O‘ZBEKISTON RESPUBLIKASI OLIY VA O‘RTA MAXSUS TA‘LIM VAZIRLIGI

O‘ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

MUHAMMAD AL –XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

T.A.XO‘JAKULOV

“MA’LUMOTLAR BAZASI”

DARSLIK

5350500 - Pochta aloqasi texnologiyasi

Toshkent 2022

UO'K: 083
K 177
KBK: 12.1981
Muallif: T.A.Xo'jaqulov / TATU, Toshkent 2022 y.

Darslik 17 ta bobdan iborat bo'lib, ma'lumotlar bazasi haqida tushuncha, ma'lumotlar bazasi tizimining arxitekturasi, ma'lumotlar bazasi modellari, relatsion ma'lumotlar modeli, relatsion algebra va relatsion hisoblash elementlari, ma'lumotlar bazasini rejalashtirish, loyihalash va administratorlash, ma'lumotlar bazasini normalashtirish mavzo'lari keltirilgan. Ma'lumotlar bazasini boshqarish uchun dasturlash tillari, SQL tili tushunchasi, ma'lumotlarni manipulyatsiya qilish, ma'lumotlarni tavsiflash, tranzaksiyalarni boshqarish, ma'lumotlar bazasini administratorlashlari ham keltirilgan. Ma'lumotlar bazasini boshqa dasturlash tillari bilan bog'lash uchun, ODBC interfeysi, C++ va SQL orqali yangi ma'lumotlar qo'shish, ADO texnologiyasidan foydalanish, ADO va C++ dan foydalanish hamda XML va ma'lumotlar bazasini prinsiplari muhokama etilib, ma'lumotlar bazasini boshqarish tizimlari me'yoriy hujjatlari keltirilgan.

Darslik oliy o'quv yurtining 5350500 - Pochta aloqasi texnologiyasi fakulteti ta'lim yo'nalishlari talabalari uchun mo'ljallangan bo'lib, unda axborot texnologiyalari sohasida faoliyat ko'rsatuvchilar foydalanishlari mumkin.

Учебник состоит из 17 глав, в которые включено следующее: понимание базы данных, архитектура системы баз данных, модели баз данных, реляционная модель данных, реляционная алгебра и реляционные вычислительные элементы, планирование базы данных, проектирование и администрирование, нормализация базы данных. Также в данном учебнике приведены языки программирования для управления базами данных, включая понятие языка SQL, манипулирование данными, описание данных, управление транзакциями и администрирование баз данных. В учебнике описано как связать базу данных с другими языками программирования, обсуждаются интерфейс ODBC, добавление новых данных через C ++ и SQL, использование технологии ADO, использование ADO и C ++, а также принципы XML и базы данных. В завершении представлены положения о системе менеджмента.

Учебник предназначен для студентов направления 5350500–Почтовый сервис высших учебных заведений и может быть использован для сферы информационных технологий.

The tutorial consists of 17 chapters, which include: Understanding the Database, Database System Architecture, Database Models, Relational Data Model, Relational Algebra and Relational Computing Elements, Database Planning, Design and Administration, Database Normalization. This tutorial also covers programming languages for database management, including the concept of SQL, data manipulation, data description, transaction management, and database administration. The tutorial describes how to bind a database with other programming languages, discusses the ODBC interface, adding new data via C ++ and SQL, using ADO technology, using ADO and C ++, and XML and database principles. Finally, the provisions on the management system are presented.

The textbook is intended for students in the direction 5350500 – Postal service of higher educational institutions and can be used for the field of information technology.

Taqrizchilar:

Nazirova E.Sh. – TATU "Multimediya texnologiyalari" kafedrasi mudiri t.f.d

Nurjanov F.R. – TDAU "Qishloq xo'jaligini raqamlashtirish va axborot-texnologiyalari masalalari bo'yicha" prorektor t.f.n.

MUNDARIJA

Kirish

Ma'lumotlar omborlari doimo axborot tizimlarini o'rganishda muhim mavzu bo'lib kelgan. So'ngi yillarda internetning ommaviyligi oshishi va internet uchun yangi texnologiyalarning jadal rivojlanishi oqibatida, ma'lumotlar bazasi texnologiyalari borasidagi bilim rivojlanishning eng muhim yo'nalishlaridan biriga aylandi. Shu bilan birga, internet texnologiyalari foydalanuvchilarga, ma'lumotlar bazasini nashr etishning standart va arzon vositalarini taqdim etadi. To'g'ri, ushbu yangi ishlanmalarning hech biri internet va biznes ehtiyojlari uchun yaratilishidan oldin paydo bo'lgan klassik ma'lumotlar bazasi ilovalariga bo'lgan ehtiyojni bekor qilmaydi. Bu ma'lumotlar bazalarini bilishning ahamiyatini kengaytiradi.

Ko'pgina talabalar ushbu mavzuni murakkabligiga qaramay yoqimli va qiziqarli deb bilishadi. Ma'lumotlar bazasini yaratish va ishlab chiqish ham san'at, ham mahorat talab qiladi. Foydalanuvchi talablarini tushunish va ularni samarali ma'lumotlar bazasi yaratish va uni joriy etilishi ijodiy jarayon deb atash mumkin. Ushbu loyihalarni bajarishda funktsional to'liq va yuqori unumli dasturlardan foydalangan holda ma'lumotlar bazalariga o'tkazish, muhandislik jarayonidir. Ikkala jarayon ham murakkabliklarga intellektual jumboqlarga to'la. Ma'lumotlar bazasi texnologiyalarini rivojlantirishga katta ehtiyoj mavjud bo'lganligi sababli, siz rivojlantiradigan ko'nikmalar va ushbu kursni o'rganish jarayonida olgan bilimlaringiz talabga ega bo'ladi.

Kitobning maqsadi – ma’lumotlar bazasi bilan ishlaydigan ushbu sohada muvaffaqiyatli martaba boshlashingiz uchun ma'lumotlar bazasi texnologiyasining asosiy printsiplariga mustahkam poydevor berishdan iboratdir.

Ushbu bo'limda biz ma'lumotlar bazalarida nima, qanday va nima uchun muhokama qilinnishi haqida tushuntirilgan. Ma'lumotlar bazalari nima uchun ishlatilishini tushintirilgan. Sizga ma'lumotlar bazasi tizimining qanday tarkibiy qismlari mavjudligini va bunday tizimlarni qanday yaratishni aytib beramiz.

Shaxsiy kompyuterlarning tarqalishi tobora kengayib borayotganligi sababli, ma'lumotlar bazasi rasmiy tashkil qilishning ahamiyati doimiy ravishda o'sib bormoqda. Asosiy nashrlar 1983-1987 yillarga oid ma'lumotlar bazalarini yaratish tushunchalari va asosiy printsiplarini qamrab olgan o'quv adabiyotlari manbalariga bog'langan. Keyinchalik adabiyotlar manbalariga havola bilan nashr etilgan "Ma'lumotlar bazarlari boshqarish tizimlari" kursi bo'yicha darsliklar ham aslida bu yillar darajasini aks ettiradi.

Shu bilan birga, so'nggi o'n yil ichida ma'lumotlar bazasi sohasida jiddiy o'zgarishlar ro'y berdi, deyarli o'quv adabiyotlarida aks etmadi. Bunga taqsimlangan, obyektga yo'naltirilgan va obyektga oid ma'lumotlar bazalari, ma'lumotlar bazasini loyihalash va dasturiy avtomatlashtirish vositalari, SQLda tuzilgan so'rovlar tili va boshqa ko'plab muammolar kiradi.

Amaldagi ma'lumotlar bazasini boshqarish tizimidagi adabiyotlar manbalariga va Windows-ning turli xil ilovalarida ko'plab adabiyotlar havolalari tugmachalarga qadar belgilanadi, bu nafaqat dasturiy vositalarning o'zlarini, balki ularni qurish tamoyillari va qoidalarini ham tushunishni qiyinlashtiradi. Bundan tashqari, ko'pincha turli xil manbalarda turli xil nomlarga ega bo'lgan, ba'zan

aniqlanmaydigan yoki yangi atamalarni kiritadi. Ma'lumotlar bazalari bo'yicha ko'plab alohida savollar xorijiy jurnallarda chop etiladi. Bundan tashqari, ma'lumotlar bazasining zamonaviy kontseptsiyasi fonida, bir qator tushunchalar va qoidalar o'zgartirildi.

Obyektga yo'naltirilgan va taqsimlangan ma'lumotlar bazasi, ma'lumotlar bazasini loyihalashni avtomatlashtirish kabi yangi va istiqbolli sohalarni o'z ichiga olgan, zamonaviy ma'lumotlar bazasining taqdimotini muntazam ravishda tashkil etadi.

Darslik uch bo’limdan iborat.

Birinchi bo'limda ma'lumotlar bazasining asosiy tushunchalari, tasnifi, tarkibi va ishlashi ko'rib chiqiladi. Ma'lumotlar bazasining kontseptsiyasi va metodologiyasi muhokama qilinadi, ma'lumotlar bazalarining umumiy nazariyasi taqdim etiladi.

Ikkinchi qismda ma'lumotlar ierarxik, tarmoq va relyatsion modellarning xususiyatlari, modellarni integratsiyalash masalalari, ma'lumotlar bazasini jismoniy amalga oshirish usullari o'rganiladi. Ma'lumotlar bazasini yaratish muammolari, uni avtomatlashtirish usullari muhokama qilinadi. Relyatsion ma'lumotlar bazasining kamchiliklarini, ob'ektga yo'naltirilgan ma'lumotlar bazasini qurish zarurati va usullarini ko'rib chiqilgan.

Uchinchi qism, taqsimlangan ma'lumotlar bazasi (TMB) ma'lumotlar bazasiga bag'ishlangan. Ularning xususiyatlari, arxitekturasi, texnologiyasi va mijoz-server tizimi muhokama qilinadi. Ma'lumotni qismlarga ajratish va lokalizatsiya qilish, ma'lumotlar bazasini loyihalashtirish bosqichlari sifatida mahalliy ma'lumotlar bazalarini birlashtirish ko’rib chiqilgan. TMB yaratish va undan foydalanish xususiyatlari o'rganilmoqda. TMB dizaynining tizimli taqdimoti berilgan.

I Bob. Ma'lumotlar bazasi faniga kirish

Sharh

Hisoblash tizimlarining rivojlanib borishi, tobora ko'payib borayotgan ma'lumotlarning elektron (mashinada o'qiladigan) rasmini saqlashni talab qiladi. Hisoblash tizimlarini takomillashtirish va yanada rivojlantirish bilan bir qatorda, qayta ishlanadigan va saqlanadigan ma'lumot hajmi o'sadi. Amaliyotda tuzilgan saqlash va ortib borayotgan hajmlarni samarali qayta ishlash muammolarini hal qilishda paydo bo'lgan qiyinchiliklar, tegishli sohalarda tadqiqotlarni rag'batlantiradi. Ma'lumotni saqlash va qayta ishlash vazifalari rasmiylashtirildi. Ushbu sinfning muammolarini hal qilish uchun nazariy baza yaratildi, uning amalda bajarilishi natijasida ma'lumotlarga ishlov berish, saqlash va ulardan foydalanishni ta'minlashga mo'ljallangan tizimlar yaratildi. Keyinchalik bunday tizimlar ma'lumotlar bazasi tizimlari deb nomlana boshladi.

Ma'lumotlar bazasi tizimlarini rivojlantirish bilan bir qatorda, katta hajmdagi ma'lumotlar bilan ishlashda foydalaniladigan kompyuter texnologiyalarining jadal rivojlanishi kuzatildi. Hisoblash kuchi va ayniqsa dastlabki hisoblash tizimlarining saqlash imkoniyatlari ma'lumotni amalda zarur bo'lgan hajmda saqlash va qayta ishlash uchun etarli emas edi.

Ma'lumotlar bazasi tizimlari rivojlanib borishi bilan ulardagi ma'lumotlarni tashkil qilish tamoyillari o'zgardi: dastlab ma'lumotlar ierarxik va keyinchalik tarmoq modeli asosida taqdim etildi. 70-yillarning oxiri va 1980-yillarning boshlarida birinchi relyatsion mahsulotlar paydo bo'la boshladi. Hozirgi kunda relyatsion modelga asoslangan ma'lumotlar bazasi tizimlari, ko'plab tadqiqotchilarning obyektga yo'naltirilgan tizimlarga o'tish to'g'risida yaqinda aytganlariga qaramay, etakchi o'rinni egallab turibdi. Hozirgi vaqtda obyektlarga yo'naltirilgan tizimlar rivojlanmoqda, garchi ularning rivojlanish sur'ati tegishli standartlarning sekin qabul qilinishi bilan to'sqinlik qilsada. Bundan tashqari, ko'plab tijorat aloqalari tizimlari obyektga yo'naltirilgan xususiyatlarga ega bo'lmoqdalar. Shundan kelib chiqqan holda, kelajakda obyektga yo'naltirilgan tizim asta-sekin relyatsion tizimlarni almashtiradi deb taxmin qilish mumkin .

Hozirgi vaqtda quyidagi yo'nalishlar bo'yicha tadqiqotlar olib borilmoqda:

1. deduktiv tizimlar;
2. ekspert tizimlar;
3. kengaytiriladigan tizimlar;
4. ob'ektga yo'naltirilgan tizimlar;
5. NoSQL

1.1 Ma'lumotlar omborlarining rivojlanish tarixi

Ma'lumotlar bazasini rivojlantirish tarixi 30 yildan ortiq. 1968 yilda IBM tomonidan birinchi sanoatda ishlatiladigan ma’lumotlar bazasini boshqarish tizimlari (MBBT), IMS tizimi foydalanishga topshirildi. 1975 yilda ma'lumotlarni qayta ishlash tizimlari tillari assotsiatsiyasining birinchi standarti paydo bo'ldi -

Conference of Data System Languages (CODASYL), u ma'lumotlar omborsi tizimlari nazariyasida tarmoq ma'lumotlari modeli uchun hali ham muhim bo'lgan bir qator fundamental tushunchalarni aniqladi.

Ma'lumotlar bazasi nazariyasining keyingi rivojlanishiga relyatsion ma'lumotlar modelini yaratuvchisi bo'lgan amerikalik matematik E.F. Codd katta hissa qo'shdi. 1981 yilda E.F. Codd relyatsion model va relyatsion algebrani yaratganligi uchun Amerika kompyuter fanlari uyushmasidan nufuzli Tyuring mukofotiga sazovor bo'ldi.

Shu paytdan boshlab yigirma yildan ko'proq vaqt o'tdi, ammo kompyuter texnologiyalarining jadal rivojlanishi, uning jamiyatdagi asosiy rolining o'zgarishi, shaxsiy kompyuterlarning rivojlanishi va nihoyat, kuchli ish stantsiyalari va kompyuter tarmoqlarining paydo bo'lishi ham ma'lumotlar bazasi texnologiyalarining rivojlanishiga ta'sir ko'rsatdi. Ma'lumotlarni qayta ishlashda ushbu yo'nalishni rivojlantirishning to'rtta bosqichi mavjud. Shu bilan birga, shuni ta'kidlash kerakki, ushbu bosqichlarda hali ham qiyinchiliklar mavjud emas: ular bir-birini silliq ravishda o'tishadi va hatto parallel ravishda birga yashaydilar, ammo shunga qaramay, ushbu bosqichlarni ajratish ma'lumotlar bazasi texnologiyasini rivojlantirishning individual bosqichlarini yanada aniqroq tavsiflashga imkon beradi, ma'lum bir bosqichning o'ziga xos xususiyatlarni ta'kidlaydi.

Ma'lumotlar bazasini rivojlantirishning birinchi bosqichi IBM 360/370, EC-EHM va PDP11 mini-EHM (Digital Equipment Corporation - DEC dan), turli xil HP modellari (Hewlett Packard-dan) kabi yirik mashinalarda ma'lumotlar bazalarini tashkil qilish bilan bog'liq.

Ma'lumotlar bazalari markaziy EHMning tashqi xotirasida saqlanar edi va bu ma'lumotlar bazasining foydalanuvchilari, asosan, to'plam rejimda bajariladigan vazifalarga kiritilgan. O'zining hisoblash resurslariga (protsessor, tashqi xotira) ega bo'lmagan va faqat markaziy kompyuter uchun kirish-chiqish moslamalari sifatida xizmat qiluvchi, konsol terminallari yordamida interaktiv kirish rejimi ta'minlandi. Ma'lumotlar bazasiga kirish dasturlari turli tillarda yozilgan va oddiy raqamli dasturlar sifatida boshlangan. Kuchli operatsion tizimlar ko'plab vazifalarning shartli ravishda parallel ravishda bajarilishini ta'minladi. Ushbu tizimlarni taqsimlangan kirish tizimlariga kiritish mumkin, chunki ma'lumotlar bazasi markazlashtirilgan, bitta markaziy EHMning tashqi xotira qurilmalarida saqlangan va unga kirish ko'plab foydalanuvchilarning vazifalari tomonidan qo'llab-quvvatlangan.

Rivojlanishning ushbu bosqichining xususiyatlari quyidagicha ifodalanadi:

- Barcha ma'lumotlar omborlari kuchli ko'p vazifali operatsion tizimlarga (MVS, SVM, RTE, OSRV, RSX, UNIX) asoslangan, shuning uchun taqsimlangan ko'p foydalanuvchilarga kirish rejimida markazlashtirilgan ma'lumotlar bazasi bilan ishlash asosan qo'llab-quvvatlanadi.
- Resurslarni taqsimlashni boshqarish funktsiyalari asosan operatsion tizim (OT) tomonidan amalga oshiriladi.

- Qo'llab-quvvatlanadigan tillar - bu ma'lumotlarga kirish uchun navigatsiya usullariga yo'naltirilgan ma'lumotlarning past darajadagi manipulyatsiyasi.
- Ma'lumot ma'muriyatiga katta ahamiyat beriladi.
- Relyatsion ma'lumotlar modelini asoslash va rasmiylashtirish bo'yicha jiddiy ishlar olib borilmoqda va relyatsion ma'lumotlar modelining mafkurasini amalga oshiradigan birinchi tizim (System R) yaratildi.
- So'rovlarni optimallashtirish va markazlashtirilgan ma'lumotlar bazasiga taqsimlangan kirishni boshqarish bo'yicha nazariy ishlar olib borilmoqda, tranzaktsiya tushunchasi kiritildi.
- Relyatsion ma'lumotlar modeli bilan ishlash uchun birinchi yuqori darajadagi tillar paydo bo'ldi. Biroq, ushbu birinchi tillar uchun standartlar yo'q.

Shaxsiy kompyuterlar tezda bizning hayotimizga kirib, kompyuter texnologiyalarining jamiyatdagi o'rni va roli haqidagi tushunchamizni o'zgartirdi. Endi kompyuterlar har bir foydalanuvchiga yanada yaqinroq va qulayroqdir. Oddiy foydalanuvchilarning tushunarsiz va murakkab dasturlashdan oldingi qo'rquvi yo'qoldi. Tayyor bo'lmagan foydalanuvchilar uchun mo'ljallangan ko'plab dasturlar mavjud. Ushbu dasturlardan foydalanish oson va qulay bo’lgan.

Bular asosan turli xil matn muharrirlari, jadvallar va boshqalardan tashkil topgan. Fayllarni nusxalash va ma'lumotlarni bitta kompyuterdan boshqasiga o'tkazish, matnlar, jadvallar va boshqa hujjatlarni chop etish operatsiyalari sodda va tushunarli bo'ldi. Har bir foydalanuvchi ushbu kuchli va qulay qurilmaning to'liq egasi kabi his qilishi mumkin, bu sizga faoliyatning ko'p qirralarini avtomatlashtirish imkonini beradi. Va albatta, bu, ma'lumotlar bazalari bilan ishlashga ta'sir ko’rsatdi. Ma'lumotlar bazasini boshqarish tizimlari deb ataladigan dasturlar paydo bo'ldi va ular katta hajmdagi ma'lumotlarni saqlashga imkon berdi, ma'lumotlarni to'ldirish uchun qulay interfeysga ega va turli xil hisobotlarni tuzish uchun o'rnatilgan vositalarga ega. Ushbu dasturlar ilgari qo'lda bajarilgan ko'plab buxgalteriya funktsiyalarini avtomatlashtirishga imkon berdi. Shaxsiy kompyuterlar narxlarining doimiy pasayishi ularni nafaqat tashkilotlar va firmalar, balki alohida foydalanuvchilar uchun ham taqdim etdi. Kompyuterlar hujjatlarni saqlash va o'zlarining buxgalteriya funktsiyalarini yuritish vositasiga aylandi. Bularning barchasi ma'lumotlar bazasini yaratish sohasida ham ijobiy, ham salbiy rol o'ynadi. Shaxsiy kompyuterlar va ularning dasturiy ta'minotining soddaligi va ochiqligi ko'rinishi ko'plab havaskorlarga sabab bo'ldi. Ushbu ishlab chiquvchilar o'zlarini mutaxassis deb hisoblab, real dunyo obyektlarining ko'pgina xususiyatlarini hisobga olmaydigan qisqa muddatli ma'lumotlar omborlarini yaratishni boshladilar. Rivojlanish qonunlariga va haqiqiy jismlarning o'zaro munosabatlariga to'g'ri kelmaydigan juda ko'p bir kunlik tizimlar yaratilgan. Biroq, shaxsiy kompyuterlarning mavjudligi ko'pgina bilim sohalarida ilgari kompyuter texnologiyalaridan foydalanmagan foydalanuvchilarni ularga murojaat qilishga majbur qildi. Va ishlab chiqilgan qulay ma'lumotlarni qayta ishlash dasturlariga bo'lgan talab dasturiy ta'minotni sotuvchilarni, odatda, ish stoli ma'lumotlar bazasi

deb nomlanadigan barcha yangi tizimlarni etkazib berishga majbur qildi. Ushbu tizimlarni takomillashtirishga majbur bo'lgan etkazib beruvchilar o'rtasida jiddiy raqobat, yangi xususiyatlarni taqdim etish, tizimlarning interfeysi va tezligini yaxshilash, ularning narxini pasaytirishga olib kelindi. Bozorda o'xshash funktsiyalarni bajaradigan ko'p sonli MBBT mavjudligi ushbu tizimlar uchun eksport va import usullarini ishlab chiqishni va ma'lumotlarni saqlash formatlarini ochishni talab qildi.

Ammo bu davrda ham havaskorlar bo'lgan, ular odatdagidan farqli o'laroq, standart dasturlash tillaridan foydalanib o'zlarining ma'lumotlar bazalarini ishlab chiqdilar. Bu halokatli yakun bo'lgan, chunki keyingi rivojlanish shuni ko'rsatdiki, nostandart formatlardan yangi MBBT-larga ma'lumotlarni uzatish ancha qiyin bo'lgan va ba'zi hollarda bu juda ko'p mehnat talab qilar edi, shuning uchun hammasini qayta ishlab chiqish osonroq edi, ammo ma'lumotlar hali ham yangi boshqalarga o'tkazilishi kerak bo'lgan. Bu, shuningdek, MBBT bajarishi kerak bo'lgan funktsiyalarni noto'g'ri baholagandan kelib chiqdi.

Ushbu bosqichning xususiyatlari quyidagilar:

- barcha ma'lumotlar omborlari yizimlari, ma'lumotlar bazasini, asosan, eksklyuziv foydalanish uchun yaratishga mo'ljallangan. Shaxsiy kompyuter, u tarmoqqa ulanmagan va bitta foydalanuvchi uchun ma'lumotlar bazasi yaratilgan. Kamdan kam hollarda bir nechta foydalanuvchilar ketma-ket ishlashlari kerak edi, masalan, avval buxgalteriya hujjatlarini kiritgan operator, so'ngra asosiy hujjatlarga mos keladigan operatsiyalarni aniqlagan bosh buxgalter ishlagan.
- ko'pgina ma'lumotlar bazasidagi ma'lumotlarni joylashtirish qulay foydalanuvchi interfeysiga ega bo'lgan. Ko'pincha ma'lumotlar bazasi tavsifi doirasida ham, so'rovlarni loyihalash doirasida ham ma'lumotlar bazasi bilan ishlashning interfaol rejimi mavjud. Bundan tashqari, ko'pgina MBBTlar dasturiy ta'minotsiz tayyor dasturlarni ishlab chiqish uchun ishlab chiqilgan va qulay vositalarni taklif qildi. Instrumental muhit shunchaki bitta kompleksga yig'ilishi mumkin bo'lgan ekran shablonlari, hisobotlar, yorliqlar, grafik so'rovlar dizaynerlari ko'rinishidagi tayyor dastur elementlaridan iborat.
- barcha ish stoli ma'lumotlar bazasidagi faqat relatsiya modelning tashqi ko'rinish darajasi, ya'ni ma'lumotlar strukturasining tashqi ko'rinishi qo'llab-quvvatlandi.
- relyatsion algebra va SQL kabi yuqori darajadagi ma'lumotlar manipulyatsiyasi tillari mavjud bo'lganda, ish stolining MBBT-da alohida jadval satrlari darajasida past darajadagi ma'lumotlarni boshqarish tillari qo'llab-quvvatlandi.
- ma'lumotlar bazasining tarkibiy yaxlitligini ta'minlash uchun hech qanday vosita yo'q edi. Ushbu funktsiyalar amaliy dasturlar tomonidan bajarilishi kerak, ammo dasturlarni ishlab chiqish vositalarining kamligi ba'zan bunga imkon bermadi va bu holda foydalanuvchi ushbu ma'lumotlar

bazasida saqlanadigan ma'lumotni kiritish va o'zgartirish paytida qo'shimcha nazoratni talab qiladigan bajarishi kerak.

- monopol rejimda ishlash amalda ma'lumotlar bazasini boshqarish funktsiyalarining yomonlashuviga va shu munosabat bilan ma'lumotlar bazasini boshqarish vositalarining etishmasligiga olib keldi.
- va nihoyat, oxirgi va hozirgi paytda juda ijobiy xususiyat bu ish stoli ma'lumotlar bazasining nisbatan kam talabga ega uskunalari paydo bo'ldi. Masalan, Clipper-da ishlab chiqilgan juda ko'p funktsional ilovalar, kompyuter 286-da ishlaydi.
- aslida, ularni hatto to'laqonli ma'lumotlar bazasi deb ham atash qiyin. Ushbu oilaning yorqin vakillari yaqin vaqtgacha juda keng qo'llanilgan DBMS (MBBT), Dbase (DbaseIII +, DbaselV), FoxPro, Clipper, Paradoxlardir.

1.2 Fayl tizimlari

Markazlashtirilgan fayllarni boshqarish tizimlaridan foydalanishga o'tish tarixiy qadam bo'ldi. Dastur nuqtai nazaridan, fayl - bu yozilishi va o'qilishi mumkin bo'lgan tashqi xotiraning nomlangan maydoni. Fayllarni nomlash qoidalari, faylda saqlanadigan ma'lumotlarga kirish usuli va bu ma'lumotlarning tuzilishi ma'lum fayllarni boshqarish tizimiga va ehtimol, fayl turiga bog'liq. Fayllarni boshqarish tizimi tashqi xotirani tarqatish, tashqi xotiradagi tegishli manzillarga fayl nomlarini xaritalash va ma'lumotlarga kirishni ta'minlash bilan shug'ullanadi.

Birinchi ishlab chiqilgan fayl tizimi IBM tomonidan o'zining 360 seriyali uchun ishlab chiqilgan bo'lib, hozirda u juda eskirgan va biz uni batafsil ko'rib chiqmaymiz. Biz shuni ta'kidlaymizki, ushbu tizimda faqat ketma-ket va indeksli ketma-ketlikdagi fayllar qo'llab-quvvatlandi va amalga oshirish o'sha paytlarda paydo bo'lgan disk qurilmasi kontrollerlarining imkoniyatlariga ko'proq tayanadi. OS / 360-da fayl tushunchasi har qanday tashqi ob'ektga, shu jumladan tashqi qurilmalarga mos keladigan asosiy mavhum tushuncha sifatida tanlanganligini hisobga olib, foydalanuvchi darajasida fayllar bilan ishlash juda noqulay bo'lgan. Katta hajmdagi va haddan tashqari yuklangan tuzilmalar kerak edi. Bularning barchasi IBM kompyuterlarining mahalliy analoglaridan foydalanishni boshlagan o'rta va katta dasturchilarga yaxshi ma'lum.

Fayl tuzilmalari

Keyinchalik zamonaviy fayl tizimlari tashkilotlari haqida tushintiramiz.

Fayl tuzilmalari - deyarli barcha zamonaviy kompyuterlarda asosiy tashqi xotira qurilmalari harakatlanuvchi boshli magnit disklar bo'lib, ular fayllarni saqlash uchun ishlatiladi. Bunday magnit disklar bu magnit plitalar (sirtlar) to'plami bo'lib, ular orasida magnit boshlari to'plami bir qo'lda harakatlanadi. Boshlar paketining harakatlanish bosqichi diskretdir va har bir to'plamning pozitsiyasi magnit diskning silindriga to'g'ri keladi. Har bir sirtda silindr yo'lni "o'yib qo'yadi", shunda har bir sirt silindr soniga teng bo'lgan izlarni o'z ichiga oladi. Magnit diskni belgilashda (diskni ishlatishdan oldin maxsus harakat), har bir trek bir xil blokda belgilanadi, shunda har bir blokga maksimal baytlar yozilishi mumkin. Shunday

qilib, apparat darajasida magnit disk bilan almashish uchun siz ushbu blokning boshidan boshlab yozilishi yoki o'qilishi kerak bo'lgan silindrning raqamini, sirt raqamini, tegishli trekdagi blok raqamini va baytlarning sonini ko'rsatishingiz kerak.

Faylga nom berish

Fayllarni nomlash usullari haqida qisqacha to'xtalib o'tamiz. Barcha zamonaviy fayl tizimlari maxsus tuzilishga ega bo'lgan qo'shimcha fayllar - tashqi xotirada kataloglarning saqlanishi tufayli ko'p darajali fayl nomini qo'llab-quvvatlaydi. Har bir katalogda ushbu katalogdagi kataloglar va / yoki fayllarning nomlari mavjud. Shunday qilib, to'liq fayl nomi katalog nomlari ro'yxatidan va to'g'ridan-to'g'ri faylni o'z ichiga olgan katalogdagi fayl nomidan iborat. Turli xil fayl tizimlarida fayllarni nomlash usullari o'rtasidagi farq, bu nomlar zanjiri boshlanadi.

Shu munosabat bilan ikkita ekstremal variant mavjud. Ko'pgina fayllarni boshqarish tizimlari har bir fayl arxivini (to'liq katalog daraxti) to'liq bitta disk paketida (yoki mantiqiy disk, operatsion tizimni alohida disk sifatida taqdim etadigan disk paketining bo'limi) talab qiladi. Bunday holda, to'liq fayl nomi tegishli disk o'rnatilgan disk qurilmasining nomi bilan boshlanadi.

Fayl himoyasi

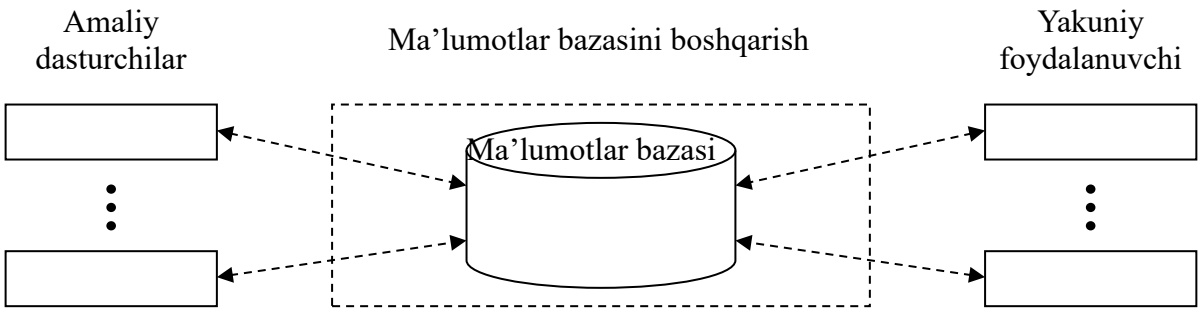
Fayl tizimlari turli xil foydalanuvchilarga tegishli bo'lgan fayllarning umumiy bazasi bo'lganligi sababli fayllarni boshqarish tizimlari fayllarga kirish uchun avtorizatsiyani ta'minlashi kerak. Umuman olganda, yondashuv shundan iboratki, ushbu hisoblash tizimining har bir ro'yxatdan o'tgan foydalanuvchisiga nisbatan mavjud bo'lgan har bir fayl uchun ushbu foydalanuvchiga ruxsat berilgan yoki taqiqlangan harakatlar ko'rsatilgan. Ushbu yondashuvni to'liq amalga oshirishga urinishlar bo'ldi. Ammo bu ortiqcha ma'lumotni saqlashda ham, undan foydalanish huquqini boshqarish uchun foydalanishda ham ortiqcha xarajatlarga olib keldi.

Ko'p foydalanuvchilarga kirish rejimi

Tarixiy jihatdan, fayl tizimlarida quyidagi yondashuv ishlatilgan. Faylni ochiq ishlashida (fayl sessiyasi boshlanishi kerak bo'lgan birinchi va majburiy operatsiya), boshqa parametrlarga qo'shimcha ravishda ish rejimi (o'qish yoki o'zgartirish) ko'rsatildi. Agar o'sha paytda ushbu operatsiya A dasturi nomidan amalga oshirilgan bo'lsa, fayl boshqa biron bir B dasturi uchun allaqachon ochilgan bo’ladi ("jarayon" deyish to'g'riroq, lekin biz terminologik noziklarga o'tmaymiz) va mavjud ochilish rejimi kerakli rejimga mos kelmadi (faqat o'qish rejimlari mos keladi), tizimning xususiyatlariga qarab, A dasturiga faylni kerakli rejimda ochishning mumkin emasligi to’g’risida xabar berilgan yoki u B dasturi yopishni tugatmaguncha bloklangan fayldir.

1.3 MB va MBBT tushunchasi

Ma'lumotlar bazasi tizimi - bu kompyuterlashtirilgan tizim bo'lib, uning asosiy vazifasi ma'lumotlarni saqlash va unga talab bo'yicha kirishni ta'minlash hisoblanadi.



1.1 -rasm. Ma'lumotlar bazasi tizimi.

Ma'lumotlar bazasi tizimiga quyidagilar kiradi (1.1-rasm):

1. ma'lumotlar bazasida bevosita saqlanadigan ma'lumotlar;
2. apparat;
3. dasturiy ta'minot;
4. foydalanuvchilar;
5. dastur dasturchilari;
6. oxirgi foydalanuvchilar;
7. ma'lumotlar bazasi ma'murlari.

Ma'lumotlar bazasidagi ma'lumotlar integratsiyalashgan va umumiy ravishda taqsimlanadi. Integratsiyalashgan ma'lumotlar kontseptsiyasi ma'lumotlar bazasini to'liq yoki qisman bir-biriga zid bo'lmagan bir nechta alohida ma'lumotlar fayllarining kombinatsiyasi sifatida taqdim etish imkoniyatini anglatadi. Umumiy tushuncha ma'lumotlar bazasida ma'lumotlarning alohida maydonlarini bir necha turli foydalanuvchilar tomonidan ishlatish imkoniyatini nazarda tutadi.

Apparat tizimiga - kiritish-chiqarish qurilmalari, qurilma dispetcherlari, dasturlarning ishlashini qo'llab-quvvatlash uchun ishlatiladigan kompyuter texnologiyalari.

Dasturiy ta'minot - shaxsiy kompyuter ma'lumotlar bazasini o'zi va tizim foydalanuvchilari o'rtasidagi oraliq qatlam bo'lib, ma'lumotlar bazasi menejeri yoki ma'lumotlar bazasini boshqarish tizimi, MBBT(DBMS) deb nomlanadi. Foydalanuvchilarning barcha so'rovlari ma'lumotlari bazasida ko'rib chiqiladi.

MBBT- bu ma'lumotlar bazasi foydalanuvchisiga dasturiy ta'minot darajasida ma'lumotlarni saqlash tafsilotlariga kirmasdan ishlash imkoniyatini beradigan maxsus dastur.

Ilova dasturchilari - ma'lumotlar bazasidan foydalanadigan amaliy dasturlarni yozish uchun javobgardir.

Yakuniy foydalanuvchilar - ma'lumotlar bazasi bilan bevosita ish stantsiyasi yoki terminal orqali ishlash tushuniladi. Yakuniy foydalanuvchi tegishli dasturiy ta'minot yordamida ma'lumotlar bazasiga kirish huquqiga ega.

Ma'lumotlar bazasi administratori - ma'lumotlar bazasini yaratadigan, ma'lumotlar bazasini texnik nazoratini va boshqa operatsiyalarni bajaradigan texnik mutaxassislardir. Ma'lumotlar bazasi ma'murlari ma'lumotlar boshqaruvchisi qarorlarini amalga oshirish uchun javobgardir. Ma'lumotlar adminstratorlash ma'lumotlar bazasida qaysi ma'lumotni saqlash kerakligini hal qiladi, ma'lumotlar

bazasida saqlanadigan ma'lumotdan foydalanish va undan foydalanish paytida tartibni ta'minlaydi.

Ma'lumotlar bazasi administratori funktsiyalari:

kontseptual doirani aniqlash; MB administratori, ma'lumotlar bazasida qaysi ma'lumotlarni saqlash kerakligini aniqlaydi. Ushbu jarayon odatda mantiqiy (yoki kontseptual) ma'lumotlar bazasini adminstratorlash loyihalash deb ataladi. Ma'lumotlar bazasini mazmunini mavhum darajada aniqlagandan so'ng, ma'lumotlar bazasi administratori kontseptual DDL yordamida tegishli konseptual sxemani yaratadi.

Ichki zanjirning ta'rifi. MB administratori, ma'lumotlarning saqlanadigan omborda qanday taqdim etilishini hal qiladi. Fizik dizaynni tugatgandan so'ng, ichki YaOD-dan foydalangan holda ma'lumotlar bazasi administratori tegishli saqlash tuzilishini yaratishi, shuningdek ichki va kontseptual dizayn o'rtasidagi xaritani aniqlab olishi kerak.

Foydalanuvchilar bilan o'zaro aloqa. MB administratori, foydalanuvchilarga kerakli ma'lumotlarni taqdim etadi. Buning uchun MB administratori, kerakli tashqi sxemalarni yozishi (yoki foydalanuvchilarga yozma ravishda yordam berishi) kerak. Bundan tashqari, tashqi va kontseptual sxemalar o'rtasidagi xaritani aniqlash kerak.

1. Xavfsizlik va yaxlitlik qoidalarini aniqlash.
2. Zaxira va tiklash protseduralarini aniqlang.
3. Unumdorlikni boshqarish va o'zgaruvchan talablarga javob berish.

Ma'lumotlar bazasi korxonaning amaliy tizimlari tomonidan foydalaniladigan doimiy ma'lumotlar to'plamidan iborat. "Doimiylar" so'zi boshqa, o'zgaruvchan ma'lumotlardan, masalan, oraliq ma'lumotlar va umuman, barcha tranzit ma'lumotlardan farq qiladigan ma'lumotlarni anglatadi. "Doimiy" ma'lumotlar aslida uzoq vaqt davomida saqlanib qolmasligi mumkin, chunki ma'lumotlar bazasidagi ma'lumotlar real dunyoning o'zgaruvchan ob'ektlarini va ular o'rtasidagi munosabatlarni aks ettirishi kerak.

Ma'lumotni saqlash uchun ma'lumotlar bazasidan foydalanish quyidagi afzalliklarni ta'minlaydigan ma'lumotlarning markazlashtirilgan boshqaruvini tashkil qilish imkonini beradi:

1. zaxira miqdorini kamaytirish imkoniyati;
2. nomuvofiqlikni (ma'lum darajada) bartaraf etish imkoniyati;
3. ma'lumot almashish imkoniyati;
4. standartlarga rioya qilish qobiliyati;
5. xavfsizlik cheklovlarini kiritish imkoniyati;
6. ma'lumotlar yaxlitligini ta'minlash qobiliyati;
7. qarama-qarshi talablarni muvozanatlash qobiliyati;
8. ma'lumotlarning mustaqilligini ta'minlash qobiliyati.

Dasturiy ta'minot ma'lumotlar bazasida saqlanadigan ma'lumotlardan ajratilganligi sababli, ko'p hollarda ma'lumotlar bazasi tuzilmasiga kiritilgan o'zgarishlar dasturiy ta'minotni tubdan o'zgartirishga olib kelmaydi.

Savollar:

1. Ma'lumotlar bazalari qanday rivojlandi?
2. Fayl tizimi nima?
3. Ko'p foydalanuvchilarga kirishda qanday muammolar mavjud?
4. Ma'lumotlar bazasi nima?
5. Ma'lumotlar bazasidan foydalanuvchilari nima?
6. Ma'lumotlar bazasi foydalanishning qanday afzalliklari bor?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

II Bob. Ma'lumotlar bazasi tizimining arxitekturasi

MBBT arxitekturasining 3 darajasi mavjud:

1. Ichki bosqich bu shaxsiy kompyuterga saqlashga eng yaqin. Bu ma'lumotni shaxsiy kompyuter qurilmalarida saqlash usullari bilan bog'liq;
2. Tashqi bosqich - foydalanuvchilarga eng yaqin ma’lumotlar bazasidir. Bu shaxsiy foydalanuvchilarga ma'lumotlarni taqdim etish usullari bilan bog'liq;
3. Kontseptual bosqich - bu birinchi va ikkinchi bosqich orasidagi oraliqdir. Bu daraja foydalanuvchilarning individual vakilliklari bilan bog'liq bo'lgan tashqi darajadan farqli o'laroq, foydalanuvchilarning umumlashtirilgan vakilliklari bilan bog'liq.

2.1 Tashqi bosqich - tashqi ishlash

Tashqi bosqich - individual foydalanuvchi bosqichi hisoblanadi. Foydalanuvchi yoki dasturchi yoki har qanday kasbiy tayyorgarlikka ega oxirgi foydalanuvchi bo'lishi mumkin. Har bir foydalanuvchi ma'lumotlar bazasi bilan aloqa qilishning o'ziga xos tiliga ega. Dasturchi uchun bu dasturlash tili, foydalanuvchi uchun, so'rovlar tili yoki rasmlar va menyularga asoslangan tildir. Ushbu tillarning har qandayida ma'lumotlar almashinuvi mavjud, ya'ni, faqat ob'ektlar va ma'lumotlar bazasi operatsiyalari bilan bog'liq bo'lgan butun tilning ko'plab operatorlaridir. Ma'lumotlar almashinuvi foydalanuvchilarning bazaviy tiliga qurilgan bo'lib, u ma'lumotlar bazasi bilan bog'liq imkoniyatlarni ham ta'minlaydi.

Tashqi arxitektura bosqichida ma'lumotlar bazasining foydalanuvchi tomonidan individual ko'rinishi tashqi ko'rinish deyiladi. Tashqi ko'rinish - bu foydalanuvchi tomonidan ko'rilgan ma'lumotlar bazasining tarkibidir. Masalan, kadrlar bo'limi xodimi ma'lumotlar bazasini xodimlar to'g'risidagi yozuvlar to'plami va bo'limlar to'g'risidagi yozuvlar to'plami sifatida ko'radi. Umumiy holda, tashqi

vakillik har bir tashqi yozuvning ko'p nusxalaridan iborat bo'lib, ular saqlangan yozuvlar bilan bir xil bo'lishi shart emas. Foydalanuvchi ma'lumotlari tashqi yozuvlar nuqtai nazaridan belgilanadi. Har bir tashqi vakillik asosan tashqi vakillik tarkibidagi har bir yozuv turining ta'riflaridan iborat tashqi sxema yordamida aniqlanadi.

2.2 Kontseptual bosqich - kontseptual ishlash

Kontseptual bosqich - bu ma'lumotlar bazasining barcha ma'lumotlarini shaxsiy kompyuterlarda saqlashning usuliga nisbatan biroz ko'proq ma'lumotlarni rasm va jadval ko'rinishida taqdim etish tushiniladi. Kontseptual bosqich foydalanuvchi ma'lumotni ma'lum bir tilda ko'rishga majbur bo'lganidek emas, balki haqiqatan ham uni taqdim etadi. Konseptual bosqich har bir turdagi kontseptual yozuvning ko'plab misollaridan iborat, ammo ba'zi tizimlarda ma'lumotlarni kontseptual taqdim etish usullari boshqacha bo'lishi mumkin. Masalan, ob'ektlar va ular o'rtasidagi munosabatlar ko'rinishida. Kontseptual vakillik kontseptual yozuvning har bir turining ta'riflaridan iborat bo'lgan kontseptual sxema yordamida aniqlanadi. Kontseptual sxemani aniqlashda tushunchalari faqat ma'lumot mazmuniga taalluqli ma'lumotlarning kontseptual ta'rifi tili qo'llaniladi. Kontseptual bosqich, ya'ni ma'lumotlarni saqlash usulidan mustaqilligini ta'minlaydi.

Ichki bosqich - ichki ko'rinish

Ichki bosqich - bu butun ma'lumotlar bazasining quyi bosqichdagi vakili hisoblanadi. Bu ichki yozuvning har bir turining ko'plab misollaridan iborat bo'ladi. Ichki yozuv saqlangan yozuvga mos keladi. Ichki bosqich jismoniy qatlam bilan bog'liq emas va jismoniy yozuvlarga murojaat qilmaydi. Ichki bosqich cheksiz chiziqli manzil maydonining mavjudligini taxmin qiladi. Tizimga kuchli bog'liqlik tufayli ushbu makonni jismoniy saqlash moslamalarini xaritalash tafsilotlari umumiy arxitekturaga kiritilmagan.

Ichki bosqich ichki ma'lumotni aniqlash tili yordamida tasvirlangan ichki sxema yordamida tavsiflanadi.

Uchta bosqich darajalari o'rtasida ikkita xaritalash darajasi mavjud. Kontseptual bosqichni ichki va tashqi bosqichga, kontseptual bosqichga tushirish. Xaritalar ma'lumotlar bazasi tuzilmasiga o'zgartirishlar kiritilganda ma'lumotlarning mustaqilligini saqlaydi.

2.3 Ma'lumotlar bazasini boshqarish tizimlari funktsiyalari

Ma'lumotlar bazasini boshqarish tizimlari funktsiyalari quyidagilardir:

1. Ma'lumotlarni aniqlash. MBBT ma'lumotlar bazasini aniqlashga imkon berishi kerak (tashqi sxemalar, kontseptual va ichki sxemalar, tegishli xaritalar). Buning uchun ma'lumotlar bazasi turli xil ma'lumotlarni aniqlash tillari uchun til protsessorini o'z ichiga oladi.
2. Ma'lumotlarni qayta ishlash. MBBT ma'lumotlar namunalarini olish, shuningdek o'zgartirishlarni kiritish bo'yicha foydalanuvchi so'rovlarini ko'rib

chiqishi kerak. Buning uchun ma'lumotlar bazasi, ma'lumotlarga ishlov berish tilining protsessor qismlarini o'z ichiga oladi.

3. Xavfsizlik va ma'lumotlar yaxlitligi. MBBTning so'rovlarni kuzatishi va xavfsizlik qoidalari va yaxlitligini buzish harakatlarining oldini olishi kerak.

4. Ma'lumotni tiklash va ko'paytirish. MBBT ishdan chiqqandan so'ng ma'lumotlarni tiklashni ta'minlashi kerak.

5. Ma'lumotlar lug'ati. MBBT ma'lumotlar lug'ati funktsiyasini taqdim etishi kerak. Lug'atning o'zi foydalanuvchi ma'lumotlar bazasi ma'lumotlarini o'z ichiga olgan tizim ma'lumotlar bazasi deb hisoblanishi mumkin, ya'ni boshqa tizim obyektlarining ta'riflarini o'z ichiga oladi. Lug'at u belgilaydigan ma'lumotlar bazasiga birlashtirilgan va shuning uchun o'zi haqida ma'lumot mavjud.

6. Ishlash. Ma'lumotlar bazasi o'z funktsiyalarini maksimal darajada bajarishi kerak.

Savollar:

1. MBBT arxitekturasining qaysi darajalarini bilasiz?
2. Uch asosiy daraja o'rtasida qanday oraliq darajalar mavjud?
3. MBBT qanday funktsiyalarga ega?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

III Bob. Ma'lumotlar bazasi modellari

3.1 Infologik modellashtirish haqida umumiy ma'lumot

Ma'lumotlar bazasi haqiqiy dunyoning bir qismini namoyish etadi. Tabiiyki, uning tavsifining to'liqligi yaratilayotgan axborot tizimining maqsadlariga bog'liq bo'ladi.

Ma'lumotlar bazasi sohasini etarlicha aks ettirishi uchun ma'lumotlar bazasi dizayneri ushbu sohasiga (dasturiy ta'minot) xos bo'lgan barcha muammolarni yaxshi bilishi va ularni ma'lumotlar bazasida aks ettira olishi kerak. MB maydoni oldindan tavsiflanishi kerak. Buning uchun, qoida tariqasida, tabiiy tildan foydalanish mumkin, ammo uning qo'llanilishida ko'plab kamchiliklar mavjud, ularning asosiylari noqulay tavsif va uni talqin qilishning noaniqligidadir. Shuning uchun odatda ushbu maqsadlar uchun sun'iy rasmiylashtirilgan til vositalaridan foydalaniladi. Shu munosabat bilan ***infologik model*** (ILM) kelajakda ishlatiladigan dasturlardan mustaqil bo'lgan maxsus til vositalari yordamida tuzilgan fan sohasining tavsifi deb tushuniladi.

Infologik model sizning ma'lumotlar tizimingizni amalga oshirish uchun har qanday MBBTdan foydalanishni davom ettirishingiz yoki boshqa dasturlardan

foydalanishingizdan qat'iy nazar qurilishi kerak.

Infologik modelga qo'yiladigan talablar.

Maqsaddan kelib chiqqan holda ILM uchun asosiy talab bu fan sohasini yetarlicha aks ettirish talabidir. ILM izchil bo'lishi kerak.

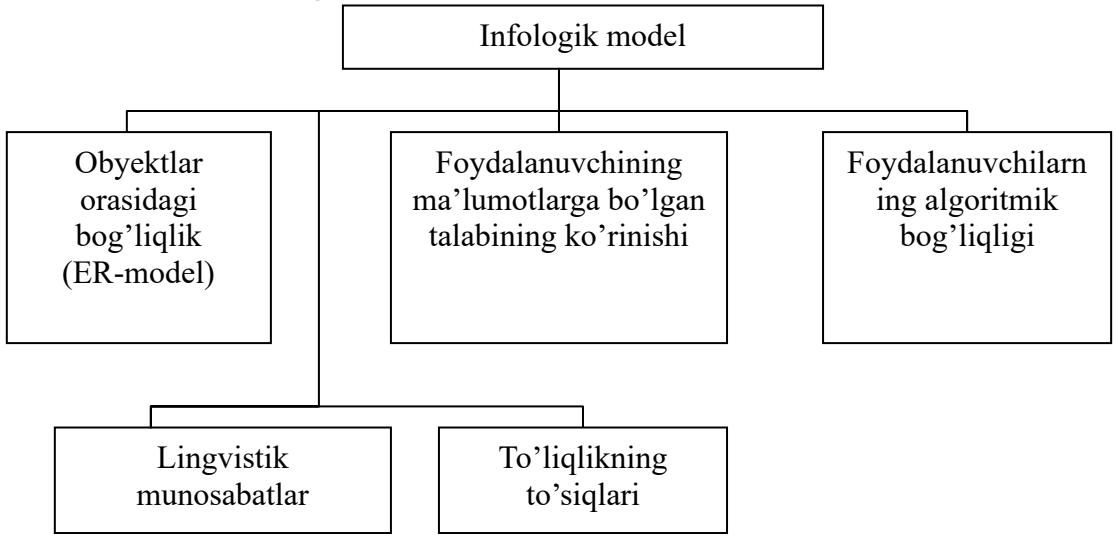
ILM-da aks ettirilgan haqiqiy dunyo tabiatan cheksiz bo'lishiga qaramay, infologik model cheklangan bo'lib, bu mavzu doirasini aniq cheklash bilan ta'minlanadi. Shu munosabat bilan, ILM oson kengaytiriladigan xususiyatga ega bo'lishi kerak, bu avval aniqlangan ma'lumotlarni o'zgartirmasdan yangi ma'lumotlarning kiritilishini ta'minlaydi. Xuddi shu narsani ma'lumotlarni yo'q qilish haqida ham aytish mumkin. Haqiqiy infologik modellarning katta o'lchamlari bilan bog'liq holda, modelning tarkibi va parchalanishi ehtimoli ta'minlanishi kerak.

Infologik modelni foydalanuvchilarning turli toifalari osonlikcha tushunishlari kerak. ILM -ni ma'lumotni qayta ishlashning avtomatlashtirilgan tizimlarining dizayneri emas, balki ushbu sohada ishlaydigan mutaxassis tomonidan qurilishi tavsiya etiladi yoki hech bo'lmaganda ushbu sohasining o'ziga xos xususiyatlari to'g'ri tushunilganligiga ishonch hosil qilish uchun tavsifni tekshiriladi. Infologik model, keyinchalik ma'lumotlar bazasi va dasturiy ta'minotni loyihalashda ishtirok etadigan barcha mutaxassislar tomonidan oson va aniq qabul qilinishi kerak.

Bu MB tizimining yadrosidir. ILM avtomatlashtirilgan axborotni qayta ishlash tizimini loyihalashtirish uchun zarur va etarli ma'lumotlarni o'z ichiga oladi.

3.2 Infologik modelning tarkibiy qismlari

MBBT sohasining infologik modeli qator tarkibiy qismlarni o'z ichiga oladi (3.1-rasm). Infologik modelning markaziy tarkibiy qismi bu domen ob'ektlarining tavsifi va ular o'rtasidagi munosabatlar (ER-model).



3.1-rasm. Infologik modelning tarkibiy qismlari

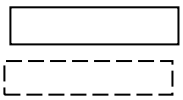
"OBYEKT - XUSUSIYAT - MUNOSABAT" modelini yaratish.

ILMni tavsiflash uchun ikkala analitik (tavsiflovchi) turdagi tillar qo'llaniladi va grafik vositalar keyinchalik «obyekt - atrebut - munosabatlar» modelini namoyish qilish uchun grafik usul sifatida ishlatiladi. Tekshirish va tahlil qilish jarayonida

ushbu sohada obyektlarning sinflari ajralib turadi.

Obyektlar sinfi - bu bir xil xususiyatlarga ega bo'lgan ob'ektlar to'plamidir. Masalan, agar biz universitetni fan sohasi deb hisoblasak, undagi obyektlarning quyidagi sinflarini ajratib ko'rsatishimiz mumkin: talabalar, o'qituvchilar, sinfxonalar va boshqalar. Obyektlar yuqorida aytib o'tilganidek real bo'lishi mumkin, masalan obyektlar ham mavhum bo'lishi mumkin.

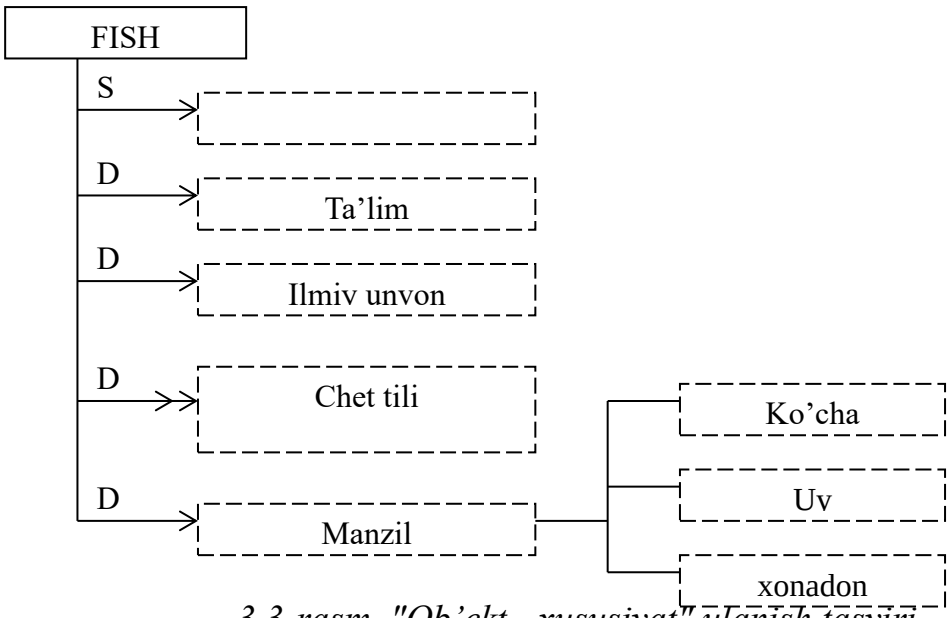
Axborot tizimida aks ettirishda har bir ob'ekt o'zining identifikatori bilan ifodalanadi, u bir sinf ob'ektini boshqasidan ajratib turadi va ob'ektlarning har bir sinifi ushbu sinf nomi bilan ifodalanadi. Misol uchun, "O'QIYDIGAN FANLAR" sinfidagi ob'ektlar uchun har bir ob'ektning identifikatori "FANLAR NOMI" bo'ladi. Aniqlovchi noyob bo'lishi kerak. Har bir ob'ekt ma'lum xususiyatlarga ega. Xuddi shu sinf ob'ektlari uchun ushbu xususiyatlarning to'plami bir xil va ularning qiymati, shubhasiz, farq qilishi mumkin. Masalan, "TALABA" sinf obyektlari uchun sinf obyektlarini tavsiflovchi bunday xususiyatlar to'plami "TUG'ILGAN YILI", "JINSI" va boshqalar bo'lishi mumkin. MBBT maydonining tavsifida mavjud sinflarning har birini va ushbu sinf ob'ektlari uchun belgilangan xususiyatlar to'plamini tasvirlash kerak. Obyektlar va ularning xususiyatlarini ko'rsatish uchun biz quyidagi yozuvdan foydalanamiz (3.2-rasm).



3.2-rasm. Ob'ektlarning belgilanishi va ularning xususiyatlari

Infologik modeldagi obyektlarning har bir sinfiga o'ziga xos nom berilgan.

Infologik modelni qurishda, ayniqsa, kontseptsiyani noaniq talqin qilish mumkin bo'lsa, har bir obyektni og'zaki talqin qilish maqsadga muvofiqdir.



3.3-rasm. "Ob'ekt - xususiyat" ulanish tasviri

MBBT sohasini tavsiflashda ob'yekt va uni tavsiflovchi xususiyatlar o'rtasidagi munosabatni aks ettirish kerak. Bu shunchaki ob'ektning belgisi va uning xususiyatlarini bog'laydigan chiziq sifatida tasvirlangan.

Ob'ekt va uning atrebuti o'rtasidagi munosabatlar har xil bo'lishi

mumkin. Ob'ekt faqat bitta qiymatga ega bo'lishi bo'ladi. Masalan, har bir inson faqat bitta tug'ilgan sanaga egadir. Bunday xususiyatlar ob'ekt birligi deyiladi. Boshqa xususiyatlar uchun bitta ob'ektning bir nechta qiymatlari bir vaqtning o'zida mavjud bo'lishi mumkin. Masalan, "Ishchilar" tavsifida u egalik qiladigan "CHET TILI" atrebut sifatida o'rnatilsin. Xodim bir nechta xorijiy tillarni bilishi mumkin bo'lganligi sababli, ushbu atrebut bir nechta deb nomlanadi. Ob'ekt va uning xususiyatlari o'rtasidagi munosabatni tasvirlashda biz bitta xususiyatlar uchun bitta o'qni va bir nechta xususiyatlar uchun ikki baravar o'qni ishlatamiz.

Bundan tashqari, ba'zi xususiyatlar doimiy bo'lib, vaqt o'tishi bilan ularning qiymati o'zgarishi mumkin emas. Biz bunday xususiyatlarni statik deymiz va vaqt o'tishi bilan qiymati o'zgarishi mumkin bo'lgan xususiyatlar dinamik deb nomlanadi.

Ob'ekt va uning atrebuti o'rtasidagi munosabatlarning yana bir o'ziga xos xususiyati bu xususiyat ushbu sinfning barcha ob'ektlarida mavjud yoki yo'qligi yoki ba'zi obyektlarda yo'qligining belgisidir. Masalan, ba'zi bir xodimlar uchun "AKADEMIK BOSQICH" atrebut bo'lishi mumkin va ushbu sinfning boshqa ob'ektlarida ko'rsatilgan atrebut bo'ladi. Biz bunday xususiyatlarni shartli ob'ekt deb ataymiz.

Infologik modelda ob'ektlarning individual holatlari emas, balki ob'ekt sinflari ko'rsatiladi. Ob'ektning belgilanishi ILM-da ko'rsatilganda, biz tavsiflangan xususiyatlarga ega ob'ektlar sinfi haqida gaplashayotganimiz aniq. Shuning uchun ko'p holatlarda ob'ektlar sinfi uchun belgini aniq infologik modelga kiritish mumkin emas. Ob'ektlar sinfining aniq tasviri faqat ob'ektlar sinfi uchun mo'ljallangan dastur nafaqat ushbu sinfning alohida ob'ektlari bilan bog'liq bo'lgan xususiyatlarni, balki butun sinfga tegishli bo'lgan yaxlit xususiyatlarni qayd etgan taqdirdagina zarurdir.

Misol uchun, agar "XIZMAT KO'MAKCHILARI" ob'ektlari sinfi uchun nafaqat har bir xodimning yoshi, balki barcha xodimlarning o'rtacha yoshi ham qayd etilgan bo'lsa, unda infologik modelda nafaqat "Xodimlar" ob'ekti, balki "Ishchilar" ob'ektlari sinfini aks ettirilishi kerak. Ob'ektlar sinfini ko'rsatish uchun siz ba'zi bir aniq belgilarni yoki ob'ektlar uchun ishlatiladigan bir xil narsani ishlatishingiz mumkin.

3.3 Semantik ma'lumotlarni modellashtirish, ER-diagrammalari

Relatsiya ma'lumotlar bazasini keng qo'llash va ulardan turli xil dasturlarda foydalanish relyatsion ma'lumotlar modeli fan sohalarini modellashtirish uchun etarli ekanligini ko'rsatadi. Biroq, biz qisqacha ko'rib chiqqan normallashtirish mexanizmiga asoslangan munosabatlar nuqtai nazaridan, relyatsion ma'lumotlar bazasini loyihalash ko'pincha foydalanuvchilar uchun juda murakkab va noqulay jarayondir.

Shu bilan birga, relatsiya ma'lumotlar modelining cheklanishi quyidagi jihatlarda namoyon bo'ladi:

- model ma'lumotlarning ma'nosini ifodalash uchun etarli vositalarni ta'minlamaydi. Haqiqiy bilim sohasining semantikasi modeldan mustaqil ravishda foydalanuvchiga bog'liq bo'lmagan tarzda taqdim etilishi

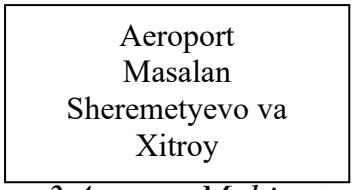
kerak. Xususan, bu biz aytib o'tgan yaxlitlikni cheklash muammosi haqidadir.

- ko'pgina dasturlar uchun tekis jadvallar asosida mavzu maydonini modellashtirish qiyin. Ba'zi hollarda, foydalanuvchining dastlabki boshlang'ich bosqichida, foydalanuvchi ushbu mavzuni bitta (ehtimol hatto normallashtirilmagan) jadval rasmida tasvirlash uchun o'ziga qarshi zo'ravonlik qilishga majbur bo'ladi.
- garchi foydalanuvchining butun jarayoni qaramlikni hisobga olishga asoslangan bo'lsa-da, relatsiya model ushbu qaramlikni namoyish etish uchun hech qanday vositani ta'minlamaydi.
- foydalanuvchi jarayoni amaliy domen ob'ekti ("mantiqiy ob’ektlar") uchun zarur bo'lgan ba'zi elementlarni tanlashdan va ushbu mantiqiy ob’ektlar o'rtasidagi munosabatlarni aniqlashdan boshlanishiga qaramasdan, relatsion ma'lumotlar modeli sub’ektlar va munosabatlarni ajratish uchun biron-bir apparatni taklif qilmaydi.

Ma'lumotlar bazasini loyihalashga turli xil zamonaviy yondashuvlar (asosan relyatsion) ER modeli turlaridan foydalanishga asoslangan. Model Chen tomonidan 1976 yilda taklif etilgan. Domenlarni modellashtirish kam sonli qismlarni o'z ichiga olgan grafik sxemalardan foydalanishga asoslangan. Ma'lumotlar bazasi sxemalarining aniq taqdimoti tufayli ER-modellari relyatsion ma'lumotlar bazalarining avtomatlashtirilgan dizaynini qo'llab-quvvatlovchi CASE tizimlarida keng qo'llaniladi. ER-modellarining ko'p navlari orasida eng rivojlanganlaridan biri ORACLE CASE tizimida qo'llaniladi. Biz buni ko'rib chiqamiz. Aniqrog'i, ushbu modelning tarkibiy qismiga e'tibor qaratamiz.

ER modelining asosiy tushunchalari mavjudlik, munosabatlar va xususiyatdir.

Korxona haqiqiy yoki tasavvur qilingan ob’ekt bo'lib, unda MB saqlanishi va foydalanishi kerak. Diagrammalar ER-model shaxsning nomini o'z ichiga olgan to'rtburchak tomonidan taqdim etiladi. Bunday holda, tashkilot nomi ushbu turdagi ba'zi bir misollar emas, balki turning nomi bo’ladi. Ko'proq ravshanlik va yaxshiroq tushunish uchun ob'ektning nomi ushbu turdagi aniq ob'ektlarning namunalari bilan birga bo'lishi mumkin. Quyida Sheremetyevo va Xitroy kabi namunaviy ob'ektlar bilan AEROPORTning mohiyati keltirilgan:



3.4-rasm. Mohiyat misoli

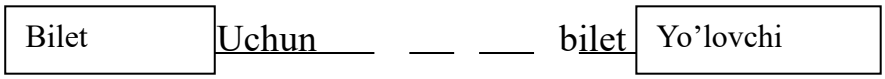
Korxonaning har bir nusxasi xuddi shu mantiqiy ob'ektning har qanday boshqa nusxasidan ajralib turishi kerak (bu talab o'zaro bog'liqlik jadvalida takroriy dubllarning mavjud emasligi talabiga o'xshashdir).

Aloqa **-** bu ikki shaxs o'rtasida o'rnatilgan grafik tasvirlangan birlashmadir. Ushbu assotsiatsiya har doim ikkilikdir va ikki xil ob'ekt o'rtasida yoki shaxs va o'zi o'rtasida mavjud bo'lishi mumkin bo’lgan holat hisoblanadi (rekursiv

munosabatlar). Har qanday aloqada, ikkita uchi bir-biridan farq qiladi (mavjud ulangan ob’ektlar juftligiga muvofiq), ularning har birida ulanishning nomi, ulanishning tugatish darajasi (ushbu ob'ektning nechta holati ulangan), ulanishning majburiyligi (ya'ni, ushbu ob'ektning har qanday misolida qatnashishi kerakmi) kerakligi tushintiriladi. Ulanish ikkita ob’ektni bir-biriga bog'laydigan yoki ob’ektdan o'zi unga olib boradigan chiziq rasmida taqdim etiladi. Shu bilan birga, mantiqiy ob’ekt bilan ulanish "joylashtirilgan" joyda, agar tashkilot uchun ko'p (many) misollardan munosabatda foydalanish mumkin bo'lsa, mantiqiy ob'ektning to'rtburchagiga uchta nuqtali kirish ishlatiladi, agar mantiqiy ob'ektning o'zaro munosabatlarida faqat bitta misol ishtirok etadigan bo'lsa. Ulanishning kerakli uchi qattiq chiziq bilan, ixtiyoriy esa chiziqli chiziq bilan ko'rsatilgan.

Biror korxona singari, munosabatlar ham odatiy tushunchadir, bularning har ikkala juftligi ham birlashma qoidalariga bo'ysunadi.

Quyidagi misolda CHIPTA va PASSAJIR subyektlari o'rtasidagi munosabatlar chiptalar va yo'lovchilarni bog'laydi. Bundan tashqari, "uchun" nomli yozuvning tugashi sizga bitta yo'lovchiga bir nechta chiptalarni bog'lashga imkon beradi va har bir chipta yo'lovchi bilan bog'liq bo'lishi kerak. Korxonaning "bor" degan yozuvi har bir chipta faqat bitta yo'lovchiga tegishli bo'lishi mumkinligini anglatadi va yo'lovchidan kamida bitta chipta bo'lishi shart emas.

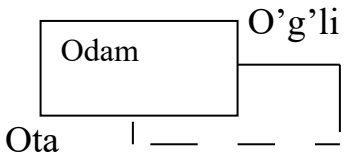


3.5-rasm. Aloqa misoli

Diagrammaning qisqa og'zaki talqini quyidagicha:

- har bir bilet bitta va bittadan yo’lovchi uchun mo'ljallangan;
- har bir yo’lovchi bir yoki bir nechta biletlarga ega bo'lishi mumkin.

Quyidagi misolda insonning mohiyatini o'zi bilan bog'laydigan rekursiv ulanish tasvirlangan. "O'g'il" nomi bilan bo'lgan munosabatlarning tugashi, bitta otada bir nechta o'g'il bo'lishi mumkinligi bilan belgilanadi. "Ota" nomi bilan munosabatlarning tugashi shuni anglatadiki, har bir odamning o'g'illari bo'lishi mumkin emas.



3.6-rasm. Rekursiv aloqa

Diagrammaning qisqa og'zaki talqini quyidagicha:

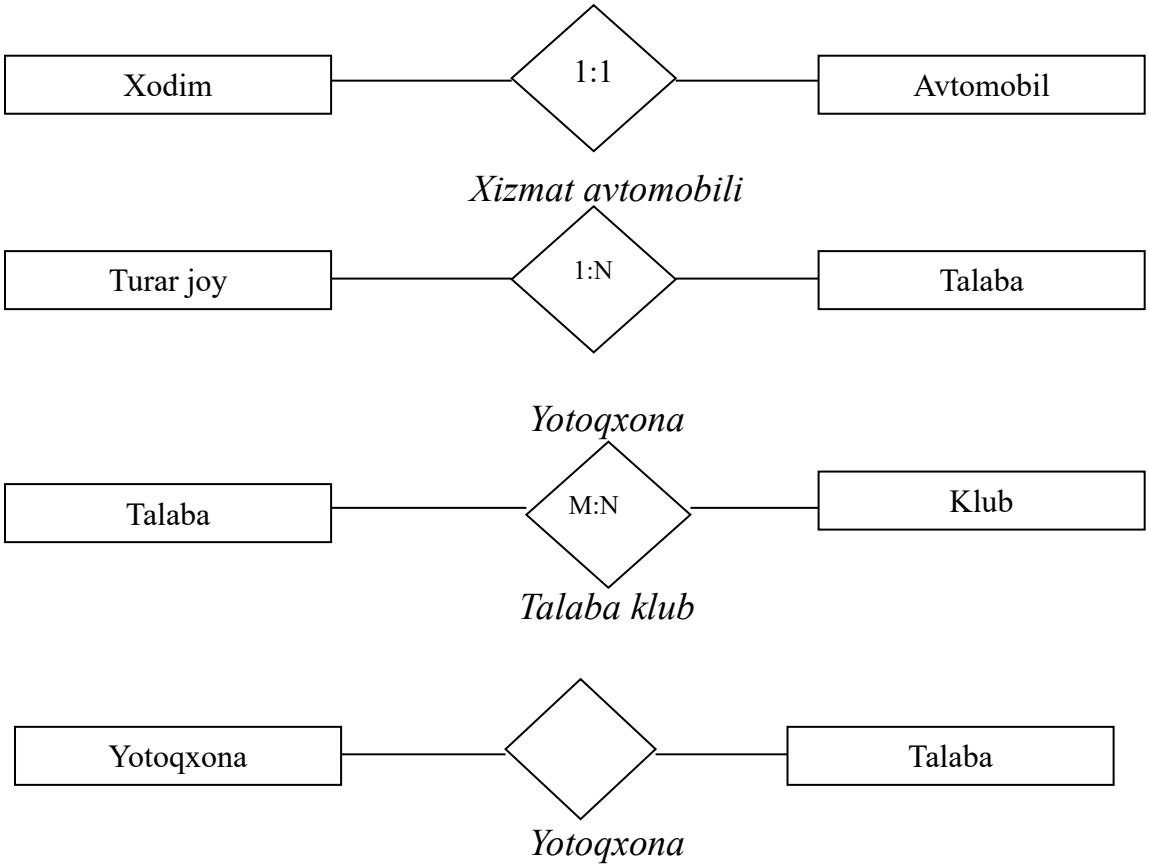
- har bir inson bitta va bitta insonning o'g'li;
- har bir erkak bir yoki bir nechta odamning otasi bo'lishi mumkin.

Mohiyatning atributi - bu uning holatini aniqlashtirish, aniqlash, tasniflash,

raqamli tavsiflash yoki ifodalash uchun xizmat qiladigan har qanday tafsilotdir. Atributlar nomlari ob’ektni ko'rsatadigan to'rtburchaklar rasmida, ob’ekt nomi ostida va kichik harflar bilan ko'rsatilgan bo'lishi mumkin.

Ikkilik aloqalarning uch turi

3.7- rasmda da ikkilik munosabatlarning uch turi ko'rsatilgan. 1:1 munosabatlarda ("bittadan bittaga") bitta turdagi bitta mantiqiy ob’ekt boshqa turning bitta mantiqiy ob’ekti bilan bog'langan. 3.7-rasmda va XIZMAT AVTOMOBILI munosabatlari, xodimlar sinfining bitta subyektini AVTOMOBIL sinfining bitta sub'ekti bilan bog'laydi. Ushbu diagrammaga binoan, bir nechta transport vositasi har qanday xodimga tayinlanmagan va bitta transport vositasi bittadan ortiq xodimga tayinlanmagan.



3.7-rasm. Ikkilik aloqalarning uch turi.

3.7- rasmda, ulanishning ikkinchi turi tasvirlangan, 1:N ("bittadan Ngacha" yoki "bittadan ko'pgacha"). Shu munosabat bilan, Yotoqxona deb ataladigan Turar joy sinfining bir namunasi Talabaning sinfining ko'plab misollari bilan bog'liq. Ushbu ko'rsatkichga ko'ra, ko'p talabalar yotoqxonada yashaydilar, ammo har bir talaba faqat bitta yotoqxonada yashaydi.

1 va N pozitsiyalarining ahamiyati. Jihoz yotoqxona joylashgan aloqa tomonida, N esa Talaba joylashgan aloqa tomonida joylashgan. Agar 1 va N lar aksincha joylashgan bo'lsa va ulanish N: l deb yozilsa, bitta talaba yotoqxonada yashashi va har bir talaba bir nechta talabalar turar joylarida yashashi aniqlanadi. Bu, albatta, unday emas. 3.7-rasmda N: M ikkilik ulanishning uchinchi turi ko'rsatilgan

("N dan M" yoki "ko'pga ko'p"). Ushbu munosabatlar Talaba, Klub deb nomlanadi va u Talaba sinf sub'ektlari misollarini Klub sinf sub'ektlari misollari bilan bog'laydi. Bitta talaba bir nechta to'garaklarga a'zo bo'lishi mumkin va bitta klubda ko'plab talabalar bo'lishi mumkin.

Romb ichidagi raqamlar, ulanishni ramziy qilib, ulanishning har ikki tomonidagi maksimal sonlarni bildiradi. Ushbu cheklovlar maksimal kardinal raqamlar deb nomlanadi va ulanishning ikkala tomoni uchun bunday ikkita cheklovning kombinatsiyasi ulanishning maksimal kardinalligi deb ataladi. Masalan, 3.3-rasmda a, b, ular maksimal 1 kardinallikka ega deb aytadilar. N kardinal raqamlar faqat 1 va N ni emas, balki boshqa qiymatlarga ham ega bo'lishi mumkin. Masalan, BASKETBALL TEAM va O'YINCHI sub'ektlari o'rtasidagi munosabatlar 1: 5 nisbatda kardiganlikka ega bo'lishi mumkin, bu esa basketbol jamoasida beshdan ortiq o'yinchi bo'lmasligi mumkinligini anglatadi.

Ba'zan "HAS" munosabatlari yoki egalik munosabatlari (HAS-A munosabatlari) deb nomlanadi. Bunday atama bitta sub'ekt boshqa sub'ekt bilan bog'langanligi uchun ishlatiladi. Masalan: xodimda mashina, talabada yotoqxona, to'garakda talabalar bor.

ER modelining yanada murakkab elementlari

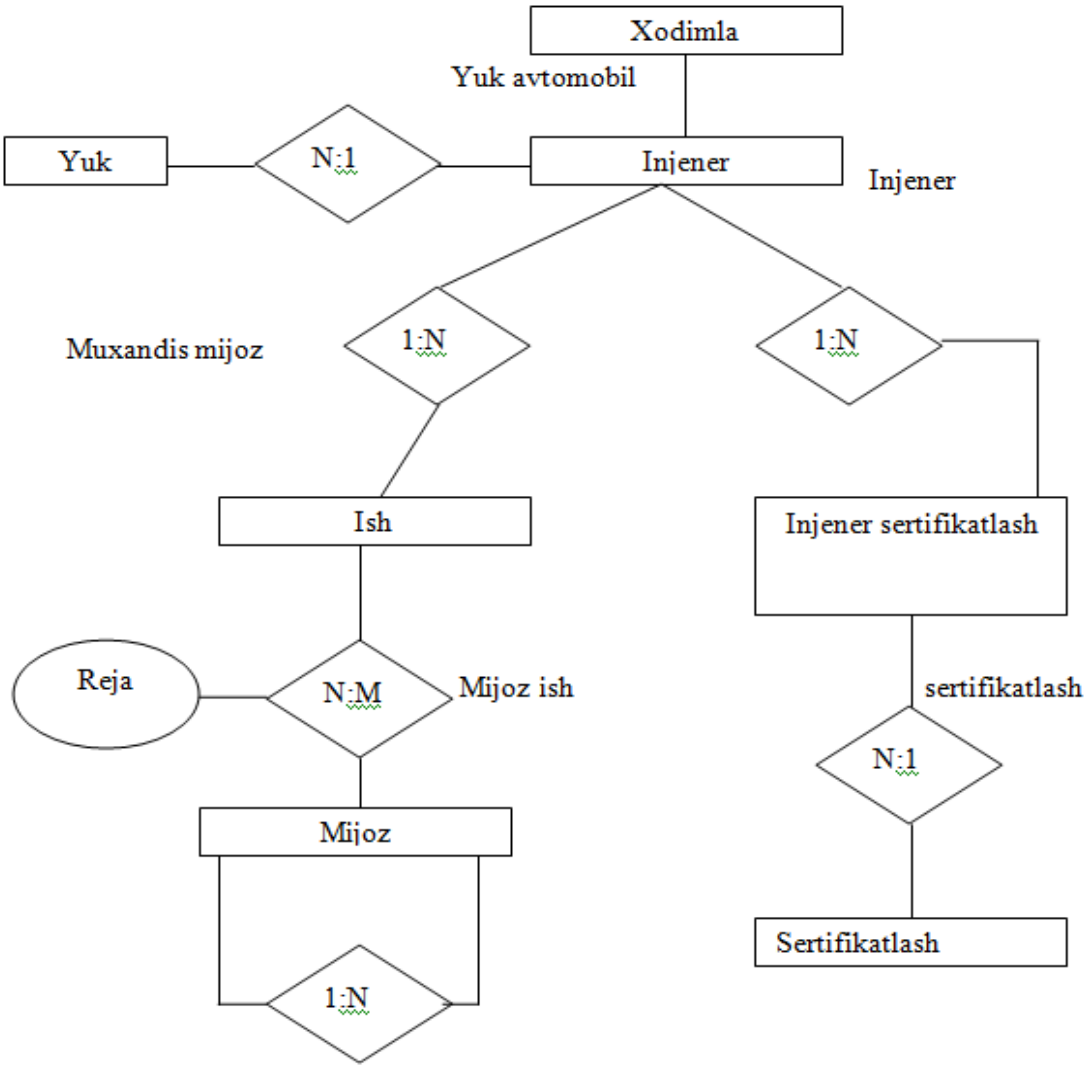
Biz faqat ER ma'lumotlar modelining eng asosiy va eng aniq tushunchalariga e'tibor qaratdik. Modelning yanada murakkab elementlari quyidagilarni o'z ichiga oladi:

- subtyp va subtypes subyektlari. Rivojlangan standart tizimlarga ega dasturlash tillarida bo'lgani kabi (masalan, ob'ektga yo'naltirilgan dasturlash tillarida), bir yoki bir nechta superteplar asosida ob'ekt turini meros qilib olish imkoniyati joriy etiladi. Qiziqarli ma'lumorlar ushbu mexanizmni grafik tasvirlash zarurati bilan bog'liq.
- muloqot "many – to – many". Ba'zan korxonalarni ulanishning ikkala uchida mavjud bo'lishi mumkin bo'lgan tarzda bog'lash kerak bo'ladi (masalan, kooperativning barcha a'zolari birgalikda kooperativning mulkiga egalik qilishadi). Bunday aloqa "ko'p - ga - ko'p" deyiladi.
- aloqaning o'ziga xos darajalari, ba'zida ma'lum bir munosabatda bo'lgan sub'ektning mumkin bo'lgan sonini aniqlash foydalidir (masalan, xodimga bir vaqtning o'zida uchtadan ortiq loyihada qatnashishga ruxsat beriladi). Ushbu semantik cheklovni ifodalash uchun ulanish oxirida uning maksimal yoki majburiy darajasini ko'rsatishga ruxsat beriladi.
- korxona nusxalarini kaskadli o'chirish ba'zi ishoratlar (bir ga bir, "bir ga - ko'p"), (aloqa yakunida "bir" to'g'ri keladi) yozuvlar shaxs misoli yo'q qilish muloqot oxirida mos shaxsning barcha nusxalarini yo'q qilish uchun, bu "ko'p". Korxonani belgilashda "kaskadli o'chirish" ga tegishli talabni rasmlantirish mumkin.
- domenlar ma'lumotlarning relatsiya modeli kabi, mantiqiy obyekt (domen) atributining mumkin bo'lgan maqbul qiymatlari to'plamini aniqlash foydali bo'lishi mumkin.

Korxona-munosabatlar diagrammalari

Sxemalar 3.8 - rasmda ko'rsatilga. Mohiyat – aloqani har bir diagrammasi "shaxs-munosabatlar", yoki ER (diagram shaxs - munosabatlar diyagramlari) hisoblanadi.

Bunday jadvallar standartlashtirilgan, ammo unchalik qiyin emas. Ushbu standartga muvofiq, ob’ekt sinflari to'rtburchaklar bilan, munosabatlar romblar bilan, har bir munosabatlarning maksimal kardinal soni esa romb ichida ko'rsatilgan. Ob’ektning nomi to'rtburchaklar ichida, munosabatlar nomi esa romb yonida ko'rsatiladi.



3.8-rasm. “Mohiyat - aloqa” diagrammasiga na’muna

Savollar:

1. Korxonada ma'lumorlarni birlashririshini tasvirlab bering?
2. Aloqa nima?
3. Nima uchun modellashtirishdan foydalaniladi?
4. Ikkilik bog’lanishlairining nechta turini bilasiz?
5. Murakkab munosabatlarning qanday murakkab elementlarini bilasiz?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

IV Bob. Relatsion ma’lumot modeli

4.1 Relatsion modelning tarixi

Ushbu modelning nazariy asosini ikki mantiq - Amerikalik Charlz Soders Pirs (1839-1914) va nemis Ernst Shreder (1841-1902) asoslagan munosabatlar nazariyasi asos bo'ldi. Aloqalar nazariyasi bo'yicha qo'llanmalarda ba'zi bir maxsus operatsiyalarga nisbatan ko'plab munosabatlar yopiq ekanligi, ya'ni bu operatsiyalar bilan birgalikda mavhum algebrani tashkil qilishi ko'rsatilgan. Aloqalarning ushbu muhim xususiyati relyatsion modelda asl algebra bilan bog'liq bo'lgan ma'lumotlar manipulyatsiyasi tilini ishlab chiqish uchun ishlatilgan.

Ma'lumoti bo'yicha matematik bo'lgan E. Codd ma'lumotlarga ishlov berish uchun o'rnatilgan nazariya apparatlaridan (ma'lumotlar birligi, kesishish, farq, kartezian mahsuloti) foydalanishni taklif qildi. U har qanday ma'lumotlarning ifodasi matematikada ma'lum bo'lgan ikki xil o'lchovli jadvallar to'plamiga qisqarishini ko'rsatdi munosabatlar - relation.

