

Федеральное государственное бюджетное образовательное учреждение
высшего образования «Кемеровский государственный университет»
Новокузнецкий институт (филиал)

Факультет информатики, математики и экономики
Кафедра информатики и вычислительной техники им. В. К. Буторина

О. А. Штейнбрехер

Практикум по экономическим пакетам прикладных программ

*Методические указания по выполнению практических работ по теме
«Разработка и модификация программных решений на основе типовых
платформ» дисциплины «Практикум по экономическим пакетам
прикладных программ» для обучающихся по направлению подготовки
09.03.03 Прикладная информатика Профиль «Прикладная
информатика в экономике»*

Новокузнецк
2020

УДК [378.147.88:004.4](072)
ББК 74.484(2Рос-4Кем)я73+ 32.972я73

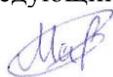
Ш88

Ш88 «Практикум по экономическим пакетам прикладных программ. Методические указания по выполнению практических работ по теме «Разработка и модификация программных решений на основе типовых платформ»» : метод. указ (текст. электрон. изд.)/ О.А. Штейнбрехер ; Новокузнец. ин-т (фил.) Кемеров. гос. ун-та – Новокузнецк: НФИ КемГУ, 2020. – 74 с.

Приводятся методические указания по выполнению практических работ по разделу «Разработка и модификация программных решений на основе типовых платформ» дисциплины «Практикум по экономическим пакетам прикладных программ».

Методические указания предназначены для студентов всех форм обучения направлений 09.03.03 «Прикладная информатика».

Рекомендовано
на заседании кафедры
информатики и вычислительной
техники им. В. К. Буторина
17 сентября 2020 года.
Заведующий кафедрой



А. В. Маркидонов

Утверждено
методической комиссией факультета
информатики, математики и экономики
24 сентября 2020 года.
Председатель методкомиссии



Г.Н. Бойченко

УДК [378.147.88:004.4](072)
ББК 74.484(2Рос-4Кем)я73+ 32.972я7

© Штейнбрехер О.А., 2020
© Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Кемеровский государственный университет»,
Новокузнецкий институт (филиал), 2020

Текст представлен в авторской редакции

Оглавление

ВВЕДЕНИЕ	4
ГЛОССАРИЙ	5
ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ	22
Создание конфигурации.....	22
Сохранение конфигурации и информационной базы в файл.....	23
Загрузка конфигурации из файла	23
Типы данных	23
Основные объекты конфигурации	25
Справочники	25
Документы	27
Регистр сведений	27
Регистр накоплений	29
Движение документа. Конструктор движения	30
Встроенный язык 1С: Предприятие	31
Формы в 1С: Предприятие	39
Настройка печатных форм и других макетов	46
Получение данных из справочника без запросов	47
Получение данных из регистра сведений.....	48
Конструктор запроса.....	48
Контроль отрицательных остатков	51
Программное создание элемента справочника, внесение изменений в элемент справочника.....	54
Программное создание и проведение документа	55
Варианты работы.....	56
Практическая работа №5. Проектирование программного решения и разработка объектов конфигурации.....	59
Практическая работа №6. Настройка прав пользователей и командного интерфейса конфигурации.....	60
Практическая работа №7. Решение задач оперативного учета	61
Практическая работа №8. Программное создание объектов конфигурации	63
Практическая работа №9. Управляемые формы	65
Практическая работа №10. Печатные формы и отчеты.....	72

ВВЕДЕНИЕ

Курс «Практикум по экономическим пакетам прикладных программ» предназначен для закрепления навыков эксплуатации, адаптации, настройки и разработки программных решений для экономических задач. Самой популярной платформой, применяемой для обеспечения автоматизации экономических процессов является «1С: Предприятие», поэтому данный курс ориентирован на решение задач с использованием типовых конфигураций и разработке решений на платформе. Система программ «1С: Предприятие 8» включает в себя платформу и прикладные решения, разработанные на ее основе, для автоматизации деятельности организаций и частных лиц. Сама платформа не является программным продуктом для использования конечными пользователями, которые обычно работают с одним из многих прикладных решений (конфигураций), разработанных на данной платформе. Такой подход позволяет автоматизировать различные виды деятельности, используя единую технологическую платформу.

В ходе изучения дисциплины обучающийся осваивает следующие компетенции:

ПК-2 - способностью разрабатывать, внедрять и адаптировать прикладное программное обеспечение;

СПК-1 - способностью разрабатывать информационные системы с учетом специфики малого и среднего предпринимательства.

Программа курса включает три раздела – «Эксплуатация пакетов программ», «Разработка и модификация программных решений на основе типовых платформ» и «Внедрение и настройка типовых программных решений и пакетов программ». Данные методические указания содержат практические задания и рекомендации по второму разделу курса.

Практические задания данного раздела рассматриваются как сквозное задание. В методических указаниях приведено краткое описание кейс-задач для создания информационных систем (по вариантам). Каждое практическое задание содержит дополнительные требования к информационной системе, которые требуется реализовать.

ГЛОССАРИЙ

Активные пользователи – список активных пользователей позволяет получать информацию о том, кто из пользователей работает с информационной базой в данный момент. Список активных пользователей содержит информацию об имени пользователя, режиме, в котором пользователь использует систему, времени начала его работы и пр. Пользователь имеет возможность отсортировать список по любой из колонок, вывести список активных пользователей на печать в виде текстового или табличного документа. Кроме этого, из списка активных пользователей можно открыть журнал регистрации системы, или просмотреть историю работы пользователя — содержимое журнала регистрации, отфильтрованное по тому пользователю, на котором установлен курсор.

Аутентификация 1С: Предприятия — это один из видов аутентификации, поддерживаемых механизмом аутентификации 1С: Предприятия. При использовании этого вида аутентификации средствами 1С: Предприятия в конфигураторе для пользователя задается пароль.

Аутентификация операционной системы — это один из видов аутентификации, поддерживаемых механизмом аутентификации 1С: Предприятия. В случае аутентификации средствами операционной системы в конфигураторе для пользователя выбирается один из пользователей операционной системы. При выполнении аутентификации средствами операционной системы, от пользователя не требуется каких-либо действий по вводу логина и пароля. Система анализирует, от имени какого пользователя операционной системы выполняется подключение к прикладному решению, и на основании этого определяет соответствующего пользователя 1С:Предприятия 8. При этом диалог аутентификации 1С:Предприятия не отображается, если не указан специальный параметр командной строки. Если для пользователя не указан ни один из видов аутентификации, — такому пользователю доступ к прикладному решению закрыт.

Бизнес-процессы — это прикладные объекты конфигурации. Они описывают бизнес-логику в карте маршрута и управляют жизненным циклом созданных бизнес-процессов (экземпляров) от момента старта до момента завершения. Необходимым свойством описания бизнес-процесса является связь с задачей, которая задает систему адресации и позволяет проектировать

карты маршрута в соответствии с поддерживаемой в прикладном решении организационной структурой.

Варианты работы системы – платформа поддерживает два варианта работы: файловый и клиент-серверный. И в том, и в другом варианте все прикладные решения работают полностью идентично. Файловый вариант работы рассчитан на персональную работу одного пользователя или работу небольшого количества пользователей в локальной сети. В этом варианте все данные информационной базы располагаются в одном файле — в файловой СУБД. Клиент-серверный вариант работы предназначен для использования в рабочих группах или в масштабе предприятия. Он реализован на основе трехуровневой архитектуры «клиент-сервер». В этом варианте информационная база хранится в одной из поддерживаемых систем управления базами данных, а взаимодействие между клиентским приложением и СУБД осуществляет кластер серверов «1С: Предприятия 8».

Внешние обработки представляют собой обработки, которые не входят в состав прикладного решения и хранятся в отдельных файлах с расширением *.erf.. Основное их преимущество заключается в том, что такие обработки можно использовать в различных прикладных решениях без изменения структуры самих решений. Кроме того, важным преимуществом внешних обработок является возможность проектировать и отлаживать их в процессе работы прикладного решения, без необходимости сохранения каждый раз конфигурации прикладного решения.

Внешние отчеты представляют собой отчеты, которые не входят в состав прикладного решения и хранятся в отдельных файлах с расширением *.erf.. Основное их преимущество заключается в том, что такие отчеты можно использовать в различных прикладных решениях без изменения структуры самих решений. Кроме того, важным преимуществом внешних отчетов является возможность проектировать и отлаживать их в процессе работы 1С:Предприятия, без необходимости сохранения каждый раз конфигурации прикладного решения.

Встроенный язык является важной частью технологической платформы «1С: Предприятия 8», поскольку позволяет разработчику описывать собственные алгоритмы функционирования прикладного решения.

Главная панель — это один из элементов командного интерфейса платформы. Он доступен пользователям любых прикладных решений. Главная панель предназначена для быстрого доступа к основным функциям

прикладного решения: меню функций, глобальному поиску, избранному, истории, к центру оповещений и главному меню.

Главное меню — это один из элементов командного интерфейса программы. Оно содержит набор команд, относящихся к прикладному решению в целом и независимых от прикладной специфики конфигурации. Главное меню расположено в главной панели основного окна программы.

Двухфакторная аутентификация — это один из видов аутентификации, поддерживаемых механизмом аутентификации 1С: Предприятия. Двухфакторная аутентификация требует, чтобы пользователь имел два из трех возможных типов аутентификационных данных:

- нечто ему известное, то, что он помнит (например, логин / пароль).
- нечто, чем он владеет (например, мобильный телефон).
- нечто ему присущее (например, отпечаток пальца).

Дерево объектов конфигурации представляет все прикладное решение в виде древовидной структуры, каждая ветвь которой описывает определенную составляющую конфигурации. Корневые ветви дерева объединяют объекты конфигурации, логически связанные между собой и имеющие общее назначение.

Диаграммы — это общие объекты встроенного языка. Они являются частью механизма формирования экономической и аналитической отчетности. Эти объекты предназначены для отображения данных в виде диаграмм различного вида. Они могут использоваться в табличных документах или непосредственно в формах (например, в форме отчета). Диаграммы являются интерактивными и поддерживают механизм расшифровок. Щелкая мышью на нужном показателе диаграммы, пользователь может получить детальную информацию или даже сформировать новый отчет.

Документы - это прикладные объекты конфигурации. Они позволяют хранить в прикладном решении информацию о совершенных хозяйственных операциях или о событиях, произошедших в "жизни" предприятия вообще.

Журналы документов - это прикладные объекты конфигурации. Они предназначены для просмотра документов разных видов. Для журнала документов могут быть определены графы, предназначенные для отображения реквизитов документов разного вида, отнесенных к данному журналу.

Журнал регистрации содержит информацию о том, какие события происходили в информационной базе в определенный момент времени или

какие действия выполнял тот или иной пользователь. Для каждой записи журнала, отражающей изменение данных, отображается статус завершения транзакции (транзакция завершена успешно, или же транзакция отменена). Это позволяет понять изменены реально данные или нет. Для событий успешной и неуспешной аутентификации в информационной базе 1С: Предприятия в журнал записывается, какой именно пользователь операционной системы выполняет эту аутентификацию. Для событий доступа к данным и отказа в доступе к данным можно гибко настроить состав регистрируемой информации. Набор полей объектов конфигурации, при доступе к которым будет регистрироваться событие, и состав дополнительной информации, которая будет записываться в журнал регистрации при наступлении этого события. Например, можно указать, что в журнал будут заноситься записи о том, что пользователь прочитал сумму начислений из регистра, хранящего данные о заработной плате. При этом в журнал будет занесена информация не только о том пользователе, который прочитал эти данные, но и информация о том, начисления какому именно сотруднику были прочитаны. Журнал регистрации доступен как в режиме 1С: Предприятие, так и в режиме Конфигуратор.

Задачи — это прикладные объекты конфигурации. Они предназначены для учета заданий и описывает способ их распределения по исполнителям, с учетом организационной структуры предприятия. Адресация заданий сотрудникам определяется реквизитами, в которых можно предусмотреть многомерную ролевую маршрутизацию, например, по ролям, рабочим группам, подразделениям, помещениям, филиалам и т. д. Задачи являются «движущей силой» механизма бизнес-процессов. При выполнении задачи породивший ее бизнес-процесс осуществляет переход на следующую точку маршрута в соответствии с картой маршрута.

Избранное — это один из стандартных элементов пользовательского интерфейса. Он предоставляется платформой и доступен пользователям любых прикладных решений. Он позволяет вести собственный список избранных ссылок.

Информационная панель — это один из элементов командного интерфейса программы. Она предназначена для отображения показателей производительности и индикации того, что включен режим имитации задержек при вызовах сервера.

Информация о текущем пользователе — это один из стандартных элементов пользовательского интерфейса. Она доступна в главной панели.

История — это один из стандартных элементов пользовательского интерфейса. Он предоставляется платформой и доступен пользователям любых прикладных решений. История содержит все действия пользователя, связанные с добавлением и изменением данных — элементов справочников, документов и т. д. Пользователь всегда может открыть список этих изменений и перейти к любому из объектов, перечисленных в этом списке. Например, для того, чтобы уточнить, правильно ли он изменил данные.

Картинки — это общие объекты конфигурации. Они позволяют включать в состав прикладного решения графические изображения — картинки. Разработчик может использовать картинки для размещения их в элементах управления, формах, макетах и даже обращаться к ним средствами встроеного языка.

Команда — это объект конфигурации, с помощью которого разработчик может описывать действия, предназначенные для выполнения пользователем. Существуют общие команды — команды, которые не имеют объектной специфики или служат для выполнения действий с объектами, которые не используют стандартные команды. Также команды могут существовать и у отдельных объектов конфигурации. Они служат для выполнения операций, связанных именно с этим объектом.

Командный интерфейс — это основное средство навигации пользователя по функциональности конфигурации. В системе 1С:Предприятие он строится на основе подсистем. Разработчик должен создать в конфигурации иерархию подсистем, отражающую для пользователя структуру функциональности прикладного решения.

Константы — это прикладные объекты конфигурации. Они позволяют хранить в информационной базе данные, которые не изменяются во времени, или изменяются очень редко. Каждая константа позволяет хранить одно значение. Например, в константе может храниться наименование предприятия, его ИНН и другая информация. В прикладном решении может быть создано произвольное количество констант.

Конструктор ввода на основании помогает создать процедуру на встроеном языке, которая будет вызываться при создании одного объекта прикладного решения на основании данных, содержащихся в другом объекте. Такая функциональность может потребоваться, например, если в прикладном решении на основании справочника Контрагенты должен создаваться документ Приход товара, содержащий те же реквизиты, что и

исходный элемент справочника. Конструктор ввода на основании можно вызвать, например, из окна редактирования справочника.

Конструктор движений — это один из инструментов разработки. Он используется только для документов и помогает создать процедуру обработки проведения документа на встроенном языке. Конструктор может быть вызван из окна редактирования документа.

Конструктор запроса — это один из инструментов разработки. Он позволяет составить текст запроса на языке запросов исключительно визуальными средствами.

Конструктор запроса с обработкой результата — это один из инструментов разработки. Он позволяет составить текст запроса и сформировать фрагмент программного кода, который исполняет запрос и выводит результаты в табличный документ или диаграмму. На первом шаге своей работы конструктор предлагает выбрать один из возможных вариантов обработки результата запроса: просто обход результата для его дальнейшей программной обработки или вывод данных в табличный документ или диаграмму.

Конструктор макета позволяет создавать макеты, используемые как объектами прикладного решения, так и самим прикладным решением в целом. Макеты могут содержать различные данные, которые требуются для отображения информации в процессе работы.

Конструктор макета оформления компоновки данных позволяет визуально настраивать оформление, которое будет использоваться генератором областей макета компоновки.

Конструктор настроек компоновки данных позволяет настроить отчет, созданный с использованием системы компоновки данных. Назначение конструктора заключается в том, чтобы предоставить разработчику или пользователю возможность быстрой настройки типичных отчетов нескольких видов: список, таблица и диаграмма.

Конструктор печати предназначен для создания макета печатной формы объекта прикладного решения и процедуры на встроенном языке, которая будет формировать печатную форму на основании этого макета. Конструктор печати может быть вызван, например, из окна редактирования справочника.

Конструктор схемы компоновки данных позволяет разработчику полностью описать схему компоновки данных исключительно визуальными средствами. Перемещаясь по закладкам конструктора можно указывать,

какие данные должны присутствовать в отчете, как они связаны, сгруппированы и какие ресурсы следует рассчитать и т. д.

Конструктор форматной строки позволяет разработчику составить текст форматной строки исключительно визуальными средствами. Форматные строки используются в конструкциях встроеного языка для того, чтобы сформировать нужное представление отображаемых данных. Конструктор поддерживает формирование форматных строк для числовых, логических значений и значений типа Дата.

Конструктор формы объекта конфигурации служит для создания различных форм, которые будут использованы системой или разработчиком при отображении данных этого объекта. Конструктор вызывается системой автоматически при создании новой формы (например, при создании новой формы справочника в окне редактирования справочника).

Конфигуратор — один из двух режимов работы системы. В этом режиме разрабатываются прикладные решения и выполняется администрирование информационных баз. Для этого используется среда быстрой разработки.

Макет — это подчиненный объект конфигурации. Макеты позволяют хранить в конфигурации различные данные, требующиеся для отображения в процессе работы, как всего прикладного решения, так и отдельных его объектов.

