополнительной группировки не производится. Возмодными значениями дополнительной группировки могут быть значения: Запись, Регистратор,

Секунда, Минута, Час, День, Неделя, Декада, Месяц, Квартал, Полугодие, Год, Авто. Значение Авто указывается в случае, ели надо рассчитать сразу несколько группировок по разным стандартным периодам или разрешить пользователю самому выбирать в диалоге формы отчета периодичность.

Создадим отчет «ОборотыСчетаАвто». Добавим его в подсистему «БухгалтерскийУчет». В качестве источника запроса укажем запрос. Текст запроса следующий:

```
ВЫБРАТЬ
    ПроводкиОбороты.КорСчет,
    ПроводкиОбороты.СуммаОборотДт,
    ПроводкиОбороты.СуммаОборотКт,
    ПроводкиОбороты.ПериодДень,
    ПроводкиОбороты.ПериодНеделя,
    ПроводкиОбороты.ПериодДекада,
    ПроводкиОбороты.ПериодМесяц,
    ПроводкиОбороты.ПериодКвартал,
    ПроводкиОбороты.ПериодПолугодие,
    ПроводкиОбороты.ПериодГод,
    ПроводкиОбороты.Организация,
    ПроводкиОбороты.Организация КАК Организация1
ИЗ
    РегистрБухгалтерии.Проводки.Обороты(, , Авто, Счет В
                ИЕРАРХИИ (&Счет), , , , ) КАК ПроводкиОбороты
```

Источником данных в запросе выступает виртуальная таблица «Обороты». В ее параметрах мы указали периодичность «Авто» и отбор по счету, который передается в параметре &Счет. В выбранные поля включаем такие поля как КорСчет, СуммаОборотДт, СуммаОборотКт и ряд интересующих нас периодов.

В список доступных полей, полученных благодаря Автозаполнению, внесем некоторые изменения настроек. Для поля «Счет» установим ограничение, т.к. он будет выбираться в параметрах.

Для периодов укажем форматы, представленные в табл. 6.1. Как видно из таблицы для разных полей применены ращзные подходы: для каких то полей мы обошлись указанием формата, для других полей использованы сложные выражения представления.

Таблица 6.1

Форматирование для периодов в отчете «АнализСчетаАвто»

Поле	Выражение представления	Оформление: формат
1	2	3
ПериодДень	–	ДФ=dd.MM.yy
ПериодМесяц	–	ДФ='MMMM yy'
ПериодКвартал	–	ДФ='к "кв." yyyy'
ПериодГод	–	ДФ=yyyy

Продолжение табл. 6.1.

1	2	3
ПериодПолугодие	ВЫБОР КОГДА Квартал(ПериодКвартал)<2 ТОГДА Формат(ПериодКвартал, "ДФ""1 полугодие" "уу") ИНАЧЕ Формат(ПериодКвартал, "ДФ"" 2полугодие""уу"") КОНЕЦ	–
ПериодДекада	Формат(НачалоПериода(ПериодДекада,"Декада"), "ДФ=dd.ММ.уу")+"- "+Формат(КонецПериода(ПериодДекада,"Декада"), "ДФ=dd.ММ.уу")	–
ПериодНеделя	Формат(НачалоПериода(ПериодНеделя,"Неделя"), "ДФ=dd.ММ.уу")+"- "+Формат(КонецПериода(ПериодНеделя,"Неделя"), "ДФ=dd.ММ.уу")	–

На закладке «Ресурсы» выберем в качестве ресурсом поля «СуммаОборотДт» и «СуммаОборотКт». На закладке «Парамерты» выполним настройки аналогичные прошлому отчету: укажем и опишем работу параметра Период (стандартный период), а также там присутствует поле Счет, доступное для пользователя.

На закладке «Настройки» добавим таблицу с группировками строк и колонок: в строках добавим группировку по полю «ПериодМесяц», в колонках – группировку по полю КорСчет (Иерархия). Установим курсор на корень дерева «Отчет» и на верхней панели по кнопке  «Свойства элемента пользовательских настроек» настроим диалог отчета следующим образом: установим флаги у «Выбранные поля» (режим редактирования = Быстрый доступ) и «Отбор» (режим редактирования = Быстрый доступ). Переставим курсор на элемент настройки «Таблица» в дереве отчета и вызовем такое же окно управления. В нем укажем флаг «Группировки строк» (режим редактирования = Быстрый доступ).

В список выбранных полей включим поля «ОборотыДт» и «ОборотыКт». При этом флаг установим только для «ОборотыКт». На закладке «Другие настройки» зададим макет «Зеленый» и заголовок «Обороты счета авто».

Отчет готов. Запускаем режим исполнения и проверяем работоспособность отчета. По умолчанию отчет формирует кредитовые обороты счета. Чтобы получить дебетовые обороты можно в выбранных полях изменить ресурс запроса на «ОборотыДт». Вариант группировки выбирается по кнопке «Настройка».

Тема 7. Организация аналитического учета в «1С:Предприятие» с помощью субконто

Под аналитическим учетом понимается более детальный, чем синтетический учет [3].

Обычная аналитика с использованием субконто подразумевает, что на разных счетах может присутствовать различная аналитика в разном количестве параллельных срезов. На каких-то счетах может не быть аналитика, на других один, два или более параллельных среза.

Для создания такого механизма нам необходимо продумать ряд аспектов:

– хранить где то список возможных аналитических разрезов, причем каждый разрез (вид субконто, аналитический счет) должен помнить тип своих значений;

– описать максимально возможное количество срезов для каждого регистра бухгалтерии (плана счетов);

– указать, на каких счетах, какие разрезы будут использоваться и в какой последовательности.

– доработать документы и отчеты, чтобы документы могли заполнять новые поля таблиц регистра бухгалтерии, а отчеты – получать итоги с отбором и группировками по новой аналитике.

1) Создание плана видов характеристик «ВидыСубконто» и справочника «Субконто». Хранение списка всевозможных аналитических срезов (видов субконто) реализуем с помощью плана видов характеристик.

Создадим новый план видов характеристик «ВидыСубконто». Зададим для него представление объекта «Вид субконто», представление списка объектов «Виды субконто». Включим созданный план видов характеристик в подсистему «БухгалтерскийУчет».

В свойстве «Тип значения характеристик» укажем составной тип и отметим следующие типы: СправочникСсылка.Склады; СправочникСсылка.Номенклатура; СправочникСсылка.Контрагенты; СправочникСсылка. ФизическиеЛица. Следует отметить, что при задании типов значений характеристик для субконто не рекомендуется задавать примитивные типы данных, т.е. это негативно сказывается на эффективности индексирования таблиц регистра бухгалтерии.

Кроме того, пользователю необходимо разрешить создавать собственные виды субконто и где-то хранить значения субконто для этих видов. Воспользуемся для реализации этого механизмом дополнительных характеристик. Создадим новый **справочник «Субконто»**. Отнесём его в подсистеме «БухгалтерскийУчет». Подчиним его плану видов характеристик «ВидыСубконто» (на закладке «Владельцы»).

Созданный справочник «Субконто» включим в составной тип данных для плана видов характеристик. Кроме того, ссылку на справочник выберем в свойстве «Дополнительные значения характеристик».

В результате пользователь сможет создавать собственные виды субконто в пользовательском режиме.

Далее создадим предопределённые виды субконто, которые будут использоваться на предопределенных и пользовательских счетах. На закладке «Прочие» плана видов характеристик введем следующие предопределенные виды субконто (рис. 7.1).

План видов характеристик ВидыСубконто: Предопределенные виды характ...

Действия

Имя	Код	Наименование	Тип
Характеристики			
Договоры	000000006	Договоры	СправочникСсылка.Договоры
Контрагенты	000000001	Контрагенты	СправочникСсылка.Контрагенты
Номенклатура	000000002	Номенклатура	СправочникСсылка.Номенклатура
Склады	000000003	Склады	СправочникСсылка.Склады
Сотрудники	000000004	Сотрудники	СправочникСсылка.ФизическиеЛица
СтатьиДДС	000000005	Статьи ДДС	СправочникСсылка.СтатьиДДС

Рис. 7.1. Предопределенные виды характеристик

2) Настройка плана счетов для аналитического учета. В свойствах плана счетов «Бухгалтерский» на закладке «Субконто» выберем в свойстве «Виды субконто» ссылку на созданный ранее план видов характеристик «ВидыСубконто».

На этой же закладке следует указать максимально возможное количество субконто. Укажем значение равное 2. Тем самым мы задаем количество дополнительных измерений регистра бухгалтерии, которые платформа «1С:Предприятие» создаст.

Далее надо подключить виды субконто к предопределенным счетам (рис. 7.2). Следует отметить, что порядок субконто важен. Специалисты советуют располагать субконто в порядке убывания количества значений в них. Кроме того, целесообразно по возможности указывать один и тот же вид субконто на одно и то же место на разных счетах. В этом случае аналитическая отчетность по всем счетам, имеющим одинаковую аналитику, будет формироваться быстрее.

План счетов Бухгалтерский: Предопределенные счета

Действия

Имя	Код	Наименование	Вид	Заба...	Поряд...	Субконто 1	Субконто 2
Счета							
Активы	1	Активы	Активный		1		
Касса	1.1	Касса	Активный		1. 1	СтатьиДДС	
Покупатели	1.2	Покупатели	Активный		1. 2	Контрагенты	Договоры
Товары	1.3	Товары	Активный		1. 3	Номенклатура	
Обязательства	2	Обязательства	Пассивный		2		
Сотрудники	2.1	Сотрудники	Пассивный		2. 1	Сотрудники	
Поставщики	2.2	Поставщики	Пассивный		2. 2	Контрагенты	Договоры
Капитал	3	Капитал	Пассивный		3		
ТоварыНаХранении	6	Товары на хранении	Активный	✔	6	Номенклатура	Контрагенты

Рис. 7.2. Предопределенные счета с субконто

3) Создание общих модулей и экспортных процедур и функций для реализации универсального механизма установки субконто на счетах

Заполнение аналитики по предопределённым счетам можно прописать прямо в процедуре проведения. Однако, когда в документах счет выбирается пользователем в диалоге формы, аналитика и даже колчиество субконто зависит от выбранного счета. Например, счет «Товары» имеет одно субконто «Номенклатура». А если пользователь выберет забалансовый счет «ТоварыНаХранении», то он имеет два вида субконто – «Номенклатура» и «Контрагенты». Следовательно, если прописывать установление субконто прямо в процедуре обработки проведения, будет необходимо делать дополнительные проверки.

