МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра Прикладной информатики

БАКАЛАВРСКАЯ РАБОТА

На тему

«Разработка автоматизированного рабочего места»

Исполнитель

Меликовский Максим Владимирович

(фамилия, имя, отчество)

Руководитель

доктор технических наук, профессор

(ученая степень, ученое звание)

Истомин Евгений Петрович

(фамилия, имя, отчество)

«К защите допускаю»

Заведующий кафедрой

(подпись)

кандидат технических наук

Слесарева Людмила Сергеевна

«23» 06 2016 г.

Санкт–Петербург

2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра прикладной информатики

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
Дипломная работа

На тему ___«Разработка автоматизированного рабочего места _____(на примере администратора кафе при гостинице) »_____

Исполнитель _____Меликовский Максим Владимирович_____
(фамилия, имя, отчество)

Руководитель _____доктор технических наук, профессор_____
(ученая степень, ученое звание)

_____Истомин Евгений Петрович_____
(фамилия, имя, отчество)

«К защите допускаю»
Заведующая кафедрой _____
(подпись)

_____заведующая кафедрой, кандидат технических наук_____
(ученая степень, ученое звание)

_____Слесарева Людмила Сергеевна_____
(фамилия, имя, отчество)

«___»_____20__г.

Санкт–Петербург
2016

Содержание

ВВЕДЕНИЕ

В настоящее время автоматизация производства проникает всё глубже и глубже во все сферы промышленности и сферы услуг, не является исключением и сфера общественного питания.

Последнее время, все больший интерес к автоматизации явно ощущается со стороны активно развивающихся предприятий общественного питания, владеющих достаточным количеством средств, желающих минимизировать потери рабочего времени персонала, сэкономить на его численности и поднять уровень сервиса в заведениях.

Автоматизация предприятий питания – это большой шаг на пути к более качественной, быстрой работе, приносящей прибыль. Конечно, программы автоматизации актуальны практически для любых организаций, при этом, для многих принципы их работы схожи. Однако автоматизация предприятий общественного питания имеет свои особенности, которые значительно отличают её от остальных. Поэтому для начала стоит говорить о тех требованиях, которые характерны именно для данной отрасли.

Автоматизация предприятий питания должна позволять легко производить расчеты ингредиентов, требуемых для приготовления продукции, помогать в проработке меню, отслеживать движение продуктов и блюд внутри предприятия, должна работать с продажей блюд оптом, а также необходима программа автоматизации розничной торговли. Кроме того, автоматизация предприятий общественного питания, как и другие подобные системы, должна облегчать работу с документами и ведение отчетности.

Быстрое управление заказами, оперативный расчет с клиентами – все это позволяет поднять обслуживание на новый уровень. Автоматизация предприятий питания ускорит работу, а значит, количество обслуживаемых клиентов увеличится, поднимая прибыль. Также стоит отметить, что повысится контроль выполнения поручений, учет продуктов станет более четким.

Как правило, организуются три вида автоматизированных рабочих мест (АРМ): место кассира (официанта), место администратора и место для обслуживания информационной базы данных системы.

Целью данной дипломной работы ставится разработка автоматизированного рабочего места администратора кафе на примере кафе при гостинице, где ведется учет клиентов.

Поставленная цель дипломной работы определила следующие задачи:

- провести исследование деятельности рассматриваемого предприятия, описать его основные бизнес-процессы, обосновать необходимость разработки автоматизированной системы;

- дать краткий анализ существующих систем и выбрать стратегию автоматизации предприятия;

- спроектировать информационную базу программного решения;

- реализовать программное решение и привести контрольный пример;

- разработать инструкцию пользователя.

1 Обследование предприятия общественного питания

1.1 Общая характеристика предприятия

Объектом автоматизации в данной дипломной работе является кафе при гостиничном комплексе «Аква».

Юридический и фактический адрес предприятия: Краснодарский край, Туапсинский район, п. Лермонтово, ул. Тепличная 3.

Тип предприятия – кафе. Это предприятие общественного питания с широким ассортиментом блюд. Назначение - обслуживание потребителей.

Организационная форма предприятия – индивидуальный предприниматель. Производственная структура кафе «Аква» представлена на рисунке 1. Ассортимент блюд, предлагаемых в меню, разнообразен, в основном представлены блюда русской и восточной кухни.

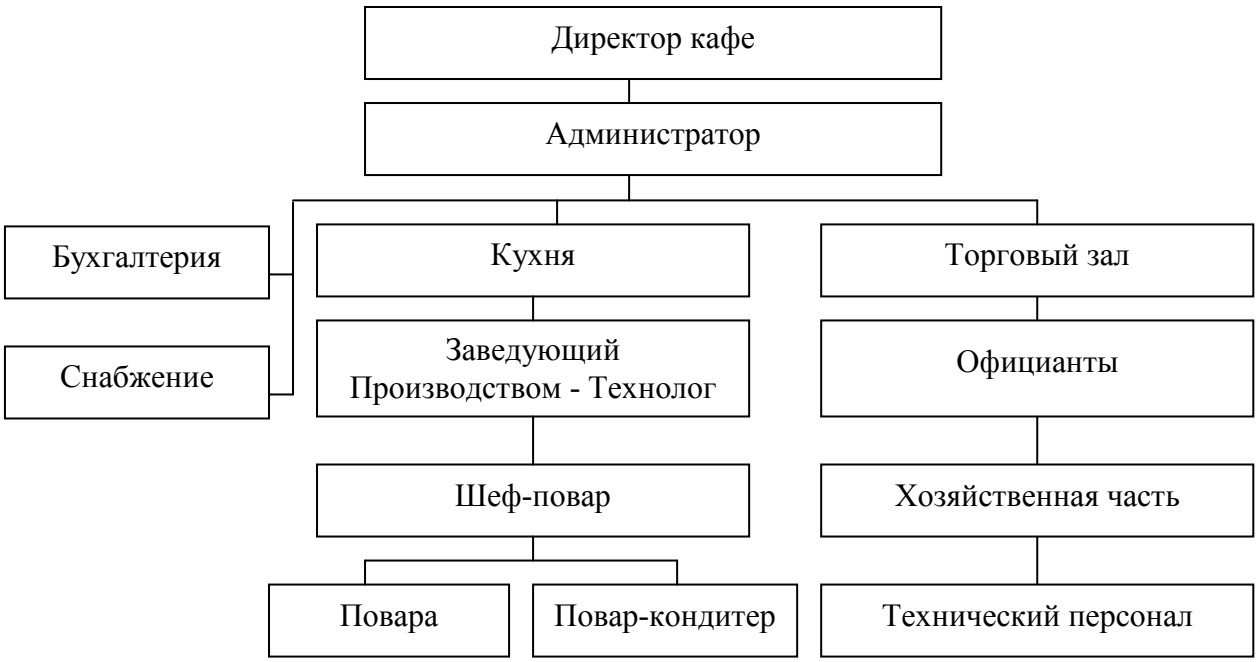


Рисунок 1 Производственная структура кафе «Аква»

Основные технико-экономические показатели работы кафе «Аква» за 2013 – 2015 гг., их динамика представлены в таблице 1.

Таблица 1 - Основные технико-экономические показатели 2013 – 2015

Показатели	2013	2014	2015
Численность работников, чел.	20	24	26
Основные фонды, тыс.руб.	4100	4450	5300
Оборотные средства, тыс.руб.	8200	8950	10600
Производственная площадь, кв.м	200	200	200
Проектная мощность, шт., тыс.руб.	70 800 заказов 28 224	95 600 заказов 31 248	104 000 заказов 35 280
Рентабельность, %	21	24	29
Количество производимых продуктов, шт.	115000	135 400	153 880
Количество заключенных договоров на производство/реализацию продуктов, шт.	3	7	11

С точки зрения масштабов производства кафе «Аква» следует стратегии роста, которая присуща молодым предприятиям любой сферы деятельности, только начинающим свое восхождение. Кафе свойственны постоянные темпы увеличения масштабов производства, объема выпускаемой продукции или услуг, измеряющиеся по всем направлениям деятельности десятками процентов в год.

Кафе «Аква» ориентировано на удовлетворение потребностей гостей гостиницы и превосходное обслуживание. По основным технико-экономическим показателям кафе следует стратегии роста, присущей всем молодым организациям.

Организационная структура кафе «Аква», построена на принципе иерархичности уровней управления, при котором каждый нижестоящий

уровень контролируется вышестоящим уровнем и подчиняется ему, а также на принципе разделения труда на отдельные функции и специализации работников по выполняемым функциям, соответствуют иерархическому типу [15,с.56].

Структура предприятия четко определена.

Всех сотрудников можно дифференцировать на три основные категории: руководители, специалисты, исполнители.

Руководители: директор кафе «Аква», администратор кафе.

Специалисты: 1 шеф-повар, 1 технолог, 2 бухгалтера, 6 официантов, 3 снабженца.

Технические исполнители: 4 повара, 2 кассира, 2 уборщицы, 1 гардеробщик.

Таким образом, система не перегружена специалистами и техническими исполнителями.

Организационная структура управления кафе «Аква» не позволяет быстро реагировать на изменения в области науки и техники, которые чаще всего приводят к «разбалансировке» отношений между функциональными подсистемами.

Результатом является замедление и сложности с передачей информации, а это приводит к снижению скорости принятия решений. Необходимость согласования действий разных функциональных руководителей резко увеличивает объем работы руководителя кафе и руководителями функциональных подразделений – производством и торговым залом.

Нижний уровень управления является источником информации для принятия управленческих решений на более высоком уровне. Если рассматривать поток информации от уровня к уровню, то количество информации, выраженное в числе символов, уменьшается с повышением уровня, но при этом увеличивается ее смысловое (семантическое) содержание [14,с.118].

Огромные объёмы перерабатываемой информации и серьёзные требования к быстроте обслуживания клиентов обуславливают необходимость широкого использования арсенала программных атрибутов.

Таким образом, объем информации высокий, необходимо использовать технические средства коммуникации.

Характеристика блоков управления кафе «Аква» представлена в таблице 2.

Таблица 2 - Характеристика блоков управления кафе «Аква»

Наименование подразделения	Цель	Решаемые задачи	Общие и специальные функции	Количество сотрудников
Дирекция	Организация, планирование и координация деятельности кафе	Внедрение международных стандартов обслуживания и обучения персонала;	Решение административно-хозяйственных вопросов, бюджетирование и планирование; оперативное управление кафе; планирование и выполнение бюджета	2
Бухгалтерия	Организация бухгалтерского учета и налогообложения кафе	Оптимизация затрат. Управление кредиторской и дебиторской задолженностью	Планирование затрат, анализ издержек производства и обращения хозяйства кафе. Анализ основных фондов, оборотных средств кафе. Эффективное использование фонда заработной платы	4

Продолжение таблицы 2

Кухня	Удовлетворение любых аппетитных желаний посетителей	Планирование производственной программы, товарооборота.	Калькуляция продажных цен на продукцию. Производство и отпуск продукции Производство и отпуск продукции	6
Зал кафе	Организация дружелюбной, расслабляющей атмосферы. Увеличение дохода кафе.	Реализация продукции кухни кафе и готовой продукции	Оказание квалифицированной помощи при выборе блюд и напитков. Распределение заказа по месту его выполнения между Барменом и Поваром	6
Снабжение	Бесперебойное производство	Снабжение кафе сырьем и товарами	Планирование снабжения. Расчет оптимального запаса сырья и товаров	3

Схема информационных потоков кафе «Аква» представлена на рисунке 2:

- информационные потоки;
- СУпр – система управления предприятием;
- СУТП – система управления технологическими процессами;
- СУР – система управления ресурсами.

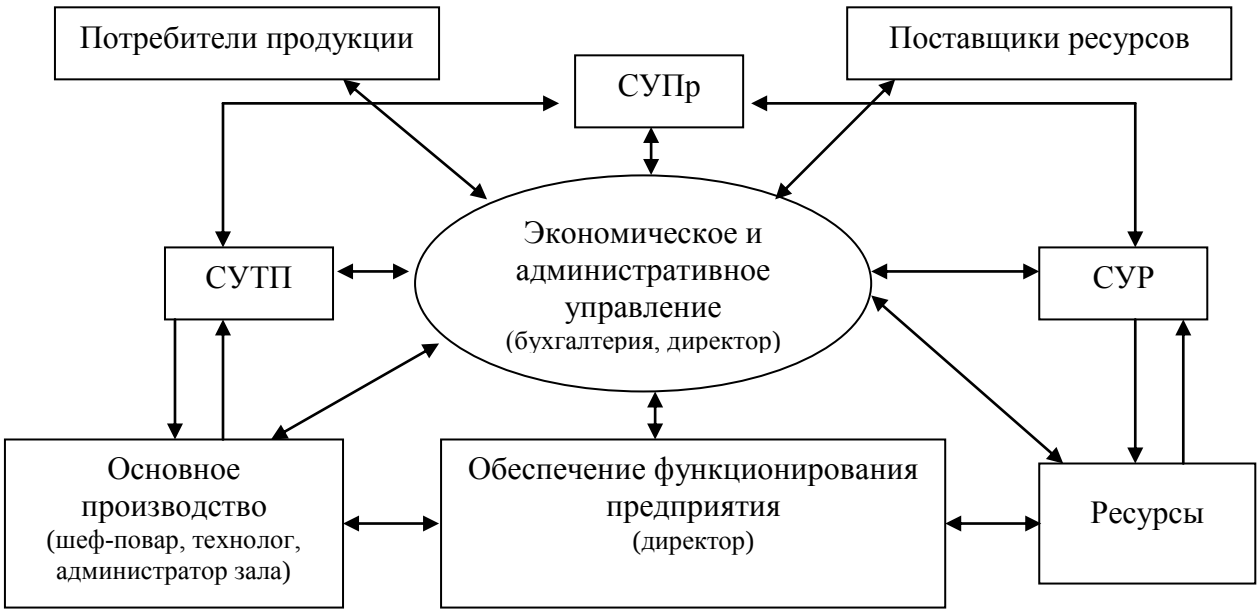


Рисунок 2 – Информационные потоки кафе «Аква»

1.2 Обоснование необходимости автоматизации рабочего места администратора кафе

Автоматизированное рабочее место администратора кафе, предполагает наличие стандартного набора устройств (компьютера, монитора, компьютерной мыши и клавиатуры). При необходимости - терминала сбора данных, обычного принтера формата А4.

Администратор кафе владеет информацией о столах которые были заказаны. АРМ позволяет администратору видеть, какие столы уже «заняты», а какие доступны для формирования заказа

Кроме эффективной связи с рабочими местами официантов, АРМ администратора наделён всеми необходимыми функциями по учёту занятости зала кафе.

Рассмотрим бизнес-процессы кафе «Аква», в которых принимает активное участие администратор.

Администратор занимается сферой обслуживания клиентов и задействован в следующих бизнес-процессах:

1.Регистрация клиента;

2.Создание заказа с непосредственным бронированием столика;

3.Контроль обслуживания заказов официантами.

Рассмотрим, какие функции предполагается автоматизировать с помощью АРМ «Администратор кафе»:

- регистрация посетителя с указанием первичной информации;
- создание заказа с непосредственным бронированием столика;
- заполнение справочников клиентов, блюд, столиков, организаций, персонала;
- построение отчетной документации

На текущий момент данная работа выполняется следующим образом: клиент звонит по телефону, желая заказать столик на определенный день и время, администратор смотрит журнал заказов и подбирает подходящий столик, фиксирует заказ в журнале, записывает номер телефона. При этом нередко администратор испытывает трудности в том, чтобы не запутаться во времени занятости столиков. В момент открытия кафе вечером, когда приходят посетители, администратору также бывает очень сложно сориентироваться, какие столики заняты, а какие свободны. Рассаживая гостей, администратор затрачивает очень много времени, что приводит к тому, что на выполнение других обязанностей администратору просто не хватает времени. Поэтому автоматизация представляется едва ли не единственным эффективным средством решить данные проблемы и усовершенствовать работу администратора.

1.3 Обзор существующих систем управления предприятием питания.

В качестве вариантов автоматизации существуют готовые типовые решения, также предлагается рассмотреть вариант индивидуальной разработки.

Готовые решения предназначены для полной автоматизации деятельности кафе.

Например, 1С: Ресторан. Программный комплекс "1С:Предприятие 8: Ресторан" предназначен для автоматизации фронт-офиса на предприятиях питания. В качестве платформы используется "1С: Предприятие 8", что позволяет легко администрировать и дорабатывать систему под специфические нужды любого предприятия, в не зависимости от его масштаба, типа и концепции. Решение может использоваться как в одиночных, так и сетевых предприятиях.

При помощи программного продукта можно автоматизировать рабочие места: кассир, официант, бармен, буфетчик, хостес, администратор (метрдотель).

Недостатком данного продукта является его высокая стоимость, оплата постоянного сопровождения. Так, стоимость 1С Ресторан на 01.11.2015 составляет 36 000 рублей (по данным сайта фирмы 1С [20]). Также программа нуждается в регулярном обслуживании, для этого необходимо заключить договор на обслуживание с компанией – франчайзинг фирмы 1С, например, ООО «Мастер-Софт». По самому дешевому тарифному плану «Стандартный» обслуживание составляет 1500 рублей/месяц, в год обслуживание составит 18000 рублей. При этом данная сумма не предусматривает консультаций по программному продукту, если у сотрудников что-то не будет получаться, программа не будет функционировать правильно или работники будут испытывать трудности при работе с программой, потребуется вызов специалиста, который будет оплачиваться отдельно по часовой ставке 1200 руб./час (данные взяты с сайта компании Мастер-Софт): [22]

Вторым вариантом автоматизации рабочего места администратора кафе является разработка индивидуального программного решения с использованием среды программирования Delphi. Преимуществами управления заказами кафе являются следующие:

- полное соответствие программного продукта требованиям заказчика;

- отсутствие ненужных (“лишних”) модулей;

- возможность быстрой доработки, т.к. разработчики и заказчик находятся в одном городе и сотрудничают по договору;

- возможность регулярных консультаций (при необходимости таковых, стоит отметить, что в первый год функционирования программного решения такая необходимость возникает достаточно часто);

- приемлемая стоимость программного решения.

Сравнение языков программирования их плюсы и минусы показаны в таблице 3.

Таблица 3 - Сравнение языков программирования

Параметры	**Delphi**	**C ++**	**Visual Basic**
Общие сведения	Создан корпорацией Borland, он вобрал в себя все лучшее, что эта фирма, славящаяся своими средствами разработки, накопила на данный момент. В его основу положен язык программирования Object Pascal	Процедурный язык программирования, который был разработан в 1970х годах. Он создавался для использования в ОС UNIX.	Средство разработки ПО, созданное корпорацией Майкрософт. Оно включает в себя среду разработки и язык программирования

Продолжение таблицы 3

Плюсы языкапп	Мощная и удобная интегрированная среда (IDE). Ни один компилятор C++, включая Visual C++, не предоставляет столь дружественной, интуитивно понятной, простой в использовании и вместе с тем столь многофункциональной оболочки как Delphi. В Delphi введены мощные средства поддержки работы с данными, позволяющие очень просто создавать приложения, связанные с базами данных.	Минимализм, обширный набор функций и лаконичность	Быстрое создание приложения для Windows с графическим интерфейсом; лёгкий синтаксис, который позволяет быстро освоить язык; предусмотрена защита от ошибок, связанных с доступом к памяти и применением указателей
Минусы языка	Сложность оптимизации, перегруженность минимальной программы, язык вытесняется более современными языками	Некоторые элементы потенциально опасны, а предсказать последствия их использования практически невозможно. Многие уязвимости невозможно увидеть ни при компиляции, ни во время исполнения. Язык чрезмерно сложен.	Можно наследовать интерфейс, но не реализацию объектов; чтобы работать с программой, нужна установка msvbvmXX.dll; сравнительно низкая скорость работы

2. Разработка автоматизированного рабочего места администратора кафе

2.1 Анализ требований к автоматизированной системе

Программа «АРМ администратора кафе» предусматривает реализацию следующих функций: работа с заказами, построение отчетов, заполнение справочников (рисунок 3).

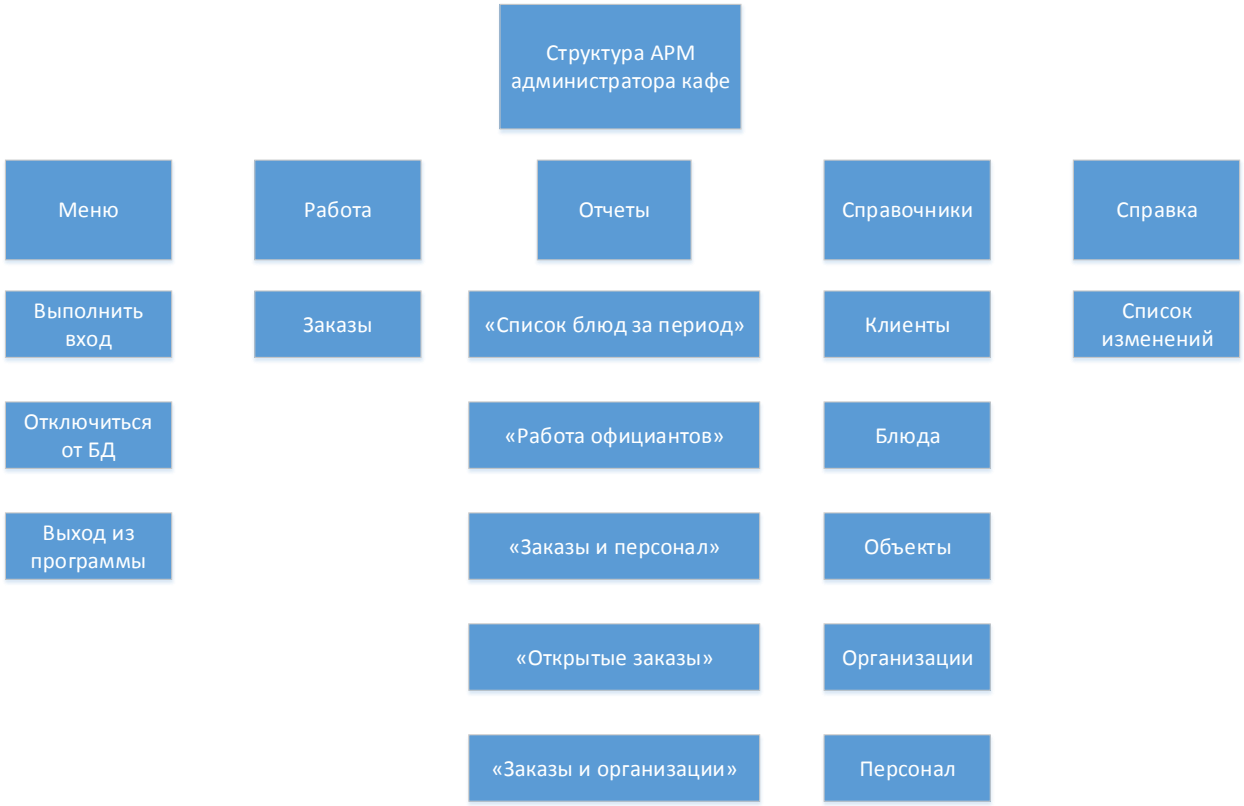


Рисунок 3 - Структура АРМ администратора кафе

Реализация функции «Меню» предусматривает подключение и отключение, а так же выход из АРМ «Администратор кафе».

Реализация функции «работа с заказами» предусматривает:

- регистрацию клиента;
- создание заказа / редактирование заказа;

- добавление позиций в заказ;

- просмотр списка заказов;

- печать чека / возможность предварительного просмотра.

Реализация функции «Отчеты» предусматривает построение аналитических отчетов по заданным критериям (дата, время, организация, блюда, персонал).

Реализация функции «Справочники» предусматривает добавление, редактирование, удаление первичной информации, необходимой для работы АРМ «Администратор кафе».

Реализация функции «Справка» предусматривает вывод списка изменений в текущей версии АРМ «Администратор кафе».

2.2 Проектирование информационной базы программного решения

В данном разделе работы будет спроектирована информационная система для создания информационной базы с целью автоматизации рабочего места администратора кафе.

Прежде чем проектировать базу данных, рассмотрим бизнес-процессы кафе «Аква», которые необходимо автоматизировать.

Для описания бизнес-процессов используем Microsoft Visio - это мощное решение для создания диаграмм, которое позволяет упростить и связать информацию, а также поделиться ей. Microsoft Visio обладает мощным интерфейсом со множеством опций для создания собственных методов организации информации. Посредством набора графических инструментов для отображения действий и объектов, с помощью Microsoft Visio мы построили схему процесса, на которой показаны исходные данные, результаты операций, ресурсы, необходимые для их выполнения, управляющие воздействия, взаимные связи между отдельными работами [18,с.55].

Схемы функциональных блоков представлены на рисунке 4.

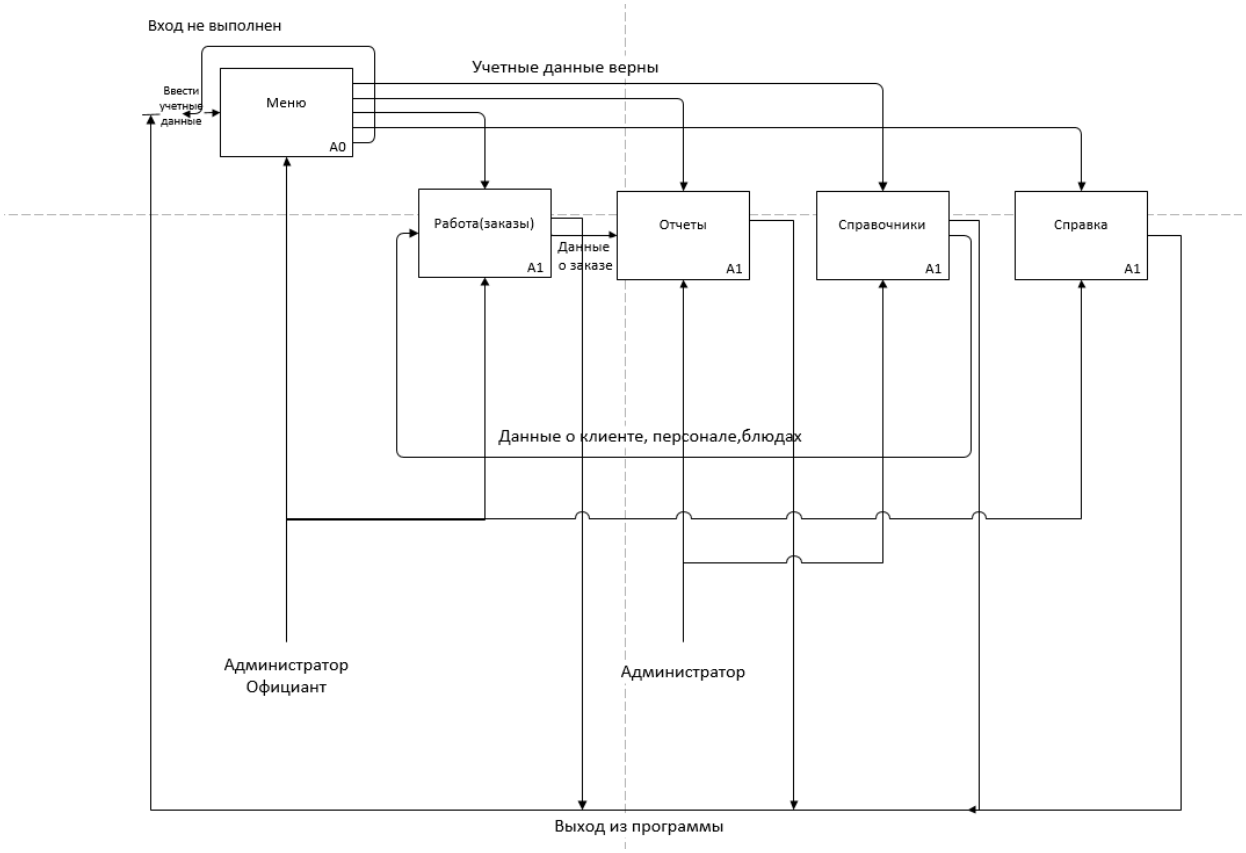


Рисунок 4 Схемы функциональных блоков АРМ «Администратор кафе»

Проектирование информационной системы начнем с описания объектов, которые необходимо создать и в дальнейшем реализовать (таблица 4,5,6,7,8,9,10).

Таблица 4 – Табчица чек в базе данных «администратор кафе»

Название столбца	**Тип**	**Описание**	**Ключ**
Id/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
id_zak/ № заказа	INT(11)/число	Обязательное, изменяемое поле.	

Продолжение таблицы 4

id_bl/ № блюда	INT(11)/ число	Обязательное, изменяемое поле.	
Kolvo/количество блюд	INT(11)/ число	Обязательное, изменяемое поле.	
Summa/сумма	DOUBLE (14,2)/ Число с плавающей запятой	Обязательное, изменяемое поле.	

