

NISHANOV A.X., BABADJANOV E.S.,
ERGASHEV A.Q., RISNAZAROV A.M.

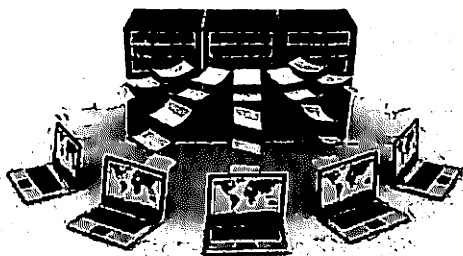
MA'LUMOTLAR BAZASI

(O'QUV QO'LLANMA)



NISHANOV A.X., BABADJANOV E.S.,
ERGASHEV A.Q., RISNAZAROV A.M.

MA'LUMOTLAR BAZASI



O'zbekiston Respublikasi Oliy va o'rta maxsus ta'lim vazirligining
Muvofiqlashtiruvchi kengashi tomonidan o'quv qo'llanma sifatida
tavsiya etilgan

«Mahalla va oila nashriyoti»
Toshkent 2021

UO'K 004.77(075.8)

KBK 32.81я7

M 37

Mualliflar: A.X.Nishanov, E.S.Babadjanov, A.K.Ergashev, A.M.Risnazarov.
«Ma'lumotlar bazasi» fanidan o'quv qo'llanma / Muhammad al-Xorazmiy
nomidagi TATU. 264 b. Toshkent, 2021.

O'quv qo'llanma OTMlarning Kredit tizimida taqsil olayotgan talabalar, mustaqil tadqiqotchi va o'qituvchilar, dasturchilar, axborot tizimlari loyihalashtiruvchi mutaxassislar kaba katta auditoriya o'quvchilari uchun yaratilgan. O'quv qo'llanma tuzilmasi oddiylikdan murakkablik qarata ishlab chichilgan bo'lib, unda nazariy ma'lumotlar amaliy masalalarni yechimi bilan bosqichma-bosqich yechib ko'rsatilgan. Ushbu qo'llanma keyingi yillarda yuqori nufuzli adabiyotlar stiliktikasi, internet ma'lumotlar taxlili natijasida tayyorlangan. Ayniqsa, ma'lumotlar bazasining asosi hisoblanuvchi SQL tilida dasturlash, tranzatsiyalarni boshqarish, triggerdar, funktsiya va protseduralar yaratish, ma'lumotlar bazasi xavfsizli va uni boshqarish amaliy tomondan keng va universal usulda yoritilgan

ISBN 978-9943-7777-5-0

**© A.X.Nishanov, E.S.Babadjanov,
A.K.Ergashev, A.M.Risnazarov.
© «Mahalla va oila nashriyoti»**

Mundarija

1-BOB. MA'LUMOTLAR BAZASI VA UNI TASHKIL ETISH	6
1.1. Kirish. Axborot tizimlar va ma'lumotlar bazasi	6
<i>Axborot tizimlar</i>	6
<i>Ma'lumotlar bazasi tushunchalari</i>	9
<i>MB turkumlanishi</i>	19
1.2. MBBTni fizik tashkil etish va undagi jarayonlar	22
<i>MBBT tashkil etuvchilari</i>	22
<i>Ma'lumotlar bazasi bilan ishlash sxemasi</i>	23
<i>Adreslash usullari</i>	24
1.3. Ma'lumotlar bazasini uch bosqichli arxitekturasi va predmet sohani tavsiflash	26
<i>Ma'lumotlar bazasini uch bosqichli arxitekturasi</i>	26
<i>Predmet sohani tavsiflash usullari</i>	29
2-BOB. MA'LUMOTLAR BAZASINI LOYIHALASH	36
2.1. Normallashtirish. Normal formalar	36
<i>Relyatsion ma'lumotlar bazasini loyihalash maqsadlari</i>	36
<i>Normallashtirish</i>	38
<i>Funksional qaramlilik</i>	40
<i>Munosabatlarning normal formalari</i>	42
<i>Munosabatlarni dekompozitsiyalashga umumiy yondashuv</i>	44
2.2. Ma'lumotlar bazasini kontseptual va datalogik loyihalash	49
<i>Kontseptual loyihalashning zarurligi</i>	49
<i>Ob'ektlarning tavsifi va ularning xususiyatlari</i>	55
<i>Ob'ektlar o'rtasidagi munosabatlarni tavsiflash</i>	58
<i>Murakkab ob'ektlarning tavsifi</i>	62
<i>Datalogik loyihalash</i>	64
<i>ILM asosidagi relyatsion MBni loyihalash usuli</i>	68
<i>ILM asosida relyatsion MBni loyihalashga misol</i>	75
3-BOB. MA'LUMOT MODYELI VA RYELYASION ALGYEBRA 80	
3.1. Ma'lumot modellari	80
<i>Ma'lumot modeli tushunchasi va sinflanishi</i>	80
<i>Ma'lumotlarni ierarxik va tarmoqli modellari</i>	81
<i>Ma'lumotlarni modellashtirish</i>	84
<i>Ma'lumotlar modellariga doir qo'shimcha tushunchalar</i>	85
3.2. Relyatsion ma'lumotlar modeli va relyatsion algebra	89
<i>Tushunchalar</i>	89
<i>Relyatsion algebra va uning amallari</i>	95

4-BOB. SQL TILIDA SO'ROVLAR.....	101
4.1. SQL tili va uning tarkibi.....	101
<i>SQL tili tarixi.....</i>	<i>101</i>
<i>SQL tilining afzalliklari va imkoniyatlari.....</i>	<i>103</i>
<i>SQL turlari yoki rejimlari.....</i>	<i>104</i>
<i>SQL tilining vazifasi.....</i>	<i>106</i>
<i>SQL ma'lumot toifalari.....</i>	<i>109</i>
4.2. DDL – ma'lumotlarni ta'riflash operatorlari.....	111
4.3. DQL sinfi operatori.....	118
4.4. Ko'p jadvalli SQL so'rovlar.....	127
<i>WHERE orqali jadvallarni bog'lash.....</i>	<i>127</i>
<i>JOIN orqali jadvallarni gorizontal birlashtirish.....</i>	<i>129</i>
<i>UNION orqali jadvallarni vertikal birlashtirish.....</i>	<i>134</i>
4.5. DML sinfi operatorlari.....	137
<i>INSERT ma'lumotlarni kiritish operatori.....</i>	<i>137</i>
<i>UPDATE ma'lumotlarni yangilash.....</i>	<i>140</i>
<i>DELETE o'chirish operatori.....</i>	<i>142</i>
4.6. SQL funktsiyalari.....	144
<i>Satrlı funktsiyalar.....</i>	<i>144</i>
<i>Matematik funktsiyalar.....</i>	<i>147</i>
<i>Sanali (date) funktsiyalar.....</i>	<i>148</i>
<i>Tizimli funktsiyalar.....</i>	<i>150</i>
<i>Agregat funktsiyalar.....</i>	<i>150</i>
4.7. SQL da guruhli operatorlar.....	153
<i>GROUP BY - guruhlash operatori.....</i>	<i>153</i>
<i>HAVING – guruhlarni tanlash sharti.....</i>	<i>156</i>
<i>So'rovni qayta ishlash tartibi va tavsifi.....</i>	<i>158</i>
5-BOB. MURAKKAB SQL SO'ROVLAR VA TASAVVURLAR	160
5.1. SQL da murakkab va ichki so'rovlar.....	160
<i>Quyı so'rovlar bo'yicha tushunchalar.....</i>	<i>160</i>
<i>Quyı so'rovlarga talablar va turlari.....</i>	<i>162</i>
<i>Bir qatorli quyı so'rovlar.....</i>	<i>164</i>
<i>Ko'p qatorli quyı so'rovlar.....</i>	<i>169</i>
<i>Korrelıatsion (mutanosib) quyı so'rovlar.....</i>	<i>173</i>
5.2. Tasavvurlar.....	183
<i>Tasavvurlar haqida tushuncha va afzalliklari.....</i>	<i>183</i>
<i>CREATE VIEW – tasavvur yaratish va ularning turlari.....</i>	<i>185</i>
<i>Tasavvurlarni yangilash yoki tasavvurlarda DML amallar.....</i>	<i>190</i>
<i>Tasavvurlarni yangilashda SNeSK OPTION parametri.....</i>	<i>192</i>

6-BOB. SQL TILIDA DASTURLASHTIRISH.....	197
6.1. Tranzaktsiyalar.....	197
<i>Tranzaksiya tushunchalari.....</i>	<i>197</i>
<i>Tranzaksiya modellari.....</i>	<i>198</i>
<i>Tranzatsiyaga misol.....</i>	<i>200</i>
6.2. Saqlanadigan protsedura va funktsiyalar.....	204
<i>Delimiter – ajratuvchi.....</i>	<i>204</i>
<i>Saqlanadigan protseduralar.....</i>	<i>205</i>
<i>Saqlanadigan funktsiyalar.....</i>	<i>212</i>
6.3. Triggerlar.....	217
7-BOB. XAVFSIZLIK VA KIRISH TEXNOLOGIYALARI.....	226
7.1. Ma'lumotlar bazasini ishlatish.....	226
7.2. Ma'lumotlar bazasini ma'murlash va SQL da xavfsizlik.....	228
<i>MBni himoyalash.....</i>	<i>228</i>
<i>Xavfsizlik modellari.....</i>	<i>231</i>
<i>Foydalanishni boshqarish.....</i>	<i>236</i>
<i>Foydalanuvchi, imtiyoz va rollar bilan ishlash operatorlari.....</i>	<i>241</i>
<i>Amaliy masala.....</i>	<i>244</i>
7.3. MBga kirish texnologiyalari.....	248
<i>MBga kirish texnologiyalari haqida.....</i>	<i>248</i>
<i>ODBC texnologiyasi.....</i>	<i>251</i>
<i>JDBC texnologiyasi.....</i>	<i>253</i>
<i>Web dasturlashda MB murojaat.....</i>	<i>257</i>
FOYDALANILGAN ADABIYOTLAR.....	261

1-BOB. MA'LUMOTLAR BAZASI VA UNI TASHKIL ETISH

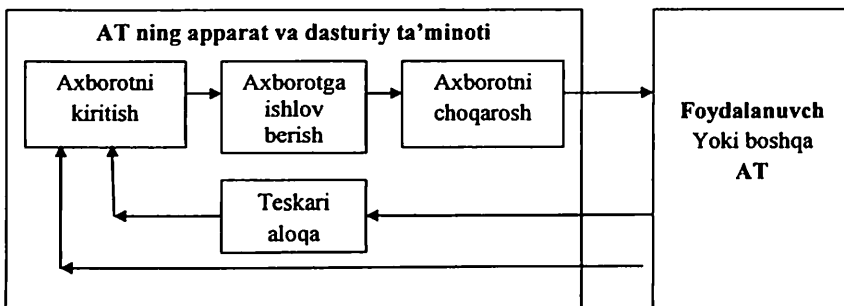
1.1. Kirish. Axborot tizimlar va ma'lumotlar bazasi

Axborot tizimlar

Ko'pgina masalalarni yechish asosida informatsiya yoki axborotlarni qayta ishlash yotadi. Informatsiyani qayta ishlashni yengillashtirish maksadida axborot tizimlar yaratiladi. Avtomatlashgan axborot tizimlar (AT) deb, shunday tizimlarga aytamizki, ularning tarkibida texnik vositalar, jumladan shaxsiy kompyuterlar ishtrok etadi. ATlarni keng ma'noda informatsiyani qayta ishlaydigan ixtiyoriy tizimni tushunish mumkin. Tadbqiq etish sohasiga qarab, ATlar ishlab chikarish sohasida, talim sohasida, sog'liqni saqlash sohasida, xarbiy soqada va boshqa sohalarda ishlatiladigan tizimlarga ajratish mumkin.

- *axborot texnologiyasi* — axborotni to'plash, saqlash, izlash, unga ishlov berish va uni tarqatish uchun foydalaniladigan jami uslublar, qurilmalar, usullar va jarayonlar;
- *axborot tizimi* — axborotni to'plash, saqlash, izlash, unga ishlov berish hamda undan foydalanish imkonini beradigan, tashkiliy jihatdan tartibga solingan jami axborot resurslari, axborot texnologiyalari va aloqa vositalari.

Ixtiyoriy ATizda uning ishlash jarayonini quyidagi tasvirda ko'rsatamiz. Asosiy ishlar 4 ta blokdan iborat (1.1-rasm).