E.Coddning takliflari ma'lumotlar bazasi tizimlari uchun shunchalik samarali ediki, ushbu model uchun u kompyuter texnologiyalarining nazariy asoslari sohasidagi nufuzli Turing mukofotiga sazovor bo'ldi.

Hozirgi kunda ma'lumotlar bazasini tanqid qilishning asosiy mavzusi ularning samaradorligi yo'qligi emas, balki quyidagi kamchiliklardir:

1. ushbu tizimlarga xos bo'lgan ba'zi bir cheklashlar (soddaligi to'g'ridan-to'g'ri natijasi) an'anaviy bo'lmagan deb ataladigan sohalarda ishlatilganda (eng oddiy misollar ma’lumotni avtomatlashtirish tizimlari) bo'lib, ular juda murakkab ma'lumotlarning tuzilishini talab qiladi.
2. MB sohasining semantikasini etarlicha aks ettirishning mumkin emasligi. Boshqacha qilib aytganda, relyatsion tizimlarda mavzu doirasining semantik o'ziga xosligi to'g'risida bilimlarni taqdim etish imkoniyatlari juda cheklangan. Post-relatsion tizimlar sohasidagi zamonaviy tadqiqotlar asosan ushbu kamchiliklarni bartaraf etishga bag'ishlangan.

Relyatsion modelda ma'lumotlarning eng kichik birligi alohida hisoblanadi ushbu model uchun yadro (yechib bo'lmaydigan) ma'lumotlar qiymati hisoblanadi. Bitta MB sohasida familiya, ism va otaning ismini bitta ma'no, boshqasida esa uch xil ma'no sifatida ko'rib chiqish mumkin.

Domen - bir xil tipdagi qiymatlari to'plamidir. Domen kontseptsiyasining eng to'g'ri intuitiv talqini bu domenni ushbu turdagi qiymatlarning maqbul potentsial

to'plami sifatida tushunishdir. Masalan, bizning misolimizdagi "Ismlar" domeni belgilar satrlarining asosiy turida aniqlangan, ammo uning qiymatlari faqat nomni ifodalaydigan satrlarni o'z ichiga olishi mumkin (xususan, bunday satrlar yumshoq belgi bilan boshlanmasligi mumkin).

Domen tushunchasining semantik yuklanishini ham ta'kidlash kerak:

ma'lumotlar taqqoslanadigan deb hisoblanadi, agar ular bitta domenga tegishli bo'lsa. Bizning misolimizda "O'tkazib yuborish raqamlari" va "Guruh raqamlari" domenlari qiymatlari butun sonlar turiga kiradi, ammo taqqoslanmaydi. Shuni yodda tutingki, ko'pgina MBBT-larida domen tushunchasi ishlatilmaydi, ammo u allaqachon Oracle V.7-da qo'llab-quvvatlanadi.

Aloqaga domen kiritilishi odatda ***atribut*** deb ataladi .

Aloqalar sxemasi - bu atribut nomi, domen nomi (turi yoki domen tushunchasi qo'llab-quvvatlanmaydigan bo'lsa) nomli juftliklar to'plamidir.

Nizomning bu dastur munosabatlariga mos - har bir ism, egalik, diagramma munosabatlarni olish uchun, MBsiga ichiga kirib juft {nomi, xususiyati qiymati}, majmui hisoblanadi. "Value" bu atributning haqiqiy domen qiymati (yoki domen tushunchasi qo'llab-quvvatlanmasa ma'lumotlar turi). Shunday qilib, juftlikning darajasi yoki "aritmi", ya'ni ulardagi elementlarning soni tegishli aloqalar sxemasining "aritmi" ga to'g'ri keladi. Oddiy qilib aytganda, tup - bu ma'lum bir turdagi nomlangan qiymatlar to'plami.

Kalit - bu satrni aniq belgilaydigan bir yoki bir nechta atributlar guruhidir. Talabalar soni , bo'lim va to'lov haqi atributlariga ega bo'lgan bo'lim (4.3-rasm) munosabatini ko'rib chiqilgan. Ushbu munosabatlar liniyasida talaba ma'lum bir haq evaziga ma'lum bir qismga tashrif buyurishi haqida ma'lumot mavjud.

Aloqa - bu yagona munosabatlar sxemasiga mos keladigan bog'lanishlar to'plamidir. Ba'zida chalkashib ketmaslik uchun ular "munosabatlar sxemasi" va "munosabatlar-misol" deyishadi, ba'zida munosabatlar sxemasi munosabatlarning sarlavhasi deb ataladi, va ulanishlar to'plami kabi munosabatlar "munosabatlar tanasi" deb ataladi. Aslida, munosabatlar sxemasi tushunchasi dasturlash tillarida tarkibiy ma'lumotlar turi kontseptsiyasiga eng yaqin. Aloqa sxemasini alohida aniqlashga, keyin esa ushbu sxema bilan bir yoki bir nechta munosabatlarga ruxsat berish mantiqan to'g'ri keladi.

Munosabatlarning odatiy kundalik namoyishi bu ikki o'lchovli jadval hisoblanadi. Jadvalning har bir satrida biron bir narsaga yoki uning bir qismiga tegishli ma'lumotlar mavjud. Jadvalning har bir ustunida bu narsaning atributi tasvirlangan. Ba'zan qatorlar to'rlar, ustunlar esa atributlar deb ataladi.

Yuqoridagi va boshqa bir qator matematik tushunchalar relyatsion ma'lumotlar bazalarini yaratish, ularning yuqori ishlashini ta'minlaydigan tegishli til vositalari va dasturiy ta'minot tizimini ishlab chiqish va ma'lumotlar bazasini loyihalash nazariyasining asoslarini yaratish uchun nazariy asos bo'lgan. Ammo, umumiy aloqador MBBT foydalanuvchisi uchun ushbu tushunchalarning norasmiy ekvivalentlaridan muvaffaqiyatli foydalanish mumkin:

- aloqa - jadval (ba'zan Fayl);

- tuple - string (ba'zan yozish);
- atribut - ustun, maydon.

Relyatsion ma'lumotlar bazasi – bu ***ma'lumotlar bazasida*** saqlanishi kerak bo'lgan barcha ma'lumotlarni o'z ichiga olgan munosabatlar to'plami. Biroq, foydalanuvchilar bunday ma'lumotlar bazasini jadvallar to'plami sifatida qabul qilishlari mumkin.

Relyatsion modelning afzalliklari:

- ushbu ma'lumotlar modeli foydalanuvchi uchun ma'lumotni eng sodda rasmda namoyish etadi.
- u ishlab chiqilgan matematik apparatga asoslanadi, bu ma'lumotlar bo'yicha asosiy operatsiyalarni aniq qisqacha tavsiflashga imkon beradi.
- sizga protsessual bo'lmagan manipulyatsiya tillarini yaratishga imkon beradi.
- chiqarilgan ma'lumotlar bazasi darajasida ma'lumotlarni boshqarish va o'zgartirish imkoniyatidir.

Relyatsion modelning kamchiliklari:

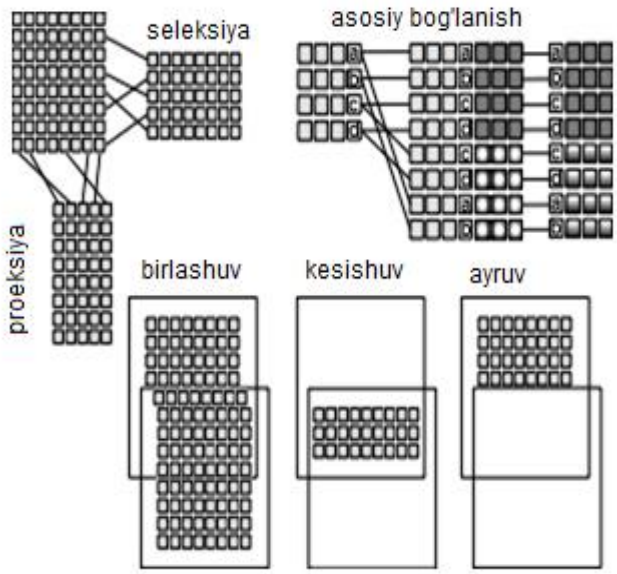
- ma'lumotlarga eng sekin kirish.
- rivojlanishning murakkabligi.

Jadval aloqador bo'lishi uchun u ma'lum cheklovlarga javob berishi kerak:

1. Birinchidan, jadval xujayralaridagi qiymatlar bitta bo'lishi kerak - na takroriy guruhlar va na massivlarga ruxsat berilmaydi.
2. Qatorlarda belgilangan maydonlar (ustunlar) va qiymatlar mavjud (bir nechta maydonlar va takrorlanadigan guruhlarga yo'l qo'yilmaydi). Boshqacha qilib aytganda, jadvalning har bir pozitsiyasida satr va ustunning kesishish joyida doimo bitta qiymat yoki hech narsa bo'lmaydi.
3. Jadval satrlari, albatta, bir-biridan kamida bitta qiymat bilan farq qiladi, bu sizga bunday jadvalning har qanday satrini noyob aniqlash imkonini beradi.
4. Ustundagi barcha yozuvlar bir xil bo'lishi kerak, masalan, agar jadvalning birinchi qatoridagi uchinchi ustunda xodimning raqami bo'lsa, jadvalning qolgan barcha satrlarida uchinchi ustunda xodimlarning raqamlari ham bo'lishi kerak. Har bir ustun o'ziga xos nomga ega; jadvaldagi ustunlarning tartibi ahamiyatli emas. Va nihoyat, aloqada ikkita bir xil chiziq bo'lishi mumkin emas va chiziqlar tartibi muhim emas.
5. Ma'lumotlar bazasining to'liq ma'lumot mazmuni aniq ma'lumotlar qiymatlari ko'rinishida taqdim etiladi va ushbu usul yagona hisoblanadi.
6. Jadval bilan operatsiyalarni amalga oshirishda, ularning tarkibiy qismlaridan qat'i nazar, uning qatorlari va ustunlari istalgan tartibda qayta ishlanishi mumkin. Bunga jadvallarning nomlari va ularning ustunlarining mavjudligi, shuningdek, ularning har qanday satrini yoki ko'rsatilgan belgilar bilan har qanday qatorlarni ajratib ko'rsatish imkoniyati yordam beradi (masalan, "Parij" yo'nalishi bo'yicha parvozlar va kelish vaqti 12 soatgacha).

Relyatsion ma'lumotlar modelini taklif qilib, Kodd shuningdek, munosabatlar bilan ishlash uchun qulay vositani - relyatsion algebrani yaratdi. Ushbu algebraning har bir operatsiyasi bir yoki bir nechta jadvallarni (aloqalarni) o'z operandalari

sifatida ishlatadi va natijada yangi jadval paydo bo'ladi, ya'ni stolni "kesish" yoki "yopishtirish" imkonini beradi (4.1-rasm).



4.1- rasm. Relyatsion algebraning ba'zi operatsiyalari.

4.1-jadvalda munosabatlarga misol keltiradi. Aloqada ettita qator mavjud, ularning har birida to'rtta ustun mavjud. Agar biz ustunlarni boshqa tartibda joylashtirgan bo'lsak (masalan, Kadrlar sonini eng chap ustunga joylashtirish orqali) yoki satrlarni (masalan, Yosh ustunining qiymatini oshirish orqali) qayta tartiblasak, biz teng nisbatni olamiz.

4.1- jadval. Xodimlar bilan munosabatlarga misol.

	Atribut 1 Ism	Atribut 2 Yosh	Atribut 3 Jins	Atribut 4 Table raqami
Kortej 1	Anderson	21	A	010110
Kortej 2	Jekker	22	E	010100
……	Glover	22	E	101000
……	Jekson	21	A	201100
……	Mur	19	E	111100
……	Nakata	20	A	111101
Kortej 7	Smit	19	E	111111

Aloqa xususiyatlari.

Relyatsion munosabatlar quyidagi xususiyatlarga ega.

- Aloqada bir-biriga o'xshash bog'ichlar yo'q.
- Qarindoshlik rishtalari buyurtma qilinmaydi (yuqoridan pastgacha).
- O'zaro munosabatlarning sifatlari tartiblashtirilmaydi (chapdan o'nga).
- Barcha atribut qiymatlari atomdir (skalalar, ajralmas).

4.2 Ma’lumotlar bazasida munosabatlar

Bu xususiyat munosabatlar tanasi matematik to'plam (tuples) ekanligidan kelib chiqadi. Klassik to'plam nazariyasida, ta'rifga ko'ra, to'plam bir xil elementlarni o'z ichiga olmaydi.

Ushbu xususiyat yuqorida qayd etilgan munosabatlar va jadval tushunchalarining bir-biriga mos kelmasligiga misoldir. Jadval (umumiy holatda) bir xil satrlarni o'z ichiga olishi mumkin va shuning uchun bir xil satrlarni o'z ichiga olgan jadval ta'rif bo'yicha o'zaro bog'liqlik bo'la olmaydi.

Aloqada bir-biriga o'xshash bir nechta satrlar mavjud emasligining muhim natijasi shundaki, munosabatlarda har doim kamida bitta potentsial kalit mavjud. Darhaqiqat, bog'lamlar noyob bo'lganligi sababli, atributlarning barchasi yoki qisman kombinatsiyasi, albatta, o'ziga xoslik xususiyatiga ega bo'ladi va shuning uchun dublларni o'ziga xos ravishda ajratib turadigan munosabatlar kaliti bo'lib xizmat qilishi mumkin.

Qarindoshlik rishtalari buyurtma qilinmaydi (yuqoridan pastgacha).

Bu xususiyat shuningdek munosabatlar tanasi matematik to'plam ekanligi va matematikada oddiy to'plamlar buyurtma qilinmaganligidan kelib chiqadi. 4.1-rasmda ko'rsatilgan munosabatda, seleksiya, proeksiya ayruv, birlashuvlar va shunga qaramay u xuddi shunday munosabatlar keltirilgan.

Yuqorida aytilganlardan kelib chiqqan holda, birinchi juftlik, oxirgi tup, o'ninchi tup, keyingi yoki oldingi juftlik va boshqalar kabi tushunchalar munosabatlarga taalluqli emas, boshqacha qilib aytganda, munosabatlarda bog'lanish joylarining mavqega oid manzili yo'q.

Muayyan guruhga murojaat qilish, uni aniqlash faqat asosiy aloqalar orqali amalga oshirilishi mumkin.

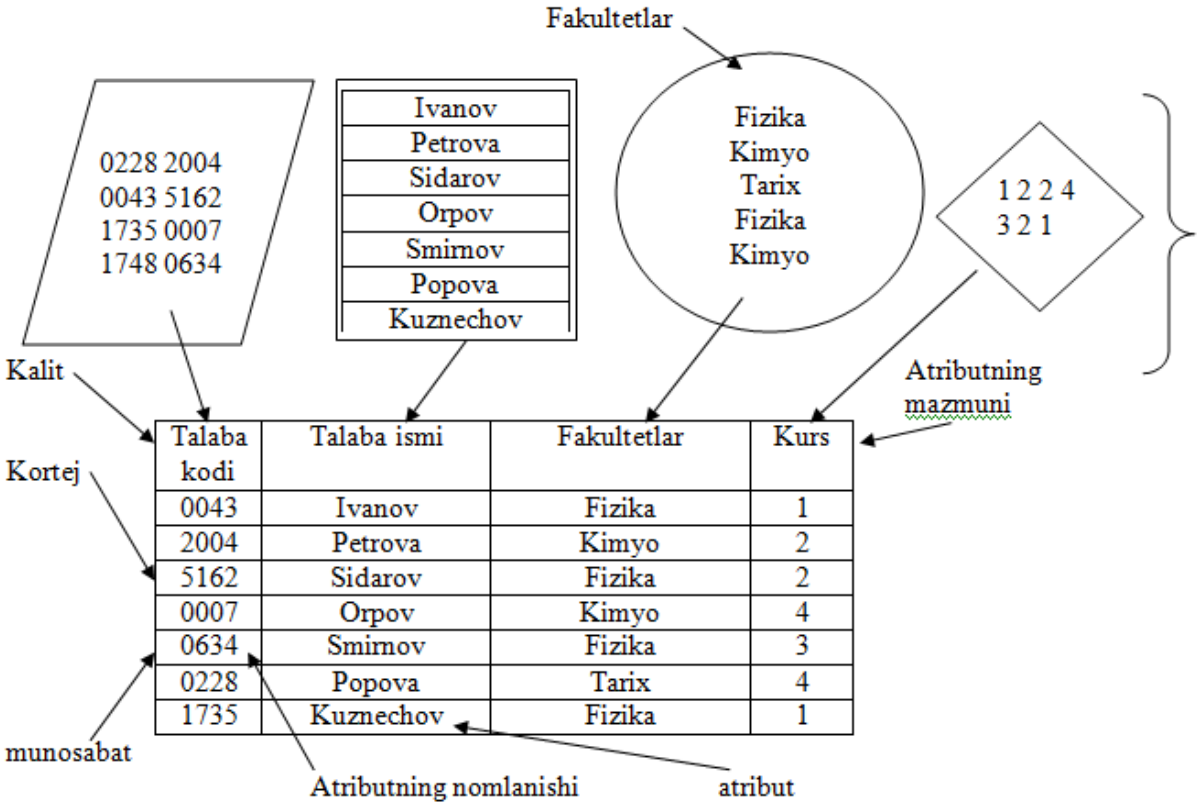
Ushbu xususiyat shuningdek munosabatlar va jadval tushunchalarining ekvivalent emasligini ham ko'rsatadi. O'zaro munosabatlardan farqli o'laroq, jadvaldagi satrlar har doim buyurtma qilinadi - birinchi qator, ikkinchi, oxirgi va boshqalar mavjud.

O'zaro munosabatlarning sifatlari tartiblashtirilmaydi (chapdan o'ngga).

Ushbu xususiyat munosabatlar sarlavhasi oddiy matematik to'plam sifatida ham aniqlanganligidan kelib chiqadi, ya'ni: juftliklar to'plami <name - attribute: domain-name>.

Bunga asoslanib, agar jadvalning 4.3-bandida bir xil atributlar boshqa tartibda joylashtirilishi kerak edi, u hali ham bir xil munosabatda bo'ladi. Shuning uchun birinchi atribut, oxirgi atribut, keyingi yoki oldingi atribut kabi tushunchalar mavjud emas.

Atribut har doim munosabati bilan emas, balki uning nomi bilan belgilanadi.



4.3-rasm. Munosabatlar va uning tarkibiy qismlari

Ushbu paragrafga muvofiq munosabatlar va jadval tushunchalari ham bir-biriga mos kelmaydi. Oddiy jadvaldagi ustunlar, munosabatlar xususiyatlaridan farqli o'laroq, har doim chapdan o'ngga buyurtma qilinadi.

Barcha atributlarning qiymatlari atamardir.

Relyatsion modelda o'zaro bog'liqlik atributlari aniqlanadigan va haqiqiy atribut qiymatlari "koplangan" domenlar faqat yardro (bo'linmas, skalyar) qiymatlarni o'z ichiga olishi mumkin. Boshqacha qilib aytganda, ustunni va jadvalni o'zaro aloqani ifodalaydigan chiziqning kesishmasida qiymatlar to'plami yoki har qanday murakkab (aralash) tuzilish emas, balki bitta qiymat bo'lishi kerak.

Ushbu shartni qondiradigan munosabatlar normallashtirilgan deb nomlanadi yoki birinchi normal rasmda taqdim etiladi (boshqa normal rasmlar keyingi bo'limda keyingi boblarda muhokama qilinadi).

Yuqorida aytilganlardan kelib chiqqan holda, relyatsion model faqat normallashtirilgan munosabatlarni ko'rib chiqadi, garchi matematik ma'noda tushunilgan munosabatlarni, umuman olganda, normallashtirish mumkin emas. Normallashtirilmagan munosabatlar atributlarining qiymatlari sifatida, murakkabroq tuzilmalar, masalan, boshqa munosabatlardan foydalanish mumkin.

G'ayritabiiy munosabatlarning misoli 4.4- rasmda keltirilgan. Yuqoridagilardan kelib chiqqan holda, relatsion modelda bunday munosabatlarga yo'l qo'yib bo'lmaydi. Biroq, shuni ta'kidlash kerakki, ushbu cheklash zarur ma'lumotlarni aks ettirish imkoniyati nuqtai nazaridan bizni hech qanday tarzda cheklamaydi. Shunday qilib, normallashmagan munosabatda taqdim

etilgan barcha ma'lumotlar, 1-munosabat, normallashtirilgan munosabatlar, 2- munosabatlar ko'rinishida to'liq taqdim etilishi mumkin.

Kod	O’zlashtirish	
	Fan	Baho
C 9	Fizika	5
	Matematika	4
	Tarix	4
	Kimyo	3
	Informatika	5
C 6	Fizika	3
	Matematika	4
	Informatika	3
C 1	Matematika	5
C 7	Informatika	5
	Kimyo	4

Kod	fan	Baho
C 9	Fizika	5
C 9	Matematika	4
C 9	Tarix	4
C 9	Kimyo	3
C 9	Informatika	5
C 6	Fizika	3
C 6	Matematika	4
C 6	Informatika	3
C 1	Matematika	5
C 7	Informatika	5
C 7	Kimyo	4

4.4-rasm. Normal va nonormal munosabatlarga misol

Matematik nuqtai nazardan, faqat normallashtirilgan munosabatlarni qo'llash talabi (ularning sodda tuzilishi tufayli) aloqalarni ishlash uchun operatorlarni soddalashtirish va ularning sonini mavzu sohasidagi axborot tarkibini etarlicha aks ettirish imkoniyatini cheklamasdan kamaytirishga imkon beradi, chunki har qanday normallashtirilmagan munosabatlar uchun ekvivalent normallashtirilgan rasm mavjud.

Relyatsion algebraning barcha operatsiyalarini va ularning deyarli har qanday kombinatsiyasini amalga oshirishga imkon beradigan ma'lumotlar manipulyatsiyasi tillari yaratilgan. Ular orasida eng keng tarqalganlari SQL (Strukturalangan so'rovlar tili -tuzilgan so'rovlar tili) va QBE (Quere - By-Example - namunaviy so'rovlar). Ularning ikkalasi juda yuqori darajadagi tillarga taalluqlidir, ularning yordamida foydalanuvchi ularni olish tartibini ko'rsatmasdan qanday ma'lumotlarni olish kerakligini ko'rsatadi.

Ushbu tillarning istalganida bitta so'rovdan foydalanib, siz bir nechta jadvallarni vaqtinchalik jadvalga qo'shishingiz va undan kerakli qatorlar va ustunlarni kesib tashlashingiz mumkin (tanlash va proektsiyalash).

4.3 ER sxemasidan relyatsion sxemani olish

1-Qadam. Har bir oddiy ob’ekt jadvalga aylanadi. Oddiy ob’ekt - bu pastki tip bo'lmagan va subtyplarga ega bo'lmagan sub'ekt. Korxonaning nomi jadvalning nomiga aylanadi.

2-Qadam. Har bir atribut bir xil nomdagi mumkin bo'lgan ustunga aylanadi, aniqroq format tanlanishi mumkin. Ixtiyoriy atributlarga mos keladigan ustunlar aniqlanmagan qiymatlarni o'z ichiga olishi mumkin, talab qilingan atributlarga mos keladigan ustunlar ishlamaydi.

3-Qadam. Noyob identifikator komponentlari jadvalning asosiy kalitiga aylanadi. Agar bir nechta noyob identifikatorlar mavjud bo'lsa, eng ko'p

ishlatiladigan tanlanadi. Agar noyob identifikator havolalarni o'z ichiga olsa, havolaning eng oxirida joylashgan ob'ektning noyob identifikatori nusxasi birlamchi kalit ustunlari soniga qo'shiladi (bu jarayon rekursiv ravishda davom etishi mumkin). Ushbu ustunlarni nomlash uchun bog'lanishning uchlari va / yoki ob'ektlarning nomlari ishlatiladi.

4- Qadam. Bir nechta ulanish (va bittadan) tashqi kalitlarga aylanadi, noyob identifikatorning nusxasi "bitta" munosabatlarning oxiridan tayyorlanadi va tegishli kalitlar tashqi kalitni tashkil qiladi. Ixtiyoriy aloqalar aniqlanmagan qiymatlarga imkon beradigan ustunlarga mos keladi; aniqlanmagan qiymatlarga yo'l qo'ymaydigan ustunlar bilan bog'lanish aloqalari.

5- Qadam. Indekslar birlamchi kalit (noyob indeks), tashqi kalitlar va so'rovlarni asoslashi kerak bo'lgan atributlar uchun yaratiladi.

6- Qadam. Agar pastki chiziqlar kontseptual sxemada mavjud bo'lsa, ikkita usul mumkin.

- bitta kichik jadvallarning barchasi (a)
- har bir kichik tip uchun - alohida jadval (b)

(A) usulni qo'llashda eng tashqi supertiplar uchun jadval yaratiladi va pastki tiplar uchun ko'rinishlar yaratilishi mumkin. Jadvalga kamida TYPE kodini o'z ichiga olgan bitta ustun qo'shiladi, u birlamchi kalitning bir qismiga aylanadi.

Birinchi darajadagi har bir kichik tip uchun (b) usuldan foydalanganda (pastki darajalar uchun, ko'rinish uchun) supertip UNION ko'rinishi yordamida qayta tiklanadi (pastki jadvallarning barcha jadvallaridan umumiy ustunlar tanlanadi - ustun ustunlar).

Hammasi bitta jadvalda	**Jadval - kichik turdagi uchun**
Foyda	
Hammasi birgalikda saqlanadi; Toifalar va pastki turlarga oson kirish; Kamroq jadval kerak.	Pastki qismlarning qoidalari aniqroq, dasturlar faqat kerakli jadvallar bilan ishlaydi
Kamchiliklari	
Juda umumiy yechim. Turli xil ustunlar to'plamlari va turli xil cheklovlar bilan ishlash uchun qo'shimcha mantiq; Ehtimoliy muammolar (qulf tufayli); Subtip ustunlar ixtiyoriy bo‘lishi kerak; Ba'zi ma'lumotlar bazalari aniqlanmagan qiymatlarni saqlash uchun qo'shimcha xotirani talab qiladi.	Jadvallar juda ko‘p; UNION ko'rinishidagi chalkash ustunlar; UNION orqali ishlash paytida potentsial yo'qotish; Supertype-ni o'zgartirish mumkin emas.

7-qadam. Eksklyuziv ulanishlar bilan ishlashning ikki yo'li mavjud:

- umumiy domen (a)
- aniq tashqi kalitlar (b)

Agar qolgan xorijiy kalitlarning barchasi bitta domenda bo'lsa, ya'ni ular umumiy formatga (usul (a)) ega bo'lganligi sababli, ikkita ustun yaratiladi: ulanish identifikatori va ob'ekt identifikatori. Aloqa identifikatori ustuni istisno yoyi bilan qoplangan havolalarni ajratish uchun ishlatiladi. Muallif identifikatori ustuni noyob ob'ekt identifikatori qiymatlarini tegishli assotsiatsiyaning eng oxirida saqlash uchun ishlatiladi.

Agar paydo bo'lgan tashqi kalitlar bitta domenga tegishli bo'lmasa, istisno yoyi bilan qoplangan har bir ulanish uchun tashqi kalitlarning aniq ustunlari yaratiladi; ushbu ustunlarning barchasida aniqlanmagan qiymatlar bo'lishi mumkin.

Savollar:

1. Relatsion model nima?
2. Domen nima?
3. Qanday bog'liqlikni bilasiz?
4. Aloqa jadvaldan qanday farq qiladi?
5. Relyatsion modelning afzalliklari va kamchiliklari qanday?
6. Qanday qilib mantiqiy munosabat modeli relyatsion modelga tarjima qilinadi?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

V Bob. Relatsion algebra va relyatsion hisoblash elementlari

5.1 Relatsiya algebraga umumiy tushuncalari

Relyatsion modelning uchinchi qismi, manipulyatsiya qismi, relyatsion ma'lumotlarga kirish relyatsion algebradan yoki unga teng keladigan relyatsion hisoblash yordamida amalga oshirilishini da'vo qilmoqda.

Muayyan relyatsion MBBT-larni amalga oshirishda hozirda na relatsion algebra, na relatsion hisob ham uning sof rasmida qo'llanilmaydi. Relyatsion ma'lumotlarga kirishning haqiqiy standarti SQL (Strukturalangan so'rovlar tili). SQL tili - bu relyatsion algebra va relyatsion kalkulyatsiyada mavjud bo'lmagan qo'shimcha funktsiyalar bilan kengaytiriladigan, ingliz tilidagi iboralarga o'xshash sintaksisdan foydalanib, relyatsion algebra operatorlari va relyatsion hisoblarning ifodalari. Umuman olganda, ma'lumotlarga kirish tili relyatsion algebraga nisbatan ekspressiv kuch (yoki teng ravishda, relyatsion hisoblar) nuqtai nazaridan kam bo'lmasa, mutanosib deb nomlanadi. Bu til yordamida relyatsion algebraning har

qanday operatorini ifodalash mumkin. SQL aynan shunday tildir.

Relyatsion algebraning yopilishi

Relatsiya algebra - bu munosabatlarni argument sifatida ishlatadigan va natijada aloqalarni qaytaradigan operatorlar to'plami. Shunday qilib, relyatsion operator argumentlar sifatida munosabatlar bilan funktsiyaga o'xshaydi:

$$R = f(R_1, R_2, \ldots . . R_n)$$

Relatsiya algebra yopiq, chunki relyatsion operatorlarga argument sifatida siz turiga mos keladigan boshqa relyatsion operatorlarni almashtirishingiz mumkin:

$$R = f(f_1(R_{11}, R_{12}, \ldots . f_2.(R_{21}, R_{22\ldots}) \ldots .)$$

Shunday qilib, o'zaro bog'liq iboralarda siz o'zboshimchalik bilan murakkab tuzilishning ichki qurilgan ifodalarini ishlatishingiz mumkin.

Har bir munosabatlar ma'lumotlar bazasida noyob nomga ega bo'lishi kerak. Relyatsion operatsiyani bajarish natijasida yuzaga keladigan munosabatlarning nomi tenglikning chap tomonida aniqlanadi. Ammo, agar aloqalar boshqa aloqador iboralardagi dalillar sifatida o'zgartirilsa, siz munosabatlardan nomlarning mavjudligini talab qila olmaysiz. Bunday munosabatlar noma'lum munosabatlar deb ataladi . Nomlanmagan aloqalar ma'lumotlar bazasida haqiqatan ham mavjud emas, lekin faqat relatsion operatorning qiymatini hisoblash paytida hisoblab chiqiladi.

Relyatsion algebraning teoretik jarayonlarii.

Edgar Codd tomonidan aniqlangan rasmdagi o'zaro algebraga sakkizta operator kiradi, ular to'rtta operatorning ikkita guruhini tashkil qiladi.

1. To'plamlarda an'anaviy operatsiyalar: birlashma, kesishish, farq va ayirma;
2. karteziya mahsuloti (ularning barchasi o'zlarining operandalari o'zaro bog'liqligini hisobga olgan holda o'zgartiriladi, o'zboshimchalik bilan emas).
3. Maxsus relyatsion operatsiyalar: tanlash, proektsiyalash, ulanish va bo'linish.

Birlik - bu ikkala munosabatlarning biriga yoki ikkalasiga tegishli bo'lgan barcha to'rlarni o'z ichiga olgan aloqani qaytaradi

Kesishish - bir vaqtning o'zida ikkita munosabatlarga tegishli bo'lgan barcha dumlarni o'z ichiga olgan aloqani qaytaradi

Farq (minus) - berilgan ikki munosabatlarning birinchisiga tegishli va ikkinchisiga tegishli bo'lmagan barcha to'dalarni o'z ichiga olgan aloqani qaytaradi.

Mahsulot (vaqt) - barcha berilgan munosabatlarga tegishli bo'lgan ikkita tutqichning birikmasidan iborat bo'lgan barcha mumkin bo'lgan tutamlarni o'z ichiga olgan aloqani qaytaradi.

Selektsiya - belgilangan shartlarni qondiradigan ma'lum bir aloqadan barcha dumlarni o'z ichiga olgan aloqani qaytaradi. Xomiylik operatsiya ba'zan cheklash operatsiyasi deb ham ataladi, shuning uchun cheklash atamasi keyinchalik ushbu kitobda, agar ushbu algebraik operatsiya nazarda tutilgan bo'lsa, ishlatiladi.

Projeksiyon - ba'zi bir atributlar undan olib tashlanganidan so'ng, ushbu

munosabatda qolgan barcha munosabatlarni (subu tuplarni) o'z ichiga olgan aloqani qaytaradi.

Ulanish - bu ikkita munosabatlarga tegishli bo'lgan ikkita tutqichning atributlari yig'indisi bo'lgan barcha mumkin bo'lgan tutamlarni o'z ichiga olgan aloqani qaytaradi, agar bu ikkita biriktiriladigan tuynuklar asl munosabatlarga xos bo'lgan bitta yoki bir nechta atributlarda bir xil qiymatga ega bo'lsa (bundan tashqari, bu umumiy qiymatlar) paydo bo'lgan qismda ikki marta emas, balki bir marta paydo bo'ladi.

Mahsulot - berilgan ikkita munosabatlarga tegishlicha ikkita tutqichning birikmasidan iborat bo'lgan barcha mumkin bo'lgan tutqichlarni o'z ichiga olgan aloqani qaytaradi.

Relyatsion operatsiyalarning xususiyatlari.

A, B va C o'zboshimchalik bilan bog'liqlik ifodalari bo'lsin. Keyin birlashtirish uchun:

- • **(A UNION B) UNION C ≡ A UNION (B UNION C)** - (assotsiativ xususiyat)
- • **A UNION B ≡ B UNION A** - (kommutativ mulk).

Xuddi shunday, qolgan operatsiyalar uchun assotsiativlik va komutativlik xususiyatlari aniqlanadi.

5.2 Relatsion hisoblash elementlari

Qarindoshlik hisob-kitobi - bu birinchi darajali predikatli kalkulyatsiya (funktsiya ko'rinishidagi bayon) asosida amalga oshiriladigan deklarativ nazariy so'rovlar tili bo'lib, ular bog'lanishlar yoki munosabatlar sohalari qanoatlantirishi kerak.

Relatsiya hisoblash yordamida tuzilgan ma'lumotlar bazasiga so'rov kerakli natijaning tavsifini o'z ichiga oladi, buning uchun uni hisoblashning bir necha yo'li bo'lishi mumkin, ular relyatsion algebraning ifodalari bilan yoki to'g'ridan-to'g'ri MBBT buyruqlari bilan taqdim etiladi. Relyatsion hisoblashning relyatsion algebradan ustunligi, foydalanuvchi so'rovlarni bajarish algoritmini o'zi tuzishi shart emas deb hisoblashi mumkin, MBBT dasturi (etarli ma'lumotga ega) samarali algoritmni o'zi yaratadi.

Hisoblashning ikkita varianti mavjud: hisob-kitoblar, to'rlar va hisob-kitoblar, domenlar. Birinchi holda, o'zaro bog'liqliklarni tavsiflash uchun foydalaniladi, ularning haqiqiy qiymatlari o'zaro bog'liqlik, ikkinchi holda domen elementlari.

MBBT asoslangan relyatsion hisob-kitoblar

Bog'lovlarni hisoblashda, shuningdek protsessual dasturlash tillarida, avval foydalanilgan o'zgaruvchilarni tavsiflashingiz kerak va keyin ma'lumotlarga so'rov ifodalarini yozishingiz kerak.

Hisoblashning tavsif qismini quyidagicha ko'rsatish mumkin:

```
RANGE OF <O'zgaruvchan> IS <ro'yxat>
```

RANGE konstruktsiyasi o'zgaruvchan (o'zgaruvchan) identifikatorni va uning haqiqiy qiymatlari oralig'ini ko'rsatadi - ro'yxat - bir yoki bir nechta elementlarning

ketma-ketligi: *x1, ..., xn*, ularning har biri aloqalar yoki munosabatlar ustidagi ifodadir (iboralarni yozish tartibi quyida keltirilgan). Bundan tashqari, har qanday vaqtda, o'zgaruvchi munosabatlar ro'yxatidan faqat bittasini qiymat sifatida qabul qiladi.

Ro'yxat munosabatlarining sxemalari teng bo'lishi kerak. O'zgaruvchining ruxsat etilgan qiymatlari maydoni ro'yxatning barcha elementlarining qiymatlarini birlashtirish orqali hosil bo'ladi.

O'zgaruvchilardan foydalanish:

```
RANGE OF Var1 IS Table1, Table2
```

Var1 o'zgaruvchisining doirasi munosabatlarning barcha qiymatlarini o'z ichiga oladi, bu Table1 va Table2-munosabatlarning birlashmasi.

MBBT relyatsion hisobi ifodasi bu <maqsadli ro'yxat> WHERE <WFF> rasmining tuzilishidir.

Ifodaning qiymati - tanasi (ko'pgina tuynuklar) WFFni (yaxshi rasmlangan formulani) qondirishi kerak bo'lgan munosabat, va sxema (atributlar to'plami va ularning nomlari) maqsadlar ro'yxati bilan belgilanadi. Maqsadli ro'yxat proektsion operatsiyani aniqlaydi va WFF formulasi bog'lovchilarni tanlashni aniqlaydi.

<variable> .tratibute> juftligida birinchi tarkibiy qism o'zgaruvchini (RANGE konstruktsiyasi bilan aniqlanadi) ko'rsatish uchun ishlatiladi, ikkinchisida esa o'zgaruvchi o'zgaradigan o'zaro bog'liqlik atributini aniqlash uchun foydalaniladi. "AS <tribute>" ixtiyoriy qismi maqsadli munosabatlarning atributini qayta nomlash uchun ishlatiladi. Agar u mavjud bo'lmasa, maqsadli munosabatlarning atributi nomi manba munosabatlarining tegishli atribut nomidan meros qilib olinadi.

O'zgaruvchan nomni maqsadli aloqaning elementi sifatida ishlatish tegishli aloqaning barcha atributlarini ro'yxatiga qo'shishga tengdir.

Yaxshi qurilgan formuladan (*Well Formed Formula, WFF*) ko'p o'zgaruvchiga qo'yilgan shartlarni ifodalash uchun foydalaniladi. WFF ning asosini oddiy taqqoslashlar tashkil etadi, ular skalyar qiymatlarni taqqoslash operatsiyalari (atribut o'zgaruvchisi yoki doimiy qiymatlari).

Oddiy shartlar bu skalyar qiymatlarni taqqoslash operatsiyalari. Ulardan ba'zilari:

```
O'zgaruvchan ism. Atribut nomi = Svalalar
qiymati
O'zgaruvchan ism A. Attribut nomi B = o'zgaruvchan ism
B. Atribut nomi G
VariableName.Atribut nomi <> ScalarValue
O'zgaruvchan ism A. Attribut nomi B <o'zgaruvchan ism
B. Atribut nomi G
```

WFFning yanada murakkab variantlari NOT, AND, OR va IF ... THEN. mantiqiy biriktirgichlardan foydalangan holda tuzilgan. Shunday qilib, agar <formula> WFF bo'lsa va <taqqoslash> oddiy taqqoslash bo'lsa, quyida keltirilgan iboralar WFF:

```
NOT <formula>
<taqqoslash> AND <rasm>
<taqqoslash> OR <formula>
IF <taqqoslash> THEN <formula>.
```

Kuchaytirgich yordamida WFFni qurishga ruxsat beriladi. Agar <formula> bu WFF bo'lsa, unda o'zgaruvchan> qatnashadi, unda yuqoridagi inshootlar ham WFF hisoblanadi.

```
EXISTS <o'zgaruvchan> (<formula>)
FORALL <o'zgaruvchan> (<rasm>.
```

Birinchi holda, WFF bu degani: "<formula> hisoblashda TRUE qiymatini berish uchun kamida <o'zgaruvchan> qiymat mavjud.»

Ikkinchi holda, WFF degani: "o'zgaruvchining barcha qiymatlari uchun <formula> hisoblashda TRUE qiymati berilgan."

WFF tarkibidagi o'zgaruvchilar bepul yoki bog'langan bo'lishi mumkin. WFF tarkibiga kirgan barcha o'zgaruvchilar, qaysi kalkulyator ishlatilmaganda, bepul hisoblanadi. Aslida, bu agar erkin eruvchan o'zgaruvchilarning ba'zi qiymatlari to'plami uchun WFFni hisoblashda TRUE qiymati olingan bo'lsa, demak, bu o'zgaruvchi o'zgaruvchilarning bu qiymatlari natijaviy aloqaga qo'shilishi mumkin.

Agar o'zgaruvchan nom `WIFT` rasmlanishida `EXISTS <variable> <formula>)` yoki `FORALL <variable>` `(<formula>)` rasmini yaratishda kvalifikatordan keyin darhol `foydalanilsa` , u holda ushbu WFFda va uning ishtirokida qurilgan barcha WFFlarda <variable> bo'ladi. chegaralangan o'zgaruvchi. Bu shuni anglatadiki, bunday o'zgaruvchi ushbu o'zgaruvchini bog'lagan minimal WFFdan tashqarida ko'rinmaydi. Bunday WFF qiymatini hisoblashda bog'liq bo'lgan o'zgaruvchining bitta qiymati emas, balki uning butun aniqlanish doirasi qo'llaniladi.

Hisoblagichlardan foydalanishni ko'rib chiqing. Var1 va Var2 jadvalga bog'liqlik bo'yicha aniqlangan ikkita ikkita o'zgaruvchi bo'lsin. Keyin, WFF:

`EXISTS Var2 (Var1.Table1_field1> Var2. Table1 _field1)` haqiqiydir, agar va har jihatdan 1-jadvalda (Var2 bilan bog'liq) atribut qiymati uning atributi qiymati ichki holatni qondirsa. taqqoslash.

`FORALL Var2 (Var1.Table1_field1> Var2. Table1 _ field1)` uchun mavjud bo'lgan `Var1 (Var1.Table1_field1> Var2.Table1_field1).`

Agar mavjud bo'lsa, 1-jadvalning o'zaro bog'liqligi (Var2 o'zgaruvchisi bilan bog'liq) Table1_field1 atributining qiymatlari taqqoslash shartini qondirsa.

Aslida, erkin va bog'liq bo'lgan o'zgaruvchilar haqida emas, balki o'zgaruvchilarning erkin va bog'langan holatlari haqida gapirish to'g'riroqdir. Agar o'zgaruvchi o'zgaruvchi WFF rasmida bog'langan bo'lsa, unda WFFning barchasida shu o'zgaruvchini o'z ichiga olgan holda erkin yoki bog'lanishi mumkin bo'lgan o'zgaruvchining nomi ishlatilishi mumkin, ammo har qanday holatda ham o'zgaruvchining WFF rasmida bo'lishi bilan hech qanday aloqasi yo'q.

O'zgaruvchilarning ikkita o'zaro bog'liqligini ishlatishga misol keltirilgan.

```
EXISTS Var2 (Var1.Table1_field2 = Var2. Table1_
field2) AND FORALL Var2 (Var1.Table1_field1 >
Var2.Table1_field1)
```

Bu erda biz Var2-ning ikkita mutlaqo boshqacha ma'nosi bor.

Ta'riflangan hisob-kitob hisob-kitoblarga imkon bermagani uchun hisoblash to'liq emas. Hisoblash funktsiyalariga hisoblash funktsiyalarini qo'shish taqqoslash operandlari va maqsadli ro'yxat elementlarining aniqlanishini kengaytirish orqali amalga oshirilishi mumkin, shunda ular skalyar ifodalarni adabiyotlar, atributlar havolalari va xulosa funktsiyalari bilan ishlatishga imkon beradi.

Quyidagi funktsiyalar jami sifatida harakat qilishi mumkin: COUNT (sanash), SUMM (miqdor), AVG (o'rtacha), MAX (maksimal), MIN (minimal).

MBBT relyatsion hisobi QUEL so'rovlari tili uchun asosdir.

5.3 Relatsiya domenini hisoblash

Domenni hisoblashda o'zgaruvchilar doirasi munosabatlar emas, balki domenlardir. Xodimlar ma'lumotlar bazasiga murojaat qilinganda, masalan, domen o'zgaruvchilari nomi (qiymatlari - haqiqiy nomi) yoki bo'lim raqami (qiymatlar - korxona bo'limlarining haqiqiy soni) haqida gapirish mumkin .

Domenni hisoblash va ko'p darajali hisoblash o'rtasidagi asosiy rasmiy farq bu a'zolik shartlarini ifoda etishga imkon beradigan qo'shimcha predikatlar to'plamining mavjudligidadir.

Qolgan barcha holatlarda domen hisoblash formulalari va iboralari ko'p kalkulyatsiya formulalari va iboralariga o'xshaydi. Xususan, albatta, domen o'zgaruvchilarining erkin va bog'liq bo'lgan holatlari ajralib turadi.

Relyatsion domen hisob-kitobi rasmlardan foydalanishga asoslangan so'rovlarning aksariyat tillarining asosidir. Xususan, bu hisob asoslangan til Query ma'lum tomonidan jadval rasmida asoslangan til oilasi birinchi (va eng qiziqarli) tili edi taqlid.

Savollar:

1. Relyatsion algebra nima?
2. Edgar Codd relyatsion algebra operatorlarini qaysi turlariga kiritdi?
3. Relyatsion operatorlar qanday xususiyatlarga ega?
4. Relyatsion hisob nima?
5. Relyatsion hisoblarning qaysi ikki turini bilasiz?
6. To'g'ri tuzilgan formula nima?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

VI Bob. Ma'lumotlar bazasini rejalashtirish, loyihalash va administratorlash

6.1 Axborot tizimining hayot sikli modeli

Har qanday ma'lumotda "umr ko'rish" mavjud. U qisqa vaqt davomida (hisob-kitob paytida xotirada), bir muncha vaqt (ba'zi ma'lumotlarni tayyorlash paytida) yoki juda uzoq vaqt (muhim shaxsiy, tijorat, ommaviy yoki davlat ma'lumotlarini saqlashda) mavjud bo'lishi mumkin. Ushbu vaqt oralig'i *ma'lumotlarning hayot aylanishini* belgilaydi.

AT-ning hayot aylanishi - bu ma'lumotlarning, axborot mahsulotlari va xizmatlari, shuningdek, apparat vositalarining hayotiy siklining hosilasidir.

Inson faoliyatining turli sohalarida axborot tizimlari *hayot siklining bosqichlari* asosan bir xil:

1. muammoning bayoni;
2. xizmat dizayni;
3. rivojlantirish va joylashtirish;
4. kafolatlangan xizmatlar;
5. xizmatni modernizatsiya qilish yoki tugatish.

Kompyuter dasturlarini yaratish va undan foydalanishning hayotiy sikllari ushbu dasturiy ta'minotga bo'lgan ehtiyoj paydo bo'lgan vaqtdan to barcha foydalanuvchilar foydalana olmagan paytgacha ularning turli xil holatini aks ettiradi.

An'anaga ko'ra, dasturiy ta'minot hayot siklining quyidagi asosiy bosqichlari ajralib turadi:

1. talablarni tahlil qilish;
2. dizayn;
3. kodlash (dasturlash);
4. sinov va disk raskadrovka;
5. foydalanish va texnik xizmat ko'rsatish.

Ishlab chiquvchilar axborot mahsulotlari va xizmatlarining hayot aylanishini iloji boricha iloji boricha amalga oshirishga intiladilar. Zamonaviy kompyuter dasturlarining aksariyati uchun o'n yoki undan ortiq yil davomida mavjud bo'lgan dasturlar mavjud bo'lsa-da, hayot siklining muddati ikki yildan uch yilgacha. Uchun bu davri oshirish doimiy ularni qo'llab-quvvatlash uchun marketing va boshqa faoliyatni amalga oshirish uchun zarur hisoblanadi.

Dasturiy mahsulot sotuvdan chiqarilgandan keyin ma'lum vaqt o'tgach, u qo'llab-quvvatlanishi mumkin. Dasturiy mahsulotni ishlab chiqarish va texnik xizmat ko'rsatishni davom ettirishdan yoki axborot xizmatlarini taqdim etishdan bosh tortish odatda ularning samarasizligi, xatolarning mavjudligi va talabning yo'qligi bilan izohlanadi.

ATning hayot aylanishi uni yaratish va undan foydalanish modelidir. Model axborot tizimining turli xil holatlarini aks ettiradi, ushbu tizim zarurati paydo bo'lgan paytdan boshlab, barcha foydalanuvchilar uchun to'liq to'xtatilgan paytgacha.

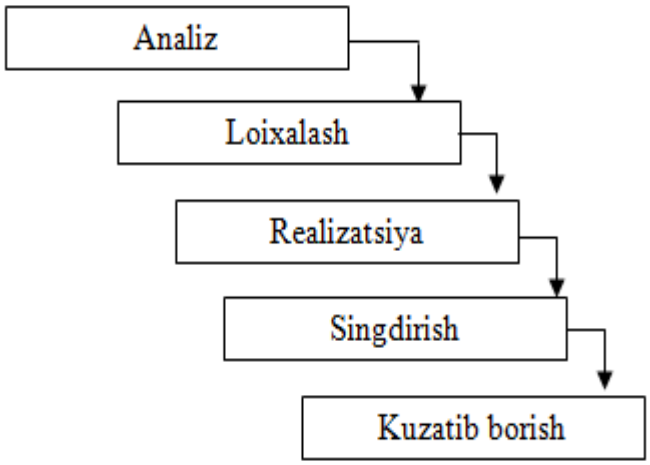
Hayot siklining modeli deganda hayotiy sikl davomida amalga oshiriladigan jarayonlar, harakatlar va vazifalarning ketma-ketligi va o'zaro bog'liqligini

aniqlaydigan tuzilma tushuniladi.

HC modeli axborot tizimining o'ziga xos xususiyatlariga, shuningdek, uni yaratish va ishlash sharoitlarining o'ziga xos xususiyatlariga bog'liq.

Axborot texnologiyalarining hayotiy siklining uchta modeli eng ko'p ishlatiladi: kaskadli, fazali va spiral.

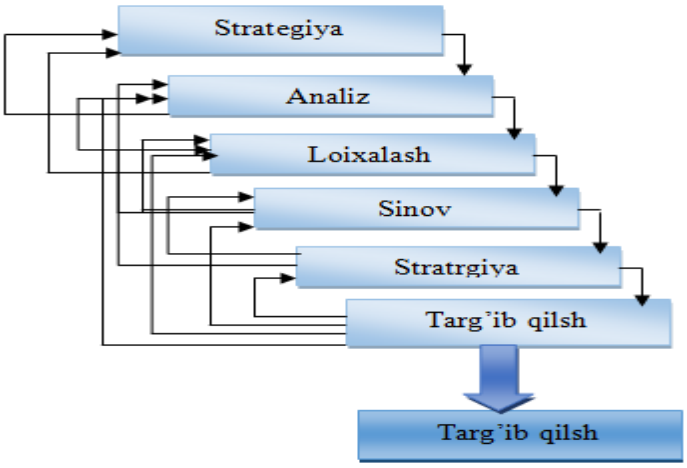
Kaskad modeli yoki "palapartishlik" oldingi bosqichda ish tugagandan so'ng keyingi bosqichga o'tishga yo'naltirilgan texnologiyalarda qo'llaniladi (6.1-rasm).



6.1- rasm. Dasturiy ta'minotni kaskadli rivojlantirish sxemasi.

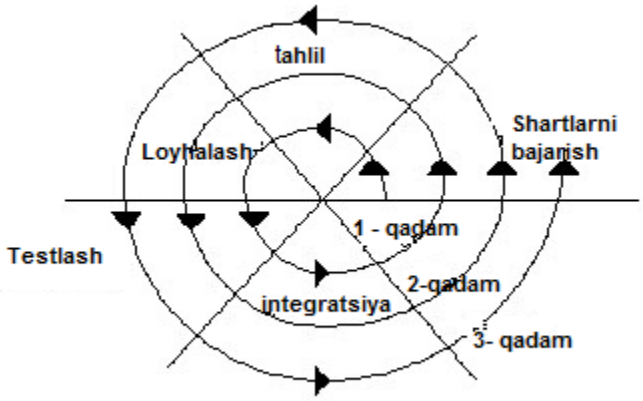
Ushbu modelning kamchiliklari shundaki, AT-larni yaratishning haqiqiy jarayoni odatda bunday qattiq sxemaga to'liq mos kelmaydi. Deyarli doimiy ravishda oldingi bosqichlarga qaytish, ilgari qabul qilingan qarorlarni aniqlashtirish yoki qayta ko'rib chiqish zarurati mavjud . Natijada, natijalarni olish muddati kechiktirildi va foydalanuvchilar faqat tizim bilan barcha ishlar tugagandan so'ng o'z fikrlarini bildirishlari mumkin. Shu bilan birga, avtomatlashtirilgan ob'ektning modellari tasdiqlangan vaqtga qadar eskirishi mumkin.

Bosqichli ***model*** odatda har qanday bosqichda va fazalararo sozlashlarda o'rta muddatli nazoratni o'z ichiga oladi. Kaskad modeliga qaraganda kamroq murakkablikni ta'minlaydi, ammo har bir bosqichning yashash muddati butun hayot aylanishiga teng bo'ladi. Bosqichlararo sozlashlar kaskad modeliga nisbatan rivojlanish jarayonining murakkabligini kamaytirishi mumkin (6.2-rasm).



6.2 – rasm. Dasturiy ta'minotni bosqichma-bosqich ishlab chiqish sxemasi.

Spiral model (6.3-rasm) SMning dastlabki bosqichlarida strategiyani ishlab chiqish, talablarni tahlil qilish va oldindan batafsil loyihalashtirish amalga oshirilganligi bilan tavsiflanadi. Shu bilan birga, texnik echimlarning maqsadga muvofiqligini tekshirish va asoslashga imkon beradigan prototiplar (masxarabozlar) yaratilgan. Spiralning har bir bobini mahsulotning parchasi yoki versiyasini yaratish bosqichma-bosqich modeliga mos keladi. Unda loyihaning maqsadi va xususiyatlari ko'rsatiladi, uning sifati aniqlanadi va spiralning keyingi bobini ishlash rejalashtirilgan. Natijada amalga oshirilayotgan oqilona variant tanlandi.



6.3- rasm. Spiral model.

Ma'lumotlar bazasining hayot aylanishi

Ma'lumotlar bazasining hayotiy aylanishi (GLCB) bu ma'lumotlar bazasini loyihalash, amalga oshirish va saqlash jarayonidir. JCBD etti bosqichdan iborat:

1. oldindan rejalashtirish;
2. texnik-iqtisodiy tekshirish;
3. talablarni aniqlash;
4. kontseptual dizayn;
5. mantiqiy dizayn;

6. jismoniy dizayn;
7. ish faoliyatini baholash va ma'lumotlar bazasini qo'llab-quvvatlash.

Har qanday iqtisodiy obyektni rivojlantirishda rivojlanishning keyingi yutuqlariga erishish uchun xodimlar tomonidan ishlatiladigan ma'lumotlar birlashtirilishi, ma'lumotlar bazasida bo'lish uchun foydalanilishi va korporativ manba sifatida qabul qilinishi zarurligi tushuniladi.

1. *Ma'lumotlar bazasini oldindan rejalashtirish,* nomutanosib ma'lumotlardan birlashtirilgan tizimga o'tish jarayonidagi muhim bosqichdir. Ushbu bosqichda ishlatiladigan dasturlar va ishlab chiqish jarayonida va ular bilan bog'liq fayllar to'g'risida ma'lumotlar to'planadi. Bu joriy dasturlar va ularning ma'lumotlari qanday ishlatilishi o'rtasida aloqa o'rnatishga yordam beradi. Bundan tashqari, u kelajakda ma'lumotlar bazasi talablarini aniqlashga imkon beradi. Ma'lumotlar ma'lumotlarning umumlashtirilgan kontseptual modeli rasmida hujjatlashtiriladi.

2. *Texnik-iqtisodiy sinov* uchta masala bo'yicha hisobotlarni tayyorlashni o'z ichiga oladi:

1. rejalashtirilgan ma'lumotlar bazasini amalga oshirish uchun texnologiya - zarur uskunalar va dasturiy ta'minot mavjudmi (texnologik asoslash);
2. ma'lumotlar bazasi rejasini muvaffaqiyatli amalga oshirish uchun xodimlar, vositalar va mutaxassislar bormi (operatsion-asoslash);
3. rejalashtirilgan ma'lumotlar bazasi to'ladimi yoki yo'qmi (iqtisodiy samaradorlik).

3. *Talablarni aniqlash* . Ushbu bosqichda quyidagilar aniqlanadi:

- Ma'lumotlar bazasining maqsadlari
- turli tarkibiy bo'linmalar va ularning rahbarlarining axborotga bo'lgan ehtiyojlari;
- apparat talablari;
- dasturiy ta'minotga qo'yiladigan talablar.

4. *Kontseptual dizayn.* Ushbu bosqichda domen ma'lumotlarini foydalanuvchi vakilliklarining batafsil modellari yaratiladi. Keyin ular ma'lumotlar bazasiga yuklanishi kerak bo'lgan korporativ ma'lumotlarning barcha elementlarini to'playdigan *kontseptual modelga* birlashtirilgan. Ushbu model *kontseptual ma'lumotlar bazasining sxemasi* deb ham ataladi.

5. *Mantiqiy dizayn.* Ushbu bosqichda ma'lumotlar modeli turini tanlash amalga oshiriladi. Kontseptual model tanlangan modelga xos bo'lgan tuzilmalar asosida *mantiqiy modelga* taqsimlangan.

6. *Fizik dizayn.* Ushbu bosqichda mantiqiy model ma'lumotlar bazasini jismoniy saqlash usullarini aniqlash uchun zarur bo'lgan xususiyatlar bilan kengaytiriladi, masalan, saqlash moslamalari, ma'lumotlar bazasiga kirish usullari, kerakli xotira hajmi, ma'lumotlar bazasini saqlash qoidalari va boshqalar.

7. *Baholash va ma'lumotlar bazasini qo'llab-quvvatlash.* Baholashda foydalanuvchilar qanday ma'lumotlarning hisobga olinmaganligini aniqlash uchun so'rov o'tkaziladi. Agar kerak bo'lsa, mo'ljallangan ma'lumotlar bazasiga

o'zgartirishlar kiritiladi. Foydalanuvchilar ma'lumotlar bazasi bilan ishlashga o'rgatiladi. Biznes kengayishi va o'zgarishi kerakligi sababli, ma'lumotlar bazasini qo'llab-quvvatlash o'zgarishlar kiritish, yangi ma'lumotlar qo'shish , ma'lumotlar bazasi bilan ishlaydigan yangi dasturlarni ishlab chiqish orqali ta'minlanadi.

6.2 Ma'lumotlar bazasini loyihalash

Kontseptual modelni qurish eng murakkab va jarayonni rasmiylashtirish qiyin. Hech qanday konstruktiv usul yo'q, tuzilish jarayoni asosan dizaynerning tajribasiga va professional - simulyatsiya qilingan mavzu sohasidagi xodimning ishtirokiga asoslangan san'atdir.

Ma'lumotlar bazalarining kontseptual dizayni - bu modelni yaratish jarayoni, ishlatiladigan ma'lumot, uni taqdim etishning har qanday fizik jihatlariga bog'liq emas - ma'lumotlar bazasining kontseptual vakillikini yaratish, shu jumladan eng muhim ob'ektlarning turlarini va ular o'rtasidagi munosabatlar va xususiyatlarni aniqlash. Keyingi - mantiqiy dizayn bosqichi.

Ma'lumotni modellashtirishga ikkita yondashuvni ajratish mumkin (ma'lumotlarning kontseptual modelini loyihalash)

- fan sohasini tizimli tahlil qilish natijasida (semantik modellashtirish ham deyiladi) - eng keng tarqalgan, hech bo'lmaganda taniqli asarlarda.
- foydalanuvchilarning axborotga bo'lgan ehtiyojini tahlil qilish natijasida (mavjudlarini normallashtirish asosida ma'lumotlar bazasini loyihalash).

Birinchi yondashuv eng keng tarqalgan (hech bo'lmaganda taniqli asarlarda), Chen tomonidan 1976 yilda aniqlangan yondashuvga asoslanadi. (Chen Peter). Of Vujudga - munosabat modeli 1976 ER -model.

6.3 Amalga oshirish

CASE vositalari - bu dasturiy ta'minotning hayot siklining individual bosqichlarini avtomatlashtiradigan CASE texnologiyalariga asoslangan avtomatlashtirilgan vositalar. Barcha zamonaviy CASE vositalari turiga va toifasiga ko'ra tasniflanishi mumkin. Turlarga ko'ra tasniflash dasturiy ta'minotning hayot aylanish jarayoniga funktsional yo'nalishini aks ettiradi. Toifalar bo'yicha tasniflash funktsiyalar bo'yicha integratsiyalashuv darajasini belgilaydi va eng avtonom vazifalarni (inglizcha vositalarda) hal qiladigan individual mahalliy vositalarni, hayot siklining ko'p bosqichlarini qamrab oluvchi qisman o'rnatilgan vositalar to'plamini (asboblar to'plami) va axborot tizimlarining butun hayot siklini qo'llab-quvvatlaydigan to'liq integratsiyalashgan vositalarni o'z ichiga oladi.

Turlarga ko'ra tasniflash quyidagi asosiy CASE-vositalarni o'z ichiga oladi:

1. Domen modellarini yaratish va tahlil qilish uchun yaratilgan tahlil vositalari (Bpwin, Design / IDEF);
2. Dizayn spetsifikatsiyalarini yaratish uchun ishlab chiqilgan tahlil va dizayn vositalari (CASE. Tahlilchi, Vantage Team Builder, Designer / 2000, Silverrun, PRO-IV);

3. Ma'lumotlar bazasini loyihalash vositalari va eng keng tarqalgan MBBT uchun ma'lumotlar sxemasini yaratishni ta'minlaydigan vositalar (Silverrun, Vantage Team Builder, Designer / 2000, ERwin, S-Designor);

4. Ilovalarni ishlab chiqish vositalari va kod generatorlari (Vantage Team Builder, Silverrun, PRO-IV);

5. Dastur kodlari tahlili, ma'lumotlar bazasi sxemalari va ularning asosida turli xil modellar va dizayn spetsifikatsiyalarining rasmlanishini ta'minlaydigan reinjiniring vositalari. Ma'lumotlar bazasi sxemasini tahlil qilish vositalari quyidagilardan iborat: (Silverrun, Vantage Team Builder, Designer / 2000, Erwin, S-Designor). Rational Rose va Object Team kabi vositalar dastur kodlarini tahlil qilish uchun ishlatiladi.

Amerikalik Computer Systems Advisers (CSA) kompaniyasining CASE-vositasi biznes-klassdagi axborot tizimlarini tahlil qilish va loyihalashda ishlatiladi va ko'proq hayot siklining spiral modeliga qaratilgan. Funktsional va axborot modellarining alohida tuzilishiga (ma'lumotlar oqimi diagrammasi va mantiqiy munosabatlarga oid diagrammalar) asoslangan har qanday metodologiyani qo'llab-quvvatlash uchun qo'llanishi mumkin. Silverrun modulli tuzilishga ega va har biri alohida mahsulot bo'lgan to'rt moduldan iborat.

Ma'lumotlar oqimi diagrammasi rasmida biznes-jarayon modellarini qurish uchun modul (BMP - Business Process Modeler) o'rganilayotgan tashkilot yoki yaratilayotgan axborot tizimining ishlashini taqlid qilishga imkon beradi. Ma'lumotni kontseptual modellashtirish moduli (ERX - Entity-Relationship eXpert) ma'lum bir amalga oshirish bilan bog'lanmagan mantiqiy obyektlar o'rtasidagi munosabatlar ma'lumotlari modellarini qurishni ta'minlaydi. Relyatsion modellashtirish moduli (RDM - Relational Data Modeler) relyatsion ma'lumotlar bazasida amalga oshirish uchun mo'ljallangan batafsil "ob'ekt-munosabatlar" modellarini yaratishga imkon beradi.

Workgroup Repository Manager (WRM) barcha modellar uchun umumiy bo'lgan ma'lumotlarni saqlash uchun ma'lumotlar lug'ati sifatida ishlatiladi va shuningdek, Silverrun modullarining yagona dizayn muhitiga integratsiyasini ta'minlaydi. Silverrun-ning turli xil modellarning tarkibiy qismlari o'rtasida qattiq o'zaro nazoratning yo'qligi (masalan, turli darajadagi ma'lumotlar oqimi diagrammalari o'rtasidagi o'zgarishlarni avtomatik ravishda tarqatish qobiliyati) vizual modellashtirish vositalarining yuqori moslashuvchanligi va xilma-xilligi uchun to'lovdir. Ammo bu kamchilik faqatgina kaskadli dasturiy hayot siklining modeli ishlatilgan taqdirdagina ahamiyatli bo'lishi mumkin. Ma'lumotlar bazasi sxemalarini avtomatik ravishda yaratish uchun Silverrun-da eng keng tarqalgan ma'lumotlar bazalari uchun ko'priklar mavjud: Oracle, Informix, DB2, Ingres, Progress, SQL Server, SQLBase, Sybase. Ilovalarni ishlab chiqish vositalariga ma'lumotlarni uzatish uchun 4GL tillariga ko'priklar mavjud: JAM, PowerBuilder, SQL Windows, Uniface, NewEra, Delphi. Silverrun tizimi uchta platformada - MS Windows, Macintosh, OS/2 Presentation Manager - ular orasida dizayn ma'lumotlarini almashish imkoniyati bilan amalga oshiriladi.

Vantage Team Builder - bu dasturiy ta'minotning hayotiy sikl modelini amalga

oshirishga yo'naltirilgan o'rnatilgan dasturiy mahsulot. Vantage Team Builder quyidagi funktsiyalarni bajaradi:

1) ma'lumotlar oqimi diagrammalarini, "mantiqiy ob'ekt - munosabatlar", ma'lumotlar tuzilmalari, dasturiy tuzilmaviy diagrammalar va ekran rasmlarining ketma-ketligini loyihalash;

2) dasturiy ta'minot jadvallari, indekslari, yaxlitligi cheklanganligi va saqlanadigan protseduralarni yaratish uchun to'liq dasturiy ta'minot va SQL-kod yaratish bilan maqsadli DBMSning 4GL tilida dasturlarni kod yaratish;

3) C dasturlashtirilgan SQL bilan;

4) versiyani boshqarish va loyihaning konfiguratsiyasi;

5) namunaviy va individual shablonlarni loyihalash hujjatlarini yaratish;

6) loyiha ma'lumotlarini eksport va import qilish.

Vantage Team Builder ishlatilgan ma'lumotlar bazasi (Oracle, Informix, Sybase, Ingress) yoki dasturlarni ishlab chiqish vositalariga (Uniface) qarab turli xil konfiguratsiyalarda etkazib beriladi. Vantage Team Builder konfiguratsiyasi bitta texnologik dizayn muhitida ikkita tizimdan birgalikda foydalanishni ta'minlaydi, ayni paytda ma'lumotlar bazasi sxemalari (SQL modellari) Uniface bazasiga beriladi va aksincha Uniface vositalari tomonidan yaratilgan amaliy modellar Vantage Team Builder bazasiga o'tkazilishi mumkin. . Ikkala tizim omborlari o'rtasidagi nomuvofiqliklar maxsus yordam dasturi yordamida o'rnatiladi. Ekran rasmlari SQL modelini import qilgandan keyin Uniface-da rasmlar ketma-ketligi diagrammasi (FSD) asosida ishlab chiqilgan. Vantage Team Builder barcha asosiy Unix platformalarida ishlaydi (Solaris, SCO UNIX, AIX, HP-UX) va VMS.

Oracle-ning Dizayneri/2000 CASE vositasi - bu CASE vositasi bo'lib, u Developer/2000 dasturlarni ishlab chiqish vositalari bilan birgalikda, Oracle DBMS-laridan foydalanadigan tizimlar uchun dasturiy ta'minotning to'liq aylanishini ta'minlaydi. Designer/2000 quyidagi tarkibiy qismlarni o'z ichiga oladi:

1) Repository Administrator - omborni boshqarish vositalari (ilovalarni yaratish, o'chirish, turli foydalanuvchilarning ma'lumotlarga kirishini boshqarish, ma'lumotlarni eksport qilish va import qilish);

2) Repository Object Navigator - omborga kirish vositasi. Omborning barcha elementlariga kirish uchun ko'p oynali ob'ektga yo'naltirilgan interfeysni taqdim etish;

3) Process Modeller - biznes jarayonlarini qayta boshqarish va global sifat menejmenti tizimiga asoslangan biznes faoliyatini tahlil qilish va modellashtirish vositasi;

4) Modeller tizimlari - ishlab chiqilgan axborot tizimining funktsional va axborot modellarini qurish uchun vositalar to'plami, shu jumladan ob'ektlar o'rtasidagi munosabatlar diagrammalarini, funktsional ierarxiya diagrammalarini, ma'lumot oqimlarining diagrammalarini tuzish vositalari va turli xil ombor ob'ektlarining o'zaro munosabatlarini tahlil qilish va o'zgartirish vositasi ;

5) tizimlar dizayneri - axborot tizimlari uchun dizayn vositalarining to'plami, shu jumladan aloqador ma'lumotlar bazasi tuzilishini, shuningdek ma'lumotlar bilan o'zaro ta'sir ko'rsatadigan diagramma tuzish vositalari, SQL tilida saqlanadigan

protseduralar yordamida amalga oshiriladigan ilovalar ierarxiyasi, tuzilishi va mantiqiyligi;

6) Server Generator - Oracle ma'lumotlar bazasi ob'ektlarini tavsiflovchi (jadvallar, indekslar, kalitlar, ketliklar va boshqalar). Oracle mahsulotlaridan tashqari MBBT Informix, DB / 2, Microsoft SQL Server, Sybase uchun, shuningdek ODBC orqali kiradigan ma'lumotlar bazalari uchun ma'lumotlar bazasini yaratish va qayta qurish mumkin;

7) Forms Generator - har xil ekran rasmlari, ma'lumotlarni boshqarish vositalari, yaxlitlikni cheklash tekshiruvi va avtomatik so'rovlarni o'z ichiga olgan ilova generatori; 8) Repository Reports - standart hisobotlar generatori. Funktsional muhit Designer/2000 - Windows 3.x, Windows 95, Windows NT.

Erwin IDEF1X metodologiyasidan foydalangan holda ma'lumotlar bazasini mantiqiy modellashtirish vositasidir. Erwin ma'lumotlar bazasi sxemasini loyihalashtirish, maqsadli MBBT tilida uning tavsifini yaratish (Oracle, Informix, DB/2, Ingres, Progress, SQL Server, SQLBase, Sybase va boshqalar) va mavjud ma'lumotlar bazasini reinjiringlashtirishni amalga oshiradi. Erwin bir nechta turli xil konfiguratsiyalarda bo'lib, eng keng tarqalgan 4GL dasturlarni ishlab chiqish vositalariga mo'ljallangan. Erwin/Open versiyasi PowerBuilder va SQLWindows dasturlarini ishlab chiqish vositalariga to'liq mos keladi va yaratilgan ma'lumotlar bazasining tavsifini to'g'ridan-to'g'ri asboblar bazasiga eksport qilishga imkon beradi.

S-Dizayn - bu CASE - ma'lumotlar bazasini loyihalash uchun vositadir. S – Designor ma'lumotlar bazasini modellashtirishning standart metodologiyasini joriy qiladi va Oracle, Informix, DB/2, Ingres, Progress, SQL Server, SQLBase, Sybase va boshqalar kabi ma'lumotlar bazalarini tavsiflarini yaratadi. Mavjud tizimlar uchun ma'lumotlar bazasini qayta qurish amalga oshiriladi.

Savollar:

1. Axborot tizimining hayot aylanishi nima?
2. IP hayotiy siklini rivojlantirish uchun asosiy modellar qanday?
3. Axborot tizimining hayot aylanishi va ma'lumotlar bazasining hayot aylanishi o'rtasidagi farq nima?
4. Kontseptual dizayn nima?
5. CASE vositalarining qaysi toifalarini bilasiz?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

VII Bob. Ma'lumotlar bazasini normallashtirish

7.1 Ma'lumotlar bazasini yaratishda muammolar

Biz loyihalash usullarini muhokama qilishdan oldin relatsion ma'lumotlar bazasi jadvallarini, ba'zi sxemalar etarli bo'lishi mumkinligini ko'rib chiqamiz. Xususan, biz SUPPLIERS (NOMI, MANZIL, TOVAR, NARXI) munosabatlar sxemasiga murojaat qilamiz.

NOMI	MANZILI	MAHSULOT	NARXI
AGATA-IMPEKS	Ata-tyurk k. 208	Kompyuter P IV	2 000000
NURON	Navoiy k. 158	Monitor LCD	254 000
TS-TECHNOLOGY	Yangiyo'l k. 64	Kompyuter P IV	1 800 000
NURON	Navoiy k. 158	Klaviatura	25 000
SHARIFA-T	Chilonzor 664	Sichqoncha	15 000

7.1- rasm. Ta'minlovchilarning o'zaro aloqasi sxemasi

Ushbu kontaktlarning bir xel formaga olib keladigan muammolar mavjud:

1. *Ko'paytirish*. Vendor manzili har ustun uchun takrorlanadi nazorat qiladi.
2. *Potentsial nomuvofiqlik (anomaliyalarni yangilash).* Ishdan bo'shatish tufayli biz provayder manzilini bitta katakchada yangilab, ikkinchisida o'zgarishsiz qoldirishimiz mumkin. Shunday qilib, ba'zi etkazib beruvchilar uchun yagona manzil yo'qligi aniqlanishi mumkin. Biroq, intuitiv ravishda biz shunday bo'lishi kerakligini his qilamiz.
3. *Inklyuzion anomaliyalar.* Ta'minlovchining manzili, agar u hozirda kamida bitta mahsulotni etkazib bermasa, ma'lumotlar bazasida yozib bo'lmaydi. Siz, albatta, ushbu sotuvchi uchun naychalarni MAHSULOT va NARXI tarkibiy qismlariga qo'yishingiz mumkin. Ammo, agar u biron bir mahsulotni etkazib berishni boshlasa, biz aniqlanmagan qiymatlar bilan kornishini o'chirishni unutmaymizmi? Bundan ham yomoni, MAHSULOT va MAHSULOT_NOMI ushbu munosabatlarning kalitini tashkil qiladi va kalitda aniqlanmagan qiymatlari bo'lgan toplilarni topish qiyin yoki imkonsiz bo'lishi mumkin.
4. *Anomaliyalarni olib tashlash.* Qarama-qarshi muammo, ma'lum bir etkazib beruvchi tomonidan etkazib beriladigan barcha tovarlarni olib tashlash zarur bo'lganda paydo bo'ladi, natijada biz o'z manzilimizni bilmasdan yo'qotamiz.

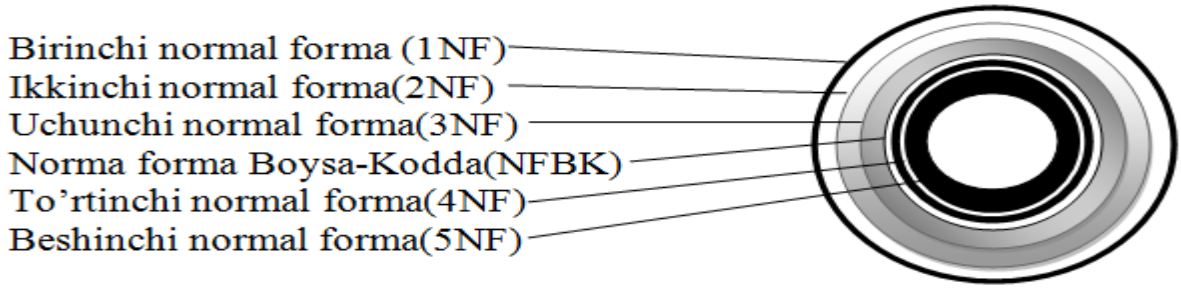
Savol tug'iladi: "Qanday qilib yomon munosabatlar sxemasining yaxshi o'rnini topish mumkin?"

7.2 Normal formalar

O'zaro munosabatlar, ular ta'sir qiladigan modifikatsiya anomaliyalari turlariga qarab tasniflanishi mumkin. 1970-yillarda relyatsion ma'lumotlar bazasi nazariyotchilari ushbu turlarni bosqichma-bosqich bekor qilmoqdalar. Kimdir anomaliyani topdi, uni tasnifladi va uning paydo bo'lishining oldini olish haqida

o'yladi. Har safar bu sodir bo'lganda, munosabatlarni o'rnatish mezonlari yaxshilandi. Ushbu munosabatlar sinflari va anomaliyalarning oldini olish usullari normal shakllar deb ataladi. O'zining tuzilishiga qarab, munosabatlar birinchi, ikkinchisida yoki boshqa normal shaklda bo'lishi mumkin.

1970 yilgi muhim maqoladan so'ng, Kodd va boshqalar birinchi, ikkinchi va uchinchi normal rasmlarni (1NF, 2NF va 3NF) aniqladilar. Keyinchalik normal Boyse-Kodd rasmi (NFBK) joriy etildi, shundan so'ng to'rtinchi va beshinchi normal formalar aniqlandi. 7.2 -rasmda, ushbu normal formalar ichkariga joylashtirilgan. Ya'ni, ikkinchi normal formaga nisbat birinchi normal formadagi nisbatdir va 5NF (beshinchi normal rasm) ning nisbati bir vaqtning o'zida 4NF, NFBK, 3NF, 2NF va 1NFda bo'ladi.



7.2-rasm. Normal formalar

Ushbu normal formalar yordam berdi, ammo ular jiddiy cheklovga ega edi. Ushbu normal formalarning har qandayida barcha anomaliyalarni yo'q qilishiga kafolat beradigan hech qanday nazariya yo'q edi: har bir rasm faqat ayrim turlarini yo'q qilishga qodir. Bu holat 1981 yilda R. Fagin domino kalitli domen rasmi yoki DKNF (domen / kalit normal formasi, DK/NF) deb nomlangan yangi normal forma kiritganda hal qilindi. Fagin o'zining muhim maqolasida DKNF-ning nisbati, ularning turlaridan qat'i nazar, barcha modifikatsiya qilingan anomaliyalardan xoli ekanligini ko'rsatdi. U shuningdek, modifikatsiyasiz anomaliyalarsiz har qanday munosabatlar DKNFda bo'lishi kerakligini ko'rsatdi.

DKNF joriy etilishidan oldin, relatsiya ma'lumotlar bazasi nazariyotchilari tobora ko'proq anomaliyalar va normal rasmlarni qidirishda davom etishlari kerak edi. Faginning isboti vaziyatni soddalashtirdi. Agar biz DKNFga munosabat bildira olsak, unda modifikatsiya qilinadigan anomaliyalar bo'lmaydi. DKNF-ga qanday munosabatda bo'lish kerakligi bilan bog'liq.

Oddiy rasmlarning asosiy xususiyatlari:

- har bir keyingi normal forma avvalgisidan yaxshiroq ma'noda;
- keyingi normal forma o'tishda avvalgi normal xususiyatlarning xususiyatlari saqlanib qoladi.

Jarayoni normallashtirish usuliga, avvalgi normal formagi munosabatlarni quyidagi normal forma talablariga javob beradigan ikki yoki undan ortiq munosabatlarga ajratish asosida amalga oshiriladi.

7.3 Normallashtirish va funktsional bog'lanishlar

Normallashtirish - bu ba'zi bir kamchiliklari bo'lgan munosabatlarni ushbu kamchiliklarga ega bo'lmagan munosabatlarga aylantirish jarayoni. Bundan ham muhimi, normalizatsiya munosabatlarni maqsadga muvofiqligi va to'g'riligini aniqlash uchun mezon sifatida ishlatilishi mumkin. Yaxshi tuzilgan munosabat nima degan savol ko'plab nazariy tadqiqotlar mavzusi bo'ldi. Normallashtirish atamasi ma'lumotlar bazasi texnologiyasining kashshoflaridan biri E.F. Koddga tegishli bo'lib, u turli xil normal munosabatlar formalarni belgilab bergan.

Amaliyotda munosabatlarning eng muhim normal formalari relyatsion ma'lumotlar bazasi nazariyasida asosiy *bo'lgan funktsional qaramlik* tushunchasiga asoslanadi. Keyingi muhokama qilish uchun bizga bir nechta ta'riflar kerak.

Funktsional bog'lanish - bu atributlar o'rtasidagi munosabatlardir.

Aytaylik, agar biz bitta atributning qiymatini bilsak, boshqa atributning qiymatini hisoblashimiz mumkin (yoki topamiz).

Masalan, agar biz mijozning hisob raqamini bilsak, u holda uning hisobining holatini aniqlashimiz mumkin. Bunday holda, mijozning hisobi holati atributi mijozning hisob raqami atributiga funktsional ravishda bog'liq deb ayta olamiz.

Umuman olganda, Y atributi X atributiga funktsional ravishda bog'liq, agar X ning qiymati Y ning qiymatini belgilasa, boshqacha qilib aytganda, agar X ning qiymatini bilsak, Y ning qiymatini aniqlashimiz mumkin.

Tenglamalar funktsional bog'liqlikni ifodalaydi. Masalan, agar biz sotib olingan mahsulotning narxini va miqdorini bilsak, sotib olish narxini quyidagi formula yordamida aniqlashimiz mumkin:

Xarajat = Narx x Soni.

Bunday holda, biz narx atributi narx va miqdor atributlariga funktsional bog'liq deb aytishimiz mumkin.

Aloqadagi atributlar orasidagi funktsional bog'liqliklar odatda tenglamalar bilan ifodalanmaydi. Aytaylik, har bir talabaga o'ziga xos identifikatsiya raqami beriladi va har bir talabada bitta va bitta mutaxassislik bo'ladi. Talaba raqamiga ega bo'lgan holda, biz uning ixtisosligini bilib olamiz, shuning uchun ixtisos atributi funktsional ravishda Student Number atributiga bog'liq. Yoki hisoblash laboratoriyasida kompyuterlarni ko'rib chiqing. Har bir kompyuterda asosiy xotiraning ma'lum o'lchamlari mavjud, shuning uchun Xotira hajmi atributi funktsional ravishda kompyuterning seriya raqami atributiga bog'liq.

Tenglamadan farqli o'laroq, bunday funktsional bog'liqliklarni arifmetikadan foydalanib hal qilib bo'lmaydi; buning o'rniga ular ma'lumotlar bazasida saqlanadi. Aslida, ma'lumotlar bazasi faqat funktsional qaramlikni saqlash va berish uchun etarli narsaga ega ekanligi haqida bahslashish mumkin.

Funktsional bog'liqliklar quyidagicha belgilanadi:

Student soni > Mutaxassisligi

Kompyuterning seriya raqami > Xotira hajmi

Birinchi ibora quyidagicha o'qiydi: " Student soni atributi mutaxassislik atributini funktsional ravishda belgilaydi", " talaba raqami atributi mutaxassislik

atributini belgilaydi" yoki "Mutaxassislik atributi talaba soni atributiga bog'liq ". O'qning o'ng tomonidagi qo'shimchalar ***aniqlovchi*** deyiladi .

Funktsional bog'liqliklar atributlar guruhlarini o'z ichiga olishi mumkin. BAHOLASH nisbati (talabalar soni , intizomi, darajasi) ni ko'rib chiqing . Talaba soni va intizomning kombinatsiyasi bahoni aniqlaydi. Bunday funktsional munosabatlar quyidagicha yoziladi:

(Talaba soni , intizomi)> Sinf

Baholarni aniqlash uchun talabalar soni ham, intizomi ham talab qilinadi. Ushbu funktsional qaramlikni ajratib bo'lmaydi, chunki na talabalar soni, na intizom bahoni o'zlari belgilamaydi.

Kalit - bu satrni aniq belgilaydigan bir yoki bir nechta atributlar guruhi. Har bir munosabatlar kamida bitta kalitga ega. Ushbu gap to'g'ri bo'lishi kerak, chunki hech qanday munosabat bir xil chiziqlarga ega bo'lolmaydi va shu sababli, kalit bilan aloqaning barcha atributlaridan iborat bo'ladi.

Birinchi normal forma

Aloqa ta'rifini qondiradigan har qanday ma'lumotlar jadvali birinchi normal forma (1NF) deyiladi. Eslatib o'tamiz, jadvalning o'zaro aloqasi bo'lishi uchun quyidagilarni bajarish kerak: jadval kataklarida bitta qiymat bo'lishi kerak va qiymat sifatida takroriy guruhlar ham, massivlar ham kiritilmaydi. Bitta ustundagi barcha yozuvlar (atribut) bir xil turda bo'lishi kerak. Har bir ustun o'ziga xos nomga ega bo'lishi kerak, ammo jadvaldagi ustunlarning tartibi ahamiyatli emas. Va nihoyat, jadval ikkita bir xil qatorga ega bo'lolmaydi va satrlarning tartibi ahamiyatsizdir.