Таблица 5 – Таблица пользователи в базе данных «администратор кафе»

Id/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
Login/логин пользователя	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	
Pass/пароль	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	
Prav/набор прав	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	
Name/имя пользователя	VARCHAR (50)/ строка	Обязательное, изменяемое поле.	
Priznak/Наличие (признак)	VARCHAR (255)/строка	Обязательное, изменяемое поле.	

Таблица 6 – Таблица справочник блюд в базе данных «администратор кафе»

Id/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
Name/наименование блюда	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	
Calories/калорийность блюда	INT(8)/ число	Обязательное, изменяемое поле.	
Cena/Цена блюда	DECIMAL (10,0)/ Точное число с десятичной точкой	Обязательное, изменяемое поле.	
Discount/Возможная скидка	DECIMAL (10,0)/ Точное число с десятичной точкой	Необязательное, изменяемое поле.	
Vid/Группа блюд	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	
Massa/масса блюда	DECIMAL (10,0)/ Точное число с десятичной точкой	Обязательное, изменяемое поле.	
Priznak/Наличие (признак)	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	

Таблица 7 – Таблица справочник клиентов в базе данных «администратор кафе»

id_kl_f/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
Name/ инициалы клиента	VARCHAR (50)/ строка	Обязательное, изменяемое поле	
Birthdate/ дата рождения	DATE/дата	Обязательное, изменяемое поле	
Dopinf/ дополнительная информация	VARCHAR (255)/ строка	Необязательное, изменяемое поле.	
Nphone/ номер телефона	VARCHAR (15)/ строка	Обязательное, изменяемое поле	
name_org/ наименование организации	VARCHAR (255)/ строка	Необязательное, изменяемое поле.	
Priznak/признак присутсвия	VARCHAR (255)/ строка	Необязательное, изменяемое поле.	

Таблица 8 – Таблица справочник столиков в базе данных «администратор кафе»

Id/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
Name/наименование объекта	VARCHAR (255)/ строка	Обязательное, неизменяемое поле.	
dop_inf/ дополнительная информация	VARCHAR (255)/ строка	Необязательное, изменяемое поле.	

Продолжение таблицы 8

date_n_a/дата начала аренды	DATETIME/дата и время	Обязательное, изменяемое поле	
id_kl/номер клиента	INT(11)/ числовой	Обязательное, изменяемое поле	

Таблица 9 – Таблица справочник организаций в базе данных «администратор кафе»

Id/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
name_org/ наименование организации	VARCHAR (255)/ строка	Обязательное, изменяемое поле	
nphone_org/ номер телефона организации	VARCHAR (255)/ строка	Обязательное, изменяемое поле	
dop_inf/ дополнительная информация	VARCHAR (255)/ строка	Необязательное, изменяемое поле.	

Таблица 10 – Таблица заказы в базе данных «администратор кафе»

Id/№ записи	INT(11)/ число	Обязательное, неизменяемое поле.	Primary
id_kl/уникальный номер клиента	INT(11)/ число	Обязательное, неизменяемое поле.	
date_n/дата начала заказа	DATETIME/дата и время	Обязательное, неизменяемое поле.	

Продолжение таблицы 10

date_o/дата окончания заказа	DATETIME/дата и время	Обязательное, изменяемое поле.	
dop_inf/ дополнительная информация	VARCHAR (255)/ строка	Необязательное, изменяемое поле.	
name_obj/ наименование столика	VARCHAR (255)/ строка	Обязательное, неизменяемое поле.	
Priznak/ признак выполнения заказа	VARCHAR (255)/ строка	Обязательное, изменяемое поле.	
Persona/персонал	VARCHAR (255)/ строка	Обязательное, неизменяемое поле.	

Далее спроектируем базу данных, для этого используем СУБД MySQL. Для визуального проектирования БД используем dbForge Studio for MySQL [11]. Благодаря ему можно наглядно представить структуру будущих таблиц и связи между ними. Это довольно простое решение, но оно позволяет спроектировать таблицы, обозначить первичные и внешние ключи, проверить связи. Схема представлена на рисунке 5.

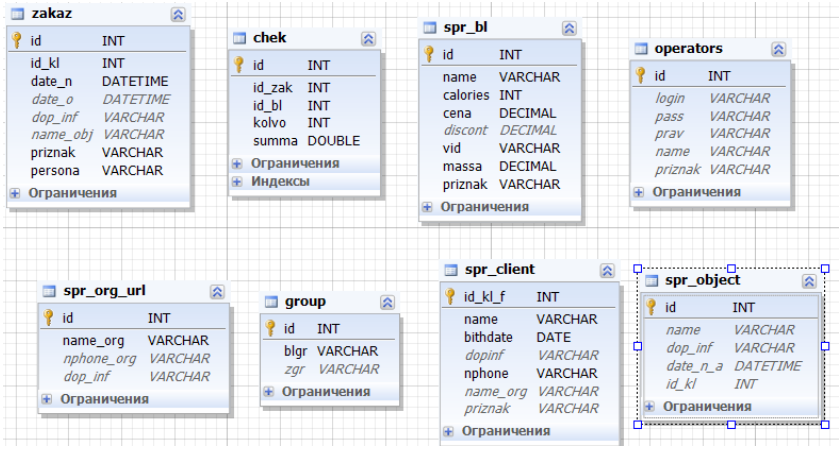


Рисунок 5 Схема базы данных АРМ администратора кафе

2.3 Обоснование проектных решений

Рассмотрим требования к техническому, программному и информационному обеспечению проекта автоматизации рабочего места администратора кафе.

Кафе «Аква» оснащено компьютером для администратора, а так же 2 рабочими местами для официантов.

Рассмотрим характеристику компьютера и требований к нему.

1. Материнская плата.

Socket-FM 2+ ;

Поддержка многоядерных процессоров;

Память: до 8 ГБ, не менее 2 DIMM слота, DDR3 1066/866;

Интегрировано:

- видеокарта, 512 Мб;

- сетевая карта 100 Мбит/с.

2. Исполнение системного блока с пониженным уровнем шума, на расстоянии 50 см от поверхности оборудования уровень шума не превышает 30 dB.

3. Корпус: ATX с блоком питания, обеспечивающим бесперебойную работу всего системного блока не менее 4-х внутренних 3.5" отсека для фиксированных HDD. Дополнительный вентилятор охлаждения 105 мм. 4 порта USB, аудио вход/выход на передней панели.

4. Процессор: кол-во ядер 2, тактовая частота не менее 3100 МГц, объем кэша не менее 2048 Кб с активным охлаждением.

5. Оперативная память: 2 Gb DDR3.

6. Жесткий диск: 500 Gb SATA

7. Дисковод: DVD+-RW SATA

8. Клавиатура PS/2; USB

9. Оптическая мышь 3-х кнопочная с переходником USB/PS2

10. Сетевой фильтр на 7 розеток, длина шнура 1,5 метра

11. Монитор 21"

Требования к техническому обеспечению также определяются разрабатываемой программой. Данная программа предъявляет следующие минимальные требования к характеристикам компьютера (с учетом требований конфигурации): процессор AMD Athlon 340 и выше, оперативная память не менее 512, жесткий диск (не менее 60 Мб свободного места), DVI дисплей. Для печати документов из программы компьютер администратора кафе должен быть оснащен локальным принтером или подключен к сетевому принтеру.

Разрабатываемый программный продукт должен иметь возможность быть установленным, запущенным на компьютере, чьи характеристики не ниже технических характеристик компьютера, параметры которого приведены в требованиях к техническому обеспечению.

Программное обеспечение копируется заказчику на компьютер. Установка, настройка и запуск программного обеспечения производится разработчиком программного обеспечения самостоятельно.

Разрабатываемый программный продукт должен соответствовать следующим требованиям:

- интерфейс на русском языке;

-должна обеспечиваться возможность работы на следующих операционных системах Windows XP и выше, а также быть совместимой с офисным пакетом MS Office 2003 rus SP3 и выше.

Требования к информационному обеспечению.

Программный продукт разрабатывается на основании информации, полученной в ходе собеседований со специалистами и сотрудниками кафе «Аква». Программный продукт должен иметь описание (руководство пользователя) на русском языке.

Руководство пользователя должно содержать разделы: описание интерфейса, технология работы с программой, расположение и структура данных.

2.4 Реализация программного решения

Для реализации программного решения используем среду программирования Delphi и базу данных MySQL .

Соединение с базой данных MYSQL

Для работы программного продукта с базой данных mysql был использован комплект компонентов доступа ZeosDBO. Данный комплект не использует дополнительных посредников для передачи информации, т.е. работает напрямую.

Для соединения с базой данных используется компонент ZConnection. Для настройки соединения необходимо указать логин, пароль, порт, название базы данных, а также протокол. Название протокола зависит от использованной версии mysql (требуется скачать dll). Пример настройки компонента ZConnection представлен на рисунке 6.

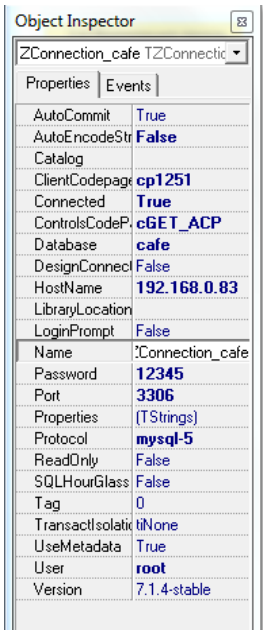


Рисунок 6 Настройки компонента ZConnection

Для соединение с таблицей базы данных используется стандартный источник данных DataSource, который подключается к компоненту выполнения запросов ZQuery(рисунок 7).

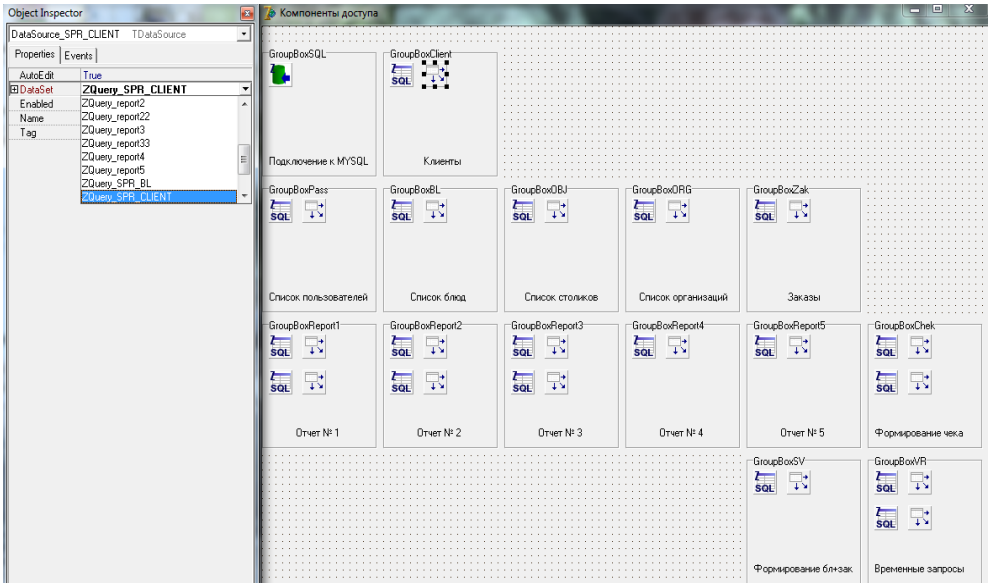


Рисунок 7 Подключение компонента DataSource к ZQuery

Компонент выполнения запросов ZQuery соединяется с ZConnection. В свойстве «SQL» данного компонента прописывается запрос к базе данных MYSQL. Запросы к базе данных можно изменять в процессе работы программы (рисунок 8).

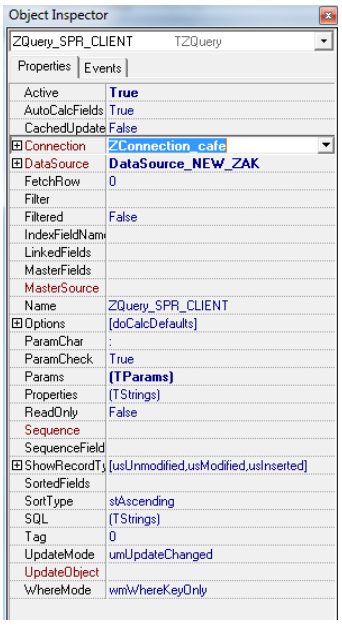


Рисунок 8 Подключение компонента ZQuery к ZConnection

Для удобства использования компонентов им даются наименования согласно их функциям, например для соединения с таблицей «Клиенты» (spr_client) используют компоненты DataSource_SPR_CLIENT, ZQuery_SPR_CLIENT. Аналогичным образом создается необходимое количество компонентов для остальных таблиц или функций.

Пример 1. подключение таблицы «Клиенты»(SPR_CLIENT).

ZQuery_SPR_CLIENT. В свойстве «sql» прописываем запрос «select * from spr_client». В свойстве «Connection» выбираем ZConnection_cafe. В свойстве «Active» выбираем «true»(включаем выполнение запроса, если запрос правильной структуры «включение» выполнится). DataSource_SPR_CLIENT подключается к ZQuery_SPR_CLIENT через свойство «dataset» в котором указывается компонент выполнения запроса ZQuery_SPR_CLIENT.(рисунок 9)

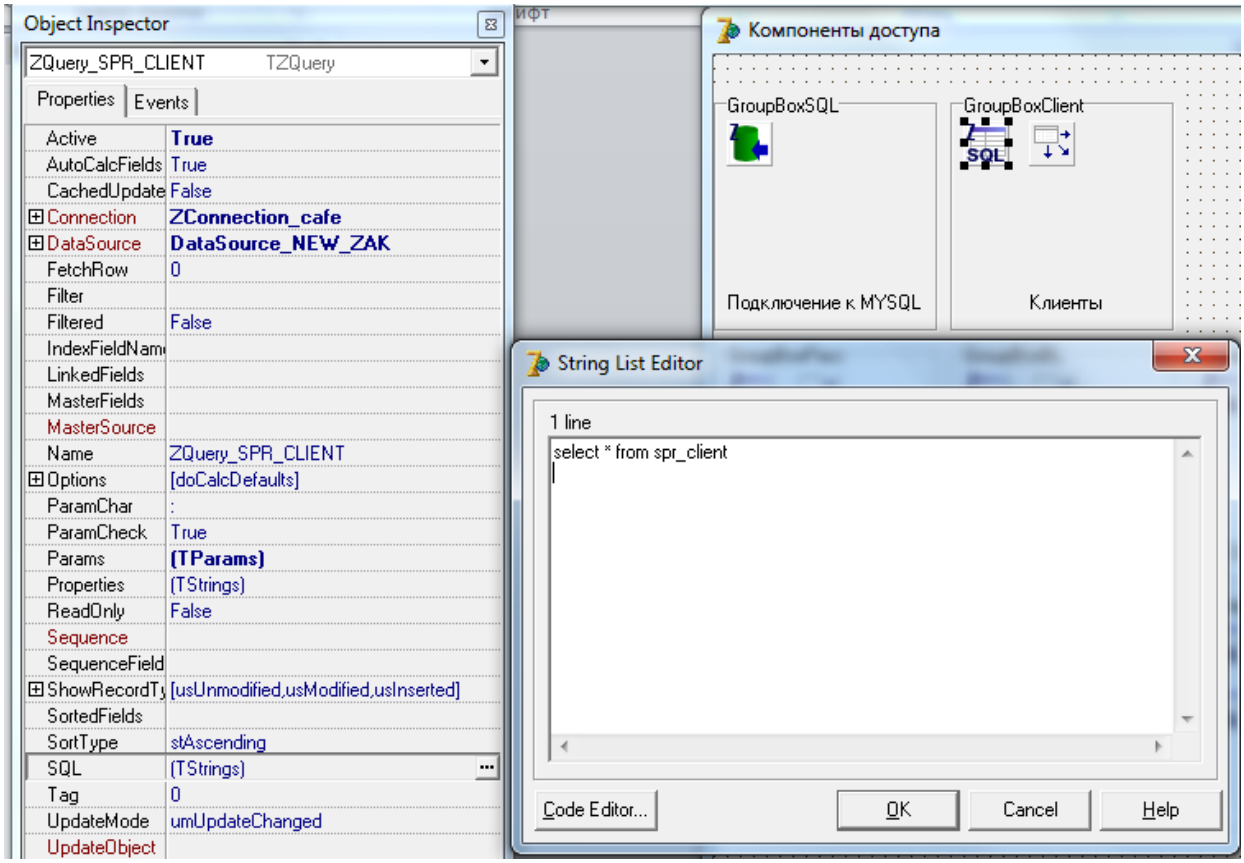


Рисунок 9 SQL запрос в компоненте ZQuery

Вывод информации после выполнения действия или запроса в виде таблицы.

Выводить информацию можно напрямую передавая ее свойству компонента. Для вывода информации в виде таблицы используется компонент DBGrid. Для соединения данного компонента используется свойство «DataSourse» в котором выбирается источник данных DataSource. Результатом заполнения данного свойства станет вывод из базы данных информации указанной в запросе нашего ZQuery. Компонент GroupBox используется для визуальной группировки компонентов (рисунок 10).

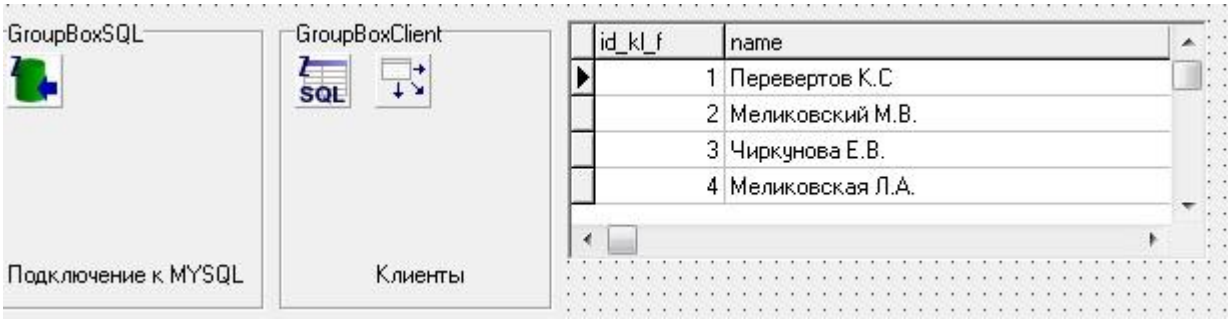


Рисунок 10 Вывод информации из компонента

Вывод информации после выполнения действия или запроса.

Вывести информацию из источника можно в любой компонент который поддерживает данную функцию. Для вывода информации в текстовую метку необходимо в процедуре компонента который будет «инициатором» написать код выполнения.

Пример 2. Вывод информации из таблицы «Пользователи»(SPR_CLIENT). в текстовое поле напрямую из запроса представлен на рисунке 11.

В данном случае «инициатором» выполнения действия будет компонент BitBtn(кнопка). В процедуре которой написан следующий код:

```
procedure TForm_PASS.BitBtn1Click(Sender: TObject);
begin
Form_DM.ZQuery_PASS.Close;
```

Form_DM.ZQuery_PASS.SQL.Text:='SELECT * FROM operators WHERE login=’1’ ';

Form_DM.ZQuery_PASS.Open;

Form_KL_ZAK.Label11.Caption:=DBGrid1.DataSource.DataSet.fieldbyname ('prav').AsString;

Form_KL_ZAK.Label11.Caption:=DBGrid1.DataSource.DataSet.fieldbyname ('name').AsString;

End;

Пояснение: отключаем компонент выполнения запроса, присваиваем свойству «sql» новый запрос, выполняем запрос, передаем значение.

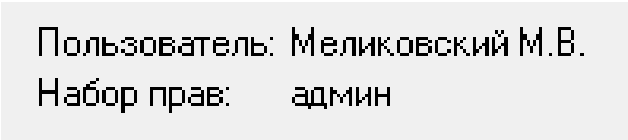


Рисунок 11 Вывод информации из таблицы «Пользователи»

Пример 2.1. Вывод информации из таблицы «Пользователи» (SPR_CLIENT) в текстовое поле с использованием параметра.

В большинстве случаев для вывода, редактирования, удаления информации по запросу в структуре запроса используют параметр, который можно получить из любого компонента.

В данном случае параметром для запроса будет выступать значение, введённое в текстовое поле

Edit.text.

procedure TForm_PASS.BitBtn1Click(Sender: TObject);

begin

Form_DM.ZQuery_PASS.Close;

Form_DM.ZQuery_PASS.SQL.Text:='SELECT * FROM operators WHERE login=: p_login ';

Form_DM.ZQuery_PASS.Params.ParamByName('p_login').Value := Edit1.text;

Form_DM.ZQuery_PASS.Open;

Form_KL_ZAK.Label11.Caption:=DBGrid1.DataSource.DataSet.fieldbyname ('prav').AsString;

Form_KL_ZAK.Label11.Caption:=DBGrid1.DataSource.DataSet.fieldbyname ('name').AsString;

End;

Структура базы данных, параметры компонентов Delphi.

Создание структуры базы данных подразумевает, что каждое поле в БД будет иметь свою структуру. Например: число, дата, дата и время, текст. Для корректного выполнения запроса необходимо, чтоб параметр, участвующий в запросе был такого же типа, как и поле для которого устанавливается параметр.

Для того чтоб изменять тип полей в Delphi используются специальные функции.

Пример 3. Изменение типа данных для использования в запросах.

В данном случае, если мы хотим выполнить запрос, который будет использовать параметр «дата и время» введенный в текстовое поле, необходимо выполнить преобразование. Аналогично преобразуются и другие типы данных. Преобразование типов данных может происходить в любой момент выполнения действия. Число преобразований не ограничено.

StrToDateTime(MaskEdit_DATE_N.Text)-из «текста» в «дата и время»

StrToInt(Edit1.Text)-из «текста» в «число».

Использование функции сравнения данных, выполнение по условию.

Большинство запросов выполняется с использованием нескольких параметров которые необходимо сравнить, а затем выполнить какое либо действие.

Пример 3.1. Сравнение параметров с использованием переменных.

Для выполнения сравнения необходимо преобразовать данные и присвоить их переменным которые будут их «хранить» для выполнения

действия. Переменные тоже имеют свой тип, который зависит от того, что собираются в нем «хранить». В данном примере выполняется преобразование параметра «текст» в «дата и время», присваивается значение переменным и выполняется сравнение, в зависимости от выполнения сравнения выводится информационное сообщение. Для выполнения условий используется конструкция if then else(если…тогда…иначе) в которую вводятся условия.

```
procedure TForm_report3_usl.BitBtn_predpClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if (a>b) then begin
ShowMessage('Дата не корректна');
end
else
ShowMessage('Дата корректна);
```

Перечень используемых типов компонентов в структуре проекта:

- текстовые: Label(метка), LabelEdit(текст+метка), Edit(текст), MaskEdit(текст), Memo(многострочный текст), DBGrid(таблица) (рисунок 12);

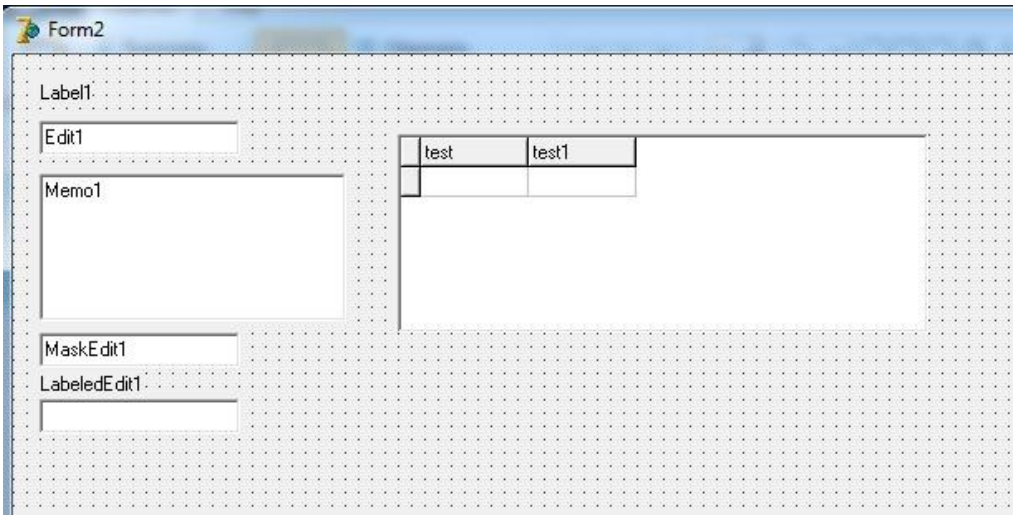


Рисунок 12 Визуальнотекстовые компоненты

-кнопки: BitBtn(графическая кнопка), Button(кнопка), RadioButton(переключатель), CheckBox(отметка) (рисунок 13);

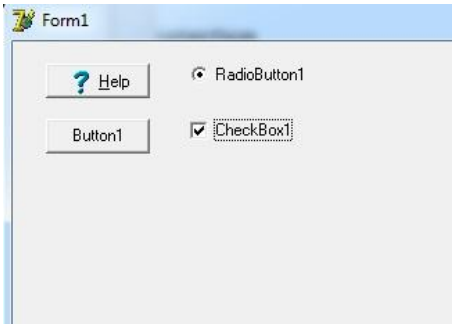


Рисунок 13 Использование кнопок и переключателей

- GroupBox(визуальные группировка), Panel(панель), MainMenu1(меню), PageControl(вкладки) (рисунок 14).

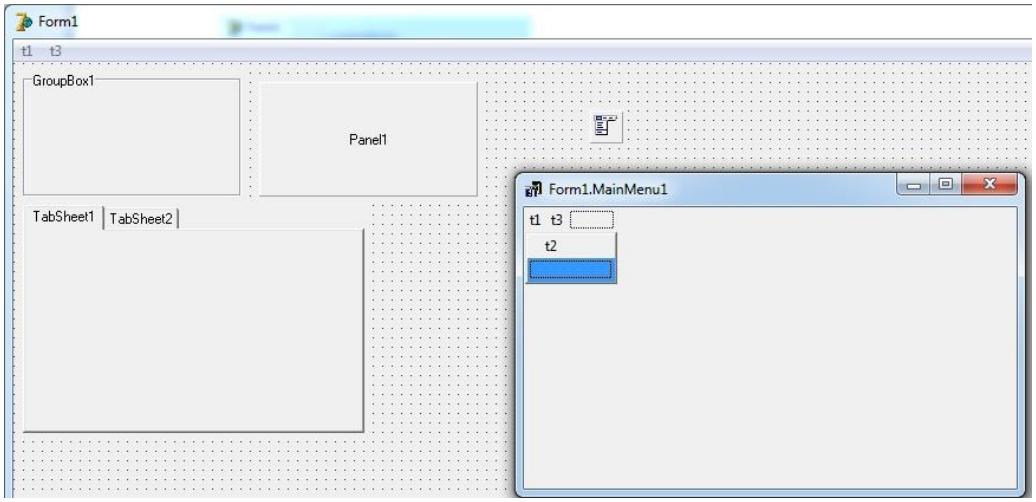


Рисунок 14 Группировщики, панели, вкладки, меню

2.5 Контрольный пример реализации проекта и его описание

Рассмотрим работу программы на конкретном примере:

1. При запуске программы выходит окно авторизации пользователей кафе. В зависимости от профиля прав пользователя набор его прав изменяется после авторизации. Для персонала с набором прав «официант» - недоступны

вкладки: справочники и отчеты. На рисунке 15 представлена форма авторизации, на рисунке 16 представлен внешний вид формы для профиля прав «администратор», на рисунке 17 представлен внешний вид формы для профиля прав «официант».