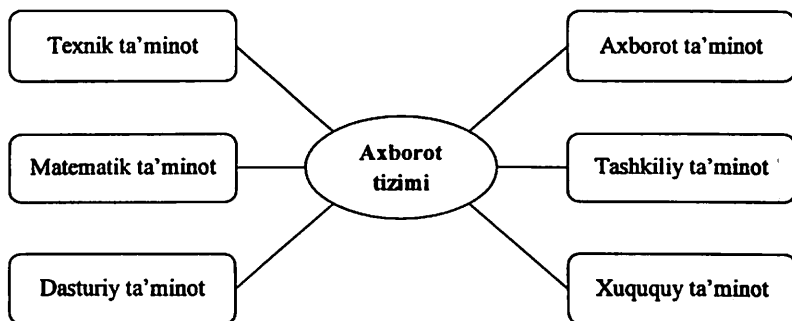


1.1-rasm. Axborot tizimida jarayonlar

- dastlabki ma'lumotni kiritish,
- unga ishlov berish,
- natijaviy ma'lumotni QQ uchun iste'molchilarga yoki boshqa AT ga uzatish,

- teskari aloqani ta'minlash, ya'ni natijani tahlil qilib dastlabki ma'lumotlarni xodimlar yoki boshqa AT tomonidan tuzatish.

Har qanday AT strukturasi, qo'llanish sohasidan qat'iy nazar, bir necha *ta'minlovchi qismlardan* iborat bo'ladi. Ta'minlovchi qismlar 6 xil bo'ladi: texnik, matematik, dasturiy, tashkiliy, huquqiy ta'minotlar.



1.2-rasm. Axborot tizimi va uni ta'minlovchi qism tizimlari

Texnik ta'minot-bu AT ishlab turishi uchun zarur bo'lgan texnik vositalar undan tashqari bu vositalar va texnologik jarayonlar uchun zarur bo'lgan hujjatlardan iborat. Texnik vositalarga: turli modeldagi kompyuterlar; axborotni yig'ish, to'plash, qayta ishlash, uzatish va chiqarish vositalari; ma'lumotlarni uzatish va aloqa vositalari; orgtexnika va axborotni avtomatik o'qish vositalari; turli xil xom ashyo va materiallar kiradi.

Matematik ta'minot - bu AT ishlab turishi uchun zarur bo'lgan matematik usullar, modellardan iborat. Matematik ta'minotga quyidagilar kiradi:

- Boshqruv jarayonlarini modellashtirish vositalari,
- Tipik masalalarni tayyor yechish usul va vositalari (jumladan, matematik analiz, algebra, hisoblash usullari, differentsial va integral tenglamalarga oid tayyor algoritmlar va usullar),
- Matematik dasturlash, matematik statistika, ommaviy xizmat ko'rsatish va h.k.larning usullari.

Dasturiy ta'minot - bu AT ishlab turishi uchun zarur bo'lgan algoritmlar, dasturlardan iborat.

Axborot ta'minot - bu AT ishlab turishi uchun zarur bo'lgan axborotni yagona (unikal) klassifikatsiyalash va kodlash usullari, unikal (yagona, normativ) hujjatlar majmui, hujjatlar oqimining harakat marshrurtlar sxemasi, MB tuzish texnologiyasidan ibordir.

Tashkiliy ta'minot - bu AT ishlab turishi uchun zarur bo'lgan xodimlarning o'zaro va texnik vositalar bilan munosabatlarini aniqlovchi usullar va vositalardan iborat. Tashkiliy ta'minot quyidagilardan iborat:

- AT o'rnatiladigan korxonaning mavjud boshqaruv tizimining tahlili, avtomatlashtirilishi zarur bo'lgan masalalarni aniqlash,
- Masalalarni kompyuterda yechishga tayyorlash. Bunga ATni texnik loyihagini tuzish uchun topshiriq va ATni foydaliligini texnik-iqtisodiy jihatdan asoslash ham kiradi.
- Tashkilotning tarkibi va strukturasi haqida boshqaruv qarorini ishlab chiqish, tashkilot boshqaruv tizimini foydaliligini oshirishga qaratilgan masalalarni yechish metodologiyasini ishlab chiqish.

Huquqiy ta'minot - bu AT ishlab turishi uchun zarur bo'lgan huquqiy normalar bo'lib, AT ni yaratish, faoliyat ko'rsatish, axborotni olish, ishlov berish va foydalanish qoidalarini aniqlab beradi.

AT lar bir necha belgilar bo'yicha *klassifikatsiyalanishi* mumkin:

- 1) echiladigan masalalarning strukturalanishi (formalizatsiya qilinish darajasi) bo'yicha: *strukturalanadigan* (formalizatsiya qilinadigan), *strukturalanmaydigan* (formalizatsiya qilinmaydigan), *qisman strukturalanadigan* (qisman formalizatsiya qilinadigan);
- 2) funktsional belgi va boshqaruv darajalari bo'yicha: ishlab chiqarish, marketing, moliyaviy, kadrlar axborot tizimlari, strategik, funktsional, operativ axborot tizimlari;
- 3) avtomatizatsiya qilinish darajalari bo'yicha: dastaki, avtomatlashgan, avtomatlashtirilgan axborot tizimlari;
- 4) foydalaniladigan axborotni xarakteri bo'yicha: axborot-qidiruv va axborot-echuvchi axborot tizimlari;
- 5) Qo'llanish sohalari bo'yicha: tashkiliy-boshqarish, texnologik jarayonlarni boshqarish, loyihalashni avtomatlashtiruvchi, integrallangan (korporativ), ta'lim, ilmiy ishlarni avtomatlashtiruvchi, geografik ...axborot tizimlari.

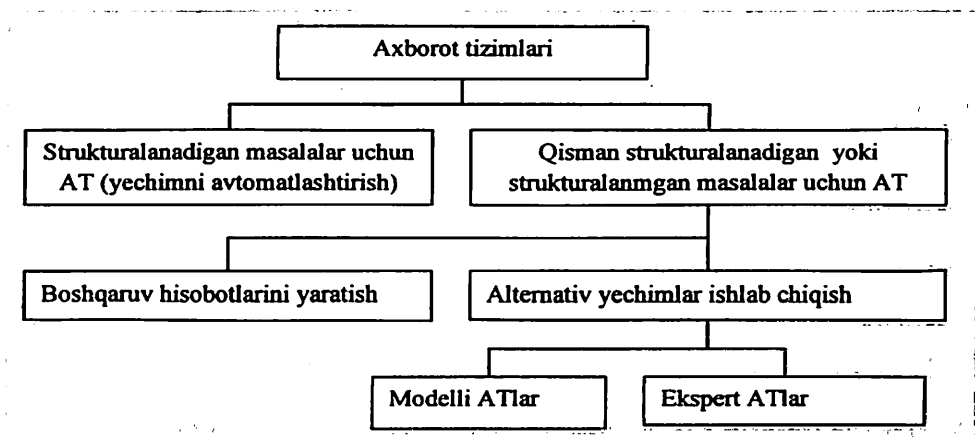
AT larda yechilishi mumkin bo'lgan masalalarni 3 turga bo'lish qabul qilingan: strukturalanadigan (formallashtiriladigan) masalalar, strukturalanmaydigan (formallashtirilmaydigan) masalalar va qisman strukturalanadigan (qisman formallashtiriladigan) masalalar.

Strukturalanadigan (formallashtiriladigan) masalalar-masalaning barcha elementlari orasida matematik bog'lanishlar aniq.

Strukturalanmaydigan (formallashtirilmaydigan) masalalar-masalaning barcha elementlari orasida matematik bog'lanishlar aniqmas.

Qisman strukturalanadigan (qisman formallashtiriladigan) masalalar-masalaning ayrim elementlari orasida matematik bog'lanishlar mavjud xolos.

Strukturalanadigan masalalarda masala matematik modellar orqali bayon etiladi va yechimi uchun aniq algoritim va dastur ishlab chiqish mumkin. Bunday ATlarda ishlash to'la avtomatlashlashgan bo'ladi va xodimning ishtiroki juda past.



1.3-rasm. AT klassifikatsiyasi

Strukturalanmaydigan masalalarda masala uchun matematik modellar tuzish mumkin bo'lmaganligidan, algoritmlar ishlab chiqish juda qiyin. Bu masalalarda ATlar ishlatish imkoniyatlar uncha ko'p emas. Bunday hollarda QQQ inson tomonidan evristik tasavvurlar asosida, tajribaga tayanib amalga oshiriladi. Masalan, guruh talabalari orasidagi munosabatlarni matematik nuqtai nazardan bayon etish mumkin emas.

Qisman strukturalanadigan masalalarda masalani yechish uchun ATlar yaratsa bo'ladi. Ularda yaratiladigan axborot xodimlar tomonidan tahlil qilinadi va QQQ. Bunday ATlar avtomatlashtirilgan deb aytiladi.

Qisman strukturalanadigan masalalarda masalani yechish uchun ATlarni klassifikatsiyalash quyidagi grafikda ko'rsatilgan.

Ma'lumotlar bazasi tushunchalari

Biror masalaga doir ma'lumotlarni to'plashda yoki aniqlashda inson real dunyoni u yoki bu sohasi bilan cheklanadi. Bunday hollarda faqat ba'zibir ob'ektlarni o'rganishgina qiziqish o'yg'otadi. Bunday ob'ektlarni

majmuasini *predmet soha* deyiladi. Shuning uchun ham ma'lum predmet soxadagi ma'lumotlar tuplami MB hisoblanadi.

Ob'ekt – bu ixtiyoriy predmet, xodisa, tushuncha yoki jarayondir.

Ma'lumot – bu uni ma'nosiga e'tibor bermay qaraladigan ixtiyoriy simvollar to'plamidir. O'zaro bog'langan ma'lumotlar *ma'lumotlar tizimi* deyiladi. Barcha ob'ektlar atributlari orkali xarakterlanadi. Masalan, ob'ekt sifatida fakultet, biblioteka, kompyuter va boshqalarni qarash mumkin. Jumladan, kompyuter ob'ektini atributi sifatida hisoblash tezligini, operativ xotira xajmi, o'lchamlari va boshqalarni ko'rish mumkin. Atributlarda saqlanadigan xabarlar *ma'lumotlarni qiymatlari* deyiladi. Masalan, operativ xotira xajmi 128 MB, EHM hisoblash tezligi sekundiga 5 mln.ta amal. Atributning qiymatlari mavjudki, ular yordamida ob'ektlarni identifikatsiyalash mumkin. Bog'langan atributlarni qiymatlarni birlashtirsak *ma'lumot yozuvlarini* hosil qilamiz.

Ma'lumotlarni ifodalashda qo'llaniladigan atamalarning ko'plab variantlari mavjud. Ma'lumotlar bazalari atamashunosligi sohasida CODASIL (ma'lumotlarga ishlov berish tizimlari assotsiatsiyasi) atamalari ko'proq nufuzga ega. Shu nuqtai nazardan, ma'lumotlar bazasi dasturlovchisi tomonidan ma'lumotlarni ifodalashda qo'llaniladigan atamalarni 1.5-rasmdagi kabi tasvirlash mumkin. Ma'lumotlar bazalari bo'yicha boshqa ishchi guruhlari tomonidan ham shunga yaqin atamalar qo'llaniladi.

Исм-шарифи	Оклад	Иш куни	Иш хаки
FIO	O	K _o	S

$$\text{Тўлиқ иш куни} - K_o \quad S = K_o O / K_r$$

Иванов Иван Иванович	1800	24	1800
-------------------------	------	----	------

Петров Петр Петрович	2200	20	1830
-------------------------	------	----	------

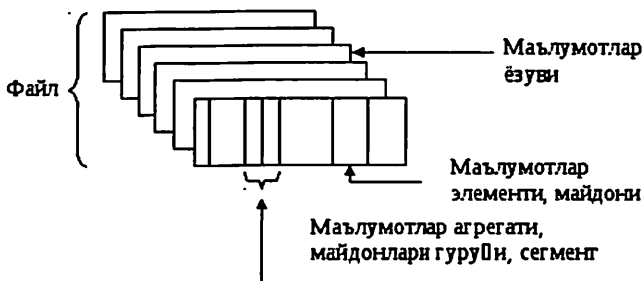
Сидоров Сидор Сидорович	2500	24	2500
----------------------------	------	----	------

1.4-rasm. Ma'lumot yozuvlariga namuna

Ma'lumotlar elementi – ma'lumotlarning eng kichik nomlangan birligi. Bu birlik ko'pincha *maydon* deb aytiladi va *bayt* va *bit*lardan iborat.

Maydonlar MB tuzilmasining asosiy elementlaridir. Ular ma'lum xususiyatlarga ega bo'ladilar. Har qanday maydonning asosiy xususiyati uning uzunligidir. Maydon uzunligi undagi belgilar soni bilan ifodalanadi.

Maydonning yana bir xususiyati, uning nomidir. Maydonda uning nomidan tashqari yana imzo xususiyati ham mavjud. Imzo-ustunning sarlavhasida aks ettiriladigan axborotdir. Uni maydon nomi bilan aralashtirib yubormaslik lozim. Agar imzo berilmagan bo'lsa sarlavhada maydon nomi yozib quyiladi. Turli tipdagi maydonlar turli maqsadlarda ishlatiladi va turli xossalarga ega bo'ladi.



