O`ZBEKISTON RESPUBLIKASI ALOQA, AXBOROTLASHTIRISH VA TELEKOMMUNIKATSIYA TEXNOLOGIYALARI DAVLAT KO`MITASI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI NUKUS FILIALI

Axborot texnologiyalari kafedrasi

Komp`yuter injiringi fakul`tetining
Korxona servis yo`nalishining
4-kurs talabasi Buranbaeva Saltanatning

BITIRUV MALAKAVIY ISHI

Mavzusi: "Kolledj o`quvchilari bilimini nazorat qilishning ma`lumotlar bazasini yaratish"

Ilmiy rahbar:_____ **«O`zbekiston pochtasi» OAJ Qoraqalpog`iston filiali direktori Masharipov B**
ass. Kuvandikova D.

Kafedra mudiri:_____t.f.n. Arzimbetov T.Z.

NUKUS - 2014 y.

MAZMUNI

KIRISH

Hozirgi vaqtda axborotlarni qayta ishlash va saqlash murakkab masalalar qatorida hisobga olinmaydi. Biroq ma`lumotlarning yo`qolishi yoki o`z vaqtida qabul qilmaslik ortiqcha harajatlarning bekorga sarflanganligini keltirib chiqaradi. Mana shu holat bo`yicha bizning mamlakatimizda va chet ellarda juda tez rivojlanayotgan komp`yuter texnikasini va elektron jadvallar bilan ma`lumotlarni boshqarish tizimining rivojlanayotganligini tushuntirish mumkin.

Har xil korxonalarning tashkilotlarning va har xil xususiy shakldagi firmalarni natijali boshqarish, fuqoralik va askariy maqsatdagi telekommunikatsion qurollar, axborot-hisoblash, ekologik, radiolakatsion tizimlar orqali, yadrosi ma`lumotlar bazasi axborot-hisoblash, (MB) bo`lgan avtomatik boshqarish tizimi orqali tadbiq qilinadi. Murakkab va katta hajmdagi ma`lumotlarda, ularni saqlash. uzatish va qayta ishlash natiyjaliligi alohida turda farqlanadi. Bugingi turmush sharoitida ma`lumotlar bazasining ortishi ular asosida yaratiladigan qo`llanmalarni taqdim etuvchi malakali mutaxassislarga qa`tiy talablar belgilanadi.

Ma`lumotlar bazasi qo`llanmasi bazibir ma`lumotlar manbai bilan bog`lanishi uchun yaratilgan ma`lumotlar bazasi bo`lib hisoblanadi.O`zaro bog`lanish ma`lumotlarni qabul qilish, ularni foydalanuvchi ko`ra oladigan belgilangan shriftda taqdim etish, dasturda biznes-algoritmlar bilan o`rnatilgan tarkibga mos redaktorlash va qayta ishlangan ma`lumotlarni ma`lumotlar bazasiga teskari uzatishi amalga oshiradi. Ma`lumotlar manbai sifatida ma`lumotlar bazasi kabi oddiy fayllarda, tekstli elektron jadval h.t.b.qatnashadi.

Ma`lumotlar bazasi maxsus dasturlar bilan xarakterlanadi. Ma`lumotlar bazasini boshqarish tizimi (MBBT) lokal, bir foydalanuvchiga mo`ljallangan, serverli-tarmoqli (oddiy oraliq), ko`p foydalanuvchiga mo`ljallangan, oraliqda joylaщgan komp`yuter –serverda harakatlanuvchi bo`lib bo`linadi.

Ma`lumotlarning ishki mexanizmlari ma`lumotlar bazasi qo`llanmasi bo`lib, u qo`llanmaga olingan ma`lumotlarning saqlanishini ta`minlaydi va qo`llanmaning boshqa bo`limlarining zaproslari bo`yicha harakatlarni bajaradi. Foydalanish

interfeysi ma`lumotlarning ko`rib o`tilishishi va redaktorlanishini ta`minlaydi, shuning bilan birga, qo`llanmani va ma`lumotlarni boshqarishni amalga oshiradi. Qo`llanmaning biznes-logikasi ma`lumotlarni qayta ishlash algoritmlarining dasturda bajarilishi.

Bu bitiruv malakaviy ish tort paragraf, xulosa, foydalanilgan adabiyotlar qismlaridan iborat.

Kirishda tanlangan mavzu dolzarbligi, komp`yuter texnikasi va elektron jadvallar bilan ma`lumotlarni boshqarish tizimining rivojlanayotganligi haqida tushunchalar aytilgan.Shuningdek, murakkab va katta hajmdagi ma`lumotlarda, ularni saqlash. uzatish va qayta ishlash natiyjaliligi alohida turda farqlanishi qisqacha bayon qilingan.

Birinchi paragrafda – Predmet soha tavsifi, Delphi dasturlash tili haqida tushunchalar keltirilgan.

Ikkinchi paragrafda- Delphida ma`lumollar bazasi bilan ishlash haqida tushunchalar keltirilgan.

Uchinchi paragrafda - Microsoft Access ma`lumotlar bazasi haqida tushunchalar keltirilgan.

To’rtinchi paragrafda- Dasturiy mahsulotni qayta ishlash bosqichlari bayon etildi.

Xulosa - Malakaviy bitiruv ishining asosiy xulosalari va natijalari keltirildi.

§ 1. Ma’lumotlar bazasini yaratishning dasturiy ta’minoti

Delphi dasturlash tili-bu bir qancha ahamiyatli texnologiyalar kombinatsiyasi:

- Mashinalar kodida yuqori unumdorli kompilyator;
- Ob`ektga yo`naltirilgan model komponent;
- Ilovalarni vizual tuzish;
- Ma`lumotlar bazasini tuzish uchun masshtablangan muhit.

Delphi o`z ichiga «klient-server» arxitekturasida ilovalarni tuzish uchun yuqori unumdorlikni ta`minlaydigan kompilyatorni oladi. U ishlab chiqarishning oddiyligini ta`minlaydi va to`rtinchi avlod tillari uchun xos bo`lgan, tayyor dasturiy blokni tezda tekshirishni amalga oshiradi, shuning bilan birga 3GL komplyatorga xarakterli bo`lgan kodning sifatini ta`minlaydi.

Qo`llanmani tuzish jarayonida ishlab chiqaruvchi rasmchi sifatida komponentlar palitrasidan tayyor komponentlarni tanlaydi.

Kompilyatsiyadan oldin u o`z ishining natijasini ko`radi-ma`lumotlar manbaiga ulangandan keyin ularning shakl ko`rinishini ko`rish mumkin. Ular ba`zi bir turlarda beriladi. Bunday qiymat bo`yicha Delphida loyihalash, interpretatsiyalanuvchi muhitda loyihalashdan unchalik farq qilmaydi, lekin kompilyatsiya bajarilgandan keyin interpretator yordamida ishlangan kodga qaraganda 10-20 marta tez bajariladigan kodga ega bo`lamiz. Delphida kompilyatsiya to`g`ridan-to`g`ri o`rnatilgan mashina kodida amalga oshadi, shuning bilan kompilyator, dastur R kodga aylantirib.keyin virtual R mashina bilan interpretatsiyalanadi. Bu tayyor qo`llanmaning haqiqiy tarizda harakatlanishiga ta`sir qilmay qo`ymaydi.

Delphi modelida asosiy kuch kodni maksimal darajada qayta ishlashga qaratiladi. Bular orqali ishlab chiqaruvchilar qo`llanmani tez orada chiqish imkoniyatiga ega bo`ladi, shuning bilan birga ular uchun Delphi muhiti uchun bag`ishlangan shaxsiy ob`ektlarni ham ta`minlaydi. Ishlab chiqaruvchilarning ob`ektlarni tiplar bo`yicha yaratishida hech qanday chegaralar kiritilmaydi. Shuning uchun ishlab chiqaruvchilar ishlab chiqarish muhitini yaratishda

qo`llaniladigan ob`ektlar bilan qurollarga bog`lanadigan mumkinchiliklarga ega. Natijada Borland tomonidan taklif qilingan ob`ektlar bilan yaratish mumkin bo`lgan ob`ektlar firmalari orasida farq bo`lmaydi.

Delphining standart o`rnatmasiga 270 bazaviy sinf ierarxiyasidan tuzilgan asosiy ob`ektlar kiritilgan. Agar Delphida qandayda bir o`ziga xos masalaning echilishi talab etilsa, uchinchi firmalar orqali ishlab chiqarilgan, kommertsiya komponentlarni yoki erkin taratilgan ro`yxatni ko`rib o`tish mumkin, bunday firmalar 250 dan oshib ketdi.

Ko`p holatlarda bu holat quyidagicha tushuntiriladi, ya`ni dasturiy Windows muhitida foydalanuvchi interfeysini o`rnatish juda murakkab edi.

Windows da holatlar bo`yicha tuziluvchi model` tushunish va ajratish uchun qiyin bo`ladi. Biroq interfeysni Delphi ishlab chiqishi dasturchilar uchun eng oddiy masala bo`lib hisoblanadi.

Delphi muhiti ma`lumotlarning korporativ bazalariga ulanishni va foydalanuvchi interfeysining ishlab chiqarilishini ta`minlovchi ilovaning yuqori tezlikda (RAD - rapid application development), ishlab chiqarilishi uchun vizual` instrumentlarning to`liq tarkibini o`z ichiga oladi. VCL- vizual` komponentlar kutubxonasi o`z ichiga foydalanuvchi interfeysini tuzuvchi standart ob`ektlarni, ma`lumotlarni boshqarish ob`ektlarni, fayllarni boshqarish ob`ektlari bilan dialoglarni, DDE va OLE boshqarishni oladi. Delphidagi kamchilik bu Borland tomonidan taklif qilinadigan tayyor komponentlarning ko`p bo`lishi mumkin edi. Biroq boshqa firmalarning ishlab chiqargan komponentlari bunday kamchilikni bartaraf qiladi.

Komponentlarning mos standarti VBX deb ataladi. Bu standart Delphida o`rnatiladi. Biroq Delphi vizual` komponentlar o`zgaruvchanlikka ega.

Delphida vizual` komponetlar, ilovaning algoritmlik qismi yoziladigan, ob`ektli paskalda yoziladi. Shuning uchun Delphida vizual` komponetlari o`rnatilishi va qayta yozilishi uchun aniq qilib belgilanadi. Delphida ma`lumotlar bazasi ob`ektlari SQLga asoslangan va o`z ichiga Borland Database Enginening to`liq quvvatliligini oladi. Delphi tarkibiga Borland SQL Link, ham kiritilgan,

shuning uchun Oracle, Sybase, Informix va InterBase MBBT iga kirish yuqori natijalikda bajariladi.

Shuning bilan birga oflayn rejimda kengaytirilgan SQL–serverlarni ishlab chiqarish uchun Delphiga Interbase lokal` serveri ulanadi. Lokal` mashinalar uchun ma`lumotlar tizimini loyihalovchi, Delphi muhitining qayta ishlovchi mutaxassisi (masalan, bir komp`yuter uchun tibbiy kartochkalarni hisobga olish tizimi) ma`lumotlarning saqlanishi uchun .dbf (dBase yoki Clipperdagi kabi) yoki .db (Paradox) format fayllarini qo`llanish mumkin. Agar u lokal` (InterBase for Windows 4.0) versiyasini qo`llansa (bu ta`minlovchi, SQL-server), u holda ilova hech o`zgarishsiz klient-server arxitekturali yirik tizimlarda ishlashi mumkin. Bir ilovani lokal` va yirik klient-server varianlarda qo`llanish mumkin.

Bugungi kunda Delphining ikki versiyasi chiqarilgan biri-(Delphi Client-Server) “klient-server” arxitekturali ilovalarni ishlab chiqaruvchilar uchun, ikkinchisi-(Delphi for Windows) boshqa dasturchilar uchun bag`ishlangan. Delphi yordamida tuzilgan ilovalarni royalty-protsentlarsiz va runtime- litsenziy to`lamlarsiz qo`llanish mumkin.

U ishchi guruhlar korporativ qo`llanishdagi yuqori unumdorli ilovalarni tuzuvchi korporativ ishlab chiqaruvchilarga bag`ishlangan.

Klient- server versiya o’zida quyidagi o`zgachaliklarni tutadi:

- SQL Links: Oraclega Sybasega, Informixga, InterBasega kirish uchun maxsus drayverlar;

- InterBase Lokal` serveri Windows uchun SQL-server. Lokal` tarmokka ulanmagan komp`yuterda korporativ kushimchalarni kayta ishlash uchun MBBT;

- ReportSmith Client/server Edition SQL- serverlar uchun hisobotlar generatori.

- Team Development Support: Intersolve komponiyasining PVCS yordami orkali kontrol` versiyasini yoki boshka dasturiy maxsulotning kontrol` versiyasini ko`zda tutadi

- Visual Query Builder –bu SQL-surovlar vizual` muxiti.

- Delphi yordamida tayyorlangan klient-server arxitekturasida kushimchalarning to`g`ri taksimlangan litsenziyasi;

- Barcha vizual` komponentlarning izlangan matni.

Delphi for Windows Delphi Client-Server qism to`plamini o`zida tutadi va yuo`ori unumdorli shaxsiy qo`shimchalarni kayta ishlovchilar uchun lokal` MBBTimida dBase i Paradox tipida ishlashni kuzda tutadi. Delphi Desktop Edition klient-server versiya sifatidagi birinchi sinf kompilyator va tez kayta ishlovchi muhitni taklif qiladi. Bu muhit dBase va Paradox tipida MBBTimida ishlovchi personal` ilovalarning tezlik bilan tayyorlanishiga imkoniyat yaratadi. Delphi tayyor dasturlar Paradox, dBase, C++orqali bajariladigan DLL kayta ishlovchilarini tuzadi:

- kompilyator Object Pascal (bu Borland Pascal 7.0 tilining kengaytirilgan versiyasi.);

- ReportSmith 2.5 hisobotlar generatori;

- Ilovalarni tuzishning vizual` muxiti;

- Vizual` komponentlar kutubxonasi;

- InterBase. InterBase lokal` serveri.

Borland komponiyasi tomonidan chiqarilgan maqsulotga, Delphini boshqarishda yordam beruvchi foydali qo`shimchalar tarkibi kiritilgan. Birinchi navbatda Delphi korporat`iv ma`lumotlar tizimini tuzuvchilar uchun mo`ljallangan. Ilovaning tezlik bilan tuzilishini ta`minlovchi ba`zi mahsulotlari oddiy ilovalarni(RAD - rapid application development) yaratishda ham yaxshi natija beradi, biroq haqiqiy murakkab tarkibni ishlab chiqarishda kutilmagan qiyinichiliklarga duch keladi.

Delphi faqat mutaxassis-dasturchilargagina emas mo`ljallangan emas. Dasturiy maqsulotlarni sotib olishni rejalashtiruvchi korxona rahbarlari rejalashtirilayotgan investitsiyaning qoplanishiga ishonchda bo`lishi kerak. Paskal` yo`nalishi bo`yicha ishlovchi dasturchi Delphini kasbiy chevarlikda boshqaradi.

InterBase lokal` serveri – bu ilovaning faqat avtonom otkladkasi uchun mo`ljallangan. Haqiqatda u InterBase serverining SQL-zaproslarining qisqartirilgan variantida berilib, unda InterBase serverining ba`zi bir imkoniyatlari kiritilgan. Bunday imkoniyatlarning o`rnini dasturning avtonom otkladkasi qoplaydi.

Team Development Support – loyihaning guruhli tarkibda ishlab chiqarilishini ta`minlovchi qurol. Yirik loyihalarni boshqarishni sezilarli darajada engillashtiradi. Bu Delphi muhitiga to`g`ridan-to`g`ri ulangan Intersolve PVCS 5.1 kabi mahsulotni ulash mumkinchiligi turida ishlangan. R- kodga o`zgaruvchi mashinali koddagi yuqori unumdorli kompilyator Delphi da dasturli tekst to`g`ridan–to`g`ri mashinali kodda kompilyatsiyalanadi, natijada Delphi-ilova10-20 marta teziroq bajariladi.(ayniqsa matematik funktsiyalarni qo`llanuvchi ilova). Tayyor ilova dastur tillarida yozilgan, ilovalarda boshqa qo`llanish mumkin bo`lgan dinamik kutubxona turida tuziladi.

Delphi yordamida tuzilgan bunday arxitekturali ilova doimiy va natijali bajariladi. Delphi o`zi yaratgan ob`ektlarining va S, S++, OLE serverda yozilgan, DLLni qo`shgan turda harakatdagi ob`ektlarning qo`llanilishini ta`minlaydi. Delphi to`lig`i bilan ob`ektli yo`naltirilgan, shuning uchun ishlab chiqaruvchilar o`z harajatlarini qisqartirish maqsadida qo`llanilgan ob`ektlarni qayta tuzish mumkin.

Delphi guruhli va individual yaratuvchilar uchun ochiq arxitekturani taklif qiladi, ular orqali qurilishda harakatlanishi mumkin. Ishlab chiqaruvchilar Delphi menyusi orqali kirish mumkinchiligiga ega.

Two-way tools- dastur tekstining klassik ko`rinishi bilan vizual` tuzilishi orasidagi moslik. Demak ishlab chiqaruvchi vizual` qurollar yordamida tuzilgan mos kodni kuzatishi mumkin va aksincha.

Interfeyslarni vizual` yaratuvchi (Visual User-interface builder)mos palitralardan komponentlarni tanlay o`t irib, klient-server ilovalarni tez orada vizual` tarizda yaratish mumkin.

Ob`ektlar kutubxonasi foydalanish interfeysini tuzishning standart ob`ektlaridan, ma`lumotlarni boshqarish ob`ektlaridan, grafik ob`ektlardan, mul`timediya ob`ektlaridan, DDE yoki OLE boshqarish, fayllarni boshqarish ob`ektlaridan iborat.

Delphi dasturlash va yuqori quvvatlilikda mosligidagi 4GL tiliga xarkterli, 3GL tilining natijaliligini ta`minlay oladigan, strukturali ob`ektga–yo`naltirilgan tilni (Object Pascal) qo`llanadi. Dasturchilar harakatdagi ilovani Windows o`zgachaliklarini o`rganmasdan turib tuzishlari mumkin. Delphi to`lig`i bilan inkapsulyatsiya, egallash, polimorfizm, holatlarni boshqarish kabi dasturiy kontseptsiyalarni ta`minlaydi.

Bu Windows muhitidagi ishlab chiqaruvchilar uchun ahamiyatli, sababi Windows –ilovalar bilan Delphi muhitining ishlab chiqaruvchilari bilan integratsiyalanish mumkinchiligiga ega.

§2. Delphida ma`lumollar bazasi bilan ishlash.

2.1. ADO texnologiyasining umumiy xarakteristikasi.

Delphida faqat MS Accessda ishlash uchun mo`ljallangan bir qator komponentlar bor bo`lib, ular ADOda joylashgan. Ma`lumotlarning ba`zi bir tarkibini qayta ishlash uchun tizim talablariga mos dastur yozilishi kerak, bunday dastur OLE DB Provider deb ataladi. Bunday ta`minlovchilar ma`lumotlarning har xil tarkibiga va har xil MBBSga mo`ljallangan, OLE DB texnologiyasi yordamida murakkab va spetsifik ma`lumotni bir xil tartibda qayta ishlashi mumkin. Biroq OLE DB da ishlash qiyin bo`lganlikdan, Microsoft firmasi oddiy komponentlar tarkibidan iborat yangi ADO texnologiyasini ishlab chiqdi. Agar aniq MBBT va ma`lumotlarni saqlash usuliga tegishli bo`lmagan ma`lumotlar bilan ishlashga yo`naltirilgan, yangi ilovani yaratish talab etilsa, u holda ADO texnologiyasini qo`llangan ma`qul.