Меню функций — это один из стандартных элементов пользовательского интерфейса. Оно предоставляет удобный доступ сразу ко всем командам раздела.

Механизм аутентификации — это один из инструментов администрирования. Он позволяет определить, кто именно из пользователей, перечисленных в списке пользователей системы, подключается к прикладному решению в данный момент. Система поддерживает несколько видов аутентификации, которые могут использоваться в зависимости от конкретных задач, стоящих перед администратором информационной базы.

Механизм ввода на основании — это один из прикладных механизмов платформы. Он позволяет упростить работу пользователя с прикладным решением и избавить его от повторного ввода данных, которые уже хранятся в информационной базе.

Механизм запросов — это один из способов доступа к данным, которые поддерживает платформа. Используя этот механизм, разработчик может читать и обрабатывать данные, хранящиеся в информационной базе;

изменение данных с помощью запросов невозможно. Это объясняется тем, что запросы специально предназначены для быстрого получения и обработки некоторой выборки из больших массивов данных, которые могут храниться в базе данных.

Механизм оперативного учета — один из прикладных механизмов платформы. Проведение документов, при котором может осуществляться изменение данных, учитываемых в прикладном решении, может осуществляться в оперативном режиме. Оперативное проведение документов пользователями выполняется в режиме «реального времени», т. е. отображает изменения, факты, свершающиеся в настоящее время. Оперативное проведение особенно актуально при многопользовательской работе. В этом режиме, как правило, осуществляется максимум проверок, способных исключить ошибки при вводе данных пользователями. Например, при оперативном проведении обычно выполняется контроль остатков на складе списываемой номенклатуры с тем, чтобы исключить одновременную продажу одного и того же товара несколькими продавцами. Оперативное проведение служит для того, чтобы в реальном режиме многопользовательской работы определить возможность или невозможность выполнения той или иной операции (и выполнить ее, если возможно).

Механизм описания характеристик — это один из прикладных механизмов платформы. Он позволяет организовать хранение свойств объектов (справочников, документов и т. д.), которые еще не известны на момент разработки прикладного решения. Таким образом, например, для номенклатуры пользователь сможет самостоятельно вводить новые свойства: цвет, размер, габариты, мощность и т. д. Для каждой группы номенклатуры может быть создан свой набор свойств: для холодильников — объем морозильной камеры, число компрессоров, уровень шума; для компьютеров — объем оперативной памяти, объем жесткого диска; для одежды — размер, рост, цвет. В дальнейшем на основе этих характеристик можно строить отчеты, анализировать объемы продаж, получать другую информацию для принятия решений.

Механизм проверки конфигурации позволяет выявить ошибки, которые не являются критичными для функционирования прикладного решения в принципе, но наличие которых может существенно снизить скорость работы прикладного решения или даже привести к возникновению ошибок при работе в некоторых специальных режимах.

Механизм сложных периодических расчетов — это один из прикладных механизмов платформы. Он позволяет реализовывать различные модели расчета заработной платы. Работа механизма основана на двух составляющих. С одной стороны механизм сложных периодических расчетов содержит средства для описания различных видов расчета, которые будут использоваться в прикладном решении. Например, это могут быть такие виды расчета как оклад, алименты, штраф и т. д. Помимо собственно описания этих видов расчета, существует возможность задать правила, по которым одни виды расчета будут влиять на другие виды расчета. С другой стороны этот механизм предоставляет возможность хранения промежуточных данных, которые используются для выполнения расчетов, и конечных результатов расчетов. Работа механизма сложных периодических расчетов обеспечивается двумя объектами прикладного решения: План видов расчета и Регистр расчета.

Механизм сравнения и объединения конфигураций позволяет сравнивать между собой два прикладных решения и объединять их полностью или выборочно по результатам сравнения. Такая возможность используется, например, когда одно прикладное решение разрабатывается несколькими независимыми разработчиками, или в случае, когда в исходную конфигурацию нужно загрузить сделанные изменения. Этот механизм обеспечивает не только сравнение общих свойств объектов прикладного решения (справочников, документов и т. д.), но и сравнение их отдельных реквизитов, табличных частей. Также выполняется сравнение форм: сравниваются тексты модулей, тексты описаний и макеты. Все результаты сравнения можно просмотреть в детальном виде.

Мобильная платформа — это один из вариантов платформы для мобильных устройств. Мобильное приложение, собранное с помощью этого варианта платформы, может работать автономно на мобильном устройстве. Если проводить аналогию с платформой для настольных компьютеров, то такое мобильное приложение является аналогом тонкого клиента, работающего с файловой информационной базой, которая находится на том же компьютере, что и тонкий клиент.

Настройка панелей — это один из стандартных интерфейсных механизмов. Он предоставляется платформой и доступен пользователям любых прикладных решений. С его помощью пользователь может самостоятельно конструировать своё рабочее пространство, располагая панели в разных областях экрана.

Настройка списков — это один из стандартных интерфейсных механизмов. Он предоставляется платформой и доступен пользователям любых прикладных решений. С его помощью пользователь может изменить внешний вид списка в соответствии со своими предпочтениями — представить список в виде дерева или в виде линейного списка, сгруппировать элементы списка, задать динамическое оформление списка в зависимости от тех данных, которые в него выводятся.

Настройка форм — это один из стандартных интерфейсных механизмов. Он предоставляется платформой и доступен пользователям любых прикладных решений. С его помощью пользователь может изменить расположение элементов формы, скрыть отдельные элементы.

Начальная страница — это стандартный раздел программы, содержащий часто используемые документы, отчеты, справочники и т. п. Как правило, работа пользователя с программой всегда начинается с начальной страницы.

Нумераторы — это прикладные объекты конфигурации. Использование нумераторов позволяет организовать сквозную нумерацию документов разных видов. Для этого всем таким документам назначается один нумератор. Контроль уникальности и присвоение нового номера будет выполняться с учетом всех документов, для которых назначен этот нумератор. В прикладном решении может быть создано произвольное количество нумераторов.

Область данных — это множество данных, доступных в текущем сеансе на основании факта использования разделителей и на основании значений разделителей, определенных в конфигурации. Данными, которые могут быть отнесены к той или иной области, являются:

- данные базы данных;
- пользователи информационной базы;
- записи журнала регистрации;
- фоновые и регламентные задания;
- часовой пояс информационной базы;
- системные и пользовательские настройки;
- история;
- активные пользователи;
- другие данные.

Обновление конфигурации информационной базы требуется тогда, когда в процессе эксплуатации прикладного решения возникают ситуации,

требующие внесения изменений в прикладное решение. Например, может выйти новая версия прикладного решения или просто потребоваться добавление новой функциональности в существующее прикладное решение. В этих случаях администратор информационной базы может выполнить обновление конфигурации прикладного решения. Если изменения не затрагивают структуру данных, обновление конфигурации может быть выполнено динамически, без прерывания работы пользователей. Активные пользователи, для того, чтобы начать работать с измененной конфигурацией, должны перезапустить клиентское приложение. Если требуется изменять структуру существующих данных, обновление конфигурации может происходить в фоновом режиме, когда основная масса изменений выполняется без прерывания работы пользователей. И лишь в короткой, заключительной фазе реструктуризации требуется монопольный режим, при котором работа пользователей с базой невозможна.

Обработки — это прикладные объекты конфигурации. Они предназначены для выполнения различных действий над информацией. Например, с их помощью можно выполнять удаление из системы устаревших данных, импорт информации из других систем и многое другое. Характер выполняемых в этом случае действий отражает название объекта конфигурации — Обработка, так как в результате информация, хранящаяся в системе, претерпевает какие-либо изменения. Обработка может содержать одну или несколько форм, с помощью которых, при необходимости, можно организовать ввод каких-либо параметров, влияющих на ход алгоритма. Вывод результатов выполнения алгоритма на экран и принтер осуществляется с помощью конструктора запроса с обработкой результата. Основное отличие обработки от отчета заключается в том, что отчет может использовать схему компоновки данных. В остальном обработка не отличается от отчета.

Объекты конфигурации — это составные элементы, «детали», из которых складывается любое прикладное решение. Они представляют собой проблемно-ориентированные объекты, поддерживаемые на уровне технологической платформы. По большому счету задача разработчика заключается в том, чтобы собрать из этих объектов, как из конструктора, необходимую структуру прикладного решения и затем описать специфические алгоритмы функционирования и взаимодействия этих объектов, отличающиеся от их типового поведения. Состав объектов, поддерживаемых технологической платформой, является результатом

анализа предметных областей использования 1С:Предприятия, и выделения и классификации используемых в этих областях бизнес-сущностей. В результате этого анализа разработчик может оперировать такими объектами как справочники, документы, регистры сведений, планы счетов и пр.

Отладчик является встроенным в конфигуратор инструментом. Он помогает отлаживать программные модули, создаваемые в процессе разработки прикладного решения. Отладчик позволяет отслеживать последовательность выполнения операторов встроенного языка и просматривать значения переменных.

Отчеты — это прикладные объекты конфигурации. Они предназначены для обработки накопленной информации и получения сводных данных в удобном для просмотра и анализа виде. Конфигуратор позволяет формировать набор различных отчетов, достаточных для удовлетворения потребности пользователей системы в достоверной и подробной выходной информации. Как правило, для формирования выходных данных отчет использует систему компоновки данных. Но, вообще говоря, отчет может содержать произвольный алгоритм формирования «бумажного» или «электронного» отчета на встроенном языке. Отчет может содержать одну или несколько форм, с помощью которых, при необходимости, можно организовать ввод каких-либо параметров, влияющих на ход алгоритма.

Отчет по конфигурации позволяет вывести информацию обо всех объектах конфигурации в текстовый или табличный документ. Возможно создание отчета, как по всей конфигурации, так и по некоторой ее части. Можно выбрать только те объекты, которые относятся к какой-либо подсистеме или указать вручную перечень объектов, информацию о которых требуется выводить в отчет.

Палитра свойств служит для редактирования объекта конфигурации. Она представляет собой окно, содержащее набор свойств объекта (как доступных, так и не доступных для редактирования), набор ссылок на связанные с объектом формы и пр. Состав свойств, расположенных в панели свойств зависит от типа редактируемого объекта конфигурации. Все свойства объекта собраны в смысловые категории. Каждую категорию свойств можно свернуть или развернуть, щелкнув мышью по треугольнику в заголовке категории.

Панель инструментов — это один из стандартных интерфейсных механизмов. Он предоставляется платформой и доступен пользователям

любых прикладных решений. Панель инструментов предназначена для быстрого доступа к основным функциям прикладного решения: меню функций, избранному, истории и поиску.

Панель открытых — это один из стандартных элементов пользовательского интерфейса. Эта панель предназначена для частого переключения между открытыми формами. Каждой открытой форме соответствует отдельная закладка. Кроме этого в начале панели располагается закладка начальной страницы.

Панель разделов — это один из элементов командного интерфейса программы. Она показывает основную, главную структуру прикладного решения и позволяет перемещаться между разделами программы.

Параметры сеанса — это общие объекты конфигурации. Они предназначены для использования в ограничениях доступа к данным для текущего сеанса (но могут применяться и для других целей). Их значения сохраняются в течение данного сеанса 1С:Предприятия. Использование параметров сеанса позволяет снизить время доступа к данным при ограничении доступа на уровне записей и полей.

Перечисления - это прикладные объекты конфигурации. Они позволяют хранить в информационной базе наборы значений, которые не изменяются в процессе работы прикладного решения. Например, это может быть перечисление состояния заказов (Запланировано, ВРаботе, Выполнено) и пр.

Планы видов расчета — это прикладные объекты конфигурации. Они используются в механизме сложных периодических расчетов и служат для описания видов расчета и их взаимного влияния друг на друга.

Планы видов характеристик — это прикладные объекты конфигурации. Они предназначены для хранения информации о характеристиках различных объектов. С их помощью пользователь может создавать всевозможные характеристики, описывать тип этих характеристик и задавать их значения. Например, для того, чтобы описывать товары произвольным количеством произвольных характеристик (цвет, размер, запах и т. д.).

Планы счетов — это прикладные объекты конфигурации. Каждый из них позволяет описать совокупность синтетических счетов, предназначенных для группировки информации о хозяйственной деятельности предприятия. Путем настройки плана счетов организуется, собственно, требуемая система учета.

Подсистемы — это общие объекты конфигурации. На их основе платформа формирует командный интерфейс прикладного решения и визуально разделяет всю функциональность программы на крупные и мелкие блоки. Подсистемы могут иметь иерархическую структуру, т. е. одна подсистема может включать в себя несколько других подсистем.

Показатели производительности — это один из инструментов разработки прикладных решений. Он показывает разработчику в реальном времени, в режиме 1С:Предприятие, информацию о количестве и длительности вызовов сервера, а также объем принятых и переданных данных.

Последовательности — это прикладные объекты конфигурации. Они предназначены для обеспечения контроля правильности изменений, внесенных документами в учетные данные. Кроме этого последовательности, в случае необходимости, позволяют восстанавливать правильную картину изменений.

Раздел — один из элементов командного интерфейса программы. Он представляет собой выделенную функциональную часть программы, которая предназначена для решения определенного фиксированного круга задач.

Регистры бухгалтерии — это прикладные объекты конфигурации. Они используются в механизме бухгалтерского учета и позволяют вести многоуровневый и многомерный аналитический учет, учет по нескольким планам счетов, опциональное ведение количественного, суммового и валютного учета по отдельным разрезам аналитики и т. д.

Регистры накопления — это прикладные объекты конфигурации. Они составляют основу механизма учета движения средств (финансов, товаров, материалов и т. д.), который позволяет автоматизировать такие направления, как складской учет, взаиморасчеты, планирование. Регистр накопления образует многомерную систему измерений и позволяет «накапливать» числовые данные в разрезе нескольких измерений. Например, в таком регистре можно накапливать информацию об остатках товаров в разрезе номенклатуры и склада, или информацию об объемах продаж в разрезе номенклатуры и подразделения компании.

Регистры расчета — это прикладные объекты конфигурации. Они используются в механизме сложных периодических расчетов и служат для хранения записей о тех или иных видах расчета, которые необходимо выполнить, а также для хранения промежуточных данных и самих результатов выполненных расчетов.

Регистры сведений — это прикладные объекты конфигурации. Они позволяют хранить в прикладном решении произвольные данные в разрезе нескольких измерений. Например, в регистре сведений можно хранить курсы валют в разрезе валют, или цены предприятия в разрезе номенклатуры и типа цен.

Редактор «Все ограничения доступа» — это один из инструментов разработки. Он позволяет просматривать, добавлять, удалять, изменять, копировать ограничения из одной роли в другую, а также отбирать ограничения по различным критериям.

Редактор «Все роли» — это один из инструментов разработки. Он позволяет изменять и анализировать состав прав сразу для нескольких или для всех ролей, существующих в прикладном решении.

Редактор графической схемы — это один из инструментов разработки. Он предназначен для редактирования карт маршрутов, которые существуют у бизнес-процессов. Также с помощью этого редактора можно создавать произвольные графические схемы, не связанные с функциональностью бизнес-процессов.

Редактор интерфейса клиентского приложения — это один из инструментов разработки. Он предназначен для того, чтобы настроить расположение панелей в основном окне приложения.

Редактор командного интерфейса — это один из инструментов разработки. Он предназначен для настройки команд какой-либо подсистемы. Редактор командного интерфейса позволяет настроить состав команд панели функций текущего раздела, порядок отображения и видимость команд для разных ролей, определенных в конфигурации.

Редактор командного интерфейса конфигурации — это один из инструментов разработки. Он предназначен для того, чтобы настроить порядок следования разделов в панели разделов и настроить видимость разделов для разных ролей, определенных в конфигурации.

Редактор формы используется для создания и редактирования форм объектов прикладного решения. Формы объектов используются системой для визуального отображения данных в процессе работы пользователя. Любая форма представляет совокупность нескольких составляющих:

- элементов — объектов, определяющих визуальное представление формы и осуществляющих взаимодействие с пользователем,
- командного интерфейса — совокупности команд, отображаемых в форме;

- реквизитов — объектов, данные которых форма использует в своей работе.
- команд — действий, которые определены в данной конкретной форме,
- параметров — объектов, значения которых характеризуют саму форму, используются при ее создании и остаются постоянными в процессе «жизни» формы,
- модуля — программы на встроенном языке, отвечающей за работу с элементами и за обработку событий.

Роли — это общие объекты конфигурации. Они предназначены для реализации ограничения прав доступа в прикладных решениях. Роль в конфигурации может соответствовать должностям или видам деятельности различных групп пользователей, для работы которых предназначена данная конфигурация.

Система компоновки данных представляет собой механизм, основанный на декларативном описании отчетов. Он предназначен для построения отчетов, а также вывода информации, имеющей сложную структуру и содержащий произвольный набор таблиц и диаграмм. Система прав доступа позволяет описывать наборы прав, соответствующие должностям пользователей или виду деятельности. Структура прав определяется конкретным прикладным решением. Кроме этого, для объектов, хранящихся в базе данных (справочники, документы, регистры и т. д.) могут быть определены права доступа к отдельным полям и записям. Например, пользователь может оперировать документами (накладными, счетами и т. д.) определенных контрагентов и не иметь доступа к аналогичным документам других контрагентов.

