ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ПОДСИСТЕМА УПРАВЛЕНИЯ СПОРТИВНЫМИ ДОСТИЖЕНИЯМИ НА ПРИМЕРЕ ШКОЛЫ №48 ГОРОДА БЕЛГОРОДА

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.03 «Прикладная информатика»
очной формы обучения, группы 07001304
Черняева Ильи Сергеевича

Научный руководитель:
старший преподаватель
Пусная О.П.

БЕЛГОРОД 2017

СОДЕРЖАНИЕ

1 Аналитическая часть 5
1.1 Технико-экономическая характеристика предметной области 5
1.2 Обоснование необходимости и цели использования вычислительной техники для решения задачи 7
1.3 Постановка задачи 12
1.4 Анализ существующих разработок и обоснование выбора технологии проектирования 15
2 Проектная часть 16
2.1 Обоснование проектных решений по техническому обеспечению 16
2.2 Обоснование проектных решений по информационному обеспечению 18
2.3 Обоснование проектных решений по программному обеспечению 21
2.4 Обоснование проектных решений по технологическому обеспечению 28
2.5 Обоснование выбора программных средств 31
2.6 Информационное обеспечение задачи 34
3 Программная реализация проектных решений 36
3.1 Информационная модель и ее описание 36
3.2 Структурная схема подсистемы 39
3.3 Проектирование клиентского приложения 44
3.4 Описание контрольного примера реализации проекта 48
3.5 Оценка экономической эффективности проекта 55
ЗАКЛЮЧЕНИЕ 62
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 64
ПРИЛОЖЕНИЕ А 68
ПРИЛОЖЕНИЕ Б 73

ВВЕДЕНИЕ

В настоящий момент в школах большое внимание уделяется развитию и пропаганде здорового образа жизни, детей с младших классов готовят к спортивным соревнованиям и нормам ГТО.

Ежегодно уровень технической и технологической оснащенности развитых школ растет. Чтобы достичь таких же показателей, необходимо активно развивать инфраструктуру спорта среди молодежи, осуществлять модернизацию систем физического воспитания подрастающего поколения.

В настоящий момент, в школах, отсутствует эффективная система детско-юношеского спорта, подготовки и отбора спортивного резерва для спортивных сборных команд города.

Именно для решения этой проблемы была создана автоматизированная подсистема управления спортивными достижениями.

Целью выпускной квалификационной работы является частичное замещение специалиста путем разработки эффективной подсистемы управления спортивными достижениями в школе.

Объектом данной выпускной квалификационной работы является методическое объединение учителей физической культуры.

Предметом исследования работы является процесс управления спортивными достижениями.

Исходя из поставленной цели, сформулирован комплекс основных задач:

– изучение объекта и выявление недостатков, существующих в учебном учреждении.

– сбор, обработка, хранение и вывод информации, определяющих необходимость создания данной подсистемы;

– построение модели деятельности учреждения, его анализ, выявление сильных и слабых сторон, формирование требований к разрабатываемой подсистеме;

- обоснование выбора основных проектных решений;
- проектирование и разработка информационной подсистемы;
- обоснование социальной эффективности проекта.

Информационной основой при написании данной квалификационной работы явились законодательные и нормативно-правовые акты РФ в области развития физической культуры и информационной безопасности, а также совокупность документации об учебной деятельности объекта исследования в сфере аппаратно-программного обеспечения, топологии локальных сетей и специализированных спортивных показателях. [3]

В процессе разработки практической части были использованы следующие программные средства: AllFusionProcessModeler 7, инструментальная среда ERWin, СУБД Firebird с утилитой IBExpert, Borland C++ Builder.

Достижение цели и задач квалификационной работы по созданию автоматизированной подсистемы управления спортивными достижениями позволит:

- отслеживать набирающие популярность виды спорта в школе;
- увеличить скорость обмена данными и скоординировать процесс взаимодействия школы №48 с Управлением образования г. Белгород;
- повысить взаимодействие учителей с методистами, за счет внедрения специализированного программного продукта.

Выпускная квалификационная работа состоит из трех глав: аналитической части, проектной части, программной реализации проектных решений.

1 Аналитическая часть

1.1 Технико-экономическая характеристика предметной области

1.1.1 Характеристика предприятия

Средняя общеобразовательная школа создана в тысяча девятьсот девяносто пятом году как одна из средних школ города Белгород. Директор школы – Виноградская Марина Викторовна. Основной деятельностью учебного заведения является предоставление среднего учебного образования всем желающим.

Школа осуществляет следующие полномочия:

– проводит официальные региональные физкультурно-оздоровительные мероприятия;

– обеспечивает подготовку кадетского класса;

– обучает вождению на базе автошколы;

– разрабатывает и реализует единый календарный план физкультурно-оздоровительных и спортивно-массовых мероприятий, организует и проводит на территории области областные спортивные соревнования.

Задачи школы в области спорта:

осуществление единой государственной политики в области физической культуры и спорта, направленной на укрепление здоровья и организацию активного отдыха школьников и гостей, формирование у них потребности в физическом совершенствовании и гармоничном развитии личности.

1.1.1 Краткая характеристика подразделения или видов его деятельности

Школа №48сотрудничает со всеми школами и лицеями города Белгород для проведения различных соревнований и спортивных мероприятий. Данная выпускная квалификационная работа описывает взаимодействие учителей физкультуры с методическим объединением физической культуры. Квалификационная работа разработана для упрощения работы учителей физкультуры, получения более высоких результатов и качественных спортивных показателей у школьников.

Автоматизированная подсистема управления спортивными достижениями позволяет грамотно выстраивать тренировочный процесс, получить более слаженное взаимодействие учителей и методистов, отслеживать спортивные результаты и достижения, осуществлять электронный документооборот школы с Управлением образования г. Белгород. Все это в итоге, повышает общий уровень физического развития школьников.

Данная подсистема решает ряд актуальных вопросов. Значительно сократятся издержки:

временные:

- позволит сократить занятость учителей;
- позволит больше уделять внимания непосредственно тренировочному процессу, чем ведению различных журналов;
- формирование отчетов также выполняет система, что способствует сокращению рабочего времени учителей;
- составление и согласование плана с Управлением образования г. Белгород спортивных и физкультурных мероприятий осуществляется быстрее.

Таким образом, можно сделать вывод, что данная подсистема является социально выгодной. Она улучшит результаты школьников, повысит конкурентоспособность белгородских школьников на внешкольных и городских турнирах, а, следовательно, повысит место школы в списке лучших в регионе.

Конечной целью всех этих изменений является вклад физической культуры и спорта в развитие человеческого потенциала России, в укрепление и сохранение, воспитание подрастающего поколения.

1.2 Обоснование необходимости и цели использования вычислительной техники для решения задачи

В настоящий момент уровень развития физической культуры и спорта не соответствует общим положительным социально-экономическим преобразованиям в Российской Федерации.

При этом расходы государства на занятия граждан физической культурой и спортом являются экономически эффективным вложением в улучшение качества жизни граждан России и развитие человеческого потенциала.

В РФ одной из основных проблем является отсутствие эффективной системы детско-юношеского спорта, подготовки и отбора спортивного резерва для спортивных сборных команд страны.

Глобальная конкуренция в спорте в перспективе будет усиливаться, что ставит задачи по разработке высокотехнологических подходов к развитию спорта высших достижений.

В настоящий момент прослеживается значительное отставание Белгородского спорта от ведущих спортивных городов во внедрении и развитии инновационных спортивных технологий.

Это существенно затрудняет развитие физической культуры и массового спорта в школах, подготовку школьников и спортивного резерва в будущем высокого класса, негативно сказывается на конкурентоспособности российского спорта.

Мировые спортивные державы перешли к формированию новой технологической базы развития физической культуры и спорта, основанной на использовании новейших достижений в области теории физического

воспитания и спортивной тренировки, педагогики, психологии, биомеханики и биотехнологий, медицины, информатики, нано технологий и управления.

В Белгородской области прослеживается увлечение большим количеством видов спорта. В школе работают хорошие учителя, есть подающие надежды школьники, созданы достаточно неплохие условия для тренировок, но нет технологической базы, которая помогла бы учителям значительно быстрее корректировать тренировочный процесс на основе результатов и достижений своих школьников, а также получать рекомендации по усовершенствованию спортивного мастерства.

Необходимо автоматизировать управленческий процесс на базе школы, который в дальнейшем можно будет распространить в качестве областного проекта по модернизации физкультурно-оздоровительной деятельности населения. Это поможет учителям обмениваться опытом быстрее и качественнее, без привлечения дополнительных трудовых и человеческих ресурсов, а школьникам думать о тренировках и соревнованиях.

С использованием вычислительной техники повысится конкурентоспособность школьного спорта на городской спортивной арене, это позволит школьникам побеждать на городских турнирах, а в дальнейшем стабильно побеждать на крупнейших российских спортивных соревнованиях. Эти успехи будут достигнуты за счет создания эффективной системы подготовки школьников, а также спортивного резерва с использованием новейших научных достижений и технологий.

В настоящий момент, вся работа по учету и управлению спортивными достижениями в школе выполняется вручную, т.е. без использования электронно-вычислительных машин (ЭВМ). Все сведения об учителях, школьниках, нормативах, тренировках, соревнованиях хранятся только в письменном виде. Это существенно усложняет работу учителю. При потере каких-либо документов, у учителя практически нет возможности их быстро, а, главное, качественно восстановить. Поскольку вычислительная техника не используется, отчеты и различную документацию приходится составлять

самостоятельно и вручную. Это занимает большую часть времени работы учителя. Приходится сверяться с бумагами и вручную составлять отчет и вносить всю информацию о проведенных соревнованиях, что значительно увеличивает и усложняет работу, в то время как в разработанной подсистеме реализована возможность генерирования отчетов.

При анализе деятельности школы №48, были выявлены некоторые задачи в работе сотрудников, которые можно автоматизировать.

Таким образом, вычислительную технику необходимо использовать, для:

- ведения баз данных об учителях и школьниках;
- отслеживания и корректировки спортивных достижений;
- осуществления быстрого обмена данными с Управлением образования г. Белгород;
- формирования электронных отчетов;
- разработки учебно-тренировочных программ по многочисленным видам спорта.

Организация работы, которая выполняется в настоящий момент, будет описана на примере взаимодействия Управления образования г. Белгород со школой №48.

Контекстная диаграмма автоматизированной подсистемы управления спортивными достижениями приведена на рисунке 1.1

Входной информацией подсистемы являются сведения.

Выходной информацией подсистемы являются: информация о школьниках, информация об учителях, информация о мероприятиях.

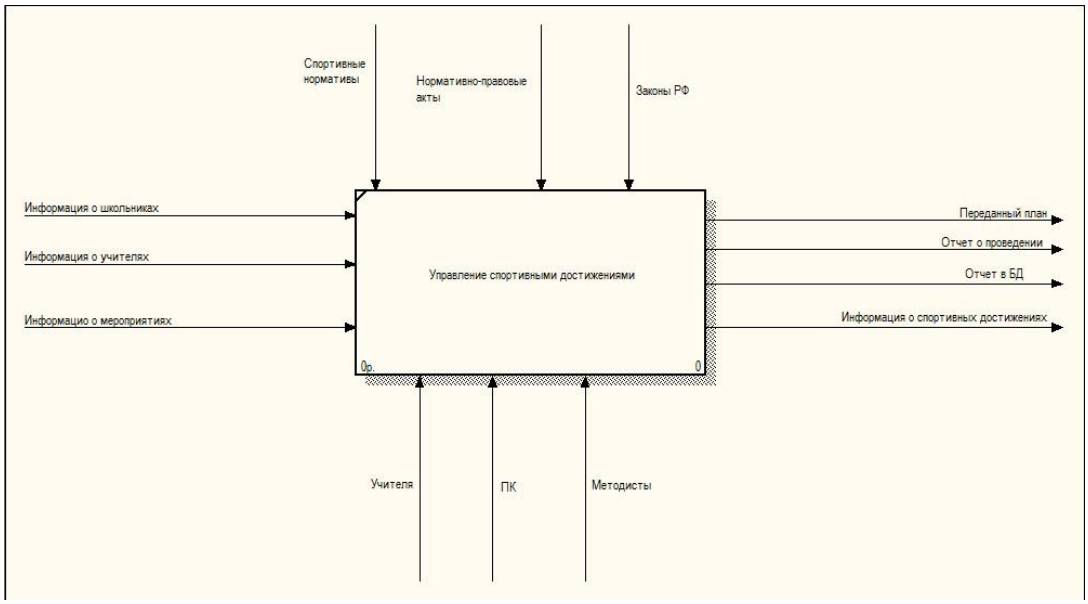


Рисунок 1.1 - Контекстная диаграмма автоматизированной подсистемы управления спортивными достижениями

Детализация контекстной диаграммы представлена на рисунке 1.2 На данном этапе с учетом Устава школы, законов РФ и нормативно-правовых актов учителям необходимо самостоятельно составить план спортивных мероприятий, провести спортивные мероприятия, произвести учет спортивных мероприятий, а также сформировать отчеты о спортивной деятельности.

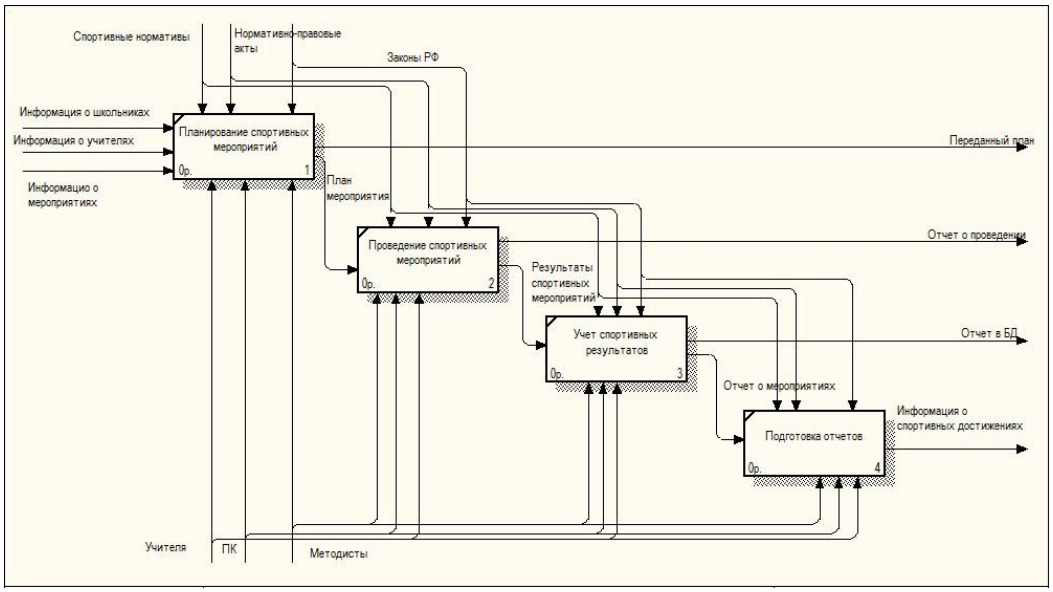


Рисунок 1.2 - Детализация контекстной диаграммы

Детализация блока «Планирование спортивных мероприятий» диаграммы представлена на рисунке 1.3 На данном этапе с учетом существующих правил учителям необходимо самостоятельно составить план мероприятий в школе, согласовать его в Управлении образования г. Белгород, если потребуется, доработать план в школе, и составить отчет о мероприятиях.

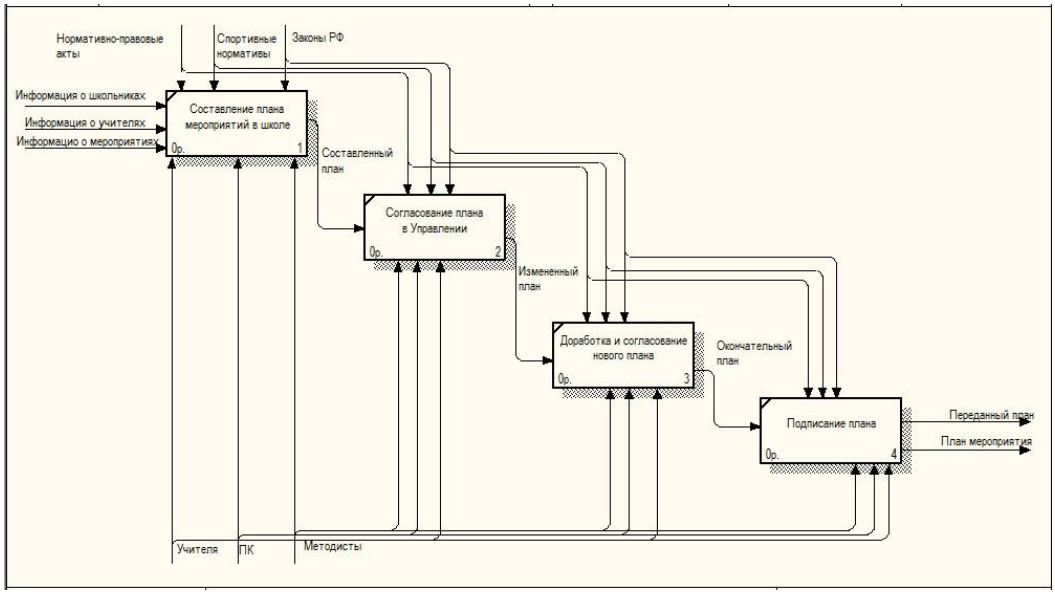


Рисунок 1.3 - Детализация блока «Планирование спортивных мероприятий»

Детализация блока «Проведение спортивных мероприятий» диаграммы представлена на рисунке 1.4 На данном этапе с учетом существующих правил учителям необходимо организовать спортивное мероприятие, провести выполнение спортивных нормативов среди школьников и произвести учет результатов.

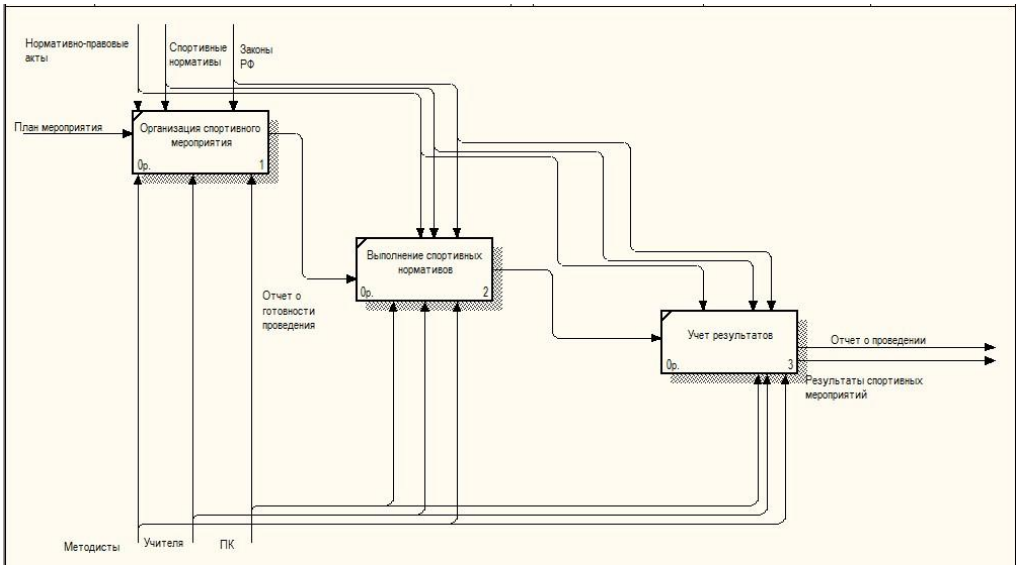


Рисунок 1.4 - Детализация блока «Проведение спортивных мероприятий»

Проанализировав все существующие требования и задачи к системе, можно сделать вывод, что фактически для достижения высших результатов необходимо внедрить использование вычислительной техники в тренировочный процесс.

Конечной целью всех этих преобразований является повышение учета спортивных достижений школы №48.

1.3 Постановка задачи

1.3.1 Цель и назначение автоматизированного варианта решения задачи

Автоматизация подсистемы управления спортивными достижениями построена на внедрении программного обеспечения в деятельность учителей физической культуры на всех этапах подготовки школьников.

Целью автоматизированной подсистемы управления спортивными достижениями является создание достойных условий, обеспечивающих возможность учителям вести белее скоординированные и продуктивные тренировки, получить доступ к развитой спортивной инфраструктуре, а также повысить конкурентоспособность школьников на городском уровне.

Автоматизированная подсистема значительно повысит качество подготовки школьников для участия в межшкольных стартах, что в перспективе позволит школьникам стабильно побеждать на крупнейших городских и областных спортивных соревнованиях. Эти успехи будут достигнуты за счет создания эффективной системы подготовки школьников и спортивного резерва с использованием новейших научных достижений.

Данная подсистема предназначена для отслеживания и корректировки спортивных достижений, создания электронного документооборота с Управлением образования г. Белгород, формирования электронных отчетов.

Деятельность школьных спортивных кружков необходимо автоматизировать, начиная с самого главного – тренировочного процесса. Данный комплекс поможет не только обрабатывать конечные результаты школьников, но и давать рекомендации по совершенствованию спортивных навыков. Подсистема управления спортивными достижениями поможет определить, какой вид спорта необходимо развивать для повышения конкурентоспособности школы.

1.3.2 Общая характеристика организации решения задачи на ЭВМ

Основной задачей автоматизированной подсистемы управления спортивными достижениями является обеспечение более эффективных и продуктивных взаимодействий между учителем и школьником, учителем и методическим объединением физической культуры.

Работа над автоматизированной подсистемой состоит из следующих основных этапов:

- построение диаграмм потоков данных;
- разработка логического и физического уровня модели данных в ERwin;
- разработка таблиц удаленной базы данных с использованием клиент-серверной технологии;

– разработка бизнес-логики на стороне SQL-сервера;

– разработка клиентского приложения с использованием технологии доступа к данным InterBaseExpress;

Поставленную задачу можно решить с помощью следующих средств и методов:

– AllFusionProcessModeler 7;

С помощью данной программы составляем диаграммы потоков данных. Это структурно-функциональная диаграмма, которая нужна для описания организации управления спортивными достижениями. Именно с помощью нее, можно просмотреть и проанализировать изменения, которые произойдут в организации после внедрения и использования автоматизированной подсистемы управления спортивными достижениями.

– База данных основана на технологии доступа «клиент-сервер».

База данных (БД) обеспечивает хранение информации, а также быстрый и удобный доступ к данным. Реляционная БД состоит из взаимосвязанных таблиц. Она представляет собой совокупность данных различного характера, организованных по определенным правилам. Каждая таблица содержит информацию об объектах одного типа, а совокупность всех таблиц образует единую БД.

Система управления базой данных (СУБД) - это совокупность языковых и программных средств, предназначенных для создания, ведения и использования БД. При создании автоматизированной подсистемы управления спортивными достижениями используется многопользовательская СУБД Firebird, позволяющая создавать информационные системы, функционирующие в архитектуре «клиент-сервер». Один сервер Firebird может обрабатывать несколько независимых баз данных, каждую с множеством пользовательских соединений.

– Для работы с СУБД Firebird используем утилиту IBExpert, при помощи которой можно администрировать сервер, управлять базами данных и правами пользователей. [10]

– Для разработки клиентского приложения выбрана среда программирования C++ Builder и используется технология доступа к данным InterBaseExpress.[5]

Технология InterBaseExpress строго ориентирована на работу с сервером InterBase версии не ниже 5.5. Преимущества технологии InterBaseExpress заключаются в том, все необходимые функции обеспечиваются путем прямого применения функций APIсервера InterBase. В результате не нужно использовать BDE, повышается скорость работы компонентов доступа к данным.

Недостатком технологии InterBaseExpress является невозможность использовать серверы баз данных, отличные от сервера InterBaseSQLServer.

1.4 Анализ существующих разработок и обоснование выбора технологии проектирования

В настоящий момент данной сфере отсутствует программный продукт по управлению спортивными достижениями. В школе №48 нет никакой автоматизированной подсистемы, нет даже никаких наработок, которые в дальнейшем можно было бы реализовать.

Разрабатываемая подсистема носит универсальный характер с учетом общих черт видов спортивной деятельности. То есть этот проект предназначен для школы №48, но может быть внедрен и любую другую. Он учитывает общие для школы виды спорта, позволяет быстрее и слаженнее координировать работу учителей и методистов.

2 Проектная часть

2.1 Обоснование проектных решений по техническому обеспечению

Развитие вычислительной техники не стоит на месте. Становясь более мощными, персональные компьютеры одновременно становятся менее дорогими и, следовательно, доступными для широкого круга пользователей.