Microsoftning Microsoft ActiveX Data Objects (ADO) texnologiyasi MB qo`llanmasining ma`lumotlar manbaiga universal bog`lanishning o`rnatilishini ta`minlaydi. Bunday mumkinchilik OLE DB spetsifikasida yozilgan va SOM ob`ektlarining umumiy modeli asosida yaratilgan interfeyslar tarkibining funktsiyalari ta`minlaydi. ADO texnologiyasi va OLE DB interfeyslari ilova uchun, har xil tipdagi ma`lumotlar manbaiga kirishni ta`minlovchi usulni o`rnatadi. Masalan, ADO qo`llanadigan ilova, korporativ SQL serverida saqlanadigan, elektron jadvallarga va lokal` MBBT ma`lumotlariga birday bo`lgan murakkab operatsiyalarni qo`llanadi. ADO orqali ma`lumotlarning xohlagan manbaiga yo`naltirilgan, SQL zaprosi, bajariladi.

BD serverlaridan xavfsizlanish mumkin emas. SQL zaproslarining qayta ishlanishi – bu ularning asosiy sharti. OLE DB ob`ektlar orasida ma`lumotlarning uzatilishini ta`minlovchi aniq ma`lumotlar manbai bilan interfeysining maxsus funktsiyalaridan va ma`lumotlarni qayta ishlashning standart funktsiyalaridan, maxsus SOM ob`ektlari tarkibidan iborat. ADO terminologiyasi bo`yicha ma`lumotlarning hohlagan manbai (ma`lumotlar bazasi, elektron jadval, fayl)

ma`lumotlar proyveyderi yordamida ilovaning o`zaro harakatini amalga oshiruvchi ma`lumotlar saqlanuvchi qurol. Ilovaning komponentlar tarkibi birlashish ob`ektidan, ma`lumotlar tarkibi ob`ektidan, zaproslar protsessori ob`ektidan iborat. ADO texnologiyasi butun holda faqat OLE DB ob`ektlaridan emas, ma`lumotlar va ilovalarning ob`ektlar bilan o`zaro bog`lanishini ta`minlovchi mexanizmlardan həm iborat. Bunday darajada har xil tipdagi ma`lumotlarning saqlanuvchi bo`lagidan va ilova harakatini tartibga keltiruvchi ADO provayderlari ahamiyatli o`ringa ega.

ADO provayderleri ADO orqali ma`lumotlarni qo`llanuvchi ilovani ma`lumotlar manbai bilan bog`laydi (SQL serveri bilan lokal` MBBT va fayl tizim) Ma`lumotlarning saqlanishini ta`minlovchi tiplar uchun ADO provayderlari beriladi.

Provayder ma`lumotlar saqlanuvchi o`rinning joylashishini va uning tarkibi haqida xabardor bo`lib, ma`lumotlarga zaproslar bera oladi va xizmat bobidagi ma`lumotlar bilan zapros natijalarini ilovaga uzatish uchun interpretatsiyalanadi. ADO orqali ma`lumotlarga kirish mexanizmi ko`p ob`ektlar bilan interfeyslar, VCL Delphida komponentlar tarkibi turida bajariladi. Komponentlar harakatini ta`minlovchi barcha kerakli interfeyslar OleDB.pas i ADODB.pas.fayllarida beriladi.

Bunday arxitektura ob`ektlar tarkibi bilan interfeyslarni kengaytiradi

ADO texnologiyasi Windows ning tizim mexanizmi bo`lgan SOM standart interfeyslarga asoslanganlikdan, bu orqali harakatdagi kodning umumiy hajmi qisqarib, MB ilovasi qo`shimcha dasturlarsiz ha’m kutubxonalarsiz ha’m tarqatiladi.

ADO komponentlari:

ADO zakladkasida joylashgan komponentlar:

a) Biriktirish komponentlari:

- ADOConnection;
- ADOCommand;

b) Standart komponentlar:

- ADODataSet-ma`lumotlarning universal tarkibi;
- ADOTable –MB jadvali;
- ADOQuery -SQL zaprosi;
- ADOStoredProc- saqlanuvchi protsedura;

ADO sahifasida Delphi, komponentlaridan boshqa ma`lumotlar tarkibini belgilaydigan va ADO ma`lumotlarining saqlanish o`rni bilan ishlanishga mo`ljallangan standart komponentlar ham bor.

ADO Connection komponenti tranzaktsiya xizmati mumkinchiligida sessiya va ma`lumotlar manbaini hisoblash mumkinchiligini beradi. ADO ning tekst komandalari ADOCommand komponentida bajariladi. Qatorlar tarkibini ADOTable, ADOQuery, AOostoredProc komponentlari yordamida olish mumkin.

Ularning har biri xraniliщdagi ma`lumotlarning aniq turiga kirishni ta`minlaydi. Ma`lumotlar saqlanuvchi o`rindan qaytarilib, beriluvchi qatorlar yig`indi Delphi komponentlariga qarata zapislar tarkibi deb ataladi. ADO komponentlarining usullari bilan xossalari tarkibi ilova uchun zarur bo`lgan funktsiyalarning bajarilishini ta`minlaydi. ADO komponentlari ma`lumotga kirishning VCL standart komponentlariga qaraganda unchalik farqi yo`q. Biroq ishlab chiqaruvchi ADO interfeysining mumkinchiliklarini mos ADO ob`ektlariga qarata foydalanish mumkin. Ob`ektlarga ssьlkalar komponentlarda joylashadi.
Ma`lumotlar bazasi bilan ADO ning birlashishini sozlash.

Birlashishdan oldin uning parametrlarini aniqlash zarur. Buning uchun ConnectionString xossasi qo`llaniladi.

Parametrlar tarkibi qo`llanilayotgan provayder turiga bog`liq o`zgaradi va qo`lda yoki redaktor yordamida sozlanadi. Biriktirish redaktorini tanlashda TADOConnection komponentini ikki marta bosamiz. Natijada dialog oynaga ega blamiz.

Bu oynada Use Connection String maydonini qo`llana o`tirib, biriktirishni sozlash mumkin yoki Use Data Link File bo`limidagi fayldan biriktirish parametrlarini yuklash mumkin. Biriktirish parametrlari biriktirish parametrlaridan tuzilgan, oddiy tekstni fayl turidagi UDL fayllarida saqlanadi.

Taklif etilgan provayder orqali biriktirishni sozlash uchun Build tugmasini bosish zarur. Oyna paydo bo`lib, unda kirish mumkin bo`lgan provayder tizimi beriladi.

Provider vkladkasidan aniq ma`lumotlar manbai uchun OLE DB ma`lumotlar provayderini tanlash mumkin. Provayderlar vkladkasida aniq operatsion tizim xizmatiga kirish uchun mo`ljallangan provayderlar ham bo`ladi. Connection vkladkasida ma`lumotlar bazasiga qaratilgan yo`lni yoki serverni ko`rsatish kerak. Advanced vkladkasi Mode xossasiga o`xshash bo`lib, kirish tartibini ko`rsatish uchun mo`ljallangan. AN vkladkasi provayderning nozik spetsifik xossalarini sozlash uchun mo`ljallangan.

Hosil bo`lgan oynada ma`lumotlar bazasiga qaratilgan yo`lni ko`rsatish kerak. Select or enter a database name maydonida dbdemos.mdb demonstratsion bazaga qaratilgan yo`lni ko`rsatish kerak. Ma`lumotlar bazasiga qaratilgan yo`l ko`rsatilgandan keyin, qolgan parametrlar Test Connection knopkasi yordamida tekshirish kerak Agar birikish parametrlari aniq berilsa, Test connection succeeded xabari paydo bo`ladi. Oyna yopilgandan keyin, birikish qatorida malumotlar paydo bo`lib, u orqali provayder ma`lumotlarga dostupni olishi mumkin.

TADOQuery komponenti ma`lumotlar bilan ishlashda ADO orqali SQL-zaproslarning bajarilishini ta`minlaydi. Ma`lumotlarning saqlanish o`rni bilan birga bog`lanish standart usul orqali amalga oshadi.

Zapros parametlari Parameters xossasida joylashadi. Agar kompoent ma`lumotlar tarkibini qaytarsa, uni Open usuli bilan ochish talab etiladi va Active xossasining True miqdori beriladi Agar zapros ma`lumotlar tarkibini qaytarmasa, (INSERT, UPDATE, DELETE i CREATE TABLE operatorlari), unda zapros ExecSQL usulini chaqirish orqali qayta ishlangan zapislarni qaytaradi.

TADOTable kompoenti ADO ma`lumotlari saqlanadigan o`ringa kirishi uchun qo`llaniladi va ulardagi ma`lumotni jadval turida taklif qiladi. TCustomADODataSet sinfining usullari bilan xossalarini egalay o`tirib, har bir zapisga ham uning maydonlariga to`g`ri birikishni taklif etadi. Komponent

ma`lumotlar bazasi bilan Connection yoki ConnectionString xossasi orqali bog`lanadi.

Jadval nomi TableName xossasida ko`rsatiladi. TableDirect xossasi ma`lumotlar tarkibi bilan ma`lumotlar saqlanadigan o`rin qanday tartibda bog`lanishini ko`rsatadi. Barcha provayderlar ma`lumotlar tarkibi bilan to`g`ri bog`lanishda bo`lmaydi, shuning uchun ba`zi holatlarda ma`lumotlar saqlanadigan o`rin bilan bog`lanish uchun SQL-operatorini qo`lanishga to`g`ri keladi. SQL-operatorы Ma`lumotlarga kirish uchun tarkibga True miqdorini o`rnatishda komponent fonlik SQL-zaproslarni qo`llanadi.

Readonly xossasini qo`llana o`tirib, berilgan jadvalga « faqat o`qish sharti» cheklanishini o`rnatish mumkin Shunday tartib orqali ma`lumotni o`zgartirishga cheklovlar kiritiladi. MasterSource tarkibida Master-Detail ssьilkali butinlik munosabatini yaratish uchun qo`llanadigan TDataSource komponenti ko`rsatiladi.

GetlndexNames usuli komponent ro`yxar sifatida tegishli indekslar ro`yxatini qaytaradi.

TDataSource komponenti- bu komponent ma`lumotlar tarkibi bilan bog`lanishadi. Bu bog`lanish ma`lumotlar tarkibi haqidagi ma`lumotdan iborat DataSet xossasi orqali amalga oshiriladi. Bu komponentda ishni olib borishni engillashtiruvchi usullar bilan xossalari tarkibi joylashgan.

Agar kiritish bilan bog`liq element fokusga ega bo`lsa, va AutoEdit xossasi True miqdorida berilsa, bu orqali ma`lumotlar tarkibi redaktorlash holatiga avtomat turda almashadi.

Edit usuli bog`lanishgan ma`lumotlar tarkibini redaktorlash holatiga almashtiradi.

OnDataChange qayta ishlovchi usuli vizual` komponentda bog`lanishgan ma`lumotlarning redaktorlanishida ishlatiladi.

Obrabotchik Post usuli bajarilishidan oldin chaqiriladi. OnStateChange holati qayta ishlovchi usul, bog`lanishgan ma`lumotlar tarkibining holati o`zgarganda chaqiriladi.

Shaxsiy zaproslar bo`yicha olingan ilovalar, zapislar massivi ma`lumotlar to`plami deb ataladi. Ma`lumotlar to`plami ob`ekt sifatida TDataSet sinfi orqali muqaddimaga ega bo`ladi va uning xossasini egallaydi.

2.2. Delphi muhitidagi ma`lumotlarga dastup komponentlari.

Ma`lumotlarga kirishni ta`minlovchi komponentlar o`zgachaligi. Ma`lumotlarga kirishni ta`minlovchi komponentlar vizual` komponentlar qatoriga kiradi. MB jadvallari diskda joylashadi va fizik ob`ektlar bo`lib topiladi. Jadvalda joylashgan ma`lumotlar bilan operatsiyani bajarish uchun, ma`lumotlar to`plami qo`llaniladi. Delphi tizimi terminlarida ta`lumotlar to`plami bu bir yoki bir nechta MB jadvalidan olingan zapislar yig`indisi bo`lib topiladi. Ma`lumotlar to`plamiga kiruvchi zapislar belgilangan tartib bo`yicha saylanadi, bunda ba`zi holatlarda ma`lumotlar to`plamida ma`lumotlar to`plami bilan bog`liq jalvalning barcha zapislari kiritiladi yoki hech qanday zapisga ega bo`lmaydi. Ma`lumotlar to`plami ilovaning bajarilishida ishlash mumkin bo`lgan mantiqiy jadval bo`lib topiladi. Jadval bilan ma`lumotlar to`plamining o`zaro bog`lanishi fizik fayl bilan faylli o`zgaruvchining o`zaro bog`lanishini eslatadi.

Delphida ma`lumotlar to`plami bilan ishlash uchun DBTable i ADOTable, DBQuery, ADOQuery, DataAccess, DataControl, DecisionQuery va StoredProckomponentlari qo`llaniladi

StoredProc komponenti uzoqlashgan MB bilan bog`lanishni tashkillashtirishni saqlanib turilgan protseduralarni chaqirish uchun qo`llaniladi, UpDateSQL komponenti bo`lsa, zapislardagi keshlangan o`zgarishlar bilan ishlashni ta`minlaydi. DecisionQuery komponenti komponenti echimlarni qabul qilish tizimini tuzishda qo`llaniladi. Universal` tarizda qo`llaniladigan ma`lumotlar to`plamini taklif etuvchi komponentlariga Table va Query ham kiradi.

MBga dostupni qator va ustunlar yig`indisi ma`lumotlar to`plamini taklif etuvchi DataSet sinfi ta`minlaydi. Bu sinf ma`lumotlar to`plamini redaktorlashni va navigatsiyaning asosiy qurollarini taklif etadi. DataSet komponenti ADO

ma`lumotlar to`plamidan ma`lumotlarni ko`rsatish uchun mo`ljallangan. U qo`llanishda oddiy, faqat bir nechta shaxsiy xossalar bilan usullarga ega.

ADO ning bu komponenti orqali ADO komandasi boshqariladi va ma`lumotlar tarkibi inkapsulyatsiyalanadi. Natijada komponent moslashtirilgan qurol sifatida jadvallardan ma`lumotlarning olinishish, protseduralar bilan fayllardan SQL,zaproslarni qabul qilishni ta`minlaydi.

Table va Query ma`lumotlari dostupni ta`minlovchi komponentlar

Table i Query komponentlari DataSet sinfining ADODataSet sinfidan kelib chiqqan. Ular bazaviy sinfga o`xshash xarakteristikalarni ko`rsatadi, lekin ularning har biri o`ziga xos o`zgachalikka ega.

ADOTable komponenti OLE DB provayderlari orqali ulangan, Delphida MB jadvallarini qo`llanishni ta`minlaydi. Qo`llanilishi va funktsional mumkinchiligi bo`yicha u standart jadvalli komponentga o`xshash. Komponent asosida ADO komandasini qo`llanish kiritilgan, lekin uning xossalarini oldindan o`rnatib o`zgarish kiritishga zarurlik bo`lmaydi. Komponentlarning boshqa xossalari Bilan usullari indekslarni qo`llanishni ta`minlaydi. ADOning barcha provayderlari MB jadvallarini to`g`ridan- to`g`ri foydalanishni ta`minlay olmaydi. Shuning uchun ularga dostup olish uchun SQL zaproslar talab etiladi. Table komponenti belgili bir vaqt oralig`ida Mbning faqat bir jadvali bilan bog`lanishda bo`ladigan ma`lumotlar tarkibini ma`lumotlarga kirishning navigatsion usuli asosida taqdim etiladi, Shuning uchun Table komponenti dBase i Paradox kabi lokal` MB uchun qo`llanish usuli taklif etiladi. Uzoqlashtirilgan MB bilan ishlashda Query komponenti qo`llangan ma`qul. Jadval va Table komponenti orasidagi bog`lanish jadval atamasini beruvchi, TableName xossasi orqali belgilanadi.

TableName xossalarini belgilashda fayl nomi va fayl nomining kengaytirilgan ko`rinishi belgilanadi. Qo`llanma ishlab chiqarish bosqichida barcha jadvallar nomi ob`ektlar inspektorining ro`yhatida ko`rinadi. Bu ro`yhatga TableName tarkibida ko`rsatilgan katalogda joylashgan jadval fayllari kiritiladi.

ADOQuery komponenti ma`lumotlar bilan ADO orqali ishlashishda SQL zaproslarning qo`llanilishini ta`minlaydli. O`zining funktsionalligi bo`yicha u zaprosning standart komponentiga o`xshash.

Query komponenti SQL-zaprosning bajarilish natijasida shakllanadigan zapislarning ma`lumotlar tarkibini taklif etadi va ma`lumotlarga kirishda relyatsion usulga asoslanadi Uzoqlashgan MB bilan ishlashda Query ma`lumotlar typlamini qo`llanish taklif etiladi. Uzoqlashgan ma`lumotlar to`plami bilan ishlash SQL tilida muhitida olib boriladi. Bu ma`lumotlar tarkibiga almashtirish yoki unga zapislarni o`rnatish kabi operatsiyalar tegishli. Agar Query komponenti uchun Next i Insert tipidagi usullar qo`llanilsa, u holda ma`lumotlarga kirishning relyatsion usuli o`rniga navigatsion usul qo`llaniladi. Natijada Query ma`lumotlar to`plami Table ma`lumotlar to`plami ozi kam farqlanadi. Table komponentidan o`zgacha bo`lgan Query ma`lumotlar to`plami MBning bir jadvalidan umumiy bo`lgan zapislarni o`z ichiga oladi. Ma`lumotlar to`plamining zapislari tanlanadigan zapros teksti Strings turidagi SQLxossasi joylashadi. Bunday zapros SQL tilidagi komandalarni o`z ichiga oladi va ma`lumotlar to`plami ochilishida bajariladi. SQL zapros ba`zida SQLdasturi deb ham ataladi. Ilovaning bajarilishida SQL-zaprosi tekstiga o`zgartirish kiritish mumkin va u dinamik tartibda amalga oshiriladi.

2.3. Ma`lumotlar bilan ishlashni ta`minlovchi komponentlar.

Delphi muhitining vizual komponentlari haqida tushuncha va ularning o`zgachaliklari.

Ma`lumotlar bilan ishlashni ta`minlovchi vizual komponentlar komponentlar politrasining DataControls betida joylashgan va ilovaning interfeys bo`lagini tuzish uchun mo`ljallangan. Ular ma`lumotlar to`plamini navigatsiyalash uchun va zapislarni ko`rsatishda redaktorlashni ko`llanadi. Odatda, bu komponentlar ma`lumotlarga sezgirligi kuchli komponentlar deb ataladi.

Ma`lumotlar bilan ishlashishni ta`minlovchi vizual komponentlar bo`lak zapis maydonindagi operatsiyalarni bajarish uchun mo`ljallangan, ular joriy zapis

maydonining ko`rsatkichlarini belgilaydi va o`zgartiradi. Bunday komponentlar bir qatorli Edit redaktori va Image garfik obzoriga tegishli.

Boshqa komponentlar bir vaqtning o`zida bir nechta zapislarni ko`rsatadi va o`zgartiradi. Bunday komponentlarga ma`lumotlar to`plami zapislarini jadval turida taklif etuvchi DBGrid va DBCtrlGrid setkalari kiradi. Ma`lumotlar bilan ishlashuvchi vizual komponentlar Standard va Additional betlarining mos komponentlariga o`xshash va asosan MB bilan ishlashishga asoslangan bo`lib, DataSource va Datafield qo`shimcha tarkibdan iborat. Birinchisi ma`lumotlar manbaini belgilaydi, ikkinchisi vizual komponent bog`lanishi o`rnatilgan ma'lumotlar to`plami maydoniga qaratilgan. Masalan, Edit qatorli miqdorni ko`rsata o`tirib, foydalanuvchi uchun uni o`zgartirishga imkon beradi.