Система типов — это особая система, по которой организуются данные, используемые прикладными решениями. Система типов позволяет представить информацию реального мира в терминах, «понятных» для «1С: Предприятия 8».

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.

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

списочный характер. Это может быть, например, список сотрудников, перечень товаров, список поставщиков или покупателей.

Ссылка — это значение, однозначно характеризующее объекты базы данных (элементы справочников, документы и так далее). Для хранения ссылок предназначены типы встроенного языка СправочникСсылка.<имя>, ДокументСсылка.<имя> и так далее.

Толстый клиент — это одно из клиентских приложений системы «1С:Предприятие 8». В операционной системе Windows исполняемый файл этого приложения — 1cv8.exe. В операционной системе Linux — 1cv8. «Толстым» клиент называется потому, что может исполнять практически всю функциональность, предоставляемую встроенным языком, в том числе умеет работать с прикладными типами данных, такими как СправочникОбъект.<имя>, ДокументОбъект.<имя> и т. д. Но, по этой же причине, он требует значительного количества аппаратных ресурсов на компьютере пользователя и может «общаться» с базой данных или с кластером серверов «1С:Предприятия 8» только посредством файлового доступа или по локальной сети.

Тонкий клиент — это одно из клиентских приложений системы «1С:Предприятие 8». В операционной системе Windows исполняемый файл этого приложения — 1cv8c.exe. В операционной системе Linux — 1cv8c. «Тонким» клиент называется потому, что умеет исполнять ограниченный набор функциональности встроенного языка. В частности на тонком клиенте недоступны все прикладные типы данных. Вместо этого тонкий клиент оперирует ограниченным набором типов встроенного языка, предназначенным лишь для отображения и изменения данных в памяти. Вся работа с базой данных, объектными данными, исполнение запросов — выполняется на стороне сервера. Тонкий клиент только получает готовые данные, подготовленные для отображения.

Файловая база данных — это файл 1Cv8.CD, в котором хранятся все данные информационной базы (конфигурация, база данных, административная информация) при работе системы в файловом варианте. Файловой базой данных управляет файловая СУБД, которая разработана фирмой «1С» и является частью платформы.

Формы в 1С: Предприятие 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.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Создание конфигурации

Для работы с конфигурацией требуется создать информационную базу: в окне «Запуск 1С: Предприятие» выбрать кнопку «Добавить», в форме «Добавление информационной базы/группы» выбрать «Создание новой информационной базы», затем «Создание информационной базы без конфигурации для разработки новой конфигурации или загрузки выгруженной ранее информационной базы», затем указать наименование и каталог информационной базы. Для работы с конфигурацией требуется выбрать ее в перечне информационных баз и открыть в режиме Конфигуратор, затем в меню «Конфигурация» выбрать пункт «Открыть конфигурацию». В рабочем окне откроется дерево конфигурации (рисунок 1), в котором будут расположены все созданные объекты конфигурации.

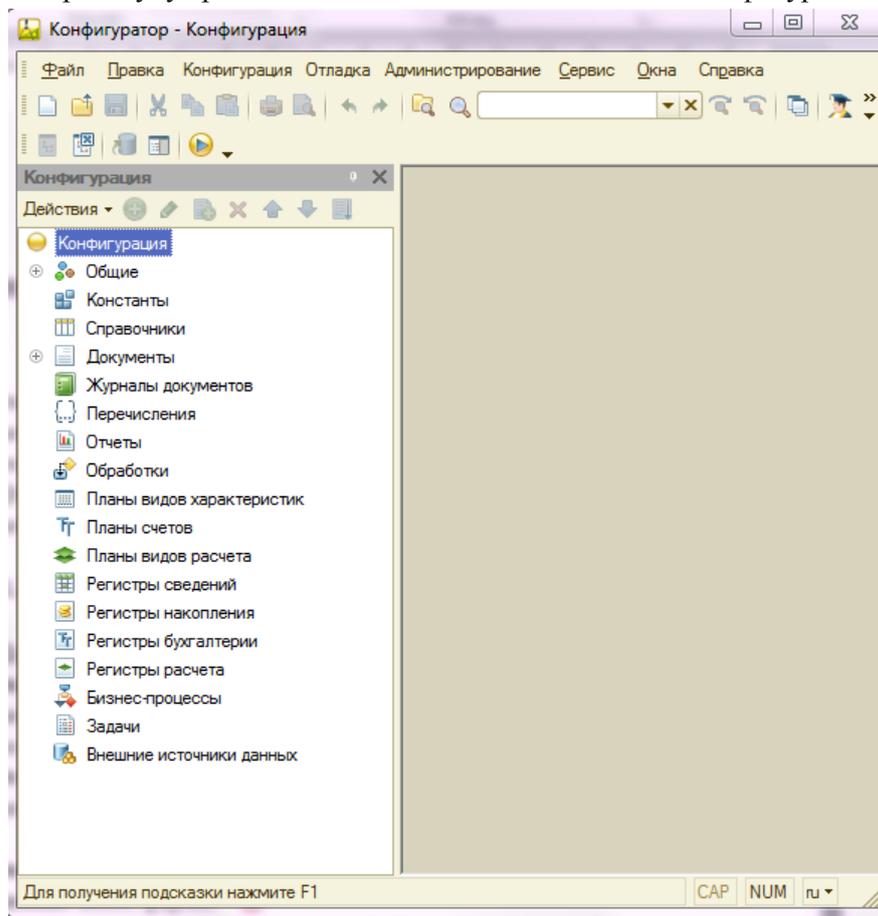


Рисунок 1 – Окно конфигуратора, дерево конфигурации

Сохранение конфигурации и информационной базы в файл

Прикладное решение состоит из конфигурации и информационной базы. Для того чтобы открыть конфигурацию на другом рабочем месте требуется выгрузить конфигурацию и информационную базу в файлы. Информационная база содержит в себе шаблон конфигурации.

Для сохранения конфигурации в файл в меню *«Конфигурация»* выбирается пункт *«Сохранить конфигурацию в файл»*, затем выбирается путь сохранения. Файл конфигурации имеет расширение *.cf. В меню *«Конфигурация»* можно также сравнить текущую конфигурацию с выгруженной ранее и объединить их. Инструменты для управления поставкой конфигурации и обновления конфигурации так же находятся в данном меню.

Для выгрузки информационной базы в файл в меню *«Администрирование»* выбирается пункт *«Выгрузить информационную базу»*. Файл выгружаемых данных имеет формат *.dt. Следует заметить, что выгрузка информационной базы невозможна при работе конфигурации в пользовательском режиме (в том числе при отладке конфигурации).

Загрузка конфигурации из файла

Для открытия конфигурации на другом рабочем месте в пустой конфигурации (созданной заново) в меню *«Конфигурация»* выбирается пункт *«Загрузить конфигурацию из файла»* и выбирается ранее выгруженный файл расширения *.cf. После загрузки конфигурации загружается информационная база. Для этого в меню *«Администрирование»* выбрать *«Загрузить информационную базу»*. При этом происходит сравнение текущей конфигурации и конфигурации информационной базы.

Если есть доступ к каталогу, в котором содержатся все файлы конфигурации (каталог, который был указан при создании базы), то можно выбрать этот каталог при создании новой конфигурации. Для этого в форме *«Добавление информационной базы/группы»* выбрать *«Добавление в список существующей информационной базы»* и указать каталог расположения.

Типы данных

Платформа «1С: Предприятие» поддерживает собственную систему типов. Система типов — это особая система, по которой организуются данные, используемые прикладными решениями. Основной особенностью системы типов является то, что есть типы, существующие в любом прикладном решении. Сами эти типы определены на уровне платформы и присутствуют всегда, независимо от действий разработчика. Наряду с ними

в конкретном прикладном решении могут существовать различные типы данных, присущие именно этому конкретному прикладному решению. Для таких типов данных на уровне платформы определены лишь общие правила их создания, шаблоны.

Можно выделить несколько основных категорий типов данных. Прimitивные типы данных — это такие типы как Строка, Число, Дата, Булево и другие. Значения примитивных типов являются простыми неделимыми значениями, в которых нельзя выделить отдельные составляющие. Например, значениями типа Число могут быть 1, 8, 15 и др.

Также, существуют более сложные типы данных. Например, платформа поддерживает целый ряд типов, которые представляют собой универсальные коллекции значений: Массив, Структура, СписокЗначений и другие. Универсальные коллекции значений не используются для определения типов реквизитам объектов конфигурации.

Кроме этого в платформе реализованы специфические типы данных, реализующие ту или иную функциональность прикладных решений: ТекстовыйДокумент, ТабличныйДокумент, ХранилищеЗначения, ПостроительЗапроса и другие. Общие типы называют также общими объектами. Значения этих типов, в отличие от значений примитивных типов, представляют собой совокупность значений отдельных свойств объекта. Поэтому их называют экземплярами объектов.

Интерфейсные типы позволяют организовывать визуальное взаимодействие прикладного решения с пользователем. В основном это типы, связанные с работой форм и их элементов. Например, тип данных НастройкиФормы.

Однако, наряду с типами данных, которые определены на уровне платформы, конкретное прикладное решение может использовать уникальные типы данных, существующие только в этом конкретном прикладном решении. Как правило, появление новых типов данных в прикладном решении связано с использованием прикладных объектов конфигурации. Поэтому такие типы называют еще прикладными типами или прикладными объектами. На уровне платформы поддерживается несколько классов (шаблонов) прикладных объектов, которые сами по себе не могут быть использованы в конкретном прикладном решении. Например, можно перечислить такие классы прикладных объектов как Справочники, Документы, Регистры сведений, Планы видов характеристик и пр.

Например, разработчик может добавить в свое прикладное решение новый справочник «Номенклатура», который будет наследовать функциональность класса Справочники, или новый документ «КассовыйОтчет», который будет наследовать функциональность класса Документы.

Сразу же после такого добавления разработчику становятся доступны новые типы данных, состав которых определяется принадлежностью объекта конфигурации к тому или иному классу прикладных объектов.

Основные объекты конфигурации

Платформа 1С: Предприятие обладает следующими стандартными видами объектов: константы, перечисления, справочники, документы, регистры и т.д. Все объекты конфигурации имеют свойство «Имя» и «Синоним». «Имя» объекта имеет ограничения: оно не должно содержать пробелы и не должно начинаться с символа или цифры. «Синоним» отображается в режиме пользователя на формах.

Сложные объекты, такие как справочники, документы, регистры и так далее имеют поля – реквизиты. Рассматривая структуру конфигурации как базу данных – объекты базы являются таблицами базы, а реквизиты объектов – полями таблицы. При этом константы и перечисления являются таблицами, состоящими из одного поля. Реквизиты объектов определяются пользователем, но каждый объект имеет несколько стандартных реквизитов.

Справочники

Справочники - это прикладные объекты конфигурации. Они позволяют хранить в информационной базе данные, имеющие одинаковую структуру и списочный характер. Каждый элемент справочника характеризуется реквизитами «Код» и «Наименование». Система поддерживает режим автоматической нумерации элементов, при котором она самостоятельно может генерировать код для нового элемента справочника. Кроме этого система позволяет осуществлять контроль уникальности кодов справочника, не разрешая создавать элементы с одинаковыми кодами.

Помимо кода и наименования, каждый элемент справочника, как правило, содержит некоторую дополнительную информацию, которая подробно описывает этот элемент. Например, для товара это может быть информация об артикуле, упаковке и т.п. Набор такой информации является одинаковым для всех элементов конкретного справочника, и для ее хранения служат реквизиты справочника.

Кроме этого, каждый элемент справочника может содержать некоторый набор информации, которая одинакова по своей структуре, но различна по количеству, для разных элементов справочника. Например, для каждого сотрудника в справочнике «Физические лица» это может быть контактная информация или информация о составе семьи, образовании. Для хранения подобных данных служат табличные части справочника.

Справочники могут поддерживать иерархическое расположение элементов. Существует два вида иерархии: иерархия групп и элементов и иерархия элементов. Иерархия групп и элементов предполагает наличие групп, в которых будут располагаться элементы, относящиеся к этим группам. Например, в справочнике «Номенклатура» могут быть созданы группы: Бытовая техника, Обувь, Продукты и т.д. Примером справочника с иерархией элементов может служить справочник «Подразделения», где элементы справочника относятся не к группам, а к другим элементам.

Разные справочники могут находиться в состоянии подчинения, т.е. элементы одного справочника могут быть подчинены элементам или группам другого справочника. Например, справочник «Кассы» может быть подчинен справочнику «Организации». Тогда при оформлении кассовых документов для некоторой организации можно будет выбрать кассу не среди всех имеющихся в программе касс, а среди касс, существующих только в этой организации.

Использование иерархии и/или подчинения активизирует встроенные реквизиты «Родитель» и «Владелец», соответственно. Тип данных реквизитов является ссылочным, одно из значений – пустая ссылка. Таким образом, при заполнении элемента справочника определяется его подчинение конкретному объекту и положение в иерархии.

Справочники допускают также создание предопределенных элементов, которые существуют в справочнике всегда, вне зависимости от действий пользователя. Такие элементы справочника создаются разработчиком при разработке прикладного решения и не могут быть удалены или перемещены пользователем. Предопределенными элементами могут быть элементы справочника или группы. Предопределенные элементы могут использоваться при кодировании обработчиков событий.

Для создания предопределенных элементов нужно зайти на вкладку «Прочее» и выбрать «Предопределенные» или в меню «Действия» - «Открыть предопределенные данные».

Документы

Документы - это прикладные объекты конфигурации. Они позволяют хранить в прикладном решении информацию о совершенных хозяйственных операциях или о событиях, произошедших в "жизни" предприятия вообще.

Каждый документ характеризуется номером, датой и временем. 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. Настройка параметров нумерации позволяет отслеживать уникальность как среди всех элементов документа, так и с учетом периода. Важными характеристиками документа являются дата и время – обязательные стандартные реквизиты (рисунок 2). Они позволяют установить строгую временную последовательность совершения операций.

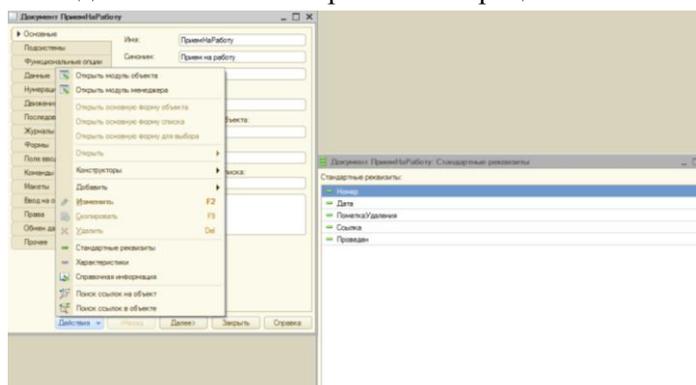


Рисунок 2 – Стандартные реквизиты документа

Кроме predetermined requisites the document contains some information, identical in its structure. Besides this, each document may contain some information, which is identical in its structure, but different in quantity, for different documents. For storage of such data, tabular parts of the document are used.

Регистр сведений

Регистры сведений - это прикладные объекты конфигурации. Они позволяют хранить в прикладном решении произвольные данные в разрезе нескольких измерений. Например, в регистре сведений можно хранить курсы валют в разрезе валют, или цены предприятия в разрезе номенклатуры и типа цен. Информация в регистре сведений хранится в виде записей, каждая из

которых содержит значения измерений и соответствующие им значения ресурсов.

Измерения регистра описывают разрезы, в которых хранится информация, а ресурсы регистра непосредственно содержат хранимую информацию. Вместе с каждой записью, находящейся в регистре сведений, можно хранить дополнительную произвольную информацию. Для этого служат реквизиты регистра сведений.

Одной из возможностей регистра сведений является хранение данных не только в разрезе указанных измерений, но и в разрезе времени. Разработчик может указать минимальную периодичность, с которой записи будут заноситься в регистр. В этом случае к каждой записи регистра будет добавляться поле «Период», хранящее дату, которой были внесены записи в регистр. Использование периодичности регистра сведений позволяет не просто хранить статические данные, но и отслеживать их изменение во времени.

Внесение изменений в регистр сведений может выполняться как вручную, так и при помощи документов. В случае, когда изменения в регистр сведений вносятся с помощью документов, к каждой записи регистра добавляется специальное поле, в котором хранится информация о регистраторе - документе, с которым связана эта запись. В процессе создания прикладного решения разработчик указывает, какой именно режим записи будет использоваться данным регистром сведений.

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

Система обеспечивает контроль уникальности записей, хранящихся в регистре сведений. Таким образом, в регистре сведений не может находиться двух одинаковых записей. Одинаковыми считаются записи, у которых совпадает ключ записи. Ключ записи формируется системой автоматически, на основании значений, содержащихся в полях записи, и зависит от вида регистра сведений.

В общем случае в формировании ключа записи будут участвовать значения регистратора, периода и значения измерений.

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

- создание, изменение и удаление записей;

- выбор записей в заданном интервале по заданным критериям;
- выбор записей по регистратору;
- получение значений ресурсов записей, соответствующих указанному периоду и значениям измерений;
- получение значений ресурсов наиболее ранних и наиболее поздних записей регистра, соответствующих указанному периоду и значениям измерений.