Ikkinchi normal forma

Agar uning barcha kalit bo'lmagan atributlari butun kalitga bog'liq bo'lsa, munosabatlar ikkinchi normal formada bo'ladi. Ushbu ta'rifga ko'ra, agar munosabatlar kalit sifatida bitta atributga ega bo'lsa, u avtomatik ravishda ikkinchi normal formada bo'ladi.

Kalit bitta atribut bo'lganligi sababli, har bir kalit bo'lmagan atribut butun kalitga bog'liq va qisman bog'liqliklar bo'lmaydi. Shunday qilib, ikkinchi normal forma faqat kompozit kalitlarga ega bo'lgan munosabatlar uchun qiziqish uyg'otadi.

Uchinchi normal forma

Ikkinchi normal formagi munosabatlar ham g'ayritabiiyga ega bo'lishi mumkin. 7.3-rasmda QO'SHIMCHA aloqasini ko'rib chiqamiz. Bu erda kalit talabalar soni bo'lib, talabalar soni > yotoqxona va yotoqxona> to'lovlari funktsional bog'liqliklar mavjud. Bu bog'liqliklar har bir talaba faqat bitta yotoqxonada yashashi sababli yuzaga keladi va har bir yotoqxonada yashovchi barcha talabalar uchun bir xil haq olinadi. Misol uchun, har bir yotoqxonada Randolf- Hall birga yashayotgan chorak uchun $ 3200 to'laydi.

Talabaraqami	Turar joy	To’lov
100	Rendolf	3200
150	Ingersol	3100
200	Rendolf	3200

250	Pitkin	3100
300	Rendolf	3200

7.3- rasm. Yotoqxonalar sxemasi.

Talaba raqami yotoqxona atributini belgilaganligi sababli, lekin Yotoqxona "To'lov" atributini, so'ngra bilvosita "Talaba raqami"> "To'lov" ni belgilaydi. Funktsional bog'liqliklarning ushbu tuzilishi o'tuvchi qaramlik deb ataladi. Chunki Talaba raqami atributi belgilaydi, xususiyat esa Yotoqxona atributi orqali to'lov qilinadi.

QO'SHIMCHA munosabatlarining kaliti bu bitta atribut bo'lgan talabalar soni, shuning uchun munosabatlar ikkinchi oddiy rasmda (Yotoqxona va to'lov miqdori talabalar soni atributi tomonidan belgilanadi). Shunga qaramay, QO'SHIMCHA aloqasi transektiv qaramlik tufayli g'ayritabiiylarga ega.

7.2-rasmdagi munosabatlarning ikkinchi qatorini o'chirsak nima bo'ladi? Biz nafaqat 150-sonli talabaning Ingersall-Xollda yashashini, balki ushbu yotoqxonada yashash 3100 dollarga tushishini ham aniqligini bilamiz. Bu o'chirish g'ayritabiiydir. Carrigg-Hallga qanday qilib 3500 dollar ekanligini yozishimiz mumkin? Hech bo'lmaganda bitta talaba u erga ko'chib o'tishga qaror qilgunga qadar. Bu qo'shilish g'ayritabiiy.

Talabaraqami	Turar joy
100	Rendolf
150	Ingersol
200	Rendolf
250	Pitkin
300	Rendolf

Talabaraqami	Turar joy
100	Rendolf
150	Ingersol
200	Rendolf
250	Pitkin
300	Rendolf

7.4-rasm. Normallashtirilgan yotoqxona sxemasi.

Ikkinchi normal formadagi munosabatdan g'ayritabiiylarni olib tashlash uchun 7.4-rasmda o'tuvchi qaramlikni yo'q qilish kerak. Agar munosabat ikkinchi normal formada bo'lsa va tranzaktiv bog'liqlikka ega bo'lmasa, munosabat uchinchi normal formada bo'ladi.

Boyke – Kodd normal formasi

Talaba raqami	Turar joy	O’qituvchi
100	Matematika	Koshi
150	Psixologiya	Yung
200	Matematika	Riman
250	Matematika	Koshi
300	Psixologiya	Riman

7.5-rasm. O'qituvchilar munosabatlari chizmasi.

Talabalar bir nechta sohalarda ixtisoslashishi mumkinligi

sababli, Student soni atributi Mutaxassislik atributini aniqlamaydi. Bundan tashqari, talabada bir nechta maslahatchilar bo'lishi mumkinligi sababli, talabalar soni O'qituvchi atributini aniqlamaydi. Shunday qilib, talabalar sonining o'zi kalit bo'lolmaydi.

Kombinatsiya (Student soni, mutaxassisligi) O'qituvchi atributini, kombinatsiya (Student soni, o'qituvchi) esa Mutaxassislik atributini belgilaydi. Shuning uchun, bu kombinatsiyalarning har biri asosiy bo'lishi mumkin. Kalit bo'lishi mumkin bo'lgan ikkita yoki undan ortiq atributlar yoki atributlar guruhiga nomzod kalitlari deyiladi. Kalit sifatida tanlangan nomzod kaliti asosiy kalit deb ataladi.

Nomzod tugmachalariga qo'shimcha ravishda yana bir funktsional bog'liqlik ham e'tiborga olinishi kerak: o'qituvchi atributi Mutaxassislik atributini belgilaydi (o'qituvchilardan har biri faqat bitta fan bo'yicha maslahatchi; shuning uchun o'qituvchining ismini bilib, mutaxassislikni aniqlashimiz mumkin). Shunday qilib, o'qituvchi aniqlovchi hisoblanadi.

Ta'rifga ko'ra, MASLAHATCHIga munosabat o'zining birinchi normal holatidadir. U ikkinchi normal formaga ham mavjud, chunki u kalit bo'lmagan atributlarga ega emas (atributlarning har biri kamida bitta kalitning qismidir). Va nihoyat, bu bog'liqlik uchinchi normal formada, chunki u o'tkinchi bog'liqliklarga ega emas. Shunga bularning bariga qaramay, modifikatsiyaning g'ayritabiiylariga bog'liq.

300-sonli talabani universitetdan chiqarib yuborishsin. Agar biz 300-sonli talabalar haqidagi ma'lumot qatorini o'chirsak, Perle psixologiya bo'yicha maslahatchi ekanligidan mahrum bo'ldi. Bu olib tashlashning g'ayritabiiyligidir. Bundan tashqari, qanday qilib ma'lumot bazasida Keynsning iqtisodiy maslahatchisi ekanligini qayd etishimiz mumkin? Yo'q, hech bo'lmaganda iqtisodiy mutaxassislikka ega talaba paydo bo'lmaguncha. Bu olib tashlashning g'ayritabiiyligidir.

Faqat tasvirlangan kabi vaziyatlar, Boyse- Kodd (Boyse-Kodd normal formada, BK/SO) normal formada aniqlash uchun bizga yordam beradi. Har bir asosiy kalit nomzod bo'lsa nisbati BKNF bo'ladi. MASLAHATCHI bilan aloqalar BKNFda mavjud emas, chunki o'qituvchining aniqlovchisi nomzod kaliti emas.

Talabaraqami	O’qituvchi
100	Koshi
150	Yung
200	Riman
250	Koshi
300	Perls

Turar joy	O’qituvchi
matematika	Koshi
Psixologiya	Yung
Matematika	Riman
Matematika	Koshi
Rendolf	Perls

7.6 –rasm. O'qituvchi normallashtirilgan sxemasi.

Boshqa misollar singari, MASLAHATCHI aloqasini g'ayritabiiylari bo'lmagan ikkita munosabatlarga bo'lish mumkin. Masalan, 7.6-rasmda STUDENT - MASLAHATCHI (talabalar soni, o'qituvchi) va MASLAHATCHI - O'QUV

(o'qituvchi, mutaxassislik) o'rtasidagi munosabatlarda g'ayritabiiylari yo'q.

To'rtinchi normal forma

Talabaraqami	Yo’nalish	To’garak
100	Musiqa	Suzish
100	buxgalteriya	Suzish
100	Musiqa	Tennis
150	buxgalteriya	tennis

7.7-rasm. Mutaxassislik aloqasi sxemasi.

7.7-rasmdagi TALABA munosabatlarini ko'rib chiqamiz. Talabalar, mutaxassisliklar va bo'limlar o'rtasidagi aloqalarni ko'rsatadigan. Aytaylik, talabalar bir nechta mutaxassislikka ega bo'lib, turli xil bo'limlarda shug'ullanishlari mumkin. Bunday holda, bitta kalit birikma (talabalar soni, mutaxassislik, bo'lim). Masalan, 100 raqamiga ega talaba musiqa va buxgalteriya sohasiga ixtisoslashgan, shuningdek suzish va tennis seksiyalariga qatnashadi, va 150 raqamli talaba faqat matematikaga va yugurishga ixtisoslashgan.

Student soni va mutaxassisligi atributlari o'rtasida qanday bog'liqlik bor? Bu funktsional qaramlik emas, chunki talabada bir nechta mutaxassisliklar bo'lishi mumkin. Student soni atributining bir xil qiymati mutaxassislik atributining ko'pgina qiymatlariga mos kelishi mumkin. Bundan tashqari, bo'lim atributining ko'plab qiymatlari Student Number atributining bir xil qiymatiga mos kelishi mumkin.

Bu atributga bog'liqlik ko'p ma'noli bog'liqlik deb ataladi. Ko'p qiymatli qaramlik modifikatsiya anomaliyalariga olib keladi. Birinchidan, sekunddagi ma'lumotlarning ko'payishiga e'tibor bering. To'rtta yozuv 100-talabaga bag'ishlangan bo'lib, ularning har biri uning mutaxassisligi va tashrif buyurgan bo'limlaridan birini ko'rsatadi. Agar bir xil ma'lumotlar kamroq satrlarda saqlangan bo'lsa (aytaylik, ikkita chiziq bor edi - bittasi musiqa va suzish uchun, ikkinchisi buxgalteriya va tennis uchun), bu foydalanuvchilarni beparvo qiladi. Aniqlanishicha, 100 raqamiga ega talaba faqat musiqaga ixtisoslashganida suzadi va u faqat buxgalteriya ixtisosiga ega bo'lganida tennis o'ynaydi. Ammo bunday talqin mantiqsizdir. Mutaxassisliklar va bo'limlar bir-biridan mutlaqo mustaqil. Shuning uchun, bunday noto'g'ri xulosalarga yo'l qo'ymaslik uchun barcha mutaxassisliklar va bo'limlarning kombinatsiyasini saqlaymiz.

Talaba raqami	Yo’nalish	To’garak
100	Musiqa	Suzish
100	Buxgalteriya	Suzish
100	Musiqa	Tennis
150	Buxgalteriya	Tennis
100	Musiqa	Chang’i

7.8-rasm. Yo’nalish normallashtirilgan munosabatlar sxemasi.

Aytaylik, 100-sonli talaba chang'i bo'limiga yozilishga qaror qildi va shuning

uchun jadvalga [100, Musiqa, Ski], 7.8- rasmda ko'rsatilganidek chiziq qo'shamiz va hozirgi paytda 100-talaba chang'i sporti bilan faqat buxgalter sifatida emas, balki musiqachi sifatida shug'ullanayotganidan ham xulosa qilish mumkin. Ma'lumotlar izchil bo'lishi uchun biz mutaxassisliklar mavjud bo'lgan qatorlarni qo'shishimiz kerak va ularning har birida chang'i qismini ko'rsatamiz. Shunday qilib, 7.8-rasmda ko'rsatilganidek, biz [100, Buxgalteriya hisobi, chang'i sporti] qatorini qo'shishimiz kerak. Bu yangilanish g'ayritabiiy: bitta oddiy o'zgartirish uchun juda ko'p o'zgartirishlar talab qilinadi.

Umuman olganda, agar munosabatlar kamida uchta xususiyatga ega bo'lsa, ularning ikkitasi ko'p qiymatga ega bo'lgan va ularning qiymatlari faqat uchinchi atributga bog'liq bo'lganda, ko'p qiymatli munosabatlar mavjud. Boshqacha qilib aytganda, R (A, B, C)ga nisbatan, agar A va B harflari ko'p qiymatli tarzda aniqlansa, ko'p qiymatli qaramlik mavjud va B va C lar bir-birlaridan mustaqil. Oldingi misoldan ko'rinib turibdiki, Student soni mutaxassislik va bo'lim atributlarini aniqlaydi, ammo mutaxassislik va bo'lim o'zlari bir-birlaridan mustaqil.

Ushbu anomaliyalarni yo'q qilish uchun biz ko'p qadriyatli qaramlikdan xalos bo'lishimiz kerak. Biz buni ikkita aloqani yaratish orqali amalga oshiramiz, ularning har biri faqat bitta ko'p qiymatli atribut uchun ma'lumotlarni saqlaydi. Olingan munosabatlar anomaliyalarga ega bo'lmaydi. Bu munosabatlar 7.9-rasmda ko'rsatilgan, TALABAT-MAXSUS (talaba soni , mutaxassisligi) va TALABA Sektsiyasi (talabalar soni, bo'lim).

Talabaraqami	Yo'nalish
100	Musiqa
100	buxgalteriya
100	Musiqa

Talabaraqami	To'garak
100	Suzish
100	Suzish
100	Tennis
150	tennis

7.9-rasm. Ko'p qiymatli qaramlikni yo'q qilish.

Agar u BKNFda bo'lsa va noaniq bog'liqliklar bo'lmasa, munosabatlar to'rtinchi normal formada bo'ladi.

Beshinchi normal forma

Beshinchi normal forma (5NF) ma'lum darajada noma'lum bo'lgan qaramliklar bilan bog'liq. Biz yuqorida aytib o'tganimizdek, bir nechta kichik munosabatlarga bo'linishi mumkin bo'lgan munosabatlar haqida gaplashmoqdamiz, ammo keyin uni tiklash mumkin emas.

Ushbu vaziyat yuzaga kelgan shartlar aniq, intuitiv talqin qilinmaydi. Bunday qaramlikning oqibatlari nima ekanligini biz bilmaymiz; ularning amaliy oqibatlari bor-yo'qligini biz ham bilmaymiz.

Domen-kalit normal forma

1981 yilda Fagin muhim bir maqolani nashr etdi, unda u domen / kalitning normal formasini (DKNF) belgilab berdi. DKNF-da munosabatlar modifikatsiya g'ayritabiiylariga ega emasligini va bundan tashqari modifikatsiya g'ayritabiiylariga ega bo'lmagan har qanday munosabatlar DKNFda bo'lishi kerakligini ko'rsatadi.

Ushbu kashfiyot normal formalarni joriy qilishni to'xtatdi va endi yuqori tartibli normal formalarga ehtiyoj qolmaydi - hech bo'lmaganda modifikatsiyadagi g'ayritabiiylarni yo'q qilishni ko'rsatdi.

Savollar:

1. G'ayritabiiylik nima?
2. Ma'lumotlar bazalarida qanday g'ayritabiiylar mavjud?
3. G'ayritabiiy rasmlarning qanday turlari mavjud?
4. Barcha normal formalarda qanday bo'liqlig mavjud?
5. Normallashtirish nima?
6. Ko'p qiymatli bog'liqliklar nima?
7. Domen-kalit normal forma nima?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

VIII Bob. SQL tili. SQL tili operatorlarini yozish

8.1 SQL tarixi

Relyatsion ma'lumotlar bazalari paydo bo'lishidan oldin yaratilgan va shaxsiy kompyuterlarning ko'plab ma'lumotlar bazalarini boshqarish tizimlari (MBBT) uchun yaratilgan barcha ma'lumotlarni boshqarish tillari (NMD) mantiqiy fayl yozuvlari rasmida taqdim etilgan ma'lumotlar bilan ishlashga qaratildi. Bu foydalanuvchilarga ma'lumotlarni saqlashni tashkil etish to'g'risida batafsil ma'lumotga ega bo'lishlari va nafaqat kerakli ma'lumotni, balki qaerda joylashganligini va uni bosqichma-bosqich qanday olish kerakligini ko'rsatishi uchun etarli kuch talab qildi.

Dastur asosan IBM bo'limlarida (ISBL, SQL, QBE tillari) va AQSH universitetlarida (PIQUE, QUEL) amalga oshirildi. Ikkinchisi MBBT INGRES (Interfaol grafika va qidiruv tizimi) uchun yaratilgan, u 70-yillarning boshlarida PC universitetida yaratilgan. Kaliforniya bugungi kunda eng yaxshi professional ma'lumotlar bazasining beshligidan biridir. Bugungi kunda ushbu tillarning barchasidan QBE (Query-By-Example - Query by Example) va SQL to'liq saqlanib qolgan va ishlab chiqilgan va qolganlaridan olingan eng qiziqarli inshootlar ichki MBBT tillarini kengaytirish uchun olingan.

IBM doktor Kodd (E. F. Kodd) da bir tadqiqotchi samarali ish erta 70-yillarida ma'lumot modeli deb ataladi (Structured Query Language English uchun tuzilgan Ingliz tili bilan bog'liq mahsulot yaratish uchun olib tariq), 1980 yilda esa SQL (Structured query language, tuzilishi bilan nomlandi turirovanny so'rovlar tili).

O'shandan beri, IBM-dan tashqari, ko'plab ishlab chiqaruvchilar SQL uchun dasturiy mahsulotlarni ishlab chiqishda qo'shilishdi. Boshqa, shuningdek relyatsion ma'lumotlar bazalarini ishlab chiqaruvchilarga, relyatsion ma'lumotlar bazasiga kirish va ularni boshqarish uchun standartlashtirilgan usul kerak edi. IBM dastlab nazariyani ishlab chiqgan kompaniya bo'lsada DAVLAT ma'lumotlar bazasi, bu texnologiyalar bilan bozorda birinchi, Oracle chiqdi. Bir muncha vaqt o'tgach, SQL bozorda etarlicha mashhurlikka erishdi va 1986, 1989, 1992, 1999 va 2003 yillarda SQL til standartlarini chiqargan Amerika Milliy Standartlar Instituti (ANSI) e'tiborini tortdi. SQL3 standartining eng so'nggi versiyasi ob'ektga yo'naltirilgan dasturlash uchun til kengaytmalarini o'z ichiga oladi.

1986 yildan beri bir nechta raqobatlashadigan tillar dasturchilar va ishlab chiquvchilarga relyatsion ma'lumotlarga kirish va ularni boshqarish imkoniyatini berdi. Biroq, ulardan juda oz qismi o'rganish oson va SQL sifatida hamma uchun qabul qilingan. Dasturchilar va hokimlar endi o'zgarishi bilan turli uchun qo'llanilishi mumkin, bir til, o'rganish mumkin, platformalar, ma'lumotlar bazalari, dasturlar va boshqa mahsulotlar.

SQL yoki tuzilgan so'rovlar tili, eng asosiysi, aloqador ma'lumotlarni boshqarish tili hisoblanadi. Bu Amerika Milliy Standartlar Instituti (ANSI) tomonidan relyatsion ma'lumotlar bazalarini boshqarish uchun standart til sifatida tavsiya etiladi va DB2, SQL/DS, Oracle, INGRES, SYBASE, SQL Server, Windows uchun dBase, Paradox va boshqa ko'plab tijorat ma'lumotlar bazalari tomonidan ma'lumotlarga kirish tili sifatida ishlatiladi. Microsoft Access va boshqalar. Mashhurligi tufayli SQL kompyuterlar o'rtasida ma'lumot almashish uchun odatiy tilga aylandi. Deyarli har qanday kompyuter va operatsion tizimda ishlaydigan SQL versiyasi mavjudligi sababli, kompyuter tizimlari ma'lumot almashish, so'rovlar va javoblarni SQL tilida bir-birlariga uzatish imkoniyatiga ega.

SQL standartining doimiy rivojlanishi turli sotuvchilar va platformalar orasida ko'plab SQL dialektlarining paydo bo'lishiga yordam berdi. Ushbu dialektlar ANSI qo'mitasi standartni ishlab chiqishdan oldin ma'lum bir ma'lumotlar bazasi foydalanuvchilari hamjamiyati ba'zi bir imkoniyatlarga muhtoj bo'lganligi sababli rivojlandi. Biroq, ba'zida raqobatlashadigan texnologiyalarning bosimi tufayli tadqiqot jamoatchiligi tomonidan yangi funktsional imkoniyatlar paydo bo'ladi. Masalan, ko'plab ma'lumotlar bazalarini etkazib beruvchilar mavjud dasturlash imkoniyatlariga Java (DB2, Oracle va Sybase kabi) yoki VBScript-ni (Microsoft kabi) qo'shadilar. Kelajakda dasturchilar va ishlab chiquvchilar ushbu dasturlash tillaridan SQLning o'zi bilan bir qatorda SQL dasturlarida ham foydalanadilar.

Ushbu dialektlarning ko'pchiligiga shartli ishlov berish vositalari (masalan, IF ... THEN bayonoti nazorati ostida), boshqaruv operatorlari (masalan, WHILE ko'chadan), o'zgaruvchilar va xatolar bilan ishlash kiradi. ANSI hali ham bunday muhim funktsionallik uchun standartlarni ishlab chiqmaganligi sababli va foydalanuvchilar allaqachon talab qilmoqdalar, shuning uchun relyatsion MBBT ishlab chiqaruvchilari o'zlarining buyruqlari va sintaksislarini yaratishda erkin edilar. Darhaqiqat, eng qadimgi dasturchilarning ko'pchiligi 1980 yildan beri eng oddiy buyruqlar (masalan, SELECT) buyruqlarini saqlab qolishgan, chunki ularning

bajarilishi standartdan oldinroq bo'lgan. Hozirda ANSI ushbu nomuvofiqliklarni bartaraf etish uchun o'z standartlarini qayta ko'rib chiqmoqda.
Ba'zi lahjalarda dasturlash tilining yanada to'liq ishlashini ta'minlash uchun protsessual buyruqlar kiritiladi. Masalan, protsessual dasturlarda xatolar bilan ishlash bo'yicha ko'rsatmalar, dasturni bajarish yo'nalishini boshqaruvchi bayonotlar, shartli buyruqlar, o'zgaruvchilar va massivlar bilan ishlash vositalari va boshqa ko'plab qo'shimchalar mavjud. Ushbu protsessual dasturlarning barchasi tilning texnik xilma-xilligi bo'lsa-da, ular bu erda dialektlar deb nomlanadi. "Doimiy saqlangan modul" (SQL / PSM) to'plami saqlangan dastur protseduralari bilan bog'liq juda ko'p funktsiyalarni taklif etadi va bunday dialektlarda mavjud bo'lgan ko'plab kengaytmalarni o'z ichiga oladi. Ba'zi mashhur SQL dialektlari:

• **PL/SQL.** Oracle-da ishlatilgan. PL / SQL protsessual tili / SQL uchun qisqa. Bu ko'p jihatdan Ada tiliga o'xshashdir.

• **Transact-SQL.** Microsoft SQL Server va Sybase Adaptive Server tomonidan ishlatiladi. Microsoft va Sybase 1990-yillarning boshlarida ishlatilgan umumiy platformadan uzoqlashganda, ularning Transact-SQL dasturlari ham turlicha.

• **PL/pgSQL.** PostgreSQL-da amalga oshirilgan dialekt va SQL kengaytmalarining nomi. Bu protsedura tili / postgreSQL uchun qisqa.

• **SQLPL.** DB2 (SQLProcedural Language) dan yangi dialekt. Standart SQL boshqaruv bayonotlari asosida. Boshqa shevalarning aksariyati standartdan oldinroq bo'lgan, demak siz ularda SQL standartidan juda ko'p farqlarni topasiz.

8.2 SQL tili

SQL tili quyidagi afzalliklarga ega:

1. muayyan MBBTdan mustaqillik. Agar ma'lumotlar bazasini yaratishda ba'zi MBBTtomonidan taqdim etilgan SQL tilining nostandart xususiyatlari ishlatilmagan bo'lsa, unda bunday ma'lumotlar bazasini boshqa ishlab chiqaruvchining ma'lumotlar bazasida o'zgartirishsiz o'tkazish mumkin. Afsuski, ko'pgina ma'lumotlar bazalari ular ishlayotgan MBBT xususiyatlaridan foydalanadi, bu esa ularni boshqa ma'lumotlar bazasiga o'zgartirishlarsiz o'tkazishni qiyinlashtiradi;
2. munosabatlar doirasi. Relyatsion model mustahkam nazariy asosga ega. SQL relyatsion modelga asoslangan va relyatsion ma'lumotlar bazalari uchun yagona til hisoblanadi;
3. SQL ingliz tiliga o'xshash yuqori darajadagi tuzilishga ega.
4. SQL turli xil foydalanuvchilar uchun turli xil ma'lumotlarning ko'rinishini yaratishga imkon beradi;
5. SQL - to'liq ma'lumotlar bazasi tili;
6. SQL til standartlari. Rasmiy SQL standarti ANSI va ISO tomonidan 1989 yilda nashr etilgan va 1992 yilda sezilarli darajada kengaydi.

SQL- dan foydalanishni boshlash uchun ko'rsatmalar qanday yozilganligini tushunishingiz kerak. Sin taksicheskie dizayn SQL to'rt asosiy guruhga bo'linadi:

• **Identifikatorlar.** Ular ma'lumotlar bazasi, jadval, jadval cheklovi, jadval

ustunlari, ko'rinishlar kabi ma'lumotlar bazasi ob'ektlari uchun foydalanuvchi yoki tizim nomlari.

• **Doimiy.** Ular foydalanuvchi yoki tizim tomonidan yaratilgan satrlar yoki identifikator yoki kalit so'z bo'lmagan qiymatlar. Konstantalar "salom" kabi satrlar, "1234" kabi raqamlar, "2002 yil 1-yanvar" kabi sanalar yoki TRUE kabi mantiqiy so'zlar bo'lishi mumkin.

• **Operatorlar.** Bir yoki bir nechta iboralarda qanday harakat amalga oshirilishini ko'rsatadigan belgilar, ko'pincha DELETE, INSERT, SELECT yoki UPDATE gaplarida. Ma'lumotlar bazasi ob'ektlarini yaratish uchun odatda operatorlardan foydalaniladi.

• **Zaxiralangan va kalit so'zlar**. SQL kod protsessori uchun maxsus ma'nolarga ega bo'ling. Masalan, SELECT, GRANT, DELETE yoki CREATE. Zaxira qilingan so'zlar, odatda SQL buyruqlari va bayonotlari ushbu platformada identifikator sifatida ishlatilishi mumkin emas. Kalit so'zlar kelajakda saqlanib qolishi mumkin bo'lgan so'zlardir.

Identifikatorlar

Esingizda bo'lsin, RMBBT o'rnatilgan nazariya asosida qurilgan. ANSI terminologiyasida klasterlar ko'p kataloglarni, kataloglar ko'plab sxemalarni, sxemalar ko'plab moslamalarni va boshqalarni o'z ichiga oladi. Ko'pgina platformalarda qo'shimcha atamalar qo'llaniladi: misollarda bir yoki bir nechta ma'lumotlar bazasi, ma'lumotlar bazalarida bir yoki bir nechta sxemalar, sxemalar bir yoki bir nechta jadvallar, ko'rinishlar, saqlangan protseduralar va har bir ob'ekt bilan bog'liq imtiyozlar. Strukturaning har bir darajasida elementlarga dasturlar va tizim jarayonlari orqali kirish uchun noyob nomlar (ya'ni identifikatorlar) kerak. Bu shuni anglatadiki, RMBBTdagi har bir ob'ekt (ma'lumotlar bazasi, jadval, ko'rinish, ustun, indeks, kalit, trigger, saqlangan protsedura yoki cheklov) o'z identifikatoriga ega bo'lishi kerak. Agar ma'lumotlar bazasi ob'ektini yaratadigan buyruqni bajaradigan bo'lsangiz, ushbu yangi ob'ektning identifikatorini (ya'ni, ismini) taqdim etishingiz kerak.

Tajribali dasturchi ob'ekt nomini tanlashda yodda tutadigan ikkita muhim qoidalar to'plami mavjud. Konvensiyani nomlash

Bunga ma'lumotlar bazasi tuzilishini va ma'lumotlarni kuzatishni yaxshilaydigan qo'llanmalar yoki nomlash konventsiyalari kiradi. Ular SQL talablari emas, balki amaliy dasturchilarning to'plangan tajribasi. Identifikatorlarni yaratish qoidalari

Ular SQL standartida aniqlangan va platformalarda amalga oshirilgan. Ushbu qoidalar, masalan, ismning maksimal uzunligi kabi parametrlarni o'z ichiga oladi. Ushbu bitimlar keyinchalik ushbu bobda har bir ishlab chiqaruvchi uchun tavsiflangan. Aniqlovchilar

Esda tutingki, RMBBTlar o'rnatilgan nazariyaga asoslanadi. Ansi Klaster terminologiyasi kataloglar bir qancha o'z ichiga oladi, katalog mikrosxemalar o'z ichiga oladi, davrlari, ishlatiladigan eng platformalarda va boshqalar ob'ektlarini bir qancha, egiluvchanlarni o'z ichiga oladi: bir yoki bir necha ma'lumotlar bazalari nusxalari (misollar) o'z ichiga oladi, ma'lumotlar bazalari bir yoki bir necha

kontaktlarning o'z ichiga oladi, davrlari bir yoki bir nechta jadval, ko'rinishlar, saqlanadigan protseduralar va har bir ob'ekt bilan bog'liq imtiyozlarni o'z ichiga oladi. Har bir darajada tuzilishi elementlari noyob ismlar (ya'ni talab fikatory) dasturlari va tizim jarayonlari kirishingiz mumkin ularga. Bu har bir ob'ekt (u ma'lumotlar bazasi, jadval, ko'rinish, ustun, indeks, kalit, trigger, saqlanadigan protsedura yoki cheklash bo'lsin) RMBBTda o'z identifikatorini olishi kerakligini anglatadi. Agar siz ma'lumotlar bazasi ob'ekti yaratadigan buyruqni ishlatsangiz, ushbu yangi ob'ektning identifikatorini (ya'ni ismini) ko'rsatishingiz kerak.

Tajribali dasturchi ob'ekt nomini tanlashda yodda tutadigan ikkita muhim qoidalar to'plami mavjud.

Anjumanlarni nomlash

Bu amaliy qo'llanmalarni yoki nomlash bo'yicha konventsiyalarni o'z ichiga oladi, ulardan foydalanish oxirida ma'lumotlar bazasi tuzilmasi va ma'lumotlar kuzatilishini yaxshilaydi. Ular amaliy tajribalar, faqat SQL so'rovlar bor zarb, dasturchilarni.

Shaxsni yaratish qoidalari

Ular SQL standartida aniqlangan va platformalarda amalga oshirilgan. Ushbu qoidalar, masalan, ismning maksimal uzunligi kabi parametrlarni o'z ichiga oladi. Ushbu bitimlar keyinchalik ushbu bobda har bir ishlab chiqaruvchi uchun tavsiflangan.

Shartnoma haqida nomlari

• *Mazmunli, tavsiflovchi va ob'ekt maqsadiga mos keladigan ismni tanlang*. Jadvalni XRO3 deb nomlamang, yaxshisi uni 2005 yildagi xarajatlarni o'z ichiga olganligini aniqlashtirish uchun uni Xarajatlar_2005 deb nomlang. Yodingizda bo'lsin, boshqa odamlar jadval va ma'lumotlar bazasidan, ehtimol siz ketganingizdan keyin uzoq vaqt foydalanishlari kerak bo'ladi. Ismlar bir qarashda aniq bo'lishi kerak. Har bir ishlab chiqaruvchi ob'ekt nomining uzunligi bo'yicha o'z cheklovlariga ega, ammo, qoida tariqasida, ismlar har kim o'z ma'nosini tushunishi uchun etarlicha uzoq bo'lishi mumkin.

• *Ma'lumotlar bazasi bo'ylab nomlarda bir xil holatlardan foydalaning*. Barcha asosiy ob'ekt nomlari uchun katta yoki kichik harflardan foydalaning. Ba'zi ma'lumotlar bazasi serverlari katta-kichiklarga sezgir bo'lib, kichik harflar aralashmasidan foydalanish keyinchalik muammolarni keltirib chiqarishi mumkin.

• *Qisqartmalardan foydalanishda izchil bo'ling. Qisqartma tanlanganidan so'ng, ma'lumotlar bazasida doimiy ravishda ishlatilishi kerak*. Masalan, agar siz EMPLni EMPLOYEE qisqartmasi sifatida ishlatsangiz, u holda ma'lumotlar bazasida EMP dan foydalaning. Ba'zi joylarda EMP-ni ishlatmang, boshqalarda EMPLOYEEE.

• *O'qish uchun pastki chiziqlar bilan to'liq, tavsiflovchi va mazmunli ismlardan foydalaning*. UPPERCASEWITHLNDERSCORES ustun nomi UPPERCASE_WITH_UNDERSCORES kabi tavsiflovchi emas.

• *Ma'lumotlar bazasi ob'ektlari nomlariga kompaniya va mahsulot nomlarini qo'ymang*. Kompaniyalar bir-birini sotib oladi va mahsulotlar nomlarini o'zgartiradi.

Bunday elementlar vaqtinchalik bo'lib, asosiy ob'ektlarning nomlariga kiritilishi shart emas.

• *Juda aniq prefiks va qo'shimchalardan foydalanmang*. Masalan, ma'lumotlar bazasi nomi uchun "MB_" ni prefiks sifatida ishlatmang, ammo "V_" barcha ko'rinishlarga prefiks sifatida. Asosiy tizim jadvallariga nisbatan oddiy so'rovlar baz administratoriga yoki dasturchiga identifikator ko'rsatadigan ob'ekt turi to'g'risida ma'lumot berishi mumkin.

• *Ob'ekt nomi uchun berilgan barcha joylarni to'ldirmang*. Agar sizning platformangiz 32 belgidan iborat jadval nomiga ruxsat bersa, oxirida kamida bo'sh joy qoldirishga harakat qiling. Ba'zi platformalar ba'zida jadvalning vaqtinchalik nusxalarini manipulyatsiya qilishda jadval nomlariga prefiks va qo'shimchalar qo'shadi.

• *Ajratilgan identifikatorlardan foydalanmang*. Ba'zan, ob'ekt nomlari ikkita tirnoq bilan qo'shib qo'yilgan. (ANSI standarti ajratilgan identifikatorlar kabi nomlarga ishora qiladi. Belgilagichni shu tarzda keltirish, undan foydalanish qiyin bo'lishi va keyinchalik muammolarni keltirib chiqarishi mumkin bo'lgan nomlarni yaratishga imkon beradi. Bunday identifikatorlar katta-kichik harflar bilan farq qiladi. Masalan, siz ularni bo'shliqlar, maxsus belgilar, har xil holatdagi belgilar va hattoki boshqariladigan belgilar, ammo ba'zi bir uchinchi tomon vositalari (va hattoki bazaning o'zi ishlab chiqaruvchisi ham) nomdagi maxsus belgilarni ishlatmasligi mumkinligi sababli, bunday identifikatorlardan keng foydalanmaslik kerak Ba'zi platformalar ikkilangan tirnoqlardan tashqari boshqa belgilarni chegaralashga ruxsat berish, masalan, ajratilgan identifikatorlarni belgilash uchun SQL Server kvadrat qavslardan foydalanadi [].

Shaxsni yaratish qoidalari

Identifikatorlar ularning doirasi ichida noyob bo'lishi kerak. Shunday qilib, ob'ektlar ierarxiyasida ma'lumotlar bazasi nomlari ma'lumotlar bazasi serverining ma'lum bir nusxasi ichida takrorlanmasligi kerak, jadvallar, ko'rinishlar, funktsiyalar, triggerlar va saqlangan protseduralar nomlari ma'lum bir sxema bo'yicha noyob bo'lishi kerak. Boshqa tomondan, jadval va saqlangan protsedura bir xil nomga ega bo'lishi mumkin, chunki ular har xil turdagi ob'ektlardir. Ustunlar, kalitlar va indeks nomlari bitta jadvalda yoki ko'rinishda noyob bo'lishi kerak va hokazo. Qo'shimcha ma'lumot olish uchun platformaning hujjatlariga murojaat qiling. Ba'zi platformalarda identifikatorning o'ziga xosligi talab qilinadi, boshqalarda esa bunday emas. Masalan, DB2 platformasi barcha indeks identifikatorlarini butun ma'lumotlar bazasida yagona bo'lishini talab qiladi, SQL Server indeks identifikatorlarini faqat ular murojaat qilgan jadval ichida yagona bo'lishini talab qiladi.

Shuni esda tutingki, ushbu qoidalarning bir qismini ishlash uchun ajratilgan identifikatorlardan foydalanishingiz mumkin (ya'ni, maxsus ajratuvchi belgilarga kiritilgan ob'ekt nomlari, odatda ikkita tirnoq). Xususan, ajratilgan identifikatorlardan zahiralangan so'zlar bilan nom berish yoki odatda u erda nomda ishlatilmaydigan belgilarni ishlatish uchun foydalanish mumkin. Masalan, ko'pincha jadval nomidagi foiz belgisidan (%) foydalana olmaysiz. Ammo, agar kerak bo'lsa,

siz jadval nomini har doim ikkita tirnoq bilan qo'shsangiz, undan foydalanishingiz mumkin. Jadvaldagi xarajatlarni %% nisbati deb nomlash uchun nomni tirnoq belgilariga kiritishingiz kerak - "xarajatlar %% nisbati". Shuni ham yodda tutingki, SQL2003 da bunday nomlar ba'zan ajratilgan identifikatorlar deb ataladi.

Konstantalar

SQL-da konstantalar - bu har qanday raqamli qiymatlar, belgilar qatorlari, vaqtni ko'rsatish bilan bog'liq qiymatlar (sana va vaqt) va identifikator yoki kalit so'z bo'lmagan mantiqiy qiymatlardir. SQL asosidagi ma'lumotlar bazalari SQL kodida har xil doimiylardan foydalanishga imkon beradi. Ma'lumotlarning raqamli, belgi va mantiqiy turlarining ko'pi, shuningdek sanalar qabul qilinadi. Masalan, SQL Server-dagi raqamli ma'lumotlar turlariga boshqalar qatorida INTEGER, REAL va MONEY kiradi. Shunday qilib, raqamli konstantalar shunday ko'rinishi mumkin.

```
30
-17
-853 3888
-6.66
70 ming dollar
2E5
7E-3
```

Misoldan ko'rinib turibdiki, SQL Server imzolangan va imzosiz raqamlarni normal va eksponent yozuvlarda qabul qiladi. Va SQL Server valyutadagi ma'lumot turiga ega bo'lganligi sababli, hatto doimiy ravishda dollar belgisini qo'shishingiz mumkin. SQL Server raqamli konstantalarida boshqa belgilarni ishlatishga ruxsat berilmaydi (0123456789+ - $. Undan tashqari), shuning uchun vergul (yoki Evropa davrlari) ni qo'shmang. Ko'pgina ma'lumotlar bazalari vergulni raqamli konstantada element ajratuvchi sifatida izohlaydi. Shunday qilib, doimiy 3,000ni 3 va alohida 000 deb talqin qilinadi.

Mantiqiyliklar, mag'lubiyat doimiylari va sanalar quyidagicha ko'rinadi:

```
TRUE
"Salom dunyo!"
10ST-28-1966 22:14:30:00 '
```

String konstantalari har doim bitta tirnoq ichiga olinishi kerak ('), ular barcha satr konstantalari uchun standart ajratuvchi hisoblanadi. String konstantalaridagi belgilar alfavitli belgilar bilan chegaralanmaydi. Aslida belgilar majmuasidagi har qanday belgi satr konstantasi sifatida ifodalanishi mumkin. Quyidagi barcha ifodalar satr konstantalari. '1998'

```
'70,000 + 14,000'
‘Bir paytlar Nantuketlik bitta odam bor edi’
'1966 yil 28 oktyabr'
```

Keltirilgan barcha misollar aslida CHARACTER ma'lumot turiga mos keladi. '1998' qatorli konstantani 1998 raqamli doimiy bilan aralashtirib yubormang. CHARACTER ma'lumotlar turi bilan faqat mag'lubiyat konstantalari bog'langanda,

ularni raqamli turga aniq aylantirmasdan arifmetik hisob-kitoblarda ishlatmaslik kerak. Ba'zi ma'lumotlar bazalari har qanday DATE yoki NUMBER qiymatlari bilan taqqoslaganda raqamlarni o'z ichiga olgan qator konstantalarini avtomatik ravishda o'zgartiradi.

Ixtiyoriy ravishda bitta tirnoq belgisini mag'lubiyatga doimiy ravishda ko'rsatishingiz mumkin. Buning uchun uni ikki marta yozish kerak; ya'ni har safar bitta tirnoq ichida bitta tirnoq yozish kerak bo'lganda, ikkitasini yozishingiz kerak. Keling, ushbu fikrni SQL Server misolida keltiraylik.

```
SELECT 'So he said  ''who''s Le Petomaine?'' '
```

Quyidagi natija olinadi.

```
So he said 'Who's Le Petomaine?'
```

SQL tili operatorlarini yozish

Operator - bu bir yoki bir nechta iboralarda bajariladigan harakatni bildiruvchi belgi. Operatorlar eng ko'p DELETE, INSERT, SELECT yoki UPDATE bayonotlarida ishlatiladi, shuningdek, odatda ma'lumotlar bazasi ob'ektlarini, masalan, saqlangan protseduralar, funktsiyalar, triggerlar va ko'rinishlar yaratishda ishlatiladi.

Operatorlar odatda quyidagi toifalarga bo'linadi:

1. *Arifmetik operatorlar*. Barcha ma'lumotlar bazalari tomonidan qo'llab-quvvatlanadi.

2. *Topshiriq operatorlari.* Barcha ma'lumotlar bazalari tomonidan qo'llab-quvvatlanadi.

3. *Bit bitli operatorlar.* Microsoft SQL Server qo'llab-quvvatlanadi.

4. Taqqoslash operatorlari. Barcha ma'lumotlar bazalari tomonidan qo'llab-quvvatlanadi.

5. *Mantiqiy operatorlar.* DB2, Oracle, SQL Server va PostgreSQL-da qo'llab-quvvatlanadi. Unary operatorlari. DB2, Oracle va SQL Serverda qo'llab-quvvatlanadi.

Arifmetik operatorlar

Arifmetik operatorlar matematik amallarni sonli toifaga kiruvchi har qanday turdagi ikkita qiymat bo'yicha bajaradilar. Arifmetik operatorlarning ro'yxati 8.1-jadvalda keltirilgan.

8.1-jadval. Arifmetik operatorlar.

Arifmetik operator	Harakat
+	Qo'shimcha
-	Ajratish
*	Ko'paytirish
/	Bo'linish

%	Bo'linishning qolgan qismi (faqat SQL Serverda). Bo'lish operatsiyasining qolgan qismini butun son sifatida qaytaradi

Belgilanish operatorlari

Topshiriq operatorlari Operator bu maqsadda ishlatiladigan Oracle bundan mustasno: - (=) tayinlash operatori o'zgaruvchiga yoki ustun sarlavhasi taxallusiga qiymat beradi. SQL Server-da AS kalit so'zi jadvallar yoki ustunlar sarlavhalarini taxallus qilish uchun operator sifatida ishlatilishi mumkin.

Inkor operatorlar

Microsoft SQL Server bitli operatorlarni taqdim etadi, ular ikkita fayl ifodalarda bitlarni boshqarish uchun qulay usulni taqdim etadi (8.2-jadvalga qarang). Bit bitli operatorlar uchun mavjud ma'lumotlar turlari *binary, hit, int, smallint, tinyint u varbinary*.

8.2-jadval. Inkor operatorlar

Bitwise operator	Harakat
&	Inkor va (ikkita operand)
|	Inkor yoki OR (ikkita operand)
^	Bitta eksklyuziv OR (ikkita operand)

Taqqoslash operatorlari

Taqqoslash operatorlari ikkita ibora orasidagi tenglik yoki tengsizlikni sinab ko'rishadi. Taqqoslash operatsiyasining natijasi mantiqiy qiymati: TRUE, FALSE yoki UNKNOWN. Shuni ham unutmangki, ANSI standartiga muvofiq, bir yoki ikkala qiymat NULL bo'lganda ifodalarni taqqoslash NULLga olib keladi. Masalan, 23 + NULL NULL, 2003 yil 23-fevral + NULL kabi.

8.3-jadval. Taqqoslash operatorlari

Taqqoslash operatori	Harakat
=	Teng
>	Katta
<	Kichik
> =	Katta yoki teng
<=	Kichik yoki teng
<>	Teng emas
! =	Teng emas (ANSI mos emas)
! <	Kichik emas (ANSI mos emas)
!>	Kata emas (ANSI mos emas)

Mantiqiy operatorlar

Mantiqiy operatorlar odatda WHERE bandida shartning to'g'riligini tekshirish uchun ishlatiladi. Mantiqiy operatorlar mantiqiy qiymatni TRUE yoki FALSE-ga qaytaradilar. Mantiqiy operatorlar 3-bobning "SELECT Statement" bo'limida ham muhokama qilinadi. Hamma ma'lumotlar bazalari barcha operatorlarni qo'llab-quvvatlamaydi. Mantiqiy operatorlarning ro'yxati 8.4 -jadvalda keltirilgan.

8.4-jadval. Mantiqiy operatorlar

Mantiqiy operator	**Harakat**
ALL	TRUE taqqoslashlar butun majmui TRUE natija beradi, agar
AND	Agar ikkala mantiqiy iboralar ham TRUE ga teng bo'lsa, TRUE
ANY	TRUE,to'plamidir bo'lsa kamida bir taqqoslash TRUE natija beradi
BETWEEN	Agar operand doirada bo'lsa, TRUE
EXISTS	Agar pastki so'rov kamida bitta qatorni qaytarsa, TRUE
IN	Agar operand ro'yxatdagi bitta iboraga yoki subquery tomonidan qaytarilgan bir yoki bir nechta satrga teng bo'lsa, TRUE
LIKE	Agar operand andozasi bilan mos bo'lsa, TRUE
NOT	Boshqa har qanday boolean operatorining qiymatini o'zgartiradi
OR	Agar biron mantiqiy ibora TRUE bo'lsa, TRUE
SOME	Agar belgilangan taqqoslashlar haqiqiy bo'lsa, TRUE

Unar operatorlari

Unar operatorlari sonli toifaga kiruvchi har qanday turdagi bitta ifodada ishlaydi. Unary operatorlari tamsayı turlariga tatbiq etilishi mumkin, ammo ijobiy va manfiy operatorlar har qanday raqamli ma'lumotlar turiga qo'llanilishi mumkin (*8.5-jadval*).

8.5-jadval. Unary operatorlari

Unar operator	**Harakat**
+	Raqamli qiymat musbat bo'ladi
-	Raqamli qiymat manfiy bo'ladi
~	Bittadan NOT, ikkilik qo'shimchani qaytaradi (Oracle va DB2 da emas)

Operator ustuvorligi

Ba'zan operatorlar ishtirokidagi iboralar juda murakkab bo'lishi mumkin. Ifoda bir nechta operator mavjud bo'lganda, ularning bajarilish tartibi operatorlarning ustunligini belgilaydi. Ijro etish tartibi hisoblash natijalariga sezilarli ta'sir ko'rsatishi mumkin. Operatorning ustuvorlik darajasi quyida keltirilgan. Yuqori

ustuvorlik bayonoti pastki ustuvorlik bayonotidan oldin bajariladi. Operatorlar eng yuqori darajadan eng past darajaga qadar tartibda ro'yxatga olingan. () (qavs ichidagi ifodalar)

1. +, -, ~ (bitta operatorlar)
2. *, /,% (matematik operatorlar)
3. +, - (arifmetik operatorlar)
4. =,>, <,> =, <=, <>,! =,!>,! <(taqqoslash operatorlari)
5. ^ (bitly eksklyuziv OR), va ((bir tomonga va) (teskari tomonga OR)
6. NOT, AND, ALL, ANY, BETWEEN IN LIKE, OR, SOME = =

(o'rtasidagi tayinlash qiymati o'zgaruvchan)

Agar operatorlar bir xil ustuvorlikka ega bo'lsa, hisoblar chapdan o'ngga amalga oshiriladi. Bayonotlarning bajarilishining sukut bo'yicha tartibini o'zgartirish uchun ifodada qavslar ishlatiladi. Qavs ichidagi iboralar birinchi navbatda, keyin qavs ichidagi hamma narsa baholanadi.

Satrlarni ajratuvchilar alfanumerik qatorning chegaralarini bildiradi.

Tizim ajratgichlari - bu ma'lumotlar bazasi serveri uchun maxsus ma'noga ega bo'lgan belgilar to'plamidan belgilar.

Cheklovchilar - bu jarayonlarning ierarxik tartibini va ro'yxat elementlarini aniqlash uchun ishlatiladigan belgilar.

Operatorlar - bu taqqoslash operatsiyalaridagi qiymatlarni, shu jumladan, arifmetik va matematik operatsiyalar uchun odatda ishlatiladigan belgilarni aniqlash uchun foydalaniladigan ajratuvchilar. 6-Jadvalda tizim cheklovlari va SQL-da ruxsat berilgan operatorlar ro'yxati keltirilgan.

8.3 Kalit so'zlar va ajratilgan so'zlar

SQL-da maxsus ma'no va funktsiyalarga ega bo'lgan belgilar bilan bir qatorda alohida ma'noga ega bo'lgan ba'zi so'zlar va iboralar ham mavjud. SQL kalit so'zlari - bu relyatsion ma'lumotlar bazasi faoliyati bilan shu qadar chambarchas bog'liqki, ularni boshqa maqsadlarda ishlatib bo'lmaydi. Odatda, bu so'zlar SQL operatorlarida qo'llaniladi. (E'tibor bering, aksariyat platformalarda ular identifikator sifatida ishlatilishi mumkin, ammo kerak emas.) Masalan, "SELECT" zaxiralangan so'z bo'lib, jadval nomi sifatida ishlatilmasligi kerak. Boshqa tomondan, zaxira qilingan so'zlar hozirda maxsus maqsadga ega emas, ammo ular kelajakda uni egallashi mumkin. Kalit so'zlardan identifikator sifatida foydalanmaslik kerakligi, ammo imkoniyat mavjudligini ta'kidlash uchun ular SQL standartida "zaxira qilinmagan kalit so'zlar" deb nomlanadi. SQL zaxiralangan so'zlar va kalit so'zlar har doim ham SQL bayonotlarida ishlatiladigan so'zlarni anglatmaydi, ular ma'lumotlar bazasi texnologiyasi bilan ham bog'liq bo'lishi mumkin. Masalan, CASCADE so'zi quyidagi ma'lumotlar jadvalidagi harakatlarni (masalan, o'chirish yoki yangilash) "pastga", ya'ni "kaskad-uy" ga tarqaladigan bunday ma'lumotlar manipulyatsiyasini tavsiflash uchun ishlatiladi. Dasturchilar ularni identifikator sifatida ishlatmasliklari va kelajakda kelajakdagi versiyalarida hech qanday muammo bo'lmasligi uchun zaxira va kalit so'zlar tez-tez nashr etiladi.

SQL buyruqlarining asosiy toifalari:

- DDL - ma'lumotlarni aniqlash tili;
- DML - ma'lumotlar manipulyatsiyasi tili;
- DQL - so'rovlar tili;
- DCL - ma'lumotlarni boshqarish tili;
- ma'lumotlarni boshqarish guruhlari;
- operatsiyalarni boshqarish guruhlari .

Ko'pincha, ikkita til ajratiladi - ma'lumotlar bazasini sxemasini aniqlash tili (SDL - Schema Definition Language) va ma'lumotlarni manipulyatsiya qilish tili (DML - Ma'lumotlarni boshqarish tili). SDL asosan ma'lumotlar bazasining mantiqiy tuzilishini aniqlash uchun xizmat qildi, ya'ni. ma'lumotlar bazasi tuzilishi foydalanuvchilarga qanday ko'rinishda bo'lsa. DML tarkibida ma'lumotlar bilan ishlash operatorlari to'plami, ya'ni. ma'lumotlar bazasiga ma'lumotlarni kiritish, mavjud ma'lumotlarni o'chirish, o'zgartirish yoki tanlashga imkon beruvchi operatorlar.

Ma'lumotlarni aniqlash tili (DDL, ***Data Definition Language***) - bu ma'lumotlar bazalari tuzilishini tavsiflash uchun kompyuter dasturlarida ishlatiladigan kompyuter tillari turkumi.

DDL funktsiyalari jumla ichidagi birinchi so'z bilan belgilanadi (ko'pincha so'rov deb ataladi), bu deyarli har doim fe'ldir. SQLda bu fayllar - "create", "alter", "drop". Ushbu so'rovlar yoki buyruqlar ko'pincha boshqa SQL buyruqlari bilan aralashtiriladi, shuning uchun DDL alohida kompyuter tili emas.

"create" so'rovi ma'lumotlar bazasi, jadval, indeks, ko'rish yoki saqlangan protsedurani yaratish uchun ishlatiladi. O'zgartirish (alter) so'rovi ma'lumotlar bazasining mavjud ob'ektini (jadval, indeks, ko'rish yoki saqlangan protsedura) yoki ma'lumotlar bazasini o'zgartirish uchun ishlatiladi. Tushirish so'rovi ma'lumotlar bazasining mavjud ob'ektini (jadval, indeks, ko'rish yoki saqlangan protsedura) yoki ma'lumotlar bazasini o'zi uchun tashlash uchun ishlatiladi. Va nihoyat, DDL-da ma'lumotlar yaxlitligini ta'minlaydigan asosiy va chet el kalit tushunchalari mavjud. " create table ", " alter table " so'rovlariga " primary key "birlamchi kalit, "foreign key" chet el tugmachasi buyruqlari kiritilgan.

Ma'lumotlarni boshqarish tili (***DML***, ***Data Manipulation Language***) - bu ma'lumotlar bazalarida ma'lumotlarni olish, kiritish, o'chirish yoki o'zgartirish uchun kompyuter dasturlarida yoki ma'lumotlar bazasi foydalanuvchilari tomonidan ishlatiladigan kompyuter tillari oilasi.

Dastlab DML-lardan faqat kompyuter dasturlari foydalangan, ammo SQL paydo bo'lishi bilan ular odamlar tomonidan ham qo'llanila boshlangan.

DML funktsiyalari gapdagi birinchi so'z bilan belgilanadi (ko'pincha so'rov deb ataladi), bu deyarli har doim fayldir. SQLda bu fe'llar "select", "insert", "update" va "delete". Bu tilning tabiatini ma'lumotlar bazasiga majburiy bayonotlar (buyruqlar) qatoriga aylantiradi.

DML-lar ma'lumotlar bazasi sotuvchisidan va sotuvchisiga juda xilma-xil bo'lishi mumkin. ANSI tomonidan o'rnatilgan SQL standarti mavjud, ammo ma'lumotlar bazasini sotuvchilari ko'pincha tilga o'zlarining "kengaytmalarini" taklif qilishadi.

DML tillari asosan ikki turga bo'linadi:

• Protsessual DMLlar - ma'lumotlar bo'yicha harakatlarni tavsiflaydi.

• Deklaratsion DML-lar - ma'lumotlarning o'zini tavsiflaydi.

Savollar:

1. SQL so'rovlar tili qanday rivojlandi?
2. SQL tilining asosiy afzalliklari nimada?
3. SQL tillarning qaysi turkumlarini o'zida birlashtiradi?
4. Operator nima?
5. SQL da qanday doimiy turlar mavjud?
6. SQL operatorlarining qaysi guruhlarini bilasiz?
7. Identifikator nima?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar omborlari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

IX Bob. Ma’lumotlarni manipulytsiya qilish

9.1 Ma'lumotlar manipulyatsiyasi tili

Ma'lumotlar manipulyatsiyasi tili 4 ta asosiy operatorga asoslanadi:

1. SELECT - jadvallardan yozuvlarni tanlash uchun ishlatiladi;
2. INSERT - jadvalga yozuvlar qo'shish uchun ishlatiladi;
3. UPDATE - jadval yozuvlarini yangilash uchun ishlatiladi;
4. DELETE - jadvaldagi yozuvlarni o'chirish uchun ishlatiladi.

Eng sodda formaga SELECT buyrug'i shunchaki ma'lumotlar bazasidan jadvaldan ma'lumot olishni buyuradi.

INSERT operatori.

SQL-dagi barcha qatorlar INSERT modifikatsiya buyrug'i yordamida kiritiladi. Oddiy shaklda INSERT quyidagi sintaksisdan foydalanadi:

```
    INSERT INTO <table_name>    [(<kolumn_1_name>    [,
<kolumn_1_name> ...])]]
   { VALUES ( <qiymat_1> [, <qiymat_2> ...]) |
| < SELECT ifoda > };
```

Masalan, sotuvchilar jadvaliga satr kiritish uchun siz quyidagi shartlardan foydalanishingiz mumkin:

```
INSERT INTO Saleseople VALUES (1001, 'Peel', 'London',
.12);
```

DML buyruqlari hech qanday natija bermaydi, ammo sizning dasturingiz

ma'lumotlardan foydalanilganligini tasdiqlashi kerak.

Shuningdek, siz ismning qiymatini kiritmoqchi bo'lgan ustunlarni belgilashingiz mumkin. Bu ismlarni istalgan tartibda kiritishga imkon beradi. Aytaylik, siz mijozlar jadvalidagi qiymatlarni printerda chop etilgan hisobotdan olasiz, ularni shu tartibda joylashtiradi: city, cname, va cnum va soddaligi uchun siz shu tartibda qiymatlarni kiritmoqchisiz:

```
INSERT INTO Mijozlar (city, cname, cnum) VALUES
('London', 'Xonman', 2001);
```

Iltimos, reyting va snum ustunlari yo'qligini unutmang. Bu shuni anglatadiki, ushbu chiziqlar avtomatik ravishda standart qiymatga o'rnatiladi. Odatiy bo'lib, NULL yoki standart sifatida belgilangan boshqa qiymat kiritilishi mumkin. Agar cheklash ushbu ustunda NULL qiymatlardan foydalanishni taqiqlasa va ushbu ustun odatiy qilib o'rnatilmagan bo'lsa, ushbu ustun jadvalga tegishli har qanday INSERT buyrug'i uchun qiymat bilan ta'minlanishi kerak.

UPDATE operatori

Endi, mavjud qatordagi ba'zi yoki barcha qiymatlarni qanday o'zgartirishni o'rganishingiz kerak. Bu UPDATE buyrug'i bilan amalga oshiriladi. Ushbu buyruqda ishlatilishi kerak bo'lgan jadvalni ko'rsatadigan UPDATE va ma'lum bir ustun uchun qilinishi kerak bo'lgan o'zgarishlarni ko'rsatadigan SET yozuvi mavjud. Masalan, barcha mijozlarning reytinglarini 200 ga o'zgartirish uchun siz kiritishingiz mumkin

```
UPDATE TABLE <table_name>
   SET <ustun nomi_1> = <qiymat #> [, <ustun nomi_2> =
<qiymat => ...]
   [WHERE  <shart>];
```

Masalan

```
UPDATE Mijozlar
   SET reytingi = 200;
```

Albatta, har doim bitta qiymatni o'zgartirish uchun jadvalning barcha satrlarini ko'rsatishni xohlamaysiz, shuning uchun UPDATE predikatlarni olishi mumkin. Masalan, Peel sotuvchisining barcha xaridorlari uchun bir xil o'zgarishlarni amalga oshirishingiz mumkin (snum = 1001 ga ega):

```
UPDATE Mijozlar
   SET reytingi = 200
   WHERE snum = 1001;
```

Biroq, UPDATE buyrug'i yordamida bitta ustunni o'zgartirish bilan cheklanib qolmaslik kerak. SET buyrug'i vergul bilan ajratilgan har qanday ustunlarni tayinlashi mumkin. Belgilangan barcha topshiriqlar har qanday jadval satriga bajarilishi mumkin, ammo bittadan. Aytaylik, Motika sotuvchisi nafaqaga chiqadi va biz uning raqamini yangi sotuvchiga tayinlashni xohlaymiz:

```
UPDATE SAVDOSI
```

```
   SET sname = 'Gibson', shahar = 'Boston', comm = .10
   WHERE snum = 1004;
```

UPDATE buyrug'ining SET gapida siz skalyar ifodalarni ishlatishingiz mumkin, shu bilan birga uni ifodaga jinsi o'zgaradi. Bu ularning iboralarni ishlatib bo'lmaydigan INSERT buyrug'ining VALUES so'zlaridan farqi; skalyar iboralarning bu xususiyati juda foydali xususiyatdir. Aytaylik, barcha sotuvchilarga ikki baravar komissiya to'lashga qaror qildingiz. Siz quyidagi iborani ishlatishingiz mumkin:

```
UPDATE SAVDOSI
   SET comm = comm * 2;
```

DELETE operatori.

DELETE - o'zgartirish buyrug'i bilan jadvaldan satrlarni o'chirishingiz mumkin. Shaxsiy maydon qiymatlarini emas, balki faqat kiritilgan satrlarni o'chirib tashlashi mumkin.

```
DELETE FROM table [WHERE <search_condition>];
```

Endi jadval bo'sh bo'lsa, uni DROP TABLE buyrug'i bilan butunlay yo'q qilish mumkin. Odatda, siz jadvaldan faqat ma'lum bir qatorlarni yo'q qilishingiz kerak. Qaysi qatorlar o'chirilishini aniqlash uchun siz so'rovlarni bajarganingiz kabi predikatdan foydalanasiz . Masalan, 1003 raqami bilan sotuvchini jadvaldan olib tashlash uchun siz kiritishingiz mumkin.

```
DELETE FROM Salespeople
   WHERE snum = 1003;
```

Biz sname maydonining o'rniga snum maydonidan foydalandik, chunki bitta va bitta satr ochilishini xohlasangiz, bu asosiy tugmachalarni ishlatishda eng yaxshi taktika. Siz uchun bu birlamchi kalitning ishlashiga o'xshaydi. Albatta, siz ushbu misolda ko'rsatilgandek, qatorlar guruhini tanlaydigan predikt bilan DELETE-dan foydalanishingiz mumkin:

```
DELETE FROM Salespeople
   WHERE city = 'London';
```

SELECT operatori.

SELECT - SQL-dagi barcha so'rovlar bitta buyruqdan iborat. Ushbu buyruqning tuzilishi aldamchi tarzda sodda, chunki siz uni murakkab baholash va ma'lumotlarni qayta ishlash uchun kengaytirishingiz kerak. Bu buyruq chaqiriladi

```
SELECT [[ ALL ] | DISTINCT ] {*
| SELECT elementi [, SELECT element ] ...}
FROM {base_table | ko'rish}
[taxalluslar]
                              [, {base_table | ko'rish}
[taxalluslar]] ...
[ WHERE ibora]
[GROUP BY iborasi [HAVING so'zlari]];
[ ORDER BY BERADI ]
```

Agar jadvalning har bir ustunini ko'rishni istasangiz, siz foydalanishingiz mumkin bo'lgan ixtiyoriy qisqartirish mavjud. Yulduzcha (*) yordamida ustunlarning to'liq ro'yxatini quyidagicha ko'rsatish mumkin:

```
SELECT * FROM sotuvchilar;
```

9.2 SQL agregat funktsiyalari va guruhlash

Umumiy funktsiyalar muayyan qatorlar qatori uchun ma'lum bir qiymatni hisoblash uchun mo'ljallangan. Agar yig'indisi funktsiyasi guruhlangan jadvalga yoki butun jadvalga qo'llanilsa, bunday qatorlar qatorlar qatori bo'lishi mumkin . SQL-da har xil turdagi ma'lumotlarga ega bo'lishga imkon beruvchi beshta agregat funktsiyalari mavjud. Ushbu funktsiyalarning sintaksisi quyida tavsiflanadi:

1. COUNT - belgilangan ustundagi qiymatlar sonini qaytaradi;
2. SUM - ko'rsatilgan ustundagi qiymatlarning yig'indisini qaytaradi;
3. AVG - belgilangan ustundagi o'rtacha qiymatni qaytaradi;
4. MIN - belgilangan ustundagi minimal qiymatni qaytaradi;
5. MAX - belgilangan ustundagi maksimal qiymatni qaytaradi.

Umumlashtiruvchi barcha funktsiyalar uchun, COUNT (*) bundan mustasno, amalda (ya'ni semantika tomonidan talab qilinadi) hisoblash tartibi quyidagicha: agregat funktsiyasining parametrlari asosida berilgan qatorlar to'plamidan qiymatlar ro'yxati tuziladi. Keyin, ushbu qiymatlar ro'yxatidan foydalanib, funktsiya hisoblab chiqiladi. Agar ro'yxat bo'sh bo'lsa, u holda COUNT funktsiyasining qiymati 0 va boshqa barcha funktsiyalarning qiymati nolga teng.

Umumiy funktsiyalar so'rovning SELECT gapida maydon nomlari kabi ishlatiladi, ammo bitta istisnosiz, ular dalil nomlarini argument sifatida oladi. SUM va AVG bilan faqat raqamli maydonlardan foydalanish mumkin. COUNT, MAX va MIN bilan raqamli yoki belgilar maydonlaridan foydalanish mumkin. Belgilar maydonlarida ishlatilganda, MAX va MIN ularni ASCII ekvivalentiga aylantiradi, bu esa MIN birinchi alfavit tartibida va MAX oxirgi qiymatni bildiradi.

Buyurtmalar jadvalidagi barcha xaridlarimizning SUM-ni topish uchun 9.1-rasmda quyidagi so'rovni kiritishimiz mumkin:

```
SELECT SUM ((amt))
FROM Orders;
```

```
================  SQL ijro jurnali =============
                        | |
                      | ------- |
                      | 26658.4 |
                        | |
================================================
```

Shuni ta'kidlash kerakki, umumiy funktsiyalar faqat SELECT ro'yxatida va HAVING bandida ishlatilishi mumkin. Boshqa barcha holatlarda ushbu funktsiyalardan foydalanish yaroqsiz. Agar SELECT ro'yxati yig'ish funktsiyasini o'z ichiga olgan bo'lsa va so'rov matnida ma'lumotlarni guruhlarga birlashtirishni ta'minlaydigan GROUP BY bandi bo'lmasa, u holda SELECT ro'yxatining biron bir

elementi ustunlar havolasini o'z ichiga olmaydi, faqat ushbu ustun bo'lgan holat bundan mustasno. Yig'ish funktsiyasining parametri sifatida ishlatiladi. Masalan, quyidagi so'rov noto'g'ri:

```
SELECT staffNo, COUNT(salary) FROM Staff;
```

Xato shundaki, ushbu so'rovda GROUP BY bandi mavjud emas, va SELECT tanlash ro'yxatidagi staffNo ustuniga yig'ish funktsiyasidan foydalanmasdan kirish mumkin.

Misol :

```
SELECT MIN (ish haqi) AS min, MAX (ish haqi) AS max,
AVG (ish haqi) AS o'rtacha hisobidan;
    ================ SQL ijro jurnali =============
                          | |
                | min | max | o‘rtacha |
                 | 9000 30.000 17,000 |
                          | |
  ================================================
```

Guruhlangan so'rovlar (GROUP by operatori)

Yuqoridagi xulosaviy ma'lumotlarning misollari, odatda hisobotlarning oxirida topilgan sarhisob qatorlariga o'xshaydi. Umumiy hisobotda hisobotning barcha batafsil ma'lumotlari bitta xulosa qatoriga siqilgan. Biroq, ko'pincha hisobotlarda subtallarni shakllantirish talab qilinadi. Shu maqsadda, SELECT bayonotida GROUP BY bandi ko'rsatilishi mumkin. GROUP BY bandini o'z ichiga olgan so'rov guruhlash so'rovi deb ataladi, chunki u SELECT operatsiyasi natijasida hosil bo'lgan ma'lumotlarni guruhlaydi va keyin har bir alohida guruh uchun bitta xulosa qatorini yaratadi. GROUP BY bandida keltirilgan ustunlar guruhlash ustunlari deb ataladi. ISO standarti SELECT va GROUP BY bandlarini chambarchas bog'liqligini talab qiladi. SELECT bayonotida GROUP BY bandidan foydalanilganda, SELECT ro'yxatidagi har bir ro'yxat elementi butun guruh uchun bitta qiymatga ega bo'lishi kerak. Bundan tashqari, SELECT bandi faqat quyidagi element turlarini o'z ichiga olishi mumkin:

• ustunlar nomlari;

• funktsiyalarni yig'ish;

• doimiylik;

• yuqoridagi elementlarning kombinatsiyalarini o'z ichiga olgan iboralar.

SELECT ro'yxatida keltirilgan barcha ustun nomlari, GROUP BY bandida ham ko'rinishi kerak, agar ustun nomi faqat agregat funktsiyasida ishlatilmasa.

Qarama-qarshi bayonot har doim ham to'g'ri kelmaydi - GROUP BY bandida SELECT ro'yxatida bo'lmagan ustun nomlari bo'lishi mumkin. Agar WHERE bandi GROUP BY bandi bilan birgalikda ishlatilsa, u holda avval u qayta ishlanadi va faqat qidirish shartini qondiradigan qatorlar guruhlanadi. ISO standarti shuni ko'rsatadiki, guruhlash amalga oshirilganda barcha etishmayotgan qiymatlar teng deb

hisoblanadi. Agar bir xil guruhlash ustunidagi ikkita jadval satrida nol qiymatlar va boshqa barcha bo'sh bo'lmagan guruhlangan ustunlarda bir xil qiymatlar bo'lsa, ular bir xil guruhga joylashtiriladi.

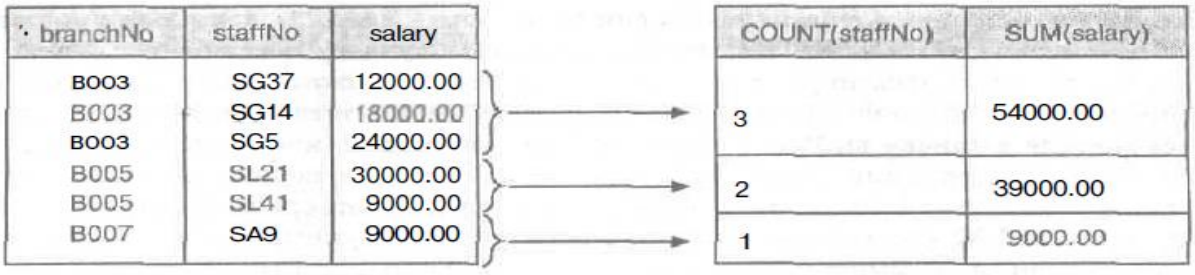
Masalan::

```
SELECT branchNo, COUNT{staffNo} AS count, SUM(salary)
AS sum FROM Staff GROUP BY branchNo ORDER BY branchNo;
```

```
================ SQL ijro jurnali =============
                          | |
              | branchNo | count | sum |
                 |    B 003 3 54000 |
                 |    B 005 2 39000 |
                  |    B 007 1 9000 |
                          | |
================================================
```

Kontseptual ravishda, ushbu so'rovni ko'rib chiqishda quyidagi harakatlar amalga oshiriladi.

1. Xodimlar jadvalidagi satrlar kompaniyaning filial raqami ustunidagi qiymatlarga ko'ra guruhlarga bo'linadi. Guruhlarning har birida kompaniyaning bir filialining barcha xodimlari to'g'risida ma'lumotlar mavjud. 9.2-rasmda ko'rsatilgandek bizning misolimizda uchta guruh yaratiladi.
2. Har bir guruh uchun, satrlarning umumiy soni bo'limdagi xodimlar soniga, shuningdek, bizni qiziqtirgan bo'limning barcha xodimlarining ish haqi miqdoriga teng bo'lgan ish haqi ustunidagi qiymatlarning yig'indisiga teng ravishda hisoblab chiqiladi. Keyin manba qatorlarining butun guruhi uchun bitta xulosa qatori yaratiladi.
3. Olingan jadvalning olingan qatorlari branchNo ustunida ko'rsatilgan filial raqamining ko'tarilish tartibida tartiblangan.



9.2- rasm. So'rov bo'yicha uchta yozuv guruhi yaratilgan.

HAVING operatori.

Aytaylik, oldingi misolda siz faqat $ 3000.00 dan yuqori bo'lgan sotib olishning maksimal miqdorini ko'rmoqchisiz. Umumlashtiruvchi funktsiyani qayerda gaplashayotganligidan foydalana olmaysiz (chunki keyinroq tasvirlangan pastki so'rovni ishlatmasangiz), chunki predikatlar bitta satr nuqtai nazaridan

baholanadi va jamlama funktsiyalar qatorlar nuqtai nazaridan baholanadi. Bu siz quyidagicha hech narsa qilolmaysiz degan ma'noni anglatadi:

```
SELECT snum, odate, MAX (amt)
FROM Oreders
WHERE MAX ((amt)) > 3000.00
GROUP BY snum, odate;
```

Bu qat'iy ANSI talqinidan voz kechish bo'ladi. Sotib olishning maksimal narxini $ 3000.00 dan yuqori narxda ko'rish uchun siz HAVING taklifidan foydalanishingiz mumkin. HAVING bandi ma'lum qatorlarni chiqindidan olib tashlash uchun ishlatiladigan mezonlarni belgilaydi, xuddi WHERE bandi alohida qatorlar uchun. To'g'ri buyruq quyidagicha bo'ladi:

```
SELECT snum, odate, MAX(amt)
FROM Orders
GROUP BY snum, odate
HAVING MAX ((amt)) > 3000.00;
```

```
    ================ SQL ijro jurnali ===============
                            | |
                 | snum odate MAX (amt) |
              | ------ ---------- -------- |
                | 1001 10/05/1990 4723.00 |
                | 1001 10.06.1990 9891.88 |
                | 1002 03/10/1990 5160.45 |
                            | |
   ================================================
```

HAVING bandidagi argumentlar SELECT bandidagi kabi qoidalarga amal qiladi, bular GROUP BY yordamida buyruqlardan iborat. Ular har bir chiqish guruhi uchun bitta qiymatga ega bo'lishi kerak. Quyidagi buyruq taqiqlanadi:

```
SELECT, MAX (amt)
FROM buyurtmalar
Snum by GROUP
HAVING odate = 10/03/1988;
```

Sana maydonini HAVING bandi bilan chaqirish mumkin emas, chunki u har bir ekran guruhi uchun bir nechta qiymatga ega bo'lishi mumkin (va). Bunday vaziyatdan qochish uchun HAVING bandi faqat GROUP BY tomonidan tanlangan agregatlar va maydonlarga tegishli bo'lishi kerak. Yuqoridagi so'rovni bajarishning to'g'ri usuli mavjud:

```
SELECT snum, MAX (amt)
FROM Orders
WHERE odate = 10/03/1990
GROUP BY snum;
```

```
    ================ SQL ijro jurnali ===============
```

```
                    | |
                  | snum |
               | ------ -------- |
                 | 1001 767.19 |
                | 1002 5160.45 |
                | 1014 1900.10 |
                | 1007 1098.16 |
                    | |
==================================================
```

Yuqorida aytib o'tilganidek, HAVING faqat bitta chiqish guruhi uchun bir xil qiymatga ega bo'lgan argumentlardan foydalanishi mumkin. Amalda, agregat funktsiyalariga havolalar eng keng tarqalgan, ammo GROUP BY bilan tanlangan maydonlar ham amal qiladi. Masalan, Serres va Rifkin uchun eng katta buyurtmalarni ko'rishni istaymiz:

```
SELECT snum, MAX (amt)
FROM Orders
GROUP BY snum
HAVING snum IN (1002,1007);
```

```
================  SQL ijro jurnali ===============
                    | |
                  | snum |
               | ------ -------- |
                | 1002 5160.45 |
                | 1007 1098.16 |
                    | |
==================================================
```

SQL ikki yoki undan ortiq ustunlar asosida so'rov natijalarini guruhlash imkonini beradi. Masalan, talabalarning familiyalari va ularning har bir semestr uchun o'rtacha baholari ro'yxatini oling.

```
SELECT StName, Semester, AVG(Mark)
FROM Marks INNER JOIN Students USING(StNo)
GROUP BY StName, Semester;
```

9.3 Guruhlangan so'rovlar bo'yicha cheklovlar

Guruhlardan foydalanadigan so'rovlar qo'shimcha cheklovlarga duch keladi. Guruhlangan ustunlar FROM gapida keltirilgan jadvallarning haqiqiy ustunlari bo'lishi kerak. Hisoblangan ifoda qiymatiga qarab satrlarni guruhlash mumkin emas.

Bundan tashqari, qaytarilgan ustunlar ro'yxatidagi elementlarga cheklovlar mavjud. Ushbu ro'yxatdagi barcha elementlar har bir satr guruhi uchun bir xil qiymatga ega bo'lishi kerak. Bu qaytib kelgan ustun bo'lishi mumkinligini anglatadi:

1. doimiy

2. guruhdagi barcha satrlar uchun bitta qiymatni qaytaradigan agregat funktsiyasi;
3. ta'rifiga ko'ra guruhning barcha qatorlarida bir xil qiymatga ega bo'lgan guruhlash ustuni;
4. yuqoridagi elementlarni o'z ichiga olgan ibora.

Amalda, guruhlash ustuni va agregat funktsiyasi har doim so'rovning guruhlarga ajratilgan ustunlari ro'yxatiga kiritilgan. Agar ikkinchisi ko'rsatilmagan bo'lsa, so'rovni DISTINCT kalit so'zidan foydalanib GROUP BY so'zidan foydalanmasdan osonroq ifodalash mumkin. Aksincha, agar siz so'rov natijalariga guruhlash ustunini qo'shmasangiz, natijalarning har bir qatoriga qaysi guruhga tegishli ekanligini aniqlay olmaysiz.

ANSI SQL-ning qat'iy talqinida siz agregat agregatidan foydalana olmaysiz. Aytaylikmi? Qaysi kuni eng katta sotib olish miqdorini bilishni xohlaysiz?

```
SELECT odati, MAX (SUM (amt))
FROM buyurtmalar
GRADA BY odate;
```

Agar siz buni qilishga harakat qilsangiz, ehtimol sizning jamoangiz rad etiladi. (Ba'zi bir amaliyotlar ushbu cheklovni belgilamaydilar, bu foydali, chunki joylashtirilgan agregatlar juda foydali bo'lishi mumkin, garchi ular biroz muammoli bo'lsa ham). Yuqorida aytib o'tilgan buyruqda, masalan, SUM odat maydonining har bir guruhiga, MAX esa barcha guruhlarga, barcha guruhlar uchun yagona qiymat ishlab chiqarish. Ammo GROUP BY bandi odat maydonining har bir guruhi uchun bitta chiqish satri bo'lishi kerakligini anglatadi.

Jadvallarga qo'shilish.

SQL-92 standarti tashqi qo'shilishni qo'llab-quvvatlash uchun mutlaqo yangi usulni aniqladi, bu esa hech qanday mashhur DBMS-larga ishonmaydi. Standart spetsifikatsiyada tashqi qo'shilishlar foydalanuvchiga manba jadvallarini so'rovga qanday qo'shilishini aniq ko'rsatishga imkon beradigan batafsil sintaksis bilan FROM bandida qo'llab-quvvatlandi.

Birlashtirish operatsiyasi SQL-da bir nechta jadvallarda saqlangan tegishli ma'lumotlarni bitta so'rovda ko'rsatish uchun ishlatiladi. Bu SQL so'rovlarining eng muhim xususiyatlaridan biri - bir nechta jadvallar o'rtasidagi munosabatlarni aniqlash va shu aloqalar ichida ulardan ma'lumotlarni ko'rsatish qobiliyati. Aynan shu operatsiya SQL tilini moslashuvchan va engil qiladi.

Birlashtirish operatsiyalari ikki turga bo'linadi - ichki va tashqi. Har ikkala qo'shilish turi SELECT so'rovining WHERE bandida maxsus qo'shilish sharti yordamida ko'rsatilgan. Tashqi birikmalar (bu haqda keyinroq gaplashamiz) ANSI-92 standarti tomonidan qo'llab-quvvatlanadi va "JOIN" so'zini o'z ichiga oladi, ichki qo'shilishlar (yoki shunchaki qo'shilishlar) bunday so'zni ishlatmasdan ko'rsatilishi mumkin (ANSI-89 standartida)) yoki "JOIN" so'zidan foydalangan holda (ANSI-92 standartida).

Jadvallarni birlashtirishda bog'lash, qoida tariqasida, bitta jadvalning asosiy kaliti va boshqa jadvalning tashqi kaliti bilan amalga oshiriladi - har bir juft jadval uchun. Bunday holda, tashqi kalitning barcha maydonlarini hisobga olish juda

muhim, aks holda natija buziladi. Birlashtiriladigan maydonlar tanlanishi mumkin bo'lgan narsalar ro'yxatida bo'lishi mumkin (lekin shart emas!). WHERE bandida bir nechta qo'shilish shartlari bo'lishi mumkin. Qo'shilish sharti WHERE bandidagi boshqa predikatlar bilan ham birlashtirilishi mumkin.

Birlashtirish - bu kartezyen mahsuloti deb nomlangan ikkita jadval ma'lumotlarining umumiy kombinatsiyasining pastki qismi. Ikki jadvalning dekartlik mahsuloti - bu ikkala jadvalning bir qismi bo'lgan barcha mumkin bo'lgan juft qatorlardan tashkil topgan yana bir jadval. Olingan jadvaldagi ustunlar to'plami birinchi jadvaldagi barcha ustunlardan keyin ikkinchi jadvaldagi barcha ustunlardir. Agar siz "WHERE" bandini ko'rsatmasdan ikkita jadvalga qarshi so'rov kiritsangiz, SQL muhitida so'rovni bajarish natijasi ushbu jadvallarning dekart natijasi bo'ladi.

SELECT operatori yordamida ikkita jadvalni birlashtirish natijalarini o'z ichiga olgan jadvalni yaratish tartibi quyidagicha. Kartezian mahsuloti bo'ladi.

1. FROM bandida ko'rsatilgan jadvallarning dekartiy mahsuloti hosil bo'ladi.

2. Agar so'rovda WHERE bandi bo'lsa, dekartlar mahsuloti jadvalining har bir qatoriga qidiruv shartlarini qo'llang va jadvalda faqat belgilangan shartlarga javob beradigan qatorlarni saqlang. Relyatsion algebra nuqtai nazaridan ushbu operatsiya dekart mahsulotining cheklanishi deb ataladi.

3. Qolgan har bir qator uchun SELECT ro'yxatida ko'rsatilgan har bir elementning qiymati aniqlanadi, natijada olingan jadvalning alohida qatori hosil bo'ladi.

4. Agar asl so'rovda SELECT DISTINCT bandi mavjud bo'lsa, natijada olingan jadvaldan barcha takrorlangan qatorlar o'chiriladi. Relyatsion algebrada 3 va 4 bosqichlar SELECT ro'yxatida ko'rsatilgan ustunlar proektsiyasiga tengdir.

5. Agar bajarilayotgan so'rovda ORDER BY bandi bo'lsa, natijada jadvaldagi qatorlar qayta tartiblanadi.

Ichki SQL-92 standartiga qo'shilish

Ichki qo'shilish faqat birlashtirish sharti to'g'ri bo'lgan satrlarni qaytaradi.

```
SELECT * | <ustunli nomlar> FROM
<table_name_1> INNER JOIN <table_name_2> ON <ustun_name
in_1_table> = <ustun_name in_2_table>
WHERE <shart>
ORDER BY <ustun "nomlari>;
```

yoki

```
SELECT * | <ustunli nomlar> FROM
<table_name_1>, <table_name_2> WHERE
<ustunli_name_in_1_table> = <ustun_name_in_2_table>
ORDER BY <ustun "nomlari>;
```

Birinchi holda, ikkita jadval JOIN operatsiyasi yordamida aniq birlashtirilgan va qo'shilishni tavsiflovchi qidiruv sharti endi FROM bandidagi ON bandida. ON kalit so'zidan keyin qidiruv sharoitida birlashtirilgan ikkita jadval qatorlarini taqqoslash uchun har qanday mezon ko'rsatilishi mumkin. Ikkinchi holda, qo'shilish sharti WHERE kalit so'zidan keyin predikat bilan birga keladi.

9.1-jadval. Filial

filiali Yo'q	b shahr	Shahar ID
B004	Bristol	1
B003	Glazgoda	3
B002	London	2

Jadval 9. 2 . PropertyForRen

mulkNo	b shahr	Shahar ID
PA14	Aberdin	4
PL94	London	2
PG4	Glazgoda	3

Ushbu jadvallarning normal (ichki) qo'shilishi quyidagi SQL so'zi yordamida amalga oshiriladi:

```
SELECT b.*, p.*
FROM filiali b, PropertyForRent p
WHERE b. shahr = p. Shahar ID;
```

Yoki SQL -92 standartiga muvofiq, ushbu so'rov quyidagicha bo'ladi:

```
SELECT b.*, p.*FROM  B filialidan IN INER JOIN
PropertyForRent b ON  b. shahr = p. Shahar;
```

```
          ====================== SQL ijro jurnali
                ========================
                                | |
  | Filial yo'q shahar shahri mulki mulkiy emas shahar
                          shahri ID |
 | -------- ------- ------ ---------- ------- ------ |
            | B003 Glasgow 3 PG4 Glasgow 3 |
             | B002 London 2 PL94 London 2 |
                                | |
 ========================================================
                              ==============
```

Ko'rib turganingizdek, natijada olingan so'rovlar jadvalida ikkala jadvaldan tanlangan shahar nomlarini o'z ichiga olgan atigi ikkita satr mavjud. Iltimos, e'tibor bering, ma'lumotlarda kompaniyaning Bristol shahridagi filiali va Aberdin shahridagi ijaraga olingan mulk uchun yozishmalar mavjud emas. Misoldan ko'rinib turibdiki, ulanish nafaqat asosiy va ikkilamchi kalitlarni o'z ichiga olgan maydonlarda amalga oshirilishi mumkin.

