O'ZBEKISTON RESPUBLIKASI OLIY TA'LIM, FAN VA INNOVATSIYALAR VAZIRLIGI

BUXORO DAVLAT UNIVERSITETI

G.B. MURODOVA

MA’LUMOTLAR BAZASINI BOSHQARISH TIZIMLARI

O`quv qo`llanma

“Durdona” nashriyoti
Buxoro – 2024

UO’K 004.65(075.8)
32.973-018.2ya73
M 89
Muradova, G.B.
Ma’lumotlar bazasini boshqarish tizimlari [Matn] : o’quv qo’llanma / G.B. Muradova; muharrir A. Qalandarov. – Buxoro: Sadriddin Salim Buxoriy, 2024. – 148 b.

KBK 32.973-018.2ya73

Ushbu o’quv qo’llanma talabalarga ma'lumotlar bazasining asosiy tushunchalari, ularning tuzilmasi, mohiyat-aloqa diagrammasini qurishni o'rgatadi. Shuningdek, o’quv qo’llanmada ma’lumotlar bazasining SQL tili va uning so'rovlarini tuzish, ular ustida amallar bajarish va so'rovlarni amalga oshirish asoslari keltirilgan.
O’quv qo’llanma 60610200-Axborot tizimlari va texnologiyalari (tarmoqlar va sohalar bo`yicha) ta’lim yo'nalishlarda tahsil olayotgan talabalar uchun mo'ljallangan.

Taqrizchilar:
A.Yu.Dauletov, Alfraganus university Raqamli taxnologiyalar fakulteti Raqamli texnologiyalar kafedrasi datsent v.b., (PhD)
Sh.E.Nazarov, Buxoro davlat universiteti Axborot texnologiyalar kafedrasi dotsenti, t.f.f.d.

O’quv qo’llanma O’zbekiston Respublikasi Oliy ta’lim, fan innovatsiyalar vazirligi Buxoro davlat universitetining 2024-yil 2-iyuldagi 452-sonli buyrug’iga asosan nashr etishga ruxsat berilgan. Ro’yxatga olish raqami 452-54.

ISBN 978-9910-682-73-5

.

MUNDARİJA

KIRISH

Hozirgi kunda barcha sohalarda ma'lumotlar bazasi (MB) kerakli axborotlarni saqlash va undan oqilona foydalanishda juda muhim rol o’ynamoqda. Jamiyat taraqqiyotining qaysi jabhasiga nazar solmaylik o’zimizga kerakli ma'lumotlarni olish uchun, albatta, MBga murojaat qilishga majbur bo’lamiz. Demak, MBni tashkil qilish axborot almashuv texnologiyasining eng dolzarb hal qilinadigan muammolaridan biriga aylanib borayotgani davr taqozasidir. Axborot texnologiyalarining rivojlanishi va axborot oqimlarining tobora ortib borishi, ma'lumotlarning tez o’zgarishi kabi holatlar insoniyatni bu ma'lumotlarni o’z vaqtida qayta ishlash choralarining yangi usullarini qidirib topishga undamoqda. Ma'lumotlarni saqlash, uzatish va qayta isMash uchun MBni yaratish, so’ngra undan keng foydalanish bugungi kunda dolzarb bo’lib qolmokda. Moliya, ishlab chiqarish, savdo-sotiq va boshqa korxonalar ishlarini ma'lumotlar bazasisiz tasavvur qilib bo’lmaydi.

Axborotlarni qavta ishlashni optimallashtirish maqsadida axborot tizimlari yaratiladi. Avtomatlashgan axborot tizimlari (AT) deb shunday tizimlarga aytiladiki, ularning tarkibida texnik vositalar, jumladan shaxsiy kompyuterlar ishtrok etadi. ATlarni keng ma’noda axborotni qayta ishlaydigan ixtiyoriy tizim sifatida tushunish mumkin. Tadbiq etish sohasiga qarab, AT ishlab chiqarish, ta’lim, sogliqni saqlash, harbiy va boshqa sohalarda ishlatiladigan tizimlarga ajratish mumkin.

Axborot texnologiyalarining rivojlanishi va axborot oqimiarining tobora ortib borishi, ma’lumotlaming tez o’zgarishi kabi holatlar insoniyatni bu ma'lumotlarni o’z vaqtida qayta ishlash choralarining yangi usuliarini qidirib topishga undamoqda. Ma'lumotlarni saqlash, uzatish va qayta ishlash uchun MBni yaratish, so’ngra undan keng foydalanish bugungi kunda dolzarb bo’lib qolmoqda. Moliya, ishlab chiqarish, savdo-sotiq va boshqa korxonalar ishlarini ma'lumotlar bazasisiz tasavvur qilib bo’lmaydi.

Ma’lumki, MB tushunchasi fanga kirib kelgunga qadar, ma’lumotlardan turli ko’rinishda foydalanish juda qiyin edi. Dastur tuzuvchilar ma’lumotlarini shunday tashkil qilar edilarki, u faqat qaralayotgan masala uchungina o’rinli bo’lardi. Har bir yangi

masalani hal qilishda ma’lumotlar qaytadan tashkil qilinar va bu hol yaratilgan dasturlardan foydalanishni qiyinlashtirar edi.

Har qanday axborot tizimining vazifasi real muhit obyektlari haqidagi ma’lumotlarga ishlov berishdan iborat. Keng ma’noda *ma’lumotlar bazasi* - bu qandaydir bir predmet sohasidagi real muhitning aniq obyektlari haqidagi ma’lumotlar to‘plamidir. Predmet sohasi deganda avtomatlashtirilgan boshqarishni tashkil qilish uchun o’rganilayotgan real muhitning ma’lum bir qismi tushuniladi. Masalan, korxona, zavod, oliy o’quv yurti va boshqalar.

1-MAVZU. KIRISH. BERILGANLAR BAZASINING NAZARIY ASOSLARI.

Reja:

1. Ma'lumotlar bazasi nazariyasiga kirish
2. Asosiy tushunchalar
3. Ma'lumotlar bazasi tizimining komponentlari

Tarixan kompyuter texnikasidan foydalanishning ikkita asosiy yo’nalishi mavjud bo’lib, ulardan birinchisi oddiy tuzilishga ega bo’lgan nisbatan kichik hajmdagi ma’lumotlarga murakkab o’zgarishlarni amalga oshirish bilan bog’liq. Bu yerda kompyuterlar hisoblash murakkab algoritmlar yordamida hisob-kitoblarni tezroq amalga oshirish imkonini berdi. Bunday muammolar birinchi kompyuterlarning yaratilishiga turtki berdi va ularning dolzarbligi hozirgi kungacha davom etmoqda.

Yana bir yo’nalish axborot tizimlarini yaratish bilan bog’liq. Ular murakkab ichki tuzilishga ega bo’lgan katta hajmdagi ma'lumotlarni nafaqat qayta ishlashlari, balki saqlashlari, kerakli ma'lumotlarni tezkor qidirishni ta'minlashlari kerak. Bunday tizimlarni yaratish ishonchli, sig’imli va tez uchuvchan bo’lmagan xotira qurilmalari paydo bo’lgandan keyin mumkin bo’ldi: birinchi navbatda, biz qattiq magnit disklar haqida gapiramiz. Ushbu turdagi tizimlarning klassik namunasi temir yo’l va aviachiptalarni bron qilish tizimlaridir. Har bir buyurtma uchun bajariladigan operatsiyalar ketma-ketligi nisbatan sodda, ammo butun tizimning to’g’ri ishlashi uchun katta hajmdagi ma'lumotlarni saqlash va doimiy ravishda yangilash, ularni qidirish va h.k.

Avtomatlashtirilgan axborot tizimi - har qanday faoliyat turini qo’llab-quvvatlash maqsadida axborotni to’plash, saqlash, yangilash va qayta ishlashni ta'minlaydigan kompyuterga asoslangan kompleks, ya'ni, avtomatlashtirilgan AT muayyan fan sohasi uchun ishlab chiqilgan.

Mavzu sohasi - bu nazoratni tashkil etish maqsadida o’rganilishi kerak bo’lgan real dunyoning bir qismidir. Axborot tizimini yaratish orqali biz ma'lum ma'noda real obyektlarning muhim xususiyatlarini va ularning munosabatlarini tasvirlash imkonini beruvchi axborot modelini shakllantiramiz.

Avtomatlashtirilgan axborot tizimi mustaqil faoliyat ko’rsatishi yoki korxonani boshqarishning avtomatlashtirilgan tizimi kabi murakkabroq tizimning tarkibiy qismi bo’lib xizmat qilishi mumkin. Saqlanadigan va qayta ishlanadigan ma'lumotlar turiga ko’ra, avtomatlashtirilgan axborot tizimlarining ikkita katta sinfi mavjud - hujjatli va faktik.

Hujjatli tizimlar tabiiy tildagi matnlar, maqolalar, ilmiy ma'ruzalar, qonun hujjatlari matnlari va boshqalar bilan ishlash uchun ishlatiladi. Hujjatli tizimlarning eng keng tarqalgan turi bu tabiiy tilda hujjatlarni to’plash va qidirish uchun mo’ljallangan axborot-qidiruv tizimlari hisoblanadi. Ular ba'zan to’liq matnli ma'lumotlar bazalari deb ham ataladi.

Bunday tizimlarda saqlanadigan hujjatlar tizim hujjatlarining qidiruv majmuasini tashkil qiladi. Har bir hujjat uchun qidiruv tasviri shakllanadi - tizim tili bo’yicha hujjatning ma'lum bir rasmiy tavsifi, uning mazmunini aks ettiradi. Masalan, kalit so’zlar to’plamini ko’rsatish orqali qidiruv tasvirini yaratish mumkin. Foydalanuvchi so’rovi so’rovning qidiruv tasviri ko’rinishida ifodalanadi, u saqlangan hujjatlarning qidiruv tasvirlari bilan taqqoslanadi. *Olingan hujjatlar so’rovga tegishli deb* nomlanadi .

Faktli tizimlar avtomatlashtirilgan axborot tizimlarining yana bir katta sinfini tashkil qiladi. Ular maxsus tashkil etilgan yozuvlar to’plami shaklida taqdim etilgan haqiqiy ma'lumotlar bilan ishlaydi. Ushbu kursning asosiy qismi ularga bag’ishlangan, chunki faktik tizimlarda ma'lumotlar bazasi nazariyasi usullari va vositalari to’liq qo’llaniladi. Ma'lumotlar bazasi texnologiyasidan foydalangan holda yaratilgan faktik tizimlar ko’pincha ma'lumotlar banklari deb ataladi (quyida ta'rifga qarang).

Ba'zan, ajratilgan ikkita sinfga qo’shimcha ravishda, leksikografik ma'lumotlar bazalari va axborot tizimlari tushunchasi kiritiladi, ularga turli xil lug’atlar va tasniflagichlar kiradi.

Ma'lumotlar bazasini rivojlantirish sohasidagi lokal normativ hujjatlar quyidagi ta'riflarni beradi.

Ma'lumotlar banki - bu ma'lumotlarning markazlashtirilgan to’planishi va jamoaviy ko’p maqsadli ishlatilishini ta'minlash uchun mo’ljallangan maxsus tashkil etilgan ma'lumotlar (ma'lumotlar bazalari), dasturiy ta'minot, texnik, til, tashkiliy va uslubiy vositalar tizimi.

Ingliz tilidagi adabiyotlarda ma'lumotlar bankiga mohiyatan yaqin bo’lgan tushuncha *ma'lumotlar bazasi tizimi* atamasi bilan belgilanadi .

Ma'lumotlar bazasi - obyektlarning holatini va ma'lum bir mavzu sohasidagi munosabatlarini aks ettiruvchi ma'lumotlar to’plami.

Ma'lumotlar bazasini elektron fayl shkafi, kompyuterga kiritilgan ma'lum ma'lumotlar to’plamining ombori deb hisoblash mumkin. Ma'lumotlar bazasida quyidagi operatsiyalarni bajaring:

- ma'lumotlar bazasiga yangi ma'lumotlarni qo’shish;
- mavjud ma'lumotlarni o’zgartirish;
- ma'lumotlar bazasidan ma'lumotlarni o’chirish;
- ma'lumotlar bazasida ma'lumotlarni topish va hokazo.

Ma'lumotlar bazalari turli xil ma'lumotlar modellari asosida tashkil etilgan. Relyatsion turdagi ma'lumotlar bazasi fragmentiga misol 1.1-jadvalda keltirilgan. Bu holatda ma'lumotlar relyatsion jadvallar shaklida tashkil qilinadi, jadvallar qatorlari yozuvlar, ustunlar esa maydonlar yoki atributlar deb ataladi. Ma'lumotlar bazalarining printsipial jihatdan muhim xususiyati shundaki, ular o’z tuzilishi haqida qo’shimcha xizmat ma'lumotlarini o’z ichiga oladi, boshqacha aytganda, ular o’z-o’zidan hujjatlashtiriladi.

1.1-jadval

Relyatsion ma'lumotlar bazasi fragmenti

StudID	FIO	Guruh
123	Hakimov O.B	382
124	Ravshanov D.K	382

Ma'lumotlar bazasi tizimining komponentlari

Ma'lumotlar bazasi tizimining soddalashtirilgan sxemasini ko’rib chiqamiz (1.1-rasm). U quyidagi asosiy komponentlarni o’z ichiga oladi: ma'lumotlar, apparat, dasturiy ta'minot, foydalanuvchilar.

1.1-rasm. Ma'lumotlar bazasi tizimining umumlashtirilgan diagrammasi

Ma'lumotlar . Ma'lumotlar bazalari doimiy ma'lumotlar to’plamidan iborat. Oraliq natijalar, kirish va chiqish ma'lumotlari kabi *tranzit ma'lumotlari* ham ajralib turadi. *Kirish ma'lumotlari* - tizimga uzatiladigan ma'lumotlar (masalan, klaviaturadan kiritilgan) hisoblanadi. Bunday ma'lumotlar doimiy ma'lumotlarning o’zgarishiga olib kelishi mumkin (u doimiy ma'lumotlarning bir qismiga aylanishi mumkin), lekin ma'lumotlar bazasining o’zi emas. *Chiqish ma'lumotlari* - tizim tomonidan ishlab chiqarilgan xabarlar va natijalardir. Ular odatda doimiy ma'lumotlardan olinadi, ammo ular ma'lumotlar bazasining bir qismi hisoblanmaydi.

Masalan, tizim har 10 daqiqada havo harorati to’g’risidagi ma'lumotlarni qabul qilsin, bir soat uchun o’rtacha qiymat ma'lumotlar bazasida saqlanadi va so’rovda o’rtacha kunlik harorat ko’rsatiladi. Bunday holda, saqlangan qiymatlar kirish va chiqishdan farq qilishi mumkin.

Ma'lumotlar bazasida mavzu sohasini tavsiflovchi ma'lumotlardan tashqari, odatda ma'lumotlar bazasining o’zi elementlari va tuzilmalarini tavsiflovchi ma'lumotlar mavjud. *Bunday tavsiflar meta-axborot* toifasiga kiradi , ya'ni. "ma'lumot haqida ma'lumot". Metaaxborotning markazlashtirilgan ombori *ma'lumotlar lug’ati* yoki *ombori deb ataladi.* Bu ma'lumotlar bazasini o’z-o’zidan hujjatlashtirish xususiyati haqida gapirishga imkon beruvchi omborning mavjudligi. Zamonaviy relyatsion ma'lumotlar bazasida bunday saqlash tizim katalogi - obyektlarning tuzilishi (ma'lumotlar bazalari, jadvallar, ko’rinishlar va boshqalar), foydalanuvchilar, ruxsatlar va boshqalar to’g’risidagi ma'lumotlar kiritilgan xizmat jadvallari to’plami shaklida amalga oshiriladi.

Foydalanuvchi va ma'lumotlar munosabatlari turiga ko’ra, ma'lumotlar bazasi tizimlarining ikki turini ajratish mumkin:

- *yagona foydalanuvchi tizimi* yagona foydalanuvchi tizimi) - ma'lumotlar bazasiga bir vaqtning o’zida faqat bitta foydalanuvchi kirishi mumkin bo’lgan tizim;
- *ko’p foydalanuvchili tizim* ko’p foydalanuvchili tizim) - bu ma'lumotlar bazasiga bir vaqtning o’zida bir nechta foydalanuvchi kirishi mumkin bo’lgan tizim. Shu bilan birga, oxirgi foydalanuvchi uchun shunday shart-sharoitlarni ta'minlash kerakki, uning ishining natijasi uning ma'lumotlar bilan bir

foydalanuvchi rejimida yoki boshqalar bilan birgalikda ishlashiga bog’liq bo’lmasligi kerak.

Ma'lumotlar bazasidagi ma'lumotlar birlashtirilgan va umumiy bo’lishi kerak. Odamlar *birlashtirilgan ma'lumotlar* haqida gapirganda, ular turli manbalardan to’plangan ma'lumotlar unga kirish uchun yagona usulda taqdim etilishini anglatadi. Masalan, tizim universitet bo’limlaridan talabalar faoliyati to’g’risida, kutubxonadan talabalarning adabiyotlardan foydalanishi to’g’risidagi ma’lumotlarni olish va ulardan ba’zi muammolarni hal qilishda birgalikda foydalanish imkonini beradi.

Umumiy ma'lumotlar o’zlarining muayyan muammolarini hal qilish uchun turli foydalanuvchilar guruhlari tomonidan umumiy ma'lumotlar bazasidan alohida ma'lumotlar to’plamidan foydalanish qobiliyatini nazarda tutadi. Masalan, onlayn-do’kon menejeri ma'lum bir buyurtma to’g’risidagi ma'lumotlar bilan ishlashi mumkin va menejer ma'lum bir davr uchun do’kon faoliyatini tavsiflovchi umumiy ma'lumotlar bilan ishlashi mumkin.

Ushbu ikki xususiyat korporativ darajadagi ma'lumotlar bazasi tizimlaridan foydalanishning eng muhim afzalligi bo’lib, "integratsiya" esa ish stoli (shaxsiy) ma'lumotlar bazasi tizimlaridan foydalanishning foydasidir.

Uskuna. Eng umumiy shaklda ma'lumotlar bazasi tizimlari uchun printsipial jihatdan muhim bo’lgan qurilmalarning ikkita guruhini ajratish mumkin:

1) ma'lumotlarni saqlash qurilmalari;

2) ma'lumotlarni qayta ishlash qurilmalari. Kichikroq tizimlar uchun ham qayta ishlash, ham saqlash bir xil kompyuterda amalga oshirilishi mumkin. Katta ma'lumotlar bazasi tizimi ma'lumotlarni qayta ishlash uchun turli xil saqlash tizimlari va ko’plab serverlardan foydalanishi mumkin. Bu yerda taqsimlangan tizimlarning rivojlanishi va ishlashi bilan bog’liq yangi muammolarning butun sinfi paydo bo’ladi.

Dasturiy ta'minot. Fizik ma'lumotlar bazasi va tizim foydalanuvchilari o’rtasida dasturiy ta'minot qatlami mavjud bo’lib, uning asosiy komponenti *ma'lumotlar bazasini boshqarish tizimidir*. ma'lumotlar bazasini boshqarish tizimi.

Ma'lumotlar bazasini boshqarish tizimi - bu ma'lumotlar bazasini yaratish, saqlash va ko’p foydalanuvchilar bilan almashish uchun

mo’ljallangan til va dasturiy vositalar to’plamidir. MBBTning asosiy vazifasi ma'lumotlar bazasi foydalanuvchisiga apparat darajasida tafsilotlarga berilmasdan u bilan ishlash imkoniyatini berishdir.

Ma'lumotlar bazasi tizimi ma'lumotlar bazasi tizimiga qo’shimcha ravishda, qoida tariqasida, bir qator dasturiy komponentlarni o’z ichiga oladi - yordamchi dasturlar, hisobot ishlab chiqaruvchilari, foydalanuvchining amaliy dasturlari va boshqalar.

Foydalanuvchilar. Ma'lumotlar bazasi tizimi foydalanuvchilarini uch sinfga bo’lish mumkin.

Ilova dasturchilari ma'lumotlar bazasidan foydalanadigan amaliy dasturlarni yozish uchun javobgardir. Ular ishlab chiqadigan dasturlar MBBTga so’rovlar beradi va so’rov natijalarini oladi. To’plamli ishlov berish dasturlari va operatsion ilovalar mavjud, ularning vazifasi tizimga interaktiv kirish huquqiga ega bo’lgan oxirgi foydalanuvchining ishini qo’llab-quvvatlashdir.

Yakuniy foydalanuvchilar ma'lumotlar bazasi tizimi bilan bevosita ish stantsiyasidan yoki terminaldan ishlaydi. Ular o’zlari uchun ishlab chiqilgan amaliy dasturlardan yoki o’rnatilgan MBBT vositalaridan (grafik yoki buyruq qatori interfeysi bilan) foydalanishlari mumkin. Ma'lumotlar bazasi tizimi oxirgi foydalanuvchilarning faoliyatini qo’llab-quvvatlash uchun yaratilganligini tushunishingiz kerak.

Ma'lumotlar ma'murlari va *ma'lumotlar bazasi ma'murlari.* Ma'lumotlar administratori - korxona ma'lumotlari uchun javobgar shaxsdir. U ma'lumotlar bazasiga qanday ma'lumotlarni kiritish kerakligi, kim qanday ma'lumotlarga kirishi mumkinligi va hokazolar haqida qaror qabul qiladi. Ba'zan bunday mutaxassislarni tahlilchilar deb atashadi. Ma'lumotlar bazasi ma'muri - ma'lumotlar ma'murining qarorlarini amalga oshirish uchun mas'ul bo’lgan texnik mutaxassis. Tizimni ishlab chiqish bosqichida u ma'lumotlar bazasini yaratish bilan shug’ullanadi; foydalanish bosqichida u konfiguratsiya, texnik xizmat ko’rsatish, zaxiralash va boshqa shunga o’xshash vazifalar bilan shug’ullanadi.

Ma'lumotlar bazasini boshqarish tizimlari va etakchi ishlab chiqaruvchilarning rivojlanish bosqichlari

MBBT paydo bo’lishidan oldin, har bir dasturni ishlab chiquvchilar OS funksiyalaridan foydalangan holda yoki hatto kiritish-chiqarish qurilmalariga bevosita kirish orqali ma'lumotlarni saqlash masalalarini mustaqil ravishda hal qilishdi. Biroq, OT fayllar bilan

ishlash funksiyalarini ta'minlaydi va fayl ichidagi yozuvlarni saqlashni tashkil etish, ma'lumotlarni qidirish va yozish cheklovlarini tekshirish masalalarini OT vositalari yordamida hal qilib bo’lmaydi. Bundan tashqari, bir nechta foydalanuvchi bir vaqtning o’zida bir xil ma'lumotlarga kirish imkoniga ega bo’lganda, ushbu jarayonni markazlashtirilgan tarzda boshqarish uchun qo’shimcha mexanizmlar kerak bo’ladi. Bu va boshqa bir qator sabablar dasturiy ta'minotning alohida sinfi - MBBTning yaratilishiga olib keldi.

Ma'lumotlar bazasini yaratishning birinchi bosqichi "katta" kompyuterlar (meynfreymlar) bilan bog’liq. Birinchi tijorat MBBT IMS (*inglizcha* axborotni boshqarish tizimi, axborotni boshqarish tizimidan) deb nomlangan va IBM tomonidan 1968 yilda IBM System/360 platformasi uchun chiqarilgan. Ushbu bosqich ma'lumotlarni markazlashtirilgan saqlash bilan tavsiflanadi. MBBTlar ma'lumotlar bazasiga jamoaviy kirishni ta'minlashi kerak edi va ular o’zlari murakkab va etarlicha rivojlangan operatsion tizimlar bilan ishlaydigan "katta" mashinalarda ishladilar.

Birinchi bosqichda tadqiqotchilar ma'lumotlar bazasi nazariyasi sohasida muhim natijalarga erishdilar. Xususan, bu ierarxik, tarmoq va relyatsion ma'lumotlar modellarini yaratish. Relyatsion model IBM da ishlagan matematik E. F. Kodd (Edgar Frank Kodd, 1923–2003; 1981 yilda Tyuring mukofotini olgan) tomonidan taklif qilingan. 1970 yilda u "Katta umumiy ma'lumotlar banklari uchun ma'lumotlarning relyatsion modeli" maqolasini nashr etdi, unda u relyatsion yondashuvning asosiy g’oyalarini tasvirlab berdi. Model ustidagi keyingi ishlarda “Ma’lumotlar bazasi tizimlariga kirish” [3|” klassik darsligi muallifi K. Date ham ishtirok etdi. Relyatsion MBBTlar bugungi kunda eng keng tarqalgan.

MBBT rivojlanishining keyingi bosqichi shaxsiy kompyuterlarning paydo bo’lishi bilan bog’liq. Ularning keng qo’llanilishi, hisoblash imkoniyatlarining cheklanganligi va o’rtacha hisobda foydalanuvchilarni tayyorlash darajasining pastligi (eng asosiy kompyuterlar bilan solishtirganda) ish stoli ma’lumotlar bazasining butun sinfi paydo bo’lishiga olib keldi. Dastlab, bular, asosan, bir foydalanuvchiga mo’ljallangan tizimlar bo’lib, imkoniyatlari cheklangan, ammo oddiy foydalanuvchi interfeysi va apparatga talablari past edi. Ularning ko’pchiligi raqobatga dosh bera olmadi va hozir qo’llab-quvvatlanmaydi. Rivojlanish jarayonida qolganlar

ma'lumotlarni almashish va himoya qilish mexanizmlari kabi ko’p foydalanuvchili MBBT xususiyatlarini o’zlashtira boshladilar.

Shu bilan birga, korporativ darajadagi ma'lumotlar bazasida sezilarli o’zgarishlar yuz berdi. Ular kompyuter tarmoqlarining tarqalishi bilan bog’liq edi, buning natijasida mijoz-server texnologiyasi, shu jumladan taqsimlangan ma'lumotlarni qayta ishlashni qo’llab-quvvatlash ustunlik qildi.

Internetning rivojlanishi ma'lumotlar bazasiga ham katta ta'sir ko’rsatdi. Veb-sahifalarni dinamik ravishda yaratishda ko’p hollarda MBBT va ular xizmat ko’rsatadigan ma'lumotlar bazalaridan foydalaniladi. Bu bir qator MBBTlarning paydo bo’lishiga olib keldi, ularning mashhurligi birinchi navbatda veb-ilovalarni yaratishda ulardan foydalanish bilan bog’liq. Eng yorqin misol MySQL relyatsion MBBTdir.

Shu bilan birga, relyatsion MBBT va ular bilan ishlash uchun ishlatiladigan SQL so’rovlar tili barcha vazifalar uchun mos emasligi ma'lum bo’ldi. *NoSQL mafkurasi* paydo bo’ldi va faol rivojlanmoqda , aloqador bo’lmagan ma'lumotlar bazalarini yaratish bilan bog’liq bir qator yondashuvlar va loyihalarni birlashtirgan.

Server MBBTlarining eng mashhur ishlab chiqaruvchisi Oracle korporatsiyasi bo’lib, u 1979-yilda birinchi tijorat relyatsion MBBT Oracle v2 ni chiqargan va shundan beri ma'lumotlar bazasi serverlari sohasida asosiy ishlab chiqaruvchi bo’lib kelgan.

Bozorda muhim o’rinni relational MBBT DB2 va ierarxik MBBT IMS ishlab chiqaruvchi IBM korporatsiyasi egallaydi. 2001 yilda Informix korporatsiyasining bo’linmasini sotib olib, IBM o’z mahsulot qatoriga xuddi shu nomdagi ma'lumotlar bazasini qo’shdi.

Microsoft korporatsiyasi o’zining server mahsuloti MS SQL Server va Microsoft Office paketining bir qismi bo’lgan ish stoli MBBT Access bilan mashhur o’rinni egallaydi. MS SQL Server faqat Windows operatsion tizimlari uchun chiqarilganiga qaramay, ushbu platformaning mashhurligi, Microsoft ishlab chiqish vositalarida qo’llab-quvvatlanishi va MBBTning o’zining keng imkoniyatlari uning keng qo’llanilishiga olib keldi.

1984-yilda tashkil etilgan Sybase-ni relyatsion MBBT ishlab chiqish sohasidagi kashshoflardan biri deb ham atash mumkin. 1980-yillarning oxiri - 1990-yillarning boshlarida. Sybase Microsoft bilan ittifoqda SQL Serverni ishlab chiqdi, ammo keyinchalik mahsulotlar

mustaqil bo’ldi. Bugungi kunda Sybase mahsulot qatoriga Adaptive Server Enterprise relyatsion ma'lumotlar bazasi serveri, o’rnatilgan relyatsion MBBT SQL Anywhere va ma'lumotlarni tahliliy qayta ishlash va ma'lumotlar omborlarini qurish vazifalari uchun mo’ljallangan ustunli ma'lumotlarni saqlash Sybase IQ bilan bog’liq bo’lmagan MBBT kiradi. 2010- yilda Sybase biznes boshqaruvi dasturiy yechimlari bo‘yicha yetakchi provayder SAP AG tomonidan sotib olindi.

Erkin dasturiy ta'minot tarafdorlari orasida dastlab Shvetsiyada yaratilgan MySQL AB kompaniyasi tomonidan ishlab chiqilgan MySQL MBBT keng shuhrat qozondi. Hozirgi vaqtda MySQL veb-ishlab chiqish sohasida qo’llaniladigan MBBT sifatida etakchi mavqega ega. 2008-yilda MySQL AB Sun Microsystems tomonidan, 2010-yilda esa Sunning o’zi Oracle tomonidan sotib olindi. MySQL ning tijorat va bepul versiyalari (MySQL Community Edition) endi mavjud. MySQL-ning MariaDB kabi jamiyat tomonidan ishlab chiqilgan, erkin foydalanish mumkin bo’lgan tarmoqlari ham mavjud.

Shuni ham ta'kidlash kerakki, ko’pgina tijorat ishlab chiquvchilari Oracle Database Express Edition, IBM DB2 Express-C, Microsoft SQL Server Express Edition kabi MBBTning erkin tarqatilgan versiyalariga ega.

Agar obyekt ma'lumotlar modeliga asoslangan MBBT haqida gapiradigan bo’lsak, bugungi kunda ushbu sohadagi eng mashhur loyiha InterSystems tomonidan ishlab chiqilgan Cache tizimidir. Ushbu MBBTning o’ziga xosligi shundaki, u ma'lumotlarning obyektli tasvirini amalga oshiradi, shu bilan birga SQL tilidan relyatsion ma'lumotlar bazasi sifatida ma'lumotlarga kirish imkoniyatini saqlaydi.

Ma'lumotlar bazasi tizimlarining afzalliklari va kamchiliklari

Har bir dastur o’z ma'lumotlarini mustaqil ravishda saqlaydigan vaziyatga nisbatan ma'lumotlarni boshqarishga markazlashtirilgan yondashuvning afzalliklari quyidagilardan iborat:

- *Ortiqchalikni kamaytirish.* Markazlashtirilgan ma'lumotlar bazasi mavjud bo’lmaganda, har bir dastur o’z ma'lumotlarini alohida saqlaydi. Bir xil ma'lumotlarning bir nechta ilovalar tomonidan ishlatilishi odatiy hol emas. Masalan, kadrlar bo’limidagi xodimlar ro’yxatida ham, korxona kutubxonasida qayd etilgan xodimlar ro’yxatida ham familiya, manzil va pasport ma'lumotlari mavjud. Markazlashtirilgan ma'lumotlar bazasida bunday ma'lumotlar

ortiqchalikni to’liq yoki qisman yo’q qilish bilan birlashtirilishi mumkin.

- *Mos kelmaslikni bartaraf etish imkoniyati.* Ko’pincha qarama-qarshiliklar ortiqcha ishlarning natijasidir. Agar bitta shaxs haqidagi bir xil ma'lumotlar ikki xil yozuvda taqdim etilsa va bu "bo’linish" hisobga olinmasa, u holda ertami-kechmi ikkita yozuv bir-biriga mos kelmasligi mumkin: masalan, bitta yozuvga o’zgartirishlar kiritiladi, lekin unga emas. boshqa. Bunday holda, ma'lumotlar bazasi nomuvofiq bo’lib qoladi. Ortiqchalikni yo’q qilish yoki nazorat qilish orqali nomuvofiqliklarning oldini olish mumkin. Ikkinchi holda, bir nechta yangilanishlardan foydalanish mumkin, bu yerda yozuvlardan biriga o’zgartirishlar kiritilganda, u bilan bog’liq yozuvlar avtomatik ravishda o’zgartiriladi.
- *Ma'lumotlarni almashish imkoniyati.* Agar markazlashtirilgan ma'lumotlar bazasi mavjud bo’lsa, turli bo’limlarning xodimlari o’z vakolatlariga muvofiq ushbu ma'lumotlarni almashishlari mumkin.
- *Standartlarga rioya qilish qobiliyati.* Ma'lumotlarni qayta ishlashning yagona standartlarini joriy etish markazlashtirilgan tizimda amalga oshirish uchun ancha osondir.
- *Xavfsizlikni ta'minlash uchun cheklovlarni joriy etish imkoniyati.* Ma'lumotlarni markazlashtirilgan saqlash va qayta ishlash bilan unga kirishni cheklash qoidalarini ishlab chiqish va amalga oshirish osonroq.
- *Ma'lumotlar yaxlitligini ta'minlash qobiliyati.* Butunlikni ta'minlashdan maqsad ma'lumotlar bazasidagi ma'lumotlarning to’g’ri va aniqligini ta'minlashdir. Qarama-qarshiliklarning paydo bo’lishi yaxlitlikning buzilishiga misoldir. Yana bir misol: markazlashtirilgan saqlash bilan ma'lumotlar bazasini zahiralash va tiklash jarayonlarini tashkil qilish osonroq.

Shu bilan birga, ma'lumotlar bazasi tizimini loyihalash va amalga oshirishning muvaffaqiyatsizligi bir qator muammolarni keltirib chiqarishi mumkin. Asosiysi, korxona axborot tizimining butun faoliyati ma'lumotlar bazasi tizimiga bog’liq bo’lishi va undagi nosozliklar halokatli oqibatlarga olib kelishi mumkinligi bilan bog’liq.

Nazorat savollari:

1. Berilganlar bazasi tushunchasi nimadan iborat?
2. Berilganlar bazasi kanday xususiyatlarga ega?
3. Berilganlar bazasi tizimi kanday asosiy tarkibiy qismlarga ega?

4. Berilganlarni berilganlar bazasida namoyon kilganda kanday masalalarni xal qilish zarur?

5. Predmetli soha muammoli sohadan nima bilan farklanadi?

6. Ma’lumotlar berilganlardan nima bilan farklanadi?

7. Berilganlar bazalaridan foydalanishning texnologiyasi nimadan iborat?

8.Berilganlar bazasidagi berilganlarning tarkibi nimaga bog’liq?

9.Berilganlar tarkibini foydalanuvchining talablariga bog’liqligi nimada namoyon bo’ladi?

2-MAVZU. MA’LUMOTLAR BAZASI TIZIMLARI ARXITEKTURASIGA KIRISH.

Reja:

1. Ma’lumotlar bazasi tizimlarining arxitekturasi.
2. Ma’lumotlar bazasi tizimlarining uch bosqichli arxitekturasi.
3. Ko’p foydalanuvchili malumotlar bazalarining arxitekturasi.

Ma'lumotlar bazasi - bu ma'lumotlarga oson kirish, qidirish va manipulyatsiya qilish uchun ma'lum bir tarzda tashkil etilgan ma'lumotlar to’plamidir. Ma'lumotlar bazasi arxitekturasi tizimning umumiy dizaynini, jumladan, ma'lumotlarni fizik saqlashni, ma'lumotlarni mantiqiy tashkil etishni va ma'lumotlarga kirish va ularni boshqarish usullarini anglatadi.

Ma'lumotlar bazasi arxitekturasi ma'lumotlar bazasini boshqarish tizimlarining (MBBT) samaradorligi va samaradorligida muhim rol o’ynaydi. Yaxshi ishlab chiqilgan arxitektura ma'lumotlarga tez va samarali kirish imkonini beradi, shu bilan birga ma'lumotlar yaxlitligi va xavfsizligini ta'minlaydi.

Ma'lumotlar bazasining fizik arxitekturasi

Ma'lumotlar bazasi tizimlarining arxitekturasi quyidagi tarkibiy qismlarga ega bo’lishi mumkin:

- *Fizikaviy qurilish (Physical Layer*): Bu qism ma'lumotlar bazasidagi ma'lumotlarni diskda, xotira qurilmalari yoki boshqa o’zgaruvchilar orqali saqlashning fizik shakllanishini ta'minlaydi.
- *Qo’llanma (Query Processor)*: Bu qism so’rovlarni qabul qiladi va ularni bajarish uchun ma'lumotlar bazasiga tegishli operatorlarga aylantiradi. Bu qism ham yuzlab so’rovlarni tahlil qiladi va ularni samarali bajarish uchun boshqaruvchi qatlamga murojaat qiladi.
- *Ma'lumotlar modeli (Data Model)*: Bu qism ma'lumotlar bazasidagi ma'lumotlar turlarini, ularning mos kelishuvlarini va ularga amal qilish protseduralarini aniqlaydi.
- *Boshqaruv tili (Data Definition Language, Data Manipulation Language)*: Bu tizim ma'lumotlar bazasidagi ma'lumotlar strukturasi va ma'lumotlarga amal qilishni ta'minlaydi.

- *Amal so’rovlari optimallashtirish (Query Optimization):* Bu qism ma'lumotlar bazasidagi so’rovlarni samarali bajarish uchun ularni optimallashtiradi.
- *Xavfsizlik (Security)*: Bu qism ma'lumotlarga kirishni boshqarish va ma'lumotlarni himoya qilishni ta'minlaydi.
- *Ma'lumotlar bazasining yuklanish va taqsimoti (Installation and Distribution)*: Bu qism ma'lumotlar bazasini o’rnatish, sozlash va ma'lumotlarni taqsim etishni ta'minlaydi.
- *Replikatsiya va rejim qo’llab-quvvatlash (Replication and Fault Tolerance):* Bu qism ma'lumotlarning ko’chirilishi, nusxalanishi va ma'lumot bazasidagi xatoliklarga qarshi qo’llab-quvvatlashni ta'minlaydi.
- *Monitoring va boshqarish (Monitoring and Management):* Bu qism ma'lumotlar bazasining ishlashi va sodir bo’lgan xatolar, tarozililiklar, va takomillar haqida ma'lumotlar beradi.
- *Ma'lumotlar bazasining integratsiyasi (Integration):* Bu qism ma'lumotlar bazasining boshqa tizimlar bilan integratsiyasini ta'minlaydi, masalan, ilova dasturlari yoki boshqa ma'lumotlar bazalari bilan bog’lanish.

Ma'lumotlar bazasining fizik arxitekturasi ma'lumotlarni qattiq disklar yoki qattiq disklar kabi saqlash qurilmalarida fizik saqlash usulini anglatadi. Fizik arxitektura ma'lumotlar bloklarining hajmi va tartibini, ma'lumotlar fayllarini tashkil qilishni, ma'lumotlarga tezkor kirishni ta'minlash uchun indekslash va boshqa ma'lumotlar tuzilmalaridan foydalanishni o’z ichiga oladi.

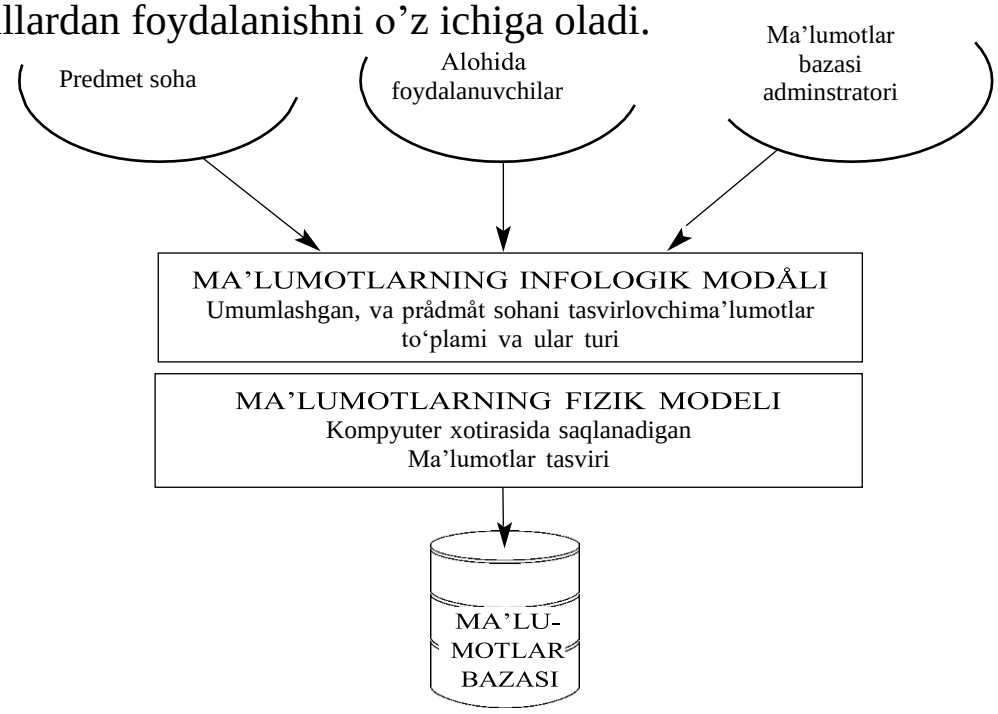
Ma'lumotlar bazasining fizik arxitekturasi ma'lumotlarni qattiq disklar yoki qattiq disklar kabi saqlash qurilmalarida fizik saqlash usulini anglatadi. Fizik arxitektura ma'lumotlar bloklarining hajmi va tartibini, ma'lumotlar fayllarini tashkil qilishni, ma'lumotlarga tezkor kirishni ta'minlash uchun indekslash va boshqa ma'lumotlar tuzilmalaridan foydalanishni o’z ichiga oladi.

Fizik ma'lumotlar bazasi arxitekturasining asosiy maqsadlaridan biri ma'lumotlarga kirish uchun zarur bo’lgan vaqtni minimallashtirishdir. Bunga ma'lumotlarni saqlash va qidirish mexanizmlarini optimallashtirish, masalan, tezkor kirish vaqtiga ega disk drayverlarini ishlatish, diskni qidirish vaqtlarini minimallashtirish va diskdan o’qilishi kerak bo’lgan ma'lumotlar miqdorini kamaytirish uchun siqish algoritmlarini qo’llash orqali erishiladi.

Mantiqiy ma'lumotlar bazasi arxitekturasi

Ma'lumotlar bazasining mantiqiy arxitekturasi ma'lumotlarni tashkil qilish va ilovalar tomonidan kirish usulini anglatadi. Mantiqiy arxitektura ilovalarga ma'lumotlarning yagona ko’rinishini taqdim etish uchun javobgardir, shu bilan birga ma'lumotlar izchil va xavfsiz bo’lishini ta'minlaydi.