Лучше эти проверки вынести в общий модуль, т.к. подобные действия нужно будет выполнять и в других документах. Сделаем этот механизм заполнения аналитики универсальным – создадим общий модуль «БухгалтерияСервер» в ветке Общие/Общие модули. В его свойствах должен быть отмечен флаг «Сервер». В общем модуле реализует следующую экспортную процедуру:

```
Процедура УстановитьСубконто(Счет, Субконто, ИмяСубконто,
ЗначениеСубконто) Экспорт
    Если НЕ ЗначениеЗаполнено(Счет) Тогда
        Возврат;
    КонецЕсли;
    ВидыСубконтоСчета = Счет.ВидыСубконто;
    Если ТипЗнч(ИмяСубконто) = Тип("Число") Тогда
        Если ИмяСубконто > ВидыСубконтоСчета.Количество() Тогда
          Возврат;
        КонецЕсли;
        ВидСубконтоССылка =
                ВидыСубконтоСчета[ИмяСубконто - 1].ВидСубконто;

    Иначе
        ВидСубконтоССылка =
                ПланыВидовХарактеристик.ВидыСубконто[ИмяСубконто];
      Если ВидыСубконтоСчета.Найти(ВидСубконтоССылка)=Неопределено
                                                        Тогда
        Возврат;
      КонецЕсли;
    КонецЕсли;
    Если
ВидСубконтоССылка.ТипЗначения.СодержитТип(ТипЗнч(ЗначениеСубконто))
                                                        Тогда
        Субконто.Вставить(ВидСубконтоССылка, ЗначениеСубконто);
    КонецЕсли;
    КонецПроцедуры
```

В качестве параметров в эту экспортную процедуру передаются следующие параметры: счет проводки, субконто (соответствие) проводки, имя для предопределенного субконто или номер вида субконто на счете и значение субконто. В зависимости от этих параметров процедура пытается установить значения субконто, а противном случае осуществляется выход из процедуры.

Для получения данных счета (количество субконто, виды субконто) целесообразно разработать еще одну экспортную функцию, которую поместить в общий модуль с повторным использованием возвращаемых значений, т.к. эта информация меняется крайне редко и будет востребована и в других механизмах конфигурации.

Создадим еще один общий модуль «БухгалтерияСерверПовторно», у которого укажем следующие свойства: отметим флаги «Сервер» и «Вызов сервера»; свойство «Повторное использование возвращаемых значений» установим в значение «На время сеанса».

В общем модуле «БухгалтерияСерверПовторно» реализуем экспортную функцию ПолучитьСвойстваСчета():

```
Функция ПолучитьСвойстваСчета(Счет) Экспорт
    ДанныеСчета = Новый Структура;
    ДанныеСчета.Вставить("Забалансовый", Счет.Забалансовый);
    ДанныеСчета.Вставить("КоличествоСубконто",
                                        Счет.ВидыСубконто.Количнство());
    МаксКоличествоСубконто =
       Метаданные.ПланыСчетов.Бухгалтерский.МаксКоличествоСубконто;
    Для Индекс=1 По МаксКоличествоСубконто Цикл
       Если Индекс <=Счет.ВидыСубконто.Количество() Тогда
          ДанныеСчета.Вставить("ВидыСубконто"+Индекс,
                        Счет.ВидыСубконто[Индекс-1].ВидСубконто);
          ДанныеСчета.Вставить("ВидыСубконто"+Индекс+"Наименование",
          Строка(Счет.ВидыСубконто[Индекс-1].ВидыСубконто));
          Счет.ВидыСубконто[Индекс-1].ВидСубконто.ТипЗначения);
       Иначе
          ДанныеСчета.Вставить("ВидСубконто"+Индекс, Неопределено);
          ДанныеСчета.Вставить("ВидСубконто"+Индекс+"Наименование",
                                                    Неопределено);
          ДанныеСчета.Вставить("ВидСубконто"+Индекс+"ТипЗначения",
                                                    Неопределено);
       КонецЕсли;
    КонецЦикла;
    Возврат ДанныеСчета;
КонецФункции
```

Эта функция создает структуру ДанныеСчета и наполняет ее основными свойствами счета.

4) Доработка документов для заполнения аналитических признаков проводки. Изменим алгоритмы проведения созданных ранее документов «ПоступлениеТоваров» и «ПродажаТоваров».

Реализуем вызов процедуры «УстановитьСубконто» общего модуля из процедуры обработки проведения документа «ПосутплениеТоваров». Жирным шрифтом выделены строки, которые добавлены для реализации установки субконто на счетах.

```
Процедура ОбработкаПроведения(Отказ, Режим)
    Движения.Проводки.Записывать = Истина;
    Для Каждого ТекСтрокаТовары Из Товары Цикл
        // регистр Проводки
        Движение = Движения.Проводки.Добавить();
        Движение.СчетДт = ТекСтрокаТовары.СчетУчета;
        БухгалтерияСервер.УстановитьСубконто(Движение.СчетДт,
  Движение.СубконтоДт, "Номенклатура", ТекСтрокаТовары.Номенклатура);
    БухгалтерияСервер.УстановитьСубконто(Движение.СчетДт,
                        Движение.СубконтоДт, "Контрагенты", Контрагент);
        БухгалтерияСервер.УстановитьСубконто(Движение.СчетДт,
                                Движение.СубконтоДт, "Склады", Склад);
        Если Не ТекСтрокаТовары.СчетУчета.Забалансовый Тогда
            Движение.СчетКт =
                                        ПланыСчетов.Бухгалтерский.Поставщики;
      БухгалтерияСервер.УстановитьСубконто(Движение.СчетКт,
                        Движение.СубконтоКт, "Контрагенты", Контрагент);
                БухгалтерияСервер.УстановитьСубконто(Движение.СчетКт,
                                Движение.СубконтоКт, "Договоры", Договор);
        КонецЕсли;
        Движение.Период = Дата;
        Движение.Организация = Организация;
        Движение.Сумма = ТекСтрокаТовары.Сумма;
        Движение.Содержание = "Поступление товара";
    КонецЦикла;
КонецПроцедуры
```

Благодаря универсальности процедуры «УстановитьСубконто» общего модуля в процедуре проведения нет необходимости делать проверки наличия субконто на счете. В модуле перечислены возможные варианты для дебета и кредита. Функция сама оставит только те, которые нужны.

Аналогично реализуйте самостоятельно изменение процедуры проведения документа «ПродажаТоваров» для заполнения аналитики.

5) Реализация универсальных документов. В учете встречаются ситуации, когда на некоторых участках учета (например, «Касса», «Банк) в документе известен только один из корреспондирующих счетов. А второй счет выбирается пользователем в диалоге формы. И у него может быть разная

аналитика, которую тоже надо выбирать в диалоге формы. В качестве примера такого универсального документа разработаем документ приходный кассовый ордер (ПКО). Документ формирует проводку в дебет счета «Касса» с кредита счета, который пользователь укажет в диалоге формы. Аналитика выбранного счета также будет выбираться пользователем на форме.

Создадим новый документ «ПКО». Добавим его в подсистему «БухгалтесркийУчет». На закладке «Данные» зададим у документа следующие реквизиты:

«Счет» – ПланСчетоСсылка.Бухгалтерский. параметры выбора Отбор.ЗапретитьИспользоватьВПроводках(Ложь) для отбора счетов только нижнего уровня;

«Субконто1» – Характеристика.ВидыСубконто; необходимо установить связь по типу со счетом (с указанием элемента связи по типу – номера субконто на счете (рис. 7.3)).

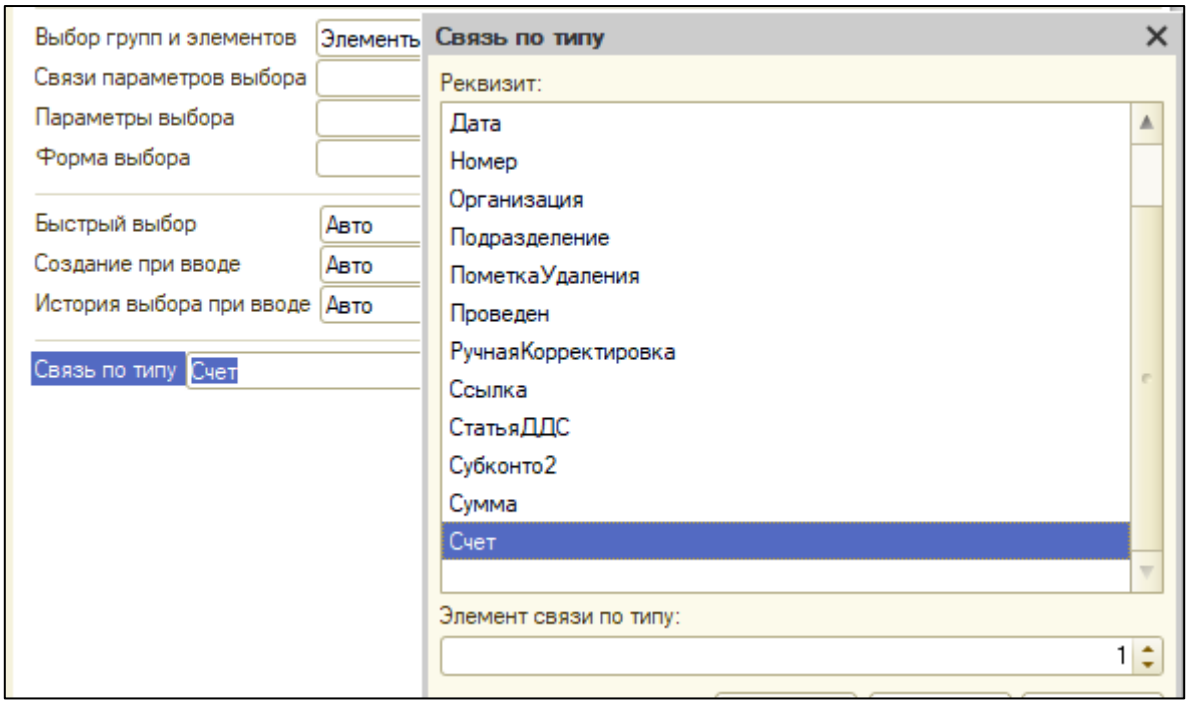


Рис. 7.3. Задание для реквизита Субконто1 связи по типу с реквизитом Счет

«Субконто2» – тип «Характеристика.ВидыСубконто»; необходимо установить связь по типу со счетом (с указанием элемента связи по типу – номера субконто на счете равным 2);

«Сумма» – тип Число, длина 15, точность 2;

«Организация» – тип «СправочникСсылка.Организации».

Для всех реквизитов следует указать свойство «Проверка заполнения» равным «Выдавать ошибку».

На закладке «Движения» укажем для документа возможность неоперативного проведения, а также отметим, что документ является регистратором для регистра бухгалтерии.

6) Реализация функциональности формы документа «ПКО». Создадим явно форму этого документа (рис. 7.4). В ней определим доступность и заголовки реквизитов «Субконто1» и «Субконто2» в соответствии с видами субконто выбранного счета и их количеством.