Tegishli ustunlar bir xil nomga ega bo'lgan ikkita jadvalning birlashishi ***tabiiy birikma*** deb nomlanadi , chunki bu odatda ikkita jadvalga qo'shilishning eng "tabiiy" usuli hisoblanadi.

SELECT * | <ustunli nomlar> FROM

```
<table_name_1> NATURAL JOIN <table_name_2> ON (<ustun
nomi>);
```

Masalan, Filial va PropertyForRent jadvallarining tabiiy birlashishi quyidagicha:

```
SELECT b.*, p.*FROM filialidan b NATURAL JOIN
mulkForRent b
ON ( shaharID );
```

SQL-92 standartida tashqi qo'shilish.

Tashqi birlashmada, qo'shilish shartini qondirmaydigan qatorlar ham natijada keltirilgan jadvalga joylashtiriladi. Tashqi ulanish operatsiyalarini bajarish xususiyatlarini tushunish uchun biz soddalashtirilgan Branch va Property For Rent jadvallaridan foydalanamiz, ularning tarkiblari yuqorida keltirilgan.

Tashqi qo'shilishning uch turi mavjud:

1. chap;

2. o'ng;

3. to'liq.

Ushbu ikkita jadvalning chap tashqi birikmasidan foydalanamiz, bu quyidagicha:

```
SELECT b.*, p.*
FROM B Filial b LEFT JOIN PropertyForRent p ON b. shahr
= p. Shahar;
```

```
      ====================== SQL bajarilish jurnali
                  ========================
                              | |
  | Filial yo'q shahar shahri mulki mulkiy emas shahar
                         shahri ID |
  | -------- ------- ------ ---------- ------- ------ |
             | B003 Glasgow 3 PG4 Glasgow 3 |
            | B004 Bristol 1 NULL NULL NULL |
|    B 002     London       2         PL 94     London
                               2 |
                             | |
 ==================================================
                        ==============
```

Ushbu misolda, chap tashqi birlashma ishlatilganligi sababli, nafaqat shaharlarning nomlari o'rtasidagi yozishmalar mavjud bo'lgan ikkita qatorni, balki ikkinchi jadvalda (o'ngda) bir-biriga mos kelmaydigan (chapda) birinchisining birinchi qatori natijaviy jadvalga kirdi. Ushbu satrda ikkinchi jadvalning barcha maydonchalari NULL qiymatlari bilan to'ldiriladi.

Ushbu ikkita jadvalning tashqi birikmasidan foydalanamiz, bu quyidagicha:

```
SELECT b.*, p.*FROM Filial b RIGHT JOIN PropertyForRent
```

```
p ON b. shahr = p. Shahar;
     ====================== SQL bajarilish jurnali
                ========================
                              | |
  | Filial yo'q shahar shahri mulki mulkiy emas shahar
                           shahri ID |
  | -------- ------- ------ ---------- ------- ------ |
            | NULL NULL NULL PA14 Aberdeen 4 |
             | B003 Glasgow 3 PG4 Glasgow 3 |
             | B002 London 2 PL94 London 2 |
                              | |
 ====================================================
                          ==============
```

Ushbu misolda, o'ng tashqi ulashni amalga oshirayotganda, natijalar jadvali nafaqat shahar nomi bilan mos keladigan ustunlarda bir xil qiymatga ega bo'lgan ikkita qatorni, balki birinchi (o'ng) jadvalning birinchi satrlari bilan, (chapda) jadval mos kelmaydigan satrlarni ham o'z ichiga oladi. Ushbu qatorda birinchi jadvalning barcha maydonlariga NULL qiymatlar berilgan .

Ushbu jadvallarning tashqi ko'rinishini quyidagicha ko'ramiz:

```
SELECT b.*, p.*FROM Filial b FULL JOIN PropertyForRent
p ON b.bCity = p.pCity;
     ====================== SQL bajarilish jurnali
                ========================
                              | |
  | Filial yo'q shahar shahri mulki mulkiy emas shahar
                           shahri ID |
  | -------- ------- ------ ---------- ------- ------ |
            | NULL NULL NULL PA14 Aberdeen 4 |
             | B003 Glasgow 3 PG4 Glasgow 3 |
             | B004 Bristol 1 NULL NULL NULL |
|    B 002     London       2         PL 94     London
                               2 |
                              | |
 ====================================================
                          ==============
```

To'liq tashqi birlashma bo'lsa, natijada jadvalda shahar nomi bilan taqqoslanadigan ustunlarda bir xil qiymatga ega bo'lgan ikkita satr emas, balki mos kelmagan asl jadvallarning qolgan barcha qatorlari joylashtiriladi . Ushbu satrlarda, mos kelmagan jadvalning barcha ustunlari NULL qiymatlari bilan to'ldirilgan.

SQL-92-dagi o'z-o'zini ulash.

Ba'zi vazifalarda maxsus usulda tanlangan ma'lumotni faqat bitta jadvaldan olish kerak.

Buning uchun ***o'z - o'zidan birikmalar*** yoki ***refleksli birikmalar***

ishlatiladi. Bu alohida qo'shilish turi emas, balki taxalluslardan foydalangan holda jadvalning o'zi qo'shilishi. Ortiqcha u shu jadval o'xshash elementlar juft olish zarur bo'lsa, aralashmalar hollarda foydalidir.

9.3-jadval. Department.

DEPT_NO	DEPARTMENT	BUDGET
1	Software Development	400000.00
2	Field Office: Canada	500000.00
3	Finance	400000.00
4	Field Office: East Coast	500000.00
5	Field Office: Japan	500000.00
6	Field Office: Singapore	300000.00
7	Field Office: Switzerland	500000.00
8	Quality Assurance	300000.00

Masalan, siz bir xil yillik byudjetga ega bo'lgan juftliklarning ro'yxatini olishni xohlaysiz:

```
SELECT d1.department, d2.department, d1.budget
FROM department d1, department d2
WHERE d1.budget = d2.budget;
```

```
        ====================== SQL Ijro jurnali
             ============================
                          | |
        |  DEPARTMENT                    DEPARTMENT
                     BUDGET     |
        |  ----------                    ----------
                     ---------  |
          |  Software Development        Finance
                     400000.00  |
   |  Field Office: East Coast   Field Office: Canada
                     500000.00  |
 |  Field Office: Japan         Field Office: East Coast
                     500000.00  |
   |  Field Office: Japan        Field Office: Canada
                     500000.00  |
 |  Field Office: Japan         Field Office: Switzerland
                     500000.00  |
    |  Field Office: Singapore    Quality Assurance
                     300000.00  |
 |  Field Office: Switzerland  Field Office: East Coast
                     500000.00  |
```

|
|
==========

Ichki so'rovlar

SQL yordamida so'rovlarni bir biri ichida joylashtirasiz. Odatda, ichki so'rov uning haqiqiy yoki yo'qligini aniqlaydigan tashqi so'rov predikatida tekshirilgan qiymatni keltirib chiqaradi. Ichki so'rovlar ko'pincha *pastki so'rovlar* deb nomlanadi.

9.4- jadval. Salespeople (sotuvchilar).

SNUM	SNAME	CITY	COMM
1001	Peel	London	0,12
1002	Serres	San-Xose	0,13
1004	Motika	London	0,11
1007	Rifkin	Barselona	0,15
1003	Axelrod	Nyu-York	0,10

9.5- Jadval. Customers (Qabul qiluvchilarni).

CNUM	NAME	CITY	RATING	SNUM
2001	Xofman	London	100	1001
2002	Govanni	Rim	200	1003
2003	Liu	Sanjose	200	1002
2004	Grass	Berlin	300	1002
2006	Clemens	London	100	1001
2008	Cisneros	Sanjose	300	1007
2007	Pereira	Rim	100	1004

9.6 –Jadval. Customers (Qabul qiluvchilarni).

ONUM	AMT	ODATE	CNUM	SNUM
3001	18.69	03/10/1990	2008 yil	1007
3003	767.19	03/10/1990	2001 yil	1001
3002	190 0,10	03/10/1990	2007 yil	1004
3005	5160.45	03/10/1990	2003 yil	1002
3006	1098.16	03/10/1990	2008 yil	1007
3009	1713.23	04.10.1990	2002 yil	1003
3007	75.75	04.10.1990	2004 yil	1002
3008	4723.00	05/10/1990	2006 yil	1001
3010	1309.95	06/10/1990	2004 yil	1002
3011	9891.88	06/10/1990	2006 yil	1001

Masalan, biz sotuvchining ismini bilamiz deylik: Motika, lekin uning maydalangan maydonining qiymatini bilmaymiz va biz buyurtma jadvalidan barcha

buyurtmalarni chiqarib olmoqchimiz. Mavjud bo'lgan yo'li, uchun, albatta, bu:

```
SELECT * FROM Orders WHERE snum = ( SELECT snum FROM
Salespeople WHERE sname = 'Motika');
```

9.4 Ichki so'rovlararga cheklovlar qo'yish

Oldingi misoldagi quyi so'rovimiz bitta va bitta qiymatni qaytarishi kerak. "WHERE sname =" Motika "o'rniga" WHERE city = 'London' "tanlangan snum maydonida bir nechta turli xil qiymatlarni olish mumkin. Bu asosiy so'rov predikatidagi tenglamani ishonchlilik yoki ishonchsizlikni baholash uchun imkonsiz qilishi mumkin va buyruq xato qiladi.

Relyatsion operatorlarga asoslangan predikatlarda sub-so'rovlardan foydalanganda, bitta va faqat bitta chiqish chizig'ini yaratadigan subquerydan foydalanganingizga amin bo'lishingiz kerak. Agar siz hech qanday qiymat ko'rsatmaydigan quyi so'rovdan foydalansangiz, buyruq bajarilmaydi; ammo asosiy so'rov hech qanday qiymatlarni chiqarmaydi.

Hech qanday natija chiqarmaydigan (yoki nol chiqadigan) so'rovlar predikatni na to'g'ri va na yolg'on deb qabul qilishga majbur qiladi. Biroq, noma'lum predikat noto'g'ri predikat kabi ta'sir qiladi: asosiy so'rov bo'yicha satrlar tanlanmaydi.

Avtomatik ravishda har qanday qatorlar uchun bitta qiymatni ishlab chiqaradigan funktsiyaning bir turi, albatta, agregat funksiya hisoblanadi.

GROUP BY bo'limisiz bitta agregat funktsiyasidan foydalanadigan har qanday so'rov asosiy predikatda foydalanish uchun bitta qiymatni tanlaydi. Masalan, 4-oktabr kuni sotib olish miqdori o'rtacha qiymatdan yuqori bo'lgan barcha buyurtmalarni ko'rishni xohlaysiz:

```
SELECT * FROM Orders WHERE amt > ( SELECT AVG (amt)
FROM Orders WHERE odate = 10/04/1990 );
```

Agar siz maxsus IN operatoridan foydalansangiz, har qanday sonli qatorlarni yaratadigan pastki so'rovlardan foydalanishingiz mumkin. Eslatib o'tamiz, IN predikatlarning haqiqat bo'lishi uchun ulardan biri predikat tenglamasidagi boshqa atama bilan mos kelishi kerak bo'lgan qiymatlar to'plamini belgilaydi. Agar sub so'rov bilan IN dan foydalansangiz, SQL shunchaki quyi so'rov natijalaridan hosil bo'ladi. Shuning uchun biz IN-dan relyatsion operator bilan ishlamaydigan sub-so'rovni bajarish va Londonda sotuvchi uchun buyurtma jadvalining barcha xususiyatlarini topish uchun foydalanishimiz mumkin:

```
SELECT * FROM Orders WHERE snum IN ( SELECT snum FROM
Salespeople WHERE city = 'LONDON' );
```

BETWEEN, LIKE va IS NULL so'rovlari quyi so'rovlarda ishlatilmaydi, ammo ALL va ANY iboralardan foydalanish mumkin.

Masalan: Rimdagi har qanday mijozga qaraganda yuqori reytingga ega

bo'lgan mijozlarni tanlang.

```
SELECT * FROM Customers WHERE rating > ANY
( SELECT rating FROM Customers WHERE city = 'Rome' );
```

Agar quyi so'rov bilan tanlangan har bir qiymat tashqi so'rov predikatidagi shartni qondirsa, ALL dan foydalanish to'g'ri bo'ladi. Agar biz avvalgi misolimizni qayta ko'rib chiqishni istasak, faqatgina reytinglari Londondagi har bir mijozdan yuqori bo'lgan mijozlarni ko'rsatish uchun:

```
SELECT * FROM Customers WHERE rating > ALL (SELECT
rating FROM Customers WHERE city = 'Rome');
```

Barcha quyi so'rovlarning ma'nosi shundaki, ularning barchasi bitta ustunni tanlaydi. Bu talab qilinadi, chunki tanlangan chiqish bitta qiymat bilan taqqoslanadi. Buni SELECT * dasturining pastki qismida ishlatib bo'lmaydiganligi tasdiqlaydi. EXISTS so'rovi bilan pastki so'rovlardan foydalanilganda, bundan istisno mavjud.

Korrelyatsiya qilingan quyi so'rovlar.

SQL-da quyi so'rovlardan foydalanganda, tashqi so'rovlar bo'limidagi ichki jadval so'roviga murojaat qilish ***bilan bog'liq so'rovni*** yaratishingiz mumkin Buni amalga oshirganingizda, pastki so'rov asosiy so'rov jadvalining har bir qatoriga bir marta takrorlanadi. Tegishli pastki so'rov, SQL-da uni baholash qiyin bo'lgani uchun juda ko'p nozik tushunchalardan biridir. Agar siz uni o'zlashtirsangiz, u juda kuchli ekanligini topasiz, chunki u juda aniq ko'rsatmalar bilan murakkab funktsiyalarni bajarishi mumkin.

Masalan, 3-oktyabr uchun buyurtmada barcha mijozlarni topishning bir usuli bor:

```
SELECT * FROM Customers outer WHERE 10/03/1990 IN (
SELECT odate FROM Orders inner WHERE outer.cnum =
inner.cnum );
```

EXISTS - bu haqiqiy yoki noto'g'ri qiymatni, boshqacha aytganda, Boolean ifodasini ishlab chiqaruvchi operator. Bu shuni anglatadiki, u predikatda yoki Boolean operatorlari AND, OR, va NOT-dan foydalanib, boshqa mantiqiy iboralar bilan birgalikda ishlashi mumkin.

U pastki so'rovni argument sifatida qabul qiladi va agar u biron-bir natija keltirsa yoki uni bajarmasa, noto'g'ri deb baholaydi. Bu boshqa noma'lum operatorlardan farq qiladi, unda noma'lum. Masalan, agar biz ushbu jadvalda bitta yoki bir nechta mijoz San-Xose shahrida bo'lsa, mijozlar jadvalidan ba'zi ma'lumotlarni olish to'g'risida qaror qabul qilishimiz mumkin:

```
SELECT cnum, cname, city FROM Customers WHERE EXISTS
(SELECT *
FROM Customers WHERE city = 'San Jose');
```

Yoki biz ko'plab xaridorlarga ega bo'lgan sotuvchilarni olib chiqishimiz

mumkin:

```
SELECT DISTINCT snum FROM Customers outer WHERE EXISTS
(SELECT * FROM Customers inner WHERE inner.snum =
outer.snum AND inner.cnum < > outer.cnum );
```

Biroq, ushbu sotuvchilar haqida nafaqat ularning raqamlarini, balki ko'proq ma'lumotni namoyish qilish biz uchun yanada foydali bo'lishi mumkin. Biz buni mijozlar jadvalini sotuvchilar jadvali bilan birlashtirish orqali amalga oshirishimiz mumkin:

```
SELECT DISTINCT first.snum, sname, first.city FROM
Salespeople first, Customers second WHERE EXISTS (
SELECT * FROM Customers third WHERE second.snum =
third.snum AND second.cnum < > third.cnum ) AND
first.snum = second.snum;
```

Savollar:

1. Ma'lumotlar qanday tanlanadi?
2. Ma'lumotlar qanday yangilanadi?
3. Ma'lumotlar qanday o'chiriladi?
4. Qanday qilib jadvallarga qatorlar qo'shiladi?
5. Umumiy funksiya nima?
6. Asosiy agregat funktsiyalarini sanab bering.
7. HAVING qayerda ishlatiladi ?
8. Qanday birikmalar mavjud?
9. O'z-o'zini bog'lash nima?
10. Bog'lanish qanday bosqichma-bosqich tushuntirib bering?
11. So’rovlar qanday amalga oshiriladi?
12. Tegishli pastki so'rov nima?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. C. J. Date - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.

X bob. SQL tili. Ma’lumotlarni tavsiflash

10.1 Ma'lumotni tavsiflash tili

Tavsiflash tilining tarkibi quyidagi operatorlarni o'z ichiga oladi.

1. CREATE - ma'lumotlar bazasi obyektlarini yaratadi
2. ALTER - obyektni o'zgartiradi
3. DROP - obyektni o’chirish
4. ANSI SQL-92 standarti quyidagi obyektlar uchun buyruqlarni belgilaydi:
5. ASSERTION - tasdiqlash uchun ko'rsatmalar
6. CHARACTER SET - belgilar to'plami
7. COLLATION - belgilar to'plami uchun tartiblash qoidalari
8. DOMAIN - domen (foydalanuvchi belgilaydigan ustunlar ma'lumotlari turi).
9. SCHEMA - sxema (nomlangan ob'ektlar guruhi)
10. Jadval - ma'lumotlar bazasi jadvali
11. TRANSLATION - bir belgi to'plamdan boshqasiga o'tkazish (tarjima qilish) qoidalari (TRANSLATE bayonida ishlatiladi)
12. VIEW - ma'lumotlar ko'rinishi

Ma'lumot turlari.

ANSI SQL quyidagi ma'lumotlar turlarini o'z ichiga oladi

Belgilar qatorlari:

- `CHARACTER (n)` yoki `CHAR (n)` - bo'shliqlar bilan ajratilgan, *n*- belgining qat'iy uzunligi;
- `CHARACTER VARYING (n)` yoki `VARCHAR (n)` - maksimal belgilar soni *n* bo'lgan o'zgaruvchan uzunlikdagi qator ;
- `NATIONAL CHARACTER (n)` yoki `NCHAR (n)` - xalqaro kodlashlarni qo'llab-quvvatlaydigan qat'iy uzunlikdagi simvol.
- `NATIONAL CHARACTER VARYING(n)` yoki `nvarchar(n)` - bir tor o'zgaruvchilar uzunligi `nchar`.

Bitli ma'lumotlar:

- `BIT (n)` - *n* bitlar qatori
- `BIT VARYING (n)` - *n* bitgacha bo'lgan qator

Sonlar:

- `INTEGER` va `SMALLINT` butun sonlar;
- `FLOAT`, `REAL` va `DOUBLE PRECISION` – haqiqiy sonlar;
- `NUMERIC` *(aniqlik, o’lchov*) yoki `DECIMAL` (*aniqlik*, *o'lchov*) - Qavslar ichida o'nlik kasrdan oldin va o'nlik kasrdan keyin sonlarni ko'rsatadigan haqiqiy son.

Sana va vaqti:

- `Sana` - sana (2010-05-30) ;
- `TIME` - vaqt (14:55:37) ;
- `TIME WITH TIME ZONE` yoki `TIMESTAMP` - bir xil , `TIME`, faqat vaqt zonasi istisno;
- `TIMESTAMP` - bu `DATE` va `TIME` bir o'zgaruvchiga birlashtirilgan (2010-05-30 14:55:37).
- `TIMESTAMP WITH TIME ZONE` yoki `TIMESTAMPTZ` - bir xil `TIMESTAMP`, faqat vaqtini o'z ichiga olmaydi.

Domen

Domen yaratish:

```
CREATE DOMAIN <DOMAIN_NAME> [ AS ] <veritipinin>
[DEFAULT {LITERAL | NULL | USER }]
  [NOT NULL] [CHEKLASH ( < shart > )]
  [COLLATE < saralash nomi > ];
```

qayerda

- ***DEFAULT*** - Boshqa kirish kiritilmaganda o'rnatilgan standart qiymatni belgilaydi.
  Qadriyatlar:
  - ***LITERAL*** - Muayyan satr, raqamli qiymat yoki sanani kiritish.
  - ***NULL*** - NULL qiymatini kiritish.
  - ***USER*** - joriy foydalanuvchining ismini kiritish. Odatiy qiymatdan foydalanish uchun ustun mos keladigan belgilar turiga ega bo'lishi kerak.
- ***NOT NULL*** - ustunga kiritilgan qiymatlar NULL bo'lmasligi kerakligini bildiradi.
- ***CHECK*** - (<shart>) domen uchun yagona CHECK cheklovini yaratadi.
- ***VALUE*** - oxirida domen nomi uchun ustun nomi.
- ***COLLATE <sort name>*** - Domen uchun saralash usulini o'rnatadi.

Quyidagi bayonot 1000 dan katta ijobiy qiymatlarni qabul qilishi mumkin bo'lgan domenni yaratadi, asl qiymati 9999. VALUE qiymati kalit so'z ushbu domen asosida ustun nomi bilan almashtiriladi.

```
CREATE DOMAIN CUSTNO
  AS INTEGER
    DEFAULT 9999
    CHECK (VALUE > 1000);
```

Quyidagi bayonnoma kiritilgan qiymatlarni domen uchun to'rtta maxsus qiymat bilan cheklaydi:

```
CREATE DOMAIN PRODTYPE
  AS VARCHAR(12)
    CHECK (VALUE IN ("software", "hardware", "other",
"N/A"));
```

Domenni o'zgartirish:

```
ALTER DOMAIN <domen nomi> {
[SET DEFAULT {LITERAL | NULL | USER}]
  | [DROP DEFAULT]
  | [ADD [CONSTRAINT] CHECK ((shart)
| [DROP CONSTRAINT]
  };
```

qayerda

- ***SET DEFAULT*** - boshqa kirish kiritilmaganda kiritiladigan standart ustun qiymatini belgilaydi. Qadriyatlar:
  - ***LITERAL -*** Muayyan satr, raqamli qiymat yoki sanani kiritish.
  - ***NULL -*** NULL qiymatini kiritish.
  - ***USER -*** joriy foydalanuvchining ismini kiritish. Odatiy qiymatdan foydalanish uchun ustun mos keladigan matn turiga ega bo'lishi kerak.

Odatiy qiymatni ustun darajasida belgilash domen darajasida standart qiymatni bekor qiladi.

- ***DROP DEFAULT*** - mavjud standart qiymatni o'chiradi.
- ***ADD [CONSTRAINT] CHECK (<shart>)*** - Domen aniqlanishiga CHECK cheklovlarini qo'shadi. Domen ta'rifida faqat bitta CHECK cheklovi bo'lishi mumkin.
- ***DROP CONSTRAINT*** - Domen ta'rifidan ***CHECK*** cheklovlarini olib tashlaydi.

Quyidagi bayonot standart domen qiymatini 9999 ga o'rnatadi.

```
ALTER DOMAIN CUSTNO SET DEFAULT 9999
```

Domenni o'chirish:

```
DROP DOMAIN <domen nomi>;
```

Chizma

Ma'lumotlar bazasini yaratish:

```
CREATE {DATABASE | SCHEMA} <ma'lumotlar bazasi nomi>
```

Ma'lumotlar bazasini o’chirish:

```
DROP { DATABASE | SCHEMA } <ma'lumotlar bazasi nomi>
```

Jadval .

Jadval yaratish

```
CREATE TABLE [ IF NOT EXISTS ] <table_name>
(
<ustun nomi_1> <ma'lumotlar turi> [DEFAULT expression ]
[ {NULL | NOT NULL} ] [ {INDEX_BLIST | INDEX_NONE} ]
```

```
<ustun nomi_2> <ma'lumotlar turi> [ DEFAULT expression
] [ {NULL | NOT NULL} ] [ {INDEX_BLIST | INDEX_NONE} ]
...
< ustun nomi _N> < ma'lumotlar turi > [ DEFAULT
expression ] [ {NULL | NOT NULL} ] [ {INDEX_BLIST |
INDEX_NONE} ]
[ CONSTRAINT < cheklov nomi >]
PRIMARY KEY (< ustun nomi _1>, < ustun nomi _2>, ...) |
FOREIGN KEY (< nomi ustun _1>, < nomi ustun _2>, ...)
REFERENCES < nomi _ stol _2> [(< nomi ustun _1>,
< nomi ustun ON _2>, ...)] [ UPDATE {NO ACTION | SET
NULL | SET DEFAULT | CASCADE} ] [ ON DELETE {NO ACTION
| SET NULL | SET DEFAULT | CASCADE} ] |

UNIQUE (<ustun nomi_1>, <ustun nomi_2>, ...) |

CHECK (<shart>) [ {INITIALLY DEFERRED | INITIALLY
IMMEDIATE} ] [ {NOT DEFERRABLE | DEFERRABLE} ]);
```

qayerda

- ***DEFAULT*** expression - standart qiymat;
- ***NULL*** | ***NOT NULL*** - bo'sh maydonga ruxsat berilganmi yoki yo'qmi;
- ***INDEX_BLIST*** | ***INDEX_NONE*** - indeks mavjud yoki yo'qligi;
- **CONSTRAINT** – cheklov
  - ***PRIMARY KEY -*** birlamchi kalit
  - ***FOREIGN KEY -*** ikkinchi darajali kalit
    - ON DELETE – qarindoshlik jadvalida o'chirishda
    - ON UPDATE - qarindoshlik jadvalida yangilanganda
    - NO ACTION - harakat yo'q
    - SET NULL - NULL ga o'zgartirish
    - SET DEFAULT - standart qiymatni o'rnatadi
    - CASCADE - kaskad
  - ***UNIQUE -*** noyob
  - ***CHECK -*** tekshirish

Masalan

```
CREATE TABLE Customer (
  number VARCHAR(40)  NOT NULL,
  name   VARCHAR(100) NOT NULL,
  ssn    VARCHAR(50)  NOT NULL,
  age    INTEGER      NOT NULL,

  CONSTRAINT cust_pk PRIMARY KEY (number),
  UNIQUE ( ssn ),  // (An anonymous constraint)
  CONSTRAINT age_check CHECK (age >= 0 AND age < 200)
);
```

Jadvalni o'zgartirish:

Jadval nomini o'zgartirish

```
ALTER TABLE <jadval_nomi> RENAME TO <yangi_ jadval_nomi
>
```

Ustunning nomini o'zgartirish

```
ALTER TABLE  <jadval nomi> RENAME [ COLUMN ] <ustun
nomi> TO <yangi ustun nomi>
```

Ustun qo'shish

```
ALTER TABLE  < jadval nomi > ADD[ COLUMN ] <ustun nomi>
<ma'lumotlar turi> [ DEFAULT expression ]
[{ NULL | EMAS NULL }]
[{ INDEX _ blist | INDEX _ NONE }]
```

Jadvalga cheklovsiz birlamchi kalit qo'shish

```
ALTER TABLE < nomi _ -jadval > ADD[CONSTRAıNT'I
< nomi cheklovlar >]
PRIMARY KEY (< ustun nomi _1>, < ustun nomi _2>, ...) |
```

Jadvalga ikkinchi darajali cheklash kalitini qo'shish

```
ALTER TABLE < nomi _ -jadval > ADD[CONSTRAINT
< nomi cheklovlar >]
FOREIGN KEY (< nomi ustun _1>, < nomi ustun _2>, ...)
REFERENCES < nomi _ stol _2> [(< nomi ustun _1>,
< nomi ustun ON _2>, ...)] [ UPDATE {NO ACTION | SET
NULL | SET DEFAULT | CASCADE} ] [ ON DELETE {NO ACTION
| SET NULL | SET DEFAULT | CASCADE} ] ||
```

Jadvalga yangi ustun qo'shish

```
ALTER TABLE  <table_name>  ADD [ CONSTRAINT  <cheklov
nomi>]
UNIQUE (<ustun nomi_1>, <ustun nomi_2>, ...) |
```

Jadvalga ustunli tekshirishni qo'shish

```
ALTER TABLE <table_name> ADD [ CONSTRAINT <cheklov
nomi>] CHECK  (<shart>) [ {INITIALLY DEFERRED |
INITIALLY IMMEDIATE} ] [ {NOT DEFERRABLE | DEFERRABLE}
]
```

Ustun ma'lumotlar turini o'zgartirish

```
ALTER TABLE < nomi _ stol > MODIFY "ustun 1", "Yangi
Data turi"
```

Ustun cheklovlari ustunini o'zgartirish

```
ALTER TABLE < nomi _ stol >, ALTER [COLUMN] column_name
SET default_expr
```

```
ALTER TABLE < nomi _ stol >, ALTER [COLUMN] column_name
DROP default
```

Ustunni o'chirish o'zgarishi

```
ALTER TABLE <Ustun _ nomi > DROP [COLUMN] column_name
```

Jadval cheklovini olib tashlash

```
ALTER TABLE < Ustun _ nomi > DROP CONSTRAINT
constraint_name
```

Birlamchi kalitlarni olib tashlash

```
ALTER TABLE < Ustun _ nomi > DROP PRIMARY KEY
```

Jadvalni o'chirish

```
DROP TABLE [ IF EXISTS ] <table_name>
```

10.2 Tasavvurlar

Tasavvurlar (VIEW) – bu ma'lumotlar bazasi haqida hech qanday ma'lumot bo'lmagan ma'lumotlar ob'ekti. Bu so'rovni bajarish orqali boshqa jadvallardan tarkib topadigan jadval turi. Ushbu jadvallardagi qiymatlar avtomatik ravishda o'zgarganligi sababli ularning qiymatlari ko'rinish bilan ko'rsatilishi mumkin. Ushbu bobda siz qanday qarashlar, qanday yaratilganligi va ularning imkoniyatlari va cheklovlari haqida bir oz ma'lumotga ega bo'lasiz. Birlashma va pastki so'rov kabi rivojlangan so'rov vositalariga asoslangan, diqqat bilan ishlab chiqilgan ko'rinishlardan foydalanish ba'zi hollarda so'rovlarga qaraganda ancha foydali bo'ladi.

Tasavvurlar - bu tarkib tanlangan yoki boshqa jadvallardan olingan jadvallar. Ular DML so'rovlari va iboralarida, xuddi asosiy jadvallar kabi ishlaydi, lekin o'zlarining biron bir ma'lumotlarini o'z ichiga olmaydi. Tasavvurlar oynada xuddi siz ko'rishingiz mumkin bo'lgan ma'lumotni (u yoki boshqa rasmda, keyinchalik ko'rasiz) oynaga o'xshaydi, u aslida asosiy jadvalda saqlanadi. Tasavvur - bu aslida buyruq mavzusiga aylanadigan har doim bajariladigan so'rov. Shu bilan birga, so'rov natijasi ko'rinishning mazmuniga aylanadi.

Siz CREATE VIEW buyrug'i bilan tasavvurlarni yaratasiz. U CREATE VIEW so'zlaridan, siz yaratmoqchi bo'lgan ***tasavvurning*** nomi, AS (AS) so'zlaridan va keyin quyidagi misolda bo'lgan so'rovdan iborat:

```
CREATE VIEW Londonstaff
    AS SELECT *
    FROM Salespeople
    WHERE city = 'London';
```

Endi sizda Londonstaff degan tasavvur bor. Siz ushbu ko'rinishni har qanday boshqa jadval kabi ishlatishingiz mumkin. Bu boshqa jadvallar va tasavvurlar uchun so'ralishi, o'zgartirilishi, joylashtirilishi, o'chirilishi va ulanishi mumkin. Keling,

bunday vakillik uchun so'rov beramiz (10.1-rasmda keltirilgan):

```
SELECT *
    FROM Londonstaff;
    ==============  SQL Execution Log ============
   |                                               |
   | SELECT *                                      |
   | FROM  Londonstaff;                            |
   |                                               |
   | ============================================= |
   |   snum      sname          city         comm  |
   | ------    ----------   -----------   -------  |
   |   1001      Peel          London       0.1200  |
   |   1004      Motika        London       0.1100  |
    ============================================
```

10.1-rasm. London staff taqdimoti .

SQL-ga buyruqdan (SELECT) barcha qatorlarni (*) tanlashni buyurganingizda, Londonstaff, degan so'rovni bajaradi va uning barcha natijalarini qaytaradi. Tasavvur so'rovida predikatga ega bo'lgan holda, siz ushbu predikatni qondiradigan ko'rinishda faqat satrlarni chiqarishingiz mumkin. Agar shunday bo'lsa, taqdimotingiz uchun boshqasini tanlashingiz kerak bo'ladi). Asosiy jadval bilan taqqoslaganda tasavvurni ishlatishning afzalligi shundaki, uning ostidagi jadval o'zgarganda har doim avtomatik ravishda ko'rinish o'zgaradi. Tasavvur tarkibi mahkamlanmagan va har safar buyruqqa murojaat qilganingizda qayta tayinlanadi. Agar ertaga Londonda yashaydigan boshqa sotuvchini qo'shsangiz, u avtomatik ravishda tasavvurda paydo bo'ladi.

Tasavvurlar ma'lumotlaringizni boshqarishni sezilarli darajada kengaytiradi. Bu jadvalda keltirilgan ma'lumotlardan ba'zilariga, ammo barchasiga hammaga ochiq bo'lishning ajoyib usulidir. Agar siz sotuvchingizni sotuvchilar jadvalida ko'rsatilishini istasangiz, lekin boshqa sotuvchilarning komissiyalari ko'rsatilmagan bo'lsa, siz quyidagi operator yordamida tasavvurni yaratishingiz mumkin (10.2-rasmda keltirilgan)

```
CREATE VIEW Salesown
    AS SELECT snum, sname, city
    FROM Salespeople:
        ============== SQL Execution Log ============
   |                                               |
   | SELECT *                                      |
   | FROM  Salesown;                               |
   |                                               |
   | ============================================= |
   |   snum      sname          city               |
   | ------    ----------   -----------            |
```

```
|   1001      Peel         London              |
|   1002      Serres       San Jose            |
|   1004      Motika       London              |
|   1007      Rifkin       Barcelona           |
|   1003      Axelrod      New York            |
 ============================================
```

10.2-rasm. Salesown ko'rinishi

Boshqacha qilib aytganda, ushbu ko'rinish Sotuvchilar jadvalidagi kabi bir xil, bundan tashqari, so'rovda comm maydoni ko'rsatilmagan va shuning uchun tasavvurga kiritilmagan.

Tasavvurlarni o'zgartirish

Endi tasavvurni DML modifikatsiya qilish buyruqlari bilan o'zgartirish mumkin, ammo modifikatsiya ko'rinishga o'zi ta'sir qilmaydi. Buyruqlar aslida asosiy jadvalga yo'naltiriladi:

```
UPDATE Salesown
    SET city = 'Palo Alto'
    WHERE snum = 1004;
```

Uning harakati xuddi shu buyruqni sotuvchilar jadvalida bajarish bilan bir xil. Ammo, agar sotuvchining komissiya qiymati UPDATE buyrug'i bilan ishlangan bo'lsa

```
UPDATE Salesown
    SET comm = .20
    WHERE snum = 1004;
```

rad qilinadi, chunki "satr" savdo maydoni ko'rinishida emas. Bu barcha nuqtai nazarlarni o'zgartirib bo'lmasligini ko'rsatadigan muhim nuqta.

Ustunlarni nomlash

Bizning misolimizda, bizning qarashlarimizning pastki qismida ularning nomlari to'g'ridan-to'g'ri asosiy jadvalning maydon nomlaridan olingan. Bu qulay. Biroq, ba'zida siz ustunlaringizni yangi nomlar bilan ta'minlashingiz kerak bo'ladi:

- ba'zi ustunlar chiqarilganda va shuning uchun nomlanmaydi.
- qo'shilishdagi ikkita yoki undan ortiq ustunlar asosiy jadvaldagi nomlarga teng bo'lganda.

Maydon nomlari bo'lishi mumkin bo'lgan ismlar qavs ichida (), jadval nomidan keyin berilgan. Ular qilinmaydi maydonlarni talab stol nomlarini mos bo'lsa talab. Ushbu maydonlarning ma'lumotlari turi va o'lchamlari ularga berilgan "so'ralgan" maydonlardan farq qiladi. Odatda siz yangi maydon nomlarini ko'rsatmaysiz, lekin agar buni amalga oshirgan bo'lsangiz, ko'rinishni har bir jins uchun bajarishingiz kerak.

10.3 Tasavvur predmetlari va asosiy so'rovlarni birlashtirish

Taqdimot so'rovini yuborganingizda, aslida siz so'rov yubormoqdasiz. SQL-ni topishning asosiy usuli ikkita so'rovlarning predikatlarini bitta biriga birlashtirish. U bilan Londonstaffga bo'lgan qarashimizga yana bir bor nazar tashlaymiz:

```
CREATE VIEW Londonstaff
     AS SELECT *
     FROM Salespeople
     WHERE city = 'London';
```

Agar biz ushbu tasavvurda quyidagi so'rovni bajarsak

```
SELECT *
     FROM Londonstaff
     WHERE comm > .12;
```

Xuddi shu narsa biz sotuvchilar jadvalida quyidagilarni bajarganimiz bilan bir xil:

```
SELECT *
     FROM Salespeople
     WHERE city = 'London'
     AND comm > .12;
```

Bu juda yaxshi, ammo taqdimotda muammo bo'lishi mumkin. Ikkita to'liq haqiqiy predikatlarni birlashtirish va ishlamaydigan predikatni olish imkoniyati mavjud. Masalan, biz quyidagi tasavvurlarni yaratamiz deylik (CREATE).

```
CREATE VIEW Ratingcount (rating, number)
     AS SELECT rating, COUNT (*)
     FROM Customers
     GROUP BY rating;
```

Bu bizga har bir reyting darajasi uchun mijozlar sonini beradi. Keyin uchta mijozga berilgan reyting mavjudligini bilish uchun ushbu yuborishni so'rashingiz mumkin:

```
SELECT *
     FROM Ratingcount
     WHERE number = 3;
```

Keling, ikkita predikatlarni birlashtirsak nima bo'lishini ko'rib chiqamiz:

```
SELECT rating, COUNT (*)
     FROM Customers
     WHERE COUNT (*) = 3
     GROUP BY rating;
```

Bu yaroqsiz so‘rov. COUNT kabi agregat funktsiyalar predikatda ishlatilmaydi. Albatta, yuqorida aytib o'tilgan talabni rasmlantirishning to'g'ri usuli, albatta:

```
SELECT rating, COUNT (*)
     FROM Customers
     GROUP BY rating;
     HAVING COUNT (*) = 3;
```

Ammo SQL o'zgarishlarni amalga oshirmasligi mumkin. Ratingcount o'rniga ekvivalent so'rov muvaffaqiyatsiz bo'ladimi? Ha, mumkin! Bu SQL-ning noaniq sohasi bo'lib, unda ko'rish usullari yaxshi natijalar berishi mumkin. Tizim hujjatlarida bu haqida hech narsa aytilmaganda amalga oshiriladigan eng yaxshi narsa bu uni aniqlashga urinishdir. Agar buyruq to'g'ri bo'lsa, so'rovlar sintaksisida ba'zi SQL cheklovlarini o'rnatish uchun siz tasavvurlardan foydalanishingiz mumkin.

Guruhli tasavvurlar.

Guruh tasavvuri - bu oldingi misolda GROUP BY iborasini o'z ichiga olgan yoki boshqa guruh tasavvuriga asoslangan Ratingcount so'rovi kabi tasavvurlar. Guruh taqdimotlari olingan ma'lumotlarni uzluksiz qayta ishlashning ajoyib usuli bo'lishi mumkin. Aytaylik, har kuni siz mijozlar raqamlari tartibini, buyurtmalarni qabul qiladigan sotuvchilar sonini, buyurtma raqamlarini, buyurtmalarning o'rtacha miqdorini va buyurtmalarni sotib olishning umumiy miqdorini kuzatib borishingiz kerak .

Har safar murakkab so'rovni tuzib, siz oddiygina tasavvurni yaratishingiz mumkin:

```
CREATE VIEW Totalforday
     AS SELECT odate, COUNT (DISTINCT cnum), COUNT
          (DISTINCT snum), COUNT (onum), AVG
          (amt), SUM (amt)
          FROM Orders
          GROUP BY odate;
```

Endi siz ushbu ma'lumotlarning barchasini oddiy so'rov bilan ko'rishingiz mumkin:

```
SELECT *
FROM Totalforday;
```

Ko'rib turganimizdek, SQL so'rovlari sizga to'liq imkoniyatlarni taqdim etishi mumkin, shuning uchun ko'rinishlar sizning ma'lumotingiz qanday ishlatilishini aniqlaydigan juda moslashuvchan va kuchli vositani taqdim etadi. Shuningdek, ular ma'lumotlarni o'zingiz uchun qulay bo'lgan tarzda qayta formatlash va ikki tomonlama ishlarni olib tashlash orqali ishingizni osonlashtirishi mumkin.

Tasavvurlar va birlashmalar.

Tasavvurlar ularni bitta asosiy jadvaldan olishni talab qilmaydi. SQL so'rovini deyarli har qanday tasavvurda ishlatish mumkinligi sababli, u bazaviy jadvallarning yoki boshqa tasavvurlarning har qanday sonidan ma'lumotlarni chiqarishi mumkin. Masalan, biz sotuvchi va xaridorlarning ismlarini aytib beradigan buyurtmalarni ko'rsatadigan tasavvurni yaratishimiz mumkin:

```
CREATE VIEW Nameorders
     AS SELECT onum, amt, a.snum, sname, cname
     FROM Orders a, Customers b, Salespeople c
     WHERE a.cnum = b.cnum
     AND a.snum = c.snum;
```

Endi siz buyurtmachining yoki sotuvchining (*) barcha buyurtmalarini (SELECT) tanlashingiz mumkin yoki siz har qanday buyurtma uchun ushbu ma'lumotni ko'rishingiz mumkin. Masalan, Rifkin sotuvchisining barcha buyruqlarini ko'rish uchun siz quyidagi so'rovni kiritishingiz kerak (10.3-rasmda keltirilgan):

```
SELECT *  FROM Nameorders
 WHERE sname = 'Rifkin';
   ==============  SQL Execution Log ==============
   |                                                |
   | SELECT *                                       |
   | FROM  Nameorders                               |
   | WHERE sname = 'Rifkin';                        |
   | ============================================== |
   |   onum       amt       snum   sname     cname   |
   |  ------   --------    -----  -------   -------  |
   |   3001       18.69     1007  Rifkin    Cisneros |
   |   3006     1098.16     1007  Rifkin    Cisneros |
   |                                                |
```

10.3-rasm. Rifkin buyurtmalari Nameorders-da ko'rsatilgan

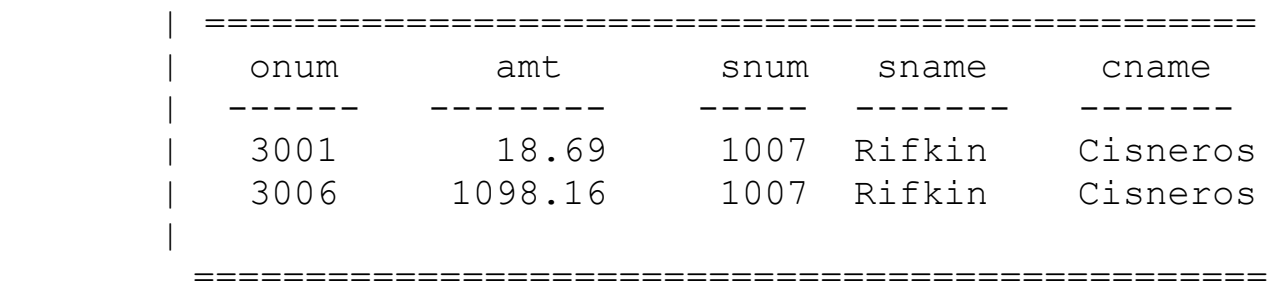
Shuningdek, siz tasavvurni boshqa jadvallar yoki tayanch jadvallar yoki tasavvurar bilan birlashtirishingiz mumkin, shunda Axelrodning barcha buyruqlarini va uning komissiya qiymatlarini har bir tartibda ko'rishingiz mumkin:

```
SELECT a.sname, cname, amt  comm
FROM Nameorders a, Salespeople b
WHERE a.sname = 'Axelrod'
AND b.snum = a.snum;
```

Ushbu so'rov natijasi 10.4-rasmda keltirilgan.

Biz predikatda " WHERE a.sname = 'Axelrod' AND b.sname = 'Axelrod' " deb yozishimiz mumkin, ammo biz bu erda ishlatgan predikat ko'proq tarqalgan. Bundan tashqari, snum maydoni Sotuvchilar jadvalining asosiy kalitidir va shuning uchun ta'rifiga ko'ra noyob bo'lishi kerak.

```
   ============== SQL Execution Log ==============
   |                                                |
   | SELECT a.sname, cname, amt * comm              |
   | FROM  Nameorders a, Salespeople b              |
   | WHERE a.sname = 'Axelrod'                      |
   | AND b.snum = a.snum;                           |
```



onum	amt	snum	sname	cname
3001	18.69	1007	Rifkin	Cisneros
3006	1098.16	1007	Rifkin	Cisneros

10.4-rasm. Tasavvur bilan asosiy jadvalga qo'shiling

Agar u erda bo'lsa, masalan, ikkita Axelrodfs bo'lsa, ularning nomlari bilan bir variant ularning ma'lumotlarini birlashtiradi. Yana afzalroq variant - bu alohida saqlash uchun snum maydonidan foydalanish.

Tasavvurlar va oddiy so'rovlar.

Tasavvurlar, shuningdek, oddiy so'rovlardan foydalanishi mumkin. Aytaylik, sizning kompaniyangiz istalgan sanada eng yuqori buyurtmaga ega bo'lgan sotuvchilarga mukofot taqdim etadi. Tasavvur orqali ushbu ma'lumotni kuzatishingiz mumkin:

```
CREATE VIEW Elitesalesforce
     AS SELECT b.odate, a.snum, a.sname,
     FROM Salespeople a, Orders b
     WHERE a.snum = b.snum
     AND b.amt =
          (SELECT MAX (amt)
          FROM Orders c
          WHERE c.odate = b.odate);
```

Agar boshqa tomondan, mukofot faqat oxirgi o'n yil ichida eng yuqori buyurtmaga ega bo'lgan sotuvchiga tayinlansa, siz ularni birinchisiga ko'ra boshqa vakolatxonada kuzatishingiz kerak bo'ladi:

```
CREATE VIEW Bonus
     AS SELECT DISTINCT snum, sname
     FROM Elitesalesforce a
     WHERE 10 < =
          (SELECT COUNT (*)
          FROM Elitesalestorce b
          WHERE a.snum = b.snum);
```

Ushbu jadvaldan mukofot oladigan sotuvchini ajratib olish oddiy savol bilan amalga oshiriladi:

```
SELECT *
FROM Bonus;
```

Endi biz SQL haqiqiy kuchini ko'rmoqdamiz. RPG yoki COBOL bilan bir xil ma'lumotni olish uzoqroq jarayon bo'ladi. SQL-da, bu oddiy so'rov bilan birgalikda tasavvurda saqlanadigan ikkita murakkab buyruqning masalasidir. Mustaqil so'rov

bilan biz har kuni bunga g'amxo'rlik qilishimiz kerak, chunki so'rovni qabul qiladigan ma'lumotlar bazasining hozirgi holatini aks ettirish uchun doimiy ravishda o'zgarib turadi.

Tasavvur nima qila olmaydi.

Taqdimot turlari juda ko'p (shu bobda bizning ko'plab misollarimiz mavjud), ular faqat o'qish uchun mo'ljallangan. Bu degani, ular so'ralishi mumkin, ammo modifikatsiya qilish buyruqlari ularga ta'sir etmaydi.Tasavvur ta'riflarida ruxsat etilmagan ba'zi so'rov turlari ham mavjud. Bitta tasavvur bitta so'rovga asoslanishi kerak; UNION va UNIOM ALL ruxsat berilmaydi. Tasavvurlarni aniqlashda ORDER BY hech qachon ishlatilmaydi. So'rov natijalari asosiy jadvalga o'xshaydigan va ta'rifiga ko'ra tartibsiz bo'lgan tasavvurning tarkibini hosil qiladi.

Tasavvurni olib tashlash.

Ma'lumotlar bazasidan tasavvurni o'chirish uchun sintaksis bazaviy jadvallarni yo'q qilish sintaksisiga o'xshaydi.

```
DROP VIEW < view name >
```

Buning keragi yo'q, lekin avval siz barcha tarkibni o'chirib tashlashingiz kerak, chunki u asosiy jadval bilan bajarilgan, chunki tasavvurning tarkibi yaratilmagan va ma'lum bir buyruq davomida saqlangan. Tasavvur olingan asosiy jadval ko'rinishni yo'q qilganda samarali bo'lmaydi. Yodingizda bo'lsin, uni o’chirish uchun siz qarashingiz kerak.

Savollar:

1. Ma'lumotni aniqlash tili nima?
2. Ma'lumotni aniqlash tilidan foydalanib qanday ob'ektlarni yaratish mumkin?
3. DDL- dan foydalanib jadvallarni qanday o'zgartirish mumkin ?
4. Tasavvur nima?
5. Taqdimot qanday tuziladi?
6. Tasavvur ma'lumotlarini yangilay olamanmi va nega?

Adabiyotlar:

1. Thomas Connolly, Carolyn Begg – Database systems. A practical Approach to Design, Implementation and Management. 4th Edition – Addison Wesley 2005 – 1373p.
2. C. J. Date – An Introduction to Database Systems – Addison-Wesley Professional – 2003 – 1024 p.

XI Bob. Tranzaktsiyalarni boshqarish. So’rovlar yaratish va qayta ishlash.

11.1 Tranzaktsiyalarni boshqarish maqsadi

Ma'lumotlarning izchilligi (ba'zida ma'lumotlarning izchilligi) bu ma'lumotlarning bir-biriga mosligi, ma'lumotlar yaxlitligi va ichki muvofiqligi.

Tahlilchi va ma'lumotlar bazasini yaratuvchisining vazifasi barcha mavjud

yaxlitlik cheklovlarini yanada aniqroq aniqlash va ularni ma'lumotlar bazasiga o'rnatishdir.

Ma'lumotlar bazasining yaxlitligi unda mavjud bo'lgan ma'lumotlarning ishonchliligini kafolatlamaydi, ammo hech bo'lmaganda, haqiqatdan ham ishonib bo'lmaydigan va mumkin bo'lmagan qiymatlarni rad etib, ushbu ma'lumotlarning to'g'riligini ta'minlaydi. Shunday qilib, ma'lumotlar bazasining yaxlitligini ma'lumotlar bazasining ishonchliligi bilan chalkashtirmaslik kerak. Ishonchlilik (yoki haqiqat) bu ma'lumotlar bazasida saqlanadigan faktlarning real dunyo bilan muvofiqligi. Shubhasiz, ma'lumotlar bazasining ishonchliligini aniqlash uchun siz ma'lumotlar bazasining mazmuni va real dunyo haqida to'liq ma'lumotga ega bo'lishingiz kerak. Ma'lumotlar bazasining yaxlitligini aniqlash uchun faqat ma'lumotlar bazasi mazmuni va ular uchun o'rnatilgan qoidalar to'g'risidagi ma'lumot talab qilinadi. Shuning uchun, MBBT ma'lumotlar bazasining yaxlitligini boshqarishi mumkin (va kerak), ammo bazaning ishonchliligini tubdan nazorat qila olmaydi. Ma'lumotlar bazasining ishonchliligini kuzatish faqat bir kishiga va hatto cheklangan miqyosda ham tayinlanishi mumkin, chunki ba'zi hollarda odamlar real dunyo to'g'risida to'liq ma'lumotga ega emaslar.

Shunday qilib, ma'lumotlar bazasi to'liq bo'lishi mumkin, ammo ishonchli emas. Buning teskarisi ham bo’lishi mumkin: ma'lumotlar bazasi ishonchli bo'lishi mumkin, ammo yaxlit emas. Ikkinchisi, agar qoidalar (yaxlitlik cheklovlari) noto'g'ri o'rnatilgan bo'lsa.

Bitim - bu ishning mantiqiy birligi. Masalan. Avval talabalar bilan aloqalar (talabalar bilan aloqalar) qo'shimcha mashg'ulot AvgMark atributini o'z ichiga oladi, deylik, hozirgi mashg'ulot natijalariga ko'ra talabalarning o'rtacha balli. Belgilangan har qanday qism uchun AvgMark qiymati joriy semestrda olingan barcha baholar uchun Marks jadvalidagi barcha Mark qiymatlarining arifmetik o'rtacha qiymatiga teng deb qabul qilinadi.

Yuqoridagi misol bu yagona, atom operatsiyasi deb taxmin qilmoqda. Aslida, "Belgilar" jadvaliga yangi reyting qo'shish bu ma'lumotlar bazasida ikkita yangilanish (bu erda yangilanish, albatta, o'zlarini kiritish, o'chirish va yangilashni anglatadi). Bundan tashqari, ushbu ikki yangilanish o'rtasidagi ma'lumotlar bazasida talaba uchun AvgMark 1-ning joriy semestrda 1-talaba uchun barcha Mark maydon qiymatlarining arifmetik o'rtacha qiymatiga tengligi sharti buzilgan. Shunday qilib, ishning mantiqiy birligi (ya'ni, tranzaktsiya) bu ma'lumotlar bazasi tizimining bitta operatsiyasi emas, balki bu kabi bir nechta operatsiyalarni muvofiqlashtirishdir. Umuman olganda, bu ma'lumotlar bazasining bitta doimiy holatini boshqasiga o'tkazish va oraliq nuqtalarda ma'lumotlar bazasi nomuvofiq holatda.

Bundan kelib chiqadiki, yangilanishlardan birini amalga oshirish mumkin emas, ikkinchisi esa amalga oshirilmaydi, chunki ma'lumotlar bazasi nomuvofiq holatda qoladi. Ideal holda, ikkala yangilanish ham amalga oshirilishi kerak. Biroq, shunday bo'lishiga 100% kafolat berishning iloji yo'q. Ehtimol, masalan, ikkita yangilash o'rtasida tizim yo'q qilinishi yoki ikkinchi yangilanishda arifmetik toshib ketishlar va boshqalar bo'lishi mumkin. Tranzaktsiyalarni qo'llab-quvvatlaydigan

tizim, agar ba'zi yangilanishlar paytida (biron bir sababga ko'ra) xato bo'lsa, ushbu yangilanishlar bekor qilinishini ta'minlaydi. Shunday qilib, tranzaktsiya to'liq bajariladi yoki butunlay bekor qilinadi (go'yo u umuman bajarilmagan kabi).

Atomiklik (yoki uning o'xshashligi) ni ta'minlaydigan tizim komponenti tranzaksiya menejeri (yoki bitim menejeri) deb nomlanadi va COMMIT TRANSACTION va ROLLBACK TRANSACTION ko'rsatmalari uning bajarilishining kaliti bo'lib xizmat qiladi.

COMMIT TRANSACTION bayonoti (qisqartirish, bajarish uchun) muvaffaqiyatli bitim to'g'risida signal beradi. Bu tranzaktsiyalar menejeriga mantiqiy ish birligi muvaffaqiyatli yakunlanganligi, ma'lumotlar bazasi yana izchil holatda (yoki bo'ladi) va mantiqiy ish birligi tomonidan qilingan barcha yangilanishlarni amalga oshirish mumkinligi to'g'risida xabar beradi.

ROLLBACK TRANSACTION iborasi (qisqa ROLLBACK uchun) muvaffaqiyatsiz bitimni bildiradi. Bu tranzaksiya menejeriga biron bir xato yuz berganligi, ma'lumotlar bazasi nomuvofiq holatda bo'lganligi va barcha yangilanishlar bekor qilinishi mumkinligi haqida xabar beradi.

Yangilanishlarni bekor qilish uchun tizim barcha yangilanish operatsiyalari tafsilotlari, xususan, o'zgartirilgan ob'ektning yangi va eski qiymatlari yozilgan diskdagi ro'yxatga olish faylini yoki jurnalni qo'llab-quvvatlaydi. Shunday qilib, ba'zi yangilanishni bekor qilish kerak bo'lsa, tizim ob'ektni asl holatiga qaytarish uchun tegishli ro'yxatga olish faylidan foydalanishi mumkin.

Yana bir muhim nuqta. Tizim alohida operatorlarning o'zlari atom bo'lishini ta'minlashi kerak (ya'ni ular to'liq bajarilgan yoki umuman bajarilmagan). Bu, ayniqsa operatorlar ko'p bosqichli va odatda bir vaqtning o'zida bir nechta tutqichlarda ishlaydigan aloqa tizimlari uchun juda muhimdir; bunday operatorni operatsiya o'rtasida buzib bo'lmaydi va tizimni nomuvofiq holatga keltiradi. Boshqacha qilib aytganda, agar bunday operatorning ishlashi paytida xatolik yuz bergan bo'lsa, ma'lumotlar bazasi to'liq o'zgarishsiz qolishi kerak. Bundan tashqari, bu operatorning xatti-harakatlari, masalan, kaskadlash, qo'shimcha ishlashga olib keladigan bo'lsa ham, to'g'ri bo'lishi kerak.

11.2 AIICH xususiyatlari

Oldingi bo'limlardan kelib chiqadigan bo'lsak, tranzaktsiyalar to'rtta muhim xususiyatga ega: atomlik, izchillik, izolyatsiya va chidamlilik (keling, buni ASID xususiyatlari deb ataymiz).

1. ***Atomiylik.*** Tranzaktsiyalar atomdir (barchasi yoki hech narsa bajarilmaydi).
2. ***Izchillik.*** Tranzaktsiyalar ma'lumotlar bazasini doimiy ravishda himoya qiladi. Bu shuni anglatadiki, tranzaktsiyalar ma'lumotlar bazasining bitta doimiy holatini boshqasiga boshqa barcha oraliq punktlarda izchillikni saqlamasdan tarjima qiladi.
3. ***Izolyatsiya.*** Bitimlar bir-biridan ajratilgan. Bu shuni anglatadiki, ko'plab raqobatbardosh tranzaktsiyalar amalga oshirilgan bo'lsa ham, ma'lum bir tranzaktsiyaning har qanday yangilanishi ushbu operatsiya bajarilguncha

boshqasidan yashirin bo'ladi. Boshqacha aytganda, har qanday ikkita T1 va T2 operatsiyalari uchun quyidagi fikr to'g'ri: T1 T2 yangilanishini faqat T2 bajarilgandan keyin ko'rishi mumkin, T2 esa T1 yangilanishini faqat T1 bajarilgandan keyin ko'rishi mumkin.

4. ***Chidamlilik.*** Tranzaktsiya tugagach, uning yangilanishlari saqlanadi, hatto keyingi safar tizim buzilsa ham.

Tranzaktsiyani tiklash.

Tranzaktsiya BEGIN TRANSACTION bayonotining muvaffaqiyatli bajarilishidan boshlanadi) va COMMIT yoki ROLLBACK ko'rsatmalarining muvaffaqiyatli bajarilishi bilan yakunlanadi. COMMIT bayonotida "majburlash nuqtasi" deb nomlangan (tijorat mahsulotlarida "sinxronlashtirish nuqtasi" deb ham ataladi. Tuzatish nuqtasi mantiqiy ish birligining oxiriga to'g'ri keladi va shuning uchun ma'lumotlar bazasi mos keladigan (yoki bo'ladigan) nuqtaga to'g'ri keladi). ROLLBACK buyrug'ini bajarish, yana ma'lumotlar bazasini BEGIN TRANSACTION operatsiyasi davomida bo'lgan holatga qaytaradi, ya'ni oldingi majburiyat nuqtasi.

Nuqtalarni mahkamlash holatlari:

1. Oldingi tuzatish nuqtasi o'rnatilgandan beri dastur tomonidan qilingan barcha yangilanishlar, ya'ni doimiy bo'lib qoling. Ishlayotgan vaqtda bunday barcha yangilanishlarni faqat sinov versiyasi sifatida ko'rib chiqish mumkin (masalan, ular bajarilmasligi mumkin , degan ma'noda). Ruxsat etilgan yangilanish bir marta saqlanib qolishi kafolatlanadi (bu "sobit" tushunchasining ta'rifi).
2. Ma'lumotlar bazasining barcha joylashuvi joylashuvi yo'qoldi va barcha blokirovka qilingan qulflar bajarildi Ma'lumotlar bazasini joylashuvi har qanday vaqtda dastur odatda ma'lum bir tutqichlarga yo'naltirilganligini anglatadi. Ushbu manzilga kirish imkoniyati yo'qoladi.

Shunday qilib, tizim tranzaktsiyani aniq ravishda orqaga qaytarishi mumkin - masalan, foydalanuvchi ishlaydigan dasturning buyrug'i bilan va noaniq ravishda - biron bir dastur uchun, biron sababga ko'ra tranzaktsiyaga kiritilgan operatsiyalarni rejalashtirish tugallanmagan bo'lsa.

Bundan ko'rinib turibdiki, tranzaktsiyalar nafaqat ishning mantiqiy birliklari, balki operatsiyalar bajarilmasa tiklanish birliklari hamdir. Tranzaktsiya muvaffaqiyatli amalga oshirilgandan so'ng, tizim yangilanishlar ma'lumotlar bazasida doimiy ravishda o'rnatilishini ta'minlaydi, garchi keyingi vaqtda tizim ishdan chiqsa. Muvaffaqiyatli COMMITdan so'ng tizim ishdan chiqishi mumkin, ammo yangilanishlar ma'lumotlar bazasiga jismoniy yozilishidan oldin (ular hali ham RAM buferida qolishlari mumkin va shu sababli tizim qulashi paytida yo'qolishi mumkin). Agar bu sodir bo'lsa ham, tizimni qayta yuklash protsedurasi ushbu yangilanishlarni ma'lumotlar bazasiga o'rnatishi kerak va ro'yxatdan o'tish faylidagi tegishli yozuvlarni ko'rib chiqadi. Bundan kelib chiqadiki, jurnal fayllari COMMIT operatsiyasi tugashidan oldin jismoniy yozilishi kerak. Jurnal faylini saqlashning ushbu muhim qoidasi jurnal protokoli deb nomlanadi (ya'ni, operatsiya

bajarilishidan oldin yozib olinadi). Shunday qilib, qayta yuklash protsedurasi har qanday muvaffaqiyatli bajarilgan operatsiyalarni tiklashi mumkin, garchi ularning yangilanishlari tizim qulashidan oldin jismoniy qayd qilinmagan bo'lsa ham. Shuning uchun, yuqorida aytib o'tilganidek, operatsiya haqiqatan ham tiklanish birligi.

Moslik

«Moslik» atamasi ma'lumotlar bazasida bir vaqtning o'zida bir xil ma'lumotlarga kirish huquqiga ega bo'lgan ko'plab operatsiyalarni qayta ishlash imkoniyatini anglatadi. Bunday tizimda ziddiyatlarsiz parallel operatsiyalarni to'g'ri qayta ishlash uchun, kelishuvni boshqarishning ba'zi usullaridan foydalanish kerak.

Qarama-qarshilikni boshqarishning har bir usuli ma'lum bir muammoni hal qilish uchun mo'ljallangan. Biroq, to'g'ri tuzilgan tranzaktsiyalarni qayta ishlash paytida ba'zi operatsiyalar o'rtasidagi o'zaro aralashuv tufayli noto'g'ri natijaga olib keladigan vaziyatlar yuzaga keladi. (Shuni inobatga olingki, xalaqit beruvchi operatsiya o'zi to'g'ri bo'lishi mumkin. Noto'g'ri yakuniy natija ikkita to'g'ri operatsiyadan nazoratsiz almashinish tufayli kelib chiqadi). Parallel ravishda ishlov berishda uchraydigan asosiy muammolar quyidagilar:

1. yangilash natijalarini yo'qotish muammosi;
2. mustaqillik muammosi
3. mos kelmaydigan tahlil muammosi.

Yangilanish natijalarini yo'qotish muammosi.

11.1-rasmda keltirilgan vaziyatni ko'rib chiqamiz. Ushbu talqinda: tranzaksiya t1 vaqtida bir nechta p-ni oladi; tranzaksiya B t2 vaqtidagi ba'zi p sonlarni oladi; tranzaktsiya A t3 vaqtidagi ba'zi bir p (yangilanadi t1 vaqtiga asoslangan qiymatlar bo'yicha); tranzaktsiya B t4 vaqtidagi bir xil p-ni yangilaydi (t2 vaqtidagi qiymatlarga asoslanib, t1 vaqtidagi qiymatlar asosida). Biroq, A bitimida bajarilgan yangilanish operatsiyasining natijasi yo'qoladi, chunki t4 vaqtida u hisobga olinmaydi va shuning uchun B tranzaktsiyasi tomonidan bajarilgan yangilash operatsiyasi "bekor qilinadi".

Tranzaksiya A	Vaqt	Tranzaksiya B
p kortejini o’zgartirish	t1	-
-	t2	p kortejini o’zgartirish
p kortejini yangilash	t3	-
-	t4	P kortejini yangilash

1.1- rasm. A bitimida bajarilgan yangilanish natijalarining t4 vaqtidagi yo'qotishlar.

Aniqlanmagan qaramlik muammosi.

Agar biron-bir operatsiya hozirgi vaqtda boshqa tranzaktsiya tomonidan yangilanadigan ma'lum bir qismni qaytarib olsa (yoki undan ham yomoni, yangilansa), lekin bekor qilinmagan bog'liqlik muammosi paydo bo'ladi. Shunday

qilib, agar yangilanish tugallanmasa, u hech qachon bajarilmasligi ehtimoli bor (Bundan tashqari, bunday holatda, tranzaktsiyani bekor qilish bilan ulanishning avvalgi holatiga qaytarish mumkin). Bunday holda, birinchi tranzaktsiyada endi mavjud bo'lmagan ma'lumotlar ishtirok etadi. Ushbu holat 11.2 –rasmda ko'rsatilgan.

Birinchi misolda (11.3-rasm), t2 vaqtida A bitim muvaffaqiyatsiz yangilanishga duch keladi (bu bajarilmagan o'zgarish deb ham ataladi). Keyin ushbu yangilanish t3 vaqtida bekor qilinadi. Shunday qilib, A bitim t2 vaqtidagi p qiymatining ma'lum qiymatiga ega ekanligi haqidagi yolg'on taxmin asosida amalga oshiriladi, aslida u t1 vaqtida mavjud bo'lgan ma'lum qiymatga ega. Natijada, A bitimi bajarilgandan so'ng noto'g'ri natija olinadi. Bundan tashqari, B bitimining bajarilishini bekor qilish B bitimining aybi bilan yuzaga kelmasligi mumkin, ammo, masalan, tizimning ishdan chiqishi natijasida. (Bu vaqtga kelib, A bitimini bajarish allaqachon yakunlangan bo'lishi mumkin va shu sababli tizimning ishdan chiqishi A bitimining bekor qilinishiga olib kelmaydi).

Tranzaksiya A	Vaqt	Tranzaksiya B
-	t1	p kortejini yangilash
p kortejini yangilash	t2	-
-	t3	Tranzaksiya jarayonini boshlash

11.2-rasm. A o'tkazmasi t2 vaqtidagi bajarilmagan o'zgarishlarga bog'liq bo'ladi.

11.2-rasmda ko'rsatilgan ikkinchi misol. 11/3, boshqa vaziyatni tasvirlaydi. A tranzaktsiyasi nafaqat t2 vaqtidagi o'zgarishlarga bog'liq bo'ladi, balki t3 vaqtida ham yangilanish natijasi yo'qoladi, chunki t3 vaqtidagi B tranzaktsiyasini bekor qilish t1 vaqtidagi boshlang'ich qiymatini asl qiymatiga qaytaradi. Bu yangilanish natijalarini yo'qotish muammosining yana bir versiyasi.

Tranzaksiya A	Vaqt	Tranzaksiya B
-	t1	p kortejini yangilash
p kortejini yangilash	t2	-
-	t3	Tranzaksiya jarayonini to’xtatish

11.3-rasm. Transaction A t2 vaqtidagi sezilarli o'zgarishlarni yangilaydi va ushbu yangilanishning natijalari t3 vaqtida yo'qoladi.

Mos kelmaydigan tahlil muammosi.

11.4-rasmda A va B bitimlari ko'rsatilgan bo'lib, ular hisob raqamlari bilan bog'lash uchun amalga oshiriladi (11.1-jadval). Shu bilan birga, A bitim qoldiqlarni yig'adi, B bitimi 10 miqdorini 3-hisobvaraqdan 1-raqamga o'tkazadi. A bitimi natijasida olingan 110 natija aniq noto'g'ri va agar u ma'lumotlar bazasida qayd etilgan bo'lsa, unda nomuvofiqlik muammosi paydo bo'lishi mumkin. Bunday holda, ular A bitim mos kelmaydigan holat bilan uchrashdi va uning asosida mos

kelmaydigan tahlil o'tkazildi, deyishadi. Ushbu misol va oldingisi o'rtasidagi quyidagi farqga e'tibor bering: biz A bitimining B bitimiga bog'liqligi haqida gapirmayapmiz, chunki B tranzaktsiyasi barcha hisob-kitoblarni 3-chi operatsiyadan oldin amalga oshirdi.

11.1-jadval. Tranzaktsiyalar oldidagi hisob qoldiqlari.

Hisob	1-QISM	2-HISOB	3-HISOB
Qolganlari	40	50 ga teng	30

Tranzaksiya A	Vaqt	Tranzaksiya B
Hisob1:kartejini almashtirish SUMMA = 40	t1	-
Hisob1:kartejini almashtirish SUMMA = 90	t2	-
-	t3	Hisob3: kartejini almashtirish
-	t4	Hisob 3: kartejini yangilash
-	t5	30 → 20
-	t6	Hisob1: kartejini almashtirish
-	t7	Hisob 1: kartejini yangilash
Hisob3:kartejini almashtirish SUMMA = 100(120 ga emas)	t8	40 → 50
		Tranzaksiya jarayonini yopish

11.4-rasm. Amaliy A mos kelmaydigan tahlilni amalga oshirdi.

11.3 Blok tushuncha

Yuqorida tavsiflangan muammolar blokirovka deb nomlangan parallel jarayonlarni boshqarish uslubi yordamida hal qilinishi mumkin. Uning asosiy g'oyasi juda sodda: operatsiyani amalga oshirishda biron bir ob'ekt (odatda ma'lumotlar bazasi tupi) oldindan aytib bo'lmaydigan darajada o'zgarishi va ushbu operatsiyani bilmagan holda (odatdagidek) bunday ob'ekt blokirovka qilinishi kerak. Shunday qilib, blokirovkalashning ta'siri "ushbu ob'ektga boshqa operatsiyalardan kirishni blokirovka qilish" dir, bu ushbu ob'ektni oldindan aytib bo'lmaydigan o'zgarishlarni oldini olish demakdir. Shunday qilib, birinchi operatsiya qayta ishlangan ob'ekt kerak bo'lguncha barqaror holatda turishini hisobga olib, barcha kerakli ishlov berishni bajarishga qodir.

1. Aytaylik, tizim ikkita blokni qo'llab-quvvatlaydi: o'zaro bloklanmagan blok (eksklyuziv qulf), X-blokirovkalari (X-lock - eksklyuziv qulflar) va S-lock (S-lock - Birgalikda bloklash) deb nomlangan o'zaro kirish bilan qulf.

Eslatma. X- va S-bloklar ba'zan mos ravishda yozish va o'qish bloklari deb

ataladi. X-va S-bloklar yagona mumkin bo'lgan deb faraz qilaylik, garchi boshqa turdagi bloklar tijorat tizimlarida mavjud bo'lsa. Bundan tashqari, tutqichlar "qulflanadigan ob'ekt" ning yagona turi, deylik, lekin tijorat tizimlarida boshqa ob'ektlar bloklanishi mumkin. Bloklash mexanizmining ishlashi quyida ko'rsatilgan.

2. o'zaro foydalanish imkoniyati bo'lmagan bitim A bloklar nizomning r bo'lsa (X - blokirovka), keyin bu shamlardan B p blokirovka bilan yana bir bitimni talab bekor qilinadi.

3. Agar A tranzaktsiyasi o'zaro ulanish imkoniyati bo'lgan p-ni bloklasa (S - lock), unda:

   a. X ustida bir operatsiya B talab - blok shamlardan rad qilish;

   b. B S bo'yicha bitim dan talab - Qulf nizomning r qabul qilinadi (masalan, bitim B ham R S yordamida nizomning to'sadi - Qulf).

Ushbu qoidalar 11.5-rasmda ko'rsatilgan moslik matritsasi shaklida keltirilgan va uni quyidagicha izohlanadi. Bir nechta p-ni ko'rib chiqamiz va A trayle p har xil turdagi bloklar bilan bloklanadi (bu S va X tegishli belgilar bilan ko'rsatiladi va tire bilan blokirovka qilinmaydi). Aytaylik, B operatsiyasining 5-rasmdagi matritsaning birinchi chap ustunida ko'rsatilgan p bandining blokirovkasini talab qiladi (to'liqligi uchun jadvalda “blok yo'q” holati ko'rsatilgan). Matritsaning boshqa hujayralarida N belgisi ziddiyatli vaziyatni bildiradi (B bitim bo'yicha so'rov qondirilmaydi va bu operatsiya kutish holatiga tushadi), Y esa to'liq mos keladi (B bitimining so'rovi qondiriladi). Shubhasiz, bu matritsa nosimmetrikdir.

	X	S	-
X	N	N	Y
S	N	Y	Y
-	Y	Y	Y

11.5-rasm. X- va S-blokirovkalash uchun moslik matritsasi.

Biz ma'lumotlarga kirish protokoli bilan tanishtiramiz, bu X va S bloklarini joriy etishga asoslanib, mos keladigan muammolarni oldini oladi.

1. O'tkazgichni olish uchun tuzilgan operatsiya avval ushbu tugmachaga S - blokirovkasini kiritishi kerak .

2. Multiplayni yangilash uchun mo'ljallangan tranzaktsiya birinchi navbatda X-blokirovkasini kiritishi kerak. Boshqacha qilib aytganda, agar, masalan, S-lock allaqachon ulanish uchun ekstraktsiya qilish / yangilash turlarining ketma-ketligi uchun o'rnatilgan bo'lsa, uni X - lock bilan almashtirish kerak.

Ko'rgan: asosiy odatda masalan, bir so'rov "ekstrakti nizomning" S-blok va iltimosiga bir shubhasiz so'rov uchun, so'zsiz bitimda belgilangan "yangilash nizomning" ga - X bilan yopilgan spam - tegishli nizomning to'sib. Bundan tashqari, "yangilash" atamasi (avvalgidek) yangilash operatsiyalariga qo'shimcha ravishda, operatsiyalarni qo'shish va o'chirishni ham anglatadi.

1. Agar A bitim tomonidagi boshqa blokirovka bilan ziddiyat tufayli B bitim tomonidagi talab qilingan blokirovka rad etilsa, u holda B bitimi bekor holatga o'tadi. Bundan tashqari, "B" tranzaksiyasi kutish holatida bo'ladi va A bitimida belgilangan blokirovka chiqarilguncha bo'ladi va tizim B bitimining cheksiz kutish holatini yo'q qilish yo'llarini ko'rsatishi shart.

2. X-qulflar bitim oxirigacha saqlanadi ("operatsiya" tugaguncha yoki "bajarishni bekor qilguncha "). S - bloklar, odatda, shu paytgacha saqlanadi.

Qarama-qarshi muammolarni hal qilish.

Bloklash mexanizmidan foydalanib, parallellik muammolarini hal qilishni ko'rib chiqamiz.

Yangilanish natijalarini yo'qotish muammosi.

11.6-rasmda ko'rsatilgan jarayonning o'zgartirilgan versiyasi ko'rsatilgan. Alternativ operatsiyalar uchun bloklash protokolidan foydalanishni hisobga olgan holda, T3 vaqtidagi A bitim uchun yangilanish operatsiyasi bajarilmaydi, chunki bu ko'p sonli p uchun X-blokirovkali so'rov bo'lib, B so'rovi B bitimi bilan o'rnatilgan S-blokirovkaga ziddir. Shunday qilib, A bitimi ketadi. kutish holati. Shunga o'xshash sabablarga ko'ra, B bitimi t4 vaqtidagi kutish holatiga o'tadi. Yangilanishlar endi yo'qoladi, ammo yangi muammo yuzaga keladi - cheksiz kutish yoki to'xtatish. Ushbu muammoni hal qilish yo'llari quyida muhokama qilinadi.

Tranzaksiya A	Vaqt	Tranzaksiya B
p kortejini sozlash (S-topshiriqi p uchun yopiq) -	t1	-
	t2	p kortejini sozlash S-topshiriqi p uchun yopiq)
p kortejini yangilash (X-topshiriq p uchun yopiq) Kutish	t3	-
Kutish	t4	p kortejini yangilash (X-topshiriq p uchun yopiq) Kurish

11.6-rasm. Yangilanishlar yo'qolmasa ham, t4 vaqtida qiyin vaziyat yuzaga keladi.

Aniqlanmagan qaramlik muammosi.

11.7-rasmda, 11.8-rasmda avval ko'rsatilgan misollar o'zgartirilgan rasmda keltirilgan. Mos ravishda ular yuqorida tavsiflangan bloklash protokoli bo'yicha uzilishlarni namoyish qiladilar. Vaqt o'tishi bilan t2 bo'yicha operatsiya amalga oshiriladi (11.7-rasmda olish va 11.8-rasmda yangilanish). Gap shundaki, bu "p" bandi uchun bloklash vazifasi bo'lgan yashirin talab va bu so'rov B bitimi tomonidan o'rnatilgan X-blokirovkaga ziddir. Shunday qilib, A bitimi bajarilish to'xtatilgunga qadar kutish holatiga o'tadi. B bitimi (operatsiya tugashi yoki B bitimining

bajarilishini bekor qilishdan oldin). Keyin B bitimi bilan o'rnatiladigan blok ochiladi va A bitim bajarilishi mumkin. Bundan tashqari, A bitimi ba'zi bir belgilangan qiymat bilan shug'ullanadi (yoki B bitimi amalga oshirilishidan oldin mavjud bo'lgan yoki B bitimi bajarilgandan keyin olingan). Qanday bo'lmasin, A bitimi endi o'tkazib yuborilmagan yangilanishga bog'liq emas.

Tranzaksiya A	Vaqt	Tranzaksiya B
-	t1	p kortejini yangilash (X-topshiriq p uchun yopiq)
p kortejini sozlash (S-topshiriqi p uchun yopiq) Kutish	t2	-
	t3	Tranzaksiya jarayoni to’xtatildi
Jami: p kortejini yangilash (S-topshiriq p uchun yopiq)	t4	(X-chiqarilishi p uchun yopiq)

11.7-rasm. A tranzaktsiyasi t2 vaqtidagi o'zgarishsiz operatsiyalarni bajarishdan himoyalangan.

Tranzaksiya A	Vaqt	Tranzaksiya B
-	t1	p kortejini yangilash (X-topshiriq p uchun yopiq)
p kortejini sozlash (X-topshiriqi p uchun yopiq) Kutish	t2	-
	t3	Tranzaksiya jarayoni to’xtatildi
Jami: p kortejini yangilash (X-topshiriq p uchun yopiq)	t4	(X-chiqarilishi p uchun yopiq)

11.8-rasm. A tranzaktsiyasi t2 vaqtidagi o'zgarishsiz operatsiyalarni bajarishdan himoyalangan.

Mos kelmaydigan tahlil muammosi.

11.9-rasmda blokirovka protokoli bo'yicha ro'yxatga olingan bitimlarni o'z ichiga olgan munosabatlarning o'zgartirilgan versiyasini (11.4-rasm) ko'rsatadi. Vaqt t6 vaqtida B tranzaktsiyasi uchun yangilash operatsiyasi bajarilmaydi. Gap shundaki, bu COUNT-bosqich 1 uchun X-lock vazifasi bo'lgan aniq so'rov bo'lib, bu talab A bitimida o'rnatilgan S-blokirovkaga ziddir. Shunday qilib, B bitimi kutish holatiga o'tadi. Xuddi shunday, t7 vaqtida A tranzaksiya uchun olish operatsiyasi bajarilmaydi. u nizomning ACCOUNT 3 uchun S-to'sib vazifaga kelgan bir shubhasiz talab, bu talab X zid ekanligi - lock allaqachon bu bitim B.

Shunday qilib, bitim A kutish holatini kiradi. Shuning uchun, blokirovka qilish bitta muammoni (aniqrog'i, mos kelmaydigan tahlil muammosini) hal qilishga yordam beradi, ammo boshqa muammoni (aniqrog'i, muammoning echimini) echishga olib keladi.

Hisob1, 40	Hisob2, 50	Hisob 3, 60
Hisob1:kartejini almashtirish (S – topshiriq Hisob 1 uchun yopiq)	t1	-
SUMMA = 40		-
Hisob1:kartejini almashtirish	t2	
SUMMA = 90		
-		
-	t3	Hisob3: kartejini
	t4	almashtirish
-		(S – topshiriq Hisob 3
-	t5	uchun yopiq)
	t6	Hisob 3: kartejini
-		yangilash
Hisob3:kartejini almashtirish		(X – topshiriq Hisob 3
(S – topshiriq Hisob 3 uchun	t7	uchun yopiq)
yopiq)	t8	30 → 20
Kutish		Kutish
		Kutish
		Kutish

11.9-rasm. Mos kelmaydigan tahlil muammosi hal qilindi, ammo t7 vaqtida qiyin vaziyat yuzaga keladi.

11.4 Muammoli vaziyat

Yuqorida ko'rsatilgandek, dublonlarga parallel ishlov berish paytida yuzaga keladigan uchta asosiy muammoni hal qilish uchun blokdan foydalanish mumkin. Afsuski, qulflardan foydalanish yana bir muammoga olib keladi - bloklash. 11.10 - rasmda ushbu muammoning umumlashtirilgan misolini ko'rsatadi, unda p1 va p2 har qanday blokirovka qilinadigan ob'ektlarni, ixtiyoriy ma'lumotlar bazasi tirgaklarini va "o'zaro kirishsiz blokirovka" kabi iboralar o'zaro kirishsiz blokni o'rnatish bilan har qanday operatsiyani ifodalaydi, aniq va o'rnatiladi.

Tranzaksiya A	Vaqt	Tranzaksiya B
p1ruxsat berish uchun yopiq	t1	-
-	t2	p1ruxsat berish uchun yopiq
		-
p2 ruxsat berish uchun yopiq	t3	p2 ruxsat berish uchun yopiq

Kutish Kutish	t4	Kutish

11.10-rasm. Muammoni hal qilishning misoli.

Bir vaqtning o'zida ikkita yoki undan ortiq tranzaktsiyalar kutilmoqda, bunda har bir operatsiya boshqa operatsiya to'xtashini kutish bilan tugaydi.

Muammoni aniqlash uchun siz kutish holatining diagrammasida siklni aniqlab olishingiz kerak, ya'ni "boshqa operatsiyalarni tugatilishini kutayotgan bitimlar" ro'yxatiga kirishdan chiqish yo'lini blokirovka qilingan bitimlardan birini jabrlanuvchi sifatida tanlash va uning bajarilishini bekor qilish kiradi. Shunday qilib, blok undan chiqariladi va boshqa operatsiyani bajarish qayta tiklanishi mumkin.

Amalda, barcha tizimlar qiyin vaziyatni aniqlay olmaydi. Masalan, ularning ba'zilari tranzaktsiyalarni bajarish vaqtidan foydalanadilar va agar biron-bir belgilangan vaqt ichida bitim amalga oshirilmagan bo'lsa, bloksiz xabar keladi.

Siz jabrlanuvchiga tegishli tranzaktsiya "noto'g'ri" deb tan olinganligiga va "o'z xatosi tufayli emas" bekor qilinishiga e'tibor berishingiz kerak. Ba'zi tizimlar operatsiyani boshidanoq avtomatik ravishda qayta boshlanishini ta'minlaydi, agar bu buzuqlikka olib kelgan holatlar yana takrorlanmasa. Va boshqa tizimlarda ushbu operatsiyani bajarish bilan bog'liq bo'lgan dasturga ushbu vaziyatni hal qilish uchun "jabrlanuvchini bitimi" haqida xabar yuboriladi, amaliy dasturlash nuqtai nazaridan, ushbu yondashuvlarning birinchisi afzalroqdir. Ammo shunga qaramay, ushbu muammoni har doim foydalanuvchi nuqtai nazaridan hal qilish tavsiya etiladi.