Yaxshi ishlab chiqilgan mantiqiy ma'lumotlar bazasi arxitekturasi ishlab chiquvchilarga murakkab ma'lumotlar munosabatlarini to’g’ridan-to’g’ri ko’rsatishga imkon beradigan ma'lumotlarni modellashtirish usullarini qo’llab-quvvatlashi kerak. Bu ma'lumotlarning aniq, yuqori darajadagi ko’rinishini yaratish uchun obyektlar o’rtasidagi munosabatlar diagrammasi (ERD) va boshqa usullardan foydalanishni o’z ichiga oladi.



2.1-rasm. MBBT arxitekturasi.

Ma'lumotlarga kirish arxitekturasi

Ma'lumotlarga kirish arxitekturasi ilovalar ma'lumotlar bazasidagi ma'lumotlarga kirishi mumkin bo’lgan mexanizmlarni anglatadi. Bunga ma'lumotlarni olish va manipulyatsiya qilish uchun Strukturaviy so’rovlar tili (SQL) va boshqa so’rovlar tillaridan foydalanish, shuningdek, ilovalarning ma'lumotlar bazasi bilan o’zaro ishlashiga imkon beruvchi API va boshqa dasturlash interfeyslari kiradi.

Ma'lumotlarga kirish arxitekturasining asosiy maqsadlaridan biri ma'lumotlarga samarali va xavfsiz kirishni ta'minlash, shu bilan birga

ma'lumotlar yaxlitligini ta'minlashdir. Bunga foydalanuvchi autentifikatsiyasi va avtorizatsiyasi kabi xavfsizlik mexanizmlari, shuningdek, ma’lumotlarga kirmasligi yoki nomuvofiq tarzda o‘zgartirilmasligini ta’minlash uchun tranzaktsiyalarni boshqarish va blokirovkalash mexanizmlaridan foydalanish orqali erishiladi.

Ma'lumotlar bazasini boshqarish tizimi arxitekturasi

Ma'lumotlar bazasini boshqarish tizimining (MBBT) arxitekturasi tizimning umumiy dizaynini, shu jumladan fizik va mantiqiy arxitekturani, shuningdek, ma'lumotlarga kirish va boshqarish mexanizmlarini anglatadi. Yaxshi ishlab chiqilgan MBBT arxitekturasi ma'lumotlarni samarali va xavfsiz saqlash, qidirish va manipulyatsiya qilishni qo’llab-quvvatlashi kerak.

MBBT arxitekturasining asosiy xususiyatlaridan biri uning katta hajmdagi ma’lumotlarni masshtablash va ishlov berish qobiliyatidir. Bunga taqsimlangan ma'lumotlar bazalari, sharding va ma'lumotlarni bir nechta serverlarda saqlash va parallel ravishda kirish imkonini beruvchi boshqa usullardan foydalanish orqali erishiladi.

Ma'lumotlar bazasi arxitekturasi ma'lumotlar bazasini boshqarish tizimlarining samaradorligi va samaradorligida muhim rol o’ynaydi. Yaxshi ishlab chiqilgan arxitektura ma'lumotlarga tez va samarali kirish imkonini beradi, shu bilan birga ma'lumotlar yaxlitligi va xavfsizligini ta'minlaydi. Fizik, mantiqiy va ma'lumotlarga kirish arxitekturalari ma'lumotlar bazasining umumiy samaradorligiga hissa qo’shadi, ma'lumotlar bazasi arxitekturasi esa kengaytiriladigan va samarali ma'lumotlarni boshqarish uchun asos yaratadi. Murakkab ma'lumotlar tizimlarini loyihalash va boshqarish bilan shug’ullanadigan har bir kishi uchun ma'lumotlar bazasi arxitekturasini to’liq tushunish juda muhimdir.

Ma’lumotlar bazasi arxitektura modellari

Ma'lumotlar bazasi arxitekturasi modellari ma'lumotlar bazasi tizimini ma'lum talablarga javob beradigan tarzda tuzilishi mumkin bo’lgan turli usullarga ishora qiladi. Ushbu modellar ma'lumotlar bazasi tizimlarini loyihalash, ishlab chiqish va boshqarish uchun asos yaratadi. Ma’lumotlar bazasi arxitekturasi modellarining bir necha turlari mavjud bo‘lib, ularning har biri o‘ziga xos xususiyat va afzalliklarga ega.

Ierarxik model ma'lumotlar bazasi arxitekturasining ilk modellaridan biri bo’lib, u bugungi kunda ham ba'zi eski tizimlarda

qo’llaniladi. Ushbu model ma'lumotlarni daraxtga o’xshash tuzilishda, ma'lumotlar elementlari orasidagi ota-ona munosabatlari bilan tartibga soladi. Ierarxik model qat'iy va bashorat qilinadigan tuzilishga ega bo’lgan ma'lumotlarni saqlash uchun foydalidir. Biroq, u ma'lumotlar tuzilmasidagi o’zgarishlarga moslashish yoki ma'lumotlar elementlari o’rtasidagi murakkab munosabatlarni boshqarish uchun etarlicha moslashuvchan emas.

Tarmoq modeli ierarxik modelning cheklovlarini hal qilish uchun ishlab chiqilgan. Ushbu model ma'lumotlar elementlari o’rtasida bir nechta ota-ona munosabatlariga imkon beruvchi yanada moslashuvchan sxemadan foydalanadi. Bu ma'lumotlar elementlari orasidagi murakkab munosabatlarni boshqarishni osonlashtiradi. Shu bilan birga, tarmoq modeli ham ierarxik modelga qaraganda murakkabroq va uni saqlab qolish qiyin bo’lishi mumkin.

Relyatsion model bugungi kunda eng ko’p qo’llaniladigan ma'lumotlar bazasi arxitekturasi modelidir. Ushbu model ma'lumotlarni jadvallarda tartibga soladi, har bir jadval ma'lum bir obyektni ifodalaydi va jadvaldagi har bir satr ushbu obyektning noyob namunasini ifodalaydi. Relyatsion model ma'lumotlarni manipulyatsiya qilish va olish uchun tuzilgan so’rovlar tilidan (SQL) foydalanadi. Ushbu model moslashuvchan, kengaytiriladigan va parvarish qilish oson. Shuningdek, u ma'lumotlar elementlari orasidagi murakkab munosabatlarni boshqarish uchun mo’ljallangan.

Obyektga yo’naltirilgan model obyektga yo’naltirilgan dasturlash tamoyillariga asoslanadi. Ushbu model ma'lumotlarni obyektlarda saqlaydi, har bir obyekt ma'lum bir sinfning noyob namunasini ifodalaydi. Obyektga yo’naltirilgan model ma'lumotlar elementlari o’rtasidagi murakkab munosabatlarga imkon beradi va u murakkab tuzilmalar va xatti-harakatlarga ega bo’lgan ma'lumotlarni qayta ishlash uchun foydalidir. Biroq, u boshqa modellarga qaraganda ancha murakkab va uni amalga oshirish qiyin bo’lishi mumkin.

NoSQL modeli katta hajmdagi tuzilmagan yoki yarim tuzilgan ma'lumotlar bilan ishlash uchun mo’ljallangan nisbatan yangi ma'lumotlar bazasi arxitekturasi modelidir. Ushbu model relyatsion model kabi qat'iy sxemadan foydalanmaydi, bu esa ko’proq moslashuvchanlik va kengayish imkonini beradi. NoSQL ma'lumotlar bazalari ko’pincha katta ma'lumotlar dasturlarida qo’llaniladi, bu yerda

qayta ishlanadigan ma'lumotlar miqdori an'anaviy relyatsion ma'lumotlar bazasiga sig’ish uchun juda katta.

Taqdimot qatlami arxitekturaning eng yuqori qatlami bo’lib, foydalanuvchi interfeysi uchun javobgardir. U mijoz qatlami sifatida ham tanilgan, chunki u foydalanuvchilar bilan bevosita aloqada bo’ladi. Ushbu qatlam foydalanuvchiga ma'lumotlarni ko’rsatish va foydalanuvchidan kiritilgan ma'lumotlarni yig’ish uchun javobgardir. Taqdimot qatlami veb-server, veb-brauzer yoki ish stoli ilovasi yordamida amalga oshiriladi.

Ilova qatlami arxitekturaning o’rta qatlami bo’lib, foydalanuvchi so’rovlarini qayta ishlash uchun javobgardir. U taqdimot qatlami va ma'lumotlar qatlami o’rtasida vositachi vazifasini bajaradi. Ilova qatlami biznes mantig’ini bajarish, foydalanuvchi so’rovlarini qayta ishlash va taqdimot qatlami uchun ma'lumotlarni tayyorlash uchun javobgardir. Ilova qatlami PHP, ASP.NET yoki Java kabi server tomonidagi texnologiyalar yordamida amalga oshiriladi.

Ma'lumotlar qatlami arxitekturaning eng pastki qatlami bo’lib, ma'lumotlarni saqlash va olish uchun javobgardir. Ushbu qatlam ma'lumotlar bazasi qatlami sifatida ham tanilgan. Ma'lumotlar qatlami ma'lumotlarning yaxlitligini ta'minlash, biznes qoidalarini qo’llash va amaliy qatlamga ma'lumotlarga kirish xizmatlarini taqdim etish uchun javobgardir. Ma'lumotlar qatlami Oracle, SQL Server yoki MySQL kabi ma'lumotlar bazasini boshqarish tizimi (MBBT) yordamida amalga oshiriladi.

Ma'lumotlar bazasining uch bosqichli arxitekturasi veb-ilovalarni ishlab chiqishda keng qo’llaniladigan dasturiy ta'minot arxitekturasi namunasidir. Arxitektura uchta qatlamdan iborat: taqdimot qatlami, amaliy qatlam va ma'lumotlar qatlami. Har bir qatlam ma'lum bir vazifalar to’plami uchun javobgardir, bu tizimni boshqarish va saqlashni osonlashtiradi. Ma'lumotlar bazasining uch bosqichli arxitekturasi bir qator afzallik va kamchiliklarga ega va ushbu arxitekturani qabul qilishdan oldin ularni ko’rib chiqish juda muhimdir.

Ilova qatlami: Ilova qatlami ma'lumotlar bazasining uch bosqichli arxitekturasining ikkinchi qatlamidir. Ushbu qatlam foydalanuvchi so’rovlarini qayta ishlash va ma'lumotlar bazasi operatsiyalarini bajarish uchun javobgardir. Ilova qatlami, shuningdek, foydalanuvchi kiritgan ma'lumotlarni tekshirish va foydalanuvchining ma'lumotlarga kirish huquqini ta'minlash uchun javobgardir. Ilova qatlami Java, PHP,

ASP.NET va Python kabi turli dasturlash tillari va ramkalar yordamida amalga oshirilishi mumkin.

Nazorat savollari:

1. Ma'lumotlar bazasi tashqi qanday tashkil etilgan? Ular qanday tashqi foydalanuvchilarga xizmat ko’rsatadi?
2. Ma'lumotlar bazasi ichki qanday tashkil etilgan? Bu qanday ma'lumotlar bazasini saqlash, ma'lumotlarga kirishni tashkil etadi?
3. Ma'lumotlarga kirish va ularga erkin kirish qanday ta'minlangan?
4. Ma'lumotlar bazasi tizimlararo aloqalari qanday tuzilgan?
5. Ma'lumotlarni qanday o’lchamda olib kelish mumkin?
6. Ma'lumotlarni saqlash va ularga murojaat qilish qanday amalga oshiriladi?
7. Ma'lumotlar bazasi xavfsizligi va himoyalash uchun qanday chora-tadbirlar mavjud?

3-MAVZU. MA'LUMOTLAR MODELLARI VA MA'LUMOTLAR BAZASI MODELLARI.

Reja:

1. Ierarxik ma'lumotlar modellari.
2. Tarmoq ma'lumotlar modeli.
3. Tarmoqli semantik modellar

Ma’lumotlar modeli - bu ma’lumotlar o’zaro bog’langan tuzilishlari va ular ustida bajariladigan operatsiyalar to’plamidir.

Ma’lumotlar modeli quyidagi tarkibiy qismlarni o’z ichiga oladi:

1. Ma’lumotlar tuzilmasi.
2. Ma’lumotlar tuzilishida bajarilish mumkin bo’lgan operatsiyalar.
3. Yaxlitlikni nazorat qilish uchun cheklashlar.

MBni tuzishda ma’lumotlar modeli bir yoki bir necha turidagi obektlarni o’z ichiga olish va ular o’rtasida aloqalar o’rnatish imkonini yaratadi.

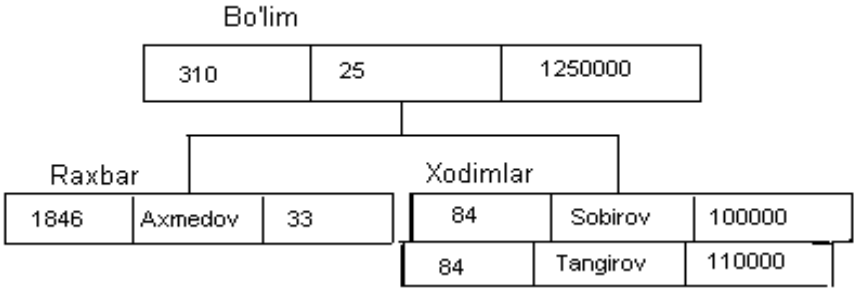
Mashina muhitidagi ma’lumotlarning murakkabroq modellari (fayl modeliga nisbatan) ***tarmoqli*** **va** ***iyerarxik*** modellar bo’lib hisoblanadi. Bu modellar ularning o’zlariga xos turdagi ma’lumotlar bazasini boshqarish tizimida ishlatiladi. MBBTda ma’lumotlarni mantiqiy tashkil etish usuli ma’lumotlarning tarmoqli yoki iyerarxik modeliga mos holda ko’rsatiladi. Bunday model o’zaro bog’lik obyektlarning majmui bo’lib ikki obyektning aloqasini va ularning bir-biriga qaramligini aks ettiradi.

Iyerarxik model

Ikki xildagi obyektlar o’rtasidagi aloqalar, ularning nusxalari o’rtasidagi qat’iy iyerarxik munosabatlar bilan aniqlanib, ular asosiy va tobe yozuvlar to’plamidan iboratdir.Daraxt turiga misol:

3.1-rasm

Bunda yozuv turlari orasida bog’lanish mavjud. Bunday sxemadagi ma’lumotlar bazasi quyidagi ko’rinishda tasvirlanadi (daraxtning bitta nusxasi):

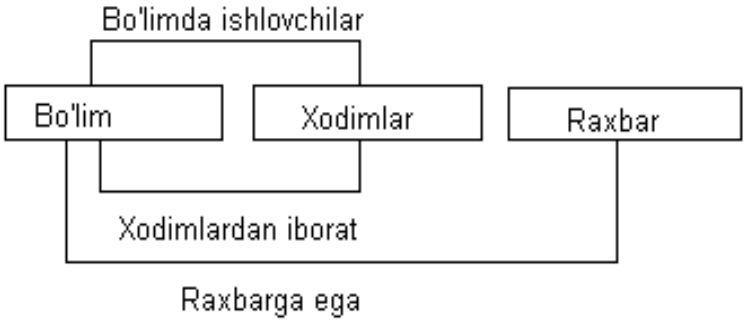


3.2-rasm

Tarmoqli model

Turli predmet sohalari uchun ma’lumotlarning tarmoqli modeli iyerarxik modeliga nisbatan mashinaning ish muhitida axborot tuzilmalarini aks ettiruvchi umumiy vosita hisoblanadi. Ko’plab predmet sohalarining malumotlari o’rtasidagi aloqalar tarmoqli ko’rinishga ega bo’ladi. Tarmoqi modellar, ma’lumotlarning iyerarxik aloqasini ham aks ettirishga imkon beradi.

Tarmoqli MBda yozuvlar va ular orasidagi bog’lanishlar tashkil topgan, ya’ni yanada aniqrog’i MB strukturasining har bir tipidagi nusxalar to’plamidagi yozuvlar turi to’plami va berilgan aloqa turlari to’plamidagi har bir turdagi nusxalar to’plamuni tashkil qiladi. MB tarmoqli sxemasiga oddiy misol:



3.3-rasm

Relyatsion modeli

Ma’lumotlarning relyatsion modeli birinchi bor 1970-yilda ye.F.Kodd tomonidan taklif qilingan bo’lib, u ma’lumotlarni tavsiflash va tasvirlashning amaliy dasturlaridan bog’liq bo’lmasligini ta’minlash masalasini hal qilishga xizmat qiladi.

Ma’lumotlarning relyatsion modeli asosida «munosabat» tushunchasi yotib, u inglizcha relation so’zidan olingan. Ba’zi bir qoidalarga amal qilgan holda munosabatlarni ikki o’lchovli jadval ko’rinishda tasvirlash mumkin. CHunki jadval har qanday odamga tushunarli va qulaydir.

Real dunyo obyektlari haqidagi ma’lumotlarini EHM xotirasida saqlash va ular orasidagi aloqalarni modellashtirish uchun munosabatlar (jadval) to’plamidan foydalanish mumkinligini ye.F.Kodd isbotlab berdi. Masalan, «talaba» mazmunini saqlash uchun TALABA munosabatidan foydalaniladi. Bu mazmunning asosiy xususiyatlarini quyidagi jadvalning ustunlari tasvirlaydi:

TALABA

Familiyasi I.O.	**Tug’ilgan sana**	**Bosqich**	**Mutaxassisligi**
Karimov M.N.	15/01/1979	4	menejement
Avazov A.V	03/11/1978	4	Iqtisodiy inf.
Ortiqov O.B.	07/07/1980	3	Iqtisodiy inf.
Lazizova V.N.	12/04/1981	3	Iqtisodiy pedag.
Xoliyorov N.A.	31/12/1982	2	Marketing
Javlonov X.B.	24/09/1983	1	Sug’o’rta ishi

Munosabat ustunlari atributlar (maydonlar) deb ataladi va ularga nomlar beriladi. Munosabat atributlarining nomlaridan iborat tuzilgan ro’yxatga munosabatlar sxemasi deyiladi. Misoldagi TALABA munosabatining sxemasi quyidagicha yoziladi:

TALABA = (Familiyasi I.O., Tug’ilgan sana, Bosqich, Mutaxassisligi)

Ma’lumotlarning relyatsion bazasi - bu o’zaro bog’langan munosabatlar to’plamidir. Har qanday munosabat (jadval) kompyuterlarning xotirasida fayl ko’rinishda joylashadi.

Jadvalda ushbu uch operatsiya bir-biri bilan chambarchas bog’langan. Bu esa ba’zi bir operatsiyalarni bajarishda ma’lum bir qiyinchiliklarga olib keladi. Masalan, ma’lumotlarni bir parametr asosida tartiblash ikkinchi bir parametr bo’yicha tartiblashni buzib yuborishi tufayli zarur ma’lumotlarni izlab topish operatsiyasi bir parametr bo’yicha osonlashsa, boshqalari bo’yicha qiyinlashadi.

Ma’lumotlarning relyatsion bazasidagi munosabatlar ustida bajariladigan asosiy operatsiyalar sakkizta bo’lib, ular quyidagilardan iborat:

- to’plamlar ustidagi ananaviy operatsiyalar, ya’ni to’plamlarning birlashmasi (yig’indisi), kesishmasi (ko’paytmasi), to’ldiruvchisi (ayirmasi), dekart ko’paytmasi, bo’lishmasi;

- maxsus relyatsion operatsiyalar, ya’ni proyeksiyalash, bog’lanish (qo’shilish), birlashtirish (ulab qo’yish) va tanlash.

Relyatsion MBBTda munosabatlar ustida operatsiyalar bajarish uchun mo’ljallangan tillarni ikki sinfga ajratish mumkin: relyatsion algebra tili (RAT) va relyatsion hisob tili (RHT).

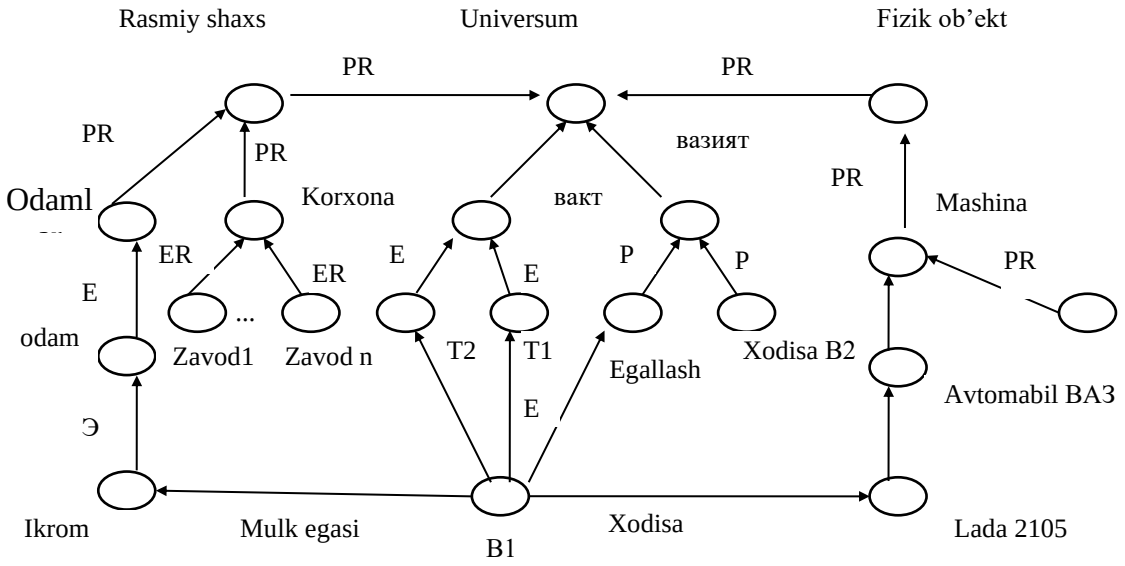
RAT relyatsion algebraga (Kodd algebrasiga, α-algebraga) asoslangan. Ma’lum tartib munosabatlar ustida operatsiyalarni ketma-ket yozish asosida hohlagan natijaga erishish mumkin. SHuning uchun RATni protsedurali til deyishadi.

RHT predikatlarni hisoblab chiqishning klassik usuliga asoslangan. Ular foydalanuvchilarga so’rovlarni yozish uchun ma’lum qoidalar to’plamini beradi. Ushbu so’rov asosida MBBT yangi munosabatlar hosil qilish yo’li bilan avtomatik tarzda zarur natijani oladi. Shu sabab RHTga protseduralimas til deyishadi.

Tarmoqli semantik modellar

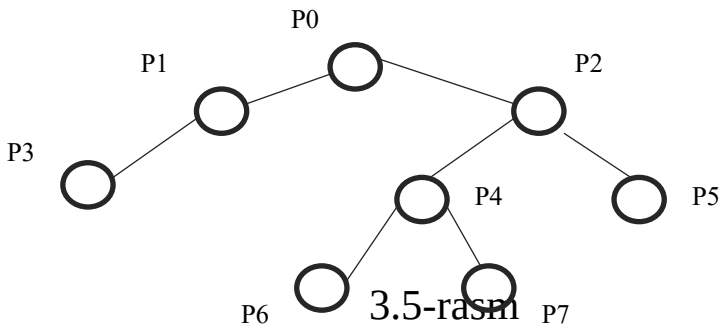
Bu modellarning asosida tarmoqlar, yoylar tushunchasi yetadi.Tarmoqlar oddiy va iyerarxik bo’ladi, unda cho’qqilar - bu ba’zi bir tushunchalar, mohiyatlar, obyektlar, vokealar, jarayenlar yoki xodisalardir. Bu mohiyatlar o’rtasidagi munosabatlar yoylar bilan aks ettiriladi. Abstrakt yoki anik obyektlar odatda tushunchalar bo’ladilar, munosabatlar esa bu («bu» («is»), “bir qismi bo’ladi”, “tegishli”, “sevadi” kurinishidagi aloqalar. Oddiy tarmoqlar ichki tuzilishga ega emas, iyerarxik tarmoqlarda esa ba’zi bir cho’qqilar ichki tuzilishga ega. Munosabatlarning quyidagi uchta turini mavjudligi semantik tarmoqlarning xarakterli xususiyatlari bo’ladi:

- sinf - sinfning elementi
- xususiyat - ma’no
- sinf elementining misoli



3.4-rasm

Iyerarxik semantik tarmoqlarda tarmoqlarni tarmoqchalarga bo’linishi ko’zda tutiladi va munosabatlar nafaqat cho’qqilar o’rtasida, balki tarmoqchalar o’rtasida ham urnatiladi.



3.5-rasm

Yuqoridagi keltirilgan chizmada joylarning daraxti aks ettirilgan. R6 joyi uchun R4, R2, R0 joylari chegarasida yetgan joylarning barcha cho’qqilari ko’rinadi, qolganlari esa ko’rinmaydi.

Iyerarxik tarmoqlarni grafik tasvirlarining qoidalari yoki bitimlarini kurib chikamiz.

1. Bir joyda yetgan cho’qqilar va yoylar tugri turt burchak yoki kup burchak bilan cheklanadilar.
2. Ey uning nomi yetgan joyga tegishli bo’ladi.
3. Rj joyning ichida tasvirlagan Ri joyi avlod (ichki daraja) xisoblanadi, ya’ni Ri dan Rj “kurinadi”. Ri Rj da yetgan super cho’qqi sifatida kurib chiqilishi mumkin.

Semantik tarmoq kurinishidagi bilimlar bazasida yechimni kidirish muammosi kuyilgan savolga mos keluvchi ba’zi bir

tarmoqgacha tegishli tarmoq qismini kidirish masalasidan iborat bo'ladi.

Tarmoqli semantik tarmoqning asosiy avzalligi - insonning uzok muddatli xotirasini tashkil qilish xakidagi zamonaviy tasavvurga mos kelishidadir. Modellarning kamchiligi - semantik tarmoqka chikishni kidirishning murakkabligidir.

Mohiyat-bog'lanish tushunchasi

Infologik modellashtirishning maqsadi – tuziladigan ma'lumotlar bazasida shakllanishi mumkin bo'lgan ma'lumotlarni tasvirlash va yig'ish usullarini odamlar uchun ayniqsa tabiiy ta'minlashdir. Shuning uchun ma'lumotlarning infologik modelini tabiiy tilga mos qilib qurishga harakat qilinadi.

Infologik modelni qurishning asosiy konstruktiv elementlari bu:

- mohiyat;
- bog'lanish;
- xossalar (atributlar)dir.

Mohiyat -bu ma'lumotlari ma'lumotlar bazasida saqlanishi kerak bo'lgan biror real yoki tasavvur qilingan obyekt bo'lib, aniq ma'noga va nomga egadir, u yagona bo'lishi kerak. Mohiyat turini uning nusxasi bilan farq qilish kerak. Mohiyat nomi uning nusxasiga emas, turiga beriladi. *Mohiyat nusxasi* - bu aniq bir xil turdagi narsalar, hodisalar va boshqalardir.

Masalan, "O'quvchi" mohiyatida "O'quvchi" mohiyat turining nomi, mohiyat nusxasi esa aniq bir o'quvchidir. Masalan, Axmedov, Toshmatov va boshqa. "Shahar" mohiyatida "Shahar" mohiyat nomi, uning nusxasi esa aniq bir shaharlardir. Masalan, Toshkent, Samarqand, Buxoro va boshqa.

Bog'lanish – bu ikki yoki undan ortiq mohiyatlarning bir-biri bilan o'zaro bog'lanishidir. Bog'lanish faqat ikkita har xil mohiyatlar orasida mavjud bo'ladi.

Atribut(xossa) - mohiyatni xarakterlovchi nomlardir. U o'zida yagona murakab bo'lmagan strukturani tasvirlab, mohiyat holatini xarakterlaydi. Masalan, "O'quvchi" mohiyati atributi - kod, familiya, ism, yosh va boshqalar bo'lishi mumkin.

Ma'lumotlar bazasini loyihalashda har xil semantik modellar ham ishlatiladi. Ulardan eng ko'p tarqalganlaridan biri - **ER** modeldir. Bu

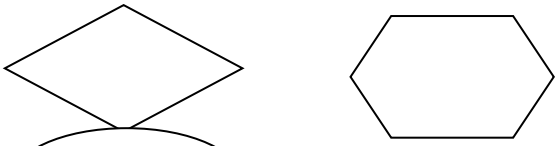
model ingilizcha **“Entity-relation”** deyilib, ma’nosi **“Mohiyat-bog’lanish”** demakdir.

Bu model 1976 yil Piter CHen tamonidan kiritilgan bo’lib u o’ziga bir qator grafik diagrammalarni oluvchi bir necha turdagi komponentalarni birlashtirgan. ER modeli diagrammasida mohiyat odatda to’rtburchak shaklida tasvirlanib, uning ichiga mohiyat nomi yoziladi.

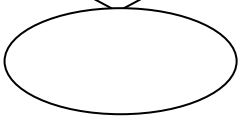
Masalan:

ER - modelining asosiy komponentalari mohiyat, bog’lanish va atribut bo’lib hisoblanadi. Infologik modelni qurishda ER diagramma tili ishlatiladi va unda quyidagi bulgilardan foydalaniladi:

- mohiyat



- assotsiatsiya (birlashish)



- atribut

- bog’lanish, uning ustida darajasi ko’rsatiladi

Nazorat savollari:

1. Ma’lumotlar bazasi nima?
2. Ma’lumotlar bazasi qanday xossalarga ega bo’lishi kerak?
3. MB jadvallari orasida qanlay bog’lanishlar bor?
4. ER modeli asoschisi kim?
5. ER modeli ma’nosi nima?
6. Mohiyat va bog’lanish nima va u qanday tasvirlanadi?
7. Atribut nima?
8. Relyatsion modellarga misollar keltiring.
9. Tarmoqli semantik modellarga misollar keltiring.

4-MAVZU. RELYATSION MA'LUMOTLAR MODELI

Reja:

1. Relyatsion ma'lumotlar modeli
2. Yaroqli axborot tuzilmalari
3. Ma'lumotlar yaxlitligi chegaralari

Relyatsion modelda ma'lumotlar tuzilmalarini ko’rib chiqishda bir qator aniq tushunchalar kiritiladi. Dastlabki ko’rib chiqish uchun biz ularni allaqachon ma'lum bo’lgan atamalar bilan aniqlaymiz.

O’zaro bog’liqlik taxminan jadvalga mos keladi, farqlar quyida muhokama qilinadi. *Kortej* jadval qatoriga, *atribut ustunga* mos keladi. *Domen* ma'lum turdagi qiymatlarning umumiy to’plamidir (masalan, shahar nomlari domeni).

Keling, taqdim etilgan tushunchalarni batafsilroq ko’rib chiqishga o’tamiz. *Ma'lumotlarning eng kichik semantik birligini skaler* deb ataymiz. Skalyar qiymatlar mantiqan bo’linmaydi. Masalan, o’quvchilarni tavsiflovchi jadvaldagi "Guruh raqami" qiymati, garchi uni alohida belgilarga bo’lish mumkin bo’lsa-da, o’z ma'nosini yo’qotadi.

Domenni bir xil turdagi skalyar qiymatlar to’plami sifatida aniqlash mumkin, bu atribut qiymatlari olinadigan qiymatlar to’plamidir. Har bir atribut bitta domenda aniqlanishi kerak. Relyatsion modeldagi domenlarning ahamiyati shundaki, ular taqqoslashni cheklaydi, ya'ni. Faqat bitta domenda belgilangan atributlarning qiymatlarini solishtirish mantiqiy. Masalan, A qismining og’irligi va B qismining o’lchamini solishtirishning ma'nosi yo’q, lekin buni rasmiy ravishda aniqlash uchun domenlar kiritiladi.

Haqiqiy ma'lumotlar bazasida domenlar qiymatlar to’plami sifatida saqlanmaydi. Ammo rasmiy ravishda relyatsion model domenlarni aniqlashni talab qiladi va atributni belgilashda tegishli domen ko’rsatilishi kerak.

Relyatsion o’zgaruvchi - bu vaqt o’tishi bilan qiymati o’zgarishi mumkin bo’lgan relyatsion munosabat bilan bog’langan nomli obyektdir. Ushbu o’zgaruvchining ma'lum bir momentdagi qiymati nisbatning qiymati deb ataladi.

Munosabatlar sxemasi deb ham ataladigan munosabat *boshi* <atribut_nomi: domen_nomi> ko’rinishidagi qat'iy juftliklar to’plamini o’z ichiga oladi. Masalan:

Asosiy raqam - bu munosabatlardagi kortejlar sonidir. *Aloqa darajasi* - atributlar soni (bizning misolimizda raqam va).

Masalan, Talabalar munosabati quyida keltirilgan (4.1-jadval).

4.1-jadval

Talabalarning munosabati

Chipta soni	**To’liq ism**	**Guruh**
928012	To’rayev I.T	3082/4
928189	Xurshidov R.H	3082/4

Keltirilgan misolda nisbatning asosiy raqami 2, daraja 3. Shuningdek, uchta domen aniqlangan deb taxmin qilinadi - talaba ID raqamlari domeni, bosh harflar bilan familiyalar domeni va guruh raqamlari domenlari. Har bir domen tegishli atribut uchun barcha mumkin bo’lgan qiymatlarni o’z ichiga oladi, ularning ba'zilari munosabatlarda ishlatilmasligi mumkin.

Relyatsion munosabat va jadval o’rtasidagi asosiy farqlar munosabat to’plam ekanligi bilan bog’liq, unda:

- bir xil elementlar (kortejlar) bo’lishi mumkin emas;
- kortejlar buyurtma qilinmaydi (va jadvaldagi qatorlar yuqoridan pastgacha tartiblangan);
- atributlar tartiblanmagan (jadvaldagi ustunlar chapdan o’ngga tartiblangan);
- barcha atribut qiymatlari atomikdir.

Munosabat munosabatlarining quyidagi turlari ajratiladi.

Nomlangan munosabat - ismga ega bo’lgan munosabat, ya'ni. munosabat o’zgaruvchisi yaratilgan biri.

Bazis munosabati o’z-o’zidan bo’ladigan nomli munosabatdir. Amalda, bu ishlab chiquvchi vaqtinchalik munosabatlardan farqli o’laroq, uni bevosita ma'lumotlar bazasining bir qismiga aylantirganligini anglatadi.

Hosil bo‘lgan munosabat nomlangan munosabatlar orqali va pirovardida tayanch munosabatlar orqali aniqlanadi. *Ekspressiv* - bu

qandaydir munosabat ifodasi orqali nomlangan munosabatlar to’plamidan olinishi mumkin bo’lgan munosabat. Ko’rsatilgan munosabatlar to’plami barcha asosiy va barcha hosila munosabatlarining yig’indisidir.

Ko’rinish - nomli olingan munosabat. Ko’rinishlar virtualdir; ular tizimda faqat nomlangan munosabatlar nuqtai nazaridan ta'rif orqali mavjud.

Snapshot - nomli olingan munosabat bo’lib, u tasvirdan farqli o’laroq, haqiqiydir, ya'ni. faqat boshqa nomdagi munosabatlar nuqtai nazaridan ta'rif bilan emas, balki uning individual ma'lumotlari bilan ham ifodalanadi. Snapshot yaratish so’rovni bajarishga o’xshaydi, faqat natijada olingan natijalar ma’lum bir nom ostida ma’lumotlar bazasida saqlanadi (va belgilangan chastotada yangilanishi mumkin).

So’rovning natijasi nomsiz hosila munosabatidir. So’rov natijalari ma'lumotlar bazasiga saqlanmaydi.

Oraliq natija - bu boshqa, kattaroq ifoda ichida joylashgan qandaydir munosabat ifodasi natijasi bo‘lgan nomsiz hosila munosabatdir.

Saqlangan munosabat bevosita fizik xotirada saqlanadigan munosabatdir.

Relyatsion ma'lumotlar bazasini foydalanuvchi tomonidan turli darajadagi aloqador munosabatlar to’plami sifatida qabul qilinadigan ma'lumotlar bazasi sifatida ta'riflash mumkin.

Ma'lumotlar yaxlitligi chegaralari

Ma'lumotlarning yaxlitligi haqida gapirganda, biz quyidagi taxminlardan kelib chiqamiz:

- istalgan vaqtda ma'lumotlar bazasi ma'lumotlar qiymatlarining ma'lum bir to’plamini o’z ichiga oladi;
- oddiygina ma'lumotlar qiymatlari to’plami, agar u haqiqatni aks ettirmasa, hech qanday ma'noga ega emas, ya'ni. haqiqiy dunyoning bir qismining vakili emas.

Shunday qilib, ma'lumotlar bazasi ta'rifi yaxlitlik qoidalarini o’z ichiga olgan holda kengaytirilishi kerak, uning maqsadi ma'lumotlar bazasini real dunyo cheklovlari haqida xabardor qilishdir.

Butunlik cheklovlari ma'lum bir ma'lumotlar bazasiga xos bo’lishi mumkin. Misol uchun, talaba a'zosi bo’lgan guruh mavjud guruhlarning aniq ro’yxatidan bo’lishi kerak. Biroq, relyatsion modelda potentsial va

tashqi kalitlar bilan tavsiflangan yaxlitlikni ta'minlashning umumiy qoidalari ham mavjud.

R nisbiy munosabat bo’lsin. Keyin *potentsial kalit* (uni K deb belgilaymiz *)* quyidagi xususiyatlarga ega bo’lgan *R* atributlarining kichik to’plami bo’ladi:

- *yagonalik* - *R* ga nisbatan *K* qiymati bir xil bo’lgan ikki xil kortej bo’lishi mumkin emas ;
- *ortiqcha bo’lmasligi* - *K ning* undan boshqa kichik to’plamlarining hech biri yagonalik xususiyatiga ega emas.

Bitta atributdan iborat nomzod kalit *oddiy kalit deb ataladi.* Masalan, 4.1-jadvalda keltirilgan munosabatlarda Ticket Number atributidan iborat oddiy nomzod kaliti bo’ladi (biz talaba karta raqami noyob deb hisoblaymiz). Bir nechta atributlardan tashkil topgan nomzod kaliti *kompozit kalit deyiladi.*

Potensial kalitlar muhim ahamiyatga ega, chunki ular kortej darajasida asosiy adreslash mexanizmini ta'minlaydi. Boshqacha qilib aytganda, kortejga ishora qilishning tizim tomonidan kafolatlangan yagona usuli, agar ular bir nechta bo’lsa, uning nomzod kalitining qiymatini yoki shunday kalitlardan birini ko’rsatishdir.

Relyatsion modelda potentsial kalitlardan birini *asosiy kalit sifatida tanlash odatiy holdir. Qolgan nomzod kalitlari muqobil kalitlar* bo’ladi . Relyatsion model nuqtai nazaridan qaysi kalitni asosiy kalit sifatida tanlash muhim emas. Biroq, ma'lumotlarni saqlashda, ko’pgina MBBTlar yozuvlarni asosiy kalit qiymati bo’yicha tartibga soladi. Bunday holda, asosiy kalit qiymatlarini taqqoslash operatsiyasini iloji boricha tezroq bajarish juda muhimdir. Shuning uchun, butun sonli qiymatlarga ega oddiy kalitdan foydalanish afzalroq bo’ladi. Agar mavjud atributlar orasida bunday rolga mos keladigan hech kim bo’lmasa, ko’pincha atribut birlamchi kalitni yaratish uchun maxsus kiritiladi, unga butun son hisoblagichining qiymatlari tayinlanadi. Bunday birlamchi kalit haqiqiy obyektni tavsiflovchi atributlardan hosil bo’lgan *tabiiy* kalitdan farqli ravishda *surrogat kalit deb ataladi.*

Ba'zi mualliflar ikkinchi darajali kalit tushunchasini kiritadilar - kortejni sezilarli darajada tavsiflovchi, ammo noyob bo’lmagan atributlar to’plami. 4.1-jadvalda, bu "To’liq ism" maydoni bo’lishi mumkin. Ushbu tushunchada ikkilamchi kalit yaxlitlik cheklovi emas, lekin uni noyob bo’lmagan indeks yaratish orqali ma'lumotlar bazasini loyihalashda hisobga olish mumkin.

Ma'lumotlar bazasida juda ko’p aloqalar bo’lishi mumkin va ularning ba'zilari bir-biri bilan bog’liq bo’lishi mumkin. Ular bilan ishlashda xorijiy kalitlar katta ahamiyatga ega.

R_1 asosiy munosabat bo’lsin . Agar quyidagi shartlar bajarilsa, A munosabatining *FK* atributlari to’plami *tashqi kalit deb ataladi:*

- *potentsial kalit K bilan asosiy R* munosabati (va *R*1 va *R* har xil emas) mavjud;
- *R* 1 munosabatning joriy qiymatidagi har bir *FK qiymati har doim R* munosabatning joriy qiymatidagi qandaydir kortejning *K* kalitining qiymatiga to’g’ri keladi.

Xorijiy kaliti, agar tegishli nomzod kaliti oddiy bo’lsa, oddiy bo’ladi. Xuddi shu narsa kompozit tashqi kalit uchun ham amal qiladi. Berilgan xorijiy kalitga kiritilgan har bir atribut tegishli nomzod kalit atributi bilan bir xil domenda aniqlanishi kerak.

Ma'lumotlar bazasi noto’g’ri xorijiy kalit qiymatlarini o’z ichiga olmasligini ta'minlash muammosi havola yaxlitligi muammosi deb ataladi. Tashqi kalitni o’z ichiga olgan munosabat havola munosabati deyiladi. Tegishli nomzod kalitini o’z ichiga olgan munosabat mos yozuvlar munosabati deb ataladi.

Relyatsion algebra

Relyatsion modelni ishlab chiqishda E. Kodd relyatsion algebrani kiritdi, u aloqalarni operand sifatida ishlatadigan va natijada munosabatlarni qaytaruvchi operatorlar to’plamidan iborat. U sakkizta operatsiyani o’z ichiga oladi:

- *to’plamlar bo’yicha an'anaviy amallar* – birlashma, kesishish, ayirish, dekart ko’paytma;
- *maxsus relyatsion operatsiyalar* – tanlash, proyeksiyalash, qo’shish, bo’lish.

Relyatsion algebra haqida gapirganda, biz *yopilish xususiyatini* e'tiborsiz qoldira olmaymiz. Bu munosabatga nisbatan munosabat amalining natijasi ham munosabat bo’lishidadir. Shunday qilib, bitta operatsiya natijasi boshqasiga kirish sifatida ishlatilishi mumkin. Shu tarzda siz ichki iboralardan foydalanishingiz mumkin.

Mumkin bo’lgan hollarda, atribut nomini meros qilib olish qoidasiga amal qilinadi: natijaning tegishli atributlari manba munosabatlari bilan bir xil nomlanadi. Ba'zi hollarda potentsial kalitlar meros qilib olinadi.

Keling, yana bir ta'rif bilan tanishamiz. Ikki munosabatli munosabatlar *turiga mos keladi,* agar:

- ularning har biri bir xil atributlarga ega;
- tegishli atributlar (ya'ni bir xil nomdagi atributlar) bir xil domenda aniqlanadi.