Рисунок 15 Форма авторизации

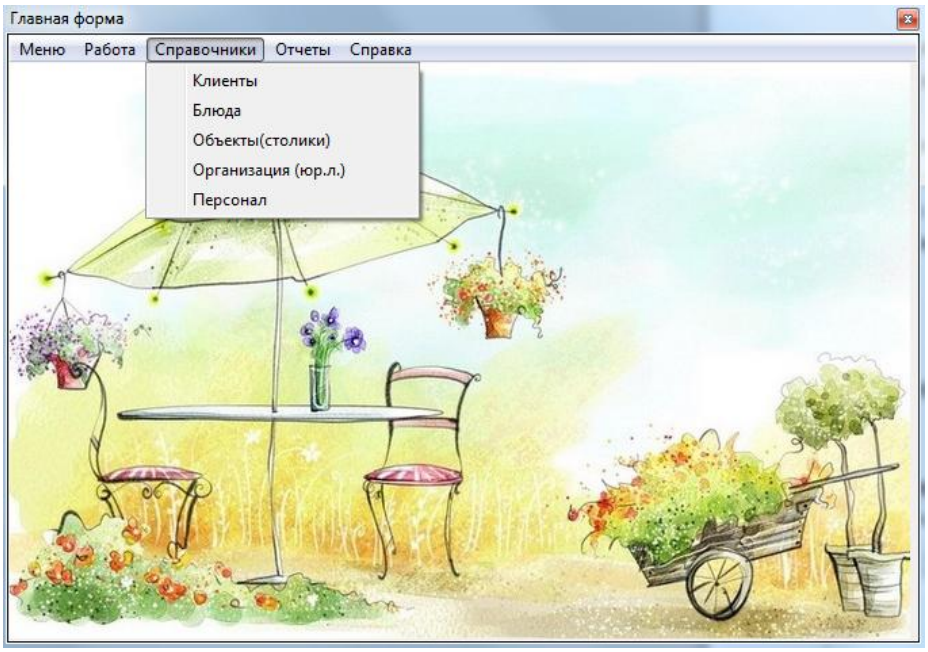


Рисунок 16 Внешний вид формы для профиля прав «администратор»

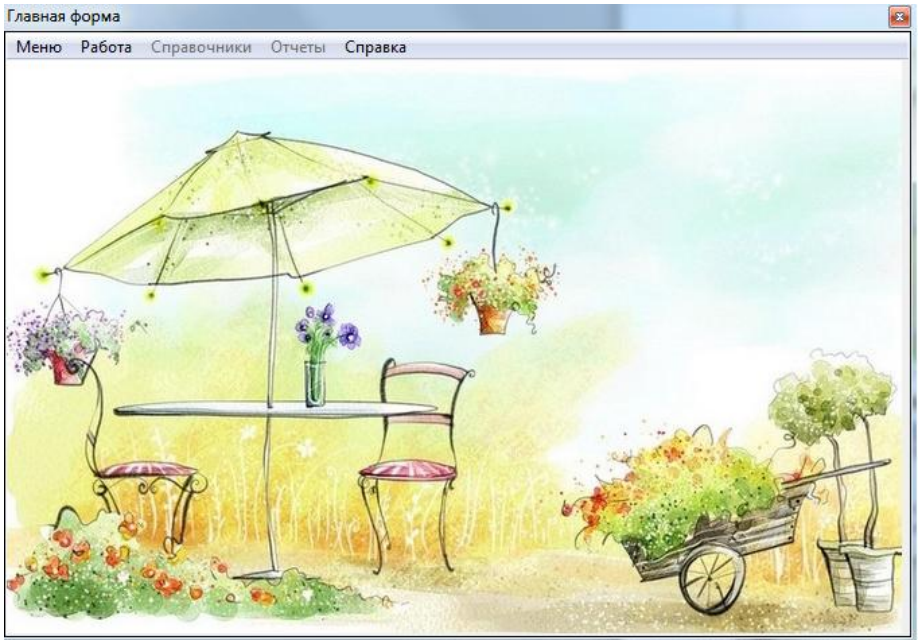


Рисунок 17 Внешний вид формы для профиля прав «официант»

Для работы с вкладкой заказы в меню программы выбираем – Работа → Заказы.

В правом верхнем углу формы фиксируется какой пользователь вошел в программу и его профиль прав. В зависимости от прав пользователя изменяется функционал.

На верхней панели формы Клиенты и заказы располагаются кнопки:

Кнопка новый клиент. При нажатии на форме появляются поля для ввода информации о новом клиенте (рисунок 18).

Редактировать - при клике по кнопке возникает форма для редактирования клиентов в справочнике. Эта форма содержит в себе такой функционал как – быстрый поиск с технологией живого поиска, кнопку обновления для отображения полного списка в случае поиска конкретного лица.

Удалить – позволяет удалять и восстанавливать удаленные записи клиентов. Удаленные записи не отображаются для набора прав официант.

Пользователь с набором прав «официант» не имеет права редактировать и удалять клиентов.

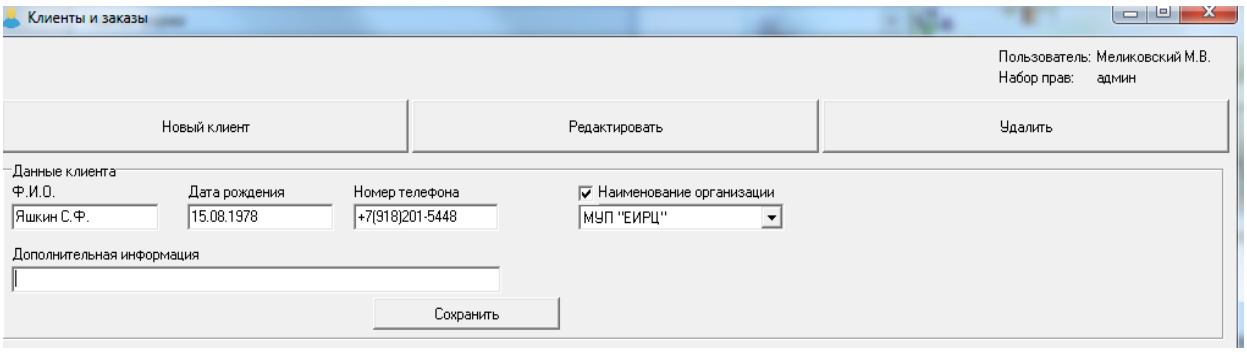


Рисунок 18 добавление нового клиента

Средняя панель формы Клиенты и заказы включается в себя таблицу клиентов с возможностью быстрого поиска(динамическое изменение параметров). Кнопка обновить список, используется для обновления списка клиентов (рисунок 19).

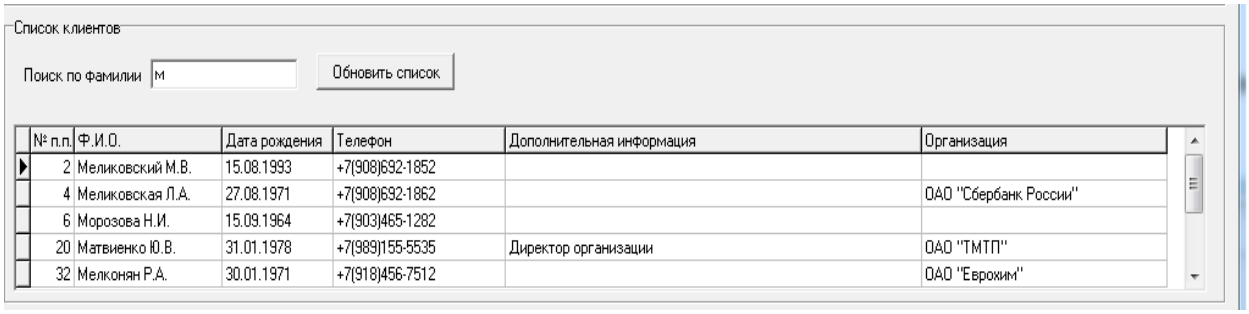


Рисунок 19 Технология живого поиска

Нижняя панель формы Клиенты и заказы взаимодействует со средней панелью по выбранному клиенту и содержит в себе 2 вкладки:

Новый заказ – позволяет создать заказ по выбранному клиенту с указанием обслуживающего персонала из выпадающего списка, времени и даты начала заказа (текущее время присваивается автоматически после выбора клиента) и номера столика(выбирается только свободный столик), также возможно внесение каких либо пометок или примечаний для этого реализовано поле дополнительная информация (рисунок 20).

Пользователи с набором прав «официант» не имеют возможности выбрать обслуживающий персонал из выпадающего списка официантов.

Рисунок 20 Создание нового заказа

Создав и сохранив заказ, автоматически переходим на 2 вкладку – список заказов. Здесь представлен полный список заказов, так же существует возможность выбора только открытых заказов (рисунок 21). Персоналу с набором прав «официант» доступны только персональные заказы.

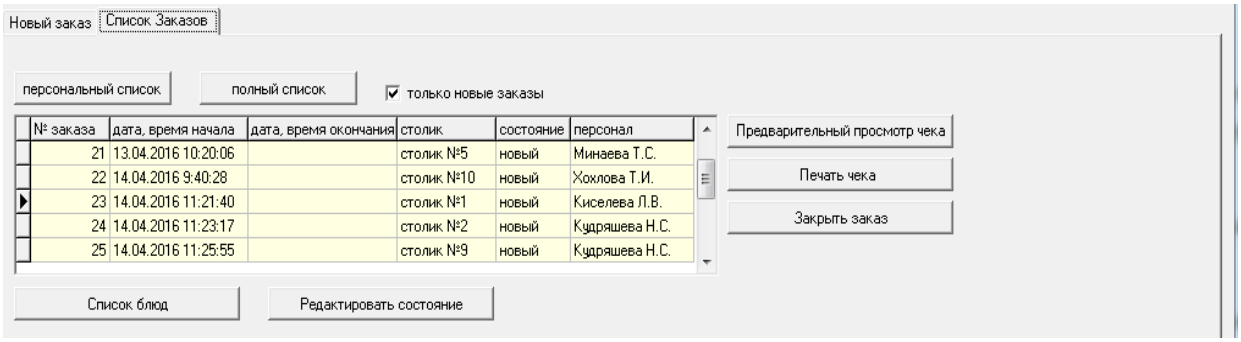


Рисунок 21 Список открытых заказов

После выбора заказа необходимо внести список блюд. Для этого необходимо нажать на кнопку «список блюд», после чего открывается форма «информация по заказу», где можно добавить отредактировать или удалить блюдо из заказа**.** Пример поиска блюда представлен на рисунке 22.

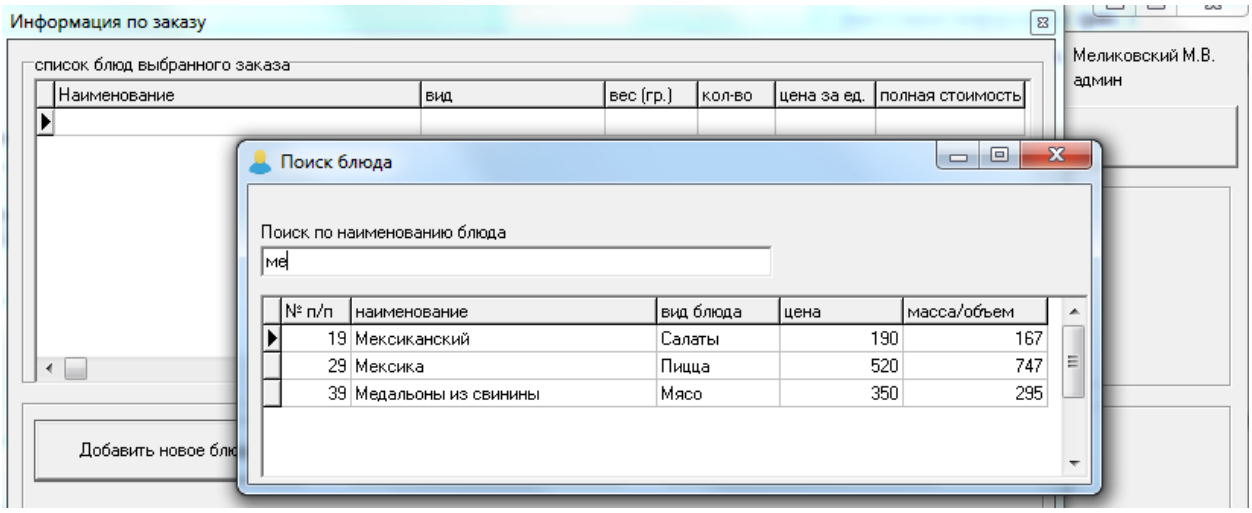


Рисунок 22 Поиск блюда

После выбора блюда из справочника необходимо заполнить количество порций. Поле «сумма» считается автоматически, учитывая количество порций, возможную скидку, а также цену за единицу. Пример выбора блюда из справочника представлен на рисунке 23.

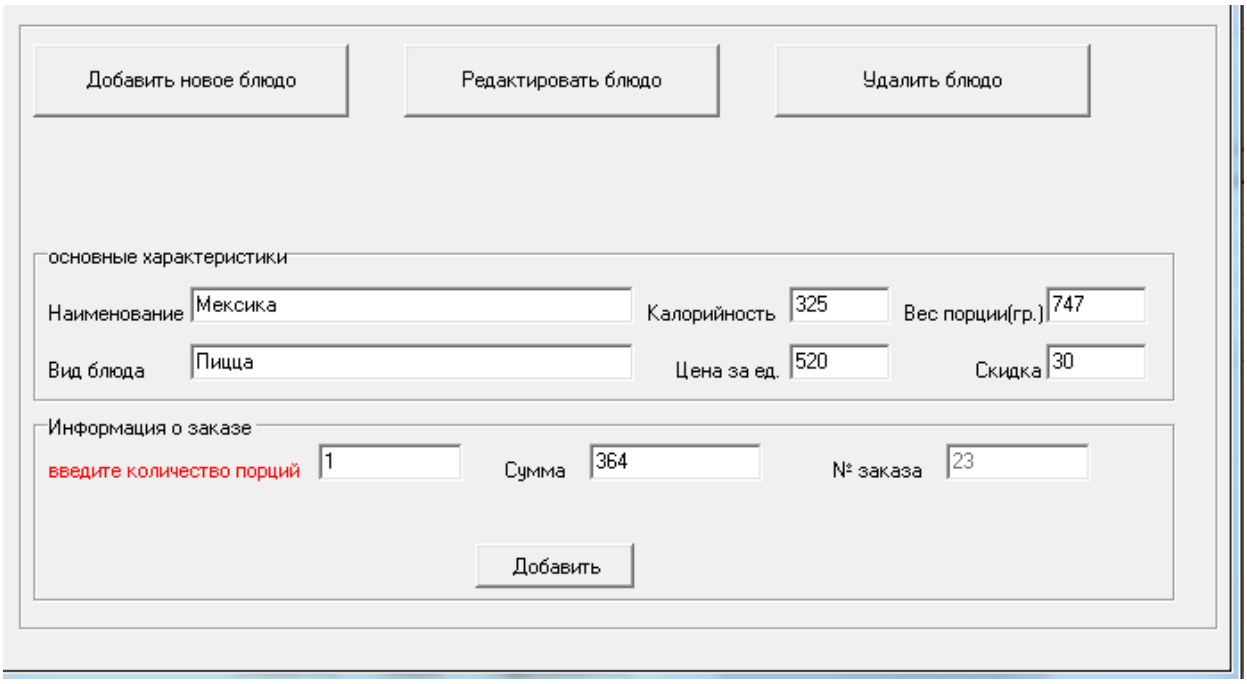


Рисунок 23 Добавление блюда в заказ

Для формирования чека необходимо выполнить действие – «закрыть заказ». При этом программа попросит указать дату и время закрытия заказа, так

же реализована защита от необдуманных ошибок пользователей – дата окончания заказа не может быть раньше даты начала, иначе появится информационное сообщение – «дата окончания некорректна». По умолчанию программа будет предлагать датой закрытия заказа текущую дату и время на компьютере (рисунок 24).

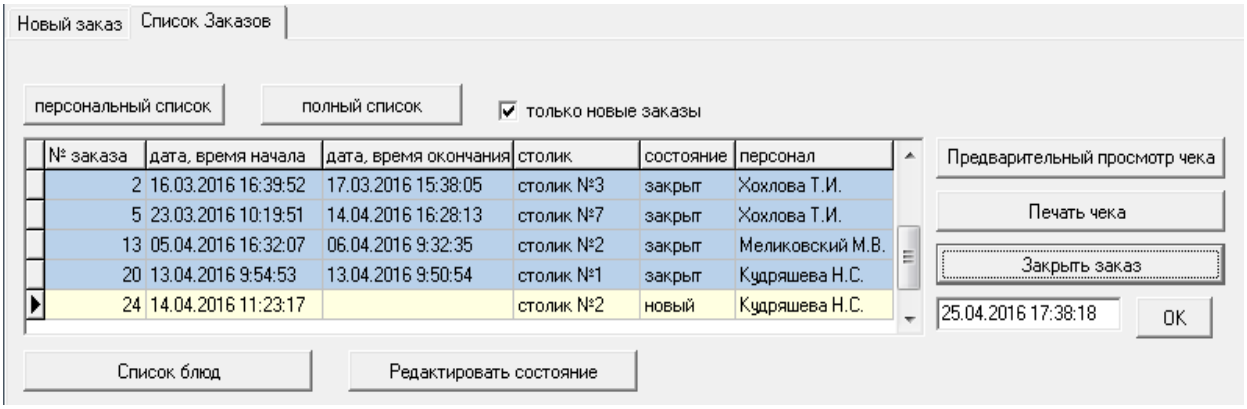


Рисунок 24 Закрытие заказа

Редактировать список блюд закрытого заказа невозможно. Для того чтобы разблокировать заказ необходимо нажать кнопку «редактировать состояние». Данный функционал доступен только пользователю с набором прав «администратор».

Далее когда заказ закрыт можем переходить к печати чека при помощи соответствующей кнопки на вкладке заказы (рисунок 25). Так же реализован функционал предварительного просмотра который доступен при открытом заказе (рисунок 26) и защита при попытке напечатать чек при открытом заказе появится информационное сообщение – закройте заказ.

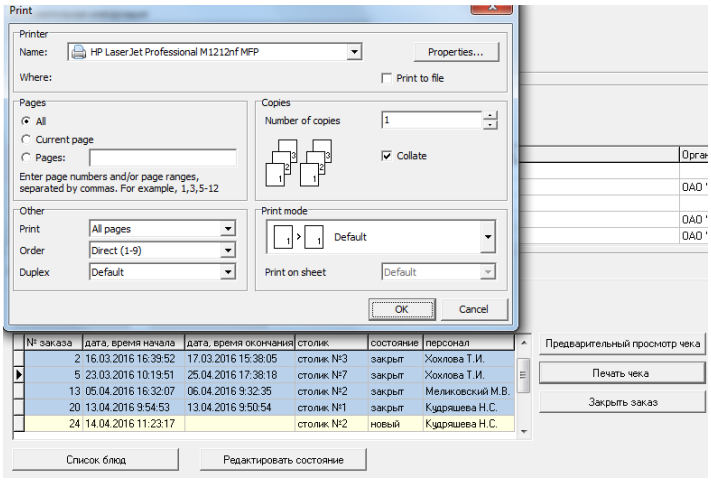


Рисунок 25 Печать чека

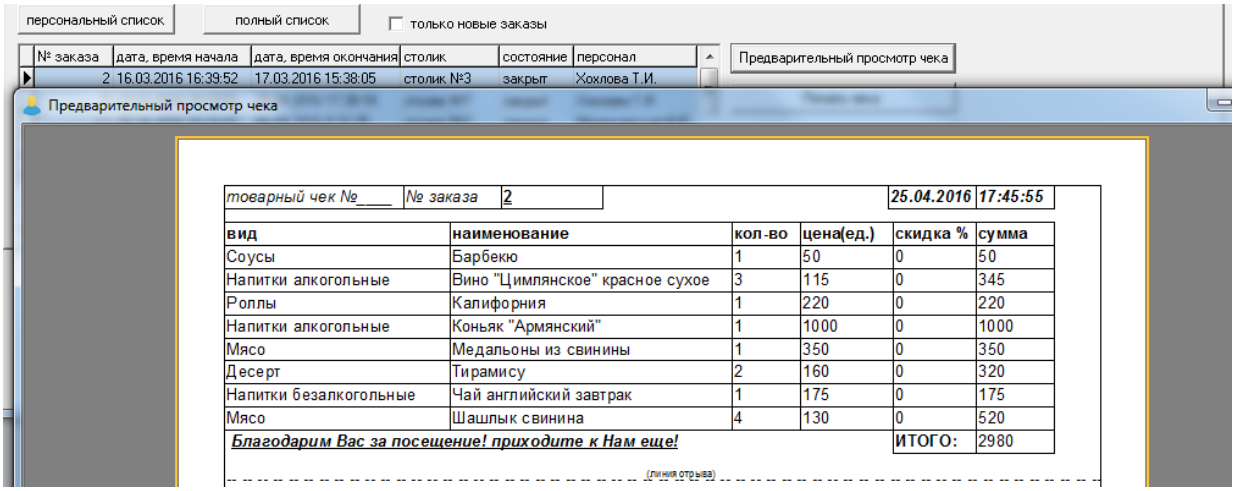


Рисунок 26 Предварительный просмотр чека

Для построения аналитических отчетов используется вкладка отчеты. Пользователю с набором прав «администратор» доступны 5 отчетов:

- Список блюд за период – формируется с учетом выбранного временного промежутка. Он включает в себя вид блюда, его наименование, а также количество и сумму (рисунок 27);

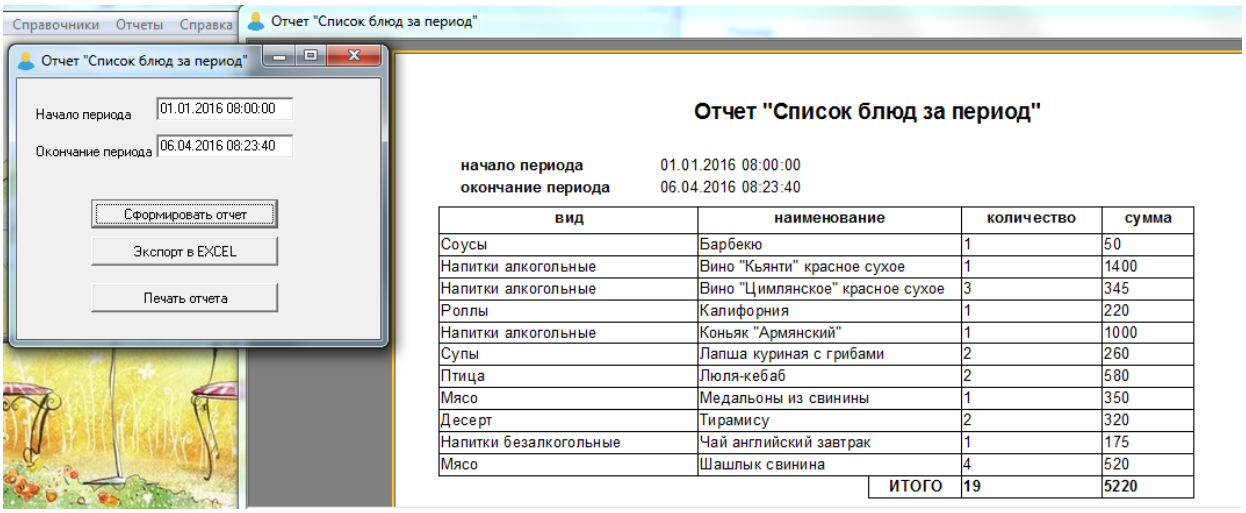


Рисунок 27 Отчет «список блюд за период »

- Работа официантов - формируется с учетом выбранного временного промежутка. Он отражает в себе фамилию официанта, общее количество обслуженных им заказов за период, а также общую сумму по всем обслуженным заказам (рисунок 28);

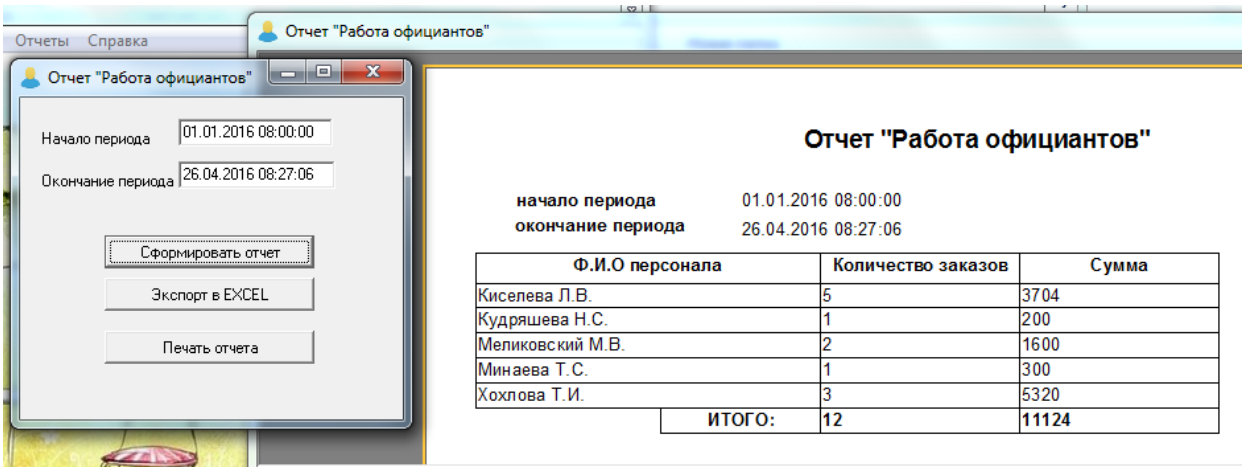
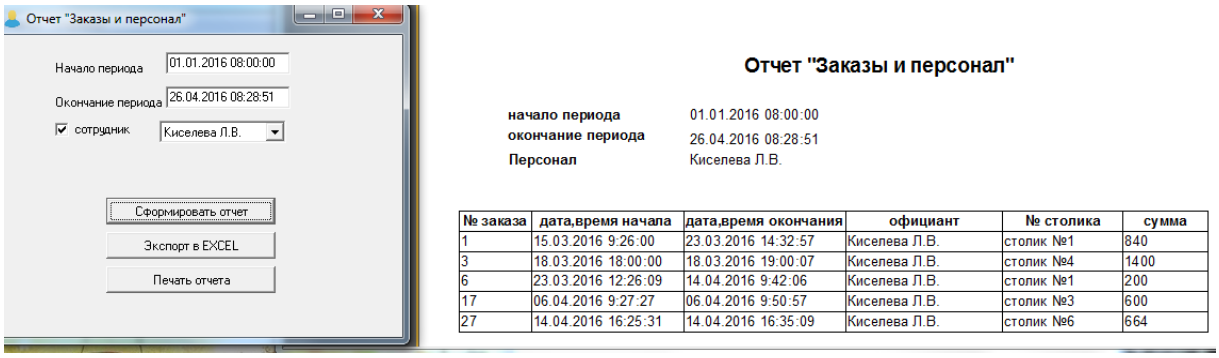


Рисунок 28 Отчет «работа официантов»

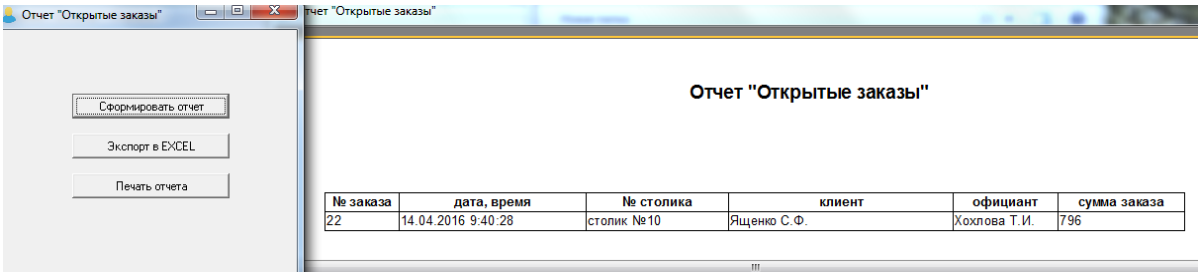
- Заказы и персонал - формируется с учетом выбранного временного промежутка, а также есть возможность указать конкретное лицо по которому будет сформирован отчет. Этот отчет отражает время и дату когда официант начал обслуживать заказ и момент закрытия этого заказа с указанием номера столика и суммы по заказу (рисунок 29);



№ заказа	дата,время начала	дата,время окончания	официант	№ столика	сумма
1	15.03.2016 9:26:00	23.03.2016 14:32:57	Киселева Л.В.	столик №1	840
3	18.03.2016 18:00:00	18.03.2016 19:00:07	Киселева Л.В.	столик №4	1400
6	23.03.2016 12:26:09	14.04.2016 9:42:06	Киселева Л.В.	столик №1	200
17	06.04.2016 9:27:27	06.04.2016 9:50:57	Киселева Л.В.	столик №3	600
27	14.04.2016 16:25:31	14.04.2016 16:35:09	Киселева Л.В.	столик №6	664

Рисунок 29 Отчет «заказы и персонал»

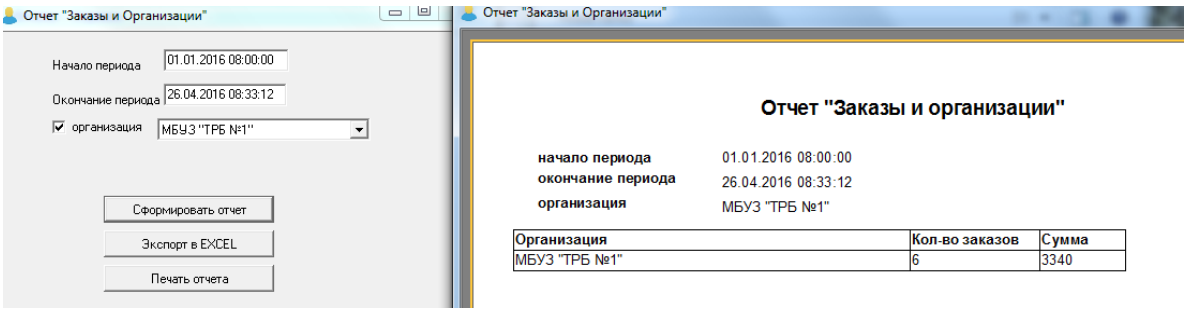
- Открытые заказы- показывает все открытые заказы с момента старта программы (рисунок 30);



№ заказа	дата, время	№ столика	клиент	официант	сумма заказа
22	14.04.2016 9:40:28	столик №10	Ященко С.Ф.	Хохлова Т.И.	796

Рисунок 30 Отчет «открытые заказы»

- Заказы и организации - формируется с учетом выбранного временного промежутка, а также есть возможность указать конкретную организацию по которой будет сформирован отчет (рисунок 31). Он включает в себя наименование организации количество заказов организации за период и общую сумму по ним (рисунок 32).