Регистр накоплений

Регистры накопления - это прикладные объекты конфигурации. Они составляют основу механизма учета движения средств (финансов, товаров, материалов и т.д.), который позволяет автоматизировать такие направления, как складской учет, взаиморасчеты, планирование.

Регистр накопления образует многомерную систему измерений и позволяет "накапливать" числовые данные в разрезе нескольких измерений. Например, в таком регистре можно накапливать информацию об остатках товаров в разрезе номенклатуры и склада, или информацию об объемах продаж в разрезе номенклатуры и подразделения компании.

Информация в регистре накопления хранится в виде записей, каждая из которых содержит значения измерений и соответствующие им значения ресурсов.

Измерения регистра описывают разрезы, в которых хранится информация, а в ресурсах регистра накапливаются нужные числовые данные.

Изменение состояния регистра накопления происходит, как правило, при проведении документа. Поэтому каждая запись регистра связана с определенным документом - регистратором, номером строки этого документа, и датой - периодом. В общем случае значение поле «Период» может не совпадать с датой документа.

Существует два вида регистров накопления: регистры накопления остатков и регистры накопления оборотов. Регистр накопления остатков позволяет хранить как итоговые значения ресурсов - остатки, так и изменения этих ресурсов - обороты. Регистр накопления оборотов является более "специализированным" видом регистра накопления и позволяет хранить только изменения ресурсов - обороты.

Существование регистра накопления оборотов связано с тем, что при автоматизации экономической деятельности существует большое количество ситуаций, когда требуется накапливать только обороты, а значения остатков не имеют смысла.

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

- выбор записей в заданном интервале по заданным критериям;
- выбор записей по регистратору;
- получение остатков и оборотов на указанный момент времени по заданным значениям измерений;
- режим работы с разделением итогов, который обеспечивает более высокую параллельность записи в регистр;
- отключение использования текущих итогов;
- расчет итогов на указанную дату;
- чтение, изменение и запись набора записей в регистр;
- возможность записи в регистр без пересчета итогов;
- полный пересчет итогов и пересчет итогов за указанный период.

Движение документа. Конструктор движения

Важным свойством документа является возможность его проведения. Если документ проводится, то он может изменить состояние тех или иных учитываемых данных. Если же документ не является «проводимым» это значит, что событие, которое он отражает, не влияет на состояние учета, который ведется в данном прикладном решении.

Алгоритм, на основании которого документ вносит те или иные изменения в состояние учетных данных при своем проведении, описывается средствами встроенного языка на этапе разработки прикладного решения. Система содержит *«Конструктор движений»*, который помогает разработчику создавать алгоритмы проведения документа.

Конструктор движений - это один из инструментов разработки. Он используется только для документов и помогает создать процедуру обработки проведения документа на встроенном языке. Конструктор может быть вызван, например, из окна редактирования документа.

Конструктор позволяет выбрать регистры, в которые будут вноситься записи и затем вручную или автоматически заполнить выражения, которые будут записаны в поля регистра. Работа с Конструктором движения похожа на работу с Конструктором ввода на основании.

Результатом работы конструктора является готовая процедура на встроенном языке с именем **ОбработкаПроведения()**. Эта процедура располагается в модуле документа и будет вызвана системой в момент проведения документа.

Встроенный язык 1С: Предприятие

Встроенный язык имеет много общих черт с другими языками, такими как Pascal, Java Script, Basic, что облегчает его освоение начинающими разработчиками. Однако он не является прямым аналогом какого-либо из перечисленных языков.

Встроенный язык поддерживает двойной синтаксис – на русском и английском языках. Каждый объект и оператор имеет два аналога. Можно использовать и русский язык, и английский, и комбинацию двух языков. Но существуют рекомендации использовать русский язык, потому что в системе достаточно большое количество объектов, которые имеют длинные названия.

Прикладные решения в 1С:Предприятии 8 не кодируются целиком. Большая часть прикладного решения создается разработчиком путем визуального конструирования — создания новых объектов конфигурации, задания их свойств, форм представления, взаимосвязей и пр. Встроенный язык используется лишь для того, чтобы определить поведение объектов прикладного решения, отличное от типового, и создать собственные алгоритмы обработки данных.

По этой причине модули, содержащие текст на встроенном языке, используются системой в конкретных, заранее известных ситуациях, которые могут возникнуть в процессе работы прикладного решения. Такие ситуации называются событиями. События могут быть связаны с функционированием объектов прикладного решения или с самим прикладным решением, как таковым. Например, с функционированием объекта прикладного решения Справочник связан ряд событий, среди которых есть событие ПередЗаписью. Это событие возникает непосредственно перед тем, как данные элемента справочника должны быть записаны в базу данных. Разработчик, используя встроенный язык, может описать алгоритм, который, например, будет проверять корректность данных, введенных пользователем.

Таким образом, можно сказать, что встроенный язык является скриптовым языком для программирования бизнес-логики, а использование модулей на встроенном языке является событийно-зависимым, т. е. выполнение модулей происходит при возникновении определенных событий в процессе функционирования прикладного решения.

Переменные

Каждый модуль описывает поведение конфигурации в определенной точке. В модуле содержится, прежде всего, раздел описания переменных. Т.е. мы можем объявить в модуле некоторые переменные.

В дальнейшем они могут быть использованы в процедурах и функциях этого модуля. Если переменная определена с ключевым словом Экспорт, то она будет доступна вне данного модуля. Пример строки объявления переменных:

Перем Склад, Подразделение, Кладовщик Экспорт;

После объявления переменных содержится раздел процедур и функций.

За ними располагается раздел основной программы, который будет выполняться в момент обращения к данному модулю.

Разделителем операторов является символ «;» (точка с запятой). Этот знак является признаком окончания оператора. Точку с запятой можно не ставить в завершающем операторе данной конструкции, например, процедуры.

Переменные предназначены для того, чтобы хранить некоторые значения любого типа данных. Они используются для промежуточного хранения информации, для обработки. 1С: Предприятие поддерживает мягкую типизацию, поэтому переменная может содержать значения различных типов данных.

Описывать переменные можно двумя способами:

- неявный способ (упоминание в левой части оператора присваивания описывает данную переменную, нет предварительного описания переменной со словом Перем, т.е. нет специального раздела описания переменных);
- явное описание переменных (Перем КонтрольныеДанные;). Явное описание переменных используется, например, если предполагается последующая передача этой переменной в функцию.

Для названия переменных используется классическое описание идентификатора. Идентификатор состоит из букв, цифр и знаков подчеркивания. Начинаться идентификатор должен либо с буквы, либо со знака подчеркивания.

Операторы языка

Операторы языка можно разделить на три группы:

1. Операторы определения переменных – выполняют объявление переменной (присваивают ей символьное имя) и ее инициализацию некоторыми данными, чтобы другие операторы могли их использовать. Символьное имя переменной называется идентификатором, а содержащиеся данные – значением;

Управляющие операторы – это операторы, которые сами не производят каких-либо вычислений, но управляют этим процессом. К ним относятся операторы ветвления, операторы циклической обработки и операторы передачи управления;

Исполняемые операторы – это операторы, выполняющие непосредственную обработку данных. К ним относят системные процедуры и функции, заложенные разработчиками платформы, а также процедуры и функции, описанные разработчиком прикладной задачи (конфигурации). Все исполняемые операторы характеризуются наличием идентификатора и, возможно, дополнительных параметров.

Оператор присваивания

Оператор присваивания = (равно) используется для инициализации переменных, реквизитов объектов или элементов массива некоторыми значениями, а также для переопределения их значений при последующих упоминаниях. В качестве источника значений могут выступать: символьная константа, имя переменной, имя реквизита объекта агрегатного типа, элемент массива или выражения с их участием.

Первое упоминание имени переменной в левой части оператора присваивания считается неявным определением переменной. Неявное определение переменных используется в том случае, когда область видимости переменной не важна и ограничена текущим контекстом выполнения.

Оператор присваивания = имеет следующий синтаксис:

Получатель = Источник;

А также альтернативный англоязычный синтаксис:

Destination = Source;

Операторы ветвления

Оператор ***Если*** управляет выполнением программы, основываясь на результаты одного или более логических выражений. Оператор может содержать любое количество групп операторов, возглавляемых конструкциями ***ИначеЕсли – Тогда***.

Синтаксис:

```
Если <Логическое выражение> Тогда
// Операторы
[ИначеЕсли <Логическое выражение> Тогда]
// Операторы
[Иначе]
// Операторы
```

КонецЕсли;

Англоязычный синтаксис:

```
If <Логическое выражение> Then  
// Операторы  
[ElsIf <Логическое выражение> Then]  
// Операторы  
[Else]  
// Операторы  
EndIf;
```

Параметры:

<Логическое выражение> – логическое выражение.

Тогда – операторы, следующие за **Тогда**, выполняются, если результатом логического выражения является значение **Истина**.

// **Операторы** – исполняемый оператор или последовательность таких операторов.

ИначеЕсли – логическое выражение, следующее за ключевым словом **ИначеЕсли**, вычисляется только тогда, когда условия в **Если** и всех предшествующих **ИначеЕсли** оказались равны **Ложь**. Операторы, следующие за конструкцией **ИначеЕсли – Тогда**, выполняются, если результат логического выражения в данном **ИначеЕсли** равен **Истина**.

Иначе – Операторы, следующие за ключевым словом **Иначе**, выполняются, если результаты логических выражений в конструкции **Если** и всех предшествующих конструкциях **ИначеЕсли** оказались равны **Ложь**.

КонецЕсли– Ключевое слово, которое завершает структуру оператора условного выполнения.

Операторы цикла

Оператор цикла **Для** предназначен для циклического повторения операторов, находящихся внутри конструкции **Цикл – КонецЦикла**. Перед началом выполнения цикла значение <Выражение 1> присваивается переменной <Имя переменной>. Значение <Имя переменной> автоматически увеличивается при каждом проходе цикла. Величина приращения счетчика при каждом выполнении цикла равна 1. Цикл выполняется, пока значение переменной <Имя переменной> меньше или равно значению <Выражение 2>. Условие выполнения цикла всегда проверяется вначале, перед выполнением цикла.

Синтаксис:

```
Для <Имя переменной> = <Выражение 1> По <Выражение 2> Цикл  
// Операторы  
[Прервать;]
```

```
// Операторы  
[Продолжить;]  
// Операторы  
КонецЦикла;
```

Англоязычный синтаксис:

```
For <Имя переменной> = <Выражение 1> To <Выражение 2> Do  
// Операторы  
[Break;]  
// Операторы  
[Continue;]  
// Операторы  
EndDo;  
Параметры:
```

<Имя переменной> – идентификатор переменной (счетчика цикла), значение которой автоматически увеличивается на 1 при каждом повторении цикла. Так называемый счетчик цикла.

<Выражение 1> – числовое выражение, которое задает начальное значение, присваиваемое счетчику цикла при первом проходе цикла.

По – Синтаксическая связка для параметра **<Выражение 2>**

<Выражение 2> – максимальное значение счетчика цикла. Когда переменная **<Имя переменной>** становится больше чем **<Выражение 2>**, выполнение оператора цикла **Для** прекращается.

Цикл – операторы, следующие за ключевым словом **Цикл**, выполняются, пока значение переменной **<Имя переменной>** меньше или равно значению **<Выражение 2>**.

// Операторы – исполняемый оператор или последовательность таких операторов.

Прервать – позволяет прервать выполнение цикла в любой точке. После выполнения этого оператора управление передается оператору, следующему за ключевым словом **КонецЦикла**.

Продолжить – немедленно передает управление в начало цикла, где производится вычисление и проверка условий выполнения цикла. Операторы, следующие в теле цикла за ним, на данной итерации обхода не выполняются.

КонецЦикла – ключевое слово, которое завершает структуру оператора цикла.

Оператор цикла **Для каждого** предназначен для циклического обхода коллекций значений. При каждой итерации цикла возвращается новый элемент коллекции. Обход осуществляется до тех пор, пока не будут

перебраны все элементы коллекции, или может быть завершен досрочно при выполнении оператора *Прервать*.

Синтаксис:

```
Для каждого <Имя переменной 1> Из <Имя переменной 2> Цикл
// Операторы
[Прервать;]
// Операторы
[Продолжить;]
// Операторы
КонецЦикла;
```

Англоязычный синтаксис:

```
For each <Имя переменной 1> In <Имя переменной 2> Do
// Операторы
[Break;]
// Операторы
[Continue;]
// Операторы
EndDo;
```

Оператор цикла *Пока* предназначен для циклического повторения операторов, находящихся внутри конструкции *Цикл – КонецЦикла*. Цикл выполняется, пока логическое выражение равно *Истина*. Условие выполнения цикла всегда проверяется вначале, перед выполнением цикла.

Синтаксис:

```
Пока <Логическое выражение> Цикл
// Операторы
[Прервать;]
// Операторы
[Продолжить;]
// Операторы
КонецЦикла
```

Англоязычный синтаксис:

```
While <Логическое выражение> Do
// Операторы
[Break;]
// Операторы
[Continue;]
// Операторы
EndDo;
```

Параметры:

<*Логическое выражение*> – логическое выражение

Цикл – операторы, следующие за ключевым словом *Цикл*, выполняются, пока результат логического выражения равен *Истина*.

// **Операторы** – Исполняемый оператор или последовательность таких операторов.

Прервать – позволяет прервать выполнение цикла в любой точке. После выполнения этого оператора управление передается оператору, следующему за ключевым словом **КонецЦикла**.

Продолжить – немедленно передает управление в начало цикла, где производится вычисление и проверка условий выполнения цикла. Операторы, следующие в теле цикла за ним, на данной итерации обхода не выполняются.

КонецЦикла – ключевое слово, которое завершает структуру оператора цикла.

Процедуры и функции

Ключевое слово **Процедура** начинает секцию исходного текста, выполнение которого можно инициировать из любой точки программного модуля, просто указав **ИмяПроцедуры()** со списком параметров (если параметры не передаются, то круглые скобки, тем не менее, обязательны). Если в модуле приложения или общем программном модуле в теле описания процедуры использовано ключевое слово **Экспорт**, то это означает, что данная процедура является доступной из всех других программных модулей конфигурации.

При выполнении оператора **Возврат** процедура заканчивается и возвращает управление в точку вызова. Если в тексте процедуры не встретился оператор **Возврат**, то после выполнения последнего исполняемого оператора происходит выполнение неявного оператора **Возврат**. Конец программной секции процедуры определяется по оператору **КонецПроцедуры**.

Переменные, объявленные в теле процедуры в разделе Объявления локальных переменных, являются локальными переменными данной процедуры, поэтому доступны только в этой процедуре (за исключением случая передачи их как параметров при вызове других процедур, функций или методов).

Ключевые слова **Процедура**, **КонецПроцедуры** являются не операторами, а операторными скобками, поэтому не должны заканчиваться точкой с запятой (это может приводить к ошибкам выполнения модуля).

Синтаксис:

```
Процедура <ИмяПроцедуры>([[Знач] <Парам 1> [=<ДефЗнач>], ...  
, [Знач] <Парам N> [=<ДефЗнач>]]) [Экспорт]  
// Объявления локальных переменных;
```

```

// Операторы;
...
[Возврат;]
// Операторы;
...
КонецПроцедуры
    Англоязычный синтаксис:
Procedure <ИмяПроцедуры>([[Val] <Парам 1> [=<ДефЗнач>], ... , [Val]
<Парам N>[=<ДефЗнач>]]) [Export]
// Объявления локальных переменных;
// Операторы;
...
[Return;]
// Операторы;
...
EndProcedure

```

Ключевое слово **Функция** начинает секцию исходного текста функции, выполнение которой можно инициировать из любой точки программного модуля, просто указав **ИмяФункции** со списком параметров (если параметры не передаются, то круглые скобки, тем не менее, обязательны). Если в модуле приложения или общем программном модуле в теле описания функции использовано ключевое слово **Экспорт**, то это означает, что данная функция является доступной из всех других программных модулей конфигурации.

Выполнение функции заканчивается оператором **Возврат**. Функции отличаются от процедур только тем, что возвращают **ВозвращаемоеЗначение**. Конец программной секции функции определяется по оператору **КонецФункции**.

Вызов любой функции в тексте программного модуля можно записывать как вызов процедуры, т. е. в языке допускается не принимать от функции возвращаемое значение.

Если ключевое слово **Возврат** в теле функции не указано или строка модуля, его содержащая, не выполнена, то функция возвращает значение типа **Неопределено**.

Переменные, объявленные в теле функции в разделе объявления локальных переменных, являются локальными переменными данной функции, поэтому доступны только в этой функции (за исключением случая передачи их как параметров при вызове других процедур, функций или методов).

Ключевые слова **Функция**, **КонецФункции** являются не операторами, а операторными скобками, поэтому не должны заканчиваться точкой с запятой (это может приводить к ошибкам выполнения модуля).

