Қазақстан Республикасының Білім және ғылым министрлігі

С. Торайғыров атындағы Павлодар мемлекеттік университеті

Физика, математика және ақпараттық технологиялар факультеті

050703 «Ақпараттық жүйелер» мамандығының студенттеріне

Деректер базасының жүйесі пәні бойынша

ЗЕРТХАНАЛЫҚ ЖҰМЫСҚА

АРНАЛҒАН

ӘДІСТЕМЕЛІК НҰСҚАУЛЫҚ

Павлодар

Лист утверждения методического указания для лабораторных работ студентов

Ф

ФСО ПГУ 7.18.2/11

БЕКІТЕМІН
ФМ және АТ факультетінің деканы
_____ Тлеукенов С.К.
«__»_____200 ж.

Құрастырушы: Аға оқытушы Ақанова А.С.

Информатика және ақпараттық жүйелер кафедрасы
Деректер базасының жүйесі пәні бойынша
050703 «Ақпараттық жүйелер» мамандығының студенттеріне

зертханалық жұмысқа арналған

ӘДІСТЕМЕЛІК НҰСҚАУЛЫҚ

Зертханалық жұмысқа арналған әдістемелік нұсқаулық «__»_____200 жылы бекітілген жұмыс жоспары негізінде әзірленген.

Кафедраның отырысында қарастырылған
«__»_____200 ж. №_____ хаттама

Кафедра меңгерушісі _____ Ж.К.Нұрбекова

ФМ және АТ факультеттің әдістемелік кеңесінде құпталған
«___»_____200 ж. №_____ хаттама

ӘК төрайымы _____ А.Т.Кишубаева

Зертханалық жұмыс №1.

Тақырып: Деректер қорын құру және индекстеу**.**
Мақсаты: Database Desktop утилитасын қолданып кесте құру оны Delphi-мен байланыстыру.

Мысал №1. «Телефон анықтамасы» деректер қорын құру және оны индекстеу.

Қарапайым электронды Телефон анықтамасының деректер базасын құрудың мысалын қарастырайық. Әр адам үшін оның аты-жөні, тегі, туған жылы, күні, жынысы, үйдің және жұмыс телефоны жазылады. Адамдар жайлы мәліметтерді сақтау үшін People атты кестені пайдаланамыз.

People деректер базасының кестесінің структурасының сипаты:

Ключ	*Имя поля*	*Тип*	*Размер, байт*	*Дополнительно*	*Описание*
#	IDPeople	Счетчик		Обязательное, ключевое	Идентификатор записи
	Family	Строка	30	Обязательное	Фамилия
	Name	Строка	15		Имя
	SecName	Строка	15		Отчество
	Birthday	Дата			День рождения
	Sex	Логическое		Обязательное	Пол
	Notes	Memo	100		Комментарий

Бұл кестеге кілттік өріс болатын IDPeople өрісі жасанды түрде қосылды. Егер бір адамның тек бір телефоны ғана бар болса, онда біз People кестесіне Tel өрісін қосып, сонымен аяқтаумызға болар еді. Бірақ бір адамның бірнеше үй және жұмыс телефондары болуы мүмкін. Егер бір ғана кестені қолдансақ, онда кестеде адамдар жайлы мәлімет жолдары телефондар санына сәйкес болу керек. Ол өте ыңғайсыз және қателер санының өсуіне әкеліп соғады. Сондықтан телефондар жайлы мәліметтер бөлек People кестесімен байланысқан Tel кестесі қолданады.

Tel деректер базасының кестесінің структурасының сипаты:

Кілт Ключ	*Өріс атауы Имя поля*	*Тип Түрі*	*Размер, Байт Өлшемі*	*Дополнительно Қосымша*	*Описание Сипаттау*
#	IDTel	Счетчик		Ключевое	Идентификатор записи
	IDPeople	Число	4	Обязательное, индексированное	Идентификатор человека
	Number	Строка	9	Обязательное	Номер телефона
	TypeTel	Строка	4		Тип телефона

IDPeople – басқа кестедегі счетчик типті өріспен байланыс орнату үшін қолданылатын өріс ұзын бүтін және индектелген типті болу қажет. Сөйтіп, «Телефон анықтамасы» деректер базасының құрылымы екі байланысқан People және Tel кестелерінен тұрады. People кестесі негізгі болады, ал байланыс IDPeople өрістері арқылы орнатылады.

Біздің мәліметтер қорымызды құруға кірісейік:

1. DataBase Desktop 7.0 бағдарламасын қосайық. File|New|Table командасын орындаңыз. Нәтижесінде форматтар тізімінде Paradox 7.0 мәні бар Create Table терезесі ашылады. Біз кестені Paradox 7 форматында құратындықтан бұл терезеде OK батырмасын басамыз. Нәтижесінде кестенің құрылымы анықталатын Create Paradox 7 Table (Untitled) терезесі ашылады. Біріншіден People кестесін құрыңыз.
2. Кестенің өрістерін анықтаңыз.

Құрылымды сипаттаудың бірінші жазбасында Field Name бағанында IDPeople мәнін анықтап келесі өріске өту үшін Tab батырмасын басыңыз.
Кейін Type бағанында пробелды басып мәндер тізімінен +(Autoincrement) мәнін таңдап келесі өріске өту үшін Tab батырмасын басыңыз.
Келесі Key бағанында ағымды өрістің кілттік екенін анықтау үшін пробелды басыңыз. Кейін бұл бағанда жұлдызша белгісі пайда болады.
Тышқан тетігі арқылы Required field (міндетті өріс) қосыңыз. Кейін Key бағанын таңдап Tab батырмасын басыңыз.

Келесі өрістер сәйкесінше типтерді таңдап толтырылады; тек Family, Name, SecName және Notes өрістері үшін Size бағанында сәйкес ұзындығын анықтау қажет. Ал Required field Family және Sex өрістері үшін қосылады. Келесі өрістер кілттік болмайтынын және олар үшін Key бағаны толтырылмайтынын белгілеп кеткен жөн. Birthday өрісі үшін Picture (Шаблон) параметрін келесідей орнату қажет. Енгізуді өңйлататын және тексеру үшін Assist (көмекші) батырмасын басамыз. Мұнда Picture жолында шаблонның тексті енгізіледі. Оның дұрыстығын Verify Sintax батырмасы арқылы тексеруге болады. Ал Sample Value жолына енгізілген шаблонға сәйкес мысал енгізіледі. Тексері Test Value батырмасын басу арқылы жүзеге асады. Батырманы басқаннан кейін тексерудің нәтижесін хабарлама түрінде көруге болады. #[#]:#[#]:#[#] {AM,PM}; шаблонын таңдаңыз. Мұндағы #[#]:#[#]:#[#] – уақыт, дата – түсетін тізімнен Sample Picture-ны таңдап, Use батырмасын басыңыз. Енді Picture енгізу жолында {AM,PM}; бөлігін өшіріп тастаңыз. Себебі бізге тек дата ғана керек. Кейін OK батырмасын басыңыз.