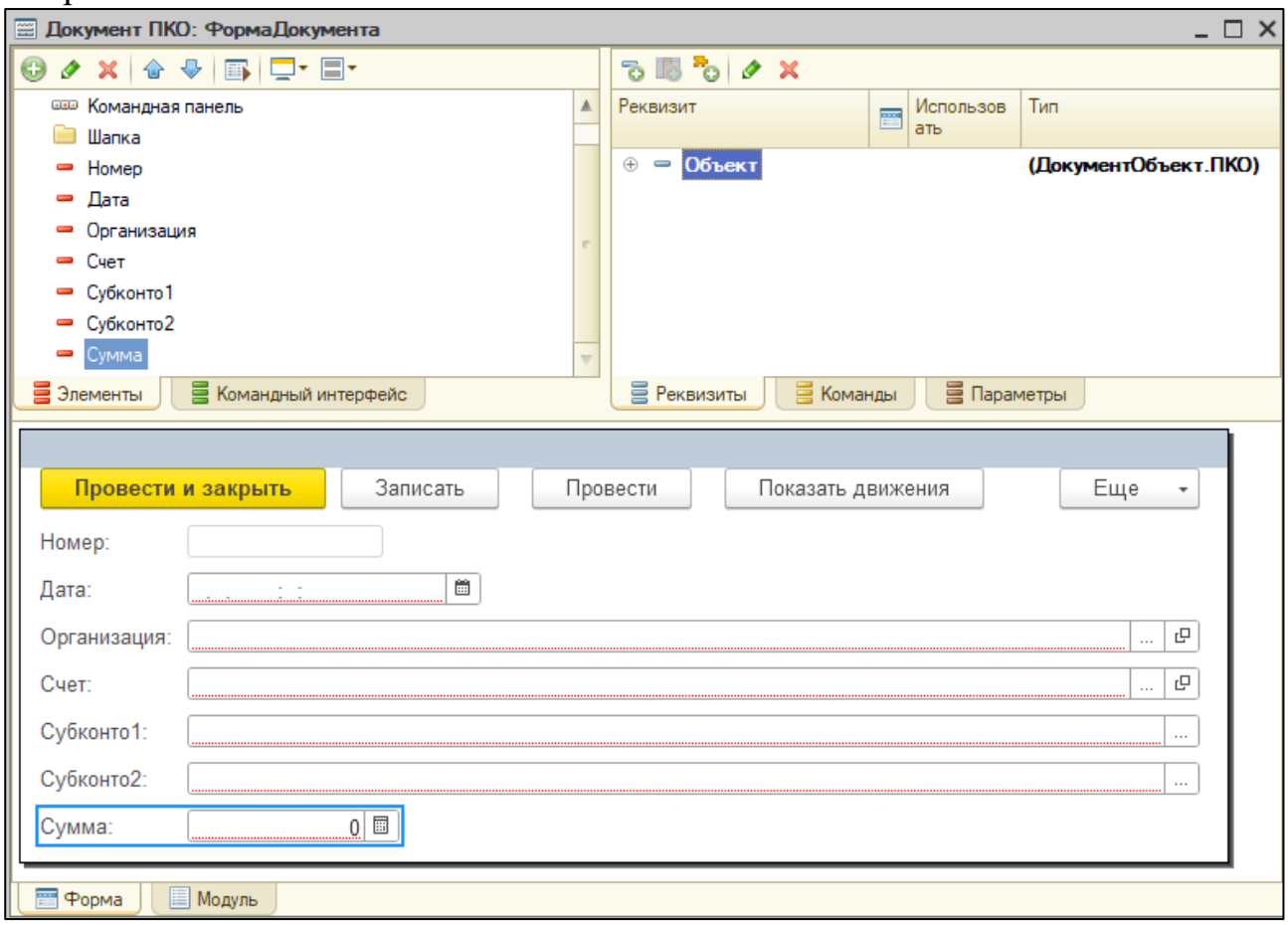


Рис. 7.4. Форма документа «ПКО»

В модуле формы реализует следующие обработчики:

```
&НаКлиенте
Процедура УстановитьЗаголовкиИДоступностьСубконто()
ДанныеСчета =
            БухгалтерияСерверПовторно.ПолучитьСвойстваСчета(Объект.Счет);
Для Индекс=1 По 2 Цикл
    Если Индекс<=ДанныеСчета.КоличествоСубконто Тогда
        Элементы["Субконто"+Индекс].Заголовок =
            ДанныеСчета["ВидСубконто"+Индекс+"Наименование"];
        Элементы["Субконто"+Индекс].Доступность=Истина;
    Иначе
        Элементы["Субконто"+Индекс].Заголовок = "Не задан";
        Элементы["Субконто"+Индекс].Доступность = Ложь;
    КонецЕсли;
КонецЦикла;
КонецПроцедуры
```

```
&НаКлиенте
Процедура СчетПриИзменении(Элемент)
    УстановитьЗаголовкиИДоступностьСубконто();
КонецПроцедуры

&НаКлиенте
Процедура ПриОткрытии(Отказ)
    УстановитьЗаголовкиИДоступностьСубконто();
КонецПроцедуры
```

7) Проведение документа «ПКО». В модуле объекта документа «ПКО» реализует следующую процедуру обработки проведения.

```
Процедура ОбработкаПроведения(Отказ, Режим)
    Движения.Проводки.Записывать = Истина;
    // регистр Проводки
    Движение = Движения.Проводки.Добавить();
    Движение.СчетДт = ПланыСчетов.Бухгалтерский.Касса;
    Движение.Период = Дата;
    Движение.СчетКт = Счет;
    Движение.Организация = Организация;
    Движение.Содержание = "Поступление денег";
     БухгалтерияСервер.УстановитьСубконто(Движение.СчетКт,
                                          Движение.СубконтоКт, 1, Субконто1);
     БухгалтерияСервер.УстановитьСубконто(Движение.СчетКт,
                                          Движение.СубконтоКт, 2, СУбконто2);
КонецПроцедуры
```

8) Реализация простого аналитического отчета по одному участку учета «ДвижениеТоваров». Создадим отчет «ДвижениеТоваров», формирующий данные о движении товаров, анализирующий остатки и обороты за период с группировкой по аналитике, итоги должны быть сгруппированы по аналитике. Упрощенно условимся, что ответ будет выполняться с отбором по одному счету «Товары». Запрос для отчета будет следующим:

```
ВЫБРАТЬ
    ПроводкиОстаткиИОбороты.Субконто1 КАК Номенклатура,
    ПроводкиОстаткиИОбороты.Организация,
    ПроводкиОстаткиИОбороты.СуммаНачальныйОстаток,
    ПроводкиОстаткиИОбороты.СуммаОборотДт,
    ПроводкиОстаткиИОбороты.СуммаОборотКт,
    ПроводкиОстаткиИОбороты.СуммаКонечныйОстаток,
ИЗ
    РегистрБухгалтерии.Проводки.ОстаткиИОбороты(, , , , Счет =
                ЗНАЧЕНИЕ(ПланСчетов.Бухгалтерский.Товары), , ) КАК
                                        ПроводкиОстаткиИОбороты
```

Сделаем следующие настройки в доступных полях отчета:

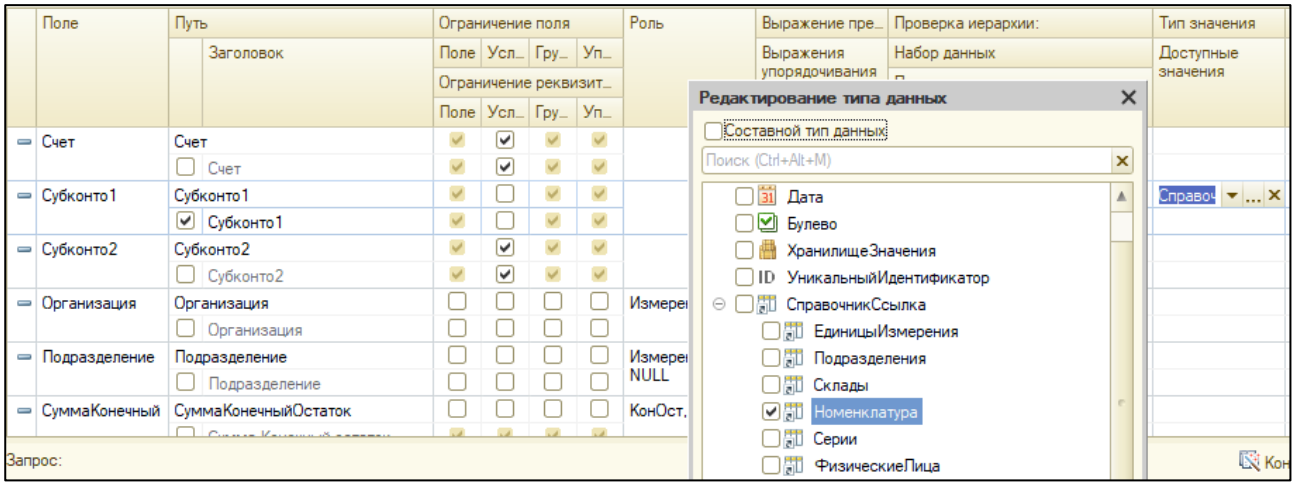


Рис. 7.5. Настройки в доступных полях отчета

Для поля Субконто1, которое изначально имеет составной тип, назначим тип значения «СправочникСсылка.Номенклатура». Поля «Счет» и «Субконто2» скроем (Счет жестко задан в параметрах и Субконто2 на счете «Товары» отсутствует).

На закладке «Ресурсы» в качестве ресурсов укажем поля «СуммаНачальныйОстаток», «СуммаОборотДт», «СуммаОборотКТ», «СуммаКонечныйОстаток».

На закладке «Параметры» добавим и опишем работу параметра Период (стандартный период) (рис. 7.6). Это даст возможность пользователю в режиме исполнения использовать стандартный период при выборе периода формирования отчета.

