**МИНОБРНАУКИ РФ**

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«ДАГЕСТАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Задание по подготовке выпускной квалификационной работы

Студенту Гарунову Зайнулабиду Омаровичу

1. Тема работы Разработка конфигурации для автоматизации и поиска кадров на платформе 1С: Предприятие

утверждена приказом по университету от 28.08.20 № 1533-с

2. Срок сдачи студентом законченной работы 25.06.2020

3. Исходные данные к работе Конфигурация для автоматизации и поиска кадров на платформе 1С: Предприятие, списки выпускников вузов Республики Дагестан за ряд лет

4. Перечень подлежащих разработке выпускной квалификационной работе вопросов или краткое содержание выпускной квалификационной работы:

а) Теоретико-методические аспекты разработки информ
б) Разработка и реализация конфигурации в программе 1
в) Выводы и приложение

5. Перечень графических материалов (с точным указанием обязательных чертежей)

Скриншоты программы

6. Консультанты по работе (с указанием относящихся к ним разделов)

7. Дата выдачи задания «15» 01 2020 г.

Кафедра *Информационных технологий и моделирования экономических процессов* ИТиМЭП

Утверждаю «15» 01 2020 г.

Зав. кафедрой *д.т.н., проф. Адамадзиев К.Р.* «_____»
подпись

Руководитель доцент ИСиТП Рабаданов «_____»
подпись

Задание принял к исполнению «15» 01 2020 г.

Подпись студента _____

**ОТЗЫВ**
**РЕЦЕНЗЕНТА О ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ**

Автор (студент/ка) Гарунов Зайнулабид Омарович
Факультет Информатики и информационных технологий .
Кафедра «Информационных систем и технологий программирования» .
Направление 09.03.03 Прикладная информатика .
Профиль Прикладная информатика в экономике .
Наименование темы: Разработка конфигурации для автоматизации и ведения кадров на платформе 1С: Предприятие

Рецензент Дадаева Б. Ш. К-т.н. доц. кафедры „Математическое моделирование, эконометрика и статистика"
(Фамилия, И., О., место работы, должность, ученое звание, степень)

ОЦЕНКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

| № п/п | Показатели | Оценки | | | | |
|---|---|---|---|---|---|---|
| | | 5 | 4 | 3 | 2 | * |
| 1. | Актуальность тематики работы | V | | | | |
| 2. | Степень полноты обзора состояния вопроса и корректность постановки задачи | V | | | | |
| 3. | Уровень и корректность использования в работе методов исследований, математического моделирования, расчетов | V | | | | |
| 4. | Степень комплексности работы, применение в ней знаний профессиональных и специальных дисциплин | V | | | | |
| 5. | Ясность, четкость, последовательность и обоснованность изложения | V | | | | |
| 6. | Применение современного математического и программного обеспечения, компьютерных технологий в работе | V | | | | |
| 7. | Качество оформления (общий уровень грамотности, стиль изложения, качество иллюстраций, соответствие требованиям стандартов) | V | | | | |
| 8. | Объем и качество выполнения графического материала, его соответствие тексту | V | | | | |
| 9. | Обоснованность и доказательность выводов работы | V | | | | |
| 10. | Оригинальность и новизна полученных результатов, научно-исследовательских или производственно-технологических решений | V | | | | |

***-не оценивается (трудно оценить)**

Отмеченные достоинства Актуальность выпускной квалификационной работы Гаврилова З.О достаточно высока
Материал в работе достаточно структурирован, написан научным стилем изложения и состоит из введения двух глав, заключения, списка литературы и приложений.
В первой главе на высоком теоретическом и методологическом уровне проведено исследование и описание модулей используемых на платформе 1С. Проведён обзор по кадровому учёту и программе 1С
Вторая глава посвящена разработке конфигурации для автоматизации учёта кадров на платформе 1С: Предприятие. Приведена разработка списка запросов предназначенная для быстрой обработки данных и получения необходимых результатов.
Научные в работе результаты и выводы достаточно обоснованы, структура и содержание раскрывает тему исследования

Отмеченные недостатки В качестве недостатков выпускной квалификационной работы можно отметить:
1) Имеется небольшая неряшливость в оформлении работы, текста, списка литературы
2) Желательно было добавить немного больше скриншотов и кодов для большей наглядности при получении результатов при написании запросов
В целом, выявленные недостатки не влияют на качество написания исследования

Заключение Выпускная квалификационная работа Гаврилова З.О выполнена на достаточно высоком уровне. Рекомендуемая оценка: „отлично"

Рецензент _____ «25» 01 2020 г.
(подпись)

**ОТЗЫВ**
**РУКОВОДИТЕЛЯ ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ**

Тема квалификационной работы Разработка конфигурации для автоматизации поиска кадров на платформе 1С:Предприятие

Автор (студент/ка) Гарипов Зайнуллабид Ильдарович
Факультет Информатики и информационных технологий .
Кафедра «Информационных систем и технологий программирования» .
Направление 09.03.03 Прикладная информатика .
Профиль Прикладная информатика в экономике .
Руководитель Габбасова Г. И. К. Э. Н., доцент кафедры "Информационных систем и технологий программирования"
(Фамилия И.О., место работы, должность, ученое звание, степень)

## Оценка соответствия требованиям ФГОС подготовленности автора выпускной квалификационной работы

| Требования к профессиональной подготовке | Соответствует | В основном Соответствует | Не соответствует |
|---|---|---|---|
| Уметь корректно формулировать и ставить задачи (проблемы) своей деятельности при выполнении квалификационной работы, анализировать, диагностировать причины появления проблем, их актуальность | V | | |
| Устанавливать приоритеты и методы решения поставленных задач(проблем) | V | | |
| Уметь использовать информацию в сфере профессиональной деятельности | V | | |
| Владеть компьютерными методами сбора, хранения и обработки (редактирования) информации применяемой в сфере профессиональной деятельности | V | | |
| Владеть современными методами анализа и интерпретации полученной информации, оценивать их возможности при решении поставленных задач (проблем) | V | | |
| Уметь рационально планировать время выполнения работы, определять грамотную последовательность и объем операций и решений при выполнении поставленной задачи | V | | |
| Уметь объективно оценивать полученные результаты расчетов и вычислений | V | | |
| Уметь анализировать полученные результаты интерпретации данных | V | | |
| Знать методы системного анализа | V | | |
| Уметь осуществлять деятельность в кооперации с коллегами, находить компромиссы при совместной деятельности | V | | |
| Уметь делать самостоятельные обоснованные и достоверные выводы из проделанной работы | V | | |
| Уметь пользоваться научной литературой профессиональной направленности | V | | |

Отмеченные достоинства Структура содержания и оригинальность работы соответствует требованиям. К достоинствам работы можно отнести:
1) Актуальность темы исследования
2) Умение корректно формулировать и ставить задачи при выполнении научного исследования
3) Изучено большое количество теоретического материала
4) Была разработана конфигурация для автоматизации и поиска кадров, с помощью которого можно быстро обработать данные и получить необходимые результаты.
5) Практическая направленность исследования

Отмеченные недостатки
Имеется небольшая погрешность в оформлении работы, текста, списка литературы.
Желательно было добавить немного больше опрошенных и кадров для большей наглядности при получении результатов или выполнения запросов.
В целом выявленные недостатки не влияют на качество написания исследования

Заключение Выпускная квалификационная работа Гаврилова Э.О выполнена в соответствии с требованиями на достаточно высоком уровне.
Рекомендуемая оценка "отлично"

Руководитель _____ «____» _____ 20___ г.
(подпись)

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Дагестанский государственный университет»
Факультет информатики и информационных технологий

**Выпускная квалификационная работа бакалавра**
по направлению 09.03.03 – Прикладная информатика
(уровень бакалавриата)
студента 4 курса очного отделения

**Гарунова Зайнулабида Омариевича**

# «Разработка конфигурации для автоматизации и поиска кадров на платформе 1С:Предприятие»

Научный руководитель:
к.э.н., доцент Рабаданова Р.М.

Рецензент:
к.э.н. доц. кафедры «ММЭ и С»
Дадаева Б.Ш.

Работа допущена к защите:
Зав. Кафедрой ИСиТП,
к.э.н., доц. Исмиханов З.Н.
«16» июня 2020 г.

Махачкала – 2020

## Оглавление

# Введение

На современном этапе перехода к стандартам нового поколения, основанном на модульных технологиях, вопрос о качественной организации работы с кадрами, оценки кадрового потенциала и подбор этих кадров становится особенно актуальным. Правильный выбор кандидата на вакантную должность – важный фактор его успешной трудовой деятельности, профпригодности и максимальной самоотдачи.

Глобализация, изменение потребительского поведения, переход к сетевой цифровой экономике являются основными тенденциями современности, что приводит к необходимости поиска новых конкурентных моделей управления организациями. Развитие цифровых технологий, интегрированных со стратегией управления человеческими ресурсами, становится ключевым условием обеспечения трансформации бизнеса от традиционной к технологичной компании.

Поэтому задача разработки качественного инструментария для обеспечения оперативного и эффективного подбора персонала становится одной их наиболее значимых, ключевых в работе руководителей департаментов по управлению человеческими ресурсами и менеджеров по персоналу.

Цель выпускной квалификационной работы заключается в разработке конфигурации для автоматизации и поиска кадров на платформе 1С: Предприятие.

Для достижения нашей цели необходимо решить следующие задачи:

- изучить методологическую и техническую литературу по теме исследования;
- создать информационную базу в программе 1С;
- создать необходимые подсистемы;
- создать справочники;
- выбрать средство для реализации поиска кадров;
- освоить язык SQL;

– разработать конфигурации для автоматизации поиска и выбора кадров на платформе 1С:Предприятие .

Данная выпускная квалификационная работа состоит из введения, двух глав, заключения, списка использованной литературы и приложения.

Во введении рассматривается актуальность данного исследования, цель и задачи исследования.

В первой главе были изучены теоретико-методологические аспекты разработки конфигурации. Были изучены особенности разработки подсистем и справочников. Приведена целесообразность использования программы 1С:Предприятие.

Во второй главе описана разработка конфигурации в программе 1С: Предприятие. Для достижения цели нашего исследования были разработаны следующие подсистемы: Бухгалтерия, Учёт Материалов, Оказание Услуг, Предприятие, Расчёт Зарплаты. Также были разработаны справочники: Клиенты, Сотрудники, Номенклатура, Склады, Кадры. Разработанный программный модуль наполнен контентом. Приведено описание программного обеспечения, его компонентов и рекомендации пользователю.

В заключении подводятся итоги проделанной работы.

Выпускная квалификационная работа выполнена на 46 страницах, список использованной литературы содержит 49 источников. Квалификационная работа содержит 37 рисунков и 1 приложение.

# Глава 1. Теоретико-методические аспекты разработки модулей в программе 1С:Предприятие.

## 1.1 Кадровый потенциал, как объект автоматизации в условиях цифровой экономики

Одним из ключевых факторов повышения эффективности деятельности предприятия является отношение к кадрам предприятия. Чтобы иметь высококвалифицированных специалистов, составляющих ядро промышленного предприятия, создать у них стимул к эффективной работе, руководители вынуждены использовать систему управления кадровым потенциалом.

Программа управления кадровым потенциалом способствует не только продвижению персонала, но и его развитию, повышению результативности мероприятий по повышению квалификации. Что в свою очередь является первостепенной задачей службы управления персоналом предприятия.[1]

Автоматизацию производства можно обеспечить с помощью дополнительного программного обеспечения.

Компания **Электронные Офисные Системы** создала специальную систему **КАДРЫ**. Она оптимизирует и автоматизирует кадровое делопроизводство:

- табельный учет рабочего времени;
- оформление командировок;
- оформление отпуска;
- ведение личных карточек;
- оформление отпуска;
- ведение штатного расписания;
- формирование и ведение приказов по личному составу и др.

Система **КАДРЫ**, которая обеспечивает кадровое делопроизводство, успешно используется в ведомственных структурах с подчиненными и территориально-географически распределенными компаниями, позволяя обрабатывать кадровые данные во всех этих компаниях.

Она отличается следующими главными преимуществами:

- гибко настраивается;
- реализована на базе новейших технологий;
- система успешно используется как в небольших компаниях, так и в крупных организациях (т.е. не зависит от количества кадров предприятия).

**Отдел кадров**

Отдел кадров – это программа с широким функционалом, котрая способна заменить даже такую программу как 1С. Следует сразу учесть, что это приложение платное. Разработчики предоставляют бюджетным и государственным организациям скидку в размере 30%.

Интерфейс программы интуитивно понятен, поэтому сложностей в работе с ней возникнуть не должно. Но конечно, сначала могут возникнуть некоторые неясности, так как «Отдел кадров» оснащен большим количеством встроенных функций. Со временем вся работа будет выполняться практически автоматически.

Основные функции приложения:

- импорт/Экспорт данных из 1С;
- экспорт данных для отправки в ПФР;
- составление штатного расписания;
- подключение любого типа классификатора;
- учет отпусков и командировок;
- создание кадровых документов;
- расчет всех видов стажей;
- составление штатного расписания;
- работа со стандартными отчетами;
- создание подробной карточки сотрудника;
- статистика организации (свободные и занятые единицы).

На самом деле это не весь список того, что умеет делать эта программа. Вы можете работать с программой по сети. При необходимости можно также подключать дополнительного сотрудника, но за каждый аккаунт придется

платить.[3]

## 1.2. Основные принципы и этапы разработки модулей в программе 1С:Предприятие.

В основу системы программ **1С: Предприятие 8** заложен принцип модульности. Модули представляют собой типовые решения для типовой автоматизации конкретных задач управления и учета. Например, для автоматизации ведения всех разделов бухгалтерского учета служит модуль **1С: Бухгалтерия**, для учета любых видов торговых операций — **1С: Управление торговлей**.

Программы системы **1С: Предприятие** поставляются с типовыми конфигурациями, реализующими наиболее общие схемы учета, которые используются в большинстве организаций. В случае необходимости программные продукты могут быть адаптированы к любым особенностям учета.

**Состав системы 1С: Предприятие 8:**

В состав системы входит конфигуратор, который обеспечивает:

- возможность создания печатных форм документов и отчетов;
- настройку внешнего вида форм ввода информации;
- организацию справочников и документов произвольной структуры;
- реализацию произвольной методологии учета;
- возможность представления информации в виде диаграмм;
- настройку системы на различные виды учета;
- настройку поведения и алгоритмов работы системы в различных ситуациях;
- быстрое изменение конфигурации с помощью «конструкторов».

Конфигуратор позволяет не только изменять элементы типовой конфигурации, но и создать собственную конфигурацию «с нуля».

Программное обеспечение 1С содержит разнообразные средства для связи с другими программами и аппаратными средствами: средства импорта и экспорта информации через текстовые файлы, файлы формата DBF и XML,

сохранение печатных форм в форматах MS Excel и HTML, возможность экспорта данных в «Диспетчер контактов для малого бизнеса» MS Office 2000.

Для организации **единой системы автоматизированного учета** на предприятиях, которые имеют территориально удаленные подразделения *(центральный офис, магазин, склад и т. д.)*, существует дополнительный компонент «Управление распределенными информационными базами». Возможности, предоставляемые этим компонентом, позволяют организовать работу распределенной информационной системы с неограниченным количеством автономно работающих информационных баз.

**Основные этапы разработки в 1С**

- проектирование;
- кодирование;
- тестирование;
- отладка.[6,7]

**Модуль 1С**

Любая программа состоит из программного кода, то есть собственно из написанных на каком-либо языке последовательности действий, которые должны быть выполнены.

Однако эта самая программа должна быть где-то написана, то есть где-то находиться. В большинстве случаев код программы пишется в обычных текстовых файлов. Разница только в том, что расширение в них не .txt, а .cpp или .php.

Код 1С можно написать в каком-нибудь текстовом файле. Однако есть понятие Конфигурация 1С – которое включает в себя не только список настроек, шаблонов форм и прочего, но и программный код 1С. Поэтому код 1С хранится в конфигурации.

Конфигурация состоит из объектов 1С. Каждый объект 1С содержит в себе

вложенные объекты, например справочник имеет несколько форм.

Каждый объект 1С, включая некоторые вложенные, имеет свой Модуль – некий текстовый файл, который содержит программный код.

Также есть независимые от объектов модули, в которых может быть написан программный код, независимый от конкретного объекта.

Таким образом в 1С нет «единой» программы. Есть набор модулей для написания программного кода для каждого объекта конфигурации 1С.

Как используются Модули 1С?

Всю программу можно грубо поделить на два вида:

- метод объекта;
- реакция на события.

**Методы**. Как мы уже говорили ранее – объект 1С является цельной структурой, которая включает в себя как данные, так и способы их обработки. Эти способы – это набор действий (методов), которые можно вызывать для обработки данных. Пример такого действия СправочникОбъект.Записать() – записывает элемент справочника в базу данных.

Методы многих объектов 1С могут быть стандартными (т.е. запрограммированными в платформе 1С) и написанными программистом на языке 1С. С помощью вторых – можно расширять функционал объектов 1С по своему желанию.

**События**. События есть во множестве других средств разработки. Цель программы не только что-то вычислить при запуске, но и поддерживать работу пользователя.

Пользовательское события – пользователь нажал кнопку. В ответ какая-то часть кода выполнится, осуществив реакцию на действия пользователя.

Системные события – мы записали объект 1С в базу данных. Возникло системное событие «Запись объекта». Возможно настроить реакцию, которая возникнет на события, вызванные не пользователем (которые нажал на кнопку или что-то еще сделал), а самой системой. Яркий пример такого события – при

запуске программы.

Порядок выполнения модулей 1С

Во многих языках есть такое понятие как «точка входа». Это та самая первая строчка или функция которая будет выполнена при запуске программы.

В 1С таких точек входа несколько – на каждый вид клиента. То есть при запуске толстого клиента точка входа одна, при запуске тонкого клиента – другая. Это позволяет запрограммировать особенности, различные в разных видах клиентов.

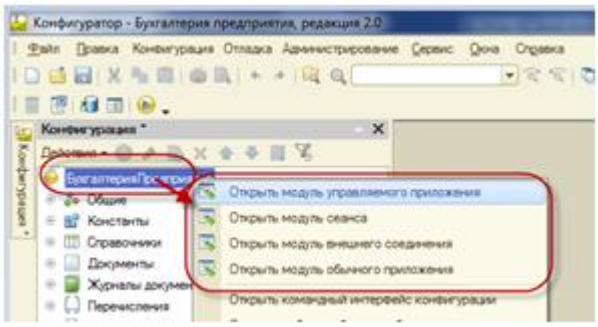



Рис 1. Модули.

Точкой входа в соответствующем модуле является обработчики системного события ПередНачаломРаботыСистемы() и ПриНачалеРаботыСистемы() соответственно (т.е. по порядку). Эти функции выполняется первыми, они может запустить что-то автоматически.

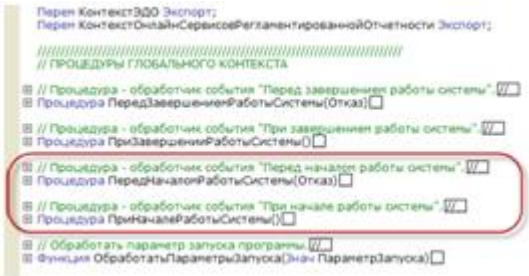



Рис 2. Функции.

Если ничего запущено автоматически не было, то перед пользователем открывается интерфейс 1С и далее все зависит от него. Он нажимает на кнопку – происходит выполнение обработчика нажатия этой кнопки (который в свою очередь тоже может что-то запустить автоматически).

Работа с модулями 1С

Производится в конфигураторе. Открыть модуль можно с помощью окна Конфигурация.

Модули точки входа (в разрезе разных клиентов) можно открыть нажав правой кнопкой на верхнем элементе окна конфигурации. Называться он может по разному, но находится всегда в самом верху.

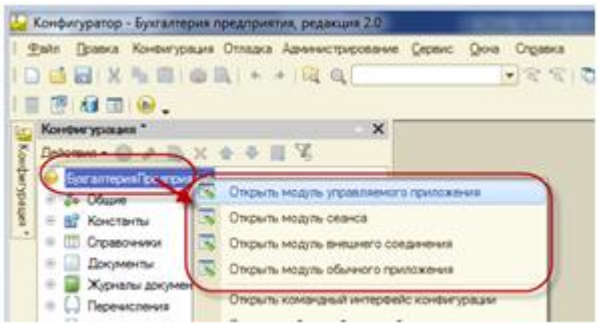



Рис 3. Модули точки входа.

Общие, независимые от объектов 1С модули, находятся в ветке Общие / Общие модули. Просто нажмите на него два раза мышкой и он откроется.

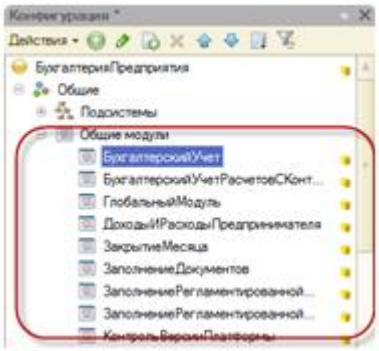



Рис 4.Общие модули.

Модули форм, где прописывается реакция на нажатие кнопок, меню и прочего интерфейсного находятся непосредственно в редакторе формы. Раскройте любую ветку объекта (справочник, документ и т.п.) до форм, нажмите два раза мышкой на форму – отроется редактор. Внизу редактора будет три закладки, одна из которых – модуль.

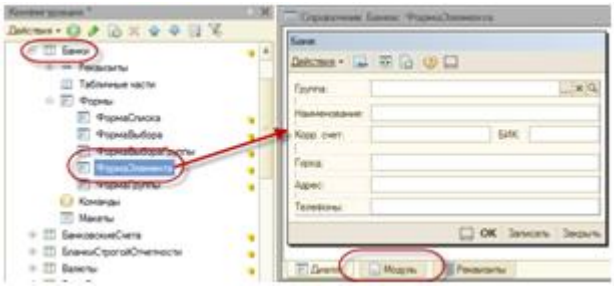


Рис 5. Модули форм.

В 1С версии 8.2 появилось понятие Команда 1С. Это самостоятельное действие, которое можно вытащить на форму в виде кнопки или меню. В ветке Общие / Общие команды находятся команды, которые можно использовать в любом другом месте конфигурации. Каждая из команд имеет свой собственный модуль.

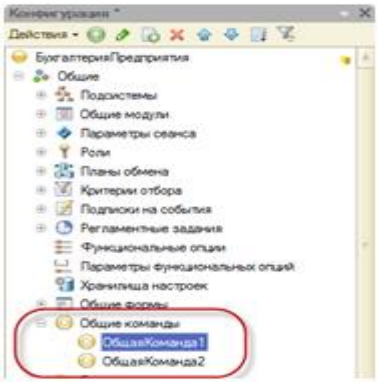



Рис 6. Общие команды.

Модули объектов. У каждого объекта 1С (справочник, документ и прочее) есть свой модуль. Там могут быть прописаны реакции на такие системные события как Запись() или Удаление(), а также созданы новые методы объекта. Нажмите правой кнопкой мыши на конкретный объект, например справочник Номенклатура. Модуль объекта – это модуль конкретного элемента этого справочника. Модуль менеджера – это модуль управления элементами справочника (документа..).

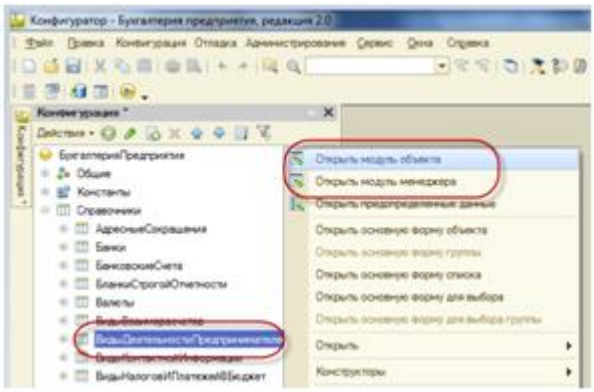



Рис 7. Модули обьектов.

**Состав модуля 1С**

Модуль состоит из набора функций и процедур – т.е. обработчиков событий и методов.

В самом начале модуля можно расположить переменные, которые будут использовать обработчики и события в этом модуле.

В самом низу модуля можно сформировать программный код без оформления функции или процедуры. Это можно назвать автостартом определённого модуля. При попытке первого обращения к любой функции или процедуре этого модуля – будет выполнен автостарт (код внизу модуля).

Как правило его используют для начального заполнения переменных расположенных в начале модуля.

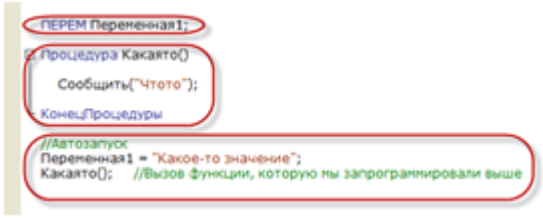



Рис 8. Состав модуля.

**Клиент и сервер 1С**

В серверном части 1С разные части программы выполняются на клиентской машине или на сервере.

У общих модулей есть свойства. Нажмите правой кнопкой мыши и далее – свойства. Галочками можно отметить, где будет доступен этот модуль.

Также один и тот же модуль может выглядеть совершенно по-разному,

если он запущен на клиенте или сервере. Для этого есть специальные разработанные директивы, которые указываться прямо в коде нашей программы: #Если Клиент и #Если Сервер.

Сервер и Клиент могут видеть только те части модуля, которые находятся между этими директивами. Если директив нет – модуль виден весь.

**Контекст**

Контекст – представляет собой весь тот набор процедур, функций и переменных, которые доступны на определенной строке выполнения программы, с теми значениями переменных, которые они уже имеют на данном этапе выполнения программы.

На общих модулях лежит обязанность хранения процедур и функций, которые вызываются из других мест системы 1С. Считается хорошим тоном размещение кода, вызывающегося несколько раз, в процедуре в общем модуле. Это правило универсально для всех конфигураций, поэтому любой разработчик 1С должен уметь работать с этими объектами конфигурации. Для этого нужно понимать все нюансы и уметь правильно использовать предоставленные платформой возможности.