3. Кестені Save As батырмасы арқылы People.db атымен сақтаңыз. Кейін Options тобынан Display Table-ды активтеу қажет. Ол кестені сақтаған соң кесте мәліметтермен жұмыс істеу үшін экранда көрсетілу үшін қажет.
4. Ашық ксте оны тек мәліметтерді қарауға ғана рұқсат етеді. Table|Edit Data командасын не F9 батырмасын басыңыз. Енді кестеге бірнеше жазуларды енгізіңіз.
5. Енді Tel кстесін құрыңыз. Ол үшін People кестесін құрғандағы әрекеттерді Tel кестесіне сәйкес орындаңыз. Мұнда тек бір ғана өзгешелік бар. Мұнда IDPeople өрісінде екінші индекс құру қажет. Ол келесідей орындалады:

5.1 Table Properties түсетін тізімде Secondary Indexes қасиетін таңдаңыз.
5.2 Пайда болған Define батырмасын басыңыз.
5.3 Define Secondary Indexes терезесінде Field тізімінде IDPeople мәнін таңдап OK батырмасын басыңыз.
5.4 Пайда болған Save Index As терезесінде IDPeopleIndex жолын енгізіп OK батырмасын басыңыз.
6. Кестені Tel.db атымен сақтаңыз.
7. Кестелер арасында байланыс орнатыңыз.
7.1 «Телефон анықтамасы» деректер базасының кестелерін сақтаған буманы жұмыс бумасы ретінде орнатыңыз. Ол үшін File|Working Directory командасын орындап, ашылған терезеде сәйкес буманы таңдаңыз.
7.2 Tel.db кестесін ашыңыз.
7.3 Table|Restructure командасын орындап, нәтижесінде сіздің алдыңызда кестенің құрылымын анықтайтын керезе ашылу тиіс.
7.4 Table Properties түсетін тізімде Referential Integrity қасиетін таңдаңыз.
7.5 Пайда болған Define батырмасын басыңыз.
7.6 «Referential Integrity» терезесінде Field тізімінде IDPeople[I] мәнін таңдаңыз да тізімнің қасындағы оңға бағытталған көрсеткішті басыңыз.
7.7 Table тізімінде People мәнін таңдаңыз да тізімнің қасындағы солға бағытталған көрсеткішті басыңыз. Терезенің ортасында кестелердің байланысы схема түрінде көрсетіледі. Кейін «Referential Integrity» терезесінде OK батырмасын басыңыз.
7.9 «Save Referential Integrity As» терезесінде PeoplesTel байланыс атауын жазып OK батырмасын басыңыз. Нәтижесінде кесте құрылымын анықтайтын терезенің оң жағында орналасқан тізімде енгізілген атау пайда болады.
7.10. Құрылымды анықау терезесінде Save батырмасын басыңыз.
8. Table|Edit Data ны пайдаланып кетеге бірнеше жазулар енгізіңіз.
9. Tel.db кестесін жабу үшін File|Close командасын орындаңыз.

Мысал №2. «Телефон анықтамасы» деректер базасының кестелеріндегі мәліметтерді өңдеу.

1. People.db құжатын ашыңыз.
2. Table|Edit Data командасын орындаңыз. Енді кестедегі мәліметтерді өңдеуге болады.
3. Кестедегі кез келген жазбаны активтеп өңдеңіз.
4. Сөйтіп бірнеше жазбаларды өңдеңіз.
5. Insert немесе Record|Insert командаларының көмегімен жазбаны орнатыңыз. Жазба қазіргі кезде активтелген жазбадан соң орнатылады.
6. Кез келген жазбаны өшіріп тастаңыз. Ол үшін кез келген жазбаны активтеп Ctrl+Del немесе Record|Delete командаларын орандаңыз. Бірақ физикалық өшіру орындалу үшін бізге Pack Table тетігін активтеу керек. Ол үшін Table|Restructure-ге кіріп тетікті активтеп Save батырмасын басыңыз.
7. Record менюінің командаларымен кесте ішінде орынауыстыруды қарап көріңіз.
8. Tel.db кестесімен де сондай әрекеттерді орындаңыз. Кестелердегі мәліметтерді өзгерту кезінде қателер жиі пайда болады. Ол екі кестенің байланыс нәтижесінде болады. Сондықтан мәліметтерді форма түрінде өңдеген жөн. Оның тиімділігін келесі зертханалық жұмыстарда көресіз.

Мысал №3. «Телефон анықтамасы» деректер базасының People кестесінің мәліметтерін сұрыптау.

1. Tools|Utilities|Sort командасын таңдаңыз.
2. Ашылған «Select Table» сұхбат терезесінде People.db құжатын таңдаңыз. Open батырмпасын басыңыз. Нәтижесінде «Sort Table:...People.db» терезесі ашылады.
3. осы терезеде Sorted Table тобында New Table тетігі активтелген. Онда сұрыптаудың нәтижесін сақтайтын құжаттың атын енгіземіз, яғни People-s.
4. Сол топтың Display sorted table тетігін сұрыптаудың шарттарын белгілеген соң оның нәтижесін көру үшін активтеңіз. Ал Sort just selected fields тетігін сол қалпында қалдырыңыз себебі біз барлық кестені сұрыптаймыз. Бұл дегеніміз Sort Oder тізімінде ғана емес, Fields тізімінде анықталған барлық өрістер бойынша.
5. Терезеде тағы екі тізім бар, олар сол – Fields – кестеге кіретін барлық өрістер атауы бар, оң - Sort Oder – сұрыптау жүргізілетін өрістер атауы бар тізімдер. Сол жақтағы тізімнен Family өрісін таңдап, тізімнің қасындағы оңға бағытталған көрсеткішке басып оны оң жақтағы тізімге орналастырыңыз.
6. Біз кестені Family өрісі бойынша және өсу ретімен сұрыптайтын боламыз. Оған Sort Oder тізімінің алдында + белгішесі тұрғаны сәйкес. Керек болса, біз сұрыптау түрін Sort Direction батырмасы арқылы өзгертуімізге болады. Мұнда тағы өрістердің ретін Change oder батырмасы арқылы өзгертуге болады.
7. Барлық әрекеттерді орындап болған соң OK батырмасын басыңыз. Сіздің алдыңызда Фамилиялар өрісі алфавит бойынша сұрыпталған кесте шығады.

№ 1 зертханалық жұмысқа арналған тапсырмалардың нұсқалары.

1. Бірнеше кестеден тұратын деректер қорын жобала және ол кестелер өзара 1:1, 1:N (N:1)байланыс түрлерімен байланысу керек.

1нұсқа.

Қарапайым қойма жүйесіне арналған мәліметтер жинақтау. Деректер қорында келесі ақпараттар болу қажет: тасымалдаушының ерекше нөмері, фамилиясы, аты, әкесінің аты, тасымалдаушының тұрғын қаласы (орны), және құралдың ерекше нөмері, аталуы, түсі, салмағы және осындай түрдегі құралдары бар қалалардың аты.

2 нұсқа.

Сәнді билер конкурсна қатысушылар туралы мәліметер. Деректер қорында келесі ақпараттар болу қажет: фамилиясы, аты, әкесінің аты, қаласының аты, тренердің аты – жөні, әр бидің бағасы.

3 нұсқа.

Студенттердің оқу үлгерімі туралы ақпарат. Деректер қорында келесі ақпараттар болу қажет: фамилиясы, аты, әкесінің аты, тобының нөмірі, пән аттары, тапсырманың нөиері, тапсырманың күрделігінің коэффициенті, тапсырма үшін алған баға (0-1 дейін)

4 нұсқа.

Жұмысшылардың бір айлығы туралы мәлімет. Деректер қорында келесі ақпараттар болу қажет: фамилиясы, аты, әкесінің аты, цехтың аты, жұмысқа кірген уақыты. Айлық туралы келесі мәліметтер болу керек: көлемі, жұмысшының стажы, оның разряды және қызметі.

Зертханалық жұмыс №2.

Тақырып: Бағынышты ішкі форма(подформа), өрістерге түзетулер жасау**.**
Мақсаты: Бағынышты ішкі форманың (подформа) өрістеріне түзетулер енгізуді үйрену.

Мысал №1. «Телефон анықтамасы» деректер базасының мәліметімен жұмыс істеу үшін подформасы бар форма құру.

1. Басты менюдің File|New Application командасы арқылы жаңа проекті ашамыз. *Caption* қасиетін «Информация о людях» деп өзгертеміз. Бұл атау біздің деректер базасының атауы болады.
2. Форманы MyExunitDB.pas атауымен сақтаймыз, ал проекті MyExampleDB.dpr деп сақтаймыз. Ол үшін File|Save Project As командасын орындаймыз. Онда құжат атауын таңдайтын терезе ашылады, бұнда біз модуль үшін буманы таңдаймыз. Модульге MyExUnitDB деген атау береміз де кейін проектіге MyExampleDB атауын береміз.
3. Form1 формасына People.db құжатында орналасқан компоненттерді байланыстыру үшін компоненттерді орналастырамыз.