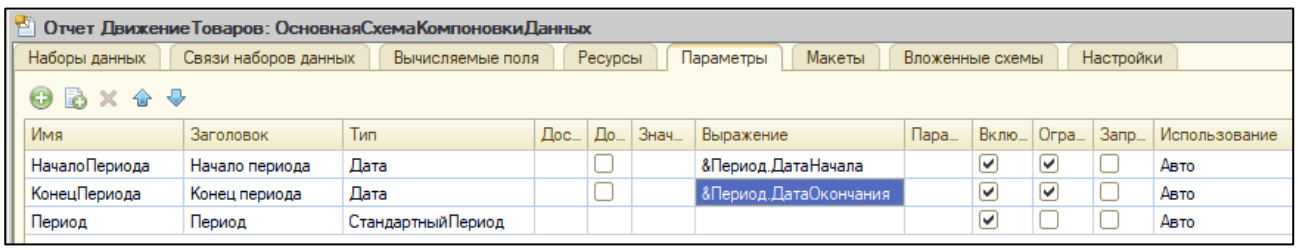
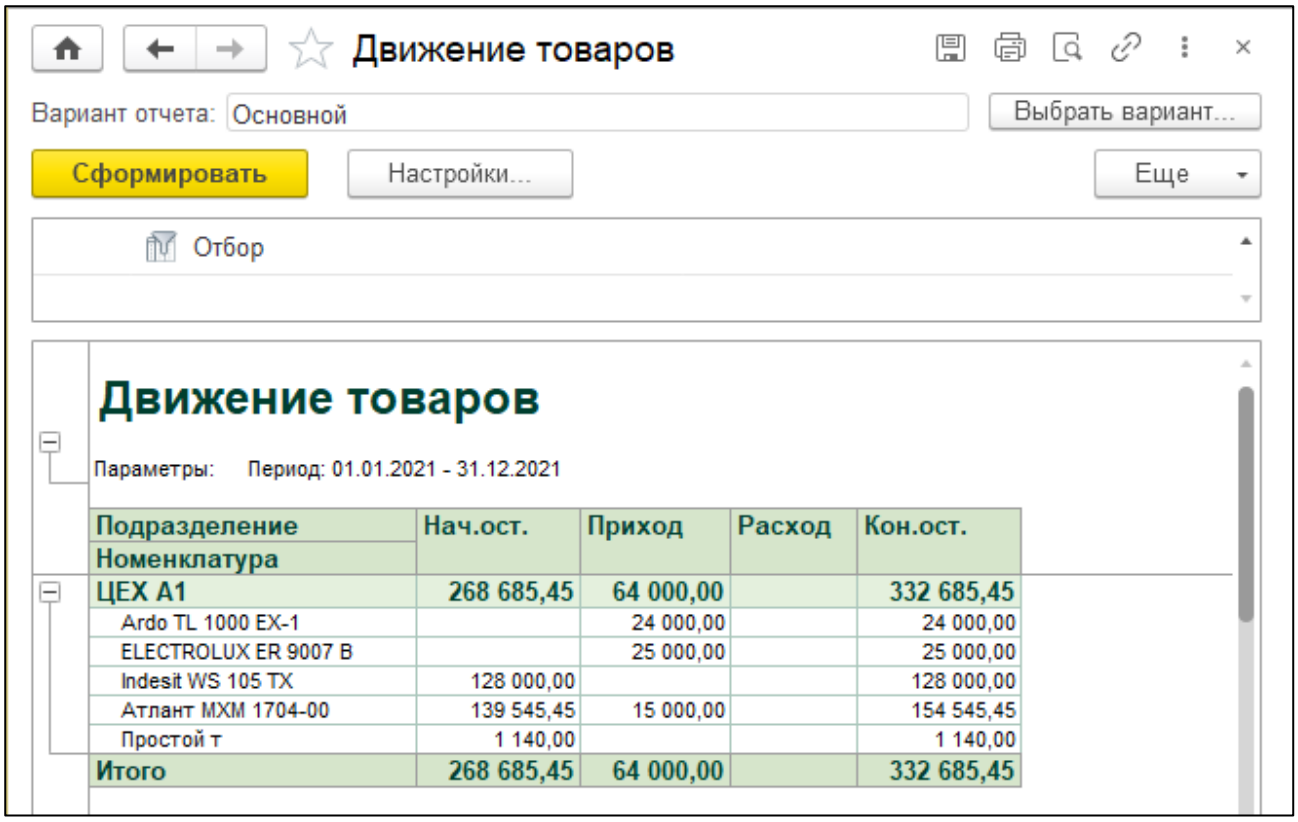


Рис. 7.6. Настройка параметра Период

На закладке «Настройки» выполним настройки внешнего вида отчета. Добавим группировку по полю Субконто1 и выберем поля «СуммаНачальныйОстаток», «СуммаОборотДт», «СуммаОборотКТ», «СуммаКонечныйОстаток». Отчет готов.

Запустим режим исполнения и сформируем отчет по движению товаров, например, за период с 01.01.2021 по 31.12.2021, т.е. за стандартный период 2021 год (рис. 7.7).



Подразделение / Номенклатура	Нач.ост.	Приход	Расход	Кон.ост.
ЦЕХ А1	268 685,45	64 000,00		332 685,45
Ardo TL 1000 EX-1		24 000,00		24 000,00
ELECTROLUX ER 9007 B		25 000,00		25 000,00
Indesit WS 105 TX	128 000,00			128 000,00
Атлант МХМ 1704-00	139 545,45	15 000,00		154 545,45
Простой т	1 140,00			1 140,00
Итого	268 685,45	64 000,00		332 685,45

Рис. 7.7. Отчет «ДвижениеТоваров» в режиме исполнения

Итак, ведение аналитического бухгалтерского учета в системе 1С:Предприятие обеспечивается тремя объектами конфигурации [3]: План счетов, План видов характеристик и Регистр бухгалтерии (рис. 7.8).

Рис. 7.8. Создание плана счетов и настройка его свойств

В рамках практических занятий по теме №7 закреплены на практике теоретические знания по механизмам платформы «1С:Предприятие» для реализации аналитического учета, детализирующего данные синтетического бухгалтерского учета. Получены навыки правильного выбора и проектирования объектов платформы «1С:Предприятие» для решение задач аналитического учета (план видов характеристик и справочник для реализации дополнительных видов субконто), конфигурирования, настройки их свойств, настройки плана счетов для аналитического учеа, реализации движений с аналитикой с помощью документов-регистраторов в регистре бухгалтерии, настройки интерфейса (видимости и заголовков субконто в зависимости от выбираемого пользователем на форме счета) для более удобного отражения информации для пользователя.

Тема 8. Создание объектов конфигурации для решения расчетных задач: планы видов расчета, регистры расчета

Расчетные задачи связаны с периодическими расчетами, т.е. с вычислениями, которые производятся с определенной периодичностью, тесно связанные друг с другом по некоторым правилам и взаимно влияющие друг на друга в пределах некоторого периода. Механизмы периодических расчетов позволяют настроить порядок и взаимосвязь вычислений и организовать учет их результатов [3].

Наиболее типичным примером использования периодических расчетов является расчет заработной платы, при котором производятся расчеты начислений и удержаний. Расчеты обычно выполняются с месячной периодичностью, а результаты расчетов одного вида могут зависеть от наличия и результатов расчетов другого вида.

Ведение периодических расчетов в системе «1С:Предприятие» обеспечивают объекты конфигурации **Планы видов расчета** и **Регистры расчета** [3].

Как и справочник, план видов расчета является ссылочным объектом, имеет стандартные реквизиты Код и Наименование, может иметь произвольное число создаваемых разработчиком дополнительных прикладных реквизитов и табличных частей. В плане видов расчетов могут быть созданы предопределенные виды расчета. В отличие от справочника в плане видов расчета не поддерживается автонумерации, автоматической генерации значения поля Код для нового вида расчета, не контролируется уникальность кода платформой. Кроме того, не поддерживается иерархическая организация видов расчета.

Типичным примером периодических расчетов является расчет заработной платы. Реализуем в нашей конфигурации расчет заработной платы. Наша конфигурация будет рассчитывать некоторый набор видов расчета, каждый из которых имеет определенную расчетную формулу. Приведем примеры расчетных формул:

Вид расчета «Оклад». Расчетная формула: **[Результат] = [Ставка оклада] * [Число отработанных дней] / [Число рабочих дней в месяце]**. Назовем способ расчета, использующий такую расчетную формулу, «По окладу».

Вид расчета «Оклад на выезде». Расчетная формула: **[Результат] = [Ставка оклада] * [Число отработанных дней] / [Число рабочих дней в месяце]**. Способ расчета тоже «По окладу».

Вид расчета «Доплата за квалификацию». Расчетная формула: [**Результат] = [Процент доплаты] * [СуммаБазы]**. Способ расчета, использующий такую формулу, назовем «Процентом».

Эти формулы для получения правильного результата для вида расчета, который рассчитывается по базе, требуют знать состав базовых его базовых видов расчета, а также необходимо придерживаться строгой очередности расчета. Ведь результаты базовых видов расчета должны быть получены раньше,

чем начнется расчет того вида расчета, для которого они являются базовыми, иначе сумма базы не может быть получена.

Также следует учитывать тот факт, что для получения правильного результата для вида расчета, который зависит от отработанного времени, нужно знать состав его вытесняющих видов расчета, чтобы, в случае их наличия за расчетный период, определить интервалы фактического периода действия и просуммировать по ним фактически отработанное время.

Если эти алгоритмы разрабатывать в универсальных программных средах, т.е. не с использованием предметно-ориентированной на решение учетных задач платформе «1С:Предприятие», то задачи получения суммы базовых начислений и суммы рабочего времени по фактическому периоду действия будут весьма трудоемкими. В приложениях на платформе «1С:Предприятие» эффективное решение этих задач возлагается на два основных объекта «План видов расчета» и «Регистр расчета», в функциональность которых заложены механизмы, обеспечивающие решение задач сложных периодических расчетов. Разработчики платформы встроили реализацию этих двух алгоритмов в платформу.

1) Создание подсистемы «РасчетЗарплаты». Добавим в конфигурацию новую подсистему «РасчетЗарплаты», к которой будем относить все объекты, создаваемые в рамках решения задач периодических расчетов.

2) Создание плана видов расчетов «ОсновныеНачисления». Создадим новый план видов расчета «ОсновныеНачисления» в соответствующей ветви конфигурации. Добавим новый объект к подсистеме «РасчетЗарплаты».

На закладке «Данные» изменим настройки стандартных реквизитов: Для Кода установим длину 5, а для Наименования – 100.

Создадим перечисление «СпособыРасчета». В качестве его значений зададим следующие:

- «ПоОкладу»,
- «Процентом»,
- «ЧасовойТариф»,
- «СвободныйГрафик»,
- «ШтрафЗаПрогул»,
- «Фиксировано»,
- «ПропорциональноБазе».

Создадим справочник «КатегорииРасчета», без реквизитов и табличных частей, зададим тип Число для реквизита Код. В этот справочник внесем несколько предопределенных элементов: «Первичное», «ЗависимоеПервогоУровня», «ЗависимоеВторогоУровня». Созданный справочник добавим в подсистему «РасчетЗарплаты».

Создадим справочник «ГрафикиРаботы». Добавим в него реквизит «ВыходныеДни» (тип Строка, длина 7).

Для плана видов расчета «ОсновныеНачисления» зададим реквизиты:

«СпособРасчета», тип ПеречислениеСсылка.СпособыРасчета;

«КатегорияРасчета», тип СправочникСсылка.КатегорииРасчета.

На закладке «Расчет» настроим ряд свойств плана видов расчета.

«ИспользуетПериодДействия», тип «Булево». Установим для нашего плана видов расчета значение этого свойства в значение Истина. Установка этого свойства означает, что в плане видов расчета будут храниться виды расчета, которые по расчетной формуле зависят от отработанного времени, следовательно, для них может быть задана зависимость по периоду действия от других, для них может быть задан состав вытесняющих видов расчета. При свойстве, установленном в значение Истина, конфигуратор создает для плана видов расчета стандартную табличную часть «ВытесняющиеВидыРасчета».

«ЗависимостьОтБазы» – это свойство, которое представляет собой переключатель с тремя значениями «Не зависит», «Зависит по периоду действия», «Зависит по периоду регистрации». Если установить переключатель в значение «Не зависит», то это будет означать, что в данном плане нельзя будет хранить такие виды расчета, формула которых зависит от суммы базы. В нашем плане видов расчета будет использоваться зависимость от базы. Поэтому механизму получения суммы базы надо задать один из двух вариантов критерия суммирования результата базового вида расчета:

– попадание периода действия этого результата регистра в базовый период;

– попадание периода регистрации этого результата регистра в базовый период;

Для начислений обычно выбирают вариант «Зависит по периоду действия», а для удержаний – «Зависит по периоду регистрации».

Наш план видов расчета содержим виды расчета, которые относятся к начислениям. Поэтому установим это свойство «Зависимость от базы» в значение «Зависит по периоду действия».

«Базовые планы видов расчета» – эта настройка определяет, где могут для какого-либо вида расчета, зависящего от базы, находиться базовые планы видов расчета. Если в конфигурации несколько планов видом расчета, то в качестве базовых планов видов расчета может быть отмечено несколько. А нашем случае пока план видов расчетов один, поэтому ставим флаг напротив «ОсновныеНачисления».