Maydon ko`rsatkichlarini o`zgartirishga va tasvirlashga mo`ljallangan vizual` komponentlar foydalanuvchi tomonidan o`zgartirilsa, ma`lumotdar to`plami avtomat turda redaktorlash rejimiga o`tadi. Bu komponentlar yordamida amalga oshirilgan o`zgarishlar belgilangan holatlar paydo bo`lganda ular bilan bog`liq maydonlarda avtomat turda saqlanadi.

Vizual komponentlar tarkibi dastur tomonidan o`zgartirilsa, ma`lumotlar to`plami redaktorlash rejimiga avtomat tartibda o`tkazilmaydi. Buning uchun kodda oldindan Edit to`plami chaqiriltirishi kerak. Joriy zapis maydonidagi (maydonlaridagi) o`zgarishlarni sozlash uchun, mos harakatlarni belgilash talab etiladi, masalan, Post usulini chaqirish yoki ma`lumotlar to`plamining boshqa zapisiga siljish.

Vizual komponentlar haqida umumiy xarakteristika.

Vizual` komponentlar kutubxonasida ma`lumotlar bilan ishlashish uchun, mo`ljallangan komponentlar ichida, bazaviy komponent Control. Sinfi bo`lib topiladi. U element o`lchami va holati uning boshlanishi, rangi va parametrlariga o`xshagan asosiy funktsional` atributlarni ta`minlaydi. Control sinfi vizual` komponentlar uchun umumiy bo`lgan xossalarni, usullarni va holatlarni o`z chiga oladi.

Oyna orqali boshqariladigan element aniq masalani echish uchun maxsus oynada taklif qilinadi.

Edit, DBEdit, Memo ili DBMemo kabi komponentlar kiritish fokusini qabul qiladi,o`z sohasida redaktorlash kursorini ko`rsatadi. Malumotning redaktorlanishi bilan bog`liq bo`lmagan komponetlar, kiritish fokusini qora to`g`ri burchakli chiziq yordamida ko`ratadi.

Oynasiz boshqariladigan elementlarning asosiy Control sinfining GraphiControl,sinfi bo`lib topiladi. Oynasiz boshqarish elementlari kiritish fokusini qabul qilmaydi. Ularning o`zgachaligi resurslarni kam sarf qilishida.

2.4 Ma`lumotlar bilan ishlashda vizual komponentlar xossalari tarkibi.

Ilovalarni proektlashtirishda va uning bajarilishida komponentlar xossalari belgilangan o`zgachaliklar asosida boshqariladi. Komponentlarning xossalari loyihalash bosqichida ob`ektlar inspektori yordamida o`rnatiladi.

Name xossasi ilovaning bajarilishida komponentlarni boshqarish uchun qo`llaniladigan, komponent nomini ko`rsatadi. Formaga ornatadigan yangi komponent formaga ornatish tartibi bo`yicha avtomat tarizda qo`shiladi va umolchanie bo`yicha nomga ega bo`ladi. Ilovaning ishlab chiqarish bosqichida mos komponent belgilari bo`yicha komponent nomini o`zgartirish mumkin.

Aling xossasi komponent joylashgan shaklda komponentning tenglashtirish usulini aniqlaydi. Agar qandayda bir interfeysli element belgilangan holatda talab etilsa, tenglashtirish holati qo`llaniladi. Caption xossasi komponentning nomini yozish uchun mo`ljallangan qator. Komponent nomidagi ayrim belgilar chizilib ko`rsatiladi.

Color xossasi fonning rangini aniqlaydi. Odatda, ranglarni doimiylar yordamida o`rnatish qulayliroq. Rangning ko`rsatilishi o`rnatilgan rang ko`rsatkichlari, videokarta va monitorning parametrlariga bog`liq.

Vizual` komponentlar har xil turdagi holatlarni qayta ishlash va generatsiyalash usulini ta`minlaydi. Holatlarning umumiy guruhiga quyidagi harakatlarni belgilash mumkin:

-Boshqarish elementini tanlash.

- Sichqoncha ko`rsatkichini siljitish;

- Klaviatura klavishlarini bosish;

-Kiritio` fokusining boshqarish elemenetini qabul qilish va yo`qotish;

- drag-and-drop. Usuli orqali ob`ektlarni siljitish.

Shuning bilan birga qo`shimcha parametrlarning berilishi talab etilishi, Murakkab holatlar ham bor. Masalan, sichqoncha ko`rsatkichini siljitish bilan bog`liq holat, ko`rsatkich koordinatalarini beradi.

2.5. Ma`lumotga kirish usullari.

Ma`lumotlarga kirishning navigatsion usulining mazmuni.

Navigatsion usul bu ma`lumotlar tarkibining har bir zapisini qayta ishlash. Bu usulning afzalligi operatsiyaning oddiy tartibda kodlashtirilishi, kamchiligi bo`lsa, ilova qancha miqdordagi zapisni qayta ishlashni talab etishiga qaramasdan barcha zapislar tarkibini qabul qiladi. Shuning uchun ma`lumotlarga kirishning navigatsion usuli lokal` ma`lumotlar bazasi bilan cheklanadi. Ma`lumotlarga kirishning navigatsion usulida operatsiyalar bo`lak zapislarda bajariladi. Ma`lumotlarning har bir to`plami harakatlanuvchi zapis` ko`rsatkichiga ega, ya`niy redaktorlash yoki yo`qotish operatsiyalari amalga oshadigan maydonlari kabi yozuvlar Table va Query komponentlari shu ko`rsatkich holatlarini boshqarishni taklif etadi. Ma`lumotlarga kirishning navigatsion usuli quyidagi operatsiyalarni bajarishga mumkinchilik beradi:

- yozuvlarni(zapis`) saralash;

- ma`lumotlar to`plami bo`yicha navigatsiya;

- yozuvlarning redaktorlanishi redaktirovanie zapisey;

- yozuvlarni o`rnatish va uzoqlashtirish;

- zapislarni fil`tratsiyalash;

- yozuvlarni izlash;

Yozuvlarni redaktorlash, o`rnatish va uzoqlatish.

Yozuvlarni redaktorlash ularning maydon qiymatlarii o`zgartirish bilan belgilanadi.

Yozuvlarda faqat joriydagilarigina redaktorlanadi, shuning uchun redaktor Bilan bog`liq harakatdan oldin, talab etilgan yozuvni izlash va siljitish bo`yicha operatsiyalar bajariladi. Joriy yozuv ko`rsatkich kerakli yozuvga belgilangandan keyin, ma`lumotlar to`plamiga ko`rib o`tish rejimida bo`ladi va yozuvlarning redaktorlanishi uchun quyidagilar bajariladi:

- Ma`lumotlar to`plamini redaktorlash holatiga;
- yozuvlar maydonlarining miqdorini;
- O`zgarishlarni tasdiqlash va asoslash.

Insert usuli ma`lumotlar to`plamini o`rnatish rejimiga o`tkazadi va unga Yangi bo`sh yozuvni qo`shadi:

- Ma`lumotlar to`plamini yozuv rejimiga o`tkazish;
- Yangi yozuv maydonining qiymatini berish;
- O`zgarishni tasdiqlash va bekorlash talab etiladi.

Joriy yozuvni Delete usuli orqali o`chirish mumkin va u faqat modifikatsiyalangan ma`lumotlar to`plami Bilan ishlashadi. Yozuv natijali o`chirilgandan keyin yozuv kelasi yozuv bo`lip siljiydi, agar eng so`ngi yozuv o`chirilsa, bunda kursor oldingi yozuvga siljiydi, yozuv o`chirilgandan so`ng,u so`ngi bo`lib o`rnatiladi.

Yozuvlar fil`tratsiyasi

Filtratsiya – bu yozuvlar uchun chegalar topshirig`i.

Fil`tratsiyani foydalanishda yozuvlarni tanlash shartini taklif etuvchi fil`tr talabini qanoatlantiruvchi yozuvlar chegaralanadi.

Fil`tratsiyaning afzalligi shunda, ya`niy u hohlagan maydon uchun qo`llaniladi, shuning ichida indeksatsiyalanmagan maydonlar uchun ham tegishli. Saylash protsessida jadvalning barcha yozuvlari baholanadi, fil`tratsiya mazmuni bo`yicha uncha katta bo`lmagan yozuvlar miqdorida natijali bo`lib hisoblanadi. Shartini belgilash uchun quyidagi tarkibdagi elementlar tuzilishi orqali aniqlanadi:

- jadvallar maydonlarining atamalar;

- literallar;
- taqqoslash operatsiyalari;
- arifmetika operatsiyalar;
- mantiqiy operatsiyalar;
- aylana va kvadrat qavslar.

Agar maydon nomi probellardan iborat bo`lsa,u holda kvadrat qavsda belgilanadi. Literal ko`rsatkich turida taklif etilib u son, qator yoki simvol turida bo`ladi. Agar fil`tr tarkibiga o`zgaruvchi miqdor yoki qandayda bir komponent tarkibini kiritish talab qilinsa, u holda ko`rsatkich qatorli tipga o`zgartirilishi kerak. Taqqoslash opertsiyalari oddiy <,>,=,<=,>=,<> munosabatlar turida taklif qilinadi, arifmetik operatsiyalar +, -,* va /. Mantiqiy operatsiyalar sifatida AND, OR va NOT ko`rsatkichlari qo`llanilishi mumkin. Aylanma qavslar arifmetik va mantiqiy operatsiyalarning bajarilish tartibini o`zgartirish uchun qo`llaniladi.

Table ma`lumotlar to`plami fil`tratsiya shartining ikki usulini qo`llanadi: ular Filter fil`tr tarkibi yordami orqali amalga oshadigan va OnFilterRecord holatlarni qayta ishlash orqali bajariladi. Query ma`lumotlar to`plamida yozuvlar uchun SQL-zaprosni, holatlarning qayta ishlanishini ta`minlovchi OnFilterRecord va fil`tr tarkibini qo`llanishi mumkin.

Diapozon bo`yicha fil`tratsiyada ma`lumotlar to`plami maydonlar miqdori diapozonga tushuvchi yozuvlar orqali kiritiladi, ya`niy fil`tratsiya sharti «qiymat> quyi chegara AND qiymat <yuqori chegara > turidagi qiymat ko`rsatiladi. Bunday fil`tratsiya Table ma`lumotlar to`plamiga qo`llaniladi. Diapozon bo`yicha fil`tratsiyalashning afzalligi shunda ya`ni yozuvlarning qayta ishlashi yuqori tezlikda bajariladi.Jadvalning barcha yozuvlari ketma-ket tartibda kuzatiladigan fil`tratsiyadan farqlanadigan diapozon bo`yicha fil`tratsiyalash usulida indeksli-ketma-ketlik usuli qo`llanladi, shuning uchun fil`tratsiyaning bu usuli faqat indeksatsiyalangan maydonlar uchun qo`llaniladi. Maydon indeksi diapozoni IndexName va IndexFieldName xossasi yordamida taklif etiladi. Agar joriy indeks o`rnatilmagan bo`lsa, unda umolchanie bo`yicha bosh indeks qo`llaniladi.

Diapozon bo`yicha fil`tratsiyani qo`shish yoki ajratish uchun ApplyRange va CancelRange usullari qo`llaniladi.Ulardan birinchisi fil`tratsiyani harakatlantiradi, ikkinchisi to`xtatadi. Fil`tratsiya bajariladigan maydonlar yoki indeksli maydonlar uchun oldindan shartli ko`rsatkichlar diapozoni taklif etiladi. Agar diapozon chegarasining biri berilmasa, unda diapozon ochiq, ya`niy quyi chegara minimal` mumkin miqdorga tenglashadi, yuqori chegara shu maydonning maksimal` mumkin miqdoriga teng bo`ladi.

Yozuvlarni izlash

Belgilangan shartlarni qanoatlantiruvchi , yozuvlarga ishlov berish,shu yozuvga o`tish shartini belgilaydi. Izlash fil`trtsiya shartiga o`xshash, sababi, izlash jarayonida ba`zi shartlar bo`yicha yozuvlar maydonini tekshirish bajariladi. Bundan farq shundan iborat, `niy izlash berish natijasida ma`lumotlar tarkibidagi yozuvlar miqdori o`zgarmaydi.

Yozuvlarni izlashda, izlash olib boriladigan, maydonlardagi indekslar ahamiyatli o`ringa ega. Indekslar ma`lumotlarning qayta ishlanish tezligini sezirarli darajada orttiradi, bundan boshqa bir qator usullar faqat indekslangan maydonlardagina ishlashi mumkin.

Maydonlar bo`yicha yozuvlarni izlash uchun Locate va Lookup usullari qo`llaniladi, sababi maydonlar indekslanmagan bo`lishi mumkin. Locate funktsiyasi maydonda berilgan ko`rsatkichlar yozuvini izlaydi. Agarda izlash shartlarini qanoatlantiruvchi yozuvlar bor bo`lsa, u holda joriy yozuv ko`rsatkichi ularning eng birinchisiga o`rnatiladi. Agar yozuv topilsa, funktsiya True ko`rsatkichini beradi, aks holda False ko`rsatkichini. Odatda ilovani ishlab chiqarishda foydalanuvchiga shaklda joylashgan, boshkarish elementlari yordamida izlash jarayoniga ta`sir etish mumkinchiligi taklif etiladi. Bunda ma`lumotlar to`plami izlashni boshqarish bo`yicha foydalanuvchi harakati, fil`tratsiyaning urinlanishi harakatidan kam farq qiladi. Lookup funktsiyasida belgilangan shartlarni qanoatlantiruvchi yozuvlarni izlashni amalga oshiradi, lekin Locate usulidan farqi shunda, joriy yozuv ko`rsatkichini almashtirmasdan ma`lumotlarni yozuvlar maydonidan hisoblaydi. Ikki usul orasidagi yana bir farq,

Locate usuli harflar registrini hisobga olib, yozuvlar maydonidagi ko`rsatkichkichlarni izlash aniq ko`rsatkichkichlarga mos bajariladi.

Table ma`lumotlar to`plami uchun faqat indeksli maydonlar bo`yicha yozuvlarga izlash berishni ta`minlovchi usullar mo`ljallangan. Bu usullarning biri chaqirilmasdan oldin izlash maydoni bo`yicha tuzilgan indeksni o`rnatish zarur.Izlash usullarini ikki guruhga bo`lish mumkin.Usullarning birinchi guruhida izlash anik moslik belgilari bo`yicha bajariladi, ikkinchisi usul izlash ko`rsatkichkichi bilan yozuvlar maydoni ko`rsatkichkichlarining mosligi asosida bajariladi. Agar berilgan izlash shartiga bir nechta yozuvlar mos kelsa,unda joriy yozuv ko`rsatkichkichi ularning eng birinchisiga o`rnatiladi.

Ma`lumotlar to`lamida saralash,tartibi va yozuvlarning joylashish tartibi, izlash olib boriladigan ,joriy indeks orqali aniqlanadi.

2.6. SQL yozuvlar tili.

SQL yozuvlarning strukturalashgan tilini belgilaydi. Bu til orqali jadvallarda saqlashuvchi, o`zaro bog`langan ma`lumotlar to`plami bo`lib topiladigan ma`lumotlarning relyatsion bazasida ishlash mumkinchiligini ta`minlaydi. Axborot fazosining belgili darajada unifikatsiyalanishi, har xil komp`yuter muhitida qo`llanilishi mumkin bo`lgan, standar tilning yaratilish zarurligiga olib keladi. Standart til orqali foydalanuvchi komandalarning bir tarkibi orqali ma`lumotlarni yaratishni aniqlash o`zgart`irish va uzatish mumkinchiliklariga ega. O`zaro bog`langan komp`yuter olamida bunday til bilan ta`minlangan foydalanuvchi ko`plagan usullar yordamida bir nechta manbalar orqali ma`lumotlarni jamlashtirishda va qo`llanishda katta afzalliklarga ega bo`ladi. Komp`yuter texnologiyalaridan mustaqila va o`zgacha bo`lgan shuning bilan birga relyatsion ma`lumotlar bazasi texnologiyasi xossalarining qo`llab quvvatlanishi SQL tilini asosiy standart til qatoriga olib keladi.

SQL tili tarkibi.

SQL tili relyatsion ma`lumotlar bazasidagi ma`lumotlarni egallash, ma`lumotlar bazasi strukturasini aniqlash va ko`plagan foydalanuvchilar

muhitidagi ma`lumotlarga kirish huquqini boshqarish uchun mo`ljallangan. Shuning uchun, SQL tilida tarkibiy qism sifatida quyidagilar kiritiladi:

- Ma`lumotlarni egallash tili (Data Manipulation Language, DML)
- Ma`lumotlarni aniqlash tili (Data Definition Language, DDL)
- ma`lumotlarni boshqarish tili (Data Control Language, DCL).

Bular bo`lak til emas, bir tilning har komandalari. Bunday qilib bo`lib ko`rsatish komandalarning funktsional` belgilari bo`yicha belgilangan. Ma`lumotlarni egallash tili ma`lumotlarni egallash uchun qo`llaniladi.U 4 asosiy komandalardan iborat:

SELECT (tanlash)

INSERT (o`rnatish)

UPDATE (yangilash)

DELETE(o`chirish)

Ma`lumotlarni aniqlash tili ma`lumotlar bazasi strukturasini uning tarkibiy bo`laklarini-jadvallarni, indekslarni, tasvirlarni( virtual` jadvallar), triggerlarni, saqlangan elementlarni tuzish va o`zgartitrish uchun qo`llaniladi.

CREATE DATABASE (ma`lumotlar bazasini CREATE TABLE (jadval yaratish)

CREATE VIEW (virtkal jadval yaratish)

CREATE INDEX (indeks yaratish)

CREATE TRIGGER (trigger yaratish)

CREATE PROCEDURE (saqlangan protsedurani yaratish)

ALTER DATABASE (ma`lumotlar bazasini yangilash)

ALTER TABLE (jalvalni yangilash)

ALTER VIEW (virtkal jadvalni yangilash)

ALTER INDEX (indeksni yangilash)

ALTER TRIGGER (triggerni yangilash)

ALTER PROCEDURE (saqlangan protsedurani yangilash)

DROP DATABASE (ma`lumotlar bazasini yo`qotish)

DROP TABLE (jadvalni yo`qotish)

DROP VIEW (virtkal jadvalni yo`qotish)

DROP INDEX (indeksni yo`qotish)

DROP TRIGGER (trigger yo`qotish)

DROP PROCEDURE (saqlangan protsedurani yo`qotish)

Ma`lumotlarni boshqarish ma`lumotlarga kirishni boshqarish va ko`p foydalanuvchilik muhitida amallarning bajarilishi uchun qo`llaniladi. U asosiy ikki komandadan iborat:

GRANT (vazifalarni taklif etish)

REVOKE (vazifalarni olish)

Amaliy interfeys ko`z-qarashi bo`yicha SQL komandalari ikki turga bo`linadi:

- Interaktiv SQL

-o`rnatilgan SQL.

Interaktiv SQL interaktiv rejimida SQL komandalarini qo`llana o`tirib, zaproslarni kiritadi, ularning bajarilishi uchun serverga uzatishni oyna bo`yicha natijalarning olinishini ta`minlovchi maxsus utilitalarda (WISQL yoki DBD turidagi)qo`llaniladi.

O`rnatilgan SQL amaliy dasturlarda zaproslarni serverga jo`natishni va olingan natijalarni qayta ishlashni ta`minlaydi, shuning bilan birga set-yo`nalishidagi va record-yo`nalishidagi usullarni kombinatsiyalaydi.

Ma`lumotlarni egallash tilining ahamiyatli komandasidan biri SELECT komandasi. Tanlab olish operatsiyasi bir jadvalning barcha qatorlarini yoki qatorlarning ba`zi bir qismini olishni taklif etadi. Logik operatorlarga har xil logik harakatlarning bajarilishini taklif etuvchi AND, OR, NOT operatorlari kiradi: ular

logik ko`paytirish (AND, “shartlarning kesishishi),

logik qo`shish, (OR “shartlarning birikishi”),

logik bekorlash(NOT,“shartlarning bekorlanishi”).