Assotsiatsiya . Asl boshi bilan bir xil bo’lgan va A yoki B ga yoki ikkalasiga tegishli barcha kortejlar to’plamidan iborat bo’lgan tanaga ega bo’lgan munosabat A va B ikkita mos keladigan munosabatlarning *birlashishi* deb ataladi.

Munosabatlar ikkita bir xil kortejga ega bo’lishi mumkin emasligi sababli, qo’shilish operatsiyasi dublikatlarni olib tashlash bilan birga bo’lishi mumkin. Bu A va B birlashgan munosabatlarda to’liq mos keladigan kortejlar topilsa sodir bo’ladi: yangi munosabat tanasini tashkil etuvchi kortejlar to’plamida to’liq mos keladigan elementlar bo’lishi mumkin emas. Natija darajasi dastlabki munosabatlar darajasiga teng bo’ladi va asosiy raqam asl munosabatlarning asosiy raqamlari yig’indisidan katta bo’lmaydi.

Munosabat operatsiyalarini qayd etish qoidalari turli mualliflar orasida bir oz farq qiladi. Birlashma yozish uchun odatda ikkita belgidan biri ishlatiladi:

A ∪ B

UNION B

Ushbu belgi shakllarining birinchisi to’plam nazariyasida qabul qilinganiga o’xshash, ikkinchisi ma'lumotlar bazasi so’rovlari tillarida qo’llaniladigan belgi shakliga yaqinroqdir.

Kesishish. Sarlavhasi asl bilan bir xil bo’lgan munosabat va bir vaqtning o’zida ikkala munosabatga tegishli bo’lgan barcha kortejlar to’plamidan iborat bo’lgan tana ikkita mos keluvchi A va B munosabatlarining *kesishishi* deyiladi . Amaliyot quyidagicha yoziladi:

A ∩ B

Kesishma natijasining kardinal soni A va B munosabatlarining asosiy sonlarining eng kichigidan katta bo’lmaydi, daraja asl munosabatlarning darajalariga teng bo’ladi.

Ayirish. Asl sarlavhalari bilan bir xil bo’lgan munosabat va A ga tegishli va B ga tegishli bo’lmagan barcha kortejlar to’plamidan iborat bo’lgan tanaga ikkita mos keluvchi A va B munosabatlarini *ayirish* deyiladi. Belgilanish shakllari:

A - B

A MINUS B

Natijaning asosiy soni A kardinal sonidan katta bo’lmaydi, daraja asl munosabatlarning darajalariga teng bo’ladi. Keling, kiritilgan ta'riflarni misollar bilan tushuntiramiz. A va B munosabatlari jadvalda ko’rsatilganidek belgilansin. 4.2 va 4.3 mos ravishda. Pastki chiziq StudID atributi asosiy kalit ekanligini bildiradi. Keyin A va B munosabatlarini birlashtirish, kesish va ayirish natijalari mos ravishda 4.2 va 4.6-jadvalda keltirilgan.

4.2-jadval

Aloqa

StudID	To’liq ism	Guruh
123	Ivanov I.I.	382
124	Petrov P.P.	382

4.3-jadval

B nisbati

StudID	**To’liq ism**	**Guruh**
123	Ivanov I.I.	382
127	Sidorov S.S.	383

4.4-jadval

Birlik A ∪ B

Studiya	**To’liq ism**	**Guruh**
123	**Ivanov I.I.**	**382**
124	**Petrov P.P.**	**382**
127	**Sidorov S.S.**	**383**

4.5-jadval

A ∩ B kesishmasi

StudID	To’liq ism	Guruh
123	**Ivanov I.I.**	382

4.6-jadval

Ayirish A B

StudID	To’liq ism	Guruh
124	Petrov P.P.	382

Quyidagi fikrlarga e'tibor qaratish lozim. Birinchidan, kalitlar va atribut nomlarini meros qilib olish natijasida hosil bo’lgan munosabatning sarlavhasini aniqlash imkonini berdi. Ikkinchidan, har ikkala munosabatda uchraydigan <123, I.I.Ivanov, 382> kortej

birlashish natijasida bir nusxada mavjud, kesishish natijasida munosabat tanasida yagona kortej bo’ladi. Xuddi haqiqiy sonlarni ayirishda A B natijasi odatda B A natijasiga teng emas.

Ikki A va B munosabatlarining *dekart mahsuloti* (bu yerda A va B umumiy atribut nomlariga ega emas) A va B asl munosabatlarining ikki boshining birlashuvi ("birlashtiruvchi") bo’lgan bosh bilan munosabat sifatida aniqlanadi.

$$A \times B$$

Dekart mahsulotiga misol keltiraylik. Dastlabki A va B nisbatlari jadvalda keltirilgan. 4,7 va 4,8 mos ravishda. Jadvalda 4.9-rasmda Dekart mahsulotining natijasi ko’rsatilgan.

4.7-jadval

Aloqa

StudID	To’liq ism
123	Ivanov I.I.
124	Petrov P.P.
127	Sidorov S.S.

4.8-jadval

B nisbati

Element
Matematika
Fizika

4.9-jadval

Dekart mahsuloti A × B

StudID	**To’liq ism**	**Element**
123	Ivanov I.I.	Matematika
123	Ivanov I.I.	Fizika
124	Petrov P.P.	Matematika
124	Petrov P.P.	Fizika
127	Sidorov S.S.	Matematika
127	Sidorov S.S.	Fizika

Mahsulot darajasi (4.9-jadvalga qarang) asl munosabatlar darajalari yig’indisiga (2+1) teng ekanligini va asosiy raqam asl munosabatlarning asosiy raqamlarining ko’paytmasiga teng ekanligini tekshirishingiz mumkin (3×2). Olingan munosabat kompozit asosiy kalitga ega {StudID, Item}, asl munosabatlar esa oddiy birlamchi kalitga ega edi.

Ushbu operatsiyaning quyidagi xususiyatlariga va quyida muhokama qilinadigan th-birlashma operatsiyasiga e'tibor qaratish lozim. Agar A va B munosabatlari bir xil atributlarga ega bo’lsa, natijada ularni farqlash uchun bunday atribut nomini <Munosabatlar nomi> ko’rinishida shakllantirishga rozi bo’lishingiz mumkin.<Atribut nomi> Masalan, A.FIO va B. FIO.

Ba'zan sizga o’zingiz bilan bo’lgan munosabatlarning Kartezian mahsuloti kerak bo’lishi mumkin. Bunday holda, siz yana bitta munosabat o’zgaruvchisidan foydalanishingiz kerak, aks holda natija atributlarining nomlarini to’g’ri shakllantirish mumkin bo’lmaydi: A1 = A, R = A × A1. Bu yerda "=" belgilash operatorini bildiradi.

Namuna olish. th-tanlamaning qisqartirilgan nomi bo‘lib, bu yerda t har qanday skaler taqqoslash operatorini (=, >, <,... va hokazo), th-tanlamani A munosabatidan X va Y atributlari (shu tartibda) bildiradi. - bu A munosabat bilan bir xil sarlavhaga ega bo’lgan munosabat va A munosabatidagi barcha kortejlar T to’plamini o’z ichiga olgan tana, buning uchun "X th t" shartini tekshirish "to’g’ri" qiymatini beradi. Yozib olish shakllari:

A[X th Y]

A WHERE X th Y

Qabul qilish operatsiyasi X va Y atributlari bir xil domenda aniqlanishini talab qiladi va D operatori buning uchun mantiqiy bo’lishi kerak. Konstanta atributlardan biri o’rniga yoki ikkalasi o’rniga ko’rsatilishi mumkin.

4.10-jadval

TALABALARGA munosabat

StudID	To’liq ism	Guruh
123	Ivanov I.I.	382
124	Petrov P.P.	382
127	Sidorov S.S.	383

4.11-jadval

Talabalar namunasi [Guruh = 382]

StudID	**To’liq ism**	**Guruh**
123	Ivanov I.I.	382

124	Petrov P.P.	382

th-namuna olish operatsiyasining ta'rifini kengaytiramiz. th-namuna uchun taqqoslash sharti mantiqiy bog’lovchilar bilan bog’langan ixtiyoriy sonli oddiy taqqoslashlarni o’z ichiga olishi mumkin deb faraz qilamiz. Agar kerak bo’lsa, qavslardan foydalanish mumkin.

Mantiqiy bog’lovchilar uchun muqobil yozuv shakllari ham qo’llaniladi, ulardan biri matematik mantiqda qabul qilinganiga, ikkinchisi ma'lumotlar bazasi so’rovlari tillaridagi yozuv shakliga yaqinroqdir:

- "not" ¬ yoki NONE deb yozilishi mumkin;
- “va” – ?, ^ yoki AND;
- "yoki" – v yoki OR.

Endi siz, quyidagi ifodani yaratishingiz mumkin:

Students WHERE ((Guruh = 382) AND (StudID < 124))

Yoki:

Students [Guruh = 382 & StudID < 124]

A munosabatining V, W,..., Z atributlari ustidagi proyeksiyasi (bu yerda atributlarning har biri A munosabatiga tegishli) {V, W, ...Z} sarlavhali va o’z ichiga olgan tanaga ega munosabatdir. barcha kortejlar to‘plami { V:v, W:w, ...Z:z}, ular uchun A munosabatida V atributining qiymati **V ga teng bo‘lgan kortej,** W – w,.. atributi mavjud. Shunday qilib, proyeksiya natijasida dastlabki munosabatning ba'zi atributlari chiqarib tashlanadi, shundan so’ng takroriy kortejlarni olib tashlash kerak.

A munosabatining V, W, Z atributlari ustidagi proyeksiyasi A [V, V, Z] sifatida belgilanadi. Masalan, Students munosabatimizni biroz o’zgartiramiz (4.12-jadval) va "Ism" atributi asosida proyeksiya qilamiz (4.13-jadval).

4.12-jadval

Studentslarga munosabat

StudID	To’liq ism	Guruh
123	Ivanov I.I.	382
124	Petrov P.P.	382
127	Sidorov S.S.	383
128	Ivanov I.I.	384

4.13-jadval

Proyeksiya Talabalar [to’liq ismi]

To’liq ism
Ivanov I.I.
Petrov P.P.
Sidorov S.S.

Ko’rib turganingizdek, StudID va "Group" atributlarini o’chirib tashlaganingizdan so’ng, dublikat <"Ivanov I.I."> syujeti olindi, natijada u bitta nusxada qoldi. Agar proyeksiya qilingan atributlar to’plami dastlabki munosabatning bitta potentsial kalitini o’z ichiga olmasa, shunga o’xshash vaziyatlar paydo bo’lishi mumkin.

Murakkab. Birlashma operatsiyasining ikki turi *aniqlanadi:* tabiiy birlashma va th-birlashma.

A va B munosabatlarida {X, Y} va {T, Z} sarlavhalari bo’lsin, bunda X, Y, Z atributlari kompozit bo’lishi mumkin. Boshqacha qilib aytganda, birinchi va ikkinchi munosabatda faqat Y atributi (yoki Y bilan belgilangan atributlarning kichik to’plami) mavjud. Xuddi shu nomdagi atributlar bir xil domenlarda aniqlanadi. U holda A va B munosabatlarining *tabiiy aloqasi* bu {X, Y, Z) sarlavhali va barcha kortejlarni o’z ichiga olgan {X:x, Y:y, Z:z} tanasi bilan bog’liqlik bo’lib, ular uchun A munosabatida atribut qiymati bo’ladi. X ga teng, Y qiymat atributi y, B munosabatida Y atributining qiymati y ga teng. A va B ning tabiiy birikmasi A JOIN B sifatida belgilanadi. Agar A va B munosabatlarning umumiy atribut nomlari bo’lmasa, ularning tabiiy birikmasi Dekart ko’paytmasiga ekvivalent bo’ladi.

4.14-jadval

TALABALARGA munosabat

StudID	To’liq ism	Guruh
123	Ivanov I.I.	382
124	Petrov P.P.	382
127	Sidorov S.S.	383

4.15-jadval

GROUPS munosabatlari

Boshliq	Guruh
Petrov P.P.	382

Sergeev S.S.	384

4.16-jadval

Tabiiy aloqa Talabalar GURUHLARGA QO’SHILADI

StudIP	To’liq ism	Guruh	Boshliq
123	Ivanov I.I.	382	Petrov P.P.
124	Petrov P.P.	382	Petrov P.P.

Birinchi munosabatda (4.14-jadval) “Guruh” atributining qiymati “384” ga teng bo’lgan kortejlar yo’qligi sababli, ikkinchi munosabatda (4.15-jadval) esa “Guruh” atributining qiymati bo’lgan kortejlar mavjud emas. Xuddi shu atribut "383", natijada "382" guruhi talabalari haqidagi ma'lumotlarni o’z ichiga olgan kortejlar qoldi.

Birlamchi kalit uchun yagonalik va ortiqchalik talablaridan kelib chiqadiki, natijaviy munosabatda asosiy kalit {StudID} dir.

Bo’lish. A va B munosabatlari mos ravishda {X, Y} va {Y} sarlavhalariga ega bo’lsin, X va Y atributlari kompozit bo’lishi mumkin. Ikki munosabatlarning bir xil nomlangan atributlari bir xil domenlarda aniqlanadi.

A munosabatini B ga bo’lish natijasi {X} sarlavhasi va barcha kortejlar to’plamini o’z ichiga olgan {X:x} tanasi bilan bog’liqlik bo’ladi.

A/B

4.17-jadvalda qaysi talaba qaysi fanni olganligi haqidagi ma'lumotlarni o’z ichiga olgan A munosabati ko’rsatilgan. B munosabati obyektlar ro’yxatini o’z ichiga oladi (4.18-jadval). B-jadvalda keltirilgan barcha fanlardan o’tgan talaba identifikatorlari ro’yxatini olish kerak.

Masala A ni B ga bo’lish yo’li bilan yechiladi (4.19-jadval).

4.17-jadval

Aloqa

StudlD	**Element**
123	Fizika
123	Matematika
124	Matematika
127	Fizika

4.18-jadval

B nisbati

Element
Fizika
Matematika

4.19-jadval

A/B bo’limi

Stud ID
123

Nazorat savollari:

1. Relyatsion ma'lumotlar modeli (RMM) nima?
2. RMM ni qanday amalga oshiriladi?
3. Bu model qanday maqsadlarga muvofiq ishlaydi?
4. RMM ni qanday tahlil qilish mumkin?
5. Ushbu model qaysi sohada foydalaniladi?
6. RMM ni qanday rivojlantirish mumkin?
7. Bu modelning afzalliklari va chegaralari nima?
8. Ma'lumotlar yaxlitligi chegaralari haqida ma’lumot bering.

5-MAVZU. RELYATSION MA’LUMOTLAR OMBORINI NORMALLASHTIRISH.

Reja:

1. Relyatsion ma'lumotlar bazalarini normallashtirish
2. Birinchi normal shakl
3. Ikkinchi normal shakl
4. Uchinchi normal shakl

Bu mavzuda relyatsion ma'lumotlar bazalarini mantiqiy loyihalashga yondashuvlarni ko’rib o’tiladi. Ushbu bosqichda ma'lumotlar bazasining mantiqiy tuzilishi ishlab chiqiladi: aloqador munosabatlar, ularning atributlari, potentsial va tashqi kalitlari aniqlanadi. Bu vazifa juda muhim, chunki ma'lumotlar bazasi strukturasini loyihalashda yo’l qo’yilgan xatolar ma'lumotlarni saqlash, o’chirish va o’zgartirishda nomaqbul oqibatlarga olib kelishi mumkin, bu *modifikatsiya anomaliyalari* deb ataladi.

O’chirish anomaliyasi bir obyekt (obyekt, hodisa va boshqalar) bilan bog’liq faktlarni o’chirishda biz boshqa obyektga tegishli faktlarni ixtiyoriy ravishda o’chirib tashlashimizda namoyon bo’ladi.

Qo’shish anomaliyasi shuni anglatadiki, bir obyekt to’g’risidagi ma'lumotlar bazasiga boshqa bir faktni qo’shimcha ravishda ko’rsatmasdan turib, ma'lumotlar bazasiga joylashtirish mumkin emas.

Yangilanish anomaliyasi shundan iboratki, bitta fakt (hodisa, obyekt) haqidagi ma'lumotlarga o’zgartirish kiritish uchun ma'lumotlar bazasiga bir nechta o’zgartirishlar kiritish kerak bo’ladi. Anomaliyalarga misollar quyida keltirilgan.

Relyatsion modelning muhim afzalliklaridan biri bu normallashtirish nazariyasidan foydalangan holda ma'lumotlar bazasining mantiqiy tuzilishi sifatini baholashning rasmiy mexanizmining mavjudligi.

Normallashtirish - ma'lumotlarni kiritish, o’zgartirish va o’chirishda munosabatlarning tuzilishini yaxshiroq xususiyatlarga ega bo’lgan shaklga qisqartirish jarayoni hisoblanadi. Normallashtirishning yakuniy maqsadi ma'lumotlar bazasi dizaynini olishdir, unda har bir fakt faqat bitta joyda paydo bo’ladi, ya'ni ma'lumotlarning ortiqchaligi istisno qilinadi. Xotiradan samaraliroq foydalanish vazifasiga qo’shimcha ravishda, normallashtirish ma'lumotlar bazasida ichki

qarama-qarshiliklar paydo bo’lishi sababli uning yaxlitligini buzish tahdidini kamaytirishga imkon beradi.

Aloqalarning normal shakllari (NF) tushunchasi kiritiladi, ularning har biri ma'lum cheklovlar to’plamiga mos keladi. Munosabat belgilangan cheklovlarni qondirsa, berilgan normal shaklda bo’ladi. Har bir keyingi NF barcha oldingilarning talablarini o’z ichiga oladi, ya'ni. yanada qattiqroq cheklovdir. Muayyan NFlarni ko’rib chiqishga o’tishdan oldin bir qator ta'riflarni kiritish kerak.

R nisbiy munosabat, *X* va *Y* esa bu munosabat atributlarining ba'zi kichik to’plamlari bo’lsin . *Y* to’plamning har bir qiymati uchun Y to’plamining faqat bitta qiymati bo’lsa va faqat X to’plamiga bog’liq bo’ladi. Boshqacha qilib aytganda, agar munosabatlarning ikkita korteji X qiymatiga to’g’ri kelsa , u holda ular *Y* qiymatida mos keladi. Funksional bog’liqlik *X* → Y shaklida yoziladi , "X funksional ravishda D ni aniqlaydi" deb o’qiladi . *Agar X* → Y federal qonuni mavjud bo’lsa , unda *X determinant* deb ataladi , *Y* esa *qaram qismdir*.

Keling, bir misolni ko’rib chiqaylik. Talabalar munosabati 5.1-jadvalda keltirilgan tuzilishga ega bo’lsin va StudID atributi asosiy kalit hisoblanadi. Taqdim etilgan munosabat quyidagi funksional bog’liqliklarga ega:

- {StudID} → {Name};
- {StudID, to‘liq ismi} → {Group};
- {StudID, to‘liq ismi} → {StudID} va boshqalar.

5.1-jadval

TALABALARGA **munosabat**

StudID	To’liq ism	**Guruh**
123	Ivanov I.I.	382
124	Petrov P.P.	382
127	Sidorov S.S.	383

Shu bilan birga, umumiy holatda {Group} atributlarining kichik to’plami funksional jihatdan {Name} ga bog’liq emas, chunki talabalar ro’yxatida bir xil bosh harflar bilan, lekin turli guruhlarda o’qiyotgan ismlar bo’lishi mumkin.

Misoldan ko’rinib turibdiki, fizik qonunlar to’plami, hatto kichik darajadagi munosabatlar uchun ham, juda katta bo’lishi mumkin. Shu bilan birga, Federal qonun yaxlitlik cheklovidir va uning bajarilishi ma'lumotlarni qo’shish, o’chirish yoki o’zgartirish kabi operatsiyalar

davomida MBBT tomonidan tekshirilishi kerak. Shu sababli, boshqa federal qonunlardan kelib chiqishi mumkin bo’lgan bunday federal qonunlarni ko’rib chiqishdan chiqarib tashlash tavsiya etiladi.

Funksional bog’liqlik, agar uning bog’liq qismi aniqlovchining kichik to’plami bo’lsa, *ahamiyatsiz* deyiladi. Ko’rib chiqilgan misolda {StudID, to’liq ismi} → {StudID} ahamiyatsiz qaramlikdir.

Agar quyidagi shartlar bajarilsa, funksional bog’liqlik *kamaytirilmas* deb ataladi:

- o’ng (qaram) qism bir elementli to’plam, ya’ni o’ng tomonda faqat bitta atribut mavjud;
- chap tomoni (aniqlovchining) qaytarilmas, ya'ni FZni yo’qotmasdan aniqlovchidan biron bir xususiyatni tashlab qo’yish mumkin emas.

Ma'lum bo’lganlar asosida yangi funksional bog’liqliklarni olish qoidalari mavjud.

- *reflekslik* – agar B atributlar to’plami A ning kichik to’plami bo’lsa, u holda A → B;
- *qo’shish* – agar A → B bo’lsa, u holda A!C haqida → B!C haqida;
- *tranzitivlik* – agar A → B va B → C bo’lsa, u holda A → C.
- *o’z taqdirini o’zi belgilash* – A → A;
- *parchalanish* – agar A → B! haqida C, keyin A → B va A → C;
- *birlashma* – agar A → B va A → C bo’lsa, u holda A → B!C haqida;
- *kompozitsiya* – agar A→B va C→D bo’lsa, A!C→B!D.

Birinchi oddiy shakl

Aloqa *birinchi normal shaklda* (1NF) bo’ladi, agar u faqat skaler atribut qiymatlarini o’z ichiga olsa va asosiy atributlarning hech biri NULL bo’lmasa. *Kalit* atribut har qanday potentsial kalitga kiritilgan xususiyatdir.

NULLga kelsak, bu atribut qiymatining aniqlanmaganligini ko’rsatadigan maxsus qiymat ekanligini aniqlashtirish kerak. Masalan, “Talabalar” munosabati “Imtihon bahosi” atributiga ega bo’lsin. Agar ba'zi bir talabaning ba'zi bir imtihon uchun bahosi noma'lum bo’lsa, u holda kortejning tegishli atributi NULL ga o’rnatiladi. NULLni 0 raqami bilan aralashtirib yubormaslik kerak.

Faqatgina 1NFdagi munosabatlar relyatsion modelni ko’rib chiqish boshida kiritilgan munosabat ta'rifiga mos keladi. 5.2-advalda 1NF talablarini buzadigan guruhlangan jadvalning namunasini ko’rsatadi.

Misolda "Element" va "Reyting" atributlarining qiymatlari atomik emas. Shunga ko’ra, bu struktura 1NFda munosabat emas.

5.2-jadval

Guruhlangan jadval misoli STUD

StudlD	**To’liq ism**	**Guruh**	**Element**	**Baho**
123	Ivanov I.I.	382	Matematika	4
			Fizika	5
124	Petrov P.P.	382	Matematika	5
			Fizika	5
127	Sidorov S.S.	383	Ma'lumotlar bazasi	4

5.3-jadval

STUD munosabati

Stitdin	To’liq ism	Guruh	Element	Baho
123	Ivanov I.I.	382	Matematika	4
123	Ivanov I.I.	382	Fizika	5
124	Petrov P.P.	382	Matematika	5
124	Petrov P.P.	382	Fizika	5
127	Sidorov S.S.	383	Ma'lumotlar bazasi	4

Ikkinchi normal shakl

Relyatsion munosabat *ikkinchi normal shaklda* (2NF) bo’ladi, agar u 1NF ta'rifiga javob bersa va uning birlamchi kalitga kiritilmagan barcha atributlari unga qaytarilmas tarzda bog’liq bo’lsa.

2NF va ZNF ni tavsiflashda hamma joyda, bu aniq ko’rsatilgan hollar bundan mustasno, munosabatlar munosabatlari faqat bitta potentsial kalitga ega, deb taxmin qilinadi, bu asosiy kalitdir.

5.3-jadvalda keltirilgan munosabatlarni ko’rsatish oson. Masalan, {StudID} → {Name} funksional bog’liqligi mavjud. Shunday qilib, "Full Name" atributining asosiy kalitga bog’liqligi kamaytirilmaydi: asosiy kalit {StudID, Subject} va "Mavzu" atributi Federal qonun determinantidan {StudID, Subject} → { To’liq ism} qaramlikni yo’qotmasdan.

Siz munosabatlarning tuzilishini 2NF da ikkiga bo’lish orqali yaxshilashingiz mumkin. Bu yerda yo’qotishsiz parchalanish

muammosi paydo bo’ladi, ya'ni. munosabatni ikkiga bo’lish, bu protsedura natijasida hech qanday ma'lumot yo’qolishi sodir bo’lmaydi.

R {A,B,C} munosabatini R1{A,B}, R2{A,C} munosabatlariga bo‘lish jarayoni proyeksiya , R1 va R2 munosabatlari *proyeksiyalar deyiladi.* Bu yerda A, B va C - bu asl munosabat atributlarining bir nechta ajratilgan kichik to’plamlari bo’lib, ularning birlashishi butun atributlar to’plamini beradi. Agar yo’qotishsiz parchalanish amalga oshirilgan bo’lsa, u holda R1 va R2 proyeksiyalarini ulash R nisbatini berishi kerak.

Parchalanishda ular ko’pincha Xit teoremasiga tayanadilar: R {A, B, C} *relyatsion munosabat* bo’lsin , bunda A, B, C *bu munosabatning* atributlari (*ehtimol kompozit).* Agar R A → B *bog’liqligini qanoatlantirsa , u holda R uning proyeksiyalarining {A, B} va {A, C} ga birlashishiga teng.*

5.4-jadval

TALABALARGA munosabat

StuHIO	To’liq ism	Guruh
123	Ivanov I.I.	382
124	Petrov P.P.	382
127	Sidorov S.S.	383

5.5-jadval

NATIJALAR munosabati

SuulID	Element	Baho
123	Matematika	4
123	Fizika	5
124	Matematika	5
124	Fizika	5
127	Ma'lumotlar bazasi	4

Uchinchi normal shakl

Aloqa *uchinchi normal shaklda* (3NF) bo’ladi, agar u 2NF ta'rifiga javob bersa va uning asosiy bo’lmagan atributlaridan hech biri funksional jihatdan boshqa kalit bo’lmagan atributga bog’liq bo’lmasa.

Shuni ta'kidlash kerakki, agar kalit bo’lmagan atributlar o’rtasida FL mavjud bo’lsa va bu FLning determinanti, o’z navbatida, birlamchi kalitga bog’liq bo’lsa, biz tranzitiv FLni

olamiz. Boshqacha qilib aytganda, agar munosabatda faqat bitta potentsial kalit mavjud bo’lsa va o’tish FLlarni aniqlash mumkin bo’lsa, bu munosabat 3NF ga mos kelmasligini ko’rsatadi.

Misolni ko’rib chiqamiz (5.6-jadval). T1 munosabati, ularning lavozimlari va ish haqini ko’rsatadigan xodimlar ro’yxati mavjud. Asosiy kalit ID atributidir, muqobil kalitlar yo’q, munosabat 2NFda. Ma'lumki, xodimning ish haqi miqdori to’liq egallab turgan lavozimiga qarab belgilanadi. Boshqacha qilib aytadigan bo’lsak, Federal qonun mavjud {Lavozim} - "{Ish haqi}, bu munosabatlarni Federal qonun talablariga mos kelmaydigan qiladi va unda o’zgartirish anomaliyalari mavjud.

5.6-jadval

T1 nisbati

ID	To’liq ism	Lavozim	Ish haqi
1	Ivanov I.I.	Muhandis	50 000
2	Petrov P.P.	Muhandis	50 000
3	Sidorova S.S.	Buxgalter	30 000
4	Vasilyeva V.V.	Buxgalter	30 000

Xis teoremasidan foydalanamiz va yuqoridagi FZ yordamida parchalanishni bajaramiz. Olingan relyatsion munosabatlarni T2 va TK deb belgilaymiz (5.7, 5.8-jadvallar), T2 da TK munosabatlarining birlamchi kalitiga murojaat qilib, tashqi kalit {Position} aniqlanishi kerak.

5.7-jadval

T2 nisbati

ID	To’liq ism	Lavozim
1	Ivanov I.I.	Muhandis
2	Petrov P.P.	Muhandis
3	Sidorova S.S.	Buxgalter
4	Vasilyeva V.V.	Buxgalter

5.8-jadval

TK munosabati

Lavozim	Ish haqi
Muhandis	50 000
Buxgalter	30 000

Bu yerda 2NF talablari bajarilishini ko’rsatish oson: T2 ga nisbatan "Ism" va "Lavozim" atributlari o’rtasida funksional bog’liqliklar yo’q va TK ga nisbatan asosiy bo’lmagan atribut faqat bitta.

oddiy Boyce-Codd shakli

5.3-bandda keltirilgan 2NF ta'rifi quyida sanab o’tilgan shartlarga javob beradigan munosabatlar tuzilishini tahlil qilish uchun to’liq mos kelmaydi:

- munosabatda ikki yoki undan ortiq nomzod kalitlari mavjud;
- bir nechta potentsial kalitlar mavjud va ular murakkab;
- nomzod kalitlari bir-biriga mos keladi (ya'ni murakkab kalitlar kamida bitta umumiy atributga ega).

Bunday holda, odatda "kuchli" Boyce-Codd normal shakli ishlatiladi. Munosabat *Boyce - Codd normal ko’rinishida* bo’ladi, agar ahamiyatsiz va kamaytirilmaydigan FZlarning barcha determinantlari potentsial kalit bo’lsa.

To’rtinchi normal shakl

Amalda, normallashtirish jarayoni ko’pincha JB munosabatlarini NFBC talablariga muvofiqlashtirish bilan tugaydi, lekin ba'zi hollarda munosabatlarni yuqori NF darajasiga yetkazish kerak.

Misolni ko’rib chiqamiz (5.9-jadval). STUD1 munosabati talabalar haqidagi ma'lumotlarni o’z ichiga oladi. Bir nechta bo’lim va mutaxassislik bo’lishi mumkin, deb taxmin qilinadi (ikkinchi ta'lim parallel ravishda olinadi va hokazo), shuning uchun barcha uchta atribut {StudID, Mutaxassislik, Sports} kompozit asosiy kalitiga kiritilgan.

5.9-jadval

Aloqa STUD1

StudID	Mutaxassislik	Sport
123	Axborot tizimlari	Kayaklar
123	Boshqaruv	Kayaklar
123	Axborot tizimlari	Yugurish
123	Boshqaruv	Yugurish
124	Axborot tizimlari	Suzish

Bu holda BCNF talablari qondiriladi, ammo munosabatlarda ortiqcha va modifikatsiya anomaliyalari mavjud. Xususan, sport bo'limi (qo'shish anomaliyasi) to'g'risidagi ma'lumotlarni kiritmasdan talabaning mutaxassisligi to'g'risidagi ma'lumotlarni kiritish mumkin emas, chunki "Sport" atributi asosiy hisoblanadi va NULL qiymatiga ega bo'lolmaydi.

R {A, B, C} ga nisbatan A → B *ko'p qiymatli bog'liqlik* mavjud, agar A va C juft qiymatlarga mos keladigan B qiymatlari to'plami faqat A ga bog'liq bo'lsa va C ga bog'liq bo'lmasa.

Bu holda A →→ C ham o'rinli ekanligini isbotlash mumkin, shuning uchun yozish mumkin: A → → B|C.

Ko'p qiymatli qaramlik - bu har bir fizik qonun ko'p qiymatli bo'lgan ma'noda federal qonunning umumlashtirilishi (ammo, umumiy holatda teskari emas).

Agar mutaxassislik va sport bo'limini tanlash faqat talabaning o'zi tomonidan belgilanadi deb hisoblasak, jadvalda. 5.9-rasmdagi misolda ikkita ko'p qiymatli bog'liqlik mavjud:

{StudlD} →→ {Mutaxassislik} | {Sport}.

Ularning hech biri federal qonun bo'lmaydi, chunki bitta talaba bir nechta bo'lim va mutaxassisliklarga mos kelishi mumkin.

R munosabati *to'rtinchi normal ko'rinishda* (4NF) bo'ladi, agar A → → B notrivial ko'p qiymatli bog'liqlik mavjud bo'lsa, R ning boshqa barcha atributlari funksional jihatdan A ga bog'liq bo'lsa.

Ko'rib chiqilayotgan misolda (5.9-jadval) munosabat 4NF talablariga javob bermaydi, lekin tegishli FLlarning yo'qligi sababli Heese teoremasidan parchalanish uchun foydalanish mumkin bo'lmaydi: barcha atributlar kompozit kalitga kiritilgan va har qanday FLlar ahamiyatsiz bo'ladi. Bunday hollarda Ron Fegin teoremasidan foydalaniladi: *A, B va C R{A, B, C} munosabat atributlarining kichik to'plamlari bo'lsin. R munosabati uning {A, B} va {A, C} proyeksiyalari birikmasiga teng bo'ladi, agar* R uchun *ko'p qiymatli A →→ BC bog'liqligi bajarilsa va faqat.*

Ushbu teoremadan foydalanib, biz normallashtirishni amalga oshiramiz va 4NF ni qanoatlantiradigan STUD2 (5.10-jadval) va STUD3 (5.11-jadval) munosabatlarini olamiz, ularning kombinatsiyasi dastlabki munosabatni beradi.

5.10-jadval

STUD2 **munosabati**

StudlD	Mutaxassislik
123	Axborot tizimlari
123	Boshqaruv
124	Axborot tizimlari

5.11-jadval

STUD3 **munosabati**

StudlD	Sport
123	Kayaklar
123	Yugurish
124	Suzish

Beshinchi normal shakl

Ba'zi hollarda munosabatlar munosabatlari yuqorida muhokama qilinganidan ko’ra murakkabroq bog’liqliklarni o’z ichiga olishi mumkin. Keling, bir nechta ta'riflarni keltiramiz.

U, V ,..., Z R munosabatlari atributlarining kichik to’plamlari bo’lsin. R *munosabati * (U, V ,....* Z) bilan belgilanadigan *birlashma bog’liqligini* qanoatlantiradi.

Ikki proyeksiyaga yo’qotishsiz ajralish mumkin bo’lmagan, lekin n ta proektsiyaga *(n* > 2) yo’qotishsiz parchalanishi mumkin bo’lgan munosabat *n-parchalangan* deb ataladi.

5.12-jadval

Nisbat

Provayder	Tafsilot	Obyekt
P1	D1	02
P1	D 2	01
P2	D1	01
P1	D1	01

Biz R ning proyeksiyalarini atributlar juftligiga chaqiramiz {Yetkazib beruvchi, Qism}, {Ta'minlovchi, Obyekt} va {Qism, Obyekt} *R, R2, R3* (5.13-5.15-jadvallar).

5.13-jadval

Proyeksiya *Ri = R* **[Yetkazib beruvchi, qism]**

Yetkazib beruvchi	Tafsilot
P1	D1
P1	D 2
P2	D1

5.14-jadval

Proyeksiya *R2 = R* [Yetkazib beruvchi, Obyekt]

Provayder	Obyekt
P1	02
P1	01
P2	01

5.15-jadval

Proyeksiya *R3 = R* [Qism, obyekt]

Tafsilot	Obyekt
D1	02
D 2	01
D1	01

Keling, ushbu prognozlarni ulash natijalarini ko'rib chiqaylik. *R1 JOIN R2* operatsiyasining natijasi asl munosabatni bermaydi, chunki u bitta qo'shimcha kortejga ega bo'ladi (5.16-jadval, o'q bilan belgilangan oxirgi qator). Va *R3* proyeksiyasi bilan bog'langandan keyingina asl munosabat olinadi (5.16-jadval).

5.16-jadval

***R3 = R1* JOIN *R2 ga* qo'shiling**

Provayder	Tafsilot	Obyekt
P1	D1	02
P1	D 2	01
P2	D1	01
P1	D1	01
P1	D 2	02

5.17-jadval

R5 = R4 JOIN *R3 ga* **qo'shiling**

Provayder	Tafsilot	Obyekt
P1	D1	02
P1	D 2	01
P2	D1	01
P1	D1	01

R munosabati *beshinchi normal shaklda* (5NF) yoki *proyeksiya-qo'shilishning* normal ko'rinishida bo'ladi , agar *R da biron bir qo'shilishga bog'liqlik R* da biron bir nomzod kalit mavjudligidan kelib chiqsa.

Domen kalitining oddiy shakli. Denormalizatsiya

Normalizatsiya sohasidagi keyingi tadqiqotlar 5NF bilan bog’liq yashirin anomaliyalar paydo bo’lishi mumkin bo’lgan vaziyatlarni izlash yo’lidan bordi. 1981-yilda R. Feigin maqola e'lon qildi, unda u domen kaliti NF ni aniqladi va DCNFdagi munosabatlar hech qanday modifikatsiya anomaliyalari yo’qligini ko’rsatdi. Bundan tashqari, modifikatsiya anomaliyalari bo’lmagan har qanday munosabat DKNFda bo’lishi kerak. Shundan so’ng, hech bo’lmaganda modifikatsiya anomaliyalarini bartaraf etish uchun yuqori darajadagi NF endi kerak emas edi.

Agar bu munosabatga qo’yilgan har bir cheklov domenlar va kalitlar ta'rifining mantiqiy natijasi bo’lsa, munosabat *domen kaliti NFda bo’ladi. Cheklov* bu yerda atributlarning mumkin bo’lgan statik qiymatlarini tartibga soluvchi har qanday qoida sifatida aniqlanadi va u qanoatlantiriladimi yoki yo’qmi aniqlanishi mumkin bo’lgan darajada aniq. DCNFga muvofiqlikni tahlil qilishda vaqtga bog’liq cheklovlar hisobga olinmaydi, masalan, "joriy davr uchun ish haqi oldingi davrdagidan kam bo’lishi mumkin emas".

Yuqorida ko’rib chiqilgan TALABLAR munosabati (5.4-jadval) nafaqat 5.4-bandda isbotlangan NFBC talablariga, balki DCNF talablariga ham mos kelishini ko’rsatamiz. Shu munosabat bilan birlamchi kalit cheklovi (StudID) mavjud va talaba identifikatorlarini yaratish qoidalari, o’quv guruhi raqamlari, ism va familiyalarni yozish tartibi bo’yicha cheklovlar bo’lishi mumkin. Yuqoridagi barcha cheklovlar kalitlarga yoki domenlarning ta'rifiga bog’liq, shuning uchun munosabatlar DKNFga mos keladi.

Mualliflar DKNFga munosabatni o’zgartirish algoritmini aniq belgilamagan, shuning uchun biz aytishimiz mumkinki, DKNF munosabatlar qanday talablarga javob berishi kerakligini ko’rsatadi, ammo bunga qanday erishish mumkinligi haqida javob bermaydi.

Normallashtirish haqidagi suhbatni yakunlab, munosabatlar munosabatlari strukturasini SF talablarini buzadigan shaklga ataylab qisqartirish sifatida tavsiflanadigan *denormalizatsiyani* eslatib o’tish kerak. Normallashtirish jarayonida munosabatlar munosabatlari odatda parchalanadi va bir nechtaga bo’linadi. Biroq, bir nechta munosabatlardan ma'lumotlarni o’qish uchun siz qo’shimcha vaqt talab qiladigan qo’shilish operatsiyalarini bajarishingiz kerak.

Ko’pgina hollarda jadvallarni birlashtirishda yo’qotilgan vaqt nisbatan kichik bo’lib, modifikatsiya anomaliyalarini o’z ichiga olmaydi ma'lumotlar bazasi strukturasini loyihalash muhimroqdir. Ammo bir qator hollarda tizimning ishlashiga bo’lgan g’amxo’rlik ishlab chiquvchilarni saqlangan ma'lumotlarning ba'zi ortiqchaligi va individual aloqadorlik munosabatlari uchun yuqoriroq normal shakllarning yo’qligi bilan rozi bo’lishga majbur qiladi (ma'lumotlar bazasi jadvallari). Buni oxirgi chora sifatida ko’rib chiqish mumkin va denormalizatsiya to’g’risida qaror juda ehtiyotkorlik bilan qabul qilinishi kerak.

Nazorat savollari.

1. Infologik modålni qurishda qanday konstruktiv elåmåntlar ishlatiladi?
2. Mohiyat va atribut deganda nimani tushinasiz?
3. “Entity-relyation” modeli kim tomonidan ishlab chiqilgan va uning ma’nosi nima?
4. Mohiyatlar orasida qanday bog’lanishlar mavjud bo’lishi mumkin?
5. Ma’lumotlarning rålyatsion modåli kim tomonidan ishlab chiqilgan va qachon?
6. Birlamchi va tashqi kalitlar haqida tushuncha bering.
7. Ma’lumotlarni normallashtirish deganda nima tushuniladi?
8. Qanday normal shakllar mavjud?

6-MAVZU. MA’LUMOTLAR BAZALARINI INFOLOGIK LOYIHALASH.

Reja:

1. Ma'lumotlar bazasini loyihalash bosqichlari.
2. Infologik loyihalash. Mohiyat-aloqa usuli.
3. Ma'lumotlar bazasini ishlab chiqish hayotiy sikli.
4. ER diagrammasi.

Ma'lumotlar bazasini loyihalash bosqichlari. Ma'lumotlar bazasini loyihalashning to’liq sikli quyidagilarni o’z ichiga oladi:

- kontseptual loyihalash
- mantiqiy loyihalash
- fizik loyihalash

Ma'lumotlar bazasini loyihalashda uchta asosiy muammo hal qilinadi:

- Kontseptual modelda predmet soha va foydalanuvchilarning axborot bo’lgan ehtiyojlarini qanday adekvat aks ettirish.
- Predmet soha obyektlarini ma'lumotlar modeli abstakt obyektlari bilan qanday qilib aks ettirilsa predmet soha semantikasiga zid bo’lmasligi va iloji bo’lsa, eng yaxshiroq bo’ladi.
- Ma'lumotlar bazasiga so’rovlarni bajarish samaradorligini qanday ta'minlash mumkin, ya'ni ma'lum bir MBBT xususiyatlarini hisobga olgan holda, ma'lumotlarni tashqi xotirada qanday tartibga solish va hokazo.

Infologik loyihalash. Mohiyat-aloqa usuli.

- Infologik modellashtirishning (kontseptual dizayn) maqsadi yaratilishi rejalashtirilgan ma'lumotlar bazasida saqlanishi kerak bo’lgan ma'lumotlarni to’plash va taqdim etishni inson uchun eng tabiiy usullarini ta'minlashdir.
- Infologik loyihalash bosqichidagi eng mashhur semantik ma'lumotlar modellaridan biri bu “Mohiyat-aloqa" (Entity-Relationship – ER - model) hisoblanadi. Model 1976 yilda Chen (Chen) tomonidan taklif qilingan. Predmet sohani modellashtirish kam sonli bir jinsli komponentlarni o’z ichiga olgan grafik diagrammalardan foydalanishga asoslangan.