DataAccess парағынан Table компонентін орналастырып TableName қасиетіне People.db-ны орналастырамыз. Мұнда бағдарлама мен People.db кестесі байланыты. Кейін Table1 компонентінің Active қасиетін True деп орнатамыз. Ол Кестенің мәндерін көрсетеді.

Осы беттен DataSource компонентін орналастырамыз. Ол компоненттерді мәліметтермен байланыстырады. DataSource1 компонентінің DataSet қасиетін Table1 деп белгілейміз. Бұдан кейін компоненттер кестедегі мәліметтерге рұқсат алады.

4. Tel.db кестесіндегі мәліметтермен байланыс орнату үшін алдыңғы пунктегі әрекеттерді орындап, тек Table2 және DataSource2 компоненттері Tel.db кестесімен байланытады да, DatabaseName және DataSet қасиеттері сәйкесінше Table2 және DataSource2 мен байланысты.
5. Деректер базасының мәліметтерін екі кестеден көрсететін подформасы бар форма құру үшін деректер жиындар арасында байланыс орнатамыз. Ол үшін Table2 компонентін ерекшелеп, MasterSource қасиетіне DataSource1 мәнін орнатамыз. Тышқан тетігі арқылы MasterFields қасиетін активтейміз. Available Index енгізу жолында IDPeopleIndex мәнін орнатамыз, себебі байланыс индекстелген өрістер арқылы жүзеге асады. Кейін сол жақтағы тізімнен IDPeople мәнін және оң жақтағы тізімнен IDPeople мәнін таңдап Add батырмасын басыңыз. Сонымен байланыс орнатылды, OK батырмасын басыңыз. Нәтижесінде IndexName қасиеті IDPeopleIndex ал байланыста қатысатын IDPeople өріс аты MsterFields қасиетінде көрсетіледі.
6. Формаға деректер базасын көрсететін компоненттерді орналастырамыз. Ол үшін келесі әрекеттерді орындаймыз:

Standart бетінен Label компонентін орналастырып, Caption қасиетін «Фамилия, имя, отчество» деп өзгертеміз.

DataControls бетінен 3 DBEdit компонентін Label1 компонентінің астына қатарынан орналастырамыз. Барлық компоненттер үшін DataSource қасиетін DataSource1, ал DataField қасиетін сәйкесінше Family, Name және SecName деп орнатамыз.

Standart бетінен тағы бір Label компонентін таңдап оны DBEdit1 компонентінің астына орналастырып, Caption қасиетін «День Рождения» деп ауыстырамыз.

DataControls бетінен DBEdit компонентін орналастырамыз. DataSource қасиетіне DataSource1 мәнін, ал DataField қасиетіне Birthday қасиетін орнатамыз.

Бұл жлдың оң жағына DataControls парағынан DBCheckBox компонентін орналастырып, Alignment қасиетін LeftJustify, ал Caption қасиетін – «Пол» деп орнатамыз. Кейін оның DataSource қасиетіне DataSource1 мәнін, ал DataField қасиетін тізімнен Sex мәнін таңдаймыз.
Label2 компонентінің астына Standart бетінен тағы бір Label компонентін орналастырып, Caption қасиетін «Комментарий» деп өзгертеміз.
Бұл компоненттің астына DataControls бетінен DBMemo компонентін орналастырамыз. Оның өлшемін горизонталь өлшемін бүкіл формаға созып, ал вертикаль өлшемін кішкене қысқартамыз. Оның DataSource қасиетін DataSource1 деп онратамыз, ал DataField қасиетін Notes мәнін таңдаймыз.
DataControls бетінен форманың жоғары оң жағына DBNavigator компонентін орналастырамыз. Оның DataSource қасиетін DataSource1 деп онратамыз.
7. DataControls бетінен формаға DBGrid компонентін формадағы барлық компоненттерден төмен орналастырамыз. Оның вертикаль өлшемін сәл қысқартамыз. Оның DataSource қасиетін DataSource2 деп онратамыз. Егер жоғарыдағы әрекеттер дұрыс орындалса, онда DBGrid1 компонентінде Tel.db кестесінің мәліметтері шығарылады.
8. DBGrid компонентінде көрсетілетін өрістер тізімін анықтаймыз. Ол үшін келесі әрекеттерді орындау қажет:
8.1 Тышқан тетігі көмегімен Columns қасиетін активтеп, Editing DBGrid1.Columns кестесінің өрістер тізімінің редакторын шақырамыз.
8.2 Бұнда Add батырмасын басамыз. Пайда болған «0-TColumn» жазбасын ерекшелейміз. Объектілер инспекторы көмегімен FieldName қасиетіне Number мәнін орнатамыз. Күрделі Titel қасиеті үшін Caption параметріне Номер мәнін береміз.
8.3 Кестенің өрістер тізімінің редакторының терезесінде Add батырмасын басамыз. Пайда болған «1-TColumn» жазбасын ерекшелейміз. Объектілер инспекторының көмегімен FieldName қасиетіне Type мәнін орнатамыз. PickList қасиетін активтеп, пайда болған текстік редакторда келесі екі жолды енгіземіз:
Дом. Раб.
Осы әрекеттерден кейін OK батырмасын басамыз. Сөйтіп біз телефон типін анықтайтын мәндер тізімін анықтадық. Кейін күрделі Titel қасиеті үшін Caption нараметріне Тип мәнін анықтаймыз.
8.4 Кестенің өрістер редакторымен жұмысты аяқтау үшін терезенің шетіндегі стандартты батырманы басу жеткілікті.
9. Standart бетінен формаға Label компонентін DBGrid1 компонентінің жоғары сол жағына орналастырамыз да оның Caption қасиетін Телефоны мәніне ауыстырамыз.
10. DataControls бетінен формаға DBNavigator компонентін орнатамыз. Оны DBGrid1 компонентінің жоғары оң жағына орналастырамыз. DataSource қасиетіне DataSource2 мәнін орнатамыз.
11. Басты менюдің Run|Run командасы арқылы бағдарламаны іске қосамыз.
12. Подформасы бар формаға мәліметтер енгізудің өз ерекшеліктері бар. Оларды белгілеу үшін мәліметтерді формаға енгіземіз. Ол үшін келесі әрекеттерді орындауымыз қажет. Деректер базасының жазбаларындағы навигатордың панелінде «Плюс» батырмасын басыңыз. Бұдан кейін People кестесінде жаңа жазба пайда болады, оған адам жайлы мәлімет енгіземіз. «Фамилиясы, Аты, Әкесінің аты» өрістеріне сәйкес «Арманов» «Канат» «Мурзатович» мәндерін енгізіп, «День рождения» өрісіне 14.01.79 мәнін енгіземіз. «Пол» тетігі қосылған болу қажет. Навигатордың «☑» батырмасын басамыз. Құрылған жазба People кестесінде сақталады. Адам туралы мәліметтерді телефонды енгізудің алдында сақтау қажет, әйтпесе олар сол адамға сәйкес болмайды, ал деректер базасында артық жазбалар саны өте көп болып кетеді. Пайдаланушы ондай әрекетті орындай алмау үшін DBGrid1 компонентіне мәліметтерді енгізуге және DBNavigator2 батырмасын баскға рұқсат етпеу қажет. Ол үшін келесі әрекеттерді орындау қажет:

12.1.1 бағдарламаның жұмысын тоқтатыңыз. Формадағы DBNavigator1 компонентін таңдап, объектілер инспекторындағы Events қалтасына өтіңіз.