Организация	Кол-во заказов	Сумма
МБУЗ "ТРБ №1"	6	3340

Рисунок 31 Отчет «заказы и организации» на примере выбранной организации

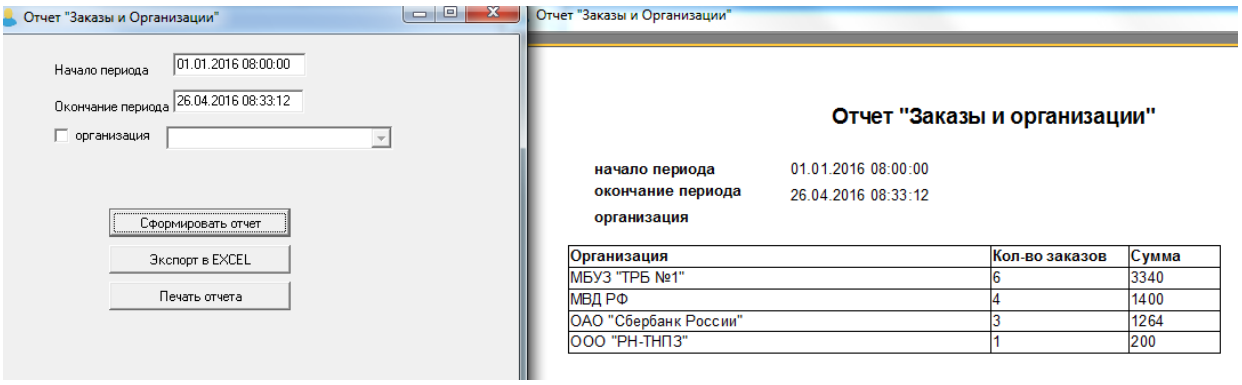
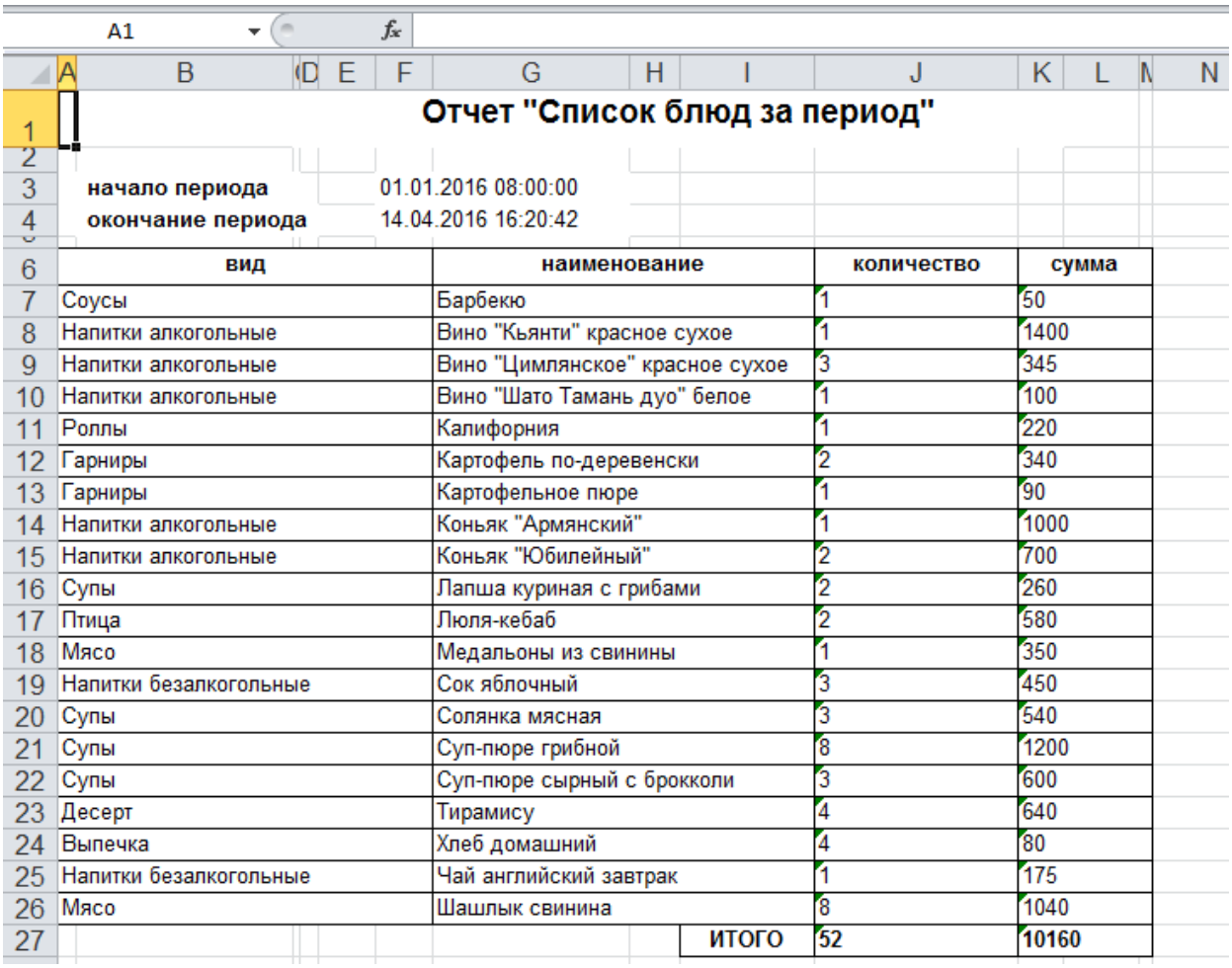


Рисунок 32 Отчет «заказы и организации»

Все отчеты имеют функцию предварительного отчета, вывода на печать и экспорта в Microsoft Excel (рисунок 33).



Отчет "Список блюд за период"

начало периода	01.01.2016 08:00:00
окончание периода	14.04.2016 16:20:42

вид	наименование	количество	сумма
Соусы	Барбекю	1	50
Напитки алкогольные	Вино "Кьянти" красное сухое	1	1400
Напитки алкогольные	Вино "Цимлянское" красное сухое	3	345
Напитки алкогольные	Вино "Шато Тамань дуо" белое	1	100
Роллы	Калифорния	1	220
Гарниры	Картофель по-деревенски	2	340
Гарниры	Картофельное пюре	1	90
Напитки алкогольные	Коньяк "Армянский"	1	1000
Напитки алкогольные	Коньяк "Юбилейный"	2	700
Супы	Лапша куриная с грибами	2	260
Птица	Люля-кебаб	2	580
Мясо	Медальоны из свинины	1	350
Напитки безалкогольные	Сок яблочный	3	450
Супы	Солянка мясная	3	540
Супы	Суп-пюре грибной	8	1200
Супы	Суп-пюре сырный с брокколи	3	600
Десерт	Тирамису	4	640
Выпечка	Хлеб домашний	4	80
Напитки безалкогольные	Чай английский завтрак	1	175
Мясо	Шашлык свинина	8	1040
	ИТОГО	52	10160

Рисунок 33 Экспорт в Microsoft Excel

АРМ «администратор» использует для работы справочники. Список справочников представлен в разделе «справочники» и состоит из 5 справочников:

- Клиенты - содержит в себе информацию о клиентах (рисунок 34) ;

Регистрация клиента необходима для создания заказа. Регистрацию можно произвести 2 способами:

1)для персонала с набором прав «официант» через форму «Заказы», выбрав необходимое действие «Новый клиент», заполнив все необходимые поля. Если клиент представляет из себя юридическое лицо, необходимо заполнить поле «организация».

2) для персонала с набором прав «админ» регистрация клиента возможна способом описанным выше и через справочник «Клиенты».

Редактирование удаление, восстановление клиентов из справочников возможно только с набором прав «админ», что аналогично и для других справочников. При удалении клиента из справочника записи присваивается признак «удален» и для такого клиента оформление заказа становится невозможным. Персонал с набором прав «официант» не увидит эту запись в таблице клиентов.

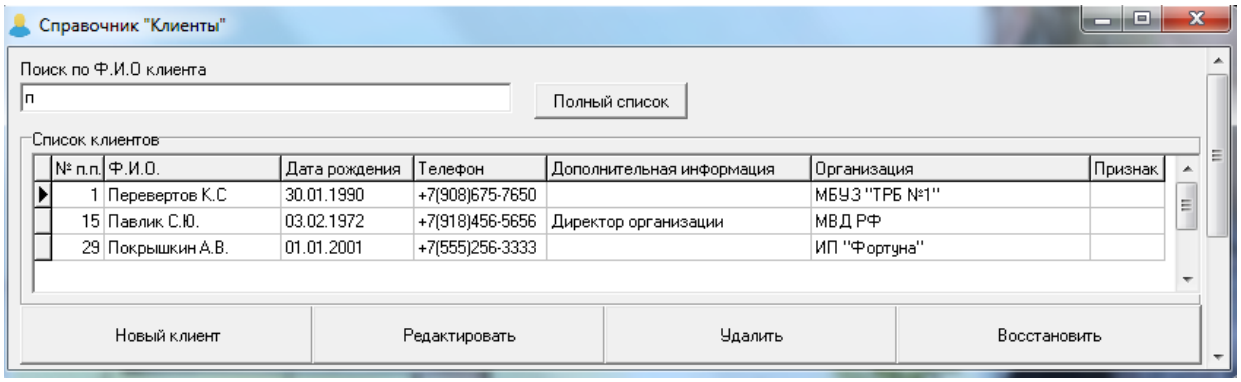


Рисунок 34 Справочник «клиенты»

- Организации – в случае если клиент представляет собой юридическое лицо для него при регистрации указывается наименование организации из

выпадающего списка. Заполнять справочник организаций имеет право только пользователь с набором прав «администратор» (рисунок 35) ;

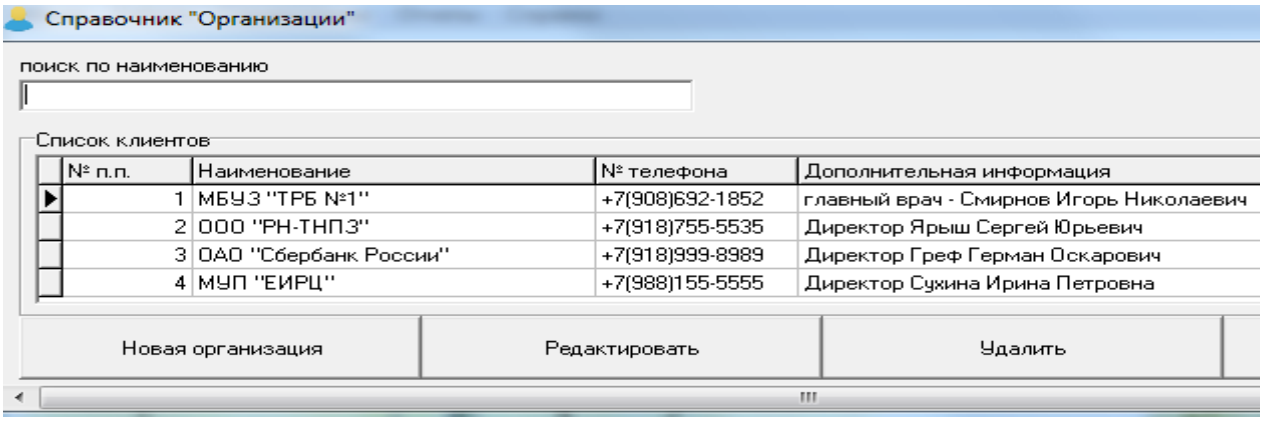


Рисунок 35 Справочник «организации»

- Объекты - содержит в себе информацию о списке столиков, которые необходимы для создания нового заказа персоналом (рисунок 36) ;

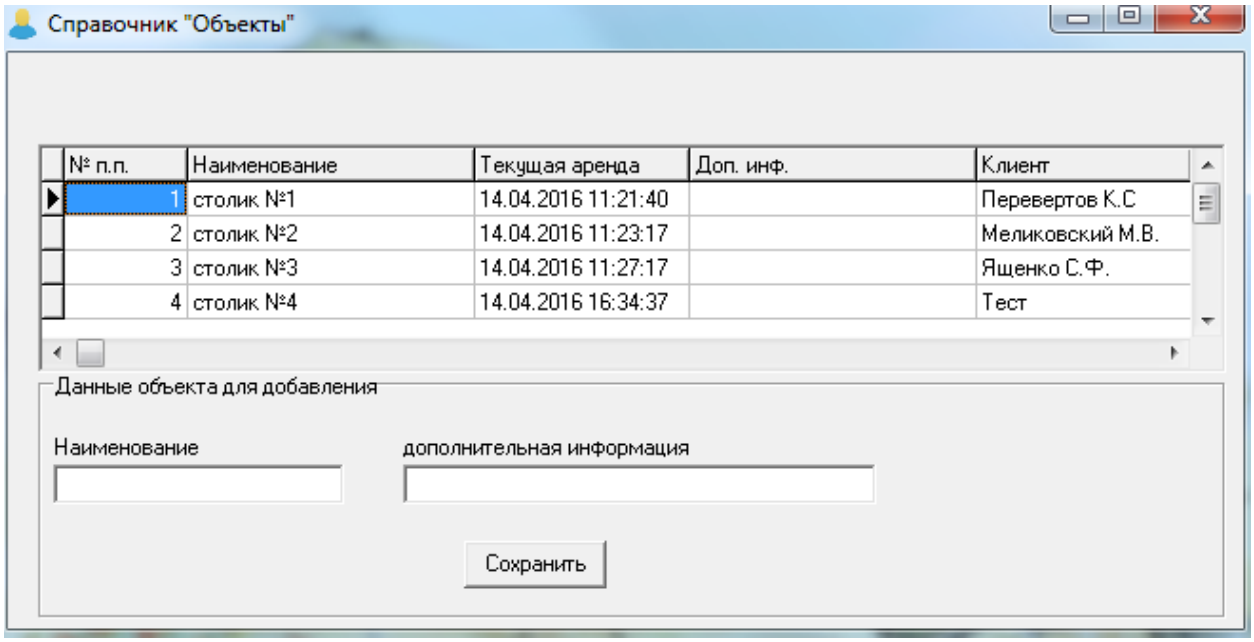


Рисунок 36 Справочник «объекты»

- Персонал - содержит в себе список лиц, имеющих право работать в программе. Для персонала существует 2 набора прав: «официант» (ограниченные права), «администратор» (полный набор прав) (рисунок 37) ;

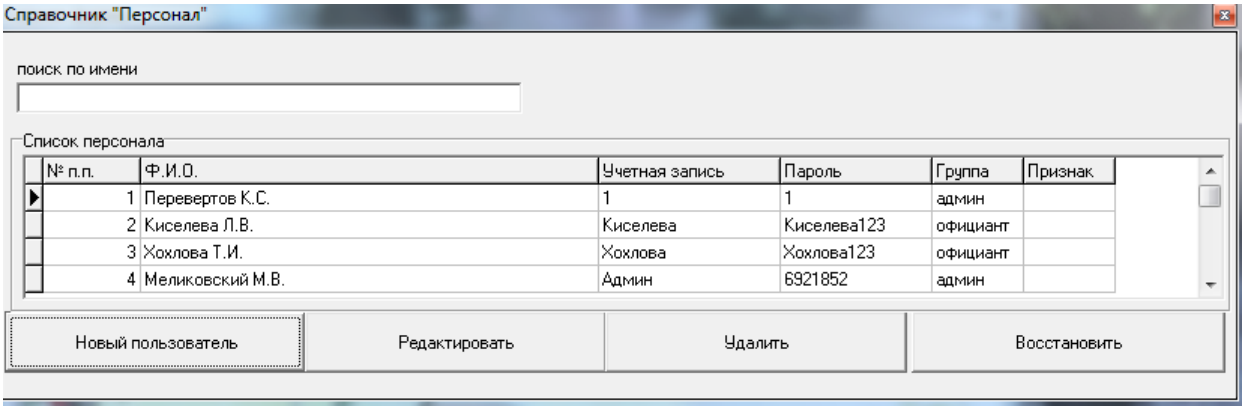


Рисунок 37 Справочник «персонал»

- Блюда - включает в себя список блюд из которых формируется заказ. Администратор имеет возможность «скрывать» блюда которые временно отсутствуют в ассортименте, т.е. официант при формировании заказа не сможет выбрать «скрытое» блюдо из справочника. Ведется учет скидки для конкретного блюда, т.е. при формировании заказа программа будет учитывать скидку на конкретное блюдо (рисунок 38) .

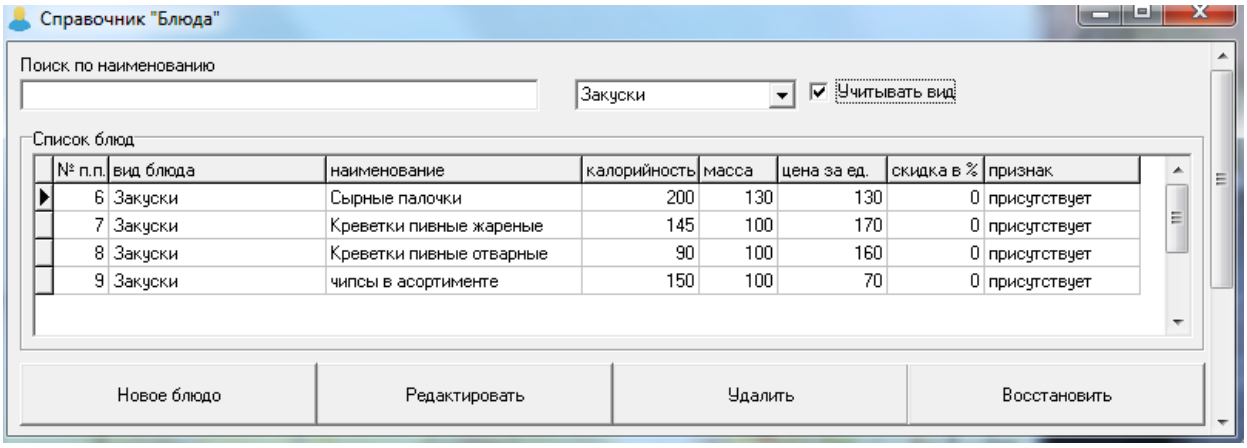


Рисунок 38 Справочник «блюда»

Далее разработаем инструкцию для работы пользователя в программе.

2.6 Инструкция пользователя.

Запуск программы.

Для начала работы с АРМ «администратор» запустите ярлык «АРМ администратор» расположенный на рабочем столе. Для того чтобы выполнить авторизацию пользователя необходимо выбрать «Меню – выполнить вход». Далее необходимо ввести учетные данные пользователя, если данные верны – то Вам будут доступны разделы: работа (все пользователи), справочники (администратор), отчеты (администратор) .

Добавление заказа.

Для работы с заказами необходимо выбрать «работа - заказы». Для добавления нового клиента необходимо нажать кнопку «новый клиент». Далее заполните форму и нажмите кнопку «сохранить». Кнопки «редактировать» и «удалить» доступны только пользователям с набором прав «администратор». Для добавления нового заказа необходимо из таблицы «список клиентов» выбрать посетителя. Для быстрого поиска клиента воспользуйтесь функцией «поиск по фамилии». После выбора посетителя в нижней части формы на вкладке «новый заказ» заполните строку «персонал» из выпадающего списка официантов. Строка «персонал» с выпадающим списком доступна только пользователям с набором прав «администратор», для пользователей с набором прав «официант» строка «персонал» заполняется автоматически. Так же необходимо заполнить строку «выбор столика», выбрав свободный столик из выпадающего списка.

Для сохранения нового заказа нажмите кнопку «сохранить».

Добавление блюд в заказ.

Для того чтобы добавить блюда в заказ нажмите кнопку «список блюд». На открывшейся форме расположена таблица «список блюд выбранного заказа» в которой представлены блюда, выбранные нами из справочника. При нажатии на кнопку «добавить новое блюдо» необходимо из справочника блюд выбрать позицию, воспользовавшись быстрым поиском по наименованию.

После выбора необходимого блюда заполните поля «количество порций» и «сумма». Поле «сумма» считается автоматически, учитывая количество порций из поля «количество порций» и поля «цена за ед.», а также скидки из поля «скидка». Для сохранения используйте кнопку «сохранить».

Редактирование/удаление блюд в заказе.

Для редактирования блюда необходимо выбрать его в таблице «список блюд выбранного заказа» и нажать кнопку «редактировать». При редактировании доступны только поля «количество порций» и «сумма». После внесения изменений сохраните результат, нажав на кнопку «сохранить». Для удаления позиции из таблицы «список блюд выбранного заказа» необходимо выбрать блюдо и нажать кнопку «удалить блюдо».

Предварительный просмотр/печать чека.

Для предварительного просмотра стоимости заказа необходимо выбрать в таблице «список заказов» позицию и нажать кнопку «предварительный просмотр чека». Для печати чека необходимо закрыть заказ, нажав на кнопку «закрыть заказ» и заполнив дату и время. Кнопка «ОК» подтверждает закрытие заказа. После выполнения данной операции распечатайте чек, нажав кнопку «печать чека». Редактирование закрытого заказа невозможно. Для того чтобы внести изменения в закрытый заказ необходимо нажать кнопку «редактировать состояние», данная функция доступна только для администратора. Обратите внимание, что новые заказы подсвечиваются желтым цветом.

Работа с отчетами.

Для работы с отчетами перейдите в раздел «отчеты» и выбирете из списка отчетов необходимый отчет.

Все отчеты кроме отчета «Открытые заказы» формируются после указания временного промежутка. Отчет «Открытые заказы» формируется за весь период работы программы. Часть отчетов позволяет детализировать данные по конкретной позиции, для этого устанавливается «отметка» о выборе конкретной позиции и из выпадающего списка выбирается нужная позиция.

Для формирования отчета используется кнопка «сформировать отчет». Для экспорта отчета используется кнопка «экспорт в excel».Для печати отчета следует нажать кнопку «печать отчета». Раздел «отчеты» доступен только пользователям с набором прав «администратор» .

Работа со справочниками.

Для работы со справочниками перейдите в раздел «справочники» и выберите необходимый справочник.

На каждой форме справочников доступны 3 функциональные клавиши для добавления, редактирования и удаления записи из справочника. Чтобы добавить новую запись в справочник необходимо нажать кнопку «добавить» и заполнив поля на предложенной форме, нажать кнопку «сохранить». Для того чтобы внести изменения в запись справочника выберите необходимую строку в таблице и нажмите кнопку «редактировать» в предложенной форме внесите необходимые изменения и нажмите «сохранить». Для удаления записи из справочника выделите необходимую строку в таблице и нажмите «удалить». На каждой из форм справочников реализована функция быстрого поиска, чтобы выполнить поиск в поле быстрого поиска введите первые символы нужного наименования.

Смена пользователя/выход из программы.

Для смены пользователя нажмите «меню - отключиться от базы данных».

Для выхода из программы воспользуйтесь кнопкой «меню – выход из программы».

ЗАКЛЮЧЕНИЕ

Целью данной дипломной работы ставилась разработка автоматизированного рабочего места администратора кафе на примере кафе «Аква» п. Лермонтово.

Для достижения поставленной цели дипломной работы были решены следующие задачи:

- проведено исследование деятельности рассматриваемого предприятия, описаны его основные бизнес-процессы, обоснована необходимость разработки АРМ;
- дан краткий анализ существующих систем и выбрана стратегия автоматизации предприятия;
- спроектирована информационная база программного решения;
- реализовано программное решение и приведен контрольный пример;
- разработана инструкция пользователя.

Объектом исследования выступает предприятие кафе при гостинице «Аква».

Администратор кафе занимается сферой обслуживания клиентов и задействован в следующих бизнес-процессах:

1. Регистрация клиента
2. Создание заказа
3. Формирование справочников
4. Построение отчетов
5. Обслуживание клиентов

С помощью АРМ администратора кафе было предложено автоматизировать следующие функции:

- регистрация посетителя с указанием первичной информации;
- создание заказа с непосредственным бронированием столика;
- заполнение справочников клиентов, блюд, столиков, организаций, персонала;

- построение отчетной документации.

Реализация функции «бронирование столиков» предусматривает:

- данные о всех столиках;

- возможность выбора столика для бронирования с указанием даты и времени прихода гостей;

- возможность внесения пожеланий и предпочтений клиента (например, зал для некурящих, бронирование 4-х местного столика на пятерых, то есть чтобы был принесен дополнительный стул, сервировка на пять человек, заказ определенного блюда на число гостей (чтобы были заготовлены необходимые продукты) и т.д.

Для реализации программного решения было предложено использовать среду программирования Delphi и базу данных СУБД MySQL.

АРМ администратора кафе позволит совершенствовать бизнес-процесс приема и обслуживания клиентов, что повысит конкурентоспособность и эффективность данного предприятия.

В перспективе данное программное решение, полученное по итогам работы, будет внедрено на конкретном предприятии, и позволит автоматизировать рабочее место администратора кафе.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. ГОСТ 34.601-90 «Автоматизированные системы. Стадии создания»
2. ГОСТ 34.698-90 «Автоматизированные системы. Требования к содержанию документов»
3. ГОСТ Р ИСО/МЭК 12207 – 99. «Информационная технология. Процессы жизненного цикла программных средств».
4. ГОСТ Р ИСО/МЭК 15910 – 2002. «Информационная технология. Процесс создания документации пользователя программных средств».
5. ГОСТ 19.701-90 «Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения».
6. ГОСТ Р ИСО/МЭК ТО 9294 «Руководство по управлению документированием программного обеспечения».
7. Аакер, Д. Стратегическое рыночное управление. – 7-е изд., пер. с англ./ под ред. С.Г. Божук/ Д. Аакер. – СПб.: Питер, 2007.
8. Автоматизированные информационные технологии в экономике. Учебник. Под ред. Г.А.Титоренко. М.: ЮНИТИ. 2002 г.
9. Аникин, Б.А. Высший менеджмент для руководителя: уч. пособие/ Б.А. Аникин. – М.: ИНФРА-М, 2000.
10. Балдин, К.В. Информационные системы в экономике/ К.В. Балдин, В.Б. Уткин. – М.: ИТК «Дашков и Ко», 2004.
11. Бородакий, Ю.В. Информационные технологии. Методы, процессы, системы/ Ю.В. Бородакий, Ю.Г. Лободинский. – М.: Радио и связь, 2002.
12. Васильев, Г.А. Управленческое консультирование: учеб. пособие/ Г.А. Васильев, Е.М. Деева. – М.: ЮНИТИ-ДАНА, 2004.
13. Веревченко, А.П. Информационные ресурсы для принятия решений: уч. пособие/ А.П. Веревченко, В.В. Горчаков, И.В. Иванов, О.В. Голодова. – М.: Академический Проект; Екатеринбург: Деловая книга, 2002. – 560 с.
14. Годин, В.В. Информационное обеспечение управленческой деятельности: учебник/ В.В. Годин, И.К. Гордеев. – М.: Мастерство: Высшая школа, 2001.

15. Гущина, И.Э. Управленческий учет: основы теории и практики: уч. пособие/ И.Э. Гущина, Н.М. Балакирева. – М.: КНОРУС, 2004.
16. Дейт К., "Введение в системы баз данных", М.: , «Наука», 2006 г.
17. Джеффри Д. Ульман, Дженнифер Уидом. Основы реляционных баз данных. М.: «Лори», 2007 г.
18. Избачков Ю. , Петров В. Информационные системы 2-е издание ПИТЕР, 2005г.
19. Национальный открытый университет - [Электронный ресурс] URL: http://www.intuit.ru/ (дата обращения: 14.03.2016)
20. "Бух.1С" - информация для бухгалтеров [Электронный ресурс] http://www.1c.ru/ (дата обращения: 17.03.2016).
21. Интернет дизайнер схемы БД [Электронный ресурс] http://www.dbdsgnr.appspot.com/ (дата обращения: 04.03.2016).
22. Компания “Мастер-Софт” [Электронный ресурс] http://www.msoft-catalog.ru/partner1c.php (дата обращения: 07.03.2016).

