O’ZBEKISTON RESPUBLIKASI OLIY VA O’RTA MAXSUS TA’LIM VAZIRLIGI
BUXORO DAVLAT UNIVERSITETI

Qo’lyozma huquqida

UDK 681.3.06:004.42

Shirinov Ziyomat Zoyirovich

Kafedra o’quv yuklamasi va dars taqsimotini qo’llab quvvatlovchi dasturiy tizim yaratish

Mutaxassislik: 5A130202 - “Amaliy matematika va axborot texnologiyalari”

Magistr akademik darajasini olish uchun yozilgan

DISSERTATSIYA

Ilmiy rahbar:
f.mf.n. dots. O.I. Jalolov

Buxoro-2014y.

MUNDARIJA

Kirish

O’zbekiston Respublikasi mustaqillik odimlarini dadil qo’yayotgan hozirgi davrda axborotlashgan jamiyat qurish masalasi mamlakatimiz uchun naqadar katta ahamiyat kasb etayotgani hech kimga sir emas. Internet hayotimizning bir bo’lagiga aylandi, Biz uning xizmatlaridan har kuni foydalanishga odatlandik.

Biz mustaqil O’zbekiston yoshlariga har tomonlama yetuk barkamol avlod bo’lib yetishimiz uchun keng imkoniyatlar yaratib bergan Prezidentimiz Islom Abdug’aniyevich Karimov yosh avlod tarbiyasiga juda kata e’tibor berib kelmoqdalar. Jumladan, asosiy vazifamiz vatanimiz taraqqiyoti va xalqimiz farovonligini yanada yuksaltirish nomli Prezidentimiz Islom Abdug’aniyevich Karimovning 2009-yilning asosiy yakunlari 2010-yilda O’zbekistonning ijtimoiy–iqtisodiy rivojlanishini eng muhim ustuvor yo’nalishlarga bag’ishlangan Vazirlar Mahkamasining majlisidagi ma’ruzasida:

Biz farzandlarimiz nafaqat jismoniy va ma’naviy sog’lom o’sishi, balki ularning eng zamonaviy intelektual bilimlarga ega bo’lgan, uyg’un rivojlangan insonlar bo’lib, XXI asr talablariga to’liq javob beradigan barkamol avlod bo’lib voyaga yetishi uchun zarur bo’lgan barcha imkoniyat va sharoitlarni yaratishni o’z oldimizga maqsad qilib qo’yganmiz.

“O’z oldimizga qo’yilgan yuksak vazifa va marralarga erishishda zamonaviy bilim va tarbiya sohibi bo’lgan, yangicha fikrlaydigan yoshlarimiz bizning asosiy tayanchimiz va suyanchimiz”.[1]

Respublikamizda o’qitish texnologiyalarini zamonaviylashtirishni jadallashtirish rivojlangan iqtisodiyotli mamlakatlarga qaraganda yanada dolzarb ahamiyatga ega. Chunki hozirgi kunda milliy ta`lim tizimining salohiyati iqtisodiy rivojlanishning yanada yuqori pog’onasiga ko’tarilishga amaliy imkoniyat ta`minlovchi asosiy ijtimoiy resurs sifatida gavdalanadi.

Respulikamiz ta’lim tizimidagi asosiy vazifa jahon talablariga mos keluvchi axborot texnologilarini o’qitish jarayoniga qo’llashdan iborat.

O’zbekistonda ta’lim tizimining axborotlashtirilishi xalqaro hamjamiyatda ham tan olindi. Masofaviy ta’limni rivojlantirish bo’yicha bir qator dasturlar ishlab chiqilmoqda. Iqtisodiyot va jamiyatda islohotlarning o’tkazilishi natijasida o’quv jarayonining zahira hajmini keskin oshirish bo’yicha yangi talablar qo’yildi. Hozirgi kunda axborot eng asosiy ishlab chiqaruvchi resurslardan biriga, iqtisodiyot va umuman jamiyatning rivojlanish poydevoriga aylanmoqda.

Yangi axborot – kommunikatsion texnologiyalar hozirgi kunda eng dolzarb mavzulardan biri bo’lib kelmoqda, sababi har bir sohani o’rganish, izlanish va tajriba ortirish uchun turli usullardan foydalanish kerak bo’ladi. Shuning uchun yangi axborot kommunikatsion texnologiyalardan foydalanish maqsadga muvofiqdir.

Respublikamizda olib borilayotgan islohotlarning tarkibida yuqori malakali mutaxassislarning roli benihoyat kattadir. Prezidentimiz ta’kidlaganidek, “Ertangi kun yangicha fikrlay oladigan zamonaviy bilimga ega bo’lgan yuksak malakali mutaxassislarni talab etadi”. Shu sababli xalqimizning boy intelektual merosi va umumbashariy qadriyatlari, zamonaviy madaniyat, iqtisodiyot, fan, texnika va texnologiyalar asosida yuksak mutaxassislar tayyorlash tizimi ishlab chiqildi va jadal sur’atlar bilan hayotga tadbiq etilmoqda.[2]

O’zbekistonda o’qitish texnologiyalarini zamonaviylashtirish, jadallashtirish rivojlangan iqtisodiyotli mamlakatlarga qaraganda yanada dolzarb ahamiyatga ega. Chunki hozirgi kunda milliy ta’lim tizimining salohiyatli tizimli rivojlanishi yanada yuqori pog’onaga ko’tarildi.

Mustaqillik yillarida axborot texnologiyalari sohasida juda katta o’zgarishlar yuz bermoqda, shu bilan bir qatorda bu yoshlarga keng imkoniyatlar yaratilmoqda. Ijtimoiy hayotning barcha sohalarida axborot texnologiyalari har yerda jadallik bilan kirib kelmoqda va bilimlarni oshishi jamiyat a’zolarini o’z ustlarida ko’proq ishlashga, yangi-yangi o’zgarishlar bilan

undashga olib kelmoqda. Texnikaviy bilim, murakkab texnologiyani egallash qobilyati ma’naviy barkamollik bilan birga borishi kerak.

Mavzuning dolzarbligi va o’rganilganlik darajasi. Bilamizki hozirgi kunda biz dasturlashda zamonaviy texnologiyalardan biri ADO texnoloiyasidan foydalanamiz. ADO.NET bu shunday sinflardan tashkil topganki, unda C# va .NET Frameworkda relyatsion jadvallar va ma’lumotlar bilan ishlashda qulaylik yaratadi. Relyatsion ma’lumotlar bazalari Microsoft SQL Server va Microsoft Access bilan bir qatorda boshqa relyatsion ma’lumotlar bazalari va hattoki relyatsion bo’lmagan ma’lumotlar manbalarni ham o’z ichiga oladi. ADO.NET texnologiyasi barcha .NET dan foydalanuvchi dasturlash tillarida .NET Framework va loyihalashni yaxlid holda birlashtirishda foydalaniladi, u C#ning o’ziga xos xususiyati hisoblanadi.

ADO.NET System.Data sinfi va uning sinf ostilari bo’lmish System.Data.SqlClient va System.Data.Linq bilan bir qarorda System.Xml sinfi va sinf ostilarining bir qismini o’z ichiga olgan sinflardan tashkil topgan.

ADO.NET ActiveX Data Objects (ADO) ning kengaytirilgan sinflar to'plami bo’lib, u ma’lumotdan Microsoftning oldingi avlod texnologiyasiga nisbatan tezroq foydalanish imkonini beradi.

ADO.NET- ADO ga nisbatan ma’lumotlardan tez foydalanuvchi sinflar va .NET dasturlash muhitlarini yangilash va kengaytirishda xizmat qiladi. Ushbu magistrlik dissertasiyasida yangi texnologiyalarni qo’llab C#dasturlash tilida kafedra o’quv yuklamasini va dars taqsimotini avtomatlashtiruvchi tizim yaratildi.

Tadqiqot ob’yekti. Oliy o’quv yurtlaridagi kafedralar va ularda tayyorlanadigan hujjatlar. O’quv rejalar, namunaviy dasturlar

Tadqiqot predmeti. Visual Studio muhiti, C#dasturlash tili, ADO.NET texnologiyasi, ADO ob’yektlari va sinflari. O’quv rejalar, namunaviy dasturlar va kafedra hujjatlari.

Tadqiqot maqsadi. Visual Studio muhitida C#dasturlash tili yordamida ADO.NET texnologiyasidan foydalanilgan holda kafedra o'quv yuklamasini va dars taqsimotini avtomatlashtiruvchi tizim yaratish.

Tadqiqot vazifalari. Tadqiqot maqsadidan kelib chiqqan holda quyidagi tadqiqot vazifalari belgilandi. ADO.NET texnologiyasi imkoniyatlarini to'liq o'rganish, Visual Studio muhitida C#dasturlash tilida amalga oshirish, kafedra o'quv yuklamasini va dars taqsimotini avtomatlashtiruvchi tizim yaratishda uning imkoniyatlarini ko'rsatish.

Tadqiqotning metodologik asosi. Tadqiqotning metodologik asosi sifatida Visual Studio muhiti, C#dasturlash tili, ADO.NET texnologiyasi, ADO ob'yektlari va sinflari asos qilib olindi.

Olingan asosiy natijalar. Visual Studio muhiti, C#dasturlash tili, ADO.NET texnologiyasi imkonyatlari to'liq o'rganildi. O'rganilgan bilimlar asosida oliy o'quv yurtlari uchun kafedra o'quv yuklamasini va dars taqsimotini avtomatlashtiruvchi tizim yaratildi.

Natijalarning ilmiy yangiligi va amaliy ahamiyati. Ushbu tizimni yaratishda yangi texnologiyalar qo'llanildi va ularning funksional imkoniyatlari to'liq ko'rsatib berildi. Bu juda katta amaliy ahamiyatga ega.

Tadbiq etish darajasi va iqtisodiy samaradorligi. Dissertasiya nazariy va amaliy xarakterga ega. Ushbu tizimdan barcha kafedralar kafedra o'quv yuklamasini va dars taqsimotini tayyorlashda foydalanishlari mumkin.

Ishning hajmi va tuzilishi. Ushbu magistrlik dissertasiyasi 80 betdan iborat bo'lib, kirish, 3 ta bob, xotima, foydalanilgan adabiyotlar ro'yxatidan iborat.

I BOB. Berilganlar bazasini boshqarish tizimlari.

1.1 Berilganlar bazasini boshqarish tizimlari haqida.

1.1.1 Ma`lumotlar bazasi va uni tashkil qilish tamoyillari.

Insonning kundalik mehnat faoliyati tashqi muhit to`g`risidagi axborotlarni qabul qilish va to`plash, turli Masalalarni yechish uchun zarur bo`lgan ma`lumotlarni aniqlash, qayta ishlash kabi amallarni bajarish bilan bog`liq bo`ladi. Shu sababli, ham yuqoridagi amallar majmuasi, ularni tatbiq etish usullarini vositalari axborot tizimlarini (AT) yaratish uchun asos bo`lib xizmat qiladi.

Axborot tizimlarining asosiy maqsadi foydalanuvchilarni tegishli sohaga taaluqli bo`lgan axborot bilan ta`minlashiga qaratilgan. EHMlarning yaratilishi natijasida avtomatlashtirilgan axborot tizimlarini (AAT) hosil qilish imkoniyatlari paydo bo`ldi.

Hozirgi kunda AATning rivojlanishi ikki yo`nalishda olib borilmoqda.

Birinchi yo`nalish – avtonom fayllar asosida axborot tizimlarini hosil qilish. Bunday ATning imkoniyat doiralari chegaralangan va oddiy tuzilishiga ega. Ular avtonom fayllar to`plamini qayta ishlash hamda hujjatlarni chiqarish amallarini bajaradigan dasturlar majmuasidan tashkil topadi. Bunday tizimlar quyidagi kamchiliklarga ega:

- ma`lumotlarning takrorlanishi;
- fayllarni yuritish murakkabligi;
- fayllar bilan birgalikda ishlash qiyinligi;
- dasturlarning ma`lumotlarga bog`liqligi va boshqalar.

Ikkinchi yo`nalish - ma`lumotlar bazasini hosil qilish. Ma`lumotlar bazasi asosida hosil qilingan AT foydalanuvchilar majmuasiga xizmat ko`rsatadi va yuqorida ko`rsatilgan tizimlar juda keng tarqalmoqda.

AATning faoliyati axborotlarni to`plash va qayta ishlash bilan bog`liq. Tizimga kiritilayotgan va foydalanuvchiga berilayotgan axborotlar hujjatlar ko`rinishda shakllanadi. Shu sababali ham hujjat moddiy ob`ekt hisoblanadi va

ma`lum bir tartib asosida chizmaviylashtirilgan axborotlar to`plamidan iborat bo`ladi.

AATda axborot manbai sifatida odamlar va texnik vositalar hisoblansa, iste`molchi sifatida turli foydalanuvchilarni uch guruhga ajratish mumkin: tizimning ma`muriyati, dasturchilar va oxirgi iste`molchilar.

Foydalanuvchilarning AAT ga murojaati talab asosida amalga oshiriladi. Talab-mavsumlashtirilgan xabar bo`lib, unda tegishli ma`lumotlarni qidirish shartlari va ular ustidan bajarilishi lozim bo`lgan vazifalar ko`rsatiladi.

Talablarni qabul qilish va kiritish, ko`rsatilgan amallarni bajarish, tegishli ma`lumotlarni tayyorlash va hujjat ko`rinishda foydalanuvchiga taqdim qilish har qanday AAT ish faoliyatining asosiy bosqichlari hisoblanadi.

Hozirgi kunda AATlar inson faoliyatining turli sohalarida, Masalan, xalq xo`jaligi tarmoqlarini boshqarishda, ilmiy-tadqiqot ishlarini boshqarishda, ma`rif sohasida loyihalashtirishda qo`llanilmoqda. Bunda quyidagi ikki usulning biridan foydalaniladi:

AATdan avtonom foydalanish. Bunda AAT boshqa tizim tarkibiga kirmaydi, balki mustaqil faoliyat ko`rsatadi. Bunga, Masalan, tayyora va temir yo`l chiptalarini sotish tizimlari («Sirena», «Ekspress»), talab bo`yicha tegishli hujjatlarni tayyorlovchi axborot - qidirish tizimlari va boshqalar misol bo`ladi.

AAT dan yuqori darajali boshqarish tizimining tarkibiy qismi sifatida foydalanish. Bunda hosil qilingan chiquvchi ma`lumotlardan tizimning boshqa elementlari faoliyatida ham qo`llaniladi. Bunday AATga, Masalan, axborot - o`qitish tizimlari, loyihalashtirishning avtomatlashtirilgan tizimlari, avtomatlashtirilgan boshqarish tizimlari misol bo`ladi.

Hujjatli axborot qidirish tizimi (XAQT) hujjatlashtirilgan ma`lumotlarni saqlash va qayta ishlashni amalga oshiradi. Kutubxona faoliyatining avtomatlashtirilgan tizimi XAQT ga misol bo`ladi.

Faktografik axborot qidirish tizimi (FAQT) raqmli va mantli ma`lumotlarni saqlashda va qayta ishlashda qo`llaniladi. Tashkil qilinayotgan AATning asosiy qismi FAQT turidagi tizimga misol bo`ladi.

Ma`lumotlarni ishlash usuliga ko`ra AAT ikki qismga: axborot - ma`lumotnoma tizimi (AMT) va ma`lumotlarni ishlashning avtomatlashtirilgan tizimi (MIAT)ga bo`linadi.

AMT talab-javob tartibida ishlaydi. Bunday tizimda tegishli axborotlar talab bo`yicha qidiriladi va foydalanuvchiga qayta ishlamagan holda beriladi. Ikkinchi turdagi tizimda esa topilgan ma`lumotlar tegishli dasturlar yordamida ishlanadi va foydalanuvchiga beriladi.

Ma`lumotlarni integratsiyalashtirish darajasiga ko`ra AAT avtonom va ma`lumotlar bazasidan tashkil topgan turlarga bo`linadi. Avtonom fayli tizimlarda (AFAAT) to`plangan ma`lumotlar o`zaro bog`lanmagan holatda bo`ladi. Shu sababli bunday turdagi tizimlar o`rniga ma`lumotlar bazasidan (MB) foydalanilmoqda.

Taqsimlash darajasiga ko`ra AAT elementlari bitta EXMda (lokal) va hisoblash tarmog`ida (taqsimlangan) joylashgan turdagi tizimlarga bo`linadi.

Ma`lumotlar bazasini tashkil qilish tamoyillari. Axborotga bo`lgan talablarning turli-tumanligi, Masalalar ko`lamining tobora ortib borishi va boshqalar zamonaviy ATlari oldiga bir qator talablar qo`ymoqda. Bunday talablar jumlasiga quyidagilar kiradi:

Ma`lumotlarning aniqligi. Ma`lumki, ma`lumotlar bazasi tegishli sohaning axborot modelini tashkil qiladi. Shu sababli ham MB da saqlanayotgan axborotlar ob`ektlarning holati, xususiyati va ular o`rtasida aloqalarni to`liq va aniq ifodalash lozim. Aks holda tashkil qilingan MB xatarli bo`lishi va zarar keltirishi mumkin.

Tezkorlik va unumdorlik. Tizimning tezkorligi qo`yilgan talabga javob berish vaqti bilan aniqlanadi. Bunda nafaqat EHM ning tezkorligini, balki ma`lumotlarning joylanishi, izlash usullari, talabning qiyinligini va boshqa omillarni ham hisobga olish zarur. Tizimning unumdorligi esa vaqt birligi ichida bajarilgan talablarning miqdori orqali aniqlanadi.

MBdan foydalanishning odiyligi va qulayligi. Bu talab tizimdan foydalanuvchi barcha is`temolchilar tomonidan qo`yiladi. Shu sababli ham MB

dan foydalanishning oson, sodda va qulay usullarini yaratish muhim ahamiyatga ega.

Ma`lumotlarni himoyalash. Tizim ma`lumotlar bazasida saqlanilayotgan axborot va dasturlarni tashqi ta`sirlardan, begona foydalanuvchilardan himoyalashni ta`minlashi lozim.

Tizimning rivojlanishi. Tizim tarkibi doimo yangi elementlar, dasturlar bilan taxminlanishi, axborot massivlari o`zgartirilishi va yangilanib borishi zarur.

Yuqorida keltirilgan talablarga javob beradigan MB quydagi tamoyillarga asoslangan holda tashkil qilinishi mumkin:

Ma`lumotlarning integratsiyalashtirish tamoyili. Bu tamoyilning mohiyatiga ko`ra o`zaro bog`lanmagan axborotlar yagona ma`lumotlar bazasiga birlashtiriladi. Buning natijasida ma`lumotlar foylalanuvchi va uning amaliy dasturlariga axborot massivlari ko`rinishida taqdim etiladi. Axborat massivlaridan foydalanilganda kerakli ma`lumotlarni qidirish, qayta ishlash jarayonlarini boshqarish osonlashadi, ma`lumotlarning ortiqchaligi kamayadi, MBni yuritish yengillashadi.

Ma`lumotlarning yaxlitligi tamoyili. Bu tamoyil orqali MBda saqlanayotgan axborotlarning aniqligi ortadi, ya`ni ularning xususiyatlari va tavsifnomalari tegishli soha ob`ektlari to`liq ifodalaniladi. Ma`lumotlarning yaxlitligi noto`g`ri axborotni kiritish yoki uning ma`lum bir qismini xotiradan o`chirib tashlash natijasida buzilishi mumkin. Shuning uchun ham kiritilayotgan axborotlarni nazorat qilish, saqlanayotgan ma`lumotlarni doimo tekshirish, maxsus tizim yordamida tiklash va boshqa tadbirlar orqali MBning yaxlitligini taxminlash mumkin.

Ma`lumotlarning aloqadorligi tamoiyli. Bu tamoyilning mohiyatiga ko`ra MBdagi barcha axborotlar o`zaro bog`langan bo`lib, ob`ektlar o`rtasidagi munosabatlarni ifodalaydi. Axborot turlari va ular o`rtasidagi munosabatlar majmuasi ma`lumotlarning mantiqiy tuzilishini tashkil qiladi. Buning natijasida ish yengillashadi va tezlashadi.

Ma`lumotlarning yetarli bo`lish tamoyili. Bu tamoyilning mohiyatiga ko`ra, tegishli axborotlar MBda yagona nusxa saqlanadi va ular istalgan Masalani yechish uchun o`zaro bog`lanadi hamda yetarli bo`ladi. Masalan, avtonom fayllardan iborat bo`lgan AATda ba`zi bir axborotlar takrorlansa, MB da esa ularning takrorlanishi butunlay barham topadi.

MBni boshqarishini markazlashtirish tamoyili. Bu tamoyilga ko`ra ma`lumotlarni boshqarishning barcha funktsiyalari yagona boshqarish dasturi-ma`lumotlar bazasini boshqarish tizimi (MBBT)ga beriladi. Bu tamoyilga rioya qilish asosida ATdan foydalanishning samaradorligi barcha jarayonlar MBBT orqali amalga oshiriladi.

Ma`lumotlarning ifodalanishini ularni qayta ishlash jarayonlaridan ajratish tamoyili. Bu tamoyilga ko`ra, ma`lumotlarning ifodalanishi amaliy dasturlardan tashqarida tayyorlanadi va MBda saqlanadi. Bu esa o`z navbatida dasturlash jarayonini yengillashtiradi, dastur uchun zarur bo`lgan holda axborotlarning hajmini kamaytiradi. MBni yuritishni yaxshilaydi va x.k.

Shunday qilib, yuqorida ko`rib o`tilgan tamoyillar asosida MBning tarkibi yaratildi, ya`ni ATning mantiqiy, fizik va dasturiy elementlari o`rtasidagi o`zaro bog`lanish ishlab chiqiladi.

Ma`lumotlar bazasining tarkibi va uni tashkil etish. ATning tarkibiy elementlari unga yuklatilgan vazifalar va yechiladigan Masalalarning xususiyati orqali aniqlanadi. Shunga ko`ra ma`lumotlar bazasining asosiy vazifalari quyidagilardan iborat:

-axborotlarni saqlash va himoyalash;

-axborotlarni doimo o`zgartirish (yangilash, yangi ma`lumotlarni kiritish, ortiqcha ma`lumotlarni o`chirish va x.k.)

-foydalanuvchi va amaliy dasturlar talablariga ko`ra ma`lumotlarni izlash va tanlash;

-aniqlangan ma`lumotlarni qayta ishlash va tegishli usulda natijaviy axborotlarni chiqarish va boshqalar.

Yuqorida ko`rsatilgandek, axborotlar ma`lumotlar bazasida saqlanadi. MB - amaliy dasturlarga bog`liq bo`lmagan holda ma`lum bir tartib asosida o`zaro bog`langan ma`lumotlar to`plamidir.

Har qanday ma`lumot fayl kabi, MB ham yozuvlardan tashkil topadi. Yozuvlar esa o`z navbatida maydonchalardan hosil qilinadi. Yozuv tezkor va tashqi xotiralar o`rtasida ma`lumotlar almashish jarayonning eng kichik o`lchov birligi bo`lsa, maydoncha - ma`lumotlarni qayta ishlashdagi eng kichik birlik hisoblanadi.

MBni tashkil qilish oddiy fayllarni tashkil qilishdan quyidagi ikkita xususiyatiga ko`ra farqlanadi:

-yozuv maydonlarining ifodalanishi ma`lumotlar bilan birgalikda saqlanadi;

-ma`lumotlarni qidirishda maxsus usullardan foydalaniladi.

Operatsion tizimning muhitida faoliyat ko`rsatayotgan MB bilan turli amallarni bajarish mumkin emas. Shu sababli ham operatsion tizim asosida ishlaydigan maxsus amaliy dasturlar majmuasi yaratilgan. Bu majmua ma`lumotlar bazasini boshqarish tizimi deb yuritiladi. MBBT - ma`lumotlar bazasini hosil qilish, uni yuritish va foydalanish uchun mo`ljallangan dasturlar va til vositalarining to`plami.

MBBTning asosiy qismini boshqarish dasturi tashkil qiladi. Bu dastur MB bilan muloqatni o`rnatishga bog`liq bo`lgan barcha jarayonlarni avtomatlashtiradi. MBBT ishga tushishi bilan uning boshqarish dasturi doimo asosiy xotirada bo`ladi va talablarni qayta ishlashni tashkil qiladi, ularning bajarilish tartibini ta`minlaydi, amaliy dasturlar va operatsion tizim o`rtasidagi aloqalarni o`rnatadi. MB dan tegishli amallarni bajarish jarayonlarini nazorat qiladi va boshqalar. MBga kelayotgan talablarni parallel bajarishni tashkil qilish boshqarish dasturining asosiy funktsiyasi hisoblanadi.

MBBTning boshqa qismini ma`lumotlarni qayta ishlash dasturlarining to`plami tashkil qiladi. Bu to`plamga tarjimonlar (translyatorlar), talab va dasturlash tillari, muharrirlar, servis dasturlari va boshqalar kiradi.

Shunday qilib, ma`lumotlar banki bir necha ma`lumotlar bazasi, boshqarish va amaliy dasturlardan tashkil topadi. Bu elementar AT ga yuklatilgan vazifalarni bajarishda asosiy rol o`ynaydi. Shu bilan birga, ATning samarali faoliyati uning ta`minlovchi elementlariga ham bog`liqdir. Bu ta`minot tarkibiga quyidagi elementlar kiradi.

Texnik ta`minot MB va foydalanuvchilarning ish faoliyatini avtomatlashtirish imkoniyatini yaratadigan texnik vositalardan tashkil topadi. Bunday vositalar jumlasiga EHM, tashqi qurilmalar, axborotni tashish, uzatish vositalari, aloqa tarmoqlari, abonent punktlari va boshqalar kiradi.

Matematik ta`minot - funktsional Masalalarni yechish va MBni boshqarish usullari, matematik modellar va algoritmlar to`plamidan tashkil topadi.

Dasturiy ta`minot-MBning faoliyatini amalga oshirish dasturlari va turli xil qo`shimcha vazifalarni bajarish uchun mo`ljallangan servis dasturlarning to`plamidan iborat bo`ladi.

Axborot ta`minoti-ma`lumotlarni turkumlash va ixchamlashtirish, ifodalash va taqdim etish tizimlaridan tashkil topadi.

Lingvistik ta`minot - MBBTda foydalaniladigan tillar, lug`atlar majmuasi orqali tashkil qilinadi. Tashkiliy ta`minot -MBning kundalik faoliyatini ifodalovchi chizmaiy hujjatlar, me`yoriy ko`rsatmalar to`plamidan iborat bo`ladi.

1.1.2 Ma`lumotlar bazasini boshqarish tizimlari.

Ma`lumotlar bazasini loyihalashtirish jarayoni ikki bosqichga bo`linadi: MB mantiqiy tuzilishini tashkil qilish va tashuvchilarda MBni hosil qilish.

MBning mantiqiy tuzilishi - ob`ektga tegishli bo`lgan axborotlarning MBda joylanishini ifodalaydi. Hosil bo`lgan MBning mantiqiy bog`lanish modeli birinchi bosqichning natijasi hisoblanadi. Bu modelda uch turdagi axborot ifodalanadi: ob`ekt to`g`risidagi xabarlar, ularning xususiyati va o`zaro munosabatlari. Har bir ob`ekt modelda yozuv turlari orqali ko`rsatiladi. Ularning xususiyatlari yozuv maydonlari orqali ifodalanadi, munosabatlar esa - yozuv va

maydon turlari o`rtasidagi aloqalar yordamida tasvirlanadi. Bunday model EHM, operatsion tizim, MBBTning mohiyatiga bog`liq bo`lmaydi, ya`ni axborotning ma`nosiga bog`liq bo`lmagan holda ularni ifodalash usuli va aloqasini ta`minlaydi.

Mantiqiy modelni chizmali va jadvalli usullar yordamida ifodalash mumkin. Chizmali usulda ma`lumotlar o`rtasidagi bog`lanish graflar yordamida tasvirlanadi. Bunda grafning uchlari yozuvlarni ifodalaydi. Graflarning qirralari yozuvlar o`rtasidagi aloqalarni ko`rsatadi. Jadvalli usulda esa ob`ekt to`g`risidagi ma`lumotlar bir yoki bir necha ustundan iborat bo`lgan jadvallar orqali ifodalanadi.