Bizning misolimizda AND operatorini qo`llangan edik. Bu operatorlarni qo`llanish orqali, yozuvlarni tanlash sharti qulayli turda o`rnatiladi.

Umumiy predikat haqiqiy bo`ladi, agar AND bo`yichap bog`liq shartlar haqiqiy bo`lsa va bu holat “AND” operatorlari orqali belgilanadi

Umumiy predikat haqiqiy bo`ladi, agar OR bo`yicha bog`langan shartlarning biri haqiqiy bo`lsa, ham bu OR operatori menen belgilanadi. Ymumiy predikat haqiqiy boladi, agar shu operatordan oldin joylashgan shart yolg`on bo`lsa va ham bu holat NOT operatori bilan belgilanadi. Bir predikatda logik operatorlar, quyidagicha tartibda bajariladi: dastlab NOT operatori bajariladi, keyin AND va bundan keyin OR operatori.

Chiqarariluvchi qatorlar tartibini o`zgartirish (ORDER BY)

Chiqariluvchi qatorlar tartibi SQL-zaprosining oxirida optsional (qo`shimcha) ORDER BY taklifi yordamida o`zgartilishi mumkin. Bu taklif ko`rinishi quyidagicha vid:

ORDER BY <qatorlar tartibi> [ASC | DESC]

Qatorlar tartibi quyidagi ikki usuldan biri orqali berilishi mumkin:

- ustunlar nomi bilan;

- ustunlar nomerlari bilan.

Tartiblashtirish usuli qo`shimcha rezervlangan ASC va DESC so`zlari orqali aniqlanadi. Agar buyruq berilmasa-tartiblashtirish “o`sish bo`yicha” beriladi. Agar “DESC” so`zi ko`rsatilsa, u holda tartiblashtirish“kamayish bo`yicha” bajariladi

§3. Microsoft Access ma`lumotlar bazasi

Talabalarning darsga qatnashishi va o`zlashtirishi bo`yicha tuzilgan elektron jadval MS ACSESS ma`lumotlar bazasi yordamida tuzilgan edi.

Microsoft Access ilovasi relyatsion ma`lumotlar bazasini boshqarishning yuqori quvvatlilikdagi va yuqori unumdorli 32-razryadli tizimi bo`lib hisoblanadi.

Ma`lumotlar bazasi – bu malumotlarni ko`rsatishni va saylashni qo`shishni ta`minlovchi, o`zaro bog`langan va tarkiblashgan ma`lumotlar va usullar yig`indisi.

Relyatsion ma`lumotlar bazasi. Tajribaning ko`rsatishicha barcha MBBT jalvalga yangi malumotlarning kiritilishini ta`minlaydi. Bu ko`z qarash boyicha MBBT, ma`lumotlar bazasining ba`zi bir funktsiyalarini emul`yatsiyalovchi elektron jadval (Excel) dasturidan ham farqi yo`q.

MBBT bilan elektron jadvallar dasturlari orasida uch printsipial` bog`lanish bor.

1) MBBT, elektron jadvallar orqali bajariladigan ma`lumotlardan ham katta, ma`lumotlar hajmining natijali qayta ishlanishini ta`minlash maqsadida tuziladi;

2) MBBT, foydalanuvchi uchun bir jadval bo`lib, taklif etiladigan ikki jadvalni oson birlashtirish mumkin. Bunday mumkinchilikni elektron jadvallarda bajarish mumkin emas.

3)MBBT ma`lumotlar bazasining umumiy hajmini minimallashtiradi. Buning uchun qaytalanib keluvchi ma`lumotlarga ega jadvallar o`zaro bog`lanishgan bir nechta jadvallarga bo`linadi.

Access – Windowsning kuchli ilovasi. Bunda MBBT unumdorligi Windowsning afzalliklari bilan qulayliklariga oson moslashadi. Relyatsion MBBT kabi Access ma`lumotlarning barcha tiplariga kirishni ta`minlaydi va bir vaqtning o`zida ma`lumotlar bazasining bir nechta jalvalini qo`llanishga imkoniyat beradi. Shuningdek Paradox va dBase o`rtasida yaratilgan jadvallarni ham qo`llanish mumkin.

Access ma`lumotlar bazasi fayllari tarmaq bo`linuvchi resurslari bo`la oladigan ko`plagan foydalanuvchi ilovalarni yaratish uchun maxsus tarizda tuzilgan.

Accessda fayllarga kirishni nazorat qiluvchi ishonchli mudofaa tizimi o`rnatilgan.

Ma`lumotlar bazasi bir faylda saqlanadi, biroq mutaxassis qo`llanuvilar ma`lumotlar bazasini ikki faylga bo`lishni ma`qul ko`radi, birisida ma`lumotlar ob`ektlari saqlanadi(jadvallar, zaproslar), ikkinchisida qo`shimcha ob`ektlar (formalar, hisobotlar, makroslar) modullar).

Accessning eng keyingi versiyalarida VBA-kodi ulamasdan, ilovani yaratish mumkin bo`lgan,kutubxona-faylining yangi formati(MDE) kiritilgan.

Asosiy funktsiyalari:

1) Ma`lumotlarni tuzish. Jadvallarni yaratish va ularni boshqarish;

2)Jadvallarni bog`lash va ma`lumotlarga kirishni ta`minlash. Access bir nechta jadvallarni bir jadvalga biriktirish maqsalida maydon ko`rsatkichlari mos keluvchi jadvallarni bog`laydi;

3) Ma`lumotlarni qo`shish va o`zgartirish;

4) Ma`lumotlarni ko`rsatish;

5) Makroslar. Makroslarni foydalanish takrorlanib keluvchi operatsiyalar orqaoi avtomatlashtirish imkoniyatlariga ega bo`ladi.spol`zovanie makrosov pozvolyaet avtomatizirovat` povtoryayuщiesya operatsii. Accessning so`ngi versiyalarida sakroslar biriktirish uchun qo`llaniladi.V poslednix versiyax Access makrosы ispol`zuyut dlya sovmestimosti;

6) Modullar. Modullar Accessning VBA (dialekt Visual Basic Application) yozilgan protseduralarni yoki funktsiyalarni ko`rsatadi. i predstavlyayut soboy protseduru ili funktsiyu, napisannыe na Access VBA (dialekt Visual Basic Application). Bu protseduralarni murakkab hrsoblashlar uchun qo`llanish mumkin;

7) Ma`lumotlar bazasini himoya qilish. Bu qurollar ilova ishini ko`p foydalanuvchi muhitida tashkillashtirish imkoniyatini beradi;

8) Bosmaga chiqarish quroli. Bu funktsiya yordamida Access ma`lumotlar bazasidagi ma`lumotlarni bosmaga chiqarish mumkinchiligini beradi;

Access yangi ilovani tarqatish uchun (Office Developer Edition Tools yordamda) distributiv tuzish imkoniyatini beradi. Tarqatsh qandayda bir taratilishida barcha kerakli fayllarning o`rnatilishini nazarda oladi. ODE Tools paketi tarqatish qurolini yaratish va o`rnatish dasturini avtrmatlashtiruvchi, o`rnatish masterini ulaydi. U Access o`rnatilmagan komp`yuterlarda ilovaning bajarilishini ta`minlaydi. Ma`lumotlar bazalarining elementlari:

1) Jadvallar. Ma`lumotlar bazasida axborotlar ikki o`lchamli jadval turida saqlanadi.

Shuning bilan birga jadvallarni boshqa MBBTidan yoki elektron jadvallarni boshqarish tizimlaridan olish yoki bog`lash mumkin. Bir vaqtning o`zida 1024 jadval ochilishi mumkin;

2) Zaproslar. Zaproslar yordamida har xil jadvallardan qandayda bir kriteriya bo`yicha ma`lumotlarni tanlash bajarilishi mumkin. Zaprosga 255maydongacha ulash mukin.

3) Formalar. Formalar jadvaldagi va zaproslardagi ma`lumotlarni qulayli turda qo`llanilishini ko`rsatadi. Formalar yordamida jadval tarkibidagi ma`lumotlarni o`zgartirish yoki qo`shib borish mumkin;

4) Hisobotlar. Hisobotlar jadvallar va zaproslarda joylashgan ma`lumotlarni bezatilgan ko`rinishda bosmaga chiqarish uchun mo`ljallangan. Shuning bilan birga, hisobotlar modullarni ulashi mumkin. 5) Makroslar

6) Modullar. Modullar holatlarni qayta ishlay olishini ko`rsatish uchun mo`ljallangan VBA-kodidan iborat, masalan, sozlash funktsiyalarini yaratish uchun ma`lumotlar bazasi ob`ektlari ustilan operatsiyalarning avtomat bajarilishi uchun, va operatsiyalarni dasturiy boshqarish uchun formadagi yoki hisobotdagi knopkalarni bosish, ya`niy, VBA-kodini qo`shish orqali menyu, instrumntlar paneli va boshqa mumkinchiliklar o`rnatilgan ma`lumotlar bazasiga yaratish mumkin.

Access tarkibiga ma`lumotlar bazasining ob`ektlarini yaratsh protsessini osonlashtiruvchi, o`rnatuvchilar, masterlar. Tuzuvchilar ko`pligi kiritiladi. vxodit V sostav Access vxodit mnojestvo masterov, postroiteley i nadstroek, kotorыe pozvolyayut uprostit` protsess sozdaniya ob`ektov bazы dannых.

Jadvallar

1) Jadval masteri rejimida jadvallarni yaratish.

Master yordamida ma`lumotlar bazasi ob`ektini yaratish protsessi bir nechta qadamlarga bo`linadi va ularning har birida yaratilayotgan ob`ektning talab etiladigan belgilarini o`rnatish mumkin. Jadvallar masteri ish yuzasida qo`llaniladigan jadvallarning 33 namunasi va shaxsiy foydalanish jadvallarining 20 namunasi asosida yangi jadvallarni yaratishni ta`minlaydi. MA`lumotlar bazasining ko`pchilik jadvallari jadvallar masterining natijasi asosida yaratilgan. Mnogie tablitsы bazы dannых sozdanы na osnove obraztsov Mastera tablits.

Jadval masterini qo`llanish orqali, Accessdagi masterlar bilan ishlashish amali oson tushiniladi.

Jadvallar masteri natijalari asosida jadvallarni yaratish, bunday ilovalarda cheklangan mumkinchilikka ega. Ko`pchilik holatlarda jadvallarni tuzish uchun import sharti qo`llaniladi yoki ma`lumotlarni boshqa ma`lumotlar bazasi bilan yoki elektron jadvallar bilan bog`lash

2) Jadvallarni konstrukturi rejimida jadvalalrni yaratish.

Agar ma`lumotlarni import etish yoki bog`lash imkoniyati bo`lmasa, jadvallar jvadvallarning talab etuvchi strukturasini aniqlovchi konstruktor rejimida yaratiladi.Jadvallar konstruktori rejimida ma`lumotlar tipini, atamasini, belgilarini va jadval maydonlarining qo`shimcha xossalarini ko`rish mumkin. Konstruktor rejimida ochiladigan, jadval oynasining yuqori tomonida jadval strukturasining blankasi yoki jadval blankasi deb ataladi.

3) Tablitsa rejimida jadvallarni to`g`ridan to`g`ri yaratish.

Access jadval rejimida jadvallarni to`g`ridan–to`g`ri yaratish mumkinchiligini beradi. Bunda 20 maydonga va 30 to`liqtirilmagan yozuvlar strukturasi orqali, umalchanie bo`yicha foydalaniladigan jadvallar asosida

to`liqtirilmagan jadvallarni tuzadi. Keyin jadvalga ma`lumotlarni to`g`ridan-to`g`ri kiritish mumkin. Uning saqlanishida Access ma`lumotlarni analizlaydi va har bir maydon uchun kiritilgan ma`lumotlarga mos keluvchi turni tanlaydi. Jadval rejimida jadvallarni tuzishda cheklangan shartlar belgilangan. Bunday jadvallar maydoni ma`noli nomga ega bo`lmaganlikdan, maydon atamasini o`zgartirish uchun, strukturani o`zgartirib turish kerak.

Bundan boshqa Access hamma vaqt ma`lumotlar tipini to`g`ri belgilay olmaydi. Jadval rejimida tuzilgan jadvallar OLE ob`ektlarini va menyu-maydonlarni qo`sha olmaydi. Agar bunday maydonni hohlasangiz, siz jadval strukturasini o`zgartirishingiz kerak. Maydonlarni aniqlash va to`liqtirilmagan jadval strukturasida ularning xossalarini o`rnatishning dasturiy usuliga toqqoslaganda jadval rejimida jadal tuzishda vaqtning tejalishiga erishish mumkinchiligi yo`q. Jadval strukturasini ishlab chiqarishda unda qanday ma`lumot o`rnatilishini ko`rsatish zarur.

Jadvalga kiritiluvchi ma`lumot aniqlangandan keyin, uni maydonlar bo`yicha bo`lish kerak. Bu protsess jadvalda maxsus o`ringa ega maydon nomini tanlashni o`z ichiga oladi. Maydon nomiga uning tarkibi haqidagi ma`lumotni kiritish zarur. Shuning bilan birga nom qisqa turda berilishi kerak.

Odatda, ma`lumotlar bazasini ishlab chiqarishda har xil jadvallarni bunday atamadagi maydonlar kiritilishi mumkin ( ko`pincha jadvallar shu maydonlar bo`yicha boqlangan).

Access har xil jadvallarda maydonlarning birday atamasini qo`llanishi mumkin, biroq Access maydanlar atamasini ma`lumotlar shartin aniqlash orqali qo`llanadi, shuning uchun maydonlar atamasini takrorrlashga yo`l qo`yilmasligi kerak.

Jadvalni to`ltirmasdan oldin, jadval ma`lumotlar bazasining boshqa jadvallari bilan, bog`lanishi belgilanadi, asosiy (kalit) maydon aniqlanadi, indekslar tuziladi.

Jadvallar orasida bog`lanish asosiy maydonlardagi mos keluvchi ko`rsatkichlar orasidagi munosabatni o`rnatadi- odatda,har xil jadval maydonlari

orasida ko`plagan holatlarda bir jadvalning asosiy maydonini ikkinchi jadvaldagi asosiy tashqi maydon deb ataluvchi mos maydon ()bilan bog`lash mumkin

Asosiy maydon ichidagi ichidagi maydon bosh jadval deb ataladi,u tashqi kalitdan iborat bo`lagi bog`lanishgan maydon deb yuritiladi. Bog`lanishlarning to`rt turi bor:

- Birga bir «Birga bir» munosabatini qo`llanishda "A" jadvaldagi yozuv ( bosh jadval) "V" (bog`lanishgan) jadvaldagi yozuv bilan bog`lanishi birdan ortiq bo`lmaydi. Bunday bog`lanish turi ko`plagan holatlarda qo`llanilmaydi, sababi bunday ma`lumotlar bir jadvalga joylashtirilishi mumkin. «Birga bir» munosabatdagi bog`lanish kengaytirilgan jalvallarni bo`laklash uchun qo`llaniladi yoki himoya tarkibi bo`yicha jadval bo`limlarini ajratadi. «Birga bir» munosabat bog`lanishidagi jadvallarda asosiy maydonlar o`zgacha bo`lishi kerak

- Birdan ko`plikka «Birdan ko`plikka» munosabat bog`lanishi jadvallar orasidagi bog`lanishning keng qo`llaniladigan turi.

- Bunday bog`lanishda «A» jadvaldagi har bir yozuvga "V"jadvaldagi bir nechta yozuvlar mos kelishi mumkin, "V" jadvaldagi yozuvning birisini egallashi mumkin.

- Ko`plikdan-ko`plikka «Ko`plikdan-ko`plikka» munosabatini qo`llanishda «A» jadvaldagi bir yozuvga «V» jadvavldagi bir yozuv mos kelishi mukin, «V» jadvaldagi yozuv «A» jadvaldagi birdan ortiq mos yozuvga ega bo`lishi mumkin. «A» jadvaldagi asosiy maydon o`zgacha bo`lishi mumkin. «A» həm «V» jadvallarnining « birlikdan ko`plikka» munosabat bog`lanishini, «V» həm «A»jadvallarning «Ko`plikdan ko`plikka» munosabat bog`dlanishi sifatida qarash kerak.

- Ko`plikdan-ko`plikka. «Ko`plikdan-ko`plikka» munosabat bog`lanishida A jadvaldagi bir yozuvga V jadvaldagi bir nechta yozukvlar mos kelishi mumkin, V jadvaldagi bir yozuvga A jadvalining bir nechta yozuvlari mos keltiriladi.

Bunday holatlarda A həm V jadvalasosiy maydonlariga o`zgachalikni belgilash talab etilmaydi. Bunday bog`lanish tashqi kalit tarkibiga ega jadvallarni biriktirish uchun qo`llaniladi.

Birikkan maydonlar bunday atamaga ega bo`lishi shart emas,.biroq ular ma`lumotlarning birday tipiga ega bo`lishi kerak. Bundan boshqa sonli tipidagi birikuvchi ma`lumotlar «maydon o`lchami » xossasining birday ko`rsatkichlariga ega bo`lishi kerak.Krome Bu qoidaga tegishli bo`lmagan holat ketma-ketlik numeratsiyadagi hisoblash maydoni bo`lib hisoblanadi, bunda keng butinlik o`lchamdagi sonli maydonlar bilan bog`lanish mumkin. Shunday har xil uzunlikdagi tekstda maydonlar orasidagi bog`lanish tuzish

Ma`lumotlarni butin turda ko`rsatishni avtomat ta`minlash Accessning ahamiyatli o`zgachaligi bo`lib hisoblanadi. Agar jadvallar orasidagi bog`lanishda butin turda ko`rsatish sharti o`rnatilgan bo`lsa,u holda Access bog`lanishli yozuvlar jadvaliga mos emas yozuvlarning kiritilishiga yo`l qo`ymaydi.va bosh jadvaldagi yozuvlar o`zgarishi shunday bo`ladi, ya`niy bundan keyin bog`lanishgan jadvalda bosh yozuvlarga ega bo`lmagan yozuvlar paydo bo`ladi; shunig bilan birga bog`lanishgan jadvaldagi yozuvlarning bosh jadvaldagi yozuvlari yo`q etiladi.

Ma`lumotlarning butinlik sharti bog`lanishgan jadvallardagi yozuvlar orasidagi bog`lanishning o`rnatilishi uchun Accessda qo`llaniladigan shartlar tizimini aniqlaydi. Bunday shartlar bog`lanishgan ma`lumotlarning o`zgartilishiga va kutilmagan holatda o`chib ketishiga yo`l qo`ymaydi.

Ma`lumotlarning butinlik sharti o`rnatilgandan keyin bog`lanishli jadval operatsiyalariga cheklovlar engiziladi. Bog`lanishgan jadvalning asosiy tashqi maydoniga, bosh jadvalning asosiy maydonda bo`lmagan ko`rsatkichni kiritish mumkin emas.

Yana bir ahamiyatli holat bu dastlabki shartni aniqlash. Agar jadval hech qachon asosiy sifatida qo`llanimasa,uning uchun shartni belgilash kerak emas. Bosh jadvallarda odatda haqiqiy ob`ektlar haqidagi ma`lumotlar kiritiladi, bunda har ob`ektga faqat bir yozuv yo`naltiriladi. Jadvalning shartini aniqlash jadvaldagi birday yozuvlarning paydo bo`lishining oldini olish usuli. Bog`lanishning bosh jadvalida dastlabki shart belgilanishi kerak. Shartlar belgilanmagan jadvallarning Accessning konstruktor rejimida ochilishida jadval sharti belgilanmaganligi

haqidagi xabar, dialog oynada paydo bo`ladi. Sharti bog`lanishgan jadvallarda ham belgilash mumkin, bunday holat takrorlanuvchi ma`lumotlarning takrorlanishining oldini oladi.

