

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

**Институт информационных технологий и управления
в технических системах**

**«ИССЛЕДОВАНИЕ МЕТОДОВ ПОСТРОЕНИЯ
ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ В СИСТЕМЕ
1С: ПРЕДПРИЯТИЕ 8»**

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Часть 3

по дисциплине

«Платформа 1С»

для студентов дневной формы обучения специальности 09.03.02

«Информационные системы и технологии»

**Севастополь
2017**

УДК 004.42

Методические указания к лабораторной работе «Исследование методов построения пользовательских интерфейсов в системе 1С: Предприятие 8» по дисциплине «Платформа 1С» для студентов дневной формы обучения. Сост. Шишкевич В.Е., Лагуткина Т.В., Савин А.Ю.– Севастополь: Изд-во СГУ, 2017. – 48 с.

Целью настоящих указаний является оказание помощи студентам в изучении основных объектов конфигурации , встроенного языка конфигураторе программного комплекса 1С:Предприятие.

Методические указания рассмотрены и утверждены на заседании кафедры Информационных систем (протокол №10 от 21.04 2017 года).

Допущено учебно-методическим центром СевГУ в качестве методических указаний.

Рецензент: к.т.н., доцент кафедры «Информационные технологии и компьютерные системы» ИИТиУвТС СевГУ Кожаяев Е.А.

СОДЕРЖАНИЕ

1. ИССЛЕДОВАНИЕ МЕТОДОВ ПОСТРОЕНИЯ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ В СИСТЕМЕ 1С: ПРЕДПРИЯТИЕ 8».....	4
1.1. ЦЕЛЬ РАБОТЫ	4
1.2. ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ	4
1.2.1. Интерфейс пользователя 1С. Основные сведения.....	4
1.2.2. Интерфейс «Такси»	6
1.2.3. Разработка пользовательского интерфейса	10
1.2.4. Формы	10
1.2.4.1. Общие формы	12
1.2.4.2. Формы объектов конфигурации	12
1.2.4.3. Реквизиты формы. Основной реквизит формы	14
1.2.4.4. Создание и редактирование форм	15
1.2.4.5. Элементы управления в формах.....	16
1.2.4.6. Использование встроенного языка при работе с формами	17
1.3. ПРИМЕРЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ	24
1.4. ПРОГРАММА ВЫПОЛНЕНИЯ РАБОТЫ	27
1.5. СОДЕРЖАНИЕ ОТЧЕТА	27
1.6. КОНТРОЛЬНЫЕ ВОПРОСЫ	28
2. «ИССЛЕДОВАНИЕ ЯЗЫКА ЗАПРОСОВ В СИСТЕМЕ 1С:ПРЕДПРИЯТИЕ 8»	29
2.1. ЦЕЛЬ РАБОТЫ	29
2.2. ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ	29
2.2.1. Описание конструкций языка запросов	30
2.2.2. Описание и установка внешней обработки «Консоль запросов» для управляемого приложения.....	32
2.2.3. Примеры использования функций и конструкций языка запросов.....	33
2.3. ПРИМЕРЫ ВЫПОЛНЕНИЯ	37
2.4. ПРОГРАММА ВЫПОЛНЕНИЯ РАБОТЫ	44
2.5. СОДЕРЖАНИЕ ОТЧЕТА	44
2.6. КОНТРОЛЬНЫЕ ВОПРОСЫ	45
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	46
ПРИЛОЖЕНИЕ А	47

1. ИССЛЕДОВАНИЕ МЕТОДОВ ПОСТРОЕНИЯ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ В СИСТЕМЕ 1С: ПРЕДПРИЯТИЕ 8»

1.1. ЦЕЛЬ РАБОТЫ

Целью данной лабораторной работы является исследование методов построения пользовательских интерфейсов в системе 1С: Предприятие 8.

1.2. ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

1.2.1. Интерфейс пользователя 1С. Основные сведения.

Платформа 1С поддерживает три вида пользовательского интерфейса:

- обычный интерфейс;
- управляемый интерфейс;
- интерфейс Такси.

Возможность использования каждого вида интерфейса зависит от используемого клиента 1С: Предприятие 8.

В Таблице 1 приведена совместимость клиентов 1С: Предприятие и видов пользовательских интерфейсов.

Таблица 1 – Совместимость клиентов 1С

Вид клиента / Интерфейс	Обычный интерфейс	Управляемый интерфейс	Такси
Обычное приложение	Да	Да	
Тонкий клиент		Да	Да
WEB-клиент		Да	Да

В рамках данного предмета будет рассмотрен управляемый интерфейс и его развитие Такси.

Интерфейсы, реализованные в платформе «1С:Предприятие», отвечают современным требованиям. Пользовательский интерфейс «1С:Предприятие» называется «управляемым» потому, что с помощью прав и ролей пользователей, функциональных опций и настроек пользователя платформа автоматически строит командный интерфейс, управляемые формы и отчеты.

Пользовательский интерфейс «1С:Предприятие» можно охарактеризовать как задаче-ориентированный, позволяющий в каждом отдельном окне приложения (основном или вспомогательном) решать конкретную задачу.

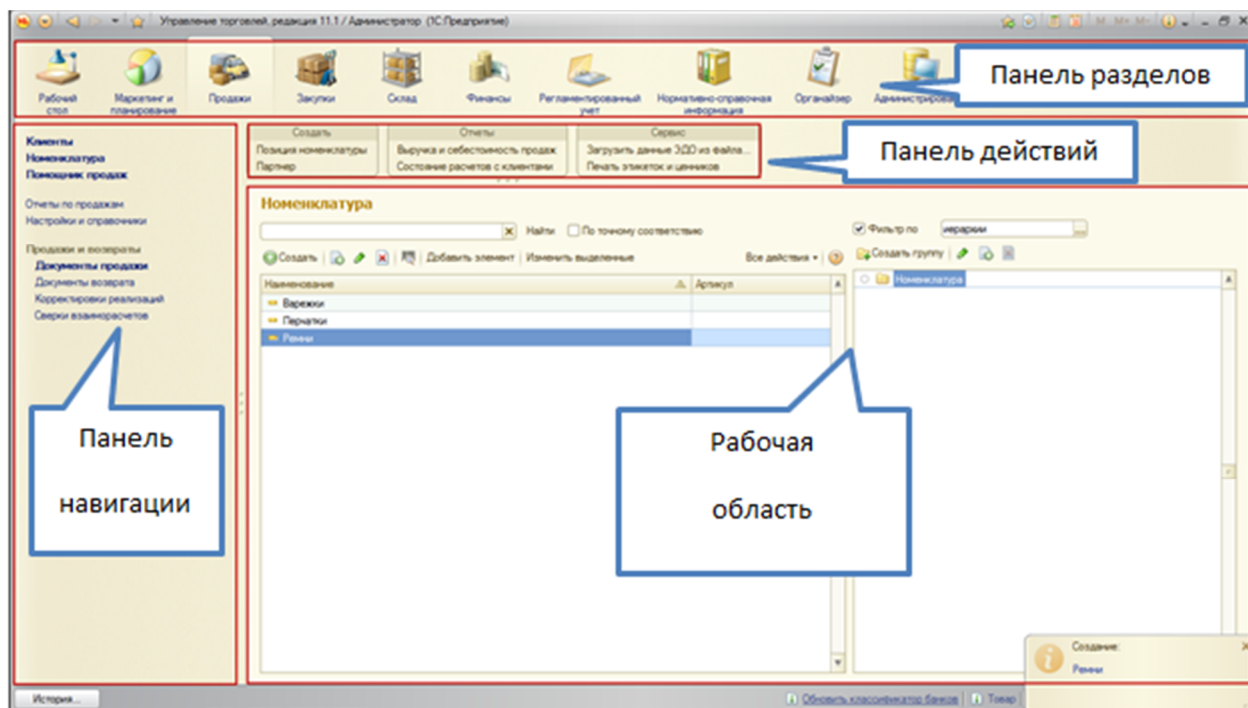


Рисунок 1.1 – Пользовательский интерфейс «1С:Предприятие»

Основное окно приложения предназначено для навигации по прикладному решению и вызова различных команд.

Вспомогательное окно предназначено для работы с объектами информационной базы (например, с документами или элементами справочников), построения отчетов или выполнения обработок данных.

Рабочий стол – это стандартный раздел программы, содержащий часто используемые документы, отчеты, справочники и т. п. Этот раздел автоматически активизируется при запуске прикладного решения.

Панель разделов – это наиболее крупное разделение функциональности прикладного решения. Она расположена в верхней части основного окна и соответствует верхнему уровню подсистем, добавленных в конфигурацию. С ее помощью осуществляется переход к другим разделам программы.

Панель навигации содержит своеобразное «оглавление» раздела. Она включает навигационные команды, которые позволяют перейти к той или иной точке этого раздела.

Панель действий содержит наиболее востребованные в повседневной работе команды, позволяющие быстро создавать новые объекты, выполнять типовые обработки или строить популярные отчеты.

Основное окно приложения также включает в себя главное меню системы и набор вспомогательных команд (калькулятор, календарь и пр.). А также предоставляет различные сервисные возможности, такие как: просмотр истории работы пользователя и возможность перемещаться по разделам прикладного решения, с которыми уже работал пользователь; добавление разделов, списков, объектов базы данных, отчетов и обработок в избранное; отображение в информационной панели (внизу окна) оповещений о последних действиях пользователя; возможность получить ссылку на любые разделы, списки, объекты

базы данных, отчеты и обработки и перейти по полученной ссылке к этим данным и т. д.

Управляемый интерфейс может отображаться в *трех вариантах* внешнего вида:

- формы в закладках. При этом все окна будут отображаться в отдельных закладках одного родительского окна. Это больше похоже на классический интерфейс.

- формы в отдельных окнах. В этом режиме в основном окне отображается только одна форма. Новые формы могут отображаться в других окнах (переход между ними будет доступен за счет средств ОС). Это менее привычный вид приложения, но он имеет свои преимущества.

Кроме уже ставших привычными вариантов интерфейса в платформе 1С:Предприятие 8.3 появился новый интерфейс – Такси.

Интерфейс такси – новое слово в развитии технологической платформы 1С:Предприятие 8.3. Новое лицо программы адаптировано под стандарты веб-приложений.

1.2.2. Интерфейс «Такси»

При создании нового пользовательского интерфейса разработчики платформы ставили перед собой ряд задач. Поскольку с конфигурациями теперь возможно работать через интернет при помощи обычного веб-браузера, заметна ориентация платформы на работу с мобильными устройствами, на ввод информации при помощи сенсорного экрана, реагирующего на прикосновения к нему.

Новый интерфейс имеет отличительные визуальные особенности:

- максимизация рабочего пространства на мониторах с различным разрешением;

- современный дизайн;

- крупный шрифт;

- удобная навигация за счет использования вспомогательных панелей: "Инструменты", "Избранное", "История" и "Полнотекстовый поиск";

- понятный и удобный поиск в списках, управление поиском;

- возможность самостоятельно конструировать рабочее пространство (методом drag&drop).

- адаптация под стандарты веб-приложений;

- расширенные возможности выбора в полях ввода.

При использовании интерфейса Такси приложение представляет из себя основное окно, внутри рабочей области которого открываются все остальные окна. Окна не отображаются отдельно на панели задач Windows, даже если визуально рисуются поверх основного окна (например, формы выбора элемента справочника). Работа начинается с начальной страницы. Эту страницу невозможно закрыть.

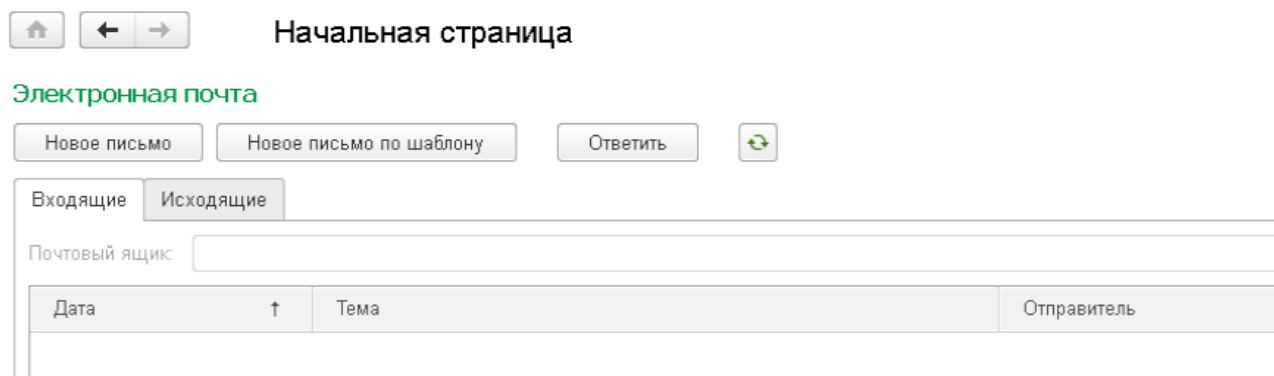


Рисунок 1.2 – Начальная страница

Предусмотрена возможность настраивать в пользовательском режиме формы, отображаемые на начальной странице. Для этого в главном меню следует выбрать пункт Вид → Настройка начальной страницы.

Основное окно приложения состоит из нескольких панелей и областей, каждая из которых имеет свое функциональное назначение. При помощи клавиши F6 можно переключать фокус между формами, расположенными на начальной странице.