Tartibga solish qobiliyati.

Berilgan bitimlar to'plamining o'zaro bajarilishi, agar u buyurtma qilingan bo'lsa to'g'ri bo'ladi, ya'ni. u bajarilganda, xuddi shu operatsiyalarni ketma-ket bajarayotganda xuddi shunday natija olinadi. Quyidagi kuzatuvlar ushbu fikrni tasdiqlashga yordam beradi:

1. Agar ma'lumotlar bazasi bitta izchil holatdan boshqa izchil holatga o'tilsa, individual bitimlar haqiqiy deb hisoblanadi.
2. Bitimlarni ketma-ket ketma-ketlikda bajarish ham to'g'ri. Bundan tashqari, "har qanday ketma-ketlik" iborasi bir-biridan mustaqil bo'lgan bitimlar ishlatilishini anglatadi.
3. Tranzaktsiyalarning o'zaro bog'liqligi, agar u ba'zi bir ketma-ket bajarishga teng bo'lsa, ya'ni to'g'ri bo'ladi, ya'ni uni tartibga keltirish kerak bo'lsa.

Yuqoridagi misollarga qaytsak (11.1-rasm - 11.4-rasm), shuni ta'kidlash mumkinki, har bir holatda muammo shundaki, bitimlarning o'zaro tartibda bajarilishi tartiblashtirilmagan, ya'ni. u avval A bitimni, so'ng B operatsiyani yoki avval B bitimni, so'ngra A bitimni bajarishga teng kelmadi.

Berilgan bitimlar to'plami uchun ularni amalga oshirishning har qanday tartibi (o'zaro yoki boshqa) ishga tushirish jadvali deb ataladi. Bitimlarni ketma-ket

almashtirishsiz ketma-ket bajarilishi ketma-ket ishga tushirish jadvali, nomuvofiq tranzaktsiyani esa navbatma-navbat yoki mos bo'lmagan boshlash jadvali deb nomlanadi. Ikkala grafik, agar ma'lumotlar bazasining dastlabki holatidan qat'iy nazar bir xil natijani beradigan bo'lsa, ekvivalent deb ataladi. Shunday qilib, agar ba'zi ketma-ket ishga tushirish jadvaliga to'g'ri keladigan bo'lsa, ishga tushirish jadvali to'g'ri (ya'ni buyurtma berishga imkon beradi).

Bir xil tranzaktsiyalarni o'z ichiga olgan ketma-ket ikki xil ketma-ket ishga tushirishda siz mutlaqo boshqa natijalarga erishishingiz mumkin. Shu sababli, bir xil tranzaktsiyalar bilan ikkita turli xil alternativ boshlash jadvalining bajarilishi, shuningdek haqiqat sifatida qabul qilinishi mumkin bo'lgan turli xil natijalarga olib kelishi mumkin.

Ikki fazali qulflash teoremasi (ikki fazali fiksatsiya protokoli bilan bog'liq emas) quyidagicha tuzilishi mumkin.

Agar barcha operatsiyalar "ikki fazali blokirovka protokoli" ga rioya qilsa, barcha mumkin bo'lgan alternativ boshlash jadvallari uchun buyurtma berish imkoniyati mavjud.

Bunday holda, ikki fazali blokirovka protokoli, o'z navbatida, quyidagicha tuziladi.

1. Ba'zi bir ob'ekt bilan biron bir operatsiyani amalga oshirishdan oldin (masalan, ma'lumotlar bazasi tupi bilan), tranzaktsiya bu tupni blokirovka qilishi kerak.
2. Qulfni bo'shatgandan so'ng, tranzaktsiya boshqa qulflarni qo'ymasligi kerak.

Shunday qilib, ushbu protokolga bo'ysunadigan tranzaktsiya ikki bosqich bilan tavsiflanadi: bloklash vazifasi va bloklash fazasi.

Buyurtmaning xarakteristikasi quyidagicha ifodalanishi mumkin. Agar A va B buyurtmalarni amalga oshirishga imkon beradigan ma'lum bir ishga tushirish jadvalining ikkita ikkita bitimi bo'lsa, u holda A mantiqiy ravishda B, yoki B mantiqiy ravishda A, ya'ni oldin, ya'ni, yoki B bitim A bitimining natijalarini ishlatadi yoki A bitim natijalarini ishlatadi (Agar A, t, p, q, ... r operatsiyalari yangilanadi va B bitimi bu tutqichlarni kirish sifatida ishlatsa, unda barcha yangilanganlar ham ishlatiladi a dizilerini yoki dizilerini butunlay bir bitim oldin yangilangan emas, balki bir aralashmasi). Aksincha, Run jadvali yaroqsiz va bitim natijasida birinchi mos kelmaydi, agar buyurtma berish mumkin emas bitta operatsiyaning bajarilishini bir, so'ngra ranzaktsii B, birinchi bitim bajarish B, keyin bitim A.

Hozirgi vaqtda resurslarga bo'lgan ehtiyojni pasaytirish va shu bilan real tizimlarda samaradorlikni va o'tkazuvchanlikni oshirish uchun odatda ikki fazali tranzaktsiyalardan emas, balki "qulfni muddatidan oldin bo'shatish" (bitimni bekor qilish operatsiyasidan oldin ham) operatsiyalari va bir nechta qulflarni o'rnatish uchun foydalaniladi. Ammo shuni tushunish kerakki, bunday operatsiyalardan foydalanish katta xavfga ega. Darhaqiqat, A ikki fazali tranzaktsiyadan foydalanilganda, ushbu tizimda B bilan almashinadigan boshqa hech qanday tranzaktsiya bo'lmaydi (aks holda tizimda noto'g'ri natijalar olinishi mumkin).

Tranzaksiyani izolyatsiya qilish darajasi.

Izolyatsiya darajasi atamasi, taxminan aytganda, parallel operatsiyalarning berilgan operatsiyaga aralashish darajasini tavsiflash uchun ishlatiladi. Ammo buyurtma berish imkoniyatini yaratishda hech qanday aralashishga yo'l qo'yilmaydi, boshqacha qilib aytganda, izolyatsiya darajasi maksimal bo'lishi kerak. Biroq, yuqorida ta'kidlab o'tilganidek, haqiqiy tizimlarda turli sabablarga ko'ra odatda maksimal darajadan past bo'lgan izolyatsiya darajasida ishlaydigan operatsiyalarga ruxsat beriladi.

Izolyatsiya darajasi odatda bitimning ba'zi mulki sifatida qabul qilinadi. Haqiqiy ma'lumotlar bazasida turli xil izolyatsiya darajalarini amalga oshirish mumkin.

Bundan tashqari, bog'lanishga qo'shimcha ravishda, boshqa ma'lumotlar bloklari blokirovka qilinishi mumkin, masalan, butun munosabat, ma'lumotlar bazasi yoki (qarama-qarshi tabiatning namunasi) berilgan to'plam ichidagi atribut qiymati.

11.5 So’rovlar yaratish va qayta ishlash

SQL operatsiyalarni tegishli ravishda ishlash va qaytarish uchun COMMIT va ROLLBACK operatsiyalarini qo'llab-quvvatlaydi.

Maxsus SET TRANSACTION so'rovi, kirish rejimi va izolyatsiya darajasi kabi boshlanishi kerak bo'lgan ba'zi xususiyatlarni aniqlash uchun ishlatiladi.

SQL til standarti aniq blokirovkani qo'llab-quvvatlamaydi (aslida blokirovka umuman aytilmaydi). SQL ko'rsatmalarini bajarishda bloklar aniq belgilanadi.

So'rovlarni optimallashtirish.

So'rovlarni optimallashtirish - bu 1) ma'lumotlar bazasini so'rovni bajarish rejasini qidirib topadigan ma'lumotlar bazasi funktsiyasi, 2) so'rovni bajarishda hisoblash resurslaridan foydalanishni qisqartirish uchun so'rovni va / yoki ma'lumotlar bazasi tuzilishini o'zgartirish jarayoni. Xuddi shunday natijani ma'lumotlar bazasi ma'lumotlar bazasida har xil usulda olish mumkin (so'rovlarni bajarish rejalari), ular ham resurslar narxida, ham bajarilish vaqtlarida sezilarli farq qilishi mumkin. Optimallashtirish vazifasi eng yaxshi usulni topishdir.

Relyatsion MBBTda so'rovlarni bajarishning eng maqbul rejasi - ma'lumotlar bazasining ma'lum bir joriy holati (tuzilishi va tarkibi) uchun hisoblash resurslaridan minimal foydalanish bilan bajarilishi mumkin bo'lgan relyatsion algebra operatorlarini boshlang'ich va oraliq munosabatlarga tatbiq etish ketma-ketligidir.

Hozirgi vaqtda maqbul rejani topish uchun ikkita ma'lum strategiya mavjud:

- qo'shilgan jadvallarning barcha o'zgarishlari, jadvallarga kirish usullari va qo'shilish turlarini baholagan holda shafqatsiz kuch (ya'ni variantlarning to'liq ro'yxati);
- cheklangan miqdordagi almashtirishlarni baholab, genetik algoritmga asoslanadi.

Shuningdek, ba'zi MBBTlar dasturchiga minimal ta'sirdan tortib, qaysi so'rov rejasidan foydalanishni aniq va aniq ko'rsatishga qadar turli xil darajadagi optimal rejani qidirishga aralashishga imkon beradi.

So'rovlarni bajarish rejalari ko'pgina omillar asosida taqqoslanadi (turli xil ma'lumotlar bazalarida amalga oshirish farqlanadi), shu jumladan:

- statistikadan olingan har bir jadvaldan olingan qatorlarning potentsial soni;
- indekslarning mavjudligi;
- birlashishni amalga oshirish qobiliyati (birlashtirish-qo'shilish);
- yozuvlar / jadvallar / indekslar bloklarini o'qish usuli.

Umuman olganda, ulanish ichki o'rnatilgan pastadir orqali amalga oshiriladi. Biroq, ushbu algoritm ixtisoslashgan algoritmlarga qaraganda samaraliroq bo'lishi mumkin. Masalan, agar birlashtirilgan jadvallar birlashtirilgan maydonlarda indekslarga ega bo'lsa yoki bitta yoki ikkala jadval xotirada saralanadigan darajada kichik bo'lsa, unda birlashishni amalga oshirish imkoniyati ko'rib chiqiladi.

Yuqorida ta'kidlab o'tilganidek, optimallashtirishning mohiyati jadvallarni o'zgartirishdan minimal xarajatlar funktsiyasini topishdir. Strategiyadan qat'i nazar, optimizator o'zboshimchalik bilan qayta tashkil etish xarajatlarini tahlil qilishi kerak, shu bilan birga qayta tahrirlash boshqa algoritm tomonidan ta'minlanadi. O'rganilgan permutatsiyalar to'plami butun permutatsiya maydonidan farq qilishi mumkin. Bunga asoslanib optimizatorning umumlashtirilgan algoritmi quyidagicha yozilishi mumkin:

```
Eng yaxshisini izlashda barcha rejalarni sanab o'tish:
Joriy jadval buyurtmasi: = boshlang'ich jadval
tartibini toping;
Eng yaxshi jadval buyurtmasi: = Mavjud jadval
buyurtmasi;
Eng kam xarajat: = Mumkin bo'lgan maksimal xarajat;
Bajaring
  Xarajat: = smeta qiymati (jadvallarning hozirgi
tartibi);
  Agar narx <Eng kam xarajat bo'lsa
    Eng yaxshi jadval buyurtmasi: = Mavjud jadval
buyurtmasi;
    Eng kam xarajat: = xarajat;
  End Agar;
   Joriy jadval buyurtmasi: = Keyingi jadval tartibini
toping;
Bye (Mavjud navbatdagi jadvallar mavjud);
```

Qo'pol kuch ishlatish strategiyasi. Nazariy jihatdan, shafqatsiz kuch strategiyasidan foydalanganda, so'rovni optimallashtiruvchi barcha dastlabki tanlangan jadvallarning permütasyon maydonini o'rganadi va har bir o'zgartirish uchun qo'shilishni bajarish xarajatlarini umumiy taqqoslaydi. Amalda, Tizim R ni ishlab chiqishda tadqiqot maydonini faqat chap tomonli ulanishlar bilan cheklash taklif qilindi, shunda so'rovni bajarishda jadvallardan biri doimo diskda namoyish

etiladi. O'ng tarafdagi bo'lmagan qo'shilishlarni tekshirish, agar qo'shilishlar jadvallari bir nechta tugunlarda joylashgan bo'lsa, mantiqiy bo'ladi.

Genetika algoritmiga asoslangan strategiya.

So'rovdagi jadvallar sonining ko'payishi bilan, mumkin bo'lgan o'zgartirishlar soni n ga o'sadi, shuning uchun ularning har birini baholash vaqti mutanosib ravishda oshadi. Bu ko'p sonli jadvallar asosida so'rovlarni optimallashtirishni muammoli qiladi. 1991 yilda ushbu muammoning echimini izlashda Kristin Bennett, Maykl Ferris, Yannis Ioannidis so'rovlarni optimallashtirish uchun genetik algoritmdan foydalanishni taklif qilishdi, bu esa chiziqli vaqt ichida suboptimal echim beradi.

Genetika algoritmidan foydalanganda permutatsiya maydonining faqat bir qismi o'rganiladi. So'rovda qatnashgan jadvallar xromosomalarga kodlangan. Ularda mutatsiyalar va xochlar amalga oshiriladi. Har bir iteratsiyada xromosomalarni tiklash jadvallarning mazmunli permutatsiyasi va minimal xarajatlar smetasini beradigan xromosomalarni tanlash uchun amalga oshiriladi. Tanlash natijasida oldingi iteratsiyaga nisbatan xarajat funktsiyasining ozroq qiymatini beradigan faqat xromosomalar qoladi. Shunday qilib, xarajatlar funktsiyasining mahalliy minimalarini o'rganish va topish sodir bo'ladi. Taxmin qilinishicha, global minimal eng yaxshi mahalliy minimal darajaga nisbatan muhim afzalliklarni ta'minlamaydi. Algoritm bir nechta iteratsiyalarni takrorlaydi, shundan so'ng eng samarali echim tanlanadi.

Statistikaga muvofiq har bir o'zgartirish uchun har bir jadval uchun indekslardan foydalanish imkoniyati baholanadi. Minimal smeta bilan aniqlash - so'rovni bajarish uchun yakuniy reja.

Talabni bajarish holatini aniqlash.

Turli xil MBBT amalga oshirish tizimning turli qismlarining resurs xarajatlarini hisobga olishi mumkin. Oracle 9i ma'lumotlar bloklarini o'qilishini, protsessor sikllarining sonini va qo'shimcha disk xotirasining miqdorini hisobga oladi. Bundan tashqari, ushbu ko'rsatkichlar normallashtiriladi va bir blokli o'qishlar soniga bo'linadi [7]. Xarajatlar formulasi quyidagicha:

$$Cost = (\#SRds * sreadtim + \#MRds * mreadtim + \#CPUCycles / CPUSpeed)/ sreadtim$$

qayerda

#SRds - bitta blokli disk o'qilishining taxminiy soni;

#MRds - ko'p blokli disk o'qilishining taxminiy soni;

#CPUCycles - protsessor tomonidan bajariladigan operatsiyalar sonini asemptomatik ravishda aniq hisoblash;

#sreadtim - bitta blokli diskni o'qishning o'rtacha vaqti;

#mreadtim - bitta ko'p blokli diskni o'qishning o'rtacha vaqti;

#CPUSpeed - protsessor tomonidan birlik vaqtiga bajarilgan operatsiyalar soni.

Savollar:

1. Tranzaksiya nima?
2. Asosiy ACID xususiyatlarini sanab bering .

3. Tranzaktsiyalarni sinxronizatsiya qilish mexanizmi qanday amalga oshiriladi?
4. S- blok nima ?
5. So‘rovni bajarish vaqti qanday hisoblanadi?

Adabiyotlar:

1. Thomas Connolly, Carolyn Begg – Database systems. A practical Approach to Design, Implementation and Management. 4th Edition – Addison Wesley 2005 – 1373p.
2. C. J. Date – An Introduction to Database Systems – Addison-Wesley Professional – 2003 – 1024 p

XII Bob. Ma'lumotlar bazasini administratorlash va xavfsizligi ta’minlash

12.1 Ma'lumotlar bazasini tiklash

Ba'zan kompyuter tizimlari ishlamay qoladi. Buning sabablari ko'p bo'lishi mumkin: uskunaning ishdan chiqishi, dasturlarning kamchiliklari, qo'lda bajariladigan protseduralarning tavsifidagi noaniqliklar va insonning noto'g'ri xatti-harakatlari. Ushbu turdagi barcha xatolar ma'lumotlar bazasida qo'llanilishi mumkin va bo'lishi mumkin. Ma'lumotlar bazasi ko'plab odamlar tomonidan baham ko'rilganligi va ko'pincha tashkilot faoliyatining muhim elementi bo'lganligi sababli uni imkon qadar tezroq tiklash juda muhimdir.

Bu biz uchun bir nechta qiyinchiliklarni tug'diradi. Birinchidan, ish nuqtai nazaridan, ish davom etishi kerak. Masalan, buyurtmalarni bajarish, moliyaviy operatsiyalarni bajarish va qadoqlash varaqalarini tuzish qo'lda tashkil etilishi kerak. Keyinchalik, ma'lumotlar bazasi dasturi yana ishlay boshlaganda, siz yangi ma'lumotlarni kiritishingiz mumkin. Ikkinchidan, kompyuter tizimidan mas'ul xodimlar uni iloji boricha tezroq tiklashlari va ishlamay qolishidan oldingi holatga imkon qadar yaqinroq bo'lishlari kerak. Uchinchidan, foydalanuvchilar tizim qayta ishlay boshlaganda nima qilish kerakligini bilishlari kerak. Ba'zi ma'lumotlarni qayta kiritishingiz kerak bo'lishi mumkin va foydalanuvchilar ular orqaga qaytishlari kerakligini bilishlari kerak.

Agar ishlamay qolsa, siz muammoni hal qila olmaysiz va uni qayta ishlashni davom ettira olmaysiz. Bir vaqtning o'zida hech qanday ma'lumot yo'qolmagan bo'lsa ham (barcha turdagi xotiralar o'zgaruvchan bo'lmasligi kerak, bu mutlaqo real bo'lmagan taxmin), sinxronizatsiya va qayta ishlashni rejalashtirish tizimning holatini aniq takrorlash uchun juda murakkabdir. To'liq to'xtatilgan joydan ishni davom ettirish uchun operatsion tizim juda katta miqdordagi ortiqcha ma'lumotlarni va uni qayta ishlashni talab qiladi. Oldinga siljish va barcha elektronlarni xato sodir bo'lgan vaqtda bir xil konfiguratsiyaga qaytarish mumkin emas. Ma'lumotlar bazasini tiklashda ikkita yondashuv mumkin: qayta ishlash va orqaga qaytarish.

Qayta ishlash orqali tiklash.

Qayta ishlashni aniq to'xtatilgan joydan boshlab qayta tiklash mumkin emasligi sababli, keyingi imkoniyat ma'lum bir nuqtaga qaytish va undan qayta

ishlashni davom ettirishdir. Buni amalga oshirishning eng oson usuli vaqti-vaqti bilan ma'lumotlar bazasining nusxasini yaratish (ma'lumotlar bazasini saqlash deb ataladi) va oxirgi nusxadan beri bajarilgan barcha operatsiyalarning yozuvlarini saqlash. Keyin, agar biron bir nosozlik yuzaga kelsa, xodimlar ma'lumotlar bazasini o'z rasmidan tiklashlari va barcha operatsiyalarni qayta bajarishlari mumkin.

Afsuski, bu oddiy strategiyani amalga oshirish mumkin emas. Birinchidan, tranzaktsiyalarni qayta bajarish xato yuzaga kelguncha davom etadi. Agar kompyuter band bo'lsa, tizim hech qachon asl holatiga tushib qolmasligi mumkin. Ikkinchidan, tranzaktsiyalarga parallel ravishda ishlov berishda hodisalar asenkron ravishda sodir bo'ladi. Biror kishining harakatlaridagi kichik tafovutlar - masalan, foydalanuvchi disketani biroz sekinroq qo'ysa yoki ilova so'roviga javob berishdan oldin elektron pochta xabarini o'qisa, bir vaqtning o'zida operatsiyalarni qayta ishlash tartibini o'zgartirishi mumkin. Shunday qilib, dastlab ushbu reysning so'nggi chiptasi A mijoziga berilgan bo'lsa ham, ikkinchi qayta ishlash paytida uni B mijozi oladi, shu sababli, qayta ishlash odatda parallel ishlov berish tizimidagi nosozliklar tufayli tiklanishning real rasmi emas.

Orqaga qaytarish orqali tiklash.

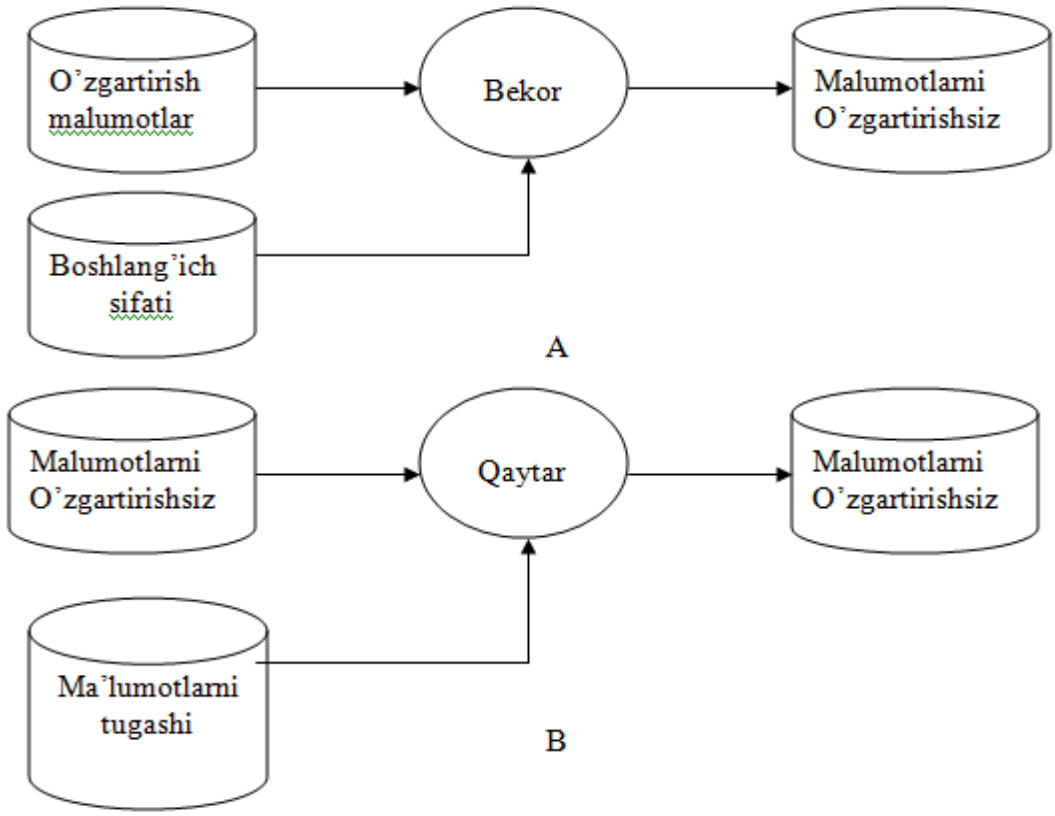
Qayta tiklashning ikkinchi usuli - bu ma'lumotlar bazasida vaqti-vaqti bilan nusxalar (suratlar) yaratish va oxirgi nusxadan boshlab ma'lumotlar bazasida amalga oshirilgan barcha o'zgarishlar jurnalini yuritish. Bunday holda, ishlamay qolganda, siz ikkita usuldan birini ishlatishingiz mumkin. Rollforward deb nomlangan birinchi usul ma'lumotlar bazasini saqlangan holatga qaytaradi, shundan so'ng barcha to'g'ri operatsiyalar bajariladi (Biz tranzaktsiyalarni qayta ishlamaymiz, chunki dasturlar tiklanish jarayonida ishtirok etmaydilar. Bularning barchasi jurnal yozuvlari bo'yicha ma'lumotlar bazasiga o'zgartirishlar kiritishdir).

Ikkinchi usul - qaytarish deb ataladi. Ushbu usul yordamida biz noto'g'ri yoki qisman bajarilgan operatsiyalar orqali ma'lumotlar bazasida kiritilgan o'zgarishlarni bekor qilamiz. Keyin, to'g'ri operatsiyalar qayta ishga tushiriladi, ular muvaffaqiyatsizlikka uchragan paytda amalga oshirildi.

Ushbu ikkala usul ham tranzaksiya natijalarini qayd etishni talab qiladi. Ushbu jurnalda ma'lumotlarga xronologik tartibda kiritilgan o'zgarishlar to'g'risida yozuvlar mavjud. Tranzaktsiya bajarilishidan oldin u yozib olinishi va jurnal bo'lishi kerak. Keyin, agar tizim operatsiyani jurnalga yozish va uni bajarish o'rtasidagi vaqt oralig'ida buzilsa, eng yomon holatda, biz muvaffaqiyatsiz bitimni qayd etamiz. Agar, aksincha, avval tranzaktsiyalar amalga oshirilsa, so'ngra jurnalga yozilgan bo'lsa, ma'lumotlar bazasiga o'zgartirishlar kiritilganda istalmagan tanlov mumkin, ammo bu o'zgarishlar haqida yozuvlar yo'q. Bunday holatda, tajribasiz foydalanuvchi allaqachon tugallangan bitimni qayta kiritishi mumkin.

Muvaffaqiyatsiz bo'lsa, jurnal bekor qilish uchun ham, qayta-qayta bajarish uchun ham qo'llaniladi (12.1-rasm). Tranzaktsiyalarni bekor qilish uchun jurnalda ma'lumotlar bazasini o'zgartirishdan oldin qilingan har bir yozuv (yoki sahifa) nusxasi bo'lishi kerak. Bunday yozuvlarga manba tasvirlari deyiladi (avvalgi rasmlar). Tranzaktsiyalar ma'lumotlar bazasiga kiritilgan barcha o'zgarishlarning dastlabki rasmlarini ketma-ket yozish orqali bekor qilinadi. Tranzaktsiyalarni qayta

bajarish imkoniyatiga ega bo'lish uchun jurnalda ma'lumotlar bazasi o'zgarganidan keyin har bir yozuv (yoki sahifa) nusxasi bo'lishi kerak. Bunday yozuvlar keyingi tasvirlar deb nomlanadi. Tranzaktsiya ma'lumotlar bazasiga u tomonidan kiritilgan barcha o'zgarishlarning oxirgi rasmlarini ketma-ket yozish bilan takrorlanadi.



12.1- rasm. Tranzaktsiyalarni orqaga qaytarish va orqaga qaytarish: a - ma'lumotlar bazasidagi o'zgarishlarni bekor qilish (orqaga qaytarish); b - o'zgarishlarni qayta bajarish (rulon).

Agar manba va maqsad rasmlari bo'lgan jurnal bo'lsa, unda bekor qilish va qayta bajarish elementar hisoblanadi (ammo biz baribir bu jarayonni tasvirlaymiz). Tranzaktsiyani bekor qilish uchun tiklash dasturi shunchaki har bir o'zgartirilgan yozuvni asl qiyofasi bilan almashtiradi. Barcha asl rasmlar tiklanganda, tranzaktsiya bekor qilinadi. Tranzaktsiyani qayta bajarish uchun qutqarish dasturi ushbu operatsiya boshlanganda mavjud bo'lgan ma'lumotlar bazasining versiyasidan boshlanadi va ma'lumotlar bazasiga barcha yakuniy rasmlarni yozadi. Yuqorida ta'kidlab o'tilganidek, ushbu harakat ma'lumotlar bazasining avvalgi versiyasi bilan qisman mavjudligini anglatadi.

Ma'lumotlar bazasini oxirgi suratiga tiklash va barcha operatsiyalarni qayta bajarish uchun ko'p vaqt ketishi mumkin. Kechikishni kamaytirish uchun MBBT ba'zan uzilish nuqtalarini ishlatadi. ***Malumot nuqtasi (nazorat nuqtasi)*** - bu ma'lumotlar bazasi va Tranzaksiyalar jurnalining o'rtasida sinxronizatsiya nuqtasi hisoblanadi. Tekshirish punktini kiritish uchun ma'lumotlar bazasi ma'lumotlar bazasi yangi so'rovlarni rad etadi, joriy so'rovlarni qayta ishlashni yakunlaydi va buferlarini diskka soladi. Keyin ma'lumotlar bazasi ma'lumotlar bazasini operatsion tizim diskka va jurnalga yozishning barcha operatsiyalari muvaffaqiyatli

yakunlanganligini xabar qiladi. Endi ma'lumotlar bazasi va jurnallar sinxronlashtirildi. Shundan so'ng, jurnalga nazorat punkti kiritiladi. Keyinchalik ma'lumotlar bazasi nazorat punktidan tiklanishi mumkin va siz faqat nazorat punktidan keyin boshlangan operatsiyalarning yakuniy rasmlarini yozib olishingiz kerak.

Tekshirish punktini kiritish arzon operatsiya bo'lib, siz soatiga uch yoki to'rt (yoki undan ko'p) operatsiyalarni bajarishingiz mumkin. Shunday qilib, tiklanish 15-20 daqiqadan ko'proq vaqtni talab qiladi. Ko'pgina ma'lumotlar bazalarida ma'lumotlar uzilish nuqtalari avtomatik ravishda kiritiladi, bu esa inson aralashuvini keraksiz qiladi.

12.2 Ma'lumotlar bazasi tuzilishini boshqarish

Ma'lumotlar bazasi tuzilmasini boshqarish ma'lumotlar bazasini loyihalashtirish va amalga oshirishda qatnashishni, shuningdek unga o'zgartirishlar kiritish jarayonida rahbarlik va nazoratni o'z ichiga oladi. Ideal holda, ma'muriyat bo'limi ma'lumotlar bazasini va uning ilovalarini ishlab chiqishning dastlabki bosqichida qatnashadi va talablarni o'rganishda, alternativalarni baholashda, shu jumladan MBBT-dan foydalanishni afzal ko'rgan holda va ma'lumotlar bazasi tuzilishini ishlab chiqishda qatnashadi. Katta tashkiliy dasturlar uchun ma'lumotlar bazasi ma'muri odatda ma'lumotlar bazasini loyihalashtirish uchun texnik yo'naltirilgan xodimlarning ishini boshqaradigan menejer hisoblanadi.

Yuqorida aytib o'tilganidek, ma'lumotlar bazasini yaratishda siz bir nechta muammolarni hal qilishingiz kerak. Birinchidan, ma'lumotlar bazasi yaratiladi va ma'lumotlar bazasi o'zi va uning jurnallari uchun jismoniy muhitda bo'sh joy ajratiladi. Keyin jadvallar, indekslar, saqlanadigan protseduralar va triggerlar yaratiladi. Keyingi ikki bobda siz bunga misollarni ko'rasiz. Ma'lumotlar bazasi tuzilmalari rasmlanganda ma'lumotlar bazasi ma'lumotlar bilan to'ldiriladi. Ko'pgina ma'lumotlar bazasini boshqarish ma'lumotlari katta hajmdagi ma'lumotlarni qayd etish uchun yordamchi dasturlar bilan ta'minlaydi.

Konfiguratsiya

Ma'lumotlar bazasi va uning ilovalari amalga oshirilgach, talablar muqarrar ravishda o'zgaradi. Bu yangi ehtiyojlar, ishbilarmonlik muhitidagi o'zgarishlar, siyosatdagi o'zgarishlar va hokazolarga bog'liq bo'lishi mumkin. Agar talablar o'zgarganda ma'lumotlar bazasi tarkibini o'zgartirish zarur bo'lsa, siz juda ehtiyotkorlik bilan harakat qilishingiz kerak, chunki tarkibiy o'zgarishlar kamdan-kam hollarda bitta dasturga ta'sir qiladi.

Shunday qilib, ma'lumotlar bazasini samarali boshqarish protseduralar va siyosatlarni o'z ichiga olishi kerak, ular orqali foydalanuvchilar o'zgarishga bo'lgan ehtiyojlarini ro'yxatga olishlari kerak, va butun ma'lumotlar bazasi foydalanuvchilari hamjamiyati global miqyosda qaror qabul qilinishi uchun ushbu o'zgarishlarning samarasini muhokama qilish imkoniyatiga ega bo'lishi kerak. ularni o'z ichiga qamrab oladimi. Ma'lumotlar bazasi va uni qo'llashning kattaligi va murakkabligi tufayli o'zgarishlar ba'zida kutilmagan natijalarga olib keladi. Shuning uchun, ma'lumotlar bazasi ma'muri ma'lumotlar bazasini tiklash kerakligiga tayyor bo'lishi

va muvaffaqiyatsizlikka olib kelgan muammoni tashxislash va tuzatish uchun etarli ma'lumotga ega bo'lishi kerak. Ma'lumotlar bazasi uning tuzilishiga o'zgartirishlar kiritgandan so'ng muvaffaqiyatsizlikka moyil bo'ladi.

Hujjatlar

Ma'lumotlar bazasining tuzilishini boshqarish ma'murining majburiyatlari hujjatlarning saqlanishini ham o'z ichiga oladi. Qanday o'zgartirishlar, qachon va qanday qilinganligini bilish juda muhimdir. Ma'lumotlar bazasi tarkibidagi o'zgarish olti oy davomida sodir bo'lmaydigan xatoga olib kelishi mumkin; o'zgarishlarning tegishli hujjatlari bo'lmasa, bunday muammoni tashxislash deyarli imkonsiz bo'ladi. Birinchi alomatlar qachon paydo bo'lishini aniqlash uchun o'nlab takroriy yugurish talab qilinishi mumkin va shuning uchun modifikatsiyani tekshirish uchun o'tkazilgan sinov protseduralari va yugurish qaydlarini yozish muhimdir. Agar standartlashtirilgan sinov protseduralari, test rasmlari va yozib olish usullari qo'llanilsa, test natijalarini yozib olish vaqtni talab qilmasligi kerak.

Hujjatlarni yuritish zerikarli va cheksiz jarayon bo'lishiga qaramay, unga sarflangan harakatlar tabiiy ofatlar yuz berganda to'lanadi va hujjat kerak bo'lmay turib jiddiy (va qimmat) muammoni hal qilib bo'lmaydi. At hozirgi vaqtda, yuritishni yukini engillashtirish ko'p mahsulotlar bor. Masalan, ma'lumotlar bazasining mantiqiy tuzilishini hujjatlashtirish uchun CASE-ning ko'plab vositalari ishlatilishi mumkin. O'zgarishlarni kuzatish uchun siz versiyani boshqarish dasturlaridan foydalanishingiz mumkin. Lug'atlar ma'lumotlar bazasida ma'lumotlarning tuzilishini o'qish va izohlash uchun hisobotlarni va boshqa vositalarni taqdim etadi.

Ma'lumotlar bazasi tarkibidagi o'zgarishlarni ehtiyotkorlik bilan hujjatlashtirishning yana bir sababi - bu tarixiy ma'lumotlardan to'g'ri foydalanish. Agar, masalan, sotuvchilar ikki yil davomida arxivda bo'lgan uch yil oldin sotuvlar to'g'risidagi ma'lumotlarni tahlil qilishni xohlasalar, ma'lumotlar arxivga yuborilishidan oldin ma'lumotlar bazasining tuzilishi nima ekanligini bilib olishlari kerak. Ma'lumotlar bazasi tarkibidagi o'zgarishlarni aks ettiruvchi yozuvlar ushbu savolga javob berishga yordam beradi. Agar buzilgan ma'lumotlar bazasini tiklash uchun olti oy oldin zaxira nusxasini ishlatishingiz kerak bo'lsa, shunga o'xshash vaziyat yuzaga keladi (garchi bunday bo'lmasligi kerak, lekin ba'zida shunday bo'ladi). Zaxira nusxasidan ma'lumotlar nusxasini ushbu holatda bo'lgan holatga qaytarish uchun foydalanish mumkin. Keyinchalik, ma'lumotlar bazasini hozirgi holatiga qaytarish uchun tarkibiy o'zgarishlarni amalga oshirish uchun xronologik ravishda operatsiyalarni bajarishingiz mumkin. Ro'yxatda ma'lumotlar bazasi tuzilishini boshqarish uchun ma'mur majburiyatlari ro'yxati mavjud.

Ma'lumotlar bazalari va ilovalarni yaratishda ishtirok etish.

- Talablarni aniqlash va alternativalarni baholash bosqichida yordam.
- Ma'lumotlar bazasini yaratish va yaratishda faol rol.
- Ma'lumotlar bazasi tuzilishini o'zgartirishda yordam
- Foydalanuvchilar bilan o'zaro aloqada echimlarni qidirish.
- Rejalashtirilgan o'zgarish har bir foydalanuvchiga qanday ta'sir qilishini baholash.

- Konfiguratsiya forumini tashkil qilish.
- O'zgarishlarni amalga oshirgandan keyin yuzaga keladigan muammolarni hal qilishga tayyorlik.
- Hisobga olish

12.3 Ma'lumotlar bazasi xavfsizligi

Ma'lumotlar bazasi xavfsizligini himoya qilish muayyan harakatlarni amalga oshirish huquqi faqat ma'lum foydalanuvchilarga va ma'lum bir vaqtda berilishiga asoslanadi. Ushbu maqsadga erishish qiyin va unga biron bir darajada yaqinlashish uchun ma'lumotlar bazasini ishlab chiqish guruhi loyiha talablarini aniqlash bosqichida barcha foydalanuvchilar uchun qayta ishlash huquqlari va majburiyatlarini belgilashlari kerak.

Ushbu xavfsizlik talablarining bajarilishi ma'lumotlar bazasining tegishli imkoniyatlari, agar ular etarli bo'lmasa, amaliy dasturlarning mantiqiyligi bilan ta'minlanishi mumkin.

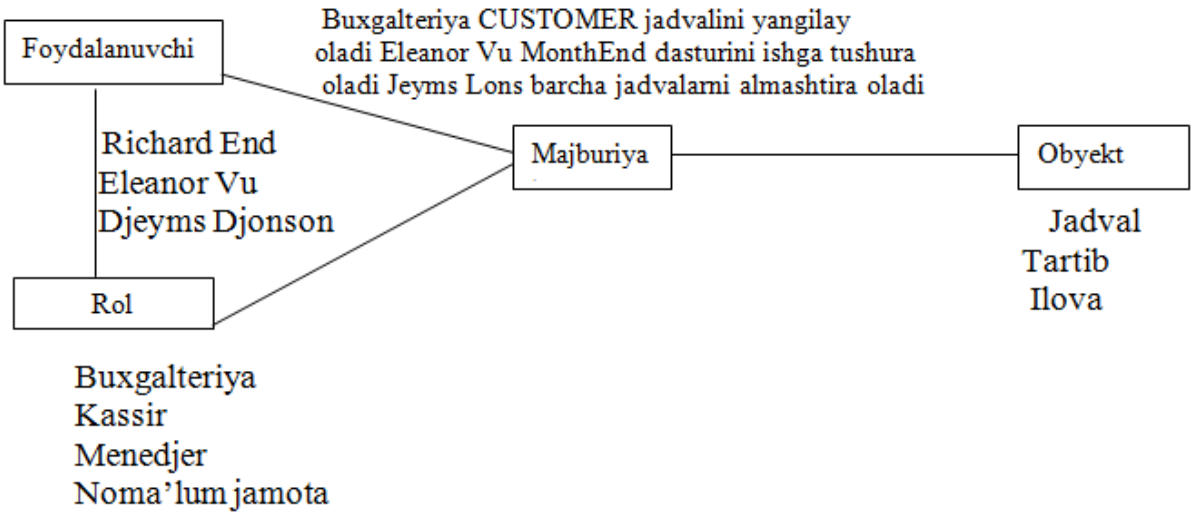
Qayta ishlashning huquq va majburiyatlari.

Ma'lumotlar bazasi ma'muriyatining vazifalariga ishlov berish huquqlari va majburiyatlarini boshqarish kiradi. Bu vaqt o'tishi bilan huquq va majburiyatlar o'zgarishi mumkinligini anglatadi. Ma'lumotlar bazasi bilan ishlash jarayonida, ilovalar va ma'lumotlar bazasining tarkibiga o'zgartirishlar kiritilganligi sababli, yangi huquqlar va majburiyatlarni kiritish yoki o'zgartirish zarurati tug'iladi. Ma'lumotlar bazasi ma'muri bunday o'zgarishlarni muhokama qilish va amalga oshirishda etakchi rol o'ynaydi.

Qayta ishlash huquqlari aniqlanganda, ulardan foydalanish kerak. Ushbu vazifa turli xil elementlarga berilishi mumkin: operatsion tizim, tarmoq, veb-server, ma'lumotlar bazasi yoki dastur. Keyingi ikkita bo'limda biz MBBT va ilova yordamida qayta ishlash huquqlarini amalga oshirishni ko'rib chiqamiz. Qolgan xususiyatlarning tavsifi ushbu kitobning doirasiga kirmaydi.

Ma'lumotlar bazasini boshqarish vositasi yordamida xavfsizlikni ta'minlash.

MBBT terminologiyasi, imkoniyatlari va xavfsizlik xususiyatlari mahsulotdan mahsulotga qarab farq qiladi. Amalda, barcha ma'lumotlar bazalarida ma'lumotlar bazasini muayyan vaqt va foydalanuvchilar doirasi bilan cheklash mumkin. MBBTning umumiy xavfsizlik modeli 12.2 – rasmda keltirilgan.



12.2-rasm. MBBT xavfsizlik modeli.

Foydalanuvchi bir yoki bir nechta rollarni tayinlashi mumkin, va rol bir yoki bir nechta foydalanuvchiga tegishli bo'lishi mumkin. Ikkala foydalanuvchi ham, rollarda ham har xil ruxsatlarga ega bo'lishi mumkin. Har bir ob'ekt (so'zning keng ma'nosida) ma'lum kuchlarga ega. Har bir imtiyoz bitta foydalanuvchi yoki guruhga va bitta ob'ektga tegishli.

Foydalanuvchi ma'lumotlar tizimiga kirganda, ma'lumotlar bazasi ma'lumotlar bazasi o'z harakatlarini ushbu foydalanuvchi uchun alohida belgilanadigan imtiyozlar bilan, shuningdek, ushbu foydalanuvchiga tayinlangan rol bilan cheklaydi. Umuman aytganda, foydalanuvchi o'zi kimligini da'vo qiladigan odam yoki yo'qligini aniqlash juda qiyin. Barcha tijorat ma'lumotlar bazasi parollarni himoyalashning bu yoki boshqa usulidan foydalanadi, garchi agar foydalanuvchilar shaxsiy identifikatorlariga beparvo bo'lsa, xavfsizlikni ta'minlashning bu usuli osonlikcha buzilishi mumkin.

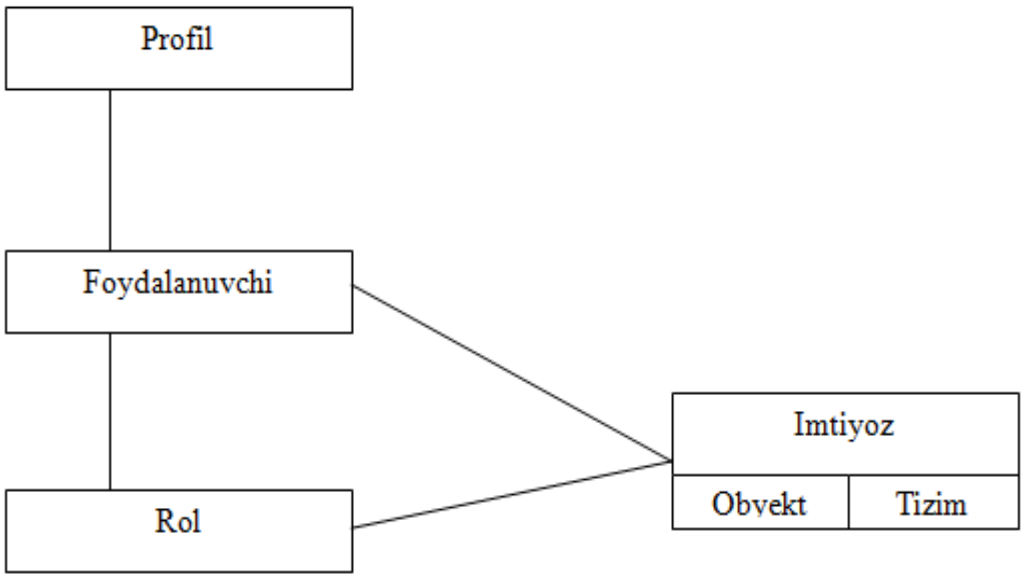
Ma'lumotni boshqarish tili (DCL) ko'p foydalanuvchilar muhitida ma'lumotlarga kirish huquqlari va protseduralarini boshqarish **uchun** ishlatiladi. Aniqrog'i, uni "kirishni boshqarish tili" deb atash mumkin. U ikkita asosiy jamoadan iborat:

- **GRANT** - huquqlar berish;
- **REVOKE** - huquqlarini olish.

Odatda, foydalanuvchilar o'zlarining foydalanuvchi nomlari va parollarini o'zlari kiritadilar, ba'zi dasturlarda foydalanuvchi va foydalanuvchi parolini tizim foydalanuvchi nomidan kiritadilar. Masalan, Windows 2000 SQL Serverga kirish uchun foydalanuvchi nomi va parolni to'g'ridan-to'g'ri yuborishi mumkin. Boshqa hollarda, foydalanuvchi nomi va parol ilova tomonidan ta'minlanadi.

Onlayn - ilovalar odatda (omma uchun noma'lum) «Noma'lum Public» kabi ismli bir guruh tomonidan belgilanadi, va Anonim foydalanuvchilar tizimiga, jumladan, bu guruhda yoziladi. Shunday qilib, Dell Computer kabi kompaniyalar har bir foydalanuvchini tizimga o'zlarining ismlari va parollari bilan kiritish zaruratidan qochishadi.

Ko'pgina ma'lumotlar bazalarida foydalaniladigan xavfsizlik tizimlarining modellari 12.3-rasmda keltirilgan. Ushbu modelga muvofiq, har bir foydalanuvchida ushbu foydalanuvchi qaysi tizim resurslaridan foydalanishi mumkinligini ko'rsatadigan profil mavjud. Biror foydalanuvchiga ko'plab rollar tayinlanishi mumkin va bir xil rol ko'plab foydalanuvchilarga berilishi mumkin. Har bir foydalanuvchi yoki rolda ko'p imtiyozlar mavjud. Imtiyozlarning ikki turi mavjud. Ob'ekt imtiyozlari ma'lumotlar bazasi ob'ektlari - jadvallar, ko'rinishlar va indekslarga nisbatan bajariladigan amallarni belgilaydi. Tizim imtiyozlari buyruqlar yordamida mumkin bo'lgan harakatlarni aniqlaydi.



12.3-rasm. Xavfsizlik modeli.

Ilova vositalari yordamida xavfsizliklarni ta’minlash.

Garchi Oracle yoki SQL Server kabi ma'lumotlar bazasi muhim xavfsizlik imkoniyatlarini ta'minlasada, bu qobiliyatlar juda keng tarqalgan. Agar dastur muayyan xavfsizlik choralarini talab qilsa, masalan, "foydalanuvchi jadval satrlarini yoki xodimning ismi o'z nomidan farq qiladigan jadvallarning birikmalarini ko'ra olmaydi ", u holda MBBT buni amalga oshirishga qodir bo'lmaydi. Bunday holatlarda xavfsizlik tizimi ilovaning o'zida bajarilgan funktsiyalar bilan to'ldirilishi kerak.

Maxsus ma'lumotlar bazasida xavfsizlikka tegishli ma'lumotlarni saqlashni tashkil qilishingiz mumkin, unga dastur yoki saqlanadigan protseduralar va triggerlar kirishlari mumkin.

Ilovaning ma'lumotlar bazasidagi xavfsizlik xususiyatlarini kengaytirishning boshqa ko'plab usullari mavjud. Biroq, birinchi navbatda, ma'lumotlar bazasini boshqarish tizimidan foydalanishingiz kerak. Faqatgina uning funktsiyalari etarli bo'lmasa, ularni amaliy dasturlar bilan to'ldirishga arziydi. Xavfsizlik darajasi qanchalik past bo'lsa, ruxsatsiz bostirish uchun shuncha kam imkoniyatlar mavjud. Bundan tashqari, DBMS xavfsizlik tizimi tezroq ishlaydi, undan foydalanish xarajatlarning pasayishiga olib keladi va, ehtimol, o'z tizimini

rivojlantirishdan ko'ra yaxshiroq natijalar beradi.

Shifrlash

Shifrlash - maxsus algoritmdan foydalanib ma'lumotlarni o'zgartirish, natijada ma'lumotlar shifrlash kalitiga ega bo'lmagan har qanday dastur tomonidan o'qib bo'lmaydigan bo'lib qoladi,

Agar ma'lumotlar bazasi bo'lgan tizim juda muhim maxfiy ma'lumotlarni o'z ichiga olgan bo'lsa, tashqaridan ruxsatsiz kirish tahdidining oldini olish uchun ma'lumotlar bazasini shifrlash maqsadga muvofiqdir (ma'lumotlar bazasi bilan bog'liq). Ba'zi ma'lumotlar bazalarida bunday maqsadlar uchun foydalanishga mo'ljallangan shifrlash vositalari mavjud. Bunday MBBT-larning ma'lumotlari ma'lumotlarga vakolatli kirishni ta'minlaydi (shifrlanganidan keyin), garchi bu ikki marta konvertatsiya qilish zarurati tufayli ba'zi ko'rsatkichlarning yomonlashishi bilan bog'liq bo'lsa. Shifrlash, shuningdek, aloqa liniyalari orqali uzatishda ma'lumotlarni himoya qilish uchun ishlatilishi mumkin. Berilgan ma'lumotni yashirish uchun ma'lumotlarni shifrlash uchun juda ko'p turli xil texnologiyalar mavjud bo'lib, ulardan ba'zilari qaytarib bo'lmaydigan, boshqalari esa qaytariladigan deb nomlanadi. Qaytarib bo'lmaydigan usullar, ularning nomidan ko'rinib turibdiki, dastlabki ma'lumotlarni aniqlashga imkon bermaydi, garchi ulardan ishonchli statistik ma'lumotlarni to'plash uchun foydalanish mumkin. Qayta tiklanadigan texnologiyalar ko'proq ishlatiladi. Xavfsiz tarmoqlar orqali ma'lumotlarning xavfsiz uzatilishini tashkil qilish uchun quyidagi tarkibiy qismlarni o'z ichiga olgan shifrlash tizimlaridan foydalanish kerak:

- manba ma'lumotlarini shifrlash uchun shifrlash kaliti (oddiy matn);
- shifrlash algoritmi, oddiy matnni shifrlash kaliti yordamida shifrlanishga qanday o'zgartirishni tasvirlaydi;
- shifrlangan matnni ochish uchun dekodlash kaliti;
- shifrlangan matnni oddiy manbaga aylantirish uchun shifrlash kalitidan qanday foydalanishni tasvirlaydigan shifrlash algoritmi.

Simmetrik deb ataladigan ba'zi shifrlash tizimlari ham shifrlash, ham shifrlash uchun bir xil kalitdan foydalanadi va kalitlarni almashtirish uchun xavfsiz aloqa liniyalari mavjud deb taxmin qilinadi. Biroq, ko'pchilik foydalanuvchilar xavfsiz aloqa liniyalaridan foydalana olmaydilar, shuning uchun ishonchli himoya olish uchun kalit uzunligi xabarning uzunligidan kam bo'lmasligi kerak. Biroq, aksariyat operatsion tizimlar xabarlarning o'zlaridan qisqa kalitlarga asoslangan.

Shifrlash tizimining yana bir turi xabarlarni shifrlash va shifrlash uchun turli xil kalitlardan foydalanishni o'z ichiga oladi - bunday tizimlar odatda assimetrik deb ataladi . Bunday tizimning misoli ikkita kalitni o'z ichiga olgan ochiq kalit tizimidir, ulardan biri ochiq va boshqasi sir saqlanadi. Shifrlash algoritmi ham ochiq bo'lishi mumkin, shuning uchun kalit egasiga shifrlangan xabarni yuborishni istagan har bir foydalanuvchi o'zining ochiq kalitidan va tegishli shifrlash algoritmidan foydalanishi mumkin. Biroq, ushbu shifrni faqat bitta juft shifrlash kalitiga ega bo'lgan kishi hal qilishi mumkin. Ochiq kalitlarni shifrlash tizimlaridan, shuningdek, haqiqatan ham ochiq kalit egasi tomonidan yuborilganligini tasdiqlovchi "raqamli imzo" xabarini yuborish uchun foydalanish mumkin .

12.4 Ma'lumotlar bazasini administratorlash va boshqarish

Ma'lumotlar bilan ishlashni va ma'lumotlar bazasining tuzilishini boshqarishdan tashqari, ma'mur MBBT-ni o'zi boshqarishi kerak. U tizimning ishlash statistikasini to'plashi va tahlil qilishi va muammolarga olib kelishi mumkin bo'lgan joylarni aniqlashi kerak. Eslatib o'tamiz, ma'lumotlar bazasi ko'plab foydalanuvchilar guruhlariga xizmat qiladi. Ma'lumotlar bazasi ma'muri tizimning sekin javobi, noto'g'ri ma'lumotlar, foydalanishda qiyinchiliklar va hokazolar bo'yicha barcha shikoyatlarni ko'rib chiqishi kerak. Agar biron bir o'zgartirish kerak bo'lsa, ma'mur ushbu o'zgarishlarni rejalashtirishi va amalga oshirishi kerak.

Ma'mur vaqti-vaqti bilan ma'lumotlar bazasi bilan ishlashda foydalanuvchi faoliyatini kuzatishi kerak. Hisobotlarda qaysi foydalanuvchilar eng faol bo'lganligi, qaysi fayllar va ehtimol ma'lumotlar elementlari ishlatilganligi, qaysi kirish usullari ishlatilganligi to'g'risida ma'lumotlar bo'lishi mumkin. Shuningdek, hisobotlarda xatolarning turlari va chastotalari to'g'risidagi ma'lumotlar bo'lishi mumkin. Ma'mur ish faoliyatini yaxshilash yoki foydalanuvchi tajribasini soddalashtirish uchun ma'lumotlar bazasi tuzilmasiga biron bir o'zgartirish kiritish kerakligini aniqlash uchun ushbu ma'lumotlarni tahlil qiladi. Agar bunday o'zgartirishlar kerak bo'lsa, ma'mur ularning amalga oshirilishini ta'minlaydi.

Ma'lumotlar bazasi ma'muri ma'lumotlar bazasi va foydalanuvchi faoliyati to'g'risidagi joriy statistikani tahlil qilishi kerak. Agar ishlashda biron bir muammo aniqlansa (hisobotni tahlil qilish paytida yoki foydalanuvchi shikoyati bo'yicha), ma'mur ma'lumotlar bazasi yoki tizim tuzilmasiga o'zgartirish kiritish kerak yoki kerak emasligini aniqlashi kerak. Mumkin bo'lgan tarkibiy o'zgartirishlarga yangi kalitlarni kiritish, ma'lumotlarni tozalash, kalitlarni o'chirish va ob'ektlar o'rtasida yangi aloqalarni o'rnatish kiradi.

Ishlatilgan MBBT ishlab chiqaruvchisi mahsulotning yangi xususiyatlarini e'lon qilganda, ma'lumotlar bazasi ma'muri ularni foydalanuvchilar hamjamiyati ehtiyojlari asosida ko'rib chiqishi kerak. Agar u ushbu yangi funktsiyalarni amalga oshirishga qaror qilsa, ishlab chiquvchilar xabardor qilinishi va ulardan foydalanish bo'yicha o'qitilishi kerak. Shunga muvofiq, ma`lumotlar bazasi tuzilishini boshqarish bilan bir qatorda, ma'mur ma'lumotlar bazasini o'zgartirish va boshqarish kerak.

Ma'lumotlar bazasi ma'murining vazifalari tizimdagi boshqa o'zgarishlarni ham o'z ichiga olishi mumkin; qaysi biri - bu ma'lumotlar bazasi ma'lumotlari va foydalanilgan boshqa dasturiy-texnik vositalarga bog'liq. Masalan, operatsion tizim yoki veb-serverdagi o'zgarishlar MBBT ba'zi funktsiyalari, funktsiyalari yoki parametrlarini o'zgartirish zaruratini keltirib chiqarishi mumkin. Shuning uchun ma'lumotlar bazasi ma'muri ishlatiladigan boshqa dasturiy ta'minot bilan ishlash uchun ma'lumotlar bazasini boshqarish tizimini sozlashi kerak.

MBBT parametrlarini (masalan, tranzaktsiyalarni izolyatsiya qilish darajasi) dastlabki tanlash tizimning ma'lum foydalanuvchi muhitida qanday ishlashi haqida kam ma'lumot bo'lgan bir vaqtda sodir bo'ladi. Shuning uchun, tizim bilan ishlash tajribasi va uning ishlashini tahlil qilish natijalari o'zgarishga ehtiyoj borligini ko'rsatishi mumkin. Ishlash maqbul ko'rinadigan bo'lsa ham, ma'lumotlar bazasi

ma'muri uning parametrlarini o'zgartirish ishlashga qanday ta'sir qilishini o'rganishni xohlashi mumkin. Ushbu jarayon tizimni sozlash yoki optimallashtirish deb nomlanadi. Ro'yxatda ma'lumotlar bazasi ma'murining ma'lumotlar bazasini boshqarish bo'yicha majburiyatlari ro'yxati keltirilgan.

- Ma'lumotlar bazasida hisobotlarni tayyorlash.
- Tizimdan foydalanuvchilarning shikoyatlarini ko'rib chiqish.
- Ma'lumotlar bazasi yoki dasturlarning tuzilishidagi o'zgarishlarni baholash.
- Ma'lumotlar bazasi tuzilishini o'zgartirish.
- Yangi MBBT funktsiyalari va funktsiyalarini baholash va amalga oshirish.
- Ma'lumotlar bazasini sozlash.

Ma'lumotlar bazasini yuritish

Internetda musiqa sotish kabi elektron tijorat kompaniyalari foydalanadigan o'xshash katta va faol onlayn ma'lumotlar bazasini tasavvur qiling. Bunday tizim uchun ma'lumot bir nechta turli xil ma'lumotlar bazalari, o'nlab veb-sahifalar va yuzlab, ammo minglab foydalanuvchilar tomonidan ta'minlanishi mumkin.

Aytaylik, ushbu ilovadan foydalanuvchi kompaniya sport mollari bilan taklif etiladigan tovarlar turlarini kengaytirishni xohlaydi. Kompaniyaning yuqori rahbariyati ma'lumotlar bazasi ma'muridan yangi mahsulot qatorini qo'llab-quvvatlash uchun ma'lumotlar bazasini yaratish uchun zarur bo'lgan vaqt va boshqa manbalarni baholashni so'rashi mumkin.

Ma'lumotlar bazasi ma'muri ushbu so'rovga javob berishi uchun unga ma'lumotlar bazasi, uning ilovalari va ushbu ilovalarning tarkibiy qismlari, foydalanuvchilar va ularning huquqlari va imtiyozlari, shuningdek tizimning boshqa elementlari tavsiflangan batafsil metadata kerak bo'ladi. Ushbu meta-ma'lumotlarning bir qismi ma'lumotlar bazasining tizim jadvallarida saqlanadi, ammo ushbu qismning o'zi yuqori rahbariyat tomonidan berilgan savollarga javob berish uchun etarli bo'lmaydi.

Ma'lumotlar bazasi ma'muriga COM va ActiveX ob'ektlari, skript protseduralari va funktsiyalari, ASP sahifalari, uslublar jadvallari, hujjatlar turlarining ta'riflari va hokazolar haqida qo'shimcha ma'lumotlar kerak. Bundan tashqari, MBBT xavfsizlik mexanizmlari foydalanuvchi, guruh va imtiyoz ma'lumotlarini hujjatlashtiradi, buni juda tuzilgan va ko'pincha noqulay tarzda bajariladi.

Shu sabablarga ko'ra, ko'plab tashkilotlar ma'lumotlar omborlarini ishlab chiqadilar va saqlaydilar, bu ma'lumotlar bazasini, uning ilovalarini, veb-sahifalarini, foydalanuvchilarini va boshqa amaliy komponentlarini tavsiflovchi metadata to'plamidir. Rezervuar virtual bo'lib, uning meta-ma'lumotlari turli xil manbalardan, shu jumladan MBBT, versiyani boshqarish dasturi, kod kutubxonalari, veb-sahifalarni yaratish va tahrirlash vositalari va boshqalardan to'planishi ma'nosida virtual bo'lishi mumkin.

CASE-vositalari ishlab chiqaruvchisi yoki Microsoft yoki Oracle kabi boshqa kompaniyalar tomonidan etkazib beriladigan mahsulot.

Ushbu holatlarning har qandayida, ma'lumotlar bazasi ma'muri yuqori rahbariyat savol bermasdan oldin, ma'lumotlar bazasini yaratishni ko'rib chiqishi kerak. Aslida, tizimni ishlab chiqish vaqtida omborxona qurilishi kerak va uni tizimning muhim tarkibiy qismi sifatida ko'rib chiqish kerak. Aks holda, ma'lumotlar bazasi ma'muri mavjud dasturlarni saqlashga, ularni yangi ehtiyojlarga moslashtirishga va qandaydir tarzda ombor uchun metadata to'plashga harakat qilib, "abadiy qo'lga kiritadi".

Repozitoriyalarning eng yaxshi turi - bu faol omborxonalar. Ular tizim tarkibiy qismlarini yaratishda metadata avtomatik ravishda yaratilganligi sababli tizimni rivojlantirish jarayonida rasmlanadi. Kamroq istalgan, ammo baribir samarali variant - passiv omborxonalar: bunday omborlarni to'ldirish va ular uchun metadata yaratish qo'lda amalga oshiriladi.

Internet mijozlar bazasini kengaytirish va biznesda sotish va daromadlilikni oshirish uchun misli ko'rilmagan imkoniyatlarni yaratdi. Kompaniyalar o'z ishlarida foydalanadigan ma'lumotlar bazalari va ilovalar ushbu muvaffaqiyatning asosiy elementini tashkil etadi. Afsuski, o'sishiga dasturlarini o'stirish yoki ularni o'zgaruvchan ehtiyojlarga moslashtirish imkoniyati to'sqinlik qiladigan tashkilotlar mavjud. Ko'pincha mavjud tizimni moslashtirishdan ko'ra yangi tizimni qurish osonroq; shubhasiz, uni almashtirishda eski tizim bilan uyg'unlashadigan yangi tizimni yaratish juda qiyin vazifa bo'lishi mumkin.

Savollar:

1. Ma'lumotlar bazasida ma'lumotlarni tiklashning qanday usullari mavjud?
2. Ma'lumotlar bazasi konfiguratsiyasini boshqarish qanday ta'minlanadi?
3. Ma'lumotlar bazasi xavfsizligini boshqarish qanday ta'minlanadi?
4. Administrator ma'lumotlar bazasini qo'llab-quvvatlash jarayonida qanday ishtirok etadi?
5. Shifrlash qanday ta'minlanadi?
6. Ma’lumotlar ombori nima o'zi?
7. Repozitoriyalarning eng yaxshi turi?
8. CASE-bu?

Adabiyotlar:

1. Thomas Connolly, Carolyn Begg – Database systems. A practical Approach to Design, Implementation and Management. 4th Edition – Addison Wesley 2005 – 1373p.
2. C. J. Date – An Introduction to Database Systems – Addison-Wesley Professional – 2003 – 1024 p.

XIII Bob. Ochiq ma’lumotlar bazasi aloqasi (ODBC) interfeysi

13.1 Ma'lumotlarga kirishga ruxsat berish texnologiyalari.

IP- dizayneri yoki dasturchisi tomonidan hal qilinishi kerak bo'lgan asosiy vazifalardan biri bu ma'lumotlar bazasiga kirish texnologiyasini tanlashdir. Ma'lumotlarga kirish texnologiyalarini tanlash strategik vazifalardan biri bo'lib, uning echimi kelajakdagi tizimning ishlashiga va qo'shimcha funktsiyalarni amalga oshirish qobiliyatiga, uning boshqa dastur platformalari va texnologiyalariga mosligi, bitta platformadan boshqasiga o'tish qobiliyatiga bog'liq.

Ma'lumotlarga kirishni ta'minlash muammosini hal qilishning bir necha yo'li mavjud.

Ko'pgina zamonaviy MBBT-larda ma'lumotlarni boshqarish uchun maxsus funktsional dasturlash interfeysi (API) mavjud bo'lgan kutubxonalar mavjud. Ish stoli tizimlari uchun MBBTda, API faqat ma'lumotlar bazasini o'qiydi va yozadi. MBBT mijoz / server turida, API tarmoq orqali serverga so'rov yuborish va mijoz dasturini keyinchalik qayta ishlash uchun natijalar yoki xato kodlarini qabul qilishni boshlaydi.

Ma'lumotlarga kirishning bir usuli bu APIni to'g'ridan-to'g'ri ishlatish, ammo bu dastur ishlatilgan ma'lumotlar bazasiga to'liq bog'liqligini anglatadi. Shunday qilib, mijozga murojaat qilish uchun turli xil ma'lumotlar bazalari bilan ishlash uchun odatiy funktsiyalar, sinflar, xizmatlar va xizmatlarning standart to'plamini taqdim etadigan ma'lum universal universal mexanizm kerak. Ushbu standart funktsiyalar (sinflar yoki xizmatlar) ma'lumotlar bazasi drayverlari (provayderlar) deb nomlangan kutubxonalarga joylashtirilishi kerak. Har bir bunday kutubxona ma'lum bir ma'lumotlar bazasini boshqarish tizimiga API qo'ng'iroqlaridan foydalangan holda standart funktsiyalar, sinflar yoki xizmatlar to'plamini amalga oshiradi.

13.2 Open Database Connectivity (ODBC) interfeysi

ODBC (Open Database Connectivity) interfeysi Microsoft tomonidan ochiq ma'lumotlar bazasiga kirish interfeysi sifatida ishlab chiqilgan. U mijoz deb nomlangan dastur (yoki mijoz ilovasi) va server - ma'lumotlar bazasi o'rtasidagi o'zaro ta'sirning yagona vositasini ta'minlaydi .

ODBC interfeysi ma'lumotlar bazalariga kirish uchun standart til sifatida X / Open va ISO/IEC tomonidan ishlab chiqilgan Call-Level Interface (CLI) spetsifikatsiyasiga, shuningdek SQL (Strukturalangan so'rovlar tili) ga asoslangan.

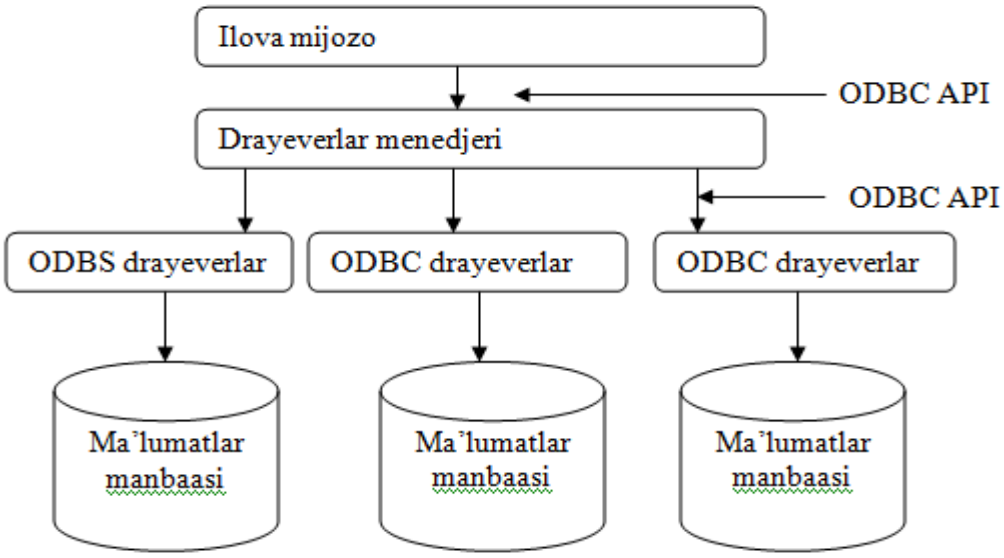
ODBC interfeysi turli xil ma'lumot manbalariga ODBCdan foydalangan holda har qanday dasturga birlashtirilgan kirishni ta'minlaydigan dasturlarning maksimal darajada ishlashini ta'minlash uchun yaratilgan. Shunday qilib, agar ODBC va SQL standartlariga mos keladigan dastur dastlab Microsoft Access ma'lumotlar bazasi bilan ishlash uchun ishlab chiqilgan bo'lsa, keyin ushbu ma'lumotlar bazasining jadvallari Microsoft SQL Server ma'lumotlar bazasiga yoki Oracle ma'lumotlar bazasiga topshirilgan bo'lsa, dastur ushbu ma'lumotlarni kiritmasdan ishlov berishni davom ettirishi mumkin. o'zgaradi.

Ma'lumotlar bazasi bilan o'zaro ishlash uchun mijoz dasturi ODBC drayverlari deb nomlangan maxsus modullarda bajariladigan ODBC interfeys funktsiyalarini chaqiradi. Odatda, ODBC drayverlari DLL bo'ladi va bitta DLL bir nechta ODBC drayverlarini qo'llab-quvvatlaydi. Kompyuteringizga biron bir SQL serverni o'rnatganingizda (SQL til standartlaridan birini qo'llab-quvvatlaydigan ma'lumotlar bazasi, masalan, SQL-92), Windows ro'yxatga olish kitobida ro'yxatdan o'tish va tegishli ODBC drayveri avtomatik ravishda amalga oshiriladi.

ODBC arxitekturasi

ODBC arxitekturasi to'rt komponentdan iborat (13.1-rasm):

- ODBC funktsiyalarini chaqiradigan mijoz dasturi.
- Mijoz dasturlari uchun zarur bo'lgan ODBC drayverlarini yuklab oladigan va chiqaradigan haydovchi menejeri. Drayver menejeri ODBC funktsiyalariga qo'ng'iroqlarni qayta ishlaydi yoki ularni drayverga uzatadi.
- SQL funktsiyalariga qo'ng'iroqlarni qayta ishlaydigan ODBC drayveri, bajariladigan SQL bayonotini SQL serveriga o'tkazish va chaqirilayotgan funktsiyani mijoz dasturiga bajarish natijasi.
- Ma'lum bir mahalliy yoki uzoq ma'lumotlar bazasi sifatida belgilangan ma'lumotlar manbai.



13.1-rasm. ODBC arxitekturasi.

Drayver menejerining asosiy maqsadi ulangan ma'lumotlar manbasiga mos keladigan drayverni yuklash va turli xil ODBC drayverlaridan foydalangan holda ma'lumot manbalarining har xil turlari bilan o'zaro aloqani o'rnatishdir.

ODBC drayverlari quyidagi funktsiyalarni bajarish uchun mijoz dasturi bilan o'zaro ishlaydi:

- mijoz ilova va ma'lumotlar manbai o'rtasidagi aloqa protokollarini boshqarish;
- MBBT so'rovlarini boshqarish;

- mijoz ilovasidan ma'lumotlar bazasi ma'lumotlar bazasiga va ma'lumotlar bazasidan mijozga buyurtma uchun ma'lumotlarni uzatish;
- mijozga buyurtmaga qaytish kodi ko'rinishidagi ODBC funktsiyasini chaqirish to'g'risidagi standart ma'lumotni qaytarish;
- kursorlar bilan ishlashni qo'llab-quvvatlaydi va tranzaktsiyalarni boshqaradi.

Mijozlar ilovasi bir vaqtning o'zida turli ODBC drayverlaridan foydalangan holda bir nechta turli xil ma'lumotlar manbalariga ulanishlarni, shuningdek bitta ODBC drayveridan foydalanib bir xil ma'lumot manbalariga bir nechta ulanishlarni o'rnatishi mumkin.

ODBC API funktsiyalari.

Barcha ODBC API funktsiyalarini shartli ravishda to'rt guruhga bo'lish mumkin:

1. Ma'lumotlar manbasi bilan o'zaro ishlash uchun ODBC yadro funktsiyalari
2. o'rnatish funktsiyalari (DLL-ni o'rnatish);
3. ODBC va ma'lumotlar manbalarini o'rnatish funktsiyalari (DLL-ni o'rnatish);
4. ma'lumotlarni o'zgartirish funktsiyalari (DLL tarjimasi).

Barcha funktsiyalarning deklaratsiyalari va ular ishlatadigan ma'lumotlar turlari sarlavha fayllarida mavjud. ODBC API asosiy funktsiyalari guruhi uchta darajaga bo'lingan:

1. ODBC yadro funktsiyalari
2. 1-darajali funktsiyalar;
3. 2 darajadagi vazifalar.

ODBC API Windows uchun qatlamli DLL funktsiyalari to'plami sifatida amalga oshiriladi. ODBC, DLL dinamik kutubxonasi ODBC drayverlarni boshqarishning asosiy kutubxonasidir, unda tizim tomonidan qo'llab-quvvatlanadigan turli xil ma'lumotlar bazalari uchun maxsus haydovchilarni chaqirish funktsiyalari mavjud. Har bir haydovchi o'z darajasiga mos keladi va ikkita toifadan biriga tegishlidir: bitta darajali yoki ko'p darajali drayverlar.

Bir darajali drayverlar ANSI SQL-dan foydalanib to'g'ridan-to'g'ri ishlov berilmaydigan ma'lumotlar manbalari bilan ishlashda foydalanish uchun mo'ljallangan. Odatda, bu shaxsiy kompyuterlardagi dBase, Paradox, FoxPro va Excel kabi mahalliy ma'lumotlar bazasi. Ushbu ma'lumotlar bazalariga mos keladigan drayverlar ANSI SQL-ni ma'lumotlar bazasini tashkil etuvchi fayllarni to'g'ridan-to'g'ri qayta ishlaydigan pastki darajadagi ko'rsatmalar to'plamiga tuzadilar.

Ko'p darajali drayverlar SQL bayonotlarini qayta ishlash uchun MBBT serveridan foydalanadilar va mijoz-server muhitida ishlashga mo'ljallangan. ANSI SQL ishlov berish bilan bir qatorda, ular ma'lum bir RMBBT uchun o'zlarining konstruktsiyalarini ham qo'llab-quvvatlashlari mumkin, chunki ODBC SQL bayonotlarini ma'lumot manbalariga tarjimasiz (pas mexanizmi mexanizmi) uzatishi

mumkin. Mijoz-server texnologiyasi tomonidan qo'llab-quvvatlanadigan ma'lumotlar bazalari uchun ODBC drayverlari deyarli barcha sanoat ma'lumotlar bazalari serverlari uchun joriy qilingan.

ODBC API orqali ma'lumot so'rash tartibiga 4 ta muhim qadam (qadam) mavjud.

- *1-qadam* - aloqani o'rnatish. Birinchi qadam ODBC drayverlari va kutubxonalari uchun operativ xotirani ajratadigan ODBC tutqichlarini joylashtirishdir. Keyin ulanish markerlari uchun xotira ajratiladi va ulanish o'rnatiladi.
- *2-qadam* - SQL bayonotini bajarish Operator ko'rsatgichi ta'kidlangan, mahalliy o'zgaruvchilar SQL bayonnomasidagi ustunlar bilan bog'langan (bu ixtiyoriy harakat) va ifoda qayta ishlash uchun asosiy ODBC drayveriga taqdim etiladi.
- *3-qadam* - ma'lumotlarni olish. Ma'lumot olishdan oldin, natijalar to'plami haqidagi ma'lumotlar, xususan, to'plamdagi ustunlar soni qaytariladi. Ushbu raqam asosida natijalar to'plami yozuv buferiga joylashtiriladi, uni ko'rish uchun sikl amalga oshiriladi va har bir ustunning mazmuni tegishli mahalliy o'zgaruvchiga joylashtiriladi. Agar siz mahalliy o'zgaruvchilar bilan ustunli bog'lanishdan foydalansangiz, bu qadam majburiy emas.
- *4-qadam* - Resurslarni bo'shatish Ma'lumotlar olgandan so'ng, resurslar operator, ulanish va atrof-muhit ko'rsatkichlarini bo'shatish funktsiyalarini chaqirish orqali ozod qilinadi. Qayta ishlash jarayonida operator va ulanish ko'rsatkichlaridan foydalanish mumkin.

ODBC texnologiyasi ma'lumotlarga kirishning umumiy, ma'lumot manbasiga bog'liq bo'lmagan usul sifatida ishlab chiqilgan. Texnologiyani qo'llash, shuningdek, turli xil ma'lumotlar bazalari atrof-muhitiga ilovalarni o'zlari qayta ishlashga ehtiyoj sezmasdan ko'chirish imkoniyatini ta'minlashi kerak edi. Shu ma'noda, ODBC texnologiyasi allaqachon sanoat standartiga aylandi, uni deyarli barcha ishlab chiqaruvchilar MBBT va ishlab chiqarish vositalari tomonidan qo'llab-quvvatlanadi.

Biroq, ko'p qirrali bo'lish qimmat. Agar dasturni ishlab chiqish jarayonida asosiy mezonlardan biri turli xil ma'lumotlar bazalari uchun qulaylik bo'lsa, ODBC-dan foydalanish asosli bo'ladi. Ilovaning samaradorligi va samaradorligini oshirish uchun ma'lumotlar bazasida SQL-ga xos kengaytmalar faol ishlatiladi, serverda saqlanadigan protseduralar va funktsiyalar qo'llaniladi. Bunday holda, ma'lumotlarga kirishning umumiy usuli sifatida ODBC ning o'rni yo'qoladi.

Bundan tashqari, turli xil MBBT uchun ODBC drayverlari har xil muvofiqlik darajalarini qo'llab-quvvatlaydi. Shuning uchun, ishlab chiqarish vositalarining ko'plab ishlab chiqaruvchilari ODBC-ni qo'llab-quvvatlashdan tashqari asosiy MBBT uchun "to'g'ridan-to'g'ri" drayverlarni etkazib berishadi.

OLE MB Modeli

OLE DB - bu mijoz dasturiga turli xil ma'lumot manbalariga birlashtirilgan kirishni ta'minlaydigan COM interfeyslari to'plami (Component Object Model).

Aytishimiz mumkinki, OLE DB bu ma'lumotlarning turiga va ularning

joylashgan joyiga qaramasdan har qanday ma'lumotni standart COM interfeyslari orqali kirish usuli. Ma'lumotlar bazasi, oddiy hujjatlar, Excel elektron jadvallari va boshqa har qanday ma'lumotlar manbalari bo'lishi mumkin. OBDC drayverlari tomonidan taqdim etilgan kirishdan farqli o'laroq, OLE DB sizga SQL tilidan (SQL serverlari) va boshqa istalgan o'zboshimchalik manbalaridan foydalangan holda ma'lumot manbalariga kirish huquqini beradi.

OLE DB texnologiyasidan foydalangan holda ma'lumotlar manbasiga kirishni ta'minlaydigan vositalar OLE DB ta'minotchilari deb nomlanadi. OLE DB provayderlaridan kirish uchun foydalanadigan mijoz dasturlari ma'lumotlar iste'molchilari deb nomlanadi.

Agar ma'lum bir ma'lumot manbasiga kirish uchun faqat ODBC drayveri mavjud bo'lsa, OLE DB texnologiyasidan foydalanish uchun ODBC ma'lumotlar manbasiga kirish uchun mo'ljallangan OLE DB provayderidan foydalanishingiz mumkin.

OLE DB arxitekturasi COMga asoslanganligi sababli, natijalar to'plamini yaratish mexanizmi quyidagi bosqichlarning ketma-ketligidan iborat:

1. ob'ekt yaratish ->
2. yaratilgan ob'ektning interfeysiga ko'rsatgichni so'rash ->
3. interfeys usulini chaqirish.

ODBC texnologiyasidan foydalanganda natijalar to'plamini yaratgandan keyin bajariladigan harakatlar majmuasiga o'xshab - OLE MB texnologiyasi kirish mexanizmidan foydalanadi. Baholovchilar ma'lumotlar iste'molchisining xotirasi maydoniga ma'lumotlar qanday yozilishini tavsiflaydi, ma'lumot iste'molchisining buferidagi xotira mintaqasi va natijalar to'plamidagi ma'lumotlar ustunlari o'rtasida manzil mosligini o'rnatadi. Ba'zan bunday munosabatlar to'plami ustun xaritasi deb nomlanadi.

OLE DB spetsifikatsiyasi OLE DB ob'ektlari tomonidan amalga oshiriladigan interfeyslar to'plamini tavsiflaydi. Har bir ob'ekt turi interfeyslar to'plami sifatida belgilanadi. OLE DB spetsifikatsiyasi har qanday OLE DB provayderlari tomonidan amalga oshirilishi kerak bo'lgan bazaviy darajadagi interfeyslarning to'plamini belgilaydi.

OLE DB asosiy modeliga quyidagi ob'ektlar kiritilgan:

- Ma'lumotlar manbasi ob'ekti ma'lumotlar manbasiga ulanish va bir yoki bir nechta seanslarni yaratish uchun ishlatiladi. Ushbu ob'ekt ulanishni boshqaradi, foydalanuvchi ma'lumotlari va autentifikatsiya ma'lumotlarini ishlatadi;
- Session ob'ekti ma'lumotlar manbai bilan o'zaro munosabatni boshqaradi - so'rovlarni bajaradi va natijalar to'plamini yaratadi. Sessiyada metadata ham qaytarilishi mumkin. Seans bir yoki bir nechta buyruqlarni yaratishi mumkin;
- Rowset (natijalar to'plami) - bu buyruq natijasida olingan yoki seansda yaratilgan ma'lumotlar.

e-mail fayllar, fayl tizimlari mazmuni va foydalanuvchi ish uchun ilişkisel va nodavlat ilişkisel o'z ichiga olgan matn, grafika va geografik ma'lumotlar - OLE DB

turli ma'lumot manbalariga ulanish imkonini beradi tizimi-darajali interfeysi - ob'ektlar. OLE DB tarkibiy ob'ektlar modeli (COM) uchun interfeyslar to'plamini belgilaydi, bu ma'lumotlarga universal kirishni ta'minlash uchun turli xil ma'lumotlar bazasini boshqarish tizimlarining xizmatlarini o'z ichiga oladi. Ushbu interfeyslardan foydalangan holda dasturchilar qo'shimcha ma'lumotlar bazasi xizmatlarini yaratishi mumkin.

OLE DB interfeyslaridan foydalanadigan har qanday dastur komponenti OLE DB iste'molchisi hisoblanadi. Bu biznes dastur, Borland Delphi kabi dasturiy ta'minotni ishlab chiqish vositasi, murakkab dasturlar yoki OLE DB interfeyslaridan foydalangan holda ActiveX Data Objects ob'ekt modeli bo'lishi mumkin. Iste'molchilar OLE DB-dan foydalangan holda ma'lumotlarga bevosita kirishni ta'minlash uchun amaliy darajadagi interfeys yoki OLE DB-provayderi yordamida ma'lumotlarga to'g'ridan-to'g'ri kirish uchun ActiveX Data Objects (ADO) dan foydalanadilar.

OLE DB nuqtai nazaridan, provayderlarning ikki turi mavjud bo'lishi mumkin: OLE DB - ma'lumotlar etkazib beruvchilar va xizmat ko'rsatuvchi provayderlar.

Xizmat ma'lumotlar (ma'lumotlar provayderi) ma'lumotlarni "egalik" bir dasturiy komponent hisoblanadi. Bu iste'molchi va to'g'ridan-to'g'ri ma'lumotlar qatori o'rtasida. OLE DB-da barcha provayderlar ma'lumotlarni jadval jadvalida (ular biz bilan aloqador ma'lumotlar bazalarida va jadvallarda tanish bo'lgan) virtual jadval rasmida taqdim etadilar. Ma'lumot ta'minotchisi quyidagi vazifalarni bajaradi.

- Iste'molchining ma'lumotlarga kirish uchun so'rovlarini qabul qiladi.
- Ma'lumotlar qatoridan ma'lumotlarni tanlaydi yoki yangilaydi.
- Ushbu ma'lumotlarni iste'molchiga qaytaradi.

Ma'lumot ta'minotchisining misollaridan biri Microsoft Jet 4.0 OLE DB Provayderidir. Ushbu ma'lumot Microsoft Jet ma'lumotlar bazalariga kirish mexanizmi bilan birgalikda ishlatiladi, bu ma'lumot Microsoft Access ma'lumotlar bazasida ma'lumotni qayta ishlash uchun ishlatiladi, shuningdek, indekslangan ketma-ket kirish usuli deb ataladigan ma'lumotlar bazalari sifatida tashkil etilgan ma'lumotlarga kirish uchundir (Jet tomonidan qo'llab-quvvatlanadigan I-ISAM). Bunday ma'lumotlarga Excel ish daftarlarida saqlanadigan jadvallar, Outlook va Microsoft Exchange pochta fayllari, dBase va Paradox jadvallari, matnli va HTML-fayllar va boshqalar kiradi.