Infologik modellarning asosiy tarkibiy elementlari quyidagilar:

- **Mohiyat** - u haqidagi axborotlar ma'lumotlar bazasida saqlanishi kerak bo’lgan har qanday farqlanuvchi obyekt (biz boshqasidan ajrata oladigan obyekt)dir. Obyektlar odamlar, joylar, samolyotlar, parvozlar, ranglar va boshqalar bo’lishi mumkin.

- **Atribut** – mohiyatni tavsiflovchi axborotning nomlangan elementi hisobanadi. Uning nomi muayyan mohiyat turi uchun nodir bo’lishi kerak, lekin har xil mohiyatlar uchun bir xil bo’lishi mumkin. Atributlar mohiyat haqida qanday ma'lumotlarni to’plash kerakligini aniqlash uchun ishlatiladi.

- **Aloqa (assotsiativ mohiyat)** - bu boshqa mohiyatlar o’rtasidagi o’zaro hamkorligini ta'minlashga xizmat qiluvchi mohiyatdir.

Fizik loyihalash. Ushbu bosqichdagi loyihalashning maqsadi MBBTga yo’naltirilgan ma'lumotlar bazasi modelining tavsifini yaratish hisoblanadi. Ushbu bosqichda bajariladi:

- mantiqiy modelda berilgan ma'lumotlar asosida relyatsion jadvallar to’plamining tavsifini va ular uchun cheklovlarni yaratish;
- ma'lumotlarni saqlashning aniq strukturalarini va ularga kirish usullarini aniqlash, ma'lumotlar bazasi bilan tizimning optimal ishlashini ta'minlash;
- yaratilayotgan tizimni himoyalash vositalarini ishlab chiqish.

Ma'lumotlar bazasini loyihalashning ahamiyati. Ma'lumotlar bazasi dizayni ma'lumotlarni rejalashtirish, saqlash va boshqarish uchun foydalaniladigan ma'lumotlar bazasi strukturasini belgilaydi. Ma'lumotlarning aniqligini ta'minlash uchun siz faqat tegishli va qimmatli ma'lumotlarni saqlaydigan ma'lumotlar bazasini yaratishingiz kerak. Yaxshi ishlab chiqilgan ma'lumotlar bazasi ma'lumotlarning izchilligini ta'minlash, ortiqcha ma'lumotlarni yo’q qilish, so’rovlarni samarali bajarish va ma'lumotlar bazasi ish faoliyatini yaxshilash uchun juda muhimdir. Ma'lumotlar bazasini loyihalashda uslubiy yondashuv ma'lumotlar bazasini ishlab chiqish bosqichida vaqtingizni tejaydi. Ma'lumotlarning ishonchliligi jadvalning tuzilishiga bog’liq bo’lib, birlamchi va yagona kalitlarning yaratilishi saqlangan ma'lumotlarning bir xilligini ta'minlaydi. Ehtimoliy qiymatlar jadvalini yaratish va qiymatni ko’rsatish uchun kalit yordamida ma'lumotlarni takrorlashdan qochishingiz mumkin. Shunday qilib, o’zgarish asosiy jadvalda har bir qiymat o’zgarganda faqat bir marta sodir bo’ladi. Umumiy ishlash uning dizayniga bog’liq bo’lganligi sababli, yaxshi ma'lumotlar bazasi dizayni oddiy so’rovlar va tezroq amalga oshirishdan foydalanadi.

Bundan tashqari, uni saqlash va yangilash oson. Boshqa tomondan, ma'lumotlar bazasi noto’g’ri ishlab chiqilgan bo’lsa, hatto kichik uzilishlar ham saqlangan voqealar, ko’rinishlar va yordamchi dasturlarga zarar etkazishi mumkin.

Ma'lumotlar bazasini ishlab chiqish hayot sikli. Ma'lumotlar bazasini ishlab chiqishning turli bosqichlari mavjud. Biroq, har bir bosqichni ketma-ket bajarish shart emas. Hayotiy siklni uch bosqichga bo’lish mumkin: talablarni tahlil qilish, ma'lumotlar bazasini loyihalash va amalga oshirish.

1- Talablarni tahlil qilish

Talablarni tahlil qilish ikki bosqichni talab qiladi:

- **Rejalashtirish:** Ma'lumotlar bazasini ishlab chiqishning ushbu bosqichi ma'lumotlar bazasini ishlab chiqishning butun hayot aylanish rejasini belgilaydi. Shuningdek, u tashkilotning axborot tizimlari strategiyasini tahlil qilishni talab qiladi.
- **Tizim ta'rifi:** Ushbu bosqichda tavsiya etilgan ma'lumotlar bazasi tizimining ko’lami tushuntiriladi.

2- Ma'lumotlar bazasini loyihalash

Haqiqiy ma'lumotlar bazasi dizayni ikkita asosiy ma'lumotlar modelini hisobga oladi:

- **Mantiqiy model:** ma'lumotlar bazasi modelini yaratish uchun berilgan talablardan foydalanadi. Ushbu bosqichda to’liq tuzilma ma'lumotlar bazasini boshqarish tizimi (MBBT) yoki uni fizik amalga oshirish uchun hech qanday maxsus talablarni hisobga olmagan holda qog’ozga tushiriladi.
- **Fizik model:** Bu bosqich mantiqiy modeldan keyin keladi va shuning uchun mantiqiy modelni fizik amalga oshirishni o’z ichiga oladi. Bu MBBT va boshqa fizik amalga oshirish omillarini hisobga oladi.

3- Amalga oshirish

Ma'lumotlar bazasini ishlab chiqishning hayot aylanishini amalga oshirish bosqichi quyidagilar bilan bog’liq:

- **Ma'lumotlarni o’zgartirish va yuklash** eski tizimdan ma'lumotlarni yangi ma'lumotlar bazasiga import qilish va aylantirishni o’z ichiga oladi.
- **Sinov:** Nihoyat, ushbu bosqichda yangi tizimdagi xatolar aniqlanadi va barcha ma'lumotlar bazasi talablari bajariladi.

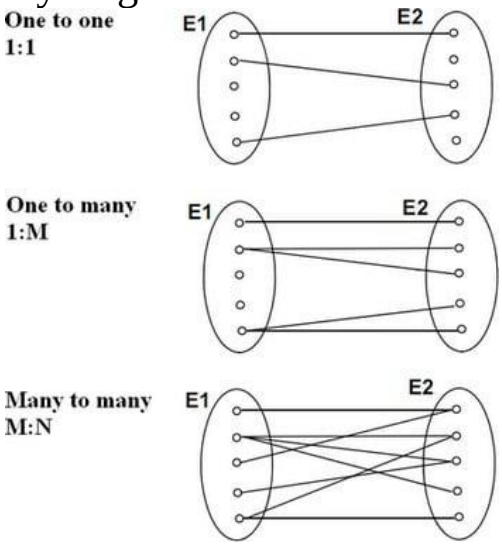
Ma'lumotlar bazasini loyihalash usullari

Ma'lumotlar bazasini loyihalashda ishlatiladigan eng keng tarqalgan ikkita usul:

- **Normallashtirish:** Jadvallar ma'lumotlarning ortiqcha va bog'liqligini kamaytiradigan tarzda tashkil etilgan. Kattaroq jadvallar kichikroq jadvallarga bo'linadi va munosabatlar yordamida bir-biri bilan bog'lanadi.
- **Shaxs-munosabat (ER) modellashtirish:** obyektlar atributlarini modellashtiradigan va real obyektlarni ifodalash uchun ular o'rtasidagi munosabatlarni belgilaydigan grafik ma'lumotlar bazasini loyihalash usulidir. Haqiqiy olamdagi har qanday obyekt atrof-muhitdan farq qiladigan yoki noyobdir.

Ma'lumotlar jadvallarga bo'lingandan so'ng, ma'lumotlar mazmunli tarzda birlashtirilishi kerak. Shunday qilib, siz har bir jadvalni tekshirishingiz va jadvallar o'rtasidagi munosabatni aniqlashingiz mumkin. Agar kerak bo'lsa, ma'lumotlar turlariga asoslangan munosabatlarni soddalashtirish uchun maydonlarni qo'shishingiz yoki yangi jadvallar yaratishingiz mumkin.

Bu vaqtda turli jadval yozuvlari o'rtasida bir-bir, bir-ko'p va ko'p-ko'p munosabatlari yaratiladi. Jadvalning bir elementi boshqa jadval elementi bilan bog'langan bo'lsa, u birma-bir munosabat deb ataladi (1:1). Birga ko'p (1:M) munosabatida bitta jadvaldagi element boshqa jadvaldagi ko'plab elementlar bilan bog'liq, masalan, bitta mijoz bir nechta buyurtmalar beradi. Ko'pdan ko'pga (M:N) munosabat jadvaldagi bir nechta elementlar boshqa jadvaldagi ko'p elementlar bilan bog'langan bo'lsa yuzaga keladi.



ER diagrammalarini kontseptsial sxema bo'yicha bajarish.

- Har bir obyekt yangi xaritaga jadval qo'shadi;

- Har bir atrlbut xaritaga yangi jadval ustuni qo’shadi;
- Har bir munosabat xaritalarga yangi ustunlar yoki yangi jadvalga (munosabat turiga qarabjo’rnatish.

Nazorat savollari

1. Ma'lumot baza modeli nima?
2. erarxik (shajara) modeli ma'hunot va uning asosiy xarakteristikalarini keltiring.
3. Tarmoqli model madumot va uning asosiy xarakteristikalarini tushuntirib bering.
4. PS mohiyat aloqa usulida tavsiflaganda qanday ishlar bajanladi?
5. Mosliklarni (munosabatlarj qanday turlari bor? Ularni tavsiflang.
6. Mohiyat-aloqa diagrammasi qanday quriladi?
7. Axborot tizimlarini loyihalashga infoiogik yondashishni asosiy qoidalarini tushuntiring.

7-MAVZU. MA’LUMOTLAR BAZALARINI INFOLOGIK LOYIHALASH.

Reja:

1. IDEF1X metodologiyasi yordamida ma‘lumotlar bazalarini loyihalash.
2. IE notatsiya.
3. Ma‘lumotlar bazasi fizik modelini yaratish.

IDEF1X dizayn metodologiyasini tavsiflovchi standart 1993 yilda AQShda ishlab chiqilgan. U oldingi versiyani takomillashtiradi.

IDEF1 va IDEF standartlar oilasining bir qismidir (*inglizcha* Integration Definition Methodology-dan), u shuningdek IDEF0 tizimlarini funksional modellashtirish, dinamik o’zgaruvchan tizimlarni modellashtirish IDEF2 va boshqa bir qator metodologiya standartlarini o’z ichiga oladi.

IDEF1X metodologiyasi korxona yoki boshqa mavzu sohasini tavsiflovchi ma’lumotlar strukturasini ifodalovchi axborot modellarini yaratish uchun ishlab chiqilgan. Bunday modellar kontseptual deb ham ataladi.

ERwin Data Modeler kabi IDEF1Xni qo‘llab-quvvatlovchi dasturiy vositalar ma’lumotlar bazasining ma’lumotlar bazasidan mustaqil mantiqiy modelini yaratish va undan ma’lum bir MBBT muhitida amalga oshirish uchun fizik modelni yaratishda foydalanish imkonini beradi. Bunday holda, tizimni loyihalashning dastlabki bosqichlarida (mantiqiy modelni yaratishda) obyektning tavsifi bo’yicha kelishib olish mumkin bo’ladi: obyektlarni, ularning munosabatlarini, individual xususiyatlarni tavsiflovchi atributlarni ajratib ko’rsatish. Agar kerak bo’lsa, bitta izchil mantiqiy model uchun siz turli xil MBBTlar uchun fizik modellarni yaratishingiz mumkin. Masalan, ERwin Data Modeler Community Editiondagi 9 IBM DB2, Microsoft SQL Server, MySQL, Oracle Database, Sybase Adaptive Server Enterprise va ODBC-mos (inglizcha *Open* Database Connectivity) MBBT uchun fizik modellarni yaratishni qo’llab-quvvatlaydi.

Fizik model asosida skript yaratilishi mumkin, uning tanlangan ma'lumotlar bazasi muhitida bajarilishi kerakli ma'lumotlar bazasi tuzilmalarini yaratishga olib keladi. Shunday qilib, fizik modelni

yaratishda ma'lumotlar bazasiga oid barcha tafsilotlarni tavsiflash kerak, masalan, tanlangan MBBT tomonidan qo’llab-quvvatlanadigan muayyan ma'lumotlar turi, mantiqiy modelni yaratishda esa siz o’zingizni umumlashtirilgan turni ko’rsatish bilan cheklashingiz mumkin (raqamli, satr va boshqalar).

Ma'lumotlar bazasining mantiqiy va fizik modeliga bo’linishi ishlab chiqilayotgan yechimni hujjatlashtirish nuqtai nazaridan bir qator afzalliklarga ega. Bir qator sabablarga ko’ra, jadvallar va ustunlar nomlari uzunligi bo’yicha MBBT tomonidan cheklovlar, ma'lumotlar bazasi obyektlari nomlarida milliy alifbolarni qo’llab-quvvatlash cheklanganligi sababli, ko’plab ishlab chiquvchilar lotin tilida ko’rsatilgan jadvallar va ustunlar uchun qisqa nomlardan foydalanishni afzal ko’rishadi. Masalan, ish stantsiyasining identifikatori ustuni "Wstld" deb nomlanishi mumkin. ERwin kabi CASE vositalari ma'lumotlar bazasining mantiqiy modelini yaratishda aniq, o’qilishi mumkin bo’lgan nomlardan foydalanishga va ularni fizik ma'lumotlar bazasi modelidagi qisqa texnik belgilar bilan almashtirishga imkon beradi. Qarorlarni hujjatlashtirish nuqtai nazaridan, model ustida ishlashda sharhlar va eslatmalar qo’shish imkoniyati ham foydalidir.

Ma'lumotlar bazasi modelini yaratishda obyektlar, ularning atributlari va obyektlar o’rtasidagi munosabatlar tavsiflanadi. IDEF1X terminologiyasi boshqa ma'lumotlar bazasini modellashtirish texnikalarida qo’llaniladigan terminlarga juda o’xshaydi. *Haqiqiy yoki mavhum obyektlar (hodisalar, hodisalar va boshqalar)ning bir* turga bo’linishi: ular bir xil belgilar majmuasiga ega va boshqa obyektlar bilan bir xil turdagi aloqalarda ishtirok etishi mumkin. Bunday to’plamdagi aniq obyekt *misol* deb ataladi. Misollar bir-biridan farq qilishi kerak. Obyektlar "Kompyuter" kabi birlik ot bilan ataladi. IDEF1X standarti shuningdek, nomga ma'lumotlar bazasi modelida noyob bo’lgan obyekt raqamini ko’rsatishga imkon beradi, masalan, "Kompyuter/1". E'tibor bering, yuqoridagi misolda "1" obyektning nusxasi emas, balki uning raqamidir.

Atribut obyektning barcha yoki ayrim holatlarida mavjud bo’lgan xususiyat yoki xususiyatga mos keladi. Atribut uchun mumkin bo’lgan qiymatlar to’plami u aniqlangan domen bilan cheklangan.

Aloqa ikki obyekt o’rtasidagi yoki bir xil obyekt misollari o’rtasidagi mantiqiy munosabatni tavsiflaydi. Ma'lumotlar bazasini modellashtirishda munosabat odatda fe'l yoki fe'l ibora deb ataladi.

Keling, IDEF1X belgisi bilan aniqlangan har xil turdagi obyektlar va munosabatlardan foydalanishni ko’rsatadigan misolni ko’rib chiqaylik. Aytaylik, siz korporativ kompyuter tizimi haqidagi ma'lumotlarni saqlash uchun ma'lumotlar bazasini ishlab chiqmoqchisiz. ERwin Data Modeler dasturidan foydalanib, biz mantiqiy model yaratishni boshlaymiz. Kompyuterlar noyob kirish raqamlari va nomlariga ega va bitta kompyuter uchun bir nechta qo’shimcha nomlar (taxalluslar) ishlatilishi mumkin. Masalan, servl.corp.ru kompyuterida mycorp.net va ftp.mycorp.net taxalluslari bo’lishi mumkin. Kompyuterlar tashkilot hududida joylashgan bo’lib, ularda dasturiy ta'minot o’rnatilgan. Mantiqiy ma'lumotlar bazasi modelining bir qismi 7.1-rasmda ko’rsatilgan.

7.1-rasm. IDEF1X yozuvidagi mantiqiy ma'lumotlar bazasi modelining fragmenti

IDEF1Xdagi obyektlar to’rtburchaklar yordamida tasvirlangan, uning ustida uning nomi ko’rsatilgan va chiziq ustidagi yuqori qismida asosiy kalitga kiritilgan atributlar keltirilgan. *Obyektlar mustaqil (* Identifikator-Mustaqil shaxs) va *qaram* (Identifikator-qaram shaxs) bo’lishi mumkin. Farqi shundaki, qaram obyektning misoli mavjud bo’lishi uchun u o’zi bog’liq bo’lgan obyekt misoli bilan bog’lanishi kerak. Bog’liq obyektlar yumaloq qirralari bo’lgan to’rtburchaklar bilan ifodalanadi. Shaklda ko’rsatilgan misolda. 6.4, "Kompyuter" mustaqil obyekti va "Kompyuter taxalluslari" qaramligi mavjud. Bu shuni anglatadiki, taxallus o’zi aniqlangan kompyuterni ko’rsatmasdan mavjud bo’lmaydi.

Mustaqil obyekt va unga bo’ysunuvchi qaram shaxs o’rtasida "birdan ko’pga" turidagi *identifikatsiya qiluvchi munosabatlar* o’rnatiladi. Munosabatlar bo’ysunuvchi shaxsning yon tomonidagi qora doira bilan uzluksiz chiziq bilan ifodalanadi. ERwinda ma'lumotlar bazasi modelini yaratishda obyektlar birinchi navbatda mustaqil

obyektlar sifatida yaratiladi va identifikatsiya qiluvchi munosabatlar o’rnatilgandan so’ng, subyekt avtomatik ravishda qaram obyektga aylanadi. Shu bilan birga, asosiy obyektning birlamchi kalitining atributlari uning asosiy kalitiga avtomatik ravishda qo’shiladi, u ham tashqi kalitni tashkil qiladi: 7.1-rasmda "Kompyuter taxalluslari" obyektining asosiy kalitining bir qismi bo’lgan "Kompyuter raqami" atributi hisoblanadi. Xorijiy kaliti diagrammada "(FK)" belgilari bilan belgilangan.

Asosiy obyektning asosiy atributlarini tashqi kalit sifatida avtomatik ravishda qo’shish (munosabat yaratishda) operatsiyasi kalitlarni *ko’chirish* deb ataladi.

Keling, diagrammada ko’rsatilgan ikkinchi bir-ko’p munosabatni ko’rib chiqaylik(7.1-rasm). Bu "Xona" va "Kompyuter" ikkita mustaqil obyekt o’rtasidagi aloqadir. IDEF1X terminologiyasida bunday munosabat aniqlanmaydigan Munosabat bo’ladi. U diagrammada bola obyekt tomonida qora doira bilan nuqta chiziq sifatida tasvirlangan. Bunday holda, biz kompyuter uchun ko’pi bilan bitta xonani belgilash mumkinligini hisobga olamiz va hech qanday xona belgilanmagan kompyuterlar bo’lishi mumkin. Relyatsion ma'lumotlar bazasini loyihalash nuqtai nazaridan, bu "Kompyuter" jadvalida "Xona raqami" tashqi kaliti NULL qiymatiga ega bo’lishi mumkinligini anglatadi.

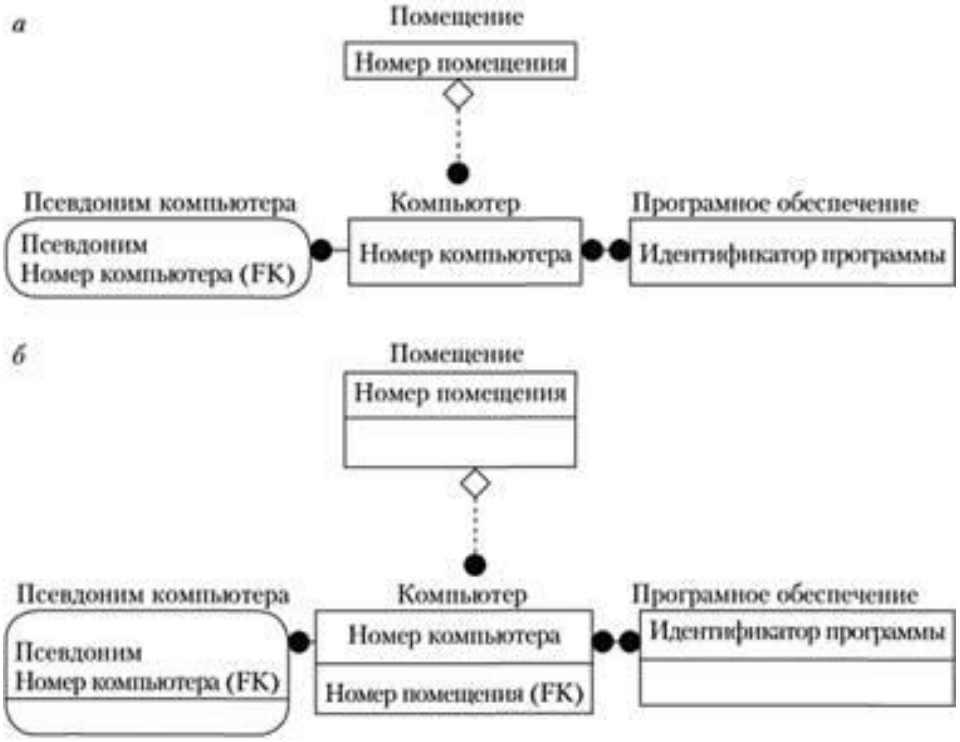
Bunday munosabat *ixtiyoriy* aniqlanmaydigan munosabatlar deb ataladi. Agar olmos bo’lmasa, bu tegishli tashqi kalit uchun NO NULL cheklovi bilan amalga oshiriladigan ushbu munosabatlarda bo’ysunuvchi obyektning majburiy ishtirokini anglatadi (ilovaviy ma'lumotlar bazasi jadvallarini yaratishda belgilangan cheklovlar haqida ko’proq ma'lumot olish uchun 7-bobga qarang). IDEF1Xda bunday munosabat *majburiy* aniqlanmaydigan munosabatlar deb ataladi.

Modelga identifikator bo’lmagan munosabatni qo’shsangiz, kalitlar ko’chishi ham sodir bo’ladi: tashqi kalit avtomatik ravishda asosiy obyektning asosiy kaliti asosida yaratilgan asosiy obyektga qo’shiladi. Biroq, identifikatsiya munosabatlaridan farqli o’laroq, bu holda tashqi kalit asosiy obyektning asosiy kalitining bir qismiga aylanmaydi.

Shaxs-munosabat diagrammasi (ERD) mavzu sohasida aniqlangan asosiy obyektlar va ular o’rtasidagi munosabatlarni o’z ichiga oladi. Atributlar belgilanmagan. Diagrammalar tafsilotlar bilan

ortiqcha yuklanmagan va ular prognoz qilinadigan bazaning tuzilishini umuman muhokama qilish uchun ishlatilishi mumkin. Bunday diagrammaning namunasi 7.2-rasmda ko’rsatilgan.

7.2-rasm. Shaxs-aloqa diagrammasi



7.3-rasm. Kalitga asoslangan ma'lumotlar bazasi modeli ko’rinishlari

Kalitga asoslangan ma'lumotlar bazasi modeli batafsilroq va barcha obyektlar, ularning kalitlari va munosabatlarini o’z ichiga oladi. ERwin Data Modeler interfeysida model bilan ishlashda siz faqat asosiy kalitlardan yoki asosiy va xorijiy kalitlardan foydalanadigan ko’rinishga o’tishingiz mumkin.

Mantiqiy modelning pastki darajasi barcha obyektlar, ularning atributlari va ulanishlarini o’z ichiga olgan *to’liq atribut modelidir (ing. ful-attributed, FA).* Loyihalashtirilgan ma'lumotlar bazasining tuzilishi uchinchi normal shaklga muvofiqlashtirilishi kerak.

Fizik model uchun ikkita daraja aniqlanadi:

- 1) *transformatsiya modeli (English* Transformation Model, TM) mantiqiy relyatsion modelni amalga oshirish uchun tanlangan MBBT tomonidan qo’llab-quvvatlanadigan tuzilmalarga

aylantirishni tavsiflaydi. Ushbu tuzilmalar MBBT imkoniyatlari, ma'lumotlar hajmlari, kutilayotgan ma'lumotlar so’rovlari asosida optimallashtirilgan va denormalizatsiya qilinishi mumkin (uchinchi normal shakl talablariga javob bermaydi);

- 2) *ma'lumotlar bazasi modeli* transformatsiya modeli asosida yaratilgan va ishlab chiqilayotgan ma'lumotlar bazasini tavsiflovchi ma'lumotlar bazasi server tizimi katalog tuzilmalarining ekranini o’z ichiga oladi.

Taqdimot davomida illyustratsiyalar ma'lum bir rasmda ko’rsatish uchun eng muhim narsaga qarab turli darajadagi modellarning parchalarini ko’rsatadi.

Keling, IDEF1X metodologiyasiga muvofiq loyihalashda hisobga olingan obyektlar o’rtasidagi aloqalarning xususiyatlarini ko’rib chiqishga qaytaylik. Ko’rib chiqilayotgan *belgi* , agar kerak bo’lsa, asosiy obyektning namunasi bilan bog’lanishi mumkin bo’lgan bo’ysunuvchi obyekt misollarining sonini ko’rsatishga imkon beradi. Quvvat identifikatsiya qiluvchi yoki aniqlanmaydigan birdan ko’p munosabatlar uchun belgilanishi mumkin. Quyidagi variantlarni aniqlash mumkin:

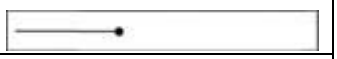
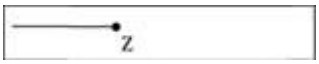
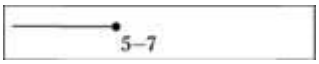
- sukut bo’yicha, asosiy obyektning bir nusxasi 0, 1 yoki bir nechta subyektning bir nechta nusxalariga mos kelishi mumkin; ushbu parametr diagrammada maxsus belgilanmagan;
- P (*ingliz tilidan* musbat - ijobiy) ota-ona obyektining namunasi 1 yoki undan ortiq subyektning misoliga mos kelishini bildiradi, belgi ulanish tasviridagi qora nuqta yoniga qo’yiladi;
- Z (*inglizcha* noldan – nol) asosiy obyektning namunasi 0 yoki 1 ta subyektning misoliga mos kelishini bildiradi;

ERwinda ulanish kuchini xususiyatlar oynasida o’rnatishingiz mumkin (ulanish tasvirini o’ng tugmasini bosganingizda ochiladigan kontekst menyusidagi "Xususiyatlar" bandi). Shuni ta'kidlash kerakki, standart sozlamalar bilan ERwin diagrammada ulanishning nomi va kuchini ko’rsatmaydi. Agar siz diagrammada munosabatlarning tubdan ko’rinishini yoqishingiz kerak bo’lsa, kontekst menyusini ochish uchun diagramma maydonidagi bo’sh joyga sichqonchaning o’ng tugmachasini bosing, undagi "Xususiyatlar" bandini va "Munosabatlar" yorlig’ini tanlang, "Munosabatlar tubdanligini ko’rsatish" variantini belgilang. Agar siz diagrammadagi munosabatlar nomini ko’rsatishni

istasangiz, xuddi shu yorliqda "Mantiqiy aloqa nomini ko’rsatish" opsiyasini belgilashingiz kerak.

7.1-jadval

Aloqa kuchini tasvirlash qoidalari

Tavsif	Misol
0, 1 yoki ko’p	
1 yoki ko’p	
0 yoki 1	
Aynan *N*. bu yerda *N* manfiy bo’lmagan butun son	
N dan L/ gacha, bu yerda *N* va *M* manfiy bo’lmagan butun sonlardir	
Qavslar ichidagi izohga ko’ra	

Ko’p hollarda tashqi kalit yordamida ma'lumotlar bazasi darajasida saqlanadigan ERwin'da bittadan ko’pga munosabatni belgilash orqali siz ma'lumotlarga o’zgartirishlar kiritilganda nima sodir bo’lishini belgilashingiz mumkin, masalan, asosiy obyekt nusxasini o’chirish tegishli bolalar aniqlangan. Bunday o’zgarishlar havolaning yaxlitligini buzishga olib kelishi mumkin va bunday holatlar maxsus ko’rib chiqilishi kerak. Har bir aniq holatda MBBT harakatini belgilaydigan qoida referent yaxlitlik *(* RI) qoidasi deb ataladi. Ushbu qoidalar ERwin muhitida loyihalash jarayonida aniqlanishi mumkin va MBBT darajasida ular xorijiy kalitlarni yaratishda variantlardan yoki triggerlardan foydalangan holda amalga oshiriladi. *Trigger* - bu SQL-da tasvirlangan va belgilangan hodisa sodir bo’lganda avtomatik ravishda bajariladigan harakatlar ketma-ketligi hisoblanadi. *Yo’naltiruvchi yaxlitlik triggeri* (RI trigger) - bu bir-biri bilan bog’liq bo’lgan ikkita jadval o’rtasida mos yozuvlar yaxlitligini saqlash uchun ishlatiladigan triggerning maxsus turidir. Agar bitta jadvaldagi qator qo’yilsa, o’zgartirilsa yoki o’chirilsa, RI triggeri boshqa jadvallardagi tashqi kalit qiymati kiritilgan (o’zgartirilgan, o’chirilgan) qatorning asosiy kalit qiymatiga mos keladigan qatorlar bilan nima qilish kerakligini aniqlaydi.

Qoidalar oltita holat uchun o‘rnatilishi mumkin: sub’yektning nusxasini o‘chirish, qo‘shish, o‘zgartirish *(* mos ravishda ingliz tilidagi

bolalarni o‘chirish qoidasi, pastki qo‘shish qoidasi, bolani yangilash qoidasi) yoki ota-ona *(inglizcha* ota-onani o‘chirish qoidasi, ota-ona kiritish qoidasi, ota-ona yangilash) qoida). Quyidagi harakatlar aniqlanishi mumkin:

7.4-rasm. **Yo’naltiruvchi yaxlitlik qoidalarini aniqlash**

- None – havola yaxlitligini saqlash uchun hech qanday harakat talab etilmaydi;
- Hech qanday harakat - hech qanday chora ko’rilmaydi;
- Kaskad – harakat “kaskadli” bo’lib, masalan, asosiy ob’yektning namunasi o’chirilganda, asosiy obyektning barcha tegishli nusxalari o’chiriladi;
- Cheklash – agar u referent yaxlitligini buzishga olib kelishi mumkin bo’lgan harakat taqiqlanadi; masalan, asosiy obyektning namunasini o’chirishga urinish, agar unda subyektning tegishli nusxalari bo’lsa, rad etiladi;
- Nullni o’rnatish – xorijiy kaliti havolasi obyektini o’chirib tashlaganingizdan so’ng, tashqi kalit Null (aniqlanmagan) ga o’rnatiladi;
- Standartni o’rnatish – oldingi holatga o’xshash vaziyatda tashqi kalit, agar u uchun belgilangan bo’lsa, standart qiymatni oladi.

7.4-rasmda ko’rsatilgan "Shartnoma" subyekti *assotsiativdir* - u bir nechta ota-onalar bilan bog’langan va ushbu obyektlarning ulanishi haqida qo’shimcha ma'lumotlarni o’z ichiga oladi. Agar qo’shimcha atributlar bo’lmasa, faqat xorijiy kalitlari bo’lsa, bunday obyekt

nomlash deb ataladi (*Inglizcha:* designative, tarjima "indikativ" va boshqa shunga o’xshash variantlar bo’lishi mumkin).

7.5-rasm. **N-ariy bog’lanishga misol**

Yana bir qiziqarli maxsus holat - bu rekursiv munosabatlar bo’lib, unda bir xil obyekt ota-ona va bola rolini bajaradi. Bunday bog’lanishlar ko’pincha predmet sohasida mavjud bo’lgan bir xil turdagi obyektlar ierarxiyasini tavsiflashda qo’llaniladi.

"Xodim" obyektining atributlari orasida "Boss ID" atributi mavjud bo’lib, u xuddi shu obyektning asosiy kalitiga ishora qiluvchi xorijiy kalitidir. Shunday qilib, xodim kimga bo’ysunishini ko’rsatish mumkin bo’ladi va munosabatlar aniqlanmaydigan va ixtiyoriy ravishda belgilanganligi sababli, ma'lumotlar bazasida hozirda hech kimga bo’ysunmaydigan xodimlar to’g’risidagi ma'lumotlarni saqlash mumkin bo’ladi.

ERwin muhitida rekursiv munosabatni belgilashda ulanish xossalarida kalit migratsiyasi tugallangandan so’ng unga tayinlanadigan tashqi kalit atributining nomini ko’rsatish kerak. *Bu nom "rol nomi*" deb ataladi , u atributning qaysi roli *ekanligini ko’rsatadi*.

7.6-rasm. **Tashkilotdagi xizmat ierarxiyasini tavsiflovchi rekursiv munosabatlar:**

a - Rol nomlarini ko’rsatish opsiyasi o’chirilgan;

b – Rol nomlarini ko’rsatish opsiyasi yoqilgan xorijiy kalit, bola shaxsida o’ynaydi.

Rekursiv ulanishni belgilashda, shuningdek, ulanishning mustahkamligini tekshirish tavsiya etiladi, chunki u har doim ham avtomatik ravishda to’g’ri o’rnatilmaydi.

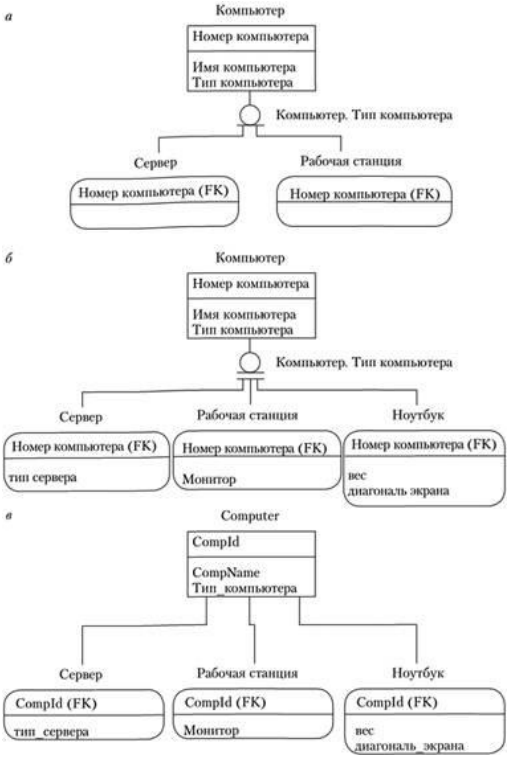
Yuqorida ta'kidlab o’tilganidek, rekursiv munosabatni ko’rsatishning ko’rib chiqilgan varianti ierarxiyalarni tavsiflashda qulaydir, agar obyektning har bir mantiqiy bo’ysunuvchi misolida bittadan ortiq mantiqiy original bo’lmasa. Bizning misolimizda, har bir xodimning faqat bitta bevosita rahbari bo’lishi mumkin va boshliqning bir nechta bo’ysunuvchisi bo’lishi mumkin. *Bu munosabat ierarxik rekursiya* deb ham ataladi. Misol uchun, ishlab chiqaruvchi kompaniya ko’plab yetkazib beruvchilardan tovarlar sotib olishi va ko’plab mijozlarga tovarlar yetkazib berishi mumkin. Bunday holda, ma'lumotlar bazasini loyihalashda siz qo’shimcha bog’liq obyekt va ikkita aniqlovchi bir-ko’p munosabatlardan foydalanishingiz kerak.

7.7-rasm. Xorijiy kalit nomini kvalifikatsiya qilish

7.8-rasm. Tarmoq rekursiyasini belgilashga misol

Men sizning e'tiboringizni qaratmoqchi bo’lgan IDEF1X notatsiyasining navbatdagi xususiyati toifalarning ta'rifidir. Ba'zi hollarda bir obyekt boshqasining *kichik turi* yoki *toifasi hisoblanadi.* Masalan, kompyuter server, ish stantsiyasi yoki noutbuk bo’lishi mumkin. Obyektlar o’rtasidagi ko’rib chiqilayotgan munosabatlar mohiyatan ierarxikdir, lekin ierarxik rekursiyadan farqli o’laroq, bu yerda misollar emas, balki obyektlar ierarxiyasi qurilgan. Ushbu ierarxiyadagi asosiy obyekt *toifali obyekt deb ataladi.* Keltirilgan misolda toifali obyektlar "Server", "Ish stantsiyasi", "Noutbuk" dir.

Xuddi shunday, *toifalar ierarxiyasi* yoki *meros ierarxiyasi* tuzilishi mumkin , bu umumiy xususiyatlarga ega bo’lgan obyektlar birlashmasining maxsus turidir.



7.9-rasm. Kategoriyalar ierarxiyasi:

a - to’liq bo’lmagan toifalar to’plami;

b – toifalarning to’liq to’plami;

c - fizik modelda ko’rsatish

Siz *diskriminatorni* - bir toifali shaxsni boshqasidan qanday ajratish mumkinligini ko’rsatadigan umumiy ajdodning atributidir.

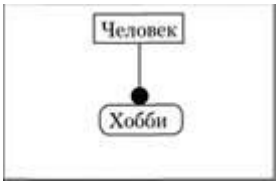
Bir nechta toifalar *to’liq* yoki *to’liq bo’lmasligi* mumkin. Birinchi holda, asosiy obyektning namunasi ko’rsatilgan toifalardan biriga

tegishli bo’lishi kerak, ikkinchi holda, sanab o’tilgan toifalarning birortasiga tegishli bo’lmagan holatlar bo’lishi mumkin.

6.13-rasmda mantiqiy ma'lumotlar bazasi modelidan fizik modelga o’tishda toifalar ierarxiyasi ERwin Data Modeler muhiti tomonidan qanday o’zgartirilishi *ko’rsatilgan*: jadvallarni bog’lash uchun tashqi kalitlardan foydalaniladi.

7.2-jadval

Bog’liq obyektlarning tasnifi

Ism turi	Tavsif	Misol
Xarakterli	Obyekt misolida bir necha marta sodir bo’lishi mumkin bo’lgan atributlar guruhini ifodalash uchun foydalaniladi.	
Assotsiativ (*inglizcha* assotsiativ) va nomlash (*inglizcha* dcsignative) obyektlari	Ikki yoki undan ortiq obyektlar o’rtasidagi munosabatlarni tasvirlash uchun ishlatiladi. Agar obyektda faqat asosiy obyektlarga ishora qiluvchi xorijiy kalitlarga kiritilgan atributlar bo’lsa, u *nomlash obyekti deb ataladi.* Agar obyekt qo’shimcha aloqa parametrlarini tavsiflovchi o’z atributlariga ega bo’lsa, u *assotsiativ deb ataladi.*	
Turkum YOKI *SUBTYPE*	Kategoriyali obyekt - bu asosiy obyektning kichik turi.	

7.2-jadvalda IDEF1X yozuvidagi diagrammalarda paydo bo’lishi mumkin bo’lgan qaram obyektlar tasnifi keltirilgan. Uni ko’rib chiqib, biz ushbu belgining qisqacha sharhini to’ldiramiz, batafsilroq tanishish uchun standartning o’ziga murojaat qilish tavsiya etiladi.

Axborot muhandisligi yozuvi

Yuqorida ta'kidlab o’tilganidek, ko’plab mashhur CASE ma'lumotlar bazasini loyihalash vositalari bir nechta ER diagramma tasvir belgilarini qo’llab-quvvatlaydi. IDEF1X yozuvi bilan bir qatorda 1980-yillar boshida taklif qilingan Informasion Engineering (qisqartirilgan IE) yozuvi ham keng tarqaldi.

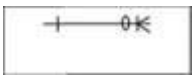
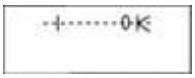
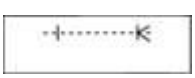
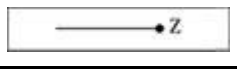
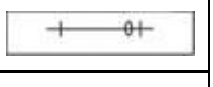
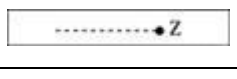
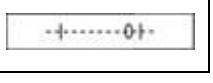
U yoki bu belgini tanlash ishlab chiquvchi kompaniyaning korporativ standartlari va mijozning talablari bilan belgilanishi mumkin. Shuning uchun ma'lumotlar bazasini ishlab chiquvchisi uchun bir nechta belgilarni bilish foydalidir. IDEF1X va IE ga muvofiq tuzilgan diagrammalardagi asosiy farqlar obyektlar o'rtasidagi munosabatlar tasvirlarida bo'ladi.

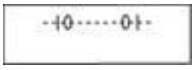
Bundan tashqari, IE yozuvi toifalarni tavsiflashda biroz boshqacha yondashuvni taklif qiladi. Ushbu belgi toifalarning to'liq va to'liq bo'lmagan to'plamlariga bo'linishni kiritmaydi, lekin *eksklyuziv* va *eksklyuziv bo'lmagan* yoki *inklyuziv kategorik* bog'lanishlar tushunchasidan foydalanadi. Birinchi holda, asosiy obyektning har bir nusxasi ko'pi bilan bitta toifaga tegishli bo'lishi mumkin. Masalan, kompaniya xodimi to'liq kunlik ("II") yoki yarim kunlik ("C") ishlashi mumkin. Bunday holda, agar "Xodim" bosh korxonasi diskriminator atributiga ega bo'lsa, u bandlik turini ko'rsatish uchun faqat "P" yoki "S" qiymatlarini olishi mumkin.

Agar toifali munosabatlar eksklyuziv bo'lmasa, asosiy obyektning namunasi bir nechta toifaga tegishli bo'lishi mumkin.

7.3-jadval

IDEF1X va IE yozuvlarida **ulanishlarni tasvirlash qoidalari**

Aloqa turi	**IDEF1X belgisi**	**IE belgisi**
"1 dan 0 gacha, 1 yoki undan ko'p" ni aniqlash		
Aniqlanmaydigan "1 dan 0, 1 yoki Ko'proq"		
Aniqlash "1 dan 1 gacha yoki Ko'proq"		
Aniqlanmaydigan "1 dan 1 gacha yoki Ko'proq"		
"1 dan 0 gacha yoki 1" ni aniqlash		
Aniqlanmaydigan "1 dan 0 yoki 1"		

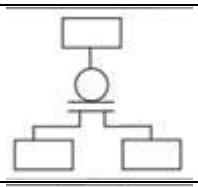
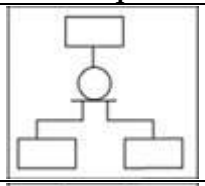
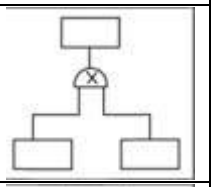
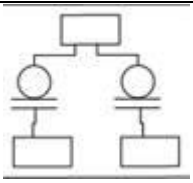
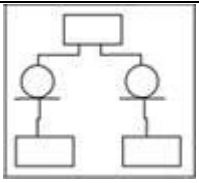
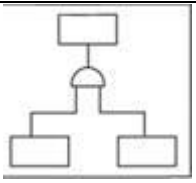
Aniqlanmaydigan "0 yoki 1 dan 0, 1 yoki undan ko’p"		
Aniqlanmaydigan "0 yoki 1 dan 0 yoki 1"		

Yuqorida muhokama qilingan xodimning ish bilan ta'minlanganligi misolida, agar "to’liq ish kuni" va "to’liq ish kuni" ning ikkita toifasi kiritilgan bo’lsa va munosabatlar eksklyuziv bo’lmagan deb ta'riflangan bo’lsa, u holda ham to’liq, ham yarim vaqtda ishlaydigan xodimlar bo’lishi mumkin. Shunga ko’ra, diskriminator atributi uchta qiymatni olishi mumkin.