12.1.2 BeforeAction жолында тышқан тетігінің оң жақ батырмасын екі реет шертіп, MyExUnitDB.pas терезесінде келесі процедураны толықтырыңдар:

Form1.DBNavigator1BeforeAction келесі фрагментпен:

```
if ((Button=nbInsert) or (Button=nbEdit)) then
begin
```

```
DBNavigator2.Enabled:=false;
DBGrid1.Enabled:=false;
end;
if Button=nbPost then
begin
DBNavigator2.Enabled:=true;
DBGrid1.Enabled:=true;
end;
```

12.1.3 Өзгертулерді сақтап бағдарламаны іске қосыңыз.

12.2 Төменгі навигатор панеліндегі «+» батырмасын басыңыз. Tel кестесінде жаңа жазба құрылады. «Номер» ұяшығына телефон номерін енгізейік. «Тип» ұяшығында ашылатын тізімде «дом» типін таңдайық.

12.3 Төменгі навигаторда «+» батырмасын басамыз. «☑» батырмасын сақтау үшін басу қажет емес, себебі енгізу кезінде мәліметтер автоматты түрде сақталады.

13. Бағдарламаның жұмысын терезі жабу батырмасын басумен аяқтаймыз.

14. File|Save All командасы арқылы барлық өзгертулерді сақтаймыз.

Ескертулер:

1. Егер бір жазбаны өшіріп тастау керек болса, онда сәйкес жолды белгілеп навигатордағы «-» батырмасын басыңыз. Ал жазбаларға өзгертулер енгізу қажет болса, сәйкес элементті таңдап төменгі навигатордың «Стрелка вверх» батырмасын басу қажет. Барлық өзгертулерді енгізіп болған соң «☑» батырмасын басамыз.
2. People кестесінен жазбаны өшіріп тастау үшін біріншіден Tel кестесіндегі сәйкес барлық жазбаларды өшіріп тастау қажет. Ондай өшірулерді қолмен жасамау үшін Form1.DBNavigator1BeforeAction процедурасын келесі фрагментпен толықтыру қажет:

```
if Button=nbDelete then
begin
Table2.First;
while not Table2.Eof do
begin
Table2.Delete;
end;
end;
```

Мысал №3. «Телефон анықтамасы» деректер базасының формасын құру кезінде өрістер редакторымен жұмыс.

1. File|Open командасы арқылы MyExUnitDB.dpr құжатына енгізілген проекті ашамыз.
2. Жазбалардың қосымша реттік номерлерін шығаратын боламыз (IDPeople өрісінің мәндерін). Ол үшін келесі әрекеттерді орындау қажет:

2. 1 Form1 формасында Table1 компонентін активтейміз. Пайда болған редакторда Add Fields командасы арқылы барлық өрістерді көшіреміз. Пайда болған өрістер тізімінен IDPeople өрісін ерекшелейміз. Объект инспекторында DisplayFormat қасиетіне келесі жолды енгіземіз:

‘Порядковый номер: ‘000

2. 2 DataControls бетін пайдаланып Form1 формасының жоғары сол жақ бұрышына навигатордың алдына DBEdit компонентін орналастырамыз. Компоненттің горизонталь өлшемін шығарылатын мәлімет толығымен сиятндай сәл үлкейтеміз. Объект инспекторын пайдаланып DataSource қасиетіне DataSource1 мәнін меншіктеп, DataField қасиетінде тізімнен IDPeople мәнін таңдаймыз. Егер бәрі бұрыс орындалатын болса, DBEdit5 жолында сәйкес жазба пайда болу керек.

3 Алдыңғы мысалдарда біз «Жыныс» өрісін DBCheckBox1 компоненті арқылы шығардық. Ол сонша ыңғайлы емес. Бұл мәнді «Ер» немесе «Әйел» мәндерін таңдау арқылы шығарамыз. Ол үшін келесі әрекеттерді орындау қажет:

3.1 Form1 формасында Table1 компонентін активтейміз. Өрістер тізімінде Sex өрісін ерекшелейміз. Объект инспекторындағы DisplayValues қасиетіне келесі жолды енгіземіз:

Ер; Әйел

3.2 Формадан DBCheckBox1 компонентін оны ерекшелеп Del батырмасы арқылы өшіріп тастаймыз. DataControls бетін пайдаланып оның орнына DBComboBox компонентін орналастырамыз да горизонталь өлшемін сәл қысқартамыз. Объект инспекторын көмегімен

DataSource қасиетіне DataSource1 мәнін, ал DataField қасиетіне Sex мәнін меншіктейміз. Егер барлық әрекеттер дұрыс орындалған болса, DBComboBox1 компонентінің жолында сәйкес жазба шығу тиісті (Мужской не Женский).

DBComboBox1 компонентінің Items қасиетін активтеп, пайда болған редакторда келесі жолдарды енгіземіз:

Ер

Әйел

Редатордың жұмысын OK батырмасын басымен аяқтаймыз.

3.3 Standart бетінен формаға DBComboBox1 компонентінің сол жағынан Label компонентін қойып, оның Caption қасиетіне «Жыныс» мәнін енгіземіз.

4. Енді туған күнді шығаратын дата үшін форматты таңдау керек. Ол үшін келесі әрекеттерді орындау керек:

4.1 Form1 формасындағы Table1 компонентін активтейміз. Өрістер тізімінде Birthday өрісін таңдаймыз.

4.2 Объект инспекторында DisplayFormat қасиетін таңдап, бір форматты енгізу керек, мысалға «dddddd». Енгізуді Enter батырмасын басумен аяқтаймыз.

5. Run|Run басты менюінің командасы арқылы бағдарламаны іске қосамыз. Экранда біз енгізген барлық өзгертулер көрсетіледі.

6. Жазбалар арасында орын ауыстырғанда қосылған енгізу жолында реттік номері рпайда болатынын атап кеткен жөн. Ал комбинилренген жолда дамның жынысы жайлы мәлімет орналасады. Бұл мәнді қажет болса ашылатын тізім арқылы өңдеуге болады.

7. Бағдарламаның жұмысын терезені жабумен аяқтаймыз.

8. File|Save All командасы арқылы барлық өзгерістерді сақтаймыз.

Зертханалық жұмыс №3

Тақырып: Есеп беруді жобалау.

Мақсаты: QuickReport2.0 пакетін қолдануын үйрену.

1. Мәліметтер қоры ақпаратты өңдеу үшін қолданылады. Өңдеулердің қорытындылары есептерде көрінеді. Есеп – принтерден басылатын құжат, кейін оның негізінде шешімдер қабылданады. Делфиде есеп мәліметтер қорынан арнайы ресімделген ақпаратты білдіреді және ерекше форма түрінде көрінуге арналған. Оны экраннан көруге болады, ал содан кейін принтерден басылады. Осындай форма мәліметтерді және және және басу кезінде есептің сыртқы түрін анықтайтын арнайы компоненттерді құрайды. Мәліметтерді ұсынуды формадан есепке ауыстыру есеп генераторы көмегімен жүзеге асырылады. Делфи 3.0 пакетінде **QuickReport2.0** есептер генераторы бар, ол есепті басу және файлдан оқу, файлда сақтау, қарау мүмкіндігін туғызады. Есептер формалар сияқты әр түрлі болуы мүмкін:
2. - **Тізім** – жол теру арқылы ақпаратты шығарады, оның әрбіреуі бір жазбаны білдіреді.
3. - **Бланкіде** бір жазба бір парақта орналасады.
4. - **Есеп есеп берушілікпен** бір кестеден екінші кестеге қосымша ретінде ақпаратты бейнелейді.
5. - **Пошталық этикеткаларды** басу есебі парақта есепті бірнеше үлкен емес тікбұрышты облыстарды орналастыра алады, кейін олар пошта бойынша жіберілетін заттарға жабыстырылады.
6. **Есеп құрастыру кезінде** бірнеше сатыларды айқындауға болады.
7. **Есепті дайындау сатысы.** Бұл сатыда есептің компоненттері және олардың параметрлері таңдалады.
8. **Есепті құру сатысы**. Бұл сатыда таңдалған компоненттерден есеп қалыптасады. Ол есеп генераторы көмегімен орындалады.
9. **Есепті қарау сатысы.** Бұл сатыда экраннан алынған есептің түрін қарастыруға болады және есептің дайындық сатысына оралып мүмкіндігінше оған өзгерістер енгізуге болады.
10. **Есепті басу сатысы.** Бұл сатыда есепті принтерден басу жүзеге асырылады.
11. **Есепті мәліметтер қорында құру.**
12. Әрбір есеп келесі бөлімдерден құралуы мүмкін: есептің аты, есеп бетінің аты, топталған мәліметтердің, мәліметтер облыстарының, есеп берушілік үшін облыстың аты, топталған мәліметтердің аяқталуы, есеп бетінің аяқталуы, есептің аяқталуы. Сонымен қатар **QuickReport** бірнеше бағаналары бар есепте әрбір бағананың аты және әрбір бетте қайталанатын фрагментті