Панель разделов

Панель разделов отображает список подсистем конфигурации верхнего уровня. Первым слева всегда отображается раздел с заголовком Главное.



Рисунок 1.3 – Панель разделов

При щелчке мышью на конкретном разделе в панели функций текущего раздела представлены команды выбранного раздела.

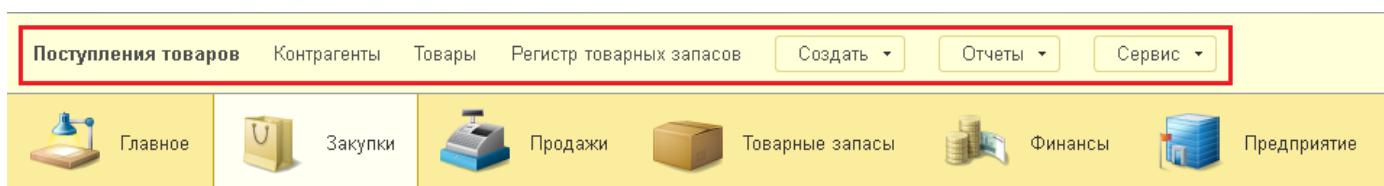


Рисунок 1.4 – Команды раздела «Закупки»

При повторном щелчке мышью на экране разворачивается список всех команд выбранного раздела – меню функций.

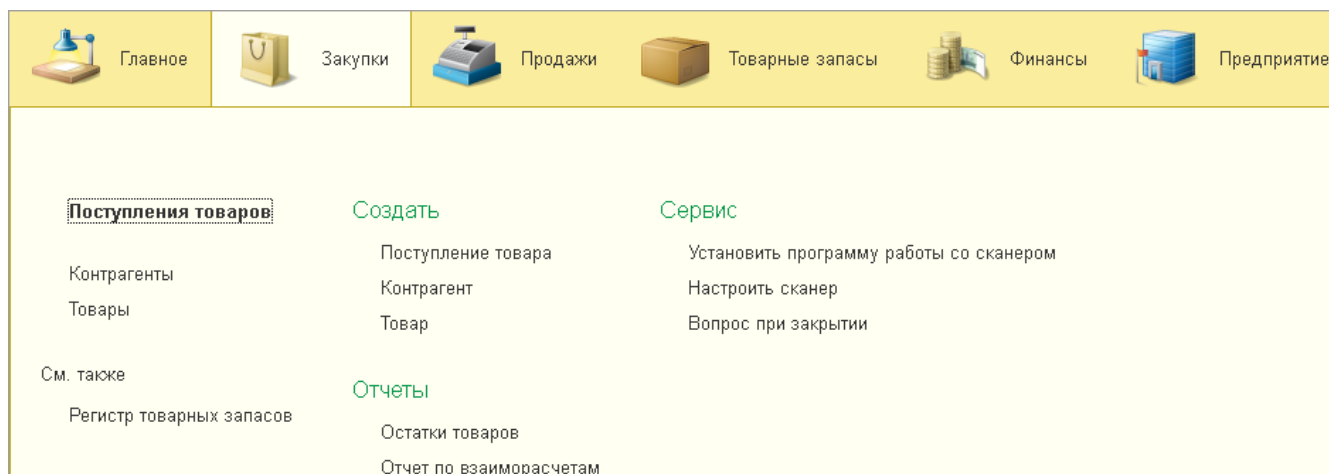


Рисунок 1.5 – Меню функций раздела «Закупки»

Панель функций текущего раздела

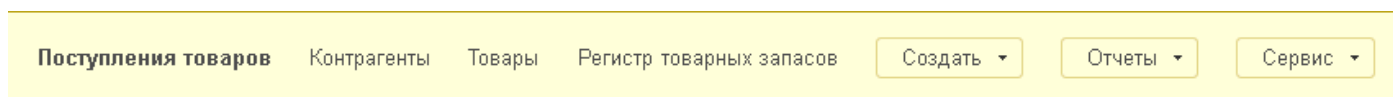


Рисунок 1.6 – Панель функций текущего раздела

На ней отображаются команды выбранного пользователем раздела.

Область системных команд

В левой части области расположен значок доступа к главному меню, содержащему классические пункты Файл, Правка, Вид и т.д. С клавиатуры активизировать главное меню можно при нажатии кнопки F10.

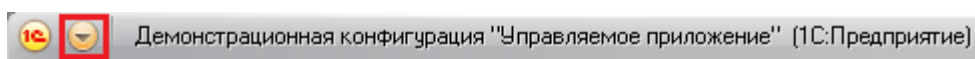


Рисунок 1.7 – Значок доступа к главному меню

В правой части области расположены команды сохранения в файл, печати, предварительного просмотра, работы со ссылками, калькулятор, календарь и т.д.



Рисунок 1.8 – Значки команд

Панель инструментов

На этой панели расположены кнопки открытия меню функций текущего раздела, доступа к избранному, просмотр истории, поиск данных.

Панель открытых

На этой панели выводится список открытых в настоящий момент окон. Формы рабочей области начальной страницы не отображаются как отдельные формы в панели открытых.

Панель истории

Панель истории ускоряет доступ к использованным в последнее время объектам – открытым, отредактированным или добавленным справочникам, документам и т.д. Записи разделены по дням, а напротив каждой строчки указано время использования. В пределах одного дня записи упорядочены по времени доступа.

Панель избранного

В избранном хранятся данные и команды.

Добавить объект в избранное можно при помощи щелчка мышью на значке с изображением звездочки.

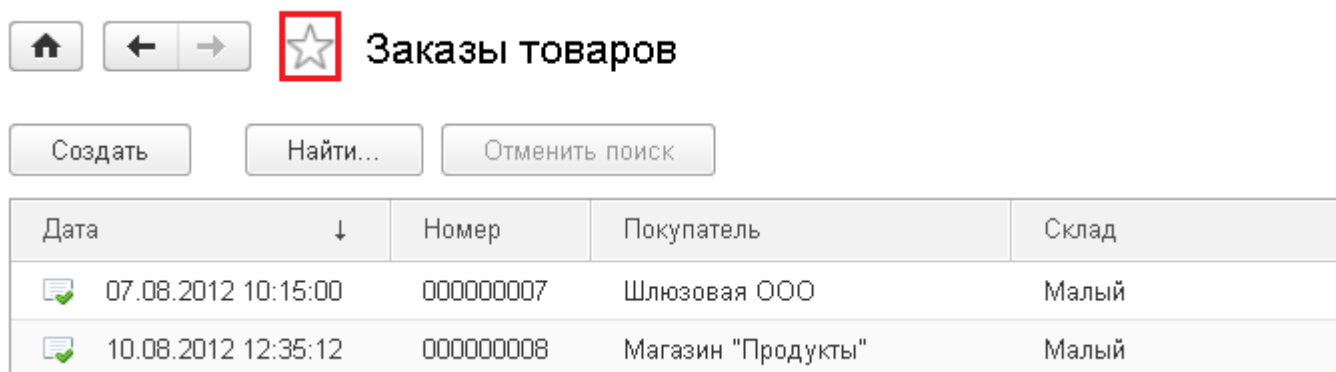


Рисунок 1.9 – Добавление объекта в избранное

На панели избранного также присутствует **строка поиска**. Заголовки элементов, расположенных в избранном, можно редактировать. Для этого предназначена небольшая кнопка с карандашом в правой части строки избранного. В платформе реализована специальная форма полнотекстового поиска. Эта форма доступна только в том случае, если полнотекстовый поиск включен.

Вызвать форму поиска с клавиатуры можно при помощи сочетания клавиш **Ctrl + Shift + F**.

В нижней части основного окна приложения может существовать **информационная панель**. Она предназначена для отображения показателей производительности и индикации того, что включён режим имитации задержек при вызовах сервера. Показатели производительности отображаются в виде отдельной панели основного окна приложения.

Панель отображается, если в параметрах системы установлен флаг «Отображать показатели производительности»:

Для того, что бы **включить** интерфейс Такси достаточно установить в свойствах конфигурации значение «Режим совместимости интерфейса» режим «Такси».

Версия 8.2 – клиентское приложение работает в интерфейсе 8.2. Переключение в интерфейс Такси невозможно.

Версия 8.2. Разрешить Такси – по умолчанию используется интерфейс версии 8.2. Возможно переключение на интерфейс Такси с помощью диалога параметров или командной строки.

Такси. Разрешить Версия 8.2 – по умолчанию используется интерфейс Такси. Возможно переключение на интерфейс версии 8.2 с помощью диалога параметров или командной строки.

Такси – клиентское приложение работает в интерфейсе Такси. Переключение в режим интерфейса 8.2 невозможно.

1.2.3. Разработка пользовательского интерфейса

При организации пользовательского интерфейса разработчик решает две задачи:

- реализация форм прикладных объектов и отчетов для ввода и анализа данных
- реализация интерфейсных элементов приложения: элементов меню, командных панелей, АРМ и т.д.

Для этого в конфигураторе предусмотрены специальные инструменты, рассмотренные ниже.

1.2.4. Формы

Формы в 1С:Предприятии предназначены для отображения и редактирования информации, содержащейся в базе данных. Формы могут принадлежать **конкретным объектам конфигурации** или существовать отдельно от них и использоваться всем прикладным решением в целом – **общие формы** (не принадлежащие конкретным объектам конфигурации).

Основное назначение формы – предоставить пользователю удобное средство для ввода и просмотра информации. Как и бумажный документ, форма позволяет быстро ввести необходимую информацию и запомнить ее для последующей обработки, а при необходимости – вновь вернуться к ранее введенным данным для просмотра или корректировки.

Формы, существующие в «1С:Предприятии», предназначены для интерактивной работы пользователя с данными информационной базы. Для обеспечения этой возможности форма «наполняется» необходимой функциональностью.

Функциональность формы определяется составом ее реквизитов и команд.

Реквизиты формы – это данные, с которыми работает форма.

Все данные формы должны быть описаны в виде реквизитов.

Команды формы – это действия, которые может выполнить форма над данными.

Все команды, которые могут присутствовать в форме, можно разделить на два класса:

- локальные команды формы – обеспечивают доступ к функциональности формы;

- глобальные команды – обеспечивают доступ к функциональности приложения.

Для использования команд, для отображения и редактирования данных, хранимых в реквизитах, служат *элементы формы*, связанные с соответствующими командами и реквизитами.

Все элементы формы подчинены корневому элементу Форма.

Элементы могут представлять собой поля ввода, флажки, переключатели, кнопки и т.д. Кроме этого, элемент может быть группой, включающий другие элементы. Структура элементов описывает то, как будет выглядеть форма.

Свойства элементов дерева зависят от свойств, которыми наделены элементы вышестоящего уровня. При этом каждый из элементов обладает своими уникальными свойствами, характерными только для него.

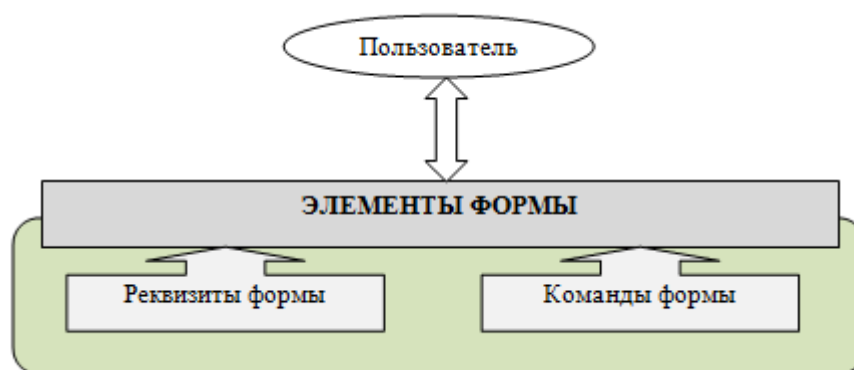


Рисунок 1.10 – Взаимодействие пользователя с элементами формы

Новая форма не прорисовывается детально разработчиком, а *строится системой автоматически*. Разработчик же в режиме конфигурирования:

- определяет состав формы в виде дерева элементов;
- описывает поведение формы, задавая значения для ее свойств и/или реализуя процедуры на встроенном языке.

Разработчик может *самостоятельно* создать форму в конфигураторе. Для создания форм используется *редактор форм*. С помощью этого редактора разработчик создает для формы необходимый набор реквизитов и команд, а также элементы формы, которыми они отображаются (процесс создания формы будет рассмотрен в п.п. 3.3.3).

В редакторе формы определяются:

Реквизиты формы – предназначены для хранения данных, с которыми работает форма.

Параметры формы – предназначены для организации связи между формами и управления функциональностью формы при ее открытии.

Команды формы – предназначены для выполнения различных действий внутри формы.

Модуль формы – содержит программный код, связанный с функционированием формы. Это программа на встроенном языке, отвечающая за работу с элементами управления формы, отработку предопределенных процедур и выполняющий различные вспомогательные вычисления. С их помощью производится подготовка и обработка реквизитов формы, передача управления к исполнению, а также обработка действий пользователя.

Элементы формы – предназначены для отображения и редактирования реквизитов формы, а также для отображения и выполнения команд.

Командный интерфейс – содержит команды, которые могут выполняться в форме и предоставлены глобальным командным интерфейсом и командами формы.