**Создание общего модуля в 1С**

После создания функции в одном из модулей объекта возникла потребность использовать аналогичный алгоритм в другом месте. Самое правильно, что можно здесь сделать – перенести код в общий модуль, но перед этим необходимо создать его. Чтобы это сделать, нам нужно зайти в конфигуратор и в дереве конфигурации найти вкладку «Общие». Затем выделить «Общие модули» и воспользоваться кнопкой в виде белого плюса на зеленом кружке.

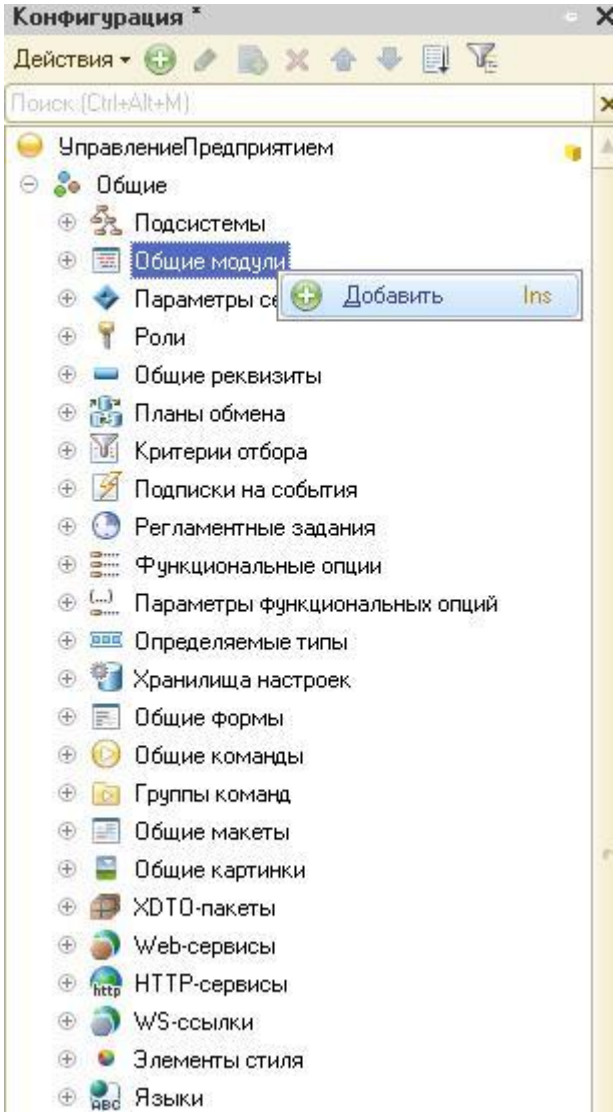



Рис 9. Создание общего моудля.

Справа откроются свойства добавленного общего модуля, и нам предстоит разобраться, что обозначает каждое из них. Они могут быть различной направленности, поэтому, перед тем как настраивать новый объект, желательно определиться, что мы там будем хранить. Если что, в будущем можно будет изменить свойства в соответствии с задачами:

- «Глобальный». Данный флаг ставится, если модуль предназначен для хранения процедур и функций, которые должны вызываться без указания имени модуля. Естественно, они должны быть экспортными, а их имена уникальными в разрезе всего глобального контекста. По использованию они не будут отличаться от стандартных функций платформы;

– «Клиент». Зависит от настроек системы и регламентирует, могут ли процедуры модуля выполняться на стороне клиента;

– «Сервер». Помечаются общие модули, в составе которых планируется помещать алгоритмы для выполнения на сервере;

– «Внешнее соединение». Процедуры модуля с активацией этого свойства смогут выполняться через подключение внешнего источника;

– «Вызов сервера». Отвечает за разрешения процедурам из модуля вызывать сервер, выполняясь на клиенте;

– «Привилегированный». Активация этой настройки позволит при работе кода процедур модуля не проверять права доступа. Вызвать общий модуль с такой настройкой можно только на сервере. Настройки «Клиент» и «Внешнее соединение» будут сброшены;

– «Повторное использование». Может принимать значения: «Не использовать», «На время сеанса», «На время вызова». При многократном вызове одной процедуры система может использовать рассчитанные ранее данные в рамках процедуры (вызов) или жизни всего сеанса (запуска 1С). Стоит быть очень осторожным с этой настройкой, так как из-за неправильного использования таких модулей могут возникать ошибки.

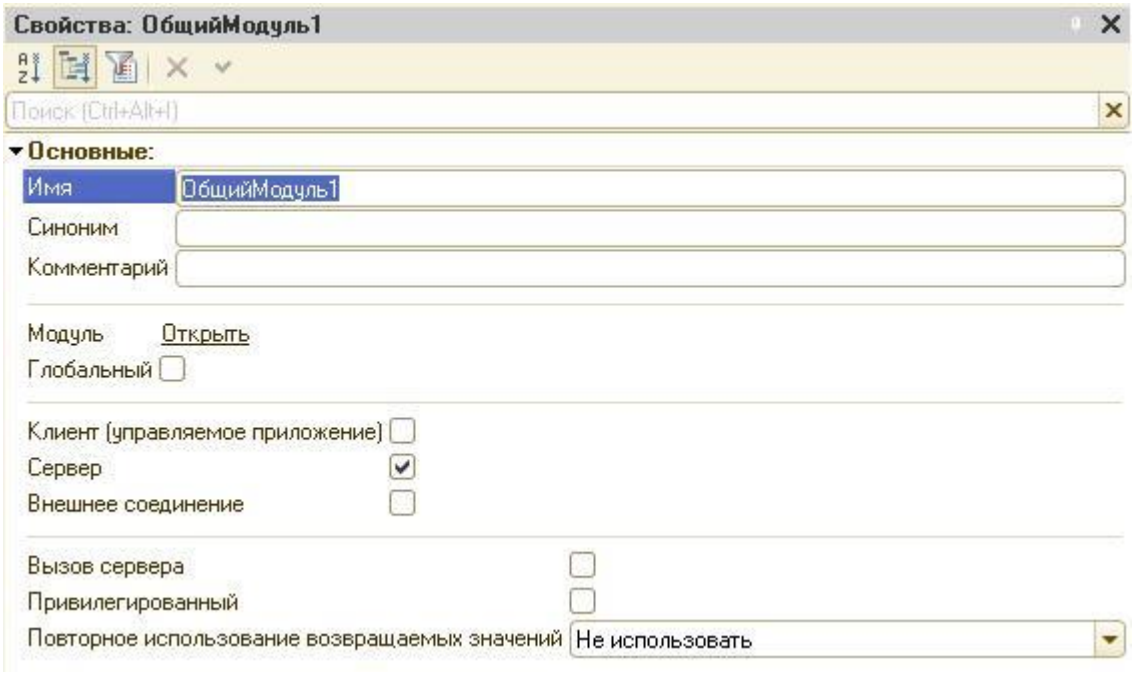



Рис10.Общиймодуль

Бывают ситуации, когда требуется создать общий модуль с вызовами процедуры на сервере и клиенте с отличиями в алгоритме. Для разграничения кода используются директивы препроцессора с проверкой. В результате для серверного вызова это будет один код, а для клиентского – другой.[5]

## Глава 2. Разработка и реализация конфигурации в программе 1С: Предприятие.

### 2.1. Постановка задачи и информационное обеспечение задачи исследования

Цель исследования заключается в разработке конфигурации для автоматизации и поиска кадров на платформе 1С:Предприятие.

Задачами нашего исследования является:

– изучить методологическую и техническую литературу по теме исследования;

– создать информационную базу в программе 1С;

– создать необходимые подсистемы;

– создать справочники;

– выбрать средство для реализации поиска кадров;

– освоить язык SQL;

– разработать конфигурации для автоматизации поиска и выбора кадров на платформе 1С:Предприятие .

При разработке конфигурации для поиска кадрового потенциала региона в программе 1С:Предприятие необходимо знать основные возможности программа 1С:Предприятие, а также основные положения управления кадровым составом.

Постановкой задачи при разработке конфигурации для автоматизации и поиска кадров на платформе 1С: Предприятие является создание подсистем и справочников, затем их наполнение контентом.

Целью выпускной квалификационной работы является разработка конфигурации для поиска и автоматизации кадров на платформе 1С:Предприятие. Для этого была использована консоль запросов и язык запросов SQL.

Также необходимо знать состав программы 1С:Предприятие и возможности программы.

Программа 1С- это продукт фирмы 1С, который предназначен для автоматизации компании. Эта программа используется во всех фирмах и предоставляет множество типовых решений различных бизнес-задач и большое количество конфигурации.

Программа 1С может работать в двух режимах: Предприятия и Конфигуратор. Эти режимы отличаются по предоставляемым возможностям.
Режим Предприятие предназначен для использования его рядовым сотрудником. В этом режиме можно создавать списки, группы, заполнять справочники и т.д. Главное меню режима Предприятия при запуске программы предоставлено на РИС 11.

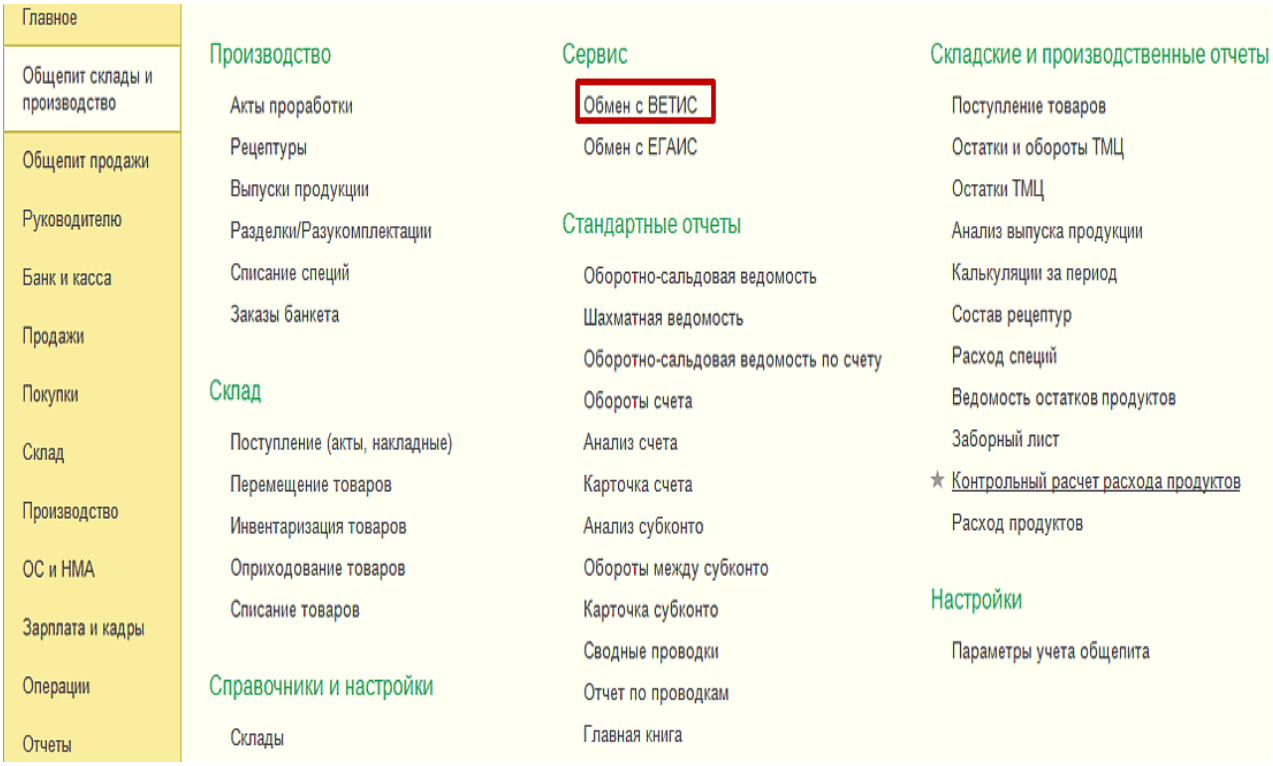



Рис 11. Меню режима Предприятие.

Интерфейс главного меню программы в этом режиме может отличатся у разных пользователей. Так как интерфейс и все что на нём расположено настраивается в режиме Конфигуратора.

Режим Конфигуратор предназначен для квалифицированных пользователей. В этом режиме настраивается интерфейс программы. Создаются модули и располагаются в нужном нам порядке, создаются справочники и настраиваются, выбирают где они будут отображаться. Главное меню этого режима предоставлено на РИС 12.

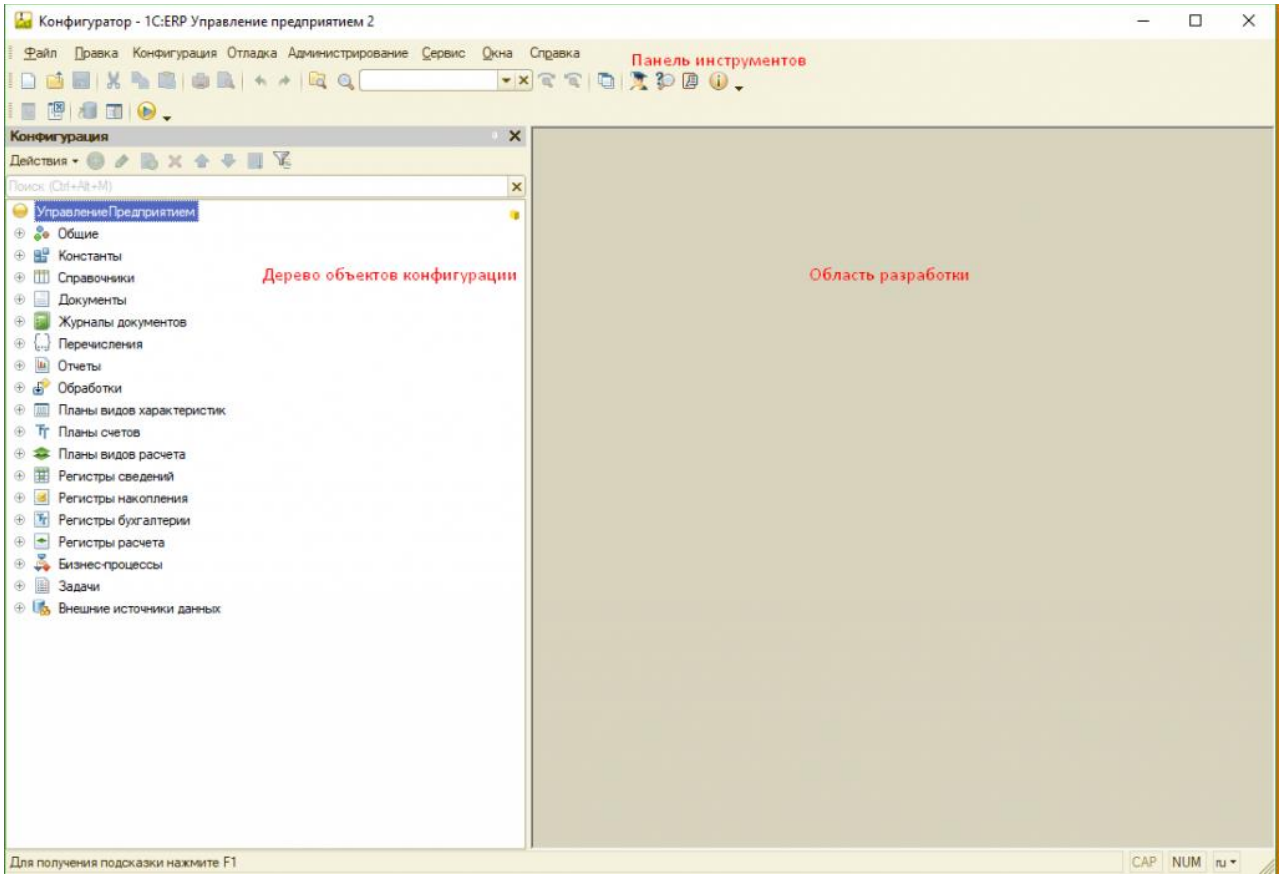



Рис 12. Меню режима Конфигуратор.

Интерфейс в этом режиме у всех пользователей при начале работы выглядит одинаково. Также для разработки конфигурации необходимо было выбрать путь к достижению этой цели. Есть два способа: Создать программно кнопку через конфигуратор в программе 1С или реализовать поиск кадров с помощью запросов.

Для первого варианта необходимо изучить особенности программирования в программе 1С. Так как каждый язык имеет свой уникальный синтаксис. Пример кода в 1С предоставлен на РИС 13.

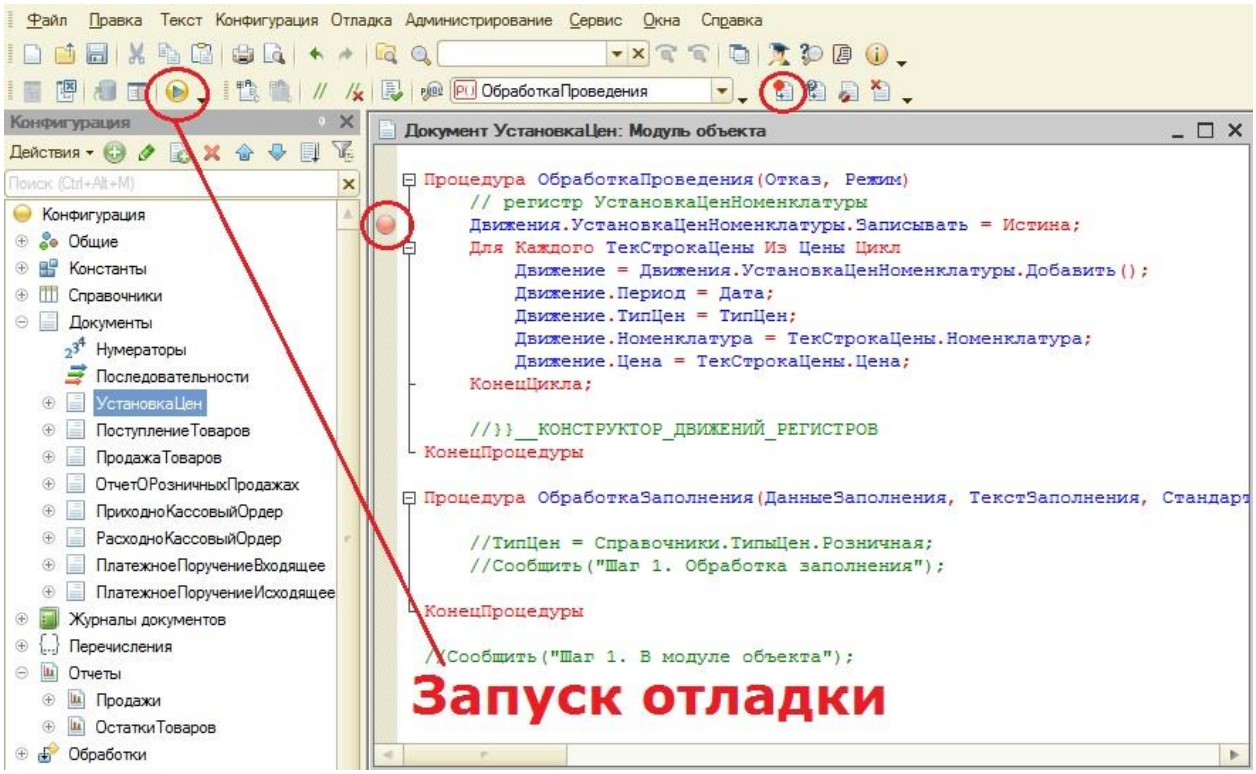



Рис 13.Пример кода.

И второй способ с помощью запросов. Этот способ основан на использовании языка SQL.

SQL — это язык программирования, который применяется для разработки, модификации и управления данными в реляционных базах данных, которые управляются соответствующей системой управления базами данных.

Также необходимо установить обработчик. Обработчик – это специальное программное обеспечение позволяющее реализовывать какие-то функции намного проще, благодаря уже установленным там сценариям. Пример кода написанного на языке запросов SQL представлен на РИС 14.

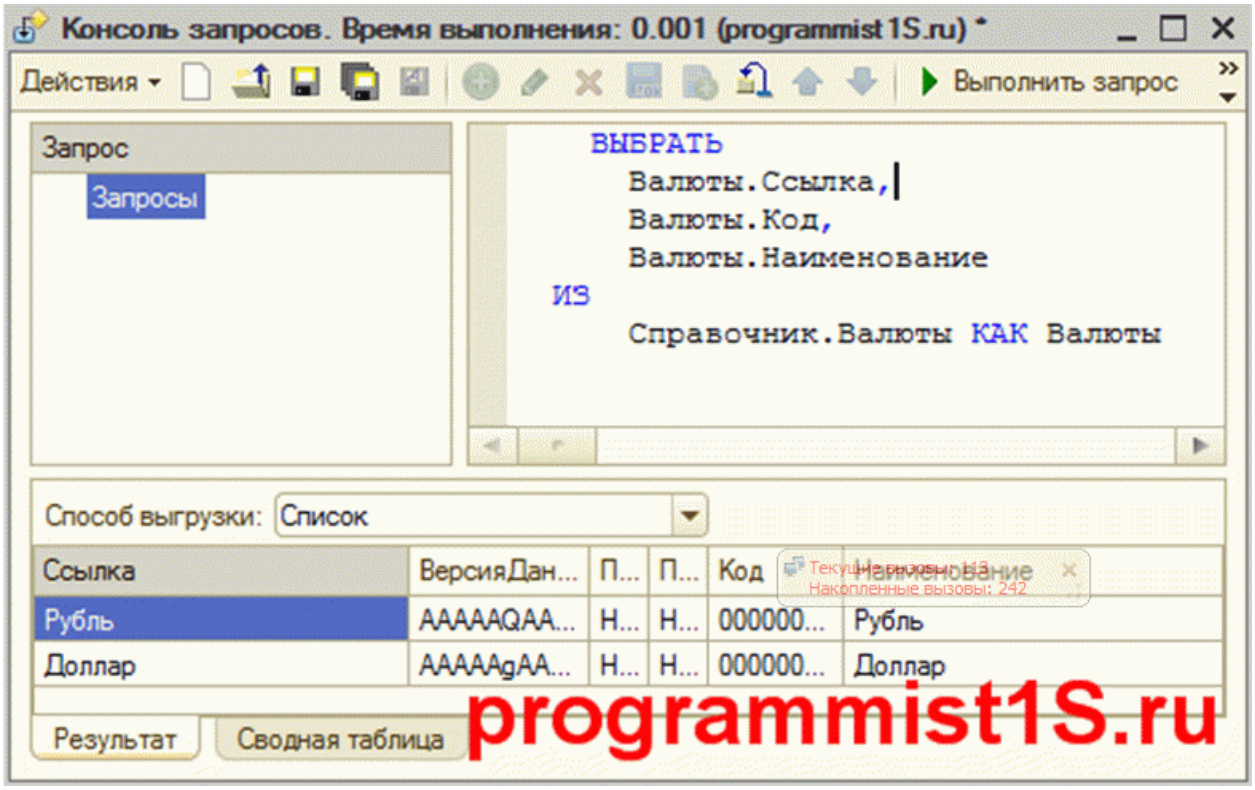



Рис 14.Код на языке SQL.

## 2.2. Состав и структура разработанной конфигурации в программе 1С:Предприятие

Для разработки конфигурации для поиска кадров в программе 1С:Предприятие можно использовать 2 способа: Создать программно кнопку через конфигуратор в программе 1С или реализовать поиск кадров с помощью запросов.

Мной был выбран второй способ: реализовать поиск кадров с помощью запросов. Этот способ основан на использовании языка SQL.

SQL — это язык программирования, который применяется для разработки, модификации и управления данными в реляционных базах данных, которые управляются соответствующей системой управления базами данных.

Для этого необходимо было изучить язык SQL и научиться писать запросы. Но перед этим необходимо было установить программу 1С:Предприятие. Установить её можно с официального сайта.

Далее мне необходимо было создать свою информационную базу. Создание информационной базы интуитивно понятно, необходимо лишь выполнить последовательность шагов и ваша собственная информационная базы готова.

Изначально она пуста, но это поправимо. Мы можем сами настроить интерфейс нашей программы и создать нашу собственную конфигурацию. Для этого необходимо войти в программу в режиме Конфигуратора. Главное окно представлено на РИС 15.