Синтаксис:

```
Функция <ИмяФункции> ([[Знач] <Парам 1>[=<ДефЗнач>], ... , [Знач]
<Парам N>[=<ДефЗнач>]]) [Экспорт]
// Объявления локальных переменных;
// Операторы;
...
Возврат <Возвращаемое значение>;
// Операторы;
...
КонецФункции
```

Англоязычный синтаксис:

```
Function <ИмяФункции> ([[Val] <Парам 1>[=<ДефЗнач>], ... , [Val]
<Парам N>[=<ДефЗнач>]]) [Export]
// Объявления локальных переменных;
// Операторы;
...
Return <Возвращаемое значение>;
// Операторы;
...
EndFunction
```

Формы в 1С: Предприятие

Формы в 1С:Предприятие 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. Каждый объект конфигурации может использоваться для выполнения некоторых стандартных действий. Например, для любого справочника может потребоваться отображать список его элементов, отображать отдельные элементы справочника, отображать группу справочника, выбирать элементы и группы элементов из справочника. Для любого документа список таких действий будет гораздо меньше: просмотр списка документов, выбор из списка документов и просмотр отдельного документа.

Важной особенностью системы 1С:Предприятие 8 является механизм автогенерируемых форм. Этот механизм освобождает разработчика от необходимости создания всех возможных форм для каждого из объектов конфигурации. Разработчику достаточно добавить новый объект

конфигурации, а система сама генерирует в нужные моменты работы пользователя необходимые формы для отображения информации, содержащейся в этом объекте.

Таким образом, разработчику нужно создавать собственные формы объектов прикладного решения лишь в том случае, если они должны иметь отличия (другой дизайн или специфическое поведение) от форм, автоматически генерируемых системой.

Принадлежность формы тому или иному объекту конфигурации не определяет состав данных, которые отображаются в форме. То, что форма принадлежит, например, справочнику Номенклатура, позволяет назначить ее одной из основных форм для этого справочника, но никак не определяет, какие же именно данные будет отображать эта форма, и каково будет ее поведение.

Для того чтобы связать форму с данными, используются реквизиты формы, в которых указывается перечень данных, отображаемых формой. Все формы, сами по себе, имеют одинаковое поведение, независимо от того, какие данные они отображают. Однако один из реквизитов формы может быть назначен для нее основным (он выделяется жирным шрифтом), и в этом случае стандартное поведение формы и ее свойства будут дополнены в зависимости от того, какой тип имеет основной реквизит формы.

Управляемые формы

Управляемое приложение поддерживает следующие типы клиентов:

- толстый клиент (обычный и управляемый режим запуска)
- тонкий клиент
- веб-клиент

В управляемом приложении используются формы, построенные на новой технологии. Они называются Управляемые формы. Для облегчения перехода прежние формы (т.н. Обычные формы) также поддерживаются, но их функциональность не развивается и они доступны только в режиме запуска толстого клиента.

Основные отличия управляемых форм для разработчика:

- декларативное, а не «по пикселям» описание структуры, конкретное размещение элементов выполняется системой автоматически при отображении формы;
- вся функциональность формы описывается в виде реквизитов и команд; реквизиты – это данные, с которыми работает форма, а команды – выполняемые действия;

- форма выполняется и на сервере и на клиенте;
- в контексте клиента, недоступны практически все прикладные типы, и соответственно невозможно изменить данные в информационной базе;
- для каждого метода или переменной формы обязательно должна быть указана директива компиляции, определяющая, место выполнения (клиент или сервер) и доступ к контексту формы.

Основные директивы компиляции методов формы:

- &НаКлиенте;
- &НаСервере;
- &НаСервереБезКонтекста;
- &НаКлиентеНаСервереБезКонтекста.

Структура формы

Основная особенность форм заключается в том, что они не нарисованы разработчиком детально, «по пикселям». Форма в конфигурации представляет собой логическое описание состава формы. А конкретное размещение элементов выполняется системой автоматически при отображении формы.

Редактор формы используется для создания и редактирования форм объектов прикладного решения. Формы объектов используются системой для визуального отображения данных в процессе работы пользователя.

Любая форма представляет совокупность нескольких составляющих:

- элементов — объектов, определяющих визуальное представление формы и осуществляющих взаимодействие с пользователем,
- командного интерфейса — совокупности команд, отображаемых в форме;
- реквизитов — объектов, данные которых форма использует в своей работе.
- команд — действий, которые определены в данной конкретной форме,
- параметров — объектов, значения которых характеризуют саму форму, используются при ее создании и остаются постоянными в процессе «жизни» формы,
- модуля — программы на встроенном языке, отвечающей за работу с элементами и за обработку событий.

Редактор формы содержит несколько закладок, обеспечивающих редактирование всех составляющих формы.

В отдельном окне, в нижней части редактора, отображается внешний вид формы в режиме 1С:Предприятие.

Отображаемая часть формы (видимая пользователю) описывается как дерево, включающее элементы формы (рисунок 3).

Элементы могут представлять собой поля ввода, флажки, переключатели, кнопки и т. д. Кроме того, элемент может быть группой, включающей другие элементы. Группа может представляться как панель с рамкой, панель со страницами (закладками), собственно страница, командная панель. Помимо этого элемент может представлять собой таблицу, которая тоже включает элементы (колонки). Структура элементов описывает то, как будет выглядеть форма (рисунок 4).

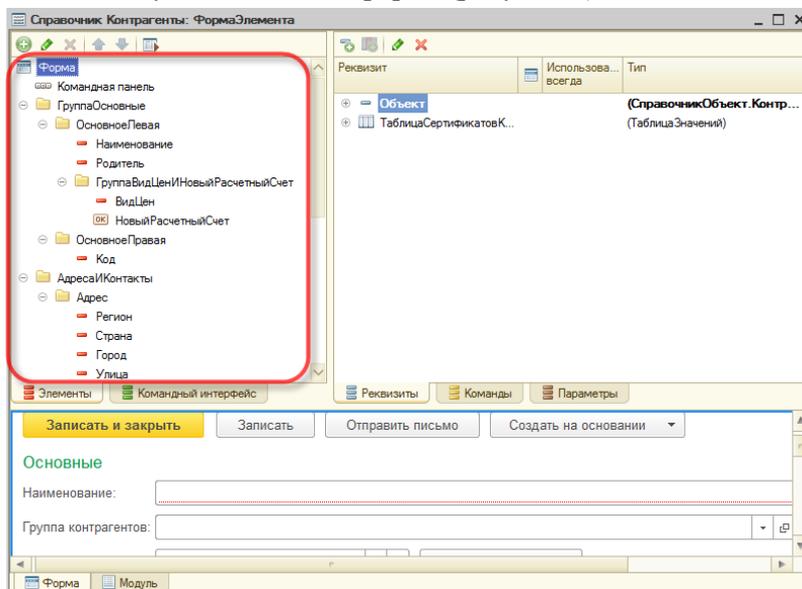


Рисунок 3 – Редактор форм для формы элемента Справочника

Редактор форм позволяет добавлять в форму специальные элементы, которые помогают придать форме собственный узнаваемый стиль. Редактор позволяет добавить в форму несколько элементов Группа — Страницы, каждая из которых может содержать несколько элементов Группа — Страница. Например, форма документа может содержать один элемент Группа — Страницы, которому подчинены несколько элементов Группа — Страница с заголовками Изображение, Характеристики и Описание (рисунок 5). Заголовок каждой группы — страницы отображается на отдельной закладке.

Разработчик имеет возможность задать режим отображения закладок: снизу или сверху. Редактор позволяет добавлять в форму различные

элементы. Добавлять элементы можно с помощью команды добавления или путем перетаскивания реквизитов формы в дерево элементов. Все элементы формы представляются в виде иерархической структуры, корнем которой является сама форма. Это позволяет быстро перемещаться к нужному элементу формы. Располагая элементы выше/ниже в дереве, подчиняя их другим элементам и задавая свойства элементов-групп можно задавать порядок, в котором пользователь будет обходить элементы управления формы при вводе и редактировании данных. В режиме 1С: Предприятие элементы формы будут обходиться в порядке их иерархии и в соответствии с тем, какой тип группировки выбран для групп: вертикальная или горизонтальная.

Разделители являются специальными элементами, с помощью которых возможно перераспределение пространства формы без изменения ее размеров. Платформа в режиме 1С:Предприятие самостоятельно добавляет эти элементы в форму. Разделитель обладает способностью «захватываться» мышью и перемещаться внутри формы в ее пределах с учетом возможности расположения других элементов и ориентации разделителя.

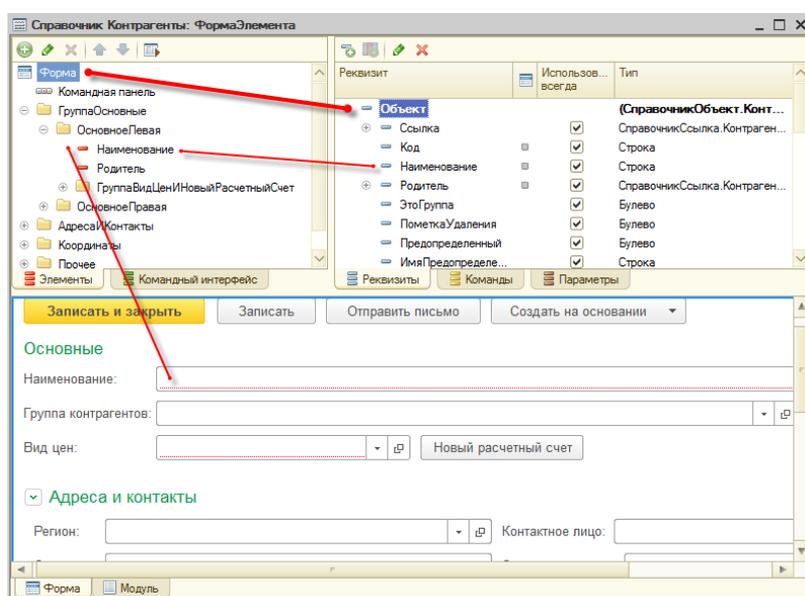


Рисунок 4 – Соответствие элементов, объектов и вида формы

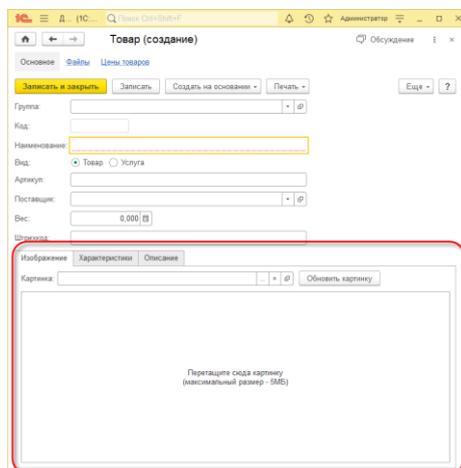


Рисунок 5 – Пример использования элементов настройки форм

Вся функциональность формы описывается в виде реквизитов и команд. Реквизиты — это данные, с которыми работает форма, а команды — выполняемые действия. Таким образом, разработчик в редакторе формы должен включить в форму необходимые реквизиты и команды, создать отображающие их элементы формы и, если необходимо, скомпоновать элементы в группы.

На основе этого логического описания система автоматически формирует внешний вид формы для отображения пользователю. При этом системой учитываются различные свойства отображаемых данных (например, тип), чтобы максимально удобно для пользователя расположить элементы формы. Разработчик может влиять на расположение элементов различными установками. Он может определять порядок элементов, указывать желаемую ширину и высоту. Однако это является только некоторой дополнительной информацией, помогающей системе отобразить форму.

В формах разработчик может использовать не только команды самой формы, но и глобальные команды, используемые в командном интерфейсе всей конфигурации (рисунок 5). Кроме того, реализована возможность создания параметризуемых команд, которые будут открывать другие формы с учетом конкретных данных текущей формы. Например, это может быть вызов отчета по остаткам на том складе, который выбран сейчас в форме расходной накладной.

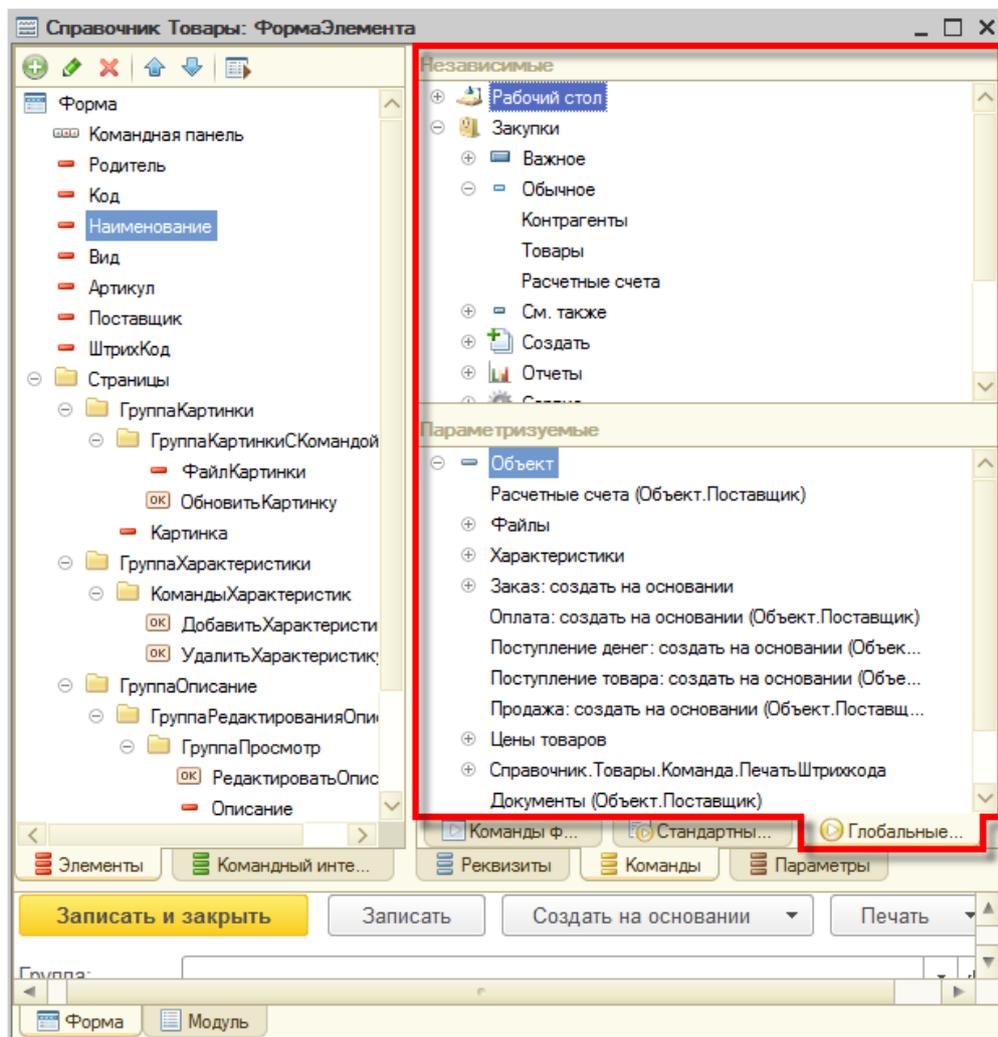


Рисунок 5 – Пример настройки команд формы

Пример

Рассмотрим настройку форм для элемента справочника. Для того чтобы пользователь мог просматривать и изменять данные, содержащиеся в справочнике, система поддерживает несколько форм представления справочника. Система может автоматически генерировать все нужные формы справочника. Наряду с этим разработчик имеет возможность создать собственные формы, которые система будет использовать вместо форм по умолчанию.

Для просмотра и изменения данных отдельных элементов справочника используется форма элемента. Как правило, она представляет данные в удобном для восприятия и редактирования виде. Форма элемента имеет

«Модуль формы». Данный модуль предназначен для того, чтобы обработать действия пользователя. Например, описать алгоритм реакции программы при нажатии кнопки. Или, например, в момент ввода в поле значения сразу же выполнить проверку на корректность. Кроме событий, связанных с элементами управления формы (кнопки, поля ввода) существуют события, связанные непосредственно с самой формой. Например, можно обработать событие открытия формы и провести некую начальную инициализацию. Также можно обработать событие закрытия формы и проверить, а все ли правильно ввел пользователь.

Программный код основной программы будет выполняться в момент инициализации формы, т.е. когда пользователь начинает ее открывать. Список событий управляемой формы виден также в списке свойств непосредственно для самой формы. Данный список вызывается в редакторе управляемых форм.

Обращение к элементам формы в модуле через объект *Элементы.<ИмяЭлементаФормы>*, а непосредственно к реквизитам справочника через объект *Объект.<ИмяРеквизитаСправочника>*. Обращение к элементам табличной части происходит через объявление и инициализацию переменной, содержащей текущие данные: *Элементы.<ИмяТабличнойЧасти>.ТекущиеДанные*. Дальнейшее обращение к элементам таблицы происходит следующим образом: *<ИмяПеременной>.<ИмяРеквизитаТабличнойЧасти>*.

Настройка печатных форм и других макетов

Документу и справочнику могут быть сопоставлены несколько макетов, содержащих данные, необходимые для обеспечения работы документа. Макеты могут использоваться для формирования печатных форм документа или для отображения дополнительной информации, имеющей отношение к документу.

Для создания печатной формы можно воспользоваться Конструктором печати (рисунок 6).

Печатная форма (рисунок 7) содержит: область заголовка, область шапки, область табличных частей и область подвала. В режиме редактирования можно добавить пользовательские области и изменить текущие.