Поскольку немаловажной задачей выпускной квалификационной работы является автоматизация деятельности работников школы №48, то необходимо грамотно подходить к вопросу выбора технического обеспечения.

В школе работниками используются пять компьютеров, входящих в сеть. Т.е. поддерживается клиент-серверная архитектура. «Клиент-сервер» - это вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемых серверами, и заказчиками услуг, называемых клиентами.

Нередко клиенты и серверы взаимодействуют через компьютерную сеть и могут быть как различными физическими устройствами, так и программным обеспечением.

Преимущества использования данной архитектуры:

– делает возможным, в большинстве случаев, распределение функции вычислительной системы между несколькими независимыми компьютерами в сети. Это позволяет упростить обслуживание вычислительной системы. В частности, ремонт, замена, перемещение или модернизация сервера, не затрагивают клиентов. Все данные хранятся на сервере, который, как правило, защищён гораздо лучше большинства клиентов.

– на сервере проще обеспечить контроль полномочий, чтобы разрешать доступ к данным только клиентам с соответствующими правами доступа.

– позволяет объединить различных клиентов. Использовать ресурсы одного сервера часто могут клиенты с разными аппаратными платформами, операционными системами и т.п.

В качестве рабочих мест, на которых планируется внедрение данной системы, выступают клиентские компьютеры, конфигурация которых представлена в таблице 2.1

Таблица 2.1 - Технические характеристики

№ п/п	Наименование компонента	Технические характеристики
1	2	3
1.	Корпус	DEXP AWS-DE6, 2xUSB, Black/Silver/Black 12cm FAN
2.	Материнскаяплата	ASUS P5K SE, S775, P35, ATX, DDR2, PCl-Exl6, SATA2, 1394, GbLAN (P5K SE/Retail)
3.	Процессор	Core 2 Duo E4600 BX80557E4600 2.40/800/2M Box LGA775 (Conroe)
4.	Видеокарта	ASUS SVGA <PCI-e> 256Mb GeForce 8400 GS SILENCER, 500MHz, 256Mb DDR2 667MHz/64 bit, PCI-Exl6, DVI, D-SUB, TV-Out, HDTV-Out
5.	Оперативнаяпамять	Kingston 1 Gb DDR2 PC-6400 (800MHz) DIMM CL5 ValueRAM 2 шт. Итого 2 Gb
6.	Жесткийдиск	Seagate 250.0GB ST3250410AS SATA2 7200rpm 16 MB
7.	Оптическийпривод	ASUSTek DVD±RWDRW-2014L1 SATA Black+White RTL
8.	МониторЖК	19" ViewSonic VA1926W, wide , silver-black , 1440*900, 170°/ 160°, 300кд/м, 2000:1, 5ms, DVI, TCO"03
9.	Клавиатура	Genius SlimStar 110 USB brown box
10.	Мышь	Logitech Grey Optical 3 button S96 PS2
11.	Операционная система и программное обеспечение	Windows XP Professional SP2 Russian Microsoft Office 2003 Pro Rus Kaspersky Antivirus 7.0 SQL Standard Edition 2000 Firebird 2.5 Borland C++ Builder 2010 (14)
12.	Дисковод	ALPS 3,5", 1.44M6

В таблице 2.2 представлена конфигурация компьютера, на котором установлен сервер.

Таблица 2.2 – Техническая характеристика сервера

№ п/п	Наименование компонента	Технические характеристики
1	2	3
1.	Корпус	Aquarius AquaServer P40 D30
2.	Материнская плата	Intel 5000V SSI EEB 3.6 (Extended ATX) Dual Intel Xeon Server Motherboard
3.	Процессор	Intel Xeon 5410 BOX 2U Active 2.33 ГГц/ 12Мб L2/ 1333МГц 771-LGA *2 штуки
4.	Видеокарта	ATI Radeon HD 5550 800-900 MHz DDR3 or 900-1000 MHz GDDR5
5.	Оперативнаяпамять	Samsung DDR2-FB-DIMM 2 Gb 667МГц ECC – 4 шт.
6.	Жесткийдиск	HDD Seagate 300Gb ST3300657SS SAS 15K.7 3.5" 16Mb*2 штуки
7.	Контроллер	Intel SRCSASBB8I (Black Butte) PCIe x8, 16P SAS RAID Controller, 256MB 667MHz memory embedded
8.	Оптическийпривод	Nec 7240 sata
9.	МониторЖК	19" ViewSonic VA1926W, wide , silver-black , 1440*900, 170°/ 160°, 300кд/м, 2000:1, 5ms, DVI, TCO"03
10.	Клавиатура	Genius SlimStar 110 USB black box
11.	Мышь	Genius Traveler 100 metallic optical (1200dpi) USB
12.	ADSL – модем	Zyxel P660RT2/RU2
13.	Операционная система и программное обеспечение	Microsoft Windows Server 2003 x64 Edition SQL Server Standard Edition 2000 64-bit Russian и 5 лицензийнапользователей
14.	Сетевое оборудование	D-Link DGS-1008D 8-ports 10/100/1000 Мбит/сек
15.	Источник бесперебойного питания	Powercom Black Knight Pro UPS-1000A BNT-1000AP
16.	ADSL – модем	Zyxel P660RT2/RU2
17.	Операционная система и программное обеспечение	Microsoft Windows Server 2003 x64 Edition SQL Server Standard Edition 2000 64-bit Russian и 5 лицензийнапользователей

2.2 Обоснование проектных решений по информационному обеспечению

Информационное обеспечение - это создание информационных условий функционирования системы, включение в систему средств поиска, обеспечение необходимой информацией, накопления, получения, хранения, передачи, обработки информации, организация банков данных.

Создание информационного обеспечения - непременное условие построения и функционирования автоматизированных систем управления.

Сбор данных. В момент каких-либо изменений в спортивных секциях школы (например, перевод детей, сдача контрольных нормативов, и т. д.) каждое происходящее действие сопровождается соответствующими записями данных. Любая новая информация фиксируется, заносится в журнал. С каждого школьника и учителя взято согласие на предоставление персональных данных. Работа с данными осуществляется строго в соответствии с законом РФ «О персональных данных» и Конституцией РФ. [1]

Обработка данных. Информационная технология обработки данных предназначена для решения хорошо структурированных задач, по которым имеются необходимые входные данные и известны алгоритмы и другие стандартные процедуры их обработки. Эта технология применяется на уровне исполнительской деятельности персонала невысокой квалификации в целях автоматизации некоторых рутинных постоянно повторяющихся операций управленческого труда.

На уровне операционной деятельности решаются следующие задачи:

– Обработка данных об операциях, происходящих в спортивных секциях и уроках физической культуры;

– Создание периодических контрольных отчетов о состоянии дел;

В настоящий момент в школе №48 работа практически не автоматизирована. Поэтому все документы, предназначенные для обработки, хранятся в бумажном виде. Есть специально созданные и разработанные бланки и журналы, в которых фиксируются все происходящие действия. Обработка большей части операций осуществляется вручную, а результат выводится на бланки строгой отчетности.

Обработку числовой информации производят табличные процессоры (электронные таблицы). Их основное назначение – автоматизация «рутинных» расчетов. В организации для обработки числовых результатов и проведения различных расчетов используется Microsoft Excel, при обработке числовой информации создаются электронные таблицы, которые можно просматривать, хранить, а также записывать на носители внешней памяти.

Хранение данных. Многие данные на уровне операционной деятельности необходимо сохранять для последующего использования либо здесь же, либо на другом уровне.

В настоящий момент все документы, созданный в ходе работы организации хранятся исключительно на бумажных носителях. Данные конфиденциальны, и доступ к ним может осуществлять только работник, имеющий для этого служебные полномочия.

В организации происходит ежегодный отбор документов на временное и постоянное хранение. Осуществляется описание дел временного хранения (например, по личному составу).

Создание отчетов. В информационной технологии обработки данных необходимо создавать документы для руководства и работников организации, а также для Управления по физической культуры, спорта и туризма Белгородской области. Отчеты составляются вручную без использования вычислительной техники за определенный период времени или по требованию начальства. Для создания отчетов у сотрудников имеются бланки, а также документы строгой отчетности. Все сведения необходимые для отчетов работники получают из журналов или созданных ранее документов.

Проанализировав созданные информационные условия функционирования системы, можно сделать вывод, что средства поиска, получения, хранения, обработки информации используются не на полную мощность. Необходимо внедрять в деятельность учителей физической культуры использование информационных технологий. Создание информационного обеспечения - непременное условие построения и функционирования автоматизированных систем управления. Все основные операции, производящие с документами необходимо усовершенствовать, а точнее автоматизировать. Это значительно упростит работу сотрудников организации, ускорит документооборот. Появится возможность создания, ведения и хранения документов не только на бумажном носителе, но и в электронном виде. Наладится процесс поиска необходимой информации.

Отчеты можно будет генерировать с помощью вычислительной техники, по определенным критериям. В итоге появится возможность наладить электронный документооборот с Управлением образования г. Белгород.

2.3 Обоснование проектных решений по программному обеспечению

Программное обеспечение - совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ. Также, это совокупность программ, правил и процедур, а также документации, относящихся к функционированию системы обработки данных.

Программное обеспечение является одним из видов обеспечения вычислительной системы, наряду с техническим (аппаратным), информационным, математическим, лингвистическим, организационным и методическим обеспечением. [13]

Операционная система в наибольшей степени определяет облик всей вычислительной системы в целом. Операционная система - это комплекс взаимосвязанных системных программ, назначение которого - организовать взаимодействие пользователя с компьютером и выполнение всех других программ. Она исполняет роль связующего звена между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем, с другой стороны.

В функции операционной системы входит:

- распределение ресурсов;
- запуск программ на выполнение;
- осуществление диалога с пользователем;
- ввод-вывод и управление данными;
- планирование и организация процесса обработки программ;
- всевозможные вспомогательные операции обслуживания;
- передача информации между различными внутренними устройствами;

– программная поддержка работы периферийных устройств.

Исходя из этого, можно сделать вывод, что выбор операционной системы, ее версии и класса является немаловажным фактором, который надо учесть при автоматизации деятельности школы №48 города Белгорода.

2.3.1 Выбор операционной системы

Операционные системы (ОС) могут различаться особенностями реализации внутренних алгоритмов управления основными ресурсами компьютера (памятью, процессорами, устройствами), особенностями использованных методов проектирования, типами аппаратных платформ, областями использования и многими другими свойствами. [8] Ниже приведена классификация ОС по нескольким наиболее основным признакам:

– Реализация сетевых возможностей. Специфика ОС проявляется и в том, каким образом она реализует сетевые функции: распознавание и перенаправление в сеть запросов к удаленным ресурсам, передача сообщений по сети, выполнение удаленных запросов. При реализации сетевых функций возникает комплекс задач, связанных с распределенным характером хранения и обработки данных в сети: ведение справочной информации обо всех доступных в сети ресурсах и серверах, адресация взаимодействующих процессов, обеспечение прозрачности доступа, тиражирование данных, согласование копий, поддержка безопасности данных;

– Особенности алгоритмов управления ресурсами.

В зависимости от особенностей использованного алгоритма управления процессором, операционные системы делят на многозадачные и однозадачные, многопользовательские и однопользовательские, на многопроцессорные и однопроцессорные системы.

Поддержка многозадачности. По числу одновременно выполняемых задач операционные системы могут быть разделены на два класса:

однозадачные (например, MSX, MS-DOS) и многозадачные (ОС ЕС, OS/2, Windows, UNIX).

Однозадачные ОС в основном выполняют функцию предоставления пользователю виртуальной машины, делая более удобным и простым процесс взаимодействия пользователя с компьютером.

Многозадачные ОС, кроме вышеперечисленных функций, управляют разделением совместно используемых ресурсов, таких как процессор, оперативная память, внешние устройства и файлы.

Поддержка многопользовательского режима. По числу одновременно работающих пользователей ОС делятся на однопользовательские (MS-DOS, Windows 3.x, ранние версии OS/2) и многопользовательские (UNIX, Windows).

Главным отличием многопользовательских систем от однопользовательских является наличие средств защиты информации каждого пользователя от несанкционированного доступа других пользователей.

Поддержка многонитевости. Важным свойством операционных систем является возможность распараллеливания вычислений в рамках одной задачи.

Многопроцессорная обработка. Другим важным свойством ОС является отсутствие или наличие в ней мультипроцессирования. Сейчас поддержка этой функции является обязательной.

– Особенности областей использования. Многозадачные ОС подразделяются на три типа в соответствии с использованными при их разработке критериями эффективности: системы пакетной обработки (например, ОС ЕС), системы разделения времени (UNIX, VMS), системы реального времени (QNX, RT/11).

– Особенности аппаратных платформ. По типу аппаратуры различают операционные системы персональных компьютеров, мини-компьютеров, мейнфреймов, кластеров и сетей ЭВМ.

– Особенности методов построения. Способы построения ядра системы - монолитное ядро или микроядерный подход. Большинство ОС использует монолитное ядро, которое компонуется как одна программа,

работающая в привилегированном режиме и использующая быстрые переходы с одной процедуры на другую, не требующие переключения из привилегированного режима в пользовательский и наоборот.

Построение ОС на базе объектно-ориентированного подхода дает возможность использовать все его достоинства, хорошо зарекомендовавшие себя на уровне приложений, внутри операционной системы, а именно: аккумуляцию удачных решений в форме стандартных объектов, возможность создания новых объектов на базе имеющихся с помощью механизма наследования, хорошую защиту данных за счет их инкапсуляции во внутренние структуры объекта, что делает данные недоступными для несанкционированного использования извне, структурированность системы, состоящей из набора хорошо определенных объектов.

Наличие нескольких прикладных сред дает возможность в рамках одной ОС одновременно выполнять приложения, разработанные для нескольких ОС. Многие современные операционные системы поддерживают одновременно прикладные среды MS-DOS, Windows, UNIX (POSIX), OS/2 или хотя бы некоторого подмножества из этого популярного набора. Распределенная организация операционной системы позволяет упростить работу пользователей и программистов в сетевых средах.

Проанализировав данную классификацию можно сделать вывод, что в нашем случае, наиболее удобной и гибкой будет операционная система Windows 7 professional. Системные требования для Windows 7 - 32-разрядныйпроцессор с тактовой частотой 1 ГГц или выше, 1 ГБ RAM, 16 ГБ GB HDD, CD-ROM.

Основные возможности - настраиваемый интерфейс, гибкая, а так же удобная эксплуатация и настройка (система интерактивных подсказок, более удобная работа с объектами, поиск по различным категориям), расширенный многопользовательский режим, который быстро и легко подключает к работе новых пользователей, новые технологии и программы - дистанционная помощь и диагностика неисправностей, быстрое подключение устройств, находящихся

на удаленном сетевом компьютере, технология Microsoft.NET - технология интеграции с сетью.

Операционная система Windows 7 Professional является стандартизированной системой, она широко используется, по данной системе разработано много справочной литературы. В настоящий момент существует множество специалистов, занимающихся ее установкой, настройкой и эксплуатацией. По тем же самым причинам для нашей серверной части подходит операционная система Windows Server 2003.

Операционная система Windows 7 обеспечивает высокий уровень стабильности, предоставляя пользователям возможность сосредоточиться на выполняемой работе. Windows 7 Professional обеспечивает высокое быстродействие в сочетании с возможностью масштабирования по мере увеличения объема памяти и мощности процессора. Аварийное восстановление системы - это функция средства архивации, позволяющая восстановить состояние системы и все файлы в системном разделе, когда проблемы или изменения в операционной системе становятся причиной нестабильной работы и сбоев при загрузке. Общий доступ к подключению Интернета позволяет осуществлять одновременный вход в интернет с нескольких компьютеров через одно и то же широкополосное подключение или подключение удаленного доступа. Брандмауэр подключения к интернету обеспечивает защиту ваших компьютеров от атак хакеров и вторжения через интернет.

Решение типичных задач происходит достаточно быстро благодаря более понятному для пользователей интерфейсу и наличию новых визуальных подсказок.

Также Windows 7 содержит пакет исправлений ServicePack 1, акцент в котором сделан на безопасность. Пакет обеспечивает более полный контроль над коммуникационными протоколами. Операционная система защищена от переполнений буфера, используемых авторами многих «троянских коней».

Windows Server 2003 - операционная система семейства Windows NT от компании Microsoft, предназначенная для работы на серверах.

Windows Server 2003 является наиболее быстрой, надежной и безопасной операционной системой в семействе серверных ОС Windows. Надежность данной системы обусловлена интегрированной инфраструктурой, гарантирующей безопасность деловой информации, а также надежностью, доступностью и масштабируемостью сетевой инфраструктуры. Данная операционная система предоставляет средства, позволяющие развертывать, управлять и использовать сетевую инфраструктуру с максимальной производительностью.

2.3.2 Характеристика базы данных

Системы управления базами данных (СУБД) – это совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Классификации СУБД:

- по модели данных выделяют: иерархические, сетевые, реляционные и объектно-ориентированные;
- файл-серверные;
- по степени распределенности выделяют: локальные СУБД (все части локальной СУБД размещаются на одном компьютере), распределённые СУБД (части СУБД могут размещаться на двух и более компьютерах);
- по способу доступа к БД выделяют:

В файл-серверных СУБД файлы данных располагаются централизованно на файл-сервере. СУБД располагается на каждом клиентском компьютере (рабочей станции). Доступ СУБД к данным осуществляется через локальную сеть. Синхронизация чтений и обновлений осуществляется посредством файловых блокировок. Преимуществом этой архитектуры является низкая нагрузка на центральный процессор сервера. Недостатки: потенциально высокая загрузка локальной сети; затруднённость централизованного управления; затруднённость обеспечения таких важных характеристик как

высокая надёжность, высокая доступность и высокая безопасность. Применяются чаще всего в локальных приложениях, которые используют функции управления БД. Примеры: MicrosoftAccess, Paradox, dBase, FoxPro, VisualFoxPro.

– клиент-серверные;

Клиент-серверная СУБД располагается на сервере вместе с БД и осуществляет доступ к БД непосредственно, в монопольном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно. Недостаток клиент-серверных СУБД состоит в повышенных требованиях к серверу. Достоинствами является: потенциально более низкая загрузка локальной сети; удобство централизованного управления; удобство обеспечения таких важных характеристик как высокая надёжность, высокая доступность и высокая безопасность. Примеры: Oracle, Firebird, Interbase, IBM DB2, MS SQL Server, Sybase, PostgreSQL, MySQL, ЛИНТЕР, MDBS.

– встраиваемые;

Встраиваемая СУБД - библиотека, которая позволяет унифицированным образом хранить большие объёмы данных на локальной машине. Доступ к данным может происходить через SQL либо через особые функции СУБД.

Встраиваемые СУБД быстрее обычных клиент-серверных и не требуют установки сервера, поэтому востребованы в локальном программном обеспечении (ПО), которое имеет дело с большими объёмами данных. Примеры: OpenEdge, SQLite, BerkeleyDB, один из вариантов Firebird, MySQL, SavZigzag, Microsoft SQL ServerCompact, ЛИНТЕР.

Проанализировав разновидности СУБД, пришли к выводу, что для нашей автоматизированной системы полностью подходит и удовлетворяет основные требования СУБД Firebird. [6] Одной из основных причин выбора именно этой СУБД считается то, что Firebird является полностью свободным сервером от лицензионных отчислений даже для коммерческого использования. Firebird абсолютно бесплатен для использования и распространения.

Firebird - компактная, кроссплатформенная, свободная система управления базами данных, работающая на GNU/Linux, MicrosoftWindows и разнообразных Unix платформах.

Firebird создан специально, чтобы удовлетворять требованиям «атомарности, целостности, изоляции и надёжности» транзакций. Основная особенность Firebird - версионная архитектура, позволяющая серверу обрабатывать различные версии одной и той же записи в любое время таким образом, что каждая транзакция видит свою версию данных, не мешая соседним. Используя язык PSQL (процедурный SQL) Firebird, возможно создавать сложные хранимые процедуры для обработки данных полностью на стороне сервера. Для генерации отчётов особенно удобны хранимые процедуры с возможностью выборки, возвращающие данные в виде набора записей. Немаловажным достоинством является резервное копирование на лету, т.е. для резервного копирования не надо останавливать сервер. Такие процедуры можно использовать в запросах точно так же, как и обычные таблицы.

Процесс резервного копирования сохраняет состояние базы данных на момент своего старта, не мешая при этом работе с базой.

2.4 Обоснование проектных решений по технологическому обеспечению

Технологическое обеспечение ИС соответствует разделению информационной системы на подсистемы по технологическим этапам обработки различных видов информации: первичной информации, организационно-распорядительной документации, технологической документации, баз данных и знаний, научно-технической информации. [9]

Технологический процесс обработки данных - это процесс преобразования исходной информации в выходную с использованием технических средств и ресурсов.

Технологический процесс можно разделить на четыре основных этапа:

– начальный или первичный (сбор исходных данных, их регистрация и передача);

В настоящий момент в школе №48 города Белгорода применяется механизированный способ сбора и регистрации данных. Т.е сбор и регистрация информации осуществляется непосредственно работником с использованием простейших приборов. Оформление документов происходит только на бумажном носителе. Входная информация регистрируется в специализированных журналах без создания их электронной копии. Передача документов для их последующей обработки происходит из рук в руки.

– основной (непосредственно обработка информации);

Обработка полученной информации осуществляется сотрудником собственноручно, практически без использования вычислительной техники. Он сопоставляет данные, оценивает различные показатели, формирует требования и вопросы. Если необходимо произвести какие-либо расчеты или подсчитать результаты используется программа Microsoft Excel. Так же с помощью нее строятся диаграммы для наглядного оценивания полученных результатов.

– Заключительный (контроль, выпуск и передача результатной информации, ее размножение и хранение);

Отчеты составляются за определенный временной период или по требованию руководства. Для получения результатной информации сотрудник заполняет соответствующие бланки. Отчет об исполнении документа работник предоставляет вышестоящему должностному лицу. После этого документ подшивается в соответствующую папку, а копия этого документа направляется в архив на хранение. Передача результатной информации в Управление образования г. Белгород осуществляется путем личной доставки отчета сотрудником школы в Управление.

Проанализировав возможности модернизации существующих информационных технологий, используемых школой №48 города Белгорода, пришли к выводу, что имеется ряд незадействованных ресурсов: трудовых,

организационных и информационных, которые могут быть наиболее эффективно задействованы и реализованы.

Например, способ сбора и регистрации информации может быть не механическим, а автоматизированным. Т.е. сотруднику поступает документ на исполнение, он его регистрирует в журнале событий, а затем вносит соответствующие данные в специально разработанную программу на компьютере.

Запись и передача информации в ЭВМ имеет следующие преимущества:

- упрощает процесс формирования и контроля информации;
- соблюдается принцип однократной регистрации информации в первичном документе и машинном носителе;
- обеспечивается высокая достоверность информации, поступающей в ЭВМ;

Технологический процесс обработки информации с использованием ЭВМ включает в себя следующие операции:

- прием и сортировка первичных документов - проверка полноты и качества их заполнения и т. д.;
- подготовка машинного носителя и контроль;
- ввод данных в ЭВМ;
- контроль; Различают визуальный и программный контроль, позволяющий отслеживать информацию на полноту ввода, нарушение структуры исходных данных. При обнаружении ошибки производится исправление вводимых данных, корректировка и их повторный ввод. Доступом к базе данных обладает работник с соответствующими полномочиями [2]. Следовательно, в организации будет реализовано разграничение прав доступа и конфиденциальность вводимой информации.
- сортировка (если в этом есть необходимость);
- обработка данных;
- контроль и выдача результатной информации;

Получается, что отчеты теперь составляются как в бумажном, так и в электронном виде. А именно благодаря этому можно наладить электронный документооборот между школой и Управлением образования г. Белгород.

Перечисляя операции технологического процесса, хотелось бы несколько слов сказать об операции хранения информации. Современные экономические информационные системы с мощными процессорами, оснащенными съемными винчестерами, CD-ROM'ами с лазерными дисками, обеспечивают более высокую скорость обработки информации и предоставляют пользователю работать с большими объемами данных, обеспечивая удобство в работе и надежность в сохранности информации. [11]

2.5 Обоснование выбора программных средств

Технология проектирования программных средств определяется как совокупность трех составляющих:

- пошаговой процедуры, определяющей последовательность и время операций проектирования ПО;
- нотаций (графических и текстовых средств), используемых для описания проектируемой системы;
- критериев и правил, используемых для оценки результатов выполнения этих операций;

Методологии, технологии и инструментальные средства проектирования (CASE-средства) составляют основу проекта любой информационной системы, а тем более автоматизированной системы [4]. Методология реализуется через конкретные технологии и поддерживающие их стандарты, методики и инструментальные средства, которые обеспечивают выполнение процессов жизненного цикла.