Hozirgi kunda mantiqiy modellarning pog`onali, tarmoqli va relyatsion turlaridan foydalanilmoqda. Shaxsiy EHMlarning paydo bo`lishi relyatsion modellarning keng tarqalishiga sababchi bo`ldi.

Pog`onali model chizmali usul asosida tashkil qilinadi. Bunda ma`lumot yozuvlari grafikning uchlarini ifodalaydi va har bir yozuv oldingi pog`ona uchlariga bog`langan bo`ladi. Bunday tuzilishdagi MBdan tegishli axborotlar hamma vaqt bitta yo`nalish bo`yicha qidiriladi va uning joylashgan o`rni to`liq ko`rsatiladi. Masalan, «Talaba» to`g`risidagi ma`lumotlarni olish uchun «Fakultet», «Kurs», «Guruh» yozuvlari ko`rsatilishi lozim.

Tarmoqli model ham chizmali usul yordamida tashkil qilinadi. Lekin, bunda tegishli axborotlar bir necha yo`nalish bo`yicha olinishi mumkin. Masalan, «Talaba» to`g`risidagi ma`lumotlarni olish uchun yuqoridagi tasvirga «Muallim-fan» va «Fan-talaba» tarmoqli modeli hosil bo`ladi

Relyatsion model jadvalli usul asosida tashkil qilinadi. Bunda tegishli ma`lumotlar jadvalning ustun va qatorlarida joylashadi. Ustunlar ma`lumotning maydonlarini, qarorlar esa yozuvlarni ifodalaydi. Bir ustunda ma`lum sohaga tegishli bo`lgan bir qancha ma`lumotlar joylashadi. Qatorda esa ustunlarda joylashgan ma`lumotlar ko`rsatiladi. Ustun va qator o`rtasidagi bog`lanish munosabat deb ataladi. Har bir ustun, qator va munosabat o`z nomiga ega bo`ladi.

Relyatsion modeldagi munosabatlar quyidagi talablar orqali hosil qilinadi:

-ustun va qator kesishgan erda joylashgan ma`lumotlar elemenlar hisoblanadi;

-munosabatlarda ikkita bir xil qator bo`lmaydi;

-ustun va qatorlarning tartibli joylashishi va nomlanishi majburiy emas.

MBni tashuvchilarda hosil qilish bosqichi fizik tuzilishni tashkil etadi. Fizik tuzilish tashqi xotiralarda ma`lumotlarni joylashtirish usullari va vositalaridan iborat bo`lib, uni natijasida ichki model hosil qilinadi.

Ichki model ma`lumotning mantiqiy modelini tashuvchilarida aks ettiradi va yozuvlarning joylanishi, aloqasi va tanlab olinishini ko`rsatadi. Ichki model MBBT orqali hosil qilinadi:

-ma`lumotlarning mantiqiy tuzilishini saqlash;

-tashqi xotiradan maksimal foydalanish;

-MBni yuritish xarajatlarini kamaytirish;

-ma`lumotni qidirish va tanlash jarayonlarining tezkorligini oshirish va boshqalar.

Ma`lumki, bir algoritm bo`yicha turli tillar yordamida ekvivalent dasturlarini yaratish mumkin. Shunga bog`liq holda bitta mantiqiy model orqali bir qancha kichik (fizik) modellarni yaratish mumkin. Lekin yaratilgan modellardan biri optimal bo`ladi. Shu sababali, MBni hosil qiluvchi mutaxassislar oldida ichki modelning optimal variantini topish Masalasi turibdi. Bunda optimallik mezoni sifatida yuqoridagi talablarni olish mumkin.

Saqlanayotgan ma`lumotlarning tuzilishi, ularni qidirish usullari va ifodalanish tillari fizik modellashtirishning asosiy vositalari hisoblanadi.

Ma`lumotlarning tuzilishini fayl yozuvlar ko`rinishida tasvirlash mumkin. Bunday holda yozuvlar maydonlardan, ularning joylanish tartibidan, turi va uzunligidan iborat bo`ladi. Ma`lumotlarni qidirish vaqtini kamaytirish maqsadida turli qidirish usullari yaratilmoqda. Agar ma`lumotlarning tuzilishi yozuvlarning tezroq topish yo`lini ko`rsatadi. Shuning uchun ham, MB fizik tashkil qilishda ikkita tamoyilga: ma`lumotlarning tuzilishi va qidirish usullari

asosida MBni hosil qilishga rioya qilinadi. Har qanday MB fizik tashkil qilish natijasida fayllar hosil qiladi. Shaxsiy kompyuterlarda bu fayllar ketma-ket yoki ixtiyoriy tartibda joylanishi mumkin. Bunday fayllarni bajarishda, ya`ni MBBT da chiziqli va zanjirli ro`yxat, tartiblashmagan va tartiblashgan qidirish usullaridan foydalaniladi.

Chiziqli ro`yxat-MBni fizik tashkil qilishning eng oddiy usuli hisoblanadi. Bunda MBning fayllari bog`lanmagan holda bo`ladi va tegishli yozuvlarni qidirish ma`lum bir algoritmlar asosida amalga oshiriladi. Chiziqli usul orqali xotiradan samarali foydalanish mumkin, lekin ma`lumotlarni qidirish uchun boshqa usullarga qaraganda ko`p vaqt sarf qiladi.

Zanjirli ro`yxat usulida hosil qilingan faylda har bir yozuv boshqa yozuv bilan bog`langan bo`ladi. Bunda aloqa vositasi sifatida ko`rsatkichlar ro`yxatidan foydalaniladi. Ko`rsatkichlar ro`yxati yozuvning qo`shimcha maydonlarida ko`rsatiladi va ular orqali kerakli ma`lumotlarni olish tartibi o`rnatiladi.

Ro`yxatga kirish uchun ro`yxatning boshlang`ich manzilgohini (RBM) ko`rsatish lozim. Bu manzilgoh ro`yxat sarlavhasida (RS) saqlanadi.

Ma`lumki, MBning yozuvlari asosiy maydon bo`yicha tartiblashgan bo`ladi. Lekin yozuvlarni asosiy maydon bo`lmagan ustunlar orqali tartiblashgan holda izlash mumkin. Buning uchun tartiblashmagan fayllar hosil qilinadi

Tartiblashmagan fayllar kerakli ma`lumotlarni tez qidirish imkoniyatini bersada, ularda saqlanayotgan ma`lumotlar bir necha marta takrorlanadi. Natijada xotiradan foydalanish samaradorligi kamayadi. Bu kamchilikni tugatish maqsadida fayllar tartiblashgan holga keltiriladi. Bunday holatlarda yozuvlar emas, balki ularning joylashgan manzilgohlari saqlanadi. Kerakli ma`lumotlar manzilgohlar bo`yicha qilinadi va u xotirada kam joyni egallaydi.

Bunday fayl MBBT orqali avtomatik tarzda hosil qilinadi. Tegishli ma`lumotlar manzilgoh indekslarini izlash orqali chiqariladi.

MBBTning asosiy vazifalari va xususiyatlari. Ma`lumki, MBBT dasturiy va til vositalarining to`plamidan iborat bo`lib, ular yordamida MBni hosil qilish, yuritish, tahrirlash va boshqa vazifalarni bajarish mumkin. Bunday tizim yordamida operatsiya tizimining ma`lumotlarini boshqarish bo`yicha imkoniyatlari kengayadi.

MBBTning vazifalarini uch guruhga ajratish mumkin:

-fayllarni boshqarish; ya`ni faylni ochish, nusxa olish, nomini o`zgartirish tuzilishini o`zgartirish, qayta hosil qilish, tiklash, hisobot olish, bekitish va boshqalar;

-yozuvlarni boshqarish, ya`ni yozuvlarni o`qish, kiritish, tartiblashtirish, o`chirish va boshqalar;

-yozuv maydonlarini boshqarish.

Shuni ta`kidlash lozimki, ma`lumotlarni xarflar dastasi yordamida kiritish, hisoblash, takroriy jarayonlarini amalga oshirish, ma`lumotlarni ko`rsatuv oynasi yoki bosmaga chiqarish MBBTning vazifalari qatoriga kirmaydi. Bu vazifalar amaliy dasturlar yordamida bajariladi. Bunday dasturlar MBBTning maxsus dasturlash tillari orqali hosil qilinadi.

Yuqorida keltirilgan vazifalar to`plami MBBT da uch turdagi dasturlarning bo`lishini talab qiladi: boshqaruvchi dastur, qayta ishlovchi (translyator) dastur va xizmat ko`rsatuvchi dastur. MBBT ishga tushishi bilan asosiy boshqaruvchi dastur xotirasiga yuklanadi. Boshqa dasturlar tegishli holda ishga tushiriladi.

MBBTni turkumlashda mantiqiy tuzilish asos qilib olingan. Shuning uchun ham tarmoqli, pog`onali va relyatsion MBBTlari mavjud. Relyatsion MBBTlari keng tarqalgan bo`lib, ular jumlasiga dBase III Plus, FoxBase, Fox Pro, Clipper, dBase IV, Paradox va boshqalar kiradi.

MBBT ikki tartibda: *interpretator* va *kompilyator* tartibda ishlashi mumkin.

Interpritator tartibda dasturlarning buyruqlari bosqichma-bosqich, birin-ketin bajariladi. Unda har bir buyruq nazorat qilinadi, so`ngra mashina tiliga

aylantirib, bajariladi. Tegishli amallar bajarilgandan keyin, ular xotiradan o`chiriladi, tizim qayta ishlash bosqichiga o`tadi va keyingi buyruqni bajarishga kirishadi, interpritator tartibida «Exe» kengaytirmali fayl hosil qilinmaydi. Bunday faylini hosil qilish uchun kapilyator tartibida foydalaniladi.

Kompilyator tartibida buyruqlar bevosita bajarilmaydi, balki ular «exe» faylga yoziladi. Exe faylni hosil qilish jarayoni ikki bosqichdan iborat bo`ladi: boshlang`ich dasturni nazorat qilish va uni obj turga aylantirish; matn muharriri yordamida dasturni exe faylga aylantirish. Exe faylning bajarilishi uchun MBBTning mavjud bo`lishi shart emas, Interpretator tartibida ishlaydigan MBBTga dBase III Plus, FoxBase va Karat kiradi, kompilyator tartibida Clipper, panel tartibida esa Clario ishlaydi.

MBBT foydalanuvchi bilan ma`lumotlar bazasi o`rtasidagi aloqani ta`minlovchi dastur sifatida ishtirok etadi. Uning funktsiyalari menyu va dasturlar ko`rinishida namoyon bo`ladi.

Menyu tartibida MBBTning funktsiyalari ekranda tasvirlanadi. Foydalanuvchi kursorni xarakatlantirish orqali tegishli funktsiyani aniqlashi va bajarishga chaqirishi lozim. Tizim aniqlangan funktsiyalarni bajarib bo`lgandan so`ng yana menyu holatiga qaytadi.

Dasturiy tartibda tegishli buyruqlar kiritiladi, dasturlar qayta ishlanadi va bajarishga chaqiriladi. Bu holda MBBT interpritator tartibida ishlaydi va foydalanuvchidan dasturlash tillarini bilish talab qilinadi.

MBBTda foydalaniladigan dasturlash tillariga umumiy talablar bilan bir qatorda quyidagilar ham qo`yiladi:

- tilning to`liq bo`lishi;
- vazifalarni bajarish uchun tegishli vositalarning bo`lishi;
- aniqlangan ma`lumotlarni to`liq qayta ishlash va boshqalar.

Dasturlash tillari bir qator belgilarga ko`ra turkumlarga ajratiladi.

-o`zgaruvchanlik;

-jarayonlilik;

-foydalanilayotgan matematik apparat va boshqalar.

MBBT dagi dasturlar tegishli bo`yruqlarning to`plamidan tashkil topadi. Echilayotgan Masala, larning qiyinligiga qarab, dasturlar oddiy yoki murakkab tuzilishiga ega bo`ladi. Oddiy tuzilishga ega bo`lgan dasturlarda buyruqlar ketma-ket joylashadi. Murakkab tuzilishli dasturlarda esa buyruqlar modullar holatida, ya`ni asosiy modul va quyi dasturlar to`plamidan iborat bo`ladi. Ma`lumotlar bazasini hosil qilishda modullik tamoyilidan foydalanish qulay va samaralidir

1.2 SQL tili.

1.2.1 SQL tili va ma'lumotlar tiplari.

SQL tilida quyidagi asosiy ma'lumotlar tiplari ishlatilib, ularning formatlari har xil MBBT lar uchun farq qilishi mumkin (1.1 jadval).

INTEGER	- butun son (odatda 10 tagacha qiymatli raqam va ishora).
SMALLINT	- "qisqa butun" (odatda 5 tagacha qiymatli raqam va ishora).
DECIMAL(p,q)	- o`nli son, p raqam va ishoradan iborat ($0 < p < 16$). O`nli nuqtadan so`ng raqamlar soni q orqali beriladi ($q < p$, agar $q = 0$ bo`lsa, tashlab yuborilishi mumkin).
FLOAT	- haqiqiy son 15 ta qiymatli raqam va butun darajadan iborat. Daraja MBBT tipi bilan aniqlanadi (masalan, 75 yoki 307).
CHAR(n)	- uzunligi o`zgarmas, n ga teng bo`lgan simvolli qator ($0 < n < 256$).
VARCHAR(n)	- uzunligi o`zgaruvchi, n simvoldan oshmagan simvolli qator ($n > 0$ va har xil MBBTlarda har xil lekin 4096 dan kam emas).
DATE	- maxsus komanda orqali aniqlanuvchi formatdagi sana (Subaseda ko`zda tutilgan bo`yicha yy/mm/dd); sana maydonlari bizning eramizdan oldin bir necha mingyilliklardan boshlanuvchi va bizning eramiz beshinchi-o`ninchi mingyilligi bilan cheklangan haqiqiy sanalarni o`z ichiga olishi mumkin.

TIME	-maxsus komanda orqali aniqlanuvchi formatdagi vaqt (ko`zda tutilgan bo`yicha hh.mm.ss).
DATETIME	- sana va vaqt kombinatsiyasi. (Sybaseda TIMESTAMP).
MONEY	-maxsus komanda orqali aniqlanuvchi formatdagi pul. Format o`z ichiga pul birligi simvoli ($, rub, ...) va uning joylashuvi (suffiks yoki prefiks), kasr qism aniqligi va pul qiymatini ko`rsatish shartlarini oladi.

1.1-jadval. SQL tilida tiplar.

1.2.2 Jadvallar bilan ishlash.

Jadvallarni yaratish.

Jadvallar CREATE TABLE komandasi bilan yaratiladi. Bu komanda qatorlarsiz bo`sh jadval yaratadi. CREATE TABLE komandasi jadval nomini va jadval o`zini ma'lum tartibda ko`rsatilgan ustunlar nomlari ketma - ketligi ta'rifi ko`rinishida aniqlaydi. U ma'lumotlar tiplari va ustunlar o`lchovini aniqlaydi. Har bir jadval juda bo`lmaganda bitta ustunga ega bo`lishi kerak.

CREATE TABLE komandasi sintaksisi:

CREATE TABLE <table-name >

(<column name> <data type>[(<size>)],

<column name> <data type>[(<size>)], ...);

Argument qiymati kattaligi ma'lumot turiga bog`liqdir. Agar siz maxsus ko`rsatmasangiz, tizim avtomatik qiymatni o`rnatadi.

Jadvallarni o`chirish.

Jadvalni o`chirish imkoniga ega bo`lish uchun, jadval egasi (Ya'ni yaratuvchisi) bo`lishingiz kerak. Faqat bo`sh jadvalni o`chirish mumkin. Qatorlarga ega bo`lgan, to`ldirilgan jadvalni o`chirish mumkin emas, Ya'ni jadval o`chirishdan oldin tozalangan bo`lishi kerak. Jadvalni o`chirish komandasi quyidagi ko`rinishga ega:

DROP TABLE < table name >;

Jadvalni yaratilgandan so`ng o`zgartirish.

Jadvalni o`zgartirish uchun ALTER TABLE komandasidan foydalaniladi. Bu komanda jadvalga Yangi ustunlar qo`shish, ustunlarni o`chirish, ustunlar kattaligini o`zgartirish,hamda cheklanishlarni qo`shish va olib tashlash imkoniyatlariga ega. Bu komanda ANSI standarti qismi emas, shuning uchun har xil tizimlarda har xil imkoniyatlarga ega.

Jadvalga ustun qo`shish uchun komandaning tipik sintaksisi:

ALTER TABLE <table name> ADD <column name>
<data type> <size>;

1.2.3 Jadvallar uchun cheklanishlar.

Cheklanishlarni kiritish.

Jadval yaratayotganingizda (yoki uni o`zgartirayotganingizda), siz maydonlarga kiritilayotgan qiymatlarga cheklanishlar o`rnatishingiz mumkin. Bu xolda SQL cheklanishlarga to`g`ri kelmaydigan hamma qiymatlarni rad etadi. Cheklanishlar ikki asosiy turi mavjud: - ustun va jadval cheklanishlari. Ularning farqi shundaki ustun cheklanishi faqat ayrim ustunlarga qo`llanadi, jadval cheklanishi bo`lsa bir yoki bir necha ustunlar guruxiga qo`llanadi. Ustun cheklanishi ustun nomi oxiriga ma'lumotlar tipidan so`ng va verguldan oldin qo`yiladi. Jadval cheklanishi jadval nomi oxiriga so`nggi dumaloq verguldan oldin qo`yiladi.

Cheklanishlar hisobga olingan CREATE TABLE komandasi sintaksisi:

CREATE TABLE < table name >
(<column name> <data type> <column constraint>,
<column name> <data type> <column constraint> ...
<table constraint> (<column name>
[, <column name>])...);

Maydonga bo`sh (NULL) qiymatlar kiritilishi oldini olish uchun CREATE TABLE komandasida NOT NULL cheklanishi ishlatiladi. Bu cheklanish faqat har xil ustunlar uchun o`rnatiladi.Ko`p xollarda ustunga kiritilgan qiymatlar bir biridan farq qilishi kerak. Agar ustun uchun UNIQUE

cheklanishi o`rnatilsa, bu ustungsha mavjud qiymatni kiritishga urinish rad etilapdi. Bu cheklanish bo`sh bo`lmaydigan (NOT NULL) deb e'lon qilingan maydonlarga qo`llanishi mumkin.[11]

Unikalligi talab qilinadigan maydonlar(birlamchi kalitlardan tashqari) kandidat kalitlar yoki unikal kalitlar deyiladi.

Birlamchi kalitlar cheklanishlari.

SQL birlamchi kalitlarni to`g`ridan to`g`ri birlamchi kalit (PRIMARY KEY) cheklanishi orqali ta'riflaydi. PRIMARY KEY jadvalni yoki ustunlarni cheklashi mumkin. Bu cheklanish UNIQUE cheklanishi kabi ishlaydi, faqat jadval uchun faqat bitta birlamchi kalit (ixtiyoriy sondagi ustunlar uchun) aniqlanishi mumkin bo`lgan xoldan tashqari. Birlamchi kalitlar NULL qiymatga ega bo`lishi mumkin emas.

Maydon qiymatlarini tekshirish (CHECK cheklanishi).

CHECK cheklanishi jadvalga kiritilayotgan ma'lumot qabul qilinishidan oldin mos kelishi lozim bo`lgan shart kiritishga imkon beradi. CHECK cheklanishi CHECK kalit so`zi ko`rsatilgan maydondan foydalanuvchi predikat ifodapdan iboratdir.

Ko`zda tutilgan qiymatlarni o`rnatish.

Biror bir maydon uchun qiymat ko`rsatmagan xolda jadvalga satr qo`shsangiz, SQL bunday maydonga kiritish uchun ko`zda tutilgan qiymatga ega bo`lishi kerak, aks xolda komanda rad etiladi. Eng umumiy ko`zda tutilgan qiymat NULL qiymatdir. CREATE TABLE komandasida ko`zda tutilgan qiymat DEFAULT operatori orqali, ustun cheklanishi sifatida ko`rsatiladi.

Ma'lumotlar yaxlitligini ta'minlash.

Jadval bir maydonidagi hamma qiymatlar boshqa jadval maydonida aks etsa, birinchi maydon ikkinchisiga ilova qiladi deyiladi. Bu ikki maydon orasidagi bog`liqlikni ko`rsatadi

Tashqi kalit bitta maydondan iborat bo`lishi shart emas. Birlamchi kalit kabi, tashqi kalit bitta modul sifatida qayta ishlanuvchi bir necha maydonlarga ega bo`lishi mumkin. Maydon tapshqi kalit bo`lsa ilova qitlayotgan jadval bilan

ma'lum usulda bog`liqdir. Tashqi kalit har bir qiymati (satri), ajdod kalitning bitta va faqat bitta qiymatiga(satriga) ilova qilishi kerak. Bu xolda tizim ilovali yaxlit xolatda deyiladi.

Cheklanish FOREIGN KEY.

SQL ilovali yaxlitlikni FOREIGN KEY yordamida ta'minlaydi. Tashqi kalit vazifasi ajdod kalitda ko`rsatilmagan qiymatlarni tashqi kalit maydonlariga kiritmaslikdir. FOREIGN KEY cheklanishi sintaksisi:

FOREIGN KEY <column list> REFERENCES

<fktable> [<column list>];

Birinchi ro`yxat komanda tomonidan o`zgartiriluvchi ustunlar ro`yxatidir. fktable - bu ajdod kalitli jadval. Ikkinchi ustunlar ro`yxati bu ajdod kalitni tashkil qiluvchi ustunlardir.

Kalitlarga cheklanish.

Ilovali yaxlitlikni ta'minlash tashqi kalit yoki ajdod kalit maydonlari qiymatlariga cheklanishlar o`rnatishni talab qiladi. Ajdod kalit tarkiblangan bo`lib, tashqi kalit har bir qiymati bitta satrga mos kelishi ta'minlangan bo`lishi kerak. Bu kalit unikal bo`lib, bo`sh (NULL) qiymatlarga ega bo`lmasligi kerak. Shuning uchun ajdod kalit maydonlari RIMARY KEY cheklanishiga ega bo`lishi yoki NOT NULL cheklanishi bilan birga UNIQUE deb e'lon qilinishi kerak.

Tashqi kalit ajdod kalitda majud qiymatlarga yoki bo`sh (NULL) qiymatga ega bo`lishi mumkin. Boshqa qiymat kiritishga urinish rad etiladi. Tashqi kalitga NOT NULL deb e'lon qilish mumkin, lekin bu maqsadga muvofiq emas.

Cheklanishlar ta'siri.

Tashqi kalit maydonlariga INSERT yoki UPDATE yordamida kiritilayotgan qiymatlar ajdod kalitlariga oldin kiritilgan bo`lishi kerak. Tashqi kalit ixtyoriy satrini DELETE yordamida o`chirish mumkin. ANSI ta'rifi bo`yicha: tashqi kalit yordamida ilova qilinayotgan ajdod kalit qiymatini

o`chirib yoki o`zgartirib bo`lmaydi. ANSI tarkibiga kirmagan ajdod kalit maydonlarini o`zgartirish yoki o`chirish qoidalari mavjud:

Cheklangan (RESTRICT) o`zgartishlar. Siz (ANSI usulida) ajdod kalitlarda cheklangan deb ko`rsatishingiz yoki man qilishingiz mumkin.

Kaskadlanuvchi (CASCADE) o`zgartishlar. Agarda ajdodkalitda o`gartish kiritsangiz, tashqi kalitda xuddi shunday o`zgartishlar avtomatik yuz beradi.

Bo`sh (NULL) o`zgartishlar. Siz ajdod kalitda uzgartirish kiritganingizda tashqi kalit maydonlari avtomatik NULL qiymat oladi (tashqi kalitda NULL qiymat ruxsat etilgan bo`lsa).

Yuqorida ko`rsatilgan effektlar UPDATE va DELETE komandalari bajarilganda ajdod kalit o`zgarishini ko`rsatadi va quyidagicha aniqlanadi:

CREATE TABLE <table-name >
(<column name> <data type>[(<size>)],
<column name> <data type>[(<size>)],
...
FOREGIGN KEY (<column name>,..) REFERENCES <table name>[(<column name>, ...)]
ON UPDATE [CASCADE|RESTRICT|SET NULL]
ON DELETE [CASCADE|RESTRICT|SET NULL],
...);

1.2.4 Maydonlar qiymatlarini kiritish, o`chirish va o`zgartirish.

Qiymatlarni kiritish.

Hamma satrlar SQLda INSERT komandasi yordamida kiritiladi. INSERT quyidagi formatga ega:

INSERT INTO <table name | view name> [(column [,column] ...)]
VALUES (<value> [,<value>] ...);

Satrlarni o`chirish.

Satrlarni jadvaldan DELETE komandasi bilan o`chirish mumkin. U aloxida qiymatlarni emas faqat satrlarni o`chiradi.

DELETE quyidagi formatga ega:

DELETE FROM <table name | view name>

[WHERE search-condition];

Maydon qiymatlarini o`zgartirish.

Bu o`zgartirish UPDATE komandasi yordamida bajariladi. Bu komandada UPDATE ifodasidan so`ng jadval nomi va SET ifodasidan so`ng ma'lum ustun uchun o`zgartirish ko`rsatiladi. UPDATE ikki formatga ega. Ulardan birinchisi:

UPDATE <table name | view name>

SET column = expression [, column = expression] ...

[WHERE search-condition]

bu erda expression - bu ustun | ifoda | konstanta | o`zgaruvchi.

Ikkinchi variant:

UPDATE <table name>

SET column = expression, ...

[FROM table-list]

[WHERE search-condition]

1.2.5 SELECT so`rov operatori.

SELECT operatori MB jadvallaridan natijaviy to`plam olish uchun mo`ljallangan ifodadir. Biz SELECT operatori yordamida so`rov beramiz, u bo`lsa ma'lumotlar natijaviy to`plamini qaytaradi. Bu ma'lumotlar jadval shaklida qaytariladi. Bu jadval keyingi SELECT operatori tomonidan qayta ishlanishi mumkin va xokazo.[11]

Operator SQL92 standartiga ko`ra quyidagi ko`rinishga ega:

SELECT -- ALL ------- sxema , ustun -------

-- DISTINCT -- ---- * -------------------------

FROM -- sxema , Jadval .. ----------------------

WHERE -- izlash sharti ------------------------

GROUP BY -- sxema , ustun ------------------

HAVING -- izlash sharti ----------------------

ORDER BY - tartiblash spetsifikatori --------

Birinchi qoida, SELECT ifodasi o`z ichiga albatta FROM ifodasini olishi kerak. Qolgan ifodalar kerak bo`lsa ishlatiladi.

SELECT ifodasidan so`ng so`rovda qaytariluvchi ustunlar ro`yxati yoziladi.

FROM ifodasidan so`ng so`rovni bajarish uchun jadvallar nomi yoziladi.

WHERE ifodasidan so`ng agar ma'lum satrlarni qaytarish lozim bo`lsa, izlash sharti yoziladi.

GROUP BY ifodasi guruxlarga ajratilgan natijaviy so`rov yaratishga imkon beradi.

HAVING ifodasidan guruxlarni qaytarish sharti yoziladi va GROUP BY bilan birga ishlatiladi.

ORDER BY ifodasi ma'lumotlar natijaviy to`plamini tartiblash yo`nalishini aniqlaydi.

1.2.6 Mantiqiy operatorlar.

BETWEEN va IN Operatorlari.

BETWEEN ifodasi bu qiymatlar diapazoniga tegishlilikni tekshirishdir. Ifoda sintaksisi quyidagicha:

--- tekshirilayotgan ifoda ------- BETWEEN ----- quyi ifoda AND yuqori ifoda

- NOT -

NOT ifodasi shartni teskarisiga o`giradi,Ya'ni tegishli emas ma'noni bildiradi. NOT ifodasi yordamida berilgan diapazonga tegishlilikni tekshirish mumkin.

Operator LIKE.

LIKE ifodasi sintaksisi SQL92 standarti bo`yicha quyidagi ko`rinishga ega:

--- IMYA STOLBTSA --------- LIKE (shablon) ----------

NOT ESCAPE (o`tkazish nomi)

LIKE '%n' operatori 'n' harfiga tugaydigan hamma yozuvlarni ko`rsatadi, agar '%' shabloni birinchi kelsa.

Operator IS NULL.