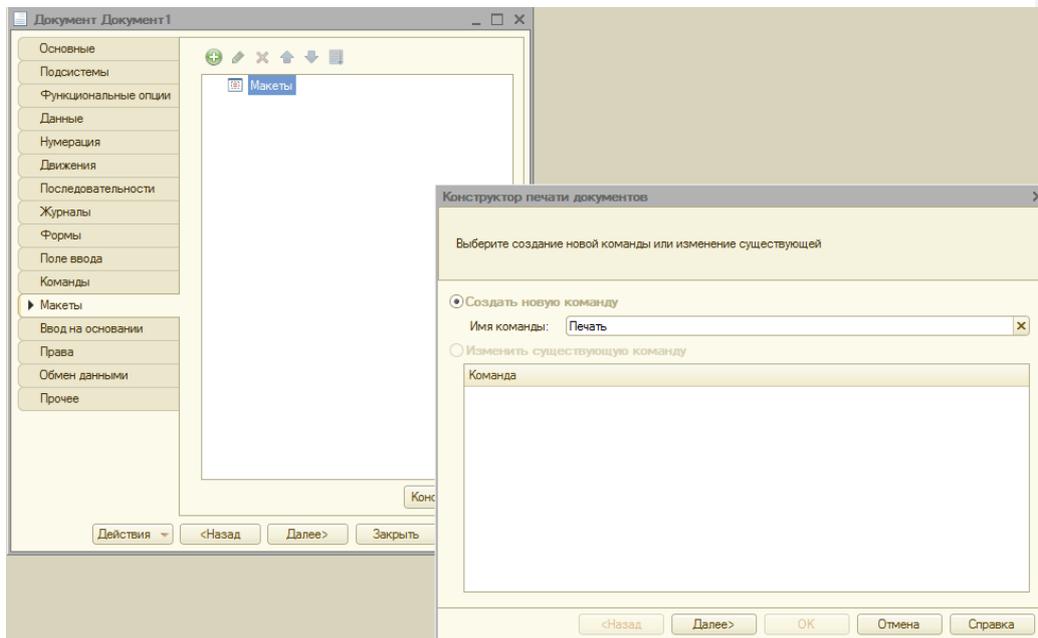


Рисунок 6 - Форма Конструктор печати

Документ ПриемНаРаботу: Печать		1	2	3	4	5	6	7	8	9	10	11	12	13
Заголовок	1													
	2	Прием на работу												
	3													
Шапка	4													
	5	Номер												
	6	Дата												
	7	Должность												
	8	Дата приема												
	9	Фамилия												
	10	Имя												
	11	Отчество												
	12	Дата рождения												
	13													
КонтактыШ	14													
	15													
	16	№	Тип связи	Номер										
Контакты	17	номерСтроки	<ТипСвязи>	<Номер>										
	18													
ЗарплатаШ	19													
	20	№	Тип выплаты	Сумма	Периодичность									
	21	номерСтроки	<ТипВыплаты>	<Сумма>	<Периодичность>									
	22													
Подвал	23													
	24	Фамилия	<Фамилия>											
	25	Дата приема	<ДатаПриема>											

Рисунок 7 – Редактирование печатной формы документа

Получение данных из справочника без запросов

Для получения данных из справочника по ссылке можно получить следующим образом. Для элемента формы (связанного с объектом справочника типа *СправочникСсылка*) добавить обработчик события *ОбработкаВыбора*. На сервере создается функция, которая обращается к содержимому объекта по ссылке:

`&НаСервереБезКонтекста`

Функция ПолучитьЗначение (ВыбранноеЗначение)
Возврат ВыбранноеЗначение.<ИмяРеквизита>;
КонецФункции

Получение данных из регистра сведений

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

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

Конструктор запроса

Для облегчения труда разработчика технологическая платформа содержит два специальных конструктора.

Конструктор запроса - это один из инструментов разработки. Он позволяет составить текст запроса на языке запросов исключительно визуальными средствами. С помощью кнопок «Далее» и «Назад» можно перемещаться по закладкам конструктора и указывать, какие данные должны присутствовать в результате запроса, как они связаны, сгруппированы, какие

итоги следует рассчитать, работать с временными таблицами, редактировать пакет запросов.

Результатом работы конструктора будет являться синтаксически правильный текст запроса. Таким образом, разработчик может составить работоспособный запрос, даже не владея синтаксисом языка запросов - необходимые синтаксические конструкции конструктор сгенерирует автоматически. Готовый текст запроса может быть сразу же вставлен в текст модуля или скопирован в буфер обмена.

Кроме этого конструктор запросов позволяет редактировать уже имеющийся в программе текст запроса. Для этого достаточно установить курсор внутри существующего текста запроса и вызвать конструктор. Имеющийся текст запроса будет проанализирован и представлен в конструкторе в виде соответствующих выбранных полей базы данных и набора заданных связей, группировок, условий и т.д.

Конструктор запроса с обработкой результата - это один из инструментов разработки. Он позволяет составить текст запроса и сформировать фрагмент программного кода, который исполняет запрос и выводит результаты в табличный документ или диаграмму.

На первом шаге своей работы конструктор предлагает выбрать один из возможных вариантов обработки результата запроса: просто обход результата для его дальнейшей программной обработки или вывод данных в табличный документ или диаграмму (рисунок 8).

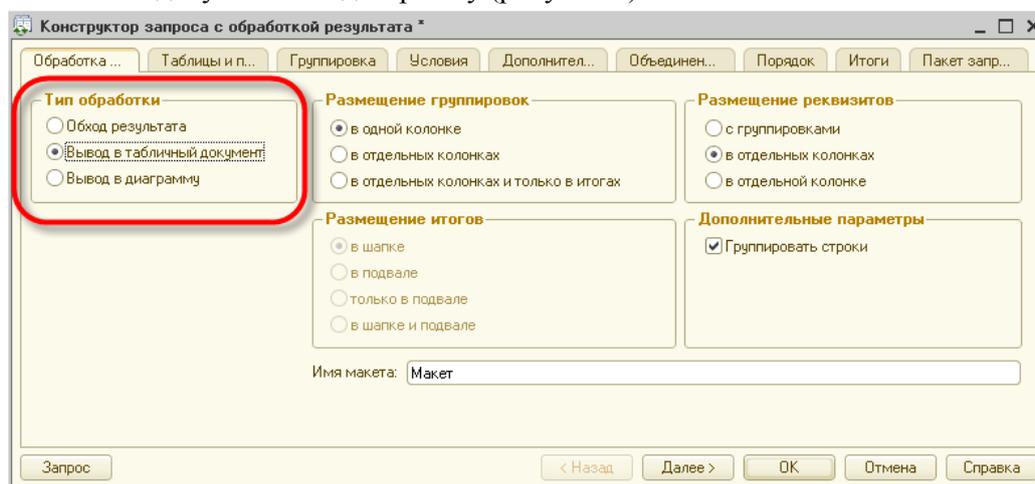


Рисунок 8 – Модуль Конструктора запроса с обработкой результата

Следующие шаги работы конструктора позволяют создать текст запроса к базе данных. Эти возможности аналогичны тем, которые предоставляет конструктор запроса.

Результатом работы конструктора запроса с обработкой результата является готовый фрагмент программного кода (рисунок 9) и, например, макет табличного документа. Разработчик может внести в них, при необходимости, свои изменения.

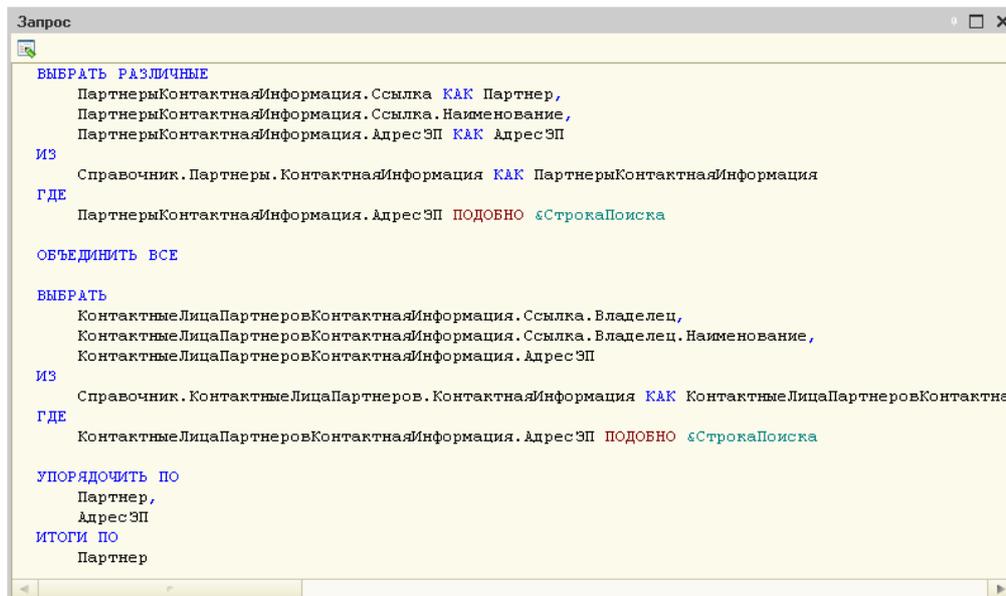


Рисунок 9 – Пример готового фрагмента кода, генерируемого конструктором запроса

Выполнение запроса 1С из программного кода осуществляется при помощи объекта встроенного языка «*Запрос*». Пример написания запроса к базе данных с использованием встроенного языка программирования:

```

Запрос = Новый Запрос;
Запрос.Текст =
    "ВЫБРАТЬ
    | Синоним.Ссылка КАК Ссылка
    |ИЗ
    | Справочник.Справочник1 КАК
Синоним";
Выборка = Запрос.Выполнить().Выбрать();
Пока Выборка.Следующий() Цикл
    // Вставить обработку выборки
Выборка.ДетальныеЗаписи;
КонецЦикла;

```

Метод «*Выполнить*» выполняет запрос, метод «*Выбрать*» возвращает значение типа «*ВыборкаИзРезультатаЗапроса*». Также можно использовать метод «*Выгрузить*», который возвращает таблицу значений.

Параметры запроса хранятся в свойстве *«Параметры»* (в данном случае это структура, поэтому все методы структуры тут применимы – вставить, удалить и т.д.).

Пример установки параметра *«Запрос.Параметры.Вставить»* (*«Справочник», СправочникСсылка*). В запросе обратиться к параметрам можно через амперсанд *«&Справочник»*. Ниже пример запроса с использованием параметров:

```
Запрос = Новый Запрос;
Запрос.Текст =
    "ВЫБРАТЬ
    |     Пользователи.Ссылка КАК Ссылка,
    |     Пользователи.Родитель КАК Родитель,
    |     Пользователи.Наименование КАК
Наименование
    | ИЗ
    |     Справочник.Пользователи КАК Пользователи
    | ГДЕ
    |     Пользователи.Ссылка = &Справочник";
Запрос.Параметры.Вставить ("Справочник", СправочникСсылка);
Выборка = Запрос.Выполнить().Выбрать();
Пока Выборка.Следующий() Цикл
    // Вставить обработку выборки Выборка.ДетальныеЗаписи
КонецЦикла;
```

Язык запросов предназначен только для чтения данных из базы, поэтому в нем отсутствуют аналоги таких операторов SQL, как INS ERT и UPDATE. Данные можно модифицировать только через объектную модель встроенного языка программирования 1С.

Контроль отрицательных остатков

При проведении некоторых видов документов (например, «Реализация товара») необходимо организовать контроль остатков. Если товара на остатках недостаточно, документ не проводится и выдается диагностическое сообщение.

Для реализации контроля остатков записи будут списываться, а далее будет проводиться проверка, образовались ли отрицательные остатки по товарам. При получении отрицательных остатков будет проводиться отмена проведения документа. Таким образом, требуется провести документ и проверить остатки в регистре накопления.

Процедура *ОбработкаПроведения()* расположена в «Модуле объекта» документа. Алгоритм будет включать в себя: получение списка записей из табличной части, формирование движения по регистру, получение остатков

из регистра. Для получения данных из табличной части документа и регистра данных используются запросы. Для формирования запроса можно использовать «Конструктор запроса».

Рассмотрим модификацию процедуры **ОбработкаПроведения()** для документа «СписаниеТовараСоСклада» пошагово.

2. Вызвать модуль объекта документа «СписаниеТовараСоСклада»
3. В обработке проведения создать запрос на получение списка товаров из табличной части документа «СписаниеТовараСоСклада», используя конструктор запроса:
 - a. На вкладке Таблицы и поля перенести в колонку таблицы табличную часть документа «Товары», а в колонку поля – наименование (СписаниеТовараСоСкладаТовары.Наименование) и количество (СписаниеТовараСоСкладаТовары.Количество).
 - b. На вкладке Группировка в групповое поле перемести «СписаниеТовараСоСкладаТовары.Наименование», а в суммируемое поле – «СписаниеТовараСоСкладаТовары.Количество» – с функцией *Сумма*.
 - c. На вкладке Условия перенести вправо поле «Ссылка».
 - d. На вкладке Дополнительно выбрать тип запроса Создание временной таблицы, указав имя временной таблицы: «ДокТЧ».
 - e. На вкладке Пакет запросов переименовать «Запрос пакета 1» в «ДокТЧ» и добавить новый запрос. В новом запросе из временной таблицы ДокТЧ перенести в колонку поля – «ДокТЧ.Наименование» и «ДокТЧ.Количество». Общий вид получения списка товаров будет иметь вид:

```
//Получение списка товаров из документа
Запрос= Новый Запрос;
Запрос.МенеджерВременныхТаблиц = Новый
МенеджерВременныхТаблиц;
Запрос.Текст = "ВЫБРАТЬ
| СписаниеТовараСоСкладаТовары.Наименование,
| СУММА (СписаниеТовараСоСкладаТовары.Количество) КАК
Количество
| ПОМЕСТИТЬ ДокТЧ
| ИЗ
```

```

| Документ.СписаниеТовараСоСклада.Товары КАК
СписаниеТовараСоСкладаТовары
| ГДЕ
| СписаниеТовараСоСкладаТовары.Ссылка = &Ссылка
|
| СГРУППИРОВАТЬ ПО
| СписаниеТовараСоСкладаТовары.Наименование
| ;
|
| ВЫБРАТЬ
| ДокТЧ.Наименование,
| ДокТЧ.Количество
| ИЗ
| ДокТЧ КАК ДокТЧ";
Запрос.УстановитьПараметр ("Ссылка", Ссылка);
РезультатЗапроса=Запрос.Выполнить ();

```

4. Для Формирования движения по регистру добавить обход выборки:

```

//Формирование движения
// регистр Склад Расход
Движения.Склад.Записывать = Истина;
Выборка = РезультатЗапроса.Выбрать ();
Пока Выборка.Следующий () Цикл
    Движение = Движения.Склад.Добавить ();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Номенклатура = Выборка.Наименование;
    Движение.Количество = Выборка.Количество;
КонецЦикла;
Движения.Записать ();

```

5. Для получения остатков из регистра построить запрос, получающий из регистра сведений СкладОстатки поля Товар и Количество. При этом на вкладке Условия Конструктора запросов добавить условие СкладОстатки.Количество<0:

```

//Получение остатков из регистра
Запрос.Текст ="ВЫБРАТЬ
| СкладОстатки.Номенклатура,
| СкладОстатки.КоличествоОстаток
| ИЗ
|
РегистрНакопления.Склад.Остатки(,Номенклатура В (ВЫБРАТЬ
ДокТЧ.Наименование ИЗ ДокТЧ КАК ДокТЧ)) КАК СкладОстатки
| ГДЕ

```

```

        | СкладОстатки.КоличествоОстаток < 0";
    РезультатЗапроса = Запрос.Выполнить ();
    б. Затем в цикле производится обход результатов запроса и
    выводится сообщение:
//Обработка результата остатков из регистра
    Выборка = РезультатЗапроса.Выбрать ();
    Пока Выборка.Следующий () Цикл
        Сообщение=Новый СообщениеПользователю;
        Сообщение.Текст ="Не хватает товара "+
    Выборка.Номенклатура "+", после проведения документа остаток
    составит "+ Выборка.КоличествоОстаток;
        Сообщение.Сообщить ();
        Отказ =Истина;
    КонецЦикла;

```

Программное создание элемента справочника, внесение изменений в элемент справочника

При сохранении элементов справочника или проведение документов может возникнуть необходимость перемещения или создания нового элемента другого справочника.

Элемент справочника для изменения можно найти по реквизиту, наименованию или коду. Для этого можно использовать запросы или методы объекта Справочники **НайтиПоКоду("<Код>")**, **НайтиПоНаименованию("<Наименование>")**, **НайтиПоРеквизиту("<Реквизит>", <ЗначениеРеквизита>)**. Данные методы возвращают ссылку на запись справочника. Например:

```
СсылкаНаКлиента = Справочники.Клиенты.НайтиПоКоду("000000001");
```

Для работы с полученной записью требуется по ссылке получить объект - **ПолучитьОбъект()**. В полученном объекте можно изменять значения реквизитов. После изменения экземпляра справочника требуется записать изменения на сервер. Например:

```
СотрудникОбъект = СправочникСсылка.ПолучитьОбъект ();
СотрудникОбъект.Родитель = Справочники.Сотрудники.Работающие;
СотрудникОбъект.Записать ();
```

Создание элемента справочника производится через новый объект типа справочники. При этом прямое объявление не обязательно. Например:

```
Спр = Справочники.Сотрудники;
НовЭл = Спр.СоздатьЭлемент ();
```

Дальнейшая работа с объектом аналогична работе с имеющимся справочником.