7.4-jadvalda IDEF1X va IE yozuvlarida toifalarni tasvirlash variantlarini ko’rsatadi. Men ota-ona obyekti va uning kichik turlari o’rtasidagi eksklyuziv bo’lmagan munosabatlarni IDEF1X belgisi yordamida ham tasvirlash mumkinligiga e'tibor qaratmoqchiman. Buning uchun asosiy obyekt va toifalarning har biri o’rtasida qo’shimcha kategorik bog’lanishlar kiritiladi.

7.4-jadval

IDEF1X va IE yozuvlarida **toifalarni ko’rsatish qoidalari**

Turi	IDEF1X		I.E.
	to’la	to’liqsiz	
Eksklyuziv (eksklyuziv)			
Eksklyuziv bo’lmagan (shu jumladan)			

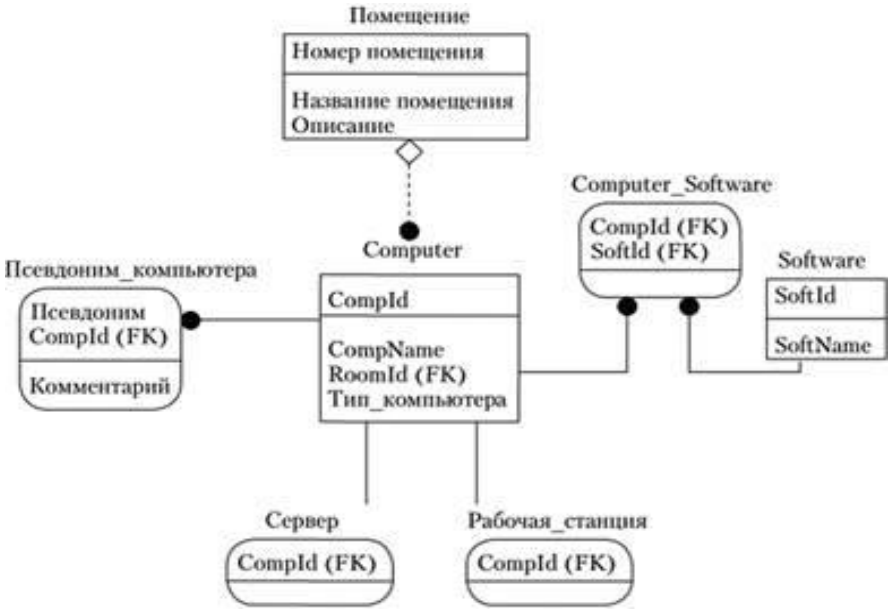
7.11-rasmda IE yozuviga muvofiq tuzilgan mantiqiy ma'lumotlar bazasi modelining fragmenti ko’rsatilgan. Agar IE yozuvida ulanishlarni tasvirlashning yuqorida tavsiflangan xususiyatlarini hisobga oladigan bo’lsak, bir belgini ishlatishdan boshqasiga o’tish hech qanday maxsus muammolarni keltirib chiqarmaydi.

7.11-rasm. IE yozuvidagi mantiqiy ma'lumotlar bazasi modelining fragmenti

Ma'lumotlar bazasining fizik modelini yaratish

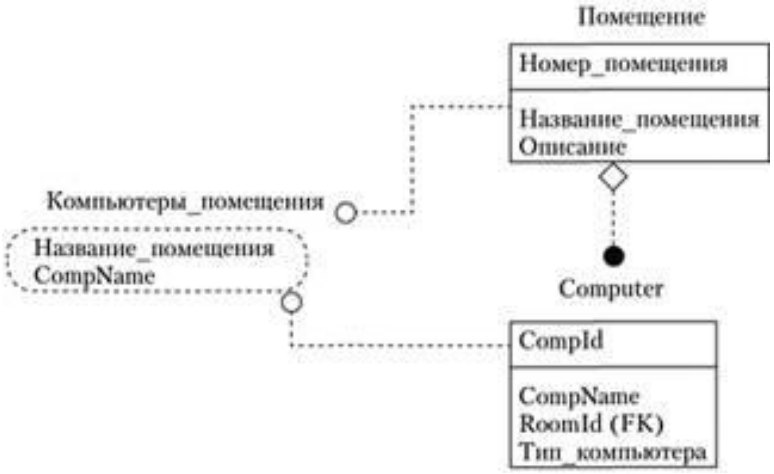
Ma'lumotlar bazasining mantiqiy modelini tavsiflashda "obyekt", "munosabat", "atribut" kabi atamalar qo’llaniladi. Relyatsion ma'lumotlar bazasining mantiqiy modelidan fizik modeliga o’tishda siz jadvallar, yozuvlar, maydonlar va boshqa obyektlar va tanlangan MBBT tomonidan qo’llab-quvvatlanadigan cheklovlar bilan ishlashingiz kerak bo’ladi. Shu bilan birga, IDEF1X va IE belgilaridan ERwin muhitida fizik model uchun diagrammalarni qurish uchun ham foydalanish mumkin.

Ma'lumotlar bazasining fizik modeli ko’p-ko’p munosabatlarni, shuningdek, kategorik munosabatlarni ishlata olmaydi. Belgilangan ulanish turlari uchun transformatsiya avtomatik ravishda amalga oshiriladigan fizik modelning namunasi 7.12-rasmda keltirilgan.



7.12-rasm. IDEF1X yozuvidagi fizik ma'lumotlar bazasi modelining fragmenti

Ma'lumotlar turlarini aniqlashtirish va jadvallar va maydonlar nomlarini MBBT talablariga muvofiq keltirishdan tashqari, fizik model ko’rinishlar, indekslar, saqlangan protseduralar, triggerlar va boshqa ma'lumotlar bazasi obyektlarini o’z ichiga olishi bilan ajralib turadi.



7.13-rasm. Fizik modeldagi viewlardan foydalanish

Ma'lumotlar bazasining fizik modeli ERwinda qurilgandan keyingi qadam ma'lumotlar bazasini o’zi yaratishdir. Buning uchun siz tanlangan serverga ulanishingiz kerak, so’ngra unda ma'lumotlar bazasi sxemasini yaratishingiz kerak (ERwin menyusida Action → → Forward Engineer → Schema...).

7.14-rasm. Erwin Data Modeler v-da tanlangan ma'lumotlar bazasi serveriga ulanishni o’rnatish.

Nazorat savollari:

1. Ma'lumotlar bazasini loyihalash bosqichlarini ayting .
2. Shaxs-munosabat diagrammalarining asosiy elementlarini ayting
3. Axborot modellarini qurishni qo’llab-quvvatlaydigan CASE vositalarining asosiy imkoniyatlarini sanab o’ting .
4. Mustaqil va qaram obyekt o’rtasidagi farq nima?
5. Atributlar turlarini sanab bering .
6. Obyektlar o’rtasidagi munosabatlarning asosiy belgilarini ayting
7. Mantiqiy loyihalash maqsadini ayting .
8. Ma'lumotlarning relyatsion modeliga mos kelmasligi mumkin bo’lgan kontseptual modelning asosiy elementlarini sanab o’ting.
9. Butunlik cheklovlarini ayting.
10. Ma'lumotlarni denormalizatsiya qilish nima?
11. Ma'lumotlarni normalizatsiya qilishning asosiy usullarini sanab o’ting va ularni tavsiflang .
12. Relyatsion ma'lumotlar bazalarida qo’llaniladigan asosiy xavfsizlik mexanizmlarini ayting .

8-MAVZU. SQL TILI ASOSLARI.

Reja:

1. SQL tarixi.
2. SQL tilida ma’limotlar tiplari.
3. Jadvallar yaratish va tahrirlash.
4. Ma’lumotlarni ajratib olish:SELECT.

Hozirgi vaqtda SQL *(* strukturalangan so’rovlar tili) standart relyatsion ma'lumotlar bazasi tili bo’lib, barcha umumiy relyatsion va obyekt bilan bog’liq MBBTlar tomonidan qo’llab-quvvatlanadi.

1970-yillarda *IBM da relyatsion ma'lumotlar modelini amaliy amalga oshirish imkoniyatlarini sinab ko’rish uchun eksperimental MBBT System R ishlab chiqildi.Uning uchun SEQUEL (* Structured English QUEry Language) tili yaratildi . Til juda mashhur bo’ldi va biroz vaqt o’tgach, ro’yxatdan o’tgan SEQUEL savdo belgisi mavjudligi sababli u SQL deb o’zgartirildi.

Bir qator ishlab chiqaruvchilarning MBBTlarida mustaqil ravishda amalga oshirilgan SQL tilini qo’llab-quvvatlash ushbu til uchun standartlarni ishlab chiqish zaruratini keltirib chiqardi. SQL spetsifikatsiyasini ishlab chiqish 1982-yilda Amerika Milliy Standartlar Instituti (ANSI) tomonidan boshlangan. Standart 1986-yilda ANSI tomonidan qabul qilingan va 1987-yilda Xalqaro Standartlashtirish Tashkiloti (ISO) tomonidan tasdiqlangan. SQL standartining ushbu versiyasi odatda SQL-86 deb ataladi.

Keyingi versiya 1989-yilda chiqarilgan (SQL–89). Unda, xususan, ma'lumotlarni qidirish va ma'lumotlarni manipulyatsiya qilish operatorlarining sintaksisi va semantikasi aniq standartlashtirildi, ma'lumotlar bazasi yaxlitligini cheklash vositalari belgilandi.

Standartning SQL2 deb ham ataladigan SQL-92 versiyasi oldingi versiyani sezilarli darajada kengaytirdi, ma'lumotlar bazasi sxemasini manipulyatsiya qilish, tranzaktsiyalar va seanslarni boshqarish, ma'lumotlar bazasiga ulanish va hokazolarni aniqladi.

1999-yilda chiqarilgan SQL:1999 (SQL3) muntazam ifodalar, rekursiv so’rovlar, triggerlarni qo’llab-quvvatlash, asosiy protsessual kengaytmalar, skalyar bo’lmagan ma'lumotlar turlari va ba'zi obyektga yo’naltirilgan imkoniyatlarni qo’shdi.

SQL tili uchta quyidagi tilni o’z ichiga oladi:

- *Data Definition Language* (DDL) – ma’lumotlarni aniqlash tili, jumladan CREATE, ALTER, DROP kabi operatorlar;
- *Data Manipulation Language* (DML) – ma’lumotlarni qayta ishlash tili bo‘lib, u sizga ma’lumotlarni so‘rash va o‘zgartirish imkonini beradi hamda SELECT, INSERT, UPDATE, DELETE operatorlarini o‘z ichiga oladi;
- *Ma'lumotlarni boshqarish tili* (DCL) – ma'lumotlarga kirish ruxsatlarini boshqarish imkonini beruvchi va GRANT va REVOKE bayonotlarini o’z ichiga olgan ma'lumotlarni boshqarish tili.

Shuni esda tutingki, SQL standarti ma'lum bir ma'lumotlar bazasi tomonidan to’liq qo’llab-quvvatlanmasligi mumkin. Aksincha, ma'lum bir MBBT standartda ko’zda tutilmagan SQL ifodalarida qo’shimcha funksiyalardan foydalanish imkonini berishi mumkin. Shunday qilib, ko’pchilik ishlab chiquvchi kompaniyalar SQL standartining versiyalaridan biriga mos kelishi mumkin bo’lgan o’zlarining SQL dialektini saqlab qolishadi. Quyida ba'zi mashhur MBBTlar ro’yxati va SQL dialektlarining nomlari keltirilgan:

- Microsoft SQL Server – Transact-SQL yoki T-SQL;
- Microsoft Access – Jet SQL;
- Oracle ma'lumotlar bazasi – PL/SQL;
- IBM DB2 – SQL PL.

Materialni taqdim etar ekanmiz, misol tariqasida SQL standarti va Transact-SQL da individual ifodalarni yozish qoidalaridagi ayrim nomuvofiqliklar keltiriladi. Xuddi shunday nomuvofiqliklar boshqa MBBT ishlab chiqaruvchilari orasida ham mavjud.

SQLda ishlatiladigan atamalar relyatsion algebrada qo’llaniladigan atamalardan farq qiladi. Xususan, “munosabat”, “tuple”, “atribut” o’rniga “jadval”, “satr” va “ustun” nomlari qo’llaniladi. Shuni ham ta'kidlash kerakki, SQL so’rovi natijasi ikki nusxadagi qatorlarni o’z ichiga olishi mumkin. Bu uni takroriy kortejlar bo’lishi mumkin bo’lmagan munosabat ifodasi natijasidan ajratib turadi.

Ma'lumotlar turlari

SQL tili standarti bir qator ma'lumotlar turlarini belgilaydi. Eng ko’p ishlatiladiganlar quyida keltirilgan. Yana bir bor ta'kidlashni istardimki, ba'zi ma'lumotlar bazasi ma'lumotlar bazasi sanab o’tilgan

turlarning ayrimlarini qo’llab-quvvatlamasligi yoki aksincha, o’z ma'lumotlar turlarini qo’shishi mumkin.
INTEGER, SMALLINT – imzolangan butun sonlar. Taqdimotning aniqligi va o’lchami amalda aniqlanadi va SMALLINT aniqligi INTEGER aniqligidan oshmasligi kerak. Ko’pgina MBBTlar INTEGER TO INT yozuvini qisqartirish imkonini beradi.

NUMERIC (r, s), DECIMAL (r, s) - umuman aniq son, butun son yoki kasr qismi bilan formatini ko’rsatish. **NUMERIC aniqlikdagi p (raqamlar soni) va o’lchov s** (o’nlik kasrlar soni) bilan raqamni ifodalaydi. Agar o’lchov o’tkazib yuborilsa, u 0 (ya'ni, butun son) deb qabul qilinadi va agar aniqlik o’tkazib yuborilsa, uning standart qiymati amalga oshirish bilan belgilanadi. DECIMAL turi w £ p bo’lishi uchun amalga oshirishga bog’liq m aniqligidan foydalanishni talab qiladi. DEC qisqartmasi qabul qilinadi.

FLOAT (p), REAL, DOUBLE PRECISION - suzuvchi nuqta tasviridagi raqamlar (taxminan raqamli turlar). REAL aniqligi amalga oshirish bilan belgilanadi, DOUBLE PRECISION aniqligi REAL aniqligidan kattaroq bo’lishi kerak. FLOAT ning aniqligi p parametri bilan belgilanadi, agar parametr mavjud bo’lmasa, u amalga oshirish orqali aniqlanadi.

CHARACTER (n) – n ta belgidan iborat qator, CHAR(n) qisqartmasi ruxsat etiladi. "CHAR" yozuvi CHAR(l) ga mos keladi. Agar saqlangan qiymat uzunligi n dan kichik bo’lsa, oxirida tegishli bo’shliqlar qo’shiladi. NATIONAL CHARACTER(n) turi ma'lumotlarni UNICODE kodlashda saqlaydi (milliy alifbolarni qo’llab-quvvatlash uchun har bir belgi uchun 2 bayt). NCHAR belgisiga ruxsat berilgan.

CHARACTER VARYING (n) – n ta belgidan ortiq bo’lmagan o’zgaruvchan uzunlikdagi qator, VARCHAR (n) qisqartmasi ruxsat etiladi. Agar saqlangan qiymat uzunligi w bo’lsa, bu yerda w < n, aynan m ta belgi saqlanadi.

CHARACTER LANG OBJECT - katta hajmdagi matnni saqlash imkonini beruvchi belgilar turi. CLOB kirishiga ruxsat beriladi. NCLOB turi ham mavjud.

TIME - vaqt belgilarini saqlash uchun ma'lumotlar turi, shu jumladan <hours>:<daqiqa>:<sekundlar>:<soniya kasrlari>. TIMESTAMP - sana va vaqtni birga saqlash uchun ma'lumotlar turi.

Jadvallar yaratish

Birinchidan, SQL jadvallarining relyatsion model munosabatlari bilan solishtirganda ikkita muhim xususiyatini qayd etamiz:

- SQL jadvallarida bir xil satrlarga ruxsat berilgan, shuning uchun potentsial kalitlarga ega bo'lish shart emas;
- SQL jadvallarida ustunlar chapdan o'ngga tartibda ko'rib chiqiladi, munosabatda esa atributlar tartibi muhim emas.

Asosiy jadvallar CREATE TABLE operatori yordamida tuziladi, uning formati CREATE TABLE table name> (jadval-element-commalist) bu yerda <jadval nomi> - yaratiladigan asosiy jadval nomi; table-element-commalist - jadval darajasidagi ustun ta'riflari yoki cheklovlarining vergul bilan ajratilgan ro'yxati. Elementlar ro'yxatida kamida bitta ustun ta'rifi bo'lishi kerak.

Ustun ta'rifi ustun nomini va ustun aniqlangan asosiy turdagi yoki domen ko'rsatkichini o'z ichiga olishi kerak. Shuningdek, u standart qiymat va ustun darajasidagi cheklovni belgilashni ham o'z ichiga olishi mumkin.

NOT NULL cheklovi ustunda NULL qiymati bo'lmasligini talab qiladi (barcha qiymatlar ko'rsatilishi kerak). Odatiy bo'lib yoki agar siz ustun ta'rifida NULLni aniq ko'rsatsangiz, null qiymatlarga ruxsat beriladi.

PRIMARY KEY kalit so'zi birlamchi kalitni va UNIQUE - muqobil kalitni belgilash imkonini beradi. Ushbu cheklovlarning ikkalasi ham qiymatlarning noyob bo'lishini talab qiladi, ammo PRIMARY KEY qo'shimcha ravishda NOT NULL cheklovini o'z ichiga oladi. Shuning uchun, agar ustunda UNIQUE cheklovi bo'lsa, bu ustun faqat bir marta NULLni o'z ichiga olishi mumkin.

Agar CHECK (<shart>) cheklovi belgilansa, shart qiymatga nisbatan tekshiriladi. Agar shartli ifodani baholash natijasi noto'g'ri bo'lsa, satr yaratishga urinish sinov shartining buzilishi hisoblanadi.

Keling, bir nechta misollarni ko'rib chiqaylik. Aytaylik, ikkita ustunli T5 jadvalini yaratishingiz kerak - butun birlamchi kalit Id va haqiqiy Sura ustuni, ularning qiymati aniqlanishi kerak.

```
CREATE TABLE T5 (Id INTEGER PRIMARY KEY,
SUM REAL NO NULL)
```

Quyidagi misol Studld, Fio, Group ustunlari bilan talabalar jadvalini yaratishdir. Oxirgi ikkita ustunni belgilashda Dname va

Dgroup domenlaridan foydalanamiz va Guruh ustuni uchun standart qiymatni aniqlaymiz. SQL kodi quyidagicha ko’rinadi:

```
CREATE TABLE Students (Studld INTEGER PRIMARY KEY,
Fio Dname, group DEFAULT 383)
```

Bu yerda biz quyidagilarga e'tibor qaratmoqchimiz: Dname domeni va Dgroup domeni uchun standart qiymatlar belgilangan. Biroq, Guruh ustuni jadval yaratilganda aniq ko’rsatilgan standart qiymatga ega bo’ladi va Fio ustuni domen ta'rifidan meros qilib olingan standart qiymatga ega bo’ladi.

Ma'lumotlar bazasi tizimiga qarab, ustun nomi SQL kalit so’ziga mos kelganligi sababli, siz Guruh ustun nomi ekanligini aniq ko’rsatishingiz kerak bo’lishi mumkin. Microsoft SQL Server holatida buning uchun kvadrat qavslar ishlatiladi - [Group].

Asosiy jadvalni ALTER TABLE bayonoti bilan o’zgartirish mumkin. Quyidagi o’zgarishlarga ruxsat beriladi:

- ustunlarni qo’shish va olib tashlash;
- mavjud ustun uchun standart qiymatni aniqlash va avval belgilangan standart qiymatni olib tashlash.
- jadval uchun yangi yaxlitlik cheklovini yaratish va avval belgilanganini o’chirish.

Ma'lumotlarni olish: SELECT bayonoti

SQLda ma'lumotlarni tanlash uchun so’rovlarni yaratish uchun SELECT iborasi ishlatiladi. Uning formati quyida keltirilgan:

```
TANLASH [ ALL | DISTINCT ] select_item_cominalist
FROM table_reference_commalist
[ WHERE conditional_expression ]
```

SELECT - bu bir yoki bir nechta jadvallardan ma'lumotlarni tanlash, guruhlashni amalga oshirish, agregat funksiyalardan foydalangan holda ma'lumotlarni qayta ishlash, ichki so’rovlar yaratish va h.k. imkonini beruvchi juda murakkab operator hisoblanadi. SELECT ifodasi odatda dasturlash tillarida bo’lgani kabi "satr satr" emas, balki butun sifatida qayta ishlanadi. Bir oz umumlashtirilgan shaklda SELECT iborasini bajarish sxemasi quyidagicha ko’rinadi:

- FROM bo’limi bajariladi;
- WHERE bandi bajariladi (agar mavjud bo’lsa);
- GROUP BY bajariladi (mavjud bo’lsa);
- HAVING bajariladi (agar mavjud bo’lsa);
- SELECT bo’limidagi ta'riflar bajariladi;

- ORDER BY bajariladi (agar mavjud bo’lsa).

Kerakli SELECT bo’limini ko’rib chiqishdan boshlaylik. U bo’sh bo’lmasligi kerak bo’lgan select-item-commalist tanlangan elementlar ro’yxatini belgilaydi. ALL yoki DISTINCT kalit so’zidan ham foydalanish mumkin. Ulardan birinchisi so’rov natijasi ikki nusxadagi qatorlarni o’z ichiga olishi mumkinligini ko’rsatadi, ikkinchisi takroriy nusxalar o’chirilganligini bildiradi. Misol uchun, agar DISTINCT kalit so’zi ishlatilsa va uchta mos keladigan qator mavjud bo’lsa, natija ulardan faqat bittasi bo’ladi. Hech narsa aniq ko’rsatilmagan bo’lsa, ALL nazarda tutiladi.

Ichki so’rovlar (pastki so’rovlar) quyida batafsilroq ko’rib chiqiladi. Shu o’rinda shuni ta'kidlashni istardimki, yuqoridagi kabi so’rovlarda pastki so’rov faqat bitta ustunni qaytarishi kerak. Masalan, quyidagi so’rov ko’pchilik MBBTlarda xatolikka olib keladi:

```
SELECT * FROM T1
WHERE NULL shahar_nomi (
SELECT Id, shahar_nomi FROM T2 )
```

Ichki so’rovlar bilan ishlashda [NOT] EXISTS (<subquery>) ko’rsatmasi ham qo’llaniladi. Agar so’rov osti natijalari kamida bitta qatorni o’z ichiga olgan bo’lsa, EXISTS rost qiymatini qaytaradi. Agar so’rov osti bo’sh qatorlar to’plamini qaytarsa, EXISTS noto’g’ri bo’ladi. Ushbu dizayndan foydalanish misollari quyida keltirilgan.

Belgilar satrlari bilan ishlashda siz ko’pincha qandaydir naqshga mos keladigan satrni topishingiz kerak bo’ladi. Shunga o’xshash muammolarni WHERE bandidagi LIKE naqshli taqqoslash operatori yordamida hal qilish mumkin.

LIKE predikatida naqshni tavsiflashda siz istalgan o’lchamdagi qatorni, jumladan bo’sh qatorni bildiruvchi “%” belgisidan va “_” belgisidan istalgan bitta belgidan foydalanishingiz mumkin. Naqsh bitta tirnoq ichiga olingan. Shunday qilib, agar Talabalar jadvalidan familiyasi "Iv" bilan boshlangan talabalar haqidagi yozuvlarni tanlash kerak bo’lsa, so’rov quyidagicha ko’rinadi:

```
SELECT *
FROM Students
WHERE FIO LIKE 'Iv%'
```

Bir qator SQL dialektlari LIKE imkoniyatlarini yanada kengaytiradi. Masalan, MS SQL Server 2008 shablonni ro’yxat yoki belgilar qatori ko’rinishida ishlatishga imkon beradi. Belgilar ro’yxati

bilan kvadrat qavslar ro’yxatda ko’rsatilganlardan bitta belgini, belgilar oralig’i bilan - belgilangan diapazondagi belgini anglatadi; ^ belgisi va belgilar ro‘yxati yoki diapazoni – belgilanganlardan boshqa har qanday belgi.

Quyidagi so’rov familiyasi "A" dan "I" gacha bo’lgan harflar bilan boshlangan talabalarning yozuvlarini qaytaradi:

```
SELECT *
FROM Students
WHERE FIO LIKE '[A-I]%'
```

Ustun-kommalist bo’yicha GURUHLASH

Agar ushbu bo’lim so’rovda mavjud bo’lsa, FROM va WHERE bandlaridan olingan jadval guruhdagi ustun qiymatlarining har bir kombinatsiyasi faqat bir marta paydo bo’lishi uchun qayta tartibga solinadi.

Keling, bir misolni ko’rib chiqaylik. Asl NATIJALAR jadvali 8.2-jadvaldagidek ko’rinishga ega bo’lsin. GROUP BY StudID operatori bajarilgandan so’ng, jadval 8.3-jadvalda ko’rsatilganidek, qayta guruhlanadi. Umuman olganda, guruhlangan jadval 1NF talablariga javob bermaydi (ustun qiymatlari atomik emas) va uni so’rov natijasi sifatida qaytarib bo’lmaydi.

8.2-jadval

NATIJALAR **jadvali**

StudID	Mavzu	Mark
123	Matematika	4
123	Fizika	5
124	Matematika	5
124	Fizika	5
127	TPP	4

8.3-jadval

Guruhlangan jadval NATIJALARI

StudID	Mavzu	Mark
123	Matematika	4
	Fizika	5
124	Matematika	5
	Fizika	5
127	TPP	4

Yuqoridagi misol uchun GROUP BY dan foydalanilganda SELECT bandi faqat guruhlash amalga oshiriladigan ustunlarni belgilashi mumkin. Boshqa ustunlar, agar ular qiymatlar guruhini bittaga moslashtiradigan funksiyalarni yig'ish uchun argument sifatida harakat qilsa, ko'rsatilishi mumkin.

Quyidagilar asosiy agregat funksiyalari:

- o'rtacha qiymat – AVG ([DISTINCT I ALL] <ifoda>);
- maksimal – MAX ([DISTINCT | ALL] <ifoda>);
- minimal – MIN ([DISTINCT|ALL] <ifoda>);
- summa – SUM ([DISTINCT|ALL] <ifoda>);
- miqdor – COUNT ([DISTINCT I ALL] <ifoda>) yoki COUNT (*).

Agar funksiya DISTINCT kalit so'zi bilan ishlatilsa, avval takrorlashlar o'chiriladi, so'ngra funksiya qiymati hisoblanadi.

COUNT(*) funksiyasi berilgan to'plamdagi qatorlar sonini hisoblash yo'li bilan hisoblanadi. Barcha satrlarda NULL qiymati bo'lgan bitta ustundan iborat bo'lsa ham, barcha satrlar alohida hisoblanadi).

Guruhlash va jamlash funksiyalaridan foydalanishga misol sifatida quyida so'rov keltirilgan. Ma'lumotlar to'plami har bir talaba uchun uning identifikatori, o'tgan imtihonlar soni (SubjNum ustuni) va o'rtacha baho (AvgMark ustuni):

```
SELECT SubjNum AS StudID, COUNT(Subj),
AVG([Mark]) AS AvgMark
FROM Natijalar
GROUP BY StudID
```

MS SQL Server 2008 da shunga o'xshash so'rovni bajarayotganda, agar Mark bahosi bilan ustun uchun butun son turi (masalan, smallint) tanlangan bo'lsa, to'g'ri javobni olish uchun ma'lumotlarni aylantirish zarurligini aniq ko'rsatishingiz kerak bo'ladi. Quyidagi misolda CONVERT funksiyasi konvertatsiyani amalga oshiradi. Aks holda, operatsiyalar butun sonlar sifatida bajariladi va o'rtacha hisoblash natijasi kutilganidan farq qilishi mumkin:

```
SELECT SubjNum AS StudID, COUNT(Subj),
AVG(CONVERT(real,[Mark])) AS AvgMark
FROM Natijalar
GROUP BY StudID
```

Guruhlash bir nechta ustunlar bo’yicha ham amalga oshirilishi mumkin: buning uchun GROUP BY bo’limida ularni vergul bilan ajratib ko’rsatish kerak. Bundan tashqari, siz qatorlarni ustunlardagi ma'lumotlarning ba'zi funksiyalarining qiymati bo’yicha guruhlashingiz mumkin.

HAVING bandi guruh uchun shartni sinab ko’rish imkonini beradi. Formati quyidagicha:

HAVING shart-ifoda

Agar so’rovda GROUP BY operatori mavjud bo’lsa, u holda HAVING bo’limida shartni tekshirish natijasida shartli-ifoda shartining tekshiruvi "to’g’ri" ni qaytarmaydigan guruhlar guruhlangan jadvaldan o’chiriladi. Agar so’rovda GROUP BY bo’lmasa, lekin HAVING mavjud bo’lsa, FROM va WHERE bandlari bajarilgandan so’ng olingan jadval bitta guruh sifatida qabul qilinadi.

Faraz qilaylik, siz kamida ikkita imtihondan o’tgan talabaning identifikatori va o’rtacha bahosini ko’rsatmoqchisiz. Yuqorida ko’rib chiqilgan Transact-SQL-ga turdagi konvertatsiya qilish xususiyatlarini hisobga olgan holda, so’rov quyidagicha ko’rinadi:

```
SELECT StudID,
AVG(CONVERT(real, [Mark]) ) AS AvgMark
FROM Natijalar
GROUP BY StudID
HAVING COUNT (Subj)>=2
```

Boshqa holatda, kamida ikkita imtihonni ijobiy baho bilan topshirgan talabalar uchun o’rtacha ijobiy baho ("3", "4" yoki "5") talab qilinishi mumkin. Bu yerda siz individual yozuv va guruh uchun shartni tekshirishingiz kerak bo’ladi:

```
StudID ni tanlang,
AVG(CONVERT(real, [Mark])) AS AvgMark
FROM Natijalar
WHERE Belgi IN (3,4,5)
GROUP BY StudID
HAVING COUNT (Subj)>=2
```

Bunday so’rovlarda alohida satrlarga tegishli shartlar WHERE bandida, guruhlarga tegishli shartlar esa HAVING bandida yozilishi kerakligini yodda tutish kerak. Bundan tashqari, WHERE bandi iboralari GROUP BY va HAVING bandlaridan oldin qayta ishlanadi. Yuqoridagi misolda, agar talaba uchta imtihondan o’tgan bo’lsa, lekin

ulardan faqat bittasi ijobiy baho olgan bo’lsa, u haqidagi yozuv so’rov natijasiga kiritilmaydi.

Nazorat savollari.

1. SQL nima va uning yangi standarti qachon qabul qilingan?
2. DDL nima va u qanday vazifani bajaradi?
3. DML nima va u qanday vazifani bajaradi?
4. Qaysi buyruq yordamida yangi jadval yaratiladi?
5. Qaysi buyruqlar yordamida jadvaldagi zarur ma’lumotlar olinadi?
6. SELECT buyrug’i nima vazifani bajaradi?
7. FROM buyrug’i nima vazifani bajaradi?
8. So’rovlarda qanday agregat funksiyalari qo’llaniladi?

9-MAVZU. SQL TILI ASOSLARI

Reja:

1. Bir nechta jadvallardan ma'lumotlarni olish
2. So‘rov osti. Relyatsion algebra amallarini SQL tili yordamida amalga oshirish.
3. Ko‘rinishlar (VIEW)

Amalda, ko’pincha SELECT iborasi bir nechta jadvallardan ma'lumotlarni oladi. Xususan, bir nechta jadvallarning qo’shilish yoki Dekart mahsuloti natijasi ishlatilishi mumkin. Keling, ushbu operatsiyalarni SQL da yozish variantlarini ko’rib chiqaylik.

Yuqorida ta'kidlab o’tilganidek, FROM bo’limiga vergul bilan ajratilgan bir nechta jadvallarning nomlarini yozish ularning Dekart mahsuloti (barcha satrlarning "har biri bilan" kombinatsiyasi) natijasini olish imkonini beradi. Masalan, T1 va T2 jadvallari uchun shunday ko’rinadi.

```
SELECT  *
FROM T1, T2
```

WHERE bandiga jadvalni birlashtirish shartini qo’shish orqali siz Dekart ko’paytmasidan 9-qo’shilishni olishingiz mumkin. Quyida biz T1 va T2 jadvallaridagi Id ustunlari qiymatlarining tengligiga asoslangan birikmadan foydalanamiz:

```
SELECT  *
FROM T1, T2
WHERE Tl.Id=T2.ID
```

Xuddi shunday tarzda ikkitadan ortiq jadvallarni birlashtira olasiz. Umuman olganda, format quyidagicha bo’lishi mumkin:

```
SELECT <jadval>.<ustun> [,... p]
FROM <jadval1>, <jadval2> [,... p]
[WHERE <jadval1>.<ustun1>=
<jadval2>.<ustun2>
[AND <2-shart> [...]]]
```

Turli xil ulanish turlari qanday ishlashini tushuntirish uchun keling, talabalar va o’quv guruhlari bilan bir misoldan foydalanamiz.

Students

	StudID	Name	Group
1	123	Иванов И.И.	3082
2	124	Петров П.П.	3084
3	125	Сидоров С.С.	3092

Groups

	Group	Department
1	3082	ФТК
2	3083	ФТК
3	3092	РФФ

9.1-rasm. Manba jadvallari

Talabalar jadvalida talaba identifikatori, familiyasi va bosh harflari hamda o‘quv guruhi raqami (mos ravishda StudID, Name va Group) ko‘rsatilgan ustunlar mavjud. Guruhlar jadvalida o’quv guruhi (Guruh) va guruh tegishli bo’lim (bo’lim) raqami mavjud. Jadvallar xorijiy kalitlar bilan bog’lanmagan. Talabalarda Guruhlarda bo’lmagan 3084-guruh talabasi haqida yozuv mavjud, xuddi shunday Guruhlarda 3083-guruh mavjud, ularning soni Talabalarda ishlatilmaydi.

Yuqorida ta'kidlab o’tilganidek, ikkita jadvalning Dekart mahsuloti barcha qatorlarning "har biri bilan" kombinatsiyasini beradi.

So’rovning natijasi 9.2-rasmda ko’rsatilgan. Bizning misolimizda har bir talaba haqidagi ma'lumot o’quv guruhi haqidagi ma'lumotlar bilan barcha kombinatsiyalarda bu juda mazmunli emas.

а

	StudID	Name	Group	Group	Department
1	123	Иванов И.И.	3082	3082	ФТК
2	124	Петров П.П.	3084	3082	ФТК
3	125	Сидоров С.С.	3092	3082	ФТК
4	123	Иванов И.И.	3082	3083	ФТК
5	124	Петров П.П.	3084	3083	ФТК
6	125	Сидоров С.С.	3092	3083	ФТК
7	123	Иванов И.И.	3082	3092	РФФ
8	124	Петров П.П.	3084	3092	РФФ
9	125	Сидоров С.С.	3092	3092	РФФ

б

	Group	Department	Group	Department
1	3082	ФТК	3082	ФТК
2	3083	ФТК	3082	ФТК
3	3092	РФФ	3082	ФТК
4	3082	ФТК	3083	ФТК
5	3083	ФТК	3083	ФТК
6	3092	РФФ	3083	ФТК
7	3082	ФТК	3092	РФФ
8	3083	ФТК	3092	РФФ
9	3092	РФФ	3092	РФФ

9.2-rasm. So’rov natijalari:

a - Dekart mahsulotining natijasi; *b* - o’zaro bog’liqlik natijasi

Jadvalning dekart mahsulotini o’zi bilan olish kerak bo’lganda, o’z-o’zidan o’zaro bog’lanish holatiga *alohida e'tibor qaratish lozim.* Guruhlar jadvali uchun SQL da buni quyidagicha yozish mumkin:

```
SELECT *
FROM Groups CROSS JOIN
Groups AS Gr
```

Bu yerda birlashtirilgan jadvallar misollaridan kamida bittasi uchun FROM bo’limida taxallusni ko’rsatish muhim: berilgan misolda bu Gr taxallus. Aks holda, jadval namunalarini ajratib bo’lmaydi va ma'lumotlar bazasi ma'lumotlar bazasi katta ehtimollik bilan natija o’rniga xatoga yo’l qo’yadi.

Agar talaba, uning guruhi va guruh tegishli bo’lgan fakultet haqida ma'lumot olishingiz kerak bo’lsa, sizga *ekvikonnektsiya* kerak bo’ladi. "Equi" prefiksi ulanish sharti tenglikni o’z ichiga olishini bildiradi: bizning holatlarimizda, "Talabalar" va "Guruhlar" jadvallaridagi guruh raqamlarining tengligi.

Jadvallarning ichki teng qo’shilishi natijasi qatorlari asl jadvallarning dekart ko’paytmasiga tegishli bo’lgan jadval bo’ladi va ular uchun birlashma sharti - ustun qiymatlarining tengligi bajariladi.

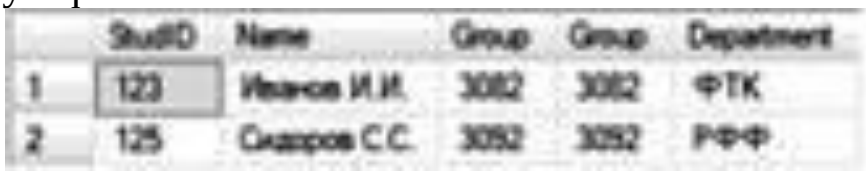
Bizning misolimizda "Talabalar" va "Guruhlar" jadvallari uchun ichki tenglik (SQL kalit so’zlari bilan nomga mos keladigan ustunlarni yozishning o’ziga xos xususiyatlarini hisobga olgan holda) quyidagicha yoziladi:

```
SELECT *
FROM Students CROSS JOIN
ON Students.[Group] = Groups.[Group];
```

yoki

```
SELECT *
FROM Students INNER JOIN
ON Students.[Group] = Groups.[Group];
```

JOIN konstruksiyasidagi INNER kalit so’zi sukut bo’yicha ishlatiladi, shuning uchun uni o’tkazib yuborishingiz mumkin. So’rovning natijasi 9.3-rasmda ko’rsatilgan. Ko’rinib turibdiki, talaba Petrov haqida hech qanday yozuv yo’q, chunki uning guruhi Guruhlar jadvalida yo’q.



9.3-rasm. Ichki tenglik natijasi (INNER JOIN)

Amalda ko’pincha ichki birikma ishlatiladi. Biroq, ba'zi hollarda, so’rov natijasida boshqa jadvalda mos kelmaydigan qatorlarni olish kerak bo’lishi mumkin. Bu tashqi (tashqi) ulanishlar yordamida amalga oshirilishi mumkin: chap (chap), o’ng (RIGHT) va to’liq (FULL). So’rovda OUTER kalit so’zini yozish shart emas, chunki LEFT, RIGHT yoki FULL JOIN allaqachon tashqi birlashma haqida gapirayotganimizni bildiradi.

Chap tashqi qo’shilishni amalga oshirishda so’rov natijasi ichki tenglamaning barcha satrlarini, shuningdek chap jadvaldagi satrlarni (JOIN konstruktsiyasida chapda joylashgan) o’z ichiga oladi, ular uchun o’ngda mos kelmaydi. Bunday qatorlar uchun o’ng jadvaldagi ma'lumotlar bilan to’ldirilgan ustunlar NULLni o’z ichiga oladi.

Yuqorida ta'kidlab o’tilganidek, OUTER kalit so’zi shart emas, quyida keltirilgan tashqi birlashma misollarida u o’tkazib yuboriladi. So’rovning natijasi 9.4-rasmda ko’rsatilgan. O’ng tomonda NULL qiymatlari bo’lgan qatorga e'tibor bering.

To'g'ri tashqi qo'shilish holatida rasm aksincha bo'ladi. So'rov natijasi o'ng jadvaldagi barcha satrlarni va chapdan ularga mos keladigan qatorlarni oladi. Agar mos kelmasa, chap jadvaldagi ma'lumotlar o'rniga NULL qo'yiladi.

So'rovning namunasi quyida ko'rsatilgan va uning natijasi 9.4-rasmda ko'rsatilgan. Bu yerda jadvalning chap tomonidagi NULL qiymatlari bo'lgan qatorga e'tibor bering:

```
SELECT * FROM Students LEFT JOIN
ON Students.[Guruh] = Guruhlar.[Guruh]
```

To'liq tashqi qo'shilish natijasi ichki tenglamaning barcha qatorlarini va o'ng va chap jadvallardagi mos kelmaydigan yozuvlarni o'z ichiga oladi. So'rovning namunasi quyida ko'rsatilgan va uning bajarilishi natijasi 9.4-rasmda ko'rsatilgan.

```
SELECT * FROM Students FULL JOIN
ON Students.[Guruh] = Guruhlar.[Guruh]
```

a

	StudID	Name	Group	Group	Department
1	123	Иванов И.И.	3082	3082	ФТК
2	124	Петров П.П.	3084	NULL	NULL
3	125	Сидоров С.С.	3092	3092	РФФ

б

	StudID	Name	Group	Group	Department
1	123	Иванов И.И.	3082	3082	ФТК
2	NULL	NULL	NULL	3083	ФТК
3	125	Сидоров С.С.	3092	3092	РФФ

в

	StudID	Name	Group	Group	Department
1	123	Иванов И.И.	3082	3082	ФТК
2	124	Петров П.П.	3084	NULL	NULL
3	125	Сидоров С.С.	3092	3092	РФФ
4	NULL	NULL	NULL	3083	ФТК

9.4-rasm. Tashqi ekvialoqa natijalari:

a - chap; *b* - o'ng; *c* - to'liq

Keling, tashqi birlashma natijasiga kiritilgan, ammo ichki birlashma natijasida etishmayotgan yozuvlarni qanday olish kerakligini ko'rib chiqaylik. Buni amalga oshirish uchun siz manba jadvallarining asosiy ustunlari NULL qiymatlarga ega bo'lgan qatorlarni tashqi birlashmadan olishingiz kerak:

```
SELECT *
FROM Students FULL JOIN
ON Students.[Guruh] = Guruhlar.[Guruh]
WHERE (Students.[StudID] IS NULL )
OR (Guruhlar. [Guruh] NULL)
```

Ko’rib chiqilayotgan misollarda jadvallar xorijiy kalitlari bilan bog’lanmagan. Biroq, masalan, o’ng jadvalda chapga havola qiluvchi tashqi kalit bo’lsa va tashqi kalit bo’lgan ustun (yoki ustunlar) uchun NOT NULL yaxlitlik cheklovi aniqlangan bo’lsa, bunday ikkita jadval uchun RIGHT natijasi bo’ladi. Tegishli qiymatlar teng bo’lishi sharti bilan JOIN har doim INNER JOIN natijasi bilan bir xil bo’ladi, chunki o’ng jadvaldagi tashqi kalit tegishli potentsial kalitda mavjud bo’lmagan qiymatlarni o’z ichiga olmaydi. chap stol.