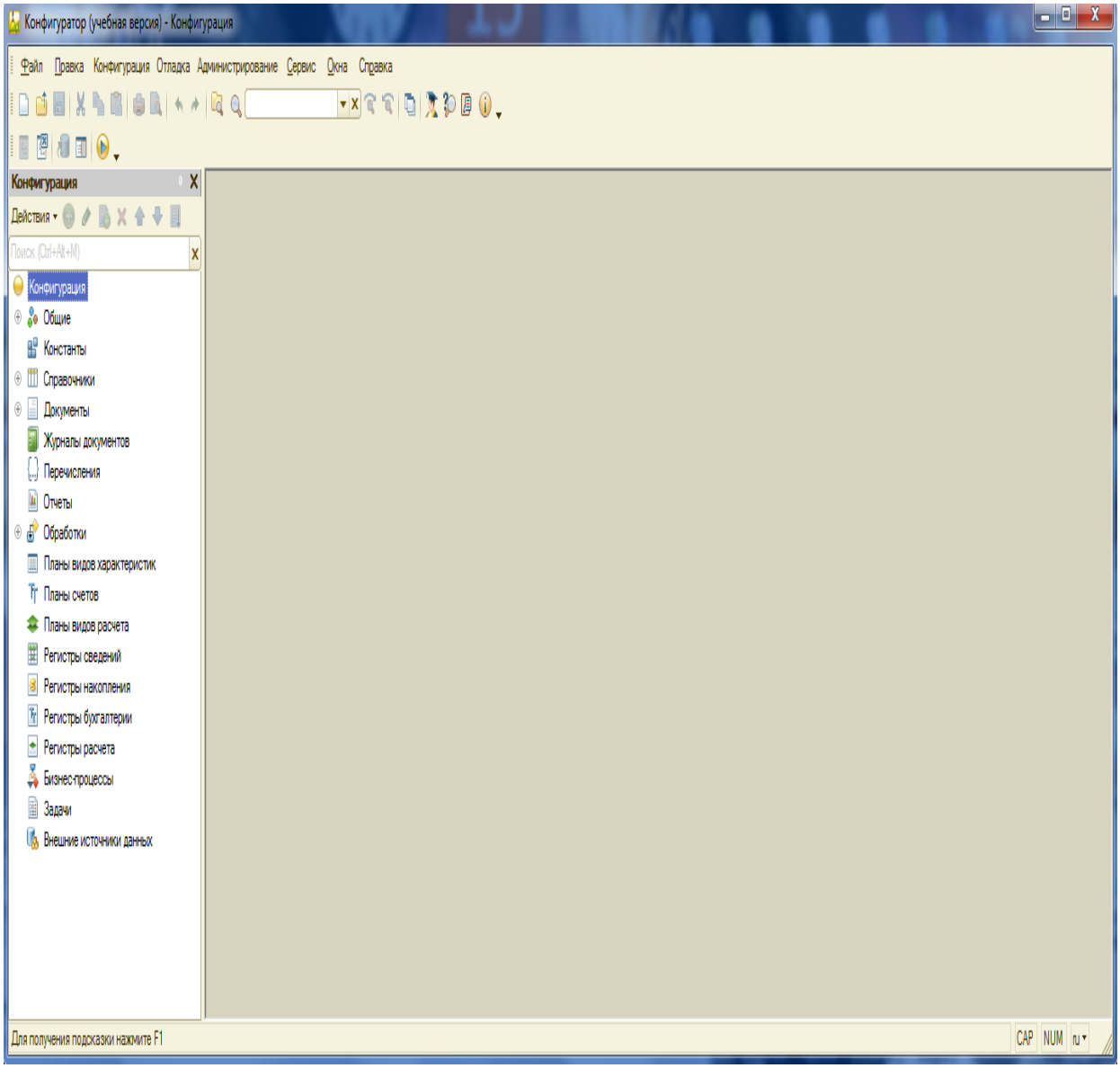



Рис 15.Глвное окно режима Конфигуратор.

Затем нам необходимо создать наши первые подсистемы. Для этого нужно перейти по вкладке общие  и найти пункт подсистемы. По клику правой кнопки мыши по этому пункты у нас появится возможность создать новую подсистему. Также мы можем выбрать картинку, которая будет отражаться рядом с подсистемой. Помимо этого мы можем определить что будет входить в подсистему на вкладке Состав. В неё могут  входить различные модули, общие картинки, справочники и обработчики, если они имеются.  Окно создания подсистемы представлено на РИС 16.

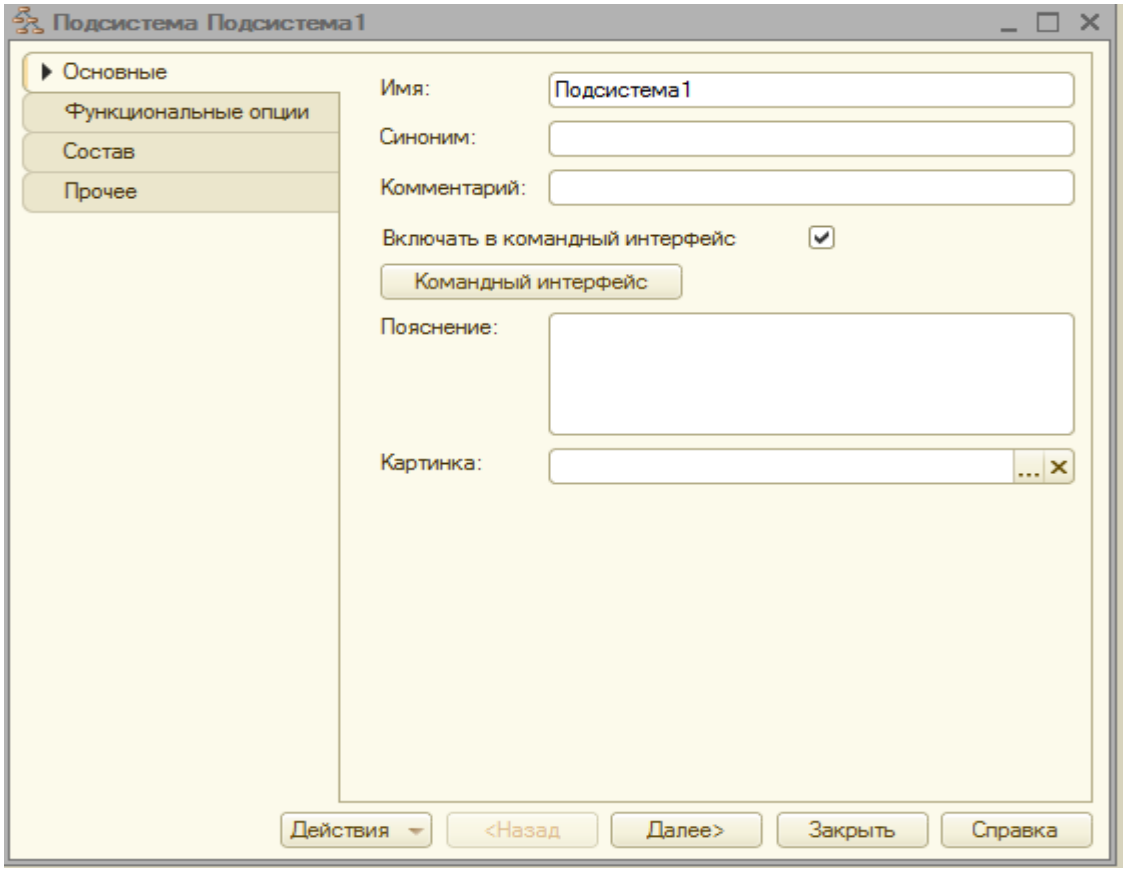



Рис 16.Окно создания подсистемы.

Каждую подсистему можно настроить индивидуально. Можно определить, что будет отображаться при переходе к этой подсистеме в режиме Предприятие. Это сделать очень легко. Необходимо лишь кликнуть правой кнопкой мыши по пункту подсистемы и выбрать все подсистемы. Нам будет представлен весь список созданных нами подсистем. И теперь мы можем определить, что будет отражаться при переходе к какой-либо подсистеме. Эта настройка представлена на РИС 17.

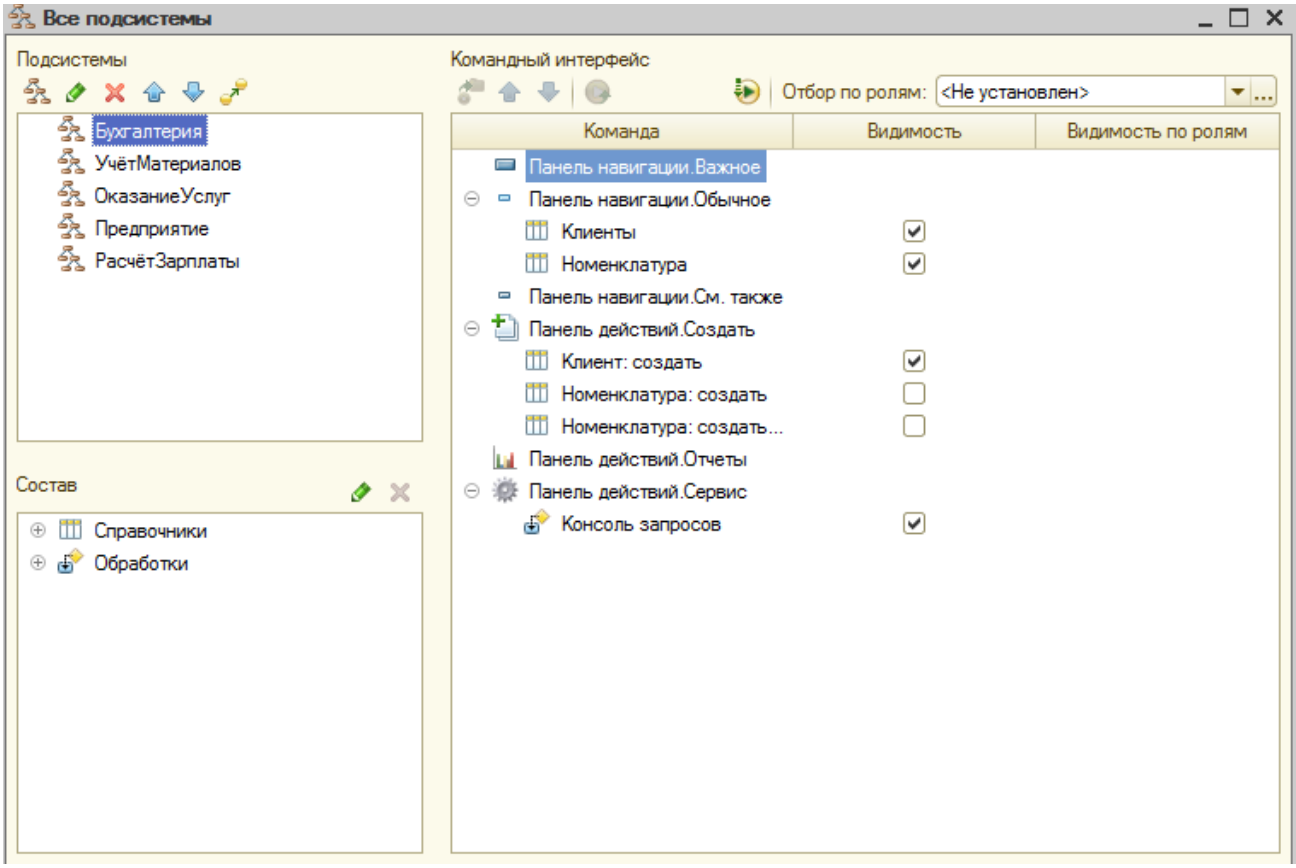



Рис 17.Настройка подсистем.

На представленном рисунке видно, что переходе к системе Бухгалтерия будут отражаться не всё. Остальные элементы будут скрыты в этой подсистеме и с ними нельзя будет взаимодействовать. В режиме Предприятия мы увидим это, так как представлено на РИС 18.

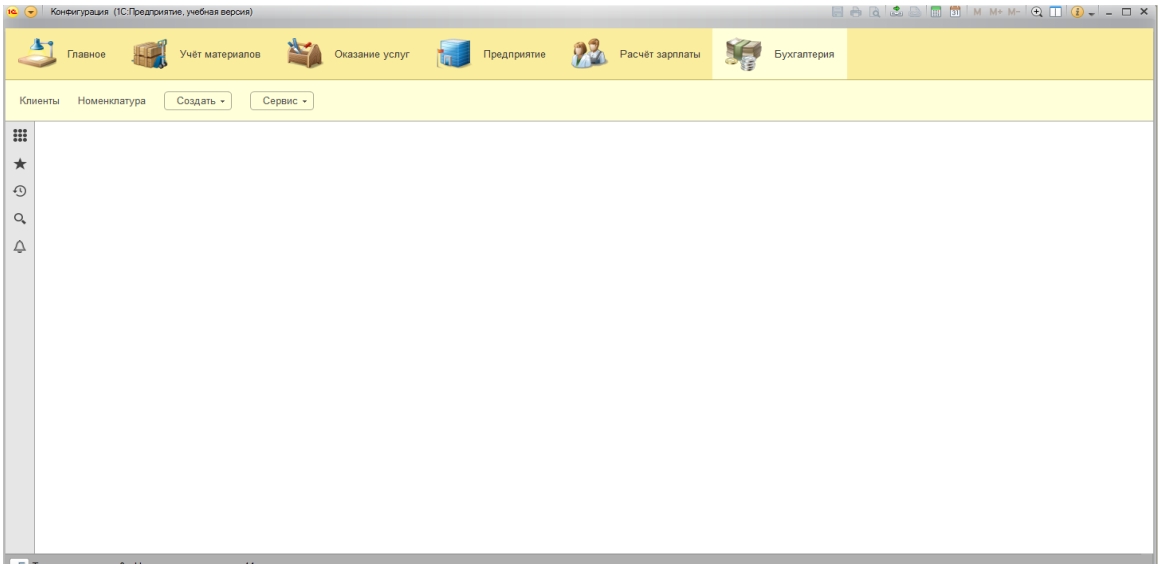
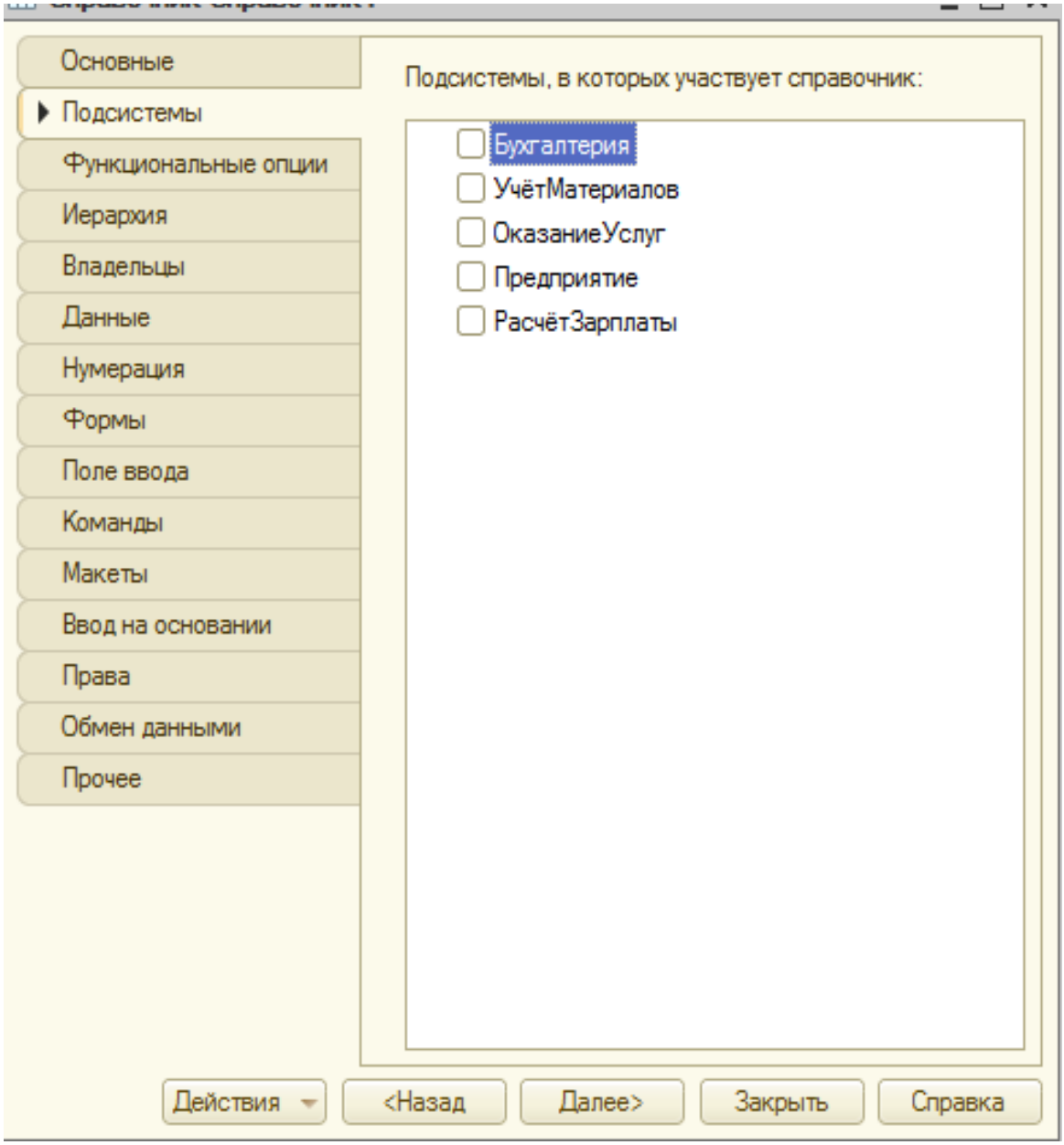


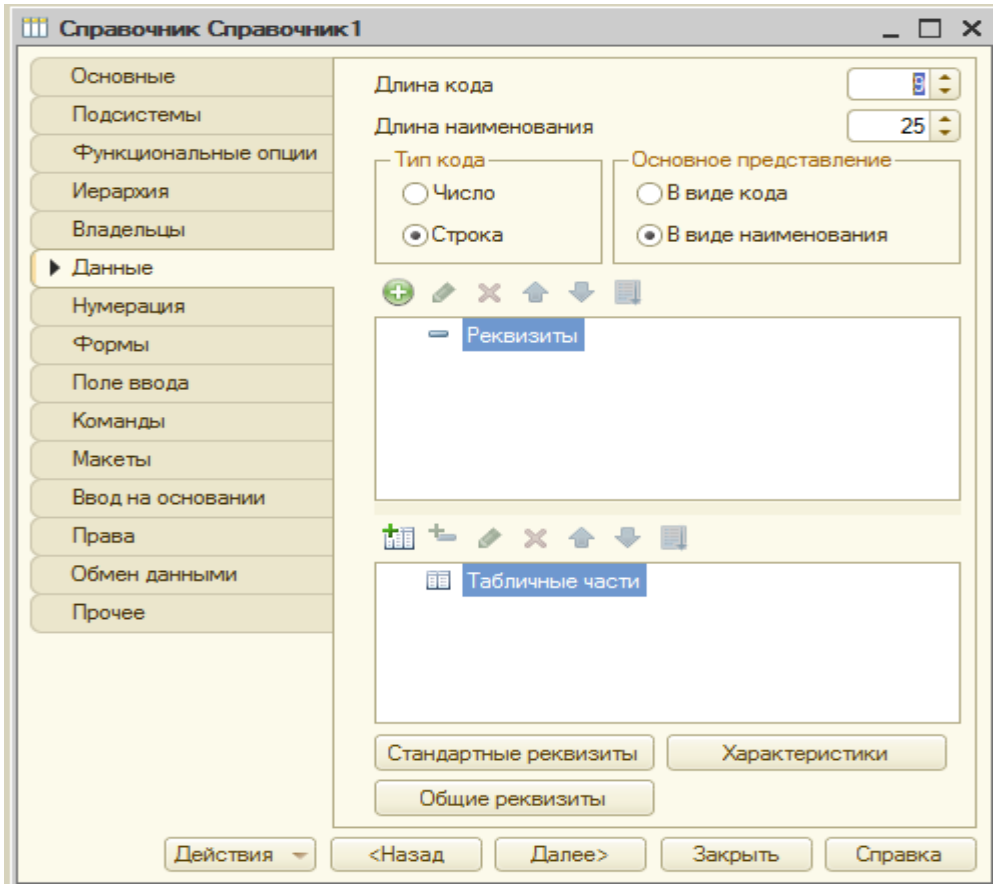

Рис 18.Подсистема в режиме Предприятия.

На представленном рисунке как отображаются системы и элементы в

режиме Предприятия. Также в режиме Конфигуратора можно менять порядок расположения подсистем по нажатию кнопки вверх или вниз.

Далее необходимо создать справочники. Они представляю собой элементы предназначенные для хранения информации и отражаются в подсистемах. Для создания справочника необходимо кликнуть правой кнопкой мыши по пункту справочники и выбрать создать справочник. При создании справочника можно многое определить, например, в какой подсистеме он будет отражаться, определить тип вводимых данных и так далее. Также на вкладке Иерархия можно поставить галочку иерархический справочник. Это позволить нам создавать группы в этом справочнике для более удобного поиска информации не во всём справочнике, а в группе, в которой содержится информации по определенной теме. Меню настройки справочника представлено на РИС 19.


Основные
Подсистемы
Функциональные опции
Иерархия
Владельцы
Данные
Нумерация
Формы
Поле ввода
Команды
Макеты
Ввод на основании
Права
Обмен данными
Прочее
Подсистемы, в которых участвует справочник:
Бухгалтерия
УчётМатериалов
ОказаниеУслуг
Предприятие
РасчётЗарплаты
Действия
<Назад
Далее>
Закрыть
Справка



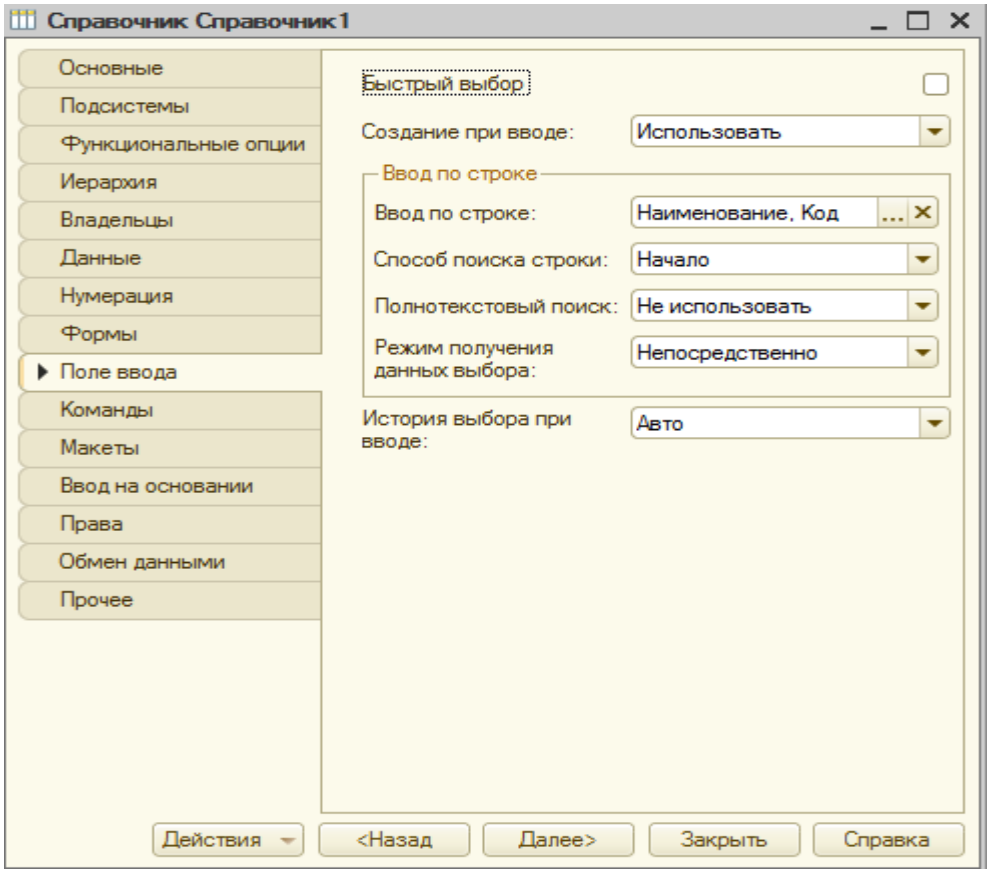




Рис 19.Настройка справочника.

В режиме предприятия справочники выглядят, так как представлено на РИС 10.

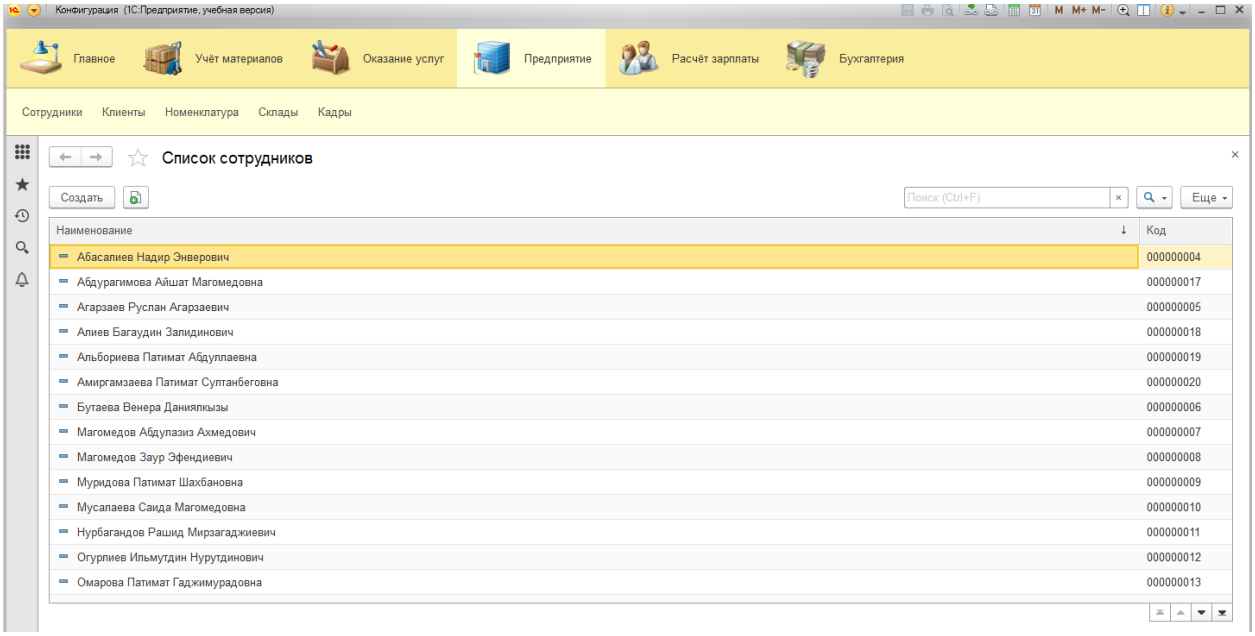



Рис 20.Содержание заполненного справочника в режиме Предприятия.

Вводятся данные в справочники по нажатию кнопки создать и записываются в текущий справочник как представлено на РИС 21.