Jadval shartini bir nechta maydon ko`rsatkichlari bo`yicha ham olish mumkin. Access shart ko`rsatkichi bo`yicha jadvalni avtomat turda indekslaydi, biroq boshqa maydon ko`rsatkichlari bo`yicha qo`shimcha indekslarni tuzish talap etilishi mumkin. Jadvalning indekslangan maydonlarida ma`lumotlarni izlash tezlik bilan bajariladi.

Accessning har bir jadvali 32 indeksgacha ega bo`lishi mumkin, ulardan 5 tasi murakkablashgan bo`ladi.

Shuningdek,jadvalning har bir maydoni uchun yoki ularning barcha kombinatsiyalariga indekslarni yaratish shart emas,sababi, bu holat jadvalni to`ltirish protsessini sezilarli darajada pasaytiradi.

Relyatsion ma`lumotlar bazasini ishlab chiqarishning asosiy printsiplaridan biri, bu jadval maydonida joylashgan, barcha ma`lumotlar birday tipda bo`lishi kerak. Jadvalning har bir maydoni uchun ma`lumotlar tipini berish zarur, Umal`chanie bo`yicha «Tekstovыy» ma`lumotlar tipii qo`llaniladi. Maydonning ma`lumotlar tipini belgilashda maydon ko`rsatkichlariningk o`rinishiga tasir etuvchi o`lchamni, formatni, boshqa parametrlarni va sanli ma`lumotlarni aniq ko`rsatish shart.

Ma`lumotlarning asomiy tiplari:

1) Tekstli. Hisoblashlarning bajarilishini talab qilmaydigan teks yoki sonlar;

2) MeMO. Bunday tip maydoni unchalik katta kata bo`lmagan ma`lumotlarni saqlash uchun mo`ljallangan. (64000 simvolovgacha) Bunday tip maydoni shartli yoki indekslangan bo`lishi mumkin emas.

3) Sonli. Ma`lumotlarning bu tipii tiplar ko`pligidan iborat. Tip ko`pligini tanlash orqali hisoblashning aniq bajarilii taminlanadi;

4) Hisoblagich. Ketma-ket ortip boruvchi, ajratilgan sonlar;

5) Mantiqiy. Mantiqiy miqdorlar yoki ikki mumkin ko`rsatkichning biridaniborat maydonlar;

6) Ko`rsatkichli (Pulli). Matematik hisoblashlarda qo`llanilayotgan , pulli ko`rsatktchlar yoki sonli miqdorlar;

7) Sana/vaqt.Sana va vaqt maxsus fiksatsiyalangan formatda saqlanadi;

8) OLE ob`ektlari maydoni.Tovushli yozuvni rasmni yokima`lumotlarning boshqa tiplarini ulaydi.Bu tipdagi maydon shartli yoki inlekslangan bo`lishi mumkin emas.;

9) Giperbog`lanish. Web –betlar adresidan iborat.

Zaproslar. ma`lumotlar. Ular jadvalga yangi yozuvlarni qo`shish,ayirish,yangilash uchun qo`llaniladi.Odatda yozuvlar belgilangan kriteriyalarni qanoatlantirish uchun, yozuvlarning maxsus guruhlarini bo`lish uchun qo`llaniladi. Shuningdek,ma`lumotlarning bog`lanishgan elementlarining butin bir ko`rinishi orqali ularni har xiljadvallardan ma`lumotlarni olish uchun qo`llanish mumkin.

Accessda har xil maqsat uchun mo`ljallangn yozuvlvrning 4 tipi bar:

Tanlash uchun, mo`ljallangan yozuvlar bir yoki bir nechta jadvallardagi ma`lumotlarni jadvallar turida ko`rsatadi. Tanlangan zaproslar elektron jadval formatiga elektron jadval formatiga o`xshash, formatda ma`lumotlarni bir yoki bir nechta jadvallardan to`playdi.

Bu zaproslar ba`zi bir yozuvlar ko`pligidagi sonli ko`rsatktchlarining summar miqdorlariga asoalangan diagrammalarni tuzish va ma`lumotarning analizi uchun qo`llaniladi.

O`zgartirish uchun zaproslar, zaposlar natijalaridan yangi jadvallarni tuzi shva berilgan jadval ma`lumotlariga o`zgarish kiritish uchun qo`llaniladi

Ular yordamida jadvaldagi yozuvlarni qo`shish yoki fyirish,shuning bilan birga zapros kontruktori rejimida berilgan, qiymatlarga mos,yozuvlarni o`zgartirish mumkin.

Parametrli zaproslar –foydalanuchi tomonidan o`zgartiriladigan zaproslar. Parametrli zaproslarlar qo`llanilganda, tanlash sharti kiritilishi kerak bo`lgan, dialog oyna paydo bo`ladi. Zaprosning bu tipi o`zgachalangan emas, ya`niy parametrlarni ixtiyoriy tipdagi zaprosga qo`shish mumkin.

3.1 Ma`lumotlar bazasini proektlash bosqishlari.

Proektlash bosqishlari va ma`lumotlar bazasini tuzish quyidagicha ketma-ketlikda belgilanadi:

- predmetli soha ma`lumotlarining axborot- mantiqiy modelini tuzish;
- relyatsion ma`lumotlar bazasining mantiqiy tarkibini ariqlash.;
- ma`lumotlar bazasining jadvalini tuzish;
- ma`lumotlarni jadvalga kiritish;
- Zarur bo`lgan formalarni, zaproslarni,makrslarni,modullarni. Hisobotlarni ishlab chiqish;
- Foydalanish interfeyslarini ishlab chiqish.

Ma`lumotlar modelini ishlab chiqish protsessida ma`lumotlar normalizatsiyasi talablariga mos, axborot ob`ektlarini belgilab, ular orasidagi bog`lanishlar aniqlanadi. Bu model`, dastlabki yuklashda va korrektirovkalashda ma`lumotlarning faqat bir marta kiritilishini va o`zgarishlarni kiritishni ma`lumotlarning butinligini ta`minlovchi ,relyatsion ma`lumotlar bazasini dublersiz (bir ishni ikkijoyda baravar bvjvrish)yaratadi.

Ma`lumtlar modelin ishlab chiqishda ikki usul qo`llaniladi.Birinchi usulda dastlab echilishi uchun baza yaratilayotgan asosiy masala belgilanadi,masalaning ma`lumotlarga bo`lgan talabi aniqlanib, unga mos holda axborot ob`ektlarning tarkibi va tuzilishi belgilanadi. Ikkinchi usulda predmet sohada tiplik ob`ektlari belgilanadi. Ikkala usulning ratsional mosligi asosan, ushbu holatga bog`liq turda, ya`niy dastlabki bosqishda qoida bo`yicha barcha masala haqida noto`g`ri ma`lumotlar bo`lmaydi. Bunday topologiyani qo`llanish sinovdan natijali o`tgan, sababi, ma`lumotlarning relyatsion bazasini yaratuvchi moslashuvchan qurollar, ishlab chiqarishning ixtiyoriy bosqishida ma`lumotlar bazasi o`zgarish kiritishga imkoniyat beradi va uning tarkibini. Odindan kiritilgan a`lumtga zarasiz tartipbda modifikatsiyalashni amalga oshiradi.Pri Normalizatsiya talabiga javob beruvchi, predmet sohaning axborot ob`ektlarini bo`lish protsessii induktiv va formal usul asosida amalga oshadi. Formal usulning nazariy usullari belgili amerikan olimi Dj.

Martinning ma`lumotlar bazasini tashkillashtirish bo`yicha ishlangan ilmiy monografiyalarida berilgan.

Intuitiv (ichki his bilan sezib) usulda haqiqiy ob`ektlarga mos keluvchi axborot ob`ektlar qulayli tartibda bo`linib ko`rsatiladi. Biroq, u orqali, olinadigan axborot- mantiqiy model` tartib bo`yicha o`zgarishlarning bajarilishini talab etadi.shuning ichida ob`ektlar orasidagi ko`p belgili bog`lanishlarning o`zgartilishini. Agar etarli tajriba bo`lmasa,bunday usulda, xatolar sezirarli darajada,paydo bo`ladi. Normalizatsiya talabining bajarilishini tekshirishning kelasi bosqichida axborot ob`ektlarini aiqlashtirish zarurligini ko`rsatadi. Axborot ob`ektlarni qo`llaniladigan, formal modellar Axborot ob`ektlarida qo`llaniladigan formal щartlarni ko`rib o`tamiz:

- Predmet soha asosida ma`lumotlar bazasida saqlanishga tegishli hujjatlarni va atributlarni belgilash;

- Atributlar olrasidagi funktsional bog`lanishni belgilash.

- Tegishli barcha atributlarni tanlash va har biri uchun asosiy atributlarni ko`rsatish, ya`niy;

- Bir xil atributlarni asosiy atributilardan bo`laklab guruhlash. G`arazli atributlardan olingan guruhlar o`zlarining asosiy atributlari Bilan, birgalikda axborot ob`ektlarini paydo etadi.

Model asosida ma`lumotlarning relyatsion bazasining mantiqiy strukturasini belgilaydi., har bir axborot hajmi relyatsion jadvalda adekvat tarzda beriladi,jadvallar orasidagibog`lanish, axborot ob`ektlar orasidagi bog`lanishga moslashadi. Yaratish protsessida ma`lumotlar modeli tomonidan tuzilgan,axborot ob`ektlariga mos, ma`lumotlar bazasining jadvallari quriladi.Keyin jadvallar orasidagi mantiqiy bog`lanish fiksatsiyasi bajariladigan ma`lumotlar sxemasi tuziladi. Bu bog`lanishlar axborot ob`ekti bog`lanishiga mos keladi. Agar ma`lumotlar modeli normalizatsiya talablariga mos ishlab chiqarilgan bo`lsa, ma`lumotlar sxemasida, ma`lumotlar bazasining butinligini ta`minlaydigan parametrlar berilishi mumkin.

Ma`lumotlarning butunligi , MB o`rnatilgan har xil jadvallarning yuklanishida, qo`shilishida va yozuvlarning uzoqlashishida va asosiy maydon miqdorlarining o`zgarishida yozuvlar orasidagi bog`lanishning to`g`ri o`rnatilganligini anglatadi.

Ma`lumotlarning sxemasi tuzilgandan keyin, predmet soha hujjatlaridan kerakli ma`lumotlarni kiritilishi amalga oshadi.

Tuzilgan ma`lumotlar bazasi asosida bazaning ma`lumotlarini va ularning ko`rinishlarini qayta ishlashda talab etiladigan kerakli zaproslar, formalar, makroslar,modullar,hisobotlar, shakllanadi.

Ma`lumotlar bazasining o`rnatilgan qurollari va instrumentlari yordamida ma`lumotlar bazasi ma`lumotlarni kiritish, saqlash, qayta ishlash, yangilash protsesslarini boshqarishni ta`minlaydigan foydalanish interfeysi tuziladi.

§4. DASTURIY MAHSULOT ISHLAB CHIQARISH BOSQICHLARI

Delphida ilova yordamida MBdan foydalanish uchun

Ilova=> BDE=> MB

ketma-ketligi bajariladi. Bu shuni bildiradiki har qanday MBga ilovadan murojaat qilinganda uning aniq adresi BDEga uzatiladi. BDE o`zining maxsus funktsiyalarini MB bilan bog`lanishda ishlatadi. BDE aniq bir MB bilan ishlaganda quyidagilarni bilishi kerak.

- MB qaysi joyda joylashganligini;
- MB parametrlari haqidagi ma`lumotni.

MB parametrlari va uning joylashishi MB psevdonimida aniqlanadi. Psevdonim - MB ga berilgan biror nom bo`lib MBga mantiqiy murojaat qilinganda ishlatiladi.

MB psevdonimi BDE adminisrator utilitasi yordamida aniqlanadi.

MBning har bir jadvali uchun fayl tuziladi. Xuddi shunday jadval indekslari va memo maydonlari uchun maxsus fayllar tuziladi. MBga dastur va utilitalardan murojaat qilinish MBning psevdonimidan amalga oshiriladi. Psevdonim BDE administrator utilitasi yordamida aniqlanadi. Psevdonim - biror bir nom bo`lib MBga Delphi komponenti ilovalari (masalan Ttable va Tquery) tamonidan mantiqiy murojaat qilishda ishlatiladi.

Aytaylik biz tashkil qilishimiz kerak bo`lgan MB “C:\PROBA\” katologida joylashsin. Biz tashkil qiladigan psevdonim nomi aytaylik “Proba” bo`lsin. BDE administrator utilitasini ishga tushiramiz. Asosiy menyudan Object/New buruqlarini tanlaymiz. Hosil bo`lgan darchadan tuziladigan MB turini aniqlash uchun “Standart” parametrini o`zgartirmasdan Ok tugmasini bosamiz. Chap oynada (MB administratori oynasida) biz “Standart1” nomini ko`ramiz va uni “Proba”ga o`zgartiramiz. ¤ng oynada MB parametrlari berilgan, u erda faqat Path (yo`l) o`zgartiriliadi. Bu parametr MB katalogiga yo`lni ko`rsatadi. Path nomini ko`rsatib, o`ng darchadan uch nuqtali tugmani va keyin chap oynadan “Proba”ni tanlab Ok tugmasini bosamiz. MB uchun aniqlangan psevdonimni saqlash uchun chap oynaga kelib, sichqoncha o`ng tugmasini chiqillatib dialog oynasidan

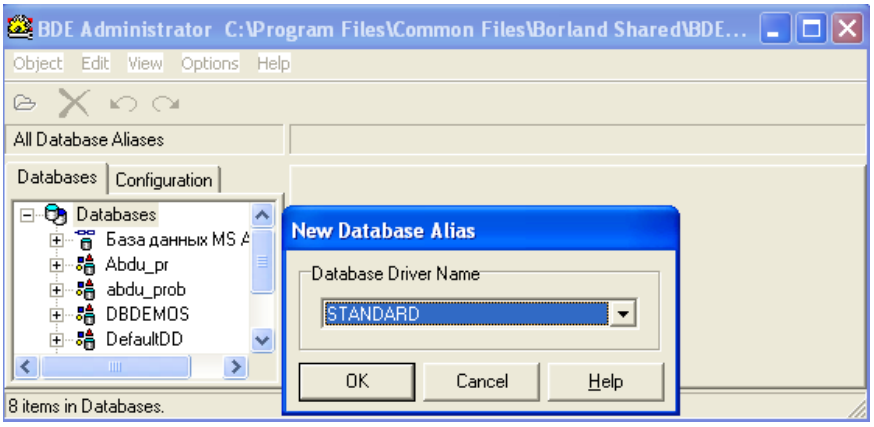
“Apply” elementini tanlaymiz va Ok tugmasini bosamiz. Keyin esa BDE administratoridan chiqamiz.

MBni psevdonimini tuzishni yanada chiqurroq ko`rib chiqaylik. Uning algoritmi quyidacha:

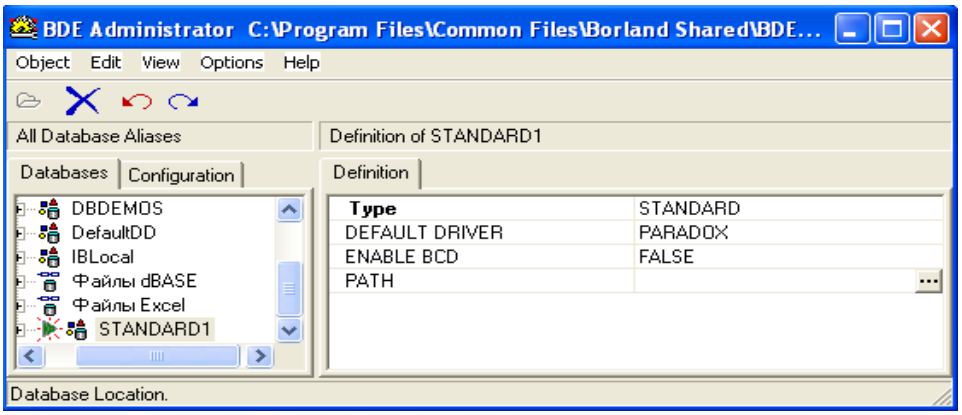
1.BDE administrator utilitasi ishga tushiriladi.

2.Menyudan Object=>New buyrug`i beriladi.

3.Darchadan Standart turi o`zgartirilmasdan Ok tugmasi bosiladi.

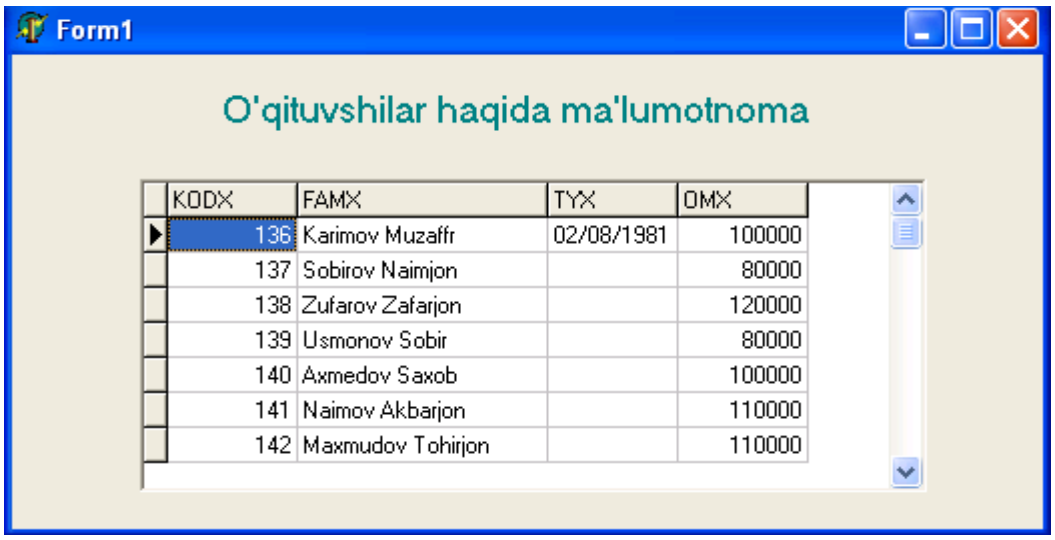


4. Chap oynadan Standart1 nomi yangi psevdonim nomiga o`zgartiriladi.
5. O`ng oynada Path qatoriga o`tilib, uch nuqtali tugmacha va keyin Ok bosiladi.
6. Menyudan Object=>Apply=>Ok buyrug`i beriladi.
7. BDE administratordan chiqiladi.



MB jadvali bilan ishlash uchun oddiy ilova yaratish algortmi quyidagi ketma-ketlikda bajariladi:

1. Delphi tizimi ishga tushirilib BDE komponentalar palitrasidan Ttable komponentasi formaga qo`yiladi.
2. Formadagi Ttable komponentasi belgilanib, DataBase Name xossasida Mbning psevdonimi aniqlanadi.
3. TableName xossasidan MB jadvali nomi aniqlanadi.
4. Active xossasi True qiymat bilan o`rnatiladi.
5. Data Acciss komponentalar palitrasidan TdataSource komponentasi formaga qo`yiladi.
6. TdataSet xosasi Table1 nom bilan o`rnatiladi.
7. Data Controls komponentalar palitrasidan TDbGrid komponentasi formaga qo`yiladi.
8. DataSourse xossasi DataSourse1 nom bilan o`rnatiladi.
9. Menyudan File=>Save Project As buyrug`i berilib, oldin forma keyin loyiha saqlanadi.
10. Loyihani ishga tushirish uchun F9 tugmasi bosiladi. Natijada quyidagi formaga ega bo`lamiz.



TDBNavigator komponenti. MB jadvalida ma`lumotlarni surish, o`chirish, yozuvni siljitish va taxrirlash uchun Data Controls komponentalar palitrasida maxsus TDBNavigator komponentasi mavjud.