басуға мүмкіндік туғызады. Осының барлығы жоғарыда айтылған компоненттердің көмегімен құрылады.

13. **Басуға тапсыруды өңдеу. TQRPrinter принтер классы.**
14. **TQRPrinter** классы **TPersistent** классының тармаға болып табылады. Бұл кластың компоненті көрінбейтін болып табылады. Бұл объектінің данасы қосымшаны орындау кезінде автоматты түрде құрылады және **QuickRep** компонентінің болуынсыз пайдаланыла алады. **TQRPrinter** атына ие бұл объект есепті құруды, алдын ала қарауды, басуды, сонымен қатар файлда есепті сақтауды және онымен әрі қарай жұмыс істеуді қамтамасыз етеді. Негізінен бұл қызметтер мен мүмкіндіктер есепті қарау бойынша, **TQRPrinter** класымен қамтамасыз етілген, **QuickReport** 2.0 пайдалануды алдын ала қару үшін стандартты формада ұсынылған.
15. Есеп бейнелеген бұл форманы бірнеше тәсілдермен шақыртуға болады. Есепті жобалау кезінде **QuickRep** компонентін таңдап, содан кейін тышқанның оң жақ кнопкасын басып, шыққан мәзірден *Preview* командасын таңдаймыз. Есепті қарау стандартты терезеде көріну үшін қосымшаны орындау кезінде *Preview* тәсілін қолданған жөн. Стандартты терезеде барлық қызметтер кнопкалық панельдің көмегімен жүзеге асырылады.
16. Беттердің санын көрсететін көрсетілген бет нөмірінің индикаторы қараудың стандартты терезесінің жолында орналасады. Сол сияқты есептің қалыптасу процесінің индикаторы да көрінеді. **TQRPrinter** классы бірнеше қосымша қасиеттерді, тәсілдерді анықтайды, бірақ біз оларды қарастырмаймыз.
17. **Есепті тез қалыптастыру.**
18. Есепті тез қалыптастыруды сарапшы ретінде емес, анықталған түрдің дайын өнімі ретінде қарастыруға болады. Бұл дайын өнімде ақпарат есептен шығарылатын мәліметтер көзімен байланыс жоқ. **Delphi Standart 3.0**-де мәліметтер көзін анықтау үшін, есеп формасына орналастырылған Table компонентінің мінездемесін анықтау қажет.
19. Келесі операциялар есептерді модификациялау үшін, сәйкес дайын өнімдер жасауға көмектеседі:
20. **File/New/Forms/QuickReport Labels**- пошталық зтикеткаларды басу үшін есептің дайын өнімі жасалады. Бұл есеп үш бағанадан тұрады, оларда тікбұрышты пошталық этикеткалар шығарылады.
21. **File/New/Forms/QuickReport List**- тізім түрінде ақпарат шығарылатын есептің дайын өнімі шығарылады, оның құрамына есеп атауының бөлігі, беттер атауының бөлігі, мәліметтер қорытындысының және беттің аяқталу бөлігі кіреді.
22. **File/New/Forms/QuickReport Master/Detail**- есеп берушілікпен дайын өнімі жасалады. Оның құрамына есеп атауының бөлігі, негізгі есептің мәліметтер қорытындысының бөлігі, есеп берудің мәліметтер қорытындысының бөлігі, есеп беруде қорытындыларды шығару және беттің аяқталу бөліктері кіреді. Айтылған бөліктер тізім бойынша жасалынады және мүмкіндігінше оларға басқаларды қосуға болады.
23. Сонымен қатар есептің бос дайын өнімінің формасын жасау мүмкіндігі бар. Ол үшін **File/New/New/Report** операциясын қолданған жөн. Қорытындысында **QuickReport** компоненті бар форма шығады.
24. **Report Wizard**- есепті құрастыру бойынша сарапшы.
25. **Report Wizard**- диалогтық терезесі бар сарапшы, онда есепте қандай ақпарат шығарылатынын анықтау қажет. Және де ақпарат тізім түрінде шығарылатын есеп жасалады. Бұл сарапшы локальды мәліметтер қорының негізінде есептерді жасауға мүмкіндік туғызады. Кез келген қадамда Cancel кнопкасын басып есеп жасау процесін тоқтатуға болады, сонымен қатар кез келген диалогтық терезеден (біріншісінен басқа) <Back кнопкасын басып алдыңғысына қайтып оралуға болады. Бір сарапшы терезеден екіншісіне Next> кнопкасын басып ауыстыруға болады.
26. Осы сарапшыны операциясы арқылы шақыртуға болады **File/New/Bisuness/QuickReport Wizard.**

Зертханалық жұмыс № 4

Қосымшаны жобалау кезінде сұраныстарын пайдалану.

Мәліметтер қорымен жұмыс істеген кезде оған сұраныс ұғымы қолданылады. Сұраныс ұғымына мәліметтер қорынан ақпаратпен бірге белгілі бір қызмет жасау кіреді. Сұралған шартқа сәйкес мәліметтерді таңдауға болады және қорытындыны – таңдау сұранысын шығаруға болады. Мәліметтер қорында сәйкес сұранысларды қолданып, белгілі бір жазбаларды қоюға, жоюға, жаңартуға болады. Осындай талаптар бағдарламалаудың аранайы **QBE** және **SQL** тілдерінде құралады. SQL делфиде кең таралған, оны мәліметтер қорының жұмысы үшін қосымшаны жасау кезінде қолдануға болады.

SQL –сұранысы

SQL тілінде бірнеше сұраныстарды жасауға болады:
- мәліметтерді таңдау сұранысы;
- қосымшаға сұраныс;
- жаңартуға сұраныс;
- жоюға сұраныс.

SQL –сұраныс, DBD көмегімен жасалған, кеңейтілген sql файлында орналасады. SQL тілінде әрбір сұраныс бірнеше бөліктен тұрады, олар белгілі бір кейінге сақталған сөзден басталады. Сұраныстарды жазу кезінде бас әріптер сияқы кішкентай әріптерді де қолдануға болады. Ақпаратты әртүрлі тәсілмен жолдарға орналастыруға болады, бірақ белгілі бір мағынасы бар ақпаратты жеке жолдарға орналастырған жөн. Сөздерді, идентификаторларды, сандарды, жолдарды және т.б. операцияларды бөлу үшін пробелды қолданған жөн.

Таңдауға сұраныс.

Бұл сұраныс кестенің бір немесе бірнеше өрістерінде орналасқан мәліметтерді алу үшін арналған. Сұранысдың конструкциясы келесідей:

SELECT [DISTINCT] <шығарылған өрістердің тізімі>
FROM <қолданылатын кестелердің тізімі>
[WHERE < таңдаудың шарты>]
[ORDER BY<іріктеу тәртібі>]
[GROUP BY<топтау тәртібі>]
[HAVING < топтарды таңдау шарты>]
[UNION<сұрыптауға араналған қосынды сұраныс>

Сұранысдың жеке бөліктерін анықтап қарастырайық.