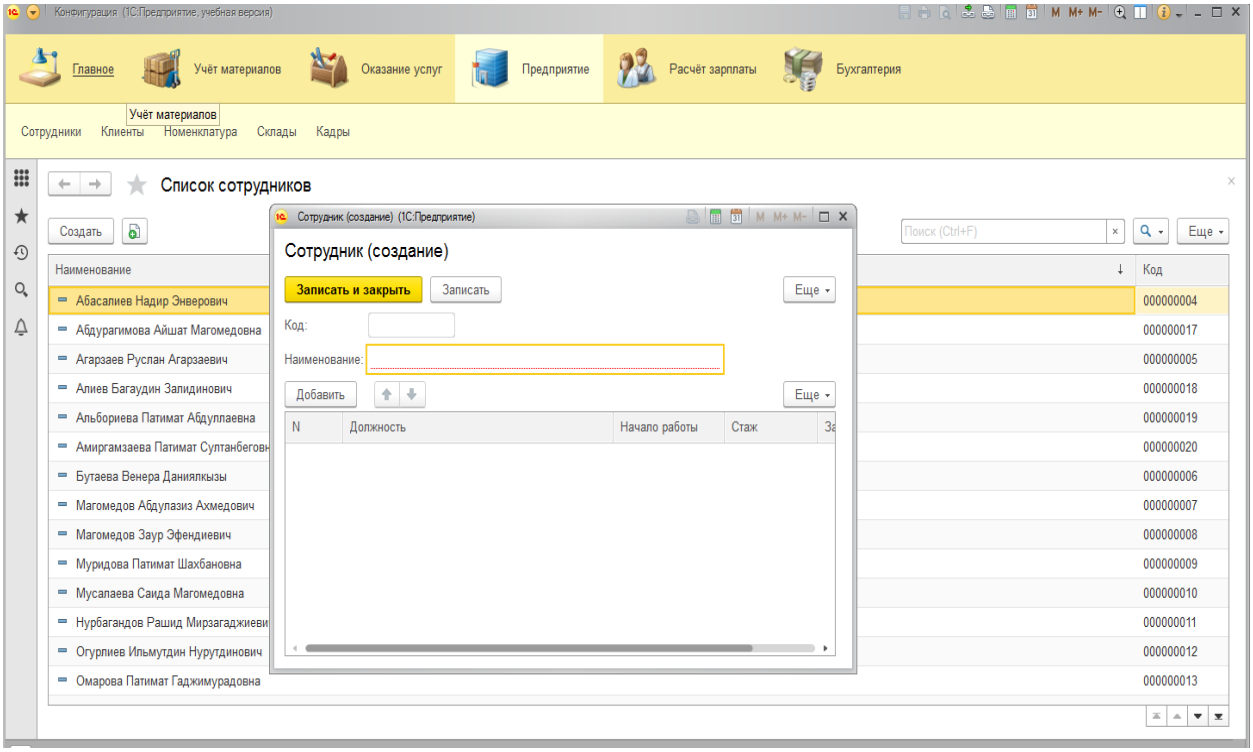



Рис 21.Ввод данных в справочник.

После заполнения всех обязательных полей новая запись будет создана и её можно будет отредактировать в любое время.

Также каждый справочник состоит из реквизитов и табличных записей, представленных на РИС 22. Именно они определяю поля, которые нужно

заполнить при создании записи. Также можно создать дополнительные реквизиты ии назвать их как нам необходимо. Это позволить добавить дополнительные поля для наших записей.

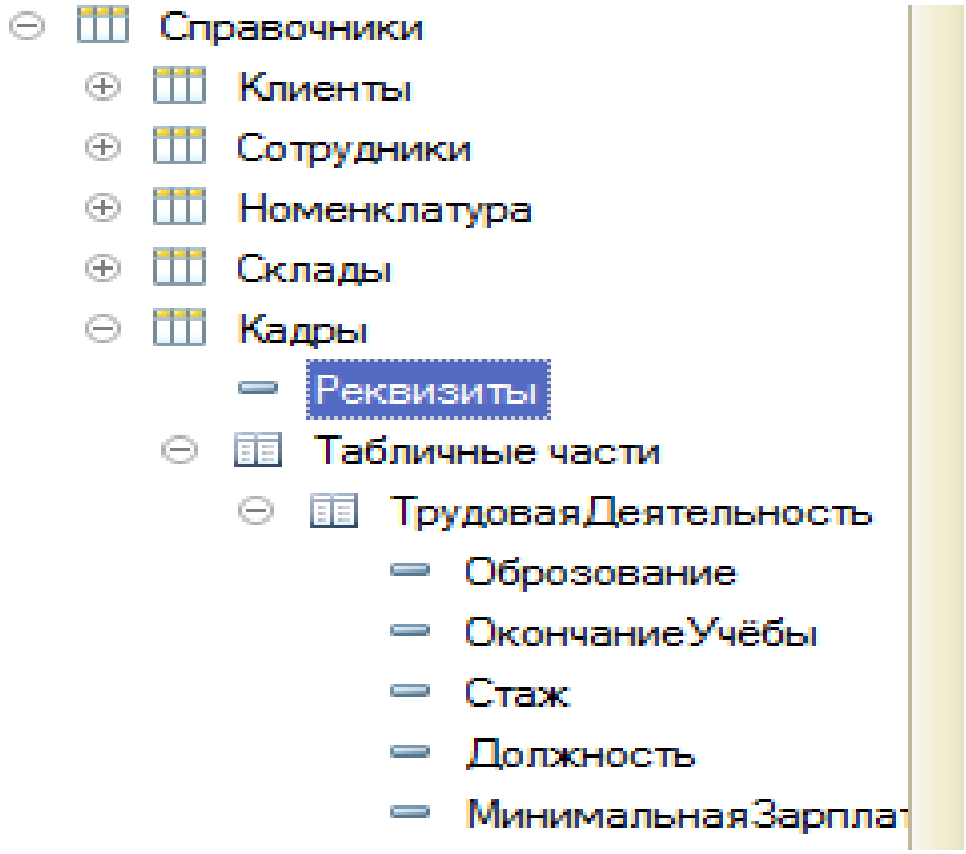



Рис 22.Содержимое справочника в режиме Конфигуратор.

Затем необходимо было определиться, как писать запросы на языке SQL. После изучения языка SQL стало ясно, что это можно реализовать только в консоли запросов. Для этого необходимо было установить обработчик-специальное программное обеспечение позволяющее реализовать обработку данных и выполнения необходимых нам задач. Интерфейс обработчика можно настроить индивидуально. Он создать кнопку на одной из подсистем в режиме Предприятия выбранным мною и будет, называется консоль запросов как это представлено на РИС 23.

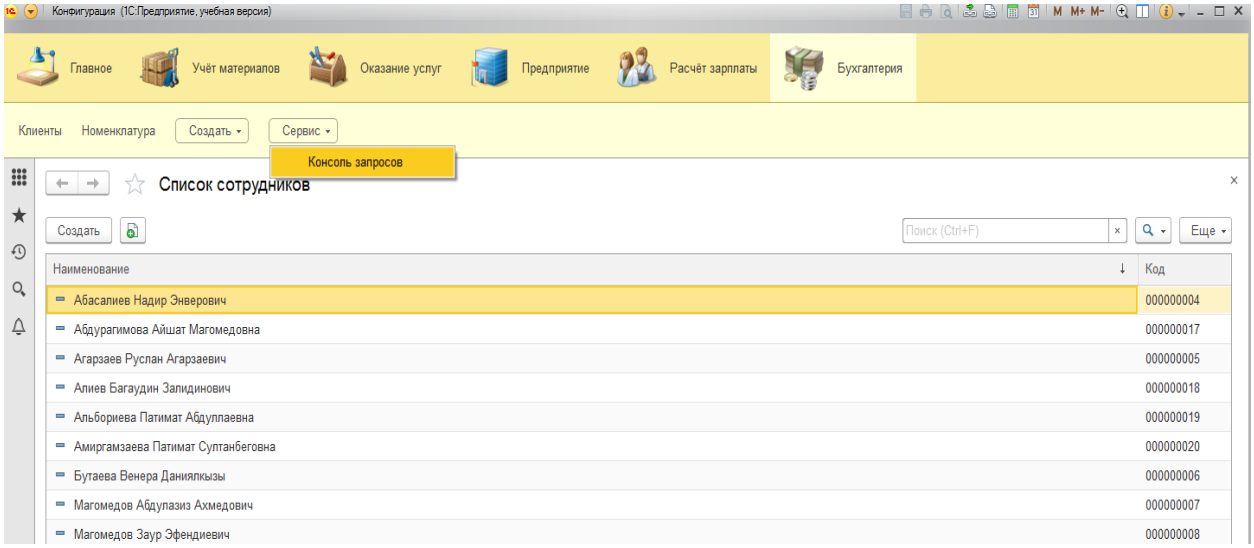



Рис 23. Источник данных для запроса

По щелчку этой кнопки мы перейдём к консоли запросов, которая представлена на РИС 24.

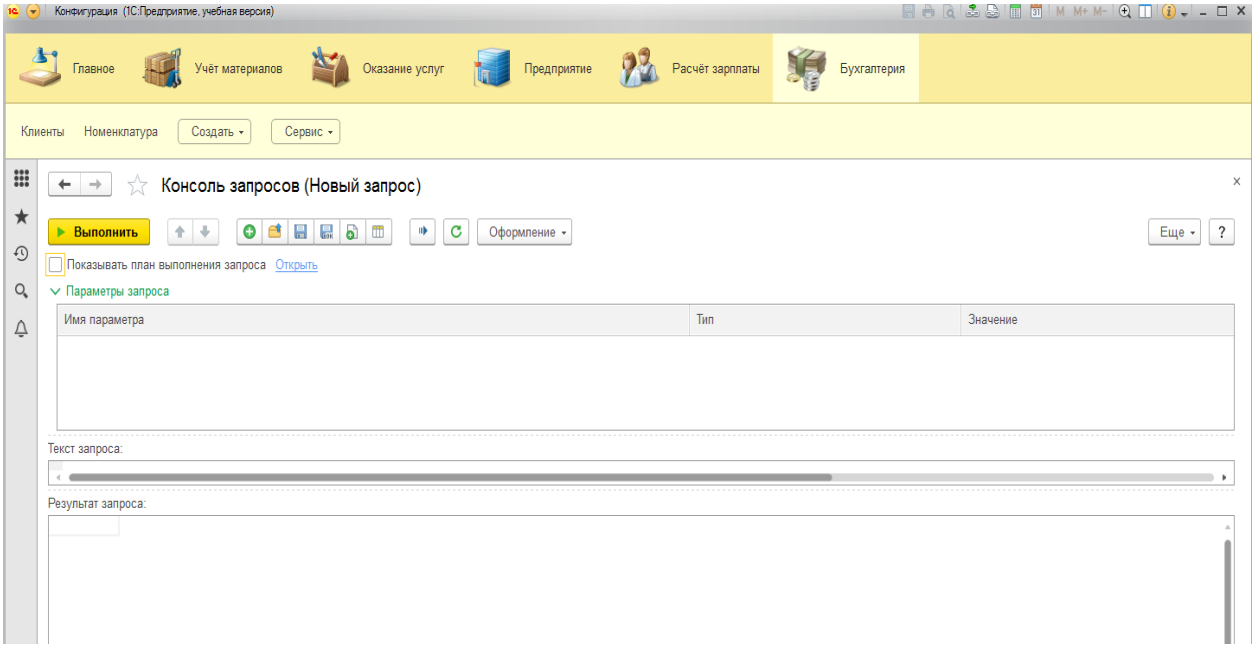



Рис 24.Консоль запросов.

Для начало работы с консолью запросов нужно в начале определиться каким пользователем мы являемся. Мы имеем достаточно глубокие знания необходимые для работы с программой 1С на достаточном уровне и можем писать код в программе 1С для составления запросов или мы просто пользователь, который не умеет писать программы и не значком с программированием в программе 1С и не умеем писать запросы.

После того как мы определимся, то можем приступить к работе. Если мы пользователь с достаточно глубокими знаниями в области программирования в программе 1С, то мы можем писать запросы в консоли запросов для обработки входных данных и получения необходимых нам результатов.

Как видно на представленном рисунке область ввода кода для запроса и область вывода результатов, а также кнопка для запуска нашего запроса. Программист 1С может писать код для запроса в области ввода кода на языке SQL, а не квалифицированный пользователь может воспользоваться конструктором запросов. Это позволит не программисту составит запрос без труда. Ему необходимо лишь выбрать необходимые ему данные, например такие которые представлены на рис. 25.

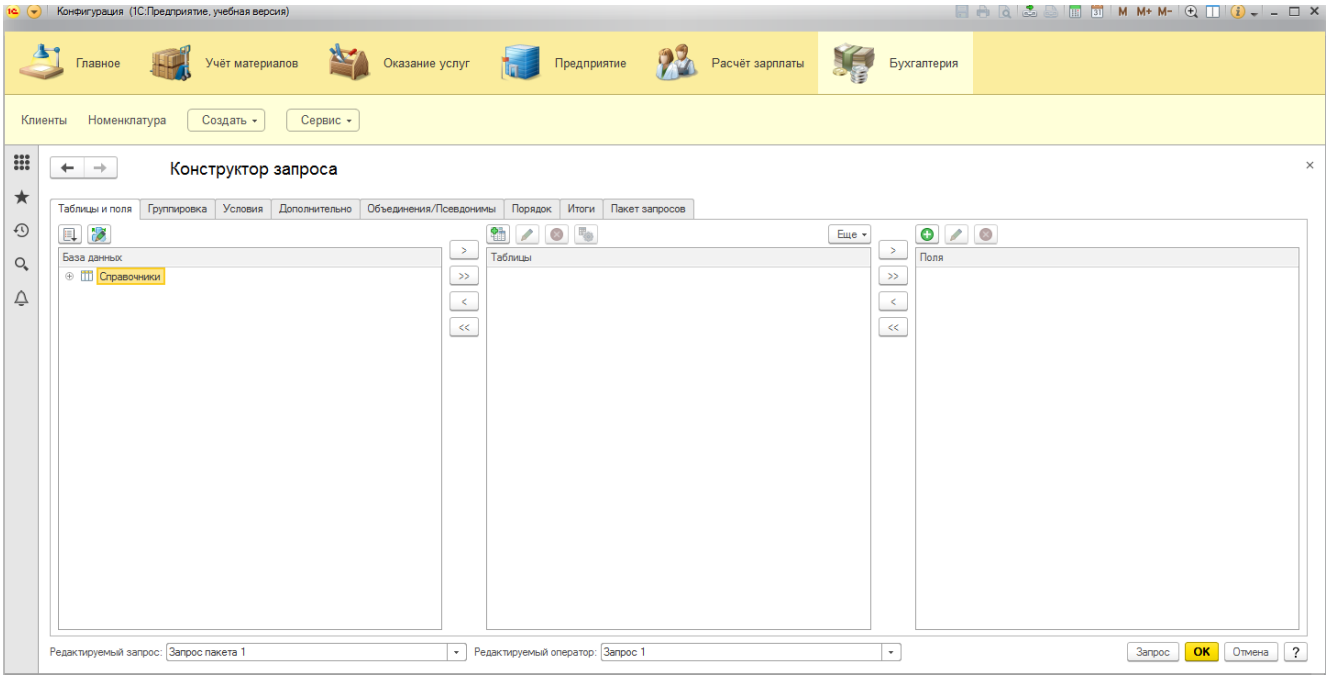



Рис 25.Конструктор запросов.

Выполнив все шаги, пользователь получит результат по своему запросу.

Пример кода для поиска кадров с указанием их минимальной зарплаты представлен на РИС 26.

```
Выбрать
Наименование,
Зарплата
Из Справочник.Кадры
```

ытат запроса (количество строк = 20, время выполнения = 0 с):

**Запрос: Справочник.Кадры (Записей в результате: 20)**

| Наименование | Зарплата |
|---|---|
| Амирова Малика Касимовна | 40 000 |
| Балабеков Тамирлан Ефратович | 30 000 |
| Гаджиев Малик Саласкарович | 40 000 |
| Гасанова Мадина Абсалутдиновна | 60 000 |
| Гасретов Девран Шахвелиевич | 90 000 |
| Гитиномагомедов Гаджи Магомедович | 50 000 |
| Гусейнова Зумрият Рафиковна | 60 000 |
| Дарбишов Шамиль Магарамович | 80 000 |
| Магомедов Али Гусенович | 100 000 |
| Магомедова Аминат Макмагомедовна | 60 000 |
| Магомедова Умгани Алиевна | 150 000 |
| Магомедшапиев Гамзат Ахмедович | 250 000 |
| Махачева Нина Мухучевна | 300 000 |
| Махмудова Фатима Хирамагомедовна | 300 000 |
| Моллалиева Наира Джалиловна | 200 000 |
| Мурсалов Алимет Арифович | 60 000 |
| Омарова Патимат Абдулаевна | 40 000 |

Рис 26. Поиск кадров по уровню зарплаты.

Также можно указать условие в запросе, например чтобы получить кадры с зарплатой больше 80000. Пример представлен на РИС 27.

```
Выбрать
Наименование,
Зарплата
Из Справочник.Кадры
Где Зарплата > 80000
```

езультат запроса (количество строк = 20, время выполнения = 0,001 с):

**Запрос: Справочник.Кадры (Записей в результате: 20)**

| Наименование | Зарплата |
|---|---|
| Амирова Малика Касимовна | 40 000 |
| Балабеков Тамирлан Ефратович | 30 000 |
| Гаджиев Малик Саласкарович | 40 000 |
| Гасанова Мадина Абсалутдиновна | 60 000 |
| Гасретов Девран Шахвелиевич | 90 000 |
| Гитиномагомедов Гаджи Магомедович | 50 000 |
| Гусейнова Зумрият Рафиковна | 60 000 |
| Дарбишов Шамиль Магарамович | 80 000 |
| Магомедов Али Гусенович | 100 000 |
| Магомедова Аминат Макмагомедовна | 60 000 |
| Магомедова Умгани Алиевна | 150 000 |
| Магомедшапиев Гамзат Ахмедович | 250 000 |
| Махачева Нина Мухучевна | 300 000 |
| Махмудова Фатима Хирамагомедовна | 300 000 |
| Моллалиева Наира Джалиловна | 200 000 |
| Мурсалов Алимет Арифович | 60 000 |
| Омарова Патимат Абдулаевна | 40 000 |

Рис 27. Пример запроса с условием.

Можно ещё найти именно определённую запись и большого списка, например выпускника определённого университета с определенного факультета. Пример такой выборки представлен на РИС 28.

Текст запроса:

```
Выбрать
Наименование,
ТрудоваяДеятельность.МинимальнаяЗарплата,
ТрудоваяДеятельность.Оброзование
Из Справочник.Кадры
Где ТрудоваяДеятельность.Оброзование ПОДОБНО "ДГУ Факультет Информатики и Информационных Технологий"
```

Результат запроса (количество строк = 15, время выполнения = 0,002 с):

| | | |
|---|---|---|
| Амирова Малика Касимовна | 40 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Балабеков Тамирлан Ефратович | 30 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гаджиев Малик Саласкарович | 40 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гасанова Мадина Абсалутдиновна | 60 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гасретов Девран Шахвелиевич | 90 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гитиномагомедов Гаджи Магомедович | 50 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гусейнова Зумрият Рафиковна | 60 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Дарбишов Шамиль Магарамович | 80 000 | ДГУ Факультет Информатики и Информационных Технологий |

Рис 28. Пример запроса для поиска студентов факультета информатики.

## 2.3. Методические указания пользователю.

Для начало работы с консолью запросов нужно вначале определиться каким пользователем мы являемся. Мы имеем достаточно глубокие знания необходимые для работы с программой 1С на достаточном уровне и можем писать код в программе 1С для составления запросов или мы просто пользователь, который не умеет писать программы и не значком с программированием в программе 1С и не умеем писать запросы.

После того как мы определимся, то можем приступить к работе. Если мы пользователь с достаточно глубокими знаниями в области программирования в программе 1С, то мы можем писать запросы в консоли запросов для обработки входных данных и получения необходимых нам результатов.

Для начала мы принимаем решение о выборе источнике данных для нашего запроса, например, который представлен на Рис 29.

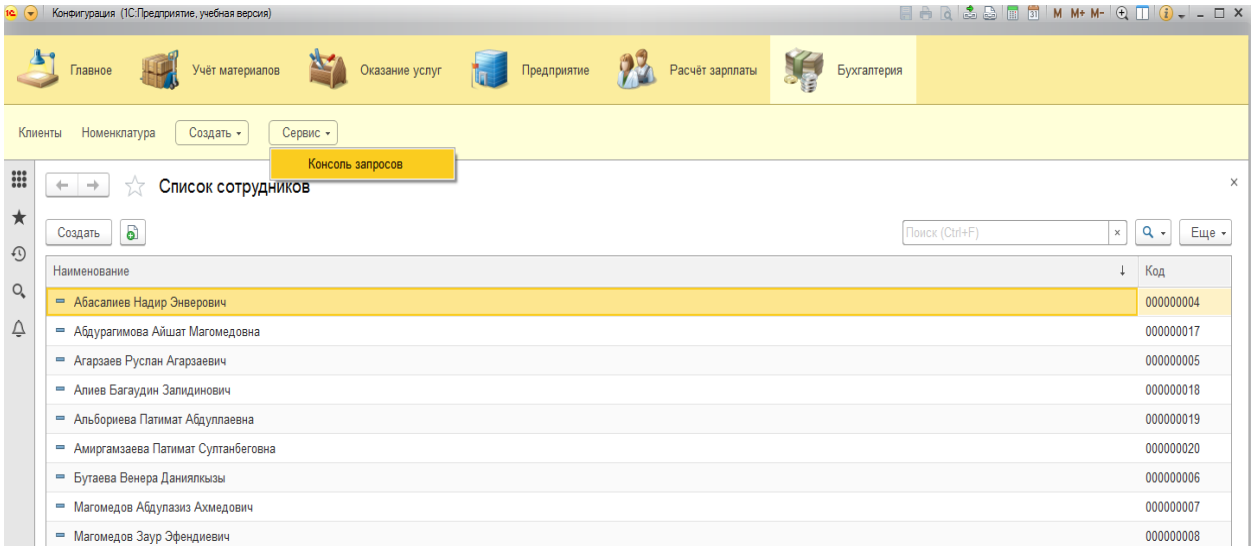



Рис 29.Источник данных для запроса.

Далее после выбора источника данных для нашего запроса мы переходим к консоли запросов и перед нами предстаёт главное окно нашей консоли запросов, которое представлено на Рис 20.

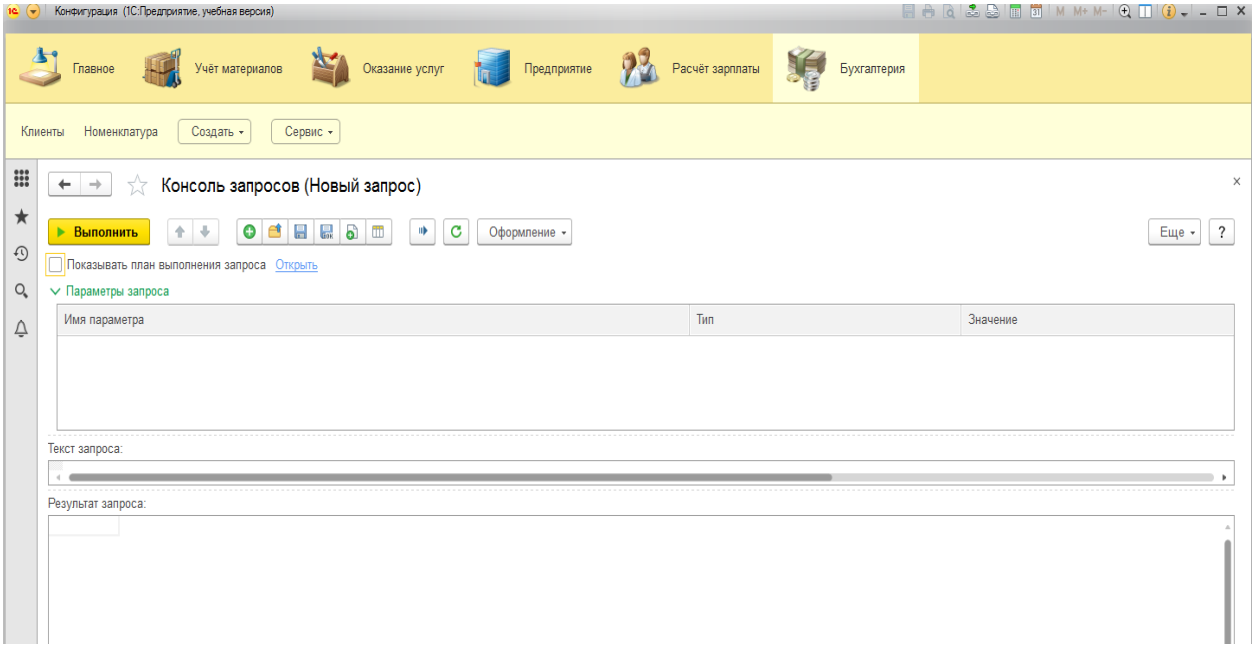



Рис 30. Главное окно консоли запросов.

В окно ввода мы вводим наш запрос. Мы можем писать код на языке SQL как на английском языке, так и на русском что является очень удобным так, как не все пользователи хорошо знают английский язык. В области результата запроса мы получаем результат нашего запроса оформленного в виде таблицы. Как это выглядит на практике можно показать на нескольких наглядных примерах.

Пример кода для поиска кадров с указанием их минимальной зарплаты представлен на РИС 31.