SELECT operatori uchun NULL qiymati bilan ishlash qoidalarini ko`ramiz. Konkret misol ko`ramiz:

NULL = NULL

Bunday tekshirish yana NULL qaytaradi! Qiymat tekshiruvchi operator uchun agar natija TRUE bo`lmasa, satr natijaviy to`plamga kirmaydi! Lekin bunday satrlar aslida mavjuddir! Bu holda NULL qiymatiga tekshirish to`g`ri operatorini qo`llash lozim:

---------- ustunning nomi IS ---------- NULL ---------- NOT

Izlashning "qo`shma" shartlarini ko`rib chiqamiz. WHERE operatorida OR, AND, NOT operatorlari bilan bog`langan bir necha izlash shartlarini qo`llash mumkin. Bu operatorlar sintaksisi quyidagicha: NOT, OR, AND operatorlarning sintaksisi.

(-------- WHERE ---------- SHART ----------)

(--- NOT ---)

(-------- AND ----------)

(-------- OR ----------)

Qo`shma izlash operatorlarining har biri o`z ustivorligiga ega. Eng yuqori ustivorlik NOT ga tegishli, undan so`ng AND va oxirida OR! SQL92 standartida IS operatori yordamida ifoda rost, yolg`on yoki aniqlanmaganligini tekshirish mumkin. Uning sintaksisi quyidagicha:

IS operatori sintaksisi.

--------- Solishtirish ---------- IS (------ TRUE ----------)

(------ FALSE ----------)

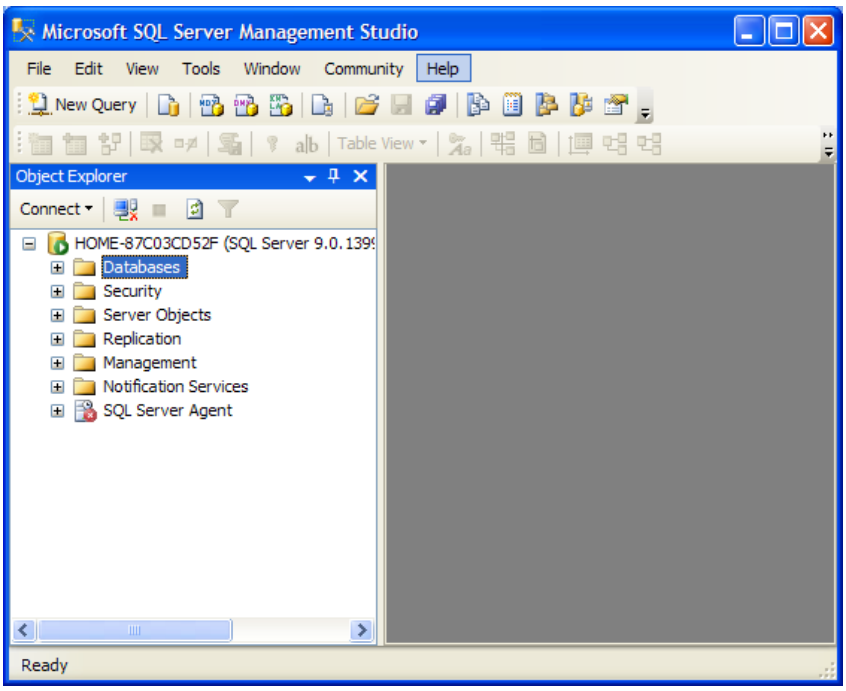
--- Mantiqiy ifoda --- (------ UNKNOWN ----------)

1.3 Microsoft SQL Server Berilganlar bazasini boshqarish tizimi.

Jadvallar va ma’lumotlar bazasini yaratish ishi bo’yicha biz Microsoft SQL Server Management Studio bilan ishlaymiz.

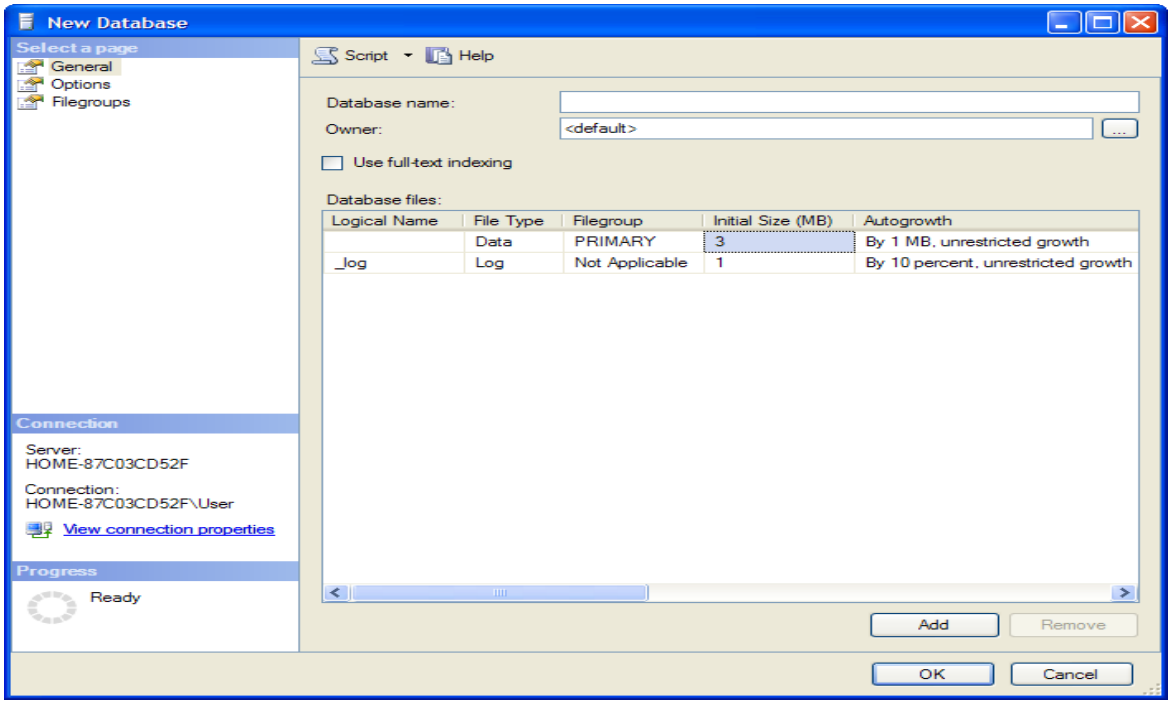
Ma’lumotlar bazasi strukturasini aniqlash.

Microsoft SQL Server Management Studio ning tashqi ko’rinish oynasi quyidagicha (1.1-chizma).



1.1- chizma Microsoft SQL Server Management Studio oynasining ko'rinishi.

Ma'lumotlar bazasini yaratish uchun <<Databases>>(База данных) ning ustiga sichqonchaning o'ng tugmasini bosamiz, hosil bo'lgan kontekstli menyudan <<New Database>> ni bosamiz. Shundan so'ng bizga quyidagi ma'lumotlar bazasini yaratish oynasi paydo bo'ladi(1.2-chizma).



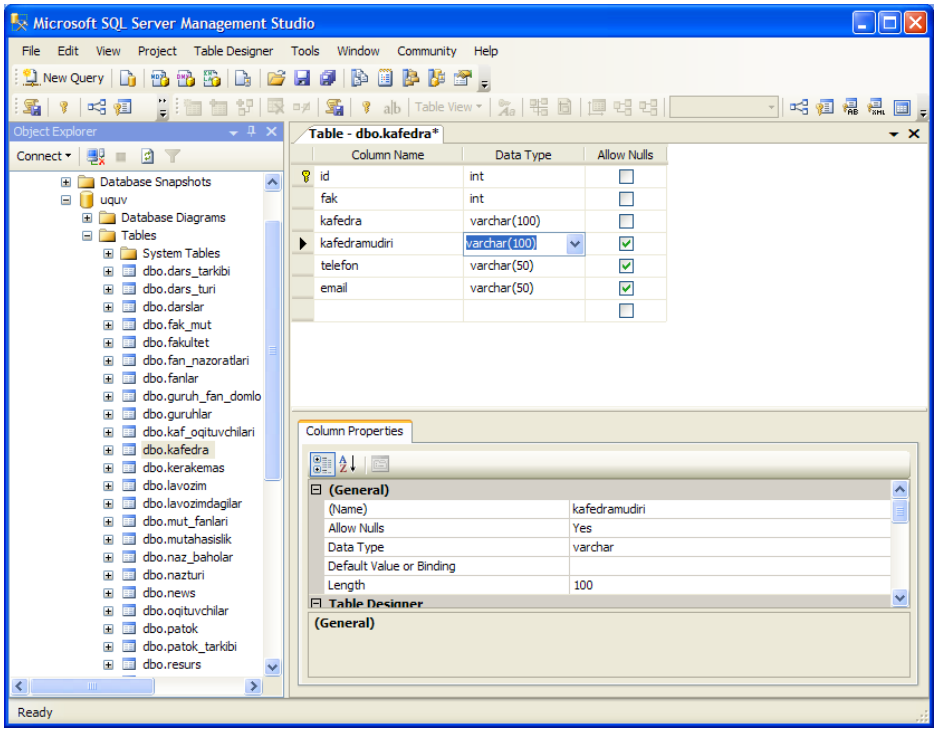
1.2- chizma Ma'lumotlar bazasini yaratish oynasi

Berilgan ushbu oynada ma'lumotlar bazasining nomi hamda ma'lumotlar bazasining fayllari turgan joyning yo'li ko'rsatiladi. OK tugmasini bosgandan

so’ng yaratmoqchi bo’lgan ma’lumotlar bazamiz yaratiladi va oldindan yaratilgan ma’lumotlar bazalarining ro’yxatida paydo bo’ladi.

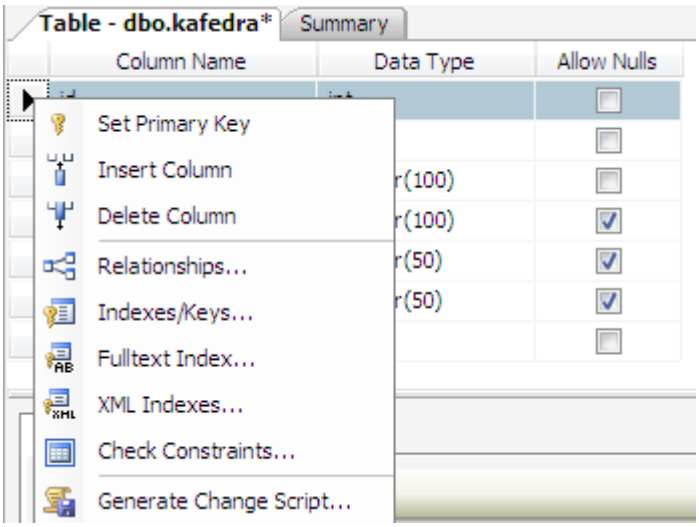
Yaratilgan ma’lumotlar bazamiz bo’sh, ya’ni hech qanday jadvallari yo’q. Shuning uchun keyingi vazifamiz jadvallarni yaratish bo’ladi. Jadvallarning tuzilishi xuddi Access da tuzilgan jadvallar kabi bo’ladi.[11]

Jadvallarni yaratish uchun <<Tables>> novdasining kontekstli menyusidan <<New Table>> ni tanlaymiz. Shundan so’ng Management Studio muhitining ko’rinishi quyidagi ko’rinishga ega bo’ladi(1.3-chizma).



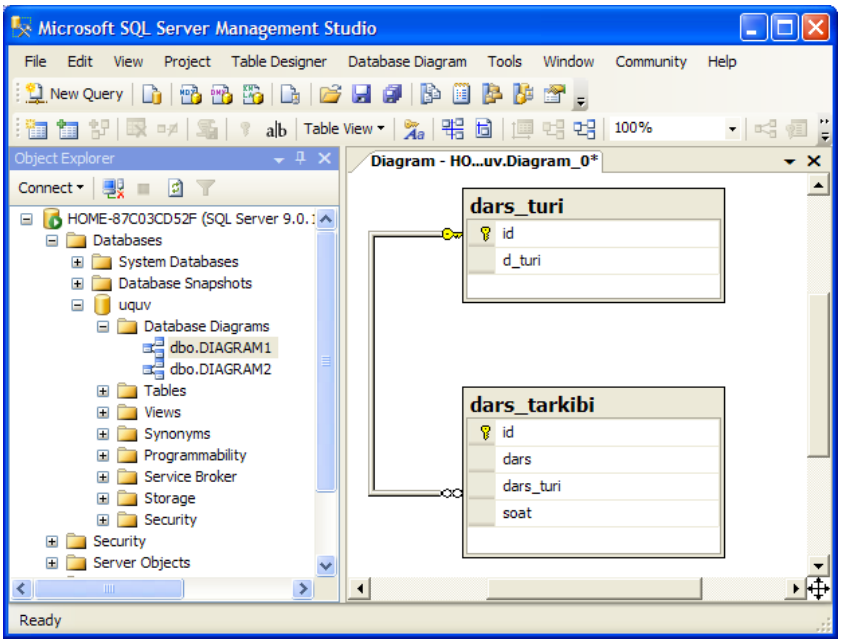
1.3-chizma . Management Studio ning jadval yaratish paytidagi ko’rinishi

Jadvallar orasidagi bog’lamni aniqlash uchun birlamchi kalitni berish zarur. Buning uchun zarur bo’lgan maydonning kontekstli menyusidan <<Set primary key>> ni tanlash kerak(1.4-chizma).



1.4-chizma . Birlamchi kalitni berish.

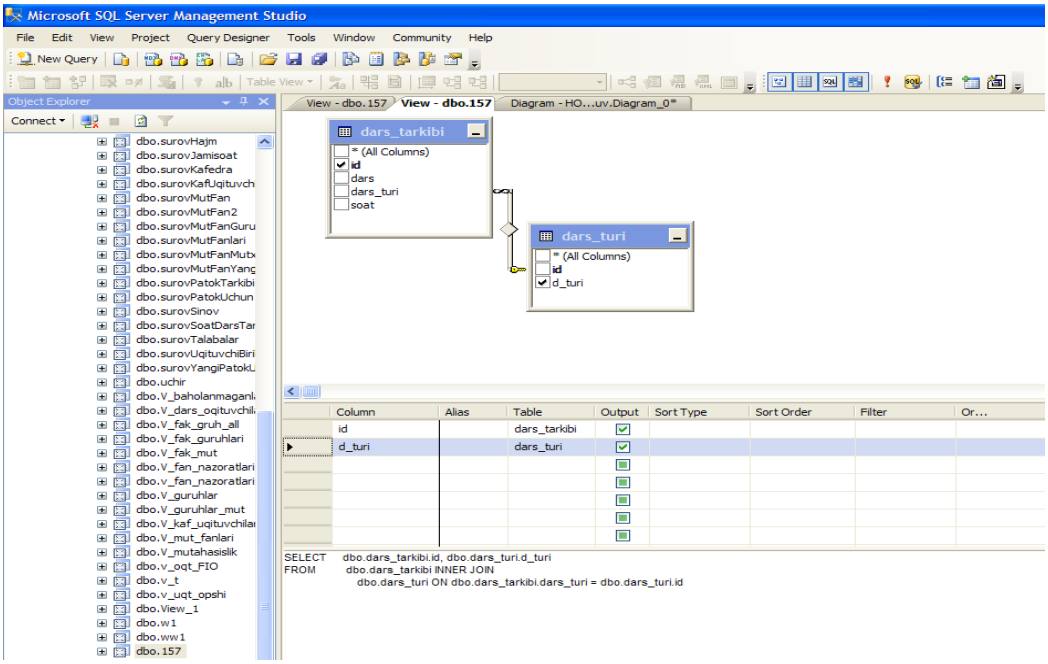
Jadvallar orasidagi bog'lamni va ma'lumotlar sxemasini yaratish uchun yangi ma'lumotlar bazasining diagrammasini yaratish zarur. Buning uchun <<Database Diagrams>> novdasining konteksli menyusidan kerakli punktni tanlaymiz. Hosil bo'lgan oynada, diagrammaga kerakli bo'lgan jadvallarni qo'shamiz, Management Studio muhitining ko'rinishi quyidagicha o'zgaradi(1.5-chizma).



1.5-chizma . Jadvallarni bir biriga bog'lash.

Management Studio muhitida so'rov yaratish quyidagicha amalgam oshiriladi. <<Views>> novdasining konteksli menyusidan <<New view>> punktini tanlaymiz. Kerakli jadvallarni tanlaganimizdan kerakli ustunlarni

tanlaganimizdan so’ng Management Studio muhitining ko’rinishi quyidagicha o’zgaradi(1.6-chizma).



1.5-chizma . So’rov yaratish.

Xulosa

Bu bobda berilganlar bazasini boshqarish tizimlari, ma`lumotlar bazasining asosiy vazifalari haqida. SQL tili unda jadvallar yaratish, ularga ma’lumot kiritish. SQL tilda so’rovlar yozish. Microsoft SQL Server berilganlar bazasini boshqarish tizimi unda jadvallar yaratish va boshqa ma’lumotlar keltirilgan.

II BOB. Microsoft Visual Studio 2010 muhitida ADO.Net texnologiyasi.

2.1 C#dasturlash tili.

2.1.1 C#da arifmetik amallar.

Ko'p programmalar ijro davomida arifmetik amallarni bajaradi. C#dagi amallar quyidagi jadvalda berilgan. Ular ikkita operand bilan ishlatdi. C#dagi amal arifmetik operator, algebraik ifoda C#dagi ifodasi

Qo'shish + h+19 h+19

Ayirish - f-u f-u

Ko'paytirish * sl s*l

Bo'lish / v/d, vod v/d

Modul olish % k mod 4 k%4

C#da qavslarning ma'nisi xuddi algebradagidekdir. Undan tashqari boshqa boshqa algebraik ifodalarning ketma-ketligi ham odatdagidek. ko'paytirish, bo'lish va modul olish operatorlari ijro ko'radi. Agar bir necha operator ketma-ket kelsa, ular chapdan o'nga qarab ishlanadi. Bu operatorlardan keyin esa qo'shish va ayirish ijro etiladi.

Mantiqiy solishtirish operatorlari.

C# bir necha solishtirish operatorlariga ega. Algebraik ifoda C#dagi operator C#dagi ifoda Algebraik ma'nosi tenglik guruhi

= == x==y x tengdir y ga

teng emas != x!=y x teng emas y ga

solishtirish guruhi

> > x>y x katta y dan

< < x<y x kichkina y dan

katta-teng >= x>=y x katta yoki teng y ga

kichik-teng <= x<=y x kichik yoki teng y ga

==, !=, >= va <= operatorlarni yozganda oraga bo'sh joy qo'yib ketish sintaksis xatodir.

2.1.2 Boshqaruv ifodalari.

Bu bo'limda biz strukturali dasturlashning asosiy prinsip va qismlarini ko'rib chiqamiz. Ma'lum bir dasturni yozish uchun belgilangan qadamlarni bosib o'tish kerak.

Masala aniqlangandan so'ng uni yechish uchun mo'ljallangan algoritm tuziladi. Keyin esa psevdokod yoziladi. Psevdokod algoritmda bajariladigan qadamlarni ko'rsatadi. Bunda faqat bajariladigan ifodalar ko'rib chiqiladi. Psevdokodda o'zgaruvchi e'lonlari yoki boshqa ma'lum bir dasturlash tiliga mansub bo'lgan yordamchi amallar bo'lmaydi. Psevdokodni yozish dasturlashni ancha osonlashtiradi, algoritm mantig'ini tushunishga va uni rivojlanritishga katta yordam beradi.

Misol uchun bir dasturning rejasi va psevdokodi 3-4 oy yozilgan bo'lsa va yuqori darajada detallashtirilgan bo'lsa, ushbu dasturning C# yoki boshqa tildagi kodini yozish 2-3 hafta vaqt oladi xolos. Bu yozilgan programmada xato ancha kam bo'ladi, uni keyinchalik takomillashtirish arzonga tushadi. Hozirgi paytda dastur o'zgarishi favqulotda hodisa emas, balki zamon talabidir.

Asosan dasturdagi ifodalar ketma-ket, navbatiga ko'ra ijro etiladi. Gohida bir shart bajarilishiga ko'ra, ijro boshqa bir ifodaga o'tadi. Navbatdagi emas, dasturning boshqa yerida joylashgan ifoda bajariladi. Yani sakrash yoki ijro ko'chishi vujudga keladi. 60-chi yillarga kelib, dasturlardagi ko'pchilik xatolar aynan shu ijro ko'chishlarining rejasiz ishlatilishidan kelib chiqishi ma'lum bo'ldi. Bunda eng katta aybdor deb bu ko'shishlarni amalga oshiruvchi goto (...ga bor) ifodasi belgilandi. goto dastur ijrosini deyarli istalgan yerga ko'chirib yuborishi mumkin. Bu esa programmani o'qishni va uning strukturasini murakkablashtirib yuboradi. Shu sababli "strukturali dasturlash" atamasi "goto ni yo'q qilish" bilan tenglashtirilardi. Shuni aytib o'tish kerakki, goto kabi shartsiz sakrash amallarini bajaruvchi ifodalar boshqa dasturlash tillarida ham bor.

Tadqiqotlar shuni ko'rsatdiki, istalgan programma goto siz yozilishi mumkin ekan. goto siz yozish uslubi strukturali dasturlash deb nom oldi. Va

bunday dastur yozish metodi katta iqtisodiy samara beradi. Strukturali dasturlash asosi shundan iboratki, har bir programma faqatgina uch xil boshqaruv strukturalaridan iboratdir. Bular ifodalarni ketma-ket ijro etish strukturasi (sequence structure), tanlash strukturasi (selection structure) va amalni qayta ijro etish strukturasidir (repetition structure).

Ifodalarni ketma-ket ijro etish strukturasi C# tomonidan ta'minlanadi. Normal sharoitda C# ifodalari dasturdagi navbatiga ko'ra bajariladi. Tanlash buyruqlari uchtadir. Bular if, if/else va switch dir. Qayta ijro etish buyruqlari guruhiga ham uchta a'zo bor, bular while, do/while va for. Bularni har birini keyinroq tahlil qilib chiqamiz.

Yuqoridagi buyruqlar nomlari C#dasturlash tilining maxsuss so'zlaridir. Dasturchi bu so'zlarni o'zgaruvchi yoki funksiyalar nomi sifatida qo'llashi ta'qiqlanadi.

if strukturasi.

Biz shartga ko'ra bir necha harakat yo'lidan bittasini tanlaymiz. Misol uchun agar bolaning yoshi 7 ga teng yoki katta bo'lsa u maktabga borishi mumkin bo'lsin. Buni C#da if ni qo'llab yozaylik.[3]

```
if (yosh >= 7)
maktab();
```

Bu yerda shart bajarilishi yoki bajarilmasligi mumkin. Agar yosh o'zgaruvchisi 7 ga teng yoki undan katta bo'lsa shart bajariladi va maktab() funksiyasi chaqiriladi. Bu holat true (to'g'ri) deyiladi agar yosh 7 dan kichik bo'lsa, maktab() tashlab o'tiladi. Ya’ni false (noto'g'ri) holat yuzaga keladi.

Biz shart qismini mantiqiy operatorlarga asoslanganligini ko'rib chiqqan edik. Aslida esa shartdagi ifodaning ko'rinishi muhim emas – agar ifodani nolga keltirish mumkin bo'lsa false bo'ladi, noldan farqli javob bo'lsa, musbatmi, manfiymi, true holat paydo bo'ladi va shart bajariladi. Bunga qo'shimcha qilib o'tish kerakki, C#da maxsus bool tipi mavjud. Bu tipdagi o'zgaruvchilarning yordamida bul (mantiqiy) arifmetikasini amalga oshirish mumkin. bool o'zgaruvchilar faqat true yoki false qiymatlarini olishlari mumkin.

if/else strukturasi.

if ni qo'llaganimizda ifoda faqat shart haqiqat bo'lgandagina bajariladi, aks holda tashlanib o'tiladi. if/else yordamida esa shart bajarilmaganda (false natija chiqqanda) else orqali boshqa bir yo'ldan borishni belgilash mumkin. Misolni takomillashtirsak. Bola 7 yosh yoki undan katta bo'lsa maktabga, 7 dan kichkina bo'lsa bog'chaga borsin.

```
if (yosh >= 7)
maktab(); //nuqta-vergul majburiydir
else
bogcha();
```

Yuqorida if ga tegishli bo'lgan blok bitta ifodadan (maktab()) iborat. Shu sababli nuqta-vergul qo'yilishi shart. Buni aytib o'tishimizning sababi, masalan Pascalda hech narsa qo'yilmasligi shart. C#da bitta ifosa turgan joyga ifodalar guruhini {} qavslarga olingan holda qo'ysa bo'ladi.[4]

switch strkturasi.

if-else-if yordami bilan bir necha shartni test qilishimiz mumkin. Lekin bunday yozuv nisbatan o'qishga qiyin va ko'rinishi qo'pol bo'ladi. Agar shart ifoda butun son tipida bo'lsa yoki bu tipga keltirilishi mumkin bo'lsa, biz switch (tanlash) ifodalarini ishlata olamiz. switch strukturasi bir necha case etiketlaridan (label) va majburiy bo'lmagan default etiketidan iboratdir. Etiket bu bir nomdir. U dasturnig bir nuqtasidaga qo'yiladi. Programmaning boshqa yeridan ushbu etiketga o'tishni bajarish mumkin. O'tish yoki sakrash goto bilan amalga oshiriladi, switch blokida ham qo'llaniladi.

while takrorlash strukturasi.

Takrorlash strukturasi bir ifoda yoki blokni ma'lum bir shart to'g'ri (true) bo'lishi davomida qaytarish imkonini beradi. Qaytarilayatgan ifoda shartga ta'sir ko'rsatishishi kerak. Ma'lum bir vaqt o'tkandan keyin shart false ga o'zgartilishi kerak. Bo'lmasam while (davomida) tugatilmaydi. while faqat o'zidan keyin kelgan ifodaga ta'sir qiladi. Agar biz bir guruh amallarni qaytarmoqchi bo'lsak, ushbu blokni {} qavslar ichiga olishimiz kerak. Shart takrorlanuvchi blokning

boshida tekshirilgani sababli, agar shart noto'g'ri bo'lib chiqsa, blokni hech ijro ko'rmasligi ham mumkin.

```
while (shart) {
ifoda1;
ifoda2;
....
.....
}
```

do/while takrorlash strukturasi.

do/while ifodasi while strukturasiga o'xshashdir. Bitta farqi shundaki while da shart boshiga tekshiriladi. do/while da esa takrorlanish tanasi eng kamida bir marta ijro ko'radi va shart strukturaning so'ngida test qilinadi. Shart true bo'lsa blok yana takrorlanadi. Shart false bo'lsa do/while ifodasidan chiqiladi. Agar do/while ichida qaytarilishi kerak bo'lgan ifoda bir dona bo'lsa {} qavslarning keragi yo'qdir. Quyidagicha bo'ladi:[5]

```
do
ifoda;
while (shart);
```

Lekin {} qavslarning yo'qligi dasturchini adashtirishi mumkin. Chunki qavssiz do/while oddiy whilening boshlanishiga o'hshaydi. Buni oldini olish uchun {} qavslarni har doim qo'yishni tavsiya etamiz.

```
do {
ifoda1;
ifoda2;
....
.....
} while (shart);
```

Qiymat berish operatorlari.

Bu qismda keyingi bo'limlarda kerak bo'ladigan tushunchalarni berib o'tamiz. C#da hisoblashni va undan keyin javobni o'zgaruvchiga beruvchi bir

necha operator mavjuddir. Misol uchun:

k = k * 4; ni

k *= 4;

deb yozsak bo'aladi.

Bunda += operatorining chap argumenti o'ng argumentga qo'shiladi va javob chap argumentda saqlanadi. Biz har bir operatorni ushbu qisqartirilgan ko'rinishda yoza olamiz (+=, -=, /=, *= %=). Ikkala qism birga yoziladi. Qisqartirilgan operatorlar tezroq yoziladi, tezroq kompilyatsiya qilinadi va ba'zi bir hollarda tezroq ishlaydigan mashina kodi tuziladi.[6]

1 ga oshirish va kamaytirish operatorlari (increment and decrement).