Teng qo’shilish yasashda qo’shilish shartida tenglik operatori qo’llaniladi. Boshqa operatorlardan ham foydalanish mumkin; <, <=, >, >= va boshqalar. Ingliz tilidagi adabiyotda bunday aloqa "pop-equi join" ("non-equi-join") deb ataladi.

Masalan, siz Students jadvalidagi barcha yozuvlar kombinatsiyasini boshqa talabalarga nisbatan ushbu jadvaldagi barcha yozuvlar bilan olishni xohlaysiz.

```
SELECT *
FROM Students INNER JOIN Students
AS St ON Students.StudID <> St.StudID
```

	StudID	Name	Group	StudID	Name	Group
1	124	Петров П.П.	3084	123	Иванов И.И.	3082
2	125	Сидоров С.С.	3092	123	Иванов И.И.	3082
3	123	Иванов И.И.	3082	124	Петров П.П.	3084
4	125	Сидоров С.С.	3092	124	Петров П.П.	3084
5	123	Иванов И.И.	3082	125	Сидоров С.С.	3092
6	124	Петров П.П.	3084	125	Сидоров С.С.	3092

9.5-rasm. Tengsizlik sharti bilan qo’shilish natijasi

So’rov osti

So’rov osti boshqa SELECT ichida joylashgan SELECT bayonotidir. Keling, bir misolni ko’rib chiqaylik. Kitob jadvalida nashrlar haqida ma'lumotlar mavjud. Uning ustunlari BookID - nashr identifikatori, Muallif - muallif, Sarlavha - sarlavha, Nashriyot - nashriyot, yil - nashr etilgan (9.1-jadval).

9.1-jadval

Jadval kitobi

BookID	Muallif	Sarlavha	Nashriyotchi	Yil
1	K. J. Sana	Ma'lumotlar bazasi tizimlariga kirish	Uilyams	1999 yil
2	T.S. Karpova	Ma'lumotlar bazalari: modellar,	Piter	2002 yil

		ishlab chiqish, amalga oshirish		
3	V.V.Kirillov, G.Yu. Gromov	Relyatsion ma'lumotlar bazalariga kirish	BHV-Peterburg	2009 yil

Book1nLib jadvali kutubxonada saqlangan nashrlarning nusxalari haqidagi ma'lumotlarni o’z ichiga oladi: LibID - kitob inventar raqami, BookID - nashr identifikatori (xorijiy kalit), Status - holati (9.2-jadval).

9.2-jadval

Book1nLib **jadvali**

LibID	BookID	Holat
10	1	saqlanadi
o’n bir	1	berilgan sana

Kichik so’rovni o’z ichiga olgan so’rovga misol keltiramiz. Aytaylik, siz Kitob jadvalida tasvirlangan nashrlar haqidagi ma'lumotlarni ko’rsatishingiz kerak, lekin ular Book1nLib jadvalida yo’q. So’rov quyidagicha ko’rinadi:

```
SELECT *
FROM Book
WHERE  BookID NO IN
(SELECT BookID FROM Book1nLib DISTINCT)
```

Bu o’zaro bog’liq bo’lmagan so’rov ostini qayta ishlashning bir misoli : pastki so’rov qaytaradigan natija tashqi SELECT iborasi tomonidan qaysi jadval qatori hozirda qayta ishlanayotganiga bog’liq emas. Bunda kichik so’rov faqat bir marta bajariladi, uning natijalari vaqtinchalik obyektda saqlanadi va so’rovning tashqi qismi bajarilganda, allaqachon shakllangan to’plam uchun NO IN sharti tekshiriladi. Korrelyatsiyasiz ishlov berishda pastki so’rovni o’z ichiga olgan so’rov ichkaridan tashqariga amalga oshiriladi: birinchi navbatda pastki so’rov, keyin esa tashqi so’rov bajariladi.

SQL tilidan foydalangan holda relyatsion algebra operatsiyalarini amalga oshirish

Ushbu bo’limda relyatsion algebra operatsiyalarini SQLda qanday yozish mumkinligi ko’rib chiqiladi. Biroz oldinga qarab, shuni ta'kidlashimiz mumkinki, relyatsion algebraning barcha operatsiyalarini SQL vositalaridan foydalangan holda amalga oshirish qobiliyati ushbu tilning relyatsion to’liqligini isbotlaydi.

A va B toifalariga mos keladigan ikkita munosabat munosabatlarining birlashishi quyidagi usullardan birida yozilishi mumkin:

A UNION B

SQL dan foydalanganda siz UNION bayonotidan foydalanib ikkita so’rov natijalarini birlashtirishingiz mumkin. Bu so’rovlar bir xil sarlavhali jadvallarni qaytarishini talab qiladi, aks holda xatolik yuzaga keladi.

Agar A va B jadvallarida mos keladigan qatorlar mavjud bo’lsa, dublikatlar UNION operatsiyasi natijasiga kiritilmaydi.

Ikki mos keladigan munosabatlarning *kesishishi quyidagicha yoziladi:*

A ∩ B

Kesishish

Xuddi shunday vaziyatda SQL so’rovini yozishda INTERSECT operatoridan foydalaniladi. Oldingi holatda bo’lgani kabi, SELECT iboralari tomonidan yaratilgan sarlavhalar o’rtasida moslik talab qilinadi:

```
SELECT *
FROM B INTERSECT *
```

Agar oddiy kalit bilan bitta jadvaldan tanlangan yozuvlar to’plami uchun kesishuvni olishingiz kerak bo’lsa, siz IN shartidan foydalanishingiz mumkin. Masalan, A jadvalida ID kaliti aniqlangan deylik. Quyidagi so’rov fragmenti tashqi so’rovning pastki so’rov shartini ham, qo’shimcha shartlarini ham qondiradigan ko’plab qatorlarni olish imkonini beradi (aslida bu kesishmadir):

```
SELECT *
FROM A
WHERE ID IN (SELECT FROM A WHERE ID)
```

Relyatsion algebrada *ayirishni quyidagicha yozish mumkin*: A-B

A va B munosabatlari turi bo’yicha mos bo’lishi kerak. SQL-da shunga o’xshash funksiya EXCEPT operatori tomonidan amalga oshiriladi, bu esa operand jadvallari sarlavhalarining mos kelishini talab qiladi:

```
SELECT *
FROM A
EXCEPT
SELECT * FROM IN
```

Agar bitta jadvaldagi ikkita qator kichik to’plami orasidagi farqni oddiy kalit bilan olishingiz kerak bo’lsa, ayirishni yozish uchun WHERE bandidagi NOT IN bandidan foydalanish mumkin.

View

Ko’rinish , mohiyat, ma'lumotlar bazasida alohida nom ostida saqlanadigan tanlov so’rovidir . Ko’rinish o’zining saqlangan ma'lumotlariga ega emas, lekin ma'lumotlar bazasining joriy holati uchun SELECT ifodasini qayta ishlash natijasini qaytaradi.

Keling, ko’rinishni yaratish uchun biroz soddalashtirilgan formatni ko’rib chiqaylik:

CREATE VIEW <nom> [(ustunlar ruyxati)]
AS <surov>

Quyida "3082" guruhidagi talabalar ro’yxatini o’z ichiga olgan misol ko’rinishi keltirilgan. SQL kalit so’ziga mos keladigan Guruh ustunining nomi Transact-SQLda odatdagidek kvadrat qavslar ichiga olinadi.

Ko’rinish nomli ma'lumotlar bazasi obyekti bo’lib, u yaratilgandan so’ng, oddiy jadvaldagi kabi undan ma'lumotlarni so’rashingiz mumkin. Ma'lumotlarni qo’shish, o’zgartirish va o’chirishga kelsak, bir qator cheklovlar mavjud. Ba'zi ko’rinishlar yangilanishi mumkin, ba'zilari esa yangilanmaydi. Bu ko’rinish aniqlangan saqlangan jadvallarga kiritilgan o’zgarishlarni aniq ko’rsatish mumkinmi yoki yo’qligiga bog’liq. Turli MBBTlar o’rtasida ba'zi farqlar bo’lishi mumkin. Yuqorida tavsiflangan Stud3082 taqdimoti shu nuqtai nazardan juda qulay, u sizga ma'lumotlarni yangilash va o’zgartirish imkonini beradi.

Boshqacha qilib aytadigan bo’lsak, 3082-guruh talabalari haqidagi ma'lumotlarni qaytaruvchi ko’rinish orqali jadvalga 3083-guruh talabalari haqidagi yozuvni qo’shish mumkin edi. Buning oldini olish uchun siz SQL Serverda ko’rinish yaratishingiz kerak bo’ladi.

Ko’rinishlar ko’pincha ma'lumotlarga kirishni boshqarish vazifalari uchun ishlatiladi. Bunday hollarda faqat foydalanuvchi kirishi kerak bo’lgan ma'lumotlarni qaytaradigan ko’rinish yaratiladi. Bu foydalanuvchiga ko’rinishdagi ma'lumotlarni o’qish huquqini beradi va manba jadvallaridan ma'lumotlarni o’qish huquqlarini olib tashlaydi.

Ko’rishlar, shuningdek, ANSI/SPARC arxitekturasini ko’rib chiqishda avval muhokama qilinganidek, mantiqiy ma'lumotlar

mustaqilligini ta'minlashi mumkin. Agar ilova ma'lumotlarni bevosita saqlangan jadvallardan emas, balki ko’rinishlardan olsa, bu sizga jadvallar tuzilishiga o’zgartirish kiritish va shunga mos ravishda ko’rinishni moslash orqali dastur dasturidan o’zgarishlarni yashirish imkonini beradi. Bunday holda, ma'lumotlar bazasining ichki tuzilishiga o’zgartirishlar kiritilganda, foydalanuvchi dasturiy ta'minotida hech narsani o’zgartirish kerak bo’lmaydi.

Ko’pgina MBBTlar ma'lumotlar bazasida ko’rinishlarni oldindan qayta ishlangan, kompilyatsiya qilingan shaklda saqlaydi. Bu shunga o’xshash so’rovga nisbatan ularning bajarilishi tezligini oshiradi. Biroq, bu yerda ma'lum bir MBBT ishlashining kutilmagan xususiyatlari paydo bo’lishi mumkin.

Nazorat savollari

1. Bir nechta jadvallardan ma'lumotlarni olish uchun qanday SQL so‘rovlarini yozish mumkin?
2. JOIN operatsiyasini qanday ishlatish mumkin?
3. Subquery (ichki so‘rov) qanday ishlatiladi?
4. SELECT, UPDATE, DELETE va INSERT so‘rovlari qanday ishlatiladi?
5. WHERE shart operatorlari nima?
6. Agregat funksiyalar (SUM, COUNT, AVG, MAX, MIN) qanday ishlatiladi?
7. View (qarash) nima?
8. View yaratish va unga murojaat qanday amalga oshiriladi?
9. View-ni yangilash va o‘chirish qanday bajariladi?

10-MAVZU. TRANZAKSIYALAR.

Reja:

1. Tranzaktsiya nima?
2. Tranzaksiya haqida tushuncha. Tranzaksiyalarni boshqarish va ma'lumotlar bazasi yaxlitligi
3. Tranzaksiya xususiyatlari. Tranzaktsiyalarni bajarish usullari
4. Tranzaktsiyalar jurnali.

Tranzaksiya - bu ma'lumotlar bazasida operatsiyalar ketma-ketligidir, ular butun MBBT tomonidan ko’rib chiqiladi. Tranzaktsiya muvaffaqiyatli yakunlangan va MBBT ushbu operatsiyani tashqi xotirada amalga oshirgan ma'lumotlar bazasini o’zgartiradi yoki ushbu o’zgarishlarning hech biri ma'lumotlar bazasining holatiga ta'sir qilmaydi. Tranzaksiya tushunchasi ma'lumotlar bazasining mantiqiy yaxlitligini ta'minlash uchun zarurdir. Agar siz Employee va Department fayllari bilan bizning ma'lumot tizimimiz misolini eslasangiz, yangi xodimni yollashda ma'lumotlar bazasining yaxlitligini buzmaslikning yagona usuli bu Employee va Department fayllaridagi elementar operatsiyalarni bitta operatsiyaga birlashtirishdir. Shunday qilib, tranzaksiyalar mexanizmini saqlab turish hatto bitta foydalanuvchida joylashgan ma'lumotlar bazasi uchun ham zaruriy shartdir (agar bunday tizim MBBT nomiga loyiq bo’lsa). Ammo ko’p foydalaniladigan ma'lumotlar bazalarida tranzaksiya tushunchasi muhimroqdir.

Har bir operatsiya ma'lumotlar bazasining izchil holati bilan boshlanadigan va ushbu holat tugallangandan so’ng uni izchil qoldiradigan xususiyat foydalanuvchiga ma'lumotlar bazasiga nisbatan harakatlarning birligi sifatida foydalanish tushunchasini ishlatishni juda qulay qiladi. Birlamchi tranzaktsiyalarni MBBT tomonidan tegishli ravishda boshqarish bilan, har bir foydalanuvchi, prinsipial ravishda, ma'lumotlar bazasining yagona foydalanuvchisi kabi his qilishi mumkin (aslida bu biroz idealizatsiya qilingan ko’rinishdir, chunki ba'zi holatlarda ko’p foydalanuvchi MBBT foydalanuvchilari hamkasblarining borligini sezishlari mumkin). Ko’p foydalanuvchi MBBTda operatsiyalarni boshqarish bilan bog’liqlik operatsiyalarni seriyalashtirishning muhim tushunchalari va tranzaksiyalar aralashmasi uchun ketma-ketlashtirilgan ijro rejasi hisoblanadi. Bir vaqtning o’zida

bajariladigan bitimlarni ketma-ketlashtirish deganda, ularning ishini rejalashtirishning shunday tartibini tushuniladi, bunda tranzaksiyalar aralashmasining umumiy samarasi ularning ba'zi bir ketma-ket bajarilishiga ta'sir qiladi. Tranzaksiyalar aralashmasi uchun ketma-ketlashtirilgan ijro rejasi bu operatsiyalarni seriyalashtirishga olib keladi. Agar bitimlar aralashmasining chinakam ketma-ket bajarilishiga erishish mumkin bo’lsa, unda har bir foydalanuvchi tashabbusi bilan tuzilgan bo’lsa, boshqa operatsiyalarning mavjudligi ko’rinmas bo’ladi (bitta foydalanuvchi rejimiga nisbatan ba'zi sekinlashuvlardan tashqari). Tranzaksiya bo’yicha ma'lumotlar bazasida bajariladigan va uni bir izchil holatdan ikkinchisiga mos holatga o’tkazadigan operatsiyalar ketma-ketligi. Tranzaksiya ma'lumotlar bazasida bo’linmaydigan, foydalanuvchi nuqtai nazaridan ma'noga ega bo’lgan harakat sifatida qaraladi, ya'ni bu tizimning mantiqiy ishlash birligi hisoblanadi. Ma'lumotlar bazasi sessiyasi sodir bo’lganda, tranzaksiya boshlanadi.

Masalan, bankomat orqali pul o’tkazmasi bo’lishi mumkin. Miqdori 100$ joriy hisobdan karta hisobiga o’tkaziladi. Dastur joriy hisobdan summani chiqaradi va keyin uni karta hisobiga qo’shadi. Dastur ishlayotganda, birinchi o’zgartirish amalga oshirilgandan so’ng, elektr uzilishi sodir bo’ladi va karta hisobi ko’paymaydi. Bunday vaziyatga yo’l qo’ymaslik uchun har ikkala jamoa ham bitim tuzishi kerak. Agar bitimning barcha buyruqlari bajarilmasa, tranzaksiya qaytariladi.

Tranzaksiya haqida tushuncha. Tranzaksiyalarni boshqarish va ma'lumotlar bazasi yaxlitligi: ma'lumotlar bazalari bilan har qanday axborot tizimlarining ishlashi uchun eng muhim talablardan biri saqlangan ma'lumotlarni izchil, mantiqiy izchil holatda saqlashdir. Faqatgina ushbu talab bajarilganda, foydalanuvchilar uchun har qanday qiymatdagi ma'lumotlar. Ma'lumotlar bazasi, agar u ma'lumotlar bazasi uchun belgilangan barcha yaxlitlik cheklovlarini qanoatlantirsa (qondirsa) izchil (mos keladigan) holatda bo’ladi. Amalda esa, ma'lumotlar bazasidagi ma'lumotlar statik emas, balki vaqt o’tishi bilan o’zgarib tursa, bu talabni ta'minlash bir qator jiddiy qiyinchiliklar bilan bog’liq. Gap shundaki, ma'lumotlar ustida bir qator operatsiyalarni bajarishda, undagi ma'lumotlar mantiqiy jihatdan mos kelmaydigan, integral bo’lmagan holatda bo’lganida, ma'lumotlar bazasi holatidan qochish mutlaqo mumkin emas. Bir misolni ko’rib chiqing.

Ma’lumotlar bazasida ikkita munosabat bo‘lsin: ikki atributga ega Student_Name va GPA atributlariga ega bo‘lgan Talaba munosabati va Talaba_Name, Kurs, Baho atributlari bilan Progress munosabati. Talaba munosabatlarining Average_ball atributining qiymati ma'lum bir talabaning u o’tgan barcha fanlar bo’yicha olgan baholarining o’rtacha qiymatidir.

Ma'lumotlar bazasida ma'lumotlar nomuvofiqligining yuzaga kelishining yana bir mumkin bo’lgan sababi bitta ma'lumotlar bazasining bir nechta foydalanuvchilari tomonidan ma'lumotlarni o’zgartirish operatsiyalarini bir vaqtning o’zida bajarishdir. Ma'lumotlar o’zgartirilganda ma'lumotlar bazasi bir izchil holatdan ikkinchisiga to’g’ri o’tkazilishini ta'minlash uchun MBBTda ishlab chiqilgan mexanizm *tranzaksiyalarni boshqarish mexanizmidir*. Tranzaksiya mexanizmining mavjudligi va qo’llab-quvvatlanishi ma'lumotlar bazasining rivojlanish darajasi, u tomonidan boshqariladigan ma'lumotlar bazasidagi ma'lumotlar yaxlitligi darajasi va ularning apparat va dasturiy ta'minotning mumkin bo’lgan nosozliklari va ma'lumotlari bilan ishlashda ziddiyatlarga chidamliligining muhim ko’rsatkichlaridan biridir. Tranzaksiya ma'lumotlar bazasining mantiqiy ish birligi bo’lib, u bir butun sifatida bajariladigan va ma'lumotlar bazasini bir izchil holatdan ikkinchisiga o’tkazadigan ma'lumotlarni manipulyatsiya qilish bayonotlarining ketma-ketligidir. Bitimning shiori "hammasi yoki hech narsa". Bitim to’rtta muhim xususiyatga ega:

• Atom,
• Muvofiqlik,
• Izolyatsiya
• Chidamlilik.

Ular odatda AMICH xossalari deb ataladi (xususiyatlarning birinchi harflaridan olingan).

Tranzaksiya xususiyatlari. Tranzaktsiyalarni bajarish usullari

Tranzaktsiyalarning har xil modellari mavjud bo’lib, ularni har xil xususiyatlarga qarab tasniflash mumkin, jumladan bitim tuzilishi, bitim ichidagi kelishuv, davomiyligi va boshqalar.

Hozirgi vaqtda bitimlarning quyidagi turlari ajratiladi: tekis yoki klassik bitimlar, zanjirli bitimlar va ichki operatsiyalar. Yassi operatsiyalar atomiklik, mustahkamlik, izolyatsiya va chidamlilikning klassik xususiyatlari bilan tavsiflanadi.

- *Atom* xususiyati shundaki, bitim umuman bajarilishi yoki umuman bajarilmasligi kerak.
- *Muvofiqlik* xususiyati, tranzaksiya davom etar ekan, ma'lumotlar bir izchil holatdan ikkinchisiga mos holatga o’tishini ta'minlaydi - tranzaksiya ma'lumotlarning o’zaro muvofiqligini buzmaydi.
- *Izolyatsiya* xususiyati shuni anglatadiki, ma'lumotlar bazasiga kirish uchun raqobatlashadigan tranzaktsiyalar bir -biridan ajratilgan holda fizik tartibda qayta ishlanadi, lekin foydalanuvchilarga ular parallel ravishda bajarilgandek ko’rinadi.
- *Chidamlilik* xususiyati shuni anglatadiki, agar bitim muvaffaqiyatli yakunlansa, u holda kiritilgan ma'lumotlar hech qanday holatda, hatto keyingi xatolar bo’lsa ham, yo’qolishi mumkin emas.

Tranzaktsiyani bajarish uchun ikkita variant mavjud:

- agar barcha bayonotlar muvaffaqiyatli bo’lsa va tranzaktsiya paytida hech qanday apparat yoki dasturiy ta'minot buzilmasa, bitim tuziladi. (Majburiy
  bu diskka tranzaksiya paytida qilingan ma'lumotlar bazasidagi o’zgarishlar yozilishi). Agar bitim tuzilmagan bo’lsa, bu o’zgarishlarni qaytarish mumkin va ma'lumotlar bazasi operatsiya boshlangan holatga qaytarilishi mumkin. Bitimni amalga oshirish bitimning barcha natijalari doimiy ekanligini bildiradi. Ular boshqa bitimlarga faqat joriy bitim tuzilgandan keyingina ko’rinadi.
- agar operatsiyani bajarish paytida xatolik yuz bersa, ma'lumotlar bazasi asl holatiga qaytarilishi kerak. Tranzaktsiyani qaytarish - bu SQL bayonotlari tomonidan kiritilgan barcha o’zgarishlarni joriy kutilayotgan tranzaktsiya tanasiga qaytaradigan harakat.

Tranzaktsiyalar jurnali

Oraliq davlatlarni saqlab qolish, bitimni tasdiqlash yoki qaytarish tamoyilini amalga oshirish, uni qo’llab-quvvatlash uchun maxsus mexanizm yordamida amalga oshiriladi, uni tuzish jurnali deb nomlangan tizim tuzilishi yaratilgan. Tranzaktsiyalar jurnalida ma'lumotlar bazasini o’zgartirish yozuvlari ketma -ketligi mavjud. Ma'lumotlar bazasida ishonchli ma'lumotlarni saqlashni ta'minlash uchun mo’ljallangan. Bu har qanday uskuna va dasturiy ta'minotdagi nosozliklardan so’ng ma'lumotlar bazasining izchil holatini tiklash

qobiliyatini bildiradi. Ma’lumotlar bazasini tiklashning umumiy tamoyillari:

- bajarilgan operatsiyalar natijalari ma'lumotlar bazasining tiklangan holatida saqlanishi kerak;
- bajarilmagan operatsiyalar natijalari ma'lumotlar bazasining tiklangan holatida bo’lmasligi kerak.

Bu shuni anglatadiki, ma'lumotlar bazasining oxirgi izchil holati tiklanadi.

Ma'lumotlar bazasi holatini tiklash zarur bo’lgan quyidagi holatlar bo’lishi mumkin:

- RAM tarkibining to’satdan yo’qolishidan qutulish (yumshoq nosozlik). Bu holat quyidagi holatlarda yuz berishi mumkin: favqulodda o’chirish paytida yoki o’lik protsessor ishlamay qolganda. Vaziyat muvaffaqiyatsizlikka uchragan vaqtda RAM buferlarida bo’lgan ma'lumotlar bazasining yo’qolishi bilan tavsiflanadi.
- Ma'lumotlar bazasining asosiy tashqi ma'lumotlari buzilganidan keyin tiklash (qattiq ishlamay qolishi).

Tizim ham kichik uzilishlardan (masalan, muvaffaqiyatsiz bitimlar), ham katta uzilishlardan keyin (masalan, elektr uzilishlari, qattiq uzilishlar) tiklanishi kerak.

Jurnal ma'lumotlarini saqlashning ikkita asosiy varianti mavjud. 1-variantda, har bir tranzaksiya uchun ma'lumotlar bazasini o’zgartirishning alohida lokal jurnali yuritiladi. Bu jurnallar lokal jurnallar deb ataladi. Ular lokal operatsiyalarni qaytarish uchun ishlatiladi. Bundan tashqari, ma'lumotlar bazasini o’zgartirishning umumiy jurnali yuritiladi, u yumshoq va qattiq nosozliklardan keyin ma'lumotlar bazasini tiklash uchun ishlatiladi. Ushbu yondashuv sizga individual tranzaktsiyalarni tezda bajarishga imkon beradi, lekin natijada lokal va umumiy jurnallarda ma'lumotlar takrorlanadi. Shuning uchun, ikkinchi variant tez -tez ishlatiladi - ma'lumotlar bazasini o’zgartirishning umumiy jurnalini yuritish, bu ham individual qaytarish paytida ishlatiladi.

Jurnalning umumiy tuzilishi ketma-ket fayl ko’rinishida ifodalanishi mumkin, bunda tranzaksiya paytida yuzaga keladigan ma'lumotlar bazasidagi har bir o’zgarish qayd etiladi. Barcha tranzaktsiyalarda ichki raqamlar mavjud, shuning uchun barcha

tranzaktsiyalar tomonidan kiritilgan barcha o’zgarishlar operatsiyalar jurnalida qayd etiladi.

Har bir jurnal yozuvi tegishli bo’lgan tranzaksiya raqami va u o’zgartiradigan atributlarning qiymatlari bilan belgilanadi, bundan tashqari, jurnaldagi har bir tranzaksiya uchun tranzaktsiyani boshlash va tugatish buyrug’i qayd etiladi. Ishonchliligi oshishi uchun tranzaktsiyalar jurnali ko’pincha ma'lumotlar bazasi ma'lumotlar bazasidagi ma'lumotlar hajmidan ko’p marotaba ko’p bo’lgan ma'lumotlar bazasi tizimli vositalar yordamida takrorlanadi.

Nazorat savollari:

1. Tranzaksiya nima?
2. Tranzaksiyalar qanday shakllarda bo’lishi mumkin?
3. Tranzaksiya qanday holatlarda yuzaga keladi?
4. Tranzaksiyalar qanday maqsadlar uchun amalga oshiriladi?
5. Tranzaksiyalar qanday turdagi ma'lumotlar orqali identifikatsiya qilinadi?
6. Tranzaksiyalar qanday ko’rinishda saqlanadi?
7. Tranzaksiyalar nima uchun ajratiladi?
8. Tranzaksiyalar qanday shaklda aniqlanadi va haqiqiylik sinovdan o’tkaziladi?
9. Tranzaksiyalar qanday shaklda muhokama qilinishi mumkin?
10. Tranzaksiyalar qanday shaklda baholay oladi?

11-MAVZU. MA’LUMOTLARNI FIZIK SAQLASHNI TASHKIL ETISH VA INDEKSLARNI YARATISH.

Reja:

1. Fizik ma'lumotlarni saqlashni tashkil qilish va indekslarni yaratish
2. Ma’lumotlarni saqlashni tashkil etish.
3. Indekslarni tashkil qilish.
4. Indekslarni yaratish va ularni boshqarish.

Ushbu bobda ma'lumotlarni diskda saqlashni tashkil etish va indekslar yordamida ma'lumotlar bazasi so’rovlarini bajarish tartibini optimallashtirish bilan bog’liq masalalar muhokama qilinadi. *Indeks* - saqlangan yozuvlarni mantiqiy tartibga solish va ma'lumotlarni olish tezligini oshirish uchun yaratilgan ma'lumotlar bazasi obyekti. Ma'lumotlar bazasi jadvali indeksi ma'lumotnoma yoki darslikdagi alifbo indeksi bilan bir xil rol o’ynashi mumkin (ingliz tilida alifbo indeksi ham indeks hisoblanadi).

Ma'lumotlarni saqlash va indekslarni qurish mavzulari o’zaro bog’liqdir, bu ularni birgalikda ko’rib chiqishni tushuntiradi.

Ma'lumotlarni saqlashni tashkil etish

Agar zamonaviy server relyatsion MBBTlar haqida gapiradigan bo’lsak, ular ma'lumotlar bazasi bilan ishlash uchun foydalanadigan disk maydonini tranzaktsiyalar jurnallari va ma'lumotlarni saqlash uchun bo’sh joyga bo’lish mumkin.

Har bir tranzaksiyani amalga oshirishdan oldin, MBBT tranzaktsiya tomonidan kiritilgan o’zgarishlar haqida jurnalga yozuv kiritadi. Ma'lumotlar bazasini tiklash uchun jurnallar talab qilinadi va normal rejimda ishlaganda, kiritilgan o’zgartirishlar u yerda ketma-ket yoziladi va jurnaldan hech qanday o’qish amalga oshirilmaydi.

Ma'lumotlarni saqlash maydoni bilan ishlash yanada xilma-xildir: bu yerda yozish ham, o’qish ham amalga oshiriladi va bu operatsiyalar navbat bilan diskning turli sohalariga ta'sir qilishi mumkin. Shuning uchun ko’p ishlatiladigan ma'lumotlar bazalari uchun jurnal va ma'lumotlarni iloji boricha alohida fizik disklarga joylashtirish tavsiya etiladi. Agar jurnal va ma'lumotlar bir xil diskda bo’lsa, disk boshlarini diskning turli joylariga o’tkazish uchun qo’shimcha vaqt talab etiladi.

Bu jurnal bilan ishlash tezligini pasaytiradi va shunga mos ravishda vaqt birligida qayta ishlangan tranzaktsiyalar sonini kamaytiradi.

Biroq, saqlash joyiga qaytaylik. MBBTga qarab, ma'lumotlar bazasini saqlash uchun konteyner sifatida alohida fayllar, kataloglar yoki hatto ajratilgan disk bo’limlaridan foydalanish mumkin. Masalan, SQL Server fayllardan konteyner sifatida foydalanadi. Qabul qilingan nomlash konventsiyasiga ko’ra, jurnal fayli ldf kengaytmasini, asosiy yoki asosiy ma'lumotlar bazasi faylini - mdf kengaytmasini va boshqa barcha ma'lumotlar fayllarini - ndf kengaytmasini oladi. Ma'lumotlar bazasi tizimi obyektlari har doim birlamchi ma'lumotlar faylida (mdf fayli) joylashgan. Foydalanuvchi ma'lumotlar bazasi obyektlari ham asosiy, ham boshqa ma'lumotlar fayllarida joylashtirilishi mumkin. Ma'lumotlar fayllari guruhlarga ajratilgan. SQL Server ma'lumotlar bazasi har doim PRIMARY asosiy fayl guruhiga ega va qo’shimchalarini yaratish mumkin.

Shuni ta'kidlash kerakki, agar SQL Server ma'lumotlar bazasi fizik jihatdan fayllar to’plami sifatida tashkil etilgan bo’lsa, mantiqan ma'lumotlar bazasi *sxemalardan* iborat. *Sxema* ma'lumotlar bazasi obyektlari tegishli bo’lgan mantiqiy darajadagi konteynerdir. Yangi yaratilgan ma'lumotlar bazasi allaqachon bir nechta sxemalarga ega bo’ladi: dbo, sys, information_schema. Dbo sxemasi yangi foydalanuvchi obyektlari uchun standart sxema bo’lib, tizim obyektlari tomonidan sys va information_schema ishlatiladi. Ma'lumotlar bazasidagi CREATE SCHEMA bayonotidan foydalanib, siz yangi sxemalar yaratishingiz mumkin.

Har qanday obyekt qandaydir sxemaga tegishli. Misol uchun, to’liq malakali jadval nomi lib.Book1.MyTest bo’lishi mumkin. Bu Book1 jadvali MyTest ma'lumotlar bazasidagi lib sxemasida ekanligini anglatadi. Agar foydalanuvchi obyektining nomi ma'lumotlar bazasini o’z ichiga olmasa, joriy ma'lumotlar bazasidan foydalaniladi, agar sxema nomi bo’lmasa, standart sxema ishlatiladi (ko’pincha dbo).

Sxemalardan foydalanish ma'lumotlar bazasi obyektlarini mantiqiy guruhlash imkonini beradi. Misol uchun, marketing bo’limi tomonidan ishlatiladigan jadvallar va ko’rinishlar marketing sxemasida joylashtirilishi mumkin. Bundan tashqari, sxemalardan foydalanish ma'lumotlar bazasi obyektlariga kirishni boshqarish konfiguratsiyasini soddalashtirishga imkon beradi.

Ko’pgina zamonaviy server ma'lumotlar bazasi ma'lumotlar bazasi ma'lumotlarni saqlash uchun konteynerlarni (xususan, fayllarni) sahifalarni tashkil qilishdan foydalanadi: ma'lumotlar bazasi ma'lumotlar bazasi ma'lumotlarini diskka qat'iy belgilangan o’lchamdagi bo’laklarda o’qiydi yoki yozadi. Microsoft SQL Serverda sahifa hajmi qat'iy belgilangan, u 8 KB ga teng. Xizmat tuzilmalari mavjudligi sababli sahifadagi ma'lumotlar uchun 8060 bayt ajratilgan. Boshqa MBBTlarda sahifa hajmi ishlab chiquvchilar tomonidan belgilangan ro’yxatdan tanlanishi mumkin, masalan, IBM DB2 4, 8, 16 yoki 32 KB o’lchamdagi sahifalardan foydalanishi mumkin.

Ma'lumotlar fayli sahifasida birinchi navbatda sarlavha, so’ngra ma'lumotlar qatorlari joylashtiriladi va sahifaning oxirida qatorlarni o’zgartirish jadvali mavjud. Ofset jadvali sahifadagi har bir satr uchun bitta yozuvni o’z ichiga oladi, bu satrning birinchi bayti sahifa boshidan qanchalik uzoqda ekanligini ko’rsatadi. Qator ofset jadvali yozuvlari sahifadagi qatorlar ketma-ketligiga teskari tartibda joylashgan.

Biroq, sahifa ham nisbatan kichik obyektdir, shuning uchun qo’shni sahifalar *kengaytmalarga* birlashtiriladi. Kengaytmalar ma'lumotlarni saqlash uchun joyni tashkil qilishning asosiy birliklari hisoblanadi.

SQL Serverda kengaytma fizik jihatdan ketma-ket joylashgan sakkiz sahifadan iborat. Ma'lumotlar bazasida kichik jadval kabi kichik obyektlarni topish mumkinligi sababli, SQL Server ikki xil kengaytmalarga ega:

11.1-rasm. SQL Server ma'lumotlar sahifasi

- *bir xil hajmlar* bitta obyektga tegishli bo’lib, barcha sakkiz sahifadan faqat u foydalanishi mumkin;

- *aralash miqyoslarni* baham ko’rish mumkin. Sakkiz kenglik sahifalarining har biri boshqa obyektga tegishli bo’lishi mumkin, lekin faqat bitta.

Endi saqlash vaqtida ma'lumotlar bazasi jadvallarini tashkil qilishni ko’rib chiqamiz. Bu yerda arxitektura qarorlari asosan MBBT ishlab chiquvchilari tomonidan belgilanadi. SQL Server MBBTda jadvalni tashkil qilish diagrammasi 11.2-rasmda ko’rsatilgan.

11.2-rasm. SQL Serverda jadvallarni tashkil qilish

Odatiy bo’lib, jadval yoki indeksda jadval yoki indeksning barcha sahifalarini o’z ichiga olgan bitta *bo’lim* mavjud. Bo’lim bitta fayl guruhida joylashgan.

Belgilangan bo’lim funksiyasi qiymatlari asosida qatorlar guruhlari bo’lingan bo’lingan jadvallar yoki indekslarni yaratishingiz mumkin. Bo’limlar turli xil ma'lumotlar bazasi fayl guruhlarida saqlanishi mumkin. Shu bilan birga, bunday jadval so’rovlarni bajarish yoki ma'lumotlarni yangilashda mantiqiy yagona obyekt sifatida qaraladi.

Keling, 11.2-rasmda mavjud bo’lgan *"uy"* va *"muvozanatlangan daraxt"* atamalarni aniqlaylik. Bo’lim ichida SQL Server ma'lumotlarni tartibga solish uchun ikkita yondashuvdan birini qo’llashi mumkin:

- *klasterli jadvallar* - bu klasterli (ba'zan "klasterli" atamasi) indeksga ega jadvallar. Ma'lumotlar qatorlari klasterlangan indeks kaliti bo’yicha tartiblanadi va indeksning o’zi muvozanatli daraxt sifatida amalga oshiriladi. Indekslarni tashkil qilish quyida batafsilroq muhokama qilinadi. Jadvalning asosiy kaliti odatda ushbu turdagi indeks yordamida saqlanadi;
- *to’p* - bu klasterli indeksga ega bo’lmagan jadval. Bunday jadvalning satrlari o’z atributlarining qiymatlari bo’yicha

tartiblanmaydi: MBBT sahifalarni ajratadi va jadvallarga yangi yozuvlar qo’shilishi sababli ulardagi ma'lumotlarni saqlaydi.

Agar to’p yoki klasterli jadval bir nechta bo’limlarni o’z ichiga olsa, har bir bo’lim mos ravishda to’p yoki B-daraxt tuzilishiga ega.

Jadvalda turli xil ma'lumotlar bo’lishi mumkin. SQL Serverda *ajratish birligi* ma'lum bir turdagi sahifalar to’plamidir. Uch turdagi tarqatish birliklari mavjud:

- IN_ROW_DATA – sahifalar ma’lumotlar qatorlari yoki indekslarni o‘z ichiga oladi, bundan *katta obyektlar (* LOB), masalan, tasvirlar yoki XML ma’lumotlari bundan mustasno. Sahifalar "Ma'lumotlar" yoki "Indeks" turiga ega;
- LOB_DATA – sahifalar SQL Serverda matn, ntext, image, XML, varchar(max), nvarchar(maks), varbinary(maks) yoki foydalanuvchi tomonidan aniqlangan katta ma’lumotlar turlari (CLR UDT) ma’lumotlar turlarini o‘z ichiga olgan katta ob’yektiv ma’lumotlarni o‘z ichiga oladi. . Sahifalar "Matn/Rasm" turiga ega;
- ROW_OVERFLOW_DATA — varchar, nvarchar, varbinary yoki sql_variant ustun turlarida saqlanadigan o‘zgaruvchan uzunlikdagi ma’lumotlar qator o‘lchami chegarasi 8060 baytdan (sahifaga yozilgan ma’lumotlarning maksimal miqdori) oshadi. Ushbu to’plamdagi sahifalar ham "Matn/Rasm" turiga ega.

Bitta jadval bo’limida sanab o’tilgan turlarning har birining faqat bitta ajratish birligi bo’lishi mumkin.

Nazorat savollari

1. Fizik saqlash nima uchun kerak?
2. Fizik ma'lumotlar bazasi qanday tashkil etiladi?
3. Ma'lumotlar bazasida ma'lumotlar qanday ko'rsatiladi?
4. Indekslar ma'lumotlar bazasida nima uchun ishlatiladi?
5. Indekslar qanday yaratiladi va qanday saqlanadi?
6. Indekslar qanday ma'lumotlarni tezkor izlashda yordam beradi?
7. Fizik ma'lumotlar bazasini yaratishda qanday ma'lumotlar kerak?
8. Ma'lumotlar bazasida indekslar qanday o'rganiladi va tuziladi?
9. Fizik saqlash uchun moslashtirilgan platformalar va vositalar qanday ishlaydi?
10. Indekslar ma'lumotlarga qanday qidiruv va filtrlash imkoniyatini oshiradi?

12-MAVZU. DASTURLASHTIRILGAN MA’LUMOTLAR BAZASI OBYEKTLARI.

Reja:

1. Dasturlashtiriladigan ma'lumotlar bazasi obyektlari
2. O‘zgaruvchilar va vaqtinchalik jadvallar.
3. Shartlarni tekshirish va dasturni bajarilish tartibini boshqaruvchi operatorlar.

Dasturlashtiriladigan ma'lumotlar bazasi obyektlariga quyidagilar kiradi: saqlangan protseduralar, funksiyalar, triggerlar, kursorlar va ko’rinishlar hisoblanadi. Ular quyidagi maqsadlarning biriga yoki kombinatsiyasiga erishish uchun ishlatilishi mumkin:

- jadval yoki domen yaratishda birlamchi va tashqi kalit cheklovlari, yagonalik cheklovlari va CHECK iborasi yordamida ko’rsatilgan atribut qiymatlari bo’yicha cheklovlar orqali ifodalab bo’lmaydigan cheklovlarni belgilash;
- ma'lumotlar bazasi server tomonida ma'lumotlarni qayta ishlash mantig’ining (biznes-mantiq) bir qismini amalga oshirish;
- xavfsizlik nuqtai nazaridan ham ma'lumotlar bazasi tuzilmalarini foydalanuvchi ilovalariga o’zgartirishlar kiritmasdan o’zgartirish imkoniyatini ta'minlash uchun ma'qul bo’lishi mumkin bo’lgan ma'lumotlar bazasi tuzilmalarini amalga oshirish xususiyatlarini foydalanuvchilardan va amaliy dasturlardan yashirish;
- samaradorlikni oshirish; xususan, saqlangan protsedurani bajarish rejasini kompilyatsiya qilish va yaratish birinchi marta ishga tushirilganda amalga oshiriladi, bu esa keyinchalik vaqtni tejaydi.

Dasturlashtiriladigan obyektlarni yaratish uchun SQL tilining kengaytmalari o’zgaruvchilar bilan ishlash, sikllarni tashkil qilish, shartli o’tishlar va hk.

Aksariyat SQL dialektlarida mos keladigan operatorlar to’plami amalda takrorlangan bo’lsa-da, turli ma'lumotlar bazasi tizimlari turli xil o’zgaruvchilarni nomlash qoidalarini qabul qilgan va turli xil tizim obyektlaridan foydalangan. Ushbu qo’llanmada, agar boshqacha ko’rsatilmagan bo’lsa, tavsif Microsoft SQL Serverda qo’llaniladigan Transact-SQL (qisqacha T-SQL) dialektiga asoslanadi.

O’zgaruvchilar va vaqtinchalik jadvallar

SQL Server ikki turdagi o’zgaruvchilardan foydalanishga imkon beradi - *lokal* va *global*. Global o’zgaruvchilar faqat qiymatni o’qishga imkon beradi. Lokal o’zgaruvchilar yaratilishi, qiymatlarni belgilashi va o’qilishi mumkin. Taqdim etilgan global o’zgaruvchilar to’plami juda keng, 12.1-jadval eng ko’p ishlatiladiganlar ro’yxatini ko’rsatadi.