Создадим для плана видов расчета форму списка. В качестве полей, выносимых на форму, отметим: «Код», «Наименование», «ПериодДействияБазовый», «СпособРасчета», «КатегорияРасчета». Создадим также форму вида расчета.

В режиме исполнения внесем несколько видов расчета:

– Вид расчета ***«Оклад»***. Код: 00001; Наименование – Оклад. Базовый период действия флаг не установлен. Способ расчета: «По окладу». Категория расчета: «Первичное».

– Вид расчета ***«Оклад на выезде».*** Код: 00002; Наименование – Оклад на выезде. Базовый период действия флаг не установлен. Способ расчета: «По окладу». Категория расчета: «Первичное». Этот вид расчета отличается от вида расчета «Оклад» повышенной ставкой оклада. Начисляется он сотруднику, в

период, когда он работает на выезде (например, выезжает в налоговую инспекцию или на другую территорию, где не комфортные условия труда). Когда сотрудник находится на выезде, ему должен начисляться только вид расчета «Оклад на выезде», а когда сотрудник работает на своем рабочем месте – только вид расчета «Оклад».

Значит вид расчета «Оклад на выезде» является вытесняющим для вида расчета «Оклад». Поэтому вернемся к первому виду расчета, откроем вкладку «Вытесняющие виды расчета». Добавим в табличную часть новую строку и введем в нее вид расчета «Оклад на выезде».

– Вид расчета ***«Доплата за квалификацию».*** Код: 00003; Наименование – Доплата за квалификацию. Базовый период действия флаг не установлен. Способ расчета: «Процентом». Категория расчета: «ЗависимоеПервогоУровня». В табличную часть «Базовые виды расчета» внесем два вида расчета: «Оклад» и «Оклад на выезде».

3) Создание регистра расчета «ОсновныеНачисления». Регистр расчета в механизме периодических расчетов играет роль хранилища первичной информации и «калькулятора» результирующих показателей, таких как, количество отработанного времени по плану в рассчитываемом периоде, количество фактически отработанного времени, сумма базовых начислений и др.

Алгоритм реализации расчетов с использованием регистров расчета следующий:

– Регистру расчета передаются исходные данные, которые нужны для получения необходимых данных: период действия, состав базовых видов расчета и т.п.

– Выполняются запросы к виртуальным таблицам регистра расчета, результаты которых будут содержать необходимые данные: количество рабочего времени (плановое и фактическое) и/или сумму базовых начислений.

Полученные необходимые данные подставляются в расчётные формулы, получаются результаты расчета каждого вида расчета в рублях и эти результаты записываются в регистр расчета.

Каждая запись регистра расчета хранит как исходные данные вида расчета, так и полученный результат расчета.

Создадим новый регистр расчета «ОсновныеНачисления». Добавим его в подсистему «РасчетЗарплаты». На закладке «Основные» настроим ряд свойств регистра:

«План видов расчета» – введен в это свойство ссылку на созданный ранее план видов расчета «ОсновныеНачисления».

«Период действия» – поскольку в нашем регистре будут рассчитываться виды расчета «оклад» и «Оклад на выезде», которые зависят от количества отработанного времени, то для нашего регистра этот флаг должен быть установлен. Установка этого флага означает, что регистр расчета будет снабжен функционалом получения рабочего времени – планового и фактического. Если бы мы его установили это флаг, то регистр не смог бы выдавать эти показатели, обращение к регистру для их получения вызывала бы программную ошибку.

После установки флажка «Период действия» стали доступны свойства: «График», «Значение графика», «Дата графика».

Для настройки этих свойств предварительно надо создать ***регистр сведений «ГрафикиРаботы»***. Его свойства настроим следующим образом:

Периодичность: Непериодический.

Режим записи: Независимый.

Измерения:

Дата, тип Дата;

ГрафикРаботы (тип СправочникСсылка. ГрафикиРаботы).

Ресурс: Значение (тип Число, длина 5, точность 2).

Назначение этого регистра – хранить сведения о длительности рабочего времени для каждой календарной даты. В нашем случае он будет хранить 0 (нерабочий день) или 1 (рабочий день).

В свойстве «График» регистра расчета укажем ссылку на этот регистр сведений «ГрафикиРаботы». В свойство «Значение» графика нужно занести имя ресурса, значения которого будут суммироваться для подсчета рабочего времени (в нашем случае это ресурс регистра сведений «Значение»). В свойство «Дата графика» регистра расчета надо указать им измерения регистра сведений, в котором хранятся календарные даты (в нашем случае это измерение регистра сведений «Дата».

«Базовый период» **–** установка этого флага означает, что регистр расчета будет обладать функционалом получения суммы базы. Для нашего регистра установим значение этого свойства в Истина, поскольку в нашем регистре будут рассчитываться результат вида расчета «Доплата за квалификацию», который зависит от базы.

«Периодичность» свойство, которое может принимать значения «День», «Месяц», «Квартал», «Год». Для целей расчета заработной платы оставим значение свойства Периодичность по умолчанию «Месяц».

Как и любой регистр, регистр расчета может иметь (и как правило имеет) ряд прикладных полей, создаваемых разработчиком. Как минимум одно измерение необходимо для указания сотрудника, для которого ведется этот расчет. Если расчеты сотрудника требуется детализировать, например, по должностям, в случае если есть внутреннее совместительство, то у регистра расчета может быть создано более одного измерения.

Результаты расчета хранятся в регистре расчета в ресурсах, которые для этого типа регистра всегда имеют тип Число. Их может быть несколько, если требуется, например, рассчитывать результаты расчета в нескольких валютах.

Кроме того, регистр расчета, как и любой другой регистр, может иметь реквизиты, которые являются вспомогательными полями. В них можно записывать информацию, относящуюся к конкретной записи. И у регистра расчета реквизиты могут активно участвовать в работе его функционала.

Зададим для нашего регистра расчета прикладные поля: измерения, ресурсы и реквизиты.

Измерения:

«Сотрудник», тип СправочникСсылка.ФизическиеЛица;

«Должность», тип СправочникСсылка.Должности.

Ресурсы:

«Результат». Ресурсы регистра расчета всегда имеют тип Число. Однако, настроим для него длину 15, точность 2.

Реквизиты:

«Размер», тип Число, длина 15, точность 2. Во многих формулах (способах расчета) имеется какая-то «опорная» числовая величина, имеющая в разных формулах разный смысл (например, месячная ставка оклада, процент премии и т.п.). Это значение удобно хранить прямо в записи регистра расчета для удобства подстановки в формулу при расчетах.

«ГрафикРаботы», тип СправочникСсылка.ГрафикиРаботы. Чтобы этот реквизит мог служить для передачи значения графика механизму регистра расчета, установим значение свойства этого реквизита, называемое «Связь с графиком» (рис. 8.1).

В этом свойстве укажем имя измерения регистра-календаря (регистр сведений «ГрафикиРаботы»), в котором содержатся значения графиков (измерение «ГрафикРаботы»).

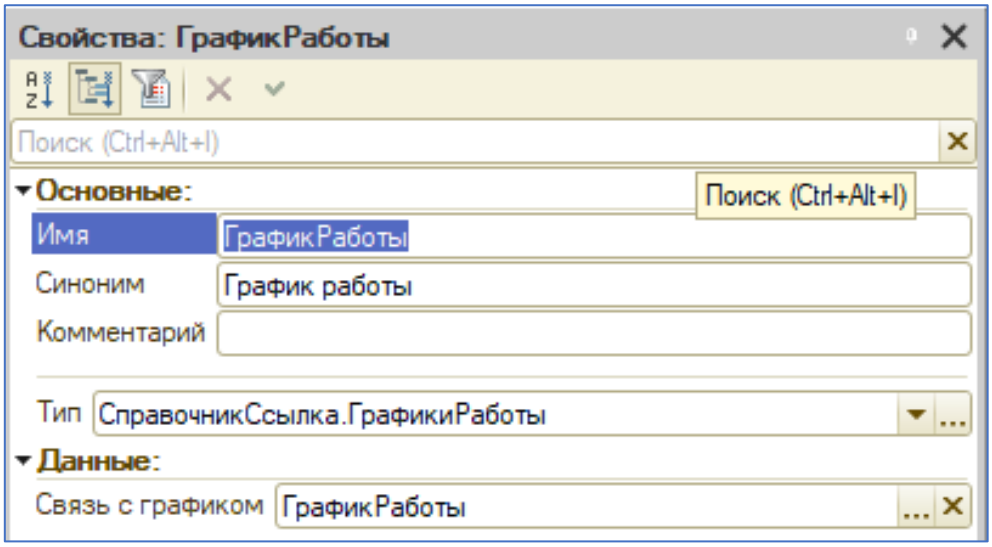


Рис. 8.1 Настройка свойств реквизита «ГрафикРаботы» регистра расчета

4) Назначение регистратора регистра расчета «ОсновныеНачисления». Регистры расчета имеют подчиненный режим записи, поэтому для регистра расчета обязательно должен быть указан хоть один регистратор.

Создадим документ «РасчетЗарплаты». Добавим в его структуру следующий реквизит: «ПериодРегистрации», тип Дата, состав даты – Дата.

Табличную часть документа назовем «ОсновныеНачисления» и добавим в нее следующие реквизиты:

«Сотрудник», тип СправочникСсылка.ФизическиеЛица;

«Должность», тип СправочникСсылка.Должности;

«ВидРасчета», тип ПланВидовРасчетаСсылка.ОсновныеНачисления;

«ПериодДействияНачало», тип Дата, состав даты – Дата;

«ПериодДействияКонец», тип Дата, состав даты – Дата;

«БазовыйПериодНачало», тип Дата, состав даты – Дата;

«БазовыйПериодКонец», тип Дата, состав даты – Дата;

«Размер», тип Число, длина 10, точность 2;

«ГрафикРаботы», тип СправочникСсылка.ГрафикиРаботы; «Сторно», тип Булево.

На закладке «Движения» отметим, что документ является регистратором для регистра расчета «ОсновныеНачисления».

Создадим для документа форму документа. Настроим в ней видимость движений по регистру расчета (закладка «Командный интерфейс», группа «Перейти», установить флаг «Видимость» в строке регистра «ОсновныеНачисления»).

5) Создание плана видов расчета «Премии». Создадим в конфигурации новый объект – план видов расчета «Премии», который будет хранить несколько видов расчета («ПерсональнаяПремия», «МесячнаяПремия» и др.), формулы которых не содержат в качестве операнда плановое количество рабочего времени или количество фактического времени. Включим новый объект в подсистему «РасчетЗарплаты».

На закладке «Данные» установим длину кода 5, длину наименования – 100. В качестве реквизитов укажем:

«СпособРасчета», тип ПеречислениеСсылка.СпособыРасчета;

«КатегорияРасчета», тип СправочникСсылка.КатегорииРасчета.

На закладке «Расчет» выполним следующие настройки:

«Использует период действия» – флаг не отмечен. Поскольку ни одна расчетная формула видов расчета, входящих в данный план, не использует данные графика, то использование периода действия нашему плану видов расчета не требуется. У плана видов расчета «Премии» не будет стандартной табличной части «Вытесняющие виды расчета».

«Зависимость от базы» – переключатель установлен на значение «Зависит по периоду действия».