C#da bir argument oluvchi inkrenet (++) va dekrement (--) operatorlari mavjuddir. Bular ikki ko'rinishda ishlatilinadi, biri o'zgaruvchidan oldin (++f - preinkrement, --d - predekrement), boshqasi o'zgaruvchidan keyin (s++ - postinkrement, s-- - postdekrement) ishlatilgan holi. Bularning bir-biridan farqini aytib o'taylik. Postinkrementda o'zgaruvchining qiymati ushbu o'zgaruvchi qatnashgan ifodada ishlatilinadi va undan keyin qiymati birga oshiriladi. Preinkrementda esa o'zgaruvchining qiymati birga oshiriladi, va bu yangi qiymat ifodada qo'llaniladi. Predekrement va postdekrement ham aynan shunday ishlaydi lekin qiymat birga kamaytiriladi. Bu operatorlar faqatgina o'zgaruvchining qiymatini birga oshirish/kamaytirish uchun ham ishlatilinishi mumkin, yani boshqa ifoda ichida qo'llanilmasdan. Bu holda pre va post formalarining farqi yo'q..

Mantiqiy operatorlar.

Bosqaruv strukturalarida shart qismi bor dedik. Shu paytgacha ishlatgan shartlarimiz ancha sodda edi. Agar bir necha shartni tekshirmoqchi bo'lganimizda ayri-ayri shart qismlarini yozardik. Lekin C#da bir necha sodda shartni birlashtirib, bitta murakkab shart ifodasini tuzishga yordam beradigan mantiqiy operatorlar mavjuddir. Bular mantiqiy VA - && (AND), mantiqiy YOKI - || (OR) va mantiqiy INKOR - ! (NOT). Bular bilan misol keltiraylik.

Faraz qilaylik, bir amalni bajarishdan oldin, ikkala shartimiz (ikkitadan ko'p ham bo'lishi mumkin) true (haqiqat) bo'lsin.[7]

if (i < 10 && l >= 20){...}

Bu yerda {} qavslardagi ifodalar bloki faqat i 10 dan kichkina va l 20 dan katta yoki teng bo'lgandagina ijro ko'radi.

AND ning (&&) jadvali:

ifoda1 ifoda2 ifoda1 && ifoda2

false (0) false (0) false (0)

true (1) false (0) false (0)

false (0) true (1) false (0)

true (1) true (1) true (1)

Bu yerda trueni qiymati o'rniga 1, falseni qiymati o'rniga 0 ni qo'llashimiz mumkin.

for takrorlash strukturasi.

for strukturasi sanovchi (counter) bilan bajariladigan takrorlashni bajaradi. Boshqa takrorlash bloklarida (while, do/while) takrorlash sonini kontrol qilish uchun ham sanovchini qo'llasa bo'lardi, bu holda takrorlanish sonini o'ldindan bilsa bo'lardi, ham boshqa bir holatning vujudga kelish-kelmasligi orqali boshqarish mumkin edi. Ikkinchi holda ehtimol miqdori katta bo'ladi. Masalan qo'llanuvchi belgilangan sonni kiritmaguncha takrorlashni bajarish kerak bo'lsa biz while li ifodalarni ishlatamiz. for da esa sanovchi ifodaning qiymati oshirilib (kamaytirilib) borilvuradi, va chegaraviy qiymatni olganda takrorlanish tugatiladi. for ifodasidan keyingi bitta ifoda qaytariladi. Agar bir necha ifoda takrorlanishi kerak bo'lsa, ifodalar bloki {} qavs ichiga olinadi. for strukturasi uch qismdan iboratdir. Ular nuqta-vergul bilan bir-biridan ajratiladi. for ning ko'rinishi:

for(1. qism ; 2. qism ; 3. qism){

takror etiladigan blok

}

1. qism - e'lon va initsalizatsiya.

2. qism - shartni tekshirish (oz'garuvchini chegaraviy qiymat bilan solishtirish).
3.qism - o'zgaruvchining qiymatini o'zgartirish.

Qismlarning bajarilish ketma-ketligi quyidagichadir:
Boshida 1. qism bajariladi (faqat bir marta), keyin 2. qismdagi shart tekshiriladi va agar u true bo'lsa takrorlanish bloki ijro ko'radi, va eng ohirda 3. qismda o'zgaruvchilar o'zgartiriladi, keyin yana ikkinchi qismga o'tiladi.

2.1.3 Funksiyalar.

C#da dasturlashning asosiy bloklaridan biri funksiyalardir. Funksiyalarning foydasi shundaki, katta Masalan, bir necha kichik bo'laklarga bo'linib, har biriga alohida funksiya yozilganda, Masalan, yechish algoritmi ancha soddalashadi. Bunda dasturchi yozgan funksiyalar C#ning standart kutubxonasi va boshqa firmalar yozgan kutubxonalar ichidagi funksiyalar bilan birlashtiriladi. Bu esa ishni osonlashtiradi. Ko'p holda dasturda takroran bajariladigan amalni funksiya sifatida yozish va kerakli joyda ushbu funksiyani chaqirish mumkin. Funksiyani programma tanasida ishlatish uchun u chaqiriladi, yani uning ismi yoziladi va unga kerakli argumentlar beriladi. () qavslar ushbu funksiya chaqirig'ini ifodalaydi.[10]

Masalan:

```
foo();
k = square(l);
```

Demak, agar funksiya argumentlar olsa, ular () qavs ichida yoziladi. Argumentsiz funksiyadan keyin esa () qavslarning o'zi qo'yiladi.

Ma`lumotlar tipi (data types).

Shu paytgacha ma'lumotlar tipi deganda butun son va kasrli son bor deb kegan edik. Lekin bu bo'limda malumotlar tipi tushunchasini yaxshiroq ko'rib chiqish kerak bo'ladi. Chunki funksiyalar bilan ishlaganda argument kiritish va qiymat qaytarishga to'g'ri keladi. Agar boshidan boshlaydigan bo'lsak, kompyuterda hamma turdagi ma'lumotlar 0 va 1 yordamida kodlanadi. Buning sababi shuki, elektr uskunalar uchun ikki holat tabiiyidir, tok oqimi bor yoki yo'q, kondensatorda zaryad bor yoki yo'q va hakozo. Demak biz bu holatlarni

oladigan jihozlarni bir quti deb faraz qilsak, quti ichida yo narsa bo'ladi, yo narsa bo'lmaydi. Mantiqan buni biz bir yoki nol deb belgilaymiz. Bu kabi faqat ikki holatga ega bo'lishi mumkin bo'lgan malumot birligiga biz bit deymiz. Bu birlik kichik bo'lgani uchun kompyuterda bitlar guruhi qo'llaniladi. Bittan keyingi birlik bu bayt (byte). Baytni sakkizta bit tashkil etadi. Demak bir bayt yordamida biz 256 ta holatni kodlashimiz mumkin bo'ladi. 256 soni ikkining sakkizinchi darajasiga tengdir. Bitimiz ikki holatga ega bo'lgani uchun biz kompyuterni ikkili arifmetikaga asoslangan deymiz. Ammo agar kerak bo'lsa, boshqa sistemaga asoslangan mashinalarni ham qo'llash mumkin. Masalan uchli sanoq sistemasiga asoslangan kompyuterlar bor. Informatika faniga ko'ra esa, hisoblash mashinasi uchun eng optimal sanoq sistemasi e ga teng bo'lar ekan. Demak amaldagi sistemalar ham shu songa iloji borisha yaqin bo'lishi kerakdir.

C#da baytga asoslangan tip char dir. char tipi butun son tipida bo'lib, chegaraviy qiymatlari -128 dan +127 gachadir. O'zgaruvchilar vergul bilan ayriladi. E'lon bilan bir vaqtning o'zida boshlang'ich qiymat ham berish imkoni bor. Mashina ichida baytdan tashkil topgan boshqa kattaliklar ham bor. Ikki baytdan tuzilgan kattalik so'z (word) deyiladi, unda 16 bit bo'ladi. 4 ta bayt guruhi esa ikkili so'z (double word) bo'ladi. Bu birlik 32 bitli mashimalarda qo'llaniladi. Hozirda qo'llanilmoqda bo'lgan mashinalar asosan 32 bitlidir, Masalan Pentium I/II/III sistemalari. C#da butun sonlarning ikki tipi bor. Biri char - uni ko'rib chiqdik. Ikkinchisi int dir. Mashinalarning arxitekturasi qanday kattalikda bo'lsa, int tipining ham kattakigi xuddi shunday bo'ladi. 16 bitlik mashinalarda int 16 bit edi. Hozirda esa int ning uzunligi 32 bitdir. int (integer - butun son) tipi charga o'xshaydi. Farqi bir baytdan kattaligidadir. 16 bitli int ning sig'imi -32768 dan +32767 gachadir. 32 bitli int esa -2 147 483 648 dan +2 147 483 647 gacha o'rin egallaydi. Bu ikki butun son tipidan tashqari C#da ikki tur vergulli, (nuqtali) yani haqiqiy son tipi mavjud. Bulardan biri float, xotirada 4 bayt joy egallaydi. Ikkinchisi esa double, 8 bayt kattalikka ega. Bularning xarakteristikalari quyidagi jadvalda berilgan. Ushbu tiplar bilan ishlaganda unsigned(ishorasiz, +/- siz), signed (ishorali) long (uzun) va short (qisqa)

sifatlarini qo'llasa bo'ladi. unsigned va signedni faqat butun son tiplari bilan qo'llasa bo'ladi. unsigned qo'llanganda sonning ishorat biti bo'lmaydi, ishorat biti sonning kattaligini bildirish uchun qo'llaniladi.

2.1.4 Massivlar.

Bu qismda dasturdagi ma'lumot strukturalari bilan tanishishni boshlaymiz. Dasturda ikki asosiy tur ma'lumot strukturalari mavjuddir. Birinchisi statik, ikkinchisi dinamikdir. Statik deganimizda xotirada egallagan joyi o'zgarmas, dastur boshida beriladigan strukturalarni nazarda tutamiz. Dinamik ma'lumot tiplari dastur davomida o'z hajmini, egallagan xotirasini o'zgartirishi mumkin. Agar struktura bir xil kattalikdagi tiplardan tuzilgan bo'lsa, uning nomi massiv (array) deyiladi. Massivlar dasturlashda eng ko'p qo'laniladigan ma'lumot tiplaridir. Bundan tashqari strukturalar bir necha farqli tipdagi o'zgaruvchilardan tashkil topgan bo'lishi mumkin. Buni biz klass (Pascalda record) deymiz. Masalan bunday strukturamiz ichida odam ismi va yoshi bo'lishi mumkin. Massivlar xotirada ketma-ket joylashgan, bir tipdagi o'zgaruvchilar guruhidir. Alohida bir o'zgaruvchini ko'rsatish uchun massiv nomi va kerakli o'zgaruvchi indeksini yozish mumkin.

2.2 ADO.NET texnologiyasi.

2.2.1 ADO.NET nima o’zi?

ADO.NET bu shunday sinflar to’plamiki, u C# va .NET Frameworkda relyatsion jadvallar va ma’lumotlar bilan ishlashda qulaylik yaratadi. Relyatsion ma’lumotlar bazalari bo’lmish Microsoft SQL Server va Microsoft Access bilan bir qatorda boshqa relyatsion ma’lumotlar bazalari va hattoki relyatsion bolmagan ma’lumotlar manbalarni ham o’z ichiga oladi. ADO.NET texnologiyasi barcha .NET dan foydalanuvchi dasturlash tillarida .NET Framework va loyihalashni yaxlid holda birlashtirishda foydalaniladi, u C#ning o’ziga xos xususiyati hisoblanadi.

ADO.NET System.Data sinfi va uning sinf ostilari bo’lmish System.Data.SqlClient va System.Data.Linq bilan bir qarorda System.Xml sinfi va sinf ostilarining bir qismini oz ichiga olgan sinflardan tashkil topgan.

Nimaga u ADO.NET deb nomlandi?.

Balki siz o'ylayotgandirsiz nima uchun ADO.NET bo'limini .NET Framework o'z ichiga oldi? Nima uchun uni shunchaki System.Data va boshqa nom bilan nomlanmadi? ADO.NET ActiveX Data Objects (ADO) ning kengaytirilgan sinflar to'plami bo'lib, u ma'lumotlardan Microsoftning oldingi avlod texnologiyasiga nisbatan tezroq foydalanish imkonini beradi.

ADO.NET- ADO ga nisbatan ma'lumotlardan tez foydalanuvchi sinflar va .NET dasturlash muhitlarini yangilash va kengaytirishda xizmat qildi. Agarda sizni bu texnologiyaning yaratilish tarixi, boshqa interfeysdan turib Microsoft ma'lumotlar bazasi bilan ishlash qanday amalga oshirilishi, shunga o'xshash ODBC yoki OLE DB ishlashi qiziqtirsa, pastda keltirilgan qisqacha tarixga ko'z tashlang.

Ma'lumotlar bilan ishlash haqida qisqacha tarix.

Qachonlardir malumotlarni boshqarish tizimi birinchi bo'lib yaratildi, bular Oracle va DB2 IBM uchun, istalgan dasturchiga qaysidir darajada ma'lumotlardan foydalanishi zarur, ular har bir malumotlni boshqarish tizimlari uchun funksiya yaratishga ehtiyoj sezdilar. Har qaysi tizimlar uchun o'zining funksiyalar kutubxonasi mavjud bo'lib, bular Oracle uchun Oracle Call Interface (OCI), Sybase SQL Server (keyinchalik uni Microsoft sotib olgan) uchun DBLib. Dasturlarning ma'lumotlardan juda tez foydalanish va ular ma'lumotlar bazalari bilan to'g'ridan to'g'ri bog'lanish imkonini beradi. Ammo dasturchi har bir ma'lumotlar bazalari uchun ularning kutubxonalarini bilishi, ularda ishlab biror natija ola bilishi zarur edi. Shu sababli ma'lumotlarni boshqarish tizimlaridan foydalanib dastur yozish haddan tashqari murakkab masala edi. Bu shundan dalolat beradiki, agar shirkat(daturiy mahsulotlardan foydalanuvchi) oldingi ma'lumotlar bazasi o'rniga bosqasidan foydalanmoqchi bo'lsa u barcha ilova dasturlarini(приложения) qaytadan yozishiga to'g'ri kelar edi.

Bu muammo ODBC (Open Database Connectivity) yaratilishi bilan o'z yechimini topdi.ODBC ustida Microsoft boshqa shirkatlar bilan birgalikda 90-yildan boshlab ishlay boshladi. ODBC dasturlovchilarga barcha funksiyalari

yordamida xoxlagan ma’lumotlar bazasi tizimidan foydalanish imkonini beradi. Bu funksiyalar ma’lumotlar bazasidan foydalanayotganda aynan shu ma’lumotlar bazasini boshqarish tizimining drayverlarida translatsiya qilinadi.
Bu orqali koplab muammolar yechildi, patentlangan ma’lumotlar baza kutubxonalari bilan islovchi dasturchilar endi bir turdagi funksiyalar to’plamidan foydalanishadi(ODBC funksiyalaridan); agarda shirkat o’zining ma’lumotlar bazasini boshqarish tizimini o’zgartirsa ular faqat kodning ma’lumotlar bazasini bog’lovchi qisminigina o’zqartirishadi xolos. Keyinchalik yana bir muammo paydo bo’ldi. Bir qancha ofeslar shirkat ishini bir joydan turib katta hajmdagi ma’lumotlarni qog’ozlarsiz elektron ko’rinishda saqlash haqida orzu qila boshladilar yani elektron kutubxona, web sahifalar, Project 2000 fayllari va boshqalar. ODBC an’anaviy ma’lumotlar bazalari bilan juda yaxshiishlaydi, lekin ma’lumotlar tipini birinchisidan boshqasiga o’tkazishm satr bo’yicha tartiblash va ustun bo’yicha jolashtirishda bazida umuman asosiy tuzilish bilan mutanosiblikni yo’qotardi.

OLE DB paydo bo’lganidan so’ng bu muammo o’z yechimini topdi. OLE DB texnologiyasining ishlash usuli ODBCga o’xshash, ma’lumotlar bazasidan ma’lumotlar abstrak qatlam orqali ilova dasturga beriladi, faqat ruxsat berilgan ma’lumotlargina. Foydalanuvchi ilova dasturi ma’lumotlar manbasiga(источник) an’anaviy ma’lumotlar bazasi yoki xohlagan joyda saqlanayotgan ma’lumotlar OLE DB uzatuvchisi orqali yetkaziladi. Ma’lumotlar ilova dasturga jadval shaklida uzatiladi — bunda ular xuddi ma’lumotlar bazasi bilan ishlayotganday bo’ladi. OLE DB ma’lumotlarni uzatishda ODBC drayverlaridan foydalar, balki ma’lumotlar ba’zasidan foydalanishda ODBCni qo’llar. Keyingi bo’limlarda ADO.NET OLE DB va ODBCning ustida qurilgan va ularga juda o’xshash ekanligiga ishonch hosil qilasiz.

Ma’lumotlar bilan ishlashning keyingi avlod texnikasi .NET davri paydo bo’lishi bilan ActiveX Data Objects (ADO) yaratildi. ADO OLE DBdan yuqori tabaqa bo’lib OLE DBning ustiga qurilgan. Yuqori bosqichli dasturlash tillarida

dastur yozishda Visual Basic da OLE DBdan foydalanib ma’lumotlarga ishlov beriladi.

ADO.NET texnologiyasi .NET Framework 1.0 bilan bir vaqtda yaratildi va .NET har xil versiyalari uchun ADO.NETning ma’lumotni yetkazuvchilari qo’shilib rivojlantirildi, sekin asta mukammallashtirildi. NET Framework paydo bo’lishi bilan LINQ to SQL o’zining birinchi ma’lumotlar bilan ishlash uchun muhim “siljish paradigmlari”ni namoyish qildi. ADO.NET endigi rivojlanishni LINQ bilan birgasi takominlashtirdi. Kutilmaganda yangi ADO.NET for Entities mahsulot yaratildi u Visual Studio 2008 dan boshlab ishlatila boshlandi.

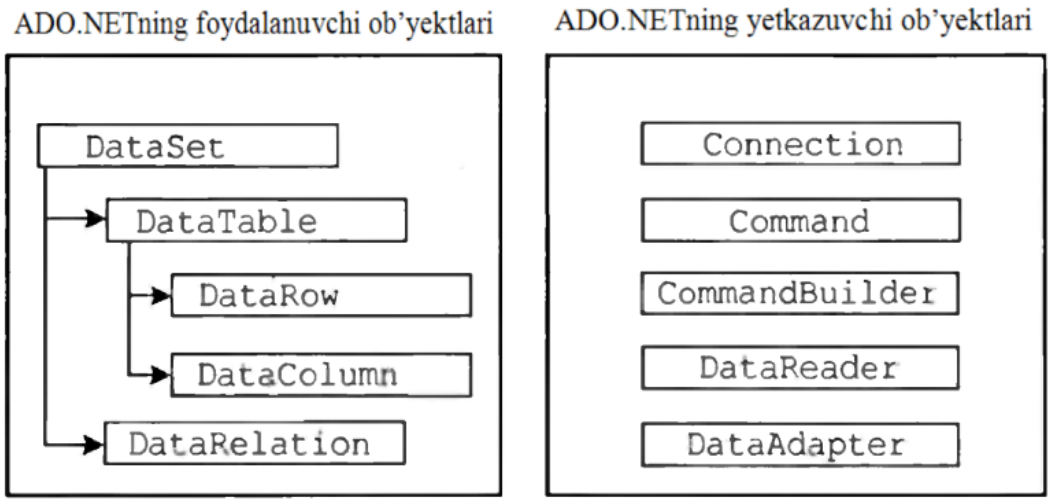
ADO.NET loyihalashning maqsadi.

Quyda ADO.NET loyihalashning asosiy maqsadlarini qisqacha keltiramiz.

- Relyatsiyon va relyatsiyon bo’lmagan malumotlar bilan oddiy bog’lanish.
- Oldingi texnologiyalarga nisbatan koproq ma’lumotlarni qamrab olishi va keng qamrovli qo’llanilishi.
- Internet uchun yuqori darajadagi ilova dasturlarda qo’llash.
- XML ma’lumotlar va relyatsion ma’lumotlardan bir xil foydalanish.

2.2.2 ADO.NET sinflari va ob’yektlari haqida.

2.1-chizma diogrammasida ADO.NETning ma’lumotlar bilan ishlovchi sinflari keltirilgan.



2.1-chizma . ADO.NETning ma’lumotlar bilan ishlovchi sinflari.

.NETning ma’lumotlar bilan ishlovchi sinflari quyidagi ADO.NETning foydalanuvchi ob’yektlari sinflari ADO.NETning yetkazuvchi ob’yektlari sinflaridan tashkil topgan.

- ADO.NETning yetkazuvchi ob’yektlari ma’lumotlar manbasidagi ma’lumotlar qaysi tipda bo’lsa o’sha tipda o’qish va yozishni ta’minlaydi.
- ADO.NETning foydalanuvchi ob’yektlari manipulyatsiyani qo’llash orqali ma’lumotlarni xotiradan o’qiydi.

ADO.NETning yetkazuvchi ob’yektlari har doim faol bo’lanishni ta’lab qiladi. Agarda siz faqatgina ma’lumotlardan o’qish uchun foydalansangiz bu ob’yektlardan tashqari ma’lumotlarni xotiradan o’qish orqali foydalanuvchi ob’yektlaridan ham foydalanishingiz mumkin. Agarda siz ma’lumotlar manbasining ma’lumotlarini o’zgartirmoqchi yoki yangilamoqchi bo’langiz unda bu ma’lumotlarni ma’lumotlar manbasiga yetkazuvchi ob’yektlaridan foydalanib kirita olasiz. Foydalanuvchi ob’yektlaridan doimiy bog’lanish bo’lmagan hallarda, ma’lumotlar bazasi bilan bog’lanishlar uzilib qolgan vaqtlarda ham ma’lumotlar bilan ishlashga to’g’ri keladi; shunda xotirada saqlangan ma’lumotlardan foydalaniladi.

ADO.NETning yetkazuvchi ob’yektlari.

ADO.NETning yetkazuvchi ob’yektlarida (provider objects) .NETning quyidagi ma’lumotlarni yetkazuvchilari o’z ichiga olgan. Masalan, oldindan mavjud bo’lgan OLE DBdagi OleDbConnection deb nomlanagan yetkazuvchining bog’lash obyekti, .NET SQL Serverdan esa SqlConnection sinfini o’z ichiga olgan.

Connection ob’yekti — ADO.NETning boshqa obyektlariga nishbattan ko’p foydalaniladigan ob’yekt bo’lib, u foydalanuvchiga ma’lumotlar manbasi bilan bazaviy bog’lanish imkonini beradi. Agarda siz ma’lumotlar bazasidan foydalanuvchining ko’rsatgich nom va kalit so’zni tarmoq serveridan o’chirgan bo’lsangiz unda boshqa foydalanuvchini ro’yxatdan o’tkazib boglanishni

qaytatdan o’rnatasiz va Connection shu o’zgarishga tegishli qismini o’zgartirasiz.

Command ob’yektida siz foydalanayotgan ma’lumotlar manbasining buyruqlaridan ishlatishda foydalanasiz, Masalan, SELECT * FROM Customers bu yerda so’rovda Customers jadvali ma’lumotlaridan foydalanildi. Bu ob’yekt SQL Server ushun SqlCommand, ODBC uchun OdbcCommand va OLE DB uchun OleCommand ko’rinishida foydalaniladi.

CommandBuilder ob’yektidan SQL buyruqlari yozishda foydalaniladi va yozilgan so’rovlar faqatgina bitta jadval ma’lumotlaridan foydalanib bajariladi. Biz bu ob’yektni ma’lumotlarni yangilashni o’rgana turib batafsil ko’rib chiqamiz. Bu ob’yekt SQL Server ushun SqlCommandBuilder, ODBC uchun OdbcCommandBuilder va OLE DB uchun OleCommandBuilder ko’rinishida foydalaniladi.

Bu ob’yektdan faqatgina bir yo’nalishdagi ma’lumotlar oqimini o’qishda va ko’p buyutmachilarga ma’lumotlar manbasidagi ma’lumotlarni tezroq yetkazishda foydalaniladi. Bu ob’yektdan shunchaki ma’lumotlarni o’qishda yuqori darajadagi unumdorlikni ta’minlaydi. Bu ob’yekt SQL Server ushun SqlDataReader, ODBC uchun OdbcDataReader va OLE DB uchun OleDataReader ko’rinishida foydalaniladi.[8]

Bu ob’yektdan ko’p maqsadlarda har xil turdagi operatsiyalarni bajarishda, ma’lumotlarni o’zgartirish va yangilash, DataSet ob’yektini ma’lumotlar bilan to’ldirish va bir qancha operatsiyalarni bajarishda foydalanildi. Bu ob’yekt SQL Server ushun SqlDataAdapter, ODBC uchun OdbcDataAdapter va OLE DB uchun OleDataAdapter ko’rinishida foydalaniladi.

ADO.NETning foydalanuvchi ob’yektlari.

Bu ob’yekt ADO.NETning ma’lumotlar manbasi bilan bo’lanish uzulib qolganda ham ma’lumotlardan foydalana olish imkoniyati. Ularda ma’lumotlar ma’nbasi bilan bog’lanish yo’q va System.Data ismlar fazosida joylashgan.

DataSet ob’yektidan turli xil ko’rinishlarda foydalanish mumkin. DataSet

unga berilayotgan jadval ma’lumotlarini yaxlid holda qolipga solish imkonini beradi. Masalan, Customers, Orders va Products alohida jadnvallar bo’lishi mumkin DataSet ulardan buyurtmachiga buyurtmachilar va mahsulotlar jadvallarida faqat uning sirkatiga tegishli buyurtmalarni namoyish qilish imkonini beradi. Uning DataTable va DataRelation ob’yektlari mavjud.

Bu ob’yekt DataSetdagi Customers, Orders yoki Products jadvallardan birini namoyish qiladi. DataTable obyektini qator va ustunlaridan foydalanish orqali ishlatish mumkin.[9]

- DataColumn ob’yekti. Jadvalning ustuni OrderID yuki CustomerName ko’rinishida bo’lishi mumkin.
- DataRow ob’yekti. Jadvalning bir qatoridagi ma’lumotlar CustomerID, name, address bo’lishi mumkin.

DataRelation ob’yekti ikkita jadvallarning barcha ustunlari orasidagi aloqa o’rnatishni ta’minlaydi. Masalan, Orders jadvalidagi CustomerID ustuni ma’lumotlari buyurtmachilarning takrorlanmas raqamlaridan iborat bo’lsin. DataRelation ob’yekti Customers va Orders jadvallarini CustomerID ustuni ma’lumotlari asosida birlashtirishi mumkin.

System.Data ismlar fazosidan foydalanish.

Birinchilardan bo’lib C#dasturlash tilida ADO.NETdan foydalanish uchun System.Data ismlar fazosiga ssilka (yo’nalish) o’rnatish kerak, chunki bu yerda ADO.NETning barcha sinflari joylashgan. Yozilajach barcha dasturlarda ADO.NET foydalanish uchun System.Data ismlar fazosiga ssilkasi using drektivasidan keyin quyidagicha yozilishi kerak:

using System.Data

Agarda siz .NETning anniq bir ma’lumotlar manbasi yetkazuvchisidan foydalanmoqchi bo’lsangiz unda bu ssilkani qudagilarning biri ko’rinishida o’rnatasiz.

.NET SQL Server ma’lumotlar yetkazuvchisi uchun

using System.Data.SqlClient;

OLE DB ma’lumotlar yetkazuvchisi uchun

using System.Data.OleDB;

ODBC.NET ma’lumotlar yetkazuvchisi uchun

using System.Data.Odbc;

Xulosa

Bu bobda C# dasturlash tili. Uning imkoniyatlari boshqarish strukturalari, mantiqiy amallar, massivlar, funksiyalar haqida ma’lumotlar. C# dasturlash tilining ADO.NET texnologiyasining qisqacha tarixi, ob’yektlari haqida ma’lumotlar keltirilgan.

III BOB. Dasturiy tizimni yaratish asoslari.