DESTINCT- сұраныс нәтижесінде қайталанатын белгілердің болмау керектігін білдіреді.

<шығарылатын өрістердің тізімі>:

[<кесте>.] <өрістер>[AS<жасырын ат>] {, [<кесте>.] <өрістер>[AS<жасырын ат>] }, мұнда кесте – кестенің атауы, егер сұранысда бір кесте болса, онда оны қолданбауға болады; өріс – өрістің атауы; жасырын ат – өрістің атауы үшін жасырын ат, сұранысда локальды болып келеді; AS – нақты өріске жалған ат қоятын оператор. Егер кестенің барлық өрістерінен мәліметтерді алу керек болса, онда барлық өрістердің аталған атаулары орнына * көрсеткен жеткілікті.

<пайдаланылатын кестелердің тізімі>:

“< кестемен файл >”,[< жасырын ат >]{,” кестемен файл >”[<жасырын ат>]}.

WHERE- сақталған сөз, одан шарттар тапсырмасының операторы басталады.

<таңдаудың шарты >:

Шарт, бірлікті бекітетін шарт:

[(<кестеМ. өріс1 >< шарт>< кестеN. өріс 2>)]{ AND/ OR
(<кестеI.өріс1 ><шарт>< кестеJ.өріс2>)}, мұндағы *кестеМ, кестеN, кестеI, кестеJ* – біріктірілген кестелердің атауы; *өріс1, өріс 2* – осы кестелердегі өрістердің атауы, олардың мағынасының сәйкестілігі бойынша кестелердің ара-қатынасы орын алады; *шарт* – өрісердегі екі кестенің мағынасының сәйкестілігі бойынша анықталады, мысалы, теңсіздік, көп, аз, тең емес және басқа да салыстыру операциялары; шарттар бірнеше болуы мүмкін, бұндай жағдайда олар дөңгелек жақшалар, AND және OR логикалық операциялар арқылы біріктіріледі. Егер біріктіру үшін шарт болмаса, кестелердің барлық мүмкін жазбаларының мүмкіндіктерін білдіретін нәтиже құрылады.

Өрістердің мағынасы бойынша шарттар:

[[<Кесте>.],<өріс>< шарт>< мағына>]{ AND/OR
[<Кесте>.]< өріс>< шарт>< мағына>}.

Математикалық салыстыру операторларынан басқа, логикалық салыстыру операторлары, **IN, LIKE** және тағы басқалары қолданылады.

ORDER BY- сақталған сөз, одан іріктеу тәртібінің тапсырма операторы басталады.
<іріктеудің тәртібі >:
[<кесте>.]< өріс> [ASC/DESC]{,[< кесте >.]<өріс> [ASC/DESC]}, мұндағы ASC немесе *DESC* - өсу немесе кему бойынша сәйкес іріктеудің тәртібін анықтайды.
GROUP BY-сақталған сөз, одан топтау тәртібінің тапсырма операторы басталады.
<топтаудың тәртібі >:
[<кесте>.], <өріс >{[<кесте>.]< өріс>}.

Қосуға сұраныс.

Бұл сұраныс кестеге жазба қосып, оның өрістеріне мағына бере алады. Сұранысдың конструкциясы келесідей:
INSERT INTO "<файл кестемен >"(<өріс>{,< өріс>})
VALUES (<мағына >{,<мағына>}).
Сұраныстың жеке бөліктерін анықтап қарастырайық:
Кестемен файл– кесте файлының атауы, оған жазба қолданылады; ол кеңейтілген болу керек;
Өріс – өрістің атауы, оған мағына қойылады.
VALUES- сақталған сөз, одан кестенің өрістер мағынасы тізімінің тапсырма операторы басталады; **мағына** – мағыналар тізімінен мағына, ол жаңа жазбаның сәйкес өрістеріне тапсырылады (олардың саны өрістердің санына сай келуі керек).

Жаңартуға сұраныс.

Бұл сұраныс тапсырылатын шарттарға сәйкес, жазбалардағы мағынаны жаңартуға көмектеседі. Сұранысдың конструкциясы келесідей:
UPDATE "<файл кестемен>"
SET<өріс>=< мағына>{,< өріс>=< мағына>}
[WHERE <таңдау шарты>]
Сұранысдың кейбір бөліктерін қарастырайық:
SET- сақталған сөз, одан мағыналар тізімінің тапсырма операторы басталады; <**өріс**>=< **мағына**>{,< **өріс**>=< **мағына**>} – осы өрістерге қолданылатын өрістер мағынасының тізімі;
[WHERE <таңдау шарты>]– мәліметтердің жазбалдарын жаңартатын тапсырма операторы.

Жоюға сұраныс:

Бұл сұраныс кестеден жазбаларды жоюға мүмкіндік береді. Сұранысдың конструкциясы келесідей:
DELECT FROM "<файл кестемен>"
[WHERE <таңдау шарты>].

Мәліметтер қорына сұраныс (Query)

Table компонентінің біреуін қолдану тек бір кестемен жұмыс істеуге мүмкіндік береді. Сондықтан, бірнеше байланысқан кестелермен жұмыс істеген кезде, Table компонентінің сәйкес санын жасау керек.. Бұл жағдайда *TDBDataSet* классына жататын *TQuery* класының *Query* компонентіне алған жөн. Бұл компонент **SQL**- сұранысының көмегімен бірнеше кестелердің негізінде мәліметтерді таңдауды анықтауға мүмкіндік береді. Сол сияқты мәліметтер қорының кестесі шамадан артық болғанда қолдануға ыңғайлы; бұндай жағдайда сұранысдың көмегімен қарастырылып отырған мәліметтердің таңдауын шектеуге болады. Бұл компонентаны пайдалану **SQL** тілінің білуіне байланысты.
Delphi **SQL** операторын Query компоненті көмегімен жасауға болады:
1) формаға **Query, DataSource** объектілерін енгізіп, оларды бір-бірімен байланыстыру;
2) *DataBaseName* қасиетінің *TQuery* объектісіне жалған ат қою;
3) *SQL* қасиетінің көмегімен **SQL** тапсырмасын белгілеу;
4) True мағынасында *Active* қасиетін бекіту.
Егер әрбір қадам дұрыс аяқталған болса, егер BDE нақты бекітілген болса, тор белгіленген кестеден жазбаны сақтау керек.
Статистикалық сияқты динамикалық та (немесе сұраныслар параметрлермен) Delphi мәліметтер қорына **SQL**-сұранысын жасай алады, онда сұранысдың мәтіні өзгермейтін болып қалады. Динамикалық **SQL**-сұраныслар бағдарлама орындау барысында өзгертіле алатын жазба таңдау шартының негізінде анықталған параметрлерді кіргізеді. Бұл жағдайда бір динамикалық сұранысдың көмегімен қосымшаны орындау кезінде әртүрлі нәтижелерді алуға болады. Динамикалық сұраныслардың синтаксистік конструкциялары статистикалыққа сай, бірақ оларда