1.5-rasm. Ma'lumotlarni tasvirlash

Ma'lumotlar agregati – ma'lumotlar yozuvi tarkibida bir butun deb qaraluvchi ma'lumotlar elementlarining nomlangan jamlanmasi. Ba'zida bu birlik ma'lumotlar guruhi deb ataladi. Ma'lumotlar agregatining ikki tipi mavjud: *vektorlar* va *qaytariluvchi* guruhlar. Ma'lumotlar elementlarining bir o'lchovli, tartiblangan jamlanmasiga *vektor* deyiladi. «Kun», «Oy» va «Yil» ma'lumotlar elementlaridan tashkil topuvchi «Sana» ma'lumot agregati vektorga misol bo'ladi. Ma'lumotlar yozuvi nusxasida bir necha bor uchraydigan ma'lumotlar jamlanmasiga *qaytariluvchi* guruh deyiladi. Bu tipdagi ma'lumotlar agregatiga misol tariqasida o'quvchining imtixon daftarchasidagi o'zlashtirish ko'rsatkichlarini keltirish mumkin. Qaytariluvchi guruhlar alohida ma'lumotlar elementlari, vektorlar, ma'lumotlar agregatlari yoki boshqa qaytariluvchi guruhlardan tashkil topishi mumkin.

Ma'lumotlar yozuvi – ma'lumotlar elementlari va agregatlarining nomlangan jamlanmasidan iborat. Yozuv umumiy holda agregat bo'lib, u boshqa agregatlarning tarkibiga kirmaydigan tarkibli agregatdan iborat.

Ta'kidlash lozimki, bu yozuvdagi ma'lumotning o'zi (qiymatga ega bo'lgan elementar ma'lumot) faqat yozuvning oxirgi uchlarida ko'rsatilgan, ma'lumotlar tuzilmasining boshqa turlari, shu jumladan agregatlar ham fakat nomlangan ma'lumotlarning to'plami bo'lib hisoblanadi.

Segment – bir, aksariyat hollarda bir necha ma'lumotlar elementlaridan tashkil topadi hamda amaliy dastur va ma'lumotlar bazasi orasida ma'lumot almashishning asosiy birligi (ma'lumot kvanti) hisoblanadi. IBM firmasida (ba'zi boshqa manbalarda ham) ma'lumotlar agregati va yozuvi atamaları o'rniga mazkur atama qo'llaniladi.

Fayl – berilgan tipdagi mantiqiy yozuvlarni barcha nusxalarining nomlangan jamlanmasidan iborat.

Ma'lumotlar bazasi – turli tipdagi ma'lumotlar yozuvlari nusxalari hamda ma'lumotlar yozuvlariaro, ma'lumotlar agregatlariaro, ma'lumotlar elementlariaro munosabatlarning jamlanmasidan iborat. Ma'lumotlar bazasi maxsus tuzilmaga ega va mantiqiy bog'liq axborot elementlarini saqlashga mo'ljallangan ombor hisoblanadi.

ATlarni asosida ma'lumotlar bazasi yotadi. *Ma'lumotlar bazasi* (MB) – bu hisoblash tizimida saqlanuvchi ma'lumotlar massividir. MB shu ma'lumotlarning o'zidan hamda ularning turli masalalarini yechishda barcha joiz hisoblash uchun foydalaniladigan tavsifdan, MB mantiqiy tuzilmasini aks ettiradigan va MBda saqlanadigan ma'lumotlar orasidagi munosabatlarni belgilaydigan ma'lumotnoma axborotdan tashkil topgan.

Ma'lumotlar bazalari tizimi – bu bir necha amaliy dasturlarni ishlashi uchun yaroqli bo'lgan ma'lumotlar bazalarining jamlanmasi. Ba'zida bu atama ma'lumotlar banki atamasi bilan ham qiyoslanadi. Biroq, ma'lumotlar banki nisbatan murakkab tizim bo'lib, u MB, ifodalash vositalari, dastur vositalari, texnik vositalar, tashkiliy-uslubiy vositalar, bank ma'muri, ma'lumotlar lug'ati kabilardan tashkil topadi. Umumiy holda MB maxsus ma'lumotlarni bazasini boshqarish tizimi (MBBT) orqali boshqariladigan o'zaro bog'langan ma'lumotlar to'plamidir. Ya'ni, MB deganda ma'lumotlarni shunday o'zaro bog'langan to'plamini tushunamizki, u mashina xotirasida saqlanib, uni MBBT to'ldirilishi, o'zgartirilishi, takomillashtirilishi mumkin.

MBBT shunday tizimli dasturiy vosita bo'lib, uning yordamida jadvallar tuzilmasini, jadvallar orasidagi bog'lanishni, jadvallarni ma'lumotlar bilan to'ldirish va taxrirlash, ya'ni, MB bilan ishlaydi

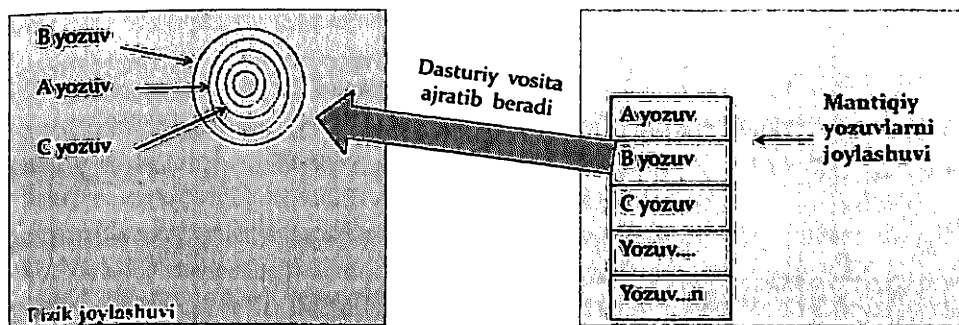
MBBT MBni yaratish, uni dolzarb holatda ushlab turish, kerakli axborotni topishni tashkil etish, foydalanuvchilarni MB bilan ishlashiga

imkon yaratish va boshqa xizmat ko'rsatish uchun zarur bo'ladigan dasturiy va til vositalari majmuasidir.

MBBT misoli sifatida quyidagilarni keltirish mumkin: *DBASE; Microsoft Access; Microsoft For Pro For DOS; Microsoft For Pro For WINDOWS; Paradox for DOS; Paradox for WINDOWS.*

MBBT deb nomlanuvchi alohida dastur majmualaridan XX asrning 60-yillardan boshlab foydalanila boshlangan. MBBT quyidagi vazifalarni bajarishda yaqin ko'makchi hisoblanadi:

- ma'lumotlarning ifodalanishi va fizik joylashuvi;
- ma'lumot qidirish;
- MBni dolzarb holatda saqlab turish;
- ma'lumotlarni nojoiz o'zgartirishlar va yo'l qo'yilmaydigan yetishishdan himoyalash;
- bir vaqtning o'zida bir necha foydalanuvchilar tomonidan bo'ladigan ma'lumotlar so'rovlariga xizmat ko'rsatish.



1.6-rasm. Ma'lumotlar saqlanishi

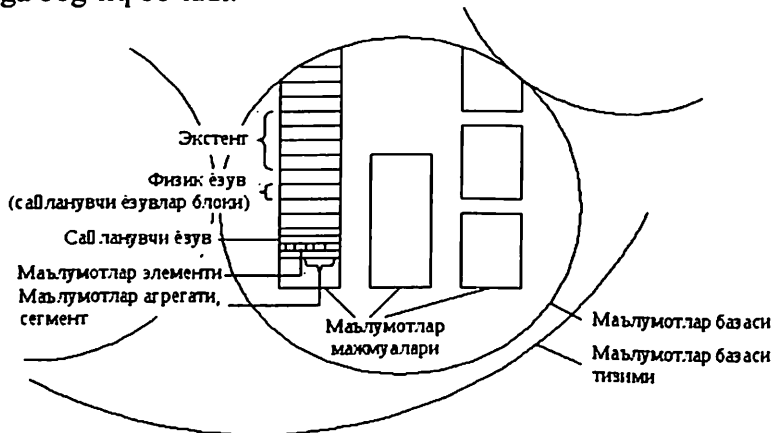
Ma'lumotlar va ular orasidagi munosabatlar ikki xil: *mantiqiy va fizik* ifodalar ko'rinishida bo'ladi. Ma'lumotlarning *fizik ifodasi* ularni kompyuterning tashqi xotirasida saqlash usullarini belgilaydi. Ma'lumotlarning *mantiqiy ifodasi* ma'lumotlarni dasturlovchi yoki dastur foydalanuvchisi tomonidan qanday tasavvur etilishini ko'rsatadi.

Ko'pincha ma'lumotlarni fizik saqlash shakli ularning mantiqiy ifodasidan anchagina farq qiladi. Ma'lumotlarning fizik saqlanishini ifodalash uchun qo'llaniladigan atamalar 1.7-rasm yordamida keltirish mumkin.

Fizik yozuv – kompyuter tashqi xotirasining bir adresi bo'yicha yoziladigan va o'qib olinadigan ma'lumotlardan iborat. Ko'pincha, bir fizik yozuv tarkibida bir necha mantiqiy yozuvlar yoki sigmentlar bo'ladi.

Saqlanuvchi yozuvlar bloki – bu fizik yozuvni tashkil etuvchi ma'lumotlar guruhidan iborat.

Ekstent – tashqi xotira qurilmasida ketma-ket joylashgan fizik yozuvlar jamlanmasidan iborat bo'ladi. Ekstentdagi yozuvlar soni xotira qurilmasining hajmi va foydalanuvchini xotiraning taqsimlanishiga doir so'roviga bog'liq bo'ladi.



1.7-rasm. Ma'lumotlarni fizik saqlanish namunasi

Ma'lumotlar majmuasi – fizik yozuvlarning nomlangan jamlanmasidan tashkil topadi. Unga yozuvlarning joylashuv o'rnini aniqlash uchun qo'llaniladigan ma'lumotlar (masalan, indekslar) ham kiradi. Ma'lumotlar majmuasi bir yoki bir necha ekstentni band qiladi. U bir necha xotira qurilmalariga taqsimlangan bo'lishi ham mumkin.

Yozuvlar uchastkasi – tashqi xotirada saqlanuvchi yozuvlarning guruhi uchun yagona adres qo'llanilganida ushbu yozuvlar sohasiga yozuvlar uchastkasi deyiladi. Fizik yozuv, yozuv yo'lagi, seksiyasi, yozuvlar guruhi yoki yozuvlar uchastkasi sifatida kelishi mumkin.

«Mijoz-server» arxitekturasida MBBT tarmoqdan mijozlarning so'rovlarini qabul qilib oladi va ularga javoblar qaytaradi. Bu xil MBBT MB serveri deyiladi. MB serveridagi asosiy amallardan biri *tranzaktsiya* hisoblanadi, yoxud serverdagi ma'lumotlarning o'zgartirilishi tranzaktsiyalar ko'rinishida ro'y beradi.

Tranzaktsiya so'zi inglizcha ikki so'zning qo'shilishi oqibatida yuzaga kelgan (transformation action – o'zgartirish amali (harakati)). Ya'ni tranzaktsiya – bu MBni bir to'laqonli holatdan ikkinchi to'laqonli holatga o'zgartiruvchi operatsiya yoki operatsiyalar guruhidan iborat. Tranzaktsiya 4 asosiy xususiyat bilan tavsiflanadi:

Bo'linmaslik (atomarlilik xususiyatiga ega bo'lish, Atomicity) – «hammasi yoki hech qaysisi» xususiyati, ya'ni tranzaktsiyaning ma'lum bir qismini bajarish mumkin emas, uni to'raligicha bajarish yoki umuman bajarmaslik zarur bo'ladi. Ushbu xususiyatga quyidagicha aniqlik kiritish mumkin. Masalan, MBda bir hisob raqamidan ikkinchisiga ma'lum miqdordagi pul o'tkazilishi zarur. Mazkur amal ikki operatsiyadan iborat:

- birinchi hisob raqamidan ko'rsatilgan miqdordagi pulni olish;
- ikkinchi hisob raqamiga ko'rsatilgan miqdordagi pulni qo'shib qo'yish.

Agarda faqat birinchi operatsiya bajariladigan bo'lsa, ko'rsatilgan miqdordagi pul butunlay yo'qolib qoladi, ya'ni barcha hisob raqamlaridagi pullar miqdori kamayadi va balans o'zgaradi. Shuning uchun, birinchi operatsiya bajarilgandan so'ng ikkinchi operatsiyani bajarish mumkin emasligi ma'lum bo'lib qolsa, birinchi operatsiya ham inkor qilinadi va MB o'zining dastlabki holatiga qaytariladi.