«Базовые планы видов расчетов» – выбраны планы видов расчета «ОсновныеНачисления», «Премии».

Создадим для плана видов расчета форму списка и форму вида расчета.

В режиме исполнения внесен в новый план видов расчета «Премии» несколько видов расчета:

– Вид расчета ***«Персональная премия»***. Код: 00001; Наименование – Персональная премия. Способ расчета: «Фиксировано». Категория расчета: «Первичное».

– Вид расчета ***«Месячная премия».*** Код: 00002; Наименование – Месячная премия. Способ расчета: «Процентом». Категория расчета: «ЗависимоеПервогоУровня». Базовые виды расчета: «Персональная премия», «Оклад», «Оклад на выезде».

– Вид расчета ***«Премия за период».*** Код: 00003; Наименование – Премия за период. Способ расчета: «Процентом». Категория расчета: «ЗависимоеПервогоУровня». В табличную часть «Базовые виды расчета» внесем вид расчета «Оклад».

– Вид расчета ***«Поощрительная надбавка».*** Код: 00004; Наименование – Поощрительная надбавка. Способ расчета: «Процентом». Категория расчета: «ЗависимоеВторогоУровня». В табличную часть «Базовые виды расчета» внесем вид расчета «МесячнаяПремия».

6) Создание регистра расчета «Премии». Создадим новый регистр расчета «Премии». Отнесем его в подсистеме «РасчетЗарплаты».

На закладке «Основные» укажем в свойстве «План видов расчета» ссылку на план видов расчета «Премии». Флаг «Период действия» оставим не отмеченным. Флаг «Базовый период» отметим, поскольку виды расчета этого плана в основном рассчитываются с использованием расчетной базы. Периодичность укажем «Месяц».

В качестве прикладных полей в регистре расчета «Премии» укажем:

Измерения:

«Сотрудник», тип СправочникСсылка.ФизическиеЛица;

«Должность», тип СправочникСсылка.Должности.

Ресурсы:

«Результат». Ресурсы регистра расчета всегда имеют тип Число. Однако, настроим для него длину 15, точность 2.

Реквизиты:

«Размер», тип Число, длина 15, точность 2.

В качестве регистратора для регистра укажем документ «РасчетЗарплаты».

7) Доработка структуры документа «РасчетЗарплаты». В документ добавим новую табличную часть «Премии». В ее состав включим следующие реквизиты:

«Сотрудник», тип СправочникСсылка.ФизическиеЛица;

«Должность», тип СправочникСсылка.Должности;

«ВидРасчета», тип ПланВидовРасчетаСсылка.Премии;

«БазовыйПериодНачало», тип Дата, состав даты – Дата; «БазовыйПериодКонец», тип Дата, состав даты – Дата;

«Размер», тип Число, длина 10, точность 2;

«Сторно», тип Булево.

Вынесем новую табличную часть на форму документа.

5) Создание плана видов расчета «Удержания». Создадим в конфигурации новый объект – план видов расчета «Удержания», который будет хранить несколько видов расчета, связанных с налогами и прочими удержаниями, формулы которых не содержат в качестве операнда плановое количество рабочего времени или количество фактического времени. Включим новый объект в подсистему «РасчетЗарплаты».

На закладке «Данные» установим длину кода 5, длину наименования – 100. В качестве реквизитов укажем:

«СпособРасчета», тип ПеречислениеСсылка.СпособыРасчета;

«КатегорияРасчета», тип СправочникСсылка.КатегорииРасчета.

На закладке «Расчет» выполним следующие настройки:

«Использует период действия» – флаг не отмечен. Поскольку ни одна расчетная формула видов расчета, входящих в данный план, не использует данные графика, то использование периода действия нашему плану видов расчета не требуется. У плана видов расчета «Удержания» не будет стандартной табличной части «Вытесняющие виды расчета».

«Зависимость от базы» – переключатель установлен на значение «Зависит по периоду регистрации».

«Базовые планы видов расчетов» – выбраны планы видов расчета «ОсновныеНачисления», «Премии».

Создадим для плана видов расчета форму списка и форму вида расчета.

6) Создание регистра расчета «Удержания». Создадим новый регистр расчета «Удержания». Отнесем его в подсистеме «РасчетЗарплаты».

На закладке «Основные» укажем в свойстве «План видов расчета» ссылку на план видов расчета «Удержания». Флаг «Период действия» оставим не отмеченным. Флаг «Базовый период» отметим, поскольку виды расчета этого плана в основном рассчитываются с использованием расчетной базы. Периодичность укажем «Месяц».

В качестве прикладных полей в регистре расчета «Удержания» укажем:

Измерения:

«Сотрудник», тип СправочникСсылка.ФизическиеЛица;

«Должность», тип СправочникСсылка.Должности.

Ресурсы:

«Результат». Ресурсы регистра расчета всегда имеют тип Число. Однако, настроим для него длину 15, точность 2.

Реквизиты:

«Размер», тип Число, длина 15, точность 2.

В качестве регистратора для регистра укажем документ «РасчетЗарплаты».

7) Доработка структуры документа «РасчетЗарплаты». В документ добавим новую табличную часть «Удержания», в состав ее реквизитов включим следующие (рис. 8.2):

«Сотрудник», тип СправочникСсылка.ФизическиеЛица;

«Должность», тип СправочникСсылка.Должности;

«ВидРасчета», тип ПланВидовРасчетаСсылка.Удержания;

«БазовыйПериодНачало», тип Дата, состав даты – Дата;

«БазовыйПериодКонец», тип Дата, состав даты – Дата;

«Размер», тип Число, длина 10, точность 2;

«Сторно», тип Булево.

Вынесем новую табличную часть «Удержания» на ранее созданную форму документа «РасчетЗарплаты».

Итак, состав расчетных механизмов платформы для решения задач сложных периодических расчетов включает:

– механизм получения суммы базовых начислений;

– механизм получения суммы рабочего времени по фактическому периоду действия, а также по периоду действия.

Основными функциональными возможностями, которые предоставляет регистр расчета разработчику, являются:

– выбор записей в заданном интервале по заданным критериям;

– выбор записей по регистратору;

– получение значения базы для записей регистра, удовлетворяющих заданному отбору;

– получение данных графика для записей регистра, удовлетворяющих заданному отбору;

– получение данных о записях, подлежащих перерасчету;

чтение, изменение и запись набора записей в регистр.

Взаимодействие разрабатываемых на платформе «1С:Предприятие» приложений с регистром расчета осуществляется по общей схеме для всех регистров:

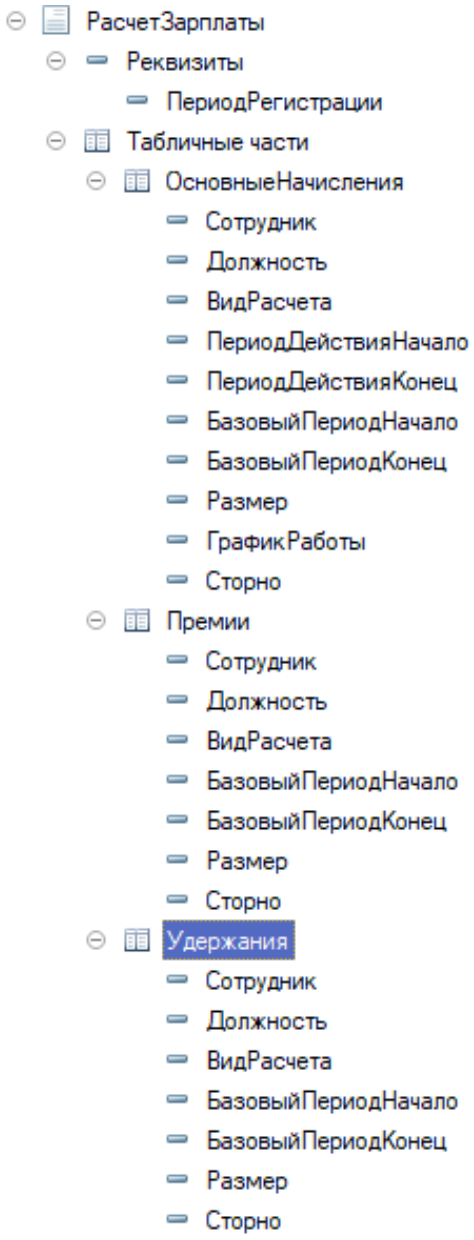


Рис. 8.2. Структура документа «РасчетЗарплаты»

– объекты «Документ» делают записи (движения) в регистр расчета, как в некое хранилище данных;

– приложения формируют к виртуальным таблицам регистра расчета запросы для получения вышеуказанных результирующих показателей суммы базы и отработанного времени, которые подлежат подстановке в расчетные формулы.

Тема 9. Настройка расчетов, формирование и расчет записей регистров расчета

Алгоритм расчета будет состоять из нескольких шагов:

1) **Шаг №1 «Создание записей регистра расчета со стандартными полями, содержащими исходные данные»**.

Для того, чтобы рассчитать по расчетным формулам определенный вид расчета нужно предварительно иметь необходимые данные (сумму базы и/или количество рабочего времени), которые являются операндами формул. Для их получения следует обратиться к регистру расчета, которому для формирования и выдачи необходимых данных нужны некоторые исходные значения, типа границ периода действия, границ базового периода, значений графика работы, на основании которых он и формирует необходимые данные. Регистр расчёта эти исходные данные может получить только из самих записей регистра расчета (стандартные поля «ПериодДействияНачало», «ПериодДействияКонец», «БазовыйПериодНачало», «БазовыйПериодКонец» и т.д.; пользовательский реквизит «ГрафикРаботы»). Таким образом, прежде чем обратиться к регистру расчета для получения необходимых данных, в регистр расчета должны быть записаны подлежащие расчету строки с исходными данными

2) Шаг №2 «Получение необходимых данных».

Для выполнения этого шага приложение обращается к функционалу регистра расчета путем выполнения запросов к виртуальным таблицам регистра расчета. В результате чего будут получены необходимые данные в виде результата запроса (т.е. табличной сущности). Их состав определяется способами расчета, т.е. расчетными формулами, соответствующих видам расчета в записях регистра расчета, созданных на шаге №1.

Для каждой записи регистра расчета, созданной в рамках шага №1, будут получены необходимые данные. При расчете результатов для каждой записи регистра расчета можно будет найти соответствующую ей запись в полученной таблице необходимых данных.

3) Шаг№3 «Расчет результата».

Последовательный обход записей регистра расчета, применение расчетной формулы. Для этого для каждой записи устанавливается соответствие со строкой таблицы необходимых данных, полученных на шаге №2. Из нее извлекаются необходимые данные (данные графика и/или сумма базы) и подставляются в выражения расчетной формулы.

Полученный результат расчета помещается в ресурс «Результат» и запись с рассчитанным результатом записывается в информационную базу.

Поскольку существует зависимость по базе одних видов расчета от других, то второй и третий шаги расчета выполняются столько раз, сколько есть категорий видов расчета. Вначале нужно рассчитать и записать в регистр расчета результаты видов расчета с категорией «Первичное». Это дает возможность получить необходимые данные (суммы базы) для расчета видов расчета категории «Зависимое первого уровня». Рассчитываются и записываются виды

расчета «Зависимое первого уровня». После чего можно получить необходимые данные для видов расчёта с категорией «зависимое второго уровня, рассчитать их и записать. Если есть еще категории цепочка расчетов продолжается по тому же принципу. В нашем случае категорий три.