1.2.4.1. Общие формы

Общие формы – формы, которые доступны из места конфигурации. Обычно общие формы используются для форм настроек программы и других сервисных функций. Например, прикрепление файла к документу, установка ЭЦП, структура подчиненности, установка прав доступа, печать документов, рабочие столы и т.д.

Общие формы располагаются в дереве конфигурации, в ветке Общие – Общие формы.

При создании **общей формы** конструктор предлагает выбрать один из типов форм:

Вызвать общую форму можно с помощью метода ОткрытьФорму(). Например:

```
ОткрытьФорму («ОбщиеФормы.ФормаНастроек»);
```

По умолчанию конструктор общих форм предлагает создать произвольную форму.

Произвольная форма – данные, с которыми будет взаимодействовать форма, будут определены разработчиком в дальнейшем, в процессе разработки;

Выбор типа «Произвольная форма» не приводит к созданию основного реквизита. В этом случае поведение формы определяется только настройками пользователя.

1.2.4.2. Формы объектов конфигурации

Существует несколько способов создать форму, подчиненную какому-либо объекту конфигурации:

- воспользоваться кнопкой *Открыть* в палитре свойств объекта конфигурации, у нужного типа формы:

– выделить в дереве конфигурации ветвь *Формы объекта*, для которого будет создаваться форма, и воспользоваться либо контекстным меню, либо кнопкой *Добавить* на панели дерева конфигурации:

– открыть окно редактирования объекта конфигурации, перейти на закладку *Формы* и воспользоваться либо кнопкой *Открыть* у необходимого типа формы, либо кнопкой командной панели этого окна *Добавить*, либо контекстным меню в списке форм.

При использовании любого из вышеперечисленных способов будет открыто окно конструктора формы объекта конфигурации.

В зависимости от вида объекта, его свойств конструктором будут предложены характерные для текущего режима создания **типы форм** (ниже рассмотрены формы изучаемых ранее объектов):

для объекта **СПРАВОЧНИК**:

- форма элемента
- форма списка
- форма выбора

Если справочник иерархический, то возможно создание формы группы и выбора группы

для объекта **ДОКУМЕНТ**

- форма списка
- форма выбора
- форма документа

для объекта **РЕГИСТР СВЕДЕНИЙ**

- форма списка
- форма записи
- форма набора записей

для объекта **РЕГИСТР НАКОПЛЕНИЯ**

- форма списка
- форма набора записей

Кроме всех перечисленных форм, для любого из объектов конфигурации можно создать тип формы **Произвольная**. Такая форма после создания не связана ни с какими данными конфигурации. Действия, которые можно будет выполнять в такой форме, целиком и полностью будут зависеть от желания и способностей разработчика формы.

Если у объекта **несколько форм одного типа**, то одну из них можно выбрать в качестве *основной*. Если при вызове формы объекта не указано явно, какую именно форму следует вызвать, на экран будет выводиться основная форма. Для указания основной формы при создании установите флажок «**Назначить форму основной**».

1.2.4.3. Реквизиты формы. Основной реквизит формы

Реквизиты формы предназначены для хранения данных, с которыми работает форма. В реквизитах хранятся те данные, которые отображаются и редактируются в элементах формы, а также те данные, которые не отображаются, но используются формой в процессе ее работы. Вся совокупность данных, хранящихся в реквизитах формы, называется *данные формы*.

Каждый *элемент формы*, позволяющий изменять данные, связан с некоторым реквизитом формы. Когда пользователь изменяет данные в элементе формы, аналогичные изменения происходят и в связанном с ним реквизите.

Верно и обратное. Когда реквизит формы изменяется программно, изменяется и значение, отображаемое в связанном с ним элементе формы.

Не все реквизиты формы обязательно должны отображаться в форме. Часть реквизитов может быть не связана с элементами формы, а данные, хранящиеся в таких реквизитах, предназначены не для отображения или интерактивного редактирования, а для обеспечения внутренних алгоритмов работы формы или для программного взаимодействия с этой формой из встроеного языка.

Реквизиты формы обладают набором свойств, позволяющих влиять на их поведение.

Добавление нового реквизита формы выполняется стандартным способом – с помощью кнопки командной панели .

Среди всех реквизитов формы, как правило, существует один **основной реквизит** (он выделен жирным шрифтом). Он определяет источник данных для формы в целом. Именно основной реквизит формы определяет ее поведение, вид, состав команд, другие дополнительные возможности, предоставляемые разработчику формы и ее пользователю. Такой реквизит автоматически добавляется конструктором формы, если форма создается подчиненной объекту конфигурации и выбирается вид формы этого объекта, а не произвольная форма. По умолчанию у такой формы после ее создания есть только один реквизит, который и является основным.

Форма обладает рядом свойств, которые присущи ей всегда.

Для редактирования свойств формы необходимо воспользоваться контекстным меню формы (пункт меню Свойства) в дереве объектов конфигурации либо контекстным меню (пункт меню Свойства) ветви Форма в области редактирования элементов формы редактора формы.

Важнейшим свойством, которое определяет не только то, что за данные будет поставлять форме реквизит, но и как будет выглядеть элемент формы, отображающий эти данные, является *Тип*.

От типа значения основного реквизита формы зависит не только то, какие данные будут отображены в элементах формы, но и поведение самой формы.

Например, если основному реквизиту формы указать тип значения ДокументОбъект.ПриходнаяНакладная, то при закрытии формы программа будет запрашивать подтверждение записи и проведения документа. Если же

основному реквизиту формы указать тип значения СправочникОбъект.Клиенты, то подобного подтверждения при закрытии формы возникать не будет.

1.2.4.4. Создание и редактирование форм

Создание новой формы начинается с *конструктора форм*.

Модификация формы, созданной конструктором, осуществляется в *редакторе формы*.

Окно редактора форм разбито на несколько областей, каждая из которых отвечает за ту или иную функциональность будущей формы.

1. Описывается состав и порядок элементов, из которых состоит форма, команды интерфейса, которые могут выполняться в форме.

2. Описывается состав реквизитов формы, а также команды, выполняемые внутри формы. Реквизит можно перетащить влево и он станет элементом формы (полем на форме).

3. Представлен внешний вид формы, как она может выглядеть на экране пользователя, с учетом описанных реквизитов, элементов, команд формы. При изменении каких-либо настроек в окнах редактора они тут же применяются и изменяют вид формы.

Программный модуль описывает работу формы на встроенном языке.

Для редактирования модуля формы можно воспользоваться закладкой Модуль, которая расположена внизу редактора форм.

В модуле формы располагаются обработчики событий формы, элементов формы, команд формы. Помимо предопределенных обработчиков событий разработчик прикладного решения может создавать в модуле формы свои процедуры и функции

Рассмотрим пример создания формы документа *Реализация товаров*.

Из контекстного меню узла Формы этого документа выберем пункт *Добавить*.

В результате откроется окно конструктора форм. В окне конструктора необходимо выбрать тип формы – Форма документа, установить флажок, назначить форму основной и задать имя ОсновнаяФорма.

После завершения работы конструктора для созданной формы откроется окно редактора форм.

В форме должны присутствовать команды для обработки данных и для управления формой. Эти возможности обеспечивают *стандартные локальные команды формы*.

Стандартные команды формы автоматически размещаются в командной панели формы. В нее всегда включаются все команды, предоставленные расширением формы, а также команда Справка, предоставленная формой

В редакторе форм они представлены на закладке Стандартные команды редактора команд:

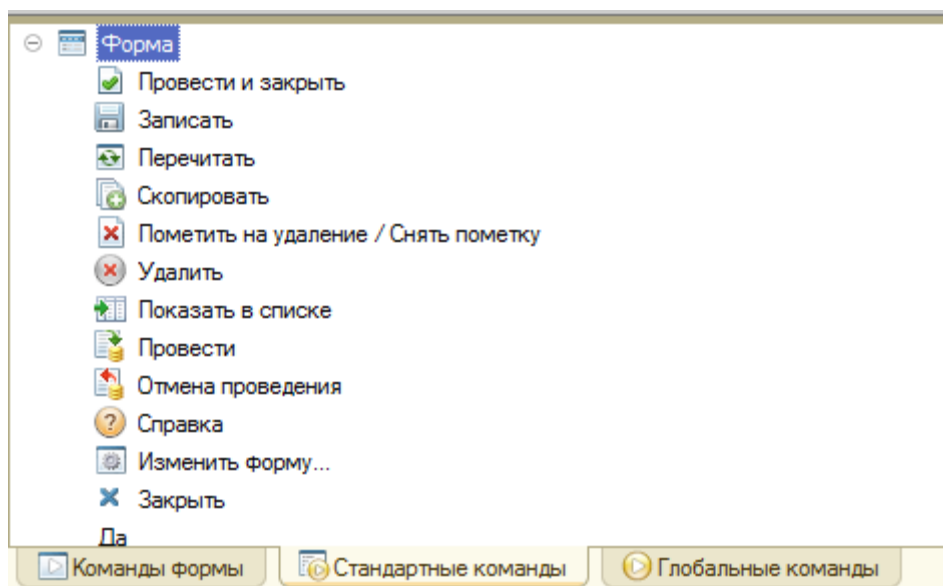


Рисунок 1.11 – Стандартные команды редактора команд

Состав команд, предоставляемых расширением, зависит от типа основного реквизита формы.

При создании формы для пользователя прикладного решения система учитывает его права, определяемые ролью, и значения функциональных опций. Из этого следует, что набор команд формы, доступный конкретному пользователю, может отличаться от состава команд, включенных в форму на этапе разработки.

1.2.4.5. Элементы управления в формах

В форме используются пять элементов:

ГРУППА:

- Обычная группа;
- Группа – Страницы;
- Группа – Командная панель;

ПОЛЕ;

КНОПКА;

ТАБЛИЦА;

ДЕКОРАЦИЯ:

- Декорация – Надпись;
- Декорация – Картинка.

Элемент формы *Поле* предназначен для отображения примитивных типов данных, текстовых, табличных, HTML-документов, диаграмм, календарей, индикаторов. Тип данных, которые отображает элемент Поле, влияет на то, какие значения может принимать его свойство Вид.

Элемент формы *Группа* предназначен для группировки других элементов формы, их выделения, для создания командных панелей, подменю, групп кнопок, страниц. Свойство Вид элемента формы Группа может принимать значения:

Обычная группа. Элемент формы, предназначенный для логической группировки других элементов формы. Элемент Обычная группа может не отображаться на форме, выполняя при этом функции группировки других элементов. Может отображаться в виде Рамка группы, Линия, Отступ.

Командная панель. Элемент формы, предназначенный для группировки кнопок и команд

Страницы. Элемент формы, представляющий собой набор страниц.

Элемент формы *Кнопка* предназначен для создания кнопок, гиперссылок. Свойство Вид элемента формы Кнопка может принимать значения:

Элемент формы *Таблица* предназначен для отображения и редактирования различных табличных данных. У Таблицы могут быть свои командные панели, контекстные меню. Поля колонок элемента Таблица могут принимать различные виды.

Декорация – представляет собой оформительский элемент формы. Не связана с данными (реквизитами формы). Может представлять собой запись или картинку. Рекомендуются использовать только для неизменяемых текстовых пояснений.

Для того чтобы добавить в состав формы новый элемент определенного вида, необходимо воспользоваться одним из этих пяти элементов.

Для отображения различных данных прикладного решения необходимо изменять свойство элемента формы Вид.

Для управления видимостью элементов формы существуют два свойства: **Видимость** (изменяется в редакторе формы (в конфигураторе) и программно) и **Пользовательская видимость** (настраивается только в конфигураторе).

1.2.4.6. Использование встроенного языка при работе с формами

В «1С:Предприятии» нет «единого пространства» для выполнения кода на встроенном языке. Каждый фрагмент кода выполняется в некотором *контексте*.

По большому счету таких контекстов два: это *контекст клиента* и *контекст сервера*.

Основная особенность формы как программного объекта заключается в том, что она существует одновременно и на клиенте, и на сервере.

Соответственно модуль формы, процедуры, написанные в этом модуле, исполняются не в каком-то одном определенном контексте (только в контексте клиента или только в контексте сервера). Для каждой процедуры в модуле формы разработчик в явном виде указывает контекст ее исполнения: на сервере или на клиенте.

Каждый объект конфигурации Форма, описанный в дереве объектов конфигурации, имеет *модуль*, в котором разработчик располагает собственные процедуры на встроенном языке.

Соответственно модуль формы, процедуры, написанные в этом модуле, исполняются не в каком-то одном определенном контексте (только в контексте клиента или только в контексте сервера). Для каждой процедуры в модуле формы

разработчик в явном виде указывает контекст ее исполнения: на сервере или на клиенте. Для этого используются *директивы компиляции*:

&НаКлиенте – означает, что метод выполняется на стороне клиента, а переменная существует все время жизни клиентской части формы.

&НаСервере – означает, что метод выполняется на стороне сервера, а переменная существует только во время вызова выполнения серверного вызова, при каждом серверном вызове значение переменной сбрасывается.

&НаСервереБезКонтекста – определяет серверную процедуру, исполняемую вне контекста формы. Переменные не могут предваряться такой директивой.

&НаКлиентеНаСервереБезКонтекста – используется, когда к процедуре необходимо обращаться как с клиента, так и с сервера. Означает, что метод выполняется как на клиенте, так и на сервере вне контекста формы. Переменные не могут предваряться такой директивой.