To'laqonlilik (Consistency) – tranzaktsiya MBni bir to'laqonli holatdan boshqasiga o'tkazishi zarur. To'laqonlilik MB rioya etishi zarur bo'lgan mantiq bilan belgilanadi va uni axborot tizimi ta'minlab beradi.

Ihtotalanganlik (Isolation) – tranzaktsiya serverda o'zi bilan birgalikda bajarilayotgan boshqa tranzaktsiyalardan bexabar bo'lishi (ularni bilmasligi, ko'rmasligi) kerak.

Kuchga kiruvchanlik (Durability) – tranzaktsiya tomonidan bajarilgan o'zgartirishlar albatta kuchga kirishi zarur.

Adabiyotlarda bu xususiyatlar jamlanmasi *ACID*-xususiyatlar deb nomlanadi. Umuman olganda, har bir so'rov alohida bir tranzaktsiya hisoblanadi.

Tranzaktsiya faqat maxsus tasdiqlash amali (commit) bajarilgandan so'ngra amalga oshirilgan hisoblanadi. Bu paytgacha boshqa foydalanuvchilar tranzaktsiya natijalaridan bexabar bo'lishadi. Tasdiqlanmagan tranzaktsiya faol tranzaktsiya deb nomlanadi.

Ma'lumotlar bazalariga oid boshqa tushunchalar, ularni yaratish uchun asos bo'lgan ma'lumotlar modeli va qo'llanilgan boshqaruv tizimi bilan uzviy bog'liq. Bu tushunchalarni to'laqonli tasavvur etish uchun, albatta, ma'lumotlarni modellash hamda ma'lumotlar modellari masalalari bilan atroflicha tanishish zarurati seziladi.

Hozirgi kunda MB tushunchasi bilan "*ma'lumotlar banki*" (MBn) tushunchasi ham keng ishlatilib, u murakkabroq tizim hisoblanadi. Bu yerda MB va MBBT o'z o'rnida MBn tarkibiy qismlari hisoblanadi.

Ma'lumotlar bankini zaruriy axborotni olish maqsadida ma'lumotlardan markazlashgan jamlash va ko'p bo'g'inli jamoa bo'lib foydalanishga mo'ljallangan axborotlashgan, matematik, dasturiy, til vositali, tashkiliy va texnik vositalar tizimi sifatida aniqlash mumkin. Ma'lumotlar bankini bajarilish va joriy etilishini ta'minlovchi dasturlar majmui bilan birgalikda olinsa, uni ma'lumotlarning avtomatlashtirilgan banki (MABn) deb ataladi.

Ma'lumotlarning avtomatlashtirilgan banki (MABn) ma'lumotlarning avtomatlashtirilgan banki bilan ishlovchi xodimlarini, foydalanuvchilarni, MBning ma'muriyatini o'z ichiga oladi, Shuningdek kompyuter, kommunikatsiyalashgan texnologiyalar bazasi asosidagi axborotni ishlash va jo'natish zaruriyatini amalga oshiruvchi axborot jarayonlari texnologiyasini tashkil qiladi.

Ma'lumotlar bankiga quyidagi asosiy talablar qo'yilgan:

1. Axborotlarni predmet soha holatiga mosligi (adekvatligi);
2. Bajarilish ishonchliligi;
3. Tezkorligi va samaradorligi;
4. Foydalanishning soddaligi va qulayligi;
5. Jamoa bo'lib foydalanish;
6. Axborotning himoyalanganligi;
7. Kengaytirish imkoniyati.

Ma'lumotlar bankining tarkibi unga qo'yilgan funktsiyalar va echilayotgan masalalar xususiyatlaridan kelib chiqqan holda aniqlanadi.

Ma'lumotlar bankining asosiy funktsiyalari:

- Axborotlarni saqlash va ularning himoyasini ta'minlash.
- Saqlanayotgan ma'lumotlarning doimiy faollashtirilishi.
- Foydalanuvchilar va amaliy dasturlar so'rovnomasi bo'yicha ma'lumotlarni izlash va tanlash.
- Topilgan ma'lumotlarni qayta ishlash va berilgan formasi ko'rinishida natijalarni chiqarish hisoblanadi.

MBn tarkibiga bir yoki bir nechta MB, MBBT, shuningdek, MBBT tilida tuzilgan amaliy dasturlar to'plami kiradi.

Yuqoridagilarni hisobga olgan holda MBn quyidagi ko'rinishda tasvirlash mumkin:

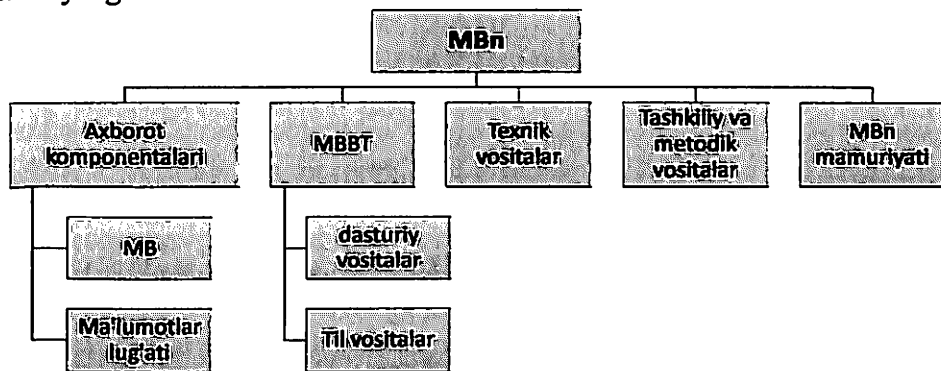
$$MBn = MB, MBBT, \bigcup_{i=1}^n ADT$$

bu yerda, ADT – amaliy dasturiy ta'minotlar, n – ADT son.

MBn komponentalariga markazlashtirilgan yig'ish va ma'lumotlardan ko'p aspektida jamoa bo'lib foydalanish, zarur axborotni olish uchun mo'ljallangan axborot, dasturli, tilli, tashkiliy va texnik vositalar tizimi kiradi (1.8- rasm).

MBn quyidagilardan tuzilgan: MB, MBBT, foydalaniladigan dasturlar majmuasi, foydalaniladigan dasturlarni boshqarish va ularning EHMdan o'tish jarayonini boshqarish tizimlari.

Axborot komponentlari. MBn yadrosini MB tashkil etadi. MBn loyihalaganda barcha foydalanuvchilar uchun axborot massivlari birligi, axborotni bir karra kiritish va undan ko'p maqsadlarda foydalanish tamoyiliga asoslanish zarur.

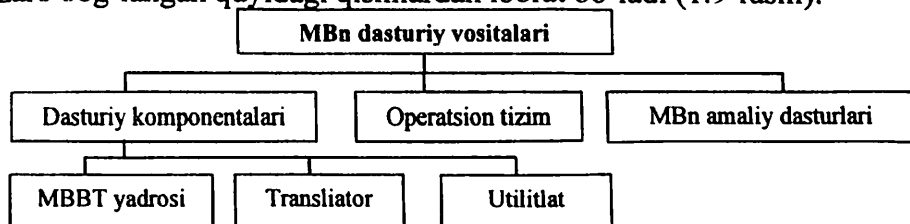


1.8-rasm. Ma'lumotlar banki komponentalari

MBn axborot komponentlari bilan ishlashning turli tartibini ta'minlaydi. MBn bilan ish ko'rilganda muloqot tartibidan keng ko'lamda foydalaniladi. Ma'lumotlarni umumiy bazaga qo'yishning xal qiluvchi faktori ulardan bir necha funktsional qisimiy tizimlarning foydalanishidir.

MB tavsifi *metaaxborot* deb yuritiladi, ya'ni axborot to'g'risidagi axborot. MBni ta'riflash *sxema* deyiladi. Sxemalar quyi sxemalarga ajralishi mumkin. Markazlashtirilgan holda saqlanayotgan metaaxborot *lug'at* yoki *rezpozitariy* deyiladi.

Ma'lumotlar bankining dasturiy ta'minoti murakkab tizim bo'lib, o'zaro bog'langan quyidagi qismlardan iborat bo'ladi (1.9-rasm):



1.9-rasm. Ma'lumotlar bankining dasturiy vositalari

MBn dasturiy ta'minotining eng muhim tarkibiy qismi bu MBBT. MBBT oxirgi foydalanuvchi va MB ma'muriyati ishini yengillashtirish uchun yaratilgan. Bu tizimlar MBni amaliy dasturlardan ajratadi. MBBTT dastur va apparat vositalarining murakkab komponenti bo'lib, foydalanuvchi shu tufayli faqat MBni mantiqiy tashkil etishnigina tasavvur qiladi. Foydalanuvchilarning ixtiyorida talablar tili bo'lib, ular yordamida foydalanuvchilar ma'lumotlarni tanlashi va o'zgartirishi mumkin.

Zamonaviy MBBTlar quyidagi talablarni ta'minlashi lozim:

- ma'lumotlarning mustaqilligi;
- talablarning kuchli tili;
- javobning qisqa vaqti;
- ma'lumotlar va kataloglarni qayta tashkil etishni qisqatirish yoki ulardan voz kechish.

Dastur vositalariga translyatorlar va MBga ma'lumotlarni kiritadigan, qayta ishlaydigan, saqlaydigan, takomillashtiridigan, testdan o'tkazadigan, ma'lumotlarni kiritish chiqarishni ta'minlaydigan boshqarish tizimlari kiradi.

Ma'lumotlar bankining tili vositalari. MBn bilan bog'langan oxirgi foydalanuvchilar turli sohalarda ishlaydilar. Shuning uchun barcha rasmiy talablarni yagona terminologiyaga keltirish uchun ma'lumotlar lug'atidan foydalaniladi.

Zamonaviy MBBTda qo'llanilayotgan til vositalari to'rtinchi avlod tizimlariga kiradi. Umuman, til vositalarining quyidagi avlodlari mavjud:

- *birinchi avlod* – mashina tili;
- *ikkinchi avlod* – assemblarning simbolik tillari;
- *uchinchi avlod* – algoritmik tillar: PL, COLOR, ALGOL va bosh..
- *to'rtinchi avlod* – ob'ektga mo'ljallangan dasturlash til vositalari.

Til vositalari rivojlanishining ikkita kontseptsiyasini farqlash mumkin:

- bo'linish (dekompozitsiya) kontseptsiyasi;
- integrallash kontseptsiyasi.

Bo'linish (dekompozitsiya) kontseptsiyasida quyidagi tillar foydalaniladi: ma'lumotlarni aniqlash tili (MAT) va ma'lumotlar bilan amallar bajarish tili (MABT).

MBBT ni paydo bo'lish tarixida 3 ta til qo'shib ishlatilgan:

1. Ma'lumotlarni tavsiflash tili – MTT (YaOD). Uni yordamida MB jadvallarini strukturalari quriladi.

2. Ma'lumotlar bilan ishlaydigan til – MIT (YaMD). Bu til MBni ma'lumotlar bilan to'ldirish va uni tiklash amallarni (olib tashlash, takomillashtirish va b.) bajarishda ishlatiladi.

3. So'rovlar tili – ST (YaZ). Bu til yordamida qidirish mezonlari asosida kerakli axborotlarni topish va ularni chiqarish uchun xizmat qiladi.

Hozirda barcha aytilgan tillarni vazifasini SQL tili bajaradi.

Texnik vositalar sifatida, asosan, shaxsiy kompyuterlar va super kompyuterlarni ishlatamiz.

Uslubiy – metodik vositalar – bu ko'rsatmalar, metodik va me'yoriy materiallarni majmuasi bo'lib, ular yordamida MB va MBBTdan foydalanish yo'llari ko'rsatiladigan vositalaridir. MBBTdan ikki gurux shaxslari foydalanadi:

MBn administratori deyilganda birorta shaxs yoki bir necha shaxslardan iborat bo'lgan va MB sini loyihalash, uzatish va samarador ishlashini ta'minlovchidir.