3.1 Kafedra o’quv yuklamasini qo’llab quvvatlovchi dasturiy tizim yaratish.

Hozirgi kunda oliy ta’lim muassasasidagi barcha kafedralarda kafedra tomonidan kafedraning o’quv yuklamasi va dars taqsimoti tayyorlanadi. Buni albatta namunaviy o’quv reja, ishchi o’quv reja va mavsumiy o’quv rejalar asosida tayyorlashadi. Bu yerda namunaviy o’quv reja oliy ta’lim vazirligi tomonidan har bir ta’lim yo’nalishi va ta’lim mutaxassisligi uchun tasdiqlab beriladi. Namunaviy o’quv reja asosida har bir kafedra o’zining ta’lim yo’nalishi uchun ishchi o’quv rejani tayyorlaydi. Ishchi o’quv rejalar asosida o’quv ishlari bo’yicha dekan muovinlari mavsumiy ishchi o’quv rejalarni tayyorlashadi. Kafedralar tomonidan tayyorlanadigan kafedra o’quv yuklamasi va dars taqsimoti turli xil shakllarda bo’lishi mumkin, qandaydir kamchiliklarga ega bo’lishi mumkin yoki uni tayyorlash jarayoni murakkablik tug’dirishi mumkin. Yuqorida keltirilgan qiyinchiliklarni va kamchiliklarni bartaraf etish maqsadida ushbu dasturiy tizim yaratildi.

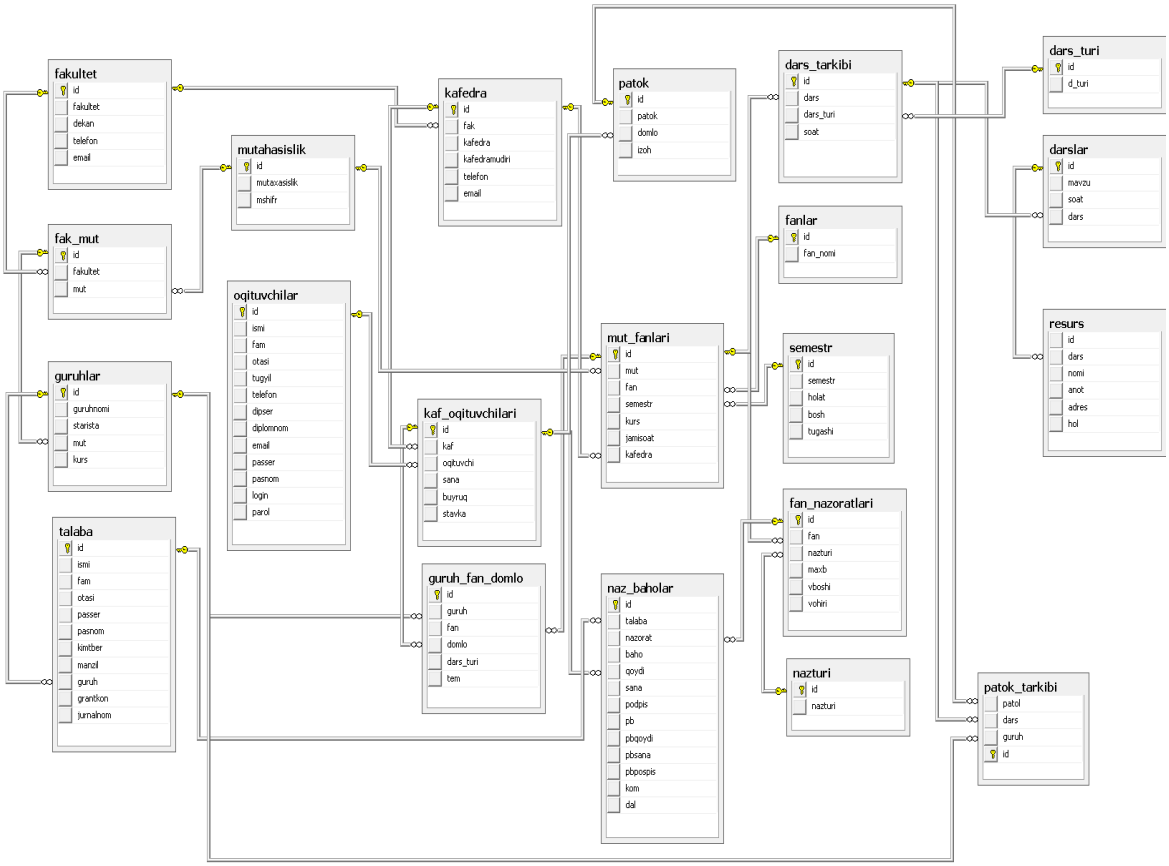
Kafedra o’quv yuklamasi va dars taqsimoti uchun kerak bo’ladigan barcha ma’lumotlarni o’rganib chiqib, uni ma’lumotlar bazasini loyihasini SQL Serverda quyidagicha tayyorlandi (3.1-chizma).

Qo’yilgan masalani loyihasini tuzishda quyidagi 14 ta jadvaldan foydalandik.

1. fakultet. 2. kafedra. 3. mutaxassislik 4. dars_turi. 5. mut_fanlari. 6. fanlar.
7. semestr 8. kaf_oqituvchilari. 9. oqituvchilar. 10. potok. 11. darslar. 12. dars tarkibi. 13. dars turi 14. potok tarkibi.

Ma’lumotlar bazasini ER diagrammasidan ko’rinadiki har bir jadvalimizni maydoni kerakli jadval bilan bog’langan, bu esa bizlarga ma’lumotlarni saqlashni va ular ustida bajariladigan amallarni tez amalga oshirishni qulaylashtiradi.[12]

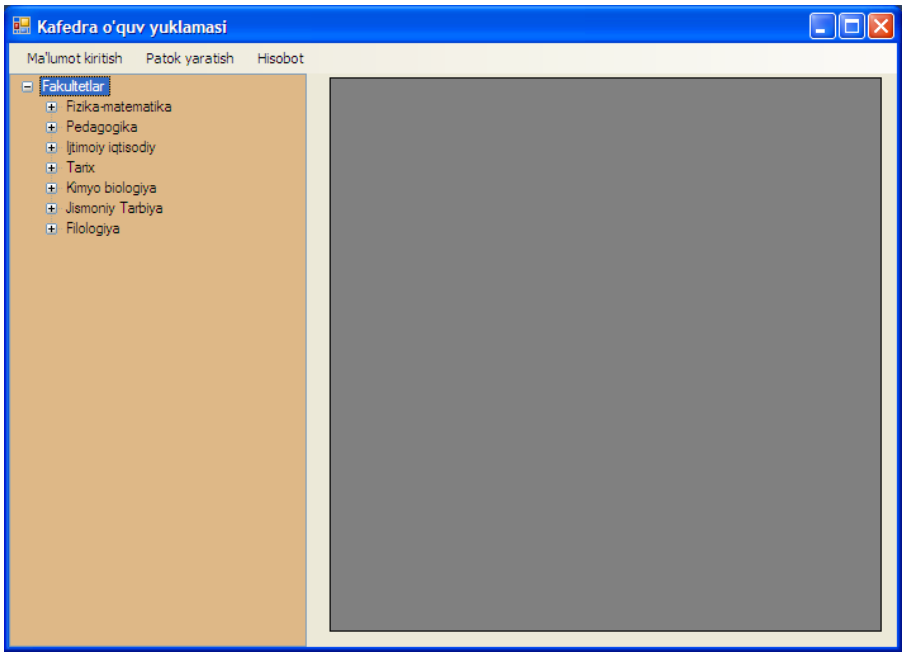
Yuqorida keltirilgan barcha jadvallaridan foydalanib ixtiyoriy kafedrani o’quv yuklamasini tayyorlab berish mumkin.



3.1-chizma. Qo’yilgan masalaning ER modeli.

Endi kafedra o’quv yuklamasini qo’llab quvvatlovchi dasturiy tizimni yaratish jarayoni va uning funksional imkoniyatlarini ko’rib chiqamiz.

Dasturiy tizimni umumiy ko’rinishi quyidagicha (3.2-chizma).



3.2-chizma. Dasturiy tizimni ko’rinishi.

Kafedra o’quv yuklanmasi ma’lum bir o’quv yili uchun tayyorlanadi. Shu sababdan oldin o’quv yilini bazaga qo’shib qo’yish talab qilinadi. Uni bazaga qo’shish quyidagicha amalga oshiriladi. Menyular satridan “Ma’lumot kiritish” bo’limining “Semestr” bandini tanlanadi.

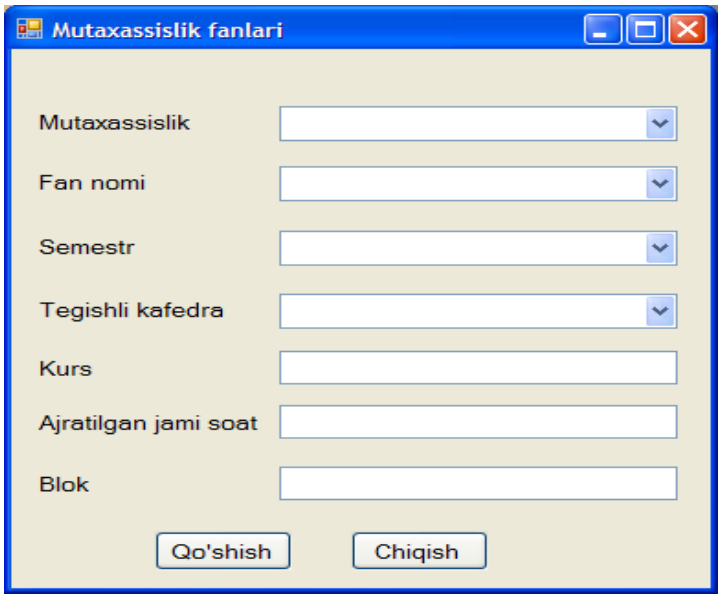
3.3-chizma. Semestr oynasining ko’rinishi.

3.3-chizmada keltirilgan oynada so’ralgan barcha ma’lumotlar to’ldirilgach “Qo’shish” tugmasi bosiladi. Obye’ktlar ADO texnologiyasida qandaydir bir hodisa yuz derganda bajariladi. Shuning uchun “Qo’shish” tugmasini bosganimizdan so’ng barcha ma’lumotlar bazaga qo’shib qo’yiladi. “Qo’shish” tugmasi bosilganda bajariladigan amallar ketmaketligini ko’raylik:

```
private void button1_Click(object sender, EventArgs e){ try{ string satr =
"insert into semestr(semestr, holat, bosh, tugashi)"; string[] bosh, tugash; bosh =
new string[2]; tugash = new string[2]; bosh[0] =
comboBox4.SelectedItem.ToString() + "." +
comboBox3.SelectedItem.ToString() + "." +
comboBox6.SelectedItem.ToString() + " 0:00:00"; bosh[1] =
comboBox13.SelectedItem.ToString() + "." +
comboBox14.SelectedItem.ToString() + "." +
comboBox12.SelectedItem.ToString() + " 0:00:00"; tugash[0] =
comboBox7.SelectedItem.ToString() + "." +
comboBox8.SelectedItem.ToString() + "." +
comboBox5.SelectedItem.ToString() + " 0:00:00"; tugash[1] =
```

comboBox10.SelectedItem.ToString() + "." +
comboBox11.SelectedItem.ToString() + "." +
comboBox9.SelectedItem.ToString() + " 0:00:00"; for (int i = 1; i < 3; i++){
yor.satr = satr + " values('"+comboBox2.SelectedItem.ToString()+" "+i+"
semestr','"+comboBox1.SelectedItem.ToString()+"','"+bosh[i-1]+"','"+tugash[i-
1]+"')"; yor.bazaUstidaAmal(); } Close(); } catch { MessageBox.Show("Barcha
to\'g\'ri ma\'lumotlarni mantiqan to\'g\'ri kiritishingiz shart!"); } }

Endigi vazifa mut_fanlari jadvalini to'ldirish. Buning uchun biz yana menyular satridan "Ma'lumot kiritish" bo'limining "Mutaxassislik fanlari" bandini tanlaymiz.



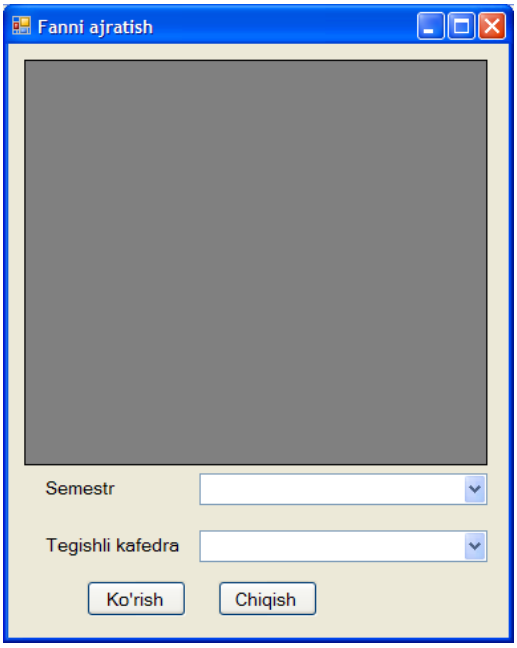
3.4-chizma. Mutaxassislik fanlari oynasining ko'rinishi.

Bu oynada ham so'ralgan ma'lumotlar kiritilgach "Qo'shish" tugmasi bosiladi. "Qo'shish" tugmasi bosilganda bajariladigan amallar ketmaketligini ko'raylik:

private void button1_Click(object sender, EventArgs e){ string satr; int mutId,
fanId, semestrId, kafedraId; yor.satr = "Select id from mutahasislik where
mutaxassislik='" + comboBox1.SelectedItem.ToString() + "'"; mutId =
yor.son(); yor.satr = "Select id from fanlar where fan_nomi='" +
comboBox2.SelectedItem.ToString() + "'"; fanId = yor.son(); yor.satr = "Select
id from semestr where semestr='" + comboBox3.SelectedItem.ToString() + "'";
semestrId = yor.son(); yor.satr = "Select id from kafedra where kafedra='" +

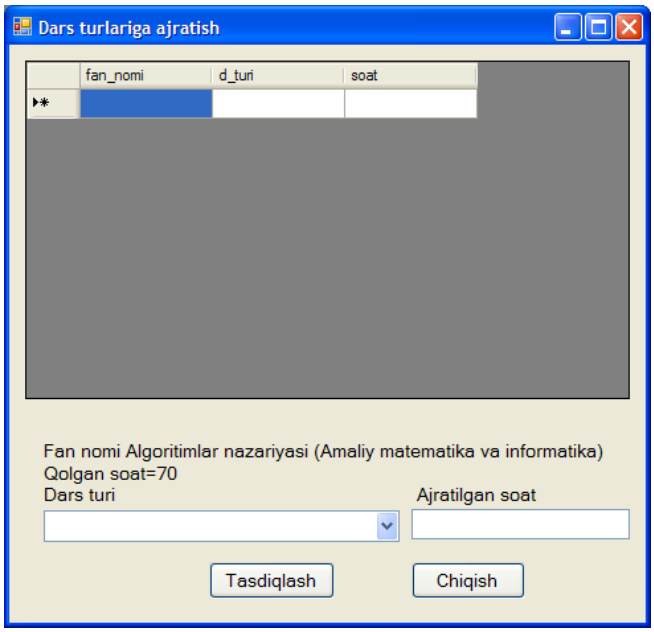
comboBox4.SelectedItem.ToString() + """; kafedraId = yor.son(); satr = "insert into mut_fanlari(mut, fan, semestr, kafedra, kurs, jamisoat,blok)"; satr = satr + " values(" + mutId + "," + fanId + "," + semestrId + "," + kafedraId + "," + textBox1.Text + "," + textBox2.Text + "," + textBox3.Text + ")"; yor.satr = satr; yor.bazaUstidaAmal(); Close(); }

mut_fanlari jadvali to'dirilgach shu mutaxassislik fanini dars turlariga ajtratish talab etiladi. Bunada biz dars_tarkibi jadvaliga malumot kiritamiz. Bu quyidagicha amalga oshiriladi. menyular satridan "Potok yaratish" bo'limining "Fanni ajratish" bandini tanlaymiz va quyidagi oyna ochib beriladi.



3.5-chizma. Fanni ajratish oynasining ko'rinishi.

Bu yerdagi ochiluvchi ro'yxatlar yordamida semestr va kafedra tanlangach "Ko'rish" tugmasi bosiladi. Oynada shu kafedraning belgilangan semestrdagi barcha fanlari ro'yxati va shu fanlardan ajdratilgan dars turlari soni namoyish qilinadi. Qo'shimcha qilish uchun fan tanlanadi va tanlangan fan ustida sichqonchaning o'ng tugmasi bosiladi, konteksli menyudan "Ajratish" bandi tanlanadi.Shundan so'ng ushbu "Dars turiga ajratish" oynasi ochib beriladi.



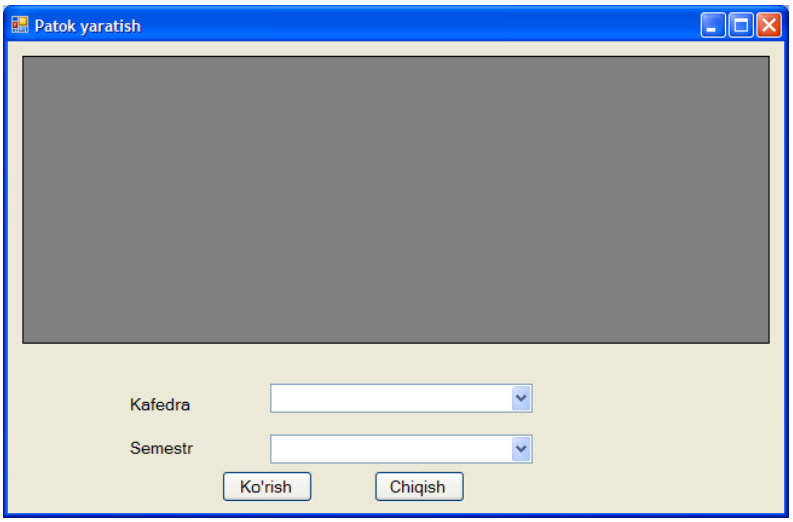
3.6-chizma. Dars turiga ajratish oynasining ko’rinishi.

Bu oynada qulaylik yaratish uchun mutaxassislik fani nomi va qavsda shu mutaxassislik undan so’ng qolgan soat yani boshlanishda jami soat keyinchalik mutaxassislik fani dars turlariga ajratilgach qolgan soat namoyish etiladi. Dars turi ochiluvchi ro’yxatidan esa dars turi yani ma’ruza, amaliyot, seminar va boshqa dars turi tanlanadi va shu dars turiga ajratilgan soat kiritilgach “Tasdiqlash” tugmasi bosiladi. “Tasdiqlash” tugmasi bsilgandan so’ng bajariladigan amallar ketma-ketligi quyidagicha:

```
private void button1_Click(object sender, EventArgs e) { int darsTuri=0, potokSoni=1; if (!(textBox2.Text.Equals("")) && !(comboBox2.Text.Equals(""))) { try { soat = int.Parse(textBox2.Text); List<int> listMutfanId=new List<int>(); yor.satr = "select dars_turi from dars_tarkibi where dars=" + mutFanId; listMutfanId.AddRange(yor.butunSonliMassiy()); yor.satr = "select id from dars_turi where d_turi='" + comboBox2.SelectedItem.ToString() + "'"; darsTuri = yor.son(); if (!listMutfanId.Contains(darsTuri)) { yor.satr = "insert into dars_tarkibi(dars, dars_turi, soat) values ('" + mutFanId + "', '" + darsTuri + "'," + soat + ")"; yor.bazaUstidaAmal(); OnLoad(e); } else MessageBox.Show("Siz oldin aniqlangan dars turini qayta aniqlashga urindingiz!\nShu sababdan o\'zgarish bazaga kiritilmadi."); } catch{MessageBox.Show(" Ma\'lumotlarni
```

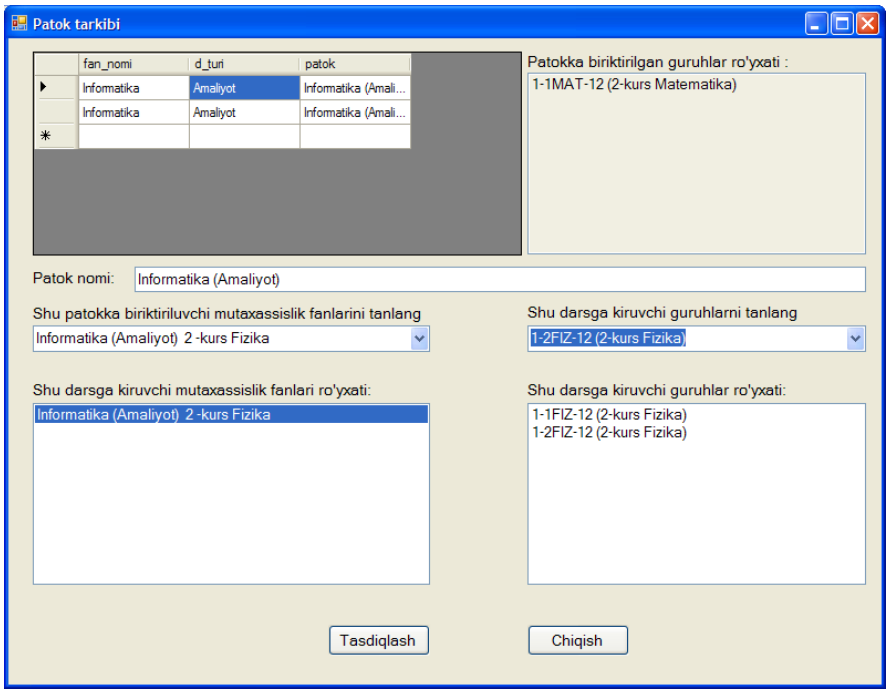
to\'ri kiritishingizni so\'raymiz! ");} } else MessageBox.Show(" Barcha qiymat kiritiluvchi maydonlar to\'ldirilishi lozim! "); }

Endi aniqlangan mutaxassisilik fanlarining dars turlari aniqlanganidan so’ng potoklar yaratish imkoniga ega bo’ldik. Potok yaratish uchun menyular satridan “Potok yaratish” bo’limining “Potok yaratish” bandini tanlaymiz va quyidagi oyna ochib beriladi.



3.7-chizma. Fanni ajratish oynasining ko’rinishi.

Bu yerdagi ochiluvchi ro’yxatlar yordamida semestr va kafedra tanlangach “Ko’rish” tugmasi bosiladi. Oynada shu kafedraning belgilangan semestrdagi barcha fanlari ro’yxati va shu fanlardan ajdratilgan dars turlari, fan o’tiladigan mutaxassislik, dars turlaridan yaratilgan patiklar soni namoyish qilinadi. Qo’shimcha qilish uchun dars turlarida biri tanlanadi va tanlangan dars tur ustida sichqonchaning o’ng tugmasi bosiladi, konteksli menyudan “Potok yaratish” bandi tanlanadi. Shundan so’ng ushbu “Potok tarkibi” oynasi ochib beriladi.



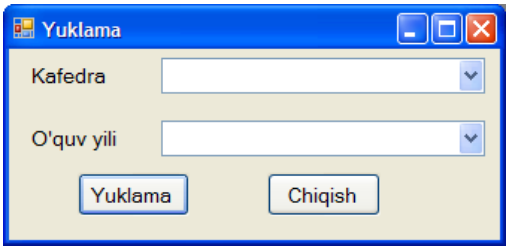
3.8-chizma. Potok tarkibi oynasining ko’rinishi.

Bu oynada qulaylik yaratish uchun oynaning yuqori chap tomon burchagida agarda mavjud bo’lsa mutaxassislik fani nomi qaysiki tanlagan dars turiga mos, dars turi, potok nomi keltirilgan. Yuqori o’ng tomon burchagida esa chap tomondan tanlangan potokka biriktilgan guruhlar ro’yxati namoyish etiladi. Potok nomi maydonida esa yaratiluvchi potok nomi tavsiya qilinadi uni o’zgartirish mumkin. Mutaxassislik fanlarini tanlang ochiluvchi ro’yxatda esa fan nomiga mos Mutaxassislik fanlari ro’yxati keltirilgan. Bu yerdan tanlangan fan mutaxassislik fanlari ro’yxatida namoyish etiladi. Barcha ma’lumotlar to’ldirilgach “Tasdiqlash” tugmasi bosiladi. “Tasdiqlash” tugmasi bosilgandan so’ng bajariladigan amallar ketma-ketligi quyidagicha:

```
private void button1_Click(object sender, EventArgs e) { foreach (int d in mutFanlist) { yor.satr = "Select dars_tarkibi_id from surovPotokUchun where mut_fan_id="+d+" and dars_turi_id="+darsturiId+" and semestr_id="+semestrId+" and kafedra_id="+kafedraId; darsTarkibiId = yor.son(); yor.satr = "Select DISTINCT guruh_id from surovMutFanGuruh where mutfan_id=" + d; vaqtincha.Clear(); vaqtincha.AddRange(yor.butunSonliMassiy()); string ssss = ""; foreach (int x in guruh){ ssss = ssss + x; if (vaqtincha.Contains(x)){ if (darsturiId >= 5 &&
```

(darsturiId <= 12)){ yor.satr="select fio from surovTalabalar where guruh_id="+x; guruhTalabalar.Clear(); guruhTalabalar.AddRange(yor.satrliMassiy()); foreach (string y in guruhTalabalar){ yor.satr = "Select max(id) from potok"; potokId = yor.son(); potokNomi = textBox1.Text+" ["+x+" -guruh]"+" "+y+"{"+potokId+"}"; potokNomi2 = "insert into potok(potok) values('" + potokNomi + "')"; yor.satr = potokNomi2; yor.bazaUstidaAmal(); yor.satr = "Select id from potok where potok='" + potokNomi + "'"; potokId = yor.son(); s = "(" + potokId + ", " + darsTarkibiId + ", " + x + ")"; yor.satr = satrValue + s; yor.bazaUstidaAmal();} potoklarYaratish = true; }else{if (!potokYaratish){ yor.satr = "Select max(id) from potok"; potokId = yor.son(); potokNomi = textBox1.Text + " {" + potokId + "}"; potokNomi2 = "insert into potok(potok) values('" + potokNomi + "')"; yor.satr = potokNomi2; yor.bazaUstidaAmal(); potokYaratish = true; } yor.satr = "Select id from potok where potok='" + potokNomi + "'"; potokId = yor.son(); yor.satr=""; s = "(" + potokId + ", " + darsTarkibiId + ", " + x + ")"; yor.satr = satrValue + s;yor.bazaUstidaAmal(); } } } } if (potokYaratish){ MessageBox.Show("Potok va potok tarkibi muvofaqiyatli yaratildi!"); Close(); } else if (potoklarYaratish){ MessageBox.Show("Siz hozir bir nechata talaba ismiga mos potoklar va potoklar tarkibini yaratdingiz!"); Close(); } else MessageBox.Show("Potok va potok tarkibi yaratilmadi."); }

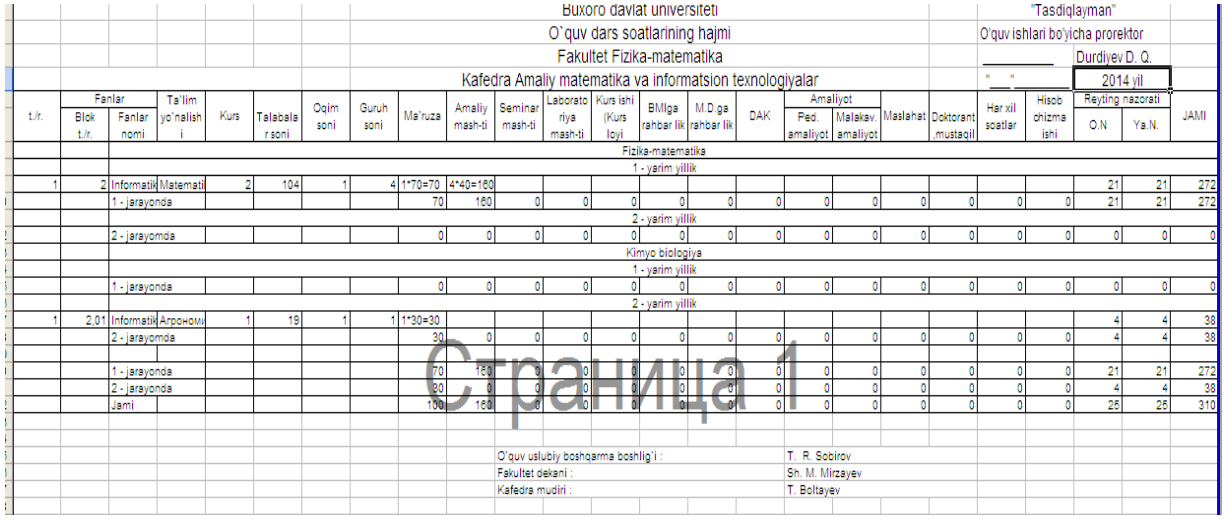
Kafedraga tegishli barcha potoklar yaratilgach kafedra yuklamasini tayyorlash mumkin bo'ladi. Buning uchun biz menyular satridan "Hisobot" bo'limining "Yuklama" bandini tanlaymiz va quyidagi oyna ochib beriladi.