При разработке автоматизированной подсистемы управления спортивными достижениями использовали структурный подход. Особенность его в декомпозиции (разбиении) на автоматизируемые функции: система

разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая подсистема сохраняет целостное представление, в котором все составляющие компоненты взаимоувязаны. [7]

2.5.1 Средства и методы решения задач

– AllFusionProcessModeler 7 (ранее BPwin) - инструмент для моделирования, анализа, документирования и оптимизации бизнес-процессов. AllFusionProcessModeler 7 можно использовать для графического представления бизнес-процессов.

Графически представленная схема выполнения работ, обмена информацией, документооборота визуализирует модель бизнес-процесса [16].

На этом этапе с помощью построения диаграмм проследили работу спортивных секций. Проанализировав диаграммы, отметили, что технические средства организацией практически не используются. И автоматизацию необходимо начинать чуть ли не с тренировочного процесса. Пришли к выводу, что результаты и достижения можно улучшить, в том числе и с помощью технической подкованности и оснащенности.

– инструментальная среда ERWin;

AllFusionERwinDataModeler - позволяет проектировать, документировать и сопровождать базы данных, хранилища данных и витрины данных (datamarts). Создав наглядную модель базы данных, можно оптимизировать структуру БД и добиться её полного соответствия требованиям и задачам организации. Визуальное моделирование повышает качество создаваемой базы данных, продуктивность и скорость её разработки [16].

На этом этапе инструментальную среду ERWin использовали для создания логической и физической модели БД. Непосредственно здесь проектировались таблицы, которые вошли в базу данных и связи между ними.

Именно благодаря данному построению таблиц, в дальнейшем смогли реализовать каскадное удаление и добавление данных.

– для создания удаленной БД использовали СУБД Firebird с утилитой IBExpert;

На этом этапе нам необходимо было зарегистрировать базу на сервере, создать домены, таблицы, задать ограничения в таблицах, создать индексы (для осуществления последующего поиска и сортировки данных), генераторы (для дальнейшего создания уникальной последовательности чисел и автоматического заполнения полей, входящих в первичный ключ, при вставке и обновлении записей), а также триггеры.

Далее необходимо приступить к разработке бизнес-логики на стороне SQL-сервера. Для этого необходимо было создать просмотры по данным различных таблиц, хранимые процедуры (для осуществления поиска, добавления и удаления данных из таблиц), исключения (для бесперебойной работы).

– для разработки клиентского приложения была выбрана среда программирования C++ Builder и использована технология доступа к данным InterBaseExpress [21];

C++ Builder был выбран потому, что в школе №48 планировалось усовершенствование программной составляющей. Именно для дальнейшего развития школы был выделен сервер, а в качестве программного продукта для взаимодействия с серверной частью была приобретена и установлена лицензионная версия C++ Builder. Поэтому для экономии финансовой составляющей, клиентское приложение было разработано в данной среде программирования.

На этом этапе разработана программа, которая позволяет просматривать и редактировать таблицы БД, в ней реализованы функции поиска, удаления, добавления, занесение результатов по контрольным нормативам. Так же здесь реализованы отчеты по школьникам, учителям, соревновательной нагрузке, контрольным нормативам и т.д.

2.6 Информационное обеспечение задачи

Для создания автоматизированной подсистемы управления спортивными достижениями в школе №48 города Белгорода есть все предпосылки. Техническое и программное обеспечение находится на достаточно высоком уровне и полностью удовлетворяет требованиям, возникающим при создании системы. А вот информационное и технологическое обеспечение явно требует доработок. Разработанная система даст толчок для усовершенствования этих пунктов.

Благодаря данной программе на начальном этапе сбор и регистрация информации (данных) будут автоматизированными. Т.е. появится возможность использования машиночитаемых документов.

Обновление данных и их обработка будут производиться децентрализовано, поскольку в школе работает много учителей, каждый учитель ведет определенный класс. Методическое объединение физической культуры это все четко отслеживает и по мере необходимости корректируют данные в программе. Появляется возможность обновлять в заданные сроки любой объем данных с высокой степенью надежности. Обработка данных осуществляется в диалоговом режиме. Данный режим используется для доступа к информации, вычислительным или программным ресурсам.

Сбор исходной информации будет осуществляться распределенным способом. Он реализован следующим образом: сбор информации может осуществляться с любой локальной машины, при этом обработка данных осуществляется одной или несколькими ЭВМ в зависимости от реальных возможностей системы. В нашем случае нецелесообразно использовать специальные технические средства, это потребует лишь дополнительных денежных вложений и никак не отразиться на работе спортивных секций.

Передача данных между автоматизированными рабочими местами будет происходить по средствам локальной сети в электронной форме после получения запроса. А передача данных в Управление образования г. Белгород

будет осуществляться за счет передачи отчетов в электронной форме по средствам сети Internet. Именно за счет этого планируется наладить электронный документооборот.

В данной системе реализовано централизованное хранение информации. Т.е. мы можем просматривать по запросу интересующие нас данные в виде результативных документов.

Появляется возможность автоматического генерирования электронных отчетов. Мы можем просмотреть их на экране, сохранить полученные результаты в файл, а можем распечатать на бумажный носитель. Функции печати в программе предусмотрены.

А непосредственно автоматизированная подсистема управления спортивными достижениями поможет грамотно скоординировать тренировочный процесс учителям по физической культуре, поможет выявить школьникам сильные и слабые стороны физической подготовленности к соревновательному периоду, упростит и ускорит процесс взаимодействия школы с Управлением образования г. Белгород.

3 Программная реализация проектных решений

3.1 Информационная модель и ее описание

Организация работы будет описана на примере взаимодействия Управления по физической культуре, спорту и туризму Белгородской области со школой №48 в городе Белгороде.

В результате внедрения автоматизированной подсистемы управления спортивными достижениями в деятельность школы, работник сможет фиксировать всю поступающую информацию, как на бумажных носителях, так и на ЭВМ.

Контекстная диаграмма автоматизированной подсистемы учета спортивных достижений приведена на рисунке 3.1.

Входной информацией системы является информация о спортивных мероприятиях, информация о школьниках, информация о учителях, выходной информацией подсистемы являются переданный план, отчет о проведении, отчет в БД, информация о спортивных достижениях. Все сведения фиксируются учителем при помощи вычислительной техники с учетом определенных правил и законов РФ.

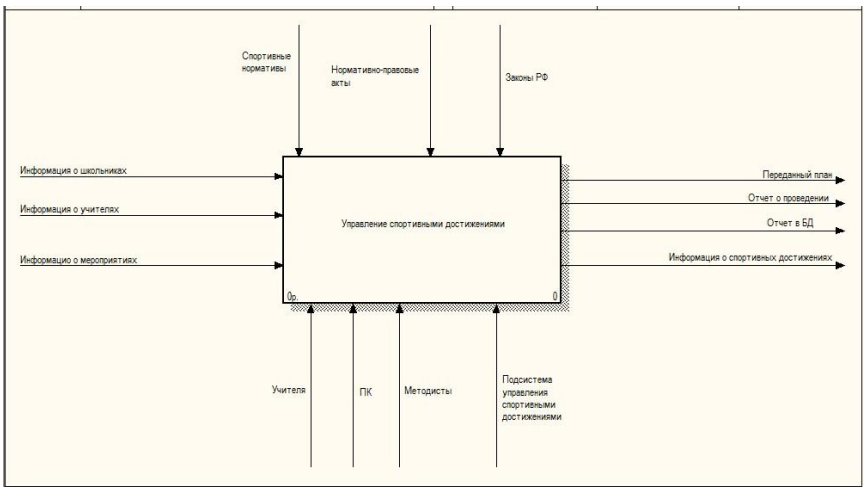


Рисунок 3.1 - Контекстная диаграмма автоматизированной подсистемы учета спортивных достижений

Детализация контекстной диаграммы представлена на рисунке 3.2. На данном этапе с учетом существующих инструкций и законов РФ учителям с помощью ПК необходимо составить план спортивных мероприятий, провести спортивные мероприятия, провести учет спортивных результатов, а также подготовить отчеты.

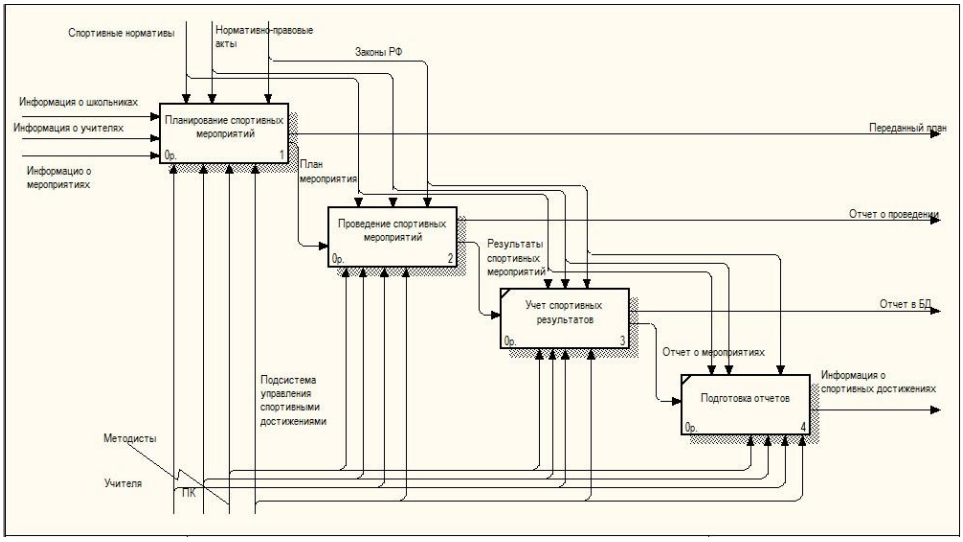


Рисунок 3.2 - Детализация контекстной диаграммы

Детализация блока «Планирование спортивных мероприятий» диаграммы представлена на рисунке 3.3. На данном этапе с учетом существующих правил учителям с помощью ПК необходимо составить план мероприятий в школе, согласовать его в Управлении образования г. Белгород, если потребуется, доработать план, утвердить новый план в Управлении образования г. Белгород и составить отчет о мероприятиях.

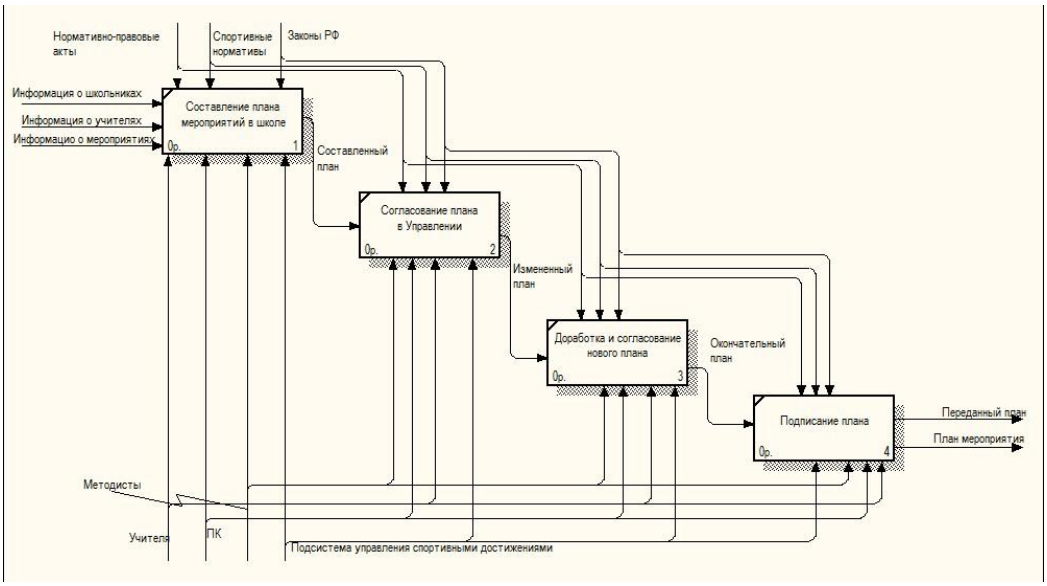


Рисунок 3.3 - Детализация блока «Планирование спортивных мероприятий»

Детализация блока «Проведение спортивных мероприятий» диаграммы представлена на рисунке 3.4 На данном этапе с учетом существующих правил учителям необходимо принять спортивные нормативы и передать результаты.

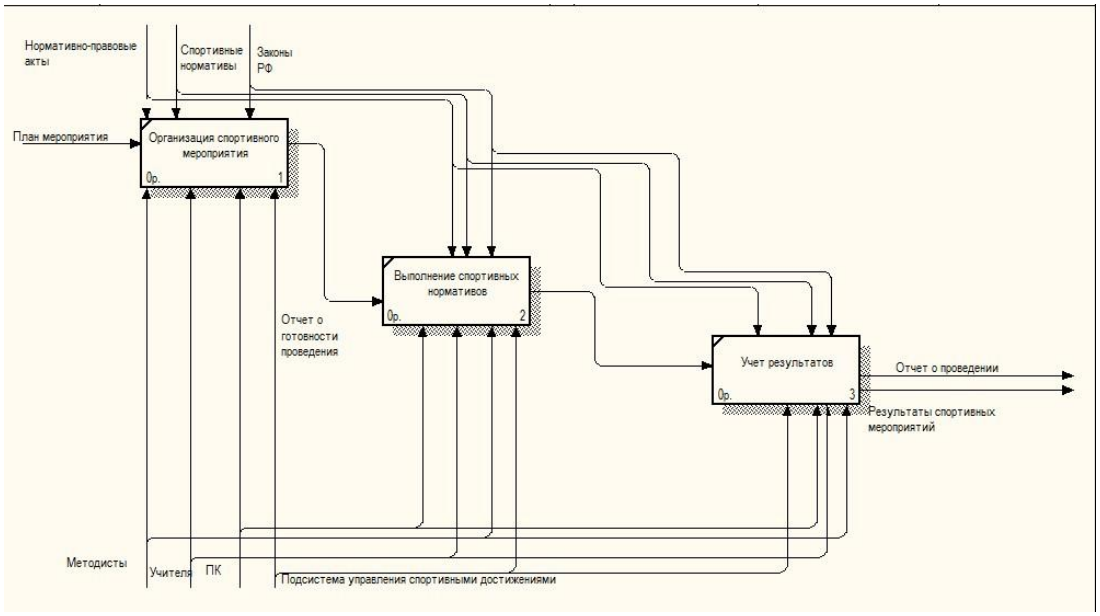


Рисунок 3.4 - Детализация блока «Проведение спортивных мероприятий»

Проанализировав получившуюся информационную модель можно сделать вывод, что деятельность учителей школы станет более автоматизированной. Они начнут применять, вычислительную технику для

решения поставленных задач. Появится возможность использования человеческих ресурсов, ранее не задействованных в работе.

3.2 Структурная схема подсистемы

3.2.1.1 Создание логической модели данных

Логическая модель создается для того, чтобы отразить взаимодействия таблиц, входящих в БД друг с другом. Именно на этом этапе необходимо грамотно определить связи между таблицами, поскольку в дальнейшем это нам понадобиться для осуществления каскадных операций над данными.

Для внесения сущности в модель использовали кнопку сущности на панели инструментов.

Для данной сущности определили имя, комментарии и описание. Этим же способом создали еще шесть сущностей, которые в дальнейшем будут таблицами в нашей БД.

Далее установили связи между сущностями. При помощи редактора связей Relationships определили родительскую и дочернюю сущности.

Затем указали первичные ключи и не ключевые атрибуты. Для задания первичных ключей и атрибутов использовали редактор атрибутов. На этом процесс логического моделирования завершен, Внешний вид полученной модели представлен на рисунке 3.5

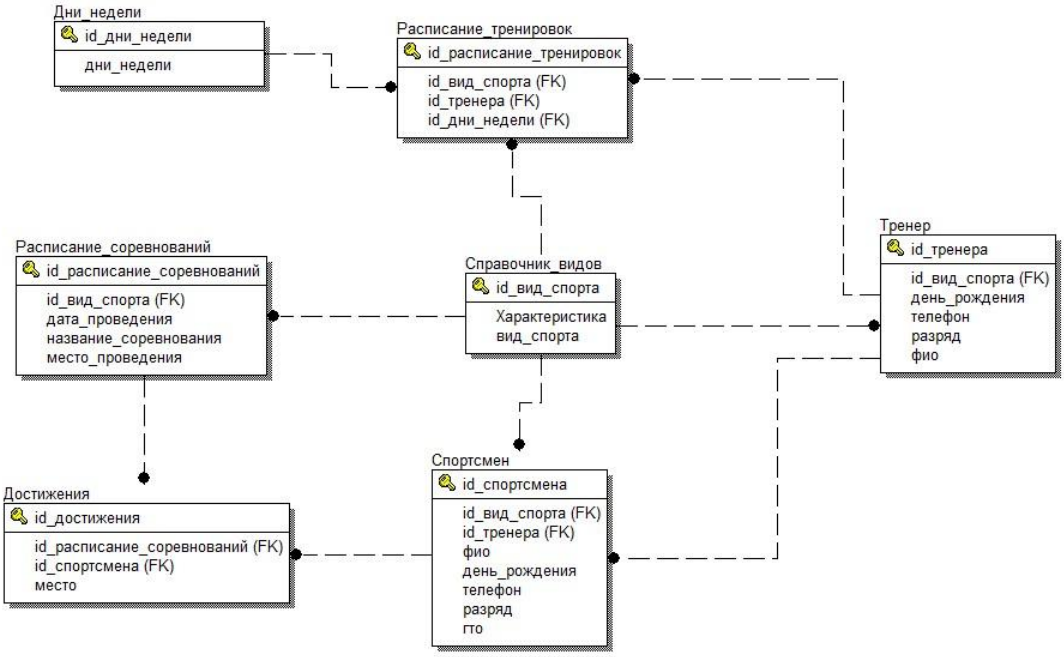


Рисунок 3.5 - Общий вид логической модели

3.2.1.2 Создание физической модели данных

Перед тем как преступить к созданию физической модели, выбрали сервер СУБД.

Поскольку логическая модель разрабатывалась на русском языке, то имена таблиц, колонок и индексов задали на английском языке.

Кроме того, для каждой колонки указали тип данных, возможность пустых значений и т.п.

Для создания английских имен таблиц воспользовались редактором таблиц, для остальных манипуляций - редактором колонок.

После того, как были выполнены все действия, физическая модель приобрела вид, показанный на рисунке 3.6

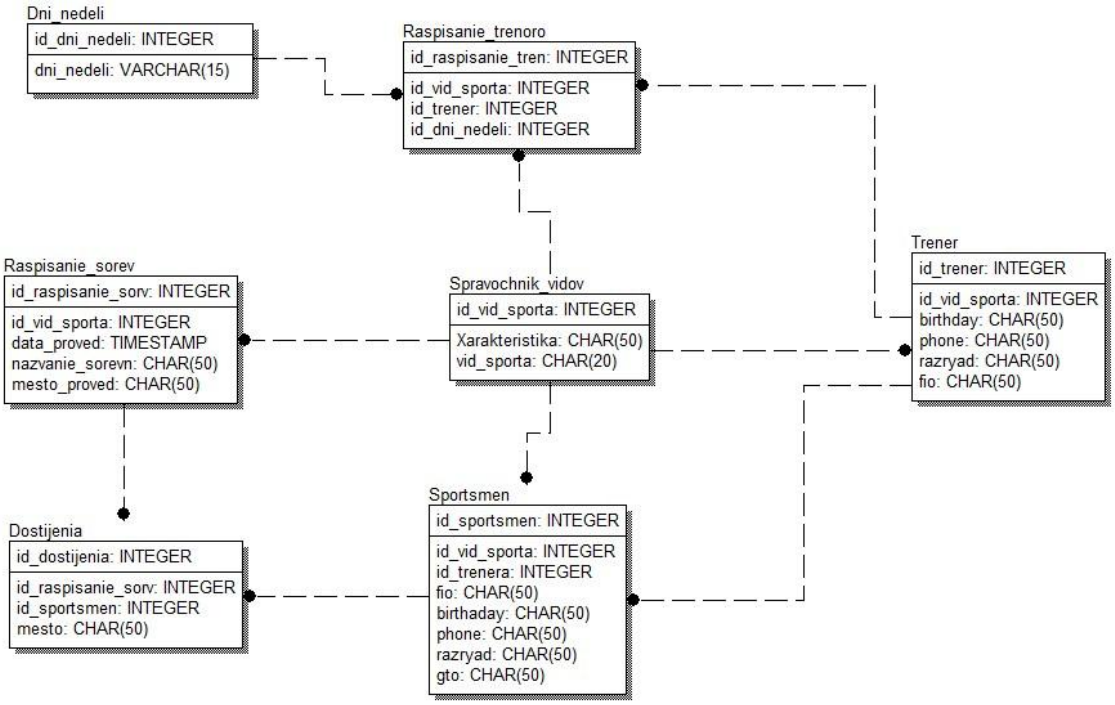


Рисунок 3.6 - Общий вид физической модели

После этого сгенерировали БД.

Полученный код приведен в приложении Б.

Генерация схемы БД запустилась нажатием кнопки Generate.Для просмотра кода, который был создан автоматически, нажали кнопку Preview. Результат представлен на рисунке 3.7

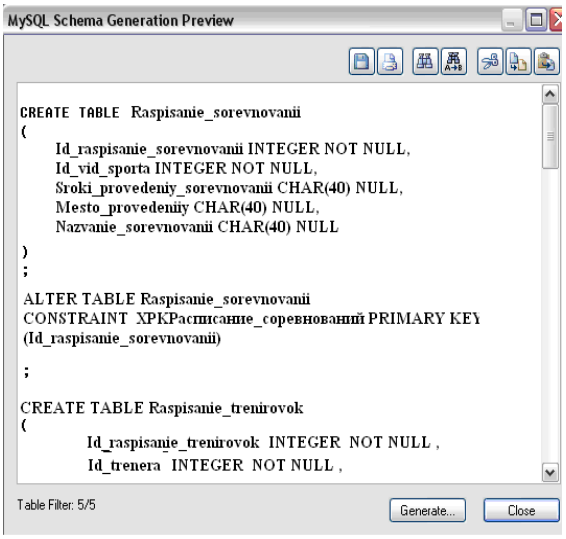


Рисунок 3.7 - Окно просмотра программного кода

3.2.2 Разработка таблиц удаленной базы данных

3.2.2.1 Создание базы данных

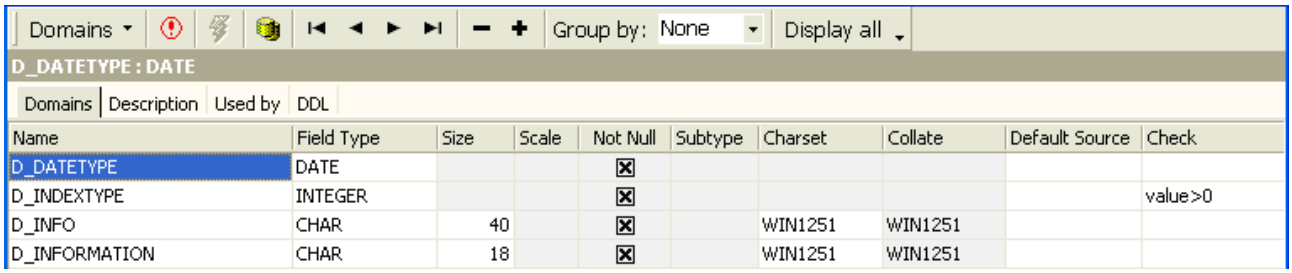
Для создания базы данных запустили утилиту IBExpert и выбрали в меню «Базы данных» пункт «Создать базу» [17]. Затем установили соединение с сервером. В поле «Файл БД» указали имя файла базы данных DATA_BASE.FDBи путь к нему. Потом зарегистрировали БД. В итоге БД DATA_BASE.FDBбыла создана.

3.2.2.2 Создание доменов

В нашей БД логично создать домены, так как таблицы БД содержат поля, идентичные по описанию.

Для этого при помощи мастера создали четыре домена. Результат выполнения представлен на рисунке 3.8

Код создания доменов приведен в приложении Б.



Name	Field Type	Size	Scale	Not Null	Subtype	Charset	Collate	Default Source	Check
D_DATETYPE	DATE			☒					
D_INDEXTYPE	INTEGER			☒					value>0
D_INFO	CHAR	40		☒		WIN1251	WIN1251		
D_INFORMATION	CHAR	18		☒		WIN1251	WIN1251		

Рисунок 3.8 - Созданные домены

3.2.2.3 - Создание таблиц

При создании таблиц использовали ранее созданные домены.