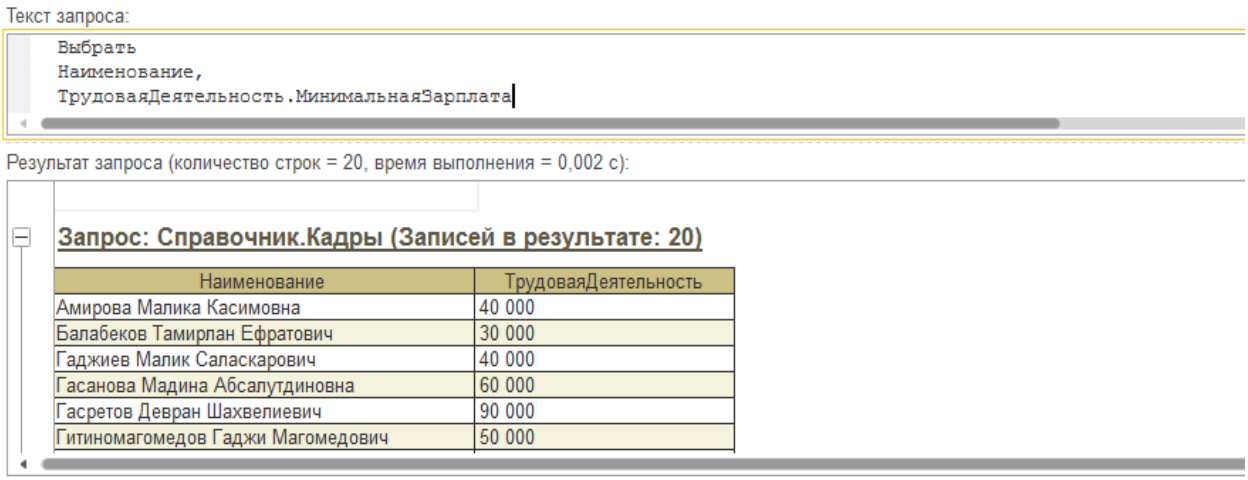



РИС 31.Поиск кадров по уровню зарплаты.

Также можно указать условие в запросе, например чтобы получить кадры с зарплатой больше 80000. Пример представлен на РИС 32.

```
Выбрать
Наименование,
ТрудоваяДеятельность.МинимальнаяЗарплата
Из Справочник.Кадры
Где ТрудоваяДеятельность.МинимальнаяЗарплата > 80000
```

Результат запроса (количество строк = 7, время выполнения = 0,002 с):

**Запрос: Справочник.Кадры (Записей в результате: 7)**

| Наименование | ТрудоваяДеятельность |
|---|---|
| Гасретов Девран Шахвелиевич | 90 000 |
| Магомедов Али Гусенович | 100 000 |
| Магомедова Умгани Алиевна | 150 000 |
| Магомедшапиев Гамзат Ахмедович | 250 000 |

РИС 32.Пример запроса с условием.

Можно ещё найти именно определённую запись и большого списка, например выпускника определённого университета с определенного факультета. Пример такой выборки представлен на РИС 33.

Текст запроса:

```
Выбрать
Наименование,
ТрудоваяДеятельность.МинимальнаяЗарплата,
ТрудоваяДеятельность.Оброзование
Из Справочник.Кадры
Где ТрудоваяДеятельность.Оброзование ПОДОБНО "ДГУ Факультет Информатики и Информационных Технологий"
```

Результат запроса (количество строк = 15, время выполнения = 0,002 с):

| | | |
|---|---|---|
| Амирова Малика Касимовна | 40 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Балабеков Тамирлан Ефратович | 30 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гаджиев Малик Саласкарович | 40 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гасанова Мадина Абсалутдиновна | 60 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гасретов Девран Шахвелиевич | 90 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гитиномагомедов Гаджи Магомедович | 50 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Гусейнова Зумрият Рафиковна | 60 000 | ДГУ Факультет Информатики и Информационных Технологий |
| Дарбишов Шамиль Магарамович | 80 000 | ДГУ Факультет Информатики и Информационных Технологий |

РИС 33. Пример запроса для поиска студентов факультета информатики.

Это всё относится для профессиональных пользователей, а для простых пользователей не обладающими такими глубокими знаниями тоже есть возможность создавать запросы, но для этого им необходимо использовать конструктор запросов. С помощью конструктора запросов можно легко и быстро создать запрос в программе 1С. Для этого необходимо лишь выполнить следующие шаги.

Во-первых, необходимо открыть конструктор запросов и первое что мы

увидим это главное окно конструктора запросов, которое представлено на Рис 34.

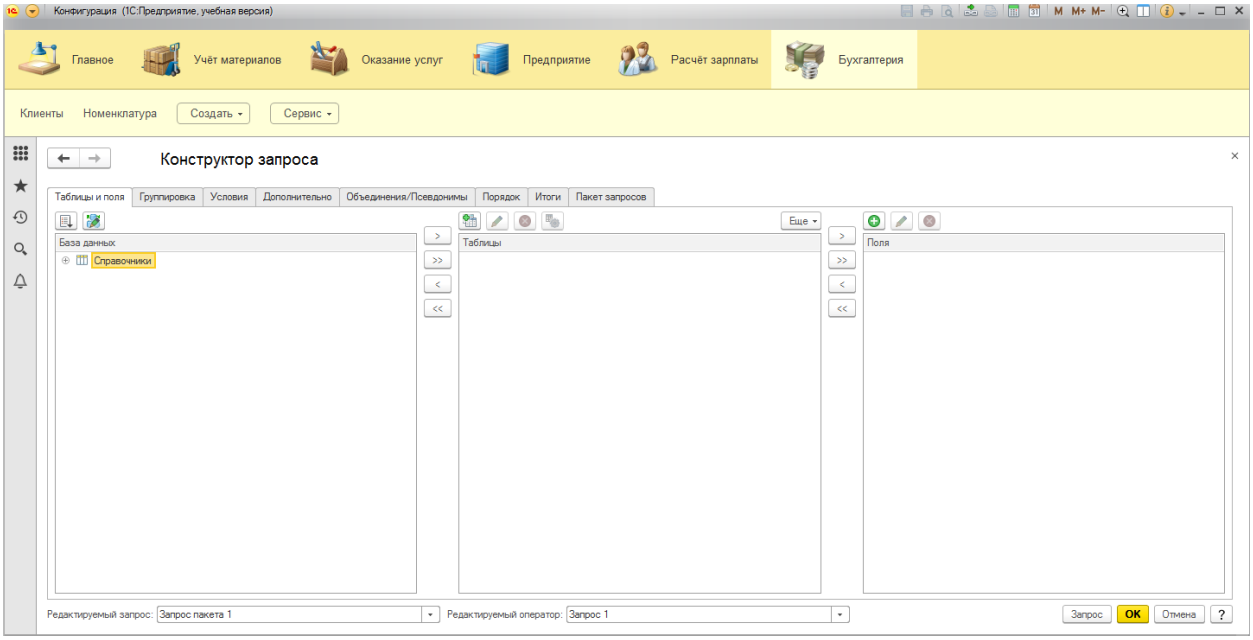



Рис 34.Главное окно конструктора запросов.

Далее нам нужно выбрать источник данных из представленного списка, выбрать таблицы и поля, по которым будет проходить обработка данных. Всё это должно выглядеть примерно, так как представлено на Рис 35.

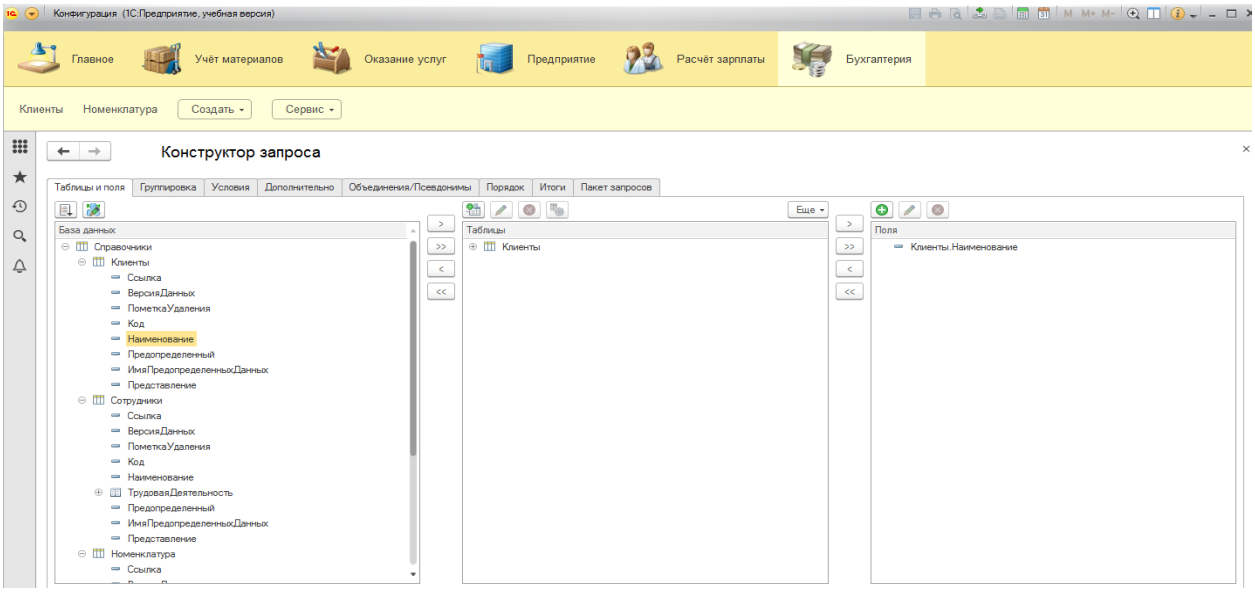



Рис 35.Выбор источника данных.

Затем нам нужно указать условия отбора если таковые имеются так, как это представлено на Рис 36.

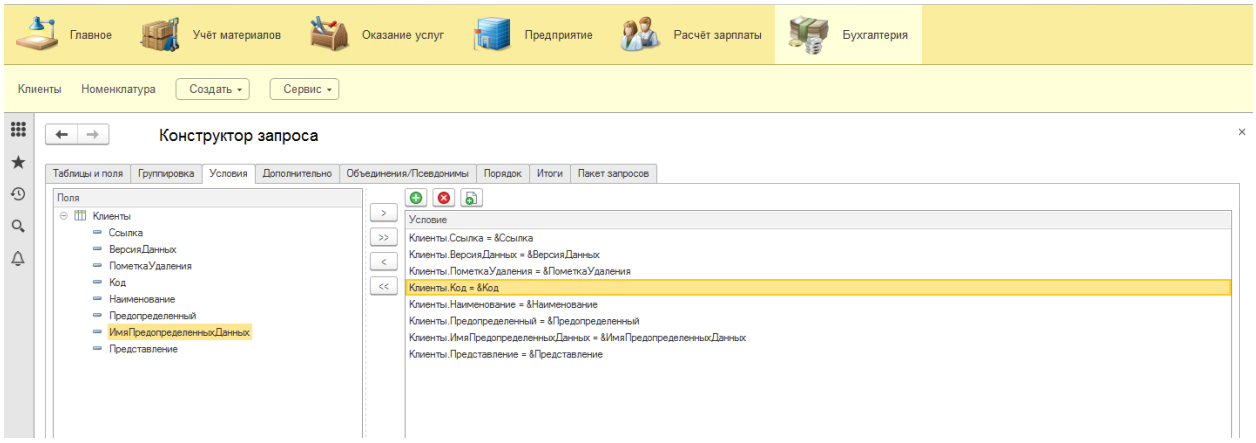



Рис 36.Условие отбора.

Затем можно определить порядок так, как это представлено на Рис 37.

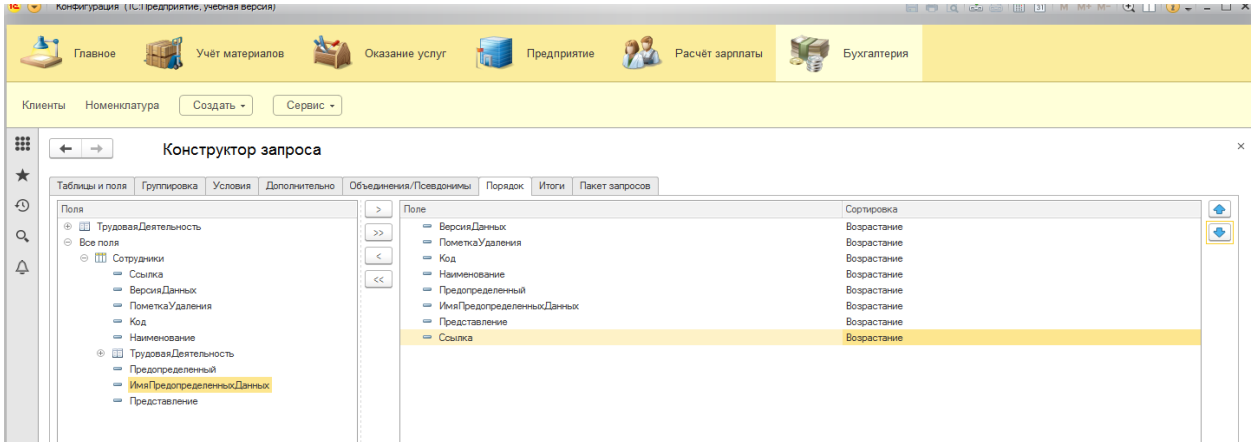



Рис 37. Порядок.

После этого нам необходимо лишь определить итоги и мы получим результат нашего запроса в виде таблицы.

## Заключение

По итогам проведенного исследования в рамках выпускной

квалификационной работы можно сделать следующие выводы и предположения.

Внедрение новых технологий (производственных, финансовых, управленческих, социальных и любых других) может привести к огромному количеству позитивных эффектов и последствий для экономики:

- повышение производительности труда;
- повышение капитализации;
- улучшение качества жизни;
- формирование новых рынков;
- повышение эффективности утилизации ресурсов (активов, капитала, компетенций);
- повышение конкурентоспособности;
- повышение безопасности.

Следует отметить, что современные тенденции в развитии средств таковы, что актуальным становится не просто появление новых гибких и мощных средств разработки, а создание семейств таких продуктов с похожими средами и принципами создания приложений. Современная технологическая платформа "1С:Предприятие 8.3." обеспечивает широкие возможности по настройке системы на решение любых отраслевых и специализированных задач, по адаптации к специфике учета на конкретном предприятии и по интеграции системы с программным обеспечением и оборудованием других производителей.

В результате проделанной работы была создана информационная база в программе 1С:Предприятие, были изучены теоретико-методологические аспекты разработки конфигурации. Изучены особенности разработки подсистем и справочников. Для создания подсистемы нужно перейти по вкладке общие и найти пункт подсистемы. По клику правой кнопки мыши поэтому пункты у нас появится возможность создать новую подсистему. Далее необходимо создать справочники. Они представляют собой элементы, предназначенные для хранения информации и отражаются в подсистемах. Для создания справочника

необходимо кликнуть правой кнопкой мыши по пункту справочники и выбрать создать справочник. При создании справочника можно многое определить, например, в какой подсистеме он будет отражаться, определить тип вводимых данных и так далее.

Описан процесс разработки конфигурации в программе 1С: Предприятие. В программе 1С: Предприятие были разработаны следующие подсистемы: Бухгалтерия, Учёт Материалов, Оказание Услуг, Предприятие, Расчёт Зарплаты. Также были разработаны следующие справочники: Клиенты, Сотрудники, Номенклатура, Склады, Кадры. Было обеспеченно наполнение контентом.

Было выбрано средство реализации автоматизации и поиска кадров. Выбор пал на консоль запросов. Она представляет собой удобное средство для обработки данных и выдачи результатов в соответствии с нашими требованиями. Помимо этого был освоен язык запросов SQL. Так как именно на этом языке необходимо писать коды для наших запросов. Для этого была использована различная литература и сайты посвященные написанию различных запросов используя консоль запросов. Для начало работы с консолью запросов нужно в начале определиться каким пользователем мы являемся. Мы имеем достаточно глубокие знания необходимые для работы с программой 1С на достаточном уровне и можем писать код в программе 1С для составления запросов или мы просто пользователь, который не умеет писать запросы. В окно ввода мы вводим наш запрос. Мы можем писать код на языке SQL как на английском языке, так и на русском что является очень удобным так, как не все пользователи хорошо знают английский язык. В области результата запроса мы получаем результат нашего запроса оформленного в виде таблицы.

Разработанный в рамках выпускной квалификационной работы программный модуль для поиска и автоматизации кадров можно использовать по назначению. Использованная мною консоль запросов и конструктор запросов упрощает поиск кадров по необходимым нам критериям и позволяет нам делать запросы на выборку по разным критериям отбора. Работать с этой конфигурацией будет понятно и удобно как профессиональному пользователю

умеющему программировать и имеющему достаточно глубокие знания в этой области, так и простому пользователю не знакомому даже с основами программирования и рабой с программой 1С.

**Литература:**


1. "1С:Предприятие 8.3. Практическое пособие разработчика. Примеры и типовые приемы"/ Радченко М. Г. ,Хрусталева Е. Ю.-Москва,2013-965 с.
2. 1С Зарплата и управление персоналом 8.3. 100 уроков для начинающих / А.А. Гладкий. - М.: Эксмо, 2015. - 272с.
3. 1С: Программирование для начинающих. Детям и родителям, менеджерам и руководителям. Разработка в системе «1С: Предприятие 8.3». / М.Г.Радченко, -Москва,2017-780 с.
4. SQL — язык реляционных баз данных/ Владимир Кара-Ушанов, Издательство Уральского университета, 2016- 156 с.
5. SQL для профессионалов. Программирование / Джо Селко. - М.: ЛОРИ, 2015. - 464с.
6. SQL запросы.{Электронный ресурс }//URL: https://habr.com/ru/post/480838/ (Дата обращения 18.04.2020).

7. SQL. Операторы SQL и программы PL/SQL / Джейсон Прайс. - М.: ЛОРИ, 2016. - 660с.

8. SQL. Полное руководство / Оппель Эндрю Дж.. - М.: Диалектика / Вильямс, 2016. - 902с.

9. SQL. Полное руководство/ Грофф Дж. Р., Вайнберг П.Н., Оппелъ Э. Дж., Вильямс, 2015-959 с.

10. SQL-запросы для простых смертных. Практическое руководство по манипулированию данными в SQL / Майкл Дж. Хернандес, Джон Л. Вьескас. - М.: ЛОРИ, 2015. - 480с.

11. Автоматизация производства, какую программу выбрать ?{Электронный ресурс}//URL: https://vc.ru/services/57133-avtomatizaciya-proizvodstva-kakuyu-programmu-vybrat. (Дата обращения 30.03.2020 )

12. Введение в «Цифровую» экономику/ А.В. Кешелава В.Г. Буданов, В.Ю. Румянцев

13. Гурвиц, Джудит. Просто о больших данных /Гурвиц Джудит, Ньюджен Алан, Халпер Ферн, Кауфман Марсия: [перевод с английского]. – Москва: Эксмо, 2015. – 400 с. – (Библиотека Сбербанка, Т.58).

14. Денисов А.Ф., Кардаш Д.С. Анализ практик применения цифровых технологий в отборе персонала // Экономика и управление. 2018. № 6 (152). С. 26–37.

15. др.; под общ. ред. А.В. Кешелава; гл. «цифр.» конс. И.А. Зимненко. – ВНИИГеоси-стем, 2017. – 28 с. (На пороге «цифрового будущего». Книга первая).

16. Запросы 1С.Полный курс. {Электронный ресурс}//URL: http://zapros-1c-8.ru/ (Дата обращения 20.04.2020).

17. Зарина И. Цифровая трансформация в HR [Электронный ресурс] – URL: http://hr-portal.ru/blog/cifrovaya-transformaciya-v-hr (дата обращения: 08.02.18).

18. Информационные технологии в управлении персоналом. {Электронный ресурс}//URL: https://docplayer.ru/37521179-Informacionnye-tehnologii-v-upravlenii-personalom.html. (Дата обращения 10.12.2019)

19. Информационные технологии в юридической деятельности.{Электронный ресурс}//URL: http://referat.co/ref/633882/read. (Дата обращения 10.12.2019)

20. Кадровая политика организций/Н.Н.Козак. -Москва ,2016-80 с.

21. Киреев В.Э. Влияние цифровой экономики на ключевые направления управления персо-налом // Формирование общекультурных и профессиональных компетенций финансиста. М., 2018. С. 85-93.

22. Книги по 1С программированию.{Электронный ресурс} //URL https://programmist1s.ru/knigi-1s-dlya-programmistov/ (Дата обращения 16.04.2020).

23. Механизм запросов. {Электронный ресурс}//URL: https://v8.1c.ru/platforma/mehanizm-zaprosov/ (Дата обращения 18.04.2020).

24. Модуль 1С. {Электронный ресурс}//URL: http://howknow1c.ru/programmirovanie-1c/modul-1s.html. (Дата обращения 10.12.2019)

25. Нагибина Н.И., Щукина А.А. HR-Digital: цифровые технологии в управлении человеческими ресурсами // Интернет-журнал «Науковедение». 2017. Том 9. №1. URL: http://naukovedenie.ru/PDF/24EVN117.pdf (дата обращения 29.10.2018)

26. Описание языка запросов 1С.{Электронный ресурс }//URL: https://programmist1s.ru/yazyik-zaprosa-1s/ (Дата обращения 20.04.2020).

27. Освой самостоятельно SQL за 10 минут / Бен Форта. - М.: Вильямс, 2015. - 288с.

28. Основы запросов в языке 1С. {Электронный ресурс}//URL: https://helpme1c.ru/zaprosy-v-yazyke-1s-8-v-primerax (Дата обращения 20.04.2020).

29. Основы управления персоналом / В. В. Лукашевич — «КноРус», 2015-272 с.

30. Особенности языка запросов 1С. {Электронный ресурс}//URL: https://infostart.ru/public/204054/ (Дата обращения 20.04.2020).

31. Понятие кадрового потенциала и его влияние на эффективность деятельности промышленного предприятия. {Электронный ресурс} //URL: https://www.cfin.ru/bandurin/article/sbrn08/07.shtml. (Дата обращения 3.12.2019)

32. Программа для отдела кадров. {Электронный ресурс} //URL: https:// programka.net/sovety/1s-kadry . (Дата обращения 03.02.2020)

33. Программирование в 1С 8.3 с нуля – краткий самоучитель. {Электронный ресурс}//URL: https://1s83.info/programmirovanie/programmirovanie-1s-s-nulya.html (Дата обращения 16.04.2020)

34. Разработка интерфейса прикладных решений на платформе «1С: Предприятие 8». / А.В.Ажеронок, А.В.Островерх, М.Г.Радченко, Е.Ю.Хрусталёва, - Москва : ООО «1С- Паблишинг», 2018. – 902с.

35. Система 1С:Предприятие 8. {Электронный ресурс} //URL: http://www.softmark.ru/catalog/1c/ . (Дата обращения 3.12.2019)

36. Управление кадровым потенциалом предприятия в современных условиях: учебное пособие/ И. В. Вотякова. – Северск: СТИ НИЯУ МИФИ, 2015. – 120с. (Дата обращения 3.12.2019)

37. Управление персоналом / В.К. Потемкин. - СПб.: Питер, 2019. - 32с. (Дата обращения 3.12.2019)

38. Управление персоналом / Т.Ю. Базаров. - М.: Academia, 2017. - 32с. (Дата обращения 3.12.2019)

39. Управление персоналом : учебное пособие / А.Я. Кибанов. — 6-е изд., стер. — Москва : КНОРУС, 2018. — 202 с. (Дата обращения 3.12.2019)

40. Управление персоналом организации / Н.В. Фёдорова, О.Ю. Минченкова. - М.: КноРус, 2018. - 190с. (Дата обращения 3.12.2019)

41. Управление персоналом организации. Краткий курс для бакалавров / Н.И. Архипова, О.Л. Седова. - М.: Проспект, 2016. - 224с. (Дата обращения 3.12.2019)

42. Управление персоналом организации: Учебник / Под ред. Кибанова А.Я.. - М.: Инфра-М, 2017. - 36с. (Дата обращения 3.12.2019)

43. Управление персоналом организации: Учебник / Под ред. Кибанова А.Я., Баткаева И.А., Ворожейкин И.Е. и др. - М.: Инфра-М, 2018. - 64с (Дата обращения 3.12.2019)

44. Управление персоналом организации: Учебник для бакалавров / А.В. Дейнека. - М.: Дашков и К, 2015. - 288с. (Дата обращения 3.12.2019)

45. Управление персоналом/ А.В.Дайнека.-2016-233 с. (Дата обращения 3.12.2019)

46. Управление персоналом/Н.Н.Козак. -Москва ,2016-370 с. (Дата обращения 3.12.2019)

47. Язык PL/SQL/Задворьев И.С.-2017-180 с. (Дата обращения 20.04.2020)

48. Язык SQL. Базовый курс/ Бондарев В., Postgres Professional, 2017- 256 с. (Дата обращения 20.04.2020)

49. Язык запросов 1С:Предприятие 8/Е.Ю.Хрусталёва.-Москва,2013-369 с. (Дата обращения 20.04.2020)