Таким образом, разработчик должен понимать, что он фактически работает с двумя контекстами исполнения: серверным и клиентским, и между ними происходит обмен данными

При этом нужно учитывать, что на клиенте поддерживается лишь ограниченный набор типов данных и интерфейсные функции. На клиенте не допускается непосредственная работа с базой данных, работа непосредственно с прикладными объектами, использование запросов, таблиц значений. Доступность описана в синтакс-помощнике и ее необходимо проверять.

Директивой по умолчанию является НаСервере.

Работа с формой из встроенного языка.

1. Открытие формы.

При открытии формы нового объекта (элемента справочника, документа) и при открытии формы существующего объекта возникает различная последовательность событий.

Если открывается форма нового объекта, то сначала происходит начальное заполнение объекта данными (событие Обработка заполнения в модуле объекта, форма которого открывается), а затем вызываются два события формы: сначала на сервере (При создании на сервере) и затем на клиенте (При открытии). Эти два события позволяют подготовить форму к открытию

Если открывается форма существующего объекта, то последовательность событий будет иной. Сначала на сервере вызываются два события формы. Одно (При чтении на сервере) – чтобы подготовить дополнительные данные, которые зависят от данных объекта. Другое (При создании на сервере) – чтобы максимально подготовить форму к открытию. И наконец, на клиенте вызывается еще одно событие формы (При открытии), чтобы выполнить действия, связанные с открытием формы, которые на сервере выполнить невозможно.

Все, что можно сделать на сервере для подготовки формы к открытию, нужно делать в обработчике события *При создании* на сервере. В обработчике

события *При открытии* нужно делать только то, что на сервере сделать невозможно. Например, выдать предупреждение, задать вопрос.

Для того чтобы открыть форму, существуют два способа:

- воспользоваться методом `ОткрытьФорму ()`, он доступен только в контексте клиентов: толстого клиента, тонкого клиента и веб-клиента. На сервере открыть форму нельзя.

- воспользоваться комбинацией метода `ПолучитьФорму ()` и метода `Открыть ()`.

В любом случае можно передать в форму параметры формы.

Открытие основной формы нового объекта:

```
ОткрытьФорму ("Справочник.Товары.ФормаОбъекта");
```

Открытие основных форм списка и выбора:

```
ОткрытьФорму ("Справочник.Товары.ФормаСписка");
```

```
ОткрытьФорму ("Справочник.Товары.ФормаВыбора");
```

```
ОткрытьФорму ("Справочник.Товары.ФормаВыбораГруппы");
```

Открытие формы констант, созданной в конфигурации:

```
ОткрытьФорму ("ОбщаяФорма.ФормаКонстант");
```

Для формы группы иерархического справочника используются *параметры формы*.

Открытие основной формы новой группы. Для этого также используются параметры формы.

Открытие основной формы существующего объекта:

```
СсылкаНаЭлементСправочника = Элементы.Список.ТекущаяСтрока;
```

```
ПараметрыФормы = Новый Структура ("Ключ", СсылкаНаЭлементСправочника);
```

```
ОткрытьФорму ("Справочник.Организации.ФормаОбъекта", ПараметрыФормы);
```

Открытие основной формы **новой** группы:

```
ПараметрыФормы = Новый Структура ("ЭтоГруппа", Истина);
```

```
ОткрытьФорму ("Справочник.Товары.ФормаГруппы", ПараметрыФормы);
```

Открыть форму, не назначенную основной для справочника Товары (произвольную форму), можно следующим оператором:

```
ОткрытьФорму ("Справочник.Товары.Форма.ПроизвольнаяФормаТовара");
```

Чтобы открыть форму существующего элемента, нужно указать, конкретный элемент. Для этого также используются параметры формы.

У расширения справочника есть параметр **Ключ**. Если в этот параметр передать ссылку на существующий элемент справочника, то будет открыта форма именно этого элемента, а не нового.

Открытие основной формы существующего объекта:

```
СсылкаНаЭлементСправочника = Элементы.Список.ТекущаяСтрока;  
ПараметрыФормы = Новый Структура("Ключ", СсылкаНаЭлементСправочника);  
ОткрытьФорму("Справочник.Организации.ФормаОбъекта", ПараметрыФормы);
```

Одна из типичных задач при работе с формами списков – это открытие формы с одновременным позиционированием на конкретном элементе списка.

Для решения такой задачи можно использовать параметр **ТекущаяСтрока**

Открытие формы списка с позиционированием на некотором элементе:

```
ПараметрыФормы = Новый Структура("ТекущаяСтрока", Объект.Ссылка);  
ОткрытьФорму("Справочник.Товары.ФормаСписка", ПараметрыФормы);
```

Другая типичная задача – открытие списка подчиненного справочника. Для ее реализации можно использовать параметр **Отбор**.

В этот параметр нужно передать условие на отбор элементов подчиненного справочника по некоторому владельцу.

Например, это может выглядеть следующим образом. Справочник **РасчетныеСчета** подчинен справочнику **Организации**. В форме элемента справочника **Организации** (ссылка на этот элемент доступна как **Объект.Ссылка**) можно выполнить следующий код.

Открытие формы подчиненного справочника с отбором по владельцу:

```
УсловияОтбора = Новый Структура("Владелец", Объект.Ссылка)  
ПараметрыФормы = Новый Структура("Отбор", УсловияОтбора);  
ОткрытьФорму("Справочник.РасчетныеСчета.ФормаСписка",  
ПараметрыФормы);
```

Кроме метода **ОткрытьФорму()** существует и другой метод глобального контекста, который может быть использован для открытия форм, – **ПолучитьФорму()**.

По составу параметров этот метод очень похож на **ОткрытьФорму()**. Разница заключается в том, что **ПолучитьФорму()** лишь возвращает форму, не открывая ее. В дальнейшем эта форма может быть открыта с помощью своего метода **Открыть()**.

```
Перем СсылкаНаТаблицуСтилей;  
ФормаРедактора =  
ПолучитьФорму("Обработка.УниверсальныйРедактор.Форма");
```

```

ФормаРедактора.РедактироватьHTMLТекст ("Пример HTML-текста",
СсылкаНаТаблицуСтилей);
ФормаРедактора.Открыть ();

```

2. Модификация формы

Существует возможность программной модификации формы. Для модификации (создания, изменения и удаления) доступны:

- 1) Реквизиты формы;
- 2) Локальные команды;
- 3) Элементы формы.

Удалять можно только те объекты, которые созданы программно.

Работа по добавлению, изменению и удалению составных частей формы возможна только на **сервере**.

Данную схему программной работы с формой можно представить следующим образом:

- изменение состава реквизитов формы;
- изменение состава команд формы;
- изменение состава элементов формы.

2.1.1. Изменение состава реквизитов. Изменение (добавление и удаление) реквизитов выполняется методом **ИзменитьРеквизиты ()**.

Рассмотрим подробнее методику создания реквизитов.

Вначале необходимо создать нужное количество объектов типа РеквизитФормы. При создании реквизита указываем его имя, тип реквизита и то место в иерархии реквизитов формы, которое будет занимать создаваемый реквизит.

Для создания таблицы значений из двух колонок программный код будет выглядеть следующим образом:

```

МоиРеквизиты = Новый Массив;
МоиРеквизиты.Добавить (Новый РеквизитФормы ("ТаблицаДанных",
    Новый ОписаниеТипов ("ТаблицаЗначений"), "Таблица значений", Ложь));
МоиРеквизиты.Добавить (Новый РеквизитФормы ("Поставщик",
    Новый ОписаниеТипов ("СправочникСсылка.Контрагенты"), "ТаблицаДанных",
    "Кто поставляет", Ложь));
МоиРеквизиты.Добавить (Новый РеквизитФормы ("Товар", Новый
    ОписаниеТипов ("СправочникСсылка.Товары"), "ТаблицаДанных", "Имя
    товара", Ложь) );

```

Аналог программного кода:

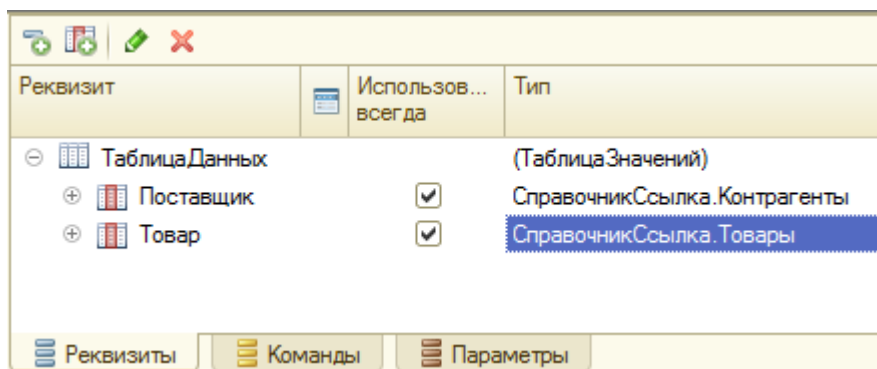


Рисунок 1.12 – Аналог программного кода

После того как были созданы все реквизиты, которые планировалось создать в форме, следует выполнить изменение списка реквизитов:

ИзменитьРеквизиты (ДобавляемыеРеквизиты, УдаляемыеРеквизиты) ;

Выполнение данного кода приведет к тому, что система вначале удалит реквизиты, перечисленные в массиве УдаляемыеРеквизиты, а затем выполнит добавление реквизитов из массива ДобавляемыеРеквизиты. После чего произойдет перестроение формы.

После добавления реквизита обращение к нему в программном коде возможно только с помощью конструкции ЭтаФорма.ИмяРеквизита. Выражение «ЭтаФорма.» является обязательным для программно добавляемых реквизитов.

Для того чтобы изменить состав или свойства реквизитов, следует вначале получить изменяемые реквизиты. Сделать это можно с помощью метода ПолучитьРеквизиты().

2.2.2. Изменение состава команд. Для того чтобы управлять составом команд формы существует специальная коллекция Команды. С помощью этой коллекции можно добавлять, удалять и изменять команды формы.

Так, для добавления команды КомандаУстановкиСтатуса с заголовком Установить статус, которая будет вызывать обработчик с именем ОбработчикПрограммныхКоманд, необходимо выполнить следующий программный код:

```
&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
Команда = Команды.Добавить ("КомандаУстановкиСтатуса") ;
Команда.Действие = "ОбработчикПрограммныхКоманд";
Команда.Заголовок = "Установить статус";
КонецПроцедуры
```

```
&НаКлиенте
Процедура ОбработчикПрограммныхКоманд(Команда)
КонецПроцедуры
```

Обработчик команды должен существовать в модуле формы и предваряться директивой компиляции &НаКлиенте.

2.2.3. Работа с элементами формы. После создания реквизитов и команд можно добавлять элементы управления.

Для управления элементами формы существует коллекция **Элементы**, с помощью которой можно добавлять, удалять, изменять свойства элементов формы, а также перемещать элементы формы между родителями.

Коллекция **Элементы** предоставляет доступ к списку элементов формы, который не учитывает возможную иерархию элементов. Для работы с иерархией у объекта коллекции **Элементы** существуют свойства *Родитель* и *ПодчиненныеЭлементы*.

Первое свойство указывает на родительский элемент формы, например, для поля формы, расположенного в группе, свойство *Родитель* будет указывать на элемент формы типа *ГруппаФормы*.

Свойство *ПодчиненныеЭлементы* существует у тех элементов формы, которые могут иметь подчиненные элементы. Например, для элемента формы типа *ГруппаФормы* коллекция *ПодчиненныеЭлементы* будет содержать те элементы, которые расположены в этой группе.

Для перемещения элемента из одной коллекции в другую (например, из одной группы в другую) существует метод **Переместить ()**. Параметры данного метода описывают перемещаемый элемент, новый родитель элемента и тот элемент формы, перед которым нужно поместить перемещаемый элемент.

Если последний параметр метода не указан, то перемещаемый элемент будет размещен в конце коллекции элементов нового родителя.

В примере показано, как разместить на форме два элемента:

- **поле** формы, связанное с реквизитом формы;
- **кнопку**, связанную с командой формы.

&НаСервере

```
Процедура ПриСозданииНаСервере (Отказ, СтандартнаяОбработка)
МоиРеквизиты = Новый Массив;
```

```
ТипСтрока = Новый ОписаниеТипов ("Строка", , Новый
КвалификаторыСтроки ());
МоиРеквизиты.Добавить (Новый РеквизитФормы ("ОписаниеОбъекта",
ТипСтрока, "" , "Описание объекта", Ложь));
ИзменитьРеквизиты (МоиРеквизиты);
```

```
Команда = Команды.Добавить ("ИзменитьСтроку");
Команда.Действие = "ОбработчикПрограммныхКоманд";
Команда.Заголовок = "Изменить строку";
```

```
Элемент = Элементы.Добавить ("ОписаниеОбъекта", Тип ("ПолеФормы")) ;
Элемент.Вид = ВидПоляФормы.ПолеВвода;
Элемент.ПутьКДанным = "ОписаниеОбъекта";
```

```
Элемент = Элементы.Добавить ("ИзменитьСтроку", Тип ("КнопкаФормы"));
Элемент.ИмяКоманды = "ИзменитьСтроку";
КонецПроцедуры
```

...