1. Chekli yoki oddiy foydalanuvchilar;

2. MB administratori;

MB administratorini xizmat doirasiga quyidagi ishlar kiradi:

a) Predmet sohani tahlili, foydalanuvchilar va axborot o'rnini aniqlash;

b) Ma'lumotlarni tuzilishini loyihalash va ularni takomillashtirish;

c) Qo'yilgan topshiriqlar va ma'lumotlarni bir butunligini ta'minlash;

d) MBni yuklash va yuritish;

e) Ma'lumotlarni himoya qilish;

f) MBni tiklashni ta'minlab berish;

g) MBga murojaatlarni yigish va statistik qayta ishlab berish;

h) MBga ko'p foydalanuvchilar rejimida ishlaganda, ma'lumotlarni o'chib ketishidan ximoya qilish;

i) Texnik vositalar nosoz bo'lib ishdan chiqqanda, ma'lumotlarni saqlash va qayta tiklash ishlarini bajarish;

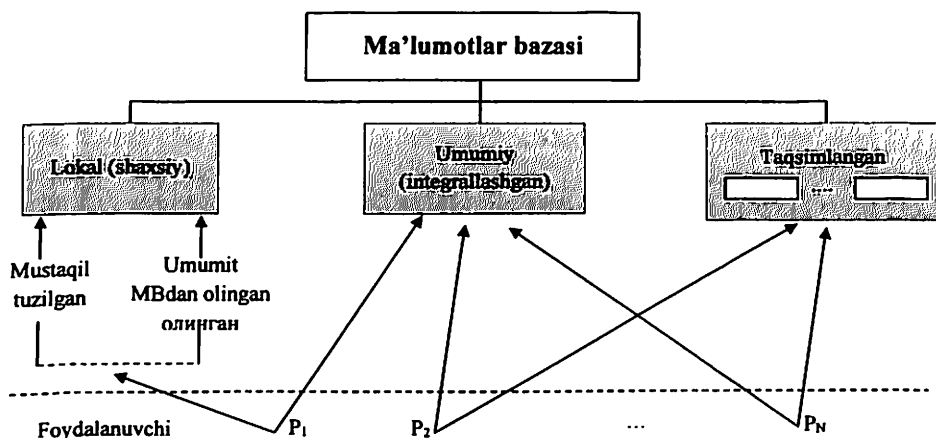
MB turkumlanishi

Ob'ektlarni sinflarga ajratish deyilganda, barcha ob'ektlar to'plamini birorta norasmiy belgi (alovati) bo'yicha qism to'plamlarga ajratishni tushunamiz. MB ni ko'pligini hisobga olib, uni sinflarga ajratish belgilari xilma – xil. Hozirgi kunda MB ni quyidagi sinflari ko'p ishlatiladi:

1. ma'lumotlarni tasvirlash shakliga qarab: *video, audio, multimedia* guruhlariga ajratish mumkin.

2. video ma'lumotlar: *matnli* va *grafik tasvirli* bo'ladi.

3. matnli MB ma'lumotlarni strukturalashganiga qarab *strukturalashgan, qisman strukturalashgan va strukturalashmagan* MB ga bo'linadi.
4. strukturalashgan MB ma'lumotlar modeliga qarab: *ierarxik, tarmoqli, relyatsion, ob'ekli relyatsion, ob'ektga yo'naltirilgan, aralash va ko'p modeli*;
5. vaqtga bog'liq *strategik va dinamik* turlarga ajratiladi.;
6. boshqarilishi jixatdan *markazlashgan va taqsimlangan* MBga bo'linadi.
7. foydalanuvchilar doirasiga qarab: *bitta va ko'p foydalanuvchili*;
8. ma'lumotlarni saqlanishiga qarab: *operatsion va analitik* turlari.;
9. MBda saqlanayotgan ma'lumotlar tipiga ko'ra: *hujjatlashtirilgan, faktografik va leksikografik* (klassifikatorlar, lug'atlar va boshqalar). Bu yerda faktografik MB qandaydir ob'ekt haqida aniq andozali ma'lumotlar (ms: xodimlarning shaxsiy ma'lumotnomasi, fuqarolik pasporti va .x.k.) bo'lsa, hujjatlashtirilgan MBga tabiiy tilda yozilgan ma'lumotlar jamlanmasi (monografiya, roman, ilmiy hisobotlar, me'yoriy-huquqiy hujjatlar) kiradi.
10. iqtisodiy nuqtai nazardan *pulli va pulsiz*;
11. murojaatlar darajasiga qarab: *ommabop va cheklangan*;
12. tashkiliy jihatdan: *maxalliy (shaxsiy), umumiy (integrallashgan va taqsimlangan)* (1.10 – rasm).



1.10-rasm. Tashkiliy jihatdan MBning turkumlanishi

Bundan tashqari MB quyidagilarga ajratiladi: ekstentsional (EMB) va intensional (IMB). *Ektentsional* MB – bu oddiy relyatsion MB.

Intensional MB esa EMB ni mohiyatini aniqlovchi qoidalar asosida quriladi.



1.11-rasm. MBning umumiy turkumlanishi

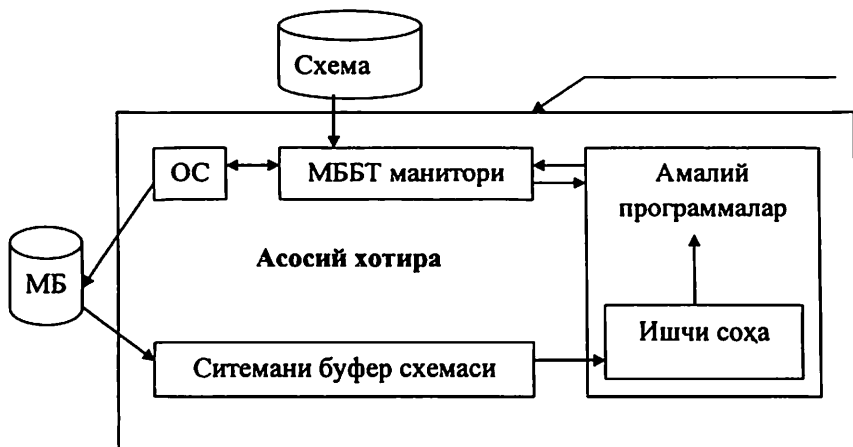
Nazorat savollari

1. Axborot tizimlar va MB o'rtasidagi farqi qanday?
2. Axborot tizimlarining namunaviy ta'minlovchi qismlari vazifasi.
3. MBning asosiy atmalar lug'ati nimalardan iborat?
4. MBBT, ma'lumotlar banki va MB farqi?
5. Ma'lumotlar banki komponentalari vazifasi.
6. MBning asosiy turkumlarini sanang.

1.2. MBBTni fizik tashkil etish va undagi jarayonlar

МБВТ tashkil etuvchilari

МБВТ komponentalari va ularni OS va amaliy programmalar bilan o'zaro bog'liqligi ma'lumotlarni fizik tasvirlashda muhim o'rin tutadi. МБВТ murakkab til programm kompleksidan iborat bo'lib, МБВТni ishlash imkoniyatini ta'minlaydi. МБВТ tarkibiga tizimli prorammlar kompleksi kiradi. Bu kompleksni markaziy komponentasi manitor yoki boshqaruvchi programma hisoblanadi. Manitor МБВТ ni komponentalarini OS va amaliy programmalar bilan o'zaro ta'sirini tashkil qiladi. Bu komponentalarning fizik tashkil etuvchilari quyidagi chizmada berilgan:



1.12-rasm. МБВТ ni komponentalari

Bu chizmada nomerlangan strelkalar bilan amaliy programma tarkibidagi ma'lumotlar bilan ishlash tili (YaMD) ni bitta operatorini bajarishiga tegishli bo'lgan amallar ketma – ketligi ko'rsatilgan.

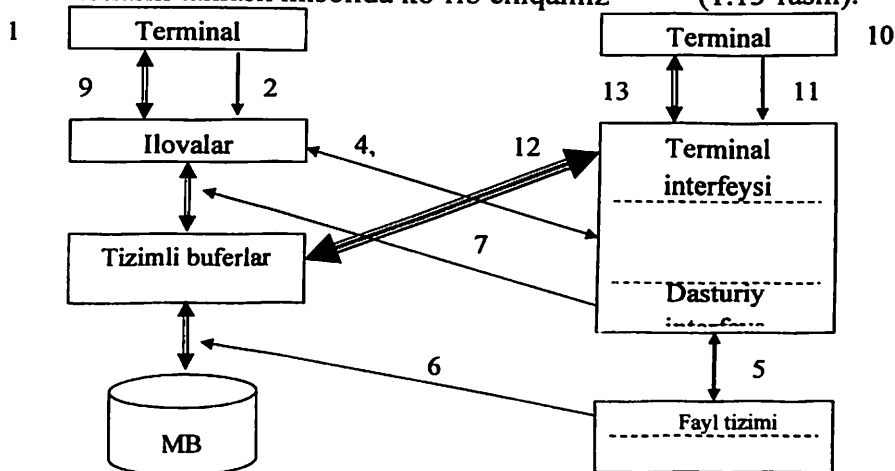
Masalan, bu МБдан ma'lumotlarni o'qish so'rovini operatori bo'lib hizmat qilsin. Unda nomerlangan strelkalar quyidagi ma'noga ega:

- 1) Amaliy programmalar МБга (YaMD) operatori orqali murojaat qilsin. Uni manitor tomonidan tahlil qilinadi.
- 2) Talqin qilish jarayonida manitor oldindan translyatsiya qilib qo'yilgan sxemani ishlatadi.
- 3) Bu so'rovga tegishli ma'lumotlar aniqlanib bo'lingandan keyin, manitor OSga tashqi xotiraga murojaat qilishni amalga oshirish talabi bilan murojaat qiladi.

- 4) OS MB ga murojaatni bajaradi. Bu xuddi fayllarga murojaat qilish kabi oddiy bajariladi.
- 5) Talab qilingan ma'lumotlar tashqi xotiradan tizim buferiga uzatiladi.
- 6) Ma'lumotlar amaliy programmalarini ishchi sohasiga jo'natiladi.
- 7) Manitor amaliy dasturga so'rovni bajarish natijalari xabarini beradi.
- 8) Amaliy programma MB dan olingan ma'lumotlar ustida kerakli amallarni bajaradi.

Ma'lumotlar bazasi bilan ishlash sxemasi

Har qanday foydalanuvchi MB bilan ishlash tartibini batafsil bilishi lozim. MBBT ning samaradorligi munosabatlar ustida munosabatlar algebrasining sakkiz amalini bajarish imkoniyati borligi bilan aniqlanadi. Bu amallar munosabatlarni quyidagilar: qo'shish (birlashmasi), ko'paytirish (kesishmasi), ayirish (to'ldiruvchisi), dekart ko'paytmasi, bo'linmasi, proektsiya, bog'lanish (qo'shilishi), tanlash. Amaliyotda ko'pincha quyidagi to'rt amal foydalaniladi: tanlash, qo'shish, modifikatsiyalash (almashtirish) va olib tashlamoq. Ularni boshqarishni ma'lumotlarni tanlash misolida ko'rib chiqamiz (1.13-rasm).



1.13-rasm. MB bilan ishlashda ma'lumotlarning almashinish sxemasi

MB bilan ishlash jarayonining quyidagi bosqichlarga ajratiladi:

1. Terminal foydalanuvchisi (1) ilova bilan ishlash jarayonida MBga rasmiy talab (2) shakllantiradi.
2. Ilova (3) dasturiy doirasida ma'lumotlar bilan amallar bajarish tili yordamida MBni boshqarish tizimiga rasmiy talab (4) ishlab chiqadi.

3. MBBT o'zining tizimli boshqarish bloklari, jadvallari va ma'lumotlar lug'atini foydalangan holda ma'lumotlar joyini aniqlash maqsadida operatsion tizimga (OT) murojaat qiladi (5).
4. Operatsion tizimning fayl sistemasi yordamida tashqi xotiradan keraqli ma'lumotlar hisoblanib (6) MBBT buferiga joylashtiriladi.
5. Olingan ma'lumotlar mos formatga keltiriladi, ularni MBBT dasturning maxsus blokiga yuboradi (7) va ishlarning oxiriga etkanligi to'g'risida xabar beradi.
6. Ilova (3) MBdan ma'lumotlarni tanlash natijasini foydalanuvchining terminalida (1) aks ettiradi (9).

Agar foydalanuvchi MBni boshqarish tizimi bilan muloqotli (interaktiv) rejimida ishlashida quyidagi bosqichlarni farqlash mumkin:

1. Terminal foydalanuvchisi (10) MBni boshqarish tizimining talab tilida, masalan QBE yordamida MBga rasmiy talab (11) xosil qiladi.
2. MBBT ma'lumotlar joyini aniqlaydi, undan so'ng operatsion tizimga murojaat qiladi (5), ma'lumotlar hisoblanadi (6) va natija MBBT tizimli buferiga joylashtiriladi.
3. Tizimli buferdagi ma'lumotlar tegishli formatga keltiriladi (12) va foydalanuvchi terminalida (10) aks ettiriladi (13).