Итак, шаг №1 выполняется один раз, а второй и третий шаги – по числу категорий, участвующих в расчете.

Для регистров расчета порядок такой: сначала выполняются необходимые расчеты для регистра расчета «Основные начисления», затем для «Премии», потом для «Удержания».

1) Общий модуль «Расчеты». Для реализации первого шага алгоритма расчета создадим общий модуль «Расчеты». В его свойствах следует указать флаг «Сервер». В этом общем модуле реализуем следующую экспортную процедуру:

```
Процедура ЗаполнитьНаборЗаписей(ТекстЗапроса, ДокументСсылка,
                                                    НаборЗаписей) Экспорт
    Запрос = Новый Запрос(ТекстЗапроса);
    Запрос.УстановитьПараметр("Ссылка", ДокументСсылка);
    Выборка=Запрос.Выполнить().Выбрать();
    Пока Выборка.Следующий() Цикл
     НоваяЗапись=НаборЗаписей.Добавить();
     ЗаполнитьЗначенияСвойств(НоваяЗапись, Выборка);
    КонецЦикла;
КонецПроцедуры
```

Процедура создает записи с исходными данными. В качестве аргументов ей передается текст запроса, который получит исходные данные из документа; ссылка на документ, используемая далее для установки параметра запроса; набор записей регистра расчета, в котором записи будут созданы. В процедуре переданный текст запроса выполняется. В цикле перебора полученная выборки результата запроса выполняется создание новой записи для каждой позиции выборки и заполнение ее полей с помощью процедуры глобального контекста «ЗаполнитьЗначенияСвойств».

2) Обработка проведения документа «РасчетЗарплаты». В модуле документа «РасчетЗарплаты» пропишем следующий код проуедуры обработки проведения документа:

```
Процедура ОбработкаПроведения(Отказ, РежимПроведения)
    // 1 шаг расчета
    НаборОсновныеНачисления=Движения.ОсновныеНачисления;
    ТекстЗапроса="ВЫБРАТЬ
    | РасчетЗарплатыОсновныеНачисления.Ссылка.ПериодРегистрации,
    | РасчетЗарплатыОсновныеНачисления.Сотрудник,
    | РасчетЗарплатыОсновныеНачисления.Должность,
    | РасчетЗарплатыОсновныеНачисления.ВидРасчета,
    | РасчетЗарплатыОсновныеНачисления.ПериодДействияНачало,
```

```
        |
        КОНЕЦПЕРИОДА(РасчетЗарплатыОсновныеНачисления.ПериодДействияКонец, ДЕНЬ) КАК ПериодДействияКонец,
        | РасчетЗарплатыОсновныеНачисления.БазовыйПериодНачало,
        | ВЫБОР
        |     КОГДА РасчетЗарплатыОсновныеНачисления.БазовыйПериодКонец = ДАТАВРЕМЯ(1, 1, 1)
        |         ТОГДА ДАТАВРЕМЯ(1, 1, 1)
        |     ИНАЧЕ КОНЕЦПЕРИОДА(РасчетЗарплатыОсновныеНачисления.БазовыйПериодКонец, ДЕНЬ)
        | КОНЕЦ КАК БазовыйПериодКонец,
        | РасчетЗарплатыОсновныеНачисления.Размер,
        | РасчетЗарплатыОсновныеНачисления.ГрафикРаботы,
        | РасчетЗарплатыОсновныеНачисления.Сторно
        |ИЗ
        | Документ.РасчетЗарплаты.ОсновныеНачисления КАК РасчетЗарплатыОсновныеНачисления
        |ГДЕ
        | РасчетЗарплатыОсновныеНачисления.Ссылка = &Ссылка
        |
        |УПОРЯДОЧИТЬ ПО
        | РасчетЗарплатыОсновныеНачисления.НомерСтроки";
        Расчеты.ЗаполнитьНаборЗаписей(ТекстЗапроса, Ссылка, НаборОсновныеНачисления);
        НаборОсновныеНачисления.Записать(, Ложь);
КонецПроцедуры
```

В тексте этой процедуры формируется текст запроса для выборки данных, на основании которых будет заполняться набор данных. Поля запроса содержат все исходные данные, требующиеся для заполнения записи регистра расчета.

3) В общем модуле «Расчеты» создадим еще одну экспортную процедуру, которая получает в качестве аргументов подлежащий расчету набор записей и выборку необходимых данных – результат выполнения запроса к виртуальным таблицам регистра расчета. В цикле осуществляется обход выборки результата запроса. На каждой итерации выполняется расчет очередной записи регистра и результат заносится в ресурс «Результат».

```
Процедура РассчитатьЗаписиРегистраРасчета(НаборЗаписей, Выборка) Экспорт
    Результат=0;
    Пока Выборка.Следующий() Цикл
      ЗаписьРегистра=НаборЗаписей[Выборка.НомерСтроки-1];
```

```
            //....
            Если
Выборка.СпособРасчета=Перечисления.СпособРасчета.ПоОкладу Тогда
                Если Выборка.План>0 Тогда

Результат=ЗаписьРегистра.Размер*Выборка.Отработано/Выборка.План;
                КонецЕсли;
            ИначеЕсли
Выборка.СпособРасчета=Перечисления.СпособРасчета.Процентом Тогда
                Результат=ЗаписьРегистра.Размер*Выборка.СуммаБазы/100;
            ИначеЕсли
Выборка.СпособРасчета=Перечисления.СпособРасчета.Фиксировано  Тогда
                Результат=ЗаписьРегистра.Размер;
        // тут далее могут быть реализованы другие способы расчета
            Иначе
                 Результат=0;
            КонецЕсли;
            ЗаписьРегистра.Результат=Результат;
          КонецЦикла;
КонецПроцедуры
```

4) Реализация шага №2 и шага №3 общего алгоритма расчета. В модуле документа «РасчетЗарплаты» после кода шага №1 допишем код реализации выполнения запроса для получения необходимых данных, получения выборки результат запроса и вызова процедуры общего модуля «РассчитатьЗаписиРегистраРасчета». Эта функциональность позволит получить первый род необходимых данных – данные графика.

```
Запрос = Новый Запрос;
        Запрос.Текст =
          "ВЫБРАТЬ
          |    ОсновныеНачисления.НомерСтроки,
          |    ОсновныеНачисления.ВидРасчета.СпособРасчета КАК
СпособРасчета,
          |
        ЕСТЬNULL(ОсновныеНачисленияДанныеГрафика.ЗначениеПериодДейс
твия, 0) КАК План,
          |
        ЕСТЬNULL(ОсновныеНачисленияДанныеГрафика.ЗначениеФактически
йПериодДействия, 0) КАК Отработано
          |ИЗ
          |    РегистрРасчета.ОсновныеНачисления КАК ОсновныеНачисления
          |            ЛЕВОЕ СОЕДИНЕНИЕ
РегистрРасчета.ОсновныеНачисления.ДанныеГрафика(
          |                        Регистратор = &Регистратор
```

```
        |                                И ВидРасчета.КатегорияРасчета =
&КатегорияРасчета) КАК ОсновныеНачисленияДанныеГрафика
        |            ПО ОсновныеНачисления.НомерСтроки =
ОсновныеНачисленияДанныеГрафика.НомерСтроки
        |ГДЕ
        |    ОсновныеНачисления.Регистратор = &Регистратор
        |    И ОсновныеНачисления.ВидРасчета.КатегорияРасчета =
&КатегорияРасчета";
        Запрос.УстановитьПараметр("КатегорияРасчета",
Справочники.КатегорииРасчета.Первичное);
        Запрос.УстановитьПараметр("Регистратор", Ссылка);
        ВыборкаДляРасчета = Запрос.Выполнить().Выбрать();
        Расчеты.РассчитатьЗаписиРегистраРасчета(НаборОсновныеНачисления,
ВыборкаДляРасчета);
        НаборОсновныеНачисления.Записать(,Истина);
```

Для получения второго рода необходимых данных – суммы базы для тех видов расчета, в расчетных формулах которых присутствует этот операнд, необходимо в процедуре «Обработка проведения» документа расчета зарплаты под кодом расчета «первичное» написать фрагмент кода для расчета вычислений с категорией «ЗависимоеПервогоУровня»:

```
Запрос.Текст =
        "ВЫБРАТЬ
        |    ОсновныеНачисления.НомерСтроки КАК НомерСтроки,
        |    ОсновныеНачисления.ВидРасчета.СпособРасчета                КАК
СпособРасчета,
        |
        ЕСТЬNULL(ОсновныеНачисленияБазаОсновныеНачисления.РезультатБ
аза, 0) КАК СуммаБазы
        |ИЗ
        |    РегистрРасчета.ОсновныеНачисления КАК ОсновныеНачисления
        |            ЛЕВОЕ                                          СОЕДИНЕНИЕ
РегистрРасчета.ОсновныеНачисления.БазаОсновныеНачисления(
        |                        &ИзмеренияОсновногоРегистра,
        |                        &ИзмеренияБазовогоРегистра,
        |                        ,
        |                        Регистратор = &Регистратор
        |                            И ВидРасчета.КатегорияРасчета =
&КатегорияРасчета) КАК ОсновныеНачисленияБазаОсновныеНачисления
        |            ПО ОсновныеНачисления.НомерСтроки =
ОсновныеНачисленияБазаОсновныеНачисления.НомерСтроки
        |ГДЕ
        |    ОсновныеНачисления.Регистратор = &Регистратор
        |    И ОсновныеНачисления.ВидРасчета.КатегорияРасчета =
&КатегорияРасчета
```

```
        |
        |УПОРЯДОЧИТЬ ПО
        |    НомерСтроки";
    // массив имен измерений
    Измерения=Новый Массив();
    Измерения.Добавить("Сотрудник");
    Измерения.Добавить("Должность");
    Запрос.УстановитьПараметр("ИзмеренияОсновногоРегистра",
Измерения);
    Запрос.УстановитьПараметр("ИзмеренияБазовогоРегистра", Измерения);

    Запрос.УстановитьПараметр("КатегорияРасчета",
Справочники.КатегорииРасчета.ЗависимоеПервогоУровня);
    Запрос.УстановитьПараметр("Регистратор", Ссылка);

    Выборка = Запрос.Выполнить(). Выбрать();

    Расчеты.РассчитатьЗаписиРегистраРасчета(НаборОсновныеНачисления,
Выборка);
    НаборОсновныеНачисления.Записать(,Истина);
```

5) Реализация расчетов для видов расчета из второго плана видов расчетов «Премии». Код шага №1 для регистра «Премии» следует поместить под кодом первого шага для регистра «ОсновныеНачисления». Напишем следующий код:

```
НаборПремии=Движения.Премии;
    ТекстЗапроса="ВЫБРАТЬ
            |    РасчетЗарплатыПпремии.Ссылка.ПериодРегистрации,
            |    РасчетЗарплатыПремии.Сотрудник,
            |    РасчетЗарплатыПремии.Должность,
            |    РасчетЗарплатыПремии.ВидРасчета,
            |    РасчетЗарплатыПремии.БазовыйПериодНачало,
            |    ВЫБОР
            |        КОГДА РасчетЗарплатыПремии.БазовыйПериодКонец =
ДАТАВРЕМЯ(1, 1, 1)
            |            ТОГДА ДАТАВРЕМЯ(1, 1, 1)
            |        ИНАЧЕ
КОНЕЦПЕРИОДА(РасчетЗарплатыПремии.БазовыйПериодКонец, ДЕНЬ)
            |    КОНЕЦ КАК БазовыйПериодКонец,
            |    РасчетЗарплатыПремии.Размер,
            |    РасчетЗарплатыПремии.Сторно
            |ИЗ
            |    Документ.РасчетЗарплаты.Премии                КАК
РасчетЗарплатыПремии
            |ГДЕ
            |    РасчетЗарплатыПремии.Ссылка = &Ссылка
```

```
                |
                |УПОРЯДОЧИТЬ ПО
                |    РасчетЗарплатыПpremии.НомерСтроки";
        Расчеты.ЗаполнитьНаборЗаписей(ТекстЗапроса, Ссылка, НаборПремии);
        НаборПремии.Записать(,Ложь);
```

6) Запрос для получения необходимых данных (шаг №2) для регистра «Премии» и расчет (шаг №3) выполним следующим образом.