Приложение А

Листинг программы

```
unit Menu;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, Buttons, jpeg, ExtCtrls;

type
  TForm_MENU = class(TForm)
    MainMenu1: TMainMenu;
    N1: TMenuItem;
    N2: TMenuItem;
    N3: TMenuItem;
    N4: TMenuItem;
    N5: TMenuItem;
    N6: TMenuItem;
    N7: TMenuItem;
    N8: TMenuItem;
    N9: TMenuItem;
    N11: TMenuItem;
    N12: TMenuItem;
    N14: TMenuItem;
    Edit1: TEdit;
    N10: TMenuItem;
    Image1: TImage;
    N13: TMenuItem;
    N15: TMenuItem;
    N16: TMenuItem;
    N17: TMenuItem;
    Edit2: TEdit;
    N18: TMenuItem;
    N19: TMenuItem;
    N20: TMenuItem;
    procedure N6Click(Sender: TObject);
    procedure N12Click(Sender: TObject);
    procedure N14Click(Sender: TObject);
    procedure N9Click(Sender: TObject);
    procedure N2Click(Sender: TObject);
    procedure N7Click(Sender: TObject);
    procedure N11Click(Sender: TObject);
    procedure N8Click(Sender: TObject);
    procedure N3Click(Sender: TObject);
    procedure N13Click(Sender: TObject);
    procedure N15Click(Sender: TObject);
    procedure N16Click(Sender: TObject);
```

```
    procedure N17Click(Sender: TObject);
    procedure N18Click(Sender: TObject);
    procedure N20Click(Sender: TObject);



  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_MENU: TForm_MENU;

implementation

uses Glav, Data_module, Org_url, Klient_zakaz, PASS, SPR_CLIENT, SPR_BL,
  SPR_PERS, SPR_OBJECT, INF_ZAK, REPORT1, REPORT1_usl, REPORT2,
  REPORT2_usl, REPORT3_usl, REPORT4_usl, REPORT5_usl, SPRAVKA;

{$R *.dfm}

procedure TForm_MENU.N6Click(Sender: TObject);
begin
Form_SPR_ORG_URL.Show;
end;

procedure TForm_MENU.N12Click(Sender: TObject);
begin
Form_KL_ZAK.Show;
if  Form_KL_ZAK.Label11.Caption='admin' then
begin
Form_KL_ZAK.DBLookupComboBox_PERSONA.Enabled:=true;
Form_KL_ZAK.BitBtn4.Enabled:=true;
end else
Form_KL_ZAK.DBLookupComboBox_PERSONA.KeyValue:=Form_KL_ZAK.Label8.Caption;
end;

procedure TForm_MENU.N14Click(Sender: TObject);
begin
Form_MENU.close;
end;

procedure TForm_MENU.N9Click(Sender: TObject);
begin
Form_SPR_CLIENT.Show;
end;

procedure TForm_MENU.N2Click(Sender: TObject);
begin
Form_PASS.Show;
```

```
end;



procedure TForm_MENU.N7Click(Sender: TObject);
begin
Form_SPR_BL.Show;
end;

procedure TForm_MENU.N11Click(Sender: TObject);
begin
Form_PERSONAL.Show;
end;

procedure TForm_MENU.N8Click(Sender: TObject);
begin
Form_OBJECT.Show;
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('select * from SPR_OBJECT , spr_client where
spr_client.id_kl_f=SPR_OBJECT.ID_KL');

Form_DM.ZQuery_SPR_OBJECT.Open;
end;

procedure TForm_MENU.N3Click(Sender: TObject);
begin
form_menu.N8.Enabled:=false;
form_menu.N7.Enabled:=false;
form_menu.N6.Enabled:=false;
form_menu.N9.Enabled:=false;
form_menu.N10.Enabled:=false;
form_menu.N11.Enabled:=false;
form_menu.N5.Enabled:=false;

form_menu.N4.Enabled:=false;
form_menu.N12.Enabled:=false;
Form_PASS.Show;
form_pass.Edit2.Text:='';
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz');
Form_DM.ZQuery_NEW_ZAK.Open;
Form_KL_ZAK.DBLookupComboBox_PERSONA.Enabled:=false;
Form_KL_ZAK.BitBtn4.Enabled:=false;



end;

procedure TForm_MENU.N13Click(Sender: TObject);
```

```
begin
Form_report1usl.show;
end;

procedure TForm_MENU.N15Click(Sender: TObject);
begin
Form_report2usl.Show;
end;

procedure TForm_MENU.N16Click(Sender: TObject);
begin
Form_report3_usl.Show;
Form_report3_usl.DBLookupComboBox_PERSONA.Enabled:=false;
end;

procedure TForm_MENU.N17Click(Sender: TObject);
begin
Form_report4_usl.Show;
end;

procedure TForm_MENU.N18Click(Sender: TObject);
begin
Form_report5_usl.Show;
Form_report5_usl.DBLookupComboBox_ORG.KeyValue:='';
Form_report5_usl.CheckBox1.Checked:=false;
Form_report5_usl.DBLookupComboBox_ORG.Enabled:=false;
end;

procedure TForm_MENU.N20Click(Sender: TObject);
begin
Form_SPRAVKA.Show;
end;
```

unit Data_module;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, ZAbstractRODataset, ZAbstractDataset, ZDataset,
  ZAbstractConnection, ZConnection, StdCtrls, Grids, DBGrids;

type
  TForm_DM = class(TForm)
    ZConnection_cafe: TZConnection;
    ZQuery_ORG_URL: TZQuery;
    DataSource_ORG_URL: TDataSource;
    ZQuery_SPR_CLIENT: TZQuery;
    DataSource_SPR_CLIENT: TDataSource;
    ZQuery_SPR_ORG_URL: TZQuery;
    DataSource_SPR_ORG_URL: TDataSource;
    ZQuery_NEW_ZAK: TZQuery;
```

DataSource_NEW_ZAK: TDataSource;
DataSource_SPR_OBJECT: TDataSource;
ZQuery_SPR_OBJECT: TZQuery;
ZQuery_SPR_BL: TZQuery;
DataSource_SPR_BL: TDataSource;
ZQuery_CHECK: TZQuery;
DataSource_CHECK: TDataSource;
ZQuery_CHEK_BL: TZQuery;
DataSource_CHEK_BL: TDataSource;
ZQuery_PASS: TZQuery;
DataSource_PASS: TDataSource;
ZQuery_CHEK_P: TZQuery;
DataSource_CHEK_P: TDataSource;
ZQuery_WRITE: TZQuery;
DataSource_WRITE: TDataSource;
GroupBox1: TGroupBox;
DataSource_report1: TDataSource;
ZQuery_report1: TZQuery;
ZQuery_report11: TZQuery;
DataSource_report11: TDataSource;
GroupBox2: TGroupBox;
ZQuery_report2: TZQuery;
ZQuery_report22: TZQuery;
DataSource_report2: TDataSource;
DataSource_report22: TDataSource;
GroupBox3: TGroupBox;
ZQuery_report3: TZQuery;
ZQuery_report33: TZQuery;
DataSource_report3: TDataSource;
DataSource_report33: TDataSource;
GroupBox4: TGroupBox;
ZQuery_report4: TZQuery;
DataSource_report4: TDataSource;
GroupBox5: TGroupBox;
DataSource_report5: TDataSource;
ZQuery_report5: TZQuery;
GroupBox6: TGroupBox;
GroupBox7: TGroupBox;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
Label7: TLabel;
GroupBox8: TGroupBox;
Label8: TLabel;
GroupBox9: TGroupBox;
Label9: TLabel;
GroupBox10: TGroupBox;
Label10: TLabel;
GroupBox11: TGroupBox;

```
    Label11: TLabel;
    GroupBox12: TGroupBox;
    Label12: TLabel;
    GroupBox13: TGroupBox;
    Label13: TLabel;
    GroupBox14: TGroupBox;
    Label14: TLabel;
    GroupBox15: TGroupBox;
    Label15: TLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_DM: TForm_DM;

implementation

{$R *.dfm}
```

unit INF_ZAK;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, StdCtrls, Buttons;

type
  TForm_INF_ZAK = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    DBGrid1: TDBGrid;
    GroupBox4: TGroupBox;
    Edit_Name_bl: TEdit;
    Label2: TLabel;
    Label3: TLabel;
    Edit_calor_bl: TEdit;
    Label4: TLabel;
    Edit_massa: TEdit;
    Edit_group_bl: TEdit;
    Label5: TLabel;
    GroupBox5: TGroupBox;
    Label6: TLabel;
    Edit_discont_bl: TEdit;
    Label7: TLabel;
    Edit_cena_bl: TEdit;
    Label8: TLabel;
    Edit_kolvo_chek: TEdit;
    Edit_summa_chek: TEdit;
```

```
    Label9: TLabel;
    BitBtn_add_chek: TBitBtn;
    Button_new_chek: TButton;
    Button_redact_chek: TButton;
    Label10: TLabel;
    BitBtn_Redact: TBitBtn;
    Button1: TButton;
    Edit1: TEdit;
    Edit_N_zak: TEdit;
    Edit_id_bl: TEdit;
    procedure Button_osm_checkClick(Sender: TObject);
    procedure Button_new_chekClick(Sender: TObject);
    procedure Button_redact_chekClick(Sender: TObject);
    procedure BitBtn_add_chekClick(Sender: TObject);
    procedure BitBtn_svernClick(Sender: TObject);
    procedure BitBtn_RedactClick(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Edit_summa_chekClick(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_INF_ZAK: TForm_INF_ZAK;

implementation

uses Data_module, Klient_zakaz, ZAKAZ_Poisk_BL, REPORT_CHEK;

{$R *.dfm}

procedure TForm_INF_ZAK.Button_osm_checkClick(Sender: TObject);
begin
Form_INF_ZAK.Height:=622;
Edit_Name_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('name').AsString;
Edit_calor_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('calories').AsString;
Edit_massa.Text:=DBGrid1.DataSource.DataSet.fieldbyname('massa').AsString;
Edit_group_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('group').AsString;
Edit_discont_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('discont').AsString;
Edit_cena_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('cena').AsString;
Edit_N_zak.Text:=Form_KL_ZAK.DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsStri
ng;
Edit_id_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('id').AsString;
end;
```

```
procedure TForm_INF_ZAK.Button_new_chekClick(Sender: TObject);
begin
Form_POISK_BL.Show;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('select * from spr_bl  WHERE priznak=:PRIZNAK');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('PRIZNAK').Value:='присутствует';
Form_DM.ZQuery_spr_bl.Open;


end;

procedure TForm_INF_ZAK.Button_redact_chekClick(Sender: TObject);
begin
Form_INF_ZAK.Height:=622;
Edit_Name_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('name').AsString;
Edit_calor_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('calories').AsString;
Edit_massa.Text:=DBGrid1.DataSource.DataSet.fieldbyname('massa').AsString;
Edit_group_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('vid').AsString;
Edit_discont_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('discont').AsString;
Edit_cena_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('cena').AsString;
Edit_N_zak.Text:=Form_KL_ZAK.DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsStri
ng;
Edit_kolvo_chek.Text:=DBGrid1.DataSource.DataSet.fieldbyname('kolvo').AsString;
Edit_summa_chek.Text:=DBGrid1.DataSource.DataSet.fieldbyname('summa').AsString;
Edit_N_zak.Text:=Form_KL_ZAK.DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsStri
ng;
Form_INF_ZAK.BitBtn_Redact.Visible:=true;
Edit_discont_bl.Enabled:=false;

end;


procedure TForm_INF_ZAK.BitBtn_add_chekClick(Sender: TObject);
begin

Form_DM.ZQuery_CHECK.Close;
Form_DM.ZQuery_CHECK.SQL.Clear;
Form_DM.ZQuery_CHECK.SQL.Add('INSERT INTO chek (id_zak, id_bl,kolvo,summa)');
Form_DM.ZQuery_CHECK.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4)');
Form_DM.ZQuery_CHECK.Params.ParamByName('Param1').Value:=Form_KL_ZAK.DBGrid_Z
AKAZ.DataSource.DataSet.fieldbyname('id').AsString;
Form_DM.ZQuery_CHECK.Params.ParamByName('Param2').Value:=StrToInt(Edit_id_bl.Text);
Form_DM.ZQuery_CHECK.Params.ParamByName('Param3').Value:=StrToInt(Edit_kolvo_chek.T
ext);
Form_DM.ZQuery_CHECK.Params.ParamByName('Param4').Value:=strtoint(Edit_summa_chek.T
ext);
Form_DM.ZQuery_CHECK.ExecSQL;
Form_DM.ZQuery_CHEK_BL.Close;
Form_DM.ZQuery_CHEK_BL.SQL.Clear;
```

```
Form_DM.ZQuery_CHEK_BL.SQL.Add('SELECT spr_bl.name,spr_bl.calories,
spr_bl.cena,spr_bl.massa,spr_bl.vid, spr_bl.discont, chek.kolvo, chek.summa, chek.id_bl  FROM
chek, spr_bl  WHERE (chek.id_bl=spr_bl.id) and (chek.id_zak=:ID)');
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('ID').Value:=StrToInt(Edit_N_zak.Text);
Form_DM.ZQuery_CHEK_BL.Open;
Form_INF_ZAK.Height:=355;
Form_INF_ZAK.BitBtn_add_chek.Visible:=false;
Edit_Name_bl.Text:='';
Edit_group_bl.Text:='';
Edit_calor_bl.Text:='';
Edit_cena_bl.Text:='';
Edit_massa.Text:='';
Edit_discont_bl.Text:='';
Edit_kolvo_chek.Text:='';
Edit_summa_chek.Text:='';

end;

procedure TForm_INF_ZAK.BitBtn_svernClick(Sender: TObject);
begin
Form_INF_ZAK.Height:=355;
end;

procedure TForm_INF_ZAK.BitBtn_RedactClick(Sender: TObject);
begin
Form_INF_ZAK.Height:=355;

Form_DM.ZQuery_CHEK_BL.Close;
Form_DM.ZQuery_CHEK_BL.SQL.Clear;
Form_DM.ZQuery_CHEK_BL.SQL.Add('UPDATE chek SET chek.kolvo=:KOLVO,
chek.summa=:SUMMA  WHERE id_bl=:ID');
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('ID').Value:=StrToInt(Edit_ID_BL.Text);
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('SUMMA').Value:=StrToInt(Edit_summa_
chek.Text);
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('KOLVO').Value:=StrToInt(Edit_kolvo_ch
ek.Text);
Form_DM.ZQuery_CHEK_BL.ExecSQL;
ShowMessage('редактирование завершено');
Form_DM.ZQuery_CHEK_BL.Close;
Form_DM.ZQuery_CHEK_BL.SQL.Clear;
Form_DM.ZQuery_CHEK_BL.SQL.Add('SELECT spr_bl.name,spr_bl.calories,
spr_bl.cena,spr_bl.massa,spr_bl.vid, spr_bl.discont,  chek.kolvo, chek.summa, chek.id_bl  FROM
chek, spr_bl  WHERE (chek.id_bl=spr_bl.id) and (chek.id_zak=:ID)');
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('ID').Value:=StrToInt(Edit_N_zak.Text);
Form_DM.ZQuery_CHEK_BL.Open;
Form_INF_ZAK.Height:=355;
Form_INF_ZAK.BitBtn_Redact.Visible:=false;

Edit_Name_bl.Text:='';
Edit_group_bl.Text:='';
Edit_calor_bl.Text:='';
Edit_cena_bl.Text:='';
```

```
Edit_massa.Text:='';
Edit_discont_bl.Text:='';
Edit_kolvo_chek.Text:='';
Edit_summa_chek.Text:='';
Edit_ID_BL.Text:='';

end;

procedure TForm_INF_ZAK.FormActivate(Sender: TObject);
begin
Form_INF_ZAK.Height:=355;
end;


procedure TForm_INF_ZAK.Edit_summa_chekClick(Sender: TObject);
var
a,b,c,f:real;
d:real;
begin
if Edit_kolvo_chek.Text= '' then
begin
ShowMessage('Введине количество')
end
else



if Edit_discont_bl.Text = '0' then
        begin
        Edit_summa_chek.Text:=IntToStr(StrToInt(Edit_cena_bl.Text) *
StrToInt(Edit_kolvo_chek.Text))
        end
        else //Иначе
        begin
a:=StrToInt(Edit_cena_bl.Text);
b:=strtoint(Edit_discont_bl.Text);
c:=StrToInt(Edit_kolvo_chek.Text);
f:=(a-(a*b/100))*c;
Edit_summa_chek.Text:=FloatToStr(f);
end;
end;

procedure TForm_INF_ZAK.Button1Click(Sender: TObject);
begin

Form_DM.ZQuery_CHEK_BL.Close;
Form_DM.ZQuery_CHEK_BL.SQL.Clear;
Form_DM.ZQuery_CHEK_BL.SQL.Add('Delete  from chek WHERE (id_bl=:ID) and
(id_zak=:IDI)');
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('ID').Value:=strtoint(Edit_ID_BL.Text);
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('IDI').Value:=strtoint(Edit_N_zak.Text);
Form_DM.ZQuery_CHEK_BL.ExecSQL;
```

```
ShowMessage('Удаление завершено');
Form_DM.ZQuery_CHEK_BL.Close;
Form_DM.ZQuery_CHEK_BL.SQL.Clear;
Form_DM.ZQuery_CHEK_BL.SQL.Add('SELECT chek.id, spr_bl.name,spr_bl.calories,
spr_bl.cena,spr_bl.massa,spr_bl.vid, spr_bl.discont,  chek.kolvo, chek.summa, chek.id_bl  FROM
chek, spr_bl  WHERE (chek.id_bl=spr_bl.id) and (chek.id_zak=:ID)');
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('ID').Value:=StrToInt(Edit_N_zak.text);
Form_DM.ZQuery_CHEK_BL.Open;
end;



procedure TForm_INF_ZAK.Button2Click(Sender: TObject);
begin
Form_report_check.Show;
Form_report_check.frxReport1.ShowReport;
end;

procedure TForm_INF_ZAK.Button3Click(Sender: TObject);
begin
Form_report_check.frxReport1.ShowReport;
Form_report_check.frxReport1.Export(Form_report_check.frxPDFExport1);
end;

procedure TForm_INF_ZAK.Button4Click(Sender: TObject);
begin
Form_report_check.frxReport1.ShowReport;
Form_report_check.frxReport1.Print;
end;

procedure TForm_INF_ZAK.DBGrid1CellClick(Column: TColumn);
begin
Edit_id_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('id_bl').AsString;
Edit_N_zak.Text:=Form_KL_ZAK.DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsStri
ng;
end;
```

unit Klient_zakaz;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Mask, DBCtrls, ComCtrls, Buttons, ExtCtrls, Grids,
  DBGrids;

type
  TForm_KL_ZAK = class(TForm)
    BitBtn_NEW_KL: TBitBtn;
    BitBtn_REDACT_KL: TBitBtn;
    BitBtn_DEL_KL: TBitBtn;
    GroupBox3: TGroupBox;
```

PageControl1: TPageControl;
TabSheet1: TTabSheet;
TabSheet2: TTabSheet;
DBLookupComboBox_SPR_OBJECT: TDBLookupComboBox;
MaskEdit_DATE_N: TMaskEdit;
Edit_zak_dop_inf: TEdit;
BitBtn_NEW_ZAK: TBitBtn;
Label1: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
GroupBox2: TGroupBox;
Label_bithdate: TLabel;
Label_nphone: TLabel;
LabeledEdit_Name: TLabeledEdit;
LabeledEdit_dopinf: TLabeledEdit;
MaskEdit_bithdate: TMaskEdit;
MaskEdit_nphone: TMaskEdit;
BitBtn_save_new_fizl: TBitBtn;
Label6: TLabel;
DBGrid_SPR_CLIENT: TDBGrid;
ComboBox_S_ZAK: TComboBox;
DBGrid_ZAKAZ: TDBGrid;
BitBtn_PERS_ZAK: TBitBtn;
BitBtn_INF_ZAK: TBitBtn;
BitBtn_REDACT_ZAK: TBitBtn;
Button2: TButton;
Edit_poisk: TEdit;
DBLookupComboBox_ORG: TDBLookupComboBox;
Label7: TLabel;
BitBtn1: TBitBtn;
MaskEdit_DATE_O: TMaskEdit;
BitBtn2: TBitBtn;
BitBtn_ZAKAZ_END: TBitBtn;
Edit_N_ZAK: TEdit;
MaskEdit1: TMaskEdit;
DBLookupComboBox_PERSONA: TDBLookupComboBox;
Label2: TLabel;
BitBtn4: TBitBtn;
Edit_Sost: TEdit;
MaskEdit_d_a_v: TMaskEdit;
BitBtn6: TBitBtn;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Edit1: TEdit;
CheckBox1: TCheckBox;
CheckBox2: TCheckBox;
procedure BitBtn_NEW_KLClick(Sender: TObject);

procedure BitBtn_save_new_fizlClick(Sender: TObject);

```
    procedure BitBtn_NEW_ZAKClick(Sender: TObject);

    procedure BitBtn_PERS_ZAKClick(Sender: TObject);
    procedure BitBtn_INF_ZAKClick(Sender: TObject);
    procedure BitBtn_REDACT_ZAKClick(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure DBLookupComboBox_SPR_OBJECTEnter(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Edit_poiskChange(Sender: TObject);
    procedure BitBtn_REDACT_KLClick(Sender: TObject);
    procedure BitBtn_DEL_KLClick(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn_ZAKAZ_ENDClick(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure DBLookupComboBox_PERSONAEnter(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure DBGrid_ZAKAZCellClick(Column: TColumn);
    procedure DBGrid_SPR_CLIENTCellClick(Column: TColumn);
    procedure BitBtn5Click(Sender: TObject);
    procedure DBGrid_ZAKAZDrawColumnCell(Sender: TObject;
      const Rect: TRect; DataCol: Integer; Column: TColumn;
      State: TGridDrawState);
    procedure BitBtn6Click(Sender: TObject);
    procedure DBLookupComboBox_ORGEnter(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);




  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_KL_ZAK: TForm_KL_ZAK;

implementation

uses Data_module, New_redact_fizl, INF_ZAK, REPORT_CHEK, SPR_CLIENT, Menu;

{$R *.dfm}

procedure TForm_KL_ZAK.BitBtn_NEW_KLClick(Sender: TObject);
begin
LabeledEdit_Name.Visible:=true;
LabeledEdit_dopinf.Visible:=true;
```

```
MaskEdit_bithdate.Visible:=true;
Label_bithdate.Visible:=true;
MaskEdit_nphone.Visible:=true;
Label_nphone.Visible:=true;
BitBtn_save_new_fizl.Visible:=true;
DBLookupComboBox_Org.Visible:=true;
Label6.Visible:=true;
CheckBox1.Visible:=true
end;



procedure TForm_KL_ZAK.BitBtn_save_new_fizlClick(Sender: TObject);
begin
if
(LabeledEdit_Name.Text='') or (MaskEdit_nphone.text='') or (MaskEdit_bithdate.Text='')
then
begin
ShowMessage('Заполните обязательные поля!');
exit;

end
else
if DBLookupComboBox_ORG.enabled=true
then begin


Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('INSERT INTO spr_client (name,
dopinf,nphone,bithdate,name_org)');
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4,
:Param5)');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param1').Value:=LabeledEdit_Name.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param2').Value:=LabeledEdit_dopinf.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param3').Value:=MaskEdit_nphone.Te
xt;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param4').Value:=StrToDate(MaskEdit_
bithdate.Text);
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param5').Value:=DBLookupComboBo
x_ORG.KeyValue;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('Добавление завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
DBLookupComboBox_Org.Keyvalue:='';
LabeledEdit_Name.Text:= '';
```

```
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';

LabeledEdit_Name.Visible:=false;
LabeledEdit_dopinf.Visible:=false;
MaskEdit_bithdate.Visible:=false;
Label_bithdate.Visible:=false;
MaskEdit_nphone.Visible:=false;
Label_nphone.Visible:=false;
BitBtn_save_new_fizl.Visible:=false;
DBLookupComboBox_Org.Visible:=false;
Label6.Visible:=false;
CheckBox1.Visible:=false;

end
else
if  DBLookupComboBox_ORG.enabled=false then begin
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('INSERT INTO spr_client (name,
dopinf,nphone,bithdate)');
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4)');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param1').Value:=LabeledEdit_Name.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param2').Value:=LabeledEdit_dopinf.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param3').Value:=MaskEdit_nphone.Te
xt;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param4').Value:=StrToDate(MaskEdit_
bithdate.Text);
//Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param5').Value:=DBLookupComboB
ox_ORG.KeyValue;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('Добавление завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
//DBLookupComboBox_Org.Keyvalue:='';
LabeledEdit_Name.Text:= '';
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';

LabeledEdit_Name.Visible:=false;
LabeledEdit_dopinf.Visible:=false;
MaskEdit_bithdate.Visible:=false;
Label_bithdate.Visible:=false;
MaskEdit_nphone.Visible:=false;
Label_nphone.Visible:=false;
BitBtn_save_new_fizl.Visible:=false;
```

```
DBLookupComboBox_Org.Visible:=false;
Label6.Visible:=false;
CheckBox1.Visible:=false;
end;
end;



procedure TForm_KL_ZAK.BitBtn_NEW_ZAKClick(Sender: TObject);
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('INSERT INTO zakaz (id_kl,date_n,
dop_inf,name_obj,priznak,persona)');
Form_DM.ZQuery_NEW_ZAK.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4,
:Param5, :Param6)');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param1').Value:=DBGrid_SPR_CLIENT.
DataSource.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param2').Value:=StrToDateTime(MaskEd
it_DATE_N.Text);
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param3').Value:=Edit_zak_dop_inf.Text;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param4').Value:=DBLookupComboBox_
SPR_OBJECT.KeyValue;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param5').Value:=ComboBox_S_ZAK.Tex
t;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param6').Value:=DBLookupComboBox_
PERSONA.KeyValue;
Form_DM.ZQuery_NEW_ZAK.ExecSQL;
ShowMessage('Добавление завершено');
PageControl1.ActivePageIndex := 1;

Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('UPDATE spr_object SET date_n_a=:DATEN,
id_kl=:IDK where NAME=:NAME');
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('DATEN').Value:=StrToDateTime(Mas
kEdit_DATE_N.Text);
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('IDK').Value:=DBGrid_SPR_CLIENT.
DataSource.DataSet.fieldbyname('id_kl_f').AsString;;
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('NAME').Value:=DBLookupComboBo
x_SPR_OBJECT.KeyValue;
Form_DM.ZQuery_SPR_OBJECT.ExecSQL;

Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('SELECT  * from spr_object');
Form_DM.ZQuery_SPR_OBJECT.Open;

Edit_zak_dop_inf.Text:='';

if label11.Caption='админ' then begin
DBLookupComboBox_PERSONA.Keyvalue:='';
```

```
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  * from zakaz');
Form_DM.ZQuery_NEW_ZAK.Open;


end else
if label11.Caption='официант' then begin
DBLookupComboBox_PERSONA.Keyvalue:=Label8.Caption;
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  * from zakaz where persona=:PERS ');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Label8.Caption;
Form_DM.ZQuery_NEW_ZAK.Open;

end;
PageControl1.ActivePageIndex := 1;
end;


procedure TForm_KL_ZAK.BitBtn_PERS_ZAKClick(Sender: TObject);
begin
 if (Label11.Caption='admin') and (CheckBox2.Checked=false) then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Open;
end
else
if (Label11.Caption='официант') and (CheckBox2.Checked=false)
then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID and persona=:PERS');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Label8.Caption;
Form_DM.ZQuery_NEW_ZAK.Open;
end
else

if (Label11.Caption='admin')and (CheckBox2.Checked=true) then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
```

```
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID and priznak=:PRI');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PRI').Value:='новый';
Form_DM.ZQuery_NEW_ZAK.Open;
end
else
if (Label11.Caption='официант')and (CheckBox2.Checked=true)
then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID and persona=:PERS and priznak=:PRI');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Label8.Caption;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PRI').Value:='новый';
Form_DM.ZQuery_NEW_ZAK.Open;

end;
end;



procedure TForm_KL_ZAK.BitBtn_INF_ZAKClick(Sender: TObject);
begin
if
Edit_Sost.Text='новый' then
begin
Form_INF_ZAK.Show;