# Приложение

Приложение 1

```
#Область ОбработчикиСобытийФормы

&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

	Если Параметры.Свойство("АвтоТест") Тогда
		Возврат;
	КонецЕсли;

	ПараметрыПередачи = ПолучитьИзВременногоХранилища(Параметры.АдресХранилища);
	Объект.ИспользоватьАвтосохранение = ПараметрыПередачи.ИспользоватьАвтосохранение;
	Объект.ПериодАвтосохранения = ПараметрыПередачи.ПериодАвтосохранения;
	Объект.ВыводитьВРезультатахЗапросаЗначенияСсылок = ПараметрыПередачи.ВыводитьВРезультатахЗапросаЗначенияСсылок;
	Объект.ТипОбхода = ПараметрыПередачи.ТипОбхода;
	Объект.ЧередованиеЦветовВРезультатеЗапроса = ПараметрыПередачи.ЧередованиеЦветовВРезультатеЗапроса;

	Элементы.ТипОбхода.СписокВыбора.Добавить("Авто");
	Элементы.ТипОбхода.СписокВыбора.Добавить("Прямой");
КонецПроцедуры

#КонецОбласти

#Область ОбработчикиКомандФормы

&НаКлиенте
Процедура Записать(Команда)
	ПараметрыПередачи = ПоместитьНастройкиВСтруктуру();

	// Передача в открывающую форму.
	Закрыть();
	Владелец = ЭтотОбъект.ВладелецФормы;

	Оповестить("ПередатьПараметрыНастроек" , ПараметрыПередачи);
	Оповестить("ПередатьПараметрыНастроекАвтоСохранения");
КонецПроцедуры

#КонецОбласти

#Область СлужебныеПроцедурыИФункции
&НаСервере
Функция ОбъектОбработки()
	Возврат РеквизитФормыВЗначение("Объект");
КонецФункции

&НаСервере
```

```
	Функция ПоместитьНастройкиВСтруктуру()
		ПараметрыПередачи = Новый Структура;
		ПараметрыПередачи.Вставить("АдресХранилища",
ОбъектОбработки().ПоместитьНастройкиВоВременноеХранилище(Объект));
		Возврат ПараметрыПередачи;
	КонецФункции

	#КонецОбласти

	#Область ОбработчикиСобытий

	&НаСервере
	Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

		Если Параметры.Свойство("АвтоТест") Тогда
			Возврат;
		КонецЕсли;

		// Считывание параметров передачи.
		ПараметрыПередачи = ПолучитьИзВременногоХранилища(Параметры.АдресХранилища);
		Объект.Запросы.Загрузить(ПараметрыПередачи.Запросы);
		Объект.Параметры.Загрузить(ПараметрыПередачи.Параметры);
		Объект.ИмяФайла = ПараметрыПередачи.ИмяФайла;
		ИдентификаторТекущегоЗапроса = ПараметрыПередачи.ИдентификаторТекущегоЗапроса;
		ИдентификаторТекущегоПараметра =
ПараметрыПередачи.ИдентификаторТекущегоПараметра;

		ОбработкаОбъект = ОбъектОбработки();
		Объект.ДоступныеТипыДанных =
ОбработкаОбъект.Метаданные().Реквизиты.ДоступныеТипыДанных.Тип;

		СписокТипов = ОбъектОбработки().СформироватьСписокТипов();
		ОбработкаОбъект.ФильтрацияСпискаТипов(СписокТипов, "");

		Фильтр = Новый Структура;
		Фильтр.Вставить("Идентификатор", ИдентификаторТекущегоЗапроса);
		СтрокиЗапросовСИдентификатор = Объект.Запросы.НайтиСтроки(Фильтр);
		Если СтрокиЗапросовСИдентификатор.Количество() > 0 Тогда
			Элементы.Запросы.ТекущаяСтрока =
СтрокиЗапросовСИдентификатор.Получить(0).ПолучитьИдентификатор();
		КонецЕсли;
		Заголовок = НСтр("ru = 'Выбрать запрос'");

	КонецПроцедуры

	&НаКлиенте
	Процедура ЗапросыПередНачаломДобавления(Элемент, Отказ, Копирование, Родитель, Группа)
		Отказ = Истина;

		ЭлементКопирования = Элемент.ТекущиеДанные;

		ИмяЗапросаПоУмолчанию = ЭтотОбъект.ВладелецФормы.ИмяЗапросаПоУмолчанию;
		ИдентификаторЗапроса = Новый УникальныйИдентификатор;

		Запрос = Объект.Запросы.Добавить();
		Запрос.Имя = ИмяЗапросаПоУмолчанию;
		Запрос.Идентификатор = ИдентификаторЗапроса;
```

```
		Если Копирование Тогда
			ИмяНовогоЗапроса = СформироватьИмяКопииЗапроса(ЭлементКопирования.Имя);
			Запрос.Имя = ИмяНовогоЗапроса;
			Запрос.Текст = ЭлементКопирования.Текст;
			ИдентификаторТекущегоЗапроса = ЭлементКопирования.Идентификатор;

			// Копирование параметров
			Фильтр = Новый Структура;
			Фильтр.Вставить("ИдентификаторЗапроса", ИдентификаторТекущегоЗапроса);
			МассивПараметров = Объект.Параметры.НайтиСтроки(Фильтр);
			Для каждого Стр Из МассивПараметров Цикл
				ЭлементПараметр = Объект.Параметры.Добавить();
				ЭлементПараметр.Идентификатор = Новый УникальныйИдентификатор;
				ЭлементПараметр.ИдентификаторЗапроса = ИдентификаторЗапроса;
				ЭлементПараметр.Имя = Стр.Имя;
				ЭлементПараметр.Тип = Стр.Тип;
				ЭлементПараметр.Значение = Стр.Значение;
				ЭлементПараметр.ТипВФорме = Стр.ТипВФорме;
				ЭлементПараметр.ЗначениеВФорме = Стр.ЗначениеВФорме;
			КонецЦикла;
		КонецЕсли;

		ВладелецФормы.Модифицированность = Истина;
	КонецПроцедуры

	// Обработчик перед удалением Запроса.
	// Удаляет параметры для данного запроса.
	//
	&НаКлиенте
	Процедура ЗапросыПередУдалением(Элемент, Отказ)
		ПараметрыВФорме = Объект.Параметры;
		ИдентификаторУдаляемогоЗапроса = Элементы.Запросы.ТекущиеДанные.Идентификатор;

		КоличествоСтрок = ПараметрыВФорме.Количество() - 1;
		Пока КоличествоСтрок >= 0 Цикл
			ТекущийПараметр = ПараметрыВФорме.Получить(КоличествоСтрок);
			Если ТекущийПараметр.ИдентификаторЗапроса = ИдентификаторУдаляемогоЗапроса
Тогда
				ПараметрыВФорме.Удалить(КоличествоСтрок);
				Модифицированность = Истина;
			КонецЕсли;
			КоличествоСтрок = КоличествоСтрок - 1;
		КонецЦикла;

		ВладелецФормы.Модифицированность = Истина;
	КонецПроцедуры

	&НаКлиенте
	Процедура ЗапросыВыбор(Элемент, ВыбраннаяСтрока, Поле, СтандартнаяОбработка)
		ОбработкаВыбораЗапроса();
	КонецПроцедуры

	&НаКлиенте
	Процедура ЗапросыИмяПриИзменении(Элемент)
		ВладелецФормы.Модифицированность = Истина;
	КонецПроцедуры

	&НаКлиенте
```

```
Процедура СравнитьРезультатыЗапросов(Команда)
	#Если Не ВебКлиент И Не ТонкийКлиент Тогда
	ВыделенныеЗапросы = Элементы.Запросы.ВыделенныеСтроки;
	Если ВыделенныеЗапросы.Количество() <> 2 Тогда
		ПоказатьПредупреждение(, НСтр("ru = 'Для сравнения необходимо выбрать только 2 запроса", "Объект'"));
		Возврат;
	Иначе
		ИдентификаторСтрокиПервогоЗапроса = ВыделенныеЗапросы.Получить(0);
		ИдентификаторСтрокиВторогоЗапроса = ВыделенныеЗапросы.Получить(1);
	КонецЕсли;

	ИдентификаторПервогоЗапроса = Объект.Запросы.НайтиПоИдентификатору(ИдентификаторСтрокиПервогоЗапроса).Идентификатор;
	ИдентификаторВторогоЗапроса = Объект.Запросы.НайтиПоИдентификатору(ИдентификаторСтрокиВторогоЗапроса).Идентификатор;

	ФайлПервогоЗапроса = Неопределено;
	ФайлВторогоЗапроса = Неопределено;

	ПолучитьТабличныеДокументыСравниваемыхЗапросов(ИдентификаторПервогоЗапроса, ИдентификаторВторогоЗапроса, ФайлПервогоЗапроса, ФайлВторогоЗапроса);

	Если ТипЗнч(ФайлПервогоЗапроса) <> Неопределено
		И ТипЗнч(ФайлВторогоЗапроса) <> Неопределено Тогда
		// Сравниваются два файла.
		Сравнение = Новый СравнениеФайлов;
		Сравнение.СпособСравнения = СпособСравненияФайлов.ТабличныйДокумент;
		Сравнение.ПервыйФайл = ФайлПервогоЗапроса;
		Сравнение.ВторойФайл = ФайлВторогоЗапроса;
		Сравнение.ПоказатьРазличияМодально();

		УдалитьФайлы(ФайлПервогоЗапроса);
		УдалитьФайлы(ФайлВторогоЗапроса);
	КонецЕсли;
	#Иначе
		ПоказатьПредупреждение(, НСтр("ru = 'Сравнивать результаты можно только в режиме толстого клиента.", "Объект'"));
	#КонецЕсли

КонецПроцедуры

////////////////////////////////////////////////////////////////////////////////
// ОБЩИЕ КОМАНДЫ

&НаКлиенте
Процедура СохранитьЗапросыВДругойФайл(Команда)
	СохранитьФайлЗапроса();
КонецПроцедуры

&НаКлиенте
Процедура СохранитьЗапросыВФайл(Команда)
	СохранитьФайлЗапроса(Объект.ИмяФайла);
КонецПроцедуры

&НаКлиенте
Процедура ВосстановитьЗапросыИзФайла(Команда)
	ОбработкаЧтенияФайла(Истина);
```

```
		ВладелецФормы.Модифицированность = Ложь;
	КонецПроцедуры

	&НаКлиенте
	Процедура ВыбратьЗапрос(Команда)
		ОбработкаВыбораЗапроса();
	КонецПроцедуры

	&НаКлиенте
	Процедура ДобавитьЗапросыИзФайла(Команда)
		ОбработкаЧтенияФайла(Ложь);
		ВладелецФормы.Модифицированность = Истина;
	КонецПроцедуры

	////////////////////////////////////////////////////////////////////////////
	// ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ

	&НаСервере
	Функция ОбъектОбработки()
		Возврат РеквизитФормыВЗначение("Объект");
	КонецФункции

	&НаСервере
	Функция ПоместитьЗапросыВСтруктуру(ИдентификаторЗапроса, ИдентификаторПараметра)
		ПараметрыПередачи = Новый Структура;
		ПараметрыПередачи.Вставить("АдресХранилища",
ОбъектОбработки().ПоместитьЗапросыВоВременноеХранилище(Объект, ИдентификаторЗапроса,
ИдентификаторПараметра));
		Возврат ПараметрыПередачи;
	КонецФункции

	&НаКлиенте
	Процедура ОбработкаВыбораЗапроса()
		ТекущаяСтрока = Элементы.Запросы.ТекущаяСтрока;
		Если ТекущаяСтрока <> Неопределено Тогда
			ТекущийЗапрос = Элементы.Запросы.ТекущиеДанные;
			ИдентификаторТекущегоЗапроса = ТекущийЗапрос.Идентификатор;
			ПараметрыПередачи =
ПоместитьЗапросыВСтруктуру(ИдентификаторТекущегоЗапроса, ИдентификаторТекущегоПараметра);

			// Передача в открывающую форму.
			Закрыть();

			Оповестить("ВыгрузитьЗапросыВРеквизиты", ПараметрыПередачи);
			Оповестить("ОбновитьФормуКлиент");
		Иначе
			ПоказатьСообщениеПользователю(НСтр("ru ='Выберите запрос.'"), "Объект");
		КонецЕсли;
	КонецПроцедуры

	// Сохранение
	&НаКлиенте
	Процедура СохранитьФайлЗапроса(ИмяФайла = "")

		ДополнительныеПараметры = Новый Структура("ИмяФайла", ИмяФайла);
		#Если Не ВебКлиент Тогда
			// В тонком и толстом клиентах расширение подключено всегда.
			СохранитьФайлЗапросаЗавершение(Истина, ДополнительныеПараметры);
```

```
                Возврат;
            #КонецЕсли

            Оповещение = Новый
ОписаниеОповещения("НачатьПодключениеРасширенияРаботыСФайламиЗавершение", ЭтотОбъект,
                Новый ОписаниеОповещения("СохранитьФайлЗапросаЗавершение", ЭтотОбъект,
ДополнительныеПараметры));
            НачатьПодключениеРасширенияРаботыСФайлами(Оповещение);

        КонецПроцедуры

        &НаКлиенте
        Процедура СохранитьФайлЗапросаЗавершение(Результат, ДополнительныеПараметры) Экспорт

            ИмяФайла = ДополнительныеПараметры.ИмяФайла;
            Если Результат Тогда

                Получаемые = Новый Массив;
                Получаемые.Добавить(Новый ОписаниеПередаваемогоФайла(ИмяФайла,
СохранитьЗапросыВоВременноеХранилище(Объект)));

                ОписаниеОповещенияПолученияФайлов = Новый
ОписаниеОповещения("ПолучениеФайловЗавершение", ЭтотОбъект);
                Если ЗначениеЗаполнено(ИмяФайла) Тогда
                    НачатьПолучениеФайлов(ОписаниеОповещенияПолученияФайлов,
Получаемые, ИмяФайла, Ложь);
                Иначе
                    Диалог = Новый
ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Сохранение);
                    Диалог.Заголовок = НСтр("ru = 'Выберите файл запросов'");
                    Диалог.ПредварительныйПросмотр = Ложь;
                    Диалог.Фильтр = НСтр("ru = 'Файл запросов (*.q1c)|*.q1c'");
                    Диалог.Расширение = "q1c";
                    Диалог.ПроверятьСуществованиеФайла = Истина;
                    Диалог.МножественныйВыбор = Ложь;

                    НачатьПолучениеФайлов(ОписаниеОповещенияПолученияФайлов,
Получаемые, Диалог, Истина);
                КонецЕсли;
            Иначе
                ПоказатьСообщениеПользователю(НСтр("ru = 'Без расширения работы с файлами
невозможно работать с файлами.'"), "Объект");
            КонецЕсли;

        КонецПроцедуры

        &НаКлиенте
        Процедура ПолучениеФайловЗавершение(Результат, ДополнительныеПараметры) Экспорт

            Если Результат <> Неопределено И ТипЗнч(Результат) = Тип("Массив") Тогда
                ВладелецФормы.Модифицированность = Ложь;
                Объект.ИмяФайла = Результат[0].Имя;
                ВладелецФормы.Объект.ИмяФайла = Результат[0].Имя;
            КонецЕсли;

        КонецПроцедуры

        // Загрузка
```

```
&НаКлиенте
Процедура ОбработкаЧтенияФайла(Удалять)

	ДополнительныеПараметры = Новый Структура("Удалять", Удалять);

	#Если Не ВебКлиент Тогда
		// В тонком и толстом клиентах расширение подключено всегда.
		ЧтениеФайлаЗавершение(Истина, ДополнительныеПараметры);
		Возврат;
	#КонецЕсли

	Оповещение = Новый
ОписаниеОповещения("НачатьПодключениеРасширенияРаботыСФайламиЗавершение", ЭтотОбъект,
		Новый ОписаниеОповещения("ЧтениеФайлаЗавершение", ЭтотОбъект,
ДополнительныеПараметры));
	НачатьПодключениеРасширенияРаботыСФайлами(Оповещение);

КонецПроцедуры

&НаКлиенте
Процедура ЧтениеФайлаЗавершение(Результат, ДополнительныеПараметры) Экспорт

	Если Результат Тогда
		// Выбор файла для загрузки.
		Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
		Диалог.Заголовок = НСтр("ru = 'Выберите файл запросов'");
		Диалог.ПредварительныйПросмотр = Ложь;
		Диалог.Фильтр = НСтр("ru = 'Файл запросов (*.q1c)|*.q1c'");
		Диалог.Расширение = "q1c";
		Диалог.ПроверятьСуществованиеФайла = Истина;
		Диалог.МножественныйВыбор = Ложь;

		ДополнительныеПараметры = Новый Структура("Удалять",
ДополнительныеПараметры.Удалять);
		Оповещение = Новый ОписаниеОповещения("ПомещениеФайловЗавершение",
ЭтотОбъект, ДополнительныеПараметры);
		НачатьПомещениеФайлов(Оповещение,, Диалог, Истина,
ЭтотОбъект.УникальныйИдентификатор);
		Иначе
		ПоказатьСообщениеПользователю(НСтр("ru = 'Без расширения работы с файлами
невозможно работать с файлами.'"), "Объект");
		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ПомещениеФайловЗавершение(Результат, ДополнительныеПараметры) Экспорт

	Если ТипЗнч(Результат) = Тип("Массив") Тогда
		Если Результат.Количество() > 0 Тогда

			Если ДополнительныеПараметры.Удалять Тогда
				Объект.Запросы.Очистить();
				Объект.Параметры.Очистить();
			КонецЕсли;

			ИмяФайла = Результат[0].Имя;
			ЗагрузитьЗапросыИзФайла(Результат[0].Хранение);
```

```
                        Объект.ИмяФайла  = ИмяФайла;
                    КонецЕсли;
                КонецЕсли;

                КоличествоЗапросов = Объект.Запросы.Количество();
                Если КоличествоЗапросов > 0 Тогда
                    ИдентификаторТекущегоЗапроса  = Объект.Запросы.Получить(0).Идентификатор;
                    ПараметрыПередачи =
ПоместитьЗапросыВСтруктуру(ИдентификаторТекущегоЗапроса, ИдентификаторТекущегоПараметра);

                    Оповестить("ВыгрузитьЗапросыВРеквизиты", ПараметрыПередачи);
                    Оповестить("ОбновитьФормуКлиент");
                КонецЕсли;

        КонецПроцедуры

        // Общие процедуры для сохранения и загрузки

        &НаКлиенте
        Процедура НачатьПодключениеРасширенияРаботыСФайламиЗавершение(РасширениеПодключено,
ДополнительныеПараметры) Экспорт

                Если РасширениеПодключено Тогда
                    ВыполнитьОбработкуОповещения(ДополнительныеПараметры, Истина);
                Иначе
                    Если Не ЗаданВопросОбУстановкеРасширения Тогда
                        ЗаданВопросОбУстановкеРасширения = Истина;
                        ОписаниеОповещения = Новый
ОписаниеОповещения("ВопросОбУстановкеРасширения", ЭтотОбъект, ДополнительныеПараметры);
                        НачатьУстановкуРасширенияРаботыСФайлами(ОписаниеОповещения);
                    Иначе
                        ВыполнитьОбработкуОповещения(ДополнительныеПараметры,
РасширениеПодключено);
                    КонецЕсли;
                КонецЕсли;

        КонецПроцедуры

        &НаКлиенте
        Процедура ВопросОбУстановкеРасширения(Оповещение) Экспорт
            ВыполнитьОбработкуОповещения(Оповещение, Истина);
        КонецПроцедуры

        // Сохранение запросов.
        //
        // Параметры:
        //  ИмяФайла - имя файла XML.
        //  Объект - объект обработки.
        //
        &НаСервере
        Функция СохранитьЗапросы(Знач Объект)

            ДвоичныеДанные = ОбъектОбработки().ЗаписатьЗапросыВФайлXML(Объект);
            Возврат ДвоичныеДанные;

        КонецФункции

        &НаСервере
```

```
Функция СохранитьЗапросыВоВременноеХранилище(Знач  Объект)

	ДвоичныеДанные = СохранитьЗапросы(Объект);
	Возврат ПоместитьВоВременноеХранилище(ДвоичныеДанные);

КонецФункции

&НаСервере
Процедура ЗагрузитьЗапросыИзФайла(АдресВоВременномХранилище)

	ДвоичныеДанные = ПолучитьИзВременногоХранилища(АдресВоВременномХранилище);
	ОбъектВнешнейОбработки  =
ОбъектОбработки().ПрочитатьЗапросыИзФайлаXML(ДвоичныеДанные);
	ЗаполнитьЗапросыИПараметрыИзОбъектаВнешнейОбработки(ОбъектВнешнейОбработки);

КонецПроцедуры

// Заполняет из объекта внешней обработки запросы и параметры.
//
// Параметры:
//	ОбъектОбработки  - объект внешней обработки.
//
&НаСервере
Процедура ЗаполнитьЗапросыИПараметрыИзОбъектаВнешнейОбработки(ОбъектОбработки)
	ЗапросыОбработка = ОбъектОбработки.Запросы;
	ПараметрыОбработка = ОбъектОбработки.Параметры;

	Объект.Запросы.Очистить();
	Объект.Параметры.Очистить();

	// Заполнение запросов и параметров в форме.
	Для каждого  ТекстЗапрос Из  ЗапросыОбработка Цикл
		ЭлементЗапроса = Объект.Запросы.Добавить();
		ЭлементЗапроса.Идентификатор  = ТекстЗапрос.Идентификатор;
		ЭлементЗапроса.Имя = ТекстЗапрос.Имя;
		ЭлементЗапроса.Текст = ТекстЗапрос.Текст;
	КонецЦикла;

	Для каждого  ТекПараметр Из ПараметрыОбработка Цикл
		ТипСтрока  = ТекПараметр.Тип;

		Значение = ТекПараметр.Значение;
		Значение = ЗначениеИЗСтрокиВнутр(Значение);

		Если ТипСтрока = "ТаблицаЗначений" ИЛИ ТипСтрока = "МоментВремени" ИЛИ
ТипСтрока = "Граница" Тогда
			ЭлементПараметр = Объект.Параметры.Добавить();
			ЭлементПараметр.ИдентификаторЗапроса  =
ТекПараметр.ИдентификаторЗапроса;
			ЭлементПараметр.Идентификатор  = ТекПараметр.Идентификатор;
			ЭлементПараметр.Имя = ТекПараметр.Имя;
			ЭлементПараметр.Тип =
СписокТипов.НайтиПоЗначению(ТипСтрока).Значение;
			ЭлементПараметр.Значение = ТекПараметр.Значение;
			ЭлементПараметр.ТипВФорме =
СписокТипов.НайтиПоЗначению(ТипСтрока).Представление;
			ЭлементПараметр.ЗначениеВФорме =
ОбъектОбработки().ФормированиеПредставленияЗначения(Значение);
```

```
                        Иначе
                                Массив = Новый Массив;
                                Массив.Добавить(Тип(ТипСтрока));
                                Описание = Новый ОписаниеТипов(Массив);

                                ЭлементПараметр = Объект.Параметры.Добавить();
                                ЭлементПараметр.ИдентификаторЗапроса =
ТекПараметр.ИдентификаторЗапроса;
                                ЭлементПараметр.Идентификатор  = ТекПараметр.Идентификатор;
                                ЭлементПараметр.Имя = ТекПараметр.Имя;
                                ЭлементПараметр.Тип = ТипСтрока;
                                ЭлементПараметр.ТипВФорме = Описание;
                                ЭлементПараметр.Значение = ЗначениеВСтрокуВнутр(Значение);
                                ЭлементПараметр.ЗначениеВФорме = Значение;
                        КонецЕсли;
                КонецЦикла;
        КонецПроцедуры

        &НаКлиенте
        Процедура ПоказатьСообщениеПользователю(ТекстСообщения,  ПутьКДанным)
                ОчиститьСообщения();
                Сообщение = Новый СообщениеПользователю();
          Сообщение.Текст = ТекстСообщения;
                Сообщение.ПутьКДанным  = ПутьКДанным;
                Сообщение.УстановитьДанные(Объект);
          Сообщение.Сообщить();
        КонецПроцедуры

        &НаСервере
        Процедура ПолучитьТабличныеДокументыСравниваемыхЗапросов(ИДПервогоЗапроса,
ИДВторогоЗапроса, ФайлПервогоЗапроса, ФайлВторогоЗапроса)
                ФильтрПервого  = Новый Структура;
                ФильтрПервого.Вставить("Идентификатор",ИДПервогоЗапроса);
                АдресПервогоДокумента =
Объект.Запросы.НайтиСтроки(ФильтрПервого).Получить(0).АдресРезультата;

                ФильтрВторого  = Новый Структура;
                ФильтрПервого.Вставить("Идентификатор",ИДВторогоЗапроса);
                АдресВторогоДокумента =
Объект.Запросы.НайтиСтроки(ФильтрПервого).Получить(0).АдресРезультата;

                Если ПустаяСтрока(АдресПервогоДокумента)  ИЛИ ПустаяСтрока(АдресВторогоДокумента)
Тогда
                        Возврат;
                КонецЕсли;

                ТДПервогоЗапроса = ПолучитьИзВременногоХранилища(АдресПервогоДокумента);
                ТДВторогоЗапроса = ПолучитьИзВременногоХранилища(АдресВторогоДокумента);

                ФайлПервогоЗапроса = ПолучитьИмяВременногоФайла("mxl");
                ТДПервогоЗапроса.Записать(ФайлПервогоЗапроса);

                ФайлВторогоЗапроса = ПолучитьИмяВременногоФайла("mxl");
                ТДВторогоЗапроса.Записать(ФайлВторогоЗапроса);
        КонецПроцедуры

        // Формирует имя копии запроса.
        //
```

```
	// Параметры:
	//		Имя - передаваемое имя запроса.
	//
	&НаКлиенте