Bu komponentani formadagi MB jadvaliga quyidagi tartibda o`rnatish mumkin.

1. MB jadvali formasi ekranga chaqiriladi.
2. Data Controls komponentalar palitrasidadan TDBNavigator komponentasi formaga joylashtiriladi.
3. TDBNavigator komponentalar xossasidan DataSourse xossasi DataSourse1 nom bilan o`rnatiladi.
4. Menyudan File=>Save Project As buyrug`i berilib, oldin forma keyin loyiha saqlanadi.
5. Loyihani ishga tushirish uchun F9 tugmasi bosiladi.

Natijada jadvali ma`lumotlarni surish, o`chirish, yozuvni siljitish va taxrirlash kabi tugmachalarga ega bo`lgan quyidagi formaga ega bo`lasiz.

Komp`yuter quyidagi dastur kodlarini avtomatik ravishda tuzadi

Ikkita jadvalda ishlash uchun ilova tuzish

Bitta formada ikkita bog`liq ma`lumotlar to`plami uchun ilova tuzishni ko`rib chiqaylik. Forma ilovasini “App21.pas” fayliga loyixa ilovasini “App2.drp” fayliga yozib qo`yaylik.

Ilovaga Ttable komponentasini qo`shamiz (Table2 nom bilan). Proba Mbning Prihod jadvali bilan ishlash uchun uchun uning xossalarini qo`yamiz (xossalari qiymati Table1 komponentasi kabi bo`lib, Table Name ga jadval nomi Prihod.dbf beriladi). Table2 Avtive xosasini “True” qiymatda o`rnatamiz. Formaga TdataSurce komponentasini joylashtiramiz (DataSource2 nom bilan). Bu komponenta uchun DataSet xosasini Table2 qiymat bilan o`rnatamiz. TDBGrid komponentasini formaga joylashtiramiz (DBGrid2 nom bilan) va DataSource xossasini o`rnatamiz (DataSource1 qiymat bilan). Ilovani ishga tushiramiz.

4.1.Ma`lumotlarni izlash va fil`trlash

Kompornent TDataset va uning davomchilari ma`lumotlar bilan ish yuritishda maxsus usullarga ega:

- maydon qiymati bo`yicha ma`lumotni izlash;
- ma`lumotlarni fil`trlash;
- zakladka qo`yish va unga o`tish.

**Ma`lumotni izlash.** Ma`lum belgilangan yozuvlarni ma`lumotlar to`plamdan izlab topish uchun ikkita usul mavjud: Locate va Lookup.

Locate - usuli biror maydonning berilgan yozuvi bo`yicha kerakli yozuvni topish imkonini beradi. Uning umumiy ko`rinishi quyidagicha:

Function Locate(Const KeyFields: String; Const KeyValues:
Variant; Options: TLocateOptions): Boolean;

Bu erda

KeyFields -ma`lumotlarni izlashda qatnashadigan maydonlar nomlari. Ular bir biridan nuqta vergul bilan ajratiladi.

KeyValues -bir yoki bir necha izlanadigan maydon qiymatlari. Agar izlanadigan qiymatlar bir necha bo`lsa massiv varianti funktsiyasi qilib berish zarur.

Options -izlanadigan parametrlar to`plami. U quyidagi qiymatlarni saqlash mumkin:

loCaseInsensitive. Registrni hisobga olmasdan izlash.

loPartialKey. Maydon qiymati to`liq berilmagan holda izlash. Masalan, ‘So’ boshlanadigan familiyalar izlanadigan bo`lsa. U holda ma`lumotlar to`plamidan So bilan boshlanadigan familiyalar ‘Sobirov’ va ‘Soatov’ lar topiladi.

Agar izlanayotgan yozuv topilsa funktsiya Locate -true qiymat qaytaradi.

Misol. 2. Misol tariqasida yuqorida tuzilgan o`qituvchilar MBni olaylik va MBdan kerakli o`qituvchini izlab topish uchun ilova yarataylik.

Ilovani yaratish algoritmi:

1.Delphini ishga tushuramiz.

2.Formaga Label1 komponentasini o`rnatamiz va uning Caption xossasini “Malumotlar bazasidan izlsh” so`ziga almashtiramiz.

3.Formaga DataSource (Ma`lumotlar manbai), Query (So`rov) va DBGrid komponentalarini joylashtiramiz. Ularning qayidagi xossalarini o`rnatamiz.

DataSource1 komponenti.

Xossa	Qiymati
DataSet	Query1

Query1 komponenti.

Xossa	Qiymati
DataBaseName RequestLive SQL	ABDU_PR True Select * From Xodims

DataSource1 komponenti.

Xossa	Qiymati
DataSourse	DataSourse1

4.Yuqoridagilarni to`g`riligini tekshirish uchun Query1 komponentasining Active xossasini True qilib o`rnatamiz.

5.Data Controls komponentalar palitrasidadan TDBNavigator komponentasini formaga joylashtiramiz.

6.TDBNavigator komponentalar xossasidan DataSourse xossasini DataSourse1 nom bilan o`rnatamiz.

7.Button1 komponentasini formaga joylashtiramiz va uning Caption xossasini “Chiqish” so`ziga almashtiramiz.

8.Edit1 komponentasini formaga joylashtiramiz.

9.Button2 komponentasini formaga joylashtiramiz va uning Caption xossasini “Kod boyicha izlash” so`ziga almashtiramiz.

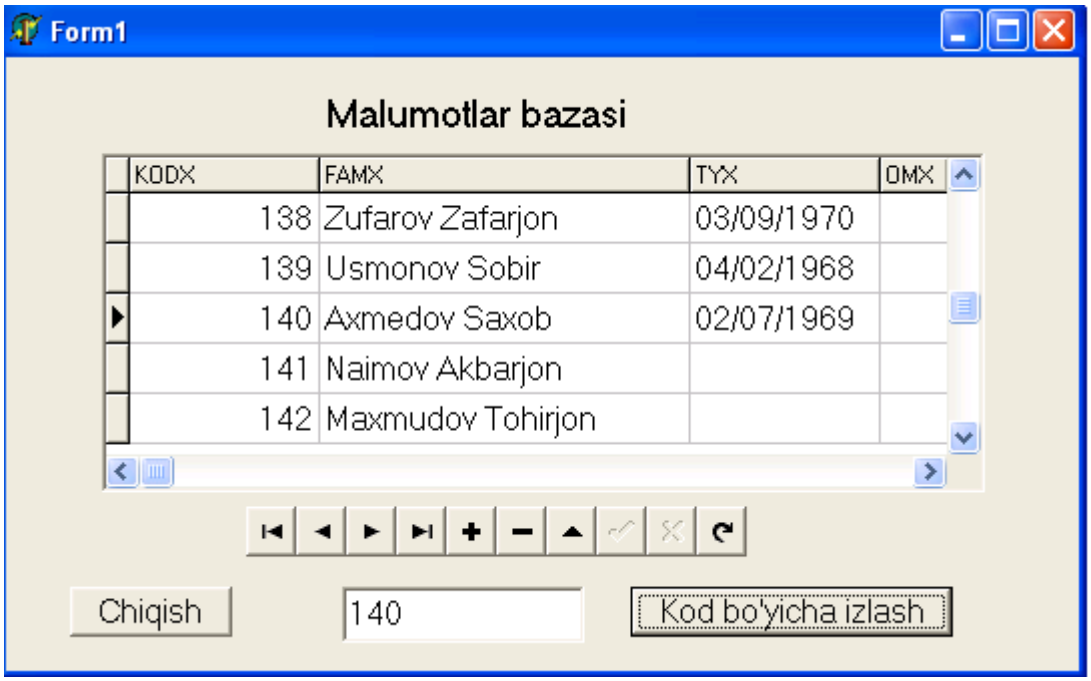
10.“Kod boyicha izlsh” tugmasini ikki marta tez-tez chiqillatib quyidagi dastur kodini kiritamiz.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
        Query1.Locate('Kodx',Edit1.Text,[LopartialKey,
         LoCaseInsensitive]);
end;
```

Bu erda 'Kodx' –xodimlar kodini tasvirlovchi maydon nomi

11. Ilovani ishga tushiramiz.

Natiijada quyidagi forma ilovasiga ega bo`lamiz.



Function LookUp(Const KeyFields: String; Const KeyValues: Variant; const ResultFields: String): Variant;

Bu funktsiyaning Locate funktsiyasidan farqi shundaki u topilgan yozuvni joriy dab aniqlamaydi, u topilgan yozuvning berilgan maydon qiymatini qaytaradi. Xuddi oldingi holdagi kabi uning parametrlari quyidagilarni aniqlaydi.

KeyFields -ma`lumotlarni izlashda qatnashadigan maydonlar nomlar ro`yxarti. Ular bir biridan nuqta vergul bilan ajratiladi.

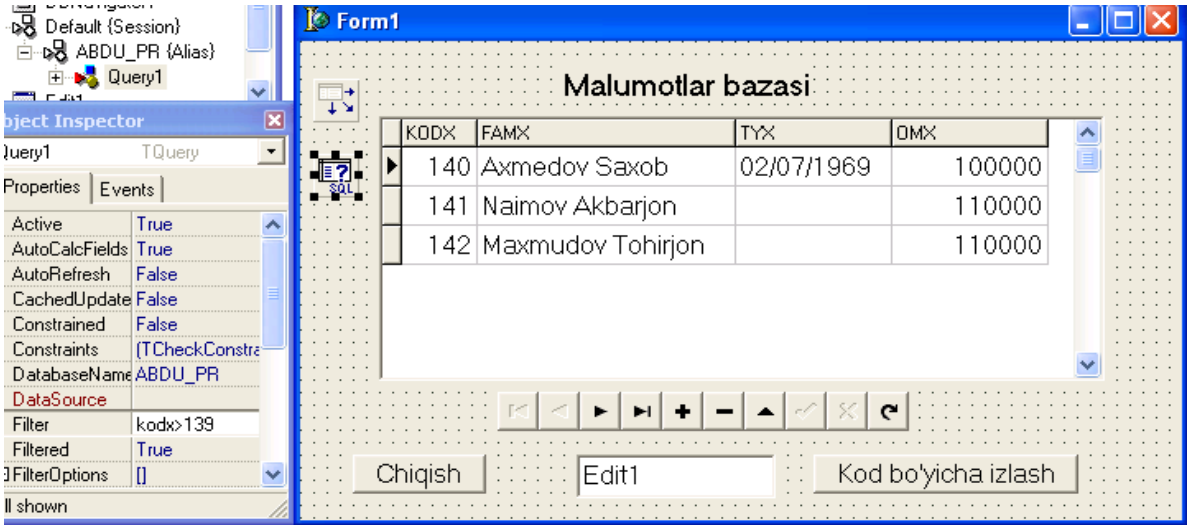
KeyValues -bir yoki bir necha izlanadigan maydon qiymatlari. Agar izlanadigan qiymatlar bir necha bo`lsa massiv varianti funktsiyasi qilib berish zarur.

4.2.Ma`lumotlarni fil`trlash.

Ma`lumotlarni fil`trlashning ikkita asosiy usuli mavjud:

- fil`tr xossasini ishlatish;
- OnFilterRecord obrabotchigini (qayta ishlovchisini) tashkil qilish.

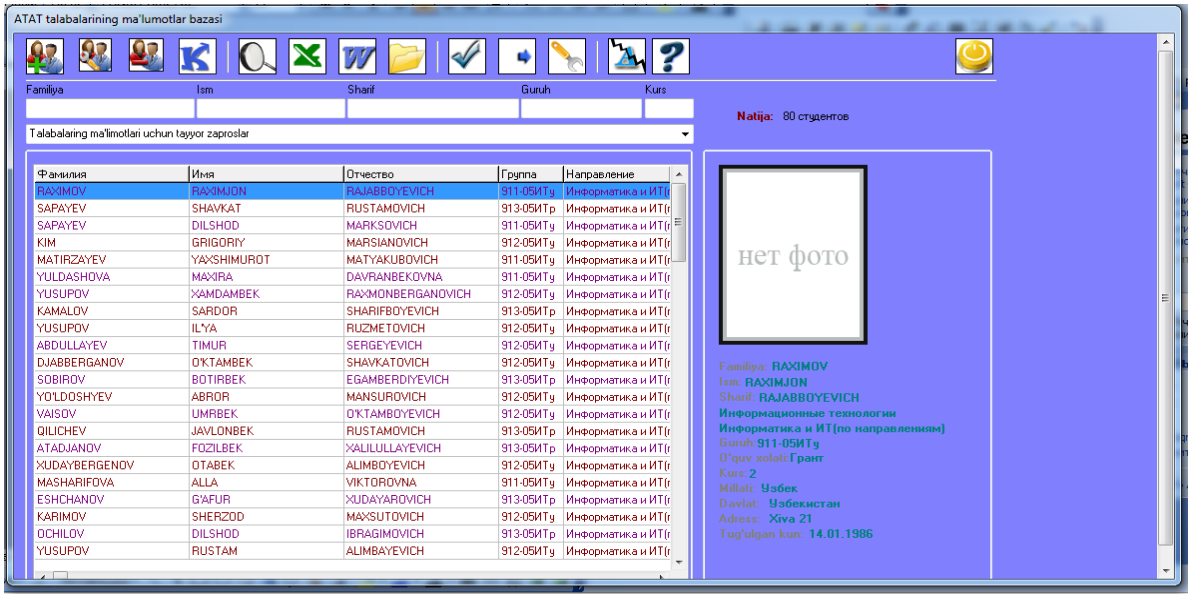
Filter xossasi MB yozuvini qanoatlantiruvchi shartni o`z ichiga oladi. Shart solishtirish va mantiqiy operatorlarni o`z ichiga olishi mumkin. Masalan, Kodx>139 And Omx>100000. Bu shart bizning yuqorida yaratgan MB uchun kodi 139dan katta va maoshi 100000dan katta xodimlarni ekranga chiqaradi. £uyidagi oynada kodi 139 dan katta xodimlarni ekranga chiqarish tasvirlangan.



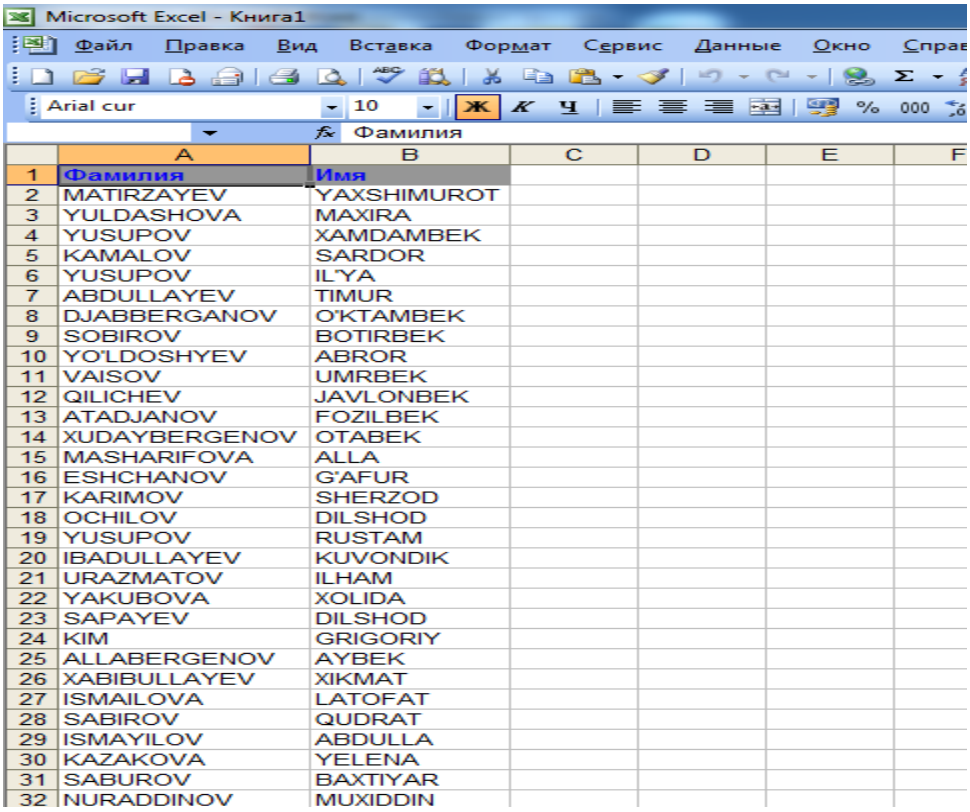
Filter xossasiga shartni kiritish uchun oldin Query1 komponentasi belgilanadi va keyin ob`ekt inspektoridan Filter xossasi topilib unig qatoridan Kodx>139 sharti kiritiladi va keyin Filtered xossasi true qiymatga almashtiriladi.

4.3.Foydalanuvchi yo`riqnomasini ishlab chiqish

Dasur Borland Delphi 7.0 dasturiy vositasi yordamida tuzilgan bo`lib, dasturga kirish parol orqali amalga oshiriladi.Dasturga kirish amalga oshirilgandan so`ng quydagicha oyna xosil bo`ladi.



Bazadagi istalgan ma`lumotlarni dasturiga eksport xam qilish mumkin.



	A	B	C	D	E	F
1	Фамилия	Имя				
2	MATIRZAYEV	YAXSHIMUROT				
3	YULDASHOVA	MAXIRA				
4	YUSUPOV	XAMDAMBEK				
5	KAMALOV	SARDOR				
6	YUSUPOV	IL'YA				
7	ABDULLAYEV	TIMUR				
8	DJABBERGANOV	O'KTAMBEK				
9	SOBIROV	BOTIRBEK				
10	YO'LDOSHYEV	ABROR				
11	VAISOV	UMRBEK				
12	QILICHEV	JAVLONBEK				
13	ATADJANOV	FOZILBEK				
14	XUDAYBERGENOV	OTABEK				
15	MASHARIFOVA	ALLA				
16	ESHCHANOV	G'AFUR				
17	KARIMOV	SHERZOD				
18	OCHILOV	DILSHOD				
19	YUSUPOV	RUSTAM				
20	IBADULLAYEV	KUVONDIK				
21	URAZMATOV	ILHAM				
22	YAKUBOVA	XOLIDA				
23	SAPAYEV	DILSHOD				
24	KIM	GRIGORIY				
25	ALLABERGENOV	AYBEK				
26	XABIBULLAYEV	XIKMAT				
27	ISMAILOVA	LATOFAT				
28	SABIROV	QUDRAT				
29	ISMAYILOV	ABDULLA				
30	KAZAKOVA	YELENA				
31	SABUROV	BAXTIYAR				
32	NURADDINOV	MUXIDDIN				

Dasturning eng asosiy imkonyatlaridan biri bu talabalar haqida turli xildagi hisobotlarni tayyorlay olishidir.

Xulosa

Diplom ishi loyihasining asosiy maqsati talabalarning o`zlashtirishi bo`yicha dasturiy mahsulotni yaratish. Bu masalining bajarilishi uchun quyidagi bosqichlar amalga oshiriladi.

- Ma`lumotlar bazasining mantiqiy modeli yaratiladi;
- Har bir oy bo`yicha mutaxassislik, guruh, talabalar va fanlar bo`yicha baholarning kiritilish tuziladi;
- Ma`lumotlarni tuzatish mumkinchiligi nazarga olinadi.

O`zlashtirish natijalari kolledj, guruh, mutaxassisliklar natijalari bo`yicha tuziladi.(o`zlashtirishning % lik tarkibi va o`zlashtirish sifati).

Oxirgi ma`lumotlar oldingilari bilan toqqoslanadi.

O`zlashtirishi qanoatlanarsiz va a`lo talabalar ro`yhatini tuzish ta`minlanadi.

Hisobotlarni tuzish ta`minlanadi.

Natijada, o`qish o`rinlarida qo`llaniladigan «O`zlashtirishni hisoblash» dasturi tuziladi.

Diplom ishining bajarilish borishida o`zlashtirishni hisoblash mumkinchiligini ta`minlovchi dasturiy mahsulot yaratildi.

Dastur juda oddiy va qulayli qilib yaratilgan bo`lib, uni egallash bo`yicha maxsus o`qish talab etilmaydi. Bu dastur normal` ishni ta`minlovchi funktsiyalarda bajaradi.

Ma`lumotlar bazasini yaratish uchun Microsoft Office Access qo`llaniladi. Ma`lumotlar bazasi bilan ishlash mumkinchiligi Delphi, dasturlash tilida olib boriladi. Ma`lumotlar bazasi bilan bog`lanish BDE texnologiyasi asosida amalga oshiriladi. Ma`lumotlarni qayta ishlash SQL tilining operatorlari yordamida bajarilib, u dastur ishini bir qancha tezlashtiradi.

Dasturiy mahsulot o`qish o`rinlarida qo`llanish uchun maqsadga muofiq.

Foydalanilgan adabiyotlar.

1. Gofman V. E., Xomonenko A. D. Delphi. Выстрыу start. — SPb.: BXV-Peterburg, 2003. — 288 s.

2. Gofman V. E., Xomonenko A. D. Rabota s bazami dannых v Delphi. — SPb.: BXV-Peterburg, 2001. — 656 s.

3. Borovskiy A. N. Programmirovanie v Delphi 2005. — SPb.: BXV-Peterburg, 2005. - 448 s.

4. Daraxvelidze P. G., Markov e. P. Delphi 2005 dlya Win32. - SPb.: BXV-Peterburg, 2005. - 1136 s.

5. Sorokin A. V. Delphi. Razrabotka baz dannых. - SPb.: Piter, 2005. — 477 s.

6. Flenov M. e. Bibliya Delphi. — SPb.: BXV-Peterburg, 2004. — 880 s.

7. Flenov M. e. Programmirovanie v Delphi glazami xakera. — SPb.: BXV-Peterburg, 2003. - 368 s.

8. Flenov M. e. Delphi v shutku i vser`ez: chto umeyut xakerы (+CD). — SPb.: Piter. 2006. — 271 s.

9. Arxangel`skiy L.Ya. Delphi 2006. Spravochnoe posobie: Yazыk Delphi, klassы, funktsii Win32 i .NET. — M.: OOO «Binom-Press», 2006 g. — 1 152 s.

10. Faronov V. V. Delphi 2005. Razrabotka prilojeniy dlya baz dannых i Interneta. — SPb.: Piter, 2006. — 603 s.

11. Karpova T. S. Bazы dannых: modeli, razrabotka, realizatsiya. — SPb.: Piter, 2001. — 304 s.

12. Illyustrirovannыy samouchitel` po Delphi dlya nachinayuщix. Elektronnoe posobie.

13. Illyustrirovannыy samouchitel` po Delphi dlya professionalov. Elektronnoe posobie.

ILOVA.