3.9-chizma. Yuklama oynasining ko'rinishi.

Bu oynadagi ochiluvchi ro'yxatlardan kerakli ma'lumotlar tanlangach "Yuklama" tugmasi bosiladi va tanlangan kafedraning tegishli o'quv yilidagi

yuklamasiga tegishli barcha ma’lumotlar Microsoft Office Excel dasturi ishga tushurilib Microsoft Office Excelning hujjatiga yuboriladi(3.10-chizma).



Buxoro davlat universiteti — "Tasdiqlayman"
O`quv dars soatlarining hajmi — O'quv ishlari bo'yicha prorektor
Fakultet Fizika-matematika — Durdiyev D. Q.
Kafedra Amaliy matematika va informatsion texnologiyalar — 2014 yil

t./r.	Fanlar		Ta'lim yo'nalishi	Kurs	Talabalar soni	Oqim soni	Guruh soni	Ma'ruza	Amaliy mash-ti	Seminar mash-ti	Laboratoriya mash-ti	Kurs ishi (Kurs loyi	BMIga rahbarlik	M.D.ga rahbarlik	DAK	Amaliyot		Maslahat	Doktorant,mustaqil	Har xil soatlar	Hisob chizma ishi	Reyting nazorati		JAMI
	Blok t./r.	Fanlar nomi														Ped. amaliyot	Malakav. amaliyot					O.N	Ya.N.	
		Fizika-matematika																						
		1 - yarim yillik																						
1	2	Informatik	Matemati	2	104	1	4	1*70=70	4*40=160													21	21	272
		1 - jarayonda						70	160	0	0	0	0	0	0	0	0	0	0	0	0	21	21	272
		2 - yarim yillik																						
		2 - jarayomda						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		Kimyo biologiya																						
		1 - yarim yillik																						
		1 - jarayonda						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		2 - yarim yillik																						
1	2.01	Informatik	Агроном	1	19	1	1	1*30=30														4	4	38
		2 - jarayomda						30	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	38
		1 - jarayonda						70	160	0	0	0	0	0	0	0	0	0	0	0	0	21	21	272
		2 - jarayonda						30	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	38
		Jami						100	160	0	0	0	0	0	0	0	0	0	0	0	0	25	25	310

O'quv uslubiy boshqarma boshlig'i : T. R. Sobirov
Fakultet dekani : Sh. M. Mirzayev
Kafedra mudiri : T. Boltayev

3.10-chizma. Kafedra o’quv yuklamasi.

Kafedra o’quv yuklamasini tayyorlash dasturi kodi quyida keltirilgan.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Collections;
using dastur = Microsoft.Office.Interop.Excel;
namespace Sqldsan{ public partial class hajm : Form { yordamchiDastur yor = new
yordamchiDastur(); DataGridView date = new DataGridView(); public List<int>
lisFakultetId = new List<int>(); public List<int> lisMutfanId = new List<int>(); public
List<int> listMutFanPotok = new List<int>(); public List<int> listPotokMutfanlar = new
List<int>(); public List<int> listSemestr = new List<int>(); public List<string>
listSemestrNomi = new List<string>(); public List<int> listDarsTuri = new List<int>();
public List<int> listGuruhlar = new List<int>(); public string ShartSatri = "" , uquvYili = ""
, shartSemestr = "" , shartPotok = "" ,sinov = "" ,hajmMutaxassislik = "" ,hajmFanNomi =
"" ,Kafedra_Mudiri="" ,Prorektor="" ,Bulim_Boshliq ; public double hajmBlok = 0 ; public
DateTime dt = DateTime.Now; int yil = 0 ,sem = 0 ,hajmSoat = 0 ,hajmPotokSoni = 0
,hajmGuruhSoni = 0 ,hajmTalabaSoni = 0 ,hajmKurs = 0 ,hajmTartibNomer = 0
```