	Функция СформироватьИмяКопииЗапроса(Имя)
		Флаг	= Истина;
		Индекс	= 1;

		Пока Флаг Цикл
			ФормируемоеИмяЗапроса=НСтр("ru = '%ИмяЗапроса% - Копия %НомерКопии%'");
			ФормируемоеИмяЗапроса=СтрЗаменить(ФормируемоеИмяЗапроса,
"%ИмяЗапроса%", Имя);
			ФормируемоеИмяЗапроса=СтрЗаменить(ФормируемоеИмяЗапроса,
"%НомерКопии%", Индекс);

			Фильтр = Новый Структура;
			Фильтр.Вставить("Имя", ФормируемоеИмяЗапроса);

			МассивЗапросовПоФильтру = Объект.Запросы.НайтиСтроки(Фильтр);

			Если МассивЗапросовПоФильтру.Количество() = 0 Тогда
				Флаг = Ложь;
			КонецЕсли;

			Индекс = Индекс + 1;
		КонецЦикла;

		Возврат ФормируемоеИмяЗапроса;
	КонецФункции

	#КонецОбласти
	#Область ОписаниеПеременных

	&НаКлиенте
	Перем ИмяКолонкиПоУмолчанию;

	#КонецОбласти

	#Область ОбработчикиСобытий

	&НаСервере
	Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

		Если Параметры.Свойство("АвтоТест") Тогда
			Возврат;
		КонецЕсли;

		// Считывание параметров передачи.
		ПараметрыПередачи	= ПолучитьИзВременногоХранилища(Параметры.АдресХранилища);
		Объект.Запросы.Загрузить(ПараметрыПередачи.Запросы);
		Объект.Параметры.Загрузить(ПараметрыПередачи.Параметры);
		Объект.ИмяФайла		= ПараметрыПередачи.ИмяФайла;
		ИдентификаторТекущегоЗапроса		=
ПараметрыПередачи.ИдентификаторТекущегоЗапроса;
		ИдентификаторТекущегоПараметра		=
ПараметрыПередачи.ИдентификаторТекущегоПараметра;

		Объект.ДоступныеТипыДанных =
```

```
ОбъектОбработки().Метаданные().Реквизиты.ДоступныеТипыДанных.Тип;
		ОбъектОбработки().СформироватьСписокТипов(СписокТипов);

		ЗаполнитьТаблицыПриОткрытии();
	КонецПроцедуры

	&НаКлиенте
	Процедура НастройкиТаблицыЗначенийТипКолонкиПриИзменении(Элемент)
		// Определение наименования колонки.
		ПервыйТип = "";

		ТекущаяКолонка		= Элементы.НастройкиТаблицыЗначений.ТекущиеДанные;
		ТипКолонки	= ТекущаяКолонка.ТипКолонки;
		СтароеИмяКолонки=  ТекущаяКолонка.НаименованиеКолонки;

		ДоступныеТипы		= ТекущаяКолонка.ТипКолонки.Типы();
		Количество		= ДоступныеТипы.Количество();
		Если Количество > 0 Тогда
			Флаг = Ложь;
			Для каждого ЭлементСписка Из СписокТипов Цикл
				Если ЭлементСписка.Представление  = Строка(ДоступныеТипы.Получить(0))
Тогда
					Флаг = Истина;
					Прервать;
				КонецЕсли;
			КонецЦикла;
			Если Флаг Тогда
				ПервыйТип = Строка(ДоступныеТипы.Получить(0));  // для примитивных
типов.
			Иначе
				ПервыйТип = Новый(ДоступныеТипы.Получить(0));
				ПервыйТип = ИмяТипаПоЗначению(ПервыйТип);
			КонецЕсли;
		КонецЕсли;

		ИдентификаторСтроки	= ТекущаяКолонка.ПолучитьИдентификатор();
		Если СтрНайти(ВРег(СтароеИмяКолонки),  ВРег(ИмяКолонкиПоУмолчанию))  <> 0 Тогда
			НовоеИмяКолонки		= СформироватьИмяКолонки(ПервыйТип,
ИдентификаторСтроки);
		Иначе
			НовоеИмяКолонки		= СтароеИмяКолонки;
		КонецЕсли;
		ТекущаяКолонка.НаименованиеКолонки  = НовоеИмяКолонки;

		ТекущаяСтрока = Элементы.НастройкиТаблицыЗначений.ТекущаяСтрока;

		ИнициализацияКолонкиВТЗКлиент(СтароеИмяКолонки,  НовоеИмяКолонки,  ТипКолонки);
	КонецПроцедуры

	&НаКлиенте
	Процедура НастройкиТаблицыЗначенийПередНачаломДобавления(Элемент,  Отказ,  Копирование,
Родитель,  Группа)
		Отказ = Истина;

		ИдентификаторСтроки		= Новый УникальныйИдентификатор;
		ИмяКолонки		=
СформироватьИмяКолонки(ИмяКолонкиПоУмолчанию,  ИдентификаторСтроки);
```

```
            МассивТипов = Новый Массив;
            МассивТипов.Добавить(Тип("Строка"));
            ТипКолонки                                                  = Новый
ОписаниеТипов(МассивТипов);

        ЭлементНастройки                                        =
НастройкиТаблицыЗначений.Добавить();
            ЭлементНастройки.НаименованиеКолонки    = ИмяКолонки;
            ЭлементНастройки.ТипКолонки                 = ТипКолонки;

            ИнициализацияКолонкиВТЗКлиент("", ИмяКолонки, ТипКолонки)
    КонецПроцедуры

    &НаКлиенте
    Процедура НастройкиТаблицыЗначенийНаименованиеКолонкиОкончаниеВводаТекста(Элемент, Текст,
ДанныеВыбора, СтандартнаяОбработка)
            ТекущаяКолонкаТЗ        = Элементы.НастройкиТаблицыЗначений.ТекущиеДанные;
            СтароеИмя                   = ТекущаяКолонкаТЗ.НаименованиеКолонки;
            ТипКолонки          = ТекущаяКолонкаТЗ.ТипКолонки;
            ИдентификаторСтроки     = ТекущаяКолонкаТЗ.ПолучитьИдентификатор();

            Текст = УбратьСимволыИзТекста(Текст);

            Если Не ПустаяСтрока(Текст) Тогда
                НовоеИмя        = СформироватьИмяКолонки(Текст, ИдентификаторСтроки);
            Иначе
                НовоеИмя        = СформироватьИмяКолонки(ИмяКолонкиПоУмолчанию,
ИдентификаторСтроки);

                ПоказатьСообщениеПользователю(НСтр("ru ='Наименование колонки не может быть
пустым.'"), "Объект");
            КонецЕсли;

            ТекущаяКолонкаТЗ.НаименованиеКолонки = НовоеИмя;

            Если ТипКолонки.Типы().Количество() <> 0 Тогда
                ИзменитьИмяРеквизитаИКолонкиСервер(СтароеИмя, НовоеИмя, ТипКолонки);
            КонецЕсли;
    КонецПроцедуры

    &НаКлиенте
    Процедура НастройкиТаблицыЗначенийПередУдалением(Элемент, Отказ)
            ТекущаяСтрока           = Элементы.НастройкиТаблицыЗначений.ТекущаяСтрока;
            ТекущаяКолонкаТЗ        = Элементы.НастройкиТаблицыЗначений.ТекущиеДанные;
            ИмяКолонки                  = ТекущаяКолонкаТЗ.НаименованиеКолонки;
            ТипКолонки          = ТекущаяКолонкаТЗ.ТипКолонки;

            Если ТипКолонки.Типы().Количество() <> 0 Тогда
                УдалитьКолонкуСервер(ИмяКолонки);
            КонецЕсли;

            ЭлементКоллекции = НастройкиТаблицыЗначений.НайтиПоИдентификатору(ТекущаяСтрока);
            ИндексЭлементаКоллекции = НастройкиТаблицыЗначений.Индекс(ЭлементКоллекции);
            НастройкиТаблицыЗначений.Удалить(ИндексЭлементаКоллекции);
    КонецПроцедуры

    ////////////////////////////////////////////////////////////////////////////
    // КОМАНДЫ
```

```
	&НаКлиенте
	Процедура ВыгрузитьТаблицуЗначений(Команда)
		ВыгрузитьТаблицуЗначенийСервер();
	КонецПроцедуры

	////////////////////////////////////////////////////////////////////////////
	// ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ

	&НаСервере
	Функция ОбъектОбработки()
		Возврат РеквизитФормыВЗначение("Объект");
	КонецФункции

	// Формирует колонки для Таблицы значений из Настроек таблицы значений.
	//
	// Изменяет реквизиты текущего параметра.
	//
	&НаКлиенте
	Процедура ВыгрузитьТаблицуЗначенийСервер()
		ПараметрыПередачи = ПоместитьЗапросыВСтруктуру(ИдентификаторТекущегоЗапроса,
ИдентификаторТекущегоПараметра);

		Закрыть();
		Владелец                                    = ЭтотОбъект.ВладелецФормы;
		Владелец.Модифицированность = Истина;

		Оповестить("ВыгрузитьЗапросыВРеквизиты", ПараметрыПередачи);
	КонецПроцедуры

	&НаСервере
	Функция ВнутрЗначениеОбъектаТЗ()
		ТЗ = РеквизитФормыВЗначение("ТаблицаЗначенийПараметр");
		Возврат ЗначениеВСтрокуВнутр(ТЗ);
	КонецФункции

	&НаСервере
	Функция ПоместитьЗапросыВСтруктуру(ИдентификаторЗапроса,ИдентификаторПараметра)
		ПараметрыФормы = Объект.Параметры;

		ПредставлениеТЗ = СформироватьПредставлениеТаблицыЗначений(ПредставлениеТЗ);

		Для каждого Стр Из ПараметрыФормы Цикл
			Если Стр.Идентификатор = ИдентификаторТекущегоПараметра Тогда
				Стр.Тип                         = "ТаблицаЗначений";
				Стр.Значение            = ВнутрЗначениеОбъектаТЗ();
				Стр.ТипВФорме           = НСтр("ru = 'Таблица значений'");
				Стр.ЗначениеВФорме      = ПредставлениеТЗ;
			КонецЕсли;
		КонецЦикла;

		ПараметрыПередачи = Новый Структура;
		ПараметрыПередачи.Вставить("АдресХранилища",
ОбъектОбработки().ПоместитьЗапросыВоВременноеХранилище(Объект, ИдентификаторЗапроса,
ИдентификаторПараметра));

		Возврат ПараметрыПередачи;
	КонецФункции
```

```
// Заполняет таблицы значений в форме по загружаемой таблице значений.
//
&НаСервере
Процедура ЗаполнитьТаблицыПриОткрытии()
	ПараметрыФормы = Объект.Параметры;
	Для каждого ТекущийПараметр Из ПараметрыФормы Цикл
		Если ТекущийПараметр.Идентификатор = ИдентификаторТекущегоПараметра Тогда
			Значение = ТекущийПараметр.Значение;
			Если ПустаяСтрока(Значение) Тогда
				Возврат;
			Иначе
				Прервать;
			КонецЕсли;
		КонецЕсли;
	КонецЦикла;

	// Формирование таблицы "Настройки".
	ТЗ = ЗначениеИзСтрокиВнутр(Значение);
	Если ТипЗнч(ТЗ) <> Тип("ТаблицаЗначений") Тогда
		Возврат;
	КонецЕсли;

	Колонки = ТЗ.Колонки;
	Для Индекс = 0 По Колонки.Количество() - 1 Цикл
		ТекущаяКолонка = Колонки.Получить(Индекс);

		ИмяКолонки = ТекущаяКолонка.Имя;
		ТипКолонки = ТекущаяКолонка.ТипЗначения;

		Настройка                                                    =
НастройкиТаблицыЗначений.Добавить();
		Настройка.НаименованиеКолонки          = ИмяКолонки;
		Настройка.ТипКолонки                   = ТипКолонки;

		ИнициализацияКолонкиВТЗСервер("", ИмяКолонки, ТипКолонки, "");
	КонецЦикла;

	// Заполнение таблицы значений.
	Для каждого Строка Из ТЗ Цикл
		ЭлементТЗ                                  = ТаблицаЗначенийПараметр.Добавить();
		Для каждого Колонка Из ТЗ.Колонки Цикл
			ЭлементТЗ[Колонка.Имя]  = Строка[Колонка.Имя];
		КонецЦикла;
	КонецЦикла;
КонецПроцедуры

&НаСервере
Функция СформироватьПредставлениеТаблицыЗначений(Представление)
	ТЗ = РеквизитФормыВЗначение("ТаблицаЗначенийПараметр");
	Представление = ОбъектОбработки().ФормированиеПредставленияЗначения(ТЗ);

	Возврат Представление;
КонецФункции

// Формирует имя добавляемой колонки.
// Оно не должно совпадать с именем реквизита формы
// и с именем колонки.
```

```
	//
	// Параметры:
	//		Имя - передаваемое имя.
	//
	&НаКлиенте
	Функция СформироватьИмяКолонки(знач ИмяКолонки, ИДТекСтроки)
		НТЗ = НастройкиТаблицыЗначений;
		Флаг = Истина;
		Индекс = 0;

		ИмяКолонки = СокрЛП(ИмяКолонки);

		Пока Флаг Цикл
			Имя = ИмяКолонки + Строка(Формат(Индекс, "ЧН=-"));
			Имя = СтрЗаменить(Имя, "-", "");

			// Если нет строки с таким именем.
			Фильтр = Новый Структура("НаименованиеКолонки", Имя);
			ОтфильтрованныеСтроки = НТЗ.НайтиСтроки(Фильтр);
			Если ОтфильтрованныеСтроки.Количество() = 0 Тогда
				Флаг = Ложь;
			Иначе
				Если ОтфильтрованныеСтроки.Получить(0).ПолучитьИдентификатор() <>
ИДТекСтроки Тогда
					Флаг = Истина;
				Иначе
					Флаг = Ложь;
				КонецЕсли;
			КонецЕсли;

			// Если нет колонки с таким именем.
			Колонки = Элементы.ТаблицаЗначенийПараметр.ПодчиненныеЭлементы;
			КолКолонок = Колонки.Количество();
			Для Индекс = 0 По КолКолонок - 1 Цикл
				Если Колонки.Получить(Индекс).Имя = Имя Тогда
					Флаг = Истина;
					Прервать;
				КонецЕсли;
			КонецЦикла;

			Результат = ?(Флаг, "", Имя);

			Индекс = Индекс + 1;
		КонецЦикла;

		Возврат Результат;
	КонецФункции

	&НаКлиенте
	Процедура ИнициализацияКолонкиВТЗКлиент(СтароеИмяКолонки, НовоеИмяКолонки, ТипКолонки)
		СообщениеСистемы = "";
		ИнициализацияКолонкиВТЗСервер(СтароеИмяКолонки, НовоеИмяКолонки, ТипКолонки,
СообщениеСистемы);
		Если НЕ ПустаяСтрока(СообщениеСистемы) Тогда
			ПоказатьСообщениеПользователю(СообщениеСистемы, "Объект");
		КонецЕсли;
	КонецПроцедуры
```

```
&НаСервере
Процедура ИнициализацияКолонкиВТЗСервер(СтароеИмяКолонки, НовоеИмяКолонки, ТипКолонки,
Сообщение = "");

	НачатьТранзакцию();
	Попытка
		ИмяУдаляемогоРеквизита = ИмяРодителя + "." + СтароеИмяКолонки;

		// Заполнение массива удаляемыми реквизитами.
		МассивУдаляемыхРеквизитов = Новый Массив;
		РекРодителя = ПолучитьРеквизиты(ИмяРодителя);
		Для каждого ТекРек Из РекРодителя Цикл
			Если ТекРек.Имя = СтароеИмяКолонки Тогда
				МассивУдаляемыхРеквизитов.Добавить(ИмяУдаляемогоРеквизита);
			КонецЕсли;
		КонецЦикла;

		// Выгрузка значений в таблицу значений.
		Если Не ПустаяСтрока(СтароеИмяКолонки) Тогда
			ТЗЗначений = ТаблицаЗначенийПараметр.Выгрузить(, СтароеИмяКолонки);
		Иначе
			ТЗЗначений = Неопределено;
		КонецЕсли;

		// Добавление нового реквизита в объект.
		ДобавляемыеРеквизиты = Новый Массив;

		НовыйРеквизит = Новый РеквизитФормы(НовоеИмяКолонки, ТипКолонки,
ИмяРодителя, НовоеИмяКолонки, Ложь);
		ДобавляемыеРеквизиты.Добавить(НовыйРеквизит);
		ИзменитьРеквизиты(ДобавляемыеРеквизиты, МассивУдаляемыхРеквизитов);

		// Поиск колонки в "ТаблицаЗначенийПараметр" с условием
ПутьКДанным=ПутьКНовомуРеквизиту.
		ИмяДобавляемогоРеквизита = ИмяРодителя + "." + НовоеИмяКолонки;
		НомерКолонки =
ПоискКолонокВТЗСЗаданнымПутемКДанным(ИмяДобавляемогоРеквизита);
		Если ТЗЗначений <> Неопределено Тогда
			Если НомерКолонки <> Неопределено Тогда
				ИмяПервойКолонки = ТЗЗначений.Колонки.Получить(0).Имя;
				Индекс = 0;
				Для Каждого Стр Из ТЗЗначений Цикл

	ТаблицаЗначенийПараметр.Получить(Индекс)[НовоеИмяКолонки] = Стр[ИмяПервойКолонки];
					Индекс = Индекс + 1;
				КонецЦикла;
			КонецЕсли;
		Иначе
			НоваяКолонкаТаблицы = Элементы.Добавить(НовоеИмяКолонки,
Тип("ПолеФормы"), Элементы.ТаблицаЗначенийПараметр);
			НоваяКолонкаТаблицы.ПутьКДанным = ИмяДобавляемогоРеквизита;
			НоваяКолонкаТаблицы.Вид = ВидПоляФормы.ПолеВвода;
		КонецЕсли;
		ЗафиксироватьТранзакцию();
	Исключение
		ОтменитьТранзакцию();
		ВызватьИсключение;
	КонецПопытки;
```

```
КонецПроцедуры

// Изменяет имя реквизита и колонки по идентификатору строки.
//
// Параметры:
//	ИДСтроки - идентификатор строки таблицы значений настроек.
//	Имя - новое передаваемое имя для реквизита и колонки.
//
&НаСервере
Процедура ИзменитьИмяРеквизитаИКолонкиСервер(СтароеИмя, НовоеИмя, ТипКолонки)

	НачатьТранзакцию();
	Попытка
		ИмяУдаляемогоРеквизита = ИмяРодителя + "." + СтароеИмя;

		// Заполнение массива удаляемыми реквизитами.
		МассивУдаляемыхРеквизитов = Новый Массив;
		РекРодителя = ПолучитьРеквизиты(ИмяРодителя);
		Для каждого ТекРек Из РекРодителя Цикл
			Если ТекРек.Имя = СтароеИмя Тогда
				МассивУдаляемыхРеквизитов.Добавить(ИмяУдаляемогоРеквизита);
			КонецЕсли;
		КонецЦикла;

		// Выгрузка значений в таблицу значений.
		ТЗЗначений = ТаблицаЗначенийПараметр.Выгрузить(, СтароеИмя);

		// Добавление нового реквизита в объект.
		ДобавляемыеРеквизиты = Новый Массив;
		НовыйРеквизит = Новый РеквизитФормы(НовоеИмя, ТипКолонки, ИмяРодителя,
НовоеИмя, Ложь);
		ДобавляемыеРеквизиты.Добавить(НовыйРеквизит);
		ИзменитьРеквизиты(ДобавляемыеРеквизиты, МассивУдаляемыхРеквизитов);

		// Поиск колонки в "ТаблицаЗначенийПараметр" с условием ПутьКДанным =
ПутьКНовомуРеквизиту.
		ИмяДобавляемогоРеквизита = ИмяРодителя + "." + НовоеИмя;
		НомерКолонки =
ПоискКолонокВТЗСЗаданнымПутемКДанным(ИмяДобавляемогоРеквизита);
		Если НомерКолонки <> Неопределено Тогда
			ИмяПервойКолонки = ТЗЗначений.Колонки.Получить(0).Имя;
			Индекс = 0;
			Для Каждого СтараяСтрока Из ТЗЗначений Цикл
				ТаблицаЗначенийПараметр.Получить(Индекс)[НовоеИмя] =
СтараяСтрока[ИмяПервойКолонки];
				Индекс = Индекс + 1;
			КонецЦикла;
		КонецЕсли;
		ЗафиксироватьТранзакцию();
	Исключение
		ОтменитьТранзакцию();
	КонецПопытки;
КонецПроцедуры

// Возвращает номер колонки с заданным путем.
//
// Параметры:
```

```
        //          ПутьКДанным - заданный путь.
        //
        //          Возвращаемое значение: номер колонки или Неопределено.
        //
        &НаСервере
        Функция ПоискКолонокВТЗСЗаданнымПутемКДанным(ПутьКДанным)
                Колонки                                 =
Элементы.ТаблицаЗначенийПараметр.ПодчиненныеЭлементы;
                КоличествоКолонок       = Колонки.Количество();
                Флаг                                    = Ложь;
                Для Индекс = 0 По КоличествоКолонок - 1 Цикл
                        ТекКолонка = Колонки.Получить(Индекс);
                        Если ТекКолонка.ПутьКДанным = ПутьКДанным Тогда
                                Возврат Индекс;
                        КонецЕсли;
                КонецЦикла;
          Возврат Неопределено;
        КонецФункции

        // Удаляет колонку по имени.
        //
        // Параметры:
        //          ИмяКолонки - имя колонки.
        //
        &НаСервере
        Процедура УдалитьКолонкуСервер(ИмяКолонки)
                ИмяУдаляемогоРеквизита = ИмяРодителя + "." + ИмяКолонки;

                // Заполнение массива удаляемыми реквизитами.
                МассивУдаляемыхРеквизитов = Новый Массив;
                РекРодителя                     = ПолучитьРеквизиты(ИмяРодителя);
                Для каждого ТекРек Из РекРодителя Цикл
                        Если ТекРек.Имя = ИмяКолонки Тогда
                                МассивУдаляемыхРеквизитов.Добавить(ИмяУдаляемогоРеквизита);
                        КонецЕсли;
                КонецЦикла;

                ИзменитьРеквизиты(, МассивУдаляемыхРеквизитов);
        КонецПроцедуры

        &НаКлиенте
        Процедура ПоказатьСообщениеПользователю(ТекстСообщения, ПутьКДанным)
                ОчиститьСообщения();
                Сообщение = Новый СообщениеПользователю();
          Сообщение.Текст = ТекстСообщения;
                Сообщение.ПутьКДанным = ПутьКДанным;
                Сообщение.УстановитьДанные(Объект);
          Сообщение.Сообщить();
        КонецПроцедуры

        &НаКлиенте
        Функция УбратьСимволыИзТекста(знач Текст)
                Результат = "";

                ДлинаТекста = СтрДлина(Текст);

                Если ДлинаТекста = 0 Тогда
                        Возврат Результат;
```

```
		КонецЕсли;

		Для Индекс = 0 По ДлинаТекста - 1 Цикл
			СимволТекста = Лев(Текст, 1);
			Если Не ЭтоСимвол(СимволТекста) Тогда
				Результат = Результат + СимволТекста;
			КонецЕсли;
			Текст = Сред(Текст, 2);
		КонецЦикла;

		Возврат Результат;
	КонецФункции

	&НаКлиенте
	Функция ЭтоСимвол(Символ)
		// Символы между 1040 и 1103 - Русские буквы.
		// Символы между 48 и 57 - Цифры.
		// Символы между 65 и 122 - Английские буквы.

		Код = КодСимвола(Символ);
		Если (Код >= 1040 И Код <= 1103) Или (Код >= 48 И Код <= 57) Или (Код >= 65 И Код <= 122)
Тогда
			Возврат Ложь;
		Иначе
			Возврат Истина;
		КонецЕсли;
	КонецФункции

	&НаСервере
	Функция ИмяТипаПоЗначению(Значение)

		Возврат Значение.Метаданные().Имя;

	КонецФункции

	#КонецОбласти

	#Область Инициализация

	ИмяРодителя          = "ТаблицаЗначенийПараметр";
	ИмяКолонкиПоУмолчанию = "Колонка";

	#КонецОбласти

	#Область ОбработчикиСобытий

	&НаСервере
	Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

		Если Параметры.Свойство("АвтоТест") Тогда
			Возврат;
		КонецЕсли;

		Текст = Параметры.ТекстЗапроса;
		ТекстЗапроса.УстановитьТекст(СформироватьТекстЗапросаДляКонфигуратора(Текст));
	КонецПроцедуры

	//////////////////////////////////////////////////////////////////////////
```

```
	// ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ

	&НаСервере
	Функция СформироватьТекстЗапросаДляКонфигуратора(Текст)
		Результат = """";
		Текст = Параметры.ТекстЗапроса;
		ПереводСтроки = Символы.ВК+Символы.ПС;
		Для Счетчик = 1 По СтрЧислоСтрок(Текст) Цикл
			ТекСтрока = СтрПолучитьСтроку(Текст, Счетчик);
			Если Счетчик > 1 Тогда
				ТекСтрока = СтрЗаменить(ТекСтрока,"""","""""");
				Результат = Результат + ПереводСтроки + "|"+ ТекСтрока;
			Иначе
				ТекСтрока = СтрЗаменить(ТекСтрока,"""","""""");
				Результат = Результат + ТекСтрока;
			КонецЕсли;
		КонецЦикла;
		Результат = Результат + """";
		Возврат Результат;
	КонецФункции

	#КонецОбласти

	#Область ОбработчикиСобытий

	&НаСервере
	Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

		Если Параметры.Свойство("АвтоТест") Тогда
			Возврат;
		КонецЕсли;

		ОбработкаОбъект                    = ОбъектОбработки();
		Объект.ПутьКФормам                 = ОбработкаОбъект.Метаданные().ПолноеИмя() +
".Форма";

		// Считывание параметров передачи.
		ПараметрыПередачи      = ПолучитьИзВременногоХранилища(Параметры.АдресХранилища);
		Объект.Запросы.Загрузить(ПараметрыПередачи.Запросы);
		Объект.Параметры.Загрузить(ПараметрыПередачи.Параметры);
		Объект.ИмяФайла        = ПараметрыПередачи.ИмяФайла;
		ИдентификаторТекущегоЗапроса       =
ПараметрыПередачи.ИдентификаторТекущегоЗапроса;
		ИдентификаторТекущегоПараметра     =
ПараметрыПередачи.ИдентификаторТекущегоПараметра;

		Попытка  // Если форма открывается не из главной формы.
			МоментВремени          = ЗначениеИзСтрокиВнутр(Параметры.Значение);
			Дата                   = МоментВремени.Дата;
			Ссылка                 = МоментВремени.Ссылка;
		Исключение
			ЗаполнитьЗначения();
		КонецПопытки;
	КонецПроцедуры

	////////////////////////////////////////////////////////////////////////////