Adreslash usullari

Bitta mashina ko'rsatmasi yordamida o'qish mumkin bo'lgan bitlar guruxi fizik yozuvlar deb ataladi. Fizik yozuvlar mashina xotirasining yacheykalarida saqlanadi va mashina adreslari yordamida identifikatsiyalanadi. Programmlar mantiqiy yozuvlarni kalitlar yordamida aniqlaydi. Programma uchun zarur bo'lgan ma'lumotni mantiqiy yozuv kalitlari yordamida fizik yozuvlarni adreslarani aniqlaydi. Programma uchun zarur bo'lgan ma'lumotni mantiqiy yozuv kalitlari yordamida fizik yozuvlarni adreslarini aniqlaymiz. Kalit qiymatlari juda ko'p bo'lganligi uchun mashina adreslar bilan munosiblikni aniqlash uchun xilma – xil adreslash usulidan foydalanamiz. Kalit sifatida har bir yozuvda joylashgan piksellangan uzunlikdagi maydonlardan foydalanamiz. Ba'zi hollarda kalit sifatida bir nechta maydon olinadi va bunda ulangan kalitlar hosil qilinadi. Fayllardagi yozuvlarni bir qiymatli aniqlash uchun albatta yagona kalit mavjud bo'lishi kerak va bunday kalitlar birlamchi kalitlar deb ataladi.

Yozuvlarni adreslashning quytsidagi usullari mavjud:

- 1) Fayllarni ketma – ket saqlash usuli. Har bir yozuvni kaliti tekshiriladi. Bunday usul ko'p vaqtni talab etadi.

- 2) Blokli qidirish. Agar yozuvlar kalit bo'yicha tartiblangan bo'lsa, fayllarni skanerlashda har bir yozuvni o'qib chiqish talab etilmaydi. Bunday xollada kerakli yozuvdarni topish uchun blokli qidirish usulidan foydalanamiz. Bunda yozuvlar bloklarga guruxlanadi va har bir blok bir martadan tekshiriladi, kerakli yozuv qidirib topilguncha.
- 3) Binar qidirish. Bunda soha o'rtasidagi yozuv topiladi va uning kaliti qidirish tartibi bilan solishtiriladi. So'ngra qidirish sohasi ikkiga ajratiladi va har bir yarmi alohida qidiriladi. Binar qidirish to'g'ridan – to'g'ri murojaat qurilmalarida ishlatib bo'lmaydi.

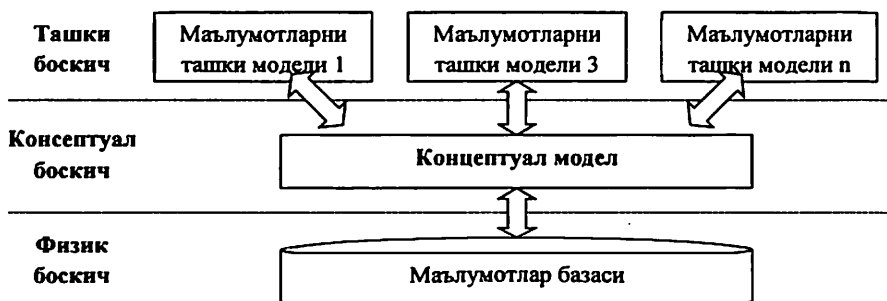
Nazorat savollari

1. MBBT komponentalari va ularning vazifaliri nimalardan iborat?
2. MBda ma'lumot almashishi jarayoni qanday tashkil etiladi?
3. Ma'lumotlar xotirada qanday saqlanadi va ularning turlari?

1.3. Ma'lumotlar bazasini uch bosqichli arxitekturasi va predmet sohani tavsiflash

Ma'lumotlar bazasini uch bosqichli arxitekturasi

MBBTni qanday bo'lishini (qurilishi) o'rganishdagi ilmiy izlanishlar, ularni amalga oshirishni xilma xil usullarini taklif qildi. Bulardan eng hayotiysi, amerika standartlashtirish komiteti ANSI (American National Standards Institute) tomonidan taqdim etilgan MB sini uch bosqichli tashkil qilish bo'lib chiqdi.



1.3.1-rasm. MBning uch bosqichli arxitekturasi

1. *Tashqi modellar* – eng yuqori bosqich, bunda har bir model o'zini ma'lumotlar tasvir qabul qiladi. Har bir ilova, o'ziga kerakli zarur bo'lgan ma'lumotlarni ko'radi va qayta ishlaydi. Masalan, ishchilarni malakasi bo'yicha taqsimlash tizimi, xizmatchi malakasi haqidagi ma'lumotlarni ishlatadi, uni oklad, manzili, telefoni haqidagi axborotlar qiziqitirmaydi va aksincha, oxirgi ma'lumotlar xodimlar bo'limi qism tizimida ishlatiladi.

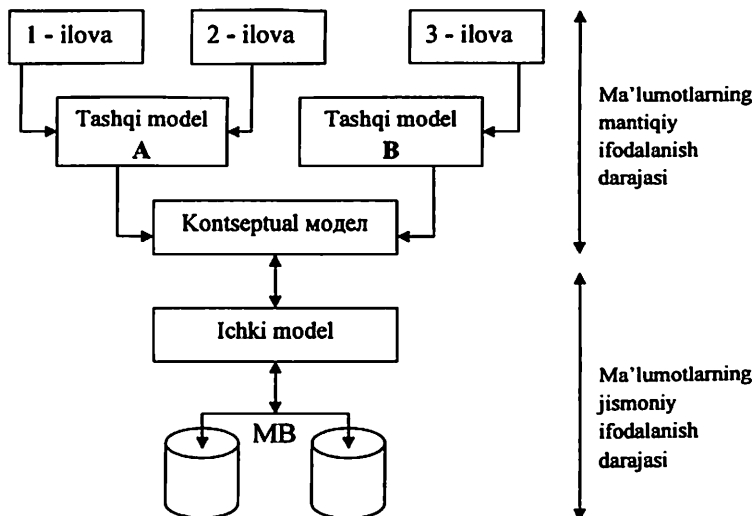
2. *Kontseptual bosqich* – markaziy boshqarish zvenosi bo'lib, bunda MB eng umumiy holda tasvirlanib, u shu MB bilan ishlaydigan barcha ilovalar ishlatiladigan ma'lumotlarni qamrab oladi. Umuman kontseptual bosqich MB yaratilgan predmet sohani umumlashgan modelini akslantiradi. Bu model ob'ektlarning muhim xossalarini akslantiradi.

3. *Fizik bosqich* – fayllarda joylashgan ma'lumotlarni tashqi axborot saqlovchilarida joylashishini belgilaydi. Bu arxitektura ma'lumotlar bilan ishlaganda mantiqiy va fizik mustaqillikni ta'minlab beradi.

Mantiqiy mustaqillik bitta ilovani o'zgartirishni, shu baza bilan ishlaydigan boshqa ilovani o'zgartirmasdan amalga oshirishni bildiradi.

Fizik mustaqiliylik, saqlanuvchi ma'lumotlarni bir qattiq diskdan boshqasiga ko'chirganda uni ishlash qobiliyatini saqlab qolgan holda o'tkazishni bildiradi.

MBni tashuvchilarda hosil qilish bosqichi fizik tuzilishni tashkil etadi. Fizik tuzilishi tashqi xotiralarda ma'lumotlarni joylashtirish usullari va vositalaridan iborat bo'lib, uning ixtiyorida ichki model hosil qilinadi (1.3.2-rasm).



1.3.2- rasm. Ma'lumotlarning MB da ko'ppog'onali ifodalanishi

Ichki model ma'lumotniig mantiqiy modelini tashuvchilarda aks ettiradi va yozuvlarning joylashishi, aloqasi va tanlab olinishini ko'rsatadi.

Ichki model MBBT orqali hosil qilinadi va unga quyidagi talablar qo'yiladi :

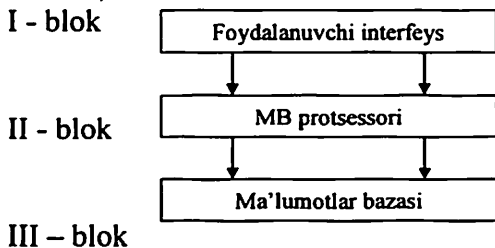
- ma'lumotlarning mantiqiy tuzilishini saqlash;
- tashqi xotiradan maksimal foydalanish;
- MBni yuritish harajatlarini kamaytirish;
- Ma'lumotlarni qidirish va tanlash jarayonlarining tezkorligini oshirish va boshqalar.

Umumiy holda ma'lumotlar bazalarini boshqarish tizimlari (MBBT) ikki xil guruxga ajratiladi:

1. Professional yoki sanoat MBBTlari. Bu guruhga quyidagi MBBTlar kiradi: Oracle, DB2, Sybase, Informix, Ingres, Progress.
2. Shaxsiy (stolga joylashtiriladiganlar). Bu guruhga kiradigan

MBBTlar: FoxBase/FoxPRO, Clipper, R:base, Paladox, Approach va Access.

Hozirgi vaqtda MBBT Accessning ishlab chiqarishda keng tarqalganligini hisobga olgan holda dasturning ma'lumotga kirish arxitekturasini ko'rib chiqamiz. Arxitektura uchta blokdan tashkil topadi (1.3.3 - rasm):



1.3.3 - rasm. Ma'lumotlarga kirish arxitekturasini

1. Foydalanuvchi interfeys bloki. Bunga MS Access ob'ektlari kiradi: jadvallar, formalar, hisobotlar va boshqalar.
2. MB. Bu blokda ma'lumotlar jadvallarining fayllari saqlanadi (Access da mdb - fayllar).
3. MB protsessori. MBBT MS Accessda ma'lumotlar disketlari MS Jet (Joint Engine Technology) MB yadrosining yangi 32 razryadli 3.5 versiyasi qo'llanilgan. Bu versiya yuqoriroq, unumli va yaxshilangan tarmoq tavsiflarga ega.

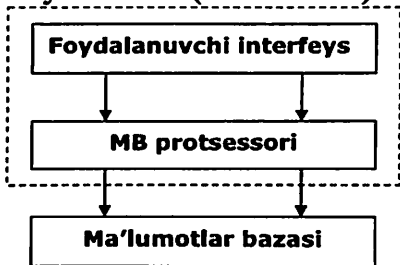
MB bitta kompyuterda joylashishi yoki bir nechta kompyuterda taqsimlanishi mumkin. Bir foydalanuvchining ma'lumotlari boshqalar uchun kira oladigan bo'lishi uchun bu kompyuter hisoblash tarmoqlari yordamida yagona hisoblash tizimiga ulanishi kerak.

Bitta kompyuterda joylashgan MB va banki mahalliy, kompyuterlarning bir necha birlashtirilgan tarmoqlaridagi taqsimlangan deb ataladi.

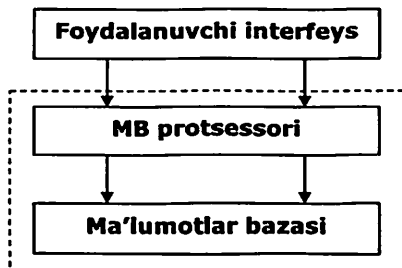
Ma'lumotlarning tarmoqli ishlab chiqarishning turli tamoyillari mavjud: «fayl-server» va «mijoz- server». «Fayl-server» (FS-model) tamoyili tarmoq operatsion tizimi yadrosi hamda markazlangan holda saqlanuvchi fayllar joylashgan va «fayl-server» uchun ajratilgan kompyuterga mo'ljallangan. Boshqa kompyuterlarda esa amaliy dasturlar va MB protsessoring nusxasi joylashtiriladi. So'ralgan ma'lumotlar «Fayl-serverdan» boshqa kompyuterlarga o'tkaziladi va ular MBBT vositalari bilan qayta ishlanadi.

«Mijoz-server» tamoyili bo'yicha ma'lumotlarni ishlab chiqish vazifasi mijoz-kompyuter (dastur) va server-kompyuter (dastur) o'rtasida taqsimlanadi. Ma'lumotlarni qayta ishlashni mijoz so'raydi va u tarmoq bo'ylab MB serveriga uzatiladi. So'rov o'Sha yerda qidiriladi. «Mijoz-server» tamoyili uchun xos xususiyat - bu MBdan so'rov uchun SQL dasturlash tilidan foydalanish (1.3.4 -rasm).

Fayl - server (FS - modeli)



Klient – server

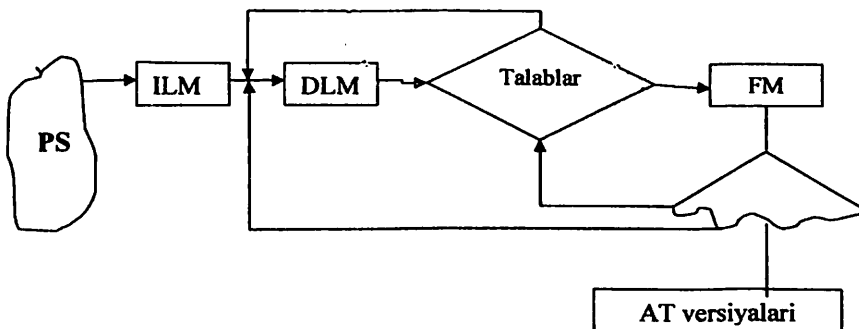


1.3.4 - rasm. «Fayl-server» va «klient-server» texnologiyalari

Predmet sohani tavsiflash usullari

Axborot tizimlarni ishlab chiqishdan asosiy maqsad qisqa vaqt ichida foydalanuvchilar uchun zarur bo'lgan ma'lumotlarni chiqarib berishdan iborat. MB li axborot tizimlarni yaratish tizimli programma ta'minotini yaratishni usullaridan biridir.