Form_DM.ZQuery_CHEK_BL.Close;
Form_DM.ZQuery_CHEK_BL.SQL.Clear;
Form_DM.ZQuery_CHEK_BL.SQL.Add('SELECT chek.id, chek.id_bl, chek.id_zak,
spr_bl.name,spr_bl.calories, spr_bl.cena,spr_bl.massa,spr_bl.vid, spr_bl.discont,  chek.kolvo,
chek.summa  FROM chek, spr_bl  WHERE (chek.id_bl=spr_bl.id) and (chek.id_zak=:ID)');
Form_DM.ZQuery_CHEK_BL.Params.ParamByName('ID').Value:=Form_KL_ZAK.DBGrid_ZA
KAZ.DataSource.DataSet.fieldbyname('id').AsString;;
Form_DM.ZQuery_CHEK_BL.Open;
end
else
ShowMessage('заказ закрыт');

end;

procedure TForm_KL_ZAK.BitBtn_REDACT_ZAKClick(Sender: TObject);
begin
```

```
if
Edit_Sost.Text='закрыт' then
begin


Form_DM.ZQuery_write.Close;
Form_DM.ZQuery_write.SQL.Clear;
Form_DM.ZQuery_write.SQL.Add('select chek.id_zak, spr_bl.vid, spr_bl.name,
SUM(chek.kolvo), spr_bl.cena, spr_bl.discont, SUM(chek.summa) from chek, spr_bl WHERE
(chek.id_bl=spr_bl.id) and (chek.id_zak=:ID) group by spr_bl.name');
Form_DM.ZQuery_write.Params.ParamByName('ID').Value:=Form_KL_ZAK.DBGrid_ZAKAZ.
DataSource.DataSet.fieldbyname('id').AsString;;
Form_DM.ZQuery_write.Open;
Form_DM.ZQuery_CHEK_P.Close;
Form_DM.ZQuery_CHEK_P.SQL.Clear;
Form_DM.ZQuery_CHEK_P.SQL.Add('SELECT  sum(summa) FROM chek  WHERE
(chek.id_zak=:ID)');
Form_DM.ZQuery_CHEK_P.Params.ParamByName('ID').Value:=Form_KL_ZAK.DBGrid_ZAK
AZ.DataSource.DataSet.fieldbyname('id').AsString;;
Form_DM.ZQuery_CHEK_P.Open;
Form_report_check.frxReport1.ShowReport;
Form_report_check.frxReport1.Print;
end
else
ShowMessage('закройте заказ');
end;

procedure TForm_KL_ZAK.Button2Click(Sender: TObject);
begin
Form_DM.ZQuery_write.Close;
Form_DM.ZQuery_write.SQL.Clear;
Form_DM.ZQuery_write.SQL.Add('select chek.id_zak, spr_bl.vid, spr_bl.name,
SUM(chek.kolvo), spr_bl.cena, spr_bl.discont, SUM(chek.summa) from chek, spr_bl WHERE
(chek.id_bl=spr_bl.id) and (chek.id_zak=:ID) group by spr_bl.name');
Form_DM.ZQuery_write.Params.ParamByName('ID').Value:=Form_KL_ZAK.DBGrid_ZAKAZ.
DataSource.DataSet.fieldbyname('id').AsString;;
Form_DM.ZQuery_write.Open;


Form_DM.ZQuery_CHEK_P.Close;
Form_DM.ZQuery_CHEK_P.SQL.Clear;
Form_DM.ZQuery_CHEK_P.SQL.Add('SELECT  sum(summa) FROM chek  WHERE
(chek.id_zak=:ID)');
Form_DM.ZQuery_CHEK_P.Params.ParamByName('ID').Value:=Form_KL_ZAK.DBGrid_ZAK
AZ.DataSource.DataSet.fieldbyname('id').AsString;;
Form_DM.ZQuery_CHEK_P.Open;
Form_report_check.Show;
Form_report_check.frxReport1.ShowReport;

end;

procedure TForm_KL_ZAK.Button1Click(Sender: TObject);
```

```
begin
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('SELECT  * from spr_object  where (date_o_a is
NULL) or (date_o_a<:DATEO)');
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('DATEO').Value:=StrToDateTime(Mas
kEdit_DATE_N.Text);
Form_DM.ZQuery_SPR_OBJECT.Open;
end;

procedure TForm_KL_ZAK.DBLookupComboBox_SPR_OBJECTEnter(Sender: TObject);
begin
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('SELECT spr_object.name from spr_object where
date_n_a is  null');

Form_DM.ZQuery_SPR_OBJECT.Open;
end;

procedure TForm_KL_ZAK.Button3Click(Sender: TObject);
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('INSERT INTO zakaz (id_kl,date_n, date_o,
dop_inf,name_obj,priznak)');
Form_DM.ZQuery_NEW_ZAK.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4,
:Param5, :Param6)');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param1').Value:=DBGrid_SPR_CLIENT.
DataSource.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param2').Value:=StrToDateTime(MaskEd
it_DATE_N.Text);
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param3').Value:=StrToDateTime(MaskEd
it_DATE_o.Text);
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param4').Value:=Edit_zak_dop_inf.Text;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param5').Value:=DBLookupComboBox_
SPR_OBJECT.KeyValue;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('Param6').Value:=ComboBox_S_ZAK.Tex
t;
Form_DM.ZQuery_NEW_ZAK.ExecSQL;
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, spr_object.name, zakaz.priznak from zakaz, spr_object  where
zakaz.name_obj=spr_object.name  and (zakaz.id_kl=:ID)');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Open;
ShowMessage('Добавление завершено');
end;

procedure TForm_KL_ZAK.Edit_poiskChange(Sender: TObject);
```

```
var
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_N.Text:=DateTimeToStr(td);
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('SELECT * FROM spr_client WHERE (name
LIKE"'+Edit_poisk.Text+'%") and priznak is null');
Form_DM.ZQuery_SPR_CLIENT.Open;
end;

procedure TForm_KL_ZAK.BitBtn_REDACT_KLClick(Sender: TObject);
begin
if Label11.Caption='admin' then
begin
Form_SPR_CLIENT.Show;
Form_SPR_CLIENT.Height:=397;
Form_SPR_CLIENT.LabeledEdit_Name.Visible:=true;
Form_SPR_CLIENT.LabeledEdit_dopinf.Visible:=true;
Form_SPR_CLIENT.MaskEdit_bithdate.Visible:=true;
Form_SPR_CLIENT.Label_bithdate.Visible:=true;
Form_SPR_CLIENT.MaskEdit_nphone.Visible:=true;
Form_SPR_CLIENT.Label_nphone.Visible:=true;
Form_SPR_CLIENT.BitBtn_Redact_fizl.Visible:=true;
Form_SPR_CLIENT.DBLookupComboBox_Org.Visible:=true;
Form_SPR_CLIENT.Label6.Visible:=true;

Form_SPR_CLIENT.LabeledEdit_Name.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldb
yname('name').AsString;
Form_SPR_CLIENT.MaskEdit_bithdate.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldb
yname('bithdate').AsString;
Form_SPR_CLIENT.MaskEdit_nphone.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldby
name('nphone').AsString;
Form_SPR_CLIENT.LabeledEdit_dopinf.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldb
yname('dopinf').AsString;
Form_SPR_CLIENT.DBLookupComboBox_ORG.KeyValue:=DBGrid_SPR_CLIENT.DataSource
.DataSet.fieldbyname('name_org').AsString;
Form_SPR_CLIENT.Edit_id_kl_f.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname
('id_kl_f').AsString;
end
else
ShowMessage('Недостаточно прав');
end;

procedure TForm_KL_ZAK.BitBtn_DEL_KLClick(Sender: TObject);
begin
if Form_Menu.Edit1.Text='admin' then
begin
Form_SPR_CLIENT.Show;
end
else
```

```
begin
ShowMessage('Не достаточно прав для удаления');
end;
end;
procedure TForm_KL_ZAK.BitBtn1Click(Sender: TObject);
begin
Edit_poisk.Text:='';
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client where priznak is null');

Form_DM.ZQuery_SPR_CLIENT.Open;
end;

procedure TForm_KL_ZAK.BitBtn2Click(Sender: TObject);
var
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_O.Text:=DateTimeToStr(td);
MaskEdit_DATE_O.Visible:=true;
BitBtn_ZAKAZ_END.Visible:=true;
MaskEdit1.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('date_n').AsString;
Edit_N_ZAK.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsString;

end;



procedure TForm_KL_ZAK.BitBtn_ZAKAZ_ENDClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_O.text);
b:=StrToDateTime(MaskEdit1.Text);
if a<b then
begin
ShowMessage('Дата окончания некорректна');
end
else
if a>b then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('UPDATE zakaz SET priznak=:PRIZ, date_o=:DATEO
where ID=:IDI');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('IDI').Value:=strtoint(Edit_N_ZAK.Text);
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PRIZ').Value:='закрыт';
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskE
dit_DATE_O.Text);
Form_DM.ZQuery_NEW_ZAK.ExecSQL;
Form_DM.ZQuery_SPR_OBJECT.Close;
```

```
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('UPDATE spr_object SET date_n_a=null, id_kl=null
where date_n_a=:date');
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('date').Value:=StrToDateTime(MaskEdi
t_d_a_v.Text);
Form_DM.ZQuery_SPR_OBJECT.ExecSQL;
ShowMessage('заказ закрыт');
MaskEdit_DATE_O.Visible:=false;
BitBtn_ZAKAZ_END.Visible:=false;

if Label11.Caption='админ' then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Open;
end
else
if Label11.Caption='официант'
then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID and persona=:PERS');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Label8.Caption;
Form_DM.ZQuery_NEW_ZAK.Open;
end
end;
end;


procedure TForm_KL_ZAK.BitBtn3Click(Sender: TObject);
begin
MaskEdit_DATE_O.Visible:=true;
BitBtn_ZAKAZ_END.Visible:=true;
MaskEdit1.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('date_n').AsString;
Edit_N_ZAK.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsString;
end;

procedure TForm_KL_ZAK.DBLookupComboBox_PERSONAEnter(Sender: TObject);
begin
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('SELECT name from operators  where prav=:PRAV');
```

```
Form_DM.ZQuery_PASS.Params.ParamByName('PRAV').Value:='официант';
Form_DM.ZQuery_PASS.Open;
end;

procedure TForm_KL_ZAK.BitBtn4Click(Sender: TObject);
begin
if Edit_N_ZAK.Text='' then begin
ShowMessage('Выберите заказ для редактирования');
end
else
if Edit_N_ZAK.Text<>'' then begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('UPDATE zakaz SET priznak=:PRIZ, date_o=NULL
where ID=:IDI');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('IDI').Value:=StrToInt(Edit_N_ZAK.Text)
;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PRIZ').Value:='новый';
Form_DM.ZQuery_NEW_ZAK.ExecSQL;

Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('UPDATE spr_object SET date_n_a=:DATEN where
name=:NAME');
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('NAME').Value:=edit1.Text;
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('DATEN').Value:=StrToDateTime(Mas
kEdit_d_a_v.Text);
Form_DM.ZQuery_SPR_OBJECT.ExecSQL;

ShowMessage('состояние изменено');

Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz  where
zakaz.id_kl=:ID');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('ID').Value:=DBGrid_SPR_CLIENT.Data
Source.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_NEW_ZAK.Open;
end;
end;

procedure TForm_KL_ZAK.DBGrid_ZAKAZCellClick(Column: TColumn);
begin
Edit_N_ZAK.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('id').AsString;
Edit_Sost.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('priznak').AsString;
MaskEdit_d_a_v.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('date_n').AsString;
edit1.Text:=DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('name_obj').AsString;
end;

procedure TForm_KL_ZAK.DBGrid_SPR_CLIENTCellClick(Column: TColumn);
var
```

```
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_N.Text:=DateTimeToStr(td);

end;

procedure TForm_KL_ZAK.BitBtn5Click(Sender: TObject);
begin

Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('INSERT INTO spr_object (date_n_a)');
Form_DM.ZQuery_NEW_ZAK.SQL.Add('VALUES (:Param2)');
//Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('Param1').Value:='столик № 1';
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('Param2').Value:=StrToDateTime(Mask
Edit_DATE_N.Text);
Form_DM.ZQuery_SPR_OBJECT.Open;
end;

procedure TForm_KL_ZAK.DBGrid_ZAKAZDrawColumnCell(Sender: TObject;
  const Rect: TRect; DataCol: Integer; Column: TColumn;
  State: TGridDrawState);
begin
if (DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('priznak').AsString='новый') then {тут
любое условие}
begin
DBGrid_ZAKAZ.Canvas.Brush.Color := clInfoBk;
DBGrid_ZAKAZ.Canvas.Font.Color := clBlack;
end
else
if (DBGrid_ZAKAZ.DataSource.DataSet.fieldbyname('priznak').AsString='закрыт') then {тут
любое условие}
begin
DBGrid_ZAKAZ.Canvas.Brush.Color := clGradientActiveCaption;
DBGrid_ZAKAZ.Canvas.Font.Color := clBlack;
end;

DBGrid_Zakaz.DefaultDrawColumnCell(Rect,DataCol,Column,State);
end;

procedure TForm_KL_ZAK.BitBtn6Click(Sender: TObject);
begin
if (Label11.Caption='админ') and (CheckBox2.checked=true)  then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz where
priznak=:PRI');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PRI').Value:='новый';
Form_DM.ZQuery_NEW_ZAK.Open;
```

```
end
else
if (Label11.Caption='официант')and  (CheckBox2.checked=true)
then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz where
persona=:PERS and priznak=:PRI');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Label8.Caption;
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PRI').Value:='новый';
Form_DM.ZQuery_NEW_ZAK.Open;
end
else

if (Label11.Caption='админ') and (CheckBox2.checked=false)  then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz');

Form_DM.ZQuery_NEW_ZAK.Open;
end
else
if (Label11.Caption='официант')and  (CheckBox2.checked=false)
then
begin
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz where
persona=:PERS');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Label8.Caption;

Form_DM.ZQuery_NEW_ZAK.Open;
end;
end;

procedure TForm_KL_ZAK.DBLookupComboBox_ORGEnter(Sender: TObject);
begin
Form_DM.ZQuery_ORG_URL.Close;
Form_DM.ZQuery_ORG_URL.SQL.Clear;
Form_DM.ZQuery_ORG_URL.SQL.Add('select * from spr_org_url');
Form_DM.ZQuery_ORG_URL.Open;
end;
procedure TForm_KL_ZAK.CheckBox1Click(Sender: TObject);
begin
if CheckBox1.Checked=true then
begin
DBLookupComboBox_ORG.Enabled:=true;
```

```
end
else
if CheckBox1.Checked=false then
begin
DBLookupComboBox_ORG.Enabled:=false;
end;
end;
```

unit Org_url;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, Mask, StdCtrls, Buttons, DBCtrls, ExtCtrls;

type
  TForm_SPR_ORG_URL = class(TForm)
    Edit_poisk: TEdit;
    Label1: TLabel;
    GroupBox3: TGroupBox;
    DBGrid_SPR_ORG_URL: TDBGrid;
    BitBtn_VORG: TBitBtn;
    BitBtn_DEL_ORG: TBitBtn;
    BitBtn1_Redact_ORG: TBitBtn;
    BitBtn_NEW_ORG: TBitBtn;
    GroupBox2: TGroupBox;
    Label_nphone: TLabel;
    LabeledEdit_Name: TLabeledEdit;
    LabeledEdit_dopinf: TLabeledEdit;
    MaskEdit_nphone: TMaskEdit;
    BitBtn_save_new_org: TBitBtn;
    BitBtn_Redact_org: TBitBtn;
    Edit_id: TEdit;
    Edit_org: TEdit;
    procedure BitBtn_NEW_KLClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure BitBtn1_Redact_ORGClick(Sender: TObject);
    procedure BitBtn_save_new_orgClick(Sender: TObject);
    procedure BitBtn_Redact_orgClick(Sender: TObject);
    procedure Edit_poiskChange(Sender: TObject);
    procedure BitBtn_DEL_ORGClick(Sender: TObject);
    procedure BitBtn_VORGClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_SPR_ORG_URL: TForm_SPR_ORG_URL;
```

```
implementation

uses Data_module, Menu;

{$R *.dfm}

procedure TForm_SPR_ORG_URL.BitBtn_NEW_KLClick(Sender: TObject);
begin
Form_SPR_ORG_URL.Height:=418;
BitBtn_save_new_org.Visible:=true;
BitBtn_Redact_org.Visible:=false;
end;

procedure TForm_SPR_ORG_URL.FormCreate(Sender: TObject);
begin
Form_SPR_ORG_URL.Height:=275;
end;

procedure TForm_SPR_ORG_URL.BitBtn1_Redact_ORGClick(Sender: TObject);
begin
MaskEdit_nphone.Text:=DBGrid_SPR_ORG_URL.DataSource.DataSet.fieldbyname('nphone_org'
).AsString;
LabeledEdit_dopinf.Text:=DBGrid_SPR_ORG_URL.DataSource.DataSet.fieldbyname('dop_inf').
AsString;
LabeledEdit_Name.Text:=DBGrid_SPR_ORG_URL.DataSource.DataSet.fieldbyname('name_org')
.AsString;
Edit_id.Text:=DBGrid_SPR_ORG_URL.DataSource.DataSet.fieldbyname('id').AsString;
Edit_org.Text:=DBGrid_SPR_ORG_URL.DataSource.DataSet.fieldbyname('name_org').AsString;



Form_SPR_ORG_URL.Height:=418;
BitBtn_save_new_org.Visible:=false;
BitBtn_Redact_org.Visible:=true;
end;

procedure TForm_SPR_ORG_URL.BitBtn_save_new_orgClick(Sender: TObject);
begin
if
(LabeledEdit_Name.Text='') or (MaskEdit_nphone.Text='')
then
begin
ShowMessage('Заполните обязательные поля!');
exit;

end
else

Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
```

```
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('INSERT INTO spr_org_url
(name_org,dop_inf,nphone_org)');
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('VALUES (:Param1, :Param2, :Param3)');
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('Param1').Value:=LabeledEdit_Name
.Text;
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('Param2').Value:=LabeledEdit_dopinf
.Text;
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('Param3').Value:=MaskEdit_nphone.
Text;
Form_DM.ZQuery_SPR_ORG_URL.ExecSQL;
ShowMessage('Добавление завершено');
Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('select * from spr_org_url');
Form_DM.ZQuery_SPR_ORG_URL.Open;
LabeledEdit_Name.Text:= '';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';
Form_SPR_ORG_URL.Height:=275;
end;

procedure TForm_SPR_ORG_URL.BitBtn_Redact_orgClick(Sender: TObject);
begin
Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('UPDATE spr_org_url SET
name_org=:NAME,dop_inf=:DOPINF, nphone_org=:NPHONE where ID=:IDI');
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('IDI').Value:=StrToInt(Edit_id.Text);
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('DOPINF').Value:=LabeledEdit_dopi
nf.Text;
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('NPHONE').Value:=MaskEdit_nphon
e.Text;
Form_DM.ZQuery_SPR_ORG_URL.Params.ParamByName('NAME').Value:=LabeledEdit_Name.
Text;
Form_DM.ZQuery_SPR_ORG_URL.ExecSQL;
ShowMessage('Редактирование завершено');
Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('select * from spr_org_url');
Form_DM.ZQuery_SPR_ORG_URL.Open;
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('UPDATE spr_client SET name_org=:NAMEN
where name_org=:NAME');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NAMEN').Value:=LabeledEdit_Name.
Text;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NAME').Value:=Edit_org.Text;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
```

```
LabeledEdit_Name.Text:= '';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';
Form_SPR_ORG_URL.Height:=275;
end;

procedure TForm_SPR_ORG_URL.Edit_poiskChange(Sender: TObject);
begin
Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('SELECT * FROM spr_org_url WHERE name_org
LIKE'''+Edit_poisk.Text+'%''');
Form_DM.ZQuery_SPR_ORG_URL.Open;
end;

procedure TForm_SPR_ORG_URL.BitBtn_DEL_ORGClick(Sender: TObject);
begin
ShowMessage('Удаление невозможно');
end;

procedure TForm_SPR_ORG_URL.BitBtn_VORGClick(Sender: TObject);
begin
ShowMessage('Восстановление не возможно');
end;
```

unit PASS;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, Grids, DBGrids, jpeg, ExtCtrls;

type
  TForm_PASS = class(TForm)
    Edit1: TEdit;
    Edit2: TEdit;
    BitBtn1: TBitBtn;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    Image1: TImage;
    Label2: TLabel;
    procedure BitBtn1Click(Sender: TObject);
    procedure Image2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  Form_PASS: TForm_PASS;

implementation

uses Data_module, Menu, INF_ZAK, Glav, Klient_zakaz;

{$R *.dfm}

procedure TForm_PASS.BitBtn1Click(Sender: TObject);
begin
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Text:='SELECT * FROM operators WHERE login = :p_login
AND pass = :p_passw and priznak is null ';
 if (Trim(Edit2.Text) <> '')
  then
  begin
    Form_DM.ZQuery_PASS.Params.ParamByName('p_login').Value := Edit1.text;
    Form_DM.ZQuery_PASS.Params.ParamByName('p_passw').Value := Edit2.Text;
    try
      Form_DM.ZQuery_PASS.Open;
    Except
      ShowMessage('Не удалось открыть таблицу пользователей');
      Exit;
    end;
    if Form_DM.ZQuery_PASS.RecordCount > 0
    then
    begin
      ShowMessage('Вход успешно выполнен');



      Form_MENU.Show;
      Form_PASS.Close;
      Form_Menu.Edit1.Text:=DBGrid1.DataSource.DataSet.fieldbyname('prav').AsString;
      Form_KL_ZAK.Label11.Caption:=DBGrid1.DataSource.DataSet.fieldbyname('prav').AsString;
      Form_KL_ZAK.Label8.Caption:=DBGrid1.DataSource.DataSet.fieldbyname('name').AsString;
      Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_pass.SQL.Clear;
Form_DM.ZQuery_pass.SQL.Add('select * from operators');
Form_DM.ZQuery_pass.Open;


      if Form_Menu.Edit1.Text='админ' then
begin
form_menu.N8.Enabled:=true;
form_menu.N7.Enabled:=true;
form_menu.N6.Enabled:=true;
form_menu.N9.Enabled:=true;

form_menu.N11.Enabled:=true;
```

```
form_menu.N5.Enabled:=true;
form_menu.N10.Enabled:=true;
form_menu.N4.Enabled:=true;
form_menu.N12.Enabled:=true;
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz');

Form_DM.ZQuery_NEW_ZAK.Open;
end
else
begin

form_menu.N8.Enabled:=false;
form_menu.N7.Enabled:=false;
form_menu.N6.Enabled:=false;
form_menu.N9.Enabled:=false;
form_menu.N11.Enabled:=false;
form_menu.N5.Enabled:=false;
form_menu.N4.Enabled:=true;
form_menu.N12.Enabled:=true;
Form_DM.ZQuery_NEW_ZAK.Close;
Form_DM.ZQuery_NEW_ZAK.SQL.Clear;
Form_DM.ZQuery_NEW_ZAK.SQL.Add('SELECT  zakaz.id_kl, zakaz.id,
zakaz.date_n,zakaz.date_o, zakaz.name_obj,persona, zakaz.priznak from zakaz where
persona=:PERS');
Form_DM.ZQuery_NEW_ZAK.Params.ParamByName('PERS').Value:=Form_KL_ZAK.Label8.C
aption;
Form_DM.ZQuery_NEW_ZAK.Open;


   end;

   end
   else
   begin
     ShowMessage('Неверный логин или пароль.Повторите ввод');
    Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('select *  from operators');
Form_DM.ZQuery_PASS.Open;
     Edit2.Text := '';
   end;
  end
  else
   ShowMessage('Пустые поля логин\пароль не допускаются');
end;

procedure TForm_PASS.Image2Click(Sender: TObject);
begin
Form_DM.ZQuery_PASS.Close;
```

```
Form_DM.ZQuery_PASS.SQL.Text:='SELECT * FROM operators WHERE login = :p_login
AND pass = :p_passw and priznak is null ';
 if (Trim(Edit2.Text) <> '')
  then
  begin
    Form_DM.ZQuery_PASS.Params.ParamByName('p_login').Value := Edit1.text;
    Form_DM.ZQuery_PASS.Params.ParamByName('p_passw').Value := Edit2.Text;
    try
      Form_DM.ZQuery_PASS.Open;
    Except
      ShowMessage('Не удалось открыть таблицу пользователей');
      Exit;
    end;
    if Form_DM.ZQuery_PASS.RecordCount > 0
    then
    begin
      ShowMessage('Вход успешно выполнен');



      Form_MENU.Show;
      Form_PASS.Close;
      Form_Menu.Edit1.Text:=DBGrid1.DataSource.DataSet.fieldbyname('prav').AsString;
      Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_pass.SQL.Clear;
Form_DM.ZQuery_pass.SQL.Add('select * from operators');
Form_DM.ZQuery_pass.Open;

      if Form_Menu.Edit1.Text='админ' then
begin
form_menu.N8.Enabled:=true;
form_menu.N7.Enabled:=true;
form_menu.N6.Enabled:=true;
form_menu.N9.Enabled:=true;

form_menu.N11.Enabled:=true;
form_menu.N5.Enabled:=true;
form_menu.N10.Enabled:=true;
form_menu.N4.Enabled:=true;
form_menu.N12.Enabled:=true;
end
else
begin

form_menu.N8.Enabled:=false;
form_menu.N7.Enabled:=false;
form_menu.N6.Enabled:=false;
form_menu.N9.Enabled:=false;
form_menu.N10.Enabled:=true;
form_menu.N11.Enabled:=false;
form_menu.N5.Enabled:=false;
form_menu.N4.Enabled:=true;
```

```
form_menu.N12.Enabled:=true;
   end;




   end
   else
   begin
     ShowMessage('Неверный логин или пароль.Повторите ввод');
    Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('select *  from operators');
Form_DM.ZQuery_PASS.Open;
     Edit2.Text := '';
   end;
  end
  else
   ShowMessage('Пустые поля логин\пароль не допускаются');
end;
```

unit REPORT_CHEK;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, frxClass, frxPreview, frxDBSet, frxExportPDF, frxExportImage;

type
  TForm_report_check = class(TForm)
    frxPreview1: TfrxPreview;
    frxDBDataset1: TfrxDBDataset;
    frxPDFExport1: TfrxPDFExport;
    frxReport1: TfrxReport;
    frxDBDataset2: TfrxDBDataset;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report_check: TForm_report_check;

implementation

uses INF_ZAK, Data_module;
```

```
{$R *.dfm}
```

unit REPORT1;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, frxClass, frxPreview, frxDBSet, frxExportXLSX;

type
  TForm_report1 = class(TForm)
    frxDBDataset1: TfrxDBDataset;
    frxPreview1: TfrxPreview;
    frxReport1: TfrxReport;
    frxDBDataset2: TfrxDBDataset;
    frxXLSXExport1: TfrxXLSXExport;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report1: TForm_report1;

implementation

uses Data_module;

{$R *.dfm}
```

unit REPORT1_usl;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, Mask, Grids, DBGrids;

type
  TForm_report1usl = class(TForm)
    BitBtn_predp: TBitBtn;
    BitBtn_export_ex: TBitBtn;
    MaskEdit_DATE_N: TMaskEdit;
    Label1: TLabel;
    MaskEdit_DATE_O: TMaskEdit;
    Label2: TLabel;
    BitBtn_P: TBitBtn;
    procedure BitBtn_predpClick(Sender: TObject);
```

```
    procedure Button1Click(Sender: TObject);
    procedure BitBtn_PClick(Sender: TObject);
    procedure BitBtn_export_exClick(Sender: TObject);
    procedure MaskEdit_DATE_OClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report1usl: TForm_report1usl;

implementation

uses REPORT1, Data_module;

{$R *.dfm}

procedure TForm_report1usl.BitBtn_predpClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if b>a then begin


Form_DM.ZQuery_report1.Close;
Form_DM.ZQuery_report1.SQL.Clear;
Form_DM.ZQuery_report1.SQL.Add('SELECT  spr_bl.vid,spr_bl.name, SUM(chek.kolvo),
SUM(chek.summa) FROM chek, spr_bl, zakaz WHERE (chek.id_bl=spr_bl.id) and
(zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)  GROUP BY
spr_bl.name');
Form_DM.ZQuery_report1.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report1.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report1.Open;

Form_DM.ZQuery_report11.Close;
Form_DM.ZQuery_report11.SQL.Clear;
Form_DM.ZQuery_report11.SQL.Add('SELECT SUM(chek.kolvo), SUM(chek.summa) FROM
chek,zakaz WHERE (zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and
(zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report11.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report11.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report11.Open;
```

```
Form_report1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report1.Show;
Form_report1.frxReport1.ShowReport;
end
else
ShowMessage('Дата не коррекtна');
end;

procedure TForm_report1usl.Button1Click(Sender: TObject);
begin
Form_DM.ZQuery_report11.Close;
Form_DM.ZQuery_report11.SQL.Clear;
Form_DM.ZQuery_report11.SQL.Add('SELECT SUM(chek.kolvo), SUM(chek.summa) FROM
chek,zakaz WHERE (zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and
(zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report11.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report11.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report11.Open;
end;

procedure TForm_report1usl.BitBtn_PClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if b>a then begin
Form_DM.ZQuery_report1.Close;
Form_DM.ZQuery_report1.SQL.Clear;
Form_DM.ZQuery_report1.SQL.Add('SELECT  spr_bl.vid,spr_bl.name, SUM(chek.kolvo),
SUM(chek.summa) FROM chek, spr_bl, zakaz WHERE (chek.id_bl=spr_bl.id) and
(zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)  GROUP BY
spr_bl.name');
Form_DM.ZQuery_report1.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report1.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report1.Open;

Form_DM.ZQuery_report11.Close;
Form_DM.ZQuery_report11.SQL.Clear;
Form_DM.ZQuery_report11.SQL.Add('SELECT SUM(chek.kolvo), SUM(chek.summa) FROM
chek,zakaz WHERE (zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and
(zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report11.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report11.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
```

```
Form_DM.ZQuery_report11.Open;


Form_report1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";

Form_report1.frxReport1.ShowReport;
Form_report1.frxReport1.Print;
end
else
ShowMessage('Дата не корректна');
end;

procedure TForm_report1usl.BitBtn_export_exClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if b>a then begin
Form_DM.ZQuery_report1.Close;
Form_DM.ZQuery_report1.SQL.Clear;
Form_DM.ZQuery_report1.SQL.Add('SELECT  spr_bl.vid,spr_bl.name, SUM(chek.kolvo),
SUM(chek.summa) FROM chek, spr_bl, zakaz WHERE (chek.id_bl=spr_bl.id) and
(zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)  GROUP BY
spr_bl.name');
Form_DM.ZQuery_report1.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report1.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report1.Open;

Form_DM.ZQuery_report11.Close;
Form_DM.ZQuery_report11.SQL.Clear;
Form_DM.ZQuery_report11.SQL.Add('SELECT SUM(chek.kolvo), SUM(chek.summa) FROM
chek,zakaz WHERE (zakaz.id=chek.id_zak) and (zakaz.date_n>=:DATEN) and
(zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report11.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report11.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report11.Open;


Form_report1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report1.frxReport1.ShowReport;
Form_report1.frxReport1.Export(Form_report1.frxXLSXExport1);
end
else
ShowMessage('Дата не корректна');
end;
```

```
procedure TForm_report1usl.MaskEdit_DATE_OClick(Sender: TObject);

var
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_O.Text:=DateTimeToStr(td);
end
```

unit REPORT2;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, frxClass, frxExportXLSX, frxDBSet, frxPreview;

type
  TForm_report2 = class(TForm)
    frxReport1: TfrxReport;
    frxPreview1: TfrxPreview;
    frxDBDataset1: TfrxDBDataset;
    frxXLSXExport1: TfrxXLSXExport;
    frxDBDataset2: TfrxDBDataset;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report2: TForm_report2;

implementation

uses Data_module;