	// КОМАНДЫ
```

```
&НаКлиенте
Процедура ЗаписатьИЗакрыть(Команда)
	ВыгрузитьМоментВремениСервер();
КонецПроцедуры

////////////////////////////////////////////////////////////////////////////
// ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ

&НаСервере
Функция ОбъектОбработки()
	Возврат РеквизитФормыВЗначение("Объект");
КонецФункции

&НаКлиенте
Процедура ВыгрузитьМоментВремениСервер()
	Владелец = ЭтотОбъект.ВладелецФормы;
	ИмяФормыВладельца = Владелец.ИмяФормы;
	ИмяОсновнойФормы = Объект.ПутьКФормам + ".Форма";

	Если ИмяФормыВладельца = ИмяОсновнойФормы Тогда
		ПараметрыПередачи =
ПоместитьЗапросыВСтруктуру(ИдентификаторТекущегоЗапроса, ИдентификаторТекущегоПараметра);
		Закрыть();
		Владелец.Модифицированность = Истина;
		Оповестить("ВыгрузитьЗапросыВРеквизиты", ПараметрыПередачи);
	Иначе
		ПредставлениеМоментаВремени = "";
		ВнутрМоментВремени = ВнутрЗначениеОбъектаМВ(ПредставлениеМоментаВремени);
		Закрыть();
		ПараметрыПередачи = Новый Структура("ВнутрМоментВремени,
ПредставлениеМоментаВремени",
			ВнутрМоментВремени, ПредставлениеМоментаВремени);
		Оповестить("ПолучениеМоментаВремени", ПараметрыПередачи);
	КонецЕсли;
КонецПроцедуры

&НаСервере
Функция ПоместитьЗапросыВСтруктуру(ИдентификаторЗапроса, ИдентификаторПараметра)
	ПараметрыФормы = Объект.Параметры;

	ПредставлениеМоментаВремени = "";
	Для каждого Стр Из ПараметрыФормы Цикл
		Если Стр.Идентификатор = ИдентификаторТекущегоПараметра Тогда
			Стр.Тип = "МоментВремени";
			Стр.Значение =
ВнутрЗначениеОбъектаМВ(ПредставлениеМоментаВремени);
			Стр.ТипВФорме = НСтр("ru = 'Момент времени'");
			Стр.ЗначениеВФорме = ПредставлениеМоментаВремени;
		КонецЕсли;
	КонецЦикла;

	ПараметрыПередачи = Новый Структура;
	ПараметрыПередачи.Вставить("АдресХранилища",
ОбъектОбработки().ПоместитьЗапросыВоВременноеХранилище(Объект,ИдентификаторЗапроса,Идентификато
рПараметра));
	Возврат ПараметрыПередачи;
КонецФункции
```

```
&НаСервере
Функция ВнутрЗначениеОбъектаМВ(Представление)
	МоментВремени = Новый МоментВремени(Дата, Ссылка);
	Представление = ОбъектОбработки().ФормированиеПредставленияЗначения(МоментВремени);

	Возврат ЗначениеВСтрокуВнутр(МоментВремени);
КонецФункции

&НаСервере
Процедура ЗаполнитьЗначения()
	ПараметрыФормы = Объект.Параметры;
	Для каждого ТекущийПараметр Из ПараметрыФормы Цикл
		Если ТекущийПараметр.Идентификатор = ИдентификаторТекущегоПараметра Тогда
			Значение = ТекущийПараметр.Значение;
			Если ПустаяСтрока(Значение) Тогда
				Возврат;
			Иначе
				Прервать;
			КонецЕсли;
		КонецЕсли;
	КонецЦикла;

	МоментВремени = ЗначениеИзСтрокиВнутр(Значение);
	Если ТипЗнч(МоментВремени) <> Тип("МоментВремени") Тогда
		Возврат;
	КонецЕсли;

	Дата    = МоментВремени.Дата;
	Ссылка  = МоментВремени.Ссылка;
КонецПроцедуры

#КонецОбласти

#Область ОбработчикиСобытийФормы

&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

	Если Параметры.Свойство("АвтоТест") Тогда
		Возврат;
	КонецЕсли;

	ОбработкаОбъект                 = ОбъектОбработки();
	Объект.ДоступныеТипыДанных =
ОбработкаОбъект.Метаданные().Реквизиты.ДоступныеТипыДанных.Тип;
	Объект.ПутьКФормам              = ОбработкаОбъект.Метаданные().ПолноеИмя() +
".Форма";

	Элементы.ВидГраницы.СписокВыбора.Добавить("Включая");
	Элементы.ВидГраницы.СписокВыбора.Добавить("Исключая");
	ВидГраницыФормы = Элементы.ВидГраницы.СписокВыбора.Получить(0).Значение;

	// Получение списка типов и его фильтрация.
	СписокТипов = ОбъектОбработки().СформироватьСписокТипов();
	ОбъектОбработки().ФильтрацияСпискаТипов(СписокТипов, "Граница");

	// Считывание параметров передачи.
	ПараметрыПередачи    = ПолучитьИзВременногоХранилища(Параметры.АдресХранилища);
```

```
		Объект.Запросы.Загрузить(ПараметрыПередачи.Запросы);
		Объект.Параметры.Загрузить(ПараметрыПередачи.Параметры);
		Объект.ИмяФайла		= ПараметрыПередачи.ИмяФайла;
		ИдентификаторТекущегоЗапроса		=
ПараметрыПередачи.ИдентификаторТекущегоЗапроса;
		ИдентификаторТекущегоПараметра		=
ПараметрыПередачи.ИдентификаторТекущегоПараметра;

		ЗаполнитьЗначения();
	КонецПроцедуры

	&НаКлиенте
	Процедура ОбработкаОповещения(ИмяСобытия,  Параметр, Источник)
		Если ИмяСобытия = "ПолучениеМоментаВремени" Тогда
			ПолучениеМоментаВремени(Параметр);
		КонецЕсли;
	КонецПроцедуры

	////////////////////////////////////////////////////////////////////////////
	// ОБРАБОТЧИКИ  СОБЫТИЙ  ЭЛЕМЕНТОВ  ФОРМЫ

	&НаКлиенте
	Процедура ТипНачалоВыбора(Элемент,  ДанныеВыбора, СтандартнаяОбработка)

		СтандартнаяОбработка  = Ложь;
		ОписаниеОповещения = Новый ОписаниеОповещения("ТипЗавершениеВыбора",  ЭтотОбъект);
		СписокТипов.ПоказатьВыборЭлемента(ОписаниеОповещения,  НСтр("ru = 'Выбрать тип'"));

	КонецПроцедуры

	&НаКлиенте
	Процедура ТипЗавершениеВыбора(ВыбранныйЭлемент, ДополнительныеПараметры) Экспорт

		Если ВыбранныйЭлемент <> Неопределено  Тогда

			ТекущийТип = ВыбранныйЭлемент;

			Если ТекущийТип.Значение = "МоментВремени" Тогда
				Тип				= ТекущийТип.Представление;
				Значение		= Тип;
				ЗначениеВФорме		= Тип;
			Иначе
				Тип		= ТекущийТип.Представление;

				Массив		= Новый Массив;
				Массив.Добавить(Тип(ТекущийТип.Значение));
				Описание		= Новый ОписаниеТипов(Массив);

				ЗначениеВФорме		=
Описание.ПривестиЗначение(ТекущийТип.Значение);
				Значение		=
Описание.ПривестиЗначение(ТекущийТип.Значение);
			КонецЕсли;

		КонецЕсли;

	КонецПроцедуры
```

```
&НаКлиенте
Процедура ЗначениеВФормеНачалоВыбора(Элемент, ДанныеВыбора, СтандартнаяОбработка)
	ПередаваемыеЗапросы = ПередачаЗапросов();
	ПередаваемыеЗапросы.Вставить("Значение",Значение);

	Если Тип = НСтр("ru ='Момент времени'") Тогда
		Путь = Объект.ПутьКФормам + "." + "МоментВремени";
		ОткрытьФорму(Путь, ПередаваемыеЗапросы, ЭтотОбъект);
	Иначе
		Возврат;
	КонецЕсли;
КонецПроцедуры

&НаКлиенте
Процедура ЗначениеВФормеПриИзменении(Элемент)
	ИзменениеЗначенияВФорме();
КонецПроцедуры

////////////////////////////////////////////////////////////////////////////
// КОМАНДЫ

&НаКлиенте
Процедура ЗаписатьГраницу(Команда)
	ВыгрузитьГраницуСервер();
КонецПроцедуры

////////////////////////////////////////////////////////////////////////////
// ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ

&НаСервере
Функция ОбъектОбработки()
	Возврат РеквизитФормыВЗначение("Объект");
КонецФункции

// Передача табличной части "Запросы","Параметры" в виде структуры.
//
&НаСервере
Функция ПередачаЗапросов()
	АдресХранилища = ОбъектОбработки().ПоместитьЗапросыВоВременноеХранилище(Объект,
ИдентификаторТекущегоЗапроса,ИдентификаторТекущегоПараметра);
	ПараметрАдрес = Новый Структура;
	ПараметрАдрес.Вставить("АдресХранилища", АдресХранилища);
	Возврат ПараметрАдрес;
КонецФункции

&НаСервере
Процедура ПолучениеМоментаВремени(СтруктураПередачи)
	Значение = СтруктураПередачи.ВнутрМоментВремени;
	ЗначениеВФорме = СтруктураПередачи.ПредставлениеМоментаВремени;
КонецПроцедуры

&НаКлиенте
Процедура ВыгрузитьГраницуСервер()
	ПараметрыПередачи = ПоместитьЗапросыВСтруктуру(ИдентификаторТекущегоЗапроса,
ИдентификаторТекущегоПараметра);
	Закрыть();
	Владелец = ЭтотОбъект.ВладелецФормы;
```

```
		Владелец.Модифицированность  = Истина;
		Владелец.ВыгрузитьЗапросыВРеквизиты(ПараметрыПередачи);
	КонецПроцедуры

	&НаСервере
	Функция ВнутрЗначениеОбъектаГраницы()
		ВидГран          = ОбъектОбработки().ОпределениеВидаГраницы(ВидГраницыФормы);
		ГраницаФормы  = Новый Граница(ЗначениеИзСтрокиВнутр(Значение),ВидГран);

		Возврат ЗначениеВСтрокуВнутр(ГраницаФормы);
	КонецФункции

	&НаСервере
	Функция ПоместитьЗапросыВСтруктуру(ИдентификаторЗапроса,  ИдентификаторПараметра)
		ПараметрыФормы         = Объект.Параметры;

		ПредставлениеГраницы  = СформироватьГраницу();

		Для каждого Стр Из ПараметрыФормы Цикл
			Если Стр.Идентификатор  = ИдентификаторТекущегоПараметра  Тогда
				Стр.Тип                                = "Граница";
				Стр.Значение              = ВнутрЗначениеОбъектаГраницы();
				Стр.ТипВФорме             = НСтр("ru ='Граница'");
				Стр.ЗначениеВФорме     = ПредставлениеГраницы;
			КонецЕсли;
		КонецЦикла;

		ПараметрыПередачи = Новый Структура;
		ПараметрыПередачи.Вставить("АдресХранилища",
ОбъектОбработки().ПоместитьЗапросыВоВременноеХранилище(Объект,ИдентификаторЗапроса,Идентификато
рПараметра));
		Возврат ПараметрыПередачи;
	КонецФункции

	&НаСервере
	Процедура ЗаполнитьЗначения()
		ПараметрыФормы = Объект.Параметры;
		Для каждого ТекущийПараметр Из ПараметрыФормы Цикл
			Если ТекущийПараметр.Идентификатор  = ИдентификаторТекущегоПараметра  Тогда
				Значение = ТекущийПараметр.Значение;
				Если ПустаяСтрока(Значение)  Тогда
					Возврат;
				Иначе
					Прервать;
				КонецЕсли;
			КонецЕсли;
		КонецЦикла;

		Граница = ЗначениеИзСтрокиВнутр(Значение);
		Если ТипЗнч(Граница) <> Тип("Граница") Тогда
			Возврат;
		КонецЕсли;

		ЗначениеЗагруженное = Граница.Значение;
		ТипЗ = ОбъектОбработки().ИмяТипаИзЗначения(ЗначениеЗагруженное);
		Тип = СписокТипов.НайтиПоЗначению(ТипЗ).Представление;
		Если Тип <> НСтр("ru  = 'Момент времени'") Тогда
			ЗначениеВФорме = ЗначениеЗагруженное;
```

```
                Иначе
                        ЗначениеВФорме =
ОбъектОбработки().ФормированиеПредставленияЗначения(ЗначениеЗагруженное);
                КонецЕсли;
                Значение = ЗначениеВСтрокуВнутр(ЗначениеЗагруженное);

                Если Граница.ВидГраницы = ВидГраницы.Включая Тогда
                        ВидГраницыФормы = элементы.ВидГраницы.СписокВыбора.Получить(0).Значение;
                Иначе
                        ВидГраницыФормы = элементы.ВидГраницы.СписокВыбора.Получить(1).Значение;
                КонецЕсли;
        КонецПроцедуры

        &НаСервере
        Функция СформироватьГраницу()
                ВидГран          = ОбъектОбработки().ОпределениеВидаГраницы(ВидГраницыФормы);
                ГраницаФормы = Новый Граница(ЗначениеИзСтрокиВнутр(Значение),ВидГран);

                Представление = ОбъектОбработки().ФормированиеПредставленияЗначения(ГраницаФормы);

                Возврат Представление;
        КонецФункции

        &НаСервере
        Процедура ИзменениеЗначенияВФорме()
                Значение = ЗначениеВСтрокуВнутр(ЗначениеВФорме);
        КонецПроцедуры

        #КонецОбласти

        #Область ОбработчикиСобытий

        &НаСервере
        Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

                Если Параметры.Свойство("АвтоТест") Тогда
                        Возврат;
                КонецЕсли;

                Результат = Параметры.РезультатЗапроса;
        КонецПроцедуры

        &НаКлиенте
        Процедура ЗакрытьФорму(Команда)
                Закрыть();
        КонецПроцедуры

        #КонецОбласти
        #Область ОбработчикиСобытийФормы

        &НаСервере
        Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

                Если Параметры.Свойство("АвтоТест") Тогда
                        Возврат;
                КонецЕсли;
```

```
		Если НЕ ЗначениеЗаполнено(Параметры.МеткаЗапроса) Тогда
			Отказ = Истина;
			Возврат;
		КонецЕсли;

		МеткаЗапроса = Параметры.МеткаЗапроса;

		ЭтотОбъект.Заголовок = НСтр("ru = 'План выполнения запроса ('") + Параметры.ИмяЗапроса + ")";

		ПолноеИмяФайлаЖурнала = ФайлТехнологическийЖурнал(Параметры.ИдентификаторПроцессаОС, Параметры.КаталогСЛогФайлами);
		Если НЕ ДанныеИзТехнологическогоЖурналаПрочитаны(ПолноеИмяФайлаЖурнала) Тогда
			Элементы.ГруппаПланВыполненияЗапроса.ТекущаяСтраница = Элементы.ГруппаПолучениеПланаВыполненияЗапроса;
			ТребуетсяПрочитатьЖурналИмя = ПолноеИмяФайлаЖурнала;
		КонецЕсли;

	КонецПроцедуры

	#КонецОбласти

	#Область ОбработчикиСобытийЭлементовФормы

	&НаКлиенте
	Процедура ТипПоказаЗапросаПриИзменении(Элемент)

		Если ВидОтображенияДанных = 0 Тогда
			Если ТипСУБД = "DBMSSQL" Тогда
				Элементы.ДеревоОператорМетаданные.Видимость=Истина;
				Элементы.ДеревоОператор.Видимость=Ложь;
			Иначе
				ПланВыполненияЗапросаТекст = ПланВыполненияЗапросаВМетаданных;
			КонецЕсли;
			СформированныйSQLТекстЗапроса = ТекстЗапросаВВидеМетаданных;
		Иначе
			Если ТипСУБД = "DBMSSQL" Тогда
				Элементы.ДеревоОператорМетаданные.Видимость=Ложь;
				Элементы.ДеревоОператор.Видимость=Истина;
			Иначе
				ПланВыполненияЗапросаТекст = ПланВыполненияЗапросаИзТехЖурнала;
			КонецЕсли;

			СформированныйSQLТекстЗапроса = ТекстЗапросаBSQL;
		КонецЕсли;
	КонецПроцедуры

	#КонецОбласти

	#Область ОбработчикиСобытийЭлементовТаблицыФормыПланВыполненияЗапросаSQLServer

	&НаКлиенте
	Процедура ДеревоПриАктивизацииСтроки(Элемент)
		Если ТипСУБД = "DBMSSQL" Тогда
			Если ВидОтображенияДанных = 0 Тогда
				ОписаниеОператора = Элемент.ТекущиеДанные.ОператорМетаданные;
			Иначе
				ОписаниеОператора = Элемент.ТекущиеДанные.Оператор;
```

```
                КонецЕсли;
            КонецЕсли;

    КонецПроцедуры

    #КонецОбласти

    #Область СлужебныеФункции

    &НаСервере
    Функция ОбъектОбработки()
        Возврат РеквизитФормыВЗначение("Объект");
    КонецФункции

    &НаСервере
    Функция ФайлТехнологическийЖурнал(ИдентификаторПроцессаОС, КаталогСЛогФайлами)

        // Использовать ТекущаяДатаСеанса() нельзя, т.к. имена лог-файлов технологического журнала формируются по дате сервера.
        ТекущаяДата = ТекущаяДата();

        ОжидаемоеИмяФайла = ИмяФайлТехнологическийЖурнал(ТекущаяДата);

        ПолноеИмяФайлаЖурнала = НайтиФайлТехнологическийЖурнал(ОжидаемоеИмяФайла, ИдентификаторПроцессаОС, КаталогСЛогФайлами);
        Если ЗначениеЗаполнено(ПолноеИмяФайлаЖурнала) Тогда
            Возврат ПолноеИмяФайлаЖурнала;
        Иначе
            ОжидаемоеИмяФайла = ИмяФайлТехнологическийЖурнал(ТекущаяДата - 3600);
            ПолноеИмяФайлаЖурнала = НайтиФайлТехнологическийЖурнал(ОжидаемоеИмяФайла, ИдентификаторПроцессаОС, КаталогСЛогФайлами);
            Если ЗначениеЗаполнено(ПолноеИмяФайлаЖурнала) Тогда
                Возврат ПолноеИмяФайлаЖурнала;
            КонецЕсли;
        КонецЕсли;

        Возврат Неопределено;

    КонецФункции

    &НаСервере
    Функция ИмяФайлТехнологическийЖурнал(ДатаФайла)
        ОжидаемоеИмяФайла = Формат(ДатаФайла, "ДФ=yyMMddHH")+ ".log";
        Возврат ОжидаемоеИмяФайла;
    КонецФункции

    &НаСервере
    Функция НайтиФайлТехнологическийЖурнал(ИмяФайла, ИдентификаторПроцессаОС, КаталогСЛогФайлами)

        СписокФайлов = НайтиФайлы(КаталогСЛогФайлами, "*.log", Истина);
        Для каждого Файл Из СписокФайлов Цикл
            Если СтрНайти(Файл.Путь, "_" + ИдентификаторПроцессаОС) > 0 Тогда
                Если Файл.Имя = ИмяФайла Тогда
                    Возврат Файл.ПолноеИмя;
                КонецЕсли;
            КонецЕсли;
```

```
		КонецЦикла;

		Возврат Неопределено;

	КонецФункции

	&НаСервере
	Функция ДанныеИзТехнологическогоЖурналаПрочитаны(ПолноеИмяФайлаЖурнала)

		ПрочитанныеДанные = Новый Структура("ТипСУБД, СКЛЗапрос, ПланВыполненияЗапроса");
		ОбъектОбработки().ПрочитатьТехнологическийЖурнал(ПолноеИмяФайлаЖурнала,
МеткаЗапроса, ПрочитанныеДанные);

		ТипСУБД = ПрочитанныеДанные.ТипСУБД;
		ТекстЗапросаBSQL = ПрочитанныеДанные.СКЛЗапрос;
		ПланВыполненияЗапросаИзТехЖурнала = ПрочитанныеДанные.ПланВыполненияЗапроса;

		Если НЕ ЗначениеЗаполнено(ТипСУБД) Тогда
			Возврат Ложь;
		КонецЕсли;

		ВВидеМетаданных = ОбъектОбработки().ПреобразоватьВМетаданные(ТекстЗапросаBSQL,
ПланВыполненияЗапросаИзТехЖурнала, ТипСУБД);

		ТекстЗапросаВВидеМетаданных = ВВидеМетаданных.ТекстЗапросаВВидеМетаданных;
		ПланВыполненияЗапросаВМетаданных =
ВВидеМетаданных.ПланВыполненияЗапросаВМетаданных;

		СформированныйSQLТекстЗапроса = ВВидеМетаданных.ТекстЗапросаВВидеМетаданных;
		ПланВыполненияЗапросаТекст = ВВидеМетаданных.ПланВыполненияЗапросаВМетаданных;

		Если ТипСУБД = "DBMSSQL" Тогда
			СуммарнаяСтоимостьОбщая = 0;
			Элементы.ГруппаПланВыполненияЗапроса.ТекущаяСтраница =
Элементы.ГруппаПланВыполненияЗапросаSQLСервер;
			ДеревоПланаЗапроса =
РеквизитФормыВЗначение("ДеревоПланаВыполненияЗапроса");

		ОбъектОбработки().ДеревоПланаВыполненияЗапроса(ПланВыполненияЗапросаИзТехЖурнала,
ПланВыполненияЗапросаВМетаданных, ДеревоПланаЗапроса, СуммарнаяСтоимостьОбщая);
			ЗначениеВРеквизитФормы(ДеревоПланаЗапроса, "ДеревоПланаВыполненияЗапроса");
			СуммарнаяСтоимостьЗапроса = СуммарнаяСтоимостьОбщая;
			Элементы.ГруппаИнформацияОСтоимостиЗапроса.Видимость = Истина;
			Элементы.ПоказыватьПланВыполненияЗапросаВВиде.Видимость = Истина;
			Максимум =
НайтиМаксимальныйПоказательСтоимости(ДеревоПланаЗапроса.Строки);
			УстановитьОформлениеДанныхВКолонкеСтоимость(Максимум);
		Иначе
			Элементы.ГруппаИнформацияОСтоимостиЗапроса.Видимость = Ложь;
			Элементы.ГруппаПланВыполненияЗапроса.ТекущаяСтраница =
Элементы.ГруппаПланВыполненияЗапросаТекстовоеПредставление;
			Элементы.ПоказыватьПланВыполненияЗапросаВВиде.Видимость = Ложь;
		КонецЕсли;

		Возврат Истина;

	КонецФункции

	&НаСервере
```

```
	Процедура УстановитьОформлениеДанныхВКолонкеСтоимость(Максимум)

		УсловноеОформление.Элементы.Очистить();
		ЭлементУсловногоОформления = УсловноеОформление.Элементы.Добавить();

		ПолеОформления = ЭлементУсловногоОформления.Поля.Элементы.Добавить();
		ПолеОформления.Поле = Новый ПолеКомпоновкиДанных("ДеревоСтоимость");
		ПолеОформления.Использование = Истина;

		ЭлементОтбора =
ЭлементУсловногоОформления.Отбор.Элементы.Добавить(Тип("ЭлементОтбораКомпоновкиДанных"));
		ЭлементОтбора.ЛевоеЗначение = Новый
ПолеКомпоновкиДанных("ДеревоПланаВыполненияЗапроса.Стоимость");
		ЭлементОтбора.ВидСравнения = ВидСравненияКомпоновкиДанных.Равно;
		ЭлементОтбора.ПравоеЗначение = Максимум;
		ЭлементОтбора.Использование = Истина;
		ЭлементУсловногоОформления.Оформление.УстановитьЗначениеПараметра("Шрифт", Новый
Шрифт(, , Истина));

	КонецПроцедуры


	&НаСервере
	Функция НайтиМаксимальныйПоказательСтоимости(СтрокиДерева, Максимум = 0)


		Для каждого Строка Из СтрокиДерева Цикл
			Если Строка.Строки.Количество() > 0 Тогда
				Максимум = НайтиМаксимальныйПоказательСтоимости(Строка.Строки,
Максимум);
			КонецЕсли;
			Если Строка.Стоимость > Максимум Тогда
				Максимум = Строка.Стоимость;
			КонецЕсли;
		КонецЦикла;

		Возврат Максимум;

	КонецФункции

	&НаКлиенте
	Процедура ПриОткрытии(Отказ)
		Если ЗначениеЗаполнено(ТребуетсяПрочитатьЖурналИмя) Тогда

	ПодключитьОбработчикОжидания("ПрочитатьДанныеИзТехнологическогоЖурналаОбработчик", 2);
			Элементы.ГруппаПолучениеПланаВыполненияЗапроса.Видимость = Истина;
			КоличествоПопыток = 0;
		КонецЕсли;
	КонецПроцедуры

	&НаКлиенте
	Процедура ПрочитатьДанныеИзТехнологическогоЖурналаОбработчик()

		Если ДанныеИзТехнологическогоЖурналаПрочитаны(ТребуетсяПрочитатьЖурналИмя) Тогда

	ОтключитьОбработчикОжидания("ПрочитатьДанныеИзТехнологическогоЖурналаОбработчик");
			ТребуетсяПрочитатьЖурналИмя = Неопределено;
			Элементы.ГруппаПолучениеПланаВыполненияЗапроса.Видимость = Ложь;
```

```
                Иначе
                        Если КоличествоПопыток < 5 Тогда
                                КоличествоПопыток = КоличествоПопыток + 1;
                        Иначе

        ОтключитьОбработчикОжидания("ПрочитатьДанныеИзТехнологическогоЖурналаОбработчик");
                                ТребуетсяПрочитатьЖурналИмя = Неопределено;
                                Элементы.ГруппаПолучениеПланаВыполненияЗапроса.Видимость = Ложь;
                                ПоказатьПредупреждение(, НСтр("ru = 'Ошибка получения плана выполнения
запроса'"));
                        КонецЕсли;
                КонецЕсли;

        КонецПроцедуры

        &НаКлиенте
        Процедура ПоказыватьПланВыполненияЗапросаВВидеПриИзменении(Элемент)

                Если ПоказыватьПланВыполненияЗапросаВВиде = 0 Тогда
                        Элементы.ГруппаПланВыполненияЗапроса.ТекущаяСтраница =
Элементы.ГруппаПланВыполненияЗапросаSQLСервер;
                Иначе
                        Элементы.ГруппаПланВыполненияЗапроса.ТекущаяСтраница =
Элементы.ГруппаПланВыполненияЗапросаТекстовоеПредставление;
                КонецЕсли;

        КонецПроцедуры



        #КонецОбласти
        Выбрать
        Наименование,
        Зарплата
        ИЗ Справочник.Кадры
        Где ТрудоваяДеятельность.МинимальнаяЗарплата < 80000

        Выбрать
        Наименование,
        Зарплата,
        Оброзование
        Из Справочник.Кадры
        Где ТрудоваяДеятельность.Оброзование ПОДОБНО "ДГУ Факультет Информатики и Информационных
Технологий"
```