Hozirgi kunda axborot tizimlarni loyihalash xilma-xil usullari mavjud. Umuman olganda, axborot tizimni programma ta'minotini yaratish iterativ xarakterga ega. Axborot tizimlarni loyihalashni asosiy bosqichlari va ular orasidagi bog'lanish quyidagi chizmada keltirilgan:



1.3.5-rasm. Namunaviy predmet soha tasvirlanishi

PS – predmet soha; ILM – infologik model; DLM – datalogik model; FM – ma'lumotlarni fizik modeli; AT – axborot tizimi;

AT loyihalashni 1 – bosqichida predmet sohasida mantiqiy axborot tuzilmasini quramiz. U PSni va foydalanuvchini talablarini o'zida mujassamlashtiradi. Bunda biz aniq MBBTga bog'lanmagan ravishda bu ishlarni bajarimiz, ya'ni PSni axborot-mantiqiy tavsifi bajariladi. Bu bosqich infologik model qurish bosqichi deb ataladi. MBBT vositasi yordamida ma'lumotlarni mantiqiy bog'lanishlarini tashkil qilish MBning datalogik modelini bildiradi. Bu model yordamida ma'lumotlar elementlari orasidagi mantiqiy bog'lanishlarni aks ettiradi. DLMni ma'lumotlarni saqlash muhiti bilan bog'laydigan bosqich ma'lumotlarni fizik modeli deyiladi.

Zamonaviy MBBTlari datalogik bosqichda xilma – xil MB bilan ishlashni ta'minlaydi. Hozirgi kunda eng ko'p o'rganilgan va keng ishlatiladigan ma'lumot modellari kiradi. Ma'lumot modeli, bu MBni ma'lumot elementlari to'plami orasidagi bog'lanish strukturalarini tasvirlovchi umumiy sxemadir. Ma'lumot modeli tushunchasini aniq ta'rifini Kodd bergan. U ma'lumot modelini 3 ta kerakli komponentasini keltirgan:

1. Mavjud bo'lgan ma'lumot strukturalarini aniqlash vositalari majmuasi;
2. Ma'lumotlarni qidirish va takomillashtirish uchun MB xolatiga qo'llaniladigan amallar to'plami.
3. Oshkor yoki oshkormas holda MB holatini aniqlovchi va bir butunligini ti'minlovchi vositalar to'plami.

Har bir MB u yoki bu model asosida yoritiladi. Har bir MBBT esa u yoki bu ma'lumot modelini ta'minlaydi deyiladi. Masalan, INYeS ierarxik ma'lumot modeliga asoslangan tizimidir. Tarmoqli modellarda esa – BANK OS, SYeTOR, relyatsion modelga asoslangan tizimlar – Access, KARAT va boshqalar.

Hozirgi kunda PSni tavsiflash uchun ko'p usullar mavjud. Shulardan biri ob'ekt-aloqa usulidir. Bu usulni ba'zan Ulman – Chen usuli ham deyiladi. Ushbu model Peter Chen tomonidan 1976 yili taklif etilgan. «Mohiyat-aloqa» modeli ER-model (inglizchadan – «entity-relationship») deb ham ataladi. U cheklangan sondagi turli jinsli komponentlardan tashkil topuvchi grafik diagrammalardan iborat. P.Chen ma'lumotlar tuzilmalarini mohiyatlar jamlanmasi va ular orasidagi aloqalar to'plami sifatida ifodalashni taklif qilgan. Chen hamda Xammer va Mak-Leodning ishlanmalari ER-model yoki «mohiyat-aloqa» modelining ishlab chiqilishiga asos bo'lgan. ER-modelning asosiy komponentlari mohiyat, aloqa va atributlar hisoblanadi.

Mohiyat – bu xohlagan bir haqiqiy yoki tasavvurdagi ob’ekt bo’lib, MBda mazkur ob’ekt to’g’risida axborot saqlanishi zarur bo’ladi. Odatda, ER-model diagrammalarida mohiyat, ichki qismiga mohiyat nomi yozib qo’yilgan to’g’ri to’rtburchak shaklida ifodalanadi. Mohiyat aniq mazmun kasb etuvchi nomlanishga ega bo’lishi hamda mohiyat nomi ot so’z turkumida, birlik sonda belgilanishi zarur. Mohiyat tipi va mohiyat nusxasi (ekzempelyari) tushunchalarini chalkashtirib yubormaslik zarur. Mohiyat nomi mohiyat nusxasi uchun emas, uning tipi uchun belgilanadi. Mohiyat nusxasi bir jinsli predmetlar, hodisalar, g’oyalar jamlanmasidagi aniq muayyan bir narsani bildiradi.

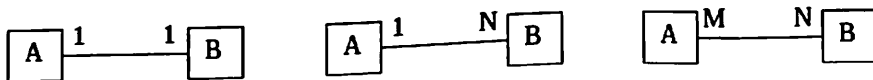
Misol uchun, «O’quvchi» mohiyat tipi rasmda ko’rsatilgani kabi tasvirlanadi. «O’quvchi» mohiyat tipining nomlanishi hisoblanadi, mazkur mohiyat nusxalari sifatida muayyan o’quvchilar – Botirov, Valiev, Po’latov va boshqalar bo’lishi mumkin. Mohiyat nusxasi tushunchasi relyatsion modeldagi kortej tushunchasi bilan taxminan mos keladi. Shu nuqtai nazardan, mohiyatning har bir nusxasi mazkur mohiyatning boshqa xohlagan bir nusxasidan farq qilishi zarur bo’ladi.

Aloqa – bu ikki va undan ortiq mohiyatlarning o’zaro birlashuvi (assotsiatsiyasi)dir. Grafik diagrammalarda aloqa ikki mohiyatni bog’lovchi chiziq sifatida ifodalanadi. Aloqa ko’pincha bir xil bo’lmagan ikki mohiyat orasida mavjud bo’ladi. Biron-bir mohiyatni o’z-o’zi bilan bog’lovchi aloqa ham mavjud bo’lishi mumkin va bu turdagi aloqa rekursiv aloqa deb nomlanadi.

«Mohiyat-alloqa» diagrammalarini ishlab chiqishning bir qancha andozaviy uslubiyatlari mavjud. Bu uslubiyatlar sifatida IDEF1X, IE, DM kabilarni ko’rsatish mumkin va ularning har birida mohiyat va aloqalarni tasvirlash uchun o’ziga xos belgilashlar qabul qilingan.

Umuman olganda, aloqaning har bir tugallanishiga aloqa ko’rsatkichi (1 yoki ko’p so’zini ifodalovchi simvol) qo’yiladi. Quyidagi xillardagi aloqa ko’rsatkichlari mavjud bo’lishi mumkin:

- *Birga-bir*. A mohiyatning har bir nusxasiga V mohiyatning 0 yoki 1 ta nusxasi mos keladi.
- *Birga-ko’p*. A mohiyatning bir vakiliga V mohiyatning 0, 1 yoki bir necha vakili mos keladi.
- *Ko’pga-ko’p*. A mohiyatning har bir nusxasiga V mohiyatning 0 yoki bir necha nusxasi mos keladi hamda V mohiyatning har bir nusxasi uchun ham ushbu vaziyat kuzatiladi.



Aloqalarda aloqa ko'rsatkichidan tashqari xususiyatlar ham mavjud. Masalan, mavjud bo'lish vaqti (doimiy, uzoq davrli va qisqa davrli aloqalar), tanlov (majburiy, nomajburiy, imkoniyatdan kelib chiqadigan va shartli aloqalar).

Atribut (xususiyat) – mohiyat tavsifining nomi. Atribut tuzilmaning elementar birligi bo'lib, u mohiyat holatini aniqlash, qayd etish, tasniflash, raqamli tavsiflash yoki ifodalash uchun xizmat qiladi. Masalan, «O'quvchi» mohiyatining atributlari – kod, familiyasi, ismi, sharifi, manzili, yoshi, jinsi va boshqalar bo'lishi mumkin.

Umumiy holda, mohiyat atributlari soni cheksiz bo'lishi mumkin. Axborot tizimlarini ishlab chiqishda atributlar soni va tarkibi foydalanuvchi talablari hamda amalga oshirilishi lozim bo'lgan masalalardan kelib chiqib belgilanadi. Odatda, infologik modelni qurishda PSga doir tushunchalarning xususiyatlarini nisbatan kengroq va to'laroq ifodalashga harakat qilinadi. Bunda mavjud vazifalarni bajarish imkoniyatini hisobga olishdan tashqari istiqbolda yuzaga kelishi mumkin bo'lgan masalalarga ham e'tibor qaratiladi.

ER-diagrammalarda atributlar qo'llanilayotgan modelga bog'liq ravishda turlicha tasvirlanadi. Masalan, atributlar nomlari mohiyatni tasvirleydigan to'g'ri to'rtburchakning ichki qismida, mohiyat nomi keltirilgan satrdan keyingi satrlarda, kichik harflarda keltirilishi mumkin. Boshqa hollarda atributlar mohiyatni tasvirlovchi to'g'ri to'rtburchakdan tashqarida, unga chiziqlar bilan bog'langan tarzda ifodalanishi mumkin. Qoidaga ko'ra, kalit toifasiga ega atributlar ajratib ko'rsatiladi. Ba'zi hollarda atributlar oval ichiga joylashtiriladi.

ER-diagrammalarning nisbatan murakkabroq elementlari qatoriga mohiyatlarning podtiplari kiradi. Mohiyat bir-birlarini o'zaro inkor etuvchi ikki va undan ortiq podtiplarga tarmoqlanishi mumkin. Ushbu podtiplarning har birida ular uchun umumiy bo'lgan atributlar va aloqalar mavjud bo'ladi. Mazkur umumiy atributlar yoki aloqalar nisbatan yuqoriroq darajada belgilanadi, podtiplarda esa faqat ularning o'zlarigagina tegishli bo'lgan atributlar va aloqalar ko'rsatiladi. Mohiyatlarni podtiplarga ajratish quyi darajalarga qarab xohlagancha davom ettirilishi mumkin. Biroq, tajribalarning ko'rsatishicha, ko'pgina hollarda ikki yoki uchta darajadan foydalanish yetarli bo'ladi.

Podtiplarni aniqlashda asos bo'lib xizmat qiladigan mohiyat supertip deb ataladi. Podtiplar birgalikda to'la to'plamni tashkil etishlari zarur, ya'ni supertipning xohlagan bir nusxasi biror-bir podtipga tegishli bo'lishi lozim. Masalan, «Jihoz» mohiyati uchun «Monitor», «Tizim bloki» va boshqa podtiplarni belgilash mumkin.

ER-diagrammalarni ishlab chiqishda universal munosabatni qurish, uni bo'laklarga ajratish, ikkinchi va uchinchi me'yoriy shakllarga keltirish zarurati bo'lmaydi.¹ Mohiyatlarni to'laqonli aniqlashning o'zi – bu universal munosabatni uchinchi me'yoriy shaklga keltirishdan iborat.

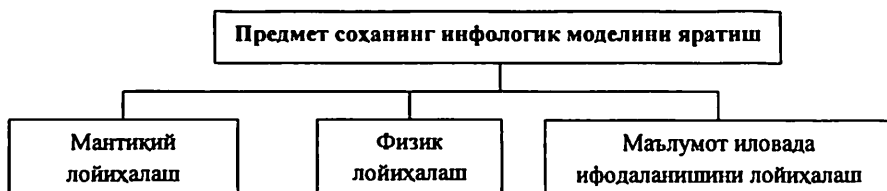
PSni mohiyat aloqa usulida tavsiflaganda quyidagi bosqichlarda ish olib boriladi:

1. PS ni ob'ektlari aniqlanadi.
2. Ob'ekt sohalari (atributlari) belgilanadi va uning kalit parametri aniqlanadi. Kalit parametri ob'ektni identifikatsiyalaydi.
3. Ob'ektlar o'rtasida aloqa o'rnatiladi va ular sinflarga ajratiladi.
4. Maxsus belgilar kiritilib, ob'ekt aloqa diagrammasi o'rnatiladi. Bu diagramma PSning infologik modeli grafik tasviri hisoblanadi.

Ob'ektlarni identifikatsiyalash uchun yoki kompyuter xotirasida yozuvlarning o'rnini aniqlashda ma'lumot elementi ishlatiladi. Bu elementni biz kalit deb ataymiz. Agar kalit ob'ektni bir qiymatli identifikatsiyalasa, u *birlamchi kalit*, aks holda *ikkilamchi kalit* deyiladi. Agar ob'ektlarni identifikatsiyalash uchun bir nechta atributlar talab etilsa, bunday kalit *ulangan kalit* deb ataladi. Agar A va V guruxdagi ob'ektlar berilgan bo'lsa, u holda yuqoridagidek munosabatlar o'rnatiladi.