Boshqa OLE DB provayderi SQL Serwer uchun Microsoft OLE DB Provayderi bo'lib, ishlash uchun ishlatiladi Microsoft SQL Server 6.5, 7.0, 2000 ma'lumotlar bazalari bilan.

Xizmat ko'rsatuvchi provayder an'anaviy ma'lumotlar etkazib beruvchilari tomonidan qo'llab-quvvatlanmaydigan va ma'lumotlarning o'zi "egalik qilmaydigan" rivojlangan funktsiyalarni amalga oshiradi. Ushbu provayder, masalan, saralash, filtrlash, tranzaktsiyalarni boshqarish, SQL so'rovlarini qayta ishlash, ko'rsatgich (kursor) funktsiyalari va boshqalarni taqdim etadi. Xizmat ko'rsatuvchi provayder to'g'ridan-to'g'ri ma'lumotlar qatorlari yoki tegishli

ma'lumotlar etkazib beruvchisi bilan ishlashi mumkin; bu holda, u iste'molchi va ta'minotchi sifatida ishlaydi.

Masalan, OLE DB uchun Microsoft Cursor Service va OLE DB uchun Microsoft Data Shaping Service kabi xizmat ko'rsatuvchi provayderlar o'zlarining funktsiyalarini kengaytirish uchun yadro OLE DB ma'lumotlar etkazib beruvchilari bilan birlashishlari mumkin.

OLE DB uchun Microsoft Cursor Service va OLE DB uchun Microsoft Data Shaping Service kabi xizmat ko'rsatuvchi provayderlar o'zlarining funktsiyalarini kengaytirish uchun asosiy OLE DB ma'lumotlar etkazib beruvchilari bilan birlashishlari mumkin.

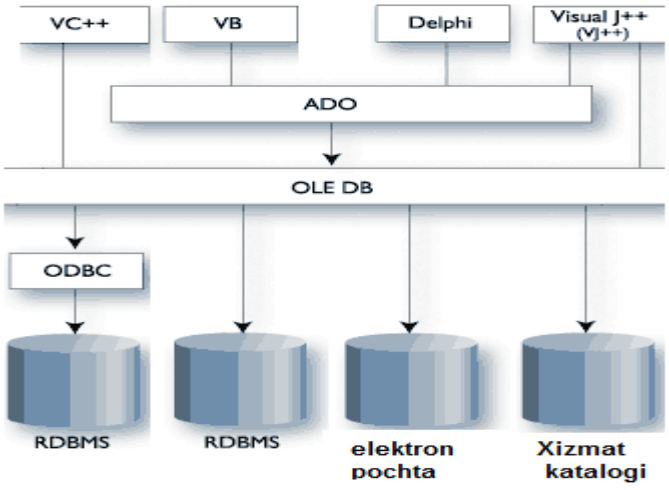
13.3 ADO

OLE DB C va C ++ dasturchilari, shuningdek C tilidagi funktsional qo'ng'iroqlarga ega tillardan foydalanuvchi dasturchilar uchun majburiydir. VB va VBScript kabi tillar ko'rsatgich ma'lumotlarini qo'llab-quvvatlamaydi (manzil o'zgaruvchilari). Shuning uchun, ular C uslubidagi ulanishdan va OLE DB-ga to'g'ridan-to'g'ri kirish huquqidan foydalana olmaydilar.

Ehtimol ko'proq chalkashlik uchun Microsoft dasturchilari yana bir ma'lumot kirish ob'ekti modelini taqdim etdilar: ADO. ADO DAO va RDO ob'ektlari bilan ishlaydi, shuningdek DAO va RDO-ga qaraganda sodda modellarni qo'llab-quvvatlaydi (garchi haddan tashqari funktsional imkoniyatlarga ega bo'lsa ham, siz operatsiyani bir necha usul bilan bajarishingiz mumkin).

ADO-da ob'ektlar ierarxiyasi DAO-ga qaraganda bir xil darajada. ADO bir nechta o'rnatilgan ob'ektlarni o'z ichiga oladi, ular ma'lumotlar omborlaridan ma'lumotlarga kirishni soddalashtiradilar.

13.2-rasmda dastur ma'lumotlar bazasi bilan bog'lanishning bir necha yo'li ko'rsatilgan. Masalan, VB dasturchisi OLE DB provayderiga dasturni ulash uchun ADO-dan foydalanishi mumkin. Agar ma'lumotlar bazasi OLE DB-ni qo'llab-quvvatlamasa, dastur ODBC-dan foydalanishi mumkin. Visual C ++ dasturchisi ADO dasturidan foydalanishi yoki to'g'ridan-to'g'ri OLE DB orqali ulanishi mumkin.



13.2-rasm. ADOda dastur yo'nalishlarining farqlanishi.

Recordset ob'ekti yozuvlar to'plami (jadval) va adOpenForwardOnly, adOpenKeyset, adOpenDynamic va adOpenStatic kursor turlarini qo'llab-quvvatlaydi. Kursor server tomonida (sukut bo'yicha) yoki mijoz tomonida bo'lishi mumkin.

ADO yozuviga kirish uchun qatorlarni ketma-ket skanerlash kerak. Bir nechta jadvallarga kirish uchun qatorlar to'plamida natijani olish uchun JOINga qo'shilish so'rovini to'ldirishingiz kerak. Recordset ob'ekti ma'lumotlarga ulanmasdan kirishni qo'llab-quvvatlasa ham, ADO dastlab unga ulangan ma'lumotlar uchun yaratilgan. Ushbu kirish usuli sizni muhim manbalarni server tomonida saqlashga majbur qiladi. Bunga qo'shimcha ravishda, satrlar to'plamini o'tkazish uchun siz COM marshalling deb nomlangan buyurtma usulidan foydalanishingiz kerak. COM marshalling - bu foydali tizim resurslarini tabiiy ravishda o'zlashtiradigan ma'lumot turlarini o'zgartirish jarayoni.

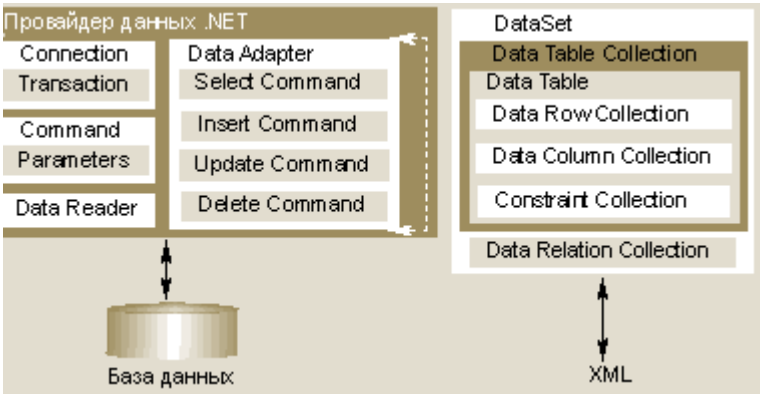
ADO 2.1-dan boshlab, Microsoft Recordset satrini to'plamini XML hujjati sifatida saqlashga imkon beradigan ADO ob'ekt modeliga XML ko'magi qo'shdi. Biroq, faqat ADO 2.5 paydo bo'lishi bilan ADO 2.1-da saqlangan bir qator XML cheklovlari olib tashlandi (masalan, Recordset ob'ektlarining qattiq ierarxiyasi). Garchi ADO XML hujjatini Recordset-ga o'zgartirishi mumkin bo'lsa-da, u faqat o'z ma'lumotlarini Advanced Data TableGram (ADTG) deb nomlanuvchi o'z sxemasida o'qiy oladi.

Bog'lanmagan ma'lumotlarga kirish mexanizmini qidirishda, Microsoft ADO-ni kengaytiradi va Remote Data Services (RDS) ni joriy qiladi. RDS ADO-dan keyin yaratilgan va faol ulanish yo'qligida Recordset ob'ektini mijozga (masalan, veb-brauzerga) uzatishga imkon beradi. Shu bilan birga, RDS, ADO singari, satrlarning to'plamini serverdan mijozga uzatish uchun COM marshaling buyrug'idan foydalanadi.

NET davri

Microsoft.NET Frameworkni ishlab chiqishni boshlaganda, ma'lumotlarga kirish modelini qayta ko'rib chiqish uchun yaxshi imkoniyatga ega bo'ldi. ADO texnologiyasini ishlab chiqishni davom ettirishga qaror qilgandan so'ng, Microsoft mutaxassislari qisqartirishni saqlagan holda ma'lumotlarga kirishning yangi tuzilishini yaratishga kirishdilar. Microsoft allaqachon tasdiqlangan ADO ob'ekt texnologiyasi asosida ADO.NET-ni rivojlantirmoqda. Ammo ADO.NET ADO tomonidan qo'llab-quvvatlanmaydigan uchta muhim xususiyatlarga e'tiborni qaratadi: Internetda ishlash uchun asosiy element bo'lgan aloqador bo'lmagan ma'lumotlarga kirish modelini qo'llab-quvvatlash; XML bilan qattiq integratsiyani qo'llab-quvvatlash; NET Framework bilan integratsiya (masalan, odatdagi tizimning bazaviy sinf kutubxonasi bilan moslik).

ADO.NET arxitekturasi 11.3-rasm ADO.NET mimarisini dalolat beradi. ADO-da juda ko'p funktsiyalarni bajaradigan Recordset ob'ekti yo'q. Buning o'rniga ADO.NET ma'lum vazifalarni bajaradigan bir nechta maxsus ob'ektlarni taqdim etadi. 1-jadvalda ulardan uchtasi tasvirlangan: DataAdapter, DataReader va DataSet.



11.3-rasm. ADO.NET arxitekturasi

NET ma'lumotlar etkazib beruvchilari. NET ma'lumotlar ta'minotchisi ADO.NETning juda muhim tarkibiy qismi ADO.NET interfeyslarini amalga oshiradi. Xususan, u DataReader dasturini ham, ham dasturni ishlatishda ishlatilishini ta'minlaydi.

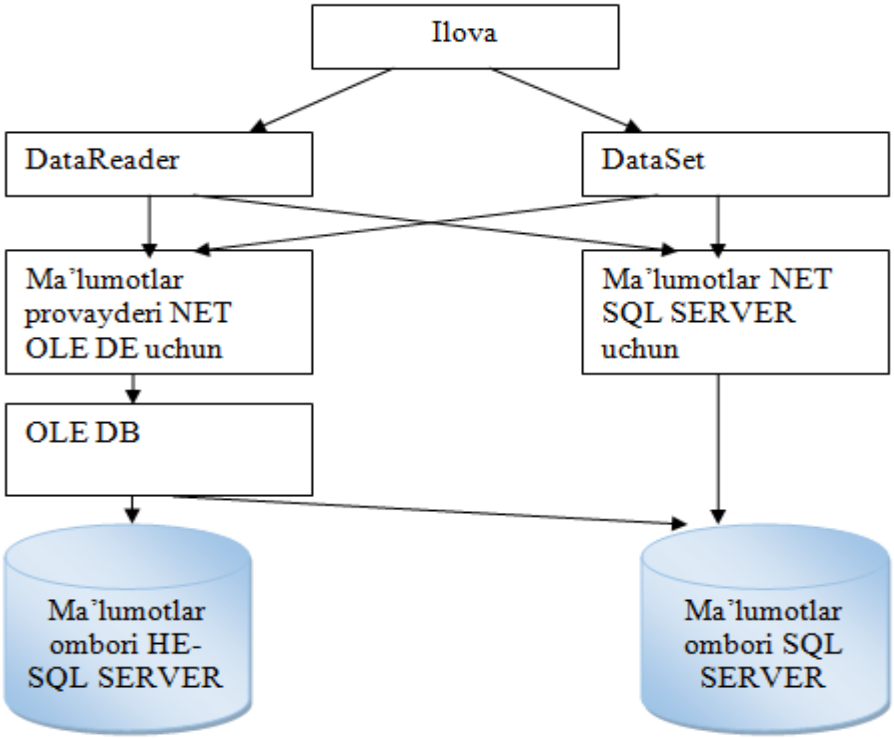
.NET ma'lumotlar etkazib beruvchisi to'rtta asosiy tarkibiy qismdan iborat: Ulanish - ma'lumotlar manbai bilan aloqa qilish uchun; Buyruq ma'lumotlar manbasidagi buyruqlarni bajaradi; DataReader ma'lumot manbaidan ma'lumotni faqat bitta yo'nalishli rejimda o'qiydi va DataAdapter ma'lumot manbasidan ma'lumotlarni o'qiydi va uni DataSet-ni to'ldirish uchun ishlatadi.

Visual Studio .NET ikkita .NET ma'lumotlar etkazib beruvchisini o'z ichiga oladi. SQL Server .NET ma'lumotlar provayderi SQL Server 7.0 va undan keyingi versiyalar bilan ulanishni ta'minlaydi. Ushbu kirish usuli SQL Server 7.0 va undan yuqori versiyalar uchun eng samaralidir, chunki SQL Server .NET ma'lumotlar ta'minotchisi to'g'ridan-to'g'ri SQL Server bilan Tabular Data Stream (TDS) protokoli orqali aloqa qiladi. OLE DB .NET ma'lumotlar provayderi Oracle yoki IBM DB2 kabi SQL Server bo'lmagan ma'lumotlar bazalariga ulanish uchun talab qilinadi. Ushbu ma'lumotlar etkazib beruvchisi tegishli ma'lumotlar bazalari uchun OLE DB-dan foydalanadi.

Ushbu yozuvni tayyorlash paytida Microsoft ishlab chiquvchilari .NET ma'lumotlar ta'minotchisining uchinchi turini, ODBC .NET ma'lumotlar etkazib beruvchisi, Beta nomzod Beta-versiyasini ishlatishdi. Buni Microsoft veb- *saytida olish mumkin: http://www.microsoft.com/data/ download_odbcnetrc.htm*.

11.4-rasmda dastur ADO.NET orqali ma'lumotlar bazasi bilan bog'lanishning turli usullari ko'rsatilgan. Yo'lni tanlashda, birinchi navbatda .NET ma'lumotlar etkazib beruvchisi ishlatilishi aniqlanadi. Agar bu SQL Server 7.0 yoki undan keyingi versiya bo'lsa, u holda SQL Server.NET ma'lumotlar ta'minotchisi ulangan. Agar sizda SQL Server 6.5 ma'lumotlar bazasi yoki SQL Server bo'lmagan ma'lumotlar bazasi (masalan, Oracle) bo'lsa, sizga OLE DB .NET ma'lumotlar ta'minotchisi kerak bo'ladi. SQL 7.0 va undan yuqori ma'lumotlar bazalari uchun OLE DB .NET ma'lumotlar provayderidan foydalanishingiz mumkinligini unutmang, ammo keyin TDS protokoli orqali SQL Serverga to'g'ridan-to'g'ri ulanishni ta'minlaydigan samaradorlik yo'qoladi. Biroq, ushbu o'ziga xos bo'lmagan usulning o'ziga xos ortiqcha - harakatchanligi, ya'ni kodni o'zgartirmasdan

ma'lumotlar bazasini o'zgartirishingiz mumkin.



11.4- rasm. ADO.NET-da yo'nalishlarning farqlanishi.

Keyinchalik, qaysi vazifani bajarishni xohlayotganingizni aniqlashingiz kerak. Agar siz faqat ma'lumot manbasidan ma'lumotlarni o'qish va namoyish etishingiz kerak bo'lsa, unda "Reader" ob'ekti kifoya qiladi. Ammo agar siz ma'lumotlarni boshqarishingiz kerak bo'lsa (masalan, tahrirlash yoki o'chirish), siz ma'lumotlar to'plamidan foydalanishingiz kerak. Garchi siz ushbu ob'ektdan faqat kerak bo'lsa foydalanishingiz kerak, chunki u ma'lumotlarni Reader-ga nisbatan sekin ishlaydi (Ma'lumotlar to'plami jadvallarni to'ldirish uchun Data Reader-dan foydalanadi).

JDBC.

JDBC juda oddiy - bu ma'lumotlarga kirish APIsi. Hurmatli o'quvchi biz ma'lumotlar bazalari haqida emas, balki "jadval ma'lumotlari" haqida gapirayotganimizni payqaydi. Bir qarashda farq unchalik katta bo'lmasligi mumkin, ammo aslida o'ta muhim. Masalan, matnli fayllarga, Microsoft Excel elektron jadvallariga kirish uchun JDBC drayverlari mavjud, ya'ni bunday ma'lumotlarga hech qanday ma'lumot kiritilmasligi mumkin: bitimlar, indekslar, munosabatlar va hokazolarni qo'llab-quvvatlash bilan.

Shuni ta'kidlash kerakki, JDBC nafaqat SQL-ga mos keladigan ma'lumotlar bazalari bilan ishlashni, balki jadval turidagi deyarli har qanday ma'lumotlar bilan ishlashni ham qo'llab-quvvatlaydi. Aytish to'g'ri bo'lsa-da, JDBC-ning kuchi, albatta, matn faylidan ma'lumotlarni o'qimaslikdir, chunki ular "chumchuq to'pidan" - tranzaktsiya mexanizmidan foydalanmasdan, saqlangan protseduralarga kirish va hk. JDBC SQL SQL ma'lumotlar bazasining "jozibasi" bo'lolmaydi.

JDBC texnologiyasini hech bir joyda Sun Microsystems bitta kompaniyasining texnologiyasi deb atash mumkin emas - Java otasi. Hozirda quyidagi tashkilotlar JDBC texnologiyasini rasman qo'llab-quvvatlamoqda:

- Oracle
- DataDirect texnologiyalari
- Go'zal
- Fujitsu
- MySQL
- INET dasturi
- Novell
- Borland
- Pointbase Inc.
- Makromedia
- Dastur

JDBC API interfeyslar kontseptsiyasidan intensiv ravishda foydalanmoqda - bu xizmat provayderi tomonidan amalga oshirilishi kerak bo'lgan usullar to'plamidir, bu holda provayder deb ataladi. JDBC drayverlari. Dasturiy ta'minot nuqtai nazaridan, JDBC drayveri JDBC API tomonidan ta'minlangan interfeyslarni amalga oshirishdan boshqa narsa emas. Amalga oshirish usuliga ko'ra haydovchilar 4 turga bo'linadi:

1 tur

Ushbu turga ODBC drayverlari ustiga o'rnatilgan drayverlar kiradi (biz bu erda ODBC nima ekanligini tushuntirmaymiz, agar kimdir bilmasa asl manbaga murojaat qiling). Ya'ni, deyarli barcha JDBC API qo'ng'iroqlar ODBC qo'ng'iroqlarga tarjima qilinadi va keyin ODBC API qo'ng'iroqni qayta ishlaydi. Ba'zan boshqa turdagi haydovchi "JDBC-ODBC ko'prigi" deb nomlanadi. Ushbu turdagi drayverlarning afzalligi shundaki, ODBC orqali kirish mumkin bo'lgan barcha ma'lumot manbalari Java dasturlarida mavjud bo'ladi. Bu drayverning kamchiliklari past tezlik, konfiguratsion qiyinchiliklar va JDBC API-ning barcha xususiyatlarini qo'llab-quvvatlamaslikdir.

2 turi

Ikkinchi turga boshqa tillarda (odatda C tilida) yozilgan dasturiy ta'minot qismlaridan foydalanadigan drayverlar kiradi. Odatda, bu holda, ma'lumotlar bazasiga kirish uchun ishlab chiqaruvchi tomonidan ishlab chiqilgan kutubxonalar ishlatiladi va mahalliy funktsiyalarni chaqirish uchun JNI - Java interfeysi ishlatiladi. Bunday haydovchining misoli deb ataladigan narsa. Oracle uchun "qalin" OCI-JDBC drayveri. Bunday drayverlar odatda juda tez, lekin yana, JDBC-ODBC drayverlarida bo'lgani kabi, mijoz mashinasida maxsus dasturiy ta'minot o'rnatilishini talab qiladi. Masalan, Oracle OCI-JDBC drayveri SQL * NET mijozini o'rnatishni talab qiladi.

3 tur

Oldingi turdagi drayverlardan farqli o'laroq, ushbu turdagi drayvlar Java-da to'liq amalga oshiriladi, ammo shu bilan birga JDBC qo'ng'iroqlari tarmoq protokoliga (RMI, HTTP va boshqalar) tarjima qilinadi, keyinchalik ma'lum

ma'lumotlar bazasi protokoliga tarjima qilinadi. Qaysidir ma'noda, bu drayver JDBC-ODBC drayverlariga o'xshaydi, farq shundaki, u to'liq Java-da amalga oshiriladi, shu sababli mijoz dasturini o'rnatishga hojat yo'q.

4 turi

3-turdagi drayverlarga o'xshab, u to'liq Java-da amalga oshiriladi, ammo tarmoq qo'ng'iroqlarini chetlab o'tib to'g'ridan-to'g'ri ma'lumotlar bazasi protokoli yordamida qo'ng'iroqlar amalga oshiriladi.

Bu erda shuni ta'kidlash kerakki, drayverlar ishlab chiqaruvchilari ko'pincha bu yoki boshqa turni, to'liq moslikni va boshqalarni e'lon qilishadi. Ammo, haqiqiy hayotda, haydovchilarning faqat chorak qismi JDBC muvofiqlik sertifikatiga ega.

Mavjud JDBC drayverlari haqida to'liq ma'lumotni JDBC-ning ildiz manbasiga kirish orqali olishingiz mumkin: http://java.sun. com/ products/ jdbc/. JDBC

drayverlarining turlari, ishlab chiqaruvchilari va sertifikatlar mavjudligi ro'yxati: bu erda:http://servlet.java.sun.com/products/jdbc/drivers.

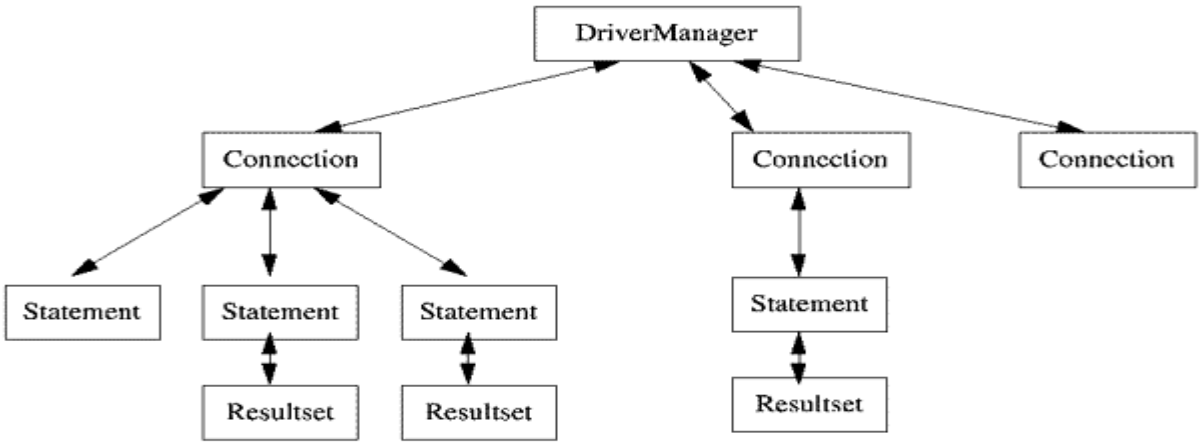
Ma'lumotlar bazasi 200 ga yaqin drayverlarga ega va qulay qidirish va navigatsiya tizimi bilan jihozlangan.

JDBC API ikkita asosiy interfeys turini o'z ichiga oladi: birinchisi dastur ishlab chiqaruvchilari uchun, ikkinchisi (pastki daraja) haydovchilar ishlab chiqaruvchilari uchun.

Ma'lumotlar bazasiga ulanish java.sql.Connection interfeysini amalga oshiruvchi sinf tomonidan tasvirlangan. Ma'lumotlar bazasiga ulanib, SQL-da ma'lumotlar bazasiga so'rovlarni bajarish uchun foydalaniladigan Statement turidagi ob'ektlarni yaratishingiz mumkin.

Maqsadida farq qiluvchi quyidagi bayonotlar turlari mavjud:

- java.sql.Statement - Umumiy maqsadlar uchun bayonot;
- java.sql.PreparedStatement - O'zgartirilgan parametrlarni o'z ichiga olgan so'rovlarni bajarish uchun ishlatiladigan so'rov (so'rov tanasida '?' bilan ko'rsatilgan);
- java.sql.CallableStatement - saqlangan protseduralarni chaqirish uchun mo'ljallangan bayonot.
- java.sql.ResultSet so'rov natijalarini qayta ishlashni osonlashtiradi.



11.5-rasm. JDBC asosiy interfeyslari.

Java.sql.Statement ifoda interfeysi boshqa ikkita muhim interfeyslarning ajdodlari vazifasini bajaradi: java.sql.PreparedStatement va java.sql. CallableStatement, birinchisi oldindan tuzilgan SQL iboralarini bajarish uchun ishlatiladi, ikkinchisi - saqlangan protseduralarni bajarish. Shunga ko'ra, Statement muntazam (statik) SQL so'rovlarini bajaradi va ko'rsatilgan ikki avlod parametrlangan SQL ifodalari bilan ishlaydi.

Savollar:

1. Ma'lumotlarga kirishning asosiy texnologiyalarini sanab bering.
2. Open database connectivity nima?
3. Nima OLEDB ?
4. OLEDB qanday ishlaydi ?
5. Nima JDBC ?
6. JDBC qanday ishlaydi ?

Adabiyotlar:

1. Thomas Connolly, Carolyn Begg – Database systems. A practical Approach to Design, Implementation and Management. 4th Edition – Addison Wesley 2005 – 1373p.
2. C. J. Date – An Introduction to Database Systems – Addison-Wesley Professional – 2003 – 1024 p.

XIV Bob. C ++ va SQL orqali yangi ma’lumotlar qo'shish, o’zgartirish va o'chirish

14.1 TData Base sinfi.

Ma'lumotlar bazalari bilan ishlashda TDataBase turidagi ob'ekt talab qilinmaydi, ammo u ma'lumotlar bazasi ulanishlarini boshqarish uchun bir qator qo'shimcha funktsiyalarni taqdim etadi. TDataBase quyidagilar uchun ishlatiladi:

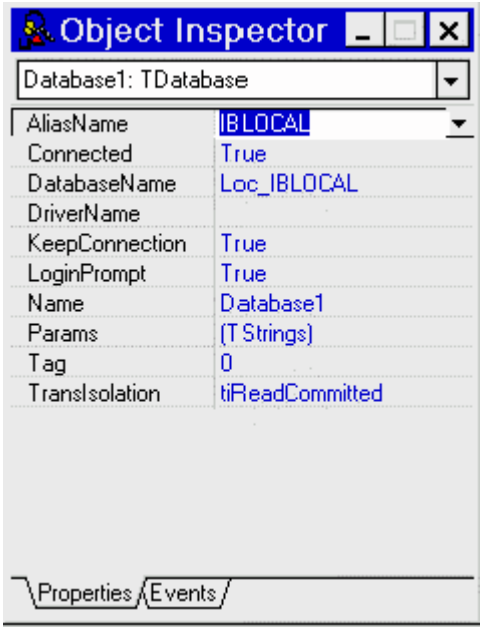
- Doimiy ma'lumotlar bazasi ulanishini yaratiladi
- Ma'lumotlar bazasiga ulanishda o'zingizning dialogingizni aniqlash (parol so'rovi)
- Mahalliy ma'lumotlar bazasi taxallusini yaratish
- Ulanishni o'zgartirish
- Tranzaktsiyalarni boshqarish

TDataBase - ko'rinmas ish vaqti. Komponentlar palitrasining ma'lumotlarga kirish sahifasida joylashgan. TDataBase-ni loyihaga kiritish uchun uni dasturning asosiy oynasiga "qo'yish" kerak.

Ma'lumotlar bazasi bilan doimiy aloqani yaratish

Agar siz ma'lumotlar bazasi bilan ishlayotgan bo'lsangiz, unda ishlashni boshlashdan oldin ushbu ma'lumotlar bazasiga ulanish tartibi amalga oshiriladi. Ulanish protsedurasi, shu qatorda, foydalanuvchi nomi va parolni so'rashni o'z ichiga oladi (IDPI orqali mahalliy Paradox va dBase jadvallari bilan ishlashdan tashqari). Agar dastur TDataBase-dan foydalanmasa, ulanish jarayoni ma'lumotlar bazasidan birinchi jadval ochilganda amalga oshiriladi. Dasturda ushbu

ma'lumotlar bazasidagi so'nggi jadval yopilganda , ma'lumotlar bazasiga ulanish to'xtatiladi (agar bu *Session* ob'ektining *Keep Connections* xususiyati " False" ga o'rnatilgan bo'lsa, lekin bundan keyin). Endi yana jadvalni ochsangiz, ulanishni sozlash jarayoni takrorlanadi va bu foydalanuvchi uchun juda noqulay bo'lishi mumkin. Ushbu ma'lumotlar bazasida ochiq jadvallar bo'lmagan taqdirda ham ulanishning oldini olish uchun TDataBase turidagi komponentdan foydalanishingiz mumkin. AliasName xususiyatida dastur ishlaydigan ma'lumotlar bazasining taxalluslarini ko'rsating; DatabaseName xususiyatida eski ma'lumotlar taxalluslari o'rniga jadvallar murojaat qiladigan har qanday nom (ma'lumotlar bazasi taxallusi). Connected xususiyatini "true" ga qo'ying - ma'lumotlar bazasiga ulanish tartibi dastur boshlanganda amalga oshiriladi. Va nihoyat, KeepConnection xususiyati "true" ga o'rnatilishi kerak (14.1-rasmga qarang).

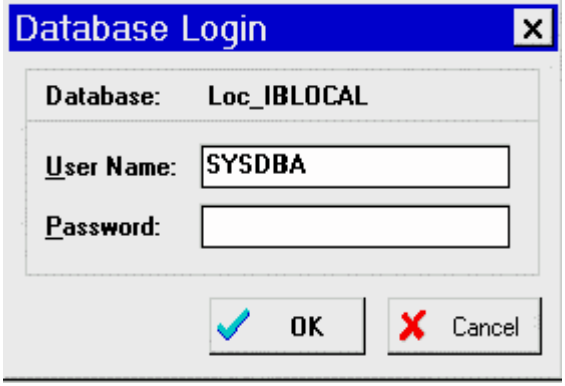


14.1-rasm. Obyekt inspektoridagi TDataBase xususiyatlari

Bizning misolimizda, DataBase1 xususiyatlarini o'rnatgandan so'ng, DatabaseName xossasida IBLOCAL bilan ishlaydigan barcha jadvallar uchun Loc_IBLOCALni o'rnatishimiz kerak.

Ma'lumotlar bazasiga ulanishda shaxsiy dialogini aniqlash

Odatiy bo'lib, ma'lumotlar bazasiga ulanishda 14.2-rasmda ko'rsatilgan foydalanuvchi nomi va parolni so'rash oynasi ishlatiladi



14.2-rasm. Foydalanuvchi bilan aloqa oynasi

Agar so'ralsa, siz dialogning ko'rinishini o'zgartirishingiz yoki uni butunlay bekor qilishingiz mumkin. Buning uchun TDataBase sinfining xususiyatlari va hodisalari - *LoginPrompt, Params* va *OnLogin ishlatiladi.*

Foydalanuvchi nomi va parolni so'rovni o'chirib qo'yish uchun, "Login" so'rovini "False" ga qo'ying. Bunday holda, Params xossasi aniq (foydalanuvchi dizayni yoki ish vaqtida) foydalanuvchi nomi va parolni ko'rsatilishini talab qiladi. Masalan, dasturda yozishingiz mumkin (ma'lumotlar bazasiga ulanmaguncha, masalan Form1 OnCreate uchun hodisada):

```
DataBase1->LoginPrompt=False;
DataBase1->Params->Clear;
DataBase1->Params->Add('USER NAME=SYSDBA');
DataBase1->Params->Add('PASSWORD=masterkey');
DataBase1->Connected=True;
```

14.2 Tranzaktsiyalarni boshqarish

TDataBase sizga ma'lumotlar bazasida tranzaktsiyalarni (StartTransaction usuli) boshlash, bajarish (bajarish) yoki orqaga qaytarish (RollBack) imkonini beradi. Bundan tashqari, siz tranzaksiyani izolyatsiya qilish darajasini o'zgartirishingiz mumkin (TransIsoltion mulki).

TransIsolation	Oracle	Sybase and Microsoft SQL	Informix	InterBase
Dirty read	Read committed	Read committed	Dirty Read	Read committed
Read committed(Default)	Read committed	Read committed	Read committed	Read committed
Repeatable read	Repeatable read	Read committed	Repeatable Read	Repeatable Read

"Dirty read" - joriy tranzaktsiyangiz ichida boshqa bitimlar tomonidan qilingan barcha o'zgarishlarni ko'rishingiz mumkin, garchi ular hali majburiyatni bajarmagan bo'lsa ham. " Read Committed " - faqat ma'lumotlar bazasiga kiritilgan "qilingan" o'zgarishlar ko'rinadi. " Repeatable Read" - tranzaktsiya ichida siz tranzaktsiyani boshlash paytida ma'lumotlar bazasida bo'lgan ma'lumotlarni ko'rishingiz mumkin, hatto u erda haqiqatan ham o'zgarishlar bo'lsa ham.

Session ob'ekti.

TSession tipidagi Session ob'ekti ma'lumotlar bazalari bilan ishlaydigan dasturda avtomatik ravishda yaratiladi (bu holda Delphi dasturga MB modulini ulaydi). Ushbu ob'ektni yaratish yoki yo'q qilish haqida tashvishlanishga hojat yo'q, lekin ba'zi hollarda uning usullari va xususiyatlari foydali bo'lishi mumkin. Ushbu komponentda dastur ishlaydigan barcha ma'lumotlar bazalari to'g'risidagi ma'lumotlar mavjud. Buni *ma'lumotlar bazasi* s-da topish mumkin . Ushbu ob'ektning *KeepConnections* xususiyati bilan biz allaqachon tanishamiz. Ushbu

dastur, agar dasturda ushbu ma'lumotlar bazasidan ochiq jadvallar bo'lmasa, ma'lumotlar bazasiga ulanishni saqlab qolish kerakligini aniqlaydi. *NetDir* - bu *PDOXUSRS.NET* umumiy tarmoq faylini BDE tomonidan talab qilinadigan katalog. *PrivateDir* - vaqtinchalik fayllarni saqlash uchun katalog.

Session ob'ektining usullaridan foydalanib, siz BDE sozlamalari haqida ma'lumot olishingiz mumkin, masalan, barcha taxalluslar ro'yxati, ma'lumotlar bazasi drayverlari yoki ma'lumotlar bazasidagi barcha jadvallar ro'yxati.

Seans ob'ektining yana bir muhim maqsadi parol bilan himoyalangan Paradoks jadvallariga kirishdir. Bunday jadvalni ochishdan oldin *AddPassword* usulini bajarish kerak :

```
Seans-> AddPassword ('my_pass');
```

Siz parolni RemovePassword yoki RemoveAllPasswords usuli yordamida olib tashlashingiz mumkin.

TDataSet sinfi.

TDataSet sinfi ma'lumotlar bazasining eng muhim ob'ektlaridan biridir. U bilan ishlashni boshlash uchun siz quyidagi ierarxiyani ko'rib chiqishingiz kerak:

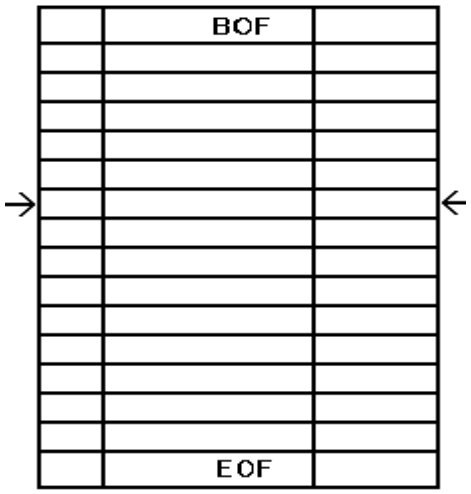
```
TDataSet
        |
   TDBDataSet
        |
        |-- TTable
        |-- TQuery
        |-- TStoredProc
```

TDataSet-da to'g'ridan-to'g'ri ma'lumotlarni boshqarish kerak bo'lgan mavhum usullar mavjud. TDBDataSet parollarni qanday boshqarish kerakligini va sizni ma'lum bir jadvalga qo'shilish uchun nima qilish kerakligini biladi. TTable jadvalni, uning indekslarini va boshqalarni qanday boshqarishni biladi (ya'ni, barcha mavhum usullar allaqachon qayta yozilgan).

Keyinchalik ko'rib turganingizdek, TQuery SQL so'rovlarini qayta ishlashning ma'lum usullariga ega.

TDataSet - bu jadvalni ochish va uning ichida harakatlanish uchun foydalanadigan vositadir. Albatta, siz hech qachon TDataSet turidagi ob'ektni yaratolmaysiz. Buning o'rniga siz TTable, TQuery yoki boshqa TDataSet avlodlaridan (masalan, TQBE) foydalanasiz. Tizim haqida to'liq tushuncha va TDataSet-ning aniq ma'nosi ushbu bobni o'qiyotganda aniqroq bo'ladi.

Eng fundamental darajada, 14.3-rasmda ko'rsatilganidek, ma'lumotlar bazasi shunchaki yozuvlar to'plamidir



14.3-rasm. Har qanday ma'lumotlar to'plami bir qator yozuvlardan iborat (har birida N maydonlar mavjud) va joriy yozuvning ko'rsatgichi.

Ko'pgina hollarda, ma'lumotlar to'plamida diskda mavjud bo'lgan jadval bilan bevosita, bittadan bitta yozishmalar bo'ladi. Ammo, boshqa holatlarda, bitta jadvaldagi yozuvlarning har qanday to'plamini yoki bir nechta jadvallar orasidagi birikmani o'z ichiga olgan ma'lumotlar to'plamini qaytaradigan so'rovni yoki boshqa amallarni bajarishingiz mumkin. Ba'zan matnda DataSet va TTable atamalaridan sinonim sifatida foydalaniladi.

Odatda, dastur TTable yoki TQuery tipidagi ob'ektlardan foydalanadi, shuning uchun keyingi bir necha boblarda 1-jadval deb nomlangan TTable ob'ekti mavjudligi taxmin qilinadi.

Shunday qilib, TDataSet-ni o'rganishni boshlash vaqti keldi. Uning imkoniyatlari bilan tanishganingizdan so'ng, Delphi ma'lumotlar bazasi sifatida diskda saqlangan ma'lumotlarga kirish uchun qanday usullardan foydalanishni tushunishni boshlaysiz. Bu erda asosiy nuqta shundan iboratki, Delphi dasturchisi deyarli har safar dasturni ochganda TDableSet uchun ba'zi bir qo'shimcha dasturlardan iborat bo'lgan TTable yoki TQuery-dan foydalanadi.

14.3 DataSet-ni ochish va yopish

Agar siz jadvalga kirish uchun TTable dasturidan foydalansangiz, unda ushbu jadvalni ochganingizda ba'zi TTable xususiyatlari to'ldiriladi (RecordCount yozuvlari soni, jadval tuzilishi tavsifi va boshqalar).

Avvalo, siz dizayn paytida TTable ob'ekti qo'yishingiz va qaysi jadval bilan ishlashni ko'rsatishingiz kerak. Buni amalga oshirish uchun Object Inspector-da DatabaseName va TableName xususiyatlarini to'ldiring. DatabaseName-da siz dBase yoki Paradox formatidagi jadvallar joylashgan katalogni belgilashingiz mumkin (masalan, C: \ DELPHI \ DEMOS \ DATA) yoki ro'yxatdan ma'lumotlar bazasi taxallusini (DBDEMOS) tanlashingiz mumkin. Ma'lumotlar bazasi taxallusi (Alias) Ma'lumotlar bazasi mexanizmi konfiguratsiya yordam dasturida aniqlangan. Endi, agar Active parametri "true" ga o'rnatilgan bo'lsa, dastur ishga tushganda jadval avtomatik ravishda ochiladi.

Dastur ishlayotgan vaqtda jadvalni ochishning ikki xil usuli mavjud. Siz

quyidagi kod satrini yozishingiz mumkin:

```
Table1->Open();
```

Yoki, agar xohlasangiz, Active xususiyatini True ga o'rnatishingiz mumkin:

```
Table1-> Active = True;
```

Ushbu ikkita operatsiya natijasida olingan natija o'rtasida hech qanday farqi yo'q. Ochiq usul, shu bilan birga Active xususiyatini True ga o'rnatish bilan yakunlanadi, shuning uchun to'g'ridan-to'g'ri Active xususiyatidan foydalanish yanada samaraliroq bo'lishi mumkin.

Jadvalni ochishning ikki yo'li mavjud bo'lganidek, uni yopishning ikki yo'li mavjud. Eng oson yo'li – Closeni chaqirishdir:

```
Table1->Close();
```

Agar xohlasangiz, yozishingiz mumkin:

```
Table1->Active = False;
```

Yana bir bor qaytaramiz, ikkalasi o'rtasida sezilarli farq yo'q. Siz faqat Ochish va Yopish usullar (protseduralar) ekanligini va Active bu xususiyat ekanligini yodda tutishingiz kerak.

Navigatsiya (yozuvlar bo'yicha navigatsiya).

Jadvalni ochganingizdan so'ng, keyingi qadamni aniqlash kerakmi? uning ichidagi yozuvlardan qanday o'tish kerak.

TDataSet-ning quyidagi keng qamrovli usullari va xossalari jadval ichidagi istalgan yozuvlarga kirish uchun zarur bo'lgan barcha narsalarni taqdim etadi:

- Chaqirish `Table1-> First` sizni jadvaldagi birinchi yozuvga olib boradi.
- `Table1-> Last` sizni oxirgi yozuvga `o'tkazadi .`
- `Table1-> Next` sizni bitta yozuvni oldinga siljitadi.
- `Table1-> Prior` sizga bitta yozuvni qaytarib beradi.
- Siz `BOF` yoki `EOF` xususiyatlarini jadvalning boshida yoki oxirida ekanligingizni tekshirishingiz mumkin.
- `MoveBy` protsedurasi jadvaldagi N yozuvlarini oldinga yoki orqaga siljitadi. `Table1->Next` va `Table1-> MoveBy (1)` chaqirish o'rtasida funktsional farq yo'q. Xuddi shunday, `Table1- >Prior-` ga qo'ng'iroq ham `Table1-> MoveBy(-1)-ga` qo'ng'iroq qilish bilan bir xil natijaga ega.

Ushbu navigatsiya usullaridan foydalanishni boshlash uchun siz oldingi darsda bo'lgani kabi TTable, TDataSource va TDBGrid-larni formaga qo'yishingiz kerak. DBGrid1-ni DataSource1-ga va 1-jadvalga DataSource1-ni ulang. Keyin jadval xususiyatlarini o'rnating:

- DatabaseName-da demo jadvallar joylashgan pastki katalog nomi (yoki DBDEMOS taxallusi);
- TableName-da, jadval nomini CUSTOMER-ga o'rnating.

Agar siz ko'rinadigan TDBGrid elementini o'z ichiga olgan dasturni ishlatsangiz, siz DBGridning pastki va o'ng tomonidagi o'tish paneli yordamida jadval yozuvlari orqali harakat qilishingiz mumkinligini ko'rasiz.

14.4 TQuery tushunchalari

TTable-dan foydalanishda bitta jadvaldagi yozuvlarning barcha to'plamiga kirish mumkin. TTable-dan farqli o'laroq, TQuery sizga o'zboshimchalik bilan (SQL doirasida) u bilan ishlash uchun ma'lumotlar bazasini tanlash imkonini beradi. Ko'p jihatdan TQuery ob'ekti bilan ishlash texnikasi TTable bilan ishlash texnikasiga o'xshaydi, ammo ba'zi o'ziga xos jihatlari ham bor.

Siz TQuery komponentasidan foydalanib SQL so'rovini quyidagicha yaratishingiz mumkin:

1. Alias DatabaseName nomini tayinlang.
2. SQL so'rovini kiritish uchun "Select * from Country" kabi SQL xususiyatidan foydalanish.
3. Active xususiyatini "true" ga o’zgartirish.

Agar chaqirish mahalliy ma'lumotlarga o'tadigan bo'lsa, unda taxallus o'rniga, jadvallar joylashgan katalogga to'liq yo'lni belgilashingiz mumkin.

SQL xususiyati

SQL xususiyati, ehtimol TQuery-ning eng muhim qismidir. Ushbu mulkka kirish ob'ektni loyihalash paytida yoki dasturni bajarish vaqtida (ishga tushirish vaqti) ob'ekt nazoratchisi orqali amalga oshiriladi.

Albatta, so'rovni dinamik ravishda o'zgartirish uchun SQL xususiyatiga ish vaqtida kirish juda qiziq. Masalan, agar siz uchta SQL so'rovini bajarmoqchi bo'lsangiz, unda uchta TQuery komponentlarini forma ustiga joylashtirishingiz shart emas. Buning o'rniga, siz birini qo'yishingiz va SQL xususiyatini uch marta o'zgartirishingiz mumkin. Eng samarali, eng oson va eng kuchli usul bu parametrlangan so'rovlar orqali amalga oshirish, bu keyingi bo'limda tushuntiriladi. Biroq, avval SQL xususiyatlarining asosiy xususiyatlarini ko'rib chiqamiz, so'ng parametrlari bo'lgan so'rov kabi murakkab mavzularni ko'rib chiqamiz.

SQL xususiyati TStrings turiga kiradi, ya'ni bu ro'yxatda saqlanadigan qatorlar qatorini anglatadi. Ro'yxat xuddi massiv kabi ishlaydi, lekin aslida u o'ziga xos imkoniyatlarga ega bo'lgan maxsus sinfdir. Keyingi bir nechta paragraf eng ko'p ishlatiladigan xususiyatlarni qamrab oladi.

TQuery dasturidan foydalanganda avval joriy so'rovni yopish va SQL xususiyatidagi qatorlar ro'yxatini tozalash tavsiya etiladi:

```
Query1-> Close ();
Query1-> SQL.Clear ();
```

Shuni yodda tutingki, siz har doim xavfsiz ravishda yaqinga chaqirishingiz mumkin. Agar so'rov allaqachon yopilgan bo'lsa ham, istisno yaratilmaydi.

Keyingi qadam so'rovga yangi qatorlar qo'shishdir.

```
Query1->SQL->Add('Select * from Country');
```

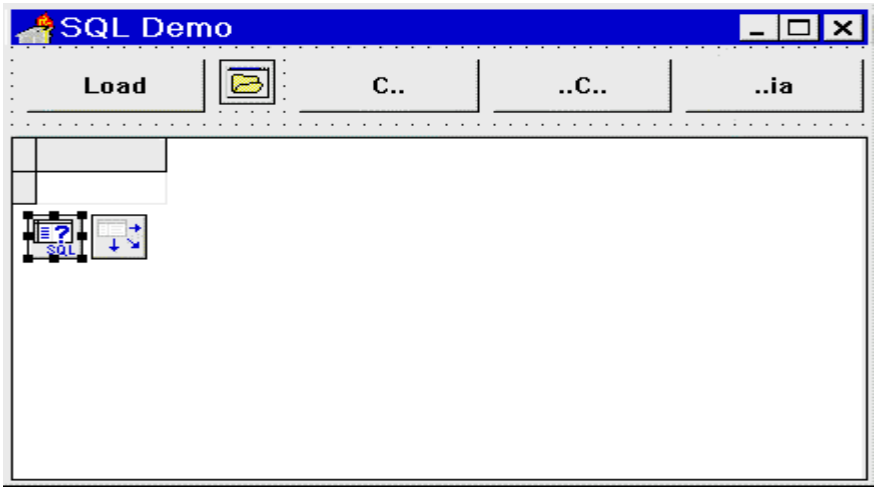
```
Query1->SQL->Add('where Name = ''Argentina''');
```

Qo'shish usuli SQL so'roviga bir yoki bir nechta satr qo'shish uchun ishlatiladi. Umumiy miqdor faqat sizning mashinangizdagi xotira miqdori bilan cheklangan.

Delphi so'rovni qayta ishlashi va natijani o'z ichiga olgan kursorni jadval rasmida qaytarishi uchun siz usulni chaqirishingiz mumkin:

```
Query1-> Open ();
```

THREESQL demo bu jarayonni namoyish etadi (14.4-rasmga qarang).



14.4-rasm. THREE SQL bitta TQuery ob'ekti yordamida bir nechta so'rovlarni qanday qilish kerakligini ko'rsatadi.

THREE SQL mahalliy SQL xususiyatidan foydalanadi. Masalan, quyidagi SQL so'rovi :

```
Select * form Country where Name like 'C%'
```

Ism maydoni 'C' harfi bilan boshlanadigan barcha yozuvlarni o'z ichiga olgan DataSet-ni qaytaradi. Quyidagi so'rov sizning ismingiz "C" harfini o'z ichiga olgan barcha davlatlarni ko'rishga imkon beradi:

```
Select * from Country where Name like '%C%';
```

Bu erda "ia" bilan tugaydigan barcha mamlakatlar topilgan so'rov:

```
Select * from Country where Name like '%ia';
```

SQL xususiyatining foydali xususiyatlaridan biri bu to'g'ridan-to'g'ri diskdan so'rov matni bo'lgan fayllarni o'qish qobiliyatidir. Ushbu xususiyat THREESQL dasturida ko'rsatilgan.

TQuery va parametrlar.

C ++ Builder parametrlangan so'rov deb nomlangan "moslashuvchan" so'rov formasini yaratishga imkon beradi. Bunday so'rovlar "where" yoki "insert" iboralaridagi bitta so'zlar o'rniga o'zgaruvchining qiymatini almashtirishga imkon beradi. Ushbu o'zgaruvchini deyarli har qanday vaqtda o'zgartirish mumkin (Agar mahalliy SQL-dan foydalansangiz, SQL-dagi deyarli har qanday so'zni almashtirishingiz mumkin, ammo ko'pgina serverlar ham shu xususiyatni qo'llab-

quvvatlamaydi).

Parametrli so'rovlardan foydalanishni boshlashdan oldin, yuqoridagi SQL ko'rsatmalaridan birini qayta ko'rib chiqing:

```
Select * from Country where Name like 'C%'
```

Siz ushbu iborani NameStr o'zgaruvchisining o'ng tomonini almashtirish orqali parametrlangan so'rovga aylantirishingiz mumkin:

```
select * from County where Name like :NameStr
```

Ushbu SQL bayonotida NameStr oldindan belgilangan doimiy emas va dizayn vaqtida yoki ish vaqtida o'zgarishi mumkin. SQL tahlil qiluvchisi (so'rov matnini tahlil qiladigan dastur), u doimiy emas, balki parametr bilan bog'liqligini tushunadi, chunki oldin yo'g'on ichakda ": NameStr" so'zi mavjud. Bu ustun Delphi-ga NameStr o'zgaruvchisini keyinchalik ma'lum bo'ladigan qiymat bilan almashtirishni buyuradi.

E'tibor bering, NameStr so'zi tasodifan to'liq tanlangan. Siz dasturdagi o'zgaruvchan identifikator tanlanganidek, har qanday haqiqiy o'zgaruvchini nomidan foydalanishingiz mumkin.

Parametrli SQL so'rovida o'zgaruvchiga qiymatni belgilashning ikkita usuli mavjud. TQuery ob'ektining Params xususiyatlaridan foydalanish. Ikkinchisi - boshqa DataSet-dan ma'lumot olish uchun DataSource xususiyatidan foydalanish.

Parametrlar so'rovidagi parametr qiymatini Params xususiyati orqali almashtirsangiz, odatda to'rt bosqichni bajarish kerak bo'ladi.

1. TQuery-ni yopish
2. Tayyorlash usulini chaqirib TQuery ob'ektini tayyorlash
3. Params xususiyatiga kerakli qiymatlarni tayinlash
4. TQuery-ni ochish

Ikkinchi savol, agar ushbu so'rov matni birinchi marta bajarilgan bo'lsa, kelajakda uni tashlab yuborish mumkin.

Buni amalda qanday bajarish mumkinligini ko'rsatuvchi kodi:

```
Query1->Close();
Query1->Prepare();
Query1->Params[0]->AsString = 'Argentina';
Query1->Open();
```

Ushbu kod biroz sirli ko'rinishi mumkin. Buni tushunish uchun sinchkovlik bilan bosqichma-bosqich tahlil qilish kerak. Boshlashning eng oson usuli uchinchi qatordan, chunki Params xususiyati bu jarayonning "yuragi" dir.

Params bu TDataSet uchun Fields xossasi kabi sintaksisga ega bo'lgan indekslangan xususiyatdir. Masalan, Params qatoridagi null elementga murojaat qilib, SQL so'rovida birinchi o'zgaruvchiga kirishingiz mumkin:

```
Params [0] -> AsString: = '"Argentina"';
```

Agar parametrlashtirilgan SQL so'rovi quyidagicha ko'rinsa:

```
select * from Country where Name = :NameStr
```

keyin yakuniy natija (ya'ni, aslida nima sodir bo'ladi) quyidagi SQL iborasi:

```
select * from Country where Name = "Argentina"
```

Hamma narsa o'zgaruvchan: NameStr-ga Params mulki orqali "Argentina" qiymati berilgan. Shunday qilib, siz oddiy SQL iborasini yaratishni tugatdingiz. Agar so'rovda bir nechta parametr mavjud bo'lsa, ularga Params xususiyatlarini o'zgartirish orqali kirishingiz mumkin;

```
Params [1] -> AsString = 'SomeValue';
```

yoki parametr nomi orqali kirishdan foydalangan holda;

```
ParamByName ('NameStr') -> AsString: = '"Argentina"';
```

Shunday qilib, parametrlashtirilgan SQL so'rovlari parametrlar qiymatlari qayerga o'tishini aniqlaydigan har doim yo'g'on nuqta bilan boshlanadigan o'zgaruvchilardan foydalanadi.

Params o'zgaruvchisidan foydalanishdan oldin, avval tayyorlab qo'yishni chaqirishingiz mumkin. Ushbu qo'ng'iroq Delphi sizning SQL so'rovingizni tahlil qilishiga va Params xususiyatini kerakli parametrlarni "qabul qilishga" tayyor bo'lishi uchun tayyorlashga majbur qiladi. Params o'zgaruvchisiga avval "Oldindan tayyorgarlik" deb qo'ng'iroq qilmasdan qiymat belgilashingiz mumkin, ammo bu biroz sekinroq ishlaydi.

Parametrlar o'zgaruvchisiga kerakli qiymatlarni tayinlaganingizdan so'ng, siz tayyorlanishga qo'ng'iroq qilganingizdan so'ng, o'zgaruvchini bog'lashni tugatish va kerakli ma'lumotlar to'plamini olish uchun "Ochish" ni chaqirishingiz kerak. Bizning holda, DataSet-da "Ism" maydonida "Argentina" yozuvi bo'lgan yozuvlar bo'lishi kerak.

Savollar:

1. TDatabase yordamida ulanishni qanday boshqarish kerak ?
2. TSession ob'ekti nima va u nima uchun kerak?
3. TDataset-da qanday parametrlarni o'rnatish mumkin ?
4. TQuery bu nima?
5. TQuery bilan qanday ishlash kerak ?
6. TQuery-da parametrlashtirilgan so'rovlardan qanday foydalanish kerak ?

Adabiyotlar:

1. Thomas Connolly, Carolyn Begg – Database systems. A practical Approach to Design, Implementation and Management. 4th Edition – Addison Wesley 2005 – 1373p.
2. C. J. Date – An Introduction to Database Systems – Addison-Wesley Professional – 2003 – 1024 p.
3. Послед Б.С. Borland C++ Builder 6. Разработка приложений баз данных СПб.: ООО «ДиаСофтЮП», 2003 — 320 с.

XV Bob. ADO texnologiyasidan foydalanish. Ma’lumotlar bazasiga murojaatni tashkil etishda ADO va C ++ dan foydalanish

15.1 ADO texnologiyasi

Microsoft ActiveX Data Objects (ADO) texnologiyasi ma'lumotlar bazasi ilovalaridan turli xil ma'lumot manbalariga kirish uchun universal mexanizmdir. ADO texnologiyasining asosi OLE DB spetsifikatsiyasida tavsiflangan COM ob'ektlarining umumiy modeli uchun interfeys to'plamidan foydalanish hisoblanadi. Ushbu texnologiyaning afzalligi shundaki, OLE DB interfeyslarining asosiy to'plami har bir zamonaviy Microsoft operatsion tizimida mavjud. Bundan kelib chiqadiki, ilovalarga ma'lumotlarga kirishni ta'minlashning soddaligi. ADO texnologiyasidan foydalangan holda ma'lumotlar bazasi ilovasi elektron jadvallardan, mahalliy va server ma'lumotlar bazalari jadvallaridan, XML fayllaridan va boshqalardan ma'lumotlardan foydalanishi mumkin.

ADO terminologiyasiga muvofiq har qanday ma'lumot manbai (ma'lumotlar bazasi, fayl, elektron jadval) ma'lumotlar bazasi deb ataladi. Ilova provayder yordamida ma'lumotlar bazasi bilan o'zaro ishlaydi. Ma'lumotlar bazasining har bir turi o'zining ADO provayderidan foydalanadi. Provayder ma'lumotlar bazasiga so'rovlar, qaytarilgan xizmat ma'lumotlarini sharhlash va ilovaga uzatish uchun so'rovni bajarish natijalari bilan kirishni ta'minlaydi.

Barcha ADO ob'ektlari va interfeyslari COM ob'ektlari va interfeyslardir. OLE DB spetsifikatsiyasiga ko'ra, COM quyidagi ob'ektlar to'plamini o'z ichiga oladi:

- Buyruq (buyruq) - buyruqlarni qayta ishlash uchun ishlatiladi (odatda SQL-so'rovlar);
- Ma'lumot manbai - ma'lumotlar etkazib beruvchisi bilan aloqa qilish uchun ishlatiladi;
- Hisoblagich (hisobchi) - ADO provayderlarini sanash uchun ishlatiladi;
- Xato - istisnolar haqida ma'lumot mavjud;
- Rowset (qatorlar to'plami) - buyruq bajarilishi natijasida olingan ma'lumotlar qatorlari;
- Sessiya - yagona ma'lumotlar bazasiga kiradigan ob'ektlar to'plami;
- Transaction - OLE DB-da operatsiyalarni boshqarish.

C++ Builder dasturlash tizimida **ADO** texnologiyasidan foydalangan holda dasturlarni yaratish uchun ishlatiladigan komponentlar komponentlar palitrasining ADO sahifasida joylashgan. Keling, ushbu tarkibiy qismlarning maqsadini qisqacha tasvirlab beramiz:

- ADOConnection - ADO ulanishi, ADO ma'lumotlar manbasi bilan aloqani o'rnatish uchun ishlatiladi va tranzaktsiyalarni qo'llab-quvvatlaydi;
- ADOCommand - ADO-buyruqlari natijasida olingan ma'lumotlar to'plamini qaytarmasdan ADO-ma'lumotlar manbasiga kirish uchun SQL-buyruqlarni bajarish uchun ishlatiladi;

- ADODataSet - bu ADO ma'lumotlar manbasining bir yoki bir nechta jadvaliga kirishni ta'minlaydigan va boshqa tarkibiy qismlarga Dataset komponentidan foydalanilgandek ma'lumotlar bazasi komponenti bilan aloqa qilish orqali ushbu ma'lumotlarni boshqarishga imkon beradigan ADO ma'lumotlar to'plami. Uni ADOTabie, ADOQuery, ADOStoredProc komponentlarida ishlatish mumkin;
- ADOTabie - ADO jadvali, ADO ma'lumotlar manbasining bitta jadvaliga kirishni ta'minlaydi va boshqa komponentlarga DataSource komponenti orqali ADOTable komponent bilan aloqa qilish orqali ushbu ma'lumotlarni boshqarishga imkon beradi;
- ADOQuery - ADO ma'lumot manbasidan ma'lumot olish uchun SQL buyruqlarini bajarishga imkon beradigan va boshqa komponentlarga DataSource komponenti orqali ADOTable komponent bilan aloqa qilish orqali ushbu ma'lumotlarni boshqarishga imkon beradigan ADO so'rovi;
- ADOStoredProc - ADO-da saqlanadigan protsedura, ilovalarga ADO interfeysi yordamida saqlangan protseduralarga kirishga imkon beradi;
- RDSConnection - Recordset ob'ektini bir jarayondan (kompyuterdan) boshqasiga o'tkazilishini boshqarish uchun ishlatiladigan RDS aloqasi. Komponent server dasturlarini yaratish uchun ishlatiladi.

ADOconnection komponenti ma'lumotlar va boshqa **ADO** komponentlari o'rtasida vositachi sifatida ishlatilishi mumkin, bu ulanishlarni yanada moslashuvchan boshqarishga imkon beradi. Ma'lumot manbasiga to'g'ridan-to'g'ri ulanish orqali boshqa **ADO** tarkibiy qismlaridan foydalanishingiz mumkin. Buning uchun **ADO** tarkibiy qismlari *ConnectionString* xususiyatiga ega, ular yordamida o'zlarining ma'lumotlarga kirish kanallarini yaratishlari mumkin.

Kirish dasturida bir yoki bir nechta jadvallar mavjud bo'lgan oddiy ma'lumotlar bazasini yarataylik. Buni qanday qilishni bilasiz deb umid qilaman. Agar yo'q bo'lsa, sizga "Microsoft Office Access for Dummies" kitobini o'qishni maslahat beraman, juda yaxshi kitob.

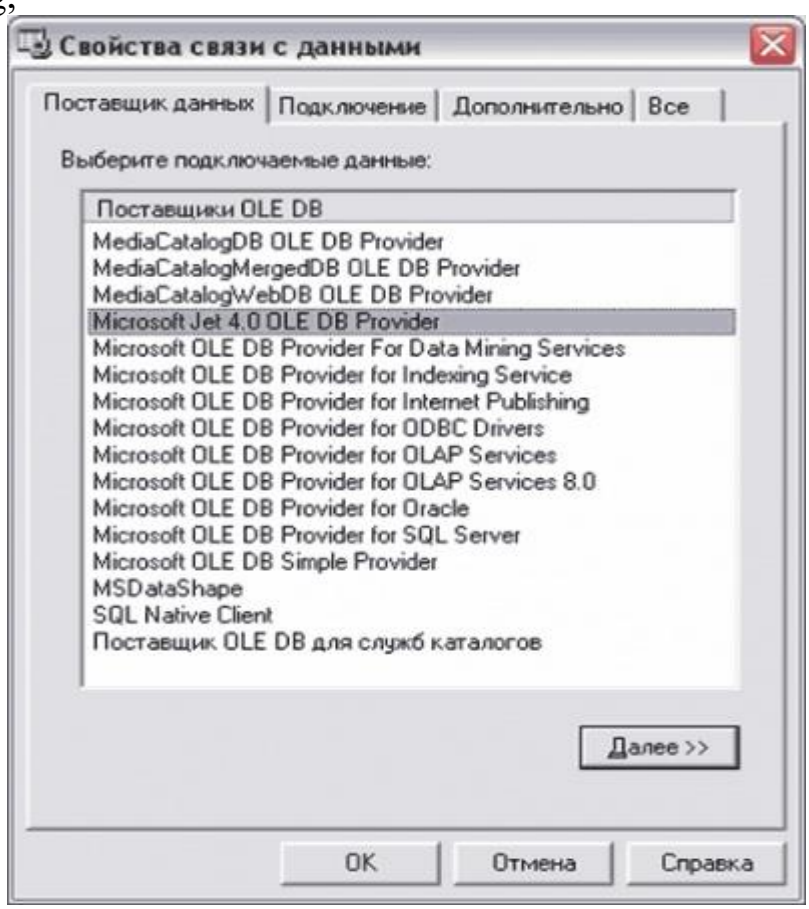
ActiveX Data Object (ADO) bilan ishlash uchun barcha komponentlar xuddi shu nom ostida yorliqda joylashgan. Boshlash uchun biz TADOConnection komponentini formaga joylashtirishimiz kerak - bu bizning dasturimizning jismoniy ma'lumotlar bazasi bilan bog'lanishiga javobgar bo'ladi. Keyinchalik, TDataSource-ni qo'yishni unutmang - bu holda biz ma'lumotlarni grafik ravishda namoyish eta olmaymiz, ma'lumotlarni boshqarish elementlarini TADOTable komponenti bilan bog'lash uchun kerak bo'ladi, uni siz formada ham joylashtirasiz (BDE dan TTable-ga o'xshash ma'lumotlar bazasi jadvali bilan bog'lanish uchun kerak).

15.1-rasm. ADO panel .

Uchala komponentni bir-biri bilan ulash uchun siz quyidagi amallarni

bajarishingiz kerak.

1. TADOTable komponentasi uchun ulanish xususiyatini formada joylashtirilgan TADOConnection komponentiga o'rnating (sukut bo'yicha bu ADOConnection1);
2. TDataSource komponentasining Dataata xususiyatini ADOTable1 ga o'rnating;



15.2-rasm. TADOConnection ulanish parametrlarini sozlash.

Ma'lumotlar bazasiga ulanish vaqti keldi, lekin avval uni sozlashingiz kerak. TADOConnection komponentini tanlang, ustiga ikki marta bosing. Ko'rsatilgan oynada "Use connection string" -ni tanlang va "Build" -ni bosing. Bunga javoban, sizdan ma'lumotlar bazasi bilan ishlash uchun drayverni tanlash so'raladi. Men darhol sizni mamnun qilmoqchiman: standart Windows tarqatish barcha kerakli drayverlarni o'z ichiga oladi, shuning uchun siz BDE-da bo'lgani kabi tarqatish to'plamini o'zingiz bilan birga olib yurishingizga hojat qolmaydi.

Biz Access MDB ma'lumotlar bazasi bilan ishlayotganligimiz sababli, Microsoft Jet 4.0 OLE DB drayverini tanlang va keyingisini bosing. Ma'lumotlar bazangizga boradigan yo'lni ko'rsating va agar siz faylga kirishni boshqarishni o'rnatgan bo'lsangiz, foydalanuvchi nomi va parolni kiriting.

Murakkab yorlig'ida faylga kirish rejimi va tarmoq sozlamalari o'rnatiladi, ammo bizning holatlarimizda bunday imkoniyat mavjud emas.

Endi siz OK tugmachasini bosishingiz mumkin va barcha sozlamalar TADOConnection komponentining String turidagi ConnectionString deb

nomlangan komponentida saqlanadi.

Ulangan xususiyatni (xuddi shu tarkibiy qism) rost qilib, ma'lumotlar bazasiga ulanish mavjudligini tekshirishingiz mumkin.

Agar IDE hech qanday xatoga yo'l qo'ymasa, demak siz hamma narsani to'g'ri sozlagansiz va ishlashni davom ettirishingiz mumkin, agar bo'lmasa, yuqoridagi barcha amallarni takrorlang.

Agar dastur avtomatik ravishda parolni va ulanish paytida foydalanuvchi ma'lumotlar bazasi fayliga kirishni so'rashini istamasangiz, buning uchun TADOConnection.LoginPrompt xususiyati mavjud. Uni “false” ga sozlang.

Endi TADOTable komponentini tanlang va TableName xususiyatida o'zingiz yaratgan ma'lumotlar bazasidan ishlamoqchi bo'lgan jadval nomini ko'rsating (TADOConnection.Connected xususiyatida haqiqiyligini tekshiring). Va nihoyat, komponentga ADOConnection yordamida ma'lumotlar bazasiga kirish uchun Active = true (TADOTable) ni o'rnating.

Aslida, bu mdb fayllari bilan ishlash uchun bizning dasturimizni sozlashni yakunlaydi. Endi grafik interfeys, ma'lumotlarni qayta ishlash va displey bilan shug'ullanamiz.

15.2 TADOTable komponentini dasturlash

Yuqoridagi operatsiyalarni bajarib, bizda ma'lumotlar bazasi bilan to'liq ishlashga imkon beradigan kichik, ammo juda jiddiy dastur mavjud. Biroq, mavjud ma'lumotlarni tahlil qilish biroz qiyin. Dasturga ma'lum belgilarga muvofiq yozuvlarni qidirish va filtrlashni qidirishni kiritish yaxshi bo'lar edi.

Izlashni boshlash uchun. Buning uchun TADOTable komponentasi ko'p funktsiyalarga ega:

- TADOTable.Locate (const AnsiString KeyFields, const System:: Variant & KeyValues, TLocateOptions parametrlari); KeyValues o'zgaruvchisining qiymati uchun KeyFields o'zgaruvchisida ko'rsatilgan kalit maydonida qidiruvlar. Agar moslik topilsa, topilgan yozuv joriy bo'ladi, ya'ni. Kursor unga joylashtirilgan. Variantlar o'zgaruvchisi qidirish uchun qiymatlarni qanday o'zgartirish kerakligini ko'rsatadi: loCaseInsensitive yoki loPartialKey.
- TADOTable.Seek (const Variant & KeyValues, TSeekOption SeekOption = soFirstEQ); Indekslar yordamida qidirishda foydalaniladi. SeekOption agar yozuv topilsa nima qilish kerakligini aniqlaydi: soFirstEQ, soLastEQ, soAfterEQ, soAfter, soBeforeEQ, va oldin. Siz ushbu qo'llanmaning batafsil tavsifini topishingiz mumkin.
- TADOTable.LookUp (const AnsiString KeyFields, const Variant & KeyValues, const AnsiString ResultFields); Natijada so'rovni qondiradigan barcha topilgan qiymatlarning qiymatlari bilan Variant turini qaytaradi.

Jadvalda ma'lumotni topishning yana bir qancha usullari mavjud: barcha elementlarni qo'lda tartiblash, FindFirst, Next va hokazolar yordamida. Ammo biz

faqat asosiy va zaruriy usullarni sanab o'tdik.

Mana, joylashishni aniqlash usulidan foydalangan holda misol. Formaga TComboBox, TButton va TEdit komponentlarini joylashtiring, jadvalingizdagi barcha maydonlar nomlarini ComboBox-ga kiriting (buning uchun "Items" xususiyati ishlatiladi). Edit-da biz qidirish uchun zarur bo'lgan qiymatni kiritamiz va ComboBox-dan qidirish kerak bo'lgan maydonni tanlang.

Endi Tugmani ikki marta bosing yoki ob'ekt nazoratchisida OnClick hodisasini tanlang.

Unga quyidagi kodni kiriting:

```
void __fastcall TForm1 :: Button1Click (TObject *
Sender)
{
ADOTable1-> Joylashtirish (ComboBox1-> Matn, Edit1->
Matn, [loCaseInsensitive]);
// bu erda Combo-da ko'rsatilgan maydon orqali biz Edit
dan qiymatni qidiramiz. Hammasi oddiy
// loCaseInsensitive bu kichik va katta harflarsiz
qidirishdir
}
```

Ko'ryapmizki, oddiy qidiruvni amalga oshirish uchun faqat bitta satr kodi kifoya qiladi.

Bizning loyihamizga filtrlashni qo'shish vaqti keldi. Rasmga boshqa TEdit va TButton qo'shamiz. Buning tamoyilini tushunish uchun etarli bo'ladi. Va ushbu kodni oching:

void __fastcall TForm1 :: Button2Click (TObject * Sender)

```
{
ADOTable1-> Filtrlangan = noto'g'ri; // Agar mavjud
bo'lsa, avvalgi filtrlashni bekor qilish
ADOTable1-> Filtr = Edit2-> Matn; // Filtrlash
mezonlarini o'rnatish
ADOTable1-> Filtrlangan = haqiqiy; // va yana
filtrlashni yoqadi
}
```

Edit2-ga nima kiritilishi mumkinligini tushuntiraman. Biz buyruq satrining juda uzoq analogini yaratdik. Bu erda filtrlash ishlaydigan shablon:

```
<maydon nomi> | parametr (=,>, <,> =, <= va boshqalar)
| <qiymat>
```

Mana bir misol:

```
Ism = Ivanov Ivan Petrovich
```

Natijada, jadvalda faqat "Ivan Petrovich" qiymati bo'lgan jadvalga jismoniy o'zgartirishlar kiritilmagan ma'lumotlar kiritiladi.

Savollar:

1. ADO orqali ma'lumotlar bazasiga ulanish uchun qanday tarkibiy qismlar talab qilinadi ?
2. Ma'lumotlar bazasida ulanishni ADO orqali qanday sozlash mumkin ?
3. TADOTable bilan ishlashda qanday usullar keltirilgan ?
4. TADOTable-da ma'lumotlar qanday filtrlanadi ?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.
2. CJ sana - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.
3. Navbat B.S. Borland C ++ Builder 6. Sankt-Peterburg uchun ma'lumotlar bazasi uchun ilovalarni ishlab chiqish: DiaSoftUP LLC, 2003 - 320 p.

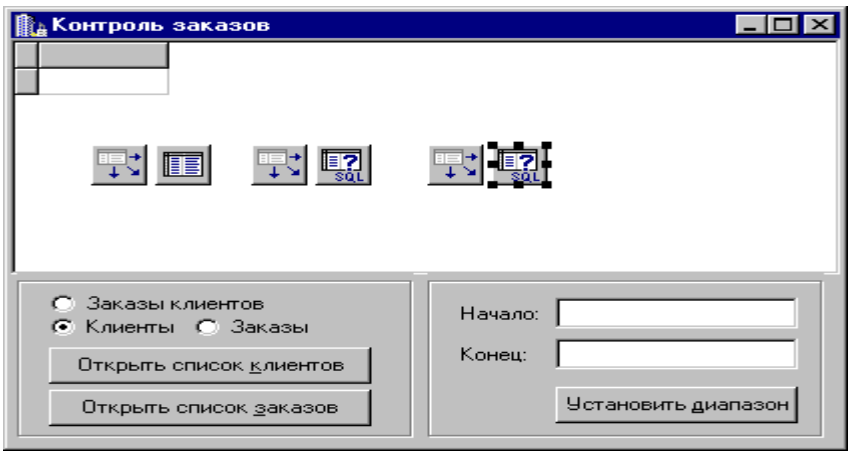
XVI Bob. ADO va C++ orqali maydon qiymatlarini kiritish, yozuvlarni o’zgartirish, qo’shish va o’zgartirish

16.1 Ma'lumotni boshqarish tili (DML).

Keling, olingan ma'lumotlarni TTable, TQuery, TDataSource, TDBGrid komponentlari yordamida dastur yaratishda qo'llashga harakat qilaylik. Buning uchun biz Borland C ++ Builder paketiga kiritilgan BCDEMOS ma'lumotlar bazasida mavjud bo'lgan Customer.db va Orders.db jadvallaridan foydalanamiz. Ushbu ilova mijozlar jadvalidagi mijozlar ro'yxatini, Buyurtmalar jadvalidagi buyurtmalarni ro'yxatlashi kerak, shuningdek mijozlar sonini tanlashga imkon beradi.

Yangi loyiha yarating va uning asosiy rasmini CUST1.CPP, va loyihani CUST.MAK sifatida saqlang.

Forma sarlavhasining nomini "Buyurtma nazorati" ga o'zgartiring. Formaga TDBGrid komponentini, ikkita TGroupBox komponentlarini, bitta TTable komponentasini, ikkita TQuery komponentlarini, uchta TDataSource komponentlarini joylashtiramiz. GroupBox1 komponentasida uchta TRadioButton va ikkita TButton komponentlarini joylashtiring. Biz ikkita TEdit, ikkita TEdit va bitta TButton komponentlarini GroupBox2 komponentiga joylashtirdik.



16.1-rasm. Qo'shimcha rasmidagi komponentlarning joylashuvi.

Ushbu komponentlar uchun quyidagi xususiyatlarni o'rnating:

Komponent nomi	Mulk	Qiymati
Table1	DatabaseName	BCDEMOS
	TableName	CUSTOMER.DB
	Active	false
DataSource1	DataSet	Table1
DBGrid1	DataSource	DataSource1
Query1	Database Name	BCDEMOS
	SQL	select * from orders
	Active	false
DataSource2	DataSet	Query1
Query2	DatabaseName	BCDEMOS
DataSource3	DataSet	Query2
Button1	Caption	Ochiq ro'yxat va mijozlar
Button2	Caption	Ochiq ro'yxat va buyurtmalar
RadioButton1	Caption	Xaridorlarga
	Checked	rost
RadioButton2	Caption	Buyurtmalar
GroupBox1	Caption	
GroupBox2	Caption	
Button3	Caption	Sozlash
Edit1	Text	
Edit2	Text	
Label1	Caption	Boshlanishi:
Label2	Caption	Tugashi:
RadioButton3	Caption	Xaridor buyurtmalari

Button1 uchun OnClick tadbir ishlovchisini yarating:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
     {
     if (Table1->Active)
```

```
{
Table1->Close();
Button1->Caption = "Ioe?uou nienie &eeeaioia";
}
else
{
Table1->Open();
Button1->Caption= "Cae?uou nienie &eeeaioia";
}
}
```

Endi ushbu tugmachani bosganingizda Mijozlar jadvali ochiladi va yopiladi va tugmadagi yozuv o'zgaradi.

Button2 uchun OnClick tadbir ishlovchisini yarating:

```
            void __fastcall
TForm1::Button2Click(TObject *Sender)
{
if (Query1->Active)
{
Query1->Active = false;
Button2->Caption = "Ioe?uou ro’yxat &caeacia";
}
else
{
Query1->Active = true;
Button2->Caption = "3ae?uou ro’yxat &caeacia";
}
}
```

Button2 tugmachasini bosganingizda, u buyurtmalar ro'yxatini o'z ichiga olgan Query1 so'rovini ochadi yoki yopadi:

RadioButton1 va RadioButton2 radio tugmachalari uchun OnClick tadbir boshqaruvchilarini yaratamiz:

```
            void __fastcall
TForm1::RadioButton1Click(TObject *Sender)
{
DBGrid1->DataSource = DataSource1;
}
//---------------------------------------------------
----------------------
void __fastcall TForm1::RadioButton2Click(TObject
*Sender)
{
DBGrid1->DataSource =DataSource2;
}
```

Endi ushbu tugmalar yordamida mijozlar ro'yxati va buyurtmalar ro'yxati o'rtasida almashish mumkin.

Button3 uchun OnClick tadbir ishlovchisini yaratamiz:

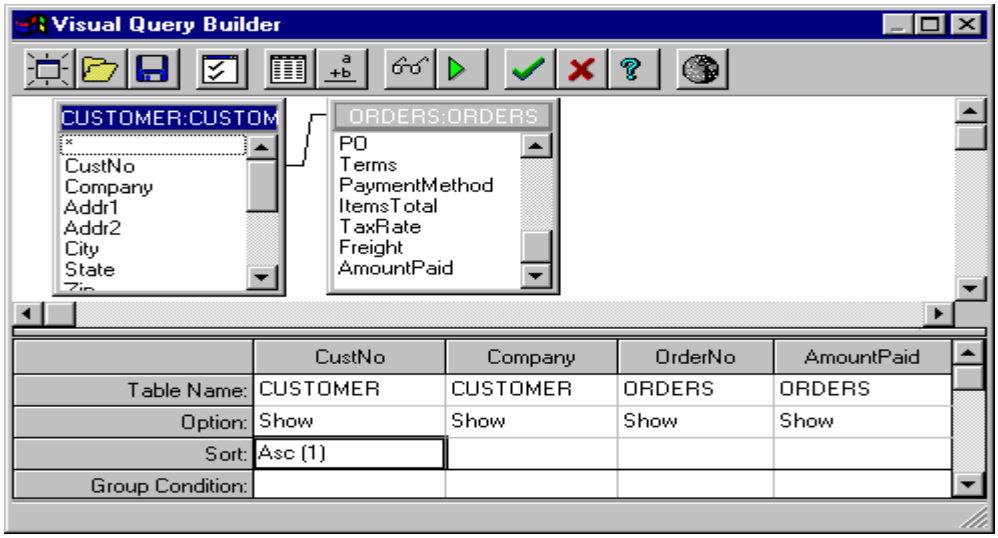
```
            void __fastcall
TForm1::Button3Click(TObject *Sender)
     {
     if (Table1->Active)
     {
     Table1->SetRangeStart();
     Table1->Fields[0]->AsString = Edit1->Text;
     Table1->SetRangeEnd();
     Table1->Fields[0]->AsString = Edit2->Text;
     Table1->ApplyRange();
     }
     }
```

Endi Edit1 va Edit2 va Button3 tahrirlash maydonlaridan foydalanib, DBGrid1-da ular to'g'risidagi ma'lumotlarni aks ettirish uchun mijoz raqamlarini tanlashingiz mumkin.

Keyin, Visual Query Builder-dan foydalanib, Query2 komponentining SQL xususiyatini o'rnating. BCDEMOS- ni ma'lumotlar bazasi nomi sifatida tanlaymiz va so'rovga CUSTOMER va ORDERS jadvallarini qo'shamiz. Keyin biz buyurtmalar jadvalidagi CUSTOMER jadvalidagi CustNo satridan CustNo maydonchasiga chiziq qo'yib jadvallar o'rtasidagi aloqani o'rnatamiz.

Biz so'rovga quyidagi maydonchalarni qo'shamiz:

- Customer.CustNo
- Customer.Company
- Orders.OrderNo
- Orders.AmountPaid



16.2-rasm. Birlashtirilgan so'rovni yaratish uchun Visual Query Builder-dan foydalanish .

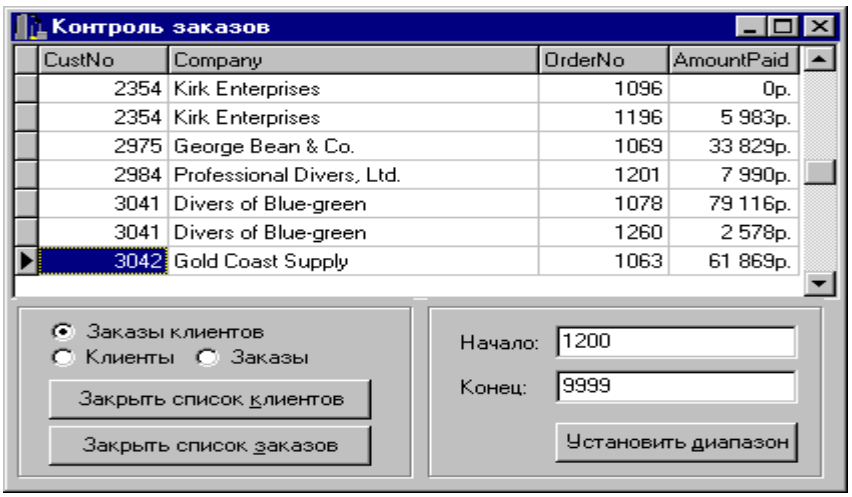
Endi so'rov natijalarini mijozlar soni bo'yicha tartiblang va Visual Query Builder-dan chiqing.

Ob'ekt inspektoridan foydalanib, Query2 komponentini tanlang va uning aktiv xususiyatini rostga o'rnatamiz.

RadioButton3 uchun OnClick tadbir ishlov beruvchisini yaratamiz:

```
            void __fastcall
TForm1::RadioButton3Click(TObject *Sender)
    {
    DBGrid1->DataSource= DataSource3;
    }
```

Ilovani tuzing. Ikkala ma'lumotlar to'plamini ochish uchun buyurtmalar ro'yxatini bosing va ochamiz. Tugmalaridan foydalanishga harakat qilib ko'ramiz.



16.3-rasm. Tugallangan ilova quyidagicha ko'rinadi .

"Mijozlar ro'yxatini ochish" tugmasini bosamiz. "Ishga tushirish" va "Tugatish" maydonlarining qiymatlarini kiriting (masalan, mos ravishda 1200 va 1700) va keyin "Diapazonni o'rnatish" tugmasini bosamiz. Buyurtma raqamlarining qiymatlari haqiqatan ham ushbu diapazonda ekanligiga ishonch hosil qilamiz.

Biz "Sotish buyurtmalari" tugmachasini bosamiz va natijada olingan ma'lumotlar to'plamida ikkala jadvaldan ham ma'lumotlar borligiga ishonch hosil qilamiz.

Savollar:

1. TADOQuery yordamida dastur yaratish uchun qanday qadamlar kerak ?
2. Komponentlarda qanday xususiyatlarni sozlash kerak?
3. Formadan kodga qanday ma'lumot uzatiladi?
4. Ulanish so'rovini qanday bajarish kerak?

Adabiyotlar:

1. Tomas Konnoli, Kerolin Begg - ma'lumotlar bazalari tizimlari. Loyihalash, amalga oshirish va boshqarish uchun amaliy yondashuv. 4-nashr - Addison Uesli 2005 - 1373p.

2. CJ sana - Ma'lumotlar bazasi tizimlariga kirish - Addison-Wesley Professional - 2003 - 1024 p.
3. B.S. Borland C ++ Builder 6. Sankt-Peterburg uchun ma'lumotlar bazasi uchun ilovalarni ishlab chiqish: DiaSoftUP LLC, 2003 - 320 p.

XVII Bob. XML va ma'lumotlar bazasi

17.1 Zaif tuzilgan ma'lumotlar

Zaif tuzilgan ma'lumotlar ma'lum tuzilishga ega deb ta'riflanadi, ammo bu struktura nomuvofiq, etarlicha o'rganilmagan yoki to'liq bo'lmagan bo'lishi mumkin. Odatda, bunday ma'lumotlarni biron bir o'zgarmas sxemadan foydalanib tasvirlab bo'lmaydi, shuning uchun ular ba'zan sxemasi bo'lmagan (sxemasi yo'q) yoki o'zlarini tavsiflovchi (o'z-o'zini tasvirlaydigan) deb nomlanadi. Yomon tuzilgan ma'lumotlarning o'ziga xos xususiyati shundaki, odatda alohida sxemada joylashtirilgan tavsiflovchi ma'lumotlar ma'lumotlarning o'zida mavjud. Yomon tuzilgan ma'lumotlarni taqdim etishning ba'zi rasmlari alohida sxemadan foydalanishni ta'minlamaydi, boshqalarida esa mavjud, ammo unda berilgan ma'lumotlarga juda cheklovlar qo'yiladi. Bundan farqli o'laroq, relatsiya ma'lumotlar bazasi ma'lumotlar bazasida jadvallarni tarqatishga imkon beradigan oldindan belgilangan sxemani talab qiladi va ushbu tizim tomonidan boshqariladigan barcha ma'lumotlar bunday tuzilishga mos kelishi kerak. Ob'ektga yo'naltirilgan MBBT relyatsion ma'lumotlar bazasi bilan taqqoslaganda yanada rivojlangan tuzilmani yaratishga imkon beradi, shu bilan birga barcha ma'lumotlarning oldindan belgilangan (ob'ektga yo'naltirilgan) sxemaga mos kelishini talab qiladi. Ammo agar yomon tuzilgan ma'lumotlar MBBTda saqlanishi kerak bo'lsa, u ushbu ma'lumotlarga asoslanib sxemani tuzishi kerak va ma'lumotlarga priori belgilangan sxemani yuklamasligi kerak.

So'nggi paytlarda ko'plab sabablarga ko'ra yomon tuzilgan ma'lumotlarga katta qiziqish paydo bo'ldi. Eng muhimlari quyida keltirilgan.

- Axborotni qayta ishlashni yanada avtomatlashtirish uchun Internetda taqdim etilgan ma'lumotlar manbalariga ma'lumotlar bazasi sifatida kirish imkoniyatiga ega bo'lish kerak, ammo ushbu manbalarga biron bir oldindan belgilangan sxemani yuklab bo'lmaydi.
- Turli xil ma'lumotlar bazalari soni doimiy ravishda o'sib bormoqda, shuning uchun har xil turdagi ma'lumotlar bazalari o'rtasida ma'lumot almashish uchun moslashuvchan format yaratish vazifasi tobora muhim ahamiyat kasb etmoqda.
- Internetda ma'lumotlarni taqdim etish va almashish uchun standart sifatida XML tilining (kengaytirilgan markup tili - belgilash tili) rasmlanishi munosabati bilan zaif tuzilgan ma'lumotlarni qayta ishlash uchun yangi usullarni yaratish istiqbollari paydo bo'ldi, chunki XML tili ularni tavsiflash uchun juda mos keladi.

Sifatsiz tuzilgan ma'lumotni boshqarishning aksariyat usullari bunday

ma'lumotlarni namoyish etish uchun xizmat qiladigan daraxt bilan belgilangan grafikadan o'tishni ta'minlaydigan so'rovlar tilidan foydalanishga asoslangan. Agar ma'lumotni sxema yordamida tavsiflab bo'lmasa, unda ularni aniqlashning yagona usuli bu uning tarkibiy xususiyatlarini rasmiylashtirish emas, balki to'plamdagi ma'lumotlar elementining o'rnini ko'rsatishdir. Bu shuni anglatadiki, ma'lumotlar so'rovlarini bajarish usullari an'anaviy deklarativ xususiyatini yo'qotadi va yanada navigatsion bo'ladi. Biz yomon tuzilgan ma'lumotlar bilan tanishishni, zaif tuzilgan tizimni qayta ishlash uchun qanday ma'lumotlardan foydalanish mumkinligini ko'rsatadigan misolni ko'rib chiqishdan boshlaymiz.

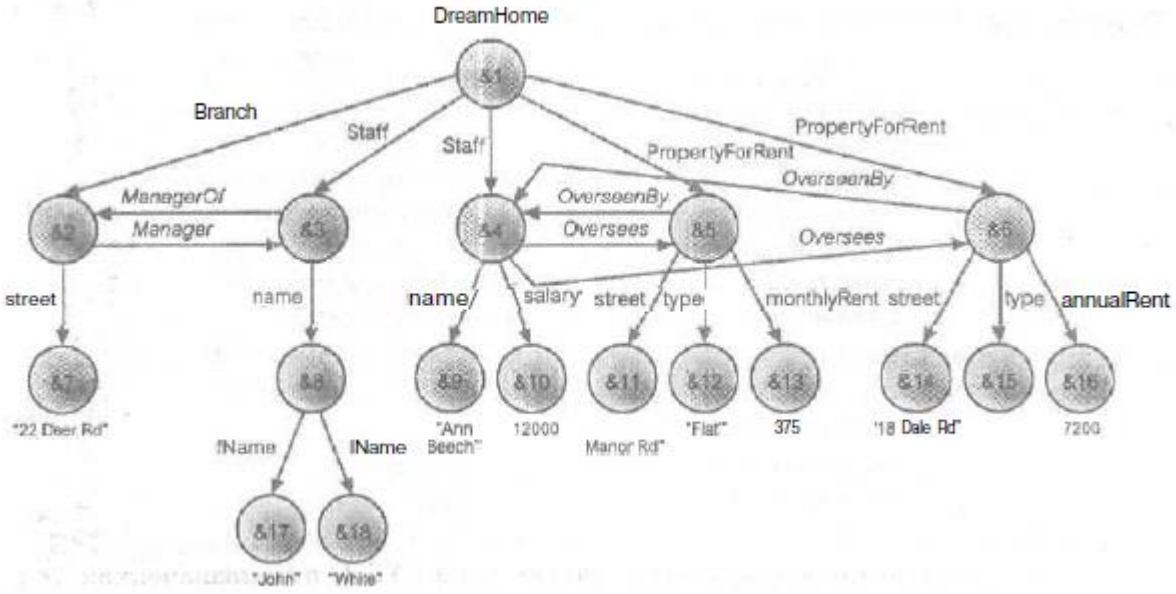
1- Listingda ko'rsatilgan strukturani ko'rib chiqamiz, unda DreamHome o'quv loyihasida keltirilgan ba'zi ma'lumotlar ko'rsatilgan. Ushbu ma'lumotlar 1- rasmda ko'rsatilganidek, grafikada ko'rsatilishi mumkin. Ular kompaniyaning bitta filialini (Deer Rd 22-da joylashgan), kompaniyaning ikki xodimi (Jon Uayt va Ann Bix) va ikkita ijaraga berilgan mulk (2 Manor Rd va 18 Deylda joylashgan) Rd); Ushbu rasm ma'lumotlar orasidagi ba'zi bir munosabatlarni ham ko'rsatadi. Xususan, shuni ta'kidlash kerakki, ma'lumotlar etarli darajada rasmiylashtirilmagan:

- Jon Uaytning xodimi uchun ism va familiya alohida ko'rsatiladi va Ann Bechning xodimi uchun ism va familiya bitta komponent sifatida saqlanadi, shuningdek ish haqi to'g'risidagi ma'lumotlar;
- 2 Manor r-nda joylashgan mulk uchun oylik ijara, 18-Deyl r-nda joylashgan mulk uchun esa yillik ijara saqlanadi;
- 2 Manor Rd-da joylashgan mulk uchun turning belgisi (kvartira) ip rasmida saqlanadi va 16 Deyl Rdda joylashgan mulk uchun belgi (uy) butun son sifatida berilgan.