```
,hajmJamiSoat = 0 ,jamiSoat = 0 ,semeatrX = 0 ,fakultetX = 0 ; bool mutFanBor = false
,semestrTugadi = false ; public hajm() { InitializeComponent(); }
 private void hajm_Load(object sender, EventArgs e) { yor.satr = "Select semestr from
semestr"; listSemestrNomi.Clear(); listSemestrNomi.AddRange(yor.satrliMassiy()); yor.satr
= "Select semestr from semestr"; comboBox1.Items.Clear(); foreach(string seme in
listSemestrNomi) { sinov=seme.Substring(0,(seme.LastIndexOf('i')+1));
if(!comboBox1.Items.Contains(sinov)) comboBox1.Items.Add(sinov); }
comboBox2.Items.Clear(); yor.satr = "Select kafedra from kafedra";
comboBox2.Items.AddRange(yor.satrliMassiy()); }
private void button1_Click(object sender, EventArgs e){int semestrId, i = 7, kafedraId,
Maruza = 0, Amaliyot = 0, Semenar = 0, Labaratoriya = 0, Kurs_ishi = 0, BMIga_rahbarlik
= 0, MDga_rahbarlik = 0, DAK = 0, Ped_amaliyot = 0 , Malakaviy_amaliyot = 0, Maslahat
= 0, Doktorant_mustaqil = 0, Har_xil_soatlar = 0 , Hisob_chizma_ishi = 0, Oraliq_nazorat
= 0, Joriy_nazorat = 0, Yakuniy_nazorat = 0 , Mustaqil_talim = 0, fakultetK = 0,
hajmJamiSoatOxirgi = 0, MaruzaJami = 0 , AmaliyotJami = 0, SemenarJami = 0,
LabaratoriyaJami = 0, Kurs_ishiJami = 0 , BMIga_rahbarlikJami = 0, MDga_rahbarlikJami
= 0, DAKJami = 0, Ped_amaliyotJami = 0 , Malakaviy_amaliyotJami = 0, MaslahatJami =
0, Doktorant_mustaqilJami = 0 , Har_xil_soatlarJami = 0, Hisob_chizma_ishiJami = 0,
Oraliq_nazoratJami = 0 , Joriy_nazoratJami = 0, Yakuniy_nazoratJami = 0,
Mustaqil_talimJami = 0 , hajmJamiSoatOxirgi2 = 0, Maruzasem = 0, Amaliyotsem = 0,
Semenarsem = 0 , Labaratoriyasem = 0, Kurs_ishisem = 0, BMIga_rahbarliksem = 0,
MDga_rahbarliksem = 0 , DAKsem = 0, Ped_amaliyotsem = 0, Malakaviy_amaliyotsem = 0,
Maslahatsem = 0 , Doktorant_mustaqilsem = 0, Har_xil_soatlarsem = 0,
Hisob_chizma_ishisem = 0 , Oraliq_nazoratsem = 0, Joriy_nazoratsem = 0,
Yakuniy_nazoratsem = 0 , Mustaqil_talimsem = 0, fakultetKsem = 0,
hajmJamiSoatOxirgisem = 0 ; uquvYili =
comboBox1.SelectedItem.ToString().Substring(0,comboBox1.SelectedItem.ToString().IndexOf
('o')); yor.satr = "SELECT id FROM semestr WHERE semestr LIKE '"+uquvYili+"%'";
listSemestr.Clear(); listSemestr.AddRange(yor.butunSonliMassiy()); string surovShart="";
shartSemestr = ""; foreach( int s in listSemestr) { if (shartSemestr.Length > 0) shartSemestr
= shartSemestr + " or semestr_id="+s; else shartSemestr = shartSemestr + " and
(semestr_id=" + s; } shartSemestr = shartSemestr + ")"; string yul = @"d:\Dars
taqsimoti\Excel\Hajm.xls", yul2 = @"d:\Dars taqsimoti\Hajm.xls" ; if (File.Exists(yul2)){
File.Delete(yul2); File.Copy(yul,yul2); } else File.Copy(yul, yul2); dastur.Application excel =
new dastur.Application(); dastur.Workbook kitob = excel.Workbooks.Open(yul2);
```

dastur.Worksheet varoq = (dastur.Worksheet)kitob.Sheets[1]; dastur.Worksheet varoq2 = (dastur.Worksheet)kitob.Sheets[2]; excel.Visible = true; yor.satr = "Select id from kafedra where kafedra='" + comboBox2.SelectedItem.ToString() + "'"; kafedraId = yor.son(); yor.satr = "Select DISTINCT fakultet_id from surovHajm where (kafedra_id=" + kafedraId + ")"+shartSemestr+" group by fakultet_id"; lisFakultetId.Clear(); lisFakultetId.AddRange(yor.butunSonliMassiy()); lisFakultetId.Sort(); varoq.get_Range("H3", "R3").Name = ("HRH3"); dastur.Range fakultetNomi = varoq.get_Range("HRH3", Type.Missing); yor.satr = "Select fak from kafedra where id=" + kafedraId; int fakultetId = yor.son(); yor.satr = "Select fakultet from fakultet where id=" + fakultetId; fakultetNomi.Value = "Fakultet " + yor.matn(); varoq.get_Range("H4", "R4").Name = ("HRH4"); dastur.Range kafedraNomi = varoq.get_Range("HRH4", Type.Missing); kafedraNomi.Value = "Kafedra " + comboBox2.SelectedItem.ToString(); varoq.get_Range("w4", "x4").Name = ("wxw4"); dastur.Range yilNomi = varoq.get_Range("wxw4", Type.Missing); yil = int.Parse(dt.Year.ToString()); yilNomi.Value = yil+" yil"; semestrTugadi = false; semeatrX = listSemestr[0]; if (lisFakultetId.Count != 0) fakultetX = lisFakultetId[0]; hajmJamiSoatOxirgi = 0; foreach (int x in lisFakultetId){ fakultetK = x; if (fakultetX != x) { fakultetX = x; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("C" + i).Value = (sem-1)+" - jarayomda"; varoq.get_Range("I" + i).Value = Maruza; varoq.get_Range("J" + i).Value = Amaliyot; varoq.get_Range("K" + i).Value = Semenar; varoq.get_Range("L" + i).Value = Labaratoriya; varoq.get_Range("M" + i).Value = Kurs_ishi; varoq.get_Range("N" + i).Value = BMIga_rahbarlik; varoq.get_Range("O" + i).Value = MDga_rahbarlik; varoq.get_Range("P" + i).Value = DAK; varoq.get_Range("Q" + i).Value = Ped_amaliyot; varoq.get_Range("R" + i).Value = Malakaviy_amaliyot; varoq.get_Range("S" + i).Value = Maslahat; varoq.get_Range("T" + i).Value = Doktorant_mustaqil; varoq.get_Range("U" + i).Value = Har_xil_soatlar; varoq.get_Range("V" + i).Value = Hisob_chizma_ishi; varoq.get_Range("W" + i).Value = Oraliq_nazorat; varoq.get_Range("X" + i).Value = Yakuniy_nazorat; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi; MaruzaJami = MaruzaJami + Maruza; AmaliyotJami = AmaliyotJami + Amaliyot; SemenarJami = SemenarJami + Semenar; LabaratoriyaJami = LabaratoriyaJami + Labaratoriya; Kurs_ishiJami = Kurs_ishiJami + Kurs_ishi; BMIga_rahbarlikJami = BMIga_rahbarlikJami + BMIga_rahbarlik; MDga_rahbarlikJami = MDga_rahbarlikJami + MDga_rahbarlik; DAKJami = DAKJami + DAK; Ped_amaliyotJami = Ped_amaliyotJami + Ped_amaliyot; Malakaviy_amaliyotJami = Malakaviy_amaliyotJami + Malakaviy_amaliyot; MaslahatJami = MaslahatJami + Maslahat; Doktorant_mustaqilJami = Doktorant_mustaqilJami + Doktorant_mustaqil;

Har_xil_soatlarJami = Har_xil_soatlarJami + Har_xil_soatlar; Hisob_chizma_ishiJami += Hisob_chizma_ishi; Oraliq_nazoratJami += Oraliq_nazorat; Joriy_nazoratJami += Joriy_nazorat; Yakuniy_nazoratJami += Yakuniy_nazorat; Mustaqil_talimJami += Mustaqil_talim; hajmJamiSoatOxirgi2 += hajmJamiSoatOxirgi; if (sem - 1 == 1) { Maruzasem += Maruza; Amaliyotsem += Amaliyot; Semenarsem += Semenar; Labaratoriyasem += Labaratoriya; Kurs_ishisem += Kurs_ishi; BMIga_rahbarliksem += BMIga_rahbarlik; MDga_rahbarliksem += MDga_rahbarlik; DAKsem += DAK; Ped_amaliyotsem += Ped_amaliyot; Malakaviy_amaliyotsem += Malakaviy_amaliyot; Maslahatsem += Malakaviy_amaliyotsem; Doktorant_mustaqilsem += Doktorant_mustaqil; Har_xil_soatlarsem += Har_xil_soatlar; Hisob_chizma_ishisem += Hisob_chizma_ishi; Oraliq_nazoratsem += Oraliq_nazorat; Joriy_nazoratsem += Joriy_nazorat; Yakuniy_nazoratsem += Yakuniy_nazorat; Mustaqil_talimsem += Mustaqil_talim; hajmJamiSoatOxirgisem += hajmJamiSoat; } Maruza = 0; Amaliyot = 0; Semenar = 0; Labaratoriya = 0; Kurs_ishi = 0; BMIga_rahbarlik = 0; MDga_rahbarlik = 0; DAK = 0; Ped_amaliyot = 0; Malakaviy_amaliyot = 0; Maslahat = 0; Doktorant_mustaqil = 0; Har_xil_soatlar = 0; Hisob_chizma_ishi = 0; Oraliq_nazorat = 0; Joriy_nazorat = 0; Yakuniy_nazorat = 0; Mustaqil_talim = 0; hajmJamiSoatOxirgi = 0; hajmJamiSoat = 0; i++; } yor.satr = "Select fakultet from fakultet where id=" + x; string ustun1 = "C" , ustun2 = "Y" ; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range(ustun1 + i, ustun2 + i).MergeCells = true; varoq.get_Range(ustun1 + i, ustun2 + i).HorizontalAlignment = dastur.XlHAlign.xlHAlignCenter; varoq.get_Range(ustun1 + i, ustun2 + i).Borders.LineStyle = 1; varoq.get_Range(ustun1 + i, ustun2 + i).Name = (ustun1 + ustun2 + ustun1 + i); varoq.get_Range(ustun1 + i, ustun2 + i).Font.FontStyle = 1; dastur.Range a = varoq.get_Range(ustun1 + ustun2 + ustun1 + i, Type.Missing); a.Value2 = yor.matn(); i++; sem = 1; semeatrX = sem; foreach (int semestr in listSemestr) { if (sem != semeatrX) { semeatrX = sem; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("C" + i).Value = (sem-1)+ " - jarayonda"; varoq.get_Range("I" + i).Value = Maruza; varoq.get_Range("J" + i).Value = Amaliyot; varoq.get_Range("K" + i).Value = Semenar; varoq.get_Range("L" + i).Value = Labaratoriya; varoq.get_Range("M" + i).Value = Kurs_ishi; varoq.get_Range("N" + i).Value = BMIga_rahbarlik; varoq.get_Range("O" + i).Value = MDga_rahbarlik; varoq.get_Range("P" + i).Value = DAK; varoq.get_Range("Q" + i).Value = Ped_amaliyot; varoq.get_Range("R" + i).Value = Malakaviy_amaliyot; varoq.get_Range("S" + i).Value = Maslahat; varoq.get_Range("T" + i).Value = Doktorant_mustaqil; varoq.get_Range("U" + i).Value = Har_xil_soatlar; varoq.get_Range("V" + i).Value = Hisob_chizma_ishi; varoq.get_Range("W" + i).Value =

Oraliq_nazorat; varoq.get_Range("X" + i).Value = Yakuniy_nazorat; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi; MaruzaJami = MaruzaJami + Maruza; AmaliyotJami = AmaliyotJami + Amaliyot; SemenarJami = SemenarJami + Semenar; LabaratoriyaJami = LabaratoriyaJami + Labaratoriya; Kurs_ishiJami = Kurs_ishiJami + Kurs_ishi; BMIga_rahbarlikJami = BMIga_rahbarlikJami + BMIga_rahbarlik; MDga_rahbarlikJami = MDga_rahbarlikJami + MDga_rahbarlik; DAKJami = DAKJami + DAK; Ped_amaliyotJami = Ped_amaliyotJami + Ped_amaliyot; Malakaviy_amaliyotJami = Malakaviy_amaliyotJami + Malakaviy_amaliyot; MaslahatJami = MaslahatJami + Maslahat; Doktorant_mustaqilJami = Doktorant_mustaqilJami + Doktorant_mustaqil; Har_xil_soatlarJami = Har_xil_soatlarJami + Har_xil_soatlar;Hisob_chizma_ishiJami += Hisob_chizma_ishi; Oraliq_nazoratJami += Oraliq_nazorat; Joriy_nazoratJami += Joriy_nazorat; Yakuniy_nazoratJami += Yakuniy_nazorat; Mustaqil_talimJami += Mustaqil_talim; hajmJamiSoatOxirgi2 += hajmJamiSoatOxirgi; if (sem - 1 == 1) { Maruzasem += Maruza; Amaliyotsem += Amaliyot; Semenarsem += Semenar; Labaratoriyasem += Labaratoriya; Kurs_ishisem += Kurs_ishi; BMIga_rahbarliksem += BMIga_rahbarlik; MDga_rahbarliksem += MDga_rahbarlik; DAKsem += DAK; Ped_amaliyotsem += Ped_amaliyot; Malakaviy_amaliyotsem += Malakaviy_amaliyot; Maslahatsem += Malakaviy_amaliyotsem; Doktorant_mustaqilsem += Doktorant_mustaqil; Har_xil_soatlarsem += Har_xil_soatlar; Hisob_chizma_ishisem += Hisob_chizma_ishi; Oraliq_nazoratsem += Oraliq_nazorat; Joriy_nazoratsem += Joriy_nazorat; Yakuniy_nazoratsem += Yakuniy_nazorat; Mustaqil_talimsem += Mustaqil_talim; hajmJamiSoatOxirgisem += hajmJamiSoat; } Maruza = 0; Amaliyot = 0; Semenar = 0; Labaratoriya = 0; Kurs_ishi = 0; BMIga_rahbarlik = 0; MDga_rahbarlik = 0; DAK = 0; Ped_amaliyot = 0; Malakaviy_amaliyot = 0; Maslahat = 0; Doktorant_mustaqil = 0; Har_xil_soatlar = 0; Hisob_chizma_ishi = 0; Oraliq_nazorat = 0; Joriy_nazorat = 0; Yakuniy_nazorat = 0; Mustaqil_talim = 0; hajmJamiSoatOxirgi = 0; hajmJamiSoat = 0; i++; } varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range(ustun1 + i, ustun2 + i).MergeCells = true; varoq.get_Range(ustun1 + i, ustun2 + i).HorizontalAlignment = dastur.XlHAlign.xlHAlignCenter; varoq.get_Range(ustun1 + i, ustun2 + i).Borders.LineStyle = 1; varoq.get_Range(ustun1 + i, ustun2 + i).Name = (ustun1 + ustun2 + ustun1 + i);varoq.get_Range(ustun1 + i, ustun2 + i).Font.FontStyle = 1;dastur.Range b = varoq.get_Range(ustun1 + ustun2 + ustun1 + i, Type.Missing); b.Value2 = sem + " - yarim yillik"; i++; hajmTartibNomer = 0; semestrId = sem; sem++; yor.satr = "Select DISTINCT mut_fanlari_id from surovHajm where (kafedra_id=" + kafedraId + ") and (semestr_id=" + semestr + ") and (fakultet_id=" + x + ")"; lisMutfanId.Clear();

lisMutfanId = yor.butunSonliMassiy(); lisMutfanId.Sort(); foreach (int y in lisMutfanId) {yor.satr = "Select DISTINCT potok_id from surovHajm where (mut_fanlari_id=" + y + ") and(kafedra_id=" + kafedraId + ") and (semestr_id=" + semestr + ") and (fakultet_id=" + x + ")"; listMutFanPotok.Clear(); listMutFanPotok.AddRange(yor.butunSonliMassiy()); shartPotok = ""; foreach (int q in listMutFanPotok){ if (shartPotok.Length > 0) shartPotok = shartPotok + " or (potok_id=" + q + ")"; else shartPotok=shartPotok+" and ((potok_id=" + q +")"; } shartPotok = shartPotok + ")"; yor.satr = "Select DISTINCT mut_fanlari_id from surovHajm where semestr_id="+semestr+shartPotok+" group by mut_fanlari_id"; listPotokMutfanlar.Clear(); listPotokMutfanlar.AddRange(yor.butunSonliMassiy()); listPotokMutfanlar.Sort(); ShartSatri = "(mut_fanlari_id=" + y ; yor.satr = "Select mutaxassislik from surovHajm where (mut_fanlari_id=" + y + ") and (semestr_id=" + semestr + ") and (kafedra_id=" + kafedraId +")"; hajmMutaxassislik = yor.matn(); mutFanBor = true; if (listPotokMutfanlar.Contains(y)) { listPotokMutfanlar.Remove(y);}foreach (int z in listPotokMutfanlar) { yor.satr = "Select DISTINCT fakultet_id from surovHajm where (mut_fanlari_id=" + z + ") and (semestr_id=" + semestr + ") and (kafedra_id=" + kafedraId + ") group by fakultet_id"; int fak; fak = yor.son(); if (x < fak) { ShartSatri = ShartSatri + " or mut_fanlari_id=" + z; yor.satr = "Select mutaxassislik from surovHajm where (mut_fanlari_id=" + z + ") and (semestr_id=" + semestr + ") and (kafedra_id=" + kafedraId + ")";hajmMutaxassislik = hajmMutaxassislik+"," + yor.matn(); } else {if (x == fak) { if (y < z) { ShartSatri = ShartSatri + " or mut_fanlari_id=" + z; yor.satr = "Select mutaxassislik from surovHajm where (mut_fanlari_id=" + z + ") and (semestr_id=" + semestr + ") and (kafedra_id=" + kafedraId + ")"; hajmMutaxassislik = hajmMutaxassislik+","+ yor.matn(); }else { mutFanBor = false; break; } } else { mutFanBor = false; break; } } } ShartSatri = ShartSatri + ")"; if (mutFanBor) { hajmJamiSoat = 0; hajmTartibNomer++; hajmTalabaSoni = 0; yor.satr = "Select DISTINCT guruh_id from surovHajm where semestr_id=" + semestr + " and " + ShartSatri + ""; listGuruhlar.Clear(); listGuruhlar.AddRange(yor.butunSonliMassiy()); hajmGuruhSoni = listGuruhlar.Count(); yor.satr = "Select DISTINCT dars_turi_id from surovHajm where semestr_id=" + semestr + " and " + ShartSatri + " group by dars_turi_id"; listDarsTuri.Clear(); listDarsTuri.AddRange(yor.butunSonliMassiy());listDarsTuri.Sort(); varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1;varoq.get_Range("A" + i).Value = hajmTartibNomer;yor.satr = "Select blok from surovHajm where semestr_id=" + semestr + " and " + ShartSatri;hajmBlok = yor.haqiqiySon(); varoq.get_Range("B" + i).Value = hajmBlok;yor.satr = "Select fan_nomi from surovHajm where semestr_id=" + semestr + " and " + ShartSatri; hajmFanNomi = yor.matn(); varoq.get_Range("C" + i).Value =

*hajmFanNomi; varoq.get_Range("D" + i).Value = hajmMutaxassislik; yor.satr = "Select kurs from surovHajm where semestr_id=" + semestr + " and " + ShartSatri;hajmKurs = yor.son(); varoq.get_Range("E" + i).Value = hajmKurs; foreach (int gu in listGuruhlar) { yor.satr = "Select count(id) from talaba where guruh=" + gu; hajmTalabaSoni=hajmTalabaSoni + yor.son(); } varoq.get_Range("F" + i).Value = hajmTalabaSoni; varoq.get_Range("H" + i).Value = hajmGuruhSoni; foreach (int darta in listDarsTuri){ yor.satr = "Select DISTINCT soat from surovHajm where semestr_id=" + semestr + " and dars_turi_id=" + darta + " and " + ShartSatri ; hajmSoat = yor.son(); yor.satr = "Select count (DISTINCT potok_id) from surovHajm where semestr_id=" + semestr + " and dars_turi_id=" + darta + " and " + ShartSatri ; hajmPotokSoni = yor.son(); hajmJamiSoat=hajmJamiSoat+(hajmSoat*hajmPotokSoni); switch (darta){case 1: { if (hajmPotokSoni != 0){ varoq.get_Range("I" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); varoq.get_Range("G" + i).Value = hajmPotokSoni;Maruza = Maruza + (hajmPotokSoni * hajmSoat); varoq.get_Range("X" + i).Value = (int)Math.Round(hajmTalabaSoni * 0.2); varoq.get_Range("W" + i).Value = (int)Math.Round(hajmTalabaSoni * 0.2); Yakuniy_nazorat=Yakuniy_nazorat+(int)Math.Round(hajmTalabaSoni*0.2); Oraliq_nazorat = Oraliq_nazorat + (int)Math.Round(hajmTalabaSoni * 0.2); hajmJamiSoat = hajmJamiSoat + (int)Math.Round(hajmTalabaSoni * 0.2)*2; }else varoq.get_Range("G" + i).Value = 1; } break; case 2: { if (hajmPotokSoni != 0){ varoq.get_Range("J" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Amaliyot = Amaliyot + (hajmPotokSoni * hajmSoat); } } break; case 3:{ if (hajmPotokSoni != 0) { varoq.get_Range("K" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Semenar = Semenar + (hajmPotokSoni * hajmSoat); } } break; case 4: { if (hajmPotokSoni != 0) { varoq.get_Range("L" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat);Labaratoriya = Labaratoriya + (hajmPotokSoni * hajmSoat); varoq.get_Range("V" + i).Value = (int)Math.Round(hajmTalabaSoni * 0.2); Hisob_chizma_ishi = Hisob_chizma_ishi + (int)Math.Round(hajmTalabaSoni * 0.2);hajmJamiSoat = hajmJamiSoat + (int)Math.Round(hajmTalabaSoni * 0.2) ; } } break; case 5: { if (hajmPotokSoni != 0){ varoq.get_Range("M" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Kurs_ishi = Kurs_ishi + (hajmPotokSoni * hajmSoat); }} break; case 6: { if (hajmPotokSoni != 0){ varoq.get_Range("N" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); BMIga_rahbarlik = BMIga_rahbarlik + (hajmPotokSoni * hajmSoat); }} break; case 7: { if (hajmPotokSoni != 0)*

{ varoq.get_Range("O" + i).Value = hajmPotokSoni + "" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); MDga_rahbarlik = MDga_rahbarlik + (hajmPotokSoni * hajmSoat); } } break; case 8: { if (hajmPotokSoni != 0) { varoq.get_Range("P" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); DAK = DAK + (hajmPotokSoni * hajmSoat); }} break; case 9: { if (hajmPotokSoni != 0) { varoq.get_Range("Q" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Ped_amaliyot = Ped_amaliyot + (hajmPotokSoni * hajmSoat); } } break; case 10: { if (hajmPotokSoni != 0) { varoq.get_Range("R" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Malakaviy_amaliyot = Malakaviy_amaliyot + (hajmPotokSoni * hajmSoat); } } break; case 11: { if (hajmPotokSoni != 0){ varoq.get_Range("S" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Maslahat = Maslahat + (hajmPotokSoni * hajmSoat); }} break; case 12: {if (hajmPotokSoni != 0){ varoq.get_Range("T" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Doktorant_mustaqil = Doktorant_mustaqil + (hajmPotokSoni * hajmSoat);} } break; case 13: { if (hajmPotokSoni != 0) { varoq.get_Range("U" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Har_xil_soatlar = Har_xil_soatlar + (hajmPotokSoni * hajmSoat); } } break;}} varoq.get_Range("Y" + i).Value = hajmJamiSoat; hajmJamiSoatOxirgi = hajmJamiSoatOxirgi + hajmJamiSoat; i++; } else yor.satr = ""; } } semestrTugadi = true; } varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("C" + i).Value = (sem - 1) + " - jarayomda"; varoq.get_Range("I" + i).Value = Maruza; varoq.get_Range("J" + i).Value = Amaliyot; varoq.get_Range("K" + i).Value = Semenar; varoq.get_Range("L" + i).Value = Labaratoriya; varoq.get_Range("M" + i).Value = Kurs_ishi; varoq.get_Range("N" + i).Value = BMIga_rahbarlik; varoq.get_Range("O" + i).Value = MDga_rahbarlik; varoq.get_Range("P" + i).Value = DAK; varoq.get_Range("Q" + i).Value = Ped_amaliyot; varoq.get_Range("R" + i).Value = Malakaviy_amaliyot; varoq.get_Range("S" + i).Value = Maslahat; varoq.get_Range("T" + i).Value = Doktorant_mustaqil; varoq.get_Range("U" + i).Value = Har_xil_soatlar; varoq.get_Range("V" + i).Value = Hisob_chizma_ishi; varoq.get_Range("W" + i).Value = Oraliq_nazorat; varoq.get_Range("X" + i).Value = Yakuniy_nazorat; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi; MaruzaJami = MaruzaJami + Maruza; AmaliyotJami = AmaliyotJami + Amaliyot; SemenarJami = SemenarJami + Semenar; LabaratoriyaJami = LabaratoriyaJami + Labaratoriya; Kurs_ishiJami = Kurs_ishiJami + Kurs_ishi; BMIga_rahbarlikJami = BMIga_rahbarlikJami + BMIga_rahbarlik; MDga_rahbarlikJami = MDga_rahbarlikJami + MDga_rahbarlik; DAKJami = DAKJami + DAK;Ped_amaliyotJami*

= Ped_amaliyotJami + Ped_amaliyot; Malakaviy_amaliyotJami = Malakaviy_amaliyotJami + Malakaviy_amaliyot; MaslahatJami = MaslahatJami + Maslahat; Doktorant_mustaqilJami = Doktorant_mustaqilJami + Doktorant_mustaqil; Har_xil_soatlarJami = Har_xil_soatlarJami + Har_xil_soatlar; Hisob_chizma_ishiJami += Hisob_chizma_ishi; Oraliq_nazoratJami += Oraliq_nazorat; Joriy_nazoratJami += Joriy_nazorat; Yakuniy_nazoratJami += Yakuniy_nazorat; Mustaqil_talimJami += Mustaqil_talim; hajmJamiSoatOxirgi2 += hajmJamiSoatOxirgi; if (sem - 1 == 1) { Maruzasem += Maruza; Amaliyotsem += Amaliyot; Semenarsem += Semenar; Labaratoriyasem += Labaratoriya; Kurs_ishisem += Kurs_ishi; BMIga_rahbarliksem += BMIga_rahbarlik; MDga_rahbarliksem += MDga_rahbarlik; DAKsem += DAK; Ped_amaliyotsem += Ped_amaliyot; Malakaviy_amaliyotsem += Malakaviy_amaliyot; Maslahatsem += Malakaviy_amaliyotsem; Doktorant_mustaqilsem += Doktorant_mustaqil; Har_xil_soatlarsem += Har_xil_soatlar; Hisob_chizma_ishisem += Hisob_chizma_ishi; Oraliq_nazoratsem += Oraliq_nazorat; Joriy_nazoratsem += Joriy_nazorat; Yakuniy_nazoratsem += Yakuniy_nazorat;Mustaqil_talimsem += Mustaqil_talim; hajmJamiSoatOxirgisem += hajmJamiSoat; }Maruza = 0; Amaliyot = 0; Semenar = 0; Labaratoriya = 0; Kurs_ishi = 0; BMIga_rahbarlik = 0; MDga_rahbarlik = 0; DAK = 0; Ped_amaliyot = 0; Malakaviy_amaliyot = 0; Maslahat = 0; Doktorant_mustaqil = 0; Har_xil_soatlar = 0; Hisob_chizma_ishi = 0; Oraliq_nazorat = 0; Joriy_nazorat = 0; Yakuniy_nazorat = 0; Mustaqil_talim = 0; hajmJamiSoatOxirgi = 0; i++; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; i++; sem = 1; foreach (int semestr in listSemestr){ sem++; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; if (sem - 1 == 1){ varoq.get_Range("C" + i).Value = (sem - 1) + " - jarayonda"; varoq.get_Range("I" + i).Value = Maruzasem; varoq.get_Range("J" + i).Value = Amaliyotsem; varoq.get_Range("K" + i).Value = Semenarsem; varoq.get_Range("L" + i).Value = Labaratoriyasem; varoq.get_Range("M" + i).Value = Kurs_ishisem; varoq.get_Range("N" + i).Value = BMIga_rahbarliksem; varoq.get_Range("O" + i).Value = MDga_rahbarliksem; varoq.get_Range("P" + i).Value = DAKsem; varoq.get_Range("Q" + i).Value = Ped_amaliyotsem; varoq.get_Range("R" + i).Value = Malakaviy_amaliyotsem; varoq.get_Range("S" + i).Value = Maslahatsem; varoq.get_Range("T" + i).Value = Doktorant_mustaqilsem; varoq.get_Range("U" + i).Value = Har_xil_soatlarsem; varoq.get_Range("V" + i).Value = Hisob_chizma_ishisem; varoq.get_Range("W" + i).Value = Oraliq_nazoratsem; varoq.get_Range("X" + i).Value = Yakuniy_nazoratsem; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgisem; i++; } else {

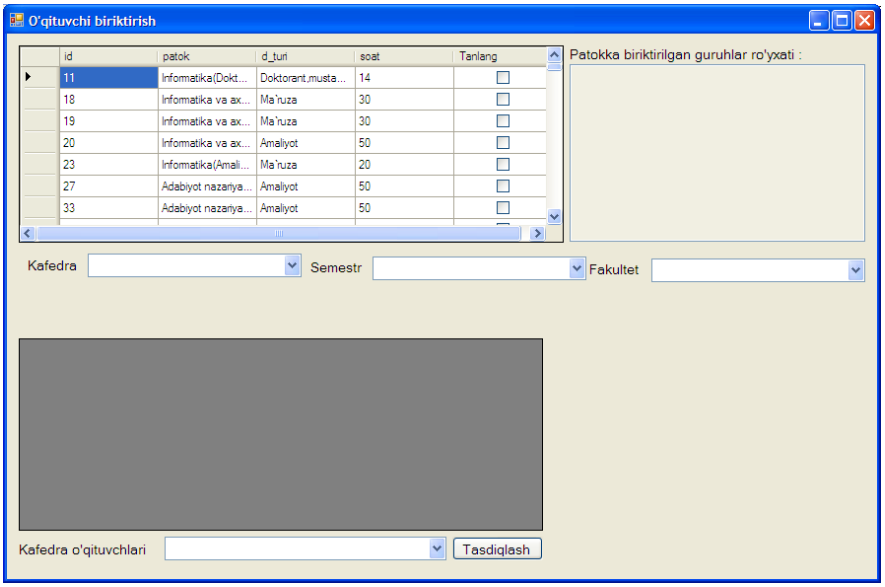
varoq.get_Range("C" + i).Value = (sem - 1) + " - jarayonda"; varoq.get_Range("I" + i).Value = MaruzaJami- Maruzasem; varoq.get_Range("J" + i).Value = AmaliyotJami- Amaliyotsem; varoq.get_Range("K" + i).Value = SemenarJami- Semenarsem; varoq.get_Range("L" + i).Value = LabaratoriyaJami- Labaratoriyasem; varoq.get_Range("M" + i).Value = Kurs_ishiJami- Kurs_ishisem; varoq.get_Range("N" + i).Value = BMIga_rahbarlikJami- BMIga_rahbarliksem; varoq.get_Range("O" + i).Value = MDga_rahbarlikJami- MDga_rahbarliksem; varoq.get_Range("P" + i).Value = DAKJami- DAKsem; varoq.get_Range("Q" + i).Value = Ped_amaliyotJami- Ped_amaliyotsem; varoq.get_Range("R" + i).Value = Malakaviy_amaliyotJami- Malakaviy_amaliyotsem; varoq.get_Range("S" + i).Value = Malakaviy_amaliyotJami- Maslahatsem; varoq.get_Range("T" + i).Value = Doktorant_mustaqilJami- Doktorant_mustaqilsem; varoq.get_Range("U" + i).Value = Har_xil_soatlarJami- Har_xil_soatlarsem; varoq.get_Range("V" + i).Value = Hisob_chizma_ishiJami - Hisob_chizma_ishisem; varoq.get_Range("W" + i).Value = Oraliq_nazoratJami- Oraliq_nazoratsem;varoq.get_Range("X" + i).Value = Yakuniy_nazoratJami- Yakuniy_nazoratsem;varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi2- hajmJamiSoatOxirgisem; i++; } } varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("C" + i).Value = "Jami"; varoq.get_Range("I" + i).Value = MaruzaJami; varoq.get_Range("J" + i).Value = AmaliyotJami; varoq.get_Range("K" + i).Value = SemenarJami;varoq.get_Range("L" + i).Value = LabaratoriyaJami; varoq.get_Range("M" + i).Value = Kurs_ishiJami; varoq.get_Range("N" + i).Value = BMIga_rahbarlikJami;varoq.get_Range("O" + i).Value = MDga_rahbarlikJami; varoq.get_Range("P" + i).Value = DAKJami; varoq.get_Range("Q" + i).Value = Ped_amaliyotJami;varoq.get_Range("R" + i).Value = Malakaviy_amaliyotJami; varoq.get_Range("S" + i).Value = MaslahatJami; varoq.get_Range("T" + i).Value = Doktorant_mustaqilJami;varoq.get_Range("U" + i).Value = Har_xil_soatlarJami;varoq.get_Range("V" + i).Value = Hisob_chizma_ishiJami;varoq.get_Range("W" + i).Value = Oraliq_nazoratJami;varoq.get_Range("X" + i).Value = Yakuniy_nazoratJami;varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi2; i++; i = i + 2; varoq.get_Range("Q" + i, "T" + i).MergeCells = true;varoq.get_Range("Q" + i, "T" + i).Name = ("QTQ" + i);dastur.Range UquUslubBosh = varoq.get_Range("QTQ" + i, Type.Missing);varoq.get_Range("K" + i, "P" + i).MergeCells = true;varoq.get_Range("K" + i, "P" + i).Name = ("KPK" + i);dastur.Range UquUslubBosh2 = varoq.get_Range("KPK" + i, Type.Missing);yor.satr = "Select oqituvchi from kaf_oqituvchilari where lavozim=" +

3;yor.satr = "Select asfio from V_kaf_uqituvchilari where oqituvchi=" + yor.son();sinov=yor.matn(); string rt=sinov.Substring(sinov.LastIndexOf(' ')+1),rs=" ";rs=rs+rt[0]; if((rt[0]=='C'||rt[0]=='S')&&(rt[1]=='h'||rt[1]=='H'))rs=rs+rt[1]; sinov=sinov.Substring(0,sinov.LastIndexOf(' ')); rs = rs + ". "; if (sinov.LastIndexOf(' ') != -1) { rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = ""+rt[0] + rt[1] + ". " + rs; else rs = rt[0] + ". " + rs; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); } rs = rs + sinov; UquUslubBosh2.Value = "O`quv uslubiy boshqarma boshlig`i :"; UquUslubBosh.Value = rs; i++; yor.satr = "Select dekan from fakultet where id=" + fakultetId; sinov = yor.matn(); rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); rs = ""; rs = rs + rt[0]; if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = rs + rt[1]; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); rs = rs + ". "; if (sinov.LastIndexOf(' ') != -1) { rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs =""+ rt[0] + rt[1] + ". " + rs; else rs = rt[0] + ". " + rs; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); } rs = rs + sinov; varoq.get_Range("Q" + i, "T" + i).MergeCells = true; varoq.get_Range("Q" + i, "T" + i).Name = ("QTQ" + i); dastur.Range dekan = varoq.get_Range("QTQ" + i, Type.Missing); varoq.get_Range("K" + i, "P" + i).MergeCells = true; varoq.get_Range("K" + i, "P" + i).Name = ("KPK" + i); dastur.Range dekan2 = varoq.get_Range("KPK" + i, Type.Missing); dekan2.Value = "Fakultet dekani :";dekan.Value = rs; i++; yor.satr = "Select kafedramudiri from kafedra where id=" + kafedraId; sinov = yor.matn(); rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); rs = ""; rs = rs + rt[0]; if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = rs + rt[1]; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); rs = rs + ". "; if (sinov.LastIndexOf(' ') != -1) { rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs =""+rt[0]+rt[1]+". "+rs; else rs = rt[0]+". "+rs; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); } rs = rs + sinov; varoq.get_Range("Q" + i, "T" + i).MergeCells = true; varoq.get_Range("Q" + i, "T" + i).Name = ("QTQ" + i); dastur.Range mudir = varoq.get_Range("QTQ" + i, Type.Missing); varoq.get_Range("K" + i, "P" + i).MergeCells = true; varoq.get_Range("K" + i, "P" + i).Name = ("KPK" + i);dastur.Range mudir2 = varoq.get_Range("KPK" + i, Type.Missing); mudir2.Value = "Kafedra mudiri :"; mudir.Value = rs; }

private void button2_Click(object sender, EventArgs e) { Close(); } }}

3.2 Kafedra dars taqsimotini qo’llab quvvatlovchi dasturiy tizim yaratish.

Kafedra dars taqsimotini tayyorlash uchun oldin barcha yaratilgan potoklarga o’qituvchilarni biriktirish kerak. Buning uchun biz menyular satridan “Potok yaratish” bo’limining “O'qituvchi biriktirish” bandini tanlaymiz va quyidagi oyna ochib beriladi.



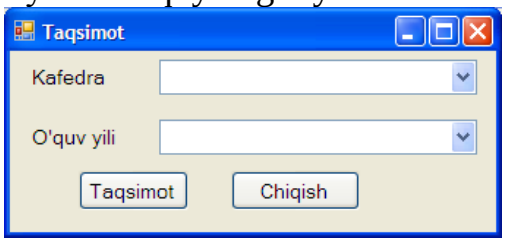
3.11-chizma. O'qituvchi biriktirish oynasining ko’rinishi.

Bu oynada qulaylik yaratish uchun oynaning yuqori chap tomon burchagida agarda mavjud bo’lsa potok takrorlanmas raqami, potok nomi, dars turi, soat, tanlagichlar keltirilgan. Yuqori o’ng tomon burchagida esa chap tomondan tanlangan potokka biriktilgan guruhlar ro’yxati namoyish etiladi. Undan so’ng ochiluvchi ro’yxatlarda tanlangan ma’lumotlarga mos potoklar namoyish etiladi. Undan so’ng eng past qismda tanlangan kafedra o’qituvchining soatlari namoyish etiladi. “Tasdiqlash” tugmasi bosilgan barcha tanlangan potoklar belgilangan o’qituvchiga beriladi. “Tasdiqlash” tugmasi bosilgandan so’ng bajariladigan amallar ketma-ketligi quyidagicha:

private void button1_Click(object sender, EventArgs e) { if (kafedraUqituvchi != 0) { for (int i = 0; i < dataGridView1.RowCount; i++) { object o = dataGridView1.Rows[i].Cells[4].Value; if (o != null) { bool b = (bool)o; if (b) { potokId = int.Parse(dataGridView1.Rows[i].Cells[0].Value.ToString()); yor.satr = "UPDATE potok SET domlo = " + kafedraUqituvchi + " WHERE id=" + potokId; yor.bazaUstidaAmal(); } } } } else MessageBox.Show("Ogohlantiramiz!\nSiz kafedra o\'qituvchisini tanlamadingiz \nshu sababli o\'zgartirish amalga oshirilmadi."); string s = ""; if

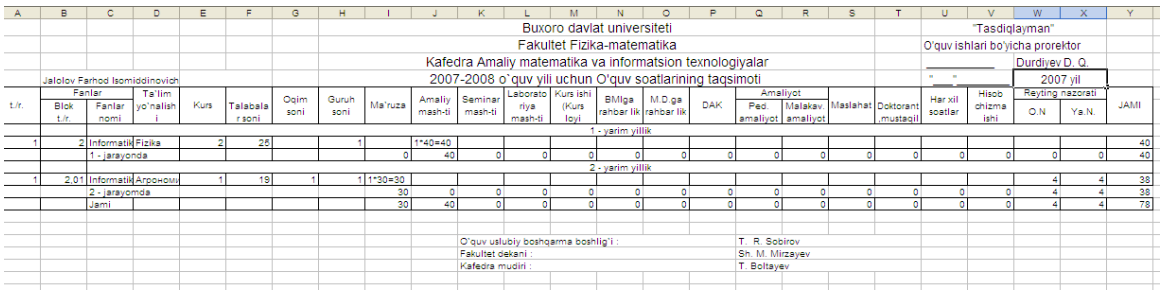
(comboBox2.Text != "") { yor.satr = "Select id from kafedra where kafedra='" + comboBox2.SelectedItem.ToString() + "'"; kafedraId = yor.son(); s = s + "(kafedra_id=" + kafedraId + ")"; comboBox3.Items.Clear(); comboBox3.Items.Add("--------"); yor.satr = "Select asfio from V_kaf_uqituvchilari where kaf=" + kafedraId; comboBox3.Items.AddRange(yor.satrliMassiy()); comboBox3.SelectedIndex = 0; } if (comboBox1.Text != "") { yor.satr = "Select id from semestr where semestr='" + comboBox1.SelectedItem.ToString() + "'"; semestrId = yor.son(); if(s.Length>0) s = s + " and (semestr_id=" + semestrId + ")"; else s = s + "(semestr_id=" + semestrId + ")"; } if (comboBox4.Text != "") { yor.satr = "Select id from fakultet where fakultet='" + comboBox4.SelectedItem.ToString() + "'"; fakultetId = yor.son(); if (s.Length > 0) s = s + " and (fakultet_id=" +fakultetId+ ")"; else s = s + "(fakultet_id=" + fakultetId + ")"; } if (s.Length > 0) { s = "Select distinct id, potok, d_turi, soat from surovYangiPotokUqituvchi where (domlo is null) and (" + s + ") group by id, potok, d_turi, soat"; } else { s = "Select distinct id, potok, d_turi, soat from surovYangiPotokUqituvchi where (domlo is null) group by id, potok, d_turi, soat"; } yor.satr = s; dataGridView1.Columns.Clear(); DataSet ds = new DataSet(); yor.dataAdapter().Fill(ds); dataGridView1.DataSource = ds.Tables[0].DefaultView; DataGridViewCheckBoxColumn cmb = new DataGridViewCheckBoxColumn(); cmb.HeaderText = "Tanlang"; cmb.Name = "Check"; dataGridView1.Columns.Add(cmb); dataGridView2.Columns.Clear(); label6.Text = ""; }*

Keraklicha potoklarga o’qituvchi biriktirilgach taqsimotni hosil qilish mumkin. Buning uchun biz menyular satridan “Hisobot” bo’limining “Taqsimot” bandini tanlaymiz va quyidagi oyna ochib beriladi.



3.12-chizma. Taqsimot oynasining ko’rinishi.

Bu oynadagi ochiluvchi ro’yxatlardan kerakli ma’lumotlar tanlangach “Taqsimot” tugmasi bosiladi va tanlangan kafedraning tegishli o’quv yilidagi dars taqsimotiga tegishli ma’lumotlarni Microsoft Office Excel dasturini ishga tushurib so’ng Microsoft Office Excelning hujjatiga yuboriladi(3.13-chizma).



Buxoro davlat universiteti
Fakultet Fizika-matematika
Kafedra Amaliy matematika va informatsion texnologiyalar
2007-2008 o`quv yili uchun O'quv soatlarining taqsimoti

"Tasdiqlayman"
O'quv ishlari bo'yicha prorektor
Durdiyev D. Q.
" " 2007 yil

Jalolov Farhod Isomiddinovich

t.r.	Fanlar: Blok t.r.	Fanlar: Fanlar nomi	Fanlar: Ta'lim yo'nalishi	Kurs	Talabalar soni	Oqim soni	Guruh soni	Ma'ruza	Amaliy mash-ti	Seminar mash-ti	Laboratoriya mash-ti	Kurs ishi (Kurs loyi	BMIga rahbarlik	M.D.ga rahbarlik	DAK	Amaliyot: Ped. amaliyot	Amaliyot: Malakav. amaliyot	Maslahat	Doktorant, mustaqil	Har xil soatlar	Hisob chizma ishi	Reyting nazorati: O.N	Reyting nazorati: Ya.N.	JAMI
1 - yarim yillik																								
1	2	Informatika	Fizika	2	25		1		1*40=40															40
		1 - jarayonda						0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	40
2 - yarim yillik																								
1	2,01	Informatika	Агрономия	1	19	1	1	1*30=30														4	4	38
		2 - jarayonda						30	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	38
		Jami						30	40	0	0	0	0	0	0	0	0	0	0	0	0	4	4	78

O'quv uslubiy boshqarma boshlig'i : T. R. Sobirov
Fakultet dekani : Sh. M. Mirzayev
Kafedra mudiri : T. Boltayev

3.13-chizma. Kafedra dars taqsimoti

Kafedra dars taqsimotini qo’llab quvvatlovchi dastur matni quyida keltirilgan.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Collections;
using dastur = Microsoft.Office.Interop.Excel;
namespace Sqldsan {
 public partial class taqsimot : Form { yordamchiDastur yor = new yordamchiDastur();
public List<int> lisDomloId = new List<int>(); public List<int> lisMutfanId = new List<int>(); public List<int> listMutFanPotok = new List<int>();
 public List<int> listPotokMutfanlar = new List<int>(); public List<int> listSemestr = new List<int>(); public List<string> listSemestrNomi = new List<string>(); public List<int> listDarsTuri = new List<int>(); public List<int> listGuruhlar = new List<int>(); public string ShartSatri = "" , uquvYili = "" , shartSemestr = "" , shartPotok = "" , sinov = "" , hajmMutaxassislik = "" , hajmFanNomi = ""; public double hajmBlok = 0 ; public DateTime dt = DateTime.Now; int yil = 0 , sem = 0 , hajmSoat = 0 , hajmPotokSoni = 0 , hajmGuruhSoni = 0 , hajmTalabaSoni = 0 , hajmKurs = 0 , hajmTartibNomer = 0 , hajmJamiSoat = 0 , jamiSoat = 0 , semeatrX = 0 , fakultetX = 0 ; bool mutFanBor = false , semestrTugadi = false ; public taqsimot() { InitializeComponent(); } private void button3_Click(object sender, EventArgs e) { Close(); } private void button4_Click(object sender, EventArgs e) { int semestrId , i = 7 , kafedraId , Maruza = 0 , Amaliyot = 0 , Semenar = 0 , Labaratoriya = 0 , Kurs_ishi = 0 , BMIga_rahbarlik = 0 , MDga_rahbarlik = 0 , DAK = 0 , Ped_amaliyot = 0 , Malakaviy_amaliyot = 0 , Maslahat = 0 , Doktorant_mustaqil = 0 , Har_xil_soatlar = 0 , Hisob_chizma_ishi = 0 , Oraliq_nazorat = 0 , Joriy_nazorat = 0 ,
```

Yakuniy_nazorat = 0 , Mustaqil_talim = 0 , fakultetK = 0 , hajmJamiSoatOxirgi = 0 , MaruzaJami = 0 , AmaliyotJami = 0 , SemenarJami = 0 , LabaratoriyaJami = 0 , Kurs_ishiJami = 0 , BMIga_rahbarlikJami = 0 , MDga_rahbarlikJami = 0 , DAKJami = 0 , Ped_amaliyotJami = 0 , Malakaviy_amaliyotJami = 0 , MaslahatJami = 0 , Doktorant_mustaqilJami = 0 , Har_xil_soatlarJami = 0 , Hisob_chizma_ishiJami = 0 , Oraliq_nazoratJami = 0 , Joriy_nazoratJami = 0 , Yakuniy_nazoratJami = 0 , Mustaqil_talimJami = 0 , hajmJamiSoatOxirgi2 = 0 , Maruzasem = 0 , Amaliyotsem = 0 , Semenarsem = 0 , Labaratoriyasem = 0 , Kurs_ishisem = 0 , BMIga_rahbarliksem = 0 , MDga_rahbarliksem = 0 , DAKsem = 0 , Ped_amaliyotsem = 0 , Malakaviy_amaliyotsem = 0 , Maslahatsem = 0 , Doktorant_mustaqilsem = 0 , Har_xil_soatlarsem = 0 , Hisob_chizma_ishisem = 0 , Oraliq_nazoratsem = 0 , Joriy_nazoratsem = 0 , Yakuniy_nazoratsem = 0 , Mustaqil_talimsem = 0 , fakultetKsem = 0 , hajmJamiSoatOxirgisem = 0 , tartibNomeri = 1 ; string kafedraUqituvchiNomi = "" ; string ustun1 = "C" , ustun2 = "Y" , rs , rt ; uquvYili = comboBox1.SelectedItem.ToString().Substring(0, comboBox1.SelectedItem.ToString().IndexOf('o')); yor.satr = "SELECT id FROM semestr WHERE semestr LIKE '" + uquvYili + "%'"; listSemestr.Clear(); listSemestr.AddRange(yor.butunSonliMassiy()); string surovShart = ""; shartSemestr = ""; foreach (int s in listSemestr) { if (shartSemestr.Length > 0) shartSemestr = shartSemestr + " or semestr_id=" + s; else shartSemestr = shartSemestr + " and (semestr_id=" + s; } shartSemestr = shartSemestr + ")"; string yul = @"d:\Dars taqsimoti\Excel\Taqsimot.xls" , yul2 = @"d:\Dars taqsimoti\Taqsimot.xls" ; if (File.Exists(yul2)) { File.Delete(yul2); File.Copy(yul, yul2); } else File.Copy(yul, yul2); dastur.Application excel = new dastur.Application(); dastur.Workbook kitob = excel.Workbooks.Open(yul2); dastur.Worksheet varoq; dastur.Worksheet varoq2 = (dastur.Worksheet)kitob.Sheets[2]; dastur.Worksheet varoq7; excel.Visible = true; yor.satr = "Select id from kafedra where kafedra='" + comboBox2.SelectedItem.ToString() + "'"; kafedraId = yor.son(); yor.satr = "Select DISTINCT kafedra_uqituvchi_id from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_id=" + kafedraId + ")" + shartSemestr + " group by kafedra_uqituvchi_id"; lisDomloId.Clear(); lisDomloId.AddRange(yor.butunSonliMassiy()); lisDomloId.Sort();varoq2.get_Range("H2", "R2").Name = ("HRH2"); dastur.Range fakultetNomi = varoq2.get_Range("HRH2", Type.Missing); yor.satr = "Select fak from kafedra where id=" + kafedraId; int fakultetId = yor.son(); yor.satr = "Select fakultet from fakultet where id=" + fakultetId; fakultetNomi.Value = "Fakultet "+yor.matn();varoq2.get_Range("H3", "R3").Name = ("HRH3"); dastur.Range kafedraNomi