При создании таблиц использовали SQL-редактор. Создание таблицы TRENERA представлено на рисунке 3.10

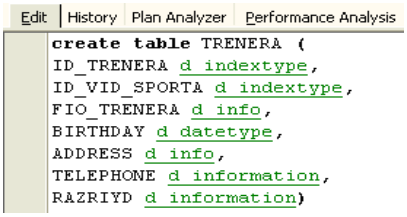


Рисунок 3.9 Создание таблицы TRENERA

Код создания таблиц приведен в приложении Б

Характеристика семи созданных таблиц БД представлена в таблице 3.1

Таблица 3.1 - Таблицы БД

Имя таблицы	Поле	Тип	Not Null	Check	Primary key	Имя домена
1	2	3	4	5	6	7
DNI_ NEDELI	ID_DNI_NEDELI	integer	true	value>0	true	d_indextype
	DNI_NEDELI	char(18)	true		false	d_information
RASPISANIE_SOREVNOVANII	ID_ RASPISANIE_SOREVNOVANII	integer	true	value>0	true	d_indextype
	ID_VID_SPORTA	integer	true		true	d_indextype
	SROKI_PROVEDENIY_SOREVNOVANII	char(40)	true		false	d_info
	MESTO_PROVEDENIIY	char(40)	true		false	d_info
	NAZVANIE_SOREVNOVANII	char(40)	true		false	d_info
SPORTSMEN	ID_ SPORTSMEN	integer	true	value>0	true	d_indextype
	ID_VID_SPORTA	integer	true		true	d_indextype
	ID_TRENERA	integer	true		true	d_indextype
	FIO_SPORTSMENA	char(40)	true		false	d_info
	BIRTHDAY	date	true		false	d_datetype
	ADDRESS	char(40)	true		false	d_info
	TELEPHONE	char(18)	true		false	d_information
	Поле	Тип	NotNull	Check	Primary key	Имя домена
	GTO	date	true		false	d_datetype
	RAZRIYD	char(18)	true		false	d_information

Продолжение таблицы 3.1

Имя таблицы	Поле	Тип	NotNull	Check	Primary key	Имя домена
SPRAVOCHNIK_PO_VIDAM_SPORTA	ID_ VID_SPORTA	integer	true	value>0	true	d_indextype
	VID_SPORTA	char(40)	true		false	d_info
	XARAKTERISTIKA	char(40)	true		false	d_info
TRENERA	ID_TRENERA	integer	true	value>0	true	d_indextype
	ID_VID_SPORTA	integer	true		true	d_indextype
	FIO_TRENERA	char(40)	true		false	d_info
	BIRTHDAY	date	true		false	d_datetype
	ADDRESS	char(40)	true		false	d_info
	TELEPHONE	char(18)	true		false	d_information
	RAZRIYD	char(18)	true		false	d_information
RASPISANIE_TRENIROVOK	ID_RASPISANIE_TRENIROVOK	integer	true	value>0	true	d_indextype
	ID_TRENERA	integer	true		true	d_indextype
	ID_VID_SPORTA	integer	true		true	d_indextype
	ID_DNI_NEDELI	integer	true		true	d_indextype
	TIME	char(40)	true		false	d_info
DOSTIJENIA	ID_DOSTIJENIA	integer	true			
	ID_SPORSMEN	integer	true			
	ID_RASPISANIE_SOREVN	integer	true			
	MESTO	char(40)	true			

3.3 Проектирование клиентского приложения

Внешний вид главной формы с размещенными на ней компонентами показан на рисунке 3.10

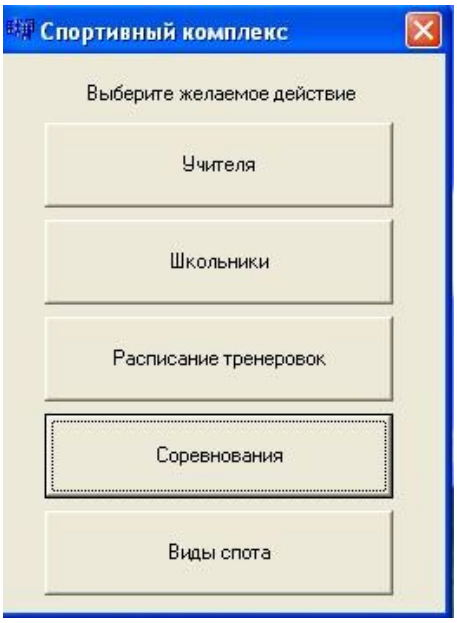


Рисунок 3.10 - Внешний вид формы «Спортивный комплекс»

Внешний вид формы, предназначенной для просмотра данных и изменения данных таблицы «Учителя» показан на рисунке3.11

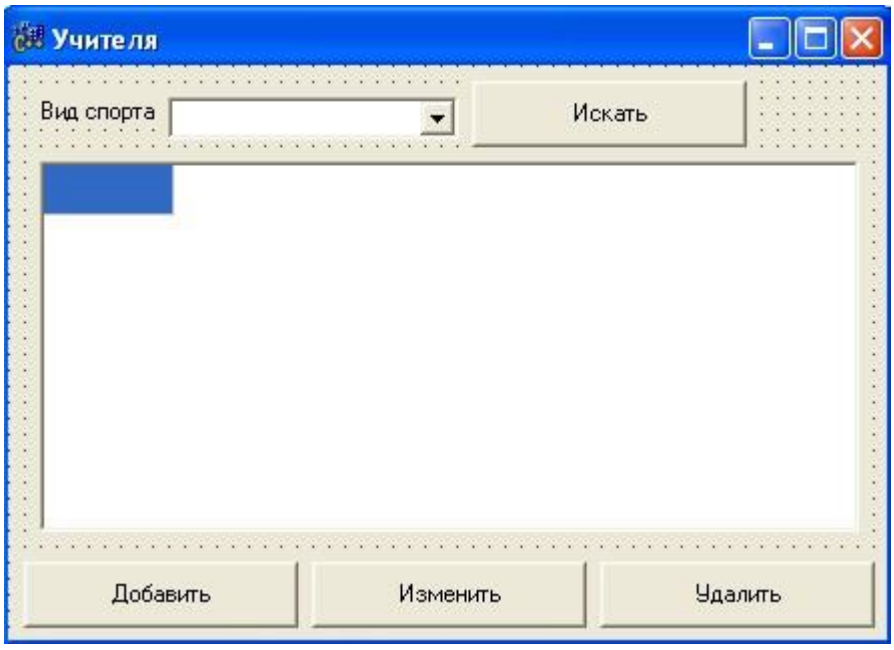


Рисунок 3.11 - Внешний вид формы «Учителя»

Внешний вид формы, предназначенной для просмотра данных и изменения данных таблицы «Виды спорта» показан на рисунке 3.12

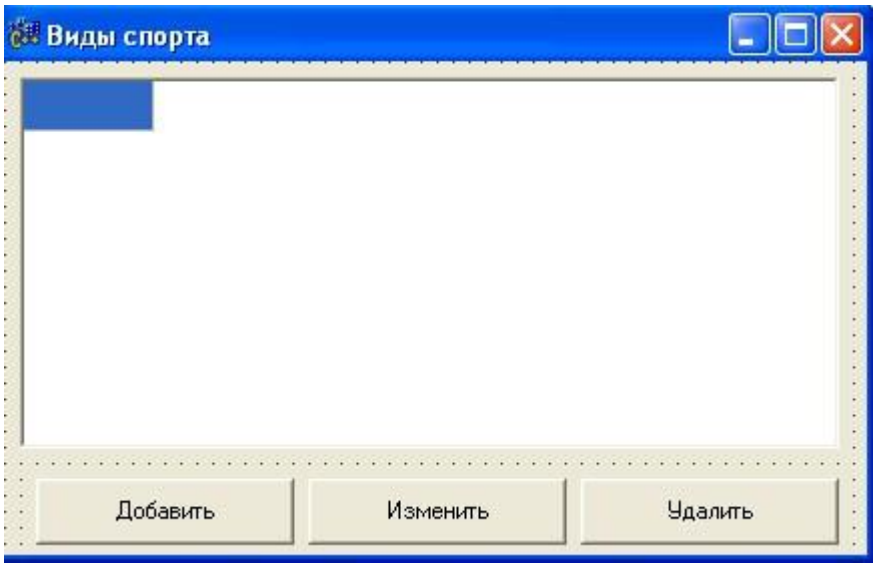


Рисунок 3.12 - Внешний вид формы «Поиск данных»

Внешний вид формы, предназначенной для просмотра данных и изменения данных таблицы «Расписание тренировок» показан на рисунке 3.13

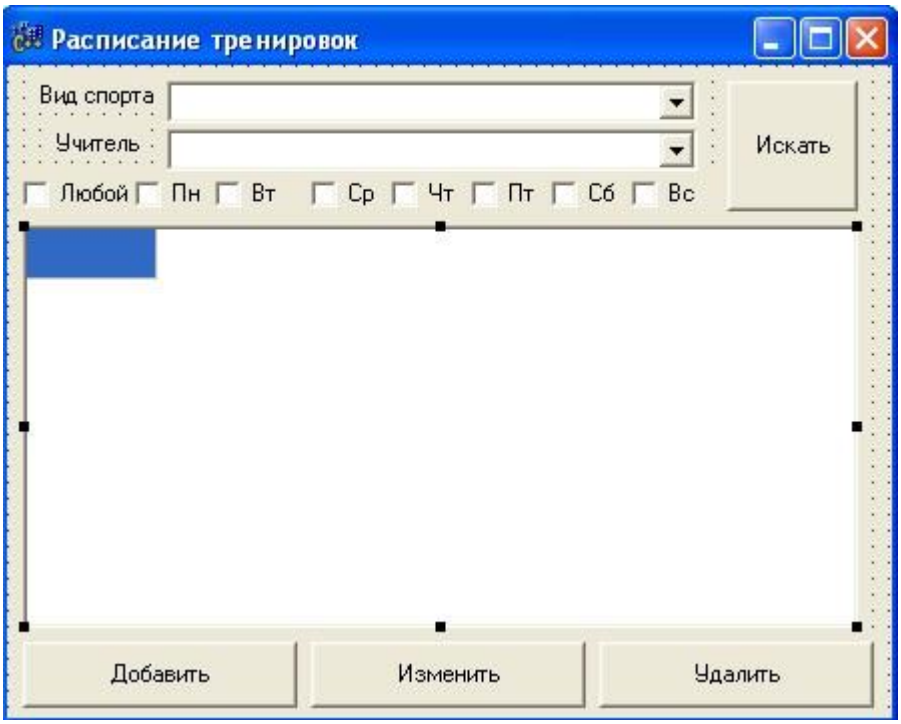


Рисунок 3.13 - Внешний вид формы «Добавление данных»

Внешний вид формы, предназначенной для просмотра данных и изменения данных таблицы «Соревнования «показан на рисунке3.14

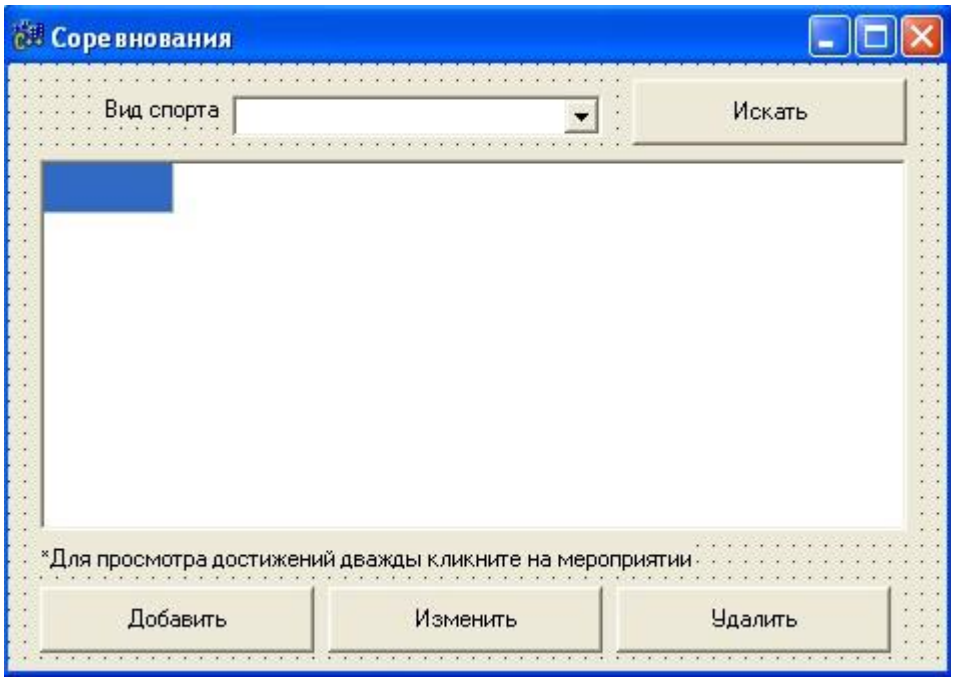


Рисунок 3.14 - Внешний вид формы «Удаление данных»

Внешний вид формы, предназначенной для просмотра данных и изменения, данных таблицы «Школьники» показан на рисунке 3.15

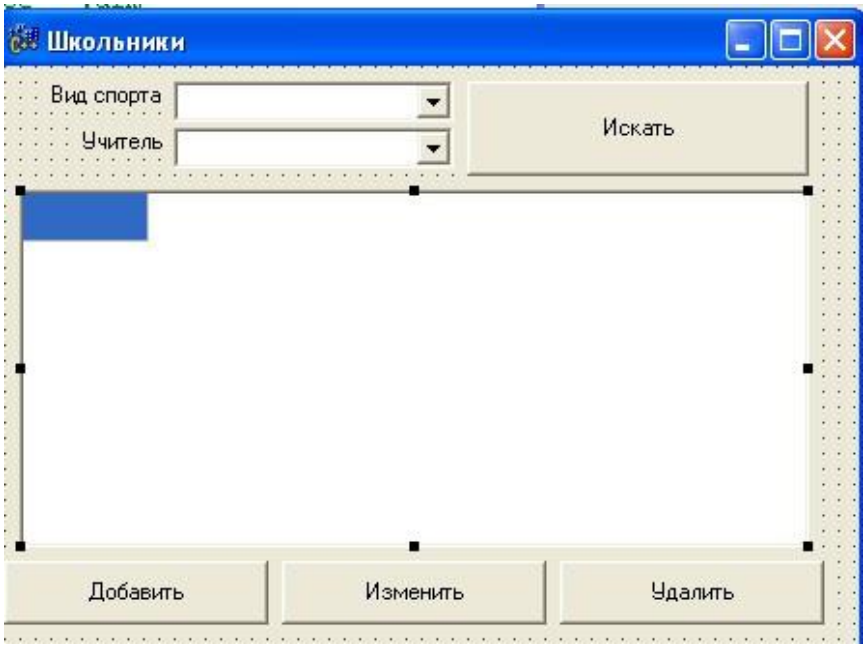


Рисунок 3.15 - Внешний вид формы «Изменение данных»

Свойства компонента IBDatabase 1, расположенного в DataModule8, показаны на рисунке 3.16. [20]

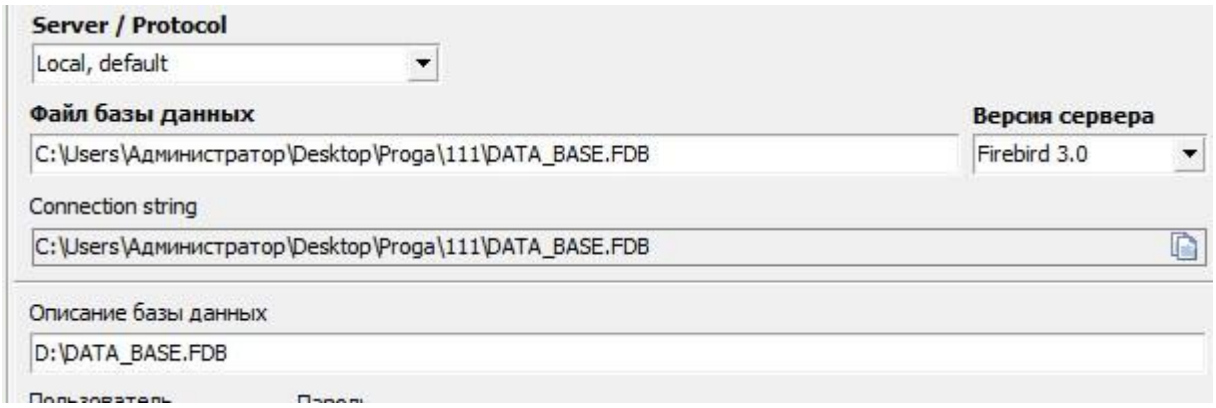


Рисунок 3.16 - Свойства компонента IBDatabase 1

3.4 Описание контрольного примера реализации проекта

3.4.1 Реализация просмотра данных БД

Составили программу, с помощью которой можно просматривать содержимое таблиц базы данных. При этом использовали ранее созданные просмотры.

Код написанной программы приведен в приложении Б.

Пример работы данной программы можно посмотреть на рисунке 3.17. Мы осуществляем просмотр таблицы «Учителя»

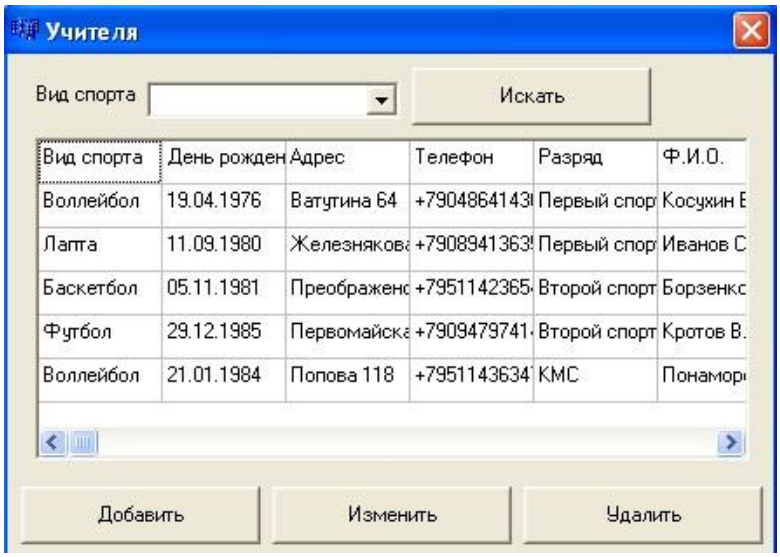


Рисунок 3.17 - Просмотр данных таблицы Учителя

Так же можно просматривать информацию о школьниках, расписании тренировок и соревнований. Возможно сортировка данных по соответствующим полям.

3.4.2 Реализация поиска данных БД

Создали программу, с помощью которой можно осуществлять поиск данных по заданным критериям. [6]

Код написанной программы приведен в приложении Б.

Пример работы данной программы можно посмотреть на рисунке3.18. Мы осуществляем поиск учителя по виду спорта.

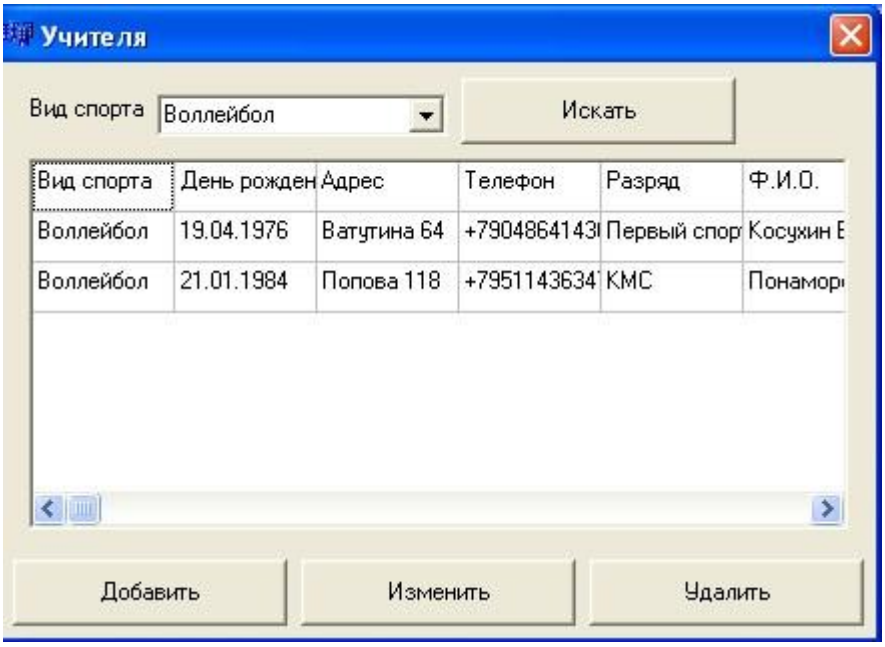


Рисунок 3.18 - Поиск данных по критерию «Волейбол»

3.4.3 Реализация добавления данных в БД

Составили программу, с помощью которой можно осуществлять добавление данных в таблицы. [14] При этом использовали ранее созданные хранимые процедуры.

Код написанной программы приведен в приложении Б.

Пример добавления данных в таблицу «Учителя» представлен на рисунке 3.19

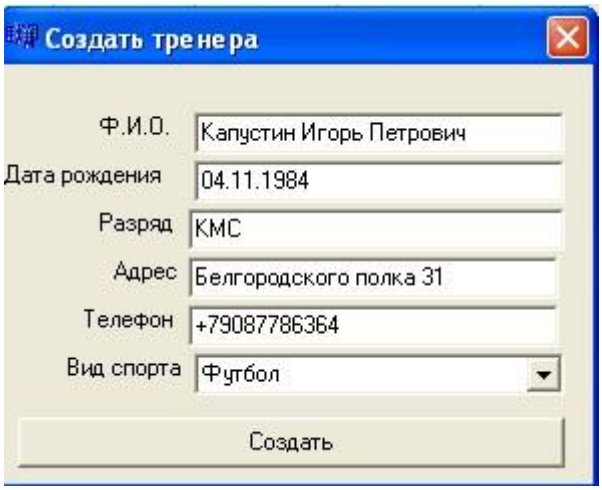


Рисунок 3.19 - Добавление данных

Результат добавления данных в таблицу можно посмотреть на рисунке 3.20

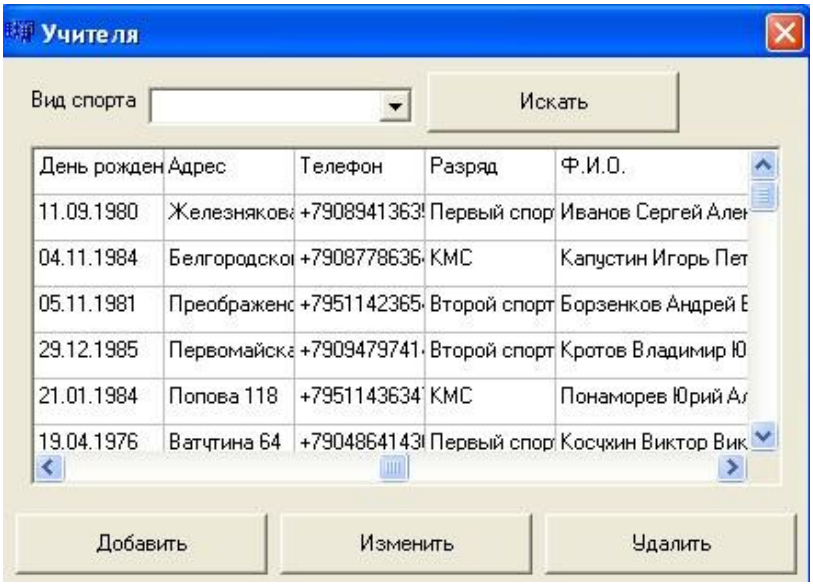


Рисунок 3.20 - Результат добавления данных

3.4.4 Реализация удаления данных из БД

Составили программу, с помощью которой можно осуществлять удаление данных.

Код написанной программы приведен в приложении Б.

Реализовали возможность каскадного удаления данных.