жазбаларды таңдау шарттарын анықтайтын секциясында мағынасының орнына <параметр >**,** мұндағы параметр – параметрдің атауы, оның орнына қосымшаны орындау кезінде мағына қойылатын болады.
TQuery класында *TDataSet, TBDEDataSet* және *TDBDataSet* кластарынан алған келесідей маңызды қасиеттерін атап айтқан жөн:
Local- кестелердің орналасуын анықтайды (*True* – локальды кестелер, *False* – *SQL*-серверіндегі кестелер); қасиет тек қана оқуға.
RequestLive- логикалық түрдің қасиеті (False мағынасын білдіреді), сұранысды орындау нәтижесінде алынған мәліметтерді өзгерту мүмкіндігін анықтайды. Бұл мүмкіндік сұранысларда да бар, егер қасиет True мағынасына ие болса. Ал қалған сұранысларда нәтиже тек қана оқуға арналған. Бұндай жағдайларда сұраныс сәтті екенін көру үшін *CanModify* қасиетін қарауға болады, егер кестені *SQL* сұранысы бойынша түзету керек болса, онда *SQL Update* командасын қолданған жөн.
SQL – Tstrings түрінің қасиеті, *Open* немесе *ExesSQL* тәсілдерін орындаған кезде пайдаланылатын *SQL*-сұранысының мәтінін анықтайды.
UniDirectional- сұраныс орындаудың нәтижесінде алынған мәліметтерді таңдау бойынша курсордың қозғалу бағытын анықтайды.
UpdateMode - *TUpdateMod*e қасиетінің түрі, аралық буферден жазбаларды жаңарту тәсілін анықтайды. *TQuery* компонентасының келесі қасиеттері динамикалық SQL-сұранысларда пайдаланылады. Олардың кеейбіреуін мысалға келтірейік:
DataSource – TDataSource түрінің қасиеті, мәліметтер көздерін анықтайды олардың өрістерінің мағынасы динамикалық сұраныс үшін параметлер сияқты пайдаланылады.
Params [Index] - TParams түрінің қасиеті, динамикалық сұранысда параметрлерді анықтайды TParams түрінің элементтер тізімін құрайды. Бұл қасиеттің көмегімен параметлер мағынасының редакторында бастапқы параметрлерінің үлкендігі құралады. Сұранысдың параметлер мағынасының редакторына ауысу үшін бұл қасиеттің мағынасын тышқанмен жандандыру керек.
Exes SQL – процедурасы SQL сұранысын орындайды.
Prepare – процедурасы синтаксисті және оптимизацияны тексеру үшін BDE-ге сұраныс жібереді. Динамикалық сұраныслар үшін орындауға ұсынылады. TQuery компонентасымен өңделген оқиғалар толықтай TDataSet класынан алынады.
Params қасиетінің көмегімен бағдарлама тәсілімен динамикалық сұранысға ауысу байланыстарын ауыстырған кезде әдетте келесі қадамдар орындалады:
1) міндетті түрде кестенің жабық екеніне көз жеткізу керек;
2) *Prepare* командасын беру көмегімен *TQuery* объектісі дайындалады;
3) *Params* қасиетіне нақты мағыналар иеленеді;
4) Сұраныс ашылады.

Зертханалық жұмыс № 5

Жобалық мәзір

Элемент мәзірі асты немесе команда, немесе айырғыш сызығы бола алады.
Break – қасиеті ағымдағы элементтің мәзірін бағандарға бөледі.
Caption – string, қасиетінің типі, берілген мәзір элементінің мәтінің құрайды. ***Checked –*** логикалық типтің қасиеті,берілген элемент белгіленгенін анықтайды.(егер True мағынасы болса, элемент
« қанат белгімен » белгіленеді).
Default – логикалық типтің қасиеті ; егер ол True мағынасы болса ,меню элемент мәтіні ***полужирный*** жазу әріпімен ,ал тышқанымен үлкен элементті екі есе басса, ағымдағы оқиғалары False болатын, OnClick үндемеуін көруге болады.
Enabled - логикалық типтің қасиеті ; егер ол True мағынасында болса ,тышқан мен клавиатурадан тәуелді болады.
GroupIndex –бүтін типті қасиеті , меню элементі жататын топтар нөмірін қамтиды, және нөлдік мағынасы болады.

Items [Index]- TMenuItem типінің массив қасиеті (Index - бүтін көрсеткіші), ағымдағыға элементтен кіші элементке тапсырма беруі. Count қасиеті бойынша элементтер саны анықталады. Сана нөлден басталып, тек оқуға арналады.
MenuIndex – қасиеттің бүтін типі, Items тізіміндегі үлкен компоненттерді қамтитын индексі бар компонент.
RadioItem - қасиеттің бүтін типі; Егер ол True ді қабылдаса,онда мәзір элементі ауыстыру қосқышы ***(переключателя)*** болып, бір топтың бір элеметі ғана белгіленген болады.
ShortCut - қасиеттің бүтін типі,шерту кезінде мәзір элементі активтендірілетін,
жылдам басқарушы пернелердің кодын анықтайды.
Жылдам басқарушы пернелердің кодын анықтау кезінде ShortCutToKey, ShortCutToText программалары қолданылады.
Visible - қасиеттің бүтін типі; егер ол True мағынасы болса , False - жоқ болса, мәзір элементі экранда суреттеледі,
TMenuItem класында OnClick оқиғасы анықталған. Класс құрылымдардын түзетуімен байланысты әдістер қатарын қамтиды.
Add (Item)- процедурасы,Item (TMenuItem класынан) элементін кіші элементтер тізіміне қосады
Delete (Index)- процедура , кіші элементтердің тізімінен Index индексі бар элементтерді жояды.Бірақ динамикалық жадтағы элементтер жойылмайды және екіншірет қолданыла береді.
Жобалау сатыларына бір немесе басқа элементке ShortCut қасиетін қолданып, тізімнен үйлестіру лайық немесе пернетақтаның теріп,жылдам басқару пернесін беруге болады.Мына операцияның көмегімен, сатыларға бағдарлама орындалуларыды жүзеге асыруға болады.
Арнайы стандартты ішкі программалар көмегімен (ShortCut , ShortCutToKey , ShortCutToText ,TextToShortCut).
Main Menu басты мәзірі.
Менюмен әдеттегі негізгі жобалау сатыларына қалыптасады. NewItem , NewLine , NewSubMenu , NewMenu . стандартты программалар көмегімен жүзеге асыра аламыз.Жергілікті Popup Menu ,басты мәзір сияқты *TMenu* класына жатады.
Standart компонент , ал Items . қасиеті содан соң қолданылады.
TPopupMenu класында келесідегідей қасиеттері бар жаңа мінездемелерді еңгіземіз. TPopupAlignment типтес Alignment - қасиеті , жергілікті мәзірмен тышқан курсоры орналасуын анықтайды.
AutoPopup –логикалық типтін қасиеті . Егер ол
True қабылдаса, жергілікті мәзір тышқанның оң пернесін шерткенде пайда болады False болғанда , мәзір пайда болмайды (бұл жерде Popup әдісін қолданған жөн).
OnPopup - оқиға , экранға жергілікті мәзірді тышқанның оң пернесін шерткенде пайда пайда болады(егер AutoPopup қасиеті True мағынасы болса) немесе Popup әдісін шақырғанда. Бұл жергілікті мәзірді өндейтін жалғыз оқиға.
Popup (X,У)- процедура , жоғарғы сол бұрышы X және Y координатасына тең меню экранына шығады.
Жобалау сатыларына жергілікті мәзір дәл осылай жасалады.

Мысал 1. Деректер қорымен жұмыс жасайтын қосымшға мәзір құру. «Телефон анықтамасы».
Меню жобалау мақсат үстіңгі db People өрістері IDPeople кестесінде екінші қайтара көрсеткіштері болуы керек, және Family мен Birthday сорттаудың,іздеу жүзеге асыруы керек. ДҚ **«Телефон анықтамасы»** қосымшасында жұмыс істейтін текстік файл құру.
телефондық кітапшаға сәйкес келетін анықтамалық жүйенін мәтінін теріп жазайық: ФИО, дата рождения,телефон номерін жазамыз.

1. 7.0 DataBase Desktop 1. ашамыз және Family мен Birthday, IDPeople өрістерінде қажетті екінші қайтаралар индекстерді құрамыз.Атауын FamilyIndex, BirthdayIndex және Number сәйкесінше қоямыз.
2. Мәзірді құру жұмысын бастау үшін Delphi 3.0 ашамыз.
3. № 2. жқмысында құрылған жоба құрылымын шақырып, аттарын өзгертпестен Қосымшаға арналған үлгі меню, ДҚпен жұмыс істеушіні тандаймыз.