Программное создание и проведение документа

Создание нового документа происходит с использованием методов класса Документы: *СоздатьДокумент()*. Например:

```
ОбъектДокумента = Документы.ОказаниеУслуги.СоздатьДокумент ();  
ОбъектДокумента.Дата = ТекущаяДата ();  
ОбъектДокумента.Клиент = Клиент;
```

При программном создании документа и справочника важно помнить о заполнении стандартных реквизитов: дата, наименование и т.д.

При сохранении созданного или измененного документа на сервере следует учитывать свойство проводимости документа. Возможно сохранение без проведения или с отменой проведения, это устанавливается в параметрах метода *Записать()* через *РежимЗаписиДокумента*. Например, *ОбъектДокумента.Записать(РежимЗаписиДокумента.Проведение);*.

Если в обработке проведения прописано создание документа именно этого наименования, при проведении документа *ОбработкаПроведения()* выполняется для вновь созданного. А выполнение текущей обработки приостанавливается – как в рекурсивных функциях.

Варианты работы

Вариант №1. Библиотека

Библиотека имеет три абонементов (детский и два взрослых) и хранилище книг. Библиотека имеет сквозную систему читательских билетов, при этом детский абонемент не обслуживает читателей старше 18 лет, а во взрослом – не обслуживаются лица младше 14.

Библиотечный фонд делится на книги, которые можно выдавать на руки и те, которыми можно пользоваться только в читальном зале. Для каждой книги библиотечного фонда ведется учет выдачи и состояния книги. Если книга приходит в негодность, то её отправляют на реставрацию или утилизируют. На реставрацию отправляют книги с небольшой степенью повреждения или представляющие собой особую ценность.

Вариант №2. Спортзал

Спортивный центр имеет бассейн, зал с тренажерами и несколько залов для занятий. Кроме постоянных сотрудников занятия могут вести приглашенные специалисты. В центре проводятся несколько видов групповых и индивидуальных занятий, кроме того возможно посещение бассейна или тренажерного зала.

Для удобства работы организована работа с абонементными, как по количеству занятий, так и на определенный срок (месяц, полгода, год). Стоимость абонемента зависит от количества занятий / срока и видов занятий. Для постоянных клиентов предоставляется система скидок.

Вариант №3. Прокат велосипедов

Предприятие занимается прокатом велосипедов, роликовых коньков, самокатов и других средств передвижения. Оно имеет филиалы на территории города, поэтому велосипед взятый на прокат в одном отделении можно вернуть в другой.

На каждого клиента заводится запись в книге учета, в которой отражаются сроки аренды и виды техники. Для постоянных клиентов возможна длительная аренда – на несколько дней. В случае, если техника возвращена с повреждениями заводится акт, на основании которого клиенту выставляется счет. Клиенты, не оплатившие счет аренды или ремонта, в дальнейшем не обслуживаются.

Вариант №4. Зоомагазин

Ассортимент магазина представлен различными видами живого товара, кормов и аксессуаров. Один из отделов зоомагазина является ветеринарной аптекой – сотрудники этого отдела могут предоставить консультацию и провести осмотр животного за отдельную плату.

Магазин имеет систему накопительных скидок для клиентов, привязанных к номеру карты. Периодически в магазине проводятся акции по распродаже кормов или игрушек для животных. Распродажи не касаются ассортимента ветеринарной аптеки – на специализированное ветеринарное питание скидка не распространяется.

Вариант №5. Учебный центр

Учебный центр проводит занятия для детей разного возраста. Для каждой возрастной категории представлен отдельный список занятий. Посещение некоторых занятий, например танцев, может начинаться только с определенного возраста.

Учебный центр ведет регистрацию детей на занятия, составление расписания для преподавателей. Расписание зависит от расписания преподавателя и занятости определенного вида помещений. В расписании учитывается время на уборку в помещениях. Личная карточка ребенка предоставляет расписание его занятий и посещений. Количество занятий на одного ребенка не может превышать 10 в неделю и 2 в день.

Вариант №6. Рабочее место администратора стоматологического кабинета

Администратор стоматологического кабинета ведет запись пациентов на прием к определенному врачу, расчет стоимости услуг и карточки пациентов. При первичном приеме администратор заполняет личную карту пациента.

В стоматологическом кабинете посменно работают три стоматолога-терапевта и один хирург. Запись на прием зависит от проводимых манипуляций и выбранного врача.

В обязанности администратора входит приглашение на прием пациентов, которые не посещали стоматолога более 8 месяцев.

Вариант №7. Автошкола

В автошколе имеется парк автотранспорта. За автотранспортом закреплено не более 3 инструкторов. У каждого инструктора не более 3 единиц возможного автотранспорта. В зависимости от вида обучения ученикам может быть предоставлен определенный вид транспорта (механика, автомат и т.д.) на базовое количество часов. Дополнительные часы оплачиваются отдельно и зависят от вида обучения. Так же автошкола проводит индивидуальные занятия и занятия по экстремальному вождению.

Ведется расписание инструкторов, учитывающее занятость автотранспортных средств и расписание теоретических занятий учеников.

Вариант №8. Рабочее место турагента

Турагент работает с несколькими туроператорами – бронирует и частично оплачивает определенное количество туров. Турагент должен продать эти туры, иначе понесет убытки. Поэтому за две недели до начала тура непроданные туры по минимальной цене – без надбавки. Количество заказанных туров должно соответствовать туристическим сезонам и спросу на направления (может основываться на статистике предыдущих продаж).

Турагент должен предложить клиенту туры, соответствующие его запросам. Постоянным клиентам могут быть предоставлены дополнительные скидки.

Вариант №9. Ателье

Ателье предоставляет услуги по пошиву и ремонту одежды. Кроме того, в ателье представлены ткани и фурнитура на продажу. Пошив одежды может осуществляться как из ткани клиента, так и из купленной ткани. Для сложных моделей существует услуга подбора ткани. Цена услуги зависит от количества и сложности операций, сложности выкройки и ткани. Ателье ведет каталог отшитых моделей, но также предоставляет услугу конструирования выкройки любой сложности.

Ателье ведет учет постоянных клиентов, предлагает накопительные скидки и оповещает о поступлении ткани для заказанной вещи.

Вариант №10. Химчистка

Химчистка работает с юридическими лицами – отелями, гостиницами, хостелами, больницами. Для клиентов отелей предоставляется услуга химчистки одежды (верхней или из сложных тканей). Оплата химчистки одежды клиентов проводится отдельно. На цену химчистки влияет сложность ткани, сложность загрязнения и общий вес. При приеме заказа составляется опись и сортировка по ткани и загрязнениям.

Оплата организацией может осуществляться авансовой, за каждый заказ или в конце месяца по факту. Вид оплаты выбирается в договоре и может быть пересмотрено раз в год.

Практическая работа №5. Проектирование программного решения и разработка объектов конфигурации

Время выполнения: 6 академических часа (5 баллов)

Задание 1. Согласно варианту выявить основные процессы и представить диаграмму прецедентов.

Задание 2. На основе анализа процесса разработать систему объектов конфигурации.

Задание 3. Построить ER-диаграмму.

Примечание [01]: Что такое диаграмма прецедентов и ER-диаграмма в МУ ни слова, а как их строить?

Практическая работа №6. Настройка прав пользователей и командного интерфейса конфигурации

Время выполнения: 2 акад. часа (3 балла)

Задание 1. Согласно диаграмме прецедентов определить объекты роли в конфигурации.

Задание 2. Создать подсистемы и определить их состав.

Задание 3. Настроить командный интерфейс конфигурации по ролям.

Практическая работа №7. Решение задач оперативного учета

Время выполнения: 6 акад. часов (10 баллов)

Задание 1. Контроль остатков

Для корректной работы системы реализовать задачу контроля остатков при проведении документа, так чтобы:

Вариант	Задание 1.а	Задание 1.б
1	книга, находящаяся на руках у читателя, не выдавалась	списанные книги не выдавались
2	при составлении расписания в каждом зале была одна группа (не касается бассейна)	при расторжении договора с клиентом, нельзя было расторгнуть не заключенный договор
3	количество средств передвижения на точке не было отрицательным	вернуть средство передвижения мог только тот клиент, который взял
4	количество товара на складе не было отрицательным	при регистрации смерти животного, отслеживать, чтобы оно было в базе
5	количество занятий ребенка не превышало 10 в неделю	при составлении расписания в каждом кабинете была одна группа
6	при списании материалов, их количество не было отрицательным	при записи клиента не возникало накладок (один врач – одно время – один клиент – один кабинет)
7	за инструктором было закреплено не более 3 ед. автотранспорта	при составлении расписания не бронировать на одно и то же время одну машину дважды
8	количество туров в наличии не было отрицательным	соблюдались сроки на оформление визы
9	при продаже или оказании услуги, количество материала на складе не было отрицательным	не выдавался уже выданный товар
10	не регистрировать акт выдачи на непринятый или уже выданный заказ	при погашении задолженности не принимать большие суммы

Задание 2. Регистры сведений и накоплений

Для автоматизации работы, обеспечьте заполнение данных в форме из регистра сведений или регистра накоплений

Вариант	Задание 2.а	Задание 2.б
1	при выборе книги, выставляется её состояние	при выборе читателя высвечивается количество книг на руках
2	при вводе клиента, выводится номер текущего абонента	при выборе клиента выводится текущее количество занятий, использованных по текущему абоненту
3	при выборе клиента, проставляется максимальное количество дней аренды	при выборе клиента, автоматически выводится его финансовая задолженность
4	при выборе услуги или товара	при вводе номера скидочной карты

	автоматически проставляется цена	проставляется текущая накопленная сумма и данные клиента
5	при составлении договора, при выборе занятия, в договоре автоматически проставляется ответственный руководитель занятий	при составлении счета на оплату выставляется количество занятий в этом месяце всего и для конкретного ребенка
6	для пациента автоматически проставляется последний принимающий врач	для пациента автоматически выставляется номер посещения
7	для ученика автоматически выставляется текущая программа	для ученика автоматически выставляется количество накатанных часов
8	для клиента автоматически выводится накопленная сумма заказов на текущую дату	для направления автоматически выставляется требования по визе и текущий сезон при заключении договоров
9	при выборе операций и типа ткани сложность должна проставляться автоматически	для клиента автоматически выводится накопленная сумма заказов на текущую дату
10	при выставлении счета на оплату в конце месяца, автоматически заполнять сумму	при выборе типа ткани сложность должна проставляться автоматически

Практическая работа №8. Программное создание объектов конфигурации

Время выполнения: 6 акад. часов (11 баллов)

Задание 1. Программное перемещение элементов справочников

Для корректной работы системы создать обработчик, так чтобы:

Вариант	Задание
1	в иерархическом справочнике Книги при проведении документов «Акт о списании», «Выдача книги», «Возврат книги», «Акт о передаче на реставрацию» элемент справочника перемещался в нужную группу (Книги на руках, Списанные, Реставрация)
2	в иерархическом справочнике Клиенты при проведении документа «Договор с клиентом» элемент справочника перемещался в нужную группу, в зависимости от статуса (Корпоративные клиенты, Постоянные клиенты, Новые клиенты)
3	в иерархическом справочнике Транспортные средства при проведении документов об аренде и возврате элемент справочника перемещался в нужную группу, в зависимости от состояния (С повреждениями, Неисправные, Без повреждений)
4	в иерархическом справочнике Клиенты при проведении документа «Договор об оказании услуг» элемент справочника перемещался в нужную группу, в зависимости от статуса (Корпоративные клиенты, Постоянные клиенты, Новые клиенты)
5	в иерархическом справочнике Учащиеся при проведении документа Отчисление, Перевод в другую группу элемент справочника перемещался в нужную группу (Отчисленные, Старшие дети, Младшие дети)
6	в иерархическом справочнике Клиенты при проведении документа «Договор на оказание услуг» элемент справочника перемещался в нужную группу, в зависимости от статуса (Корпоративные клиенты, Постоянные клиенты, Новые клиенты)
7	в иерархическом справочнике Учащиеся при проведении документа Отчисление, Договор с курсантом элемент справочника перемещался в нужную группу (Отчисленные, категория А, категория В, категория С и т.д.)
8	в иерархическом справочнике Клиенты при проведении документа «Бронирование тура» элемент справочника перемещался в нужную группу, в зависимости от статуса (Корпоративные клиенты, Постоянные клиенты, Новые клиенты)
9	в иерархическом справочнике Клиенты при проведении документа «Договор об оказании услуг» элемент справочника перемещался в нужную группу, в зависимости от статуса (Корпоративные клиенты, Постоянные клиенты, Новые клиенты)
10	в иерархическом справочнике Клиенты при проведении документа «Договор на оказание услуги» элемент справочника перемещался в нужную группу, в зависимости от статуса (Отели, Хостелы, Больницы)

Задание 2. Программное создание и проведение документа

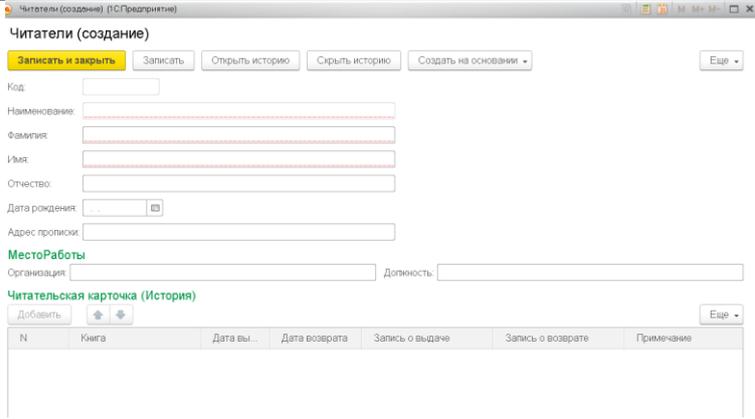
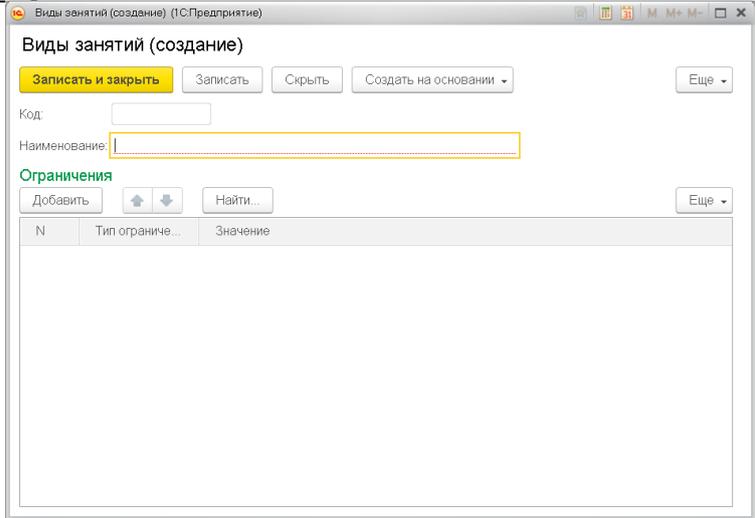
Для корректной работы системы создать обработчик, так чтобы:

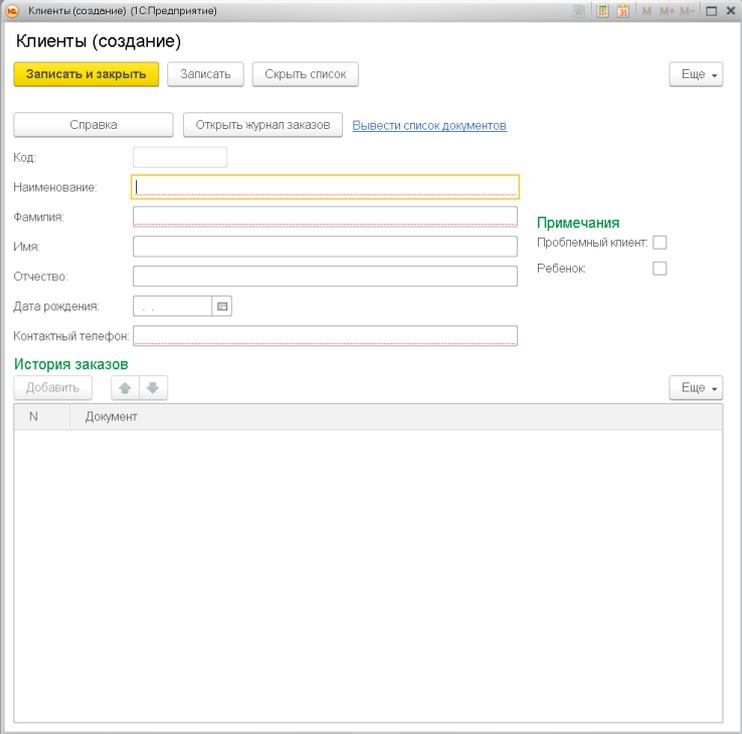
Вариант	Задание
1	В случае возврата книги с сильными повреждениями, автоматически формировался акт о передаче на реставрацию или о списании книги
2	В случае бронирования зала создавались документы для регистрации объявленного занятия
3	В случае возврата средства с повреждениями автоматически формировался акт повреждений
4	При создании чека, если у клиента достаточная сумма покупок, формировался документ на регистрацию накопительной карты
5	При проведении перевода в другую группу, формировался документ об оказании услуг
6	В случае записи к врачу формировался документ об оказании услуг
7	При проведении договора с курсантом формировался документ на бронирование времени
8	При бронировании тура создавался документ на продажу тура
9	При оформлении договора на оказание услуг, формировался чек на оплату материалов и услуг
10	При проведении акта приемки, если заказ включает личные вещи физ.лиц, автоматически формировался ордер для физического лица