&НаКлиенте

```
Процедура ОбработчикПрограммныхКоманд (Команда)
ЭтаФорма.ОписаниеОбъекта = "Описание объекта, сформированное
командой";
```

```
КонецПроцедуры
```

1.3. ПРИМЕРЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

Задание: в справочнике «Товары» необходимо сохранять и отображать в форме фотографию товара. Исходная конфигурация не содержит ничего, кроме справочника с товарами.

Для хранения самого графического файла, в справочнике необходимо создать реквизит «ДанныеКартинки», с типом «Хранилище значений».

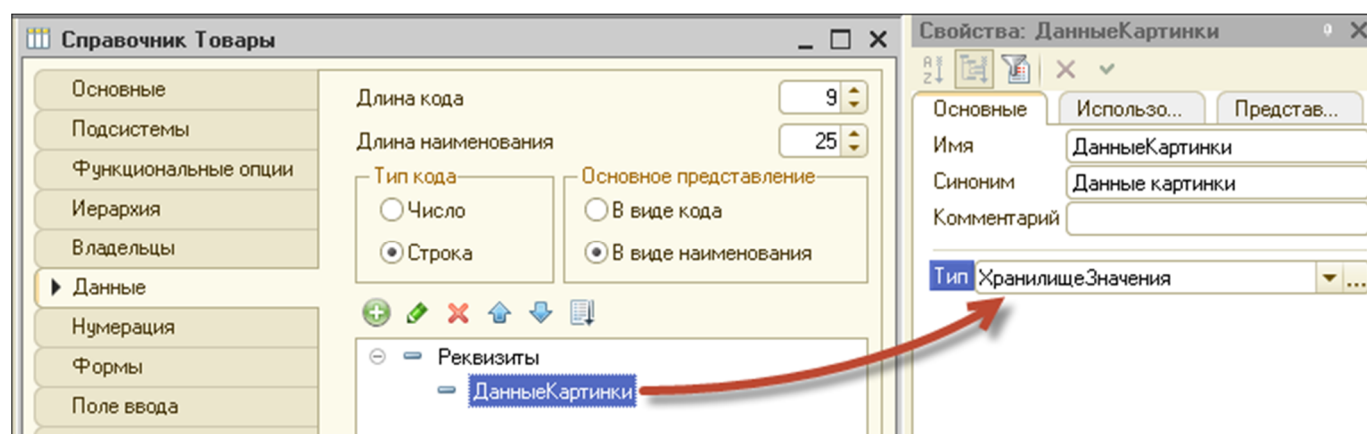


Рисунок 1.13 – Создание «ДанныеКартинки»

Как видно на рисунке, реквизит с типом хранилище значений в форме недоступен.

Добавьте реквизит формы с именем «СсылкаНаКартинку», тип «строка», длина не ограничена. И поместите его в реквизиты формы.

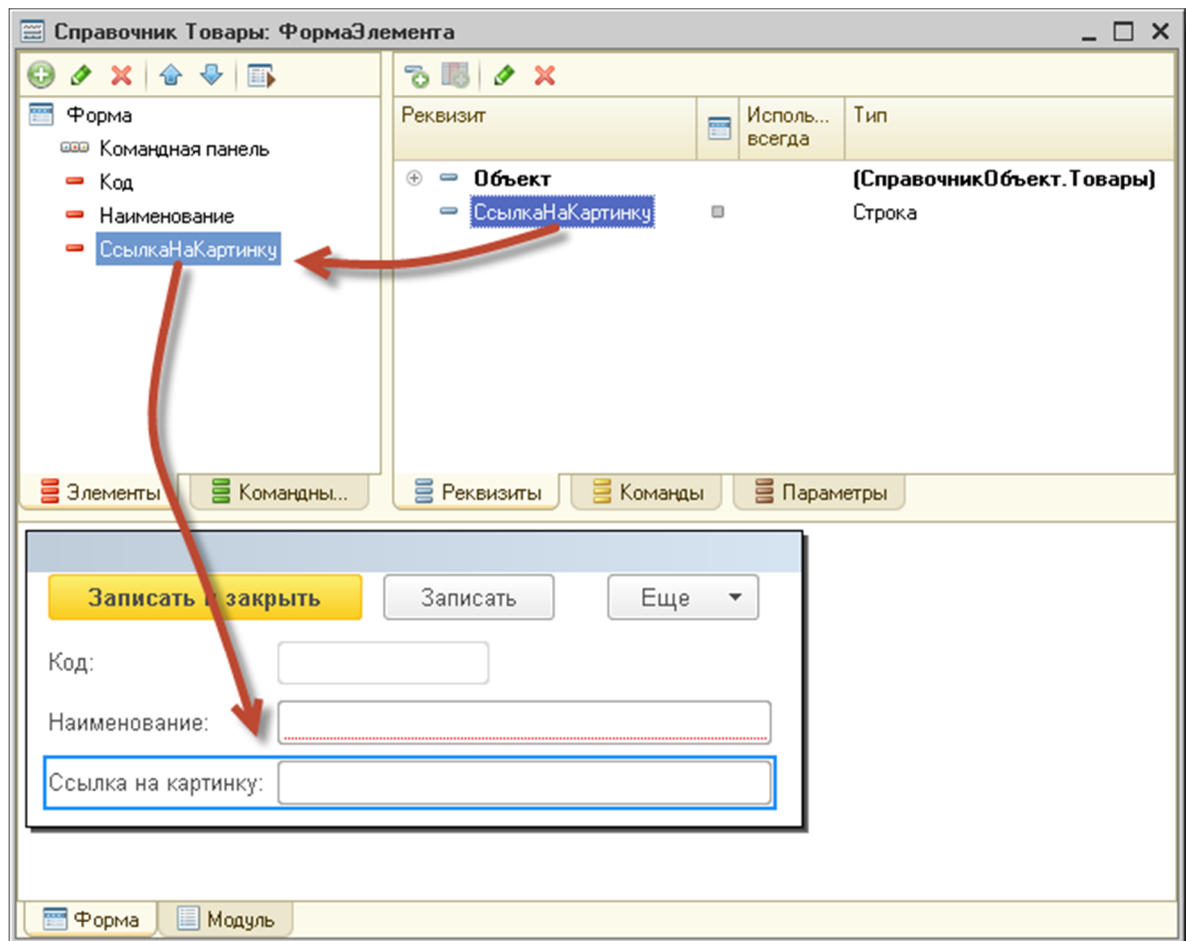


Рисунок 1.14 – Добавление реквизита «СсылкаНаКартинку»

По умолчанию, строка в форме будет выглядеть как поле ввода, нужно поменять это в свойствах элемента формы. Выберите в свойстве «Вид» «Поле картинки».

Создайте команду формы «ВыбратьФайлКартинки», в виде кнопки расположите ее на форме.

В модуле опишите обработчик события «Действие» для созданной команды формы:

```

&НаКлиенте
□ Процедура ВыбратьФайлКартинки(Команда)
|
| //Создаем оповещение, именно процедура "ОбработатьВыборФайла" будет вызвана при закрытии окна выбора файла
| Оповещение = Новый ОписаниеОповещения("ОбработатьВыборФайла", ЭтотОбъект);
|
| //Открываем интерактивно окно для выбора файла
| НачатьПомещениеФайла(Оповещение, , , Истина, УникальныйИдентификатор);
|
| КонецПроцедуры
  
```

Метод НачатьПомещениеФайла помещает выбранный файл во временное хранилище (если выбор файла был произведен) и вызывает процедуру «ОбработатьВыборФайла» в этом же модуле.

Свойство «УникальныйИдентификатор» связывает временное хранилище с текущей формой. В противном случае файл в хранилище помещен будет, но само хранилище при серверном вызове будет уничтожено, а оно еще понадобится для записи файла в реквизит справочника.

«НаКлиенте

Процедура ОбработатьВыборФайла(Результат, Адрес, ВыбранноеИмяФайла, ДополнительныеПараметры) Экспорт

Если Не Результат Тогда

Возврат;

КонецЕсли;

СсылкаНаКартинку = Адрес;

КонецПроцедуры

Если пользователь не отказался от выбора файла, то в реквизит формы «СсылкаНаКартинку» помещается адрес во временном хранилище, куда были помещены данные картинки.

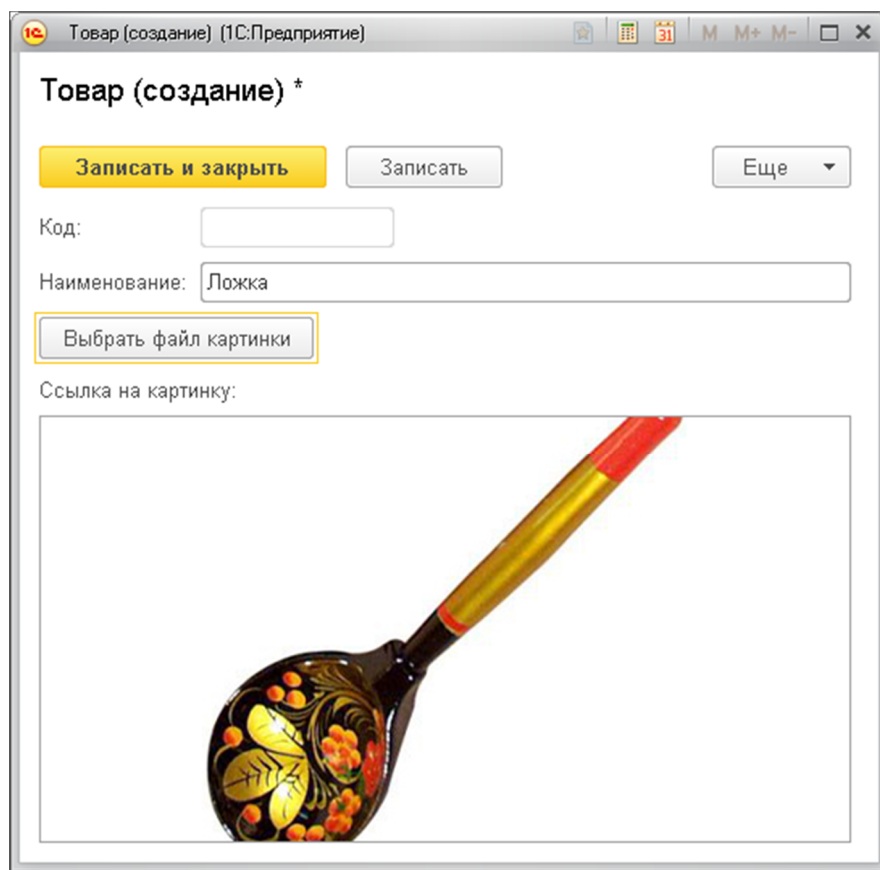


Рисунок 1.15 – Создание товара с картинкой

Событие ПередЗаписьюНаСервере:


```

&НаСервере
□ Процедура ПередЗаписьюНаСервере(Отказ, ТекущийОбъект, ПараметрыЗаписи)
    Если ЭтоАдресВременногоХранилища(СсылкаНаКартинку) Тогда
        ТекущийОбъект.ДанныеКартинки = Новый ХранилищеЗначения( ПолучитьИзВременногоХранилища(СсылкаНаКартинку) );
    КонецЕсли;
КонецПроцедуры

```

Осталось описать открытие формы, для отображения картинки уже сохраненной в реквизите элемента справочника.

```

&НаСервере
□ Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)
    СсылкаНаКартинку = ПолучитьНавигационнуюСсылку(Объект.Ссылка, "ДанныеКартинки");
КонецПроцедуры

```

1.4. ПРОГРАММА ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомится с теоретическим материалом в методических указаниях.
2. Построить интерфейс для всех объектов из предыдущих лабораторных работ.
3. Сделать задание на доработку формы по индивидуальному заданию преподавателя.
4. Сформулировать выводы.
5. Оформить отчет.

Выполнение задания подразумевает продолжение ЛР №1, 3 и 4 (см.Приложение А).

1.5. СОДЕРЖАНИЕ ОТЧЕТА

1. Цель работы.
2. Описание процесса разработки.
3. Демонстрация результатов работы в режиме «1С:Предприятие».
4. Выводы.

1.6. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое модуль формы?
2. Какие формы бывают?
3. Какие элементы формы вы знаете?
4. Как добавить кнопку на форму?
5. Как привязать обработчик события к кнопке?

2. «ИССЛЕДОВАНИЕ ЯЗЫКА ЗАПРОСОВ В СИСТЕМЕ 1С:ПРЕДПРИЯТИЕ 8»

2.1. ЦЕЛЬ РАБОТЫ

Целью данной лабораторной работы является исследование языка запросов в системе 1С: Предприятие 8.

2.2. ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Язык запросов в 1С 8 – это упрощенный аналог широкоизвестного «структурированного языка программирования» (SQL). Но в 1С он используется только для чтения данных, для изменения данных используется объектная модель данных.

Отличие языка запросов 1С – русский синтаксис (можно использовать и англоязычные конструкции).