Пример удаления данных школьника представлен на рисунке 3.21. Мы удаляем спортсмена с фамилией «Ефанов»

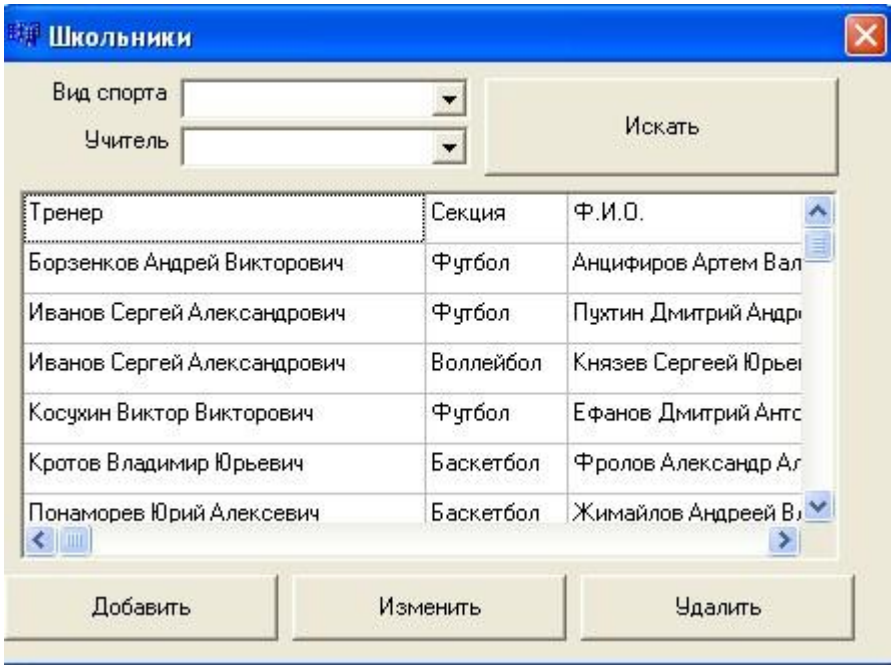


Рисунок 3.21 - Удаление данных

Результат, полученный после удаления данных можно посмотреть на рисунке 3.22

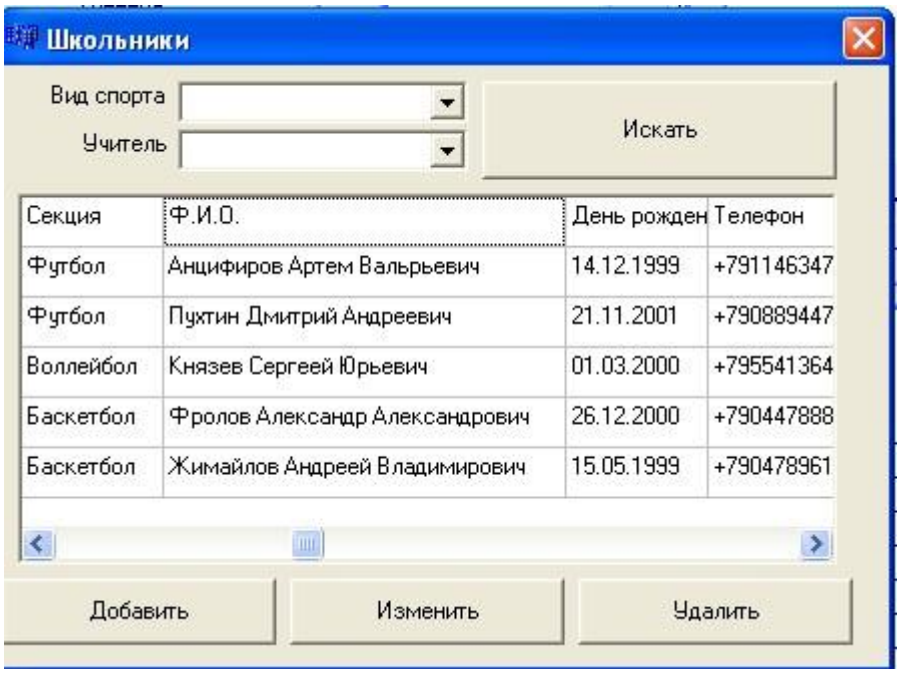


Рисунок 3.22 - Результат удаления данных

3.4.5 Реализация изменения данных в БД

Составили программу, с помощью которой можно осуществлять изменение данных.

Данные можно изменять. Необходимая информация корректируется в программе, а затем автоматически сохраняется на сервере.

Код написанной программы приведен в приложении Б.

Пример изменения данных в программе можно посмотреть на рисунке3.23 Мы изменяем место проведения соревнований.

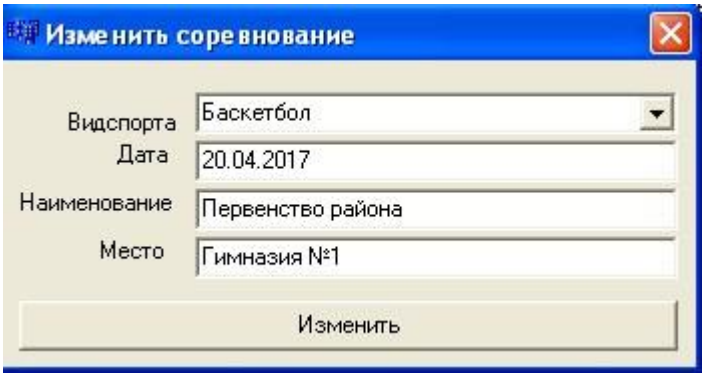


Рисунок 3.23 - Изменение данных

3.4.6 Реализация фильтрации данных в БД

Составили программу, с помощью которой можно осуществлять фильтрацию данных.

Данные можно фильтровать. Необходимая информация корректируется в программе, а затем автоматически сохраняется на сервере.

Код написанной программы приведен в приложении Б.

Пример фильтрации данных в программе можно посмотреть на рисунке 3.24 Мы фильтруем расписание тренировок по дням недели.

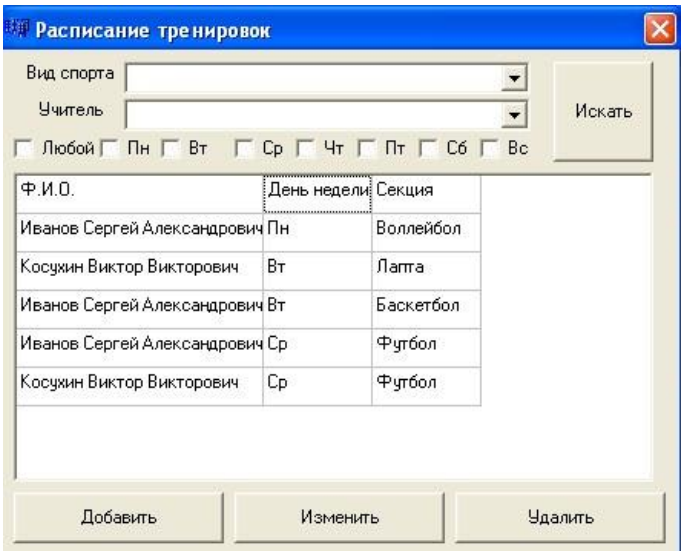


Рисунок 3.24 - Фильтрация данных

Результат, полученный после фильтрации данных можно посмотреть на рисунке 3.25

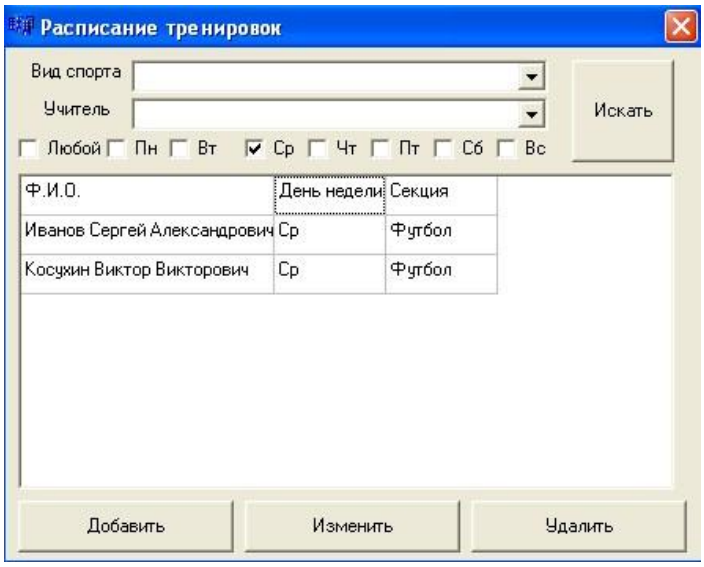


Рисунок 3.25 - Результат фильтрации данных

3.4.6 Создание отчетов в БД

Составили программу, с помощью которой можно осуществлять создание отчетов.

Код написанной программы приведен в приложении Б.

Пример создания отчета в программе можно посмотреть на рисунке3.26. Мы создаем отчет по всем соревнованиям.

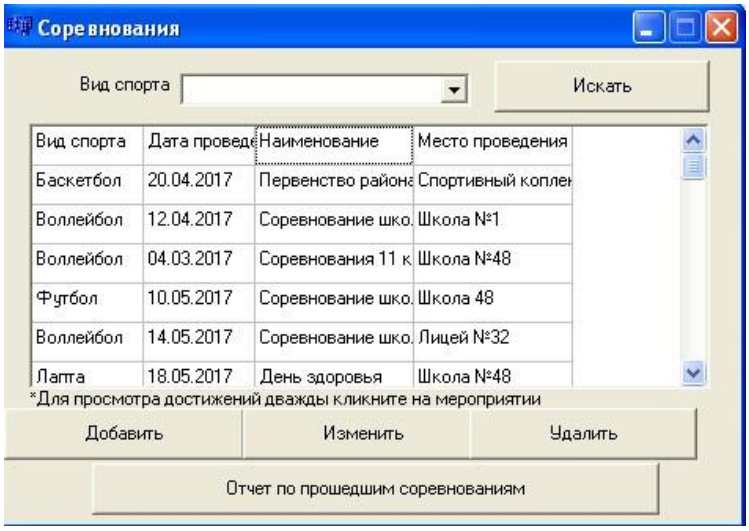


Рисунок 3.26 - Формирование отчета

Результат, полученный после формирования отчета можно посмотреть на рисунке 3.27

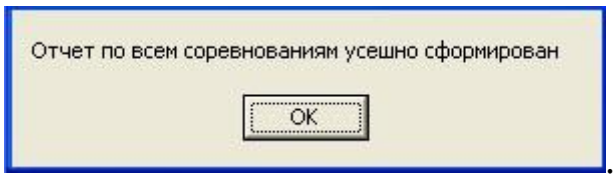


Рисунок 3.27 - Результат формирования отчета

Отчет, сохраненный в отдельный документ можно посмотреть на рисунке 3.28

```
Отчет по соревнованию 'Первенство района по баскетболу'
        Фролов Александр Александрович  |       3 место        |
        Анцифиров Артем Вальрьевич      |       3 место        |
        Кушин Андрей Сергеевич          |       3 место        |
        Павлов Игорь Михайлович         |       3 место        |
        Игнатьев Дмитрий Юрьевич        |       3 место        |

Отчет по соревнованию 'Соревнования 11 классов по воллейболу'
        Ефанов Дмитрий Антонович        |       1 место        |
        Князев Сергеей Юрьевич          |       1 место        |
        Пухтин Дмитрий Андреевич        |       1 место        |
        Анцифиров Артем Вальрьевич      |       1 место        |
        Фролов Александр Александрович  |       1 место        |

Отчет по соревнованию 'Соревнование школ по футболу'
        Мухин Владимир Викторович       |       5 место        |
        Князев Сергеей Юрьевич          |       5 место        |
        Игнатьев Дмитрий Юрьевич        |       5 место        |
        Терентьев Сергеей Сергеевич     |       5 место        |
        Жимайлов Андреей Владимирович   |       5 место        |
|
```

Рисунок 3.28 - Созданный файл отчета

Полный функционал работы подсистемы представлен в приложении А

3.5 Оценка экономической эффективности проекта

В данной работе была разработана информационная подсистема для МБОУ СОШ №48. Результатом решения этой задачи будет являться программное обеспечение современных ЭВМ, которое должно обеспечивать достаточно высокую скорость обработки информации. Целью данного раздела является расчет затрат. В результате расчета находится затраты на прикладную программу.

Для нахождения затрат необходимо учесть:

– амортизационные отчисления на полное восстановление технических средств и программного обеспечения;

- трудоемкость разработки программного продукта;
- оплату труда программиста;
- доплаты и надбавки к заработной плате;
- затраты электроэнергии, расходуемой техническими средствами;
- накладные расходы;
- единый социальный налог.

Расчет амортизационных отчислений.

В работе необходима следующая смета затрат на оборудование:

Таблица 3.2 – Смета затрат на оборудование

Наименование	Единица измерения	Количество	Цена за единицу	Сумма, руб
1	2	3	4	5
AMD совместимый компьютер	шт.	5	18000	90000
Acer монитор	шт.	5	7000	35000
Клавиатура BTC	шт.	5	1000	5000
Мышка 4Tech	шт.	5	300	1500
Итого:				131500

Закупочная цена персональных компьютеров -90000 руб., мониторов 35000 руб., клавиатуры -5000 руб., мышки - 1500 Примем норму амортизации на технические средства 20 %.

Общая стоимость технических средств, руб.:

$$Цтс=Цк+Цм, \quad (1)$$

где Цк – цена компьютера;

Цм – цена монитора.

Отсюда:

Цтс=9000+35000+5000+1500=131500 руб.

Для создания пакета программ, являющегося конечным результатом исследований, применялось следующее программное обеспечение:

- операционная система Windows 7 – 3000 руб.;
- Builder C++ – 4000 руб.

Общая стоимость программного обеспечения составляет 7000 руб.

Общая стоимость технических средств и программного обеспечения, руб.:

$$Цо=Цтс+Цпо, \quad (2)$$

Цо=131500+7000=138500 руб.

Годовые амортизационные отчисления на полное восстановление технических средств и программного обеспечения рассчитываются по формуле, руб.:

$$Ао=Цо \cdot На, \quad (3)$$

Ао=138500·0,2=27700.

Амортизационные отчисления за период создания программного продукта, руб.:,

$$A_П = \frac{A_0 * K_{ДН}}{K_{РГ}} \quad (4)$$

где Кдн=55 дн. – количество отработанных дней;

Крг=280 дн. – количество рабочих дней в году.

$A_П = \frac{27700*55}{280} = 5441$ руб.

Расчет расходов на энергопотребление

ПЭВМ, на которой была разработана программа, является потребителем электрической энергии сети переменного тока, напряжением 220 В. Согласно технической документации, суммарная мощность, потребляемая компьютером и монитором, составляет:

Мс= 250 Вт·ч.

Расход денежных средств, связанный с энергопотреблением технических средств можно найти по формуле, руб.:

$$Рэ=Кдн \cdot Враб \cdot Мс \cdot Цэн, \quad (5)$$

где Кдн – период написания программы, дн., Кдн=55 дней;

Враб – длительность рабочей смены, ч., Враб=6 часов;

Мс – мощность, потребляемая техническими средствами, кВт·ч;

Цэн – стоимость электроэнергии по действующим тарифам, р./кВт·ч; Цэн=1,5 рубля за кВт·ч.

Отсюда:

Рэ=55·6·0,25·1,5 ≈ 123,75 руб.

Расчет заработной платы программиста

Исходя из фактически отработанного времени программиста, которое составило 55 рабочих шестичасовых дней, найдем количество фактически отработанного времени, ч.:

$$Тф=Кдн·Враб, \quad (6)$$

где Тф – фактически отработанное время, ч.;

Кдн – количество отработанных дней, дн;

Враб – продолжительность рабочего дня, ч.

Тф=55·6=330 руб.

Принимая часовую заработную плату программиста в расчете 30 руб., получим основную заработную плату, руб.:

$$Зосн=Тф·Тч, \quad (7)$$

где Тч – часовая тарифная ставка программиста.

Зосн=330·30=9900 руб.

Для определения общей суммы расходов на оплату труда необходимо учесть доплаты и надбавки. Принимаем удельный вид доплат и надбавок в размере 15 % от основной заработной платы, руб.:

$$Удоп=Зосн·0,15 , \quad (8)$$

Удоп=9900 ·0,15=1485 руб.

Отсюда находим общие расходы на оплату труда, руб.:

$$Робщ=Зосн+Удоп, \quad (9)$$

Робщ=9900+1485=11385 руб.

Далее определим единый социальный налог; в соответствии со ставкой он составляет 26% от расходов на оплату труда, что составит, руб.:

$$Осоц=Робщ·0,26 \quad (10)$$

Осоц=11385 ·0,26≈ 2960 руб.

Необходимо учесть накладные расходы, отображенные в таблице 3.3

Таблица 3.3 - Накладные расходы

Материалы	Единица измерения	Количество	Цена за единицу	Сумма, руб
1	2	3	4	5
Бумага офисная	пачка	1	200	200
CD-RW	шт.	2	30	60
Ручка	шт.	1	10	10
Папка	шт.	1	50	50
Материалы	Единица измерения	Количество	Цена за единицу	Сумма, руб
Итого:				320

Расчет общих затрат на создание пакета программ

Итого, затраты на создание пакета программ, составляют, руб.:

$$Т=Ап+Рэ+Робщ+Осоц+Рнак, \quad (11)$$

Т=5441+123,75+11385+2960+220 ≈ 20129 руб.

Определение отпускной цены программы

Для нахождения отпускной цены необходимо учесть:

– прибыль от реализации 15 %;

– налог на добавленную стоимость 18 %.

$$Цн=Т+Пн, \quad (12)$$

где Пн – прибыль от реализации продукта, руб.

$$Пн=Т\cdot 0,15, \quad (13)$$

Цн=20129 +20129·0,15 ≈ 23148 руб.

Цена пакета программ с учетом НДС, руб.:

$$Цп=Цн+НДС, \quad (14)$$

где НДС=Цн·0,18 – налог на добавленную стоимость.

Цп=23148 +23148 ·0,18 ≈ 27314 руб.

Итак, отпускная цена программы составляет 27314 руб.

Расчет годовых затрат на эксплуатацию программы

Расчет годовых затрат необходимо провести для последующего анализа эффективности данного программного продукта.

Годовые затраты на эксплуатацию программы составляют:

$$C_{Р.М.ГОД} = n * C_{Р.М.} * E_n * C \quad (15)$$

где $C_{Р.М.ГОД}$ – стоимость одного непосредственного решения на ЭВМ, руб.;

E_n - нормативный коэффициент сложности (0.2–0.5);

C – себестоимость разработанной программы;

n – плотность потока заявок = 10 з/год.

Стоимость одного непосредственного решения определяется, руб.:

$$C_{Р.М.} = C_{М.Ч.} * T_р * ЗП_0^{ПР} * \mathrm{N_Р} * \mathrm{K_Р} * \mathrm{P_К} \quad (16)$$

где $C_{Р.М.}$. – стоимость одного часа работы на ЭВМ;

$T_р$ – время решения задачи на ЭВМ;

$\mathrm{N_Р}$ – трудоемкость программиста , затраченная на решение задачи на ЭВМ (50 ч.);

$\mathrm{K_Р}$ – районный коэффициент (1.3);

$\mathrm{P_К}$ – расходы косвенные.

$C_{Р.М.}$. = 8 * 50 + 30 * 50 * 1,3 * 1.1 = 2545руб.

Ср.м.год. = 10 * 2545 + 0.3 * 14264 = 29729 руб.

Для реализации подсистемы была подведена общая смета затрат.

Таблица 3.4 - Общая смета затрат.

Наименование	Сумма, руб
1	2
Материальные затраты	131790
Затраты на создание программы	20129
Затраты на эксплантацию	29729
Амортизационные отчисления	27700
Затраты на зарплату программисту	11385
Социальный налог	2960
Итого:	193964 руб.

Расчет срока окупаемости. Срок окупаемости затрат является простейшим и часто используется для оценки экономической эффективности проекта. По определению, период окупаемости – это минимальный интервал времени, за пределами которого интегральный эффект от реализации проекта становится положительным и в дальнейшем не уходит в минус.

$$PB = \frac{IC}{CF_{CP}} \quad (17) \qquad (16)$$

IC – общая стоимость затрат в проект;

CFcp – среднегодовой финансовый поток от реализации проекта.

$PB = \frac{193964}{5000} \approx 3$ года.

Расчет годовых затрат на выполнение работ ранее употреблявшимся способом

Сравнительным вариантом является такой, при котором расчеты выполняются, как правило, вручную. Для определения затрат для ручной обработки данных необходимо иметь данные по квалификации специалиста и затраты времени при числовой обработке. При отсутствии данных затраты ручной обработки рассчитываются путем хронометража работы программистом при выполнении расчетов по данному способу. Для нахождения годовых затрат на выполнение расчетов ранее употреблявшимся способом необходимо знать стоимость всех работ по выполнению одного расчета. Она составит, руб.:

$$C_{РСП} = ЗП_{СП} * T_{СП} * K_{Р} * P_{Н} \qquad (17)$$

где $C_{РСП}$. – стоимость выполненных расчетов ранее употреблявшимся способом;

$ЗП_{СП}$ – зарплата специалиста , который проводил расчеты ранее употреблявшимся способом;

$T_{СП}$ – затраты времени специалиста на один расчет (150 ч.);

$K_{Р}$ – районный коэффициент (1.3);

$P_{Н}$ – накладные расходы

$C_{РСП}$ =30 * 150 * 1.3 * 1,1 = 6435 руб.

Зная плотность потока заявок по данному расчету определяем годовые затраты, руб.:

$$C_{Р.СП.ГОД} = n * C_{Р.СП} \qquad (18)$$

$C_{Р.СП.ГОД}$ = 10 * 6435 = 64350 руб.

Таким образом, в результате проведения расчетов в экономической эффективности выпускной квалификационной работы были определенны: расчет амортизационных отчислений, расчет расходов на энергопотребление, расчет общих затрат на создание пакета программ , расчет годовых затрат, расчет срока окупаемости.

ЗАКЛЮЧЕНИЕ

В ходе выполнения практической работы бы произведен анализ деятельности учебного учреждения, в результате чего был предложен вариант автоматизации процессов в МБОУ СОШ №48.

При выполнении работы были рассмотрен программный продукт AllFusionProcessModeller 7, который и использовался для построения моделей «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

Для решения поставленной задачи в ходе преддипломной производственной практики выполнены следующие подзадачи:

- изучение и анализ деятельности учреждения, для которой будет разрабатываться подсистема;
- выявление недостатков;
- проектирование моделей «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

При формировании требований к системе были выдвинуты условия, которые необходимо было учесть при разработке данного программного продукта, а именно, подсистема должна обеспечивать более эффективные и продуктивные взаимодействия учителей и школьников, учителей и методического объединения.

Автоматизация подсистемы учета спортивных достижений основана на внедрении программного обеспечения в деятельность обучающих специалистов на всех этапах подготовки спортсменов.

Проанализировав созданные информационные условия, пришли к выводу, что средства поиска, получения, хранения, обработки информации используются не в полной мере. Разработанная проектная система предоставит возможность:

- автоматизировать операции, происходящие над документами;
- создания, ведения и хранения документов не только на бумажном носителе, но и в электронном виде;
- усовершенствования процесса поиска необходимой информации.

Проанализировав существующее в школе аппаратно-программное обеспечение, пришли к выводу:

- техническое оснащение рабочих мест предполагает внедрение разрабатываемого программного комплекса;
- не требуется дополнительных денежных вложений на закупку новых лицензионных версий программного обеспечения.

Разработаны следующие функции подсистемы управления спортивными достижениями:

- добавления, удаления, изменения информации;
- поиск по заданным критериям;
- формирование отчетов.

В результате проделанной работы была разработана подсистема учета спортивных достижений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. О персональных данных [Текст] Федеральный закон Российской Федерации от 27 июля 2006 г. № 152-ФЗ. //Редакции федеральных законов. – 2007. - № 5, (4 февр.). – С. 1485 – 1498 (ст. 375)

2. Об информации, информационных технологиях и о защите информации [Текст] Федеральный закон Российской Федерации от 27 июля 2006 г. № 149-ФЗ. // Редакции федеральных законов– 2007. - – С. 1425 – 1448 (ст. 375)

3. Распоряжение от 07.08.2009 № 1101-р Об утверждении Стратегии развития физической культуры и спорта в Российской Федерации на период до 2020 года [Текст], распоряжение Правительства РФ от 07.08.2009.

4. Титоренко, Г.А. Автоматизированные информационные технологии в экономике [Текст] / Г.А.Титоренко. – Москва: ЮНИТИ, 2013. – 400 с.

5. Архангельский, А.Я. Программирование C++Builder 6 [Текст] / А.Я.Архангельский. – Москва: Бином, 2015. – 1162 с.

6. Борри, Х. Firebird: руководство разработчика баз данных /Х.Борри.– Санкт-Петербург: БХВ-Петербург, 2012. – 1104 с.

7. Вендров, А.М. CASE-технологии современные методы и средства проектирования информационных систем [Текст] / А.М.Вендров. – Москва: Финансы и статистика, 2008. –176 с.