В конце процедуры «ОбработкаПроведения» запишем вызов новой процедуры для расчета премий с категорией «Первичное»:

```
РассчитатьПремии(НаборПремии, Запрос,
                                    Справочники.КатегорииРасчета.Первичное);
```

В модуле объекта реализуем и саму процедуру «РассчитатьПремии»:

```
Процедура РассчитатьПремии(НаборЗаписей,Запрос,КатегорияРасчета)
        Запрос = Новый Запрос;
        Запрос.Текст =  "ВЫБРАТЬ
                        | ВложенныйЗапрос.НомерСтроки КАК НомерСтроки,
                        | ВложенныйЗапрос.СпособРасчета,
                        | СУММА(ВложенныйЗапрос.СуммаБазы) КАК СуммаБазы
                        |ИЗ
                        | (ВЫБРАТЬ
                        |       Премии.НомерСтроки КАК НомерСтроки,
                        |       Премии.ВидРасчета.СпособРасчета КАК
СпособРасчета,
                        |
        ЕСТЬNULL(ПремииБазаОсновныеНачисления.РезультатБаза, 0) КАК
СуммаБазы
                        | ИЗ
                        |       РегистрРасчета.Премии КАК Премии
                        |               ЛЕВОЕ СОЕДИНЕНИЕ
РегистрРасчета.Премии.БазаОсновныеНачисления(
                        |                       &ИзмеренияОсновногоРегистра,
                        |                       &ИзмеренияБазовогоРегистра,
                        |                       ,
                        |                       Регистратор = &Регистратор
                        |                               И ВидРасчета.КатегорияРасчета
= &КатегорияРасчета) КАК ПремииБазаОсновныеНачисления
                        |               ПО Премии.НомерСтроки =
ПремииБазаОсновныеНачисления.НомерСтроки
                        | ГДЕ
                        |       Премии.Регистратор = &Регистратор
                        |       И Премии.ВидРасчета.КатегорияРасчета =
&КатегорияРасчета
                        |
                        | ОБЪЕДИНИТЬ ВСЕ
```

```
                    |
                    | ВЫБРАТЬ
                    |       Премии.НомерСтроки,
                    |       Премии.ВидРасчета.СпособРасчета,
                    |       ЕСТЬNULL(ПремииБазаПремии.РезультатБаза, 0)
                    | ИЗ
                    |       РегистрРасчета.Премии КАК Премии
                    |           ЛЕВОЕ СОЕДИНЕНИЕ
РегистрРасчета.Премии.БазаПремии(
                    |                   &ИзмеренияОсновногоРегистра,
                    |                   &ИзмеренияБазовогоРегистра,
                    |                   ,
                    |                   Регистратор = &Регистратор
                    |                       И ВидРасчета.КатегорияРасчета
= &КатегорияРасчета) КАК ПремииБазаПремии
                    |           ПО Премии.НомерСтроки =
ПремииБазаПремии.НомерСтроки
                    | ГДЕ
                    |       Премии.Регистратор = &Регистратор
                    |       И Премии.ВидРасчета.КатегорияРасчета =
&КатегорияРасчета) КАК ВложенныйЗапрос
                    |
                    |СГРУППИРОВАТЬ ПО
                    | ВложенныйЗапрос.НомерСтроки,
                    | ВложенныйЗапрос.СпособРасчета
                    |
                    |УПОРЯДОЧИТЬ ПО
                    | НомерСтроки";

        Запрос.УстановитьПараметр("КатегорияРасчета", КатегорияРасчета);
        Запрос.УстановитьПараметр("Регистратор", Ссылка);
        // массив имен измерений
        Измерения=Новый Массив();
        Измерения.Добавить("Сотрудник");
        Измерения.Добавить("Должность");
        Запрос.УстановитьПараметр("ИзмеренияОсновногоРегистра",
Измерения);
        Запрос.УстановитьПараметр("ИзмеренияБазовогоРегистра",
Измерения);
        Выборка = Запрос.Выполнить().Выбрать();
        Расчеты.РассчитатьЗаписиРегистраРасчета(НаборЗаписей,Выборка);
        НаборЗаписей.Записать(,Истина);
КонецПроцедуры
```

7) Для расчета видов расчета, имеющих категорию «ЗависимоеПервогоУровня» и «ЗависимоеВторогоУровня» обеспечим, дописав вызовы в обработку проведения документа:

```
РассчитатьПремии(НаборПремии, Запрос,
                 Справочники.КатегорииРасчета.ЗависимоеПервогоУровня);
РассчитатьПремии(НаборПремии, Запрос,
                 Справочники.КатегорииРасчета.ЗависимоеВторогоУровня);
```

8) Для реализации расчетов по регистру «Удержания» добавим в обработку проведения документа вызов новой процедуры:

```
РассчитатьУдержания(НаборУдержания, Запрос,
                                Справочники.КатегорииРасчета.Первичное);
```

И реализовать в модуле объекта документа новую процедуру РассчитатьУдержания() по аналогии с аналогичной процедурой для регистра «Премии».

```
Процедура РассчитатьУдержания(НаборЗаписей,Запрос,КатегорияРасчета)
    Запрос = Новый Запрос;
    Запрос.Текст =  "ВЫБРАТЬ
                    | ВложенныйЗапрос.НомерСтроки КАК НомерСтроки,
                    | ВложенныйЗапрос.СпособРасчета,
                    | СУММА(ВложенныйЗапрос.СуммаБазы) КАК СуммаБазы
                    |ИЗ
                    | (ВЫБРАТЬ
                    |       Удержания.НомерСтроки КАК НомерСтроки,
                    |       Удержания.ВидРасчета.СпособРасчета КАК
СпособРасчета,
                    |
    ЕСТЬNULL(УдержанияБазаОсновныеНачисления.РезультатБаза, 0)
КАК СуммаБазы
                    | ИЗ
                    |       РегистрРасчета.Удержания КАК Удержания
                    |           ЛЕВОЕ СОЕДИНЕНИЕ
РегистрРасчета.Удержания.БазаОсновныеНачисления(
                    |                   &ИзмеренияОсновногоРегистра,
                    |                   &ИзмеренияБазовогоРегистра,
                    |                   ,
                    |                   Регистратор = &Регистратор
                    |                       И ВидРасчета.КатегорияРасчета
= &КатегорияРасчета) КАК УдержанияБазаОсновныеНачисления
                    |           ПО Удержания.НомерСтроки =
УдержанияБазаОсновныеНачисления.НомерСтроки
                    | ГДЕ
                    |       Удержания.Регистратор = &Регистратор
                    |       И Удержания.ВидРасчета.КатегорияРасчета =
&КатегорияРасчета
```

```
                    |
                    | ОБЪЕДИНИТЬ ВСЕ
                    |
                    | ВЫБРАТЬ
                    |       Удержания.НомерСтроки,
                    |       Удержания.ВидРасчета.СпособРасчета,
                    |       ЕСТЬNULL(УдержанияБазаПремии.РезультатБаза, 0)
                    | ИЗ
                    |       РегистрРасчета.Удержания КАК Удержания
                    |           ЛЕВОЕ СОЕДИНЕНИЕ
РегистрРасчета.Удержания.БазаПремии(
                    |                   &ИзмеренияОсновногоРегистра,
                    |                   &ИзмеренияБазовогоРегистра,
                    |                   ,
                    |                   Регистратор = &Регистратор
                    |                       И ВидРасчета.КатегорияРасчета
= &КатегорияРасчета) КАК УдержанияБазаПремии
                    |           ПО Удержания.НомерСтроки =
УдержанияБазаПремии.НомерСтроки
                    | ГДЕ
                    |       Удержания.Регистратор = &Регистратор
                    |       И Удержания.ВидРасчета.КатегорияРасчета =
&КатегорияРасчета) КАК ВложенныйЗапрос
                    |
                    |СГРУППИРОВАТЬ ПО
                    | ВложенныйЗапрос.НомерСтроки,
                    | ВложенныйЗапрос.СпособРасчета
                    |
                    |УПОРЯДОЧИТЬ ПО
                    | НомерСтроки";

        Запрос.УстановитьПараметр("КатегорияРасчета", КатегорияРасчета);
        Запрос.УстановитьПараметр("Регистратор", Ссылка);
        // массив имен измерений
        Измерения=Новый Массив();
        Измерения.Добавить("Сотрудник");
        Измерения.Добавить("Должность");
        Запрос.УстановитьПараметр("ИзмеренияОсновногоРегистра",
Измерения);
        Запрос.УстановитьПараметр("ИзмеренияБазовогоРегистра",
Измерения);
        Выборка = Запрос.Выполнить().Выбрать();
        Расчеты.РассчитатьЗаписиРегистраРасчета(НаборЗаписей,Выборка);
        НаборЗаписей.Записать(,Истина);
КонецПроцедуры
```

Заключение

В рамках работы на практических занятиях обучающимися освоены навыки разработки учетных систем на платформе «1С:Предприятие» для заданной предметной области, использования Конфигуратора «1С:Предприятие» для создания структуры прикладных объектов и разработка командного интерфейса приложения (подсистемы, роли). Освоены навыки программирование различных интерфейсных задач в формах и работы с асинхронными вызовами в системе «1С:Предприятие».

Получены навыки конфигурирования и программирования прикладных объектов для организация решения задач оперативного учета в «1С:Предприятие», организации синтетического и аналитического бухгалтерского учета, решения расчетных задач в приложениях на платформе«1С:Предприятие», разработки сложных отчетов с помощью системы компоновки данных.

Библиографический список

1. https://its.1c.ru – сайт Информационно-технологического сопровождение пользователей системы «1С:Предприятие».

2. Профессиональная разработка в системе «1С:Предприятие»: в 2 т. / 2-е изд.– М.: 1С-Паблишинг, 2012.–Т.1.–690с.

3. Профессиональная разработка в системе «1С:Предприятие»: в 2 т. / 2-е изд.– М.: 1С-Паблишинг, 2012.–Т.2.–683с.