```
unit Unit12;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, Menus, DB, ADODB;
type
 TForm12 = class(TForm)
 MainMenu1: TMainMenu;
 N1: TMenuItem;
 N2: TMenuItem;
 N3: TMenuItem;
 N4: TMenuItem;
 N8: TMenuItem;
 N9: TMenuItem;
 ADOConnection1: TADOConnection;
 N10: TMenuItem;
 N11: TMenuItem;
 N12: TMenuItem;
 procedure N9Click(Sender: TObject);
 procedure N11Click(Sender: TObject);
 procedure N12Click(Sender: TObject);
 procedure FormCreate(Sender: TObject);
 procedure N2Click(Sender: TObject);
 procedure N3Click(Sender: TObject);
 procedure N4Click(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form12: TForm12;
implementation
uses Unit13, Unit15, Unit14, Unit16, Unit17, Unit18;
{$R *.dfm}
procedure TForm12.N9Click(Sender: TObject);
begin
 Form13.Show;
end;
procedure TForm12.N11Click(Sender: TObject);
begin
Form15.Show;
end;
procedure TForm12.N12Click(Sender: TObject);
begin
Form14.Show;
end;
procedure TForm12.FormCreate(Sender: TObject);
begin
//Form16.Show;
end;
procedure TForm12.N2Click(Sender: TObject);
begin
Form16.Show;
end;
procedure TForm12.N3Click(Sender: TObject);
begin
Form17.show;
end;
procedure TForm12.N4Click(Sender: TObject);
begin
Form18.show;
```

```
end;
end.

unit Unit13;
interfa
ceuses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,Unit12, Forms,
 Dialogs, ExcelXP, OleServer, Grids, DBGrids, StdCtrls, DB, ADODB;
type
 TForm13 = class(TForm)
 Label1: TLabel;
 DBGrid1: TDBGrid;
 ExcelWorkbook1: TExcelWorkbook;
 ExcelApplication1: TExcelApplication;
 ADOQuery1: TADOQuery;
 DataSource1: TDataSource;
 Button1: TButton;
 ADOQuery2: TADOQuery;
 ADOQuery1gruppa: TWideStringField;
 ADOQuery1fio: TWideStringField;
 ADOQuery1Datapr: TWideStringField;
 ADOQuery1Prim: TWideStringField;
 procedure Button1Click(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form13: TForm13;
implementation
{$R *.dfm}
procedure TForm13.Button1Click(Sender: TObject);
var n: OleVariant;
i:integer;
//S:String;
begin
ADOQuery2.ExecSQL;
AdoQuery1.Open;
n:='d:\55\spisok.xls';
ExcelApplication1.Workbooks.Add(n,0);
Excelworkbook1.ConnectTo(ExcelApplication1.ActiveWorkbook);
i:=2;
ADOQUERY1.Insert;
ExcelApplication1.Visible[0]:=False;
While i<2000 do
begin;
ADOQuery1.FieldByName('gruppa').AsString:=ExcelApplication1.Cells.Item[i,1].Value;
ADOQuery1.FieldByName('Fio').AsString:=ExcelApplication1.Cells.Item[i,2].Value;
ADOQuery1.FieldByName('Datapr').AsString:=ExcelApplication1.Cells.Item[i,3].Value;
ADOQuery1.FieldByName('Prim').AsString:=ExcelApplication1.Cells.Item[i,4].Value;
 If ADOQuery1.FieldByName('gruppa').AsString='' then
 begin
 ADOQUERY1.Delete;
 i:=2001;
 end;
ADOQUERY1.Insert;
i:=i+1;
 end;
ADOQuery1.Open;
Showmessage('Priem dannых vыpolnen');
end;
```

```
end.

unit Unit14;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics,Unit12, Controls, Forms,
 Dialogs, DB, ADODB, ExtCtrls, DBCtrls, Grids, DBGrids;
type
 TForm14 = class(TForm)
 DBGrid1: TDBGrid;
 DBNavigator1: TDBNavigator;
 DataSource1: TDataSource;
 ADOQuery1: TADOQuery;
 ADOQuery1Disziplina: TWideStringField;
 procedure FormCreate(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form14: TForm14;
implementation
{$R *.dfm}
procedure TForm14.FormCreate(Sender: TObject);
begin
ADOQuery1.Active:=true;
end;
end.

unit Unit15;
interface
uses
 Windows, Messages, SysUtils, Variants,Unit12,Classes, Graphics, Controls, Forms,
 Dialogs, DB, ADODB, ExtCtrls, DBCtrls, Grids, DBGrids;
type
 TForm15 = class(TForm)
 ADOQuery1: TADOQuery;
 DBGrid1: TDBGrid;
 DBNavigator1: TDBNavigator;
 DataSource1: TDataSource;
 ADOQuery1Grupa: TWideStringField;
 ADOQuery1Spez: TWideStringField;
 procedure FormCreate(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form15: TForm15;
implementation
{$R *.dfm}
procedure TForm15.FormCreate(Sender: TObject);
begin
ADOQuery1.Active:=true;
end;
end.

unit Unit16;
interface
uses
```

```
 Windows, Messages, SysUtils, Variants, Classes,Unit12, Graphics, Controls, Forms,
 Dialogs, DB, ADODB, StdCtrls, ExtCtrls, DBCtrls, Grids, DBGrids;
type
 TForm16 = class(TForm)
 ADOQuery1: TADOQuery;
 DBGrid1: TDBGrid;
 DBNavigator1: TDBNavigator;
 DataSource1: TDataSource;
 ADOQuery1stud: TWideStringField;
 ComboBox1: TComboBox;
 ComboBox2: TComboBox;
 ComboBox3: TComboBox;
 ComboBox4: TComboBox;
 ComboBox5: TComboBox;
 Label1: TLabel;
 Label2: TLabel;
 Label3: TLabel;
 Label4: TLabel;
 Label5: TLabel;
 Button1: TButton;
 Dobav: TADOQuery;
 ADOQuery1ozenka: TIntegerField;
 ADOQuery1koddis: TWideStringField;
 ADOQuery2: TADOQuery;
 ADOTable1: TADOTable;
 ADOTable2: TADOTable;
 ADOTable1DSDesigner: TAutoIncField;
 ADOTable1Grupa: TWideStringField;
 ADOTable1Disziplina: TWideStringField;
 procedure Button1Click(Sender: TObject);
 procedure ComboBox4Change(Sender: TObject);
 procedure FormCreate(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form16: TForm16;
implementation
{$R *.dfm}
procedure TForm16.Button1Click(Sender: TObject);
begin
Dobav.SQL.Clear;
ADOQuery2.SQL.Clear;
ADOQuery2.SQL.Text:='delete from ozenka where(ozenka.kodspez = '''+Combobox4.Text+'''and
ozenka.koddis = '''+Combobox3.Text+'''and ozenka.mes = '''+Combobox1.Text+'''and ozenka.god =
'''+Combobox2.Text+''' and ozenka.gruppa='''+Combobox5.Text+''')';
ADOQuery2.ExecSQL;
Dobav.SQL.Add('Insert Into Ozenka ( gruppa,stud)');
Dobav.SQL.Add('Select gruppa,fio');
Dobav.SQL.Add(' From stud where (gruppa='''+Combobox5.text+''')');
//ShowMessage(Dobav.SQL.Text); //Dobav.SQL.Add('
Dobav.ExecSQL;
Dobav.SQL.Clear;
Dobav.SQL.Add('UPDATE ozenka SET ozenka.kodspez = '''+Combobox4.Text+''', ozenka.koddis =
'''+Combobox3.Text+''', ozenka.mes = '''+Combobox1.Text+''', ozenka.god = '''+Combobox2.Text+'''
WHERE (((ozenka.gruppa)='''+Combobox5.Text+''')and(ozenka.koddis is null ));');
 Dobav.ExecSQL;
ADOQUERY1.SQL.Clear;
```

```
ADOQUERY1.SQL.Add('Select* from ozenka where(ozenka.kodspez = '"+Combobox4.Text+'"and
ozenka.koddis = '"+Combobox3.Text+'"and ozenka.mes = '"+Combobox1.Text+'"and ozenka.god =
'"+Combobox2.Text+'" and ozenka.gruppa='"+Combobox5.Text+'")');
//ShowMessage(ADOQUERY1.SQL.Text);
ADOQUERY1.Active:=True;
end;
procedure TForm16.ComboBox4Change(Sender: TObject);
begin
ADOTable2.Open;
ADOTable2.First;
ComboBox5.Items.Clear;
While not ADOTable2.Eof do
begin
if ADOTable2.Fieldbyname('Spez').AsString=ComboBox4.Text Then
ComboBox5.Items.Add(ADOTable2.fieldbyname('grupa').AsString);
ADOTable2.Next;
end;
ADOTable2.Close;
ComboBox5.Sorted:=True;
end;
procedure TForm16.FormCreate(Sender: TObject);
begin
ADOTable1.Open;
ADOTable1.First;
ComboBox3.Items.Clear;
While not ADOTable1.Eof do
begin
ComboBox3.Items.Add(ADOTable1.fieldbyname('disziplina').AsString);
//ShowMessage(ADOTable1.fieldbyname('disziplina').AsString);
ADOTable1.Next;
end;
ADOTable1.Close;
ComboBox3.Sorted:=True;
end;
end.

unit Unit17;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, DB, ADODB, StdCtrls, ExtCtrls, DBCtrls, Grids, DBGrids;
type
 TForm17 = class(TForm)
 DBGrid1: TDBGrid;
 DBNavigator1: TDBNavigator;
 Label1: TLabel;
 Label2: TLabel;
 Label3: TLabel;
 Label4: TLabel;
 Label5: TLabel;
 ComboBox1: TComboBox;
 ComboBox2: TComboBox;
 ComboBox3: TComboBox;
 ComboBox5: TComboBox;
 Button1: TButton;
 ADOQuery1: TADOQuery;
 ADOQuery1stud: TWideStringField;
 DataSource1: TDataSource;
 ADOQuery1ozenka: TIntegerField;
 ADOQuery1koddis: TWideStringField;
 ComboBox4: TComboBox;
 ADOTable1: TADOTable;
```

```
 ADOTable2: TADOTable;
 procedure Button1Click(Sender: TObject);
 procedure FormCreate(Sender: TObject);
 procedure ComboBox4Change(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form17: TForm17;
implementation
{$R *.dfm}
procedure TForm17.Button1Click(Sender: TObject);
begin
ADOQUERY1.SQL.Clear;
ADOQUERY1.SQL.Add('Select* from ozenka where(ozenka.kodspez = '''+Combobox4.Text+'''and
ozenka.koddis = '''+Combobox3.Text+'''and ozenka.mes = '''+Combobox1.Text+'''and ozenka.god =
'''+Combobox2.Text+''' and ozenka.gruppa='''+Combobox5.Text+''')');
//ShowMessage(ADOQUERY1.SQL.Text);
ADOQUERY1.Active:=True;
end;
procedure TForm17.FormCreate(Sender: TObject);
begin
ADOTable1.Open;
ADOTable1.First;
ComboBox3.Items.Clear;
While not ADOTable1.Eof do
begin
ComboBox3.Items.Add(ADOTable1.fieldbyname('disziplina').AsString);
//ShowMessage(ADOTable1.fieldbyname('disziplina').AsString);
ADOTable1.Next;
end;
ADOTable1.Close;
ComboBox3.Sorted:=True;
end;
procedure TForm17.ComboBox4Change(Sender: TObject);
begin
ADOTable2.Open;
ADOTable2.First;
ComboBox5.Items.Clear;
While not ADOTable2.Eof do
begin
if ADOTable2.Fieldbyname('Spez').AsString=ComboBox4.Text Then
ComboBox5.Items.Add(ADOTable2.fieldbyname('grupa').AsString);
ADOTable2.Next;
end;
ADOTable2.Close;
ComboBox5.Sorted:=True;
end;
end.

unit Unit18;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, StdCtrls, DB, ADODB, Buttons, ExtCtrls, ComCtrls, ExcelXP,
 OleServer, Grids, DBGrids;
type
 TForm18 = class(TForm)
 PageControl1: TPageControl;
 TabSheet1: TTabSheet;
```

```
 TabSheet2: TTabSheet;
 TabSheet3: TTabSheet;
 RadioGroup1: TRadioGroup;
 BitBtn1: TBitBtn;
 Label1: TLabel;
 Label2: TLabel;
 ComboBox1: TComboBox;
 ComboBox2: TComboBox;
 Label4: TLabel;
 Label5: TLabel;
 ADOQuery1: TADOQuery;
 Edit1: TEdit;
 Edit2: TEdit;
 Label3: TLabel;
 Label6: TLabel;
 Label7: TLabel;
 ComboBox3: TComboBox;
 ComboBox6: TComboBox;
 Label8: TLabel;
 Label9: TLabel;
 RadioGroup3: TRadioGroup;
 ExcelWorkbook1: TExcelWorkbook;
 ExcelApplication1: TExcelApplication;
 BitBtn2: TBitBtn;
 ADOQuery2: TADOQuery;
 ComboBox7: TComboBox;
 ComboBox8: TComboBox;
 Label10: TLabel;
 Label11: TLabel;
 BitBtn3: TBitBtn;
 ADOQuery3: TADOQuery;
 ADOQuery4: TADOQuery;
 ADOQuery5: TADOQuery;
 ADOTable2: TADOTable;
 ComboBox4: TComboBox;
 ComboBox5: TComboBox;
 DataSource1: TDataSource;
 ADOQuery6: TADOQuery;
 ADOTable6: TADOTable;
 Button1: TButton;
 procedure BitBtn1Click(Sender: TObject);
 procedure BitBtn2Click(Sender: TObject);
 procedure BitBtn3Click(Sender: TObject);
 procedure Button1Click(Sender: TObject);
 procedure ComboBox4Change(Sender: TObject);
 // procedure ComboBox4Change(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;
var
 Form18: TForm18;
implementation
{$R *.dfm}
procedure TForm18.BitBtn1Click(Sender: TObject);
var
k1:integer;
k2:integer;
k3:integer;
k4,k:double;
begin
```

```
//po gruppe
If radiogroup1.ItemIndex=0
then begin
ADOQuery1.SQL.Clear;
AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count1] FROM ozenka where
(((ozenka.ozenka)>3))GROUP BY ozenka.mes, ozenka.god, ozenka.gruppa ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.gruppa)="'+Combobox5.Text+'"))');
 AdoQuery1.Active:=True;
k1:=AdoQuery1.fieldbyname('Count1').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count2] FROM ozenka where
(((ozenka.ozenka)>0))GROUP BY ozenka.mes, ozenka.god, ozenka.gruppa ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.gruppa)="'+Combobox5.Text+'"))');
 AdoQuery1.Active:=True;
k2:=AdoQuery1.fieldbyname('Count2').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count3] FROM ozenka where
(((ozenka.ozenka)>2))GROUP BY ozenka.mes, ozenka.god, ozenka.gruppa ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.gruppa)="'+Combobox5.Text+'"))');
 AdoQuery1.Active:=True;
k3:=AdoQuery1.fieldbyname('Count3').AsInteger;
k4:=Round((k3/k2*100)*100)/100;
k:=Round((k1/k2*100)*100)/100;
Edit1.Text:=FloatToStr(k4);
Edit2.Text:=FloatToStr(k);
end;
 //po spetsial`nosti
If radiogroup1.ItemIndex=1
then begin
 ADOQuery1.SQL.Clear;
AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count1] FROM ozenka where
(((ozenka.ozenka)>3))GROUP BY ozenka.mes, ozenka.god, ozenka.kodspez ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.kodspez)="'+Combobox4.Text+'"))');
 AdoQuery1.Active:=True;
k1:=AdoQuery1.fieldbyname('Count1').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count2] FROM ozenka where
(((ozenka.ozenka)>0))GROUP BY ozenka.mes, ozenka.god, ozenka.kodspez ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.kodspez)="'+Combobox4.Text+'"))');
 AdoQuery1.Active:=True;
k2:=AdoQuery1.fieldbyname('Count2').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count3] FROM ozenka where
(((ozenka.ozenka)>2))GROUP BY ozenka.mes, ozenka.god, ozenka.kodspez ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.kodspez)="'+Combobox4.Text+'"))');
 AdoQuery1.Active:=True;
k3:=AdoQuery1.fieldbyname('Count3').AsInteger;
k4:=Round((k3/k2*100)*100)/100;
k:=Round((k1/k2*100)*100)/100;
Edit1.Text:=FloatToStr(k4);
Edit2.Text:=FloatToStr(k);
end;
//po kolledju
If radiogroup1.ItemIndex=2
then begin
ADOQuery1.SQL.Clear;
```