= varoq2.get_Range("HRH3", Type.Missing); kafedraNomi.Value ="Kafedra " +comboBox2.SelectedItem.ToString();varoq2.get_Range("H4", "R4").Name = ("HRH4"); dastur.Range uquvYiliExcel = varoq2.get_Range("HRH4", Type.Missing); uquvYiliExcel.Value = comboBox1.SelectedItem.ToString() + " uchun O'quv soatlarining taqsimoti "; varoq2.get_Range("w4", "x4").Name = ("wxw4"); dastur.Range yilNomi = varoq2.get_Range("wxw4", Type.Missing); yil = int.Parse(dt.Year.ToString()); yilNomi.Value = yil + " yil"; semestrTugadi = false; semeatrX = listSemestr[0]; if (lisDomloId.Count != 0) fakultetX = lisDomloId[0]; hajmJamiSoatOxirgi = 0; fakultetK = -1; foreach (int x in lisDomloId) {dastur.Worksheet varoqi = (dastur.Worksheet)kitob.Sheets[2]; varoq2.Copy(varoq2, Type.Missing); varoq = (dastur.Worksheet)kitob.Sheets[tartibNomeri]; yor.satr = "Select asfio from V_kaf_uqituvchilari where id=" + x; varoq.get_Range("B4", "D4").Name = ("BDB4"); varoq.get_Range("B4", "D4").MergeCells = true; dastur.Range kafedraUqituvchiNomiExcel = varoq.get_Range("BDB4", Type.Missing); sinov = yor.matn(); kafedraUqituvchiNomi = sinov; kafedraUqituvchiNomiExcel.Value = kafedraUqituvchiNomi; varoq.Name =kafedraUqituvchiNomi; tartibNomeri++; sem = 1; semeatrX = sem; foreach (int semestr in listSemestr) { if (sem != semeatrX) { semeatrX = sem; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("C" + i).Value = (sem - 1) + " - jarayonda"; varoq.get_Range("I" + i).Value = Maruza; varoq.get_Range("J" + i).Value = Amaliyot; varoq.get_Range("K" + i).Value = Semenar; varoq.get_Range("L" + i).Value = Labaratoriya; varoq.get_Range("M" + i).Value = Kurs_ishi; varoq.get_Range("N" + i).Value = BMIga_rahbarlik; varoq.get_Range("O" + i).Value = MDga_rahbarlik; varoq.get_Range("P" + i).Value = DAK; varoq.get_Range("Q" + i).Value = Ped_amaliyot; varoq.get_Range("R" + i).Value = Malakaviy_amaliyot; varoq.get_Range("S" + i).Value = Maslahat; varoq.get_Range("T" + i).Value = Doktorant_mustaqil; varoq.get_Range("U" + i).Value = Har_xil_soatlar; varoq.get_Range("V" + i).Value = Hisob_chizma_ishi; varoq.get_Range("W" + i).Value = Oraliq_nazorat; varoq.get_Range("X" + i).Value = Yakuniy_nazorat; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi; MaruzaJami = MaruzaJami + Maruza; AmaliyotJami = AmaliyotJami + Amaliyot; SemenarJami = SemenarJami + Semenar; LabaratoriyaJami = LabaratoriyaJami + Labaratoriya; Kurs_ishiJami = Kurs_ishiJami + Kurs_ishi; BMIga_rahbarlikJami = BMIga_rahbarlikJami + BMIga_rahbarlik; MDga_rahbarlikJami = MDga_rahbarlikJami + MDga_rahbarlik; DAKJami = DAKJami + DAK; Ped_amaliyotJami = Ped_amaliyotJami + Ped_amaliyot; Malakaviy_amaliyotJami = Malakaviy_amaliyotJami + Malakaviy_amaliyot; MaslahatJami = MaslahatJami + Maslahat; Doktorant_mustaqilJami = Doktorant_mustaqilJami + Doktorant_mustaqil; Har_xil_soatlarJami = Har_xil_soatlarJami + Har_xil_soatlar;

Hisob_chizma_ishiJami += Hisob_chizma_ishi; Oraliq_nazoratJami += Oraliq_nazorat; Joriy_nazoratJami += Joriy_nazorat; Yakuniy_nazoratJami += Yakuniy_nazorat; Mustaqil_talimJami += Mustaqil_talim; hajmJamiSoatOxirgi2 += hajmJamiSoatOxirgi; if (sem - 1 == 1) { Maruzasem += Maruza; Amaliyotsem += Amaliyot; Semenarsem += Semenar; Labaratoriyasem += Labaratoriya; Kurs_ishisem += Kurs_ishi; BMIga_rahbarliksem += BMIga_rahbarlik; MDga_rahbarliksem += MDga_rahbarlik; DAKsem += DAK; Ped_amaliyotsem += Ped_amaliyot; Malakaviy_amaliyotsem += Malakaviy_amaliyot; Maslahatsem += Malakaviy_amaliyotsem; Doktorant_mustaqilsem += Doktorant_mustaqil; Har_xil_soatlarsem += Har_xil_soatlar; Hisob_chizma_ishisem += Hisob_chizma_ishi; Oraliq_nazoratsem += Oraliq_nazorat; Joriy_nazoratsem += Joriy_nazorat; Yakuniy_nazoratsem += Yakuniy_nazorat; Mustaqil_talimsem += Mustaqil_talim; hajmJamiSoatOxirgisem += hajmJamiSoat; } Maruza = 0; Amaliyot = 0; Semenar = 0; Labaratoriya = 0; Kurs_ishi = 0; BMIga_rahbarlik = 0; MDga_rahbarlik = 0; DAK = 0; Ped_amaliyot = 0; Malakaviy_amaliyot = 0; Maslahat = 0; Doktorant_mustaqil = 0; Har_xil_soatlar = 0; Hisob_chizma_ishi = 0; Oraliq_nazorat = 0; Joriy_nazorat = 0; Yakuniy_nazorat = 0; Mustaqil_talim = 0; hajmJamiSoatOxirgi = 0; hajmJamiSoat = 0; i++; } varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range(ustun1 + i, ustun2 + i).MergeCells = true; varoq.get_Range(ustun1 + i, ustun2 + i).HorizontalAlignment = dastur.XlHAlign.xlHAlignCenter; varoq.get_Range(ustun1 + i, ustun2 + i).Borders.LineStyle = 1; varoq.get_Range(ustun1 + i, ustun2 + i).Name = (ustun1 + ustun2 + ustun1 + i); varoq.get_Range(ustun1 + i, ustun2 + i).Font.FontStyle = 1; dastur.Range b = varoq.get_Range(ustun1 + ustun2 + ustun1 + i, Type.Missing); b.Value2 = sem + " - yarim yillik"; i++; hajmTartibNomer = 0; semestrId = sem; sem++; yor.satr = "Select DISTINCT mut_fanlari_id from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_id=" + kafedraId + ") and (semestr_id=" + semestr + ") and (kafedra_uqituvchi_id=" + x + ")"; lisMutfanId.Clear(); lisMutfanId = yor.butunSonliMassiy(); lisMutfanId.Sort(); foreach (int y in lisMutfanId) { yor.satr = "Select DISTINCT potok_id from surovHajm where (kafedra_uqituvchi_id is not null) and (mut_fanlari_id=" + y + ") and(kafedra_id=" + kafedraId + ") and (semestr_id=" + semestr + ") and (kafedra_uqituvchi_id=" + x + ") group by potok_id"; listMutFanPotok.Clear(); listMutFanPotok.AddRange(yor.butunSonliMassiy()); shartPotok = ""; foreach (int q in listMutFanPotok) { if (shartPotok.Length > 0) shartPotok = shartPotok + " or (potok_id=" + q + ")"; else shartPotok = shartPotok + " and ((potok_id=" + q + ")"; } shartPotok = shartPotok + ")"; yor.satr = "Select DISTINCT mut_fanlari_id from surovHajm where (kafedra_uqituvchi_id is not null) and semestr_id=" + semestr + shartPotok + " and

(kafedra_uqituvchi_id=" + x + ")" + " group by mut_fanlari_id"; listPotokMutfanlar.Clear(); listPotokMutfanlar.AddRange(yor.butunSonliMassiy()); listPotokMutfanlar.Sort(); ShartSatri = "(mut_fanlari_id=" + y; yor.satr = "Select mutaxassislik from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and (mut_fanlari_id=" + y + ") and (semestr_id=" + semestr + ") and (kafedra_id=" + kafedraId + ")"; hajmMutaxassislik = yor.matn(); mutFanBor = true; if (listPotokMutfanlar.Contains(y)) { listPotokMutfanlar.Remove(y); } foreach (int z in listPotokMutfanlar) { if (z < y) { ShartSatri = ShartSatri + " or mut_fanlari_id=" + z; yor.satr = "Select mutaxassislik from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and (mut_fanlari_id=" + z + ") and (semestr_id=" + semestr + ") and (kafedra_id=" + kafedraId + ")"; hajmMutaxassislik = hajmMutaxassislik + "," + yor.matn(); } else mutFanBor = false; } ShartSatri = ShartSatri + ")"; if (mutFanBor) { hajmJamiSoat = 0; hajmTartibNomer++; hajmTalabaSoni = 0; yor.satr = "Select DISTINCT guruh_id from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and semestr_id=" + semestr + " and " + ShartSatri + "";listGuruhlar.Clear(); listGuruhlar.AddRange(yor.butunSonliMassiy()); hajmGuruhSoni = listGuruhlar.Count();yor.satr = "Select DISTINCT dars_turi_id from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and semestr_id=" + semestr + " and " + ShartSatri + " group by dars_turi_id"; listDarsTuri.Clear(); listDarsTuri.AddRange(yor.butunSonliMassiy()); listDarsTuri.Sort();varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("A" + i).Value = hajmTartibNomer; yor.satr = "Select blok from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and semestr_id=" + semestr + " and " + ShartSatri; hajmBlok = yor.haqiqiySon(); varoq.get_Range("B" + i).Value = hajmBlok; yor.satr = "Select fan_nomi from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and semestr_id=" + semestr + " and " + ShartSatri; hajmFanNomi = yor.matn(); varoq.get_Range("C" + i).Value = hajmFanNomi; varoq.get_Range("D" + i).Value = hajmMutaxassislik; yor.satr = "Select kurs from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and semestr_id=" + semestr + " and " + ShartSatri; hajmKurs = yor.son(); varoq.get_Range("E" + i).Value = hajmKurs; foreach (int gu in listGuruhlar) { yor.satr = "Select count(id) from talaba where guruh=" + gu; hajmTalabaSoni = hajmTalabaSoni + yor.son(); } varoq.get_Range("F" + i).Value = hajmTalabaSoni; varoq.get_Range("H" + i).Value = hajmGuruhSoni; foreach (int darta in listDarsTuri) { yor.satr = "Select DISTINCT soat from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id="

*+ x + ") and semestr_id=" + semestr + " and dars_turi_id=" + darta + " and " + ShartSatri; hajmSoat = yor.son(); yor.satr = "Select count (DISTINCT potok_id) from surovHajm where (kafedra_uqituvchi_id is not null) and (kafedra_uqituvchi_id=" + x + ") and semestr_id=" + semestr + " and dars_turi_id=" + darta + " and " + ShartSatri; hajmPotokSoni = yor.son(); hajmJamiSoat = hajmJamiSoat + (hajmSoat * hajmPotokSoni); switch (darta) { case 1: { if (hajmPotokSoni != 0) { varoq.get_Range("I" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); varoq.get_Range("G" + i).Value = hajmPotokSoni; Maruza = Maruza + (hajmPotokSoni * hajmSoat); varoq.get_Range("X" + i).Value = (int)Math.Round(hajmTalabaSoni * 0.2); varoq.get_Range("W" + i).Value = (int)Math.Round(hajmTalabaSoni * 0.2); Yakuniy_nazorat = Yakuniy_nazorat + (int)Math.Round(hajmTalabaSoni * 0.2); Oraliq_nazorat = Oraliq_nazorat + (int)Math.Round(hajmTalabaSoni * 0.2); hajmJamiSoat = hajmJamiSoat + (int)Math.Round(hajmTalabaSoni * 0.2) * 2; } else varoq.get_Range("G" + i).Value = 1; } break; case 2: { if (hajmPotokSoni != 0) { varoq.get_Range("J" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Amaliyot = Amaliyot + (hajmPotokSoni * hajmSoat); } } break; case 3: { if (hajmPotokSoni != 0) { varoq.get_Range("K" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Semenar = Semenar + (hajmPotokSoni * hajmSoat); } } break; case 4: { if (hajmPotokSoni != 0) { varoq.get_Range("L" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Labaratoriya = Labaratoriya + (hajmPotokSoni * hajmSoat); varoq.get_Range("V" + i).Value = (int)Math.Round(hajmTalabaSoni * 0.2); Hisob_chizma_ishi = Hisob_chizma_ishi + (int)Math.Round(hajmTalabaSoni * 0.2); hajmJamiSoat = hajmJamiSoat + (int)Math.Round(hajmTalabaSoni * 0.2); } } break; case 5: { if (hajmPotokSoni != 0) { varoq.get_Range("M" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Kurs_ishi = Kurs_ishi + (hajmPotokSoni * hajmSoat); } } break; case 6: { if (hajmPotokSoni != 0) { varoq.get_Range("N" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); BMIga_rahbarlik = BMIga_rahbarlik + (hajmPotokSoni * hajmSoat); } } break; case 7: { if (hajmPotokSoni != 0) { varoq.get_Range("O" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); MDga_rahbarlik = MDga_rahbarlik + (hajmPotokSoni * hajmSoat); } } break; case 8: { if (hajmPotokSoni != 0) { varoq.get_Range("P" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); DAK = DAK + (hajmPotokSoni * hajmSoat); } } break; case 9: { if (hajmPotokSoni != 0) { varoq.get_Range("Q" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" +*

*(hajmPotokSoni * hajmSoat); Ped_amaliyot = Ped_amaliyot + (hajmPotokSoni * hajmSoat); } } break; case 10: { if (hajmPotokSoni != 0) { varoq.get_Range("R" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Malakaviy_amaliyot = Malakaviy_amaliyot + (hajmPotokSoni * hajmSoat); } } break; case 11: { if (hajmPotokSoni != 0) { varoq.get_Range("S" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Maslahat = Maslahat + (hajmPotokSoni * hajmSoat); } } break; case 12: { if (hajmPotokSoni != 0) { varoq.get_Range("T" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Doktorant_mustaqil = Doktorant_mustaqil + (hajmPotokSoni * hajmSoat); } } break; case 13: { if (hajmPotokSoni != 0) { varoq.get_Range("U" + i).Value = hajmPotokSoni + "*" + hajmSoat + "=" + (hajmPotokSoni * hajmSoat); Har_xil_soatlar = Har_xil_soatlar + (hajmPotokSoni * hajmSoat); } } break; } } varoq.get_Range("Y" + i).Value = hajmJamiSoat; hajmJamiSoatOxirgi = hajmJamiSoatOxirgi + hajmJamiSoat; i++; } else yor.satr = ""; } } semestrTugadi = true; varoq = (dastur.Worksheet)kitob.Sheets[tartibNomeri - 1]; varoq.get_Range("C" + i).Value = (sem - 1) + " - jarayomda"; varoq.get_Range("I" + i).Value = Maruza; varoq.get_Range("J" + i).Value = Amaliyot; varoq.get_Range("K" + i).Value = Semenar; varoq.get_Range("L" + i).Value = Labaratoriya; varoq.get_Range("M" + i).Value = Kurs_ishi; varoq.get_Range("N" + i).Value = BMIga_rahbarlik; varoq.get_Range("O" + i).Value = MDga_rahbarlik; varoq.get_Range("P" + i).Value = DAK; varoq.get_Range("Q" + i).Value = Ped_amaliyot; varoq.get_Range("R" + i).Value = Malakaviy_amaliyot; varoq.get_Range("S" + i).Value = Maslahat; varoq.get_Range("T" + i).Value = Doktorant_mustaqil; varoq.get_Range("U" + i).Value = Har_xil_soatlar; varoq.get_Range("V" + i).Value = Hisob_chizma_ishi; varoq.get_Range("W" + i).Value = Oraliq_nazorat; varoq.get_Range("X" + i).Value = Yakuniy_nazorat; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi; MaruzaJami = MaruzaJami + Maruza; AmaliyotJami = AmaliyotJami + Amaliyot; SemenarJami = SemenarJami + Semenar; LabaratoriyaJami = LabaratoriyaJami + Labaratoriya; Kurs_ishiJami = Kurs_ishiJami + Kurs_ishi; BMIga_rahbarlikJami = BMIga_rahbarlikJami + BMIga_rahbarlik; MDga_rahbarlikJami = MDga_rahbarlikJami + MDga_rahbarlik; DAKJami = DAKJami + DAK; Ped_amaliyotJami = Ped_amaliyotJami + Ped_amaliyot; Malakaviy_amaliyotJami = Malakaviy_amaliyotJami + Malakaviy_amaliyot; MaslahatJami = MaslahatJami + Maslahat; Doktorant_mustaqilJami = Doktorant_mustaqilJami + Doktorant_mustaqil; Har_xil_soatlarJami = Har_xil_soatlarJami + Har_xil_soatlar; Hisob_chizma_ishiJami += Hisob_chizma_ishi; Oraliq_nazoratJami += Oraliq_nazorat; Joriy_nazoratJami += Joriy_nazorat; Yakuniy_nazoratJami += Yakuniy_nazorat; Mustaqil_talimJami +=*

Mustaqil_talim; hajmJamiSoatOxirgi2 += hajmJamiSoatOxirgi; Maruza = 0; Amaliyot = 0; Semenar = 0; Labaratoriya = 0; Kurs_ishi = 0; BMIga_rahbarlik = 0; MDga_rahbarlik = 0; DAK = 0; Ped_amaliyot = 0; Malakaviy_amaliyot = 0; Maslahat = 0; Doktorant_mustaqil = 0; Har_xil_soatlar = 0; Hisob_chizma_ishi = 0; Oraliq_nazorat = 0; Joriy_nazorat = 0; Yakuniy_nazorat = 0; Mustaqil_talim = 0; hajmJamiSoatOxirgi = 0; i++; varoq.get_Range("A" + i, "Y" + i).Borders.LineStyle = 1; varoq.get_Range("C" + i).Value = "Jami"; varoq.get_Range("I" + i).Value = MaruzaJami; varoq.get_Range("J" + i).Value = AmaliyotJami; varoq.get_Range("K" + i).Value = SemenarJami; varoq.get_Range("L" + i).Value = LabaratoriyaJami; varoq.get_Range("M" + i).Value = Kurs_ishiJami; varoq.get_Range("N" + i).Value = BMIga_rahbarlikJami; varoq.get_Range("O" + i).Value = MDga_rahbarlikJami; varoq.get_Range("P" + i).Value = DAKJami; varoq.get_Range("Q" + i).Value = Ped_amaliyotJami; varoq.get_Range("R" + i).Value = Malakaviy_amaliyotJami; varoq.get_Range("S" + i).Value = MaslahatJami; varoq.get_Range("T" + i).Value = Doktorant_mustaqilJami; varoq.get_Range("U" + i).Value = Har_xil_soatlarJami; varoq.get_Range("V" + i).Value = Hisob_chizma_ishiJami; varoq.get_Range("W" + i).Value = Oraliq_nazoratJami; varoq.get_Range("X" + i).Value = Yakuniy_nazoratJami; varoq.get_Range("Y" + i).Value = hajmJamiSoatOxirgi2; i++; MaruzaJami = 0; AmaliyotJami = 0; SemenarJami = 0; LabaratoriyaJami = 0; Kurs_ishiJami = 0; BMIga_rahbarlikJami = 0; MDga_rahbarlikJami = 0; DAKJami = 0; Ped_amaliyotJami = 0; Malakaviy_amaliyotJami = 0; MaslahatJami = 0; Doktorant_mustaqilJami = 0; Har_xil_soatlarJami = 0; Hisob_chizma_ishiJami = 0; Oraliq_nazoratJami = 0; Joriy_nazoratJami = 0; Yakuniy_nazoratJami = 0; Mustaqil_talimJami = 0; hajmJamiSoatOxirgi2 = 0; i = i + 2; varoq.get_Range("Q" + i, "T" + i).MergeCells = true; varoq.get_Range("Q" + i, "T" + i).Name = ("QTQ" + i); dastur.Range UquUslubBosh = varoq.get_Range("QTQ" + i, Type.Missing); varoq.get_Range("K" + i, "P" + i).MergeCells = true; varoq.get_Range("K" + i, "P" + i).Name = ("KPK" + i); dastur.Range UquUslubBosh2 = varoq.get_Range("KPK" + i, Type.Missing); yor.satr = "Select oqituvchi from kaf_oqituvchilari where lavozim=" + 3; yor.satr = "Select asfio from V_kaf_uqituvchilari where oqituvchi=" + yor.son(); sinov = yor.matn(); rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); rs = " "; rs = rs + rt[0]; if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = rs + rt[1]; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); rs = rs + ". "; if (sinov.LastIndexOf(' ') != -1) { rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = "" + rt[0] + rt[1] + ". " + rs; else rs = rt[0] + ". " + rs; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); } rs = rs + sinov; UquUslubBosh2.Value = "O`quv

uslubiy boshqarma boshlig`i :"; UquUslubBosh.Value = rs; i++; yor.satr = "Select dekan from fakultet where id=" + fakultetId; sinov = yor.matn(); rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); rs = ""; rs = rs + rt[0]; if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = rs + rt[1]; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); rs = rs + ". "; if (sinov.LastIndexOf(' ') != -1) { rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = "" + rt[0] + rt[1] + ". " + rs; else rs = rt[0] + ". " + rs; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); } rs = rs + sinov; varoq.get_Range("Q" + i, "T" + i).MergeCells = true; varoq.get_Range("Q" + i, "T" + i).Name = ("QTQ" + i); dastur.Range dekan = varoq.get_Range("QTQ" + i, Type.Missing); varoq.get_Range("K" + i, "P" + i).MergeCells = true; varoq.get_Range("K" + i, "P" + i).Name = ("KPK" + i); dastur.Range dekan2 = varoq.get_Range("KPK" + i, Type.Missing); dekan2.Value = "Fakultet dekani :"; dekan.Value = rs; i++; yor.satr = "Select kafedramudiri from kafedra where id=" + kafedraId; sinov = yor.matn(); rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); rs = ""; rs = rs + rt[0]; if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = rs + rt[1]; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); rs = rs + ". "; if (sinov.LastIndexOf(' ') != -1) { rt = sinov.Substring(sinov.LastIndexOf(' ') + 1); if ((rt[0] == 'C' || rt[0] == 'S') && (rt[1] == 'h' || rt[1] == 'H')) rs = "" + rt[0] + rt[1] + ". " + rs; else rs = rt[0] + ". " + rs; sinov = sinov.Substring(0, sinov.LastIndexOf(' ')); } rs = rs + sinov; varoq.get_Range("Q" + i, "T" + i).MergeCells = true; varoq.get_Range("Q" + i, "T" + i).Name = ("QTQ" + i); dastur.Range mudir = varoq.get_Range("QTQ" + i, Type.Missing); varoq.get_Range("K" + i, "P" + i).MergeCells = true; varoq.get_Range("K" + i, "P" + i).Name = ("KPK" + i); dastur.Range mudir2 = varoq.get_Range("KPK" + i, Type.Missing); mudir2.Value = "Kafedra mudiri :"; mudir.Value = rs; i = 7; } } private void taqsimot_Load(object sender, EventArgs e) { yor.satr = "Select semestr from semestr"; listSemestrNomi.Clear(); listSemestrNomi.AddRange(yor.satrliMassiy()); yor.satr = "Select semestr from semestr"; comboBox1.Items.Clear(); foreach (string seme in listSemestrNomi) { sinov = seme.Substring(0, (seme.LastIndexOf('i') + 1)); if (!comboBox1.Items.Contains(sinov)) comboBox1.Items.Add(sinov); } comboBox2.Items.Clear(); yor.satr = "Select kafedra from kafedra"; comboBox2.Items.AddRange(yor.satrliMassiy()); } }}

Xulosa

Bu bob 2 ta bo’limdan iborat. Birinchi bo’limda yuklamani tayyorlash haqida ma’lumot va dastur kodi keltirilgan. Ikkinchi bo’limda tayyorlangan yuklamadan taqsimot hosil qilish haqida ma’lumot va dastur kodi keltirilgan.

XOTIMA

Oliy ta’lim muassasasidagi barcha kafedralarda kafedra tomonidan kafedraning o’quv yuklamasi va dars taqsimoti tayyorlanadi. Buni albatta namunaviy o’quv reja, ishchi o’quv reja va mavsumiy o’quv rejalar asosida tayyorlashadi. Bu yerda namunaviy o’quv reja oliy ta’lim vazirligi tomonidan har bir ta’lim yo’nalishi va ta’lim mutaxassisligi uchun tasdiqlab beriladi. Namunaviy o’quv reja asosida har bir kafedra o’zining ta’lim yo’nalishi uchun ishchi o’quv rejani tayyorlaydi. Ishchi o’quv rejalar asosida mavsumiy ishchi o’quv rejalarni tayyorlashadi. Kafedralar tomonidan tayyorlanadigan kafedra o’quv yuklamasi va dars taqsimotini bir xil strukturada tayyorlash va uni tayyorlash jarayonida vujudga keladigan qandaydir qiyinchiliklarni va kamchiliklarni bartaraf etish maqsadida kafedra o’quv yuklamasi va dars taqsimotini qo’llab quvatlovchi dasturiy tizim yaratildi.

Ushbu dasturiy tizimni yaratishda albatta C#dasturlash tili va ADO texnologiyasidan foydalanildi. ADO.NET bu shunday sinflar tashkil topganki, unda C# va .NET Frameworkda relyatsion jadvallar va ma’lumotlar bilan ishlashda qulaylik yaratadi. Relyatsion ma’lumotlar bazalari Microsoft SQL Server va Microsoft Access bilan bir qatorda boshqa relyatsion ma’lumotlar bazalari va hattoki relyatsion bo’lmagan ma’lumotlar manbalarni ham o’z ichiga oladi. ADO.NET texnologiyasi barcha .NET dan foydalanuvchi dasturlash tillarida .NET Framework va loyihalashni yaxlid holda birlashtirishda foydalaniladi, u C#ning o’ziga xos xususiyati hisoblanadi. ADO.NET ActiveX Data Objects (ADO) ning kengaytirilgan sinflar to'plami bo’lib, u ma’lumotdan Microsoftning oldingi avlod texnologiyasiga nisbatan tezroq foydalanish imkonini beradi.

Magisrlik dissertasiyasi 3ta bobdan iborat bo’lib, I-bobda berilganlar bazasini boshqarish tizimlari haqida ma’lumotlar keltirilgan. II- bob Microsoft Visual Studio 2010 muhitida ADO.Net texnologiyasiga bag’ishlangan. III-bobda dasturiy tizimni tayyorlash etablari keltirilgan.

ADABIYOTLAR RO'YXATI

1. I.A.Karimov "Yuksak ma'naviyat yengilmas kuch".Toshkent,2008, 176 b.
2. I.A.Karimov "O'zbekistonning o'z istiqlol va taraqqiyot yo'li".Toshkent "O'zbekiston". 1992-yil. 173-174 b.
3. Трей Неш. C# 2008 усконренный курс для профессионалов. Москва, Санкт-Петербург, Киев, 2008, -576с.
4. Павел Агуров. C# Сборник рецептов. СПб.: БХВ – Санкт-Петербург, 2008. – 432 с.: ил
5. Дейтел Х, Дейтел П, Листфилд Дж. C# Наиболее полное руководство В Подлиннике. СПб.: БХВ – Санкт-Петербург, 2006. – 1056 с.: ил
6. Лабор В. В. Си Шарп: Создание приложений для Windows.— Мн.: Харвест, 2003. - 384 с.
7. Шилдт Герберт. Полный справочник по C#. : Пер. с англ. — М. : Издательский дом "Вильяме", 2004. — 752 с. : ил.
8. ADO.NET: Обзор технологии [Электронный ресурс].– http://www.cyberguru.ru/dotnet/ado-net/adonet-overview.html.
9. Кариев, Ч. А. Технология Microsoft ADO. NET / Ч. А. Кариев. – М. : БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий – ИНТУИТ.ru, 2007.
10. О. Н. Евсеева, А. Б. Шамшев Классы, интерфейсы и делегаты в C#. 2005 : учебное пособие / сост. – Ульяновск : УлГТУ, 2008. -740с.
11. Роберт Виейра Программирование баз данных Microsoft SQL Server 2005 "Диалектика" Москва • Санкт-Петербург • Киев 2007г. 831 ст.
12. Shirinov Ziyomat "Kafedra o'quv yuklamasi va dars taqsimotini avtomatlashtirish masalasi". Tafakkur va talqin. Buxoro ,- 2014 y. 171-173 betlar.