12.1-jadval

Tez-tez ishlatiladigan global o’zgaruvchilar

O’zgaruvchan nomi	Qaytish qiymati
@@XATO	Oxirgi bajarilgan ko’rsatmaning xato kodi
@@IDENTITY	Joriy ulanishga kiritilgan oxirgi aniqlovchi qiymat (avtomatik ravishda yaratilgan hisoblagich ustuni qiymati).
@@LANGUAGE	Joriy seans uchun til o‘rnatilgan
@@>KO W COUNT	Oxirgi ko’rsatma bo’yicha qayta ishlangan qatorlar soni
@@SERVERNAME	Server nomi. Standart misol uchun qiymat "server nomi" shaklida qaytariladi, nomlangan misollar uchun - <server_name/instance_name"
@@TRANCOUNT	Ochiq tranzaktsiyalar soni (joriy aloqada)
@@VERS10N	SQL Server versiyasi

Quyida SQL Server versiyasini va joriy seans uchun belgilangan tilni olish imkonini beruvchi so’rov matni keltirilgan. So’rov natijalarida ustunlar mos ravishda "Versiya" va "Lang" deb nomlanadi:

```
SELECT @@VERSION AS [Version],
SELECT @@LANGUAGE AS [Lang]
```

T-SQL da lokal o’zgaruvchilarni e'lon qilish uchun DECLARE operatoridan foydalaniladi, uning formati quyida ko’rsatilgan:

```
DECLARE
{ [@local_var [AS] data_type] | [ = value ] }
| { @cursor_var cursor }
| [,...n]
| {@table_var [AS] <table_type>}
```

Muntazam o’zgaruvchini belgilashda uning nomi, turi ko’rsatilishi kerak va ixtiyoriy ravishda qiymat ko’rsatilishi mumkin. O’zgaruvchi nomi va ma'lumotlar turi orasiga "AS" kalit so’zini qo’yish shart emas. O’zgaruvchi matn, matn, rasm bundan mustasno, SQL

Server tomonidan qo’llab-quvvatlanadigan har qanday o’rnatilgan yoki foydalanuvchi tomonidan belgilangan ma'lumotlar turi bo’lishi mumkin. Quyida @i butun o’zgaruvchisini e'lon qilish va uni 1 qiymati bilan ishga tushirish misoli keltirilgan:

```
DECLARE @i int =1
```

Bitta DECLARE bayonotida bir nechta o’zgaruvchilarni e'lon qilishingiz mumkin, ularning ta'riflarini vergul bilan ajrating:

```
DECLARE
@Group nvarchar(50) , @Sale_money
```

DECLARE dan jadval o’zgaruvchisini yaratish uchun ham foydalanish mumkin. Bunday holda, uning nomidan keyin (va ixtiyoriy kalit so’z "AS") "TABLE" so’zi va jadval tavsifi bo’lishi kerak. Jadval o’zgaruvchisi alohida DECLARE bayonotida yaratilishi kerak:

```
DECLARE @MUTAY_TABLE (
ID int NO NULL,
AUTHOR varchar(30),
TITLE varchar(50))
```

SET yoki SELECT iboralari yordamida lokal o’zgaruvchiga qiymat belgilashingiz mumkin. Eng tez-tez ishlatiladigan SET bayonotining soddalashtirilgan formati quyida ko’rsatilgan:

```
SET @local_var = ifoda
```

Bu yerda @local_var - lokal o’zgaruvchining nomi; ifoda - belgilangan qiymatni tavsiflovchi ifoda. Masalan, quyidagi kod parchasi @MyVar o’zgaruvchisini e'lon qiladi va unga "Test matni" qiymatini belgilaydi:

```
DECLARE @MyVar nchar(20);
SET @MyVar = 'Test matni';
```

SQL Serverda 2008-versiyasidan boshlab oddiy tayinlash operatori bilan bir qatorda murakkab belgilash operatoridan ham foydalanish mumkin. 12.2-jadvalda uni yozish variantlarini ko’rsatadi.

12.2- *jadval*

Murakkab tayinlash operatori

Yozib olish	Natija	Yozib olish	Natija
+=	Qo’shish va tayinlash	%=	Bo’limning qolgan qismini oling va tayinlang

-=	Ayirish va tayinlash	&=	Bit bo'yicha VA (VA, birikma) bajaring va tayinlang
*=	Ko'paytiring va tayinlang	**A**	Bit bo'yicha eksklyuziv OR (XOR, eksklyuziv ajratish) ni bajaring va tayinlang
/=	Ajratish va tayinlash	h	Bit bo'yicha OR (ajralish) ni bajaring va tayinlang

Agar biz muntazam o'zgaruvchi haqida gapiradigan bo'lsak, unga faqat skalyar qiymat berilishi mumkinligini hisobga olish kerak. Masalan, quyidagi kod parchasi xatoga olib keladi:

```
DECLARE @st varchar(50);
SET @st = (
SELECT Author, Title FROM Book)
```

SELECT iborasidan foydalanib, siz o'zgaruvchining qiymatini o'qishingiz va uni belgilashingiz mumkin.

Quyidagi misolda birinchi SELECT iborasi e'lon qilingan o'zgaruvchilarga qiymatlarni belgilash uchun ishlatiladi, ikkinchisi esa ushbu qiymatlarni chop etish uchun ishlatiladi:

```
DECLARE @MyVarl nchar(20);
DECLARE @i int =1;
SELECT @gyVar1 = 'Test matni', @i+=1;
SELECT gMyVar1 AS [Matn], @i AS [number]
```

Shuni ta'kidlash kerakki, SQL Server bir xil SELECT bayonotida qiymatlarni tayinlash va olish imkonini bermaydi. Shuning uchun quyidagi kod parchasini ishga tushirish xato xabariga olib keladi:

```
SELECT gMyVar1, gi+=1
```

Chalkashmaslik uchun qiymatlarni SET iborasi yordamida belgilash tavsiya etiladi. Lokal o'zgaruvchilar doirasi haqida gapirganda, ish o'rinlari to'plami tushunchasini kiritish kerak. *SQL Server uchun ish* to'plami yoki oddiygina to'plam mijoz ilovasi tomonidan ma'lumotlar bazasi serveriga yuboriladigan buyruqlar to'plami bo'lib, ular bitta blok sifatida tahlil qilinadi va bajariladi. SQL Server Management Studio'da ishlaganda paket chegaralarini belgilash uchun oldingi iboradan foydalaniladi. O'zgaruvchilar ular aniqlangan paket uchun lokaldir. Boshqa paketda aniqlangan o'zgaruvchidan foydalanishga urinish xatolikka olib keladi.

Endi jadval o’zgaruvchilari bilan ishlash misolini ko’rib chiqamiz. Aytaylik, siz jadval o’zgaruvchisini e'lon qilishingiz, unga so’rov natijalarini olishingiz va keyin olingan qiymatlarni ko’rsatishingiz kerak. T-SQL kodi quyidagicha ko’rinishi mumkin:

```
DECLARE TABLE gMyTab (
Id int PRIMARY KEY,
Author varchar(30),
Title varchar(50));
gMyTab (Id, Author, Title) INSERT INTO
SELECT DISTINCT [Id], [Author], [Title]
FROM dbo.Book1
WHERE [year]>2000;
SElECT * FROM SMyTab
```

Adabiyotlarda qayd etilishicha, unumdorlik nuqtai nazaridan kichik hajmdagi ma’lumotlar (bir necha qatorlar) bilan ishlashda jadval o’zgaruvchilari eng yaxshi qo’llaniladi, agar ma’lumotlar ko’p bo’lsa, vaqtinchalik jadvallaridan foydalanish tavsiya etiladi.

SQL Server lokal va global vaqtinchalik jadvallarni yaratish imkonini beradi. Vaqtinchalik jadvallarning barcha turlari tempdb vaqtinchalik obyektlari uchun ma'lumotlar bazasida yaratilgan.

Lokal vaqtinchalik jadval faqat uni yaratgan seansga, uning yaratilish darajasida va qo’ng’iroqlar stekining barcha ichki darajalarida (ichki protseduralar, funksiyalar va boshqalar) ko’rinadi. *Lokal vaqtinchalik jadvalning belgisi* - bu nomdagi "#" prefiksi hisoblanadi. SQL Server vaqtinchalik jadvalni qo’ng’iroqlar to’plamida yaratish darajasi chegaradan chiqib ketganda avtomatik ravishda yo’q qiladi.

Global vaqtinchalik jadval faqat uni yaratgan sessiyaga emas, balki barcha seanslarga ko’rinadi. Bunday jadvallar, u yaratilgan seans uzilganda va jadvalga faol havolalar mavjud bo’lmaganda yo’q qilinadi. Global vaqtinchalik jadval nomi "##" bilan boshlanadi.

Shartlarni tekshirish va dasturni bajarish tartibini nazorat qilish uchun operatorlar

T-SQL tili dasturni bajarish tartibini nazorat qilish va shartlarni tekshirish imkonini beruvchi bir qancha operatorlarni taklif etadi. Ular asosan dasturlashtiriladigan ma'lumotlar bazasi obyektlarida qo’llaniladi. 12.3-jadvalda ushbu operatorlarni sanab o’tadi va ularning qisqacha tavsiflarini beradi.

12.3-jadval

Sinov shartlari va dasturni bajarish tartibini nazorat qilish uchun T-SQL bayonotlari

Operator	Tavsif
BEGIN... END	Operator qavslari, ichida - bir nechta SQL ifodalari: BEGIN {sql_statementIstatement_block} END
GOTO	Belgiga shartsiz o'tish. Yorliq yorliq sifatida tavsiflanadi:
IF... ELSE	Shartli operator
RETURN	Protseduradan, to'plamdan yoki bayonotlar blokidan shartsiz chiqish. RETURN dan keyingi ko'rsatmalar bajarilmaydi.
WHILE	Takrorlash operatori
...BREAK	WHILE operatoridagi eng ichki sikldan yoki WHILE siklidagi IF...ELSE operatoridan chiqadi. END kalit so'zidan so'ng, siklning oxirini ko'rsatadigan har qanday ko'rsatma darhol bajariladi.
... CONTINUE	WHILE siklini qayta ishga tushiradi (siklning yangi iteratsiyasini boshlaydi). Siklning joriy iteratsiyasida CONTINUE kalit so'zidan keyin hech qanday bayonot bajarilmaydi.
CASE	Shartlar ro'yxatini baholaydi va bir nechta mumkin bo'lgan natija ifodalaridan birini qaytaradi.

Keling, tavsiflangan operatorlardan foydalanishning bir qator misollarini ko'rib chiqaylik. Shartli IF iborasidan boshlaylik. Yangi vaqtinchalik jadval yaratishdan oldin, siz tez-tez xuddi shu nomdagi vaqtinchalik jadval mavjudligini tekshirishingiz kerak. Agar bunday jadval allaqachon mavjud bo'lsa, u o'chiriladi.

Quyidagi misol GOTO operatoridan foydalanishni ko'rsatadi. Kodni bajarish natijasida faqat 5 raqami ko'rsatiladi.Umumiy tavsiya: iloji bo'lsa, shartsiz sakrashlardan foydalanmang, chunki bu kodni o'qishni qiyinlashtiradi:

```
DECLARE @i int =3;
IF @i > 2 GOTO L1;
SET @i+=1;
```

```
PRINT @i;
L1:
SET @i+=2;
PRINT @i;
```

12.5-jadvalda sodir bo’lgan xato haqida ma'lumot olish uchun CATCH blokida ishlatilishi mumkin bo’lgan SQL Server tomonidan taqdim etilgan funksiyalar ro’yxati keltirilgan. @@ERROR global o’zgaruvchisidan ham foydalanish mumkin (12.5-jadvalga qarang).

12.5-jadval

Xatolarni bartaraf etishda foydalaniladigan funksiyalar

Funksiya	Ma'nosi
ERROR_LINE()	Xatolik yuz bergan qator raqami
ERROR_MESSAGE()	CATCH blokirovkasini ishga tushirgan xato xabarining to’liq matni
ERROR_NUMBER()	Xato raqami
ERROR_PROCEDURE()	Xato sodir bo’lgan saqlangan protsedura nomi
ERROR_SEVERITY()	Xatoning jiddiylik darajasini tavsiflovchi butun son.
ERRORSTATEQ	Butun son - xato holati kodi. Xuddi shu raqam bilan xatolik turli vaziyatlarda yuzaga kelishi mumkin, ularga noyob holat kodlari beriladi

Nazorat savollari

1. Dasturlashtiriladigan ma'lumotlar bazasi obyektlari nima?
2. O‘zgaruvchilar ma'lumotlar bazasida qanday ishlatiladi?
3. Vaqtinchalik jadvallar ma'lumotlar bazasida qanday yaratiladi?
4. Shartlarni tekshirish operatorlari ma'lumotlar bazasida qanday ishlatiladi?
5. Dasturni bajarilish tartibini boshqaruvchi operatorlar nima?
6. Dasturlashtiriladigan ma'lumotlar bazasi obyektlari qanday yaratiladi va boshqariladi?
7. O‘zgaruvchilar ma'lumotlar bazasida qanday tanzimlanadi?
8. Vaqtinchalik jadvallar ma'lumotlar bazasida qanday ishlatiladi va qanday yaratiladi?

9. Shartlarni tekshirish operatorlari qanday ishlaydi va qanday foydalaniladi?

10. Dasturni bajarilish tartibini boshqaruvchi operatorlar qanday ishlaydi va qanday maqsadga xizmat qiladi?

11. Dasturlashtiriladigan ma'lumotlar bazasi obyektlari va o‘zgaruvchilar orasidagi farq qanday?

12. Vaqtinchalik jadvallar va shartlarni tekshirish operatorlari orasidagi farq qanday?

13-MAVZU. SAQLANGAN PROTSEDURALAR.

Reja:

1. Saqlangan protseduralar
2. Funksiyalar.
3. Trigerlar.
4. Ko‘rinishlar: T-SQL da kengaytirilgan sintaksisi

Saqlangan protsedura - bu nomli ma'lumotlar bazasi dasturiy obyektidir. SQL Serverda bir necha turdagi saqlangan protseduralar mavjud.

Tizimda saqlanadigan protseduralar MBBT ishlab chiquvchilari tomonidan taqdim etiladi va tizim katalogi bilan harakatlarni bajarish yoki tizim ma'lumotlarini olish uchun ishlatiladi. Ularning nomlari odatda "sp_" prefiksi bilan boshlanadi. Siz barcha turdagi saqlangan protseduralarni EXECUTE buyrug’i yordamida bajarasiz, uni EXEC ga qisqartirish mumkin. Masalan, parametrlarsiz ishga tushirilgan sp_helplogins saqlanadigan protsedura hisob raqamlari *(inglizcha* loginlar) va har bir ma'lumotlar bazasidagi tegishli foydalanuvchilar *(ingliz* users) haqida ikkita hisobot hosil qiladi.

```
EXEC sp_helplogins;
```

Tizimda saqlanadigan protseduralar yordamida bajariladigan harakatlar haqida tushuncha berish haqida 13.1-jadval ba'zi misollarni ko’rsatadi. Umuman olganda, SQL Serverda mingdan ortiq tizim saqlanadigan protseduralar mavjud.

13.1-jadval

SQL Server tizimida saqlangan protsedura misollari

Nomi	Vazifasi
Sp_changedbowner	Ma'lumotlar bazasi egasini o’zgartirishga imkon beradi
Sp_database	Ma'lumotlar bazalari ro’yxatini va ularning hajmini qaytaradi
Sp_help	Kirish parametri sifatida obyekt nomini oladi va u haqidagi ma'lumotlarni qaytaradi. Masalan: exec sp_help @objname= 'dbo.Book1'
Sp_storedprocedures	Joriy ma'lumotlar bazasida saqlangan protseduralar ro’yxatini qaytaradi

sp_tables	Joriy ma'lumotlar bazasidagi jadvallar va ko’rinishlar ro’yxatini qaytaradi. talab qilinishi mumkin

Foydalanuvchi ma'lumotlar bazalarida va vaqtinchalik obyektlar uchun ma'lumotlar bazasida saqlangan protseduralarni yaratishi mumkin. Ikkinchi holda, saqlangan protsedura *vaqtinchalik* bo’ladi. Vaqtinchalik jadvallarda bo’lgani kabi, vaqtinchalik saqlanadigan protsedura nomi, agar lokal vaqtinchalik saqlanadigan protsedura bo’lsa, "#" prefiksi bilan yoki global bo’lsa, "##" bilan boshlanishi kerak. Lokal vaqtinchalik protsedura faqat u yaratilgan ulanish doirasida ishlatilishi mumkin, global esa boshqa ulanishlar ichida ham ishlatilishi mumkin.

Dasturlashtiriladigan SQL Server obyektlari Transact-SQL vositalari yordamida ham, *Microsoft.Net Framework ning CRL* (Common Language Runtime) muhitida yig’ilishlar yordamida ham yaratilishi mumkin. Ushbu qo’llanma faqat birinchi usulni qamrab oladi.

Saqlangan protseduralarni yaratish uchun CREATE PROCEDURE (PROC ga qisqartirilishi mumkin) iborasidan foydalaning, uning formati quyida keltirilgan:

```
CREATE {PROC I PROCEDURE) [schema_name.]proc_name [ I
NUMBER]
[
   {gparameter [type_schema_name.] data_type }
   [VARYING] ["standart"] [OUT|OUTPUT]|[READONLY]
]
[...n]
[ WITH <protsedura_opsiyasi> [ ...n ] ]
AS {
   [BEGIN] sql_statement [I] [ ...n ] [END] }
[I]
WHERE
<protsedura_opsiyasi>::=[ENCRYPTION][RECOMPILE][BAJARI
ASLA ISHLATISH]
```

Agar saqlangan protsedura (yoki trigger, funksiya, ko’rinish) SIFIRLASH opsiyasi bilan yaratilgan bo’lsa, uning kodi matnni o’qib bo’lmaydigan qilib o’zgartiriladi. Shu bilan birga, foydalanilgan

algoritm SQL Serverning oldingi versiyalaridan ko’chirilgan va uni ishonchli himoya algoritmi deb hisoblash mumkin emas - teskari konvertatsiyani tezda amalga oshirish imkonini beruvchi yordamchi dasturlar mavjud.

RECOMPILE parametri har safar protsedura chaqirilganda tizim matnni qayta kompilyatsiya qilishini bildiradi. Oddiy holatda, birinchi ishga tushirishda tuzilgan protsedura keshda saqlanadi, bu esa unumdorlikni oshirish imkonini beradi.

Parametrlar protsedura uchun belgilanishi mumkin. Bu qiymatlarni protseduraga o’tkazish uchun ishlatiladigan lokal o’zgaruvchilar hisoblanadi. Agar parametr OUTPUT (yoki qisqacha OUT) kalit so’zi bilan e'lon qilingan bo’lsa, u chiqish qiymati hisoblanadi: protsedura tugagandan so’ng unga berilgan qiymat protsedurani chaqirgan dastur tomonidan ishlatilishi mumkin. READONLY kalit so’zi saqlangan protsedura ichida parametr qiymatini o’zgartirib bo’lmasligini bildiradi.

Parametrlarga standart qiymatlar berilishi mumkin, agar protsedurani chaqirishda parametr qiymati aniq ko’rsatilmagan bo’lsa, ular qo’llaniladi. Keling, bir misolni ko’rib chiqaylik:

```
PROC CREATE Sum(@a int, @b int=0,
@result int OUTPUT) AS
SET @result =@a+@b
```

Biz uchta parametrli protsedura yaratdik va @b parametrining standart @result=0 ga teng va @result parametri chiqish parametridir: u qiymatni chaqiruvchi dasturga qaytaradi. Amalga oshirilgan harakatlar juda oddiy - chiqish parametri ikkita kirish summasining qiymatini oladi.

Protsedurani chaqirishda siz kiritish parametrlari sifatida ham o’zgaruvchilardan, ham doimiylardan foydalanishingiz mumkin. Keling, ikkita misolni ko’rib chiqaylik. Birinchisida protseduraning kirish parametrlari aniq konstantalar sifatida ko’rsatilgan va chaqiruvdagi chiqish parametri uchun OUTPUT kalit so’zi ko’rsatilgan. Ikkinchi parametr o‘zgaruvchining qiymatini birinchi kiritish parametri sifatida ishlatadi va standart qiymatdan DEFAULT kalit so‘zidan foydalangan holda ikkinchi parametr uchun foydalanish kerakligini bildiradi:

13.1-rasm. Saqlangan protsedura yaratildi

var.1

```
DECLARATION @s int;
EXEC sum 10,5, @s OUTPUT;
PRINT @s;
```

Ba'zi hollarda buyruqni dinamik ravishda yaratish va uni ma'lumotlar bazasi serverida bajarish kerak bo'lishi mumkin. Bu muammoni EXEC operatori yordamida ham hal qilish mumkin. Quyidagi misol, year atributi o'zgaruvchi tomonidan belgilangan qiymatga teng bo'lsa, Book1 jadvalidan yozuvlarni oladi:

```
DECLARE @y int = 2000;
EXEC ('SELECT * FROM dbo.Book1 WHERE [Yil] ='+ @y) ;
```

Dinamik ravishda yaratilgan ko'rsatmalarning bajarilishi "SQL injection" kabi kompyuter hujumlarini amalga oshirish uchun zarur shart-sharoitlarni yaratadi. Misolning mohiyati shundaki, foydalanuvchi o'zining SQL kodini dinamik ravishda yaratilgan so'rovga kiritadi. Bu odatda almashtirilayotgan parametrlar foydalanuvchi kiritish natijalaridan olinganda sodir bo'ladi.

Oldingi misolni biroz o'zgartiramiz:

```
DECLARE @y varchar(100);
SET @y='2OOO';
EXEC ('SELECT * FROM dbo.Book2 WHERE [year]='+ @y);
```

Agar biz SET bayonotida berilgan satr qiymatini foydalanuvchidan oldik deb hisoblasak (qanday bo'lishidan qat'iy nazar, masalan, veb-ilova orqali), unda misol bizning kodimizning "oddiy" harakatini ko'rsatadi.

Quyidagi misolda foydalanuvchi kiritilgan parametrga SQL kodini qo'shdi. Natijada, biz nafaqat jadvaldan kerakli qatorlarni olib tashladik, balki uning barcha mazmunini ham o'chirib tashladik:

```
DECLARE @y varchar(100);
SET @y='2000';
FROM dbo.Book2 DELETE;
EXEC ('SELECT * FROM dbo.Book2 WHERE [year]='+ @y);
```

Funksiyalar

SQL Server ko’plab o’rnatilgan funksiyalarni ta'minlaydi va shuningdek, foydalanuvchi tomonidan belgilangan funksiyalarni yaratishni qo’llab-quvvatlaydi. *Foydalanuvchi funksiyasi* - bu oldindan belgilangan turdagi qiymatni qaytaradigan dasturlashtiriladigan ma'lumotlar bazasi obyektidir. SQL Server skalyar funksiyalarni (skaler qiymat qaytaradiganlar) va jadval funksiyalarini qo’llab-quvvatlaydi.

Skayar funksiyadan skaler ifoda o‘rniga istalgan joyda, jumladan hisoblangan ustunlar va CHECK cheklash ta’riflarida ham foydalanish mumkin. Jadval qiymatini qaytaruvchi funksiyani jadval ifodalariga ruxsat berilgan har qanday joyda, birinchi navbatda SELECT, INSERT, UPDATE va DELETE iboralarida chaqirish mumkin.

Funksiyalar quyidagilarni bajara olmaydi:

- server nusxasi yoki ma'lumotlar bazasi holatini o’zgartirish;
- jadvaldagi ma'lumotlarni o’zgartirish;
- vaqtinchalik jadvallarni yaratish va ularga kirish;
- dastur kodini dinamik ravishda bajarish;
- 0 dan 1 gacha bo’lgan psevdo-tasodifiy qiymatni qaytaruvchi xatolik keltirib chiqaruvchi RAISERROR funksiyasi yoki RAND funksiyasi kabi tashqi effektli funksiyani chaqirish. Ikkinchi holda, “tashqi effekt” degani RANDni ishga tushirish qiymati bilan chaqirish ushbu funksiya joriy seansda ishlab chiqaradigan butun ketma-ketlikka ta'sir qiladi.

Skalyar foydalanuvchi tomonidan belgilangan funksiyani yaratish uchun CREATE FUNCTION iborasining formatini ko’rib chiqing:

```
CREATE FUNCTION [sxema_name.]function_name(
   [
      {
         @parameter_name [AS] [type_schema.]data_type
         [=default] [READONLY]
      }
         [,...n]
   ])
RETURN_DATA_TYPE
```

```
[WITH <function_option> [, ...n]]
[AS]
BEGIN
funktion_body
skalyar_ifoda RETURN
END
(;)
```

E'tibor bering, agar funksiya hech qanday parametrga ega bo'lmasa ham, ta'rifda funksiya nomidan keyin qavslar bo'lishi kerak. Funksiya qiymatini qaytaruvchi RETURN iborasi ham bo'lishi kerak.

SCHEMABINDING opsiyasi bog'langan obyektlarni o'chirib bo'lmasligini ta'minlaydi. Misol uchun, agar ushbu parametrga ega funksiya jadvaldan foydalansa, funksiya mavjud bo'lganda uni o'chirib bo'lmaydi.

CALLED ON NULL INPUT ning standart qiymati, agar funksiya parametri NULL bo'lsa, funksiya chaqiriladi va uning kodi bajariladi. Agar RETURNS NULL ON NULL INPUT opsiyasi o'rnatilgan bo'lsa, u holda NULL qiymatiga ega bo'lgan kirish parametri olinganda, funksiya kodi bajarilmaydi va NULL qiymati darhol funksiyani chaqirgan modulga qaytariladi.

Skalyar qiymatli funksiya yaratish misolini ko'rib chiqamiz. Saqlangan protseduralar bo'yicha birinchi misolda bo'lgani kabi, funksiya ikkita kirish argumentining yig'indisini hisoblab chiqsin. Shuni e'tiboringizga qaratmoqchimanki, chop etish operatorida funksiyani chaqirishda siz funksiya nomini u tegishli bo'lgan sxema nomi bilan ko'rsatishingiz kerak. Aks holda, SQL Server ko'rsatilgan nom bilan o'rnatilgan funksiyani topishga harakat qilishi sababli xatolik yuzaga keladi.

Keling, natural sonning faktorialini hisoblaydigan funksiya yaratamiz. Agar argumentning qiymati aniqlanmagan bo'lsa, unda keyingi hisoblashning ma'nosi yo'q: funksiya NULLni qaytaradi va o'z ishini tugatadi - bu RETURNS NULL ON NULL INPUT varianti bilan ko'rsatiladi. Agar argument qiymati manfiy yoki nolga teng bo'lsa, IF iborasidagi tekshirish orqali aniqlanganidek, xatti-harakatlar o'xshash bo'ladi:

```
CREATE FUNCTION dbo.Factorial(@n int)
RETURNS int
RETURNS NULL ON NULL INPUT
```

```
BEGIN
IF (@n<=0) RETURN NULL;
DECLARE @Nf int =1, @i int =1;
@n>@i BEGIN
SET @i+=l;
SET @Nf*=@i;
END
RETURN @Nf
END
```

Funksiyalar, triggerlar, saqlangan protseduralar kabi dasturlashtiriladigan obyektlarni qo’llab-quvvatlaydi. SQL Server maksimal 32 ta joylashtirish darajasiga ruxsat beradi. Misol tariqasida, quyida rekursiya yordamida faktorial hisoblash funksiyasi keltirilgan. Bundan tashqari, kirish parametrining qiymatlari oralig’i ruxsat etilgan joylashtirish darajasidan oshmasligi uchun boshqariladi. Ammo shuni ta'kidlash kerakki, int tipidagi qiymatlar diapazoni hatto 32 ni ham hisoblashga imkon bermaydi!

```
CREATE FUNCTION dbo.Factorial2(@n int)
RETURNS int
RETURNS ON NULL INPUT BEGIN
IF ((@n<=0) OR (@n>32)) RETURN NULL;
DECLARE @Nf int =1;
IF (@n>1)
SET @NF=dbo.Faktorial2(@n-l)* @n;
RETURN @Nf
END
```

SQL Serverda SELECT operatorining WHERE bandida foydalanuvchi tomonidan belgilangan funksiyalardan foydalanish tavsiya etilmaydi, chunki bu holda funksiya so’rov bilan qayta ishlanadigan har bir satr uchun bajariladi. Bu so’rovni bajarish tezligiga salbiy ta'sir ko’rsatishi mumkin.

Triggerlar

Trigger - bu ma'lum bir hodisa sodir bo’lganda avtomatik ravishda bajariladigan saqlanadigan protseduraning maxsus turidir. Triggerlar aniq kirish yoki chiqish parametrlarini o’z ichiga olmaydi va ularni aniq bajarib bo’lmaydi. SQL Server triggerlarning uchta sinfini qo’llab-quvvatlaydi: Ma'lumotlarni boshqarish tili triggerlari (DML

triggerlari), Ma'lumotlarni tavsiflash tili triggerlari (DDL triggerlari) va kirish triggerlari.

INSERT, UPDATE yoki DELETE iboralari yordamida jadval yoki viewdagi ma’lumotlarni o‘zgartirishga urinilganda DML triggerlari ishga tushiriladi. Triggerni yaratuvchi buyruqning formati quyidagicha:

```
CREATE TRIGGER [schema_name.] trigger_name
ON [stol | view]
[ON <option> [ ,...n ] ]
[FOR |AFTER |PLACE]
{ [INSERT] [,] [UPDATE] [,] [DELETE] }
[NOT FOR REPLICATION]
AS {sql_statement[; ] [ ,...n ]}
<option>::= {[cipher]! [use]}
```

Bu yerda quyidagi tushuntirishlar zarur. AFTER trigger turi faqat triggerni yoqish haqidagi bayonot muvaffaqiyatli yakunlangandan keyingina ishga tushishini bildiradi. Bu birinchi navbatda barcha cheklash tekshiruvlarini o’tkazadi va agar ular o’tib ketsa, tetik ishga tushadi.

Birlamchi kalit qiymatining takrorlanishi tufayli jadvalga yozuv qo’shishning iloji bo’lmasa, ushbu operatsiya bilan bog’langan AFTER tipidagi trigger ishga tushmaydi.

Agar faqat FOR kalit so’zi ko’rsatilgan bo’lsa, bu AFTER triggerining yaratilishini bildiradi, ya'ni. FOR va AFTER bu holda sinonimdir. Shuni hisobga olish kerakki, INSERT, UPDATE, DELETE operatorlari yashirin tranzaksiya doirasida bajariladi. Trigger xuddi shu tranzaksiya doirasida ishga tushiriladi va butun tranzaksiya undan orqaga qaytarilishi mumkin. AFTER triggerlarini ko’rinishlarda aniqlab bo’lmaydi.

Turdagi trigger o’rniga triggerni chaqirgan operatorning harakatlari bajarilmaydi, aksincha trigger kodi bajariladi. Jadval yoki ko’rinishda har bir INSERT, UPDATE yoki DELETE iborasi uchun belgilangan ko’pi bilan bitta OF PLACE trigger bo’lishi mumkin. Shu bilan birga, jadval uchun bir nechta ko’rinish yaratishingiz mumkin, ularning har biri o’zining O’RNATISH triggeriga ega bo’lishi mumkin. WITH CHECK OPTION dan foydalanadigan yangilanishdan xabardor ko’rinishlarda triggerlar o’rniga foydalanishga ruxsat berilmaydi.

"[INSERT] [,] [UPDATE] [,] [DELETE]" ro‘yxati bajarilganda yaratilayotgan triggerni ishga tushiradigan bayonot yoki bayonotlarni belgilaydi.

Ish paytida DML triggerlari kiritilgan va o’chirilgan maxsus jadvallarga kirish huquqiga ega bo’lib, ularda trigger chaqirgan bayonot tomonidan kiritilgan o’zgarishlar natijalari saqlanadi. 13.2-jadvalda trigger aniqlangan bayonotga qarab, ushbu jadvallarda aniq nima saqlanganligi ko’rsatilgan.

13.2-jadval

DML triggerlari uchun mavjud bo’lgan maxsus jadvallar tarkibi

Operator	Jadval	Jadval tarkibi
INSERT	kiritilgan	Jadvalga yangi qatorlar kiritildi
DELETE	o’chirildi	O’chirilgan qatorlar
UPDATE	kiritilgan	O’zgartirilgan qatorlarning yangi holati

Triggerlar yordamida siz murakkab mantiqiy shartlarni tekshirishingiz mumkin, ularning bajarilishi ma'lumotlarning izchilligini ta'minlaydi. Misol uchun, jadvalga yozuvni yaratishda siz boshqa ma'lumotlar bazasida joylashgan boshqa jadvalda tegishli ma'lumotlar mavjudligini tekshirishingiz kerak. Xorijiy kalit cheklovi bu yerda yordam bermaydi, chunki ma'lumotlar bazalari har xil, ammo bunday tekshirish trigger yordamida amalga oshirilishi mumkin.

Quyidagi misolni ko’rib chiqing. MyTest ma'lumotlar bazasida Book1 jadvali mavjud bo’lib, u allaqachon misollarda ishlatilgan. MyTest2 ma'lumotlar bazasida quyidagi skript yordamida yaratilgan qora ro’yxat jadvali mavjud:

```
USE [MyTest2]
BEGIN
CREATE TABLE [dbo].[Black list](
[Id] [int] PRIMARY KEY,
[Author] [varchar] (50) NULL EMAS,
[Title] [varchar] (50) NULL EMAS)
```

Kursorlar

Kursor - bu SELECT operatori tomonidan qaytarilgan natijalar to’plamidagi qatorlarni alohida qayta ishlash imkonini beruvchi obyektdir. Keyinchalik Transact-SQL tilida qo’llab-quvvatlanadigan kursorlarni ko’rib chiqamiz. Bu ma'lumotlar bazasi server tomonida obyektlar sifatida mavjud bo’lgan server tomoni kursorlari. Bundan

tashqari, mijoz ma'lumotlar bazasi ilovalarini yaratish uchun ishlatiladigan mijoz kursorlari mavjud.

Adabiyotlarda qayd etilishicha, kursor yordamida ma’lumotlar to‘plamini satr bo‘yicha qayta ishlash aksariyat hollarda bir nechta satrlarni qayta ishlash uchun SQL vositalari tomonidan bajariladigan shunga o‘xshash harakatlarga qaraganda ancha sekinroq bo‘ladi. Shuning uchun kursorlardan faqat qatorlar to’plami bilan operatsiyalar orqali kerakli harakatlarni tavsiflash samarasiz yoki hatto imkonsiz bo’lgan hollarda foydalanish tavsiya etiladi.

Kursor bilan ishlash odatda quyidagi bosqichlarni o’z ichiga oladi:

- kursor deklaratsiyasi;
- kursorni ochish;
- atribut qiymatlarini kursorning birinchi yozuvidan o’zgaruvchilarga o’qish;
- kursor atrofida harakatlanish (odatda siklda) va kursor yozuvlarini qayta ishlash;
- kursorni yopish;
- kursorga ajratilgan xotirani bo’shatish.

Kursor DECLARE operatori yordamida e'lon qilinadi, uning formati quyida ko’rsatilgan. Shuni ta'kidlash kerakki, SQL Serverda ushbu operator ISO SQL standartining sintaksisini (standart versiyasi hujjatlarda ko’rsatilmagan) va Transact-SQL til kengaytmalari to’plamidan foydalangan holda sintaksisni qo’llab-quvvatlaydi.

GLOBAL kalit so’zini ko’rsatish, e'lon qilingan kursor serverga joriy ulanish doirasida ishlaydigan har qanday ish to’plami, trigger yoki saqlangan protsedurada mavjudligini anglatadi. Kursor faqat ulanish buzilgan taqdirdagina qo’yib yuboriladi.

"LOKAL" kursor, sukut bo’yicha yaratilgan yoki LOCALni aniq belgilash orqali yaratilgan bo’ladimi, faqat u yaratilgan ish to’plami, saqlangan protsedura yoki triggerda mavjud. To’plam, saqlangan protsedura yoki trigger bajarilishini tugatgandan so’ng, bu kursor bilvosita chiqariladi. Kursor saqlanadigan protseduraning OUTPUT parametri orqali o’tkazilganda istisno hisoblanadi. Keyin kursor unga havola qilingan barcha o’zgaruvchilar bo’shatilganda yoki u doiradan chiqib ketganda bo’shatiladi.

FORWARD_ONLY siz kursor bo’ylab faqat oldinga "harakat qilishingiz" mumkin degan ma'noni anglatadi (faqat FETCH NEXT buyrug’i mavjud). Kursordagi har bir yozuvni ko’pi bilan bir marta

qayta ishlash mumkin. Agar FORWARD_ONLY STATIC, KEYSET yoki DYNAMIC kalit so’zlarisiz ko’rsatilgan bo’lsa, u holda kursor DYNAMIC kursor sifatida ishlaydi. Agar FORWARD_ONLY yoki SCROLL belgilanmagan bo’lsa va STATIC, KEYSET yoki DYNAMIC kalit so’zlardan hech biri belgilanmagan bo’lsa, sukut bo’yicha FORWARD_ONLY hisoblanadi.

SCROLL kursor atrofida istalgan yo’nalishda “harakatlanishingiz” mumkinligini bildiradi (FIRST, LAST, PRIOR, NEXT, RELATIVE, ABSOLUTE FETCH operatorida mavjud). SCROLL parametrini FAST_FORWARD opsiyasi bilan belgilab bo’lmaydi. STATIC, KEYSET va DYNAMIC kursorlarida SCROLL standart qiymati mavjud.

STATIC kursor yangilanmasligini bildiradi. Bunday kursorning olingan ma'lumotlar to’plami ma'lumotlar bazasidan olinadi va tempdb vaqtinchalik obyektlari uchun ma'lumotlar bazasida saqlanadi. Kursor uchun asos bo’lib xizmat qiladigan jadvallarga kiritilgan o’zgarishlar bundan keyin kursorda aks etmaydi.

KEYSET - kursorning ushbu turi uchun tanlangan yozuvlarni aniqlaydigan kalit qiymatlar to’plami vaqtinchalik jadvalda saqlanadi. Kursor bo’ylab harakatlanayotganda, kalit bo’lmagan atributlarning qiymatlari tegishli jadvallardan olinadi, shuning uchun kursorni harakatlantirganda kalit bo’lmagan ustunlarga o’zgartirishlar ko’rinadi. Agar kursordagi qator FETCH operatori tomonidan olib kelinguncha jadvaldan allaqachon olib tashlangan bo’lsa, @@ FETCH_STATUS xizmat o’zgaruvchisi -2 qiymatini qaytaradi. Kursor ochilgandan so’ng jadvallarga qo’shilgan qatorlar kursorda ko’rinmaydi. Kursorni yaratuvchi so’rov noyob indeksga ega bo’lmagan kamida bitta jadvalni o’z ichiga olsa, KEYSET tipidagi kursor STATIC turiga o’tkaziladi.

DYNAMIC - eng ko’p resurs talab qiladigan kursor turi bo’lib, natijalar to’plamining qatorlarida kiritilgan barcha ma'lumotlar o’zgarishlarini, shu jumladan yangi kiritilgan qatorlarni aks ettiradi. Har bir tanlovdagi ma’lumotlar qiymatlari, tartib va qator a’zoligi farq qilishi mumkin. FETCH ABSOLUTE-ni dinamik kursorlar bilan ishlatib bo’lmaydi.

FAST_FORWARD kursorning eng tezkor turi bo’lib, bir qatordan ikkinchisiga faqat “oldinga” o’tish imkonini beradi. Bu FORWARD_ONLY va READ_ONLY parametrlari bilan e'lon qilingan kursorga ekvivalent.

READ_ONLY – “faqat o’qish uchun” kursorni belgilaydi: bunday kursor orqali ma’lumotlar bazasiga o’zgartirishlar kiritib bo’lmaydi.

SCROLL_LOCKS degani, SQL Server kursorda o’qilgan qatorlarni qulflashini anglatadi, bu esa kursor turi orqali ularni yangilash yoki o’chirishni ta'minlaydi.

OPTIMISTIC kalit so’zi bilan e'lon qilingan kursor qatorni blokirovka qilishni talab qilmaydi va ma'lumotlarni o’zgartirishga imkon beradi. Agar kursorda ma'lumotlar o’qilgandan so’ng asosiy jadvalga o’zgartirishlar kiritilgan bo’lsa, kursor orqali ushbu ma'lumotlarni o’zgartirishga urinish xatolikka olib keladi.

TYPE_WARNING kursor so’ralgan turdan boshqasiga bilvosita aylantirilganda (masalan, jadvalda yagona indeks mavjud bo’lmaganda yuqorida tavsiflangan KEYSET kursorini STATICga aylantirish) mijozga ogohlantirish yuborilishini bildiradi.

SELECT_STATEMENT - kursorning natijalar to’plamini tashkil etuvchi SELECT iborasi.

FOR UPDATE iborasi kursordagi yangilanadigan ustunlarni belgilaydi. Agar OF ustun_nomi [, . . . n] bo‘lsa, o‘zgartirishlar uchun faqat ro‘yxatdagi ustunlar mavjud bo‘ladi. Agar ustunlar ro’yxati bo’lmasa, barcha ustunlar uchun yangilash mumkin, faqat READ_ONLY parametri bilan kursor deklaratsiyasidan tashqari.

Kursorni ochish va to’ldirish uchun buyruqdan foydalaning

OPEN {{[GLOBAL] kursor_nomi} @kursor_uzgaruvchisi}

Ochilganda kursor nomi (kursor_nomi) yoki CURSOR tipidagi o’zgaruvchi (@kursor_uzgaruvchisi) orqali ko’rsatilishi mumkin. GLOBAL parametri kursor_nomi global kursor ekanligini bildiradi.

Viewlar: T-SQLda kengaytirilgan sintaksis

Ko’rinish asosan o’z nomi bilan ma'lumotlar bazasida saqlanadigan relyatsion ma'lumotlar bazasi obyektidir.

SQL Server foydalanuvchilarga ko’p sonli oldindan belgilangan tizim ko’rinishlarini taqdim etadi. Masalan, sys.check_constraints ko’rinishi joriy ma'lumotlar bazasidagi obyektlarda belgilangan CHECK cheklovlari haqida ma'lumot beradi. Joriy foydalanuvchi ruxsatlarga ega bo’lgan obyektlar haqida ma'lumot ko’rsatiladi. Quyida ushbu ko’rinishga kirish misoli keltirilgan:

USE MyTest
ON

```
SELECT * FROM sys.check_constraints
```

Foydalanuvchilar CREATE VIEW buyrug’i yordamida ko’rinishlarni yaratishi mumkin:

```
CREATE VIEW [sxema_name] view_name
[(ustun [,...n])]
[ ON <atribut_view> [,...n]]
AS tanlash_iborasi [WITH CHECK OPTION] [;]
```

Agar ko’rish nomidan keyin qavs ichida ustun nomlari ro’yxati bo’lmasa, u holda ular SELECT so’rovi natijalaridagi kabi qabul qilinadi (format tavsifida - select_statement). SQL Serverda ko’rinishda maksimal 1024 ta ustun bo’lishi mumkin.