Практическая работа №9. Управляемые формы

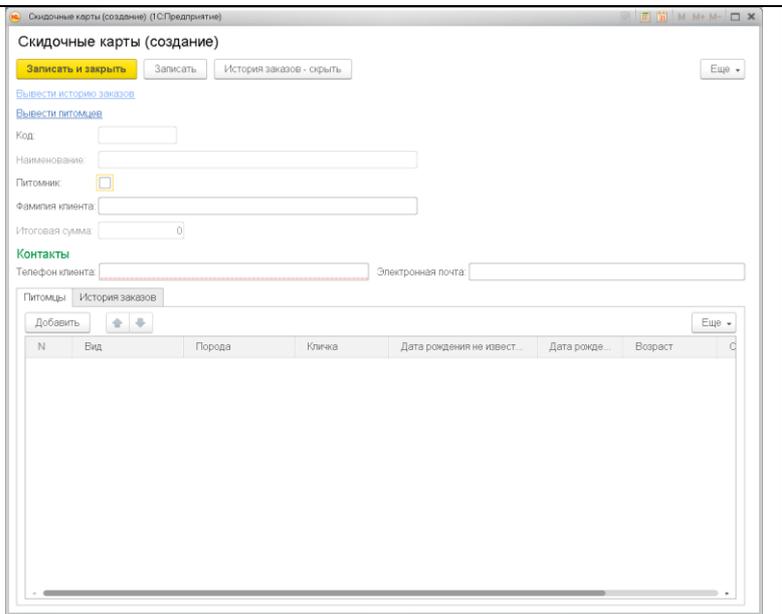
Время выполнения: 6 акад. часов (5 баллов)

Задание 1. Создать форму справочника, согласно требованиям, представленным ниже.

Вар.	Название Справочника	Вид Формы
1	Читатели	 <p>1. Реквизит Наименование должен заполняться автоматически, как Фамилия и инициалы читателя</p> <p>2. При вводе Даты рождения рядом должен появляться Абонемент (младше 14 – детский)</p> <p>3. Вкладка создать на основании позволяет создать документы Возврат книги и Выдача книги</p> <p>4. При проведении документов, данных о них отображаются в читательской карточке.</p> <p>5. Читательскую карточку (историю) можно скрыть или открыть при нажатии на кнопки. Изменить читательскую историю из справочника нельзя</p>
2	ВидыЗанятий	 <p>1. Табличная часть справочника содержит ограничения на виды занятий – это могут быть ограничения по возрасту, полу, росту или</p>

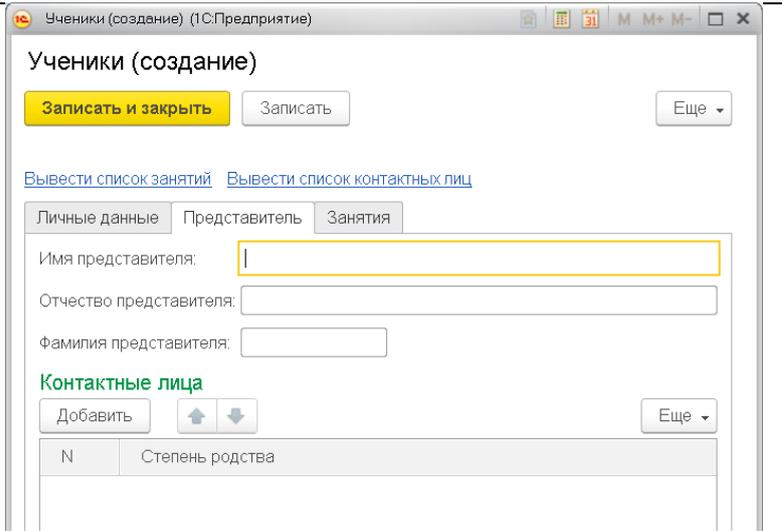
		<p>заболеваниям. В зависимости от выбранного типа ограничения (перечисление), выбираются значения (из справочника, перечисления или записывается число) (Самый простой вариант реализации – через составной тип данных, но нет защиты от пользователя)</p> <p>2. Табличная часть ограничения становится невидимой при нажатии на кнопку Скрыть. После этого кнопка скрыть меняется на кнопку Показать табличную часть.</p> <p>3. Из справочника можно создать бронирование классов под вид занятия</p>
3	Клиенты	 <p>1. Наименование должно заполняться автоматически</p> <p>2. Если возраст клиента меньше 18, то галочка отмечается автоматически</p> <p>3. История заказов заполняется автоматически при проведении документов</p> <p>4. Табличная часть ограничения становится невидимой при нажатии на кнопку Скрыть. После этого кнопка скрыть меняется на кнопку Показать табличную часть.</p>

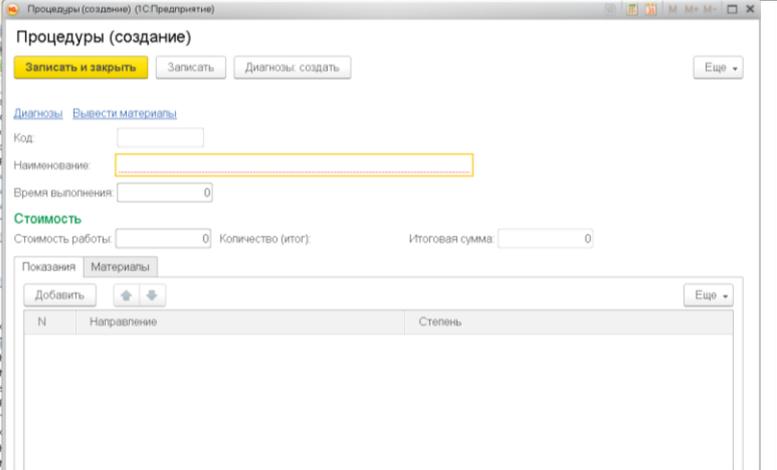
4 Скидочные карты



1. При нажатии на кнопки «Вывести ...» должны выводиться списки из табличных частей справочника
2. Данные в табличной части История заказов не должны изменяться из справочника – они должны заноситься и удаляться при проведении соответствующих документов. При этом итоговая сумма должна заполняться из суммы документов (не является реквизитом формы)
3. Если клиент является питомником, то наименование поля Фамилия меняется на название питомника. Название питомника является обязательным полем
4. Табличная часть История заказов становится невидимой при нажатии на кнопку Скрыть. После этого кнопка скрыть меняется на кнопку Показать табличную часть.

5 Ученики



		<ol style="list-style-type: none"> 1. При нажатии на кнопки «Вывести список...» должны выводиться списки из табличных частей справочника 2. Данные в табличной части занятия не должны изменяться из справочника – они должны заноситься и удаляться при проведении соответствующих документов. 3. На вкладке личные данные должно быть поле с вычисленным возрастом и указанной возрастной группой (этих данных не должно быть в реквизитах справочника) 4. Наименование должно заполняться автоматически
6	Процедуры	 <ol style="list-style-type: none"> 1. При нажатии на кнопки «Вывести список...» должны выводиться списки из табличных частей справочника 2. При нажатии на Диагнозы должен открываться справочник Диагнозы 3. Сумма по табличной части материалов должны отображаться в поле Количество (итог). Итоговая сумма должна состоять из Стоимости работы и стоимости материалов 4. Стоимость работы рассчитывается в зависимости от времени выполнения – после заполнения поля время выполнения, появляется поле сложность выполнения и проводится расчет. 5. При выборе показаний в выборе направления может быть выбран диагноз или записаны данные в виде строки (например «Гигиенические требования» или «Желание клиента») (Самый простой вариант реализации – через составной тип данных, но нет защиты от пользователя)

7 Клиент

Клиент (создание) * (ПСПредприятие)

Клиент (создание) *

Записать и закрыть Записать Отчислить курсанта

Еще ▾

Вывести список программ

Код:

Личные данные

Фамилия:

Имя:

Отчество:

Дата рождения:

Категория прав:

Программы Медицинские ограничения

Добавить

N	Наименование	Часов в плане	Часы факт	Теор...	Автодр...	Гор...
1				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Добавить

N	Программа	Дата	Часы факт
1			

Еще ▾

1. Поля в табличной части Программы и ФактическиеЧасы не должны редактироваться из формы – они должны заполняться из соответствующих документов.
2. Для определенных программ наличие форм контроля может отличаться, это должно отражаться на форме
3. Часы факт в табличной части Программы заполняются как сумма часов по программе в ФактическихЧасах
4. Кнопка Отчислить курсанта должна открывать и проводить документ отчисления.
5. Наименование должно заполняться автоматически
6. В Программе табл. часы Фактические Часы должно отображаться как название программы из справочника так и запись строкового типа

8 Туры

Туры (создание) (ПСПредприятие)

Туры (создание)

Записать и закрыть Записать Покупка тура

Еще ▾

Туроператоры Гостиницы создать

Код:

Наименование:

Туроператор:

Отель

Гостиница:

Размещение:

Питание:

Трансфер

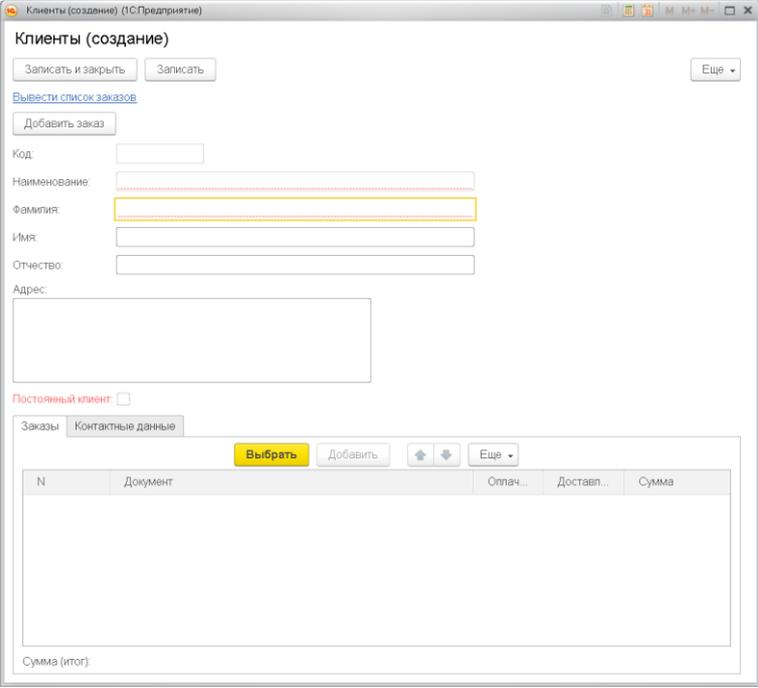
Трансфер:

Добавить

N

Еще ▾

1. Отель и Трансфер можно свернуть

		<p>2. В случае, если Трансфер не выбран, то Табличная часть не отображается</p> <p>3. При нажатии на кнопку Покупка тура открывается документ с частичным заполнением</p> <p>4. При выборе гостиницы значения размещения и питания меняется</p>										
9	Клиенты	 <p>Клиенты (создание)</p> <p>Записать и закрыть Записать Еще ▾</p> <p>Вывести список заказов</p> <p>Добавить заказ</p> <p>Код: <input type="text"/></p> <p>Наименование: <input type="text"/></p> <p>Фамилия: <input type="text"/></p> <p>Имя: <input type="text"/></p> <p>Отчество: <input type="text"/></p> <p>Адрес: <input type="text"/></p> <p>Постоянный клиент: <input type="checkbox"/></p> <p>Заказы Контактные данные</p> <p>Выбрать Добавить ⬆ ⬇ Еще ▾</p> <table border="1"> <thead> <tr> <th>N</th> <th>Документ</th> <th>Оплат...</th> <th>Доставл...</th> <th>Сумма</th> </tr> </thead> <tbody> <tr> <td colspan="5"> </td> </tr> </tbody> </table> <p>Сумма (итог):</p> <p>1. Наименование должно заполняться автоматически</p> <p>2. Поле Постоянный клиент заполняется на основе данных документов об оказании услуг</p> <p>3. Табличная часть заказы заполняется данными документов, при их проведении (оплачен и доставлен – булевы)</p> <p>4. При нажатии на кнопку ДобавитьЗаказ должен формироваться документы заказа</p>	N	Документ	Оплат...	Доставл...	Сумма					
N	Документ	Оплат...	Доставл...	Сумма								

10

Организации

1. При выборе типа организации наличие физических лиц должно проставляться автоматически. В случае, если физ.лица не предусмотрены изменение поля блокировать.
2. При нажатии вывести список заказов должна выводиться табличная часть Заказы.
3. В табличную часть заказы данные должны попадать только при проведении документов
4. Табличная часть Заказы становится невидимой при нажатии на кнопку Скрыть. После этого кнопка скрыть меняется на кнопку Показать табличную часть.
5. Предусмотреть вывод документов по организации при нажатии на кнопку Вывести документы (в меню «Еще»)

Задание 2. Автоматизировать все расчеты и все получения данных из справочников, регистров накопления и регистров сведений в формах документов.

Практическая работа №10. Печатные формы и отчеты

Время выполнения: 4 акад. часов (5 баллов)

Задание 1. Создайте отчеты, отражающие работу системы.

Вариант	В виде таблицы	В виде списка	В виде диаграммы
1	Сводная ведомость количества книг на абонементах по состояниям (на руках, на реставрации, в наличии)	Списки читателей по абонементом Списки книг, сгруппированных по абонементами и авторам Списки книг, сгруппированных по жанрам (в зависимости от абонемент)	Сводная ведомость количества книг на абонементах по состояниям (на руках, на реставрации, в наличии) Количество книг по абонементам Количество читателей по абонементам
2	Занятость залов по дням Сводная таблица количества сотрудников по квалификации	Списки клиентов Списки клиентов по абонементам	Количество абонементов по видам занятий Количество предоставленных скидок Количество абонементов Количество клиентов у тренера
3	Сводная ведомость количества средств передвижения по точкам Расстояние между точками	Список клиентов Список средств передвижения по точкам Список средств передвижения по виду	Сводная ведомость количества средств передвижения по точкам Количество транспортных средств по видам Стоимость аренды по видам транспортных средств
4	Сводная ведомость выручки в отделах по видам товаров и услуг Сводная ведомость количества товаров в отделах	Сводная ведомость товаров Список товаров Список услуг	Количество товаров по видам Сводная ведомость выручки в отделах по видам товаров и услуг Скидки по картам и видам товаров
5	Сводный список занятий по возрастным группам и преподавателям Количество учащихся по видам занятий и возрастным группам Сводная ведомость успеваемости Занятость залов по дням	Список учащихся по возрастным группам Список учащихся в зависимости от вида занятий Расписание ребенка	Количество учащихся по видам занятий и возрастным группам Занятость залов по дням Количество занятий у ребенка по дням
6	Сводная ведомость посещений по видам услуг и клиентам	Личная карточка пациентов Расписание врача на день Список клиентов	Количество посещений клиентов по месяцам Количество пациентов по

	Сводная ведомость занятости кабинетов		месяцам Количество пациентов у врача
7	Занятость автомобилей по дням Занятость инструктора Список автотранспорта по статусу Количество часов по программам и по видам занятий у курсантов	Сводный список курсантов по видам программ Расписание курсанта Расписание инструктора	Количество автотранспорта по видам Количество курсантов по программам Количество часов по программам
8	Сводный список туров по операторам и направлениям Сводный список услуг по операторам и направлениям Сводный список затрат по операторам и направлениям Полученный доход по направлениями и включенным услугам	Список клиентов Список отелей по направлениям	Количество туров по направлениям Стоимость туров по направлениям
9	Сводный список по сложности операций в зависимости от модели и ткани Количество затрат материалов на пошив и прямую продажу	Список материала в наличии Список клиентов Каталог отшитых моделей	Количество заказов по видам операций Количество затрат материалов на пошив и прямую продажу Накопительные скидки Количество оказанных услуг по месяцам
10	Сводный список задолженности клиентов по месяцам Сводный список заказов клиентов по месяцам и видам работ Сводный список работ сложности пятен и ткани	Список клиентов Количество оказанных услуг по сложности Список клиентов по видам оплаты	Количество оказанных услуг по сложности Сводный список заказов клиентов по месяцам и видам работ

Задание 2. Создайте печатные формы для вывода документов, добавив к каждому документу поле подпись и дату проведения.