```
DreamHome (&1)
Branch (&2)
     street (&7) "22 Deer Rd"
     Manager &3
Staff (&3)
     name (&8)
          fName (&17) "John"
          IName (&8) "White
     ManagerOf &2
Staff (&4)
     name (&9) "Ann Beech"
     salary (&10) 12000
     Oversees &5
     Oversees &6
PropertyForRent (&5)
     street (&11) "2 Manor Rd"
     type (&2) "Flat"
     monthlyRent (&3) 375
     OverseenBy &4
```

```
PropertyForRent (&6)
     Street (&14) "16 Dale Rd"
     type (&15) 1
     annualRent (&6) 7200
     OverseenBy &4
```

1 - Listing. DreamHome loyihasi ma'lumotlar bazasida notekis tuzilgan ma'lumotlarning taqdimoti.



17.1-rasm. Listing 1-da ma'lumotlarning grafik ko'rinishi .

1. Ma'lumotlar bazasining ishlashini ta'minlash.
2. Ob'ektlarning o'ziga xosligini saqlash.
3. Inkapsulatsiyani ta'minlash.
4. Ob'ektlarni murakkab holatda saqlash

17.2 Ob'ekt ma'lumotlarini almashish modeli (OEM)

Yomon tuzilgan ma'lumotlarni birinchi bo'lib tavsiflaganlardan biri ***Obyektlar almashish modeli (OEM)dir***. Bu dastlab TSIMMIS loyihasi uchun ishlab chiqarilgan (Stanford-IBM ko'p ma'lumot manbalarining menejeri) va turli xil manbalardan olingan ma'lumotlarning integratsiyasini ta'minlashi kerak bo'lgan ichki ob'ektlar uchun vakillik modelidir. OEM modellarida sxema yo'q va o'zlarini tavsiflaydi. Bunday modelni tugunlari ob'ektga ega bo'lgan yorliqli yo'naltirilgan grafik sifatida ko'rib chiqish mumkin (17.1-rasmda ko'rsatilgan).

Har bir OEM ob'ekti uchun noyob ob'ekt identifikatori (masalan, & 7), tavsiflovchi matn yorlig'i (ko'cha), turi (satr) va qiymati ("22 Deer Rd") o'rnatiladi. Ob'ektlar elementar va kompozitga bo'linadi. Elementar ob'ektlar bazaviy turdagi qiymatlarni o'z ichiga oladi (masalan, butun son yoki satr). Ular grafikada chiquvchi qirralarsiz tasvirlanganligi bilan ajralib turadi. Boshqa barcha ob'ektlar kompozit deb nomlanadi. Ushbu ob'ektlarning turi ob'ekt identifikatorlari to'plami sifatida belgilanadi va grafikada ular bir yoki bir nechta chiquvchi qirralarga ega tugunlar sifatida tasvirlangan. OEM kompozit ob'ekti o'zining OEM ob'ektlari uchun ota-ona hisoblanadi. Har bir alohida OEM ob'ekti

o'zboshimchalik bilan sonli ona ob'ektlariga ega bo'lishi mumkin. Bu ma'lumotlar orasidagi barcha kerakli ulanishlarni simulyatsiya qilish uchun har qanday murakkablikdagi tarmoqlarni yaratishga imkon beradi.

Ob'ektni belgilash uchun yorliq ishlatiladi va ob'ekt nima ekanligini ko'rsatadi (shuning uchun OEM modelida keltirilgan ma'lumotlar o'z-o'zini tavsiflash deb nomlanadi); Bu shuni anglatadiki, yorliq matni imkon qadar ko'proq tavsiflovchi ma'lumotlarni o'z ichiga olishi kerak. Teglarni dinamik ravishda o'zgartirish mumkin. Ism - bu ob'ektlardan biri uchun taxallus sifatida ishlatiladigan va ma'lumotlar bazasiga kirish nuqtasi sifatida xizmat qiladigan maxsus yorliq (masalan, DreamHome bu ob'ektni ifodalovchi nom va 1).

Har qanday OEM ob'ekti to'rtta {yorliq, oldingi, turi, qiymati} sifatida ko'rib chiqilishi mumkin. Masalan, quyidagicha xodimlar ob'ekti (tugun va 4) nomi va ish haqi ob'ektlari, "Ann Beech" satrini o'z ichiga olgan nom ob'ekti (tugun va 9) va 12000 kasr qiymatini o'z ichiga olgan ish haqi ob'ekti (tugun va 10) qanday ko'rsatilishi mumkinligi ko'rsatilgan. .

```
{ Staff, &4, to'plam, {&9, &10}}
{ name & 9, qator, "Ann Beech"}
{ salary, & 10, o'nlik, 12000}
```

OEM modeli to'liq bo'lmagan va rasmiylashtirilmagan ma'lumotlarni qayta ishlash uchun maxsus yaratilgan va yuqoridagi misolda model qanday qilib tartibsiz struktura va noaniq turga ega ma'lumotlarni namoyish qilishini ko'rish mumkin.

17.3 XML haqida tushuncha

Hozirgi vaqtda Internetda saqlanadigan va uzatiladigan hujjatlarning ko'pi HTML kodda taqdim etilgan. Yuqorida ta'kidlab o'tilganidek, HTML-ning afzalliklaridan biri uning soddaligi bo'lib, bu tildan foydalanuvchilarning turli toifalari uchun foydalanish imkonini beradi. Ammo, shuningdek, bu tilning soddaligi nafaqat uning afzalligi, balki noqulayligi haqida ham asosli ravishda bahslashish mumkin. Xususan, HTML tili deskriptorlar to'plamini kengaytirishga imkon bermaydi va bunga ehtiyoj ko'proq paydo bo'ladi, chunki yangi deskriptorlar ba'zi muammolarni hal qilishni soddalashtirish va HTML hujjatlarini yanada jozibali va dinamik qilishlari mumkin edi. Ushbu ehtiyojni qondirishga harakat qilib, brauzer ishlab chiquvchilari ma'lum brauzerlarga xos bo'lgan ba'zi HTML tavsiflovchilarini taqdim etishga harakat qilmoqdalar. Ammo bunday urinishlar ishlab chiqilgan, omma uchun ochiq veb-hujjatlarni yaratish yanada murakkablashishi va HTML-ning turli xil talqinlari paydo bo'lishiga olib keladi. HTML-ni dialektlarga bo'lish tendentsiyasining rivojlanishiga yo'l qo'ymaslik uchun W3C yangi standart - ***XML (kengaytirilgan markup tili - kengaytirilgan markalash tili) ni taklif qiladi***. Ushbu til ma'lumotlardan ilovalarning bir xil mustaqilligini ta'minlashga imkon beradi, shu sababli HTML bir vaqtning o'zida taqdimot vositalarining ma'lumotlardan mustaqilligini ta'minlaydigan portativ va kuchli tilga aylandi.

XML Dizaynerlarga funksionallikni amalga oshirish uchun ixtisoslashtirilgan deskriptorlarni yaratishga imkon beradigan metallangage (boshqa tillarni tasvirlash

uchun til); HTML-dan foydalanib bo'lmayapti.

XML - bu ***SGML (standart umumlashtirilgan markalash tili)*** to'plami; Web-hujjatlar uchun maxsus mo'ljallangan. XML bu meta-til bo'lib, u dizaynerlarga HTML-ga erishib bo'lmaydigan funktsiyalarni amalga oshirish uchun o'zlarining maxsus deskriptorlarini yaratishga imkon beradi. Masalan, XML bir vaqtning o'zida bir nechta hujjatlarga ishora qiladigan havolalarni qo'llash imkoniyatini beradi, HTML havolasi esa faqat bitta maqsadli hujjatga ishora qiladi.

SGML - bu turdagi hujjatlarning namunalarini ifodalovchi hujjatlarning tuzilgan turlarini va belgilash tillarini aniqlash tizimi. O'n yildan ko'proq vaqt davomida SGML standartlashtirilgan, dasturiy ta'minotsiz, tizimlashtirilgan hujjatlar uchun omborxonalarni yaratishda ishlatilgan. Ushbu til har qanday hujjatni ikkita mantiqiy mustaqil qismga bo'lish imkonini beradi; ulardan biri hujjatning tuzilishini belgilaydi, ikkinchisi matnning o'zi. Tuzilish ta'rifi Hujjat turini aniqlash (DTD) deb nomlanadi. SGML tili har qanday hujjatga ma'lum bir tuzilmani belgilashga imkon beradi va hujjat mualliflariga o'zlarining ixtisoslashgan tuzilmalarini yaratishga imkon beradi, shuning uchun bu juda kuchli hujjat aylanish tizimining asosi bo'lishi mumkin. Ammo bu til o'zining murakkabligi bilan ajralib turadi, shuning uchun u keng tarqalmagan.

XML tili shunga o'xshash funktsiyalarni bajarish uchun yaratilgan, ammo unchalik murakkab emas va shu bilan birga tarmoqdan foydalanish uchun ko'proq mos keladi. XML tili SGML-ning asosiy afzalliklarini saqlab qolganligi juda muhimdir: kengayish, tuzilish va hujjatlarning to'g'riligini tekshirish qobiliyati. XML SGML-ning quyi qismi bo'lgani uchun, SGML-ga to'liq mos keladigan har qanday tizim XML hujjatlarini qayta ishlash qobiliyatiga ega (ammo bu to'g'ri emas). Biroq, XML SGML-ni almashtirishga mo'ljallanmagan. Bundan tashqari, u SGML-ga asoslangan va ushbu tilning qo'llanilishi bo'lgan HTML tilining o'rnini bosa olmaydi. Aslida, XML HTML tiliga qo'shimcha sifatida foydalanish uchun mo'ljallangan va Internet orqali turli xil ma'lumotlarni uzatishga imkon berishi kerak. Keng imkoniyatlari tufayli XML tili aslida nafaqat matnni belgilash uchun, balki ovoz yoki tasvirni belgilash uchun ham ishlatilishi mumkin, ya'ni. multimedia ma'lumotlari. Keng tarqalgan ishlatiladigan XML asosidagi tillarga misollar orasida ***MathML (Matematik belgilash tili)*** , ***SMIL (Sinxronlashtirilgan multimedia integratsiya tili)*** va ***CML (Chemistry Markup Language) mavjud.***

XML tilidan keng foydalanish deyarli besh yil oldin boshlanganiga qaramay (XML spetsifikatsiyasi 1.0 rasmiy ravishda 1998 yil oxirida W3C tomonidan tasdiqlanganligi sababli), bu til allaqachon axborot texnologiyalarining ko'plab sohalariga, shu jumladan grafik interfeyslarga, o'rnatilgan tizimlarga sezilarli ta'sir ko'rsatdi, taqsimlangan tizimlar va ma'lumotlar bazasini boshqarish. Masalan, XML tili ma'lumotlar tuzilishini vizual ravishda tasvirlashga imkon beradi, shuning uchun bu geterogen ma'lumotlar bazalari va ma'lumotlar manbalarining tuzilishini aniqlash uchun qulay mexanizm bo'lishi mumkin. Va XML sizga butun ma'lumotlar bazasi sxemasini aniqlashga imkon berganligi sababli, asosan, tarkibni olish uchun ishlatilishi mumkin, masalan, butun Oracle DBMS sxemasi va uni Informix yoki

Sybase sxemasiga o'tkazish.

XML aslida dasturiy ta'minot sanoatida ma'lumot almashishning standart vositasiga aylandi va bir vaqtning o'zida korxonalar o'rtasida ma'lumot almashishning asosiy mexanizmi bo'lib xizmat qilgan ***EDI (Elektron ma'lumot almashish) tizimlarini*** tezda almashtirmoqda. Ba'zi bir tahlilchilar vaqt o'tishi bilan hujjatlarning aksariyati Internetda va undan tashqarida XML tilida yaratilishini va saqlanishini taxmin qilishmoqda. Ushbu bo'limda XML batafsil tasvirlangan va XML uchun sxemalarni qanday aniqlash kerakligi ko'rsatilgan. Keyingi bo'limda XML uchun so'rov tillari tavsiflanadi, avval XML-ning afzalliklarini ko'rib chiqamiz.

XML afzalliklari

Web muhitida XML-dan foydalanishning ba'zi afzalliklari:

- Oddiylik. XML nisbatan sodda til bo'lib, uning standart qiymati 50 sahifadan oshmaydi. U inson idrokini ta'minlaydigan va juda tushunarli bo'lgan matndan foydalanishga asoslangan til sifatida foydalanish uchun mo'ljallangan.
- Ochiq standart; platforma va dasturiy ta'minot mustaqilligi. XML platforma va dasturiy ta'minotdan mustaqil. Bu ISO standartida belgilangan SGML tilining quyi qismidir. XML tili standartga (ISO 10646) asoslangan bo'lib, Unicode belgilar to'plamini qo'llab-quvvatlaydi va shuning uchun uni barcha alifboda matnni ifodalash uchun foydalanish mumkin, xususan, tegishli milliy tilni va kodlashni belgilash imkoniyatini beradi.
- Kengayish qobiliyati. HTML dan farqli o'laroq, XML kengaytiriladi; bu foydalanuvchilarga ma'lum bir dasturga qo'yiladigan talablarga muvofiq o'zlarining tavsiflovchilarini aniqlashga imkon beradi .
- Qayta foydalanish mumkin. XML kengayishi, shuningdek, XML tavsiflovchi kutubxonalarini yaratishga va ularni ko'plab dasturlarda qayta ishlatishga imkon beradi.
- Tarkibni va taqdimot vositalarini ajratish. XML tili hujjat tarkibini saqlashga va uni qanday taqdim etishni mustaqil tasvirlashga imkon beradi (masalan, brauzerda). Ushbu xususiyat ixtisoslashtirilgan ma'lumotlarni namoyish qilish vositalarini yaratishni soddalashtiradi. Bunday holda, ma'lumot brauzer yordamida foydalanuvchiga etkazilishi mumkin va uni tashqi qurilmaga etkazish uchun, o'sha joyda tanlangan usul, ehtimol foydalanuvchi parametrlari yoki konfiguratsiyasi kabi omillarni hisobga olgan holda qo'llanilishi mumkin. Shunday qilib, xuddi shu tamoyil Java tilida bo'lgani kabi amalga oshiriladi, bu ba'zan sizga "dasturni bir marta ishlab chiqish va uni istalgan joyda qo'llash" imkonini beruvchi til sifatida tavsiflanadi. Shu sababli, XML deb bir xil XML hujjati turli formatlarda va ommaviy axborot vositalaridan foydalangan holda turli xil usullarda nashr etilishi mumkin bo'lgan uslublar jadvallari kabi vositalardan foydalanilganligi sababli "hujjatni bir marta tayyorlash va uni istalgan joyda nashr qilish" imkoniyatini beradigan til deyiladi.

- Yaxshilangan yuk almashish. XML tili ba'zi saytdagi hisob-kitoblar uchun ma'lumotlarni brauzerga (mijoz kompyuterida) etkazib berishga imkon beradi; Shu bilan birga, server hisoblash yuklanishidan qisman ozod qilinadi va yuklarning yanada teng taqsimlanishi ta'minlanadi.
- Bir nechta manbalardan ma'lumotlarni kiritish uchun qo'llab-quvvatlash. Turli xil manbalardan ma'lumotlarni kiritish vazifasi juda murakkab va vaqt talab etadi. Ammo XML tili turli manbalardan ma'lumotlarni birlashtirishni ancha osonlashtiradi. Server ma'lumotlari bazasi va boshqa dasturlardan keladigan ma'lumotlardan birgalikda foydalanish uchun maxsus dasturiy ta'minot agentlaridan foydalanish mumkin, ular keyinchalik ma'lumotlarni boshqa ishlov berish yoki taqdim etish uchun boshqa mijozlarga yoki serverlarga uzatadilar.
- Turli xil dasturlarga taalluqli ma'lumotlarni tavsiflash qobiliyati. XML kengaytiriladiganligi sababli undan turli xil dasturlarda mavjud bo'lgan ma'lumotlarni tavsiflash uchun ham foydalanish mumkin. Bunga qo'shimcha ravishda, XML tili ma'lumotlarni taqdim etish formatini yaratishga imkon beradi, chunki unda ma'lumotlarni o'zi tavsiflaydi, keyinchalik ma'lumotlarni uzatish va qayta ishlash hech qanday qo'shimcha ma'lumotlarning tavsifisiz amalga oshirilishi mumkin.
- Kengaytirilgan qidiruv tizimlari. Hozirgi vaqtda qidiruv tizimlari HTML meta-deskriptorlaridagi ma'lumotlardan foydalanadilar yoki kalit so'zlarning relatsiya holatini tahlil qiladilar. Va XML-dan foydalanganda, qidirish mexanizmlaridan oddiygina tavsiflovchilarni tavsiflash uchun foydalanish mumkin.
- Yangi xususiyatlar. Ko'rinishidan, XML-ning eng katta afzalliklaridan biri bu yangi texnologiyalarni joriy qilish bilan ochiladigan cheksiz imkoniyatlardir.

XML tili haqida qisqacha ma'lumot.

Ushbu bo'limda 1- Listingda ko'rsatilgan oddiy misoldan foydalangan holda XML tiliga qisqacha sharh berilgan bo'lib, unda kompaniya xodimlari to'g'risida ma'lumotlar mavjud.

```
<?xml version= "1.0" encoding= "UTF-8" standalone=
"yes"?>
<?xml:stylesheet type = "text/xsl" href =
"staff_list.xsl"?>
<!DOCTYPE STAFFLIST SYSTEM "staff_iist.dtd">
<STAFFLIST>
    <STAFF branchNo = "B005">
        <STAFFNO>SL2K/STAFFNO>
            <NAME>
                <FNAME>John</FNAME><LNAME>White</LNAME
            >
            </NAME>
```

```
        <POSITION>Manager</POSITION>
        <DOB>l-Oct-45</DOB>
        <SALARY>3OOOO</SALARY>
    </STAFF>
    <STAFF branchNo = "B003">
        <STAFFNO>SG37</STAFFNO>
            <NAME>
                <FNAME>Ann</FMAMExLNAME>Beech</LNAME>
            </NAME>
        <POSITION>Assistant</POSITION>
        <SALARY>12000</SALARY>
    </STAFF>
</STAFFLIST>
```

2- Listing. Kompaniya ishchilari to'g'risida ma'lumot beradigan XML hujjatiga misol.

XML deklaratsiyasi.

XML hujjatlari ixtiyoriy XML deklaratsiyasidan boshlanadi, bu misolda hujjat muallifi (1.0) va kodlash tizimi (UTF-8 Unicode standartiga mos keladi) tomonidan ishlatiladigan XML versiyasini o'z ichiga oladi, shuningdek hujjatning tashqi havolasi mavjudligi to'g'risida ma'lumot mavjud. tuzatish deklaratsiyalari (mustaqil = 'ha' hujjatda tashqi tuzatish deklaratsiyalari yo'qligini bildiradi). Listing 2-da ko'rsatilgan XML hujjatining ikkinchi va uchinchi satrlari uslublar jadvallari va DTD ta'riflari uchun quyida qisqacha muhokama qilinadi.

Elementlar

Belgilanuvchilarning eng ko'p ishlatiladigan rasmi XML elementlari (deskriptorlar deb ham ataladi). Hujjatning birinchi elementi boshqa elementlar (subelementlar) bo'lishi mumkin bo'lgan ildiz elementi deb nomlanishi kerak. Har bir XML hujjatida bitta ildiz elementi bo'lishi kerak, ushbu misolda <STAFFLIST>. Har qanday element boshlang'ich deskriptor bilan boshlanadi (masalan, <STAFF>) va yakuniy deskriptor bilan tugaydi (masalan, </STAFF>). XML elementlari katta-kichikligini hisobga oladi, shuning uchun <STAFF> elementi <staff> elementidan farq qiladi (shuni ta'kidlash kerakki, HTMLda bu shart bajarilmaydi). Element bo'sh bo'lishi mumkin, bu holda uni bitta tavsiflovchi sifatida qisqartirish mumkin, masalan, <EMPTYELEMENT />. Elementlar to'g'ri joylashtirilgan bo'lishi kerak, chunki list2 ning quyidagi parchasi:

```
<STAFF>
    <NAME>
        <FNAME> Jon </FNAMExLNAME> Oq </LNAME>
    </NAME>
</STAFF>
```

Bunday holda, NAME element to'liq STAFF elementiga joylashtirilgan va FNAME va LNAME elementlari NAME elementiga joylashtirilgan.

Atributlar

Atributlar - bu element haqida tavsiflovchi ma'lumotlarni o'z ichiga olgan nom-qiymat juftlari. Atributlar mos keladigan element nomidan keyin boshlang'ich deskriptorning ichiga joylashtiriladi va atribut qiymati tirnoqlarga ilova qilinadi. Masalan, ko'rib chiqilayotgan ro'yxatni tayyorlash paytida, STAFF elementining branchNo atributidan foydalanib, ushbu xodim ishlaydigan bo'limning sonini ko'rsatishga qaror qilindi:

```
<STAFF branchNo = "B005">
```

Shuni ta'kidlash kerakki, ajratish to'g'risidagi ma'lumotlar STAFF elementining pastki elementi sifatida teng ravishda taqdim etilishi mumkin. Va kompaniya xodimining maydoni haqida ma'lumotni taqdim etish uchun siz bo'sh elementning atributidan foydalanishingiz mumkin, masalan, quyidagicha:

```
<SEX jinsi = "M" />
```

Har bir o'ziga xos atribut deskriptorda faqat bitta misolda mavjud bo'lishi mumkin, shu bilan birga bir xil deskriptordagi tagliklar takrorlanishi mumkin. Shuni ta'kidlash kerakki, bu holda noaniqlik paydo bo'lishi mumkin: biron bir element yoki atributdan foydalanib, bo'lim raqami yoki xodimning maydoni haqida ma'lumot berilishi kerakmi?

Tashkilotga havolalar

Korxonalar quyidagi asosiy vazifalarni bajaradigan hujjat elementlari:

- tez-tez takrorlanadigan matn uchun qisqartirish sifatida xizmat qiladi yoki tashqi fayllar tarkibini hujjatga kiritishga imkon beradi;
- matnga ixtiyoriy Unicode belgilarini kiritish uchun foydalanilgan (masalan, to'g'ridan-to'g'ri klaviaturada kiritib bo'lmaydigan belgilarni ko'rsatish uchun);
- Sizga ajratilgan belgilar va tarkibni farqlash imkonini beradi. Masalan, chap burchakli qavs (<) boshlang'ich yoki tugatish elementlari tavsifining boshlanishini bildiradi. Ushbu markirovka belgisini tarkibning o'zida mavjud bo'lgan belgidan ajratish uchun, bu ob'ekt <belgini almashtirishga xizmat qiladigan XML tiliga kiritildi.

Har bir ob'ekt o'ziga xos nomga ega bo'lishi kerak va XML hujjatidan foydalanilishi ob'ektga murojaat qilish deb ataladi. Har qanday ob'ektga oid ma'lumotnoma & () belgisi bilan boshlanadi va nuqta-vergul (;) bilan tugaydi, masalan & lt;.

Sharhlar

Sharhlar <! - va -> deskriptorlarida joylashtirilgan va "-" harfli satridan tashqari har qanday ma'lumotlarni o'z ichiga olishi mumkin. Sharhlar XML hujjatining istalgan joyida belgilash deskriptorlari o'rtasida joylashtirilishi mumkin, ammo XML protsessorida sharhlarni dasturga berish shart emas.

17.4 CDATA bo'limlari va ishlov berish bo'yicha ko'rsatmalar

CDATA bo'limi - bu protsessor ushbu bo'limda joylashgan belgilarni e'tiborsiz qoldirishi va unga ilova qilingan matnni to'g'ridan-to'g'ri dasturga izohsiz topshirishi kerak bo'lgan XML protsessoriga ishora. Dasturga qo'shimcha ma'lumotlarni etkazish uchun ishlov berish buyruqlaridan ham foydalanish mumkin. Qayta ishlash buyrug'i `<? Name pidata?> Rasmiga ega` , bu erda `nom` dastur uchun ishlov berish buyrug'ining identifikatori bo'lib xizmat qiladi. Buyruqlar dasturga tegishli bo'lganligi sababli, har qanday XML hujjatida bir xil amallarni bajarish uchun turli xil dasturlarga buyruqlar berishga imkon beradigan bir nechta qayta ishlash buyruqlari bo'lishi mumkin, ammo ehtimol turli xil usullar bilan.

Tartibga solish

1-bo'limda tasvirlangan ma'lumotlar tizimining yomon tuzilishida yig'ilishlar tartibsiz deb taxmin qilinadi, XML-da esa elementlar buyurtma qilingan deb hisoblanadi. Shunday qilib, XML-da, FNAME va LNAME-ning qayta o'zgartirilgan elementlari bo'lgan quyidagi ikkita bo'lak boshqacha deb hisoblanadi:

```
<NAME> <NAME>
                <FNAME> Jon </ FNAME> <LNAME> Oq
</LNAME>
                <LNAME> Oq </LNAME> < FNAME > Jon
</FNAME>
</ NAME > </ NAME >
```

Bunga javoban, XML-da atributlar buyurtma qilinmagan, shuning uchun quyidagi ikkita XML elementi bir xil deb hisoblanadi:

```
<NAME FNAME = "John" LNAME * "White"/>
<NAME LNAME = "White" FNAME = "John"/>
```

Hujjat turini aniqlash (DTD)

Lug'at atamasi ba'zan ma'lum bir dasturda ishlatiladigan elementlarga murojaat qilish uchun ishlatiladi. Grammatika deb ham ataladigan sintaktik tuzilish XML sintaksisidan emas, balki ***EBNF (Kengaytirilgan Backus Naur formasi)*** yordamida aniqlanadi. Garchi DTD ta'rifi ixtiyoriy bo'lsa ham, quyida tavsiflanganidek, uning muayyan talablarga javob berishini tekshirish uchun har bir hujjat uchun taqdim etilishi tavsiya etiladi.

Xodimlar ma'lumotlari misolida davom ettirib, 2- Listingda ko'rsatilgan XML hujjati uchun 3-ro'yxatdagi DTD ta'rifini ko'rib chiqing, bu holda DTD ta'rifi alohida tashqi faylda bo'ladi, lekin uni to'g'ridan-to'g'ri XML hujjatiga qo'shish mumkin. DTD deklaratsiyalari quyidagi to'rtga bo'linadi: quyida muhokama qilinadigan turlari; elementlar deklaratsiyalari, atributlar ro'yxati deklaratsiyalari, ob'ekt deklaratsiyalari va belgilar deklaratsiyalari.

Mahsulot turi deklaratsiyalari

Element turi deklaratsiyalari XML hujjatida yuzaga kelishi mumkin bo'lgan elementlardan foydalanish qoidalarini belgilaydi. Masalan, 3-Listing `STAFFLIST` elementi uchun quyidagi qoidani belgilaydi (shuningdek, tarkib modeli deb ham ataladi) :

```
<!ELEMENT STAFFLIST (STAFF)*>
```

3- Listing. 2- Listingdagi CMC hujjatiga mos keladigan hujjat turini aniqlash

```
<! ELEMENT STAFFLIST (STAFF) *>
<! ELEMENT STAFF (NOMI, POSITION, DOB?, SALARY)>
<! ELEMENT NOMI (FNAME, LNAME}>)
<! Element elementi (#PCDATA)>
<! ELEMENT LNAME (#PCDATA)>
<! ELEMENT POSITION (ftPCDATA)>
<! ELEMENT DOB (#PCDATA)>
<! ELEKTR SALARY {#PCDATA)>
<! ATTLIST STAFF filialiNo CDATA ft IMPLIED>
```

Ushbu qoida `STAFFLIST` elementi nol yoki undan ko'p bo'lishi mumkin bo'lgan `S TAFF` elementlaridan iboratligini bildiradi. Takrorlash tezligini ko'rsatish uchun quyidagi belgilarni sinab ko'rish mumkin:

- yulduzcha (*) elementda sub-elementning paydo bo'lishi soni nol yoki undan ko'p bo'lishi mumkinligini bildiradi;
- plyus belgisi (+) berilgan element uchun bitta yoki undan ko'p bo'lishi mumkinligini bildiradi;
- savol belgisi (?) uchraydigan holatlar soni nol yoki bitta bo'lishi mumkinligini bildiradi.

Agar element nomi paydo bo'lish sonini ko'rsatadigan belgilarsiz berilgan bo'lsa, element hujjatda aniq bir marta paydo bo'lishi kerak. Element nomlari orasidagi vergullar elementlarning hujjatda belgilangan ketma-ketlikda bo'lishi kerakligini bildiradi va agar element nomlari o'rtasida vergul bo'lmasa, elementlar hujjatda istalgan tartibda paydo bo'lishi mumkin.

Masalan, STAFF elementi uchun quyidagi qoida belgilangan:

```
<! ELEMENT STAFF (NOMI, POSITION, DOB?, SALARY)>
```

bu element haqida qoida ma'lumotlarni o'z ichiga oladi `STAFF` elementlar iborat `NAME` va `POSITION`, ixtiyoriy element `DOB` va element `SALARY`, bu tartibda ta'rifi, shuningdek, reklama elementlar uchun berilishi kerak `FNAME`, `LNAME`, `POSITION`, `DOB` va `SALARY` va kontent modeli ishlatiladigan boshqa barcha elementlar. Bu sizga hujjatni tekshirish uchun XML protsessoridan foydalanishga imkon beradi. Ushbu barcha asosiy elementlar maxsus `ftPCDATA` belgilaridan foydalangan holda e'lon qilingan , bu ularda talqin qilingan belgilar to'g'risidagi ma'lumotlarning mavjudligini ko'rsatadi. Shuni ta'kidlash kerakki, har qanday element faqat elementlardan iborat bo'lishi mumkin, ammo boshqa elementlarni va `#PCDATA` parchasini qo'shish mumkin (bu holda aralash tarkibni o'z ichiga olgan atama elementi qo'llaniladi).

Atributlar ro'yxati deklaratsiyalari.

Attributlar ro'yxati deklaratsiyalari qaysi elementlar atributlarga ega bo'lishi mumkinligini, qaysi atributlar tarkibiga kirishi mumkinligini, qaysi atributlar

atributlarga ega bo'lishi mumkinligini va asl qiymati sifatida ishlatiladigan ixtiyoriy atribut qiymatlarini bildiradi, har bir atribut deklaratsiyasi uchta qismdan iborat: ism, tur va qo'shimcha qiymat sukut bo'yicha. Quyida oltita mumkin bo'lgan atribut turlari mavjud.

CDATA. Har qanday matnni o'z ichiga olgan belgilar haqidagi ma'lumotlar. Ushbu turdagi satr XML protsessor tomonidan sintaktik tahlil qilinmaydi va oddiygina to'g'ridan-to'g'ri dasturga o'tkaziladi.

ID. Hujjatda individual elementlarni ko'rsatish uchun foydalaniladigan identifikator. Identifikatorlari ID element nomini mos bo'lishi kerak, va barcha qadriyatlar ID hujjatda foydalaniladigan, turli xil bo'lishi kerak.

IDREF yoki IDREFS. Hujjatdagi element uchun bitta identifikator atributi qiymatiga mos kelishi kerak bo'lgan bir yoki bir nechta identifikator ma'lumotlari. IDREFS atributida bo'sh joy bilan ajratilgan bir nechta IDREF qiymatlari bo'lishi mumkin .

ENTITY yoki ENTITIES, bitta tashkilot nomiga mos kelishi kerak bo'lgan bir yoki bir nechta belgi. ENTITIES atributida bo'sh joy bilan ajratilgan bir nechta ENTITY qiymatlari bo'lishi mumkin .

NMTOKEN yoki NMTOKENS. Odatda bitta so'zdan iborat bo'lgan cheklangan formatli satr. NMTOKENS atributida bo'shliqlar bilan ajratilgan bir nechta NMTOKEN qiymatlari bo'lishi mumkin .

Ismlar ro'yxati. Ushbu atribut ega bo'lishi mumkin bo'lgan qiymatlar (boshqacha aytganda, ro'yxatga olingan tur).

Misol uchun, xususiyati quyidagi deklaratsiya xususiyati aniqlash uchun ishlatiladigan branchNo element XODIMLAR :

```
<LATTLIST STAFF filiali CDATA #IMPLIED>
```

Ushbu deklaratsiyada aytilishicha, • branchNo atributining qiymati satr (CDATA - belgi ma'lumotlari) va ixtiyoriy (! IMPLIED), va u uchun odatiy qiymat berilmaydi. Bundan tashqari kalit so'z uchun !IMPLIED, kalit beradi #REQUIRED xususiyati har doim element o'rnatilgan bo'lishi kerakligini ko'rsatadi. Agar ushbu talabga javob beradigan kalit so'zlarning hech biri ko'rsatilmagan bo'lsa, unda atribut e'lon qilingan standart qiymatni o'z ichiga olishi mumkin. Atribut har doim standart qiymatga ega bo'lishi kerakligini ko'rsatish uchun ttFIXED kalit so'zi ishlatiladi . Misol sifatida, SEX elementini M (standart) yoki F qiymatini o'z ichiga olgan gender atributiga (jinsga) ega deb aniqlash mumkinligini quyidagicha ko'rsatamiz:

```
<!ATTLIST SEX gender (M | F } "M">
```

Shaxslar va belgilarning deklaratsiyalari.

Mualliflik deklaratsiyalari sizga nomni ba'zi bir ma'lumot tarkibiy qismlari bilan bog'lash imkonini beradi, masalan oddiy matndan parcha, DTD ta'rifining bir qismi yoki matn yoki ikkilik ma'lumotlarni o'z ichiga olgan tashqi faylga havola. Nota deklaratsiyalari shunchaki XML protsessor tomonidan dasturga o'tkaziladigan tashqi ikkilik ma'lumotlarga ishora

qiladi. Masalan, "DreamHome Estate Agents" matni uchun siz quyidagini e'lon qilishingiz mumkin:

```
<! ENTITY DH "DreamHome Estate Agents">
```

Tashqi tushunarsiz ob'ektlarni qayta ishlash uchun javobgarlik ilovaga yuklangan. Korxonaning joylashgan joyini ko'rsatadigan identifikatordan so'ng, ushbu ob'ektning ichki formati to'g'risidagi ba'zi ma'lumotlar e'lon qilinishi kerak, masalan, quyidagilar:

```
<! ENTITYITYHomeLogo TIZIMI "dreamhome.jpg" NDATA
JPEGFormat>
<! NOTATION JPEGFormat Tizimi "http://www.jpeg.org">
```

Bu erda `NDATA` kalit so'zining mavjudligi mantiqiy ob'ektning sharhlanmaganligini ko'rsatadi; ushbu kalit so'zdan keyin o'zboshimchalik bilan berilgan nom shunchaki belgining keyingi deklaratsiyasining kalitidir. Notation deklaratsiyasida bu nom ilova qanday ishlov berilganligini aniqlash uchun foydalanadigan identifikator bilan taqqoslanadi.

17.5 Element identifikatorlari, ID identifikatorlari va identifikatorlar uchun havolalar

Yuqorida aytib o'tilganidek, XML tili `identifikator` atributi turini belgilashga imkon beradi, uning yordamida element bilan noyob kalitni bog'lashingiz mumkin. Bunga qo'shimcha ravishda, `IDREF` atribut `turi sizga` elementda ko'rsatilgan kalit bilan boshqa elementga havolani kiritishga imkon beradi, `IDREFS` atribut `turi` esa elementdagi bir nechta boshqa elementlarga havolani chizishga imkon beradi. Masalan, norasmiy `Branch Has Staff` o'zaro aloqalarini modellashtirish uchun, siz `BRANCH` va `STAFF` elementlari uchun quyidagi ikkita atributni aniqlashingiz mumkin:

```
<!ATTLIST STAFF staffNo ID #REQUIRED>
<!ATTLIST BRANCH staff IDREFS #IMPLIED>
```

Keyinchalik 4- Listingda ko'rsatilganidek, siz ushbu xususiyatlarni qo'llashingiz mumkin.

```
<STAFF staffNo = "SL21">
<NAME>
<FNAME>John</FNAME><LNAME>White</LNAME>
</NAME>
</STAFF>
<STAFF staffNo = "SL41">
<NAME>
<FNAME>Julie</FNAMExLNAME>Lee</LNAME>
</NAME>
</STAFF>
«BRANCH staff = "SL21 SL41">
```

```
<BRANCHNO>B005</BRANCHNO>
</BRANCH>
```

4-Listing. ID va IDREFS atributlari turlaridan foydalanishga misol

Hujjatni ruxsat berilganlikni tekshirish

XML spetsifikatsiyasi ishlov berish paytida hujjatni tekshirish uchun ikki bosqichni taqdim etadi: rasmiy tasdiqni tekshirish va yaroqlilikni tekshirish. Tekshiruvni ta'minlamaydigan protsessor, XML hujjati ma'lumotlarini dasturga topshirishdan oldin, hujjat rasmiy ravishda to'g'riligini tekshiradi. XML hujjatining tuzilishi va belgilanishini belgilaydigan qoidalarga mos keladigan har qanday XML hujjati shunday hisoblanadi. Bundan tashqari, rasmiy ravishda tasdiqlangan XML hujjatlari quyidagi talablarga javob berishi kerak:

- Hujjat XML deklaratsiyasi bilan boshlanishi kerak - <? Xml versiyasi "1.0"?>;
- barcha elementlar bitta ildiz elementiga kiritilishi kerak;
- elementlar bir-birining ichiga daraxt tuzilishi rasmida, elementlarning bir-birining ustiga chiqmasdan joylashtirilishi kerak;
- barcha bo'sh bo'lmagan elementlar boshlang'ich va oxirgi tutqichlarga ega bo'lishi kerak.

Tekshiruvchi nafaqat XML hujjati rasmiy ekanligini tasdiqlashi kerak, balki u DTD ta'rifiga javob berishini ham aniqlashi kerak; agar bu tekshirish amalga oshirilmasa, XML hujjati haqiqiy deb hisoblanadi. Yuqorida aytib o'tilganidek, DTD ta'rifi XML hujjatida bo'lishi mumkin yoki unda havolada ko'rsatilishi mumkin. Hozirgi vaqtda W3C DTD-ga qaraganda ifodali va XML sxemasi sifatida tanilgan hujjatni tekshirish usulidan foydalanishni tavsiya qiladi. XML Schema spetsifikatsiyasini tavsiflashga o'tishdan oldin, biz XML sxemasi ta'riflarida ishlatiladigan ba'zi bir XML bilan bog'liq texnologiyalarni ko'rib chiqamiz.

17.6 XML texnologiyalari

DOM va SAX interfeyslari.

XML API asosan ikki toifaga bo'linadi: daraxtga asoslangan va voqeaga asoslangan.

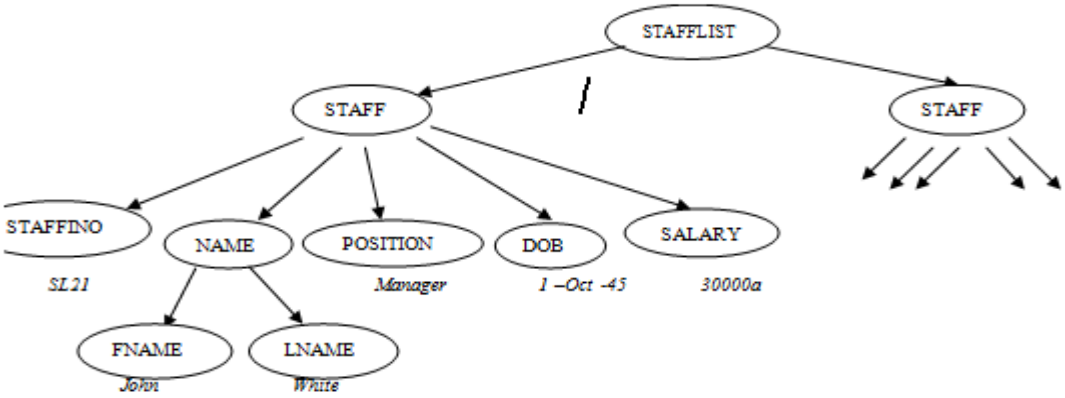
Interface ***DOM (Document Object Model - Document ob'ektni Model)*** ob'ekt yo'naltirilgan ma'lumotlar vakillik yaratish imkonini beradi daraxt ko'rinishida, asoslangan, XML API-interfeysi. Ushbu API W3C tomonidan yaratilgan va har qanday rasmiy haqiqiy XML yoki HTML hujjatlarini ichki rasmga aylantirishga imkon beradigan bir qator mustaqil (platforma va til) interfeyslarni tavsiflaydi. DOM interfeysi XML hujjatining RAMdagi daraxtga o'xshash vakilligini hosil qiladi va dasturda ushbu daraxt elementlarini ko'chirish va qayta ishlashga imkon beradigan sinflar va usullarga kirishni ta'minlaydi. Xususan, DOM modeli "Element", "Attribute" va "Character-Data" kichik sinflari bilan tugun interfeysini belgilaydi. Node interfeysi ota tugunni qaytaradigan parentNode {} kabi tugunning tarkibiy qismlariga kirish usullarini va ko'pgina bolalar tugunlarini qaytaradigan childNodes () ni o'z ichiga oladi. Umuman olganda, DOM interfeysi elementlarni qo'shish yoki o'chirish, shuningdek elementlarning tartibini

o'zgartirish kabi XML daraxti bilan tizimli manipulyatsiyalarni bajarish uchun eng mos keladi.

17.2-rasmda 2- ro'yxatda ko'rsatilgan XML hujjati daraxt tuzilishi sifatida ko'rsatilgan va bu ma'lumotni OEM grafigi sifatida taqdim etish (1-rasmga qarang) va XML elementlari daraxti kabi muhim farqni aniqlash kerak. OEM ko'rinishida grafik chetlarida yorliqlarga va XML ko'rinishida tugunlarga teglar mavjud. Agar ma'lumotlar ierarxik tuzilishga ega bo'lsa, ularni bir vakillikdan ikkinchisiga osongina aylantirish mumkin va agar ma'lumotlar grafik rasmida taqdim etilsa, bunday o'zgartirish ancha murakkablashadi.

SAX (XML uchun oddiy API) - bu hujjatni tahlil qilish paytida ro'y beradigan voqealar to'g'risida dasturga xabar yuborish uchun qo'ng'iroqni qaytarish funktsiyalaridan foydalanadigan voqeaga asoslangan XML ketma-ket kirish API. Masalan, elementlarni ishga tushirish va tugatish tavsiflovchilari aniqlanganda hodisalarni boshlash mumkin. Ilova ushbu tadbirlarni boshqarish uchun ixtisoslashtirilgan tadbirlarni boshqaruvchilardan foydalanadi. Daraxtga asoslangan API-lardan farqli o'laroq, voqealarga asoslangan API-lar XML xotirasidagi RAM-ga daraxtga o'xshash vakillikni ta'minlamaydi.

Shuni ta'kidlash kerakki, SAX API XML-DEV pochta ro'yxatiga kiritilgan ishlab chiquvchilarning birgalikdagi harakatlari natijasida yaratilgan va W3C dasturiy mahsuloti emas.



17.2- rasm. XML hujjatining 2- Listingda daraxt tuzilishi sifatida namoyishi.

Ism maydonlarining tavsifi.

Ism maydonlari spetsifikatsiyasi XML hujjatidagi elementlarning nomlari va havolalariga maxsus malakaviy qaydlarni qo'llash imkoniyatini beradi. Bu bir xil nomga ega, ammo turli lug'atlarda aniqlangan elementlardan foydalanganda nomlarning to'qnashuvidan qochadi. Bu turli nomlar hududlarida aniqlangan hujjatdagi deskriptorlardan foydalanish imkonini beradi. Agar hujjatdagi ma'lumotlar bir nechta manbalardan olingan bo'lsa, bu zarur. Nom doiralari W3C tavsiyalarida ham aniqlangan. Ularning noyobligini ta'minlash uchun elementlar va atributlarga URI havolasidan foydalanib, global miqyosda noyob nomlar beriladi. Masalan, quyidagi hujjatning parchalari ildiz elementida e'lon qilingan ikkita turli xil nomlarni ishlatadi. Ulardan birinchisi ("http://www.Dreamhome. Co.uk/branch5/") standart nomlar maydoni sifatida xizmat qiladi, shuning uchun

saralashga ega bo'lmagan barcha elementlar ushbu nom maydoniga tegishli ekanligi tushuniladi.

Ikkinchi ism maydoniga ("http://www.dreamhome.co.uk/HQ/") nom berilgan (hq), bundan keyin SALARY elementi oldingi element sifatida foydalaniladigan nom maydonini ko'rsatish uchun ishlatiladi:

```
<STAFFLIST xmlns =
"http://www.dreamhome.co.uk/branch5/1"
        xmlns: hq = "http://www.dreamhome.co.uk/HQ/">
    <STAFF branchNo = "B005">
        <STAFFNO> SL21 </STAFFNO>
        ...
        <hq: SALARY> 30000 </ hq: SALARY>
    </STAFF>
</STAFFLIST>
```

XSL va XSLT tillari.

Uslub - bu ma'lumot elementlarining brauzerda qanday namoyish etilishini rasmiylashtirilgan tavsifi. HTML-da ma'lumotlarni qayta ishlashda standart uslublar ma'lumotlari brauzer dasturiga kiritilgan. Ushbu imkoniyat HTML uchun tavsiflovchilar to'plami oldindan aniqlangan va doimiy bo'lganligi bilan bog'liq. Ishlab chiquvchilar HTML tavsiflovchilari bilan ishlashning boshqa usulini ham ko'rib chiqishi mumkin. Buning uchun ***uslublar jadvalining kaskadli spetsifikatsiyasi (CSS***) ishlatiladi. HTML-dan farqli o'laroq, XML hujjatlarida standart uslublar mavjud emas. XML hujjatini brauzerda namoyish qilish uchun CSS spetsifikatsiyasi ham ishlatilishi mumkin, ammo bu hujjatga tarkibiy o'zgartirishlarni kiritishga imkon bermaydi. Shuning uchun W3C ***kengaytirilgan*** uslublar ***jadvallari tilidan (XSL***) foydalanish bo'yicha rasmiy tavsiyalar berdi , bu XML ma'lumotlari brauzerda qanday ko'rsatilishini va shuningdek bitta XML hujjatini boshqasiga qanday o'zgartirishni aniqlash uchun maxsus yaratilgan. Ushbu til CSS spetsifikatsiyasiga o'xshashdir, ammo ancha kuchli.

Kengaytiriladigan stillar jadvallari tili islohotlarni (qo'llab-quvvatlash uchun ***kengaytiriladigan stillar jadvallari tili uchun konvertatsiya - XSLT*** XSL til pastki to'plamidir). Bu ikkala belgilash tili va dasturlash tili hisoblanadi, chunki u XML strukturasini boshqa har qanday XML tuzilishiga, HTML formatiga yoki boshqa har qanday matn formatiga (masalan, SQL kabi) o'zgartirish mexanizmlarini ta'minlaydi. XSLT veb-sahifaning ekran formatini tasvirlash uchun ishlatilishi mumkin bo'lsa-da, uning asosiy ustunligi - bu CSS-da bo'lgani kabi, chiquvchi qurilmalarda ushbu tuzilmalarni taqdim etish bilan birga, asosiy ma'lumotlar tuzilmalarini o'zgartirish qobiliyati.

XSLT muhim ahamiyatga ega, chunki u hujjat ko'rinishi va ma'lumotlarni filtrlash usulini dinamik ravishda o'zgartirish mexanizmlarini ta'minlaydi. Ushbu til shuningdek dastur algoritmlarini kodlash uchun etarli darajada ishonchli. Bundan tashqari, olingan ma'lumotlardan (va nafaqat hujjatlardan) grafik tasvirlarni yaratishga imkon beradi . XSLT tili hatto serverlar bilan (ayniqsa XSLT kodga

o'rnatilishi mumkin bo'lgan skript modullari bilan birgalikda) aloqa o'rnatishga imkon beradi va tegishli xabarlarni to'g'ridan-to'g'ri XSLT kodini yaratishga imkon beradi. 5- Listingda XML hujjatini 2- Listingda ko'rsatish uchun ishlatiladigan XSL uslublar jadvali ko'rsatilgan.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-
xsl">
<xsl:template match="/">
    <html>
    <body background ="sky.jpg">
    <center><h2><i>DreamHome</i>Estate
Agents</h2></center>
    <table border-"Iм bgcolor="#ffffff ">
    <tr>
        <th bgcolor= "#cOcOcO" bordercolor=
    "#000000">staffNo</th>
<!- Boshqa ustun sarlavhalari uchun takrorlang >
    </tr>
    <XSl:for-each select= "STAFFLIST/STAFF">
    <tr>
        <td bordercolor="#cOcOcO"><xsl:value-of
    select="STAFFNO"/>
        </td>
        <td border-color ="#cOcOcO"><xsl:value-of
    select="NAME/FNAME"></td>
<!-- Boshqa narsalar uchun takrorlang -->
    </tr>
    </xsl:for-each>
    </table>
    </body>
    </html>
</xsl:template>
</xsl:stylesheet>
```

5-Listing. XML hujjatini 2- Listingda ko'rsatish uchun XSL uslublar jadvali

XPath tili (XML Path).

XPath - bu XML hujjatining alohida qismlariga murojaat qilish uchun oddiy sintaktik konstruktsiyalarni ta'minlovchi XML uchun deklarativ so'rovlar tili. Quyida tavsiflanganidek, ushbu til XSLT tili (shablonga mos kelganda) va XPointer tili (manzilni qayta ishlash paytida) bilan ishlashga mo'ljallangan. XPath tili katalogga olib boradigan yo'l bilan bir xil printsip asosida yaratilgan yo'lni ko'rsatib, elementlarning to'plamlarini tanlashga imkon beradi. Bundan tashqari, yo'l belgisi shartli iboralarni o'z ichiga olishi mumkin yoki bo'lmasligi mumkin. XPath XML elementlaridan foydalanishga asoslangan tuzilgan sintaksisga emas, balki

ixcham torli sintaksisdan foydalanadi, shuning uchun XPath ifodalarini XML atributlarida ham, URIda ham ishlatish mumkin.

XPath-da XML hujjati mantiqiy (buyurtma qilingan) daraxt deb hisoblanadi, unda tugunlar har bir element, atribut, matn, ishlov berish ko'rsatmasi, sharh, nom maydoni va ildiz elementini belgilaydi.

Murojaat qilish mexanizmi kontekst tugunidan (masalan, yo'lning boshlang'ich nuqtasi) va mahalliylashtirish yo'lidan foydalanishga asoslangan (bu XML hujjatining bir nuqtasidan ikkinchisiga o'tish yo'lini tavsiflaydi va shu bilan XML hujjat elementlariga murojaat qilish usulini ta'minlaydi). XPointer yordamida hujjatdagi elementning mutlaq yoki relatsiya joylashuvini ko'rsatish uchun foydalanish mumkin. Lokalizatsiya yo'li katalog yo'lidagi / belgi bilan bir xil vazifani bajaradigan (/) belgi bilan ajratilgan bir qator qadamlardan iborat (bu erda lokalizatsiya yo'li atamasi keladi). Har bir kesish belgisi oldingi bosqichga nisbatan daraxt ko'rinishining pastki darajasiga o'tishga imkon beradi.

Har bir qadam asosiy va ixtiyoriy predikatlardan iborat, va asos eksa nomi va tugunni tekshirish shartidan iborat. Tugunni tekshirish holati hujjatdagi tugun turini aniqlashga imkon beradi. Odatda, unda element nomi tekshiriladi, lekin matn () (tugunning matnli ekanligini tekshirish uchun) yoki tugun () (har qanday turdagi tugunni tekshirish uchun) kabi funktsiyalardan ham foydalanish mumkin. XPath bolalarni 13 turini belgilaydi, ularning ichida ajdod (ajdod), atribut (atribut) va bola (avlod). Masalan, rasmda ko'rsatilgandek. 4 -rasmda `STAFF` bir o'qi bor `child` besh tugunlari (o'z ichiga olgan, `STAFFNO`, `NAME`, `POSITION`, `DOB` va `SALARY`). Predikat kvadrat qavslarga o'ralgan va asosdan keyin bo'lishi kerak. Agar elementda bittadan ko'p sub-element bo'lsa, ma'lum bir elementni tanlash uchun `[positionO = positionNumber]` konstruktsiyasidan foydalanish mumkin, bu erda `positionNumber` – 1-dan boshlanadigan pozitsiya raqami. XPath-da to'liq va qisqa sintaksis qo'llab-quvvatlanadi. Mahalliylashtirish yo'llarining ba'zi misollari 17.2-jadvalda keltirilgan.

17.2-jadval. Mahalliylashtirish yo'llariga ba'zi misollar

Mahalliylashtirish yo'li	**Uchrashuv**
.	Kontekst tugunini tanlaydi
..	Kontekst tugunining qarindoshlik-tugunini (ajdodini) tanlaydi
/	Ildiz tugunini tanlaydi yoki yo'ldagi qadamlar orasidagi ajratuvchi sifatida xizmat qiladi
//	Joriy tugunning bola tugunlarini (avlodlarini) tanlaydi
/child::STAFF	Ildiz tugunining bolalari bo'lgan ACE STAFF elementlarini tanlaydi
child::STAFF (yoki oddiy	Kontekst tugunining bolalar tugunlari bo'lgan ACE

STAFF)	STAFF elementlarini tanlaydi
attribute т:branchNo (yoki oddiy ©branchNo)	Kontekst tugunining branchNo atributini tanlaydi
attribute::* (yoki oddiy®*)	Kontekst tugunining barcha atributlarini tanlaydi
child::STAFF[3]	Kontekst tugunining bolalar tugmasi bo'lgan uchinchi STAFF elementini tanlang
/child::STAFF[®branchNo = "B005"]	BAT5 ga teng branchHo atributiga ega bo'lgan barcha STAFF elementlarni tanlaydi
/child::STAFF[®branchNo = "B005"][position () =1]	B 005 qiymatli filialNo qiymati bilan atributga ega bo'lgan birinchi STAFF elementni tanlaydi

XPointer tili (XML Pointer).

XPointer atribut qiymatlariga yoki XML hujjatning istalgan joyida joylashgan elementlarning tarkibiga kirishni ta'minlaydi. Har qanday XPointer ko'rsatkichi URIdagi XPath ifodasidir. Boshqa narsalar qatorida, XPointer tili sizga matnning qismlariga havolalar yaratishga, aniq elementlarni yoki atributlarni tanlashga va bir elementdan boshqasiga o'tishga imkon beradi. Ushbu til bir nechta tugunlar to'plamidagi ma'lumotlarni tanlashga imkon beradi, XPath-dan foydalanishda esa bu vazifani bajarib bo'lmaydi.

Tugunlarning ta'rifiga qo'shimcha ravishda, XPointer tili shuningdek, tugunlar bilan birgalikda ma'lumotlarning joylashuvini ko'rsatishga imkon beradigan nuqta va diapazonlarni ham belgilaydi. Nuqta - bu XML hujjatidagi mavqe, va diapazon XML-ning barcha struktura va tarkibini bildiradi, ularning har ikkalasi ham tugun o'rtasida bo'lishi mumkin. Misol uchun, keyingi ko'rsatkich XPointer boshlanadigan bola element bilan boshlanadi bir qator bildiradi `STAFF` bir xususiyati qiymati ega, `branchNo` , teng `B005` , va bola element oxirida tugaydi `XODIMLAR` bir xususiyati qiymati ega, `branchNo`, teng `BOO3`:

```
XPointer(/child::STAFF[attribute:rbranchNo = "B005"] to
/child::STAFF[attribute::branchNo = "B003"])
```

Ushbu misolda ushbu dizayn ikkala `STAFF` tugunlarini tanlashga imkon beradi .

XLink tili (XML Linking).

XLink sizga manbalar orasidagi aloqalarni yaratish va tavsiflash uchun foydalanishingiz mumkin bo'lgan XML hujjatlariga maxsus elementlarni qo'shish imkonini beradi. Oddiy, bir yo'naltirilgan HTML giperhavolalarga o'xshash havolalarni, shuningdek yanada murakkab havolalarni tavsiflash uchun tuzilmalarni yaratish uchun XML sintaksisidan foydalanadi. XLink havolalari ikki turdan biri bo'lishi mumkin: oddiy va rivojlangan. Oddiy havola manbani maqsadli manba bilan

bog'laydi va kengaytirilgan havola o'zboshimchalik bilan resurslarni ulaydi. Bundan tashqari, ushbu til havolalarni alohida ma'lumotlar bazasida saqlash imkoniyatini beradi (bu bog'lanish bazasi - linkbase deb nomlanadi). Bu resurslarning joylashuvidan ma'lum darajada mustaqillikka erishish imkonini beradi: hatto havolalarga o'zgartirishlar kiritilgan taqdirda ham, asl XML hujjatlari o'zgarishsiz qoladi va yangilanishlar faqat havolalar bazasida amalga oshiriladi.

XHTML tili.

XHTML (kengaytirilgan HTML - kengayadigan HTML tili) 1.0 versiyasi - bu HTML 1.0 formatida tuzilgan HTML 4.01 tili. U keyingi avlod HTML tili sifatida foydalanish uchun mo'ljallangan va aslida HTMLning yanada qat'iy va aniq ifodalangan versiyasidir. Masalan, u quyidagi talablarni ta'minlaydi:

- tavsiflovchi va atribut nomlari katta harf bilan yozilishi kerak;
- Barcha XHTML elementlari izohlovchi tavsifga ega bo'lishi kerak;
- atribut qiymatlari tirnoqlarga ilova qilinishi kerak va ularni qisqartirish usuliga yo'l qo'yilmaydi;
- ism atributi o'rniga ID atributidan foydalanish kerak;
- Hujjatlar XML qoidalariga muvofiq bo'lishi kerak.

XML sxemasini aniqlash.

XML 1.0 spetsifikatsiyasi DTD mexanizmidan tarkibiy modelni (elementlarning maqbul tartibini va joylashishini), shuningdek (cheklangan darajada) XML hujjat atributlarining ma'lumot turlarini aniqlashda foydalanadi, ammo bu mexanizm bir qator kamchiliklarga ega:

- DTD ta'riflari XML-dan boshqa sintaksis yordamida amalga oshiriladi;
- ular nom bo'shlig'i yordamiga ega emaslar;
- ular sizga juda cheklangan ma'lumotlar turlarini aniqlashga imkon beradi.

Shuning uchun XML hujjatining tarkibiy modelini aniqlash uchun yanada to'liq va qat'iy usulni taqdim etish zarurati tug'ildi. W3C tomonidan ishlab chiqilgan XML sxemasi spetsifikatsiyasi ushbu kamchiliklardan xoli va DTD ta'riflariga qaraganda ancha katta ifodali kuchga ega. Qo'shimcha ma'lumotlar taqdim etish imkoniyatlari faqat ma'lum bir dastur talablariga javob beradigan ishlab chiqilgan tekshirish vositalaridan foydalanish bilan solishtirganda veb-ilovalar o'rtasida ancha ishonchli XML ma'lumot almashinuvini ta'minlaydi.

XML sxemasi - bu ma'lum bir XML tuzilmaning (uning tashkil etilishi va unda ishlatiladigan ma'lumotlar turlarining) ta'rifi. XML sxemasi tili sxemada har bir element turini qanday aniqlashni tavsiflaydi va har bir element bilan bog'liq ma'lumotlar turlarini ko'rsatishga imkon beradi. Sxemaning o'zi bu sxemaning semantikasini ifoda etuvchi elementlar va atributlardan foydalanadigan XML hujjati. Sxema XML hujjati bo'lganligi sababli, u tomonidan tasvirlangan XML hujjatiga ishlov berish uchun ishlatiladigan asboblar yordamida tahrir qilinishi va ishlov berilishi mumkin. Ushbu bo'limda Listing 2-da XML hujjati uchun XML sxemasini yaratishga misol keltirilgan.

Oddiy va murakkab toifalar

Ehtimol, XML sxemasini yaratishning eng oson usullaridan biri bu hujjat tuzilishini kuzatish va har bir elementni topilganidek aniqlashdir. Boshqa elementlarni o'z ichiga olgan elementlar murakkab Type elementlari. Masalan, `STAFFLIST` ildiz elementini topsangiz, u kompleksType turiga tegishli ekanligini aniqlashingiz mumkin. `STAFFLIST` elementi bolalarining ro'yxati ketma-ketlik elementi yordamida tasvirlangan. Bu submelementlarning tartiblangan ketma-ketligini aniqlaydigan kompozitor element turiga tegishlidir:

```
<xsd:element name = "STAFFLIST">
<xsd:complexType>
<xsd:sequence>
 <! - Bu erda bolalar elementlarining ta'riflari
bo'lishi kerak ->
</xsd:sequence>
</xsd:complexType>
</xsd:element>
```

Sxemadagi har bir element odatiy xsd: prefiks bilan ko'rsatiladi, bu W3C XML sxemasi nomlari maydoni bilan xmlns deklaratsiyasi yordamida bog'lanadi: xsd = "http://www.w3.org/2000/lO/XMLSchema" (bu deklaratsiya sxemaning elementiga joylashtirilgan)) STAFF va NAME elementlari pastki qismlarni o'z ichiga oladi va ularni shunga o'xshash tarzda aniqlash mumkin. Pastki qism yoki atributlarga ega bo'lmagan elementlar oddiy tip turiga kiradi. Masalan, STAFFNO, DOB va SALARY elementlarini quyidagicha ko'rsatish mumkin:

```
<xsd:element name = "STAFFNO" type = "xsd:string"/>
<xsd:element name = "DOB11 type = "xsd:date"/>
<xsd:element name = "SALARY" type = "xsd:decimal"/>
```

Ushbu elementlar oldindan belgilangan W3C XML Sxema turlaridan foydalangan holda e'lon qilinadi, ya'ni satr, sana va o'nlik va xsd: prefiksi XML sxemasi lug'atiga a'zoligini ko'rsatish uchun kiritiladi. Bundan tashqari, har doim oxirgi pozitsiyani egallashi kerak bo'lgan `branchNo`, atributi quyidagicha e'lon qilinishi mumkin:

```
< xsd : attribute name = " branchNo " type =
" xsd : string " />
```

Kardinallik

XML sxemasi tili minOccurs (yozuvlarning eng kam soni) va maxOccurs (yozuvlarning maksimal soni) atributlaridan foydalangan holda elementning kardinallikligini namoyish etishga imkon beradi. Ixtiyoriy elementni ko'rsatish uchun minOccurs atributi 0 ga, va sodir bo'ladigan hodisalar soni cheklanmaganligini ko'rsatish uchun, maxOccurs atributi cheklanmagan qilib o'rnatilishi mumkin . Har qanday aniqlanmagan atributning standart qiymati 1 ga teng. Masalan, DOB elementi ixtiyoriy bo'lgani uchun, buni ko'rsatish uchun quyidagi qurilish ta'minlanishi mumkin:

```
<xsd:element name="DOB" type="xsd;date" minOccurs="0"/>
```

Va yaqin qarindoshlar nomlariga qadar, ro'yxatdan o'tishingiz uchun (Next-Of-Kin – NOK) dizaynga murojaat qilishingiz mumkin;

```
<xsd:element name="NOK" type="xsd:string" minOccurs="0"
naxOccurs="3"/>
```

Havolalar.

Yuqorida tavsiflangan usul (har bir aniqlangan element uchun ta'rifni ishlab chiqishni o'z ichiga olgan) nisbatan sodda bo'lishiga qaramay, ichki o'rnatilgan ta'riflarni chuqur joylashtirishga imkon bermaydi va natijada olingan sxema o'qish va saqlash uchun noqulay bo'lishi mumkin. Shuning uchun, sxemani yaratish uchun, ular ishlatiladigan elementni aniqlash sohasida bo'lgan elementlar va atributlarning deklaratsiyalariga havolalar yordamida ham foydalaniladi. Masalan, `STAFFNO` elementini quyidagicha belgilashingiz mumkin:

```
<xsd:element name="STAFFNO" type="xsd:string"/>
```

har safar `STAFFNO` elementiga duch kelganda, quyida ko'rsatilgandek, ushbu ta'rifdan foydalanamiz.

```
<xsd: element ref = "STAFFNO" />
```

Agar XML hujjatida elementga ko'plab havolalar mavjud bo'lsa, bu holda `STAFFNO` , unda havolalardan foydalanish uning ta'rifini faqat bitta joyda saqlashga imkon beradi va shu bilan sxemani saqlash qulayligini oshiradi.

Yangi turlarni aniqlash

XML sxemasi yangi ma'lumotlar turlarini deklaratsiyalash asosida atribut elementlarini yaratish uchun uchinchi mexanizmni ta'minlaydi. Bu holatda ishlatiladigan usul sinfni aniqlashga, keyin esa ob'ekt yaratishda foydalanishga o'xshaydi. Xususan, PCDATA elementlari yoki atributlari uchun oddiy turlarni, shuningdek elementlar uchun murakkab turlarni aniqlash mumkin. Yangi turlarga nomlar berilgan va ularning ta'riflari elementlar va atributlar deklaratsiyasidan tashqarida joylashtirilgan. Masalan, `STAFFNO` elementi uchun yangi oddiy turni quyidagicha aniqlash mumkin:

```
<xsd:simpleType name-l lSTAFFNOTYPE">
<xsd:restriction base="xsd:string">
<xsd:maxLength value="5"/>
</xsd:restriction>
</xsd:simpleType>
```

Ushbu yangi tur XML sxema nomlari maydonidan (ma'lumotlar atributi) cheklangan ma'lumot sifatida belgilanadi va maksimal 5 belgidan iborat bo'lishi kerakligini bildiradi (maxLengtn elementi faset deb ataladi). XML sxemasi spetsifikatsiyasi uzunligi, minLength, minlnclusive va maxlnclusive ni o'z ichiga olgan holda 15 tomonni taqdim etadi. Boshqa ikkita qulay jihatlar - bu naqsh va

ro'yxat. Naqsh elementi belgilangan qiymatni mos keladigan oddiy ifoda belgilaydi. Masalan, STAFFNO elementiga cheklov qo'yilishi mumkin, shunda uning qiymati ikkita katta harfdan iborat bo'lib, undan keyin birdan uchgacha raqamlar bo'lishi kerak (masalan, SG5, SG37, SG999). Ushbu talabni quyidagi sxema shablonidan foydalangan holda XML sxemasida taqdim etish mumkin:

```
<xsd:pattern value="[A-Z] {2} [0-9]{l,3}">
```

Hisoblash elementi oddiy turni farqlanadigan qiymatlar to'plamiga cheklash imkonini beradi. Misol uchun, pozitsiyalarni ma'lumotlarni o'z ichiga olgan element ustida, Manager , Supervisor yoki yordamchi u faqat cheklovlar qiymatlari bo'lishi mumkin. Bunday talabni sxemada quyidagi ro'yxatga olish ma'lumotlari yordamida taqdim etish mumkin:

```
<xsd:enumeration value="Manager"/>
<xsd:enumeration value="Supervisor"/>
<xsd:enumeration value="Assistant"/>
```

Guruhlar.

XML sxemasi tili elementlar guruhini va atributlar guruhining ta'riflarini sxemaga kiritishga imkon beradi. Guruh ma'lumot turi emas, balki konteyner sifatida ishlatiladigan va ma'lum elementlar yoki atributlar to'plamini o'z ichiga olgan qurilishdir. Masalan, kompaniya xodimlari to'g'risidagi ma'lumotlarni quyidagicha guruh sifatida taqdim etish mumkin:

```
<xsd:group name="STAFFTYPE">
    <xsd:sequence>
        <xsd:element name="STAFFNO"
        type="STAFFNOTYPE"/>
        <xsd:element name="POSITION"
        type="POSITIONTYPE"/>
        <xsd:element name="DOB" type="xsd:date"/>
        <xsd:element name="SALARY" type="xsd:decimal"/>
    </xsd:sequence>
</xsd:group>
```

Ushbu konstruktsiyani qayta ishlashdan so'ng, nomlangan STAFFTYPE guruhi yaratildi, bu elementlar ketma-ketligi (soddaligi uchun faqat ba'zi STAFF elementlari bu erda ko'rsatilgan). Sxemani aniqlashda siz STAFFLIST elementini quyidagicha nol yoki undan ko'p STAFFTYPE elementlar ketma-ketligi sifatida belgilangan guruhga havolasi bilan yaratishingiz mumkin :

```
<xsd:element name="STAFFLIST">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:group ref="STAFFTYPE"
                                minOccurs="0"
                                maxOccurs-"unbounded"/>
```

```
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

Choice va all kompozitsiya elementlari.

Yuqorida aytib o'tilganidek, ketma-ketlik kompozitor element turlaridan birining vakili. Kompozitsion elementlarning yana ikkita turi taqdim etiladi: tanlov va barchasi. Tanlash kompilyatori elementi bir nechta haqiqiy elementlar yoki elementlar guruhlari o'rtasidagi tanlovni aniqlaydi va kompilyator elementi barcha elementlarning tartibsiz to'plamini belgilaydi. Masalan, kompaniya xodimining ismi faqat bitta chiziq yoki ikkita chiziq (ism va familiya bilan) kombinatsiyasini ko'rsatadigan vaziyat quyidagi qurilish yordamida aks ettirilishi mumkin:

```
<xsd:group name="STAFFNAMETYPE">
    <xsd:choice>
    <xsd:element name="NAME" type="xsd:string"/>
        <xsd:sequence>
            <xsd:element name="FNAME" type="'xsd:
            string"/>
            <xsd:element name="LNAME"
            type="xsd:string"/>
        </xsd:sequence>
    </xsd:choice>
</xsd:group>
```

Ro'yxatlar va birlashuvlar.

Bo'shliqlar bilan ajratilgan narsalar ro'yxatini yaratish uchun ro'yxat elementidan foydalaning. Masalan, kompaniya xodimlari uchun xodimlar sonini o'z ichiga olgan ro'yxat quyidagicha tuzilishi mumkin:

```
<xsd:simpleType name="STAFFNOLIST">
    <xsd:list itemType=STAFFNOTYPE/>
</xsd: sitnpleType>
```

Quyida XML turida ushbu turni qo'llash misoli keltirilgan.

```
<STAFFNOLIST>"SG5" "SG37" "SG999"</STAFFNOLIST>
```

Endi, ushbu turdagi ro'yxatga asoslanib, cheklovning ba'zi bir rasmini ifodalaydigan yangi turni aniqlash mumkin, masalan, cheklangan ro'yxat e'lon qilinishi mumkin, elementlarning soni 10 bo'lgan, quyidagicha:

```
<xsd:simpleType name="STAFFNOLIST10">
    <xsd:restriction base="STAFFNOLIST">
        <xsd:Length value="10"/>
    </xsd: restriction
</xsd:simpleType>
```

Oddiy turlar (elementar deb ham ataladi) va ro'yxatlar ba'zi bir elementar tipning bir yoki bir nechta namunalaridan tashkil topgan elementlar yoki atributlarning qiymatlarini aniqlashga imkon beradi. Bundan farqli o'laroq, birlashma turi bir nechta elementar turlar yoki ro'yxatlar birlashmasidan tanlangan

bir xil turdagi bir yoki bir nechta holatlardan element yoki atributning qiymatlarini tanlashga imkon beradi. Buning uchun ishlatiladigan format yuqorida tavsiflangan tanlov deklaratsiyasiga o'xshaydi, shuning uchun uning batafsil tavsifi bu erda berilmaydi. W3C XML sxema hujjatiga qiziqqan o'quvchi murojaat qilishi mumkin. 2-Listingda ko'rsatilgan XML hujjati uchun XML sxemasi, 6 -Listingda ko'rsatilgan.