WITH bandi quyidagi ko’rinish atributlarini belgilashi mumkin:

- 1) KRİPTIYA - yaratilgan ko’rinishning kodi shifrlash yordamida yashirilishini bildiradi:
- 2) SCHEMABINDING - "bog’langan" obyektlar mavjudligini kafolatlaydi, asosiy jadval yoki jadvallarni ko’rinishning ta'rifiga ta'sir qiladigan tarzda o’zgartirib bo’lmaydi. Agar ko’rinish SCHEMABINDING atributisiz yaratilgan bo’lsa, SQL Server hujjatlari ko’rinish ta'rifiga ta'sir qiluvchi obyektlarni o’zgartirgandan so’ng darhol sp_ref reshview saqlangan protsedurasini ishga tushirishni tavsiya qiladi;
- 3) VIEW_METADATA shuni bildiradiki, tizim tegishli so’rov bo’yicha ushbu ko’rinishning asosiy jadvallari haqida emas, balki ko’rinishning metama'lumotlari haqidagi ma'lumotlarni DB-Library, ODBC va OLE DB API-lariga qaytaradi.

WITH CHECK OPTION konstruksiyasi ko’rinish orqali ma’lumotlarni o’zgartirishda qo’shilgan yoki o’zgartirilgan ma’lumotlarning ko’rinish aniqlanadigan SELECT bayonotida ko’rsatilgan tanlash mezonlariga muvofiqligi tekshirilishini bildiradi. Ma'lumotlar bazasi jadvallaridagi ma'lumotlarni o’zgartirishingiz mumkin bo’lgan ko’rinish *yangilanadigan ko’rinish deb ataladi.*

Keling, oddiy ko’rinish yaratish va undan ma'lumotlar to’plamini olish misolini ko’rib chiqaylik:

```
USE MyTest
ket
CREATE VIEW dbo.NewBooks AS
SELECT Id, author, title, publish, [YEAR]
FROM dbo.Book1 WHERE [YEAR]>2010
```

ket

```
SELECT * FROM dbo.NewBooks
```

Endi taqdimot orqali ma'lumotlarni o’zgartirish masalalarini ko’rib chiqamiz. Agar quyidagi shartlar bajarilsa, asosiy jadval ma'lumotlarini ko’rinish orqali o’zgartirish mumkin:

- har qanday o’zgarishlar, shu jumladan UPDATE, INSERT va DELETE iboralari orqali amalga oshiriladigan o’zgarishlar faqat bitta asosiy jadval ustunlariga murojaat qilishi kerak;
- ko’rinishda o’zgartirilgan ustunlar to’g’ridan-to’g’ri asosiy jadval ustunlaridagi ma'lumotlarga havola qilishi kerak;
- hisoblangan ustunlarni, shu jumladan agregat funksiyalar (AVG, COUNT, SUM va boshqalar) yordamida tuzilgan ustunlarni va UNION, UNION ALL, CROSS JOIN, EXCEPT VA KESISHI operatorlari yordamida olingan ustunlarni o’zgartirish mumkin emas;
- o’zgartirilayotgan ustunga GROUP BY, HAVING va DISTINCT ko’rsatmalari ta'sir qilmasligi kerak;
- SELECT operatoridagi ko’rinish ta'rifi, agar ko’rinish WITH CHECK OPTION parametri bilan aniqlangan bo’lsa, TOP operatoridan foydalanmasligi kerak.

Agar yuqoridagi shartlar bajarilmasa, lekin ma'lumotlarni o’zgartirish imkoniyatini ta'minlash zarur bo’lsa, ba'zi hollarda bu vazifani INSTEAD OF tipidagi DML triggerlari bilan hal qilish mumkin.

Shuni ham hisobga olish kerakki, agar jadvalda ko’rinishga kiritilmagan NOT NULL cheklovli ustunlar bo’lsa, ushbu ko’rinish orqali yangi yozuv qo’shishga harakat qilganingizda xatolik yuz berishi mumkin. Bunday holatda, etishmayotgan qiymat avtomatik ravishda yaratilsa (masalan, surrogat asosiy kalitlarni yaratish uchun ishlatiladigan turli hisoblagichlar) yoki ustun standart qiymatga ega bo’lsa, xato bo’lmaydi.

Nazorat savollari

1. Saqlangan protseduralar T-SQL da qanday yaratiladi va qanday ishlaydi?
2. Funksiyalar T-SQL da qanday yaratiladi va qanday ishlaydi?
3. Trigerlar T-SQL da qanday yaratiladi va qanday ishlaydi?
4. Ko‘rinishlar T-SQL da qanday yaratiladi va qanday ishlaydi?

5. Saqlangan protseduralar qanday qo'llaniladi ma'lumotlar bazasining ma'lumotlarini o'qish uchun?
6. Funksiyalar qanday ishlatiladi ma'lumotlar bazasida ma'lumotlarni qayta ishlash uchun?
7. Trigerlar qanday ishlatiladi ma'lumotlar bazasida avtomatik amallarni bajarish uchun?
8. Koʻrinishlar qanday qo'llaniladi ma'lumotlar bazasidan ma'lumotlar ko'rish uchun?
9. Saqlangan protseduralar, funksiyalar, trigerlar va koʻrinishlar orasidagi farq qanday?
10. T-SQL da saqlangan protseduralar, funksiyalar, trigerlar va koʻrinishlar yaratishda qanday sintaksis ishlatiladi?
11. Koʻrinishlar va saqlangan protseduralar arasidagi boshqaruv munosabatlari qanday?
12. T-SQL da trigerlar va funksiyalar orasidagi farq qanday?

14-MAVZU. XML FORMATINI QO‘LLAB-QUVVATLASH.

Reja:

1. XML formatini qo’llab-quvvatlash
2. Relyatsion ma’lumotlar jadvalidan ma’lumotlarni XML ko‘rinishida olish.

XML (Xextensible Markup Language) - bu dasturiy ta'minot tizimlari o’rtasida ma'lumotlar almashish, shuningdek, ma'lumotlarni import va eksport qilish uchun keng qo’llaniladigan kengaytirilgan ierarxik matn belgilash tilidir. Uning texnik tavsiflari Internet uchun texnologiya standartlarini yaratuvchi va joriy qiluvchi tashkilot World Wide Web Consortium (W3C) tomonidan ishlab chiqilgan.

XML tilini qo’llab-quvvatlash barcha asosiy korporativ darajadagi relyatsion MBBTlar tomonidan taqdim etiladi va quyidagi vazifalarni hal qilish mumkin:

- XML formatidagi relyatsion jadvallardan ma'lumotlarni olish;
- relyatsion jadvallarda o’zgartirish va saqlash uchun ma'lumotlar bazasi ma'lumotlarini XML formatida uzatish;
- ma'lumotlar bazasida XML hujjatlarini skalyar qiymatlar bilan birga saqlash va qayta ishlash.

Quyida XML formati haqidagi minimal ma’lumotlar keltirilgan bo‘lib, siz ushbu bobda nima bo‘lishini tushunishingiz kerak bo‘ladi.

XML hujjati prolog va ildiz elementdan iborat. Prolog deklaratsiyalarni, qayta ishlash ko’rsatmalarini, sharhlarni o’z ichiga olishi mumkin va ixtiyoriydir, ya'ni. etishmayotgan bo’lishi mumkin.

Ildiz elementi hujjatning talab qilinadigan qismidir. Ildiz elementi ichiga o‘rnatilgan elementlar, matn ma’lumotlari va izohlar kirishi mumkin. O’rnatilgan element shunga o’xshash bo’ladi.

Quyida til versiyasini *ko’rsatuvchi* XML deklaratsiyasini o’z ichiga olgan prologga ega XML hujjati mavjud . Deklaratsiyada hujjatni kodlash to’g’risidagi ma'lumotlar ham bo’lishi mumkin. XML 1.0 versiyasi deklaratsiyasining yo’qligiga ruxsat berdi, shuning uchun deklaratsiyasiz hujjat XML 1.0 yordamida yaratilgan deb hisoblanadi:

```
<?XML version="l.1"?>
<greeting>Salom, dunyo!</greeting>
```

XML hujjatidagi elementlar axborotni tartibga solish uchun javob beradi va XML tilining asosiy tarkibiy birliklari hisoblanadi. Yuqoridagi misolda bitta element bor - greeting. Bu yerda <greeting> elementning boshlang’ich tegi *(inglizcha* teg - matnni belgilash konstruksiyasi); </greeting> - bu uning yakuniy tegi; "Salom Dunyo!" - bu elementning matn ma'lumotlari.

Matn ma’lumotlari yoki ichki o‘rnatilgan elementlarni o‘z ichiga olmagan bo‘sh elementni <element/> sifatida stillashtirish mumkin, bu teg bir vaqtning o’zida ham boshlang’ich, ham yakuniy. Bo’sh element uchun atributlar o’rnatilishi mumkin.

Elementlar ularga tayinlangan qiymatlarga ega atributlarni o’z ichiga olishi mumkin. Atributlar elementning boshlang’ich teglarida joylashtiriladi va element haqida ma'lumot qo’shish imkonini beradi. Atribut qiymati bitta yoki ikkita tirnoq ichiga olinadi. Masalan, xodimni tavsiflashda siz uning shaxsiy raqamini atribut orqali ko’rsatishingiz mumkin:

```
<worker tabnuiti="11">Aliyev SH.E</worker>
```

Bir nechta atributlar bo’lishi mumkin, bu holda ular probel bilan ajratiladi.

XML hujjati quyidagi qoidalarga javob bersa, *yaxshi* tuzilgan XML hujjati hisoblanadi:

- ildiz elementi mavjud bo’lishi kerak va faqat bitta bo’lishi kerak;
- har bir element boshlang’ich teg bilan boshlanib, yakuniy teg bilan tugashi kerak;
- element boshqa elementlarni, atributlarni va matn ma'lumotlarini o’z ichiga olishi mumkin;
- atribut qiymatlari qo’shtirnoq ichiga olinadi; matn ma'lumotlari, aksincha, qo’shtirnoq ichiga kiritilmaydi.

XML hujjati quyidagi hollarda haqiqiy yoki yaroqli deb nomlanadi *(inglizcha joriy XML hujjati):*

- to’g’ri tuzilgan;
- *XML sxemasi yoki* Document Type Definition *(* DTD) yordamida belgilangan qoidalarga mos keladi.

Relyatsion MBBT bilan ishlashda ba'zi hollarda XML hujjatining prolog va ildiz elementi bo’lmagan qismlaridan foydalanish mumkin. Quyida bunday parchaning namunasi keltirilgan:

```
<dbo.Book1 Muallif="HuMK Ben-Gan"/>
<dbo.Book1 Muallif="XoTeK M."/>
```

XML sifatida relyatsion jadvallardan ma'lumotlarni olish

Ba'zi hollarda siz XML formatidagi relyatsion ma'lumotlar bazasi jadvallariga nisbatan so’rov natijalarini olishingiz kerak bo’lishi mumkin, masalan, ularni keyingi qayta ishlash uchun boshqa dasturga o’tkazish uchun. SQL Serverda buni SELECT iborasining oxiriga qo’shilgan FOR XML bayonoti yordamida amalga oshirish mumkin. Misol tariqasida, quyida bunday so’rov matni keltirilgan:

```
SELECT [Author] , [Title]
dbo.Book1 FROM DISTINCT
FOR XML AVTO
```

So’rovni bajargandan so’ng, XML hujjatining quyidagi qismi olindi, unda har bir natija to’plami yozuvi bitta elementga mos keladi va ustun qiymatlari elementning atributlarini tashkil qiladi:

```
<dbo.Book1 Author="MuMK Ben-Gan"
Title="Microsoft SQL Server 2008. T-SQL asoslari"/>
<dbo.Book1 Muallif="XoTeK M."
Title="SQL Server 2008. Amalga oshirish va texnik xizmat
ko'rsatish."/>
```

Transact-SQL da FOR XML operatori bir necha ish rejimlariga ega.

RAW [('ElementName')] - natijalar to’plamidagi har bir qatorni XML elementiga o’zgartiradi. Element identifikatori qavslar ichida aniq ko’rsatilishi yoki standart identifikator "qator" ishlatilishi mumkin. Ildiz element nomini belgilash uchun RAW dan keyin vergul bilan ajratilgan ROOT kalit so’zidan foydalanishingiz mumkin. Masalan:

```
SELECT * dbo.Book1
FOR XML RAW ('Kitob'), ROOT ('Kitoblar')
```

So’rov natijasi:

```
<Books>
<Book Id="101" Author="HuHK Ben-Gan" Title="Microsoft SQL
Server 2008. T-SQL asoslari" Publisher="BXB-neTep6ypr"
year="2009" />
<Book Id="102" Muallif="XoTeK M." Title="SQL Server 2008.
Amalga oshirish va texnik xizmat ko'rsatish." year="2011" />
</Books>
```

Agar siz yozuv maydonlarini element atributlari sifatida emas, balki ichki o’rnatilgan elementlar sifatida ko’rsatishni istasangiz, buni

vergul bilan ajratilgan ELEMENTS kalit so’zini ko’rsatish orqali amalga oshirishingiz mumkin.

Bundan tashqari, oldingi misolga e'tibor qaratsangiz, ikkinchi kitobga mos keladigan elementda Publisher atributi yo’qligini sezasiz. Relyatsion jadvalda bu maydon NULL qiymatiga ega edi va XML yaratishda sukut bo’yicha atribut yoki element (agar ELEMENTS o’rnatilgan bo’lsa) shunchaki o’tkazib yuboriladi. Agar siz NULL qiymatini qayta ishlashda elementni saqlamoqchi bo’lsangiz, FOR XML bayonotiga ELEMENTS XSINIL kalit so’zlarini qo’shishingiz kerak. Masalan:

```
REQUEST SELECT DISTINCT [Title], [Publisher]
FROM dbo.Book1
FOR XML RAW ('Kitob'), ROOT ('Kitoblar')
ELEMENTS XSINIL
OUTPUT RESULT
<Books XMLns:xsi="w3.org/2001/XMLSchemainstance">
   <Book>
<Title>Microsoft SQL Server 2008. T-SQL asoslari</Title>
<Publisher>BXB-neTep6ypr</Publisher>
   </Book>
   <Book>
      <Title>SQL Server 2008. Amalga oshirish va texnik xizmat
kursatish. </Title>
      <Publisher xsi:nil="true"/>
   </Book>
</Books>
```

Bu yerda, ildiz elementining ochilish tegida xsi deb nomlangan XML nom maydoniga havola mavjud. Bu nomlar maydoni <Publisher xsi:nil="true"/> elementida publisher nomi aniqlanmaganligini ko’rsatish uchun ishlatiladigan nil atributini belgilaydi.

Agar so’rov natijalari asosida oddiy ierarxik tuzilmalarni shakllantirish kerak bo’lsa, AVTO rejimi qulay. Odatiy bo’lib, so’rovlar chiqishi to’plamining satrlari elementga aylantiriladi va ustun qiymatlari element atributlariga aylantiriladi. Elementlarning nomlari manba jadvallari yoki ko’rinishlarining nomlari bilan bir xil bo’ladi, lekin agar ular uchun FROM bo’limida taxalluslar ko’rsatilgan bo’lsa, ular element nomida ham qo’llaniladi. Oldingi holatda bo’lgani kabi, siz ROOT va ELEMENTS ko’rsatmalaridan foydalanishingiz mumkin.

Quyidagi so’rov xorijiy kalit bilan bog’langan ikkita jadvaldan ma'lumotlarni oladi. FROM bo’limida jadvallar uchun taxalluslar belgilanadi, ular keyinchalik natijalar ustunlarini nomlashda ishlatiladi. dbo.Book1 jadvali kitob nashrlarini tavsiflaydi, dbo.Lib1 jadvali kutubxonada mavjud nashrlarning nusxalarini va ularning hozirgi holatini tavsiflaydi:

```
SELECT [Author], [Title], [LibId], [Status]
FROM dbo.[Book1] AS [publish] INNER JOIN
dbo.[Lib1] [Book] AS
ON [publish].[Id] = [Kitob].[Id]
FOR XML AUTO, ROOT ("Books")
```

So’rov natijasi quyida keltirilgan. FOR XML AUTO bayonoti dbo dan ma'lumotlarni qayta ishlaganligini unutmang. Lib1, dbo.Book1 jadvaliga havola qilib, ichki elementlar sifatida tashkil etilgan:

```
<Books>
   <publish Author="ktuMK Ben-Gan" Title="Microsoft SQL Server
2008. T-SQL asoslari">
      <Book LibId="1" Status="Kutubxonada"/>
      <Book LibId="2" Status="BunaHa"/>
   </publish>
   <publish Author="XoTeK M." Title="SQL Server 2008. Amalga
oshirish va texnik xizmat ko'rsatish.">
      <Book Libld="3" Status="Kutubxonada"/>
   </publish>
</Books>
```

FOR XML EXPLICIT rejimida ishlatilsa, XML daraxt tuzilishi aniq aniqlanadi. Bu juda murakkab sintaksisni talab qiladi, bu yerda PATH rejimi oddiyroq alternativani ifodalaganligi sababli bu yerda taqdim etilmaydi. Siz SQL Server hujjatlarida EXPLICIT sintaksisini ko’rib chiqishingiz mumkin.

FOR XML PATH rejimi natijada olingan XML hujjatidagi elementlar va atributlarni aralashtirish, shuningdek, murakkab xususiyatlarni ifodalash uchun qo’shimcha joylashtirish darajalarini joriy qilish imkoniyatini beradi. SQL Server hujjatlari ko’p hollarda FOR XML PATH dan foydalanish eng yaxshi tanlov ekanligini ta'kidlaydi.

Odatda, PATH rejimi natijalar to’plamidagi har bir qator uchun <satr> element o’ramini yaratadi. Element nomini ham belgilashingiz

mumkin. Agar nom ko’rsatilgan bo’lsa, u elementning o’ramining nomi sifatida ishlatiladi. Agar siz bo’sh qatorni taqdim qilsangiz (FOR XML PATH ('')), element o’rami yaratilmaydi. Keling, buni RAW rejimi qanday ishlashini ko’rsatish uchun avval ishlatilgan so’rov misolidan foydalanib tushuntiramiz:

```
DISTINCT SELECT [Title], [Publish]
FROM dbo.Book1
FOR XML PATH
```

So’rov natijasi quyida keltirilgan. Shuni esda tutish kerakki, bizning ma'lumotlar bazasida ikkinchi kitob uchun nashriyot ko’rsatilmagan, shuning uchun tegishli element natijalar to’plamida yo’q:

```
<row>
   <Title>Microsoft SQL Server 2008. T-SQL asoslari</Title>
<Publisher>BXB-neTep6ypr</Publisher>
</row>
<row>
   <Title>SQL Server 2008. Amalga oshirish va texnik xizmat
ko'rsatish.</Title>
</row>
```

PATH rejimida siz ROOT va ELEMENTS iboralaridan ham foydalanishingiz mumkin. Agar alohida qiymatlarni elementning atributlari sifatida formatlash kerak bo’lsa, buni SELECT bo’limida qo’sh tirnoq ichida atribut nomini ko’rsatish orqali amalga oshirish mumkin:

```
SELECT DISTINCT [Title], [Publisher] AS "@Publ"
FROM dbo.Book1
FOR XML PATH ('Publish'), ROOT ('Books')
```

So’rov natijasi quyidagicha bo’ladi:

```
<Books>
   <publish>
      <Title>SQL Server 2008. Amalga oshirish va texnik xizmat
ko’rsatish.</Title>
   </publish>
<Lib1 publish="BHV-Peterburg">
   <Title>Microsoft SQL Server 2008. T-SQL asoslari</Title>
   </publish>
</Books>
```

Shuningdek, so’rov natijasiga sharhlar, XML doimiylari, nomlar bo’shliqlari va boshqalarni qo’shishingiz mumkin. Quyidagi so’rovda kitob muallifi haqidagi ma'lumotlar sharh sifatida formatlanadi va yaratilgan hujjatga yangi bo’sh <Test/> elementi qo’shiladi:

```
SELECT [Publisher] AS "@Publ",
[Author] AS "comment()"
CAST('<Test/>' AS XML) AS "node0",
[Title]
FROM dbo.Book1
FOR XML PATH ('Publish'), ROOT ('Books')
```

So’rov natijasi quyida ko’rsatilgan:

```
<Books>
   <Publish RiY="BHV-Peterburg">
   <Test/>
      <Title>Microsoft SQL Server 2008. T-SQL asoslari</Title>
   </Publish><Publish>
   <Test/>
      <Title>SQL Server 2008. Amalga oshirish va texnik xizmat
ko'rsatish.</Title>
   </Publish>
</Books>
```

FOR XML PATH bayonoti ichki so’rovlarni amalga oshirish imkonini beradi. Quyida XML hujjatining fragmentini qaytaradigan quyi so’rovdan foydalanadigan misol keltirilgan:

```
SELECT [Publisher] AS "@Publ",
[Title] AS "@Title",
SELECT [LibId] AS "SLibld",
FROM dbo.Lib1 [Status] AS "Status"
WHERE dbo.Lib1.Id=dbo.Book1.ID
FOR XML PATH ('Book'), TYPE
FROM dbo.Book1 WHERE Id=101
FOR XML PATH ('Pulish'), ROOT ('Books')
```

Quyi so’rovda ma'lumotlarni to’g’ri formatlash uchun zarur bo’lgan TYPE bayonoti so’rov natijalarni XML turi sifatida qaytarishini bildiradi. Ushbu so’rovni bajarish natijasi:

```
<Books>
   <Publish RiY="BHV-Peterburg" Title="Microsoft SQL Server
2008. T-SQL asoslari">
```

```
<Book LibId="l" Status="Kutubxonada"/>
<Book LibId="2" Status="Kutubxonada" />
   </Publish>
</Books>
```

FOR XML PATH rejimining imkoniyatlari yuqorida sanab o’tilganlar bilan cheklanmaydi va batafsil ma'lumot uchun SQL Server hujjatlariga murojaat qilish tavsiya etiladi.

Nazorat savollari:

1. XML nima?
2. XML ni qachon ishlatish mumkin?
3. XML foydalanishining asosiy maqsadi nima?
4. XML-ga misollar qanday kiritiladi?
5. XML va HTML orasidagi farq nima?
6. XML strukturasining asosiy qismi nima?
7. XML fayllarini qanday tuzish mumkin?
8. XML-ga dastur yozish uchun qanday vositalar mavjud?
9. XML-ni qo’llashda foydali dasturlar nima?
10. XML-ni ma'lumot almashish vaqti kerak bo’lgan holatlar nima?

15-MAVZU. XML MA’LUMOTLAR TURIDAN FOYDALANISH.

Reja:

1. XML ma'lumotlar turidan foydalanish
2. XML faylni ochish
3. XML faylda ma'lumotlar kiritilishi

SQL Server ma'lumotlar bazasini boshqarish tizimi XML hujjatlari va fragmentlarini saqlash imkonini beruvchi XML ma'lumotlar turini qo’llab-quvvatlaydi. Berilgan turdagi ustun yoki o’zgaruvchini aniqlaganingizda, u javob berishi kerak bo’lgan cheklovlarni belgilashingiz mumkin:

XML ([CONTENT I DOCUMENT] XML_schema_collection)

CONTENT kalit so’zi ma'lumotlarning yaxshi shakllangan XML fragmenti ekanligini ko’rsatadi. XML ma'lumotlari 0 yoki undan ortiq yuqori darajadagi elementlarni o’z ichiga olishi mumkin, matnli ma'lumotlar yuqori darajali tugunlar uchun ruxsat etiladi. Xuddi shu cheklovlar sukut bo’yicha bo’ladi (agar siz hech narsa ko’rsatmasangiz).

Agar *DOCUMENT* kalit so’zi ishlatilsa, u holda ma'lumotlar yaxshi shakllangan XML hujjati bo’lishi kerak: faqat bitta ildiz elementi bo’lishi kerak. Yuqori darajadagi elementdagi matn ma'lumotlari ta’qiqlanadi.

Aniqlashda siz ro’yxatdan o’tgan sxemalar to’plamini belgilashingiz mumkin, XML_schema_collection. Keyin XML ma'lumotlari sxema bo’yicha tekshiriladi va u *yozilgan* XML ustuni yoki o’zgaruvchisi bo’ladi.

Quyidagi misolni ko’rib chiqing, u XML ustunli jadval yaratadi va keyin unga ma'lumotlarni qo’shadi:

```
T1 CREATE TABLE(c1 int PRIMARY KEY, c2 XML)
DECLARE @s varchar(lOO)
SET
@s='<Cust><Fname>MBaH</Fname><Lname>IIeTpoB</Lnamex/Cust>'
INSERT INTO T1 VALUES (, cast (@s AS XML))
```

Shuni ta'kidlash kerakki, ichki darajada XML ma'lumotlar turi varbinar (maks) ma'lumotlar turi yordamida saqlanadi, ya'ni, XML

matn qatori sifatida saqlanmaydi, lekin ikkilik ko’rinishda. XML ma'lumotlar turi misollarining saqlangan ko’rinishi hajmi 2 GB dan oshmasligi kerak. Yuqoridagi misol literal konstantani XML turiga aniq aylantirish uchun *cast* funksiyasidan foydalanadi.

FOR XML bayonoti bilan SELECT iborasini bajarish orqali XML turi qiymatini ham olishingiz mumkin:

```
DECLARE @xDoc XML
SET @xDoc = ([Title], [Publish]
FROM dbo.Book1
FOR XML AUTO )
```

SQL Serverda XML ma'lumotlariga qarshi so’rovlarni yozish uchun siz XML ma'lumotlar turidagi usullardan foydalanishingiz mumkin (15.1-jadval). Jadvalda keltirilgan XQuery tili XML ma'lumotlarini so’rash imkonini beradi. ***XQuery tili*** XPath so’rovlar tiliga asoslangan, takomillashtirilgan qo’llab-quvvatlash iteratsiyalar, saralash natijalari va kerakli XML tuzilmalarini qurish qobiliyatidir.

Keling, bir nechta misollarni ko’rib chiqaylik. Birinchisi, XML hujjatining birinchi ProductDescription elementining ProductID atributining qiymatini olish uchun value() usulidan foydalanadi. Bizni birinchi elementga qiziqtirganligimiz "[1]" parametri bilan ko’rsatilgan. 0ProdID o’zgaruvchisi 10 qiymati bilan tugaydi:

```
DECLARE XML @myDoc
DECLARE @ProdID int SET @myDoc = '<Root>'
      <Product description productID = "10" Product Name
"velosiped"/>
      <Product description productID = "15" Product
Name="Mashina"/>
'</Root>'
SET @ProdID = @myDoc.value (
   '/Root/ProductDescription/@ProductID) [1]', 'int')
```

15.1-jadval

XML **ma'lumotlar turi usullari**

Metod(Usul)	**Tavsif**
Request Xquery	XML ma'lumotlar turi namunasi uchun XQuery so’rovini belgilaydi. Natija XML ma'lumotlar turiga ega (tiplanmagan)

qiymat (XQuery, SQLType)	XML strukturasiga qarshi XQuery so’rovini bajaradi va qo’llab-quvvatlanadigan SQL turining skaler qiymatini qaytaradi
mavjud (XQuery)	Bit tipidagi qiymatni qaytaradi. Agar XQuery ifodasi so’ralganda bo’sh bo’lmagan natijani qaytarsa, "1" To’g’ri degan ma'noni anglatadi. "0" noto’g’ri degan ma'noni anglatadi - bo’sh natijani qaytarishda. NULL, agar so’rov qilingan XML tipidagi namunada NULL qiymati bo’lsa
tugunlar (XQuery) Jadval (ustun) sifatida	XML ma'lumotlar namunasini aloqador ma'lumotlar to’plamiga bo’lish imkonini beradi
o’zgartirish (XML_DML)	Ushbu usul XML tipidagi o’zgaruvchi yoki ustunning mazmunini o’zgartirish uchun ishlatiladi: XML DML bayonoti XML ma'lumotlaridan tugunlarni kiritish, yangilash yoki o’chirishni belgilaydi.

Quyidagi misol so’rov usulidan foydalanishni ko’rsatadi. T1 jadvali GO (harf mamlakat kodi) va REGIONS - mintaqalar ro’yxati, ularning markazlari va boshqalarni o’z ichiga olgan XML maydoni bo’lsin. Rossiyaga mos keladigan XML hujjatining fragmenti quyida keltirilgan (hujjatning ko’p qismi o’tkazib yuborilgan va bo’shliqlar o’rniga ellipslar joylashtirilgan; ular XMLning bir qismi hisoblanmasligi kerak):

```
<REGIONS>
   <REGION REGID="01" REGNAME="Adigeya Respublikasi
(Adigeya)" REGCENTER="MAYKOP" />
   <REGION REGID="02" REGNAME="Oltoy Respublikasi"
REGCENTER="GORNO-ALTAISK" />
   <REGION REGID="35" REGNAME="Vologda viloyati"
REGCENTER="BOJIOrflA" />
</REGIONS>
```

Keling, Vologda shahrida joylashgan mintaqaga mos keladigan XML hujjat elementini REG deb nomlangan ustun shaklida qaytaradigan so’rov misolini ko’rib chiqaylik. Shart REGCENTER atributining qiymati uchun tekshiriladi. XQuery (XML Query

Language) sintaksisiga ko’ra, so’rovdagi "@" belgisi bu atribut ekanligini bildiradi. E'tibor bering, WHERE bo’limida Rossiyaga mos keladigan (ID='RUS') qator oldindan tanlangan. Aks holda, so’rov manba jadvalidagi qancha mamlakatlar bo’lsa, shuncha yozuvlarni qaytaradi va Rossiyadan tashqari barcha mamlakatlar uchun ustun NULL qiymatlarga ega bo’ladi. Shunday qilib, bu yerda bitta so’rovda tanlash shartlari ham relyatsion, ham XML ma'lumotlari uchun tekshiriladi:

```
SELECT
REGIONS.Request('/REGIONS/REGION[0REGCENTER="VOLOGDA"]')
AS REG
FROM T1
WHERE ID='RUS'
```

Agar sizga butun element emas, balki uning individual atributining qiymati kerak bo’lsa, unda siz yuqorida aytib o’tilgan qiymat usulidan foydalanishingiz kerak. Quyida yuqorida aytib o’tilgan T1 jadvalidan mintaqaning nomi (@REGNAME atributi) olingan misol keltirilgan. Atribut qiymati varchar(100) turiga aylantiriladi. Shunga qaramasdan

faqat bitta REGNAME atributi bo’ladi, rasmiy ravishda bilishimiz kerakki, so’rov matnida "[1]" bilan ko’rsatilganidek, birinchi bunday atributning qiymati kerak:

```
SELECT REGIONS.value (

'(/REGIONS/REGION[@REGCENTER="VOLOGDA"]/@REGNAME)'
   varchar(100))
AS REG
FROM T1
WHERE ID='RUS'
```

XML hujjatiga ichki o’rnatilgan elementni qo’shishning yakuniy misolidir. Bu yerda modify usulidan foydalanib, birinchi r elementiga yangi ichki rl elementi qo’shiladi. As first kalit so’zlari qo’shilayotgan element birinchi o’ringa joylashtirilishini bildiradi:

```
DECLARE XML @x
DECLARE bit @f
SET @x = '<r>'
```

```
<rl Somedate = "2011-01-01Z"/>
<rl Somedate = "2012-01-01Z"/>
<rl Somedate = "2013-01-01"/> '</r>'
SET @x.modify(FIRST AS
<rl Somedate = "2010-01-01Z"/>
INPUT IN '</r)[1]>'
SELECT @x
```

Taqdim etilgan misollar XML formatida saqlangan ma'lumotlar bo'yicha so'rovlarni bajarish imkoniyatlari haqida fikr beradi. Shu bilan birga, SQL Serverda XML DML deb nomlangan XML formatidagi ma'lumotlarni o'zgartirish subtilini va XQuery tilini batafsil o'rganish alohida kitob mavzusidir. Agar siz ushbu xususiyatlardan foydalanishingiz kerak bo'lsa, maxsus buyruqlar sintaksisini SQL Server hujjatlarida aniqlashtirish mumkin.

Ma'lumotlarni XML formatidan jadval ko'rinishiga aylantirish

FOR XML bayonoti ma'lumotlarni relyatsion tasvirdan XMLga aylantirish imkonini beradi. Transact-SQL da teskari muammoni OPENXML bayonoti yordamida hal qilish mumkin. Keling, undan foydalanish misolini ko'rib chiqaylik.

Mamlakatning to'liq va qisqartirilgan nomi, uning poytaxti, mintaqalar ro'yxati va ularning poytaxtlari (XML formatida) haqidagi ma'lumotlarni saqlaydigan T1 jadvali bo'lsin. Quyida uni yaratadigan skript mavjud:

```
T1 CREATE TABLE(
   ID CHAR(3) PRIMARY KEY,
   Nomi VARCHAR(60),
   FULL NAME VARCHAR(10 0),
   Poytaxti VARCHAR(60),
   REGIONS XML
);
```

Misol sifatida, Buyuk Britaniya yozuviga mos keladigan REGIONS atributining qiymatini ko'rib chiqing:

```
<REGIONS>
   <REGION REGID="EN" REGNAME="Angliya"
REGCENTER="LONDON"/>
   <REGION REGID="NI" REGNAME="Shimoliy Irlandiya"
REGCENTER="BELFAST"/>
```

```
   <REGION REGID="SC" REGNAME="Shotlandiya"
REGCENTER="EDINBURG" />
   <REGION REGID="WL" REGNAME="Uels"
REGCENTER="CARDIFF"/>
</REGIONS>
```

Aytaylik, biz ushbu XML hujjatidagi ma'lumotlarni relyatsion formatda taqdim qilmoqchimiz. Buni quyidagicha amalga oshirish mumkin. Birinchidan, biz kerakli qiymatni tayinlaydigan XML tipidagi o’zgaruvchini e'lon qilamiz. Izohlar bilan skript kodi quyida keltirilgan.

```
DECLARE @idoc int, @reg XML;
SET @reg = (ID='GBR' T1 jadvalidan mintaqani tanlash)
/* saqlangan protsedura hujjatning ichki ko’rinishiga ishlov berishni
yaratadi */
EXEC sp_XML_preparedocument @idoc OUTPUT, @reg;
/* OPENXML bilan SELECT XML hujjatini relyatsion ma’lumotlar
to’plamiga aylantiradi */
SELECT *
OPENXMLDAN (@idoc, /REGIONS/REGION1, 1),
BILAN REGID CHAR(2),
REGNAME VARCHAR(10 0),
REGCENTER VARCHAR(60);
```

Nazorat savollari

1. XML nima uchun ishlatiladi?
2. XML fayllarini qanday yaratish mumkin?
3. XML fayllarini qanday ochish mumkin?
4. Ma'lumotlar turi XML faylda qanday belgilanadi?
5. XML faylda qo'shimcha xususiyatlar qanday belgilanadi?
6. XML faylda ma'lumotlar tuzilishi qanday bo'lishi kerak?
7. Ma'lumotlar kiritilishi uchun moslashtirilgan usullar qanday?
8. XML faylda ma'lumotlar tashqi bog'lash qanday amalga oshiriladi?
9. XML faylda ma'lumotlar validatsiyasi qanday tekshiriladi?
10. Ma'lumotlar XML faylda tahriri qanday amalga oshiriladi?

GLOSSARIY

Ma'lumotlar bazasi (MB): Ma'lumotlar bazasi, ma'lumotlar to'plamini saqlash, o'rganish va ulardan foydalanish uchun axborot tizimi. Ushbu tizim ma'lumotlarni turli usullar bilan saqlaydi va ularga foydalanishni tashkil etadi.

Boshqarish tizimi: Bu tizimlar MBni boshqarish uchun tuzilgan dasturlar va vositalardir. Ular ma'lumotlarni qo'shish, o'chirish, o'zgartirish, so'rovlar yaratish, ma'lumotlar bilan ishlash va boshqalar uchun yordamchi funksiyalarni o'z ichiga oladi.

Ma'lumotlar saqlash: Ma'lumotlar bazasida ma'lumotlarni saqlash jarayoni. Bu, ma'lumotlar bazasining tuzilishi va ma'lumotlarni saqlash usullarini, jamlashuvi va boshqarilishini o'z ichiga oladi.

Ma'lumotlar o'rganish: Bu MBdan foydalanuvchilar uchun ma'lumotlarni izlash, tartiblash va chiqarish jarayoni. Ma'lumotlar o'rganish tizimi ma'lumotlarni kerakli shakl va tartibda topish va chiqarish uchun kerakli funksiyalarni o'z ichiga oladi.

Ma'lumotlarni eksportlash va importlash: Ma'lumotlar bazasidagi ma'lumotlarni boshqa formatlarda eksport qilish va boshqalardan import qilish imkoniyatini ta'minlaydigan tizimlar.

Aniqlik va xavfsizlik: Bu qisqichina dasturlar ma'lumotlar bazasidagi ma'lumotlarni himoyalash, foydalanuvchilar ma'lumotlarga faqat kerakli huquqlarga ega bo'lishi shartini boshqarish uchun qo'llaniladi.

Monitoring va reporting: Ma'lumotlar bazasining amalga oshirilishini va ishlashini nazorat qilish va xabarlar yaratish tizimlari. Ular MBni qo'llab-quvvatlash, muvaffaqiyat darajasini baholash va ma'lumotlar bazasining davriy ishlashini ta'minlashda yordam beradi.

Ma'lumotlar integratsiyasi: Boshqa tizimlar va dasturlar bilan ma'lumotlar bazasi o'rtasidagi ma'lumotlar almashinuvi. Bu tizimlar ma'lumotlarni olingan va unga chiqarilgan tizimlar o'rtasidagi almashuv va bog'lanishni ta'minlaydi.

Tizimni monitoring va scaling qilish: Ma'lumotlar bazasining ishlashini nazorat qilish va qo'shimcha yukni qabul qilish uchun ma'lumotlar bazasini o'lchash, ishlash darajasi va optimallashtirish tizimlari.

Ma'lumotlar tahlili: Bu tizimlar ma'lumotlar bazasidagi ma'lumotlarni tahlil qilish va ma'lumotlar izini qoldirish uchun qo'llaniladi. Ular ma'lumotlar bazasidan ma'lumotlarni olib, ularni tahlil qiladi va shu ma'lumotlardan foydalanish tartibini takomillashtirishga yordam beradi.

Ma'lumotlar interfeysi (API): Bu, ma'lumotlar bazasini boshqarish tizimlari bilan tashqi tizimlar o'rtasidagi aloqalarni amalga oshirish uchun ishlatiladigan interfeyslardir. Ushbu interfeyslar ma'lumotlarga kirish va ma'lumotlardan chiqish uchun protokollarni, buyruqlarni va standartlarni ta'minlaydi.

SQL: Strukturalangan so'rovlar tili.

Cess: Microsoft Office RMBBT dasturi.

Append: Jadval oxirigacha ma'lumotlarni qo'shish.

Accending order: Eng past va eng yuqori uchun sanada asoslangan matn sohasida alifbo tartibi.

Autonumber field: Yozishga qaraganda katta maydonga qo' shimcha ravishda avtomatik saqlash.

Database: Tegishli ma'lumotlarni yig'ish.

Database sxema: Ma'lumotlar bazasida jadvallar yacheykasiga ma'lumotlarning bayoni va ma'lumotlarni uzatishni tashkil etish

Datasheet: Ma'lumotlar uchun satrlar ustunlar sohalarda va yozuvlar bilan tashkil etish.

FOYDALANILGAN ADABIYOTLAR RO'YHATI

1. O'zbekistan Respublikasi prezidentining 2017 yil 7 fevraldagi PF- 4947-son "O'zbekistan Respublikasini yanada rivojlantirish boyicha Xarakatlar strategiyasi to'g'risida"gi Farmoni.
2. Mirziyoyev Sh.M. Buyuk kelajagimizni mard va olijanob xalqimiz bilan birga quramiz. Toshkent. "O'zbekistan", NMIU, 2017. - 488 b.
3. Guliamova M.K., & Aliev R.M. (2021). Database Concept, Relevance and Expert Systems. Scientific and Educational Areas Under Modern Challenges, 2021. – PP. 125–127. Чебоксары: SCC "Interaktiv plus".
4. Tokhirov E., Aliev R. Improving the braking distance of the train before level crossing // InterConf. – 2020.
5. Бурмистров А.В., Белов Ю.С. Недостатки реляционных баз данных // Электронный журнал: наука, техника и образование. 2015. № 3 (3). – С. 25–34.
6. Драч В.Е., Родионов А.В., Чухраева А.И. Выбор системы управления базами данных для информационной системы промышленного предприятия // Электромагнитные волны и электронные системы. – 2018. – Т. 23. – № 3. – С. 71–80.
7. В.П. Базы данных. Книга 2 распределенные и удаленные базы данных: учебник.// Москва ИД «ФОРУМ» - ИНФРА-М. -2018 . - С 261.
8. Голицына О.Л. Базы данных: учеб. Пособие // - 4-е изд.,
9. иерераб. И доп. —М.: ФОРУМ: ИНФРА-М, 2018. —400 с.
10. Мартишин С.А. Базы данных. Практическое применение СУБД SQL - и NoSQL — типа для проектирования информационных систем: учеб. Пособие // - Москва: ИД «ФОРУМ» - ИНФРА-М, 2 0 19, - 368 с.

Internet saytlari:

1. www.oracle.com
2. www.library.tuit.uz:
3. www.w3schooi.com;

G.B. MURODOVA

MA’LUMOTLAR BAZASINI BOSHQARISH TIZIMLARI

O`quv qo`llanma

Muharrir: *A. Qalandarov*
Texnik muharrir: *N. Samiyeva*
Musahhih: *Sh. Qahhorov*
Sahifalovchi: *M. Bafoyeva*

Nashriyot litsenziyasi AI № 178. 08.12.2010. Original-maketdan bosishga ruxsat etildi: 18.11.2024. Bichimi 60x84. Kegli 16 shponli. «Times New Roman» garn. Ofset bosma usulida bosildi. Ofset bosma qog‘ozi. Bosma tobog‘i 9,2. Adadi 100. Buyurtma №711.

“Sadriddin Salim Buxoriy” MCHJ
“Durdona” nashriyoti: Buxoro shahri Muhammad Iqbol ko‘chasi, 11-uy.
Bahosi kelishilgan narxda.

“Sadriddin Salim Buxoriy” MCHJ bosmaxonasida chop etildi.
Buxoro shahri Muhammad Iqbol ko‘chasi, 11-uy. Tel.: 0(365) 221-26-45

Azim

[illegible]
[illegible] Azim
DN: cn=Azim gn=Azim c=
[illegible]
[illegible]
[illegible]
[illegible]
[illegible] 2024-12-10 12:53+05:00