8. Гордеев, А. В. Операционные системы [Текст] / А.В.Гордеев. – Санкт-Петербург: Питер, 2014. –416 с.

9. Дейт, К. Введение в системы баз данных [Текст] / К. Дж. Дейт. – Москва: Вильямс, 2014. – 1328 с.

10. Киселев, С. В. Аппаратные средства персонального компьютера [Текст] / С.В.Алексахин.– Москва: Академия, 2014. – 64 с.

11. Когаловский, М. Р. Перспективные технологии информационных систем [Текст] / М.Р.Когаловский.–Москва: ДМК Пресс,2013. – 288 с.

12. Культин, Н. Б. Самоучитель C++Builder [Текст] / Н.Б.Культин. – Санкт-Петербург: БХВ-Петербург, 2014. – 320 с.

13. Маклаков, С.В. BPWinи ERWin: CASE-средства разработки информационных систем [Текст] / С.В.Маклаков.– Москва: ДИАЛОГ-МИФИ, 2010. – 254 с.

14. Маклаков, С. В. Моделирование бизнес-процессов с AIIFusionProcessModeler [Текст] / С.В.Маклаков. - Москва: ДИАЛОГ-МИФИ, 2014. - 240 с.

15. Семакин, И. Г. Основы программирования [Текст] / И.Г.Семакин, А.П. Шестаков.– Москва: Академия, 2013. – 432 с.

16. Фуфаев, Э. В. Базы данных [Текст] / Э.В.Фуфаев, Д.Э.Фуфаев.– Москва: Академия, 2011. – 320 с.

17. Харченко, К. В. Муниципальное стратегическое планирование: от теории к технологии [Текст] / К.В. Харченко – Белгород: Белгородская областная типография, 2009. – 304 с.

18. Холингворт, Д. BorlandC++ Builder 6. Руководство разработчика [Текст] / Д. Холингворт, Б. Сворт, М. Кэшмэн, П. Густавсон.– Москва: «Вильямс», 2014. – 976 с.

19. Шень, А. Программирование: теоремы и задачи [Текст] / А.Шень.– Москва: МЦНМО, 2015.–296с.

20. Алексеев, В.Е. Вычислительная техника и программирование. Практикум по программированию [Текст] / В.Е.Алексеев.– Москва: ВШ, 2011 – 408 с.

21. Синицын, С.В Программирование на языке высокого уровня [Текст] /С.В. Синицын. – Москва: Академия, 2010– 355 с.

22. Сергиевский, Г.М Функциональное и логическое программирование [Текст] / Г.МСергиевский. – Москва: Академия, 2010– 320 с.

23. Павловская, Т.А. C/C++. Структурное и объектно-ориентированное программирование [Текст] /Т.А.Павловская. –Санкт-Петербург: Питер, 2010 – 352 с.

24. Линев, А.В Технологии параллельного программирования для процессоров новых архитектур [Текст] /А.В. Линев. – Москва: Московский университет, 2010. – 160 с.

25. Гуда, А.Н Информатика и программирование [Текст] / А.Н. Гуда. – Москва: Дашков и К, 2012. – 240 с.

26. Новожилов, О.П. Электротехника и электроника [Текст] / О.П.Новожилов.– Москва :Юрайт, 2013. – 653 с

27. Уткин, В.Б Информационные системы в экономике [Текст] / В.Б. Уткин.– Москва: Академия, 2010. – 395 с.

28. Гагарина, Л.Г. Технология разработки программного обеспечения [Текст] /Л.Г. Гагарина. – Москва: ИНФРА-М, 2011. – 400 с.

29. Голицина, О.Л. Программирование на языках высокого уровня [Текст] /О.Л. Голицина. – Москва: ФОРУМ, 2011. –496 с.

30. Голицына, О.Л. Программирование на языках высокого уровня [Текст] / О.Л.Голицына. – Москва: Форум, 2011. – 496 с.

31. Илюшечкин, В.М. Основы использования и проектирования баз данных: учебник для вузов [Текст] / В.М. Илюшечкин. – Москва :Юрайт, 2011. – 2013 с.

32. Волгина, О.А. Математическое моделирование экономических процессов и систем [Текст] / О.А. Волгина. –Москва: КноРус, 2011. –200с.

33. Назаров, С.В. Современные операционные системы [Текст] / С.В. Назаров. –Москва: БИНОМ Лаборатория знаний, 2013. –367 с.

34. Назаров, С.В. Операционные системы [Текст] / С.В.Назаров.– Москва:КноРус, 2012. – 376с.

35. Заботина, Н.Н. Проектирование информационных систем [Текст] / Н.Н. Заботина. –Москва: НИЦ Инфра-М, 2013. –331 с.

36. Сухарев, А. Г. Курс методов оптимизации: Учеб.пособие [Текст] / А. Г. Сухарев, А. В. Тимохов, В. В. Федоров. – Москва: ФИЗМАТЛИТ, 2011. – 384 с.

37. Основы проектирования реляционных баз данных. [Электронный ресурс], - Режим доступа: http://www.intuit.ru/goods_store/ebooks/8322, свободный.

38. Ресурсы информационных систем. [Электронный ресурс], - Режим доступа: http://www.economica-upravlenie.ru/content/view/204/, свободный.

39. Обзор ERP-систем [Электронный ресурс] : Независимый ERP портал, – Режим доступа: www.erp-online.ru/, свободный.

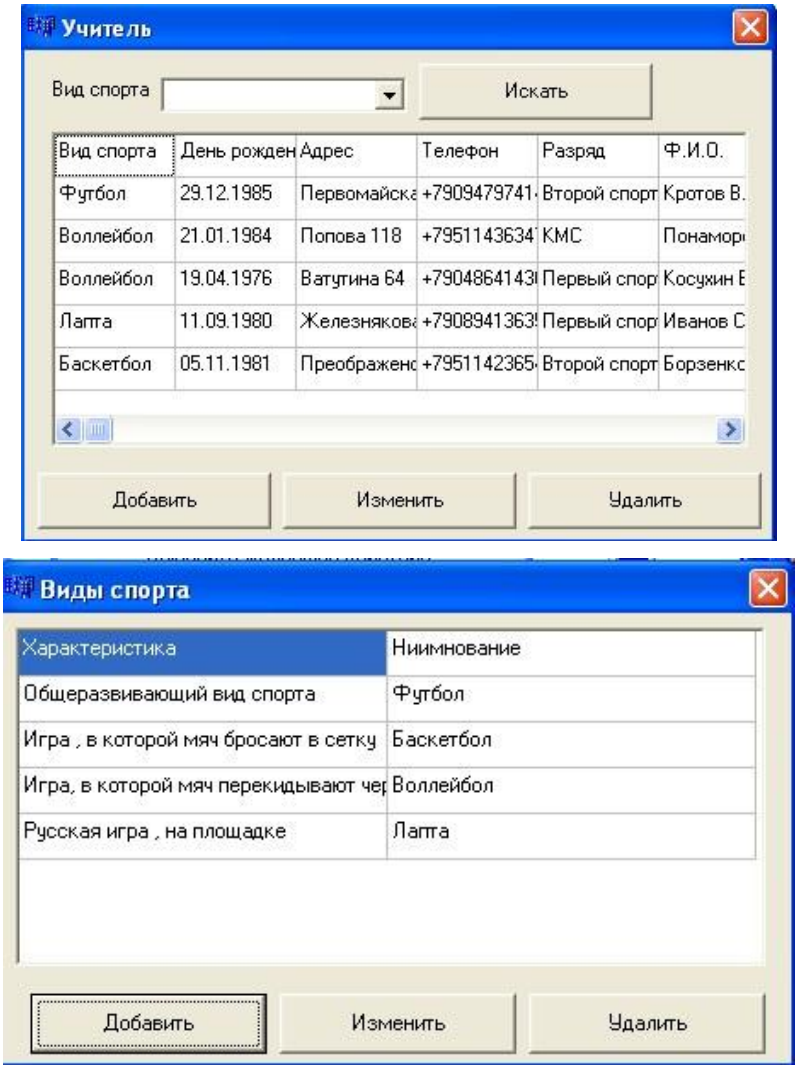
40. Грекул, В.И. Проектирование информационных систем [Электронный ресурс] / В.И. Грекул, Интернет университет информационных технологий, – Режим доступа: http://www.intuit.ru/department/se/devis/9/, свободный.

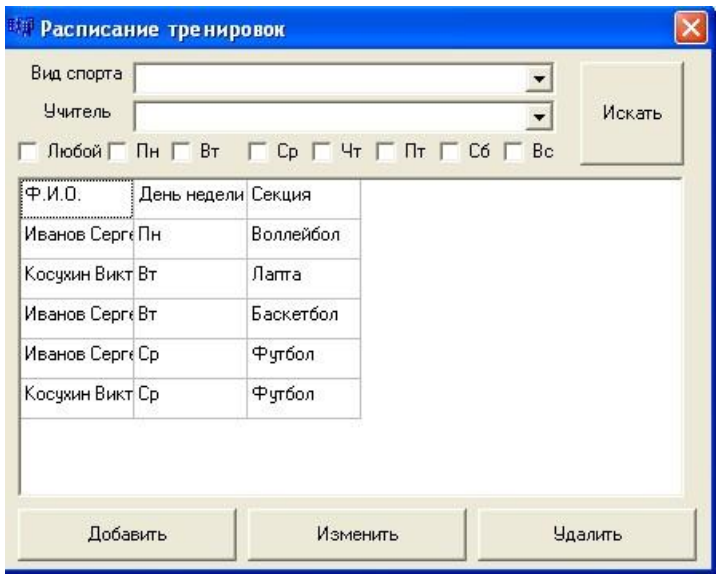
41. Мезенцев, К.Н. Автоматизированные информационные системы: [Текст] / К.Н. Мезенцев. – Москва: Академия, 2012. – 174 с.

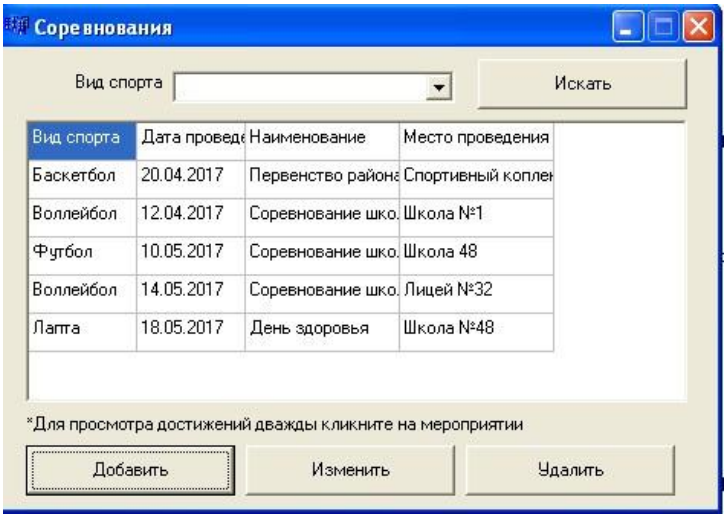
42. Маклаков С.В. BPwin и Erwin. Case – средства разработки информационных систем [Текст] / С.В. Маклаков – Москва: ДИАЛОГ – МИФИ, 2013. – 230 с.

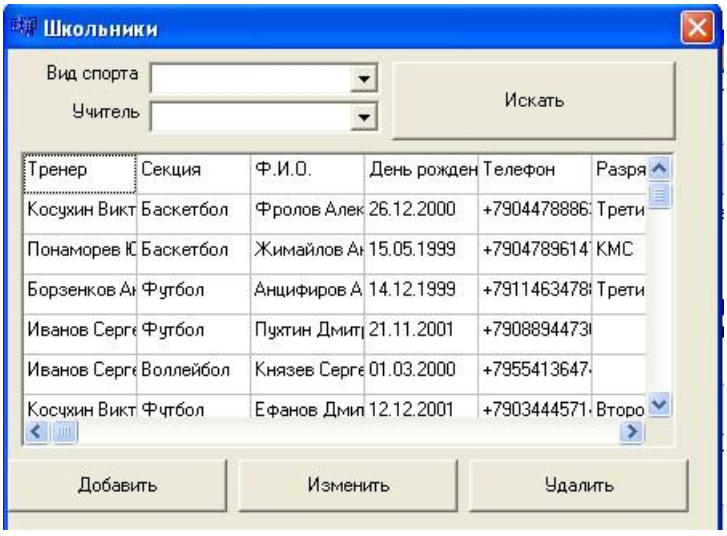
ПРИЛОЖЕНИЕ **А**

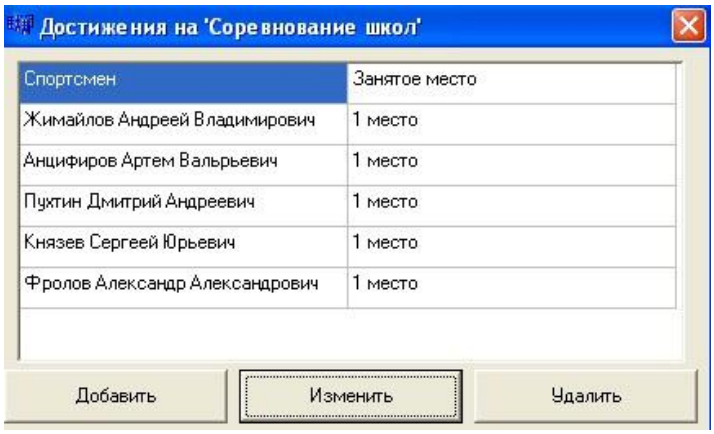
Подсистема управления спортивными достижениями

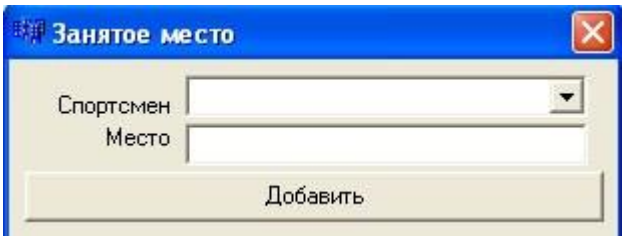


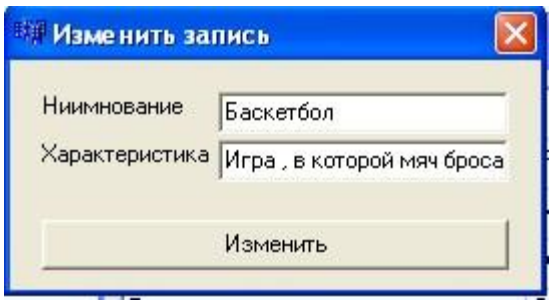

Расписание тренировок
Вид спорта
Учитель
Искать
Любой Пн Вт Ср Чт Пт Сб Вс
Ф.И.О. День недели Секция
Иванов Серге Пн Воллейбол
Косухин Викт Вт Лапта
Иванов Серге Вт Баскетбол
Иванов Серге Ср Футбол
Косухин Викт Ср Футбол
Добавить
Изменить
Удалить

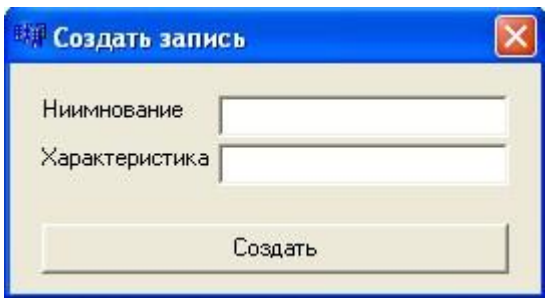

Соревнования
Вид спорта
Искать
Вид спорта Дата провед Наименование Место проведения
Баскетбол 20.04.2017 Первенство района Спортивный коплек
Воллейбол 12.04.2017 Соревнование шко. Школа №1
Футбол 10.05.2017 Соревнование шко. Школа 48
Воллейбол 14.05.2017 Соревнование шко. Лицей №32
Лапта 18.05.2017 День здоровья Школа №48
*Для просмотра достижений дважды кликните на мероприятии
Добавить
Изменить
Удалить


Школьники
Вид спорта
Учитель
Искать
Тренер Секция Ф.И.О. День рожден Телефон Разря
Косухин Викт Баскетбол Фролов Алек 26.12.2000 +7904478886 Трети
Понаморев Ю Баскетбол Жимайлов Ан 15.05.1999 +7904789614 КМС
Борзенков Ан Футбол Анцифиров А 14.12.1999 +7911463478 Трети
Иванов Серге Футбол Пухтин Дмитр 21.11.2001 +7908894473
Иванов Серге Воллейбол Князев Серге 01.03.2000 +7955413647
Косухин Викт Футбол Ефанов Дмит 12.12.2001 +7903444571 Второ
Добавить
Изменить
Удалить

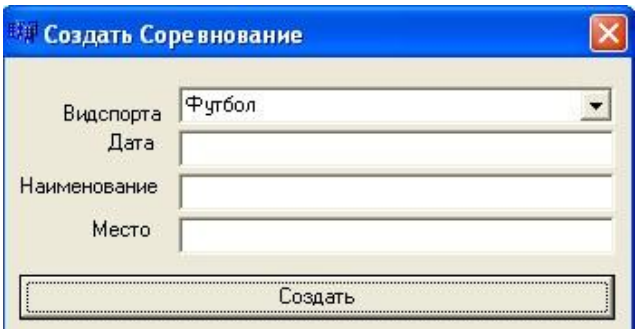

Достижения на 'Соревнование школ'
Спортсмен
Занятое место
Жимайлов Андреей Владимирович
1 место
Анцифиров Артем Вальрьевич
1 место
Пухтин Дмитрий Андреевич
1 место
Князев Сергеей Юрьевич
1 место
Фролов Александр Александрович
1 место
Добавить
Изменить
Удалить

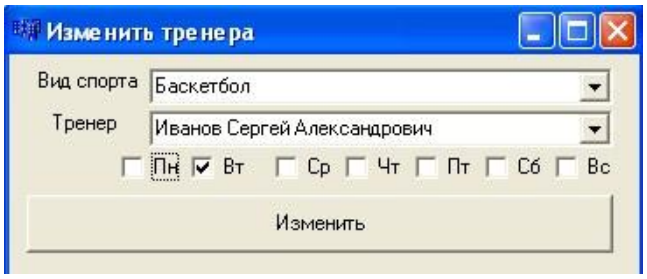

Занятое место
Спортсмен
Место
Добавить

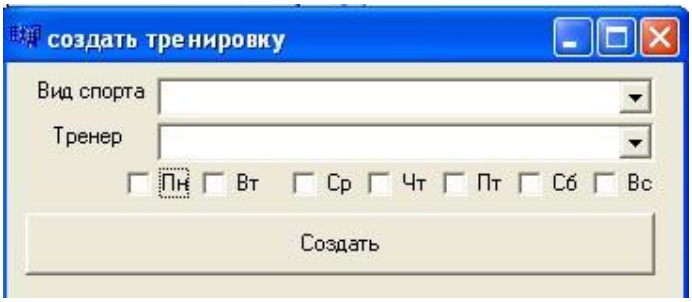

Изменить запись
Ниимнование
Баскетбол
Характеристика
Игра , в которой мяч броса
Изменить

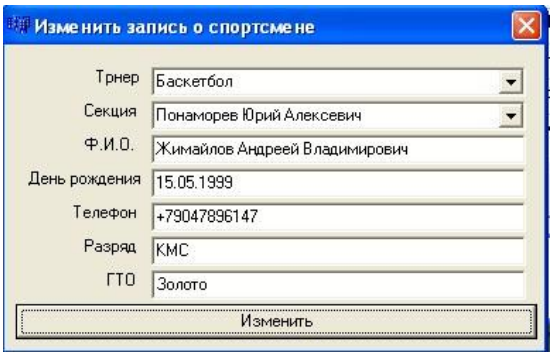

Создать запись
Ниимнование
Характеристика
Создать

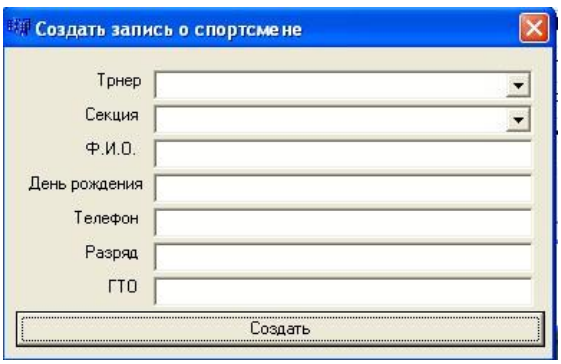

Изменить соревнование
Видспорта
Воллейбол
Дата
12.04.2017
Наименование
Соревнование школ
Место
Школа №1
Изменить

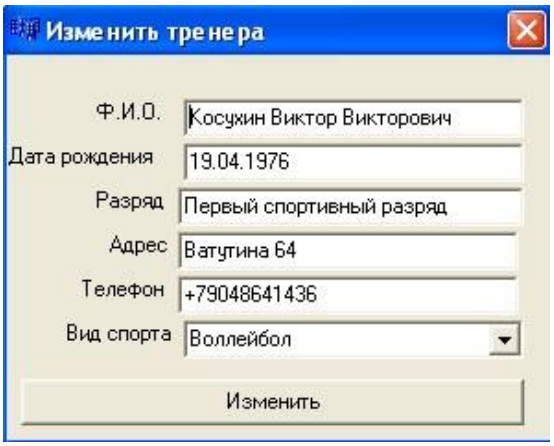

Создать Соревнование
Видспорта
Футбол
Дата
Наименование
Место
Создать

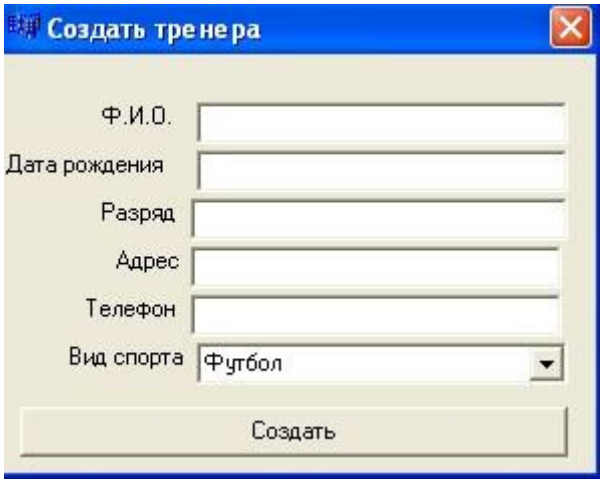

Изменить тренера
Вид спорта
Баскетбол
Тренер
Иванов Сергей Александрович
Пн
Вт
Ср
Чт
Пт
Сб
Вс
Изменить

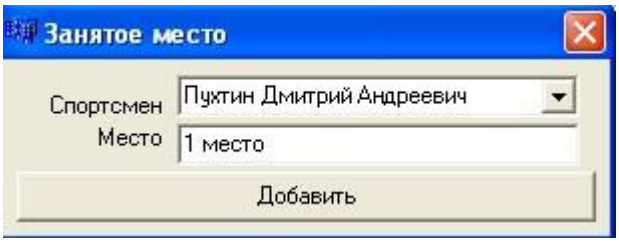

создать тренировку
Вид спорта
Тренер
Пн
Вт
Ср
Чт
Пт
Сб
Вс
Создать


Изменить запись о спортсмене
Трнер
Баскетбол
Секция
Понаморев Юрий Алексевич
Ф.И.О.
Жимайлов Андреей Владимирович
День рождения
15.05.1999
Телефон
+79047896147
Разряд
КМС
ГТО
Золото
Изменить


Создать запись о спортсмене
Трнер
Секция
Ф.И.О.
День рождения
Телефон
Разряд
ГТО
Создать


Изменить тренера
Ф.И.О.
Косухин Виктор Викторович
Дата рождения
19.04.1976
Разряд
Первый спортивный разряд
Адрес
Ватутина 64
Телефон
+79048641436
Вид спорта
Воллейбол
Изменить


Создать тренера
Ф.И.О.
Дата рождения
Разряд
Адрес
Телефон
Вид спорта
Футбол
Создать