Можно перечислить наиболее существенные возможности, реализуемые языком запросов:

1) *Обращение к подчиненным полям через точку.* Если поля какой-либо таблицы имеют ссылочный тип (хранят ссылки на объекты другой таблицы), разработчик может в тексте запроса ссылаться на них через точку («.»). При этом количество уровней вложенности таких ссылок система не ограничивает;

2) *Обращение к вложенным таблицам (например, табличным частям документов и элементов справочников).* К вложенным табличным частям можно обращаться и как к отдельным таблицам, и как к целым полям одной таблицы;

3) *Автоматическое упорядочивание.* Режим автоматического упорядочивания позволяет выводить информацию в наиболее правильном («естественном») порядке;

4) *Многомерное и многоуровневое формирование итогов.* Итоги и подытоги формируются с учетом группировки и иерархии, обход уровней может выполняться в произвольном порядке с подведением подытогов. Обеспечивается корректное построение итогов по временным измерениям;

5) *Поддержка виртуальных таблиц.* Виртуальные таблицы, предоставляемые системой, позволяют получить практически готовые данные для большинства прикладных решений без необходимости составления сложных запросов. Например, такая виртуальная таблица может предоставить данные по остаткам товаров в разрезе периодов на какой-то момент времени. При этом виртуальные таблицы максимально используют хранимую информацию, например, ранее рассчитанные итоги и т. д.;

6) *Стандартные SQL операции.* В языке запросов поддерживаются стандартные для SQL операции, такие как объединение (Union), соединение (Join) и т. д.;

7) *Использование временных таблиц.* Временные таблицы позволяют повысить производительность запросов, в некоторых случаях снизить количество блокировок и сделать текст запроса более легким для восприятия;

8) *Пакетные запросы.* Пакетные запросы позволяют более удобно работать с временными таблицами за счет того, что создание временной таблицы и ее использование помещаются в один запрос. Пакетный запрос представляет собой последовательность запросов, разделенных символом «;». Запросы исполняются один за другим. Результатом выполнения пакетного запроса в зависимости от используемого метода будет являться либо результат, возвращаемый последним запросом пакета, либо массив результатов всех запросов пакета в той последовательности, в которой следуют запросы в пакете.

2.2.1. Описание конструкций языка запросов

Текст запроса можно описать следующим правилом:

<Текст Запроса>

<Описание запроса>

[<Объединение запросов>]

[<Упорядочивание результатов>]

[Автоупорядочивание]

[< Описание итогов>]

Текст запроса состоит из нескольких частей, или секций:

1. <Описание запроса> – это единственная обязательная секция в тексте запроса, и во многих случаях достаточно указать только ее. В секции определяются источники данных запроса, поля выборки, группировки и т.д.

2. <Объединение запросов> – язык запросов позволяет объединять результаты выполнения нескольких запросов.

3. В секции <Упорядочивание результатов> можно определить условия упорядочивания строк в результате запроса.

4. Секция <Автоупорядочивание> позволяет включить режим автоматического упорядочивания строк в результате запроса.

5. В секции <Описание итогов> можно указать, какие итоги необходимо рассчитывать в запросе.

В тексте запроса должна обязательно присутствовать секция описания запроса, в которой определяются:

- поля, которые будут содержаться в результате запроса;
- источники данных запроса - исходные таблицы;
- условия, влияющие на выборку данных в запросе;
- порядок группировки результатов запроса.

Секция описания запроса состоит из нескольких взаимосвязанных предложений:<Описание запроса>

ВЫБРАТЬ [РАЗРЕШЕННЫЕ] [РАЗЛИЧНЫЕ] [ПЕРВЫЕ <Количество>]
 <Список полей выборки>
 [ИЗ <Список источников>]
 [ГДЕ <Условие отбора>]
 [СГРУППИРОВАТЬ ПО <Поля группировки>]
 [ИМЕЮЩИЕ <Условие отбора>]
 [ДЛЯ ИЗМЕНЕНИЯ [<Список таблиц верхнего уровня>]]

Описание запроса начинается с обязательного ключевого слова **ВЫБРАТЬ**.

Ключевое слово **РАЗРЕШЕННЫЕ** означает, что запрос выберет только те записи, на которые у текущего пользователя есть права. Если данное слово не указать, то запрос отработает с ошибкой, когда он выберет записи, на которые у пользователя нет прав. Данное ключевое слово может присутствовать только в предложении **ВЫБРАТЬ** верхнего уровня и распространяется на весь запрос, включая вложенные запросы.

С помощью ключевого слова **РАЗЛИЧНЫЕ** можно указать, что в результат не должны попадать повторяющиеся строки.

Конструкция **ПЕРВЫЕ** <Количество> позволяет задать предельное количество строк в результате запроса. Будут отобраны самые первые (в соответствии с правилами упорядочивания результатов запроса) строки. Количество задается целым числом. В языке запросов добавлена возможность исполнения упорядочивания во вложенных запросах в случае, если вложенный запрос содержит конструкцию **ПЕРВЫЕ**.

В секции <Список полей **Выборки**> описываются поля, которые должны содержаться в результате запроса.

В предложении **ИЗ** <Список источников> указываются источники данных – таблицы информационной базы, содержимое которых обрабатывается в запросе. Описание источников может быть опущено только в том случае, если они полностью определены в списке полей выборки.

Предложение **ГДЕ** <Условие отбора> позволяет отфильтровать результат запроса. В результат попадают только те записи, для которых указанное условие оказывается истинным.

Предложение **ДЛЯ ИЗМЕНЕНИЯ** предназначено для указания необходимости блокировки считываемых в транзакции данных. Считанные данные становятся недоступными для чтения в других сессиях. Для файлового варианта блокируются указанные таблицы, а для клиент-серверного варианта – только выбранные записи. Блокировка снимается после завершения транзакции.

Предложение **СГРУППИРОВАТЬ** позволяет описать порядок группировки результатов запроса.

Предложение **ИМЕЮЩИЕ** позволяет накладывать условия на результаты группировки.

2.2.2. Описание и установка внешней обработки «Консоль запросов» для управляемого приложения

Инструмент «Консоль запросов» предназначена для отладки и просмотра результатов выполнения запросов в режиме **1С:Предприятие**, не используя Конфигуратор. Данная обработка предназначена в основном для разработчиков конфигураций и специалистов по внедрению.

Для установки внешней обработки необходимо открыть 1С в режиме «Конфигуратор», открыть Конфигурацию → Обработки → Вставить внешнюю обработку (выбрав путь к файлу обработки).

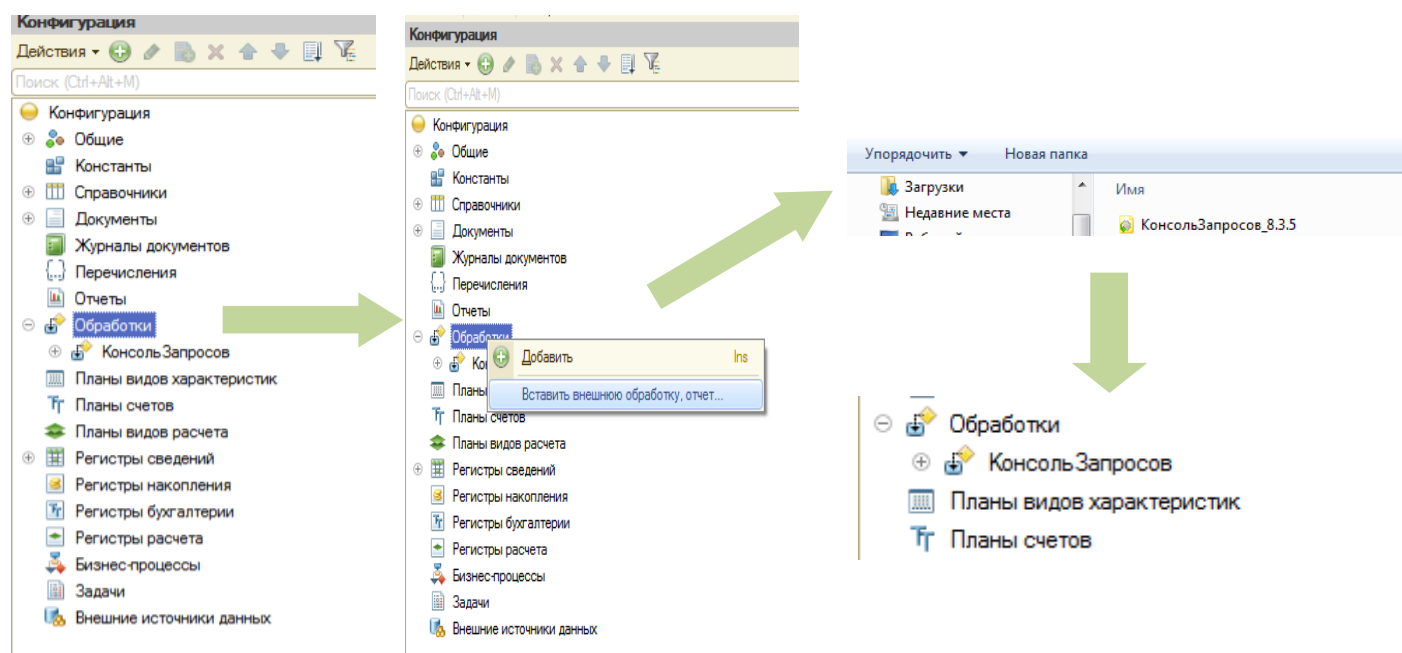


Рисунок 2.1 – Установка внешней обработки

Для работы Консоль запросов запускают в режиме 1С:Предприятие.

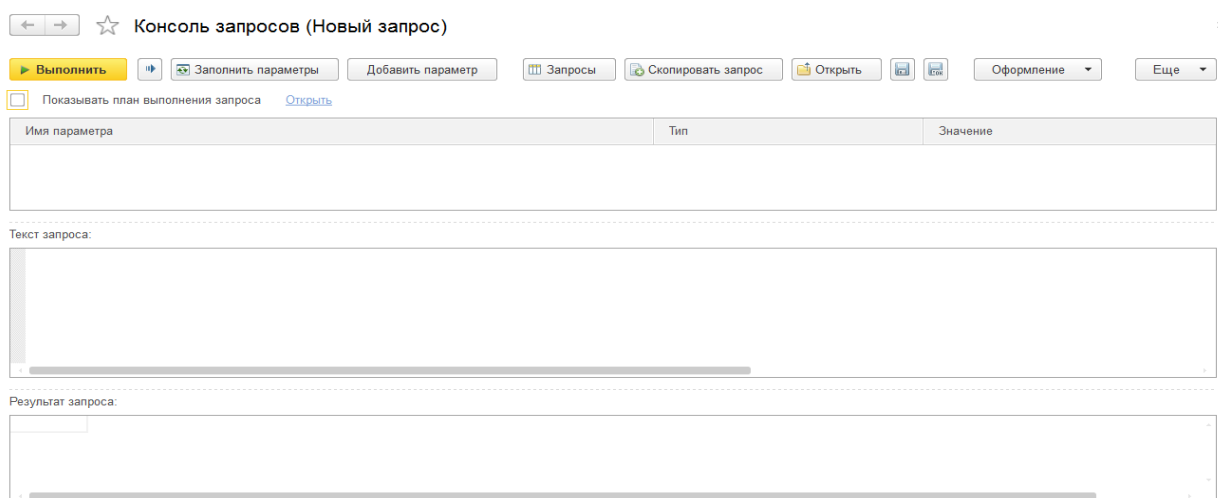


Рисунок 2.2 – Консоль запросов

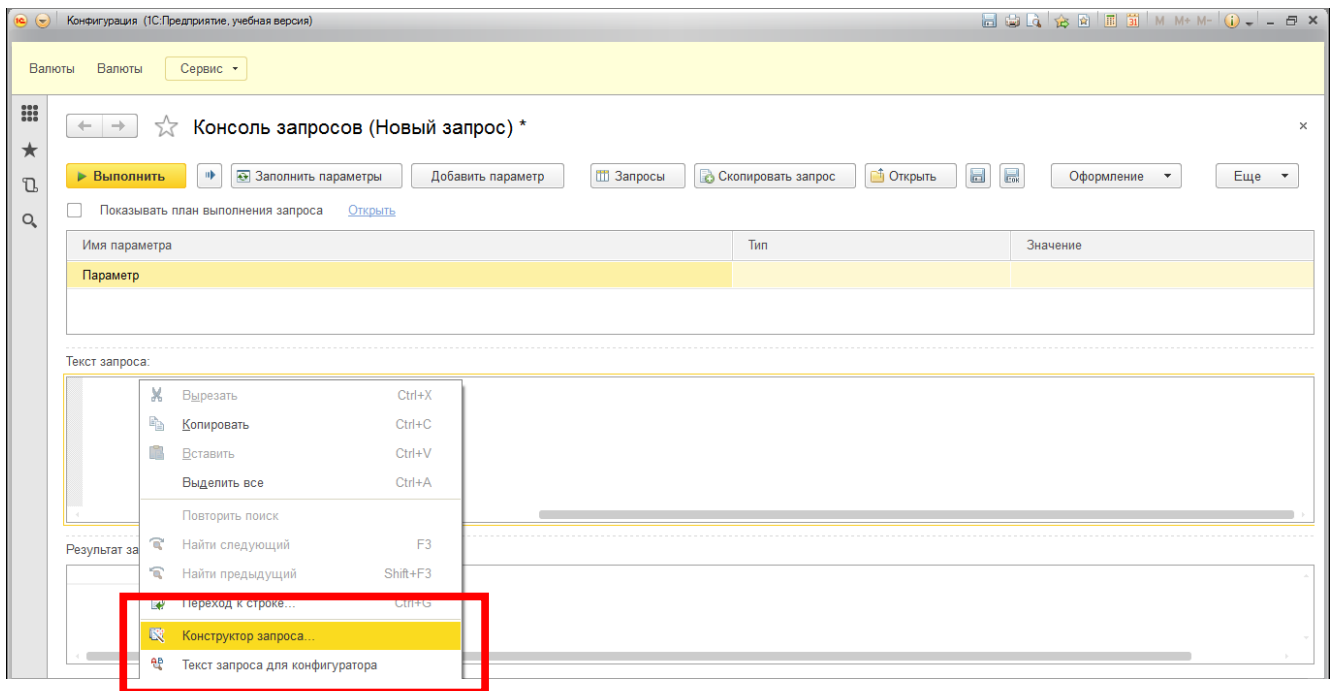


Рисунок 2.3 – Выбор конструктора запроса

2.2.3. Примеры использования функций и конструкций языка запросов

1. Выбор всех строк со значениями всех столбцов из таблицы

Процедура ЗапросВсехСтрокСоЗначениямиВсехСтолбцовТаблицы ()

```
Запрос = Новый Запрос (
    "
        | ВЫБРАТЬ
        |     *
        | ИЗ
        |     Справочник.Номенклатура
        | "
);
```