```
<?xml version= "1.011 encoding= "UTF-8"?>
<xsd:schema xmlns:xsd=
"http://www.w3.org/2000/10/XMLSchema">
 <! - STAFFLIST -> uchun guruh yarating
<xsd:group name="STAFFLISTGROUP">
    <xsd:element. name»"STAFFLIST">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:group ref="STAFFTYPE"
                                minOccurs="Q"
                                maxOccurs="unbounde
            d"/>
            </xsd:sequence>
        </xsd:complexType >
    </xsd:element>
</xsd:group>
 <! - bir yaratish uslubi uchun element STAFFNO ->
<xsd:simpleType name="STAFFNQTYPE">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="5"/>
        <xsd:pattern value="[A-Z]{2} [0-9]{1,3}">
    </xsd:restriction>
</xsd:simpleType>
<!-- branchNo elementi uchun guruh yarating -->
<xsd:sirapleType name="BRANCHNOTYPE">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="4"/>
        <xsd:pattern value="[A-Z] [0-9]{3}">
    </xsd:restriction>
</xsd:simpleType>
<]-- POSITION elementi uchun guruh yarating ->
<xsd:simpleType narae="POSITIONTYPE">
    <xsd:restriction base="xsd:atring">
        <xsd:enumeration value="Manager"/>
        <xsd:enumeration value="Supervisor"/>
        <xsd:enumeration value="Assistant"/>
    </xsd: restriction
</xsd:simpleType>
<!-- STAFF elementi uchun guruh yarating -->
```

```
<xsd:group name="STAFFTYPE">
    <xsd:element name="STAFF" >
        <xsd:complexType>
            <xsd:sequence»
                <xsd:element name="STAFFNO"
                type="STAFFNOTYPE"/>
                <xsd:element name="NAME">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="FNAME"
                        type="xsd:string"/>
                        <xsd:element name="LNAME"
                        type="xsd:string"/>
                    </xsd:sequence>
                </xsd:complexType>
                </xsd:element>
                <xsd:element name="POSITION"
                type="POSITIONTYPE"/>
                <xsd:element name="DOB"
                type="xsd:date"/>
                <xsd:element name="SALARY"
                type="xsd:decimal"/>
                <xsd:attribute name="branchNo"
                type="BRANCHNOTYPE"/>
            <xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:group>
</xsd:schema>
```

6- Listing. 2- Listingdagi XML hujjati uchun XM L sxemasi

Cheklovlar

Oldingi bo'limlarda XML hujjatida foydalanilgan ma'lumotlarni tekshirish uchun qirralardan foydalanishning ba'zi misollari ko'rib chiqilgan. XML sxemasi spetsifikatsiyasi, shuningdek, XPath vositalari asosida ma'lumotlar qiymatlarining ma'lum bir doirasi bo'yicha hurmat qilinishi kerak bo'lgan noyoblik cheklovlarini va bog'langan havolaviy cheklovlarni aniqlash vositasini ham beradi. Ushbu bo'limda ikki xil cheklovlar ko'rib chiqiladi: noyoblik va asosiy cheklovlar.

O'ziga xoslik cheklovlari.

Noyob cheklovni aniqlash uchun noyob bo'lishi kerak bo'lgan elementlar yoki atributlarni belgilaydigan noyob element ko'rsatilishi kerak. Masalan, siz yagona qurilish cheklovini quyidagi xodimlardan foydalangan holda kompaniyaning xodimining familiyasi va tug'ilgan sanasidan (tug'ilgan sana - DOB) iborat ma'lumotlar bilan aniqlashingiz mumkin:

```
<xsd:unique name="NAMEDOBUNIQUE">
```

```
    <xsd:selector xpath="STAFF"/>
    <xsd:field xpath="NAME/LNAME"/>
    <xsd:field xpath="DOB"/>
</xsd:unique>
```

Sxemadagi noyob elementning joylashuvi sizga ushbu cheklov qo'llaniladigan kontekst tugunini aniqlashga imkon beradi.

Bunday holda, cheklovni ko'rsatuvchi tavsiflovchi `STAFF` elementiga ergashadi;

Shunday qilib, ushbu cheklash faqat `STAFF` elementi nuqtai nazaridan noyob bo'lishi kerakligi ko'rsatilgan, xuddi munosabatdagi MBBTda munosabatlarni cheklash aniqlangandek. Quyidagi uchta elementda aniqlangan XPath iboralari kontekst tuguniga nisbatan. Selektor elementi bilan birinchi XPath ifodasi noyoblik chekloviga duch keladigan elementni belgilaydi (bu holda, `STAFF`), keyingi ikkita maydon elementlari noyobligi tekshirilishi kerak bo'lgan tugunlarni bildiradi.

Kalit bilan cheklovlar o’rnatish.

Kalit cheklovi noyob cheklovga o'xshaydi, bundan tashqari u bilan belgilangan qiymat bo'sh bo'lmasligi kerak. Shuningdek, u sizga kalitga murojaat qilish imkonini beradi. Quyidagi misol `STAFFNO` tugunida o'rnatilgan cheklovlarni o'rnatgan :

```
<xsd:key name="STAFFNOISKEY">
    <xsd:selector xpath="STAFF"/>
    <xsd:field xpath="STAFFNO"/>
</xsd:key>
```

Cheklovning yana bir turi ko'rsatilgan kalitlarning qiymatlariga bog'lanishni cheklashga imkon beradi. Masalan, branchNo atributi oxir-oqibat kompaniyaning biron bir filialini ko'rsatishga mo'ljallangan. Agar tegishli element BRANCHNOISKEY tugmachasi bilan yaratilgan deb faraz qilsak, unda ushbu atributning qiymati kalit qiymatlari bilan quyidagicha cheklanishi mumkin:

```
<xsd:keyref name="BRANCHNOREF" refer "BRANCHNOISKEY">
    <xsd:selector xpath="STAFF"/>
    <xsd:field xpath="@branchNo"/>
</xsd:keyref>
```

17.7 (RDF) Resurs ta'rifi doirasi

Shubhasiz, XML sxemasi spetsifikatsiyasi DTD ta'riflariga qaraganda XML hujjat tarkibini aniqlash uchun yanada to'liq va qat'iy usulni ta'minlaydi. Ammo bu hali semantik moslashuvning zarur darajasini qo'llab-quvvatlamaydi. Masalan, agar ikkita dastur XML-dan foydalanib ma'lumot almashishi kerak bo'lsa, unda ushbu jarayonda ishlatiladigan hujjatlarning maqsadi va tushunarli ma'nosi ular yaratadigan ma'lumotlar tuzilishiga mos kelishi kerak. Ammo buning uchun ma'lum bir dastur uchun zarur bo'lgan ma'lumotlarning tavsifi bilan domen modelini yaratish kerak. Bu qanday ma'lumotni bir ilovadan boshqasiga yo'naltirish va teskari

yo'nalishda o'tkazish kerakligini aniq belgilash imkonini beradi. Bunday modelni ob'ektlar yoki munosabatlar nuqtai nazaridan tavsiflash odatiy holdir (masalan, oldingi boblarda bu maqsadda UML tili ishlatilgan). Va XML sxemasi faqat hujjatning sintaktik tuzilishini tavsiflaganligi sababli, bir xil domen modelini XML sxemasi sifatida har xil yo'llar bilan ifodalash mumkin. Shuning uchun, domen modeli va ma'lum bir sxema o'rtasidagi to'g'ridan-to'g'ri yozishmalarni aniqlash mumkin emas. Ikki emas bo'lsa, bu muammo yanada murakkablashadi, ammo ma'lumot almashishda bir nechta dastur ishtirok etishi kerak. Bunday holda, bir nechta XML sxemalari o'rtasida yozishmalarni o'rnatish kifoya qilmaydi, chunki bu erda bitta sintaktik tuzilmani boshqasiga o'tkazish emas, balki ob'ektlar va bir necha predmet sohalariga tegishli munosabatlar o'rtasida yozishmalar o'rnatish. Shunday qilib, yuqorida tavsiflangan muammoni hal qilish uchun quyidagi uchta bosqichdan o'tish kerak.

1. XML sxemalaridan asl domen modellarini tiklash;
2. mavzu sohalari modellari ob'ektlari o'rtasidagi muvofiqlikni aniqlash;
3. Masalan, XSLT tilidan foydalanib, XML hujjatlarini o'zgartirish mexanizmlarini aniqlash.

Amalda, bu bosqichlar ba'zan juda qiyinlashadi, shuning uchun XML tili faqatgina tarkibiy model allaqachon ma'lum bo'lgan dasturlar o'rtasida ma'lumot almashish uchun juda mos ekanligi ayon bo'lishi mumkin, ammo tobora ko'proq yangi dasturlar ma'lumotlar almashinuviga qo'shilsa. Shu sababli , qiziqish mavzularini tasvirlashga imkon beradigan boshqa umumiy tan olingan til talab qilinadi. W3C shafe'ligida ishlab chiqilgan ***Resurslarni tavsiflash asoslari (RDF)*** strukturalangan metadata kodlash, almashish va qayta ishlatishni ta'minlaydigan axborot muhiti. Ushbu infratuzilma hujjatlarni semantikasi, sintaksisi va tuzilishi bo'yicha umumiy qabul qilingan konventsiyalarni yaratishga imkon beruvchi dizayn mexanizmlari yordamida turli xil ilovalarning metadata bilan o'zaro ishlashini ta'minlaydi. RDF infratuzilmasi har bir fan sohasining semantikasini aniqlamaydi, ammo zarurat tug'ilsa, bunday mavzular uchun metadata elementlarini yaratish qobiliyatini ta'minlaydi. RDF doirasi XML-dan metadata almashish va qayta ishlash uchun umumiy sintaksis sifatida foydalanadi. XML til vositalari yordamida RDF ma'lumotlar modellari yaratilib, ularning tuzilishi semantikaning tavsifini beradi, shuningdek yagona tavsifni yaratishga va standart metadata almashinuvini ta'minlaydi.

RDF ma'lumotlar modeli.

Eng oddiy RDF ma'lumotlar modeli uchta ob'ektdan iborat.

Resurs. Resurs bu URI bo'lishi mumkin bo'lgan hamma narsa, masalan, web-sahifa, web-sahifalar qatori yoki hatto XML elementi kabi web-sahifaning bir qismi.

Mulk. Bu manbani tavsiflovchi o'ziga xos atribut. Masalan, Muallif atributidan aniq XML hujjatini tayyorlagan kishini tasvirlash uchun foydalanish mumkin.

Operator . Bu manba, mulk va qiymat kombinatsiyasidan iborat bo'lgan qurilish. RDF operatorining tarkibiy qismlari odatda "sub'ekt", "predikat" va "ob'ekt" deb nomlanadi. Misol uchun, "muallifi bayonot

http : // www. Dreamhorae. Co. Uk / xodimlari _ ro'yxati.

XML deb Jon Oq " bir operator hisoblanadi (Hujjat muallifi ... Jon Oq emas). Ikkinchisi RDF infratuzilmasida quyidagicha ifodalanishi mumkin:

```
<rdf:RDF     xmlns:rdf="http://www.w3.org/1999/02/22-
rdf-syntax-nstf"
       xmlns:s="http://www.dreamhome.co.uk/schema/">
<rdf:Description
     about="http://www.dreamhome.co.uk/staff_list.xml">
     <s:Author>John White</s:Author>
</rdf:Description>
</rdf:RDF>
```

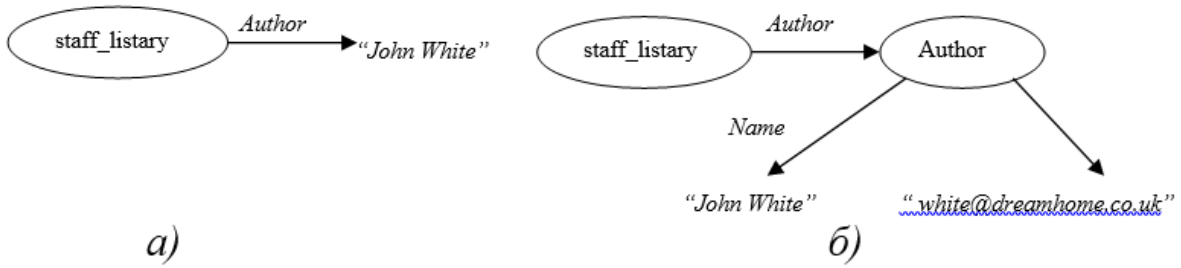
Ushbu ma'lumot shuningdek sxematik tarzda, sekstda ko'rsatilgan yo'naltirilgan yorliqli grafik yordamida taqdim etilishi mumkin (17.5 a –rasm). Agar ma'lumotni muallifning tavsifi bilan ta'minlash talab qilinsa, u holda 17.5, b - rasmda ko'rsatilganidek, manba sifatida ma'lumotlar manba sifatida modellashtirilishi mumkin.

Bunday holda, metadata tavsiflash uchun quyidagi XML bo'limi ishlatilishi mumkin:

```
<rdf:RDF     xmlns:rdf="http://www.w3.org/1999/02/22-
rdf-syntax-nstt"
       xmlns:s="http://www.dreamhome.co.uk/schema/">
<rdf:Description
         about="http://www,dreamhome.co.uk/staff_list.xm
         l">
         <s:Author
     rdf:resource="http://www.dreamhome.co.uk/Author__00
     1"/>
     </rdf:Description>
     <rdf:Description
     about="http://www.dreamhome.co.uk/Author _001">
         <s:Name>John White</s:Name>
         <s:e-mail>white®dreamhome.co.uk</s:e-mail>
     </rdf:Description> </rdf:RDF>
```

RDF sxemasi

RDF sxemasi sizga sxemaga kiritilgan sinflar, shu jumladan ularning xususiyatlari (atributlari) va manbalar (sinflar) o'rtasidagi munosabatlar to'g'risida ma'lumot berishga imkon beradi. Boshqacha qilib aytganda, RDF sxemasini aniqlash mexanizmi XML sxemasiga o'xshash RDF modellarida foydalanish uchun asosiy turdagi tizimni ta'minlaydi. RDF sxemasi ma'lum dasturlarning sxemalarini aniqlash uchun ishlatiladigan manbalar va xususiyatlarni (masalan, rdf s: sinf va rdf s: subClassOf) aniqlashga imkon beradi.



17.5-rasm. Taqdimot formatlariga misollar: a) muallif haqida ma'lumotni mulk sifatida taqdim etish; b) muallif haqida ma'lumotni manba rasmida taqdim etish.

RDF sxemalari, shuningdek, ba'zi cheklovlarni belgilash, xususan, kerakli kardinallikni belgilash va sinf namunalarining haqiqiy xususiyatlarini aniqlash imkoniyatini beradi. RDF sxemasini aniqlash uchun bilimlarni namoyish qilish sohasidagi g'oyalar (masalan, semantik tarmoqlar va predikatlar mantig'i) ta'siri ostida yaratilgan deklarativ til, shuningdek ma'lumotlar bazasi sxemalarini namoyish qilish uchun modellar, masalan, NIAM kabi ikkilik relatsion modellar va grafikaga asoslangan ma'lumotlar modellari qo'llaniladi. RDF infratuzilmasi va RDF sxemasi haqida to'liq ma'lumot ushbu kitobning doirasiga kirmaydi va qiziqqan o'quvchi qo'shimcha ma'lumot olish uchun W3C hujjatlariga murojaat qilishi mumkin.

XML so'rovlari tillari.

So'rovlar tillari ma'lumotlar to'plash, o'zgartirish va ma'lumotlarni birlashtirish kabi ma'lumotlar bazalarining ishlashini ta'minlashning yaxshi o'rganilgan muammolarini hal qilishga imkon beradi. Avvalgi boblarda tavsiflangan ma'lumotlar bazasining ikkita standart so'rovlari (xususan SQL va OQL) ushbu ma'lumotlarning tartibsiz tuzilishi tufayli XML ma'lumotlari bilan ishlashda bevosita ishlatilishi mumkin emas. XML hujjatlarini, shu jumladan Microsoft-ning XML-QL, UnQL va XQL-larini qayta ishlash uchun bir oz tuzilgan so'rov tillaridan foydalanish mumkin. Ushbu tillarda, yo'l belgisi deb ataladigan narsa, ichiga o'rnatilgan XML tuzilishini boshqarish uchun ishlatiladi. Masalan, XML-QL-da, tanlangan hujjatning qismini tasvirlash uchun XML-ga o'xshash ichki tuzilish ishlatiladi va natijada kerakli XML tuzilishi yaratiladi. Masalan, 30,000 GBP dan yuqori ish haqi bo'lgan xodimlarning ismlarini olish uchun quyidagi so'rovdan foydalanish mumkin:

```
WHERE <STAFF>
    <SALARY> $S </SALARY>
    <NAMExFNAME> $F </FNAME> <LNAME> $L </LNAMEx/NAME>
    </STAFF> IN "http://www.dreamhome.co.uk/staff.xml"
    $S > 30000
CONSTRUCT <LNAME> $L </LNAME>
```

XML Query Working Group so'rovi bo'yicha ishchi guruhi

Yaqinda W 3 C XML hujjatlari ma'lumotlari modelini tayyorlash, ushbu model uchun so'rov operatorlari to'plamini va ushbu so'rov operatorlariga asoslangan so'rov tilini aniqlash uchun XML so'rovi

bo'yicha ishchi guruhini tuzdi. So'rovlar bitta hujjat yoki muayyan hujjatlar to'plamida amalga oshiriladi va sizga hujjatlarning tarkibi va tuzilishi hisobga olinadigan shartlarga javob beradigan to'liq hujjatlarni yoki hujjatlarning pastki qismlarini tanlash imkonini beradi. So'rovlar, shuningdek, olingan ma'lumotlar asosida yangi hujjatlar yaratishga imkon beradi. Oxir oqibat, ma'lumotlar bazasi sifatida XML hujjatlari to'plamlariga kirishga imkon beradigan til yaratish rejalashtirilmoqda. Ushbu kitobni yozish paytida ishchi guruh quyidagi to'rtta hujjatni tayyorladi:

- XML Query Requirements (talabiga talablariga XML);
- XML Query Data Model (so'rovlar ma'lumotlar modeli XML);
- XML Query Algebra so'rovlari algebra;
- XQuery (XML uchun so'rovlar tili).

XML so'roviga ***talablar*** hujjati vazifalarni, odatiy dastur stsenariylarini va so'rov ma'lumotlari modeliga, so'rovlar algebrasiga va W3C XML so'rov tiliga bo'lgan ***talablarni*** belgilaydi. Ushbu talablar quyidagilardan iborat:

1. so'rovlar tili deklarativ bo'lishi kerak va u ishlatilgan protokollardan qat'i nazar aniqlanishi kerak;
2. ma'lumotlar modeli nafaqat XML 1.0 belgilarining ma'lumotlarini, balki oddiy va murakkab XML sxema spetsifikatsiyalarini ham ifodalashi kerak; u shuningdek hujjat ichida va tashqarisida belgilangan havolalarni qo'llab-quvvatlashni o'z ichiga olishi kerak;
3. so'rovlarni bajarish qobiliyati hujjat sxemasi mavjud yoki yo'qligidan qat'i nazar taqdim etilishi kerak;
4. til kollektsiyalarda aniqlangan umumiylik va mavjudlikni o'lchash vositalarini qo'llab-quvvatlashi, bo'sh qiymatlarni yig'ish, saralash va qayta ishlashni ta'minlashi kerak, shuningdek, havolalar hujjatning o'zida yoki bitta hujjatdan ikkinchisiga o'tishni ta'minlaydi.

Ilova sifatida XML so'roviga talablar hujjati kutilgan natijalarga ega XML so'rovlari ko'rinishidagi testlar to'plamini taqdim etadi.

XML so'rov ma'lumotlari modeli.

XML so'rovi ma'lumotlar modelini tavsiflovchi XML so'rovi ma'lumotlari modeli hujjati har qanday XML so'rov protsessoriga kirishda mavjud bo'lgan ma'lumotlarni belgilaydi. Ushbu ma'lumotlar modeli XML ma'lumot to'plamiga asoslangan bo'lib, rasmiy rasmiy XML hujjatida mavjud bo'lgan ma'lumotlarning tavsifini quyidagi yangi vositalar bilan ta'minlaydi:

- XML sxema turlarini qo'llab-quvvatlash;
- hujjatlar to'plamlarini, shuningdek sodda va murakkab ma'nolar to'plamlarini taqdim etish;
- havolalarni topshirish.

XML so'rov ma'lumotlari modeli yorliqli tugunlar va daraxt parchalari konstruktorlaridan foydalanadigan vakillikdir. U XML mos yozuvlar qiymatlarini (masalan, IDREF, XPointer va URI) soddalashtirish uchun tugun identifikatori

tushunchasidan foydalanadi. Ma'lumotlar modelining har bir misoli bitta yoki bir nechta to'liq hujjatlarni yoki hujjatlarning bir qismini ifodalaydi va buyurtma yoki tartibsiz bo'lishi mumkin.

Ma'lumotlar modelining asosiy tarkibiy qismi bu tugun bo'lib, u hujjat, element, qiymat, atribut, ism maydoni, ishlov berish buyrug'i, sharh yoki ma'lumot bo'lagi bo'lishi mumkin. Butun XML hujjati DocNode tugun sifatida taqdim etiladi. Hujjatning bir qismi element tugunlari (EletnNode), qiymat tugunlari (ValueNode), ishlov berish yo'riqnomasi tugmachasi (PINode) yoki sharh tugunlari (CommentNode) bilan ifodalangan hujjat substrati. Ma'lumotlar modeli, shuningdek, ma'lumotlar modelining muayyan namunasidagi tugun identifikatorlarini tekshirish va bog'lash uchun tugun ma'lumotlarini ishlatadi. Model, tugunga va Derefga ulanishni ko'rsatadigan tugmachani yaratish uchun Ref funktsiyalarini qo'llab-quvvatlaydi. Quyida XML so'rov ma'lumotlar modelining asosiy ob'ektlari haqida qisqacha ma'lumot berilgan.

Hujjat.

DocNode hujjatida URI mos yozuvlar qiymatini va bo'sh bo'lmagan buyurtma qilingan o'rmon2 ni parametr sifatida qabul qiladigan docNode konstruktori mavjud:

```
docNode :(uriReference, [EleraNode | PINode |
CommentNode])->DocNode
```

Hujjat quyidagi uchta kirish funktsiyalariga ega:

```
uri     :DocNode->uriReference
children     :DocNode->[ElemNode ( PINode |
CoramentNode]
root    :DocNode->E1emNode
```

URI funktsiyasi DocNode-ning URI mos yozuvlar qiymatini qaytaradi, children funktsiyasi DocNode bolalarining buyurtma qilingan o'rmonini qaytaradi va root funktsiyasi ota-ona DocNode-ning bolalarining tartiblangan o'rmonidagi noyob EleraNode-ni qaytaradi.

Elementlar

ElemNode tugunida ikkita ortiqcha yuklangan elemNode konstruktorlari mavjud. Birinchi konstruktor tavsiflovchi qiymatini (malakali QNameValue nomi), NSNode nomlar maydonining tartiblanmagan o'rmonini, AttrNode-ning tartibga solinmagan atribut o'rmonini, bolalar tugunlarining tartibga solinmagan o'rmonini va parametrlar sifatida tugun turiga (Def_Type turining misoli) havolasini oladi:

```
elemNode :(QNameValue, {NSNode} , {AttrNode} , [ElemNode
| ValueNode
       J PINode | CommentNode | InfoItemNode |
RefNode],
       Def_Type}->ElemNode
```

Ikkinchi konstruktor o'z parametrlari sifatida NSNode nomlar maydonining tartiblanmagan to'plamini, elementning atributlari va bolalar tugunlarini, shuningdek tugun turiga ulanishni o'z ichiga olgan tartibga solingan tugunni oladi. Tugunlarning tartibga solingan o'rmonida atributlar boshqa barcha

tugunlardan oldin bo'lishi kerak:

```
elemNode :( QNameValue , { NSNode },
[ AttrNode | ElemNode | ValueNode | PINode | CommentNod
e | InfoItemNode j RefNode ], Def _ Type ) ->
E1emNode
```

Ob'ektlar uchun quyidagi ettita kirish funktsiyalari mavjud:

```
name     :ElemNode -> QNameValue
children     :ElemNode -> [ElemNode | ValueNode |
PINode |
        CommentNode j     InfoItemNode | RefNode]
attributes   :ElemNode -> {AttrNode}
namespaces   :ElemNode -> {NSNode}
type     :ElemNode -> DefJType
parent   :ElemNode -> ElemNode | DocNode | NaR
nodes    :ElemNode -> [AttrNode | ElemNode | ValueNode |
PINode | CommentNode | InfoItemNode j RefNode]
```

Birinchi beshta kirish funktsiyalari ElemNode- ning tarkibiy qismlarini qaytaradi;

Qarindoshlik funktsiyasi ElemNode tugunining bitta asosiy tuguniga havolani qaytaradi. Agar ElemNode hujjatning ildiz tugunidir, bu funktsiya mos keladigan DocNode-ga havolani qaytaradi va agar u mavjud bo'lmasa, NaR-ni qaytaradi (Yo'naltiruvchi noto'g'ri havola). Tugunlarga kirish funktsiyasi element atributlarini o'z ichiga olgan tartiblangan o'rmonni va uning barcha elementlarini o'z ichiga oladi.

Atributlar

AttrNode tugunida atributning nomi va qiymatini parametr sifatida qabul qiladigan attrNode konstruktori mavjud:

```
attrNode: (QNameValue, ValueNode) -> AttrNode
```

AttrNode tuguniga quyidagi uchta kirish funktsiyalari taqdim etiladi:

```
name   :AttrNode -> QNameValue
value  :AttrNode -> ValueNode
parent   :AttrNode -> EleraNode | NaR
```

Ism maydonlari.

NSNode-da nsNode konstruktori mavjud bo'lib, u nom bo'shligi prefiksini (bo'sh bo'lishi mumkin) va e'lon qilingan nomlar maydonining mutlaq URI (bu ham bo'sh bo'lishi mumkin) parametr sifatida oladi:

```
nsNode: (string) null, uriReference | null) -> NSNode
```

NSNode uchun quyidagi uchta kirish funktsiyalari taqdim etiladi:

```
prefix   :NSNode -> string | null
uri  :NSNode -> uriReference | null
parent   :NSNode -> ElemNode
```

Yo'naltiruvchi tugun.

Ushbu ma'lumotlar modeli ma'lumot modelining ma'lum bir nusxasida tugun identifikatorlarini kapsülleme mexanizmi sifatida RefNode mos yozuvlar tugunlaridan foydalanish imkoniyatini ta'minlaydi. RefNode tugunida parametr sifatida tugun tugunini oladigan refNode konstruktori mavjud:

```
refNode :Node -> RefNode
```

RefNode tuguniga quyidagi ikkita kirish funktsiyalari taqdim etiladi:

```
deref    :RefNode -> Node
parent   :RefNode -> ElemNode | NaR
```

Deref accessor funktsiyasi mos keladigan tugun bilan ko'rsatilgan tugunni qaytaradi; qarindoshlik orqali kirish funktsiyasi RefNode bolalar tugunining noyob ota tugunini qaytaradi va agar ota-ona tugmasi mavjud bo'lmasa, NaR qiymatini qaytaradi.

Qadriyatlar.

ValueNode tugunlari bu 14 xil turdagi qiymatlarning ajralish birlashmasidir:

```
ValueNode    :StringValue | BoolValue | FloatValue |
DoubleValue j DecimalValue | TimeDurValue j
RecurDurValue | BinaryValue | URIRefValue j IDValue |
IDREFValue | QNameValue | ENTITYValue | NOTATIONValue
```

Ma'lumotlar modeli 14 XML sxemasi ma'lumotlari turlarining har biri uchun mos keladigan konstruktorni taqdim etadi, masalan, quyidagi konstruktorlar:

```
stringValue  :(string, Def_string, [InfoItemNode]) ->
StringValue
decimalValue :(decimal, Def_deciraal) -> DecimalValue
urirefValue  :(uriReference, Def_uriReference) ->
URIRefValue
qnameValue   :(uriReference j null, string, NCName,
Def_QName) -> QNameValue
```

ValueNode tugunida turli xil kirish funktsiyalari mavjud, masalan, quyidagilar:

```
turi : ValueNode -> Def_ST, bu erda ST mos keladigan
oddiy tur;
string : ValueNode -> StringValue, bu
satr namoyishini qaytaradi
```

Bundan tashqari, isSTValue 14- ta kirish funktsiyalari mavjud, bu erda ST oddiy turni bildiradi. Agar ValueNode tugunining belgilangan turi bo'lsa, ushbu funktsiyalar haqiqiy holatga qaytadi, masalan:

```
isStringValue: ValueNode -> boolean
```

17.8 XML Query algebra so'rovlari

W3C Query Working Group ishchi guruhi SQL va OQL kabi tillardan foydalanish tajribasidan foydalanadigan XML so'rovlari algebrasini taklif qildi. Ushbu algebra XML sxemasi tuzilmalarining mohiyatini aks ettiruvchi oddiy turdagi tizimdan foydalanadi. Bu sizga tilda statik turli ta'riflarni ishlatishga imkon beradi, shuningdek, so'rovlarni keyingi optim allashtirishni ta'minlaydi. Ushbu bo'limda, taklif qilingan algebra operatorlarini tasvirlash uchun biz 2- Listingda XML hujjatidan foydalanamiz va shuningdek 6- Listingda ko'rsatilgan XML sxemasining soddalashtirilgan versiyasidan foydalanamiz, ammo to'liqligi uchun sxemada qo'shimcha ravishda NOK elementi bor, deb taxmin qilinadi. ketma-ket uch marta hujjatlash. Ushbu ma'lumotlar va sxemani 7- Listingda ko'rsatilgandek ushbu algebradan foydalanib namoyish etish mumkin.

```
type STAFFLISTGROUP =
    STAFFLIST [ STAFFTYPE (O, *}]
type STAFFTYPE =
    STAFF [
        STAFFNO [String],
        SALARY [Decimal],
        NOK [String] {0, *}
        ©ЬгапсЬКо [String]
        ]
let STAFFLISTO : STAFFLISTGROUP =
    STAFFLIST [
    STAFF [
        STAFFNO ["SL21"],
        SALARY [30000] ,
        NOK ["Mrs Mary White1 1],
        NOK ["Mr Paul White'1],
        fflbranchNo ["B005"]
    ],
    STAFF [
        STAFFNO ["SG37"],
        SALARY [12000],
        NOK ["Mr John Beech"],
        fflbranchNo ["BOC3"]
    ]
]
```

7-Listing. Query Algebra XML Query Algebra yordamida taqdim etiladigan hujjat va sxema.

Ushbu ro'yxatda ko'rsatilgan algebraik modelda ikkita turdagi (`STAFFLISTGROUP` va `STAFFTYPE`) va bitta global o'zgaruvchi (`STAFFLISTO`) e'lon qilingan. `STAFFLISTGROUP` turi bitta `STAFFLIST` elementini anglatadi, unda nol yoki undan ko'p `STAFFTYPE` elementlardan iborat o'rmon mavjud. O'rmon - bu atributlar va elementlarning ketma-ketligi, ularning

soni noldan yoki undan ko'p bo'lishi mumkin. Element `STAFFTYPE` bir element ifodalaydi `STAFFTYPE` ikki elementlar (o'z ichiga oladi, `STAFFNO` va `SALARY` va bir atribut (-), ular nol yoki undan ko'p ob'ektlar NOK (eng yaqin qarindoshi Next-Of-Kin) tomonidan ta'qib qilingan `branchNo`). `STAFFLISTO` o'zgaruvchisi XML tom ma'nosi bilan bog'liq (`STAFFLIST` elementi ikkita `STAFF` elementi).

Quyida tavsiflangan algebraik qurilishda qo'llanilishi mumkin bo'lgan operatsiyalar tasvirlangan.

Proektsiyalash.

Quyidagi proektsion operatsiya `STAFF` elementlarida mavjud bo'lgan `STAFFLISTO` global o'zgaruvchisiga tegishli bo'lgan barcha NOK elementlarini qaytaradi :

```
STAFFLISTO/STAFF/NOK: NOK [String] { 0, *}
             ==>  NOK ["Mrs Mary White"],
                  NOK ["Mr Paul White1 1],
                  NOK ["Mr John Beech"]
```

Ushbu operatorda avval proektsion operatsiya ko'rsatiladi, keyin yo'g'on ichakdan keyin qaytarilgan natijaning turi ko'rsatiladi (bu holda, NOK elementi String turidagi nol va undan ko'p sonli yozuvlar soniga ega), shundan so'ng operatsiya natijasi strelkaga tushadi (==>).

Elementlar yoki atributlarning haqiqiy qiymatlariga kirish uchun ma'lumotlar () kalit so'zidan foydalaning, masalan:

```
STAFFLISTO/STAFF/NOK/data():  String { 0, *}
                   ⇨ "Mrs Mary White",
                     "Mr Paul White»,
                     "Mr John Beech"
```

Qaytarilish

Yuqorida aytib o'tilganidek, muhim operatsiyalardan biri hujjatni keyinchalik qayta qurish uchun hujjat elementlarini davriy qayta ishlashdir. Ko'rib chiqilayotgan algebraik modelda bunday maqsadga iteratsiya yordamida erishish mumkin. Masalan, quyidagi operatsiya sizga faqat `STAFFNO` va NOK elementlarini o'z ichiga olgan tuzilmani olishga imkon beradi (asl XML hujjatining teskari tartibida):

```
for S in STAFFLISTO/STAFF do
        STAFF [S/NOK, S/STAFFNO] : STAFF [
         NOK [String] { ! , * } ,
         STAFFNO [String]
         ]{o , *}
    ==> STAFF [
             NOK ["Mrs Mary White"],
             NOK ["Mr Paul White"],
             STAFFNO ["SL21"]
             ] ,
```

```
        STAFF [
             NOK ["Mr John Beech"],
             STAFFNO ["SG37")
]
```

Bu ifoda sizga `STAFFLISTO` o'zgaruvchisiga tegishli har bir `STAFF` elementini aylanib chiqish va s o'zgaruvchini har bir elementga bog'lash imkonini beradi. Har bir bog'liq elementga ichki ifoda qo'llaniladi va yangi `STAFF` elementi NOK elementlarini, so'ngra `STAFFNO` elementini o'z ichiga oladi.

Namuna olish.

Belgilangan predikatga mos keladigan qiymatlarni tanlash uchun WHERE so'zidan foydalanish mumkin. Misol uchun, bu siz o'zgaruvchan va 20 dan ortiq 000 funt sterling uchun ish haqining xususiyati STAFFLISTO qiymati Xodimlar barcha elementlarni olish va imkon beradi, elementlar STAFFNO o'z ichiga olgan, undan keyin, mutaxassis, yangi elementlarni hosil qilgan :

```
for S in STAFFLISTO/STAFF do
where S/SALARY/data() > 20000 do
       STAFF [S/STAFFNO, S/SALARY] : STAFF [
         STAFFNO [String],
         SALARY [Decimal]
         ]{ o , * }
    ==>  STAFF [
         STAFFNO ["SL21"],
         SALARY [30000]
]
```

Odatda, WHERE iborasi IF ... ga aylantiriladi ... ELSE rasmi (bo'sh ELSE bilan). A for if boshqa takrorlanadigan elementlardan ma'lumotlarni tanlash uchun joylashtirilishi mumkin. Masalan, quyidagi ibora NOK elementlari orasida qanday qidirishni ko'rsatadi:

```
for S in STAFFLISTO/STAFF do
    for N in S/NOK/data() do
    where N = "Mrs Mary White" do
             STAFF [S/STAFPNO] : STAFF [STAFFNO [String]
        ] { 0, *}
        ==>  STAFF [STAFFNO ["SL21"] ]
```

Ifodalash uchun qayerda ishlatilishi mumkin. O'qish qobiliyatini yaxshilash uchun mahalliy o'zgaruvchilardan foydalanish mumkin, masalan, mahalliy o'zgaruvchidan foydalanib, yuqoridagi ifoda quyidagi rasmga o'tkaziladi:

```
for S in STAFFLISTO/STAFF do
    let MrsWhite = (for N in S/NOK/data() do
        where N = "Mrs Mary White" do)
    where MrsWhite do
        STAFF [S/STAFFNO]
```

Ulanish.

Ko'rib chiqilayotgan algebraik model bir yoki bir nechta hujjatlardan olingan qiymatlarni birlashtirishga imkon beradi. Ushbu imkoniyatni namoyish qilish uchun, ma'lumotlar bazasida kompaniya xodimlariga to'lanadigan bonuslar to'g'risidagi ma'lumotlarni o'z ichiga olgan ikkinchi hujjat bor deylik.

```
type BONUSLIST =
    BONUSES [
        STAFF [
            STAFFNO [String] ,
            " BONUS [Decimal]
let BONUSLISTO: BONUSLIST -
    BONUSES [
        STAFF [
            STAFFNO ["SG37"] ,
            SALARY [1200]
        ]
        STAFF [
            STAFFNO ["SL21"] ,
            BONUS [3000]
        ]
    ]
```

Ushbu ikkita ma'lumot manbalari (`STAFFLISTO` va `BONUSLISTO`) bir-biriga `STAFFNO` qiymatlarini hisobga olgan holda ulanishi mumkin:

```
for S in STAFFLISTO/STAFF do
    for B in BOMJSLISTO/STAFF do
        where S/STAFFNO = B/STAFFNO do
            STAFF [S/STAFFNO, S/SALARY, B/BONUS] :
                STAFF [
                    STAFFNO t String] , SALARY
                [Decimal] , BONUS
                    [Decimal]
                ] ( 0, *}
              ==> STAFF [STAFFNO ["SL21"], SALARY
            [30000], BONUS [3000]
                ]
                STAFF [
                    STAFFNO ["SG37"], SALARY [12000],
                BONUS [1200]
                ]
```

Hozirgi vaqtda natijalardagi ma'lumotlar tartibini ifoda qilish uchun eng yaqinni aniqlaydi, ammo XML Query Algebra rasmiy W3C tavsiyasi sifatida qabul qilingandan keyin bu qoida o'zgarishi mumkin.

Saralash.

Ko'rib chiqilayotgan algebraik modelda o'rmon ichidagi elementlar ketma-ketligini ma'lum bir tarzda o'zgartirishga imkon beradigan turga oid ifoda berilgan. Ushbu ibora quyidagi umumiy formulaga ega:

```
sort VAR in EXP1 by EXP2
```

Bu erda VAR `EXP1` ifodasi bilan belgilangan o'rmonga tegishli bo'lgan elementlarning qatorini aniqlaydi va bu elementlarni `EXP2` ifoda bilan belgilangan qiymatdan foydalangan holda buyurtma qilishga imkon beradi. Misol uchun, elementlar `STAFFLISTO` elementlar tartibda tartiblashtiriladi u quyidagicha:

```
sort S in STAFFLISTO/STAFF by S/STAFFNO
```

Birlashtirish

Ushbu ko'rib chiqilayotgan algebraik model o'rtacha, hisoblash, yomg'ir, maksimal va yig'indisi yig'ilish funktsiyalarini qo'llab-quvvatlaydi. Masalan, yaqin qarindoshlari soni bittadan oshgan xodimlar to'g'risidagi ma'lumotlarni tanlash uchun siz quyidagi iborani ishlatishingiz mumkin:

```
for S in STAFFLISTO/STAFF do
    where count(S/NOK) > 1 do
        S: STAFF { 0, *}
```

Ushbu algebraik modelning to'liq tavsifi ushbu ma'ruzalar doirasiga kirmaydi. Ammo oxirida shuni ta'kidlash kerakki, ushbu algebra funktsiyalarni va strukturaviy rekursiyani ham qo'llab-quvvatlaydi.

XML uchun so'rovlar tili (XQuery).

W3C Query ishchi guruhi ***XQuery*** deb nomlangan XML uchun so'rovlar tilini taklif qiladi. XQuery Quilt deb nomlangan XML so'rovi tillaridan biriga asoslanadi, u o'z navbatida XPath, XML-QL, SQL, OQL, Lorel, XQL va YATL kabi bir qator boshqa tillarning ko'plab vositalarini o'z ichiga oladi. OQL singari, XQuery ham funktsional til bo'lib, unda har qanday so'rov ifoda sifatida taqdim etiladi. XQuery joylashtirilishi mumkin bo'lgan bir nechta iboralarni qo'llab-quvvatlaydi (shuning uchun u subquery tushunchasini qo'llab-quvvatlaydi). Ushbu bo'limda ushbu tilning ikkita jihati ko'rib chiqiladi: yo'l belgilari va FLWR iboralari. XQuery tilining to'liqroq tavsifi ushbu ma'ruzalar doirasiga kirmaydi.

Yo'l belgilari.

XQuery yo'l belgisi yangi ajratish operatori va diapazon predikati deb nomlangan yangi turdagi predikat bilan to'ldirilgan qisqartirilgan XPath sintaksisidan foydalanadi. XQuery-da, yo'l belgilarini qo'llash natijasi, tugunlarning buyurtma qilingan ro'yxati bo'lib, ular tarkibiga ushbu tugunlarning avlodlari ham kiradi. Yo'l belgisini qo'llash natijalari bo'yicha yuqori darajadagi tugunlar asl ierarxiyadagi mavqeiga qarab tartiblanadi; ketma-ketlikda yuqoridan pastga, chapdan o'ngga. Bunday iboraning misoli sifatida siz ko'rsatilgan hujjatning ildiz tugunini qaytaradigan hujjat (satr) funktsiyasini ko'rsatishingiz mumkin.

So'rovda, bir yoki ikkita kesishish belgilaridan (/yoki //) boshlanadigan yo'l belgisi bo'lishi mumkin, bu aniq aniqlanmagan ildiz tugunini bildiradi, uning ta'rifi

so'rov qilingan muhitga bog'liq.

(->) havolani olib tashlash operatsiyasidan ushbu atribut tomonidan ko'rsatilgan element (lar) ni olish uchun IDREF turidagi atributdan keyingi yo'lni belgilashda foydalanish mumkin. Bog'lanishni olib tashlash jarayoni maqsadli element ko'rsatilgan nomni tekshirish amaliyoti bilan amalga oshiriladi (yulduzcha (*) belgisi har qanday turdagi maqsadli elementga murojaat qilish imkonini beradi).

FLWR ifodasi.

FLWR iborasi FOR, LET, WHERE va RETURN konstruktsiyalari yordamida hosil bo'ladi. Har qanday SQL so'rovida bo'lgani kabi, ushbu konstruktsiyalar belgilangan tartibda ifodada bo'lishi kerak. FLWR ifodasi sizga qiymatlarni bir yoki bir nechta o'zgaruvchilar bilan bog'lash imkonini beradi va natijada (odatda tugunlarning buyurtma qilingan o'rmonlari) hosil qilish uchun ushbu o'zgaruvchilardan foydalaniladi.

FOR va / yoki LET konstruktsiyalari bir yoki bir nechta o'zgaruvchiga qiymatlarni bog'lash uchun ishlatiladi (masalan, yo'l belgisi). Qayta ishlashni amalga oshirish va har bir o'zgaruvchini tugunlar ro'yxatini qaytaradigan ifoda bilan bog'lash zarurati tug'ilganda FOR tuzilishi ishlatiladi. FOR konstruktsiyasini qo'llash natijasi har bir o'zgaruvchi uchun bog'lanishni belgilaydigan har bir biriktiruvchi tutqichlarning har biri barcha iboralar bilan qaytarilgan tugunlar ro'yxatlarining o'zaro bog'langan mahsulotlarini biriktiradigan birikmalar ro'yxatidir. FOR konstruktsiyasining har bir o'zgaruvchisini, uning mos keladigan ifodasi bilan qaytarilgan tugunlar orqali bog'langan deb hisoblash mumkin.

LET konstruktsiyasi sizga bitta yoki bir nechta o'zgaruvchini bitta yoki bir nechta iboralar bilan bog'lash imkonini beradi, ammo ko'chadan o'tkazmasdan, bu har bir o'zgaruvchiga bitta bog'lab turishga olib keladi. Masalan, $ S IN / STAFFLIST / STAFF konstruktsiyalari $ S o'zgaruvchini STAFFLIST ro'yxatidagi bitta STAFF elementiga bog'laydigan bir nechta natijalarni keltirib chiqaradi. Boshqa tomondan, LET $ S: = / STAFFLIST / STAFF tuzilishi $ S o'zgaruvchisini ro'yxatning barcha STAFF elementlarini o'z ichiga olgan ro'yxat bilan bog'laydi.

FLWR ifodasi bir nechta FOR va LET konstruktsiyalarini o'z ichiga olishi mumkin va har bir konstruktsiya oldingi konstruktsiyalar bilan bog'liq bo'lgan o'zgaruvchilarga havolalarni o'z ichiga olishi mumkin. FOR va LET konstruktsiyalarining ketma-ketligini qo'llash natijasida ular bog'liq bo'lgan kirish hujjati elementlarining ketma-ketligi bilan bog'liq bo'lgan o'zgaruvchilar turlarining ro'yxati; birinchi navbatda birinchi chegaralangan o'zgaruvchi, so'ngra ikkinchi chegaralangan o'zgaruvchi va boshqalar. Ammo agar FOR tuzilishida ishlatiladigan ba'zi bir iboralar tartiblanmagan bo'lsa (masalan, u DISTINCT funktsiyasini o'z ichiga olganligi sababli), unda FOR / LET ketma-ketligi natijasida hosil bo'lgan birikmalar ham tartibsizdir.

FOR bandida bog'langan va bitta tugunni ifodalaydigan o'zgaruvchilar odatda $ S / SALARY> 10000 kabi skalar predikatlarida qo'llaniladi, boshqa tomondan, LET bandida bog'langan o'zgaruvchilar tugunlarning ro'yxatlarini aks ettirishi mumkin va shuning uchun ko'pincha bunday hollarda ishlatiladi AVG ($ S /

SALARY)> 20000 sifatida ro'yxatni qayta ishlashga mo'ljallangan predikat. Esda tutingki, "WHERE" bandida FOR va LET so'zlari tomonidan hosil qilingan bog'lanish katakchalarining tartibini saqlaydi.

Savollar:

1. Ma'lumotlar XML-da qanday saqlanadi?
2. Ma'lumotlarning XML turlarini XMLda qanday saqlash mumkin ?
3. XSLT nima uchun kerak ?
4. SGML nima ?
5. XPATH so'rovlari qanday amalga oshiriladi ?
6. Ulanish va birlashtirish mumkinmi?

Adabiyotlar:

1. Thomas Connolly, Carolyn Begg – Database systems. A practical Approach to Design, Implementation and Management. 4th Edition – Addison Wesley 2005 – 1373p.
2. C. J. Date – An Introduction to Database Systems – Addison-Wesley Professional – 2003 – 1024 p.

AMALIY MASHG'ULOT MATERIALLARI

1-Amaliy mashg'ulot. MySQL-ni o'rnatish.

Ishning maqsadi:

MySQL ma'lumotlar bazasini WINDOWS oilasining operatsion tizimlariga qanday o'rnatishni o'rganish.

Topshiriq:

1. Rasmiy wveb-saytdanhttp:// mysql .com (server, ODBC drayveri , Workbench to'plami) dan yuklab oling.
2. MySQL o'rnatish uchun quyidagi ko'rsatmalarga amal qiling http://dev.mysql.com/doc/refman/5.7/en/windows-nstallation.html
3. O'rnatishdan so'ng, avval serverni sozlang (mysqld jarayonini xavfsizlik devori istisnolar ro'yxatiga qo'shishni unutmang).

Hisobot talablari:

Hisobotda quyidagi ma'lumotlar bo'lishi kerak:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Dasturni bosqichma-bosqich o'rnatishning skrinshotlari (ekran rasmlari Windows OS ish stoli bilan birga olinadi)
6. Ish bo'yicha xulosalar.

Nazorat savollar

1. MySQL- ni o'rnatishda qanday qiyinchiliklar mavjud ?
2. MySQL- ning birinchi boshlanishida qanday parametrlarni ko'rsatish kerak ?
3. Xizmat rejimida boshlash va normal ishga tushirish o'rtasidagi farq nima?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil
2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.
3. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

4. Luqo Velling, Laura Tomson - MySQL. O'quv qo'llanma - Uilyams 2005 yil

2 -Amaliy mashg'ulot. MySQL konfiguratsiyasi va uni boshqorish

Ishning maqsadi:

Ishning maqsadi MySQL ma'lumotlar bazasida muhitda ishlashni o'rganish va ma'lumotlar bazasini boshqarishning turli vositalaridan foydalanish.

Topshiriq:

1. MySQL Workbench vositalari bo'yicha qo'llanmani o'qing. http://dev.mysql.com/doc/refman/5.7/en/tutorial.html
2. Serverga ulaning
3. Bir nechta oddiy so'rovlarni bajaring
4. Ma'lumotlar bazasini yarating va USE buyrug'i yordamida tanlang.
5. View dasturlari va ma'lumotlar bazasi haqida ma'lumot axborot_ sxema.
6. MySQL-

ga ulanish qo'llanmasini sozlang : http://dev.mysql.com/doc/workbench/en/wb-getting-started-tutorial-create-connection.html

7. MySQL- dagi ma'lumotlar bazasiga APACHE serverini o'rnating va serverga kirishni sozlang .

Hisobot talablari:

Hisobotda quyidagi ma'lumotlar bo'lishi kerak

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Vazifaning barcha punktlarini bosqichma – bosqich bajarish skrinshotlari.
6. Ish bo'yicha xulosalar.

Nazorat savollar

1. MySQL serveriga ulanishda qanday parametrlarni ko'rsatish kerak ?
2. USE jamoasi nima uchun ishlatiladi?
3. Shou buyrug'i qanday ishlatiladi ?
4. Workbench- da nima qilishim mumkin ?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil

2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.

3. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

4. Luqo Velling, Laura Tomson - MySQL. O'quv qo'llanma - Uilyams 2005 yil

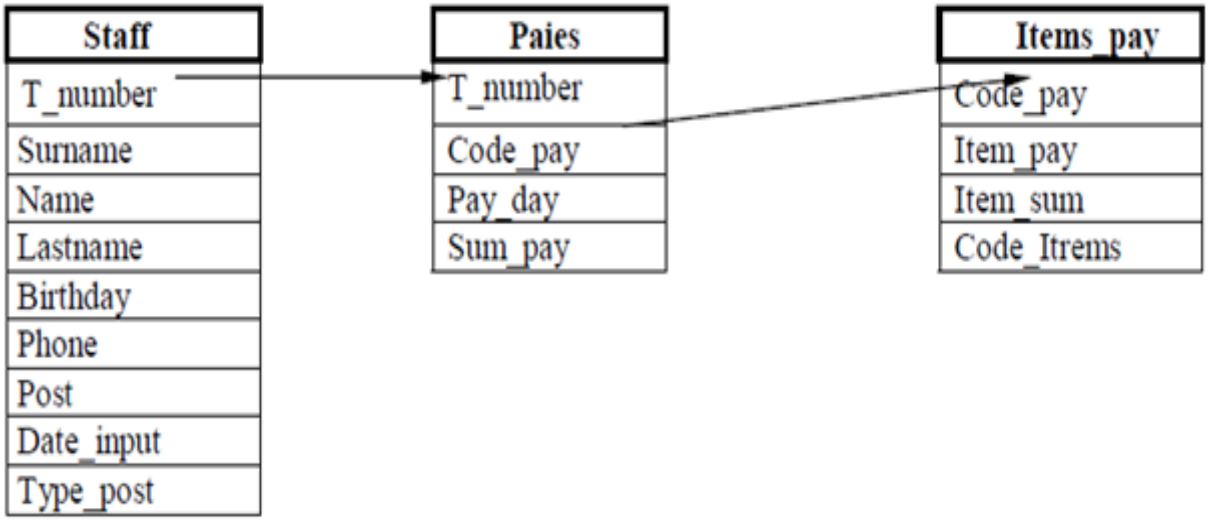
3 -Amaliy mashg'ulot. Jadvallar ustida so'rovlar (SELECT, INSERT, UPDATE, DELETE)

Ishning maqsadi:

Ma'lumotlar bazasini tuzishni o'rganish va oddiy `SELECT`, `INSERT`, `UPDATE`, `DELETE` so'rovlarini bajarish.

Topshiriq:

Quyida keltirilgan ma'lumotlar bazasining tuzilishi sizga korxonada ishlaydigan xodimlarning hisobini yuritishga, shuningdek, berilgan barcha ish haqini daromadlar va xarajatlar moddalari bo'yicha sharhlash bilan saqlashga imkon beradi (masalan, ish haqi, tuman koeffitsienti, daromad solig'i va boshqalar).



3.1-rasm. "Ish haqi" ma'lumotlar bazasining sxemasi.

Jadvallar o'rtasidagi o'zaro aloqa mos ravishda birdan to ko'pgacha bo'lgan munosabatlar turiga ega bo'lgan quyidagi juftliklar yordamida amalga oshiriladi:

1. Xodimlar.
2. Paies.Code_pay - Mahsulotlar _pay.Code_pay.

3.1-jadval. Xodimlar ro'yxati (shtat jadvali)

Maydon nomi	Maydon turi	Maydon tavsifi
T_number	Integer	Xodimlar soni (noyob)
Surname	Character	Xodimning familiyasi

Name	Character	Xodimning ismi
Lastname	Character	Xodimning otasining ismi
Birthday	Date	Xodimning tug'ilgan sanasi
Phone	Character	Xodimning aloqa telefoni
Post	Character	Xodimlarning lavozimi
Type_post	Character	Xodimning turi (muhandis, ishchi, ishchi)
Date_input	Date	Ishlagan sana

3.2- Jadval. Ish haqi hisobi jadvali (Paies jadvali) .

Maydon nomi	Maydon turi	Maydon tavsifi
T_number	Integer	Xodimning shaxsiy soni maosh olish
Code_pay	Integer	Ish haqi kodi (yagona)
Pay_day	Date	Ish haqi berilgan sana
Sum_pay	Numeric	Qo'lda jami ish haqi

3.3-jadval. Maqolalar bo'yicha har bir maosh uchun dekodlash jadvali (jadval Items_pay).

Maydon nomi	Maydon turi	Maydon tavsifi
Code_pay	Integer	Ish haqi kodi
Item_pay	Character	Hisoblangan maqola nomi ish haqi (daromad ham, xarajat ham)
Item_sum	Numeric	Ish haqini olish yoki ushlab qolish miqdori
Code_Itrems	Integer	Jadval tugmachalari maydoni

3.4-jadval. Shtat jadvalini to'ldirishga misol.

T_number	Surname	Name	Lastname	Birthday	Phone	Post	Type_post	Date_input
1	Иванов	Иван	Петрович	12.01.1971	124563	Бухгалтер	Служащий	12.04.2000
2	Сидоров	Василий	Михайлович	14.06.1954	451263	Начальник отдела кадров	ИТР	14.11.1999
3	Васильков	Петр	Аркадьевич	14.06.1981	145236	Специалист отдела кадров	Служащий	30.11.2000
67	Артемьев	Иван	Васильевич	05.12.1970	365462	Главный инженер	ИТР	10.02.1998
4	Соянов	Савел	Игнатьевич	15.05.1981	121212	Строитель	Рабочий	25.06.1980
11	Ушаков	Виктор	Семенович	30.05.1970	156462	Бухгалтер	Рабочий	18.11.2003
15	Иванова	Анна	Михайловна	12.03.1940	145214	Строитель	Служащий	12.11.1979

3.5-jadval. Paies jadvalini to'ldirishga misol

T_number	Code_pay	Pay_day	Sum_pay
1	1	01.01.2003	2544.00
1	2	01.02.2003	4521.00
1	3	01.03.2003	12542.00
2	4	01.01.2003	1452.00
2	5	01.02.2003	2145.00
2	6	01.03.2003	2135.00
3	7	01.01.2003	4511.00
3	8	01.02.2003	1542.00
3	9	01.03.2003	1542.00
4	10	01.03.2003	2456.00

3.6-jadval. Jadvalni to'ldirishga misol Items_pay

Code_pay	Item_pay	Item_sum	Code_Items
1	Премия	124.00	1
1	Налог	-451.00	2
1	Оклад	1457.00	3
1	Поощрение	4512.00	4
1	Оплата учебы	145.00	5
2	Оклад	4656.00	6
2	Налог	-415.00	7
2	Поощрение	326.00	8
3	Оклад	1654.00	9
3	Премия квартальная	1213.00	10
10	За бездетность	-154.00	11
10	Оклад	1456.00	12
10	Премия разовая	1245.00	13
10	Налог подоходный	-452.00	14

1. MySQL Workbench muhitida SQL so'rovlari yordamida ma'lumotlar bazasini yarating.

2. Jadvallarni `INSERT` buyruqlari orqali 3.4, 3.5 va 3.6 jadvallarda ko'rsatilgan ma'lumotlar bilan to'ldiring.

3. Xodimlar jadvalidagi xodimlar to'g'risidagi barcha ma'lumotlarni aks ettiring va natijani xodimlar soni bo'yicha tartiblang.

4. Xodimlarning familiyalari, ismlari, otalarining ismlarini, ularning lavozimlarini ro'yxatini ko'rsating, natijani lavozim nomi bo'yicha ko'tarilish tartibida pasayish tartibida tartiblang.

5. Paies jadvalidan ishchilarning shaxsiy raqamlarini va ish haqi olingan sanani tanlang va natijani sanaga qarab kamayib boruvchi tartibda tartiblang.

6. Quyidagicha ustunlar natijasida tartibi deb, bunday tarzda jadval xodimlari shtab barcha ma'lumotlarni chop: Name, Lastname, Surname, Post, Date_input, Phone, Birthday, T_number, Type_post.

7. Paies jadvalidagi barcha maydonlarni tanlang, shunda ustunlar tartibi quyidagicha bo'ladi: Sum_pay, Pay_day, T_Number, Code_pay.

8. Nomi "asosiy" bilan boshlanadigan lavozimga ega bo'lgan xodimlar ro'yxatini ko'rsating.

9. Ishchilar bo'lmagan ishchilar ro'yxati va ularning lavozimlarini ko'rsatish.

10. Buxgalteriya hisoblanmaydigan xodimlarning ro'yxati va ularning ishlagan sanalarini ko'rsatish.

11. 03/12/2000dan 06/15/2000 gacha bo'lgan davrda ishlagan xodimlarning ro'yxati va lavozimlari.

12. Qiymati 111111 dan 222222 gacha bo'lgan xodimlar va ularning telefonlari ro'yxatini ko'rsating.

13. "Kadrlar bo'limi boshlig'i", "Kadrlar bo'limi mutaxassisi", "Kadrlar bilan ishlash bo'yicha menejer" lavozimlari ko'rsatilgan xodimlar ro'yxatini ko'rsatish.

14. 4, 67, 45, 77 raqamlari bo'lgan xodimlar ro'yxatini ko'rsating.

15. 'N' harfi bilan boshlanadigan ish haqi moddalarining takrorlanmaydigan ro'yxatini chop eting.

16. Otalari ismlarida "wa" harflari birikmasi bo'lgan xodimlar ro'yxatini ko'rsating.

17. "El" bilan tugaydigan xabarlarning takrorlanmaydigan ro'yxatini tanlang.

18. 01.01.1950 - 01.01.1960 yilda tug'ilgan xodimlar ro'yxati yoki 10 dan 150 gacha bo'lgan xodimlar sonini ko'rsating.

19. Lavozimini "asosiy" dan boshlanadigan barcha xodimlarni ITR holatiga o'tkazing.

20. Barcha xodimlarni "faxriy nafaqaxo'r" maqomiga o'tkazing va agar ish staji 20 yoshdan oshgan bo'lsa va yoshi 60 yoshdan oshgan bo'lsa, ishdan oling.

21. Maydon qiymati bo'sh bo'lsa, "xabar yo'q" deb xabarning qiymatini o'zgartiring.

22. 80 yoshdan oshgan barcha xodimlarni stoldan olib tashlang.

23. Ish haqi haqidagi maqolalar jadvalidan (Items_pay jadvali) ish haqi to'g'risidagi maqolaning nomi = 'noma'lum' bo'lgan barcha yozuvlarni olib tashlang.

24. Ish haqi jadvalidan xodimlarning ish haqi miqdori va xodimlar soni 0 ga teng bo'lgan barcha yozuvlarni o'chirib tashlang.

Hisobot talablari:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Ma'lumotlar bazasini bosqichma - bosqich yaratish va to'ldirishning skrinshotlari.
6. SQL so'rov matni va SQL buyruq chiqishining skrinshotlari.
7. Ish bo'yicha xulosalar.

Nazorat savollar

1. `CREATE TABLE` buyrug'ining sintaksisi nima ?
2. Jadvallar bilan ishlashda qanday ma'lumotlardan foydalanishim mumkin?
3. Ma'lumot olish qanday amalga oshiriladi?
4. Ma'lumotlar qanday tartiblanadi?
5. `WHERE` predikati yordamida ma'lumotlar qanday tanlanadi ?
6. Nega `BETWEEN` va `LIKE ishlatiladi` ?
7. `SELECT` buyrug'ida ko'rsatilgan ma'lumotlarni tanlashning bir nechta shartlari qanday?
8. Ma'lumotlar qanday o'zgartiriladi?
9. Ma'lumotlar qanday o'chiriladi?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil
2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.
3. Martin Graber - SQL-ga kirish - Ed. Lori, 382 bet 1992 yil
4. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

4-Amaliy mashg'ulot. Jadvallarni qo’shishda so’rovlar (JOIN, UNION).

Ishning maqsadi:

Birlashtirish, kesishish va boshqa operatorlar yordamida bir nechta jadvallarni ustida amallar bajarishni o'rgatish.

Topshiriq:

1. 3-amaliy ishdan berilgan topshiriqda tasvirlangan ma'lumotlar bazasini qayta yarating.
2. Xodimlarning familiyalari, ismlari, otalarining ismlari ro'yxati (Surname, Name, Lastname), shuningdek ularning ish haqi qiymatlari (Sum_pay maydoni) va olingan sana (Sum_pay maydoni).
3. Shtat raqamlarini, ish haqi olingan sanani va uni maqolalar bo'yicha taqsimlashni ko'rsating, natijani xodimning shaxsiy raqamiga qarab tartiblang.
4. Maqolalar bo'yicha har bir ish haqi sxemasi ko'rsatilgan xodimlarning familiyalari va shaxsiy raqamlari ro'yxati, shuningdek, ularning ish haqi qiymatlari va qabul qilish kunlari ko'rsatilgan.

5. 01.01.2003 dan 01.01.2003 gacha bo'lgan davrda ishchilarning ro'yxati va olingan ish haqi miqdorini ko'rsating.

6. Quyidagi ish haqi qo'shimchalaridan birini oladigan xodimlar ro'yxatini ko'rsating: "bonus", "o'qish uchun to'lov", "rag'batlantirish".

7. 2003 yil 15 martda ish haqi olgan barcha xodimlarni 2000 dan 3000 rublgacha bo'lgan ish haqidan olib qo'ying.

8. 12-30 xodimlar soni yoki 5000 rubldan oshadigan ish haqi bo'lgan xodimlarning ismlari va xodimlarning ismlari bilan takrorlanmagan ro'yxatini ko'rsating.

9. Ish haqi 2000 dan 3000 rublgacha bo'lgan barcha xodimlarning takrorlanmaydigan ro'yxatini ko'rsating.

10. "Bolasizlik uchun" chegirmalar mavjud bo'lgan ish haqi kodlarini chop eting.

11. Oylik maoshida "bolasizlik uchun" ajratmasi bo'lgan barcha xodimlarning takrorlanmaydigan ro'yxatini ko'rsating.

12. Agar ular "bolalikdan" soliqni to'lamagan bo'lsa, barcha ishchilar ro'yxatini, ularning xodimlarining raqamlarini, qo'llaridagi ish haqi sana va miqdorlarini va ish haqini ko'rsating.

13. Xodimlarning ro'yxatini va ularning har birining umumiy ish haqini ko'rsating.

Hisobot talablari:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Ma'lumotlar bazasini bosqichma-bosqich yaratish va to'ldirishning skrinshotlari.
6. SQL so'rov matni va SQL buyruq chiqishining skrinshotlari.
7. Ish bo'yicha xulosalar.

Nazorat savollar:

1. Ulanish qanday amalga oshiriladi?
2. Qanday birikmalar turlarini bilasiz?
3. Qanday qilib uchdan ortiq jadvallar birlashtiriladi?
4. `JOIN` operatori nima uchun kerak ?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil

2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.

3. Martin Graber - SQL-ga kirish - Ed. Lori, 382 bet 1992 yil

4. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

5-Amaliy mashg'ulot. Qo'shimcha so'rovlar orqali amallar bajarish.

Ishning maqsadi:

Pastki so'rovlarni va guruh operatsiyalarini o'z ichiga olgan so'rovlar bilan ishlash ko'nikmalarni shakillantirish.

Topshiriq:

1. Korxonada to'langan o'rtacha ish haqini olib qo'ying.
2. Xodimlarning ro'yxatini va ularning har birining umumiy ish haqini ko'rsating.
3. O'tgan bir yil davomida har bir ishchining o'rtacha ish haqini chop eting.
4. Har bir lavozim uchun ishchilar sonini chop eting .
5. Eng birinchi va oxirgi xodimning ishlashi uchun qurilmaning sanasini ko'rsatish.
6. Ishchilar ro'yxati va har birining ish haqi ko'rsatilgan bo'lib, ular maydonga Itog nomi bilan joylashtirilishi kerak .
7. Barcha ishchilar ro'yxati, ularning shaxsiy tarkibi, ish haqi va ish haqi miqdori, agar "daromad solig'i" olinmasa, natijani Sum_With_Nalog ustuniga qo'ying.
8. Berilgan familiyalarni, ismlarni, otalarning ismlarini bitta ustunga FIO nomi bilan birlashtiring.
9. Ushbu familiyalarni, ismlarni, otalarning ismlarini va ish nomlarini bitta ustunga FIO_Post nomi bilan birlashtiring.
10. "Bonus", "o'qish", "mukofot" va ularning ish haqi kodlaridan birini oladigan xodimlarning ro'yxatini ko'rsatin .
11. Mukofotni olgan xodimlarning noyob ro'yxatini ko'rsating .
12. Hech qachon maosh olmagan ishchilar ro'yxatini ko'rsating.
13. Ish haqi 3000 rubldan kam bo'lmagan xodimlarning ro'yxatini ko'rsating.
14. Ishchilar ro'yxati va o'rtacha ish haqidan oshgan ish haqi sanalari ko'rsatilgan.

Hisobot talablari:

1. Sarlavha sahifasi
2. Amaliy ishning nomi

3. Amaliy ishning maqsadi
4. Vazifa
5. SQL so'rov matni va SQL buyruq chiqishining skrinshotlari .
6. Ish bo'yicha xulosalar.

Nazorat savollar

1. So’rovlar bilan ishlash nima?
2. Guruhlash qanday amalga oshiriladi?
3. Qaysi agregat funktsiyalarni bilasiz?
4. Murakkab so'rovlari necha marta bajariladi?
5. `IN`, `ANY`, `ALL` operatorlari nima uchun kerak ?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil
2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.
3. Martin Graber - SQL-ga kirish - Ed. Lori, 382 bet 1992 yil
4. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

6-Amaliy mashg'ulot. ORACLEni o'rnatish.

Ishning maqsadi:

Microsoft ORACLE o'rnatishni bilib olish.

Topshiriq:

1. ORACLE O'rnatish qo'llanmasini o'qing.
2. Kerakli tarqatish to'plamini yuklab oling yoki bunga mos keladigan tarqatish to'plamini yuklab olish uchun foydalaning.
3. ORACLE o'rnating.

Hisobot talablari:

Hisobotda quyidagi ma'lumotlar bo'lishi kerak:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Dasturni bosqichma-bosqich o'rnatishning skrinshotlari (ekran rasmlari Windows OS ish stoli bilan birga olinadi)

6. Ish bo'yicha xulosalar.

Nazorat savollar

1. ORACLEni o'rnatishda qanday qiyinchiliklar mavjud ?
2. WINDOWS ishga tushganda ORACLE qanday xizmatlarni ishga tushiradi?
3. default va named instance nima farqi bor?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil
2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.
3. Martin Graber - SQL-ga kirish - Ed. Lori, 382 bet 1992 yil
4. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

7-Amaliy mashg'ulot. ORACLEni boshlash va to'xtatish. Fayllar bilan ishlash. (REDO logs, Datafiles)

Ishning maqsadi:

ORACLE sozlamalarini sozlash va ORACLEning asosiy fayllarini boshqarishni o'rganish.

Topshiriq:

1. my.ini faylini sozlash uchun ORACLE server qo'llanmasi bilan tanishib chiqing.
2. Kirill harflari bilan ishlash uchun UTF -8 formatida taqqoslashni o'rnating .
3. So'rovlarni bajarish uchun ishlatiladigan kesh hajmini oshiring.
4. Ma'lumotlar bazasi fayllarini saqlash uchun katalogni o'zgartiring.
5. Ildiz parolini tiklash qo'llanmasini o'qing .
6. Yangi ildiz parolini tayinlang .

Hisobot talablari:

Hisobotda quyidagi ma'lumotlar bo'lishi kerak:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Sozlamalardagi o'zgarishlarning skrinshotlari.
6. Ildiz parolini tiklashning skrinshotlari .
7. Ish bo'yicha xulosalar.

Savollar:

1. my.ini fayli nima uchun kerak?
2. Qaysi buyruq parolni qayta tiklash uchun ishlatish kerak root foydalanuvchisi uchun?
3. Qaysi buyruq root parolini tiklaydi?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil
2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.
3. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

8- Amaliy mashg'ulot. Foydalanuvchilarni boshqarish.

Ishning maqsadi:

Foydalanuvchi yaratishni, o'chirishni, shuningdek foydalanuvchi huquqlarini tayinlashni va o'chirishni o'rganishni shakillantirish.

Topshiriq:

1. "Xodimlar" ma'lumotlar bazasi sxemasi uchun yangi foydalanuvchi yarating.
2. «Staff», « paies », « Items_Pay » foydalanuvchi jadvallariga ruxsat berish.
3. Ma'lumotlar bazasiga yangi foydalanuvchi sifatida ulaning.
4. Foydalanuvchi uchun berilgan imtiyozlarni tekshiring.
5. Xodimlar jadvalining tuzilishini «Staff» o'zgartirishga ruxsat berib, yangi foydalanuvchiga imtiyozlarni kengaytirish .
6. Xodimlar «Staff» jadvaliga 2 ustun qo'shing.
7. Yangi foydalanuvchidan barcha imtiyozlarni tanlang.

Hisobot talablari:

Hisobotda quyidagi ma'lumotlar bo'lishi kerak:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Foydalanuvchi yaratish uchun so'rov.
6. Yangi yaratilgan foydalanuvchi ulanishining skrinshoti.
7. Foydalanuvchi huquqlarini amalga oshirish buyrug'i.
8. Xodimlar «Staff» jadvalini o'zgartirish skrinshotlari .
9. Ish bo'yicha xulosalar.

Nazorat savollar

1. Foydalanuvchi qanday yaratilgan?
2. Qaysi buyruq foydalanuvchi huquqlarini tayinlaydi?
3. Agar foydalanuvchiga CONNECT huquqlari berilmagan bo'lsa nima bo'ladi ?
4. Foydalanuvchi paroli qanday o'zgartiriladi?
5. Foydalanuvchi huquqlarini qanday olib qo'yish kerak?
6. Foydalanuvchi uchun qanday huquqlar berilishi mumkin?
7. Rol nima?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil

2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.

3. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil

9 –Amaliyot mashg'ulot. Ma'lumotlar bazasida qo'shish

Ishning maqsadi:

ORACLE ma'lumotlar bazasini tiklash uchun SQL skriptlaridan foydalanish bo'yicha amaliy ko'nikmalarni hosil qilish.

Topshiriq:

Yuklab olingan arxivdan foydalaning:

http://195.158.2.210/employees_db-code-1.0.6.rar

Keyingi amaliy ishlarni bajarish uchun xodimlarning ma'lumotlar bazasini tiklash kerak.

Buning uchun administrator rejimida konsol CMD buyruqni qayta ishlash vositasini ishga tushirishingiz kerak .

Keyin **CD- papkadagi "papka yo'li"** buyrug'i yordamida ochilmagan ma'lumotlar bilan papkaga o'ting.

Masalan, **CD "C: \ Yuklab olishlar"**

Keyin MySQL serveri qayerda o'rnatilganligini toping. Odatiy bo'lib, u C:\Program Files\MySQL\My sql server 5.6\bin\mysql.exe papkasida o'rnatiladi.

skript ijrosini ishga tushirish **"ga yo'l MySQL -" U ildizi - u root –p –t < employees.sql**

Masalan, " C : \ Program Files \ MySQL \ My sql server 5.6 \ bin \ mysql.exe" –u- **root - -p <** tublish.sql

Buyruqni kiritganingizdan so'ng, administrator paroli so'raladi (agar siz

parolni to'g'ri o'rnatgan bo'lsangiz, ushbu parolni kiriting) va agar parol to'g'ri kiritilsa, skript ishga tushiriladi.

Skript tugagandan so'ng, muvaffaqiyatli xabar ko'rsatiladi.

Hisobot talablari:

Hisobotda quyidagi ma'lumotlar bo'lishi kerak:

1. Sarlavha sahifasi
2. Amaliy ishning nomi
3. Amaliy ishning maqsadi
4. Vazifa
5. Skriptni buyruq satrida ishga tushirish natijalari va SQL muhitidan ma'lumotlar bazalari jadvallarining rasmlari.

P . S . Ekran rasmlari oynaning sarlavhasi va vazifalar panelini o'z ichiga olgan butun ishlaydigan ekranni ko'rsatishi kerak.

Nazorat savollar:

1. Ma'lumotlar bazasini tiklash qanday amalga oshiriladi?
2. Ma'lumotlar bazasini tiklashdan eksport qanday farq qiladi?
3. Qanday hollarda ma'lumotlarni tiklash kerak va qaysi hollarda import qilish kerak?
4. Ma'lumotlar qanday zaxiralanadi?

Adabiyotlar:

1. C.J.Date. - ma'lumotlar bazasi tizimlariga kirish ed. Uilyams, 1328 bet 2006 yil
2. Jeffrey D. Ullman - ma'lumotlar bazasi tizimlari 3 tamoyillari rd Stenford universiteti keyin Informatika Press - Edition. - 2006 - 1137 b.
3. Mullins Kreyg S. - Ma'lumotlar bazasini boshqarish. Usullar va protseduralar bo'yicha to'liq qo'llanma. Per ingliz tilidan Nashriyotchi: Kudits-Obraz. Chiqarilgan yili: 2003 yil
4. Luqo Velling, Laura Tomson - MySQL. O'quv qo'llanma - Uilyams 2005 yil

GOLOSARRIY

Русский

1. **Администратор базы данных (АБД)** – это лицо или группа специалистов, знакомых с теорией построения информационных систем с базами данных и со спецификой предметной области данной информационной системы. Администратор базы данных осуществляет централизованное управление базой данных посредством конкретной СУБД.
2. **Аномалии операций с отношениями базы данных** – нежелательные эффекты, которые могут иметь место при осуществлении операций, связанных с изменением данных, а именно операций INSERT (вставка кортежа), DELETE (удаление кортежа) и UPDATE (обновление значений атрибута какого-либо кортежа).
3. **База данных** – совокупность связанных данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования, независимая от прикладных программ. База данных является информационной моделью предметной области. Обращение к базам данных осуществляется с помощью системы управления базами данных (СУБД).
4. **База данных (БД)** – это совокупность данных, организованная по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования данными.
5. **Домен реляционного отношения** – это множество скалярных (атомарных, неделимых) элементов, из которого могут браться значения конкретного атрибута.
6. **Запрос** – процесс обращения пользователя к БД с целью ввода, получения или изменения информации в БД.
7. **Логическая структура БД** – определение БД на физически независимом уровне, ближе всего соответствует концептуальной модели БД.
8. **Локальная автономность** – означает, что информация локальной БД и связанные с ней определения данных принадлежат локальному владельцу и им управляются.
9. **Пользователь БД** – программа или человек, обращающийся к БД на ЯМД.
10. **Словарь (справочник) базы данных** – специализированная подсистема СУБД, предназначенная для централизованного хранения единообразной информации обо всех хранимых в БД данных, используемой СУБД для доступа к данным
11. **Сущность** – конкретный опознаваемый предмет, элемент, единица (реальные или абстрактные), имеющие четко определенное функциональное назначение, четко определенные границы в данной предметной области, обусловленные контекстом задач, в рамках которых представляет интерес информация о данной предметной области.

12. **Схема базы данных** – это описание базы данных в контексте конкретной модели данных

13. **Топология БД, структура распределенной БД** – схема распределения физической БД по сети.

14. **Транзакция** – это логическая единица работы СУБД, последовательность операторов манипулирования данными, выполняющаяся как единое целое и переводящая базу данных из одного согласованного состояния в другое. Лозунг транзакции “все или ничего.

15. **Удаленный запрос** – запрос, который выполняется с использованием модемной связи.

16. **Экземпляры связей** – связи, имеющие место между конкретными экземплярами объектов.

17. **Язык манипулирования данными (или язык запросов к базе данных) (Data Manipulation Language - DML)** – набор команд, реализующих операции манипулирования данными.

18. **Язык описания данных (Data Definition Language - DDL)** – это язык высокого уровня, предназначенный для задания схемы базы данных. С его помощью описываются типы данных, подлежащих хранению в базе или выборке из нее, структура данных и их связи между собой.

Английский

1. **The database administrator (DBA)** is a person or group of specialists who are familiar with the theory of building information systems with databases and with the specifics of the subject area of this information system. The database administrator performs centralized management of the database through a specific DBMS.

2. **Anomalies in operations with database relationships** are undesirable effects that may occur when performing operations related to data changes, namely INSERT operations, DELETE and UPDATE (update of attribute values of a tuple).

3. **Database -** a set of related data, organized according to certain rules, providing general principles of description, storage and manipulation, independent of the application programs. The database is an information model of the domain. The database is accessed through a database management system (DBMS).

4. **The domain of a relational relation** is a set of scalar (atomic, indivisible) elements from which the values of a particular attribute can be taken**.**

5. **Request -** the process of accessing the user to the database in order to enter, receive or modify information in the database.

6. **Logical structure of the database -** the definition of the database at a physically independent level, closest corresponds to the conceptual model of the database.

7. **Local autonomy mea**ns that the information of the local database and associated data definitions belong to the local owner and are managed**.**

8. **The user of the database** is a program or person accessing the database on the DML.

9. **The database dictionary** is a specialized DBMS subsystem intended for centralized storage of uniform information about all data stored in the database used by the DBMS to access data

10. **The entity** is a concrete identifiable object, an element, a unit (real or abstract), having a well-defined functional purpose, clearly defined boundaries in the given subject area, conditioned by the context of the tasks within which information on the given subject area is of interest.

11. **A database schema** is a description of a database in the context of a particular data model

12. **Database topology, distributed database structure -** the scheme of physical database distribution over the network.

13. **A transaction** is a logical unit of the DBMS operation, a sequence of data manipulation operators that runs as a unit and transfers the database from one agreed state to another. The slogan of the transaction is "all or nothing."

14. **A remote request** is a request that is made using a dial-up connection.

15. **Link instances** are the relationships that occur between specific instances of objects.

16. **Data manipulation language (or data query language) (Data Manipulation Language (DML**) is a set of commands that implement data manipulation operations.

17. **The Data Definition Language (DDL)** is a high-level language intended for specifying a database schema. It describes the types of data to be stored in a database or a sample from it, the data structure and their relationships with each other.

O’zbekcha

1. **Ma'lumotlar bazasi administratori (DBA) -** bir kishi yoki ma'lumotlar bazalari va axborot tizimining o'ziga xos domen bilan axborot tizimlari nazariyasi bilan tanish mutaxassislar bir guruh. Ma'lumotlar bazasi ma'muri ma'lumotlar bazasini markazlashtirilgan boshqarish uchun maxsus DBMS orqali amalga oshiradi.

2. **Ma'lumotlar bazasi munosabatlar bilan anomaliyalar operatsiyalar -** ma'lumotlar, masalan, INSERT operatsiyalar (Insert shamlardan) o'zgarishi bilan bog'liq operatsiyalarni amalga oshirishda yuzaga kelishi mumkin kiruvchi ta'sir, o'chirish (shamlardan nomlash), va UPDATE (yangilash bir shamlardan xususiyati qiymatlari).

3. **Ma'lumotlar bazasi -** ilovaning mustaqil bayon qilish, saqlash va qayta ishlash, umumiy tamoyillari ta'minlash muayyan qoidalar asosida tashkil tegishli ma'lumotlarni to'plash. Ma'lumotlar bazasi domenning axborot modeli. Ma'lumotlar bazasiga ma'lumotlar bazasini boshqarish tizimi orqali kirish mumkin.

4. **Relyatsion algebraning maydoni,** ma'lum bir xususiyatning qiymatlari qabul qilinishi mumkin bo'lgan skalar (atom, bo'linmas) elementlarning to'plamidir.

5. **Talab -** ma'lumotlar bazasida ma'lumotlarni kiritish, olish yoki o'zgartirish uchun foydalanuvchini ma'lumotlar bazasiga kirish jarayoni.

6. **Ma'lumotlar bazasining mantiqiy tuzilishi -** jismonan mustaqil darajada ma'lumotlar bazasini aniqlash, eng yaqin ma'lumotlar bazasining kontseptual modeliga mos keladi.

7. **Mahalliy muxtoriyat –** mahalliy ma'lumotlar bazasi va unga aloqador ma'lumotlar ta'riflari mahalliy egasiga tegishli ekanligini anglatadi va boshqariladi.

8. **Ma'lumotlar bazasi foydalanuvchisi –** dastur yoki NMD bazasiga kirgan shaxs.

9. **Ma'lumotlar bazasi lug'ati -** ma'lumotlar bazasiga kirish uchun DBMS tomonidan foydalaniladigan ma'lumotlar bazasida saqlanadigan barcha ma'lumotlar haqida yagona ma'lumotni markaziy saqlash uchun mo'ljallangan maxsus DBMS kichik tizimidir

10. **Mazmuni –** aniqlangan aniq funktsional maqsadga ega bo'lgan, ushbu mavzu bo'yicha aniq belgilangan chegaralarni aniqlaydigan aniq ob'ektni, elementni, birlikni (haqiqiy yoki mavhum), ushbu mavzu bo'yicha axborotni qiziqtiradigan vazifalar konteksti bilan shartlangan.

11. **Ma'lumotlar bazasi sxemalari** – ma'lumotlar bazasining ma'lum bir ma'lumot modeli kontekstidagi tavsifi.

12. **Ma'lumotlar bazasi topologiyasi, tarqalgan ma'lumotlar bazasi tuzilishi –** tarmoq orqali jismoniy ma'lumotlar bazasini tarqatish sxemasi.

13. **Tranzaksiya –** DBMS operatsiyaning mantiqiy birligi bo'lib, bir birlik sifatida ishlaydigan ma'lumotlar bazasini boshqaruvchi operatorlarning ma'lumotlar bazasini boshqasiga o'zgartiradi. Jurnalning shiori "butun yoki hech narsa".

14. **Masofaviy so'rov –** bu modemning aloqasidan foydalanib, so'rov**.**

15. **Bog'lanish misollari** ob'ektlarning muayyan nusxalari orasidagi o'zaro bog'liqlikdir.

16. **Ma'lumotni manipulyatsiya tili (yoki ma'lumotlar so'rovining tili) (Data Manipulation Language (DML) -** ma'lumotni manipulyatsiya operatsiyalarini bajaradigan buyruqlar majmui**.**

17. **Ma'lumotni aniqlash tili (DDL) -** ma'lumotlar bazasi sxemasini aniqlash uchun mo'ljallangan yuqori darajadagi tildir. Ma'lumot bazasida yoki undan namunada saqlanadigan ma'lumotlar turlarini, ma'lumotlar strukturasini va ularning o'zaro munosabatlarini tavsiflaydi.

Foydalanilgan adabiyotlar ro’yxati

1. O‘zbekiston Respublikasini yanada rivojlantirish bo‘yicha harakatlar strategiyasi to‘g‘risida. O‘zbekiston Respublikasi Prezidentining PF – 4947 – son farmoni. Toshkent, 2017 yil 7 fevral.
2. Mirziyoev Sh.M. Buyuk kelejagimizni mard va olijanob halqimiz bilan birga quramiz. 2017 yil.
3. Mirziyoev Sh.M. Qonun ustuvorligi va inson manfaatlarini ta’minlash - yurt taraqqiyoti va halq farovonligining garovi. 2017 yil.
4. Mirziyoev Sh.M. Erkin va farovon, demokratik O‘zbekiston davlatini barpo etamiz. 2017 yil.
5. Роб П. Системы баз данных: проектирование, реализация и управление (5-е издание) издательство "БХВ - Санкт-Петербург" ·1200 стр, 2003 г. ·.
6. Григорьев Ю.А., Плутенко А.Д.. Жизненный цикл проектов распределенных баз данных. Благовещеснк АмГУ, 1999.
7. Дунаев С.С. Доступ к базам данных и техника работы в сети. Практические приемы современного программирования. М.: Диалог – МИФИ, 1999.
8. Дж.Ульман, Дж Уидом. Введение системы баз данных. Пер.с англ. М.: «Лори», 2000.
9. Диго С.М. Базы данных Проектирование и использование. издательство "Финансы и статистика" · 592 стр, 2005 г.
10. Конноли Т., Брегк К. Базы данных, проектирование, реализация и сопровождения, теория и практика, Университет Пейсли, Шотландия, изд. М.- СПБ.- Киев, 2003.
11. Четвериков, В. Н. Базы и банки данных [Текст] : учебник для вузов по спец. "Автоматизир. системы управления" / Г. И. Ревунков, Э. Н. Самохвалов. - М. : Высш. шк., 1987. - 248 с. : ил. - Библиогр.: с.246 (14 назв.). Предм. указ.: с. 247.
12. А. Д. Хомоненко, В. М. Цыганков, М. Г. Мальцев Базы данных [Текст] : учебник для вузов / - 4-е изд., доп. и перераб. - СПб : Корона принт, 2004. - 736 с. - 1 экз.
13. Б. Я. Советов, В. В. Цехановский, В. Д. Чертовский. Базы данных. Теория и практика [Текст] : учебник для студ. вузов / - М. : Высш. шк., 2005. - 463 с. : ил. - Список лит. с. 459-460. - 2 экз.
14. Астахова И.Ф., Толстобров А.П. СҚЛ в примерах и задачах. Учебное пособие. Новое знание, 176 стр, 2002 г.
15. Полякова. Л.Н. Основы СҚЛ. Курс лекций. Учебное пособие. издательство "ИНТУИТ.РУ" · 368 стр, 2004 г.
16. Бен Форта Освой самостоятельно СҚЛ. 10 минут на урок (3-е издание) издательство "Вильямс" 288 стр, 2005 г. ·
17. Клыков Ю. И. Банки данных для принятия решений [Текст] : монография / Ю. И. Клыков, Л. Н. Горьков. - М. : Сов. радио, 1980. - 208 с. - 1 экз.

18. Шомье Ж. Банки данных [Текст] : использ. электрон. вычисл. техники / Пер. с фр. Ю.Л. Смирнова ; Под ред. Б.А. Щукина. - М. : Энергоиздат, 1981. - 70 с. : ил. - (Б-ка по автоматике ; вып.619). - Библиогр.: с. 69. - 2 экз.
19. Аппак М. А. Базы данных в АСУ-связь [Текст] : монография / М. А. Аппак. - М. : Радио и связь, 1987. - 80 с. - 2 экз.
20. Лори, Питер Базы данных для микроЭВМ [Текст] : монография / Пер. с англ. Ю.К. Трубина. - М. : Машиностроение, 1988. - 135 с. : ил. - 1 экз
21. Хансен, Гэри Базы данных [Текст]: разработка и управление / Пер. с англ. под ред. С. Каратыгина. - М. : БИНОМ, 1999. - 699 с. - 1 экз.
22. Марков, А. С. Базы данных. Введение в теорию и методологию [Текст] : учебник для студ. / А. С. Марков, К. Ю. Лисовский . - М. : Финансы и статистика, 2004. - 512 с. : ил. - Рекомендуемая лит. с. 431-435. -Предм. указ. с. 499-511. - 1 экз.
23. Маллинс Крейг С. Администрирование баз данных. Полное справочное руководство по методам и процедурам. Пер. с англ. Издательство: Кудиц-Образ. Год издания: 2003.
24. Кренке Д. Теория и практика построения баз данных. 8-издание, СПб.: Питер, 2003.-880с.
25. Конноли Томас, Каролин Бегг Базы данных .Проектирование, реализация и сопровождение. Теория и практика. 3-издание – М. : Изд.дом Вильямс - 2003. – 1440 с.
26. Дунаев В.В. Базы данных. Язык СҚЛ – СПб.: --БХВ-Петербург,2006.-288с.
27. Григорьев Ю.А. Банки данных. Учеб.для вузов.- М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. -320с.
28. Т.А.Хужакулов. Системы управление базами данных. Учеб.для вузов.-Т.:Изд-во Алокачи. 2019. 370 с.

my.gov.uz
egov.uz
lex.uz
www. Ziyonet.uz
www.library.tuit.uz
www.intuit.ru

Ma’lumotlar bazasnini boshqarish tizimlari
5350500 – Pochta aloqasi texnologiyasi
ta’lim yo’nalishi talabalari uchun darslik

“Axborot texnologiyalari” kafedrasining 2022 yil “___”_____(_____ - sonly bayonnoma) majlisida ko‘rib chiqildi va chop etishga tavsiyalandi

“Kompyuter Injiniring” fakultetinig ilmiy – uslubiy Kengashida majlisida ko‘rib chiqildi va chop etishga tavsiyalandi
2022 yil “____”_____, ____- sonly bayonnoma

TATU ilmiy – uslubiy kengashi majlisida ko‘rib chiqildi va chop etishga tavsiyalandi
2022 yil “_____”_____, ___- sonly bayonnoma

Tuzuvchilar: T.A.Xo‘jakulov,

Taqrizchilar: E.Sh.Nazirova
F.R.Nurjanov

Ma’sul muharrir: H.N.Zaynidinov
Korrektor: R.T.Gaipnazarov