Занятое место
Спортсмен
Пухтин Дмитрий Андреевич
Место
1 место
Добавить

ПРИЛОЖЕНИЕ Б

Листинг подсистемы управления спортивными достижениями

```
#include "Unit1.h"
#pragma hdrstop
#include "Unit2.h"
#include "Unit4.h"
#include "Unit5.h"
#include "Unit7.h"
#include "Unit11.h"
#include <vcl.h>
#include "DB.h" //клас работы ЬД
#include "tabelTools.h" //Главный класс таблицы от него наследуются все остальные
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
DB* DataBase;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{      //инициализация параметров базы данных
       DataBase = new DB(Owner);
       DataBase->DBName = "\DATA_BASE.FDB";
       DataBase->usrname = "SYSDBA";

       DataBase->password = "masterkey";
       DataBase->lc_ctype = "win1251";
       DataBase->connect();
       tabelTools::Db = DataBase; //передаем базу класам таблицы
       Part<AnsiString,AnsiString>::Db = DataBase;//передаем базу класам строк
        /*  ListA find;
        find["ID_VID_SPORTA"]="24";
        tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
        ReturnQuery* ret = sprVidov.find(&find);
        ShowMessage(IntToStr(ret->size()));
        for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
              ShowMessage((*(*it))["ID_VID_SPORTA"]);
        }*/

}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    Form2->Show();
    Form2->intTable();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
   Form4->Show();
    Form4->intTable();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button3Click(TObject *Sender)
{
```

```
   Form5->intTable();
  Form5->Show();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button4Click(TObject *Sender)
{
   Form7->Show();
   Form7->intTable();
}
//---------------------------------------------------------------------------


void __fastcall TForm1::Button5Click(TObject *Sender)
{
  Form11->intTable();
  Form11->Show();
}
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit2.h"
#include "Unit3.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm2 *Form2;
//---------------------------------------------------------------------------
__fastcall TForm2::TForm2(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------



void __fastcall TForm2::FormCreate(TObject *Sender)
{
 StringGrid1->ColCount=6;
 StringGrid1->RowCount=1;
 StringGrid1->Cells[0][0]="Вид спорта";
 StringGrid1->Cells[1][0]="День рождения";
 StringGrid1->Cells[2][0]="Адрес";
 StringGrid1->Cells[3][0]="Телефон";
 StringGrid1->Cells[4][0]="Разряд";
 StringGrid1->Cells[5][0]="Ф.И.О.";
  StringGrid1->ColWidths[0] = 70;
 StringGrid1->ColWidths[1] = 70;
 StringGrid1->ColWidths[2] = 70;
 StringGrid1->ColWidths[3] = 70;
 StringGrid1->ColWidths[4] = 70;
 StringGrid1->ColWidths[5] = 70;
 sprVidov = new tabelTools("TRENERA");
}
void TForm2::intTable(){
 ListA find;//условия выборки
 ReturnQuery* ret = sprVidov->find(&find);
 Part<AnsiString,AnsiString> BuffRow("SPRAVOCHNIK_VIDOV");
 ListA findparam;
```

```
findparam.clear();
GridIndex.clear();

StringGrid1->RowCount=1;//удаляем строки кроме заголовка

    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
    //вытаскиваем название вида спорта
     findparam["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
     BuffRow.find(&findparam);

     int cur = StringGrid1->RowCount;
     StringGrid1->RowCount++;
     GridIndex[cur]=(*(*it))["ID_TRENERA"];
     StringGrid1->Cells[0][cur]=BuffRow["VID_SPORTA"];
     StringGrid1->Cells[1][cur]=(*(*it))["BIRTHDAY"];
     StringGrid1->Cells[2][cur]=(*(*it))["ADRESSS"];
     StringGrid1->Cells[3][cur]=(*(*it))["TELEPHONE"];
     StringGrid1->Cells[4][cur]=(*(*it))["RAZRYAD"];
     StringGrid1->Cells[5][cur]=(*(*it))["FIO"];
    }



  //инициализация выпадающего списка
  ComboBox1->Items->Clear();
  ComboBox1->Text="";
  VidsportaIndex.clear();
  tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
  find.clear();//условия выборки
  ret = sprVidov.find(&find);
  int cur = 0;
  for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
     ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
     VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
     cur++;
  };
}
//---------------------------------------------------------------------------

void __fastcall TForm2::Button1Click(TObject *Sender)
{
  Form3->Show();
  Form3->init();
}
//---------------------------------------------------------------------------

void __fastcall TForm2::Button4Click(TObject *Sender)
{
//поиск оп виду спорта
     StringGrid1->RowCount=1;//удаляем строки кроме заголовка
     Part<AnsiString,AnsiString> BuffRow("SPRAVOCHNIK_VIDOV");
     ListA findparam;
     if(ComboBox1->ItemIndex!=-1){
          findparam.clear();
          findparam["ID_VID_SPORTA"] = VidsportaIndex[ComboBox1->ItemIndex];
     }

    ReturnQuery* ret = sprVidov->find(&findparam);
    findparam.clear();
    BuffRow.clear();
```

```
        GridIndex.clear();
        for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
          findparam["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
          BuffRow.find(&findparam);

          int cur = StringGrid1->RowCount;
          GridIndex[cur]=(*(*it))["ID_TRENERA"];
          StringGrid1->RowCount++;
          StringGrid1->Cells[0][cur]=BuffRow["VID_SPORTA"];
          StringGrid1->Cells[1][cur]=(*(*it))["BIRTHDAY"];
          StringGrid1->Cells[2][cur]=(*(*it))["ADRESSS"];
          StringGrid1->Cells[3][cur]=(*(*it))["TELEPHONE"];
          StringGrid1->Cells[4][cur]=(*(*it))["RAZRYAD"];
          StringGrid1->Cells[5][cur]=(*(*it))["FIO"];

        }
      }
      //---------------------------------------------------------------------------

      void __fastcall TForm2::Button2Click(TObject *Sender)
      {    //событие клавиши изменить

        if(GridIndex[StringGrid1->Row]!=""){
          int cur=StringGrid1->Row;
          Form3->init(GridIndex[StringGrid1->Row]);
          Form3->Edit1->Text=StringGrid1->Cells[5][cur];
          Form3->Edit2->Text=StringGrid1->Cells[1][cur];
          Form3->Edit3->Text=StringGrid1->Cells[4][cur];
          Form3->Edit4->Text=StringGrid1->Cells[2][cur];
          Form3->Edit5->Text=StringGrid1->Cells[3][cur];
          Form3->Show();
        }

      }
      //---------------------------------------------------------------------------

      void __fastcall TForm2::Button3Click(TObject *Sender)
      {  //событие удаление
        ListA findparam;
        findparam.clear();
        if(GridIndex[StringGrid1->Row]!=""){
          findparam["ID_TRENERA"] = GridIndex[StringGrid1->Row];
          Part<AnsiString,AnsiString> BuffRaspisanie("RASPISANIE_TRENIROV");
          Part<AnsiString,AnsiString> BuffSportsme("SPORTSMEN");
          Part<AnsiString,AnsiString> BuffDostijenia("DOSTIJENIA");
          if(BuffRaspisanie.find(&findparam)){ShowMessage("Сначала удали все Тренировки с данным
тренером");return;}
          if(BuffSportsme.find(&findparam)){ShowMessage("Сначала удали всех Спортсменов с данным
тренером");return;}

          Part<AnsiString,AnsiString> BuffRow("TRENERA");
          BuffRow.find(&findparam);
          BuffRow.del();
          intTable();
        }

      }
      //---------------------------------------------------------------------------

      void __fastcall TForm2::StringGrid1Click(TObject *Sender)
```

```
{

}
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "Unit3.h"
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
#include "tabelTools.h" //Главный класс таблицы от него наследуются все остальные
TForm3 *Form3;
//---------------------------------------------------------------------------
__fastcall TForm3::TForm3(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm3::Button1Click(TObject *Sender)
{ //событие кнопки добавить\изменить
Part<AnsiString,AnsiString> BuffRow("TRENERA");
ListA findparam;
if(_ID_BD!=""){
        findparam["ID_TRENERA"]= _ID_BD;
        BuffRow.find(&findparam);
}
      BuffRow["FIO"]=Edit1->Text;
      BuffRow["BIRTHDAY"]=Edit2->Text;
      BuffRow["RAZRYAD"]=Edit3->Text;
      BuffRow["ADRESSS"]=Edit4->Text;
      BuffRow["TELEPHONE"]=Edit5->Text;
      if(ComboBox1->ItemIndex!=-1){
        BuffRow["ID_VID_SPORTA"]=VidsportaIndex[ComboBox1->ItemIndex];
      }else{
        ShowMessage("Выберите вид спорта");
        return;
      }
      BuffRow.save();
      Form2->intTable();
      this->Close();


}

 void TForm3::init(AnsiString ID_BD){
    //инициализация формы для изменения
 _ID_BD = ID_BD;

 //вытаскиваем id вида спорта для инициализации выпадающего списка
 Part<AnsiString,AnsiString> BuffRow("TRENERA");
ListA findparam;
 findparam["ID_TRENERA"]= _ID_BD;
 BuffRow.find(&findparam);
 //инициализируем выпадающий список
    ComboBox1->Items->Clear();
    ComboBox1->Text = "";
    VidsportaIndex.clear();
```

```
   tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
   ListA find;//условия выборки
   ReturnQuery* ret = sprVidov.find(&find);
   int cur = 0;
   for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
       VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
       ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
       if(BuffRow["ID_VID_SPORTA"]==(*(*it))["ID_VID_SPORTA"])
            ComboBox1->ItemIndex=cur;//выставляем список на нужный элемент
       cur++;
   };
   Button1->Caption = "Изменить";
   this->Caption = "Изменить тренера";
 }
void TForm3::init(){
 //инициализация формы для создания
 _ID_BD = "";

 //инициализируем выпадающий список
   ComboBox1->Items->Clear();
   ComboBox1->Text = "";
   VidsportaIndex.clear();
   tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
   ListA find;//условия выборки
   ReturnQuery* ret = sprVidov.find(&find);
   int cur = 0;
   for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
       ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
       VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
       cur++;
};

  //для создания новой записи
     Edit1->Text="";
     Edit2->Text="";
     Edit3->Text="";
     Edit4->Text="";
     Edit5->Text="";
     ComboBox1->ItemIndex=0;
     Button1->Caption = "Создать";
     this->Caption = "Создать тренера";
 }
#include <vcl.h>
#pragma hdrstop

#include "Unit4.h"
#include "Unit8.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm4 *Form4;
//---------------------------------------------------------------------------
__fastcall TForm4::TForm4(TComponent* Owner)
       : TForm(Owner)
{
}
//---------------------------------------------------------------------------


void __fastcall TForm4::FormCreate(TObject *Sender)
```

```
{
 StringGrid1->ColCount=2;
 StringGrid1->RowCount=1;
 StringGrid1->Cells[0][0]="Характеристика";
 StringGrid1->Cells[1][0]="Ниимнование";
  StringGrid1->ColWidths[0] = 200;
 StringGrid1->ColWidths[1] = 200;
  TableSkcii = new tabelTools("SPRAVOCHNIK_VIDOV");
}
//---------------------------------------------------------------------------
void TForm4::intTable(){
 ListA find;//условия выборки
 ReturnQuery* AllSekcii = TableSkcii->find(&find);
 StringGrid1->RowCount=1;//удаляем строки кроме заголовка
 for (ReturnQuery::iterator it = AllSekcii->begin(); it != AllSekcii->end(); ++it){
     int cur = StringGrid1->RowCount;
       StringGrid1->RowCount++;
       GridIndex[cur]=(*(*it))["ID_VID_SPORTA"];
       StringGrid1->Cells[0][cur]=(*(*it))["XARAKTERISTIKA"];
       StringGrid1->Cells[1][cur]=(*(*it))["VID_SPORTA"];
 }
}

/*
  tabelTools* TableSkcii;
      ListA GridIndex;//соответствие адресов индексов для таблицы
public:// User declarations
      __fastcall TForm4(TComponent* Owner);
      void intTable();//и

*/

void __fastcall TForm4::Button3Click(TObject *Sender)
{
  ListA find;//условия выборки
if(GridIndex[StringGrid1->Row]!=""){
  find["ID_VID_SPORTA"] = GridIndex[StringGrid1->Row];
  Part<AnsiString,AnsiString> Buffsorev("RASPISANIE_SOREVN");
  Part<AnsiString,AnsiString> Bufft2("RASPISANIE_TRENIROV");
  Part<AnsiString,AnsiString> Buff3("TRENERA");
  Part<AnsiString,AnsiString> Buff4("SPORTSMEN");
  if(Buffsorev.find(&find)){ShowMessage("Сначала удали все соревнования с данной секцией");return;}
  if(Bufft2.find(&find)){ShowMessage("Сначала удали все Тренировки с данной секцией");return;}
  if(Buff3.find(&find)){ShowMessage("Сначала удали всх тренеров с данной секции");return;}
  if(Buff4.find(&find)){ShowMessage("Сначала удали все спортсменов с данной секции");return;}


  Part<AnsiString,AnsiString> BuffRow("SPRAVOCHNIK_VIDOV");
  BuffRow.find(&find);
  BuffRow.del();
  intTable();
}

}
//---------------------------------------------------------------------------

void __fastcall TForm4::Button1Click(TObject *Sender)
{       //добавить

      Form8->init();
```

```
        Form8->Show();
}
//---------------------------------------------------------------------------

void __fastcall TForm4::Button2Click(TObject *Sender)
{//изменить
if(GridIndex[StringGrid1->Row]!=""){
    int cur=StringGrid1->Row;
    Form8->init(GridIndex[StringGrid1->Row]);
    Form8->Edit1->Text=StringGrid1->Cells[1][cur];
    Form8->Edit2->Text=StringGrid1->Cells[0][cur];
    Form8->Show();
}
}

#include <vcl.h>
#pragma hdrstop

#include "Unit5.h"

//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm5 *Form5;
//---------------------------------------------------------------------------
__fastcall TForm5::TForm5(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm5::FormCreate(TObject *Sender)
{
 StringGrid1->ColCount=4;
 StringGrid1->RowCount=1;
 StringGrid1->Cells[0][0]="Вид спорта";
 StringGrid1->Cells[1][0]="Дата проведения";
 StringGrid1->Cells[2][0]="Наименование";
 StringGrid1->Cells[3][0]="Место проведения";
 StringGrid1->ColWidths[0] = 70;
 StringGrid1->ColWidths[1] = 70;
 StringGrid1->ColWidths[2] = 100;
 StringGrid1->ColWidths[3] = 100;
 Raspis_sorevn = new tabelTools("RASPISANIE_SOREVN");
}
//---------------------------------------------------------------------------

void __fastcall TForm5::Button1Click(TObject *Sender)
{  //добавить
   Form9->init();
   Form9->Show();
}
//---------------------------------------------------------------------------

void __fastcall TForm5::Button2Click(TObject *Sender)
{//изменить

 if(GridIndex[StringGrid1->Row]!=""){
    int cur=StringGrid1->Row;
    Form9->init(GridIndex[StringGrid1->Row]);
    Form9->Edit1->Text=StringGrid1->Cells[1][cur];
```

```
    Form9->Edit2->Text=StringGrid1->Cells[2][cur];
    Form9->Edit3->Text=StringGrid1->Cells[3][cur];
    Form9->Show();
  }

}
//---------------------------------------------------------------------------

void __fastcall TForm5::Button3Click(TObject *Sender)
{//удалить
  ListA findparam;
 findparam.clear();
 if(GridIndex[StringGrid1->Row]!=""){
  findparam["ID_RASPISANIE_SOREVN"] = GridIndex[StringGrid1->Row];
  Part<AnsiString,AnsiString> BuffDostijenia("DOSTIJENIA");
  if(BuffDostijenia.find(&findparam)){ShowMessage("Сначала удали все Достижения с данным
соревнованием");return;}
   Part<AnsiString,AnsiString> BuffRow("RASPISANIE_SOREVN");
  BuffRow.find(&findparam);
  BuffRow.del();
  intTable();
  }
}
//---------------------------------------------------------------------------
void TForm5::intTable()
{
 ListA find;//условия выборки
 ReturnQuery* AllSorevn = Raspis_sorevn->find(&find);
 Part<AnsiString,AnsiString> BuffSPvidov("SPRAVOCHNIK_VIDOV");
 find.clear();
 GridIndex.clear();
 StringGrid1->RowCount=1;//удаляем строки кроме заголовка

 //заплняем таблицу
  for (ReturnQuery::iterator it = AllSorevn->begin(); it != AllSorevn->end(); ++it){
     //вытаскиваем название вида спорта
      find["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
      BuffSPvidov.find(&find);

      int cur = StringGrid1->RowCount;
      StringGrid1->RowCount++;
      GridIndex[cur]=(*(*it))["ID_RASPISANIE_SOREVN"];
      StringGrid1->Cells[0][cur]=BuffSPvidov["VID_SPORTA"];
      StringGrid1->Cells[1][cur]=(*(*it))["DATA_PROVEDEN"];
      StringGrid1->Cells[2][cur]=(*(*it))["NAVANIE_SOREVN"];
      StringGrid1->Cells[3][cur]=(*(*it))["MESTO_PROVEDEN"];
  }

  //инициализация выпадающего списка
   ComboBox1->Items->Clear();
   ComboBox1->Text="";
   VidsportaIndex.clear();
   tabelTools TSPvidov("SPRAVOCHNIK_VIDOV");
   find.clear();//условия выборки
   ReturnQuery* AllSPvids = TSPvidov.find(&find);
   int cur = 0;
   for (ReturnQuery::iterator it = AllSPvids->begin(); it != AllSPvids->end(); ++it){
      ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
      VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
      cur++;
```

```
   };

}
void __fastcall TForm5::Button4Click(TObject *Sender)
{//поиск
      StringGrid1->RowCount=1;//удаляем строки кроме заголовка
      Part<AnsiString,AnsiString> BuffSPvidov("SPRAVOCHNIK_VIDOV");
      ListA findparam;
      if(ComboBox1->ItemIndex!=-1){
            findparam.clear();
            findparam["ID_VID_SPORTA"] = VidsportaIndex[ComboBox1->ItemIndex];
      }

     ReturnQuery* AllSorevn = Raspis_sorevn->find(&findparam);
     findparam.clear();
     BuffSPvidov.clear();
     GridIndex.clear();
  for (ReturnQuery::iterator it = AllSorevn->begin(); it != AllSorevn->end(); ++it){
     //вытаскиваем название вида спорта
      findparam["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
      BuffSPvidov.find(&findparam);

      int cur = StringGrid1->RowCount;
      StringGrid1->RowCount++;
      GridIndex[cur]=(*(*it))["ID_RASPISANIE_SOREVN"];
      StringGrid1->Cells[0][cur]=BuffSPvidov["VID_SPORTA"];
      StringGrid1->Cells[1][cur]=(*(*it))["DATA_PROVEDEN"];
      StringGrid1->Cells[2][cur]=(*(*it))["NAVANIE_SOREVN"];
      StringGrid1->Cells[3][cur]=(*(*it))["MESTO_PROVEDEN"];
  }
}
//---------------------------------------------------------------------------

 void __fastcall TForm5::StringDBClick(TObject *Sender)
{ //удалить
  if(GridIndex[StringGrid1->Row]!=""){
    Form6->intTable(GridIndex[StringGrid1->Row]);
    Form6->Caption="Достижения на "'+StringGrid1->Cells[2][StringGrid1->Row]+"'";
    Form6->Show();
 }
}

#include <vcl.h>
#pragma hdrstop

#include "Unit6.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm6 *Form6;
//---------------------------------------------------------------------------
__fastcall TForm6::TForm6(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm6::FormCreate(TObject *Sender)
{
 StringGrid1->ColCount=2;
```

```
 StringGrid1->RowCount=1;
 StringGrid1->Cells[0][0]="Спортсмен";
 StringGrid1->Cells[1][0]="Занятое место";
  StringGrid1->ColWidths[0] = 150;
 StringGrid1->ColWidths[1] = 150;
 Tdostijenia   = new tabelTools("DOSTIJENIA");
}
//---------------------------------------------------------------------------
void TForm6::intTable(AnsiString ID_sor){
      _ID_sorevn = ID_sor;
     ListA find;//условия выборки
     Part<AnsiString,AnsiString> BuffRow("SPORTSMEN");//для парсинга тренера
     find["ID_RASPISANIE_SOREVN"]=_ID_sorevn;
     ReturnQuery* AllDosijeniaInSorevnovanie = Tdostijenia->find(&find);
     find.clear();
     GridIndex.clear();
     StringGrid1->RowCount=1;//удаляем строки кроме заголовка
      for (ReturnQuery::iterator it = AllDosijeniaInSorevnovanie->begin(); it != AllDosijeniaInSorevnovanie-
>end(); ++it){
      //вытаскиваем название вида спорта
        find["ID_SPORSMEN"]=(*(*it))["ID_SPORSMEN"];
        BuffRow.find(&find);

        int cur = StringGrid1->RowCount;
        StringGrid1->RowCount++;
        GridIndex[cur]=(*(*it))["ID_DOSTIJENIA"];
        StringGrid1->Cells[0][cur]=BuffRow["FIO"];
        StringGrid1->Cells[1][cur]=(*(*it))["MESTO"];
       }
}
void __fastcall TForm6::Button1Click(TObject *Sender)
{   //добавить
   Form10->_ID_sorevn=_ID_sorevn;
   Form10->init();
   Form10->Show();
}
//---------------------------------------------------------------------------

void __fastcall TForm6::Button2Click(TObject *Sender)
{//изменить
  if(GridIndex[StringGrid1->Row]!=""){
     Form10->_ID_sorevn=_ID_sorevn;
     int cur=StringGrid1->Row;
     Form10->init(GridIndex[StringGrid1->Row]);
     Form10->Edit1->Text=StringGrid1->Cells[1][cur];
     Form10->Show();
  }
}
//---------------------------------------------------------------------------



  void __fastcall TForm6::Button3Click(TObject *Sender)
{ //удалить
   ListA findparam;
  findparam.clear();
  if(GridIndex[StringGrid1->Row]!=""){
   findparam["ID_DOSTIJENIA"] = GridIndex[StringGrid1->Row];
   Part<AnsiString,AnsiString> BuffRow("DOSTIJENIA");
   BuffRow.find(&findparam);
```

```
  BuffRow.del();
  intTable(_ID_sorevn);
 }
}
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "Unit7.h"
#include "Unit12.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm7 *Form7;
//---------------------------------------------------------------------------
__fastcall TForm7::TForm7(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void TForm7::intTable(){

ComboBox1->ItemIndex=-1;
ComboBox2->ItemIndex=-1;
   CheckBox1->Checked = false;
   CheckBox2->Checked = false;
   CheckBox3->Checked = false;
   CheckBox4->Checked = false;
   CheckBox5->Checked = false;
   CheckBox6->Checked = false;
   CheckBox7->Checked = false;
   CheckBox8->Checked = false;

 ListA find;//условия выборки
 ReturnQuery* AllTrener = RaspisanieT->find(&find);
 Part<AnsiString,AnsiString> BuffTrenera("TRENERA");//для парса имени тренера
 Part<AnsiString,AnsiString> BuffVsporta("SPRAVOCHNIK_VIDOV");//для парса наименования спорта
 BuffTrenera.clear();
 BuffVsporta.clear();
 StringGrid1->RowCount=1;//удаляем строки кроме заголовка
    for (ReturnQuery::iterator it = AllTrener->begin(); it != AllTrener->end(); ++it){
       //вытаскиваем название вида спорта
       find.clear();
       find["ID_TRENERA"]=(*(*it))["ID_TRENERA"];
       BuffTrenera.find(&find);
       find.clear();
       find["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
       BuffVsporta.find(&find);

       int cur = StringGrid1->RowCount;
       StringGrid1->RowCount++;
        GridIndex[cur]=(*(*it))["ID_RASPISANIE_TRENIROV"];
        StringGrid1->Cells[0][cur]=BuffTrenera["FIO"];
        StringGrid1->Cells[1][cur]=dniNedeli[(*(*it))["ID_DNI_NEDELI"]];
        StringGrid1->Cells[2][cur]=BuffVsporta["VID_SPORTA"];
    }

   ComboBox1->Items->Clear();
   ComboBox1->Text="";
   VidsportaIndex.clear();
```

```
    find.clear();//условия выборки
     tabelTools TVsporta("SPRAVOCHNIK_VIDOV");
    ReturnQuery* ret = TVsporta.find(&find);
    int cur = 0;
    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
        ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
        VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
        cur++;
    };
    ComboBox2->Items->Clear();
    ComboBox2->Text="";
    TrenerIndex.clear();
    find.clear();//условия выборки