{$R *.dfm}
```

unit REPORT2_usl;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, StdCtrls, Buttons, Mask;

type
```

```
  TForm_report2usl = class(TForm)
    MaskEdit_DATE_N: TMaskEdit;
    BitBtn_P: TBitBtn;
    BitBtn_export_ex: TBitBtn;
    BitBtn_predp: TBitBtn;
    Label2: TLabel;
    MaskEdit_DATE_O: TMaskEdit;
    Label1: TLabel;
    procedure BitBtn_predpClick(Sender: TObject);
    procedure BitBtn_export_exClick(Sender: TObject);
    procedure BitBtn_PClick(Sender: TObject);
    procedure MaskEdit_DATE_OClick(Sender: TObject);



  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report2usl: TForm_report2usl;

implementation

uses REPORT2, Data_module;

{$R *.dfm}

procedure TForm_report2usl.BitBtn_predpClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if b>a then begin
Form_DM.ZQuery_report2.Close;
Form_DM.ZQuery_report2.SQL.Clear;
Form_DM.ZQuery_report2.SQL.Add('SELECT COUNT(DISTINCT(zakaz.id)),
SUM(chek.summa), zakaz.persona FROM zakaz, chek WHERE (zakaz.id=chek.id_zak) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)  GROUP BY zakaz.persona');
Form_DM.ZQuery_report2.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report2.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report2.Open;

Form_DM.ZQuery_report22.Close;
Form_DM.ZQuery_report22.SQL.Clear;
```

```
Form_DM.ZQuery_report22.SQL.Add('SELECT COUNT(DISTINCT(zakaz.id)),
SUM(chek.summa), zakaz.persona FROM zakaz, chek WHERE (zakaz.id=chek.id_zak) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report22.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report22.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report22.Open;

Form_report2.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report2.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report2.Show;
Form_report2.frxReport1.ShowReport;
end
else
ShowMessage('дата не корректна');
end;

procedure TForm_report2usl.BitBtn_export_exClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if b>a then begin
Form_DM.ZQuery_report2.Close;
Form_DM.ZQuery_report2.SQL.Clear;
Form_DM.ZQuery_report2.SQL.Add('SELECT COUNT(DISTINCT(zakaz.id)),
SUM(chek.summa), zakaz.persona FROM zakaz, chek WHERE (zakaz.id=chek.id_zak) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)  GROUP BY zakaz.persona');
Form_DM.ZQuery_report2.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report2.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report2.Open;

Form_DM.ZQuery_report22.Close;
Form_DM.ZQuery_report22.SQL.Clear;
Form_DM.ZQuery_report22.SQL.Add('SELECT COUNT(DISTINCT(zakaz.id)),
SUM(chek.summa), zakaz.persona FROM zakaz, chek WHERE (zakaz.id=chek.id_zak) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report22.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report22.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report22.Open;

Form_report2.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report2.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report2.frxReport1.ShowReport;
Form_report2.frxReport1.Export(Form_report2.frxXLSXExport1);
end
```

```
else
ShowMessage('дата не коррektна');
end;

procedure TForm_report2usl.BitBtn_PClick(Sender: TObject);
var
a,b:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if b>a then begin
Form_DM.ZQuery_report2.Close;
Form_DM.ZQuery_report2.SQL.Clear;
Form_DM.ZQuery_report2.SQL.Add('SELECT COUNT(DISTINCT(zakaz.id)),
SUM(chek.summa), zakaz.persona FROM zakaz, chek WHERE (zakaz.id=chek.id_zak) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)  GROUP BY zakaz.persona');
Form_DM.ZQuery_report2.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report2.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report2.Open;

Form_DM.ZQuery_report22.Close;
Form_DM.ZQuery_report22.SQL.Clear;
Form_DM.ZQuery_report22.SQL.Add('SELECT COUNT(DISTINCT(zakaz.id)),
SUM(chek.summa), zakaz.persona FROM zakaz, chek WHERE (zakaz.id=chek.id_zak) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO)');
Form_DM.ZQuery_report22.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report22.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report22.Open;

Form_report2.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report2.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report2.frxReport1.ShowReport;
Form_report2.frxReport1.Print;
end
else
ShowMessage('дата не коррektна');
end;


procedure TForm_report2usl.MaskEdit_DATE_OClick(Sender: TObject);

var
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_O.Text:=DateTimeToStr(td);

end;
```

unit REPORT3;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, frxClass, frxDBSet, frxPreview, frxExportXLSX;

type
  TForm1 = class(TForm)
    frxPreview1: TfrxPreview;
    frxReport1: TfrxReport;
    frxDBDataset1: TfrxDBDataset;
    frxXLSXExport1: TfrxXLSXExport;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

uses Data_module;

{$R *.dfm}
```

unit REPORT3_usl;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, Mask, DBCtrls;

type
  TForm_report3_usl = class(TForm)
    MaskEdit_DATE_O: TMaskEdit;
    Label1: TLabel;
    MaskEdit_DATE_N: TMaskEdit;
    Label2: TLabel;
    BitBtn_predp: TBitBtn;
    BitBtn_export_ex: TBitBtn;
    BitBtn_P: TBitBtn;
    CheckBox1: TCheckBox;
    DBLookupComboBox_PERSONA: TDBLookupComboBox;
    procedure BitBtn_predpClick(Sender: TObject);
    procedure DBLookupComboBox_PERSONAEnter(Sender: TObject);
```

```
    procedure BitBtn_export_exClick(Sender: TObject);
    procedure BitBtn_PClick(Sender: TObject);

    procedure MaskEdit_DATE_OClick(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report3_usl: TForm_report3_usl;

implementation

uses REPORT3, Data_module;

{$R *.dfm}

procedure TForm_report3_usl.BitBtn_predpClick(Sender: TObject);
var
a,b,d:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if (a>b) then begin
ShowMessage('Дата не коррекtна');
end
else
if CheckBox1.Checked=true
then begin
Form_DM.ZQuery_report3.Close;
Form_DM.ZQuery_report3.SQL.Clear;
Form_DM.ZQuery_report3.SQL.Add('SELECT  zakaz.id, zakaz.date_n,zakaz.date_o,
zakaz.name_obj, SUM(chek.summa),zakaz.persona FROM zakaz, chek WHERE
(chek.id_zak=zakaz.id) and (zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) and
(zakaz.persona=:PERS) GROUP BY zakaz.id');
Form_DM.ZQuery_report3.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report3.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report3.Params.ParamByName('PERS').Value:=DBLookupComboBox_PERS
ONA.KeyValue;
Form_DM.ZQuery_report3.Open;

Form1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form1.frxReport1.Variables['PERS']:="" + DBLookupComboBox_PERSONA.KeyValue + "";
Form1.Show;
Form1.frxReport1.ShowReport;
end else
```

```
if CheckBox1.Checked=false
then begin
DBLookupComboBox_PERSONA.KeyValue:='';
Form_DM.ZQuery_report3.Close;
Form_DM.ZQuery_report3.SQL.Clear;
Form_DM.ZQuery_report3.SQL.Add('SELECT  zakaz.id, zakaz.date_n,zakaz.date_o,
zakaz.name_obj, SUM(chek.summa),persona FROM zakaz, chek WHERE (chek.id_zak=zakaz.id)
and (zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) GROUP BY zakaz.id');
Form_DM.ZQuery_report3.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report3.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report3.Open;
Form1.frxReport1.Variables['DATEN']:='' + MaskEdit_DATE_N.Text + '';
Form1.frxReport1.Variables['DATEO']:='' + MaskEdit_DATE_O.Text + '';
Form1.frxReport1.Variables['PERS']:='' + DBLookupComboBox_PERSONA.KeyValue + '';
Form1.Show;
Form1.frxReport1.ShowReport;

end;
end;

procedure TForm_report3_usl.DBLookupComboBox_PERSONAEnter(Sender: TObject);
begin
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('SELECT name from operators  where prav=:PRAV');
Form_DM.ZQuery_PASS.Params.ParamByName('PRAV').Value:='официант';
Form_DM.ZQuery_PASS.Open;
end;

procedure TForm_report3_usl.BitBtn_export_exClick(Sender: TObject);

var
a,b:TDateTime;
begin

a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if a>b then begin
ShowMessage('Дата не корректна');
end
else
if CheckBox1.Checked=true
then begin
Form_DM.ZQuery_report3.Close;
Form_DM.ZQuery_report3.SQL.Clear;
Form_DM.ZQuery_report3.SQL.Add('SELECT persona,  zakaz.id, zakaz.date_n,zakaz.date_o,
zakaz.name_obj, SUM(chek.summa) FROM zakaz, chek WHERE (chek.id_zak=zakaz.id) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) and (zakaz.persona=:PERS) GROUP BY
zakaz.id');
```

```
Form_DM.ZQuery_report3.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report3.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report3.Params.ParamByName('PERS').Value:=DBLookupComboBox_PERS
ONA.KeyValue;
Form_DM.ZQuery_report3.Open;

Form1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form1.frxReport1.Variables['PERS']:="" + DBLookupComboBox_PERSONA.KeyValue + "";
Form1.frxReport1.ShowReport;
Form1.frxReport1.Export(Form1.frxXLSXExport1);
end else
if CheckBox1.Checked=false
then begin
DBLookupComboBox_PERSONA.KeyValue:='';
Form_DM.ZQuery_report3.Close;
Form_DM.ZQuery_report3.SQL.Clear;
Form_DM.ZQuery_report3.SQL.Add('SELECT persona,  zakaz.id, zakaz.date_n,zakaz.date_o,
zakaz.name_obj, SUM(chek.summa) FROM zakaz, chek WHERE (chek.id_zak=zakaz.id) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) GROUP BY zakaz.id');
Form_DM.ZQuery_report3.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report3.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report3.Open;
Form1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form1.frxReport1.Variables['PERS']:="" + DBLookupComboBox_PERSONA.KeyValue + "";
Form1.frxReport1.ShowReport;
Form1.frxReport1.Export(Form1.frxXLSXExport1);
end;
end;

procedure TForm_report3_usl.BitBtn_PClick(Sender: TObject);

var
a,b:TDateTime;
begin

a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if a>b then begin
ShowMessage('Дата не коррекtна');
end
else
if CheckBox1.Checked=true
then begin
Form_DM.ZQuery_report3.Close;
Form_DM.ZQuery_report3.SQL.Clear;
```

```
Form_DM.ZQuery_report3.SQL.Add('SELECT  persona,zakaz.id, zakaz.date_n,zakaz.date_o,
zakaz.name_obj, SUM(chek.summa) FROM zakaz, chek WHERE (chek.id_zak=zakaz.id) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) and (zakaz.persona=:PERS) GROUP BY
zakaz.id');
Form_DM.ZQuery_report3.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report3.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report3.Params.ParamByName('PERS').Value:=DBLookupComboBox_PERS
ONA.KeyValue;
Form_DM.ZQuery_report3.Open;

Form1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form1.frxReport1.Variables['PERS']:="" + DBLookupComboBox_PERSONA.KeyValue + "";
Form1.frxReport1.ShowReport;
Form1.frxReport1.Print;
end else
if CheckBox1.Checked=false
then begin
DBLookupComboBox_PERSONA.KeyValue:='';
Form_DM.ZQuery_report3.Close;
Form_DM.ZQuery_report3.SQL.Clear;
Form_DM.ZQuery_report3.SQL.Add('SELECT  persona,zakaz.id, zakaz.date_n,zakaz.date_o,
zakaz.name_obj, SUM(chek.summa) FROM zakaz, chek WHERE (chek.id_zak=zakaz.id) and
(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) GROUP BY zakaz.id');
Form_DM.ZQuery_report3.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report3.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report3.Open;
Form1.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form1.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form1.frxReport1.Variables['PERS']:="" + DBLookupComboBox_PERSONA.KeyValue + "";
Form1.frxReport1.ShowReport;
Form1.frxReport1.Export(Form1.frxXLSXExport1);
end;



end;

procedure TForm_report3_usl.MaskEdit_DATE_OClick(Sender: TObject);
var
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_O.Text:=DateTimeToStr(td);
end;

procedure TForm_report3_usl.CheckBox1Click(Sender: TObject);
begin
```

```
if CheckBox1.Checked=false then begin
DBLookupComboBox_PERSONA.KeyValue:='';
DBLookupComboBox_PERSONA.Enabled:=false;


end
else
if CheckBox1.Checked=true then begin
Form_report3_usl.DBLookupComboBox_PERSONA.Enabled:=true;
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('select * from spr_object');
Form_DM.ZQuery_SPR_OBJECT.Open;
end;
end;
```

unit REPORT4;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, frxClass, frxDBSet, frxPreview, frxExportXLSX;

type
  TForm_report4 = class(TForm)
    frxXLSXExport1: TfrxXLSXExport;
    frxPreview1: TfrxPreview;
    frxDBDataset1: TfrxDBDataset;
    frxReport1: TfrxReport;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report4: TForm_report4;

implementation

uses Data_module;

{$R *.dfm}
```

unit REPORT4_usl;

```
interface

uses
```

```
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, Mask;

type
  TForm_report4_usl = class(TForm)
    BitBtn_predp: TBitBtn;
    BitBtn_export_ex: TBitBtn;
    BitBtn_P: TBitBtn;
    procedure BitBtn_predpClick(Sender: TObject);
    procedure BitBtn_export_exClick(Sender: TObject);
    procedure BitBtn_PClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report4_usl: TForm_report4_usl;

implementation

uses Data_module, REPORT4;

{$R *.dfm}

procedure TForm_report4_usl.BitBtn_predpClick(Sender: TObject);
begin
Form_DM.ZQuery_report4.Close;
Form_DM.ZQuery_report4.SQL.Clear;
Form_DM.ZQuery_report4.SQL.Add('select zakaz.name_obj,spr_client.name, zakaz.id,
zakaz.date_n, SUM(chek.summa), zakaz.persona from zakaz, spr_client,chek where date_o is null
and zakaz.id_kl=spr_client.id_kl_f AND id_zak=zakaz.id group by zakaz.persona ');
Form_DM.ZQuery_report4.Open;

Form_report4.Show;
Form_report4.frxReport1.ShowReport;
end;

procedure TForm_report4_usl.BitBtn_export_exClick(Sender: TObject);
begin
Form_DM.ZQuery_report4.Close;
Form_DM.ZQuery_report4.SQL.Clear;
Form_DM.ZQuery_report4.SQL.Add('select zakaz. name_obj,spr_client.name, zakaz.id,
zakaz.date_n, SUM(chek.summa), zakaz.persona from zakaz, spr_client,chek where date_o is null
and zakaz.id_kl=spr_client.id_kl_f AND id_zak=zakaz.id  group by zakaz.persona ');

Form_DM.ZQuery_report4.Open;

Form_report4.frxReport1.ShowReport;
Form_report4.frxReport1.Export(Form_report4.frxXLSXExport1);
```

```
end;

procedure TForm_report4_usl.BitBtn_PClick(Sender: TObject);
begin
Form_DM.ZQuery_report4.Close;
Form_DM.ZQuery_report4.SQL.Clear;
Form_DM.ZQuery_report4.SQL.Add('select zakaz. name_obj,spr_client.name, zakaz.id,
zakaz.date_n, SUM(chek.summa), zakaz.persona from zakaz, spr_client,chek where date_o is null
and zakaz.id_kl=spr_client.id_kl_f AND id_zak=zakaz.id  group by zakaz.persona ');

Form_DM.ZQuery_report4.Open;

Form_report4.frxReport1.ShowReport;
Form_report4.frxReport1.Print;
end;
```

unit REPORT5;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, frxClass, frxExportXLSX, frxDBSet, frxPreview;

type
  TForm_report5 = class(TForm)
    frxPreview1: TfrxPreview;
    frxReport1: TfrxReport;
    frxDBDataset1: TfrxDBDataset;
    frxXLSXExport1: TfrxXLSXExport;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report5: TForm_report5;

implementation

uses Data_module;

{$R *.dfm}
```

unit REPORT5_usl;

```
interface

uses
```

```
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DBCtrls, StdCtrls, Buttons, Mask, Grids, DBGrids;

type
  TForm_report5_usl = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    MaskEdit_DATE_O: TMaskEdit;
    MaskEdit_DATE_N: TMaskEdit;
    BitBtn_predp: TBitBtn;
    BitBtn_export_ex: TBitBtn;
    BitBtn_P: TBitBtn;
    CheckBox1: TCheckBox;
    DBLookupComboBox_ORG: TDBLookupComboBox;
    procedure DBLookupComboBox_ORGEnter(Sender: TObject);
    procedure MaskEdit_DATE_OClick(Sender: TObject);
    procedure BitBtn_predpClick(Sender: TObject);
    procedure BitBtn_export_exClick(Sender: TObject);
    procedure BitBtn_PClick(Sender: TObject);
    procedure DBLookupComboBox_ORGClick(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_report5_usl: TForm_report5_usl;

implementation

uses Data_module, REPORT5;

{$R *.dfm}

procedure TForm_report5_usl.DBLookupComboBox_ORGEnter(Sender: TObject);
begin
Form_DM.ZQuery_ORG_URL.Close;
Form_DM.ZQuery_ORG_URL.SQL.Clear;
Form_DM.ZQuery_ORG_URL.SQL.Add('select * from spr_org_url');
Form_DM.ZQuery_ORG_URL.Open;
end;

procedure TForm_report5_usl.MaskEdit_DATE_OClick(Sender: TObject);
var
td:TDateTime;
begin
td:=Now;
MaskEdit_DATE_O.Text:=DateTimeToStr(td);
end;
```

```
procedure TForm_report5_usl.BitBtn_predpClick(Sender: TObject);
var
a,b,d:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if (a>b) then begin
ShowMessage('Дата не коррекtна');
end
else
if CheckBox1.Checked=true
then begin
Form_DM.ZQuery_report5.Close;
Form_DM.ZQuery_report5.SQL.Clear;
Form_DM.ZQuery_report5.SQL.Add('SELECT SUM(chek.summa), COUNT(zakaz.id),
spr_client.name_org FROM zakaz, chek,spr_client WHERE (zakaz.id_kl=spr_client.id_kl_f) AND
(chek.id_zak=zakaz.id) AND(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) and
(name_org=:ORG) group by name_org');
Form_DM.ZQuery_report5.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report5.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report5.Params.ParamByName('ORG').Value:=DBLookupComboBox_ORG.K
eyValue;
Form_DM.ZQuery_report5.Open;

Form_report5.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report5.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report5.frxReport1.Variables['ORG']:="" + DBLookupComboBox_ORG.KeyValue + "";
Form_report5.Show;
Form_report5.frxReport1.ShowReport;
end else
if CheckBox1.Checked=false
then begin
Form_DM.ZQuery_report5.Close;
Form_DM.ZQuery_report5.SQL.Clear;
Form_DM.ZQuery_report5.SQL.Add('SELECT SUM(chek.summa), COUNT(zakaz.id),
spr_client.name_org FROM zakaz, chek,spr_client WHERE (zakaz.id_kl=spr_client.id_kl_f) AND
(chek.id_zak=zakaz.id) and(name_org is  not null) AND(zakaz.date_n>=:DATEN) and
(zakaz.date_o<=:DATEO) group by name_org');
Form_DM.ZQuery_report5.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report5.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report5.Open;
DBLookupComboBox_ORG.KeyValue:='';
Form_report5.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report5.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report5.frxReport1.Variables['ORG']:="" + DBLookupComboBox_ORG.KeyValue + "";
Form_report5.Show;
Form_report5.frxReport1.ShowReport;
```

```
end;
end;
procedure TForm_report5_usl.BitBtn_export_exClick(Sender: TObject);
var
a,b,d:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if (a>b) then begin
ShowMessage('Дата не коррекtна');
end
else
if CheckBox1.Checked=true
then begin
Form_DM.ZQuery_report5.Close;
Form_DM.ZQuery_report5.SQL.Clear;
Form_DM.ZQuery_report5.SQL.Add('SELECT SUM(chek.summa), COUNT(zakaz.id),
spr_client.name_org FROM zakaz, chek,spr_client WHERE (zakaz.id_kl=spr_client.id_kl_f) AND
(chek.id_zak=zakaz.id) AND(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) and
(name_org=:ORG) group by name_org');
Form_DM.ZQuery_report5.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report5.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report5.Params.ParamByName('ORG').Value:=DBLookupComboBox_ORG.K
eyValue;
Form_DM.ZQuery_report5.Open;

Form_report5.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report5.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report5.frxReport1.Variables['ORG']:="" + DBLookupComboBox_ORG.KeyValue + "";
Form_report5.frxReport1.ShowReport;
Form_report5.frxReport1.Export(Form_report5.frxXLSXExport1);
end else
if CheckBox1.Checked=false
then begin
Form_DM.ZQuery_report5.Close;
Form_DM.ZQuery_report5.SQL.Clear;
Form_DM.ZQuery_report5.SQL.Add('SELECT SUM(chek.summa), COUNT(zakaz.id),
spr_client.name_org FROM zakaz, chek,spr_client WHERE (zakaz.id_kl=spr_client.id_kl_f) AND
(chek.id_zak=zakaz.id) AND(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) group by
name_org');
Form_DM.ZQuery_report5.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report5.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report5.Open;
DBLookupComboBox_ORG.KeyValue:='';
Form_report5.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report5.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report5.frxReport1.Variables['ORG']:="" + DBLookupComboBox_ORG.KeyValue + "";
Form_report5.frxReport1.ShowReport;
```

```
Form_report5.frxReport1.Export(Form_report5.frxXLSXExport1);

end;
end;

procedure TForm_report5_usl.BitBtn_PClick(Sender: TObject);
var
a,b,d:TDateTime;
begin
a:=StrToDateTime(MaskEdit_DATE_N.Text);
b:=StrToDateTime(MaskEdit_DATE_O.Text);
if (a>b) then begin
ShowMessage('Дата не коррекtна');
end
else
if CheckBox1.Checked=true
then begin
Form_DM.ZQuery_report5.Close;
Form_DM.ZQuery_report5.SQL.Clear;
Form_DM.ZQuery_report5.SQL.Add('SELECT SUM(chek.summa), COUNT(zakaz.id),
spr_client.name_org FROM zakaz, chek,spr_client WHERE (zakaz.id_kl=spr_client.id_kl_f) AND
(chek.id_zak=zakaz.id) AND(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) and
(name_org=:ORG) group by name_org');
Form_DM.ZQuery_report5.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report5.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report5.Params.ParamByName('ORG').Value:=DBLookupComboBox_ORG.K
eyValue;
Form_DM.ZQuery_report5.Open;

Form_report5.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
Form_report5.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report5.frxReport1.Variables['ORG']:="" + DBLookupComboBox_ORG.KeyValue + "";
Form_report5.frxReport1.ShowReport;
Form_report5.frxReport1.Print;
end else
if CheckBox1.Checked=false
then begin
Form_DM.ZQuery_report5.Close;
Form_DM.ZQuery_report5.SQL.Clear;
Form_DM.ZQuery_report5.SQL.Add('SELECT SUM(chek.summa), COUNT(zakaz.id),
spr_client.name_org FROM zakaz, chek,spr_client WHERE (zakaz.id_kl=spr_client.id_kl_f) AND
(chek.id_zak=zakaz.id) AND(zakaz.date_n>=:DATEN) and (zakaz.date_o<=:DATEO) group by
name_org');
Form_DM.ZQuery_report5.Params.ParamByName('DATEN').Value:=StrToDateTime(MaskEdit_
DATE_N.Text);
Form_DM.ZQuery_report5.Params.ParamByName('DATEO').Value:=StrToDateTime(MaskEdit_
DATE_O.Text);
Form_DM.ZQuery_report5.Open;
DBLookupComboBox_ORG.KeyValue:='';
Form_report5.frxReport1.Variables['DATEN']:="" + MaskEdit_DATE_N.Text + "";
```

```
Form_report5.frxReport1.Variables['DATEO']:="" + MaskEdit_DATE_O.Text + "";
Form_report5.frxReport1.Variables['ORG']:="" + DBLookupComboBox_ORG.KeyValue + "";
Form_report5.frxReport1.ShowReport;
Form_report5.frxReport1.Print;

end;
end;

procedure TForm_report5_usl.DBLookupComboBox_ORGClick(Sender: TObject);
begin
Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('select * from spr_org_url');
Form_DM.ZQuery_SPR_ORG_URL.Open;
end;

procedure TForm_report5_usl.CheckBox1Click(Sender: TObject);
begin
if CheckBox1.Checked=false then begin
DBLookupComboBox_ORG.KeyValue:='';
DBLookupComboBox_ORG.Enabled:=false;


end
else
if CheckBox1.Checked=true then begin
Form_report5_usl.DBLookupComboBox_ORG.Enabled:=true;
Form_DM.ZQuery_SPR_ORG_URL.Close;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Clear;
Form_DM.ZQuery_SPR_ORG_URL.SQL.Add('select * from spr_org_url');
Form_DM.ZQuery_SPR_ORG_URL.Open;
end;
end;
```

unit SPR_BL;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, StdCtrls, Buttons, ExtCtrls;

type
  TForm_SPR_BL = class(TForm)
    DBGrid_SPR_BL: TDBGrid;
    Edit_poisk: TEdit;
    Label1: TLabel;
    GroupBox1: TGroupBox;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn1: TBitBtn;
```

```
    BitBtn_NEW_BL: TBitBtn;
    GroupBox2: TGroupBox;
    LabeledEdit_Name: TLabeledEdit;
    BitBtn_redact: TBitBtn;
    BitBtn_new: TBitBtn;
    Edit_id_bl: TEdit;
    ComboBox1_Group: TComboBox;
    Label2: TLabel;
    LabeledEdit_Kalor: TLabeledEdit;
    LabeledEdit_Massa: TLabeledEdit;
    LabeledEdit_Cena: TLabeledEdit;
    LabeledEdit_Diskont: TLabeledEdit;
    CheckBoxV: TCheckBox;
    ComboBoxV: TComboBox;
    procedure Edit_poiskChange(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn_newClick(Sender: TObject);
    procedure BitBtn_NEW_BLClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure BitBtn_redactClick(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure CheckBoxVClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_SPR_BL: TForm_SPR_BL;

implementation

uses Data_module;

{$R *.dfm}

procedure TForm_SPR_BL.Edit_poiskChange(Sender: TObject);
begin
if CheckBoxV.Checked=true then
begin
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('SELECT * FROM spr_bl WHERE (name
LIKE"'+Edit_poisk.Text+'%") and vid=:VID');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('VID').Value:=ComboBoxV.Text;
Form_DM.ZQuery_SPR_BL.Open;
end
else
if CheckBoxV.Checked=false then
begin
```