```
AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count1] FROM ozenka where
(((ozenka.ozenka)>3))GROUP BY ozenka.mes, ozenka.god ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'"))');
 AdoQuery1.Active:=True;
k1:=AdoQuery1.fieldbyname('Count1').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count2] FROM ozenka where
(((ozenka.ozenka)>0))GROUP BY ozenka.mes, ozenka.god ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'"))');
 AdoQuery1.Active:=True;
k2:=AdoQuery1.fieldbyname('Count2').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count3] FROM ozenka where
(((ozenka.ozenka)>2))GROUP BY ozenka.mes, ozenka.god ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'"))');
 AdoQuery1.Active:=True;
k3:=AdoQuery1.fieldbyname('Count3').AsInteger;
k4:=Round((k3/k2*100)*100)/100;
k:=Round((k1/k2*100)*100)/100;
Edit1.Text:=FloatToStr(k4);
Edit2.Text:=FloatToStr(k);
end;
end;
procedure TForm18.BitBtn2Click(Sender: TObject);
var n: OleVariant;
i:integer;
//S:String;
begin
if radiogroup3.ItemIndex=0 then begin
AdoQuery1.Close;
ADOQuery1.SQL.Clear;
ADOQuery1.SQL.Add('SELECT ozenka.ozenka, ozenka.stud, ozenka.gruppa, ozenka.koddis,
ozenka.kodspez, ozenka.mes, ozenka.god FROM ozenka WHERE (((ozenka.ozenka)=2))');
ADOQuery1.SQL.Add('and mes="'+ combobox3.Text+'" and god="'+combobox6.Text+'"');
//Showmessage(adoquery1.SQL.Text);
ADOQuery1.Open;
ADOQuery1.First;
n:='d:\55\spisok55.xls';
ExcelApplication1.Workbooks.Add(n,0);
Excelworkbook1.ConnectTo(ExcelApplication1.ActiveWorkbook);
i:=3;
ExcelApplication1.Cells.Item[1,1].Value:='Cpisok neuspevayuщix';
ExcelApplication1.Cells.Item[1,2].Value:=Combobox3.Text;
ExcelApplication1.Cells.Item[1,3].Value:=Combobox6.Text;
ExcelApplication1.Visible[0]:=true;
While not ADOQuery1.Eof do
begin;
ExcelApplication1.Cells.Item[i,1].Value:=ADOQuery1.FieldByName('Stud').AsString;
ExcelApplication1.Cells.Item[i,2].Value:=ADOQuery1.FieldByName('koddis').AsString;
ExcelApplication1.Cells.Item[i,3].Value:=ADOQuery1.FieldByName('ozenka').AsString;
ExcelApplication1.Cells.Item[i,4].Value:=ADOQuery1.FieldByName('gruppa').AsString;
 ADOQUERY1.Next;
//ADOQUERY1.Post;
i:=i+1;
 end;
end;
if radiogroup3.ItemIndex=1 then begin
AdoQuery1.Close;
ADOQuery1.SQL.Clear;
```

```
ADOQuery1.SQL.Add('SELECT ozenka.stud, Avg(ozenka.ozenka) AS [Avg-ozenka],
ozenka.mes,ozenka.gruppa, ozenka.god FROM ozenka');
ADOQuery1.SQL.Add('GROUP BY ozenka.stud, ozenka.mes,ozenka.gruppa, ozenka.god');
ADOQuery1.SQL.Add('HAVING (((Avg(ozenka.ozenka))=5));');
//ADOQuery1.SQL.Add('and mes='"'+ combobox3.Text+'" and god='"'+combobox6.Text+'"');
ADOQuery1.Open;
ADOQuery1.First;
n:='d:\55\spisok55.xls';
ExcelApplication1.Workbooks.Add(n,0);
Excelworkbook1.ConnectTo(ExcelApplication1.ActiveWorkbook);
i:=3;
ExcelApplication1.Cells.Item[1,1].Value:='Cpisok uspevayuщix na otlichno';
ExcelApplication1.Cells.Item[1,2].Value:=Combobox3.Text;
ExcelApplication1.Cells.Item[1,3].Value:=Combobox6.Text;
ExcelApplication1.Visible[0]:=true;
While not ADOQuery1.Eof do
begin;
ExcelApplication1.Cells.Item[i,1].Value:=ADOQuery1.FieldByName('Stud').AsString;
ExcelApplication1.Cells.Item[i,2].Value:='vse';
ExcelApplication1.Cells.Item[i,3].Value:='5';
ExcelApplication1.Cells.Item[i,4].Value:=ADOQuery1.FieldByName('gruppa').AsString;
 ADOQUERY1.Next;
//ADOQUERY1.Post;
i:=i+1;
 end;
end;
end;
procedure TForm18.BitBtn3Click(Sender: TObject);
var
 n: OleVariant;
i:integer;
k1:integer;
k2:integer;
k3:integer;
k4,k:double;
begin
ADOQuery4.SQL.Text:='drop table ots1';
ADOQuery4.ExecSQL;
//ADoQuery.Active:=false;
ADoQuery2.Active:=true;
ADOQuery2.Edit;
ADoQuery2.First;
ADoQuery5.Active:=true;
ADoQuery5.First;
ADOQuery2.insert;
ADoQuery3.SQL.Text:='Select* from spez';
ADOQuery3.Active:=true;
While not ADoQuery3.Eof do
begin
ADOQuery2.FieldByName('Gruppa').AsString:=ADOQuery3.FieldByName('Grupa').AsString;
ADOQuery2.FieldByName('Spez').AsString:=ADOQuery3.FieldByName('Spez').AsString;
ADOQuery2.FieldByName('kolledg').AsString:='koledg';
Combobox1.Text:=Combobox7.Text;
Combobox2.Text:=Combobox8.Text;
Combobox5.Text:=ADOQuery2.FieldByName('Gruppa').AsString;
Combobox4.Text:=ADOQuery2.FieldByName('Spez').AsString;
//Combobox1.Text:=Combobox7.Text;
//Combobox2.Text:=Combobox8.Text;
ADOQuery1.SQL.Clear;
AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count1] FROM ozenka where
(((ozenka.ozenka)>3))GROUP BY ozenka.mes, ozenka.god, ozenka.gruppa ');
```

```
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.gruppa)="'+Combobox5.Text+'"))');
 AdoQuery1.Active:=True;
k1:=AdoQuery1.fieldbyname('Count1').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count2] FROM ozenka where
(((ozenka.ozenka)>0))GROUP BY ozenka.mes, ozenka.god, ozenka.gruppa ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.gruppa)="'+Combobox5.Text+'"))');
 AdoQuery1.Active:=True;
k2:=AdoQuery1.fieldbyname('Count2').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count3] FROM ozenka where
(((ozenka.ozenka)>2))GROUP BY ozenka.mes, ozenka.god, ozenka.gruppa ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.gruppa)="'+Combobox5.Text+'"))');
 AdoQuery1.Active:=True;
k3:=AdoQuery1.fieldbyname('Count3').AsInteger;
//ShowMessage(FloatToStr(k4));
// ShowMessage(FloatToStr(k));
 IF K2>0 then begin
k4:=Round((k3/k2*100));
k:=Round((k1/k2*100));
//Edit1.Text:=FloatToStr(k4);
//Edit2.Text:=FloatToStr(k);
ADOQuery2.FieldByName('usp').AsString:=FloatToStr(k4);
ADOQuery2.FieldByName('kas').AsString:=FloatToStr(k);
ADOQuery2.FieldByName('kol').AsString:=FloatToStr(k2);
end;
ADOQuery2.Insert;
ADOQuery3.Next;
end;
//po spetsial`nostyam
ADoQuery2.Active:=true;
ADOQuery2.Edit;
ADoQuery2.First;
ADoQuery3.SQL.Text:='Select spez from spez';
ADoQuery3.Active:=true;
ADoQuery3.First;
ADOQuery2.insert;
While not ADoQuery3.Eof do
begin
//ADOQuery2.FieldByName('Gruppa').AsString:=ADOQuery3.FieldByName('Grupa').AsString;
ADOQuery2.FieldByName('Spez').AsString:=ADOQuery3.FieldByName('Spez').AsString;
ADOQuery2.FieldByName('kolledg').AsString:='koledg';
Combobox1.Text:=Combobox7.Text;
Combobox2.Text:=Combobox8.Text;
//Combobox5.Text:=ADOQuery2.FieldByName('Gruppa').AsString;
Combobox4.Text:=ADOQuery2.FieldByName('Spez').AsString;
//Combobox1.Text:=Combobox7.Text;
//Combobox2.Text:=Combobox8.Text;
ADOQuery1.SQL.Clear;
AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count1] FROM ozenka where
(((ozenka.ozenka)>3))GROUP BY ozenka.mes, ozenka.god, ozenka.kodspez');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)="'+Combobox1.Text+'") AND
((ozenka.god)="'+Combobox2.Text+'") AND ((ozenka.kodspez)="'+Combobox4.Text+'"))');
 AdoQuery1.Active:=True;
k1:=AdoQuery1.fieldbyname('Count1').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count2] FROM ozenka where
(((ozenka.ozenka)>0))GROUP BY ozenka.mes, ozenka.god, ozenka.kodspez ');
```

```
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)='''+Combobox1.Text+''') AND
((ozenka.god)='''+Combobox2.Text+''') AND ((ozenka.kodspez)='''+Combobox4.Text+'''))');
 AdoQuery1.Active:=True;
k2:=AdoQuery1.fieldbyname('Count2').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count3] FROM ozenka where
(((ozenka.ozenka)>2))GROUP BY ozenka.mes, ozenka.god, ozenka.kodspez ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)='''+Combobox1.Text+''') AND
((ozenka.god)='''+Combobox2.Text+''') AND ((ozenka.kodspez)='''+Combobox4.Text+'''))');
 AdoQuery1.Active:=True;
k3:=AdoQuery1.fieldbyname('Count3').AsInteger;
//ShowMessage(FloatToStr(k4));
 //ShowMessage(FloatToStr(k));
 IF K2>0 then begin
k4:=Round((k3/k2*100));
k:=Round((k1/k2*100));
//Edit1.Text:=FloatToStr(k4);
//Edit2.Text:=FloatToStr(k);
ADOQuery2.FieldByName('usp').AsString:=FloatToStr(k4);
ADOQuery2.FieldByName('kas').AsString:=FloatToStr(k);
ADOQuery2.FieldByName('kol').AsString:=FloatToStr(k2);
end;
ADOQuery2.Insert;
ADOQuery3.Next;
end;
//po kolledju
ADOQuery2.insert;
//ADOQuery2.FieldByName('Gruppa').AsString:=ADOQuery3.FieldByName('Grupa').AsString;
//ADOQuery2.FieldByName('Spez').AsString:=ADOQuery3.FieldByName('Spez').AsString;
ADOQuery2.FieldByName('kolledg').AsString:='koledg';
Combobox1.Text:=Combobox7.Text;
Combobox2.Text:=Combobox8.Text;
//Combobox5.Text:=ADOQuery2.FieldByName('Gruppa').AsString;
Combobox4.Text:=ADOQuery2.FieldByName('Spez').AsString;
//Combobox1.Text:=Combobox7.Text;
//Combobox2.Text:=Combobox8.Text;
ADOQuery1.SQL.Clear;
AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count1] FROM ozenka where
(((ozenka.ozenka)>3))GROUP BY ozenka.mes, ozenka.god');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)='''+Combobox1.Text+''') AND
((ozenka.god)='''+Combobox2.Text+'''))');
 AdoQuery1.Active:=True;
k1:=AdoQuery1.fieldbyname('Count1').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count2] FROM ozenka where
(((ozenka.ozenka)>0))GROUP BY ozenka.mes, ozenka.god ');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)='''+Combobox1.Text+''') AND
((ozenka.god)='''+Combobox2.Text+'''))');
 AdoQuery1.Active:=True;
k2:=AdoQuery1.fieldbyname('Count2').AsInteger;
 ADOQuery1.SQL.Clear;
 AdoQuery1.SQL.Add('SELECT Count(ozenka.stud) AS [Count3] FROM ozenka where
(((ozenka.ozenka)>2))GROUP BY ozenka.mes, ozenka.god');
AdoQuery1.SQL.Add('HAVING (((ozenka.mes)='''+Combobox1.Text+''') AND
((ozenka.god)='''+Combobox2.Text+'''))');
 AdoQuery1.Active:=True;
k3:=AdoQuery1.fieldbyname('Count3').AsInteger;
//ShowMessage(FloatToStr(k4));
 //ShowMessage(FloatToStr(k));
 IF K2>0 then begin
k4:=Round((k3/k2*100));
k:=Round((k1/k2*100));
```

```
//Edit1.Text:=FloatToStr(k4);
//Edit2.Text:=FloatToStr(k);
ADOQuery2.FieldByName('usp').AsString:=FloatToStr(k4);
ADOQuery2.FieldByName('kas').AsString:=FloatToStr(k);
ADOQuery2.FieldByName('kol').AsString:=FloatToStr(k2);
ADOQuery2.Post;
end;
//S:String;
begin
//if radiogroup3.ItemIndex=0 then begin
AdoQuery6.Close;
ADOQuery6.SQL.Clear;
ADOQuery6.SQL.Add('SELECT ots.kolledg, ots.spez, ots.gruppa, ots.kol, ots.usp, ots.kas into ots1 FROM
ots GROUP BY ots.kolledg, ots.spez, ots.gruppa, ots.kol, ots.usp, ots.kas');
//ORDER BY ots.kolledg DESC , ots.spez DESC , ots.gruppa DESCSELECT ozenka.ozenka, ozenka.stud,
ozenka.gruppa, ozenka.koddis, ozenka.kodspez, ozenka.mes, ozenka.god FROM ozenka WHERE
(((ozenka.ozenka)=2))');
ADOQuery6.SQL.Add('ORDER BY ots.kolledg DESC , ots.spez DESC , ots.gruppa DESC');
//Showmessage(adoquery1.SQL.Text);
ADOQuery6.ExecSQL;
{ADOTable6.TableName:='ots1';
ADOTable6.Active:=true;
ADOTable6.Active:=false;
ADOTable6.Active:=true;
//ADOTable6.First;
n:='d:\55\usp.xls';
ExcelApplication1.Workbooks.Add(n,0);
Excelworkbook1.ConnectTo(ExcelApplication1.ActiveWorkbook);
i:=10;
ExcelApplication1.Cells.Item[1,2].Value:=Combobox7.Text;
ExcelApplication1.Cells.Item[1,3].Value:=Combobox8.Text;
 ADOTable6.First;
 //
 If ADOTable6.FieldByName('gruppa').AsString<>'' then
ExcelApplication1.Cells.Item[i,2].Value:=ADOTable6.FieldByName('gruppa').AsString
 else
// If (ADOQuery6.FieldByName('Spez').AsString<>'') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po spetsial`nosti
'+ADOTable6.FieldByName('Spez').AsString;
If (ADOTable6.FieldByName('Spez').AsString='') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po kolledju
';//+ADOQuery1.FieldByName('Kolledg').AsString;
ExcelApplication1.Cells.Item[i,3].Value:=ADOTable6.FieldByName('kol').AsString;
ExcelApplication1.Cells.Item[i,4].Value:=ADOTable6.FieldByName('usp').AsString;
ExcelApplication1.Cells.Item[i,5].Value:=ADOTable6.FieldByName('kas').AsString;
//While not ADOTable6.Eof do
begin;
ShowMessage('fdgiiii');
 k4:= ADOTable6.FieldByName('usp').AsFloat;
 k:=ADOTable6.FieldByName('kas').AsFloat;
 ShowMessage(FloatToStr(k4));
If ADOTable6.FieldByName('gruppa').AsString<>'' then
ExcelApplication1.Cells.Item[i,2].Value:=ADOTable6.FieldByName('gruppa').AsString
 else
// If (ADOQuery6.FieldByName('Spez').AsString<>'') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po spetsial`nosti
'+ADOTable6.FieldByName('Spez').AsString;
If (ADOTable6.FieldByName('Spez').AsString='') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po kolledju
';//+ADOQuery1.FieldByName('Kolledg').AsString;
ExcelApplication1.Cells.Item[i,3].Value:=ADOTable6.FieldByName('kol').AsString;
ExcelApplication1.Cells.Item[i,4].Value:=k4;
```

```
ExcelApplication1.Cells.Item[i,5].Value:=k;
ADOTable6.Next;
//ADOQUERY1.Post;
i:=i+1;
 end;
end;
//ExcelApplication1.Visible[0]:=true; }
ADOQuery4.SQL.Clear;
ADOQuery4.SQL.Text:='delete * from ots';
ADOQuery4.ExecSQL;
ADOQuery4.SQL.Text:='drop table ots1';
//ADOQuery4.ExecSQL;
//end;
 end;
ShowMessage('Pereraschet dannых uspeshno vыpolnen mojno delat` otchet!');
 end;
procedure TForm18.Button1Click(Sender: TObject);
var
n:Olevariant;
i:Integer;
k1:integer;
k2:integer;
k3:integer;
k4,k:double;
begin
ADOTable6.TableName:='ots1';
ADOTable6.Active:=true;
ADOTable6.Active:=false;
ADOTable6.Active:=true;
//ADOTable6.First;
n:='d:\55\usp.xls';
ExcelApplication1.Workbooks.Add(n,0);
Excelworkbook1.ConnectTo(ExcelApplication1.ActiveWorkbook);
i:=10;
ExcelApplication1.Cells.Item[1,2].Value:=Combobox7.Text;
ExcelApplication1.Cells.Item[1,3].Value:=Combobox8.Text;
 ADOTable6.First;
 // If ADOTable6.FieldByName('gruppa').AsString<>'' then
ExcelApplication1.Cells.Item[i,2].Value:=ADOTable6.FieldByName('gruppa').AsString
 else
// If (ADOQuery6.FieldByName('Spez').AsString<>'') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po spetsial`nosti
'+ADOTable6.FieldByName('Spez').AsString;
If (ADOTable6.FieldByName('Spez').AsString='') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po kolledju
';//+ADOQuery1.FieldByName('Kolledg').AsString;
ExcelApplication1.Cells.Item[i,3].Value:=ADOTable6.FieldByName('kol').AsString;
ExcelApplication1.Cells.Item[i,6].Value:=ADOTable6.FieldByName('usp').AsString;
ExcelApplication1.Cells.Item[i,7].Value:=ADOTable6.FieldByName('kas').AsString;
//While not ADOTable6.Eof do
begin;
ShowMessage(ADOTable6.FieldByName('kas').AsString);
 k4:= ADOTable6.FieldByName('usp').AsFloat;
 k:=ADOTable6.FieldByName('kas').AsFloat;
 //ShowMessage(FloatToStr(k4));
If ADOTable6.FieldByName('gruppa').AsString<>'' then
ExcelApplication1.Cells.Item[i,2].Value:=ADOTable6.FieldByName('gruppa').AsString
 else
// If (ADOQuery6.FieldByName('Spez').AsString<>'') then
ExcelApplication1.Cells.Item[i,2].Value:='itogo po spetsial`nosti
'+ADOTable6.FieldByName('Spez').AsString;
If (ADOTable6.FieldByName('Spez').AsString='') then
```

```
ExcelApplication1.Cells.Item[i,2].Value:='itogo po kolledju
';//+ADOQuery1.FieldByName('Kolledg').AsString;
ExcelApplication1.Cells.Item[i,3].Value:=ADOTable6.FieldByName('kol').AsString;
ExcelApplication1.Cells.Item[i,4].Value:=ADOTable6.FieldByName('usp').AsString;
ExcelApplication1.Cells.Item[i,5].Value:=ADOTable6.FieldByName('kas').AsString; ADOTable6.Next;
//ADOQUERY1.Post;
i:=i+1;
 end;
ExcelApplication1.Visible[0]:=true;
 end;
procedure TForm18.ComboBox4Change(Sender: TObject);
begin
ADOTable2.Open;
ADOTable2.First;
ComboBox5.Items.Clear;
While not ADOTable2.Eof do
begin
if ADOTable2.Fieldbyname('Spez').AsString=ComboBox4.Text Then
ComboBox5.Items.Add(ADOTable2.fieldbyname('grupa').AsString);
ADOTable2.Next;
end;
ADOTable2.Close;
ComboBox5.Sorted:=True;
end;
end.
```