3.1 Енді мәзірді құруды бастау үшін, Delphi ортасына ауысамыз,
3.2 лабараториялық жұмыста құрылған, дайындық проектін шақырамыз.проекттің құрылымдық бөліктерін дереу файлдарға дәл сол атымен сақтаймыз,бірақ 7 санын қосу керек: MyExampleDB7.pas.
4. Менюді орналастыру үшін форма дайындаймыз. Ол үшін келесіні орындау керек:
4.1 Басты менюдің орнын босату арқылы , барлық файлдарды төмен қарай орналастырамыз. (керек болған жағдайда форманың көлемін вертикаль бағамы бойынша өзгерту керек) Қолданушының көңілін аудару үшін ,менюды экранда ерекшелеп орнату қажет,ол үшін оның формасымен түсін өзгерту қажет.
4.2 Қосымшалардың жұмысын аяқтау үшін, Выход кнопкасын жасау керек,DBGridldің оң жағына орналастыру қажет.(Аталмыш кнопка 3,4 лабараториялық жұмыстарда жасалынған)
5. Басты менюдің ортақ түрін қалыптастырайық. Ол үшін келесі функцияларды орындау шарт:
5.1 Standart бетінен палитра компонентін Form1 формасына MainMenu компонентасын орналастырайық.
5.2 MainMenu1 компонентасындағы *Items ұызыметін активтендіреміз. Caption қызыметіне* бос үшбұрышты файл мағынасына орналастырамыз,ал *Name қ*ызыметіне File_pt. Орналастырамыз.
5.3 Шыққан мәзір асты бос ұшбұрышты *Caption* қызыметін баспаға ауыстырамыз, *Name* қызыметін Print_pt ауыстырамыз. Келесі элементте мәзір асты *Caption* «-» таңбасын береміз. *Caption шығу,і Name* – Exit_pt жазамыз.
5.4. Негізгі мәзірдің келесі элементінде Caption қасиетіне Іздеу мағынаны береміз, ал Name қасиетіне - Search - pt . онда ұқсас фамилиямен , датамен тууылар подпунктілерімен іздеуді жазамыз,сәйкесінше баспа және шығу элементтерін жазамыз.
- элементтерге - жасаймыз
5.5. Ұқсас бейнемен элемент сорттау мәзірін жасаймыз пунктілер нөмірмен (Name қасиеті -Number _ pt) және фамилиямен (Name - Family қасиет 2_ pt) сортталатын болады.
5.6. Меню конструкторын жабамыз. не Form1 формасында жасалынушы мәзірдің бейнеленуін белгілеп қоямыз.
Барлық пунктілердің жұмыстары толығымен істеп тұрады.
6. Баспа меню пунктісін таңдау кезінде №3 жұмыста істелінген отчетті алдын ала қарауға болады,онда телефон нөмірлері тіркелінген адамдар тізімі шығады.баспа баспа тәртіпте шығарылады.Бұл пунктілер жұмыс істеу үшін
Келесі әрекеттерді орындау керек:
6.1 MyUnitReport файл 2. pas жобаға қосамыз,№3 Және
Түр модулі мәтінінде pas . MyUnitReport . астында оның сақтаймыз
Form 2(түр есептеу нәтижесімен) қолданылатын жариялау бөлімін
Implementation секциясында модульдерін өзгертеміз.
USES MyExUnitDB3;
Бөлімге
USES MyExUnitDB7;
6.2. Form1 модуль мәтінінде implementation секциясында келесі сөздерді формамен байланысын жазамыз.
USES MyUnitReport ;
6.3. Form1 формасында баспа мәзір пунктін активтейміз және шыққан
OnClick оқиғасында алдын ала қарау тәртібінде есептеу нәтижесінің операторын енгіземіз:
Procedure TForm 1. Print - ptClick (Sender : TObject);
Begin
Form 2. QuickRep 1. Preview ;
End ;
7. Шығу мәзір пункті үшін,келесі әрекеттерді орындаймыз:
тап осы пункті активтейміз және шыққан өңдеушіде
OnClick оқиғасында жұмыстар аяқтауды қамтамасыз ететін операторды енгіземіз:
procedure TForm1.Exit-ptClick (Sender: TObject);
begin
Close;
end;

8. Туған жылдары немесе фамилиясы арқылы іздеу үшін тағы да екі терезе керек,әрбір терезеде іздейтін сұраныс өрістері құрылады. Оларды құру үшін келесі ірекеттерді орындаймыз:
8.1 Жобаға жаңа форма қосамыз. Форманын размерін тігінен кішірейтіп,жолынан үлкейтіп өзгертеміз. *Caption* қасиетін «Поиск по фамилии». формасына өзгертеміз. Форманы Search7.pas. файылында сақтаймыз.
8.2 Standart бетінен Label компонентасын еңгіземіз. *Caption* қасиетін «Введите фамилию искомого человека» өзгертеміз, *Font* қасиеті арқылы шрифті полужирный тандаймыз.
8.3 Label1 компонентін формаға еңгіземіз,және Edit компонентінің размерлерін жазықтығы бойынша үлкейтіп, өзгертеміз.
8.4 Additional бетінен BitBtn кнопкаларын тандаймыз. *Caption* OK, *Kind* ке bkOK, *ModalResult* mrOK қылып *Name*
OKBtnке өзгертеміз. Label1 «Введите дату рождения человека» деген атын, Label «дата вводится через точку» еңгіземіз.
8.6 Form1, Form3, Form4 арасында байланыс орнатамыз Form1 формасында **implementation** секциясында келесі hрістерді еңгіземіз.
USES MyUnitReport, Search7, Search7-1;
9. фамилиялар арқылы мәзір асты іздеу үшін *OnClick* оқиғасына келесі жолдарды еңгіземіз:

```
procedure TForm1.Family1-ptClick (Sender: TObject);
begin
IF Form3.ShowModal = mrCancel THEN Exit;
Form1.Table1.IndexName := ‘FamilyIndex’;
Form1.Table1.SetKey;
Form1.Table1.FieldByName(‘Family’).AsString := Form3.Edit1.Text;
Form1.Table1.GotoNearest;
end;
```

10. «По дате рождения» пунктімен жұмыс істеу үшін *OnClick* оқиғасынын түрі мынандай болу керек:

```
procedure TForm1.Date-ptClick (Sender: TObject);
begin
IF Form4.ShowModal = mrCancel THEN Exit;
Form1.Table1.IndexName := ‘BirthdayIndex’;
Form1.Table1.SetKey;
Form1.Table1.FieldByName(’Birthday’).AsString := Form4.Edit1.Text;
Form1.Table1.GotoNearest;
end;
```

11. Номерлары арқылы сорттау пунктінде *OnClick* ті активтендіру үшін келесі жолдарды жазу керек:

```
procedure TForm1.Number-ptClick (Sender: TObject);
begin
Form1.Table1.IndexName := ‘Number’;
Form1.Table1.SetKey;
Form1.Table1.GotoNearest;
end;
```

12. Фамилиясы бойынша сорттауда *OnClick* оқиғасында келесі кодтты жазу керек:

```
procedure TForm1.Family2-ptClick (Sender: TObject);
begin
Form1.Table1.IndexName := ‘FamilyIndex’;
Form1.Table1.SetKey;
Form1.Table1.FieldByName(‘Family’).AsString := ‘A’;
Form1.Table1.GotoNearest;
end;
```

13. Run|Run командасы арқылы бағдарламаны іске қосамыз.
14. Қосымшаны шығу батырмасын басып,немесе файл\шығу арқылы жабамыз.
15. Жобадағы барлық өзгертулерді сақтаймыз.