```
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('SELECT * FROM spr_bl WHERE name
LIKE"'+Edit_poisk.Text+'%"');
Form_DM.ZQuery_SPR_BL.Open;
end;

end;

procedure TForm_SPR_BL.BitBtn2Click(Sender: TObject);
begin
Edit_id_bl.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('id').AsString;
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('UPDATE spr_bl SET priznak=:PRIZ where ID=:IDI');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('IDI').Value:=strtoint(Edit_id_bl.Text);
Form_DM.ZQuery_SPR_BL.Params.ParamByName('PRIZ').Value:='отсутствует';
Form_DM.ZQuery_SPR_BL.ExecSQL;
ShowMessage('удаление завершено');
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('select * from spr_bl');
Form_DM.ZQuery_SPR_BL.Open;
end;

procedure TForm_SPR_BL.BitBtn3Click(Sender: TObject);
begin
Edit_id_bl.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('id').AsString;
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('UPDATE spr_bl SET priznak=:PRIZ where ID=:IDI');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('IDI').Value:=strtoint(Edit_id_bl.Text);
Form_DM.ZQuery_SPR_BL.Params.ParamByName('PRIZ').Value:='присутствует';
Form_DM.ZQuery_SPR_BL.ExecSQL;
ShowMessage('восстановление завершено');
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('select * from spr_bl');
Form_DM.ZQuery_SPR_BL.Open;
end;

procedure TForm_SPR_BL.BitBtn_newClick(Sender: TObject);
begin
if
(LabeledEdit_Name.Text='') or (LabeledEdit_Kalor.Text='') or (LabeledEdit_Massa.Text='') or
(LabeledEdit_Cena.Text='') or (ComboBox1_Group.Text='')
then
begin
ShowMessage('Заполните обязательные поля!');
exit;

end
```

```
else
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('INSERT INTO spr_bl (name, calories, cena, discont, vid,
massa, priznak)');
Form_DM.ZQuery_SPR_BL.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4, :Param5,
:Param6, :Param7)');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param1').Value:=LabeledEdit_Name.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param2').Value:=LabeledEdit_Kalor.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param3').Value:=LabeledEdit_Cena.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param4').Value:=LabeledEdit_Diskont.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param5').Value:=ComboBox1_Group.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param6').Value:=LabeledEdit_Massa.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param7').Value:='присутствует';
Form_DM.ZQuery_SPR_BL.ExecSQL;
ShowMessage('Добавление завершено');
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('select * from spr_bl');
Form_DM.ZQuery_SPR_BL.Open;
ComboBox1_Group.text:='';
LabeledEdit_Name.Text:='';
LabeledEdit_Kalor.Text:='';
LabeledEdit_Massa.Text:= '';
LabeledEdit_Cena.Text:= '';
LabeledEdit_Diskont.Text:= '0';
Form_SPR_BL.Height:=293;
end;

procedure TForm_SPR_BL.BitBtn_NEW_BLClick(Sender: TObject);
begin
Form_SPR_BL.Height:=432;
BitBtn_new.Visible:=true;
BitBtn_redact.Visible:=false;
if CheckBoxV.Checked=true then
begin
ComboBox1_Group.Text:=ComboBoxV.Text
end
else
if CheckBoxV.Checked=false then
begin
ComboBox1_Group.Text:='';
end;
end;

procedure TForm_SPR_BL.FormCreate(Sender: TObject);
begin
Form_SPR_BL.Height:=293;
end;

procedure TForm_SPR_BL.BitBtn_redactClick(Sender: TObject);
begin
```

```
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('UPDATE spr_bl SET name=:Param1, calories=:Param2,
cena=:Param3, discont=:Param4, vid=:Param5, massa=:Param6 where ID=:IDI');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('IDI').Value:=Edit_id_bl.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param1').Value:=LabeledEdit_Name.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param2').Value:=LabeledEdit_Kalor.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param3').Value:=LabeledEdit_Cena.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param4').Value:=LabeledEdit_Diskont.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param5').Value:=ComboBox1_Group.Text;
Form_DM.ZQuery_SPR_BL.Params.ParamByName('Param6').Value:=LabeledEdit_Massa.Text;
Form_DM.ZQuery_SPR_BL.ExecSQL;
ShowMessage('Редактирование завершено');
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('select * from spr_bl');
Form_DM.ZQuery_SPR_BL.Open;
BitBtn_redact.Visible:=false;
ComboBox1_Group.text:='';
LabeledEdit_Name.Text:='';
LabeledEdit_Kalor.Text:='';
LabeledEdit_Massa.Text:= '';
LabeledEdit_Cena.Text:= '';
LabeledEdit_Diskont.Text:= '';
Form_SPR_BL.Height:=293;
end;

procedure TForm_SPR_BL.BitBtn1Click(Sender: TObject);
begin
Form_SPR_BL.Height:=432;
BitBtn_redact.Visible:=true;
BitBtn_new.Visible:=false;
Edit_id_bl.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('id').AsString;
LabeledEdit_Name.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('name').AsString;
LabeledEdit_Kalor.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('calories').AsString;
LabeledEdit_Cena.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('cena').AsString;
LabeledEdit_Diskont.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('discont').AsString
;
ComboBox1_Group.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('vid').AsString;;
LabeledEdit_Massa.Text:=DBGrid_SPR_BL.DataSource.DataSet.fieldbyname('massa').AsString;;
end;



procedure TForm_SPR_BL.CheckBoxVClick(Sender: TObject);
begin
if CheckBoxV.Checked=true then
begin
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('SELECT * FROM spr_bl WHERE vid=:VID');
```

```
Form_DM.ZQuery_SPR_BL.Params.ParamByName('VID').Value:=ComboBoxV.Text;
Form_DM.ZQuery_SPR_BL.Open;
end
else
if CheckBoxV.Checked=false then
begin
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('SELECT * FROM spr_bl');
Form_DM.ZQuery_SPR_BL.Open;
end;
end;
```

unit SPR_CLIENT;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, ExtCtrls, Grids, DBGrids, DBCtrls, Mask;

type
  TForm_SPR_CLIENT = class(TForm)
    GroupBox3: TGroupBox;
    DBGrid_SPR_CLIENT: TDBGrid;
    Edit_poisk: TEdit;
    Label1: TLabel;
    GroupBox2: TGroupBox;
    Label_bithdate: TLabel;
    Label_nphone: TLabel;
    Label6: TLabel;
    LabeledEdit_Name: TLabeledEdit;
    LabeledEdit_dopinf: TLabeledEdit;
    MaskEdit_bithdate: TMaskEdit;
    MaskEdit_nphone: TMaskEdit;
    BitBtn_save_new_fizl: TBitBtn;
    DBLookupComboBox_ORG: TDBLookupComboBox;
    BitBtn_NEW_KL: TBitBtn;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn_Redact_fizl: TBitBtn;
    Edit_id_kl_f: TEdit;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    CheckBox1: TCheckBox;

    procedure Edit_poiskChange(Sender: TObject);
    procedure BitBtn_save_new_fizlClick(Sender: TObject);
    procedure BitBtn_NEW_KLClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
```

```
    procedure BitBtn_Redact_fizlClick(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure DBLookupComboBox_ORGEnter(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_SPR_CLIENT: TForm_SPR_CLIENT;

implementation

uses Data_module;

{$R *.dfm}


procedure TForm_SPR_CLIENT.Edit_poiskChange(Sender: TObject);
begin
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('SELECT * FROM spr_client WHERE name
LIKE"'+Edit_poisk.Text+'%"');
Form_DM.ZQuery_SPR_CLIENT.Open;
end;

procedure TForm_SPR_CLIENT.BitBtn_save_new_fizlClick(Sender: TObject);
begin
if
(LabeledEdit_Name.Text='') or (MaskEdit_nphone.Text='') or (MaskEdit_bithdate.Text='')
then
begin
ShowMessage('Заполните обязательные поля!');
exit;

end
else
if DBLookupComboBox_ORG.enabled=true
then begin
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('INSERT INTO spr_client (name,
dopinf,nphone,bithdate,name_org)');
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4,
:Param5)');
```

```
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param1').Value:=LabeledEdit_Name.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param2').Value:=LabeledEdit_dopinf.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param3').Value:=MaskEdit_nphone.Te
xt;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param4').Value:=StrToDate(MaskEdit_
bithdate.Text);
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param5').Value:=DBLookupComboBo
x_ORG.KeyValue;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('Добавление завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;

LabeledEdit_Name.Text:= '';
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';

LabeledEdit_Name.Visible:=false;
LabeledEdit_dopinf.Visible:=false;
MaskEdit_bithdate.Visible:=false;
Label_bithdate.Visible:=false;
MaskEdit_nphone.Visible:=false;
Label_nphone.Visible:=false;
BitBtn_save_new_fizl.Visible:=false;
DBLookupComboBox_Org.Visible:=false;
Label6.Visible:=false;
Form_SPR_CLIENT.Height:=258;
end else

if DBLookupComboBox_ORG.enabled=false
then begin
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('INSERT INTO spr_client (name,
dopinf,nphone,bithdate)');
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4)');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param1').Value:=LabeledEdit_Name.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param2').Value:=LabeledEdit_dopinf.T
ext;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param3').Value:=MaskEdit_nphone.Te
xt;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('Param4').Value:=StrToDate(MaskEdit_
bithdate.Text);

Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('Добавление завершено');
```

```
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;

LabeledEdit_Name.Text:= '';
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';

LabeledEdit_Name.Visible:=false;
LabeledEdit_dopinf.Visible:=false;
MaskEdit_bithdate.Visible:=false;
Label_bithdate.Visible:=false;
MaskEdit_nphone.Visible:=false;
Label_nphone.Visible:=false;
BitBtn_save_new_fizl.Visible:=false;
DBLookupComboBox_Org.Visible:=false;
Label6.Visible:=false;
Form_SPR_CLIENT.Height:=258;
end;
end;
procedure TForm_SPR_CLIENT.BitBtn_NEW_KLClick(Sender: TObject);
begin
Form_SPR_CLIENT.Height:=397;
LabeledEdit_Name.Visible:=true;
LabeledEdit_dopinf.Visible:=true;
MaskEdit_bithdate.Visible:=true;
Label_bithdate.Visible:=true;
MaskEdit_nphone.Visible:=true;
Label_nphone.Visible:=true;
BitBtn_save_new_fizl.Visible:=true;
DBLookupComboBox_Org.Visible:=true;
Label6.Visible:=true;
LabeledEdit_Name.Text:= '';
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';
DBLookupComboBox_Org.Keyvalue:='';
end;

procedure TForm_SPR_CLIENT.FormCreate(Sender: TObject);
begin
Form_SPR_CLIENT.Height:=258;
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
end;

procedure TForm_SPR_CLIENT.BitBtn1Click(Sender: TObject);
begin
```

```
Form_SPR_CLIENT.Height:=397;
LabeledEdit_Name.Visible:=true;
LabeledEdit_dopinf.Visible:=true;
MaskEdit_bithdate.Visible:=true;
Label_bithdate.Visible:=true;
MaskEdit_nphone.Visible:=true;
Label_nphone.Visible:=true;
BitBtn_Redact_fizl.Visible:=true;
DBLookupComboBox_Org.Visible:=true;
Label6.Visible:=true;

LabeledEdit_Name.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('name').AsStri
ng;
MaskEdit_bithdate.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('bithdate').AsSt
ring;
MaskEdit_nphone.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('nphone').AsStr
ing;
LabeledEdit_dopinf.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('dopinf').AsSt
ring;
DBLookupComboBox_ORG.KeyValue:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname
('name_org').AsString;
Edit_id_kl_f.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('id_kl_f').AsString;


end;

procedure TForm_SPR_CLIENT.BitBtn_Redact_fizlClick(Sender: TObject);
begin
if DBLookupComboBox_ORG.enabled=true
then begin
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('UPDATE spr_client SET name=:NAME,
bithdate=:BITHDATE, dopinf=:DOPINF, nphone=:NPHONE, name_org=:NAME_ORG where
ID_KL_F=:IDI');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('IDI').Value:=StrToInt(Edit_id_kl_f.Tex
t);
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NAME').Value:=LabeledEdit_Name.Te
xt;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('BITHDATE').Value:=StrToDate(Mask
Edit_bithdate.Text);
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('DOPINF').Value:=LabeledEdit_dopinf.
Text;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NPHONE').Value:=MaskEdit_nphone.
Text;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NAME_ORG').Value:=DBLookupCom
boBox_ORG.KeyValue;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('Редактирование завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
```

```
Form_DM.ZQuery_SPR_CLIENT.Open;
DBLookupComboBox_Org.Keyvalue:='';
LabeledEdit_Name.Text:= '';
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';

LabeledEdit_Name.Visible:=false;
LabeledEdit_dopinf.Visible:=false;
MaskEdit_bithdate.Visible:=false;
Label_bithdate.Visible:=false;
MaskEdit_nphone.Visible:=false;
Label_nphone.Visible:=false;
BitBtn_Redact_fizl.Visible:=false;
DBLookupComboBox_Org.Visible:=false;
Label6.Visible:=false;
Form_SPR_CLIENT.Height:=258;
end
else
if DBLookupComboBox_ORG.enabled=false
then begin
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('UPDATE spr_client SET name=:NAME,
bithdate=:BITHDATE, dopinf=:DOPINF, nphone=:NPHONE, name_org=NULL where
ID_KL_F=:IDI');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('IDI').Value:=StrToInt(Edit_id_kl_f.Tex
t);
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NAME').Value:=LabeledEdit_Name.Te
xt;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('BITHDATE').Value:=StrToDate(Mask
Edit_bithdate.Text);
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('DOPINF').Value:=LabeledEdit_dopinf.
Text;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NPHONE').Value:=MaskEdit_nphone.
Text;
//Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('NAME_ORG').Value:=DBLookupCo
mboBox_ORG.KeyValue;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('Редактирование завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
DBLookupComboBox_Org.Keyvalue:='';
LabeledEdit_Name.Text:= '';
MaskEdit_bithdate.Text:='';
MaskEdit_nphone.Text:='';
LabeledEdit_dopinf.Text:='';

LabeledEdit_Name.Visible:=false;
LabeledEdit_dopinf.Visible:=false;
```

```
MaskEdit_bithdate.Visible:=false;
Label_bithdate.Visible:=false;
MaskEdit_nphone.Visible:=false;
Label_nphone.Visible:=false;
BitBtn_Redact_fizl.Visible:=false;
DBLookupComboBox_Org.Visible:=false;
Label6.Visible:=false;
Form_SPR_CLIENT.Height:=258;
end;
end;

procedure TForm_SPR_CLIENT.BitBtn2Click(Sender: TObject);
begin
Edit_id_kl_f.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('UPDATE spr_client SET priznak=:PRIZ where
ID_KL_F=:IDI');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('IDI').Value:=strtoint(Edit_id_kl_f.Text)
;
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('PRIZ').Value:='удален';
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('удаление завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client where priznak is null');
Form_DM.ZQuery_SPR_CLIENT.Open;
end;


procedure TForm_SPR_CLIENT.BitBtn3Click(Sender: TObject);
begin
Edit_id_kl_f.Text:=DBGrid_SPR_CLIENT.DataSource.DataSet.fieldbyname('id_kl_f').AsString;
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('UPDATE spr_client SET priznak=NULL where
ID_KL_F=:IDI');
Form_DM.ZQuery_SPR_CLIENT.Params.ParamByName('IDI').Value:=strtoint(Edit_id_kl_f.Text)
;
Form_DM.ZQuery_SPR_CLIENT.ExecSQL;
ShowMessage('восстановление завершено');
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
end;

procedure TForm_SPR_CLIENT.DBLookupComboBox_ORGEnter(Sender: TObject);
begin
Form_DM.ZQuery_ORG_URL.Close;
Form_DM.ZQuery_ORG_URL.SQL.Clear;
Form_DM.ZQuery_ORG_URL.SQL.Add('select * from spr_org_url');
```

```
Form_DM.ZQuery_ORG_URL.Open;
end;

procedure TForm_SPR_CLIENT.BitBtn4Click(Sender: TObject);
begin
Edit_poisk.Text:='';
Form_DM.ZQuery_SPR_CLIENT.Close;
Form_DM.ZQuery_SPR_CLIENT.SQL.Clear;
Form_DM.ZQuery_SPR_CLIENT.SQL.Add('select * from spr_client');
Form_DM.ZQuery_SPR_CLIENT.Open;
end;

procedure TForm_SPR_CLIENT.CheckBox1Click(Sender: TObject);
begin
if CheckBox1.Checked=true then
begin
DBLookupComboBox_ORG.Enabled:=true;
end
else
if CheckBox1.Checked=false then
begin
DBLookupComboBox_ORG.Enabled:=false;
end;
end;
```

unit SPR_OBJECT;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Grids, DBGrids, Mask, DBCtrls;

type
  TForm_OBJECT = class(TForm)
    DBGrid1: TDBGrid;
    GroupBox1: TGroupBox;
    Edit_Name: TEdit;
    Label2: TLabel;
    Label4: TLabel;
    Edit_dopinf: TEdit;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure DBLookupComboBox1Enter(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  Form_OBJECT: TForm_OBJECT;

implementation

uses Data_module, SPR_PERS;

{$R *.dfm}

procedure TForm_OBJECT.Button1Click(Sender: TObject);
begin
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('SELECT  distinct name from spr_object ');
Form_DM.ZQuery_SPR_OBJECT.Open;
end;

procedure TForm_OBJECT.Button3Click(Sender: TObject);
begin

Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('SELECT  * from spr_object ');
Form_DM.ZQuery_SPR_OBJECT.Open;
end;

procedure TForm_OBJECT.Button2Click(Sender: TObject);
begin
if
(Edit_Name.Text='')
then
begin
ShowMessage('Заполните обязательные поля!');
exit;

end
else



Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('INSERT INTO spr_object
(name,date_n_a,dop_inf)');
Form_DM.ZQuery_SPR_OBJECT.sql.Add('VALUES (:Param1, :Param2, :Param3)');
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('Param1').Value:=Edit_Name.Text;
//Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('Param2').Value:=StrToDateTime(Mas
kEdit_DATE_N.Text);
Form_DM.ZQuery_SPR_OBJECT.Params.ParamByName('Param3').Value:=Edit_dopinf.Text;
Form_DM.ZQuery_SPR_OBJECT.ExecSQL;
ShowMessage('Добавление завершено');
```

```
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('select * from SPR_OBJECT , spr_client where
spr_client.id_kl_f=SPR_OBJECT.ID_KL');
Form_DM.ZQuery_SPR_OBJECT.Open;

Edit_Name.Text:= '';
Edit_dopinf.Text:='';

Form_PERSONAL.Height:=270;
end;

procedure TForm_OBJECT.DBLookupComboBox1Enter(Sender: TObject);
begin
Form_DM.ZQuery_SPR_OBJECT.Close;
Form_DM.ZQuery_SPR_OBJECT.SQL.Clear;
Form_DM.ZQuery_SPR_OBJECT.SQL.Add('SELECT  distinct name from spr_object ');
Form_DM.ZQuery_SPR_OBJECT.Open;
end;
```

unit SPR_PERS;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, StdCtrls, Buttons, Mask, ExtCtrls;

type
  TForm_PERSONAL = class(TForm)
    GroupBox2: TGroupBox;
    LabeledEdit_Name: TLabeledEdit;
    BitBtn_save_new_pers: TBitBtn;
    BitBtn_Redact_pers: TBitBtn;
    Edit_id: TEdit;
    BitBtn_NEW_ORG: TBitBtn;
    BitBtn1_Redact_ORG: TBitBtn;
    BitBtn_DEL_ORG: TBitBtn;
    BitBtn_VORG: TBitBtn;
    Edit_poisk: TEdit;
    Label1: TLabel;
    GroupBox3: TGroupBox;
    DBGrid_SPR_PERS: TDBGrid;
    LabeledEdit_login: TLabeledEdit;
    LabeledEdit_pass: TLabeledEdit;
    ComboBox_prav: TComboBox;
    Label2: TLabel;
    procedure Edit_poiskChange(Sender: TObject);
    procedure BitBtn_VORGClick(Sender: TObject);
    procedure BitBtn_DEL_ORGClick(Sender: TObject);
    procedure BitBtn_Redact_persClick(Sender: TObject);
```

```
    procedure BitBtn_save_new_persClick(Sender: TObject);
    procedure BitBtn_NEW_ORGClick(Sender: TObject);
    procedure BitBtn1_Redact_ORGClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_PERSONAL: TForm_PERSONAL;

implementation

uses Data_module;

{$R *.dfm}

procedure TForm_PERSONAL.Edit_poiskChange(Sender: TObject);
begin
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('SELECT * FROM operators WHERE name
LIKE"'+Edit_poisk.Text+'%"');
Form_DM.ZQuery_PASS.Open;
end;

procedure TForm_PERSONAL.BitBtn_VORGClick(Sender: TObject);
begin
Edit_id.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('id').AsString;
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('UPDATE operators SET priznak=NULL where ID=:IDI');
Form_DM.ZQuery_PASS.Params.ParamByName('IDI').Value:=strtoint(Edit_id.Text);
Form_DM.ZQuery_PASS.ExecSQL;
ShowMessage('восстановление завершено');
Form_DM.ZQuery_pass.Close;
Form_DM.ZQuery_pass.SQL.Clear;
Form_DM.ZQuery_pass.SQL.Add('select * from operators');
Form_DM.ZQuery_pass.Open;
end;

procedure TForm_PERSONAL.BitBtn_DEL_ORGClick(Sender: TObject);
begin
Edit_id.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('id').AsString;
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('UPDATE operators SET priznak=:PR where ID=:IDI');
Form_DM.ZQuery_PASS.Params.ParamByName('IDI').Value:=strtoint(Edit_id.Text);
Form_DM.ZQuery_PASS.Params.ParamByName('PR').Value:='запрещен';
Form_DM.ZQuery_PASS.ExecSQL;
```

```
ShowMessage('удаление завершено');
Form_DM.ZQuery_pass.Close;
Form_DM.ZQuery_pass.SQL.Clear;
Form_DM.ZQuery_pass.SQL.Add('select * from operators');
Form_DM.ZQuery_pass.Open;
end;

procedure TForm_PERSONAL.BitBtn_Redact_persClick(Sender: TObject);
begin
Edit_id.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('id').AsString;
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('UPDATE operators SET name=:Param1, login=:Param2,
pass=:Param3, prav=:Param4 where id=:IDI');
Form_DM.ZQuery_PASS.Params.ParamByName('IDI').Value:=StrToInt(Edit_id.Text);
Form_DM.ZQuery_PASS.Params.ParamByName('Param1').Value:=LabeledEdit_Name.Text;
Form_DM.ZQuery_PASS.Params.ParamByName('Param2').Value:=LabeledEdit_login.Text;
Form_DM.ZQuery_PASS.Params.ParamByName('Param3').Value:=LabeledEdit_pass.Text;
Form_DM.ZQuery_PASS.Params.ParamByName('Param4').Value:=ComboBox_prav.Text;

Form_DM.ZQuery_PASS.ExecSQL;
ShowMessage('Редактирование завершено');
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('select * from operators');
Form_DM.ZQuery_PASS.Open;
ComboBox_prav.Text:='';
LabeledEdit_Name.Text:= '';
LabeledEdit_login.Text:='';
LabeledEdit_pass.Text:='';




Form_PERSONAL.Height:=270;
end;

procedure TForm_PERSONAL.BitBtn_save_new_persClick(Sender: TObject);
begin
if
(LabeledEdit_Name.Text='') or (LabeledEdit_login.Text='') or (LabeledEdit_pass.Text='') or
(ComboBox_prav.Text='')
then
begin
ShowMessage('Заполните обязательные поля!');
exit;

end
else
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('INSERT INTO operators (name, login,pass,prav)');
```

```
Form_DM.ZQuery_PASS.SQL.Add('VALUES (:Param1, :Param2, :Param3, :Param4)');
Form_DM.ZQuery_PASS.Params.ParamByName('Param1').Value:=LabeledEdit_Name.Text;
Form_DM.ZQuery_PASS.Params.ParamByName('Param2').Value:=LabeledEdit_login.Text;
Form_DM.ZQuery_PASS.Params.ParamByName('Param3').Value:=LabeledEdit_pass.Text;
Form_DM.ZQuery_PASS.Params.ParamByName('Param4').Value:=ComboBox_prav.Text;

Form_DM.ZQuery_PASS.ExecSQL;
ShowMessage('Добавление завершено');
Form_DM.ZQuery_PASS.Close;
Form_DM.ZQuery_PASS.SQL.Clear;
Form_DM.ZQuery_PASS.SQL.Add('select * from operators');
Form_DM.ZQuery_PASS.Open;
ComboBox_prav.Text:='';
LabeledEdit_Name.Text:= '';
LabeledEdit_login.Text:='';
LabeledEdit_pass.Text:='';

Form_PERSONAL.Height:=270;

end;

procedure TForm_PERSONAL.BitBtn_NEW_ORGClick(Sender: TObject);
begin
Form_PERSONAL.Height:=406;
BitBtn_save_new_pers.Visible:=true;
BitBtn_Redact_pers.Visible:=false;
LabeledEdit_Name.Visible:=true;
LabeledEdit_pass.Visible:=true;
LabeledEdit_login.Visible:=true;
Label1.Visible:=true;
ComboBox_prav.Visible:=true;
ComboBox_prav.Text:='';
LabeledEdit_Name.Text:= '';
LabeledEdit_login.Text:='';
LabeledEdit_pass.Text:='';
GroupBox2.Visible:=true;
end;

procedure TForm_PERSONAL.BitBtn1_Redact_ORGClick(Sender: TObject);
begin
Form_PERSONAL.Height:=406;
BitBtn_save_new_pers.Visible:=false;
BitBtn_Redact_pers.Visible:=true;
LabeledEdit_Name.Visible:=true;
LabeledEdit_pass.Visible:=true;
LabeledEdit_login.Visible:=true;
Label1.Visible:=true;
ComboBox_prav.Visible:=true;
LabeledEdit_Name.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('name').AsString;
LabeledEdit_login.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('login').AsString;
LabeledEdit_pass.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('pass').AsString;
ComboBox_prav.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('prav').AsString;
```

```
Edit_id.Text:=DBGrid_SPR_PERS.DataSource.DataSet.fieldbyname('id').AsString;
GroupBox2.Visible:=true;


end;

procedure TForm_PERSONAL.FormCreate(Sender: TObject);
begin
Form_PERSONAL.Height:=265;
end;
```

unit SPRAVKA;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, ComCtrls;

type
  TForm_SPRAVKA = class(TForm)
    RichEdit1: TRichEdit;
    BitBtn1: TBitBtn;
    procedure BitBtn1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_SPRAVKA: TForm_SPRAVKA;

implementation

{$R *.dfm}

procedure TForm_SPRAVKA.BitBtn1Click(Sender: TObject);
begin
Form_SPRAVKA.Close;
end;
```

unit ZAKAZ_Poisk_BL;

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, StdCtrls;

type
  TForm_POISK_BL = class(TForm)
```

```
    Edit_poisk: TEdit;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    procedure Edit_poiskChange(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_POISK_BL: TForm_POISK_BL;

implementation

uses INF_ZAK, Data_module, Klient_zakaz;

{$R *.dfm}

procedure TForm_POISK_BL.Edit_poiskChange(Sender: TObject);
begin
Form_DM.ZQuery_SPR_BL.Close;
Form_DM.ZQuery_SPR_BL.SQL.Clear;
Form_DM.ZQuery_SPR_BL.SQL.Add('SELECT * FROM spr_bl WHERE (name
LIKE'"+Edit_poisk.Text+'%") and (priznak=:PRIZNAK)');
Form_DM.ZQuery_SPR_BL.Params.ParamByName('PRIZNAK').Value:='присутствует';
Form_DM.ZQuery_SPR_BL.Open;
end;

procedure TForm_POISK_BL.DBGrid1DblClick(Sender: TObject);
begin
Form_INF_ZAK.Edit_Name_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('name').AsString
;
Form_INF_ZAK.Edit_calor_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('calories').AsStri
ng;
Form_INF_ZAK.Edit_massa.Text:=DBGrid1.DataSource.DataSet.fieldbyname('massa').AsString;
Form_INF_ZAK.Edit_group_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('vid').AsString;
Form_INF_ZAK.Edit_discont_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('discont').AsStr
ing;
Form_INF_ZAK.Edit_cena_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('cena').AsString;
Form_INF_ZAK.Edit_N_zak.Text:=Form_KL_ZAK.DBGrid_ZAKAZ.DataSource.DataSet.fieldby
name('id').AsString;
Form_INF_ZAK.Edit_id_bl.Text:=DBGrid1.DataSource.DataSet.fieldbyname('id').AsString;



Edit_poisk.Text:='';
Form_POISK_BL.Close;
Form_INF_ZAK.Height:=622;
Form_INF_ZAK.Show;
Form_INF_ZAK.BitBtn_add_chek.Visible:=true;
```

```
end;

end.
```