    tabelTools TTrenera("TRENERA");
    ret = TTrenera.find(&find);
    cur = 0;
    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
        ComboBox2->Items->Add((*(*it))["FIO"]);
        TrenerIndex[cur]=(*(*it))["ID_TRENERA"];
        cur++;
    };
}
void __fastcall TForm7::FormCreate(TObject *Sender)
{
 StringGrid1->ColCount=3;
 StringGrid1->RowCount=1;
 StringGrid1->Cells[0][0]="Ф.И.О.";
 StringGrid1->Cells[1][0]="День недели";
 StringGrid1->Cells[2][0]="Секция";
  StringGrid1->ColWidths[0] = 150
 StringGrid1->ColWidths[1] = 70
 StringGrid1->ColWidths[2] = 100
 RaspisanieT = new tabelTools("RASPISANIE_TRENIROV");
  dniNedeli["1"]="Пн";
 dniNedeli["2"]="Вт";
 dniNedeli["3"]="Ср";
 dniNedeli["4"]="Чт";
 dniNedeli["5"]="Пт";
 dniNedeli["6"]="Сб";
 dniNedeli["7"]="Вс";
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox2Click(TObject *Sender)
{
    CheckBox1->Checked = false;
    CheckBox3->Checked = false;
    CheckBox4->Checked = false;
    CheckBox5->Checked = false;
    CheckBox6->Checked = false;
    CheckBox7->Checked = false;
    CheckBox8->Checked = false;

}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox3Click(TObject *Sender)
{
    CheckBox1->Checked = false;
    CheckBox2->Checked = false;
```

```
   CheckBox4->Checked = false;
   CheckBox5->Checked = false;
   CheckBox6->Checked = false;
   CheckBox7->Checked = false;
   CheckBox8->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox4Click(TObject *Sender)
{
   CheckBox1->Checked = false;
   CheckBox2->Checked = false;
   CheckBox3->Checked = false;
   CheckBox5->Checked = false;
   CheckBox6->Checked = false;
   CheckBox7->Checked = false;
   CheckBox8->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox5Click(TObject *Sender)
{
   CheckBox1->Checked = false;
   CheckBox2->Checked = false;
   CheckBox3->Checked = false;
   CheckBox4->Checked = false;
   CheckBox6->Checked = false;
   CheckBox7->Checked = false;
   CheckBox8->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox6Click(TObject *Sender)
{
   CheckBox1->Checked = false;
   CheckBox2->Checked = false;
   CheckBox3->Checked = false;
   CheckBox4->Checked = false;
   CheckBox5->Checked = false;
   CheckBox7->Checked = false;
   CheckBox8->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox7Click(TObject *Sender)
{
   CheckBox1->Checked = false;
   CheckBox2->Checked = false;
   CheckBox3->Checked = false;
   CheckBox4->Checked = false;
   CheckBox5->Checked = false;
   CheckBox6->Checked = false;
   CheckBox8->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox8Click(TObject *Sender)
{
   CheckBox1->Checked = false;
   CheckBox2->Checked = false;
```

```
   CheckBox3->Checked = false;
   CheckBox4->Checked = false;
   CheckBox5->Checked = false;
   CheckBox6->Checked = false;
   CheckBox7->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::CheckBox1Click(TObject *Sender)
{
   CheckBox2->Checked = false;
   CheckBox3->Checked = false;
   CheckBox4->Checked = false;
   CheckBox5->Checked = false;
   CheckBox6->Checked = false;
   CheckBox7->Checked = false;
   CheckBox8->Checked = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm7::Button1Click(TObject *Sender)
{
//добавить
AddTrenerovka->Show();
AddTrenerovka->init();
}
//---------------------------------------------------------------------------

void __fastcall TForm7::Button2Click(TObject *Sender)
{
//изменить

if(GridIndex[StringGrid1->Row]!=""){
    int cur=StringGrid1->Row;
    AddTrenerovka->init(GridIndex[StringGrid1->Row]);
   // Form3->Edit1->Text=StringGrid1->Cells[5][cur];
    AddTrenerovka->Show();
  }
}
//---------------------------------------------------------------------------

void __fastcall TForm7::Button3Click(TObject *Sender)
{
//удалить
  ListA findparam;
 findparam.clear();
 if(GridIndex[StringGrid1->Row]!=""){
  findparam["ID_RASPISANIE_TRENIROV"] = GridIndex[StringGrid1->Row];
  Part<AnsiString,AnsiString> BuffRow("RASPISANIE_TRENIROV");
  BuffRow.find(&findparam);
  BuffRow.del();
  intTable();
  }
}
//---------------------------------------------------------------------------

void __fastcall TForm7::Button4Click(TObject *Sender)
{
//поиск
      StringGrid1->RowCount=1;//удаляем строки кроме заголовка
```

```
        ListA find;
        if(ComboBox1->ItemIndex!=-1){
            find["ID_VID_SPORTA"] = VidsportaIndex[ComboBox1->ItemIndex];
        }
        if(ComboBox2->ItemIndex!=-1){
            find["ID_TRENERA"] = TrenerIndex[ComboBox2->ItemIndex];
        }
        int CursorDay = 0;
      if(CheckBox2->Checked)CursorDay = 1;
      if(CheckBox3->Checked)CursorDay = 2;
      if(CheckBox4->Checked)CursorDay = 3;
      if(CheckBox5->Checked)CursorDay = 4;
      if(CheckBox6->Checked)CursorDay = 5;
      if(CheckBox7->Checked)CursorDay = 6;
      if(CheckBox8->Checked)CursorDay = 7;
        if(CursorDay!=0)find["ID_DNI_NEDELI"]=CursorDay;
         ReturnQuery* AllTrener = RaspisanieT->find(&find);
        Part<AnsiString,AnsiString> BuffTrenera("TRENERA");//для парса имени тренера
        Part<AnsiString,AnsiString> BuffVsporta("SPRAVOCHNIK_VIDOV");//для парса наименования
спорта
        BuffTrenera.clear();
        BuffVsporta.clear();
     StringGrid1->RowCount=1;//удаляем строки кроме заголовка
      for (ReturnQuery::iterator it = AllTrener->begin(); it != AllTrener->end(); ++it){
         //вытаскиваем название вида спорта
         find.clear();
         find["ID_TRENERA"]=(*(*it))["ID_TRENERA"];
         BuffTrenera.find(&find);
         find.clear();
         find["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
         BuffVsporta.find(&find);

         int cur = StringGrid1->RowCount;
         StringGrid1->RowCount++;
          GridIndex[cur]=(*(*it))["ID_RASPISANIE_TRENIROV"];
          StringGrid1->Cells[0][cur]=BuffTrenera["FIO"];
          StringGrid1->Cells[1][cur]=dniNedeli[(*(*it))["ID_DNI_NEDELI"]];
          StringGrid1->Cells[2][cur]=BuffVsporta["VID_SPORTA"];
      }
    }
    //---------------------------------------------------------------------------
    #include <vcl.h>
    #pragma hdrstop

    #include "Unit8.h"
    #include "Unit4.h"
    //---------------------------------------------------------------------------
    #pragma package(smart_init)
    #pragma resource "*.dfm"
    TForm8 *Form8;
    //---------------------------------------------------------------------------
    __fastcall TForm8::TForm8(TComponent* Owner)
        : TForm(Owner)
    {
    }
    //---------------------------------------------------------------------------
    void TForm8::init(AnsiString ID_BD){
       //инициализация формы для изменения
        _ID_BD = ID_BD;
        Button1->Caption = "Изменить";
```

```
       this->Caption = "Изменить запись";
 }
void TForm8::init(){
    //инициализация формы для создания
    _ID_BD = "";
    //для создания новой записи
     Edit1->Text="";
     Edit2->Text="";
     Button1->Caption = "Создать";
     this->Caption = "Создать запись";

};
void __fastcall TForm8::Button1Click(TObject *Sender)
{
  Part<AnsiString,AnsiString> Tvids("SPRAVOCHNIK_VIDOV");
  ListA findparam;
if(_ID_BD!=""){
       findparam["ID_VID_SPORTA"]= _ID_BD;
       Tvids.find(&findparam);
}
     Tvids["VID_SPORTA"]=Edit1->Text;
     Tvids["XARAKTERISTIKA"]=Edit2->Text;
     Tvids.save();
     Form4->intTable();
     this->Close();
}
//---------------------------------
#include <vcl.h>
#pragma hdrstop

#include "Unit9.h"

//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm9 *Form9;
//---------------------------------------------------------------------------
__fastcall TForm9::TForm9(TComponent* Owner)
       : TForm(Owner)
{
}
void TForm9::init(AnsiString ID_BD){
    //инициализация формы для изменения
 _ID_BD = ID_BD;

 //вытаскиваем id вида спорта для инициализации выпадающего списка
 Part<AnsiString,AnsiString> BuffRow("RASPISANIE_SOREVN");
ListA findparam;
 findparam["ID_RASPISANIE_SOREVN"]= _ID_BD;
 BuffRow.find(&findparam);
 //инициализируем выпадающий список
   ComboBox1->Items->Clear();
   ComboBox1->Text = "";
   VidsportaIndex.clear();
   tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
   ListA find;//условия выборки
   ReturnQuery* ret = sprVidov.find(&find);
   int cur = 0;
   for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
       VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
```

```
        ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
        if(BuffRow["ID_VID_SPORTA"]==(*(*it))["ID_VID_SPORTA"])
             ComboBox1->ItemIndex=cur;//выставляем список на нужный элемент
        cur++;
    };
    Button1->Caption = "Изменить";
    this->Caption = "Изменить соревнование";
}
void TForm9::init(){
 //инициализация формы для создания
 _ID_BD = "";

 //инициализируем выпадающий список
    ComboBox1->Items->Clear();
    ComboBox1->Text = "";
    VidsportaIndex.clear();
    tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
    ListA find;//условия выборки
    ReturnQuery* ret = sprVidov.find(&find);
    int cur = 0;
    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
        ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
        VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
        cur++;
};

  //для создания новой записи
     Edit1->Text="";
     Edit2->Text="";
     Edit3->Text="";
     ComboBox1->ItemIndex=0;
     Button1->Caption = "Создать";
     this->Caption = "Создать Соревнование";
 }
//---------------------------------------------------------------------------
void __fastcall TForm9::Button1Click(TObject *Sender)
{
//событие кнопки добавить\изменить
Part<AnsiString,AnsiString> BuffRow("RASPISANIE_SOREVN");
ListA findparam;
if(_ID_BD!=""){
       findparam["ID_RASPISANIE_SOREVN"]= _ID_BD;
       BuffRow.find(&findparam);
}
     BuffRow["DATA_PROVEDEN"]=Edit1->Text;
     BuffRow["NAVANIE_SOREVN"]=Edit2->Text;
     BuffRow["MESTO_PROVEDEN"]=Edit3->Text;
     if(ComboBox1->ItemIndex!=-1){
       BuffRow["ID_VID_SPORTA"]=VidsportaIndex[ComboBox1->ItemIndex];
     }else{
       ShowMessage("Выберите вид спорта");
       return;
     }
     BuffRow.save();
     Form5->intTable();
     this->Close();
}

#include <vcl.h>
#pragma hdrstop
```

```
#include "Unit10.h"
#include "Unit6.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm10 *Form10;
//---------------------------------------------------------------------------
__fastcall TForm10::TForm10(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void TForm10::init(){
 //инициализация формы для создания
 _ID_BD = "";

 //инициализируем выпадающий список
    ComboBox1->Items->Clear();
    ComboBox1->Text = "";
    SportsmenIndex.clear();
    tabelTools sprVidov("SPORTSMEN");
    ListA find;//условия выборки
    ReturnQuery* ret = sprVidov.find(&find);
    int cur = 0;
    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
        ComboBox1->Items->Add((*(*it))["FIO"]);
        SportsmenIndex[cur]=(*(*it))["ID_SPORSMEN"];
        cur++;
    };
}
void TForm10::init(AnsiString ID_BD){
    _ID_BD = ID_BD;
    Part<AnsiString,AnsiString> BuffRow("SPORTSMEN");
    ListA findparam;
    findparam["ID_SPORSMEN"]= _ID_BD;
    BuffRow.find(&findparam);
    //инициализируем выпадающий список
    ComboBox1->Items->Clear();
    ComboBox1->Text = "";
    SportsmenIndex.clear();
    tabelTools sprVidov("SPORTSMEN");
    ListA find;//условия выборки
    ReturnQuery* ret = sprVidov.find(&find);
    int cur = 0;
    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
        ComboBox1->Items->Add((*(*it))["FIO"]);
        SportsmenIndex[cur]=(*(*it))["ID_SPORSMEN"];
        if(BuffRow["ID_SPORSMEN"]==(*(*it))["ID_SPORSMEN"])
            ComboBox1->ItemIndex=cur;//выставляем список на нужный элемент
        cur++;
    };
}

void __fastcall TForm10::Button1Click(TObject *Sender)
{ //добавить/изменить
  Part<AnsiString,AnsiString> BuffRow("DOSTIJENIA");
ListA findparam;
if(_ID_BD!=""){
       findparam["ID_DOSTIJENIA"]= _ID_BD;
```

```
      BuffRow.find(&findparam);
}
   if(ComboBox1->ItemIndex!=-1){
     BuffRow["ID_SPORSMEN"]=SportsmenIndex[ComboBox1->ItemIndex];
    }else{
     ShowMessage("Выберите спорсмена");
     return;
    }
    BuffRow["MESTO"]=Edit1->Text;
    BuffRow["ID_RASPISANIE_SOREVN"]=_ID_sorevn;
    BuffRow.save();
    Form6->intTable(_ID_sorevn);
    this->Close();
}

#include <vcl.h>
#pragma hdrstop

#include "Unit11.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm11 *Form11;
//---------------------------------------------------------------------------
__fastcall TForm11::TForm11(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void TForm11::intTable(){

 ListA find;//условия выборки
 ReturnQuery* AllSportsmen = Tsportsmen->find(&find);
 Part<AnsiString,AnsiString> Tsvidov("SPRAVOCHNIK_VIDOV");
 Part<AnsiString,AnsiString> TTrenera("TRENERA");
 find.clear();
 GridIndex.clear();
 StringGrid1->RowCount=1;//удаляем строки кроме заголовка
     for (ReturnQuery::iterator it = AllSportsmen->begin(); it != AllSportsmen->end(); ++it){
     //вытаскиваем название вида спорта
       find["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
       Tsvidov.find(&find);
       find.clear();
       find["ID_TRENERA"]=(*(*it))["ID_TRENERA"];
       TTrenera.find(&find);
       find.clear();

       int cur = StringGrid1->RowCount;
       StringGrid1->RowCount++;
       GridIndex[cur]=(*(*it))["ID_SPORSMEN"];
       StringGrid1->Cells[0][cur]=TTrenera["FIO"];
       StringGrid1->Cells[1][cur]=Tsvidov["VID_SPORTA"];
       StringGrid1->Cells[2][cur]=(*(*it))["FIO"];
       StringGrid1->Cells[3][cur]=(*(*it))["BIRTHDAY"];
       StringGrid1->Cells[4][cur]=(*(*it))["TELEPHONE"];
       StringGrid1->Cells[5][cur]=(*(*it))["RAZRYAD"];
       StringGrid1->Cells[6][cur]=(*(*it))["GTO"];
     }
  //инициализация выпадающего списка
  ComboBox1->Items->Clear();
```

```
    ComboBox1->Text="";
    VidsportaIndex.clear();
    tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
    find.clear();//условия выборки
     ReturnQuery* AllVid = sprVidov.find(&find);
    int cur = 0;
    for (ReturnQuery::iterator it = AllVid->begin(); it != AllVid->end(); ++it){
        ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
        VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
        cur++;
    };


    ComboBox2->Items->Clear();
    ComboBox2->Text="";
    TrenerIndex.clear();
    tabelTools Ttrenrs("TRENERA");
    find.clear();//условия выборки
    ReturnQuery* alTren = Ttrenrs.find(&find);
    cur = 0;
    for (ReturnQuery::iterator it = alTren->begin(); it != alTren->end(); ++it){
        ComboBox2->Items->Add((*(*it))["FIO"]);
        TrenerIndex[cur]=(*(*it))["ID_TRENERA"];
        cur++;
    };
}
void __fastcall TForm11::FormCreate(TObject *Sender)
{
 StringGrid1->ColCount=7;
 StringGrid1->RowCount=1;
 StringGrid1->Cells[0][0]="Тренер";
 StringGrid1->Cells[1][0]="Секция";
 StringGrid1->Cells[2][0]="Ф.И.О.";
 StringGrid1->Cells[3][0]="День рождения";
 StringGrid1->Cells[4][0]="Телефон";
 StringGrid1->Cells[5][0]="Разряд";
 StringGrid1->Cells[6][0]="ГТО";
 StringGrid1->ColWidths[0] = 70;
 StringGrid1->ColWidths[1] = 70;
 StringGrid1->ColWidths[2] = 70;
 StringGrid1->ColWidths[3] = 70;
 StringGrid1->ColWidths[4] = 70;
 StringGrid1->ColWidths[5] = 70;
 StringGrid1->ColWidths[6] = 70;


 Tsportsmen = new tabelTools("SPORTSMEN");


}
//---------------------------------------------------------------------------


void __fastcall TForm11::Button2Click(TObject *Sender)
{ //изменить
    if(GridIndex[StringGrid1->Row]!=""){
    int cur=StringGrid1->Row;
    Form13->init(GridIndex[StringGrid1->Row]);
    Form13->Edit1->Text = StringGrid1->Cells[2][cur];
    Form13->Edit2->Text= StringGrid1->Cells[3][cur];
```

```
            Form13->Edit3->Text=StringGrid1->Cells[4][cur];
            Form13->Edit4->Text=StringGrid1->Cells[5][cur];
            Form13->Edit5->Text=StringGrid1->Cells[6][cur];
            Form13->Show();
          }
        }
        //---------------------------------------------------------------------------

        void __fastcall TForm11::Button1Click(TObject *Sender)
        {    // добавить
             Form13->Show();
             Form13->init();
        }
        //---------------------------------------------------------------------------

        void __fastcall TForm11::Button3Click(TObject *Sender)
        { //удалить
          ListA findparam;
          findparam.clear();
          if(GridIndex[StringGrid1->Row]!=""){
           findparam["ID_SPORSMEN"] = GridIndex[StringGrid1->Row];
           Part<AnsiString,AnsiString> BuffDostijenia("DOSTIJENIA");
           if(BuffDostijenia.find(&findparam)){ShowMessage("Сначала удали все Достижения с данным
CGJHNCVTYJV");return;}

           Part<AnsiString,AnsiString> BuffRow("SPORTSMEN");
           BuffRow.find(&findparam);
           BuffRow.del();
           intTable();
          }

        }
        //---------------------------------------------------------------------------

        void __fastcall TForm11::Button4Click(TObject *Sender)
        {//ПОИСК
              StringGrid1->RowCount=1;//удаляем строки кроме заголовка
              ListA find;
              ListA Pfind;
              if(ComboBox1->ItemIndex!=-1){
                    Pfind["ID_VID_SPORTA"] = VidsportaIndex[ComboBox1->ItemIndex];
              }
              if(ComboBox2->ItemIndex!=-1){
                    Pfind["ID_TRENERA"] = TrenerIndex[ComboBox2->ItemIndex];
              }
              ReturnQuery* AllSportsmen = Tsportsmen->find(&Pfind);
              Part<AnsiString,AnsiString> BuffTrenera("TRENERA");//для парса имени тренера
              Part<AnsiString,AnsiString> BuffVsporta("SPRAVOCHNIK_VIDOV");//для парса наименования
спорта
              StringGrid1->RowCount=1;//удаляем строки кроме заголовка
              for (ReturnQuery::iterator it = AllSportsmen->begin(); it != AllSportsmen->end(); ++it){
             //вытаскиваем название вида спорта
               find["ID_VID_SPORTA"]=(*(*it))["ID_VID_SPORTA"];
               BuffVsporta.find(&find);
               find.clear();
               find["ID_TRENERA"]=(*(*it))["ID_TRENERA"];
               BuffTrenera.find(&find);
               find.clear();

               int cur = StringGrid1->RowCount;
```

```
        StringGrid1->RowCount++;
        GridIndex[cur]=(*(*it))["ID_SPORSMEN"];
        StringGrid1->Cells[0][cur]=BuffVsporta["VID_SPORTA"];
        StringGrid1->Cells[1][cur]=BuffTrenera["FIO"];
        StringGrid1->Cells[2][cur]=(*(*it))["FIO"];
        StringGrid1->Cells[3][cur]=(*(*it))["BIRTHDAY"];
        StringGrid1->Cells[4][cur]=(*(*it))["TELEPHONE"];
        StringGrid1->Cells[5][cur]=(*(*it))["RAZRYAD"];
        StringGrid1->Cells[6][cur]=(*(*it))["GTO"];
      }
}

#include <vcl.h>
#pragma hdrstop

#include "Unit13.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm13 *Form13;
//---------------------------------------------------------------------------
__fastcall TForm13::TForm13(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void TForm13::init(AnsiString ID_BD){
    //инициализация формы для изменения
 _ID_BD = ID_BD;

 //вытаскиваем id вида спорта для инициализации выпадающего списка
 Part<AnsiString,AnsiString> BuffRow("SPORTSMEN");
ListA findparam;
 findparam["ID_SPORSMEN"]= _ID_BD;
 BuffRow.find(&findparam);
 //инициализируем выпадающий список
   ComboBox1->Items->Clear();
   ComboBox1->Text = "";
   VidsportaIndex.clear();
   tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
   ListA find;//условия выборки
   ReturnQuery* ret = sprVidov.find(&find);
   int cur = 0;
   for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
       VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
       ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
       if(BuffRow["ID_VID_SPORTA"]==(*(*it))["ID_VID_SPORTA"])
            ComboBox1->ItemIndex=cur;//выставляем список на нужный элемент
       cur++;
   };
   ComboBox2->Items->Clear();
   ComboBox2->Text="";
   TrenerIndex.clear();
   tabelTools Ttrenrs("TRENERA");
   find.clear();//условия выборки
   ReturnQuery* alTren = Ttrenrs.find(&find);
   cur = 0;
   for (ReturnQuery::iterator it = alTren->begin(); it != alTren->end(); ++it){
       ComboBox2->Items->Add((*(*it))["FIO"]);
       TrenerIndex[cur]=(*(*it))["ID_TRENERA"];
```

```
        if(BuffRow["ID_TRENERA"]==(*(*it))["ID_TRENERA"])
            ComboBox2->ItemIndex=cur;//выставляем список на нужный элемент
        cur++;
    };
    Button1->Caption = "Изменить";
    this->Caption = "Изменить запись о спортсмене";
 }
void TForm13::init(){
 //инициализация формы для создания
 _ID_BD = "";

 //инициализируем выпадающий список
    ComboBox1->Items->Clear();
    ComboBox1->Text = "";
    VidsportaIndex.clear();
    tabelTools sprVidov("SPRAVOCHNIK_VIDOV");
    ListA find;//условия выборки
    ReturnQuery* ret = sprVidov.find(&find);
    int cur = 0;
    for (ReturnQuery::iterator it = ret->begin(); it != ret->end(); ++it){
        ComboBox1->Items->Add((*(*it))["VID_SPORTA"]);
        VidsportaIndex[cur]=(*(*it))["ID_VID_SPORTA"];
        cur++;
    };
    ComboBox2->Items->Clear();
    ComboBox2->Text="";
    TrenerIndex.clear();
    tabelTools Ttrenrs("TRENERA");
    find.clear();//условия выборки
    ReturnQuery* alTren = Ttrenrs.find(&find);
    cur = 0;
    for (ReturnQuery::iterator it = alTren->begin(); it != alTren->end(); ++it){
        ComboBox2->Items->Add((*(*it))["FIO"]);
        TrenerIndex[cur]=(*(*it))["ID_TRENERA"];
        cur++;
    };
  //для создания новой записи
     Edit1->Text="";
     Edit2->Text="";
     Edit3->Text="";
     Edit4->Text="";
     Edit5->Text="";
     ComboBox1->ItemIndex=-1;
     ComboBox2->ItemIndex=-1;
     Button1->Caption = "Создать";
     this->Caption = "Создать запись о спортсмене";
 }

void __fastcall TForm13::Button1Click(TObject *Sender)
{   //сохранить изменить
      Part<AnsiString,AnsiString> BuffRow("SPORTSMEN");
ListA findparam;
if(_ID_BD!=""){
       findparam["ID_SPORSMEN"]= _ID_BD;
       BuffRow.find(&findparam);
}
    if(ComboBox1->ItemIndex!=-1){
       BuffRow["ID_VID_SPORTA"]=VidsportaIndex[ComboBox1->ItemIndex];
     }else{
       ShowMessage("Выберите вид спорта");
```

```
      return;
     }
     if(ComboBox2->ItemIndex!=-1){
      BuffRow["ID_TRENERA"]=TrenerIndex[ComboBox2->ItemIndex];
     }else{
      ShowMessage("Выберите вид спорта");
      return;
     }
     BuffRow["FIO"]=Edit1->Text;
     BuffRow["BIRTHDAY"]=Edit2->Text;
     BuffRow["TELEPHONE"]=Edit3->Text;
     BuffRow["RAZRYAD"]=Edit4->Text;
     BuffRow["GTO"]=Edit5->Text;
     BuffRow.save();
     Form11->intTable();
     this->Close();
}
```