Результат = Запрос.Выполнить () ;

ОткрытьЗначение (Результат.Выгрузить (ОбходРезультатаЗапроса.Прямой)) ;

КонецПроцедуры

2. Выбор всех строк со значениями определенных столбцов

Процедура ЗапросВсехСтрокСоЗначениямиОпределенныхСтолбцовТаблицы ()

```
Запрос = Новый Запрос (
    "
        | ВЫБРАТЬ
        |     Код, Наименование, Родитель
        | ИЗ
```

```

        | Справочник.Номенклатура
        | "
    );

```

```

    Результат = Запрос.Выполнить();
    ОткрытьЗначение(Результат.Выгрузить(ОбходРезультатаЗапроса.Прямой)
    );
    КонецПроцедуры

```

3. Использование в запросах псевдонимов для столбцов и таблиц

```

Процедура ИспользованиеПсевдонимов()
    Запрос = Новый Запрос(
        "
        | ВЫБРАТЬ
        |     Товары.Наименование КАК Имя, Товары.Родитель КАК
Группа
        | ИЗ
        |     Справочник.Номенклатура КАК Товары
        | "
    );

```

```

    Результат = Запрос.Выполнить();
    ОткрытьЗначение(Результат.Выгрузить(ОбходРезультатаЗапроса.Прямой)
    );
    КонецПроцедуры

```

4. Добавление в результат запроса столбца с определенным значением

```

Процедура
ДобавлениеВРезультатЗапросаСтолбцаСОпределеннымЗначением()
    Запрос = Новый Запрос(
        "
        | ВЫБРАТЬ
        |     Наименование, 10000 КАК Цена, ""Оранжевый"" КАК Цвет
        | ИЗ
        |     Справочник.Номенклатура
        | "
    );

```

```

    Результат = Запрос.Выполнить();
    ОткрытьЗначение(Результат.Выгрузить(ОбходРезультатаЗапроса.Прямой)
    );
    КонецПроцедуры

```

5. Выбор неповторяющихся (различных) строк из таблицы

```

Процедура ЗапросРазличныхСтрокИзТаблицы()
    Запрос = Новый Запрос(

```



```

"
| ВЫБРАТЬ РАЗЛИЧНЫЕ
|   Родитель
| ИЗ
|   Справочник.Номенклатура
|"
);

```

```

Результат = Запрос.Выполнить();
ОткрытьЗначение(Результат.Выгрузить(ОбходРезультатаЗапроса.Прямой)
);
КонецПроцедуры

```

6. Выбор первых n строк из таблицы

```

Процедура ЗапросПервыхNСтрокИзТаблицы()
    Запрос = Новый Запрос (
        "
        | ВЫБРАТЬ ПЕРВЫЕ 10
        |   Наименование
        | ИЗ
        |   Справочник.Номенклатура
        |"
    );

```

```

    Результат = Запрос.Выполнить();
    ОткрытьЗначение(Результат.Выгрузить(ОбходРезультатаЗапроса.Прямой)
    );
    КонецПроцедуры

```

7. Выбор строк из таблицы в соответствии с правами пользователя

```

Процедура ВыборкаРазрешенныхСтрокИзТаблицы()
    Запрос = Новый Запрос (
        "
        | ВЫБРАТЬ РАЗРЕШЕННЫЕ
        |   *
        | ИЗ
        |   Справочник.Номенклатура
        |"
    );

```

```

    Результат = Запрос.Выполнить();
    ОткрытьЗначение(Результат.Выгрузить(ОбходРезультатаЗапроса.Прямой)
    );
    КонецПроцедуры

```

8. Использование функции ДАТАВРЕМЯ – данная функция создает константное поле с типом "Дата".

Синтаксис:

ДАТАВРЕМЯ (<Год>, <Месяц>, <День>, <Час>, <Минута>, <Секунда>)

Пример использования:

```
Запрос.Текст =
    "ВЫБРАТЬ
    |     ДАТАВРЕМЯ(2015, 3, 1) КАК ПервоеМарта"; // 1 марта 2015
```

9. *Использование функции РАЗНОСТЬДАТ* – возвращает разность двух дат в одном из измерений (год, месяц, день, час, минута, секунда). Измерение передается в параметре.

Синтаксис:

РАЗНОСТЬДАТ (<Дата1>, <Дата2>, <Тип>)

Пример использования:

```
Запрос.Текст =
    "ВЫБРАТЬ
    |     РАЗНОСТЬДАТ(ДАТАВРЕМЯ(2015, 4, 17), ДАТАВРЕМЯ(2015, 2,
1), ДЕНЬ)
    |     КАК КолвоДней";
```

10. *Использование функции ЗНАЧЕНИЕ* – задает константное поле с предопределенной записью из базы данных, также можно получить пустую ссылку любого типа.

Синтаксис:

ЗНАЧЕНИЕ (<Имя>)

Пример использования:

```
Запрос.Текст =
    "ВЫБРАТЬ
    //предопределенный элемент
    |     ЗНАЧЕНИЕ(Справочник.Валюты.Доллар) КАК Доллар,
    //пустая ссылка
    |     ЗНАЧЕНИЕ(Документ.ПоступлениеТоваровУслуг.ПустаяСсылка)
КАК Поступление,
    //знач. перечисления
    |     ЗНАЧЕНИЕ(Перечисление.ЮрФизЛицо.ФизЛицо) КАК ФизЛицо,
    //предопределенный счет
    |     ЗНАЧЕНИЕ(ПланСчетов.Хозрасчетный.Материалы) КАК
Счет_10";
```

2.3. ПРИМЕРЫ ВЫПОЛНЕНИЯ

Пример 1. Составить список всех лиц, работающих в фирмах-партнерах (лиц, зафиксированных в табличной части КонтактныеЛица). В таком списке напротив фамилии каждого сотрудника должен быть указан номер его телефона. Также информацию о сотруднике желательно дополнить еще и названием организации, где он работает. Список следует отсортировать по фамилиям сотрудников (по алфавиту в возрастающем порядке).

Текст запроса

ВЫБРАТЬ Сотрудник, Телефон,
Ссылка.Наименование КАК Организация
ИЗ Справочник.Фирмы.КонтактныеЛица
УПОРЯДОЧИТЬ ПО Сотрудник

В приведенном запросе использовано ключевое слово ВЫБРАТЬ, позволяющее указать список полей для данного запроса. Другое ключевое слово ИЗ дает возможность определить таблицы, участвующие в рассматриваемом запросе. В данной ситуации используется табличная часть КонтактныеЛица справочника Фирмы.

При запросе к табличной части справочника обращение к обычным (вне таблицы) реквизитам справочника производится через поле Ссылка. В данном случае информация о фирме заключена в автоматически создаваемом в любом справочнике реквизите Наименование, который располагается вне табличной части.

В приведенном запросе было использовано ключевое слово КАК, позволяющее определить синоним для сочетания Ссылка.Наименование.

Для упорядочивания данных запроса используется конструкция УПОРЯДОЧИТЬ ПО, в которой необходимо указать название поля (или полей), по которому будет производиться сортировка.

Пример 2. Упорядочить данные из предыдущего примера сначала по названиям организаций, а затем по именам сотрудников.

Текст запроса

ВЫБРАТЬ Сотрудник, Телефон,
Ссылка.Наименование КАК Организация
ИЗ Справочник.Фирмы.КонтактныеЛица
УПОРЯДОЧИТЬ ПО Организация ВОЗР, Сотрудник ВОЗР

Пример 3. Получить информацию о движении товаров из регистра сведений Количество Товаров по филиалам организации в следующем формате: Товар, Филиал, Регистратор (в регистре накопления реквизит под названием

регистратор представляет собой ссылку на документ, которому подчинена рассматриваемая запись регистра), *Количество*.

Текст запроса

```
ВЫБРАТЬ
КоличествоТоваров.Товар КАК Товар,
КоличествоТоваров.Филиал.Наименование КАК Филиал,
КоличествоТоваров.Регистратор КАК Регистратор,
КоличествоТоваров.Количество КАК Количество
ИЗ
РегистрНакопления.КоличествоТоваров КАК КоличествоТоваров
УПОРЯДОЧИТЬ ПО Филиал, Количество УБЫВ
```

Пример 4. Получить позиции в документах ПоступлениеТоваров (строки табличной части) с тремя минимальными (максимальными) суммами поступления.

Текст запроса (выбор трех минимальных сумм при поступлении товаров)

```
ВЫБРАТЬ ПЕРВЫЕ 3 Товар, Сумма,
Ссылка.Фирма КАК ФирмаПоставщик
ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров
УПОРЯДОЧИТЬ ПО Сумма ВОЗР
```

Текст запроса (выбор трех максимальных сумм поступления товаров)

```
ВЫБРАТЬ ПЕРВЫЕ 3 Товар, Сумма,
Ссылка.Фирма КАК ФирмаПоставщик
ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров
УПОРЯДОЧИТЬ ПО Сумма УБЫВ
```

Пример 5. Произвести выборку товаров из документов ПоступлениеТоваров вместе с названиями фирм без повторений данных комбинаций

Текст запроса

```
ВЫБРАТЬ РАЗЛИЧНЫЕ Товар.Наименование КАК Товар,
Ссылка.Фирма.Наименование КАК ФирмаПоставщик
ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
УПОРЯДОЧИТЬ ПО Товар ВОЗР
```

Пример 6. Получить список поступлений товаров, которые встречались в документах ПоступлениеТоваров в количестве, большем пяти единиц, при условии их суммарной стоимости (произведение цены на количество) более 75 000 рублей.

Текст запроса

```
ВЫБРАТЬ Товар, Количество, Сумма,
```

Перечень.Ссылка.Филиал КАК Филиал
 ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
 ГДЕ Количество > 5 И Сумма > 75000

Пример 7. Получить список товаров, которые поступали либо по цене не менее 30 000 рублей в любом количестве, либо более 25 000 рублей в количестве, большем двух.

Текст запроса

ВЫБРАТЬ Товар, Количество, Цена,
 Перечень.Ссылка.Дата КАК ДатаДокумента
 ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
 ГДЕ (Количество > 2 И Цена > 25000) ИЛИ Цена >= 30000

Пример 8. Подсчитать число всех документов ПоступлениеТоваров в базе данных, а также число фирм, от которых были поступления товаров.

Текст запроса

ВЫБРАТЬ КОЛИЧЕСТВО(*) КАК Всего,
 КОЛИЧЕСТВО (РАЗЛИЧНЫЕ Поступление.Фирма) КАК ЧислоФирм
 ИЗ Документ.ПоступлениеТоваров КАК Поступление

Пример 9. Прочитать движения по регистру накопления КоличествоТоваров, используя ссылку на документ. Запрос извлекает название товара, филиала, а также количество единиц поступления (либо продажи или перемещения) товара

Текст запроса

ВЫБРАТЬ Товар, Филиал, Количество
 ИЗ РегистрНакопления.КоличествоТоваров КАК КоличествоТоваров
 ГДЕ КоличествоТоваров.Регистратор=&Регистратор

Пример 10. Получить с помощью запроса информацию о документах по поступлениям товаров в определенный филиал не позднее указанной даты.

Текст запроса

ВЫБРАТЬ Дата, Номер
 ИЗ Документ.ПоступлениеТоваров
 ГДЕ Дата <= &ГраницаДаты И Филиал=&УказанныйФилиал

Пример 11. Вывести только те записи регистра накопления КоличествоТоваров, которые соответствуют выбранному интервалу по дате.

Текст запроса

ВЫБРАТЬ Товар,
 Филиал, Количество, Регистратор
 ИЗ РегистрНакопления.КоличествоТоваров
 ГДЕ РегистрНакопления.КоличествоТоваров.Период
МЕЖДУ &Дата1 И &Дата2

Пример 12. Сформировать запрос к табличной части документа ПоступлениеТоваров, в котором используется конструкция **ИМЕЮЩИЕ**. Цель рассматриваемого запроса – отобрать товары, для которых итоговая сумма по поступлениям превышает 325 000.

Текст запроса

ВЫБРАТЬ Товар, СУММА(Сумма) КАК СуммаТоваров
 ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
 СГРУППИРОВАТЬ ПО Товар
ИМЕЮЩИЕ СУММА(Сумма) > 325000

Пример 13. Объединение запросов. Составить список товаров в привязке к фирме, которая выступала в роли либо поставщика, либо продавца.

Текст запроса (формирования списка товаров с указанием фирмы)

ВЫБРАТЬ
 Перечень.Ссылка.Фирма.Наименование КАК Фирма,
 Перечень.Товар
 ИЗ
 Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
ОБЪЕДИНИТЬ
 ВЫБРАТЬ
 Перечень.Ссылка.Фирма.Наименование,
 Перечень.Товар
 ИЗ
 Документ.ПродажаТоваров.ПереченьТоваров КАК Перечень
 УПОРЯДОЧИТЬ ПО Фирма

Пример 14. Используя ключевое слово **СОЕДИНЕНИЕ**, отобрать из документов ПоступлениеТоваров и ПродажаТоваров только совпадающие товары

Текст запроса (отбор совпадающих товаров)

ВЫБРАТЬ РАЗЛИЧНЫЕ ПереченьПоступлений.Товар КАК
 ПоступившийТовар,
 ПереченьПродаж.Товар КАК ПроданныйТовар

ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК
ПереченьПоступлений

СОЕДИНЕНИЕ

Документ.ПродажаТоваров.ПереченьТоваров КАК ПереченьПродаж
ПО ПереченьПоступлений.Товар = ПереченьПродаж.Товар

Пример 15. Отобрать филиалы, в которые поступали товары от фирм-партнеров и одновременно были перемещения с выставок.

Текст запроса (извлечение филиалов по условию поступления и перемещения товаров)

ВЫБРАТЬ РАЗЛИЧНЫЕ Поступление.Филиал КАК НашФилиал
ИЗ Документ.ПоступлениеТоваров КАК Поступление

СОЕДИНЕНИЕ

Документ.ПеремещениеСВыставки КАК Перемещение
ПО Поступление.Филиал=Перемещение.Филиал

Пример 16. Выбрать фамилии всех менеджеров с указанием общей суммы выполненных ими продаж, при этом в результирующую таблицу должны также попасть фамилии менеджеров, которые еще не оформляли документов по продаже товаров. Данный запрос реализуется с помощью конструкции **ЛЕВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ**. В этом случае (в отличие от внутреннего соединения) в полученном списке данных будут представлены также менеджеры, у которых еще не зарегистрировано продаж

Текст запроса

ВЫБРАТЬ НашиМенеджеры.Наименование КАК НашМенеджер,
Продажи.СуммаОборот КАК Сумма

ИЗ Справочник.Менеджеры КАК НашиМенеджеры

ЛЕВОЕ СОЕДИНЕНИЕ

РегистрНакопления.ПродажиПоФилиалам.Обороты КАК Продажи
ПО НашиМенеджеры.Наименование=Продажи.Менеджер.Наименование

Пример 17. Сформировать список менеджеров, аналогичного рассмотренному выше примеру, но полученного с использованием правого соединения таблиц. Дополнительно следует вывести информацию о филиалах, в которых были продажи.

Текст запроса

ВЫБРАТЬ НашиМенеджеры.Наименование КАК НашМенеджер,
Продажи.Филиал КАК НашФилиал,

Продажи.СуммаОборот КАК Сумма

ИЗ РегистрНакопления.ПродажиПоФилиалам.Обороты КАК Продажи

ПРАВОЕ СОЕДИНЕНИЕ

Справочник.Менеджеры КАК НашиМенеджеры

ПО НашиМенеджеры.Наименование = Продажи.Менеджер.Наименование

Пример 18. Составить общий список товаров с учетом их количества (просуммировать количество одинаковых товаров во всех филиалах). Для этого можно использовать информацию из виртуальной таблицы остатков регистра накопления:

РегистрНакопления.КоличествоТоваров.Остатки

Однако по замыслу задачи в указанный список необходимо включить также товары, по которым еще не было поступлений (это новые товары, зарегистрированные только в справочнике Товары). Фактически о подобных товарах в регистре КоличествоТоваров записей нет. Поэтому для извлечения информации воспользуемся еще одной таблицей – справочником Товары.

В данном запросе будет использоваться **правое соединение** таблиц.

Текст запроса (извлечение информации о наличии товаров)

```
ВЫБРАТЬ Товары.Ссылка КАК НазваниеТовара,
КоличествоТоваров.КоличествоОстаток КАК Количество
ИЗ РегистрНакопления.КоличествоТоваров.Остатки КАК
КоличествоТоваров
ПРАВОЕ СОЕДИНЕНИЕ
Справочник.Товары КАК Товары
ПО Товары.Ссылка=КоличествоТоваров.Товар.Ссылка
УПОРЯДОЧИТЬ ПО Количество УБЫВ
```

В сформированный список вошли как товары, которые уже поступали, так и товары, которые в документах по поступлению не фигурировали.

Пример 19. Отобрать фирмы, которые выступали в качестве как продавцов, так и покупателей. При этом необходимо в результат включить также фирмы, которые выступали только в одном качестве (в качестве только продавца или только покупателя).

Текст запроса (использование полного внешнего соединения таблиц)

```
ВЫБРАТЬ РАЗЛИЧНЫЕ Продажа.Фирма.Наименование КАК
ФирмаПокупатель,
Поступление.Фирма.Наименование КАК ФирмаПродавец
ИЗ Документ.ПродажаТоваров КАК Продажа
ПОЛНОЕ СОЕДИНЕНИЕ
Документ.ПоступлениеТоваров КАК Поступление
ПО Продажа.Фирма.Наименование=Поступление.Фирма.Наименование
```

Пример 20. Привести к строковому типу информацию о датах документов. Вывести даты из документов ПоступлениеТоваров и ПеремещениеСВыставки.

В рассматриваемом примере производится объединение двух запросов, которые выполняют извлечение дат документов ПоступлениеТоваров и извлечение дат документов ПеремещениеСВыставки.

Текст запроса (функция ПРЕДСТАВЛЕНИЕ)

```
ВЫБРАТЬ ПРЕДСТАВЛЕНИЕ (Поступление.Дата) КАК  
ДатыПоступленияПеремещения  
ИЗ Документ.ПоступлениеТоваров КАК Поступление  
ОБЪЕДИНИТЬ  
ВЫБРАТЬ ПРЕДСТАВЛЕНИЕ (Перемещение.Дата)  
ИЗ Документ.ПеремещениеСВыставки КАК Перемещение
```

2.4. ПРОГРАММА ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомится с теоретическим материалом в методических указаниях.
2. Выполнить перечень запросов по индивидуальному заданию преподавателя.
3. Сформулировать выводы.
4. Оформить отчет.

Выполнение задания подразумевает продолжение ЛР №1, 3 и 4 (см.Приложение А).

2.5. СОДЕРЖАНИЕ ОТЧЕТА

1. Цель работы.
2. Описание процесса выполнения.
3. Демонстрация результатов работы.
4. Выводы.

2.6. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что является источником данных в запросе?
2. Что такое левое соединение?
3. В чем отличие реальных и виртуальных таблиц.
4. Что такое псевдонимы в языке запросов?
5. Для чего предназначена схема компоновки данных.
6. Что является источником данных запроса.
7. В чем отличие между реальными и виртуальными таблицами.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. М.Г. Радченко, Е.Ю. Хрусталева «1С:Предприятие 8.3. Практическое пособие разработчика. Примеры и типовые приемы» (ISBN 978-5-9677-2041-3), М.: ООО «1С-Паблишинг», 2013. Электронный аналог печатного издания.
2. Ажеронок В. А., Габец А. П., Гончаров Д. И., Козырев Д. В., Кухлевский Д. С., Островерх А. В., Радченко М. Г., Хрусталева Е. Ю. Под редакцией М. Г. Радченко. "Профессиональная разработка в системе 1С:Предприятие 8" (второе издание). Том 1. М.: ООО «1С-Паблишинг», 2012. Электронная версия книги.
3. Управляемый интерфейс. Режим доступа:
<http://xn----1-bedvffifm4g.xn--plai/articles/%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9-%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81/>
4. Новый интерфейс платформы 1С 8.3. Режим доступа:
<http://programmist1s.ru/novyiy-interfeys-platformyi-1s-8-3/>
5. Переход на интерфейс Такси в 1С 8.3.
<http://programmist1s.ru/vklyuchenie-interfeysa-taksi-v-1s/>
6. Новый интерфейс Такси в 1С:Предприятие 8.3. Режим доступа:
<http://xn----1-bedvffifm4g.xn--plai/news/%D0%BD%D0%BE%D0%B2%D1%8B%D0%B9-%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81-%D1%82%D0%B0%D0%BA%D1%81%D0%B8/>
7. Секреты управляемого интерфейса. Режим доступа: <http://xn--80aaaahcnwic8afz6cdd5e1ic.xn--plai/index.php/stati/72-sekretiy-upravlyаемого-interfejsa>
8. Под капотом управляемых форм. Режим доступа:
<http://infostart.ru/public/198766/>
9. Основы запросов в языке 1С 8.3, 8.2 (в примерах). Режим доступа:
<http://helpme1c.ru/zaprosy-v-yazyke-1s-8-v-primerax>
10. Запросы 1С 8 от А до Я. Режим доступа: <http://programmist1s.ru/zaprosyi-1s-ot-a-do-ya/>
11. Особенности языка запросов 1С. Режим доступа:
<http://infostart.ru/public/204054/>
12. Как пользоваться консолью запросов 1С. Режим доступа:
<http://programmist1s.ru/konsol-zaprosov-1s-opisanie-i-skachat/>
13. Запросы 1С 8. Режим доступа: <http://zapros-1c-8.ru/9-yazik-zaprosov-1c-8/2-intro>
14. Язык запросов 1С 8.3 для начинающих программистов: основы синтаксиса. Режим доступа: <http://helpme1c.ru/yazyk-zaprosov-1s-8-3-dlya-nachinayushhix-programmistov-osnovy-sintaksisa>

ПРИЛОЖЕНИЕ А

Таблица 1 - Варианты заданий

№ варианта	Наименование предметной области	Подсистема справочников		Подсистема документов		Пользователи	
		Наименование подсистем	Наименование справочников	Наименование подсистем	Наименование документов	Доступные роли	Наименование пользователя
1	Управление прайс- листами	Товары, Ценообразование	Товары Типы цен Поставщики	Ценовая политика	Установка цен	Администратор	Администратор
						Менеджер по закупкам	ФИО студента
2	Взаиморасчеты	Номенклатура и контрагенты	Клиенты Услуги	Покупки и продажи	Договор Оказание услуг	Администратор	Администратор
						Менеджер торгового направления	ФИО студента
3	Товарооборот	Номенклатура	Товары Характеристики	Продажи	Реализация товаров	Администратор	Администратор
						Менеджер по продажам	ФИО студента
4	Управление проектами	Проекты, Ресурсы	Проекты Этапы Сотрудники	Учет рабочего времени	Отчет о работе	Администратор	Администратор
						Руководитель подразделения	ФИО студента
5	Управление персоналом	Структура предприятия, Кадры	Организации Подразделения Сотрудники	Учет кадровой информации	Прием на работу Штатное расписание	Администратор	Администратор
						Инспектор по кадрам	ФИО студента
6	Транспортные услуги	Ресурсы предприятия	Водители Грузчики Транспорт	Доставка товаров	Заказ клиента Наряд на доставку	Администратор	Администратор
						Диспетчер службы доставки	ФИО студента
7	Активы предприятия	Внеоборотные активы	Виды оборудования Цеха	Учет оборудования	Ввод в эксплуатацию Перемещение оборудования	Администратор	Администратор
						Бухгалтер	ФИО студента

№ варианта	Наименование предметной области	Подсистема справочников		Подсистема документов		Пользователи	
		Наименование подсистем	Наименование справочников	Наименование подсистем	Наименование документов	Доступные роли	Наименование пользователя
8	Управление производством	Номенклатура	Продукция Материалы Спецификации	Производство	Требование- накладная Отчет за смену	Администратор	Администратор
						Контроллер ОТК	ФИО студента
9	Управление торговлей	Номенклатура и склад	Товары на складе Единицы измерения	Покупки и продажи	Поступление товаров Реализация товаров	Администратор	Администратор
						Заведующий складом	ФИО студента
10	Корпоративная мобильная связь	Контактная информация	Телефоны Сотрудники	Учет мобильной связи	Отчет о мобильной связи	Администратор	Администратор
						Главный бухгалтер	ФИО студента
11	Интернет услуги	Нормативно- справочная информация	Абоненты Сотрудники	Подключения	Акт подключения услуг Отчет о предоставлении услуг	Администратор	Администратор
						Мастер технической поддержки и подключений	ФИО студента
12	Предоставление рекламных услуг	Номенклатура и контрагенты	Рекламные агентства Продукты Клиенты	Отчеты	Отчет о предоставлении услуг	Администратор	Администратор
						Менеджер по рекламе	ФИО студента
13	Оплата труда на предприятии	Зарплата и персонал	Сотрудники Тарифные ставки	Расчет заработной платы	Начисление заработной платы Выплата заработной платы	Администратор	Администратор
						Бухгалтер расчетного отдела	ФИО студента
14	Использование коммунальных услуг	Нормативно- справочная информация	Жилой фонд Услуги	Учет услуг	Оплата услуг Отчет об использовании	Администратор	Администратор
						Бухгалтер- кассир	ФИО студента