MBlarini ishlab chiqishda 2 xil usuldan foydalanish mumkin. Birinchi usulda, avval asosiy masalalar aniqlanib, ularni hal qilish uchun baza yaratiladi hamda masalaning ma'lumotlarga bo'lgan ehtiyoji aniqlanadi. Ikkinchi usulda, predmet sohasining namunaviy (tipik) ob'ektlari birdaniga o'rnatiladi. Bu yerda eng optimal usul - ikkala usulni birgalikda ishlatishdir. Bu dastlabki bosqichda barcha masalalar to'g'risida yetarlicha ma'lumotlarning yo'qligi bilan bog'liq.

MB loyihalashtirish jarayoni ikki bosqichga bulinadi: predmet sohasining ma'lumotlarining axborot-mantiqiy modelini (MAMM) ishlab chiqish va MBning mantaqiy tuzilishini aniqlash. Axborot - mantiqiy model predmet sohasini axborot ob'ektlarining majmui va ular orasidagi tuzilmaviy aloqalar ko'rinishida aks etgiruvchi ma'lumotlar modelidir. Predmet sohasi ma'lumotlarini axborot-mantiqli modelini ishlab chiqish muammo sohasini tekshirish natijasida olingan tavsifga asoslanadi (1.3.6-rasm).



1.3.6- rasm. MB loyihalash bosqichlari

Muammo sohasining infologik modeli asosida kontseptual (mantiqiy), ichki (jismoniy) va tashqi modellar tuziladi.

MBning mantiqiy tuzilishi - ob'ektga tegishli bo'lgan axborotlarning MBda joylanishini ifodalaydi. Hosil bo'lgan MBning mantiqiy bog'lanish modeli ikkinchi bosqichining natijasi hisoblanadi. Bu modelda uch turli axborot ifodalanadi: ob'ekt to'g'risidagi xabarlar, ularning xususiyati va o'zaro munosabatlari. Har bir ob'ekt modeli yozuv turlari orqali ko'rsatiladi. Ularning xususiyatlari — yozuv maydonlari orqali ifodalanadi, munosabatlar esa yozuv va maydon turlari o'rtasidagi aloqalar yordamida tasvirlanadi. Bunday model EHM operatsion tizimining, MBBTning mohiyatiga bog'liq, bo'lmaydi, ya'ni axborotning ma'nosiga bog'liq bo'lmagan holda ularni ifodalash usuli va aloqasini ta'minlaydi.

Mantiqiy modelni rasmi va jadvalli usullar yordamida ifodalash mumkin. Rasmi usulda ma'lumotlar o'rtasidagi bog'lanish grafiklar yordamida tasvirlanadi. Bunda grafikning uchlari yozuvlarni ifodalaydi, qirralari esa yozuvlar o'rtasidagi aloqalarni ko'rsatadi. Jadvalli usulda ob'ekt to'g'risidaga ma'lumotlar bir yoki bir nechta ustundan iborat bo'lgan jadvallar orqali ifodalanadi.

Hozirgi vaqtda mantiqiy modellarning pog'onali (ierarxik), tarmoqli va relyatsion turlaridan foydalanilmoqda.

Pog'onali model rasmi usul asosida tashkil qilinadi. Bunda ma'lumot yozuvlari grafikning uchlarini ifodalaydi va har bir yozuv oldingi pog'ona uchlariga bog'langan bo'ladi. Bunday tuzilishdagi MBdan tegishli axborotlar hamma vaqt bitta yo'nalish bo'yicha qidiriladi va uning joylashgan o'rni to'liq ko'rsatiladi. Pog'onali (ierarxik) modelga asoslangan MB 1- va 2- avlod EHMlari yordamida ishlab chiqilgan. IBM firmasi 1968 yilda IMS (Information Management System) deb nomlangan MBni tashkil qilgan.

Tarmoqli model ham rasmi usul yordamida tashkil qilinadi. Lekin bunda tegishli axborotlar bir nechta yo'nalish bo'yicha olinishi mumkin. Tarmoqli model ierarxik modelning kengayishi hisoblanadi. Bu modelning

asoschisi - Ch. Baxman. Tarmoqli modelga asoslangan MB - Integrated Database Menegement System (IDMS) Cullinet Software Inc. Kompaniyasi tomonidan 70-yillarda ishlab chiqilgan. Ierarxik va pog'onali MBning afzalligi - ularning tezkorligi.

Shaxsiy EHMLarning paydo bo'lishi relyatsion modellarning keng tarqalishiga sababchi bo'ldi. Relyatsion model jadvalli usul asosida tashkil qilinadi. Bunda tegishli ma'lumotlar jadvalning ustun va qatorlarida joylashadi. Ustunlar ma'lumotning maydonlarini, qatorlar esa yozuvlarni ifodalaydi. Bir ustunda ma'lum sohaga tegishli bo'lgan bir qancha ma'lumotlar ko'rsatiladi. Ustun va qator o'rtasidagi bog'lanish *munosabat* deb ataladi. Har bir ustun, qator va munocabat o'z nomiga ega bo'ladi. Relyatsion modeldagi munosabatlar ushbu talablar orqali hosil qilinadi:

- ustun va qator kesishgan yerda joylashgan ma'lumotlar element hisoblanadi;
- munosabatlarda ikkita bir xil qator bo'lmaydi;
- ustun va qatorlar tartibli joylashishi va nomlanishi majburiy emas.

Relyatsion model bir nechta munosabatlardan tashkil topishi mumkin. Relyatsion modelning asoschisi Amerika olimi E.F. Kodd. Bu modelning ikkinchi nomi - Kodd modeli.

Nazorat savollari

1. MBning necha turli arxitekturalari mavjud?
2. MBni uch bosqichli arxitektura bosqichlarini tushuntirib bering?
3. «Fayl-server» va «mijoz- server» arxitekturalari farqi nimadi?
4. Ma'lumot modelining zarur komponentalari nimalar?
5. Ob'ektlar o'rtasida «mohiyat-aloqa» diagrammasida asosan necha turli bog'lanish mavjud?
6. Predmet sohani mohiyat aloqa usulida tavsiflash bosqichlar?
7. Mohiyat, aloqa va atribut atamalarini tushuntiring?

2-BOB. MA'LUMOTLAR BAZASINI LOYIHALASH

2.1. Normallashtirish. Normal formalar

Relyatsion ma'lumotlar bazasini loyihalash maqsadlari

Loyihalashning ko'plab maqsadlari ichida quyidagilar asosiylari hisoblanadi[5]:

1. barcha kerakli ma'lumotlarni saqlash mumkinchiligi;
 2. ma'lumotlarning ortiqchaligini yo'qotish;
 3. MBda saqlanuvchi munosabatlarning sonini minimumlashtirish;
 4. ma'lumotlarni yangilashga va o'chirishga bog'liq muammolarni yechigni osonlashtirish maqsadida munosabatlarni normallashtirish.
- Endi ushbu keltirilgan har bir maqsad alohia o'rganiladi.

MBda barcha kerakli ma'lumotlarni saqlash mumkinchiligi. MB yechilayotgan masalaga ta'sirini ko'rsatadigan barcha ma'lumotlarni saqlashi lozim. Loyihalash jarayonining dastlabki qadami MBga joylashtirilishi lozim bo'lgan barcha atributlarni aniqlashtirib olish bo'lib sanaladi. Loyihalashtirilivchi atributlar aniqlashtirilib olinganidan so'ng, qancha munosabatlar kerakligi va qaysi atributlar qaysi munosabatlarga kiritilishi yechiladi.

Ma'lumotlarning ortiqchaligini yo'qotish. Bu maqsadning ma'nosi, agar «ma'lumotlarning takrorlanuvchi» va «ma'lumotlarning ortiqcha takrorlanuvchi» orasidagi farqni aniqlancha tushunarliroq bo'ladi. Misol uchun 4.1-rasmda keltirilgan munosabatlarni ko'rib o'taylik. Bunda A variantda O-KB munosabati ON (o'qituvchi nomeri) va KBF (kafedra boshlig'i familiyasi) bo'lgan ikki atributdan iborat. Munosabatta, o'quv o'rnida har-bir o'qituvchining to'g'ridan-to'g'ri boshlig'ini ko'rsatuvchi ma'lumotlar saqlanadi.

O-KB	
ON	KBF
25	Uteuliev
55	Babajanov
56	Babajanov
75	Uteuliev

A

O-KB	
ON	KBF
25	Uteuliev
55	Babajanov
56	—
75	—

B

2.1.1-rasm. Ortiqcha bo'lmagan takrorlanuvchi ma'lumotlar

Rasmdan ko'rinib turganidek kafedra boshliqlari familiyalar munosabatida bir-necha marta takrorlanishi mumkin. Lekin bu munosabatda familiyaning takrorlanishi ortiqcha bo'lib sanalmaydi. Aksincha qaytarilishlar bartaraf etiladigan bo'lsa (4.1. b), u holda natijada

axborot yo'qotiladi (56 va 75 nomerli o'qituvchilarning kafedra boshliqlari aniq bo'lmay qoladi).

4.2 rasmda ma'lumotlarning ortiqcha takrorlanishi va ortiqchaligini yo'qotishga misol keltirilgan. O-KB-T munosabati yuqoridagi O-KB munosabatiga o'xshash bo'lib, qo'shimcha kafedra boshlig'ining telefon nomerini ko'rsatuvchi qo'shimcha TN atributigi ega.

O-KB-T			O-KB-T		
ON	KBF	TN	ON	KBF	TN
25	Uteuliev	1350391	25	Uteuliev	1350391
55	Babajanov	7270727	55	Babajanov	7270727
56	Babajanov	7270727	56	Babajanov	—
75	Uteuliev	1350391	75	Uteuliev	—

A

O-KB	
ON	KBF
25	Uteuliev
55	Babajanov
56	Babajanov
75	Uteuliev

b

KB-T	
KBF	TN
Uteuliev	1350391
Babajanov	7270727

v

2.1.2-rasm. Ortiqcha takrorlanuvchi ma'lumotlar va ularni yo'qotish

Keltirilgan dastlabki munosabatda Uteuliev va Babajanovlarning telefon nomerlari bir martadan ko'p qaytarilib, bu telefon nomerlari haqida takrorlanuvchi axborot ortiqcha bo'lib sanaladi. Ortiqchalikning sababi shunda, agar Uteulievning telefoni yozilgan bir axborotni o'chirib tashlasak, bu axborotni munosabatning boshqa kortejidan olishga bo'ladi. 4.2,b-rasmdan Uteuliev va Babajanovlarning telefon nomerlari keyingi ikki yozuv olib tashlangan bilan yo'qolmadi, sababi ularni munosabatning boshqa kortejidan topishga bo'ladi.

Ortiqchalikni bunday turda yo'qotishning xato ekanligini ikki sabab bilan ko'rsatishga bo'ladi. Birinchidan MBda bo'sh maydonlar bo'lishi dasturlashtirishda bo'sh maydonlarning qiymatlarini aniqlovchi qo'shimcha algoritmlar foydalanishga olib keladi. Ikkinchidan 4.2, b-rasmda keltirilgan munosabat axborotlarni o'chirish aniq muammoga olib keladigan strukturaga ega. Agarda ON=25 nomerli o'qituvchi kafedradan ishdan chiqib ketsa, unda <25, Uteuliev, 1350391> kortegi munosabatdan o'chiriladi. Natijada Uteulievning telefon nomeri haqida axborot yo'qotiladi. Sababi munosabatda bu axborot boshqa yozuvlarda ko'rsatilmagan.

Telefon nomerlari bo'yicha axborot ortiqchaliklarini yo'qotish uchun O-KB-T munosabatni 4.2,v-rasmda ko'rsatilgandek O-KB va KB-T

munosabatlariga ajratiladi. Bu usulda ON=25 nomerli o'qituvchi O-KB munosabatdan o'chirilgani bilan, Uteulievga tegishli telefon nomeri haqidagi axborot yo'qolmaydi.

MBda saqlanuvchi munosabatlarning sonini minimumlashtirish. Muammolarni yechishda bir munosabatni ikki va undan ko'p kichik munosabatlarga ajratish kerak bo'lishi bilan birga, bu MB foydalanuvchisiga noqulaylik tug'dirishi mumkin. Shuning uchun munosabatlar sonining cheksiz o'sishiga yo'l qo'ymaslik lozim.

Munosabatlarni normallashtirish. Munosabatlarning ko'pchiligida kortejlarni o'chirish va yangilash muammolari juda zarur hisoblanadi. Bunday xavfga ega munosabatlarni topish va ularni yuqorida ko'rsatilgan usuldagidek normallashtirish kerak bo'ladi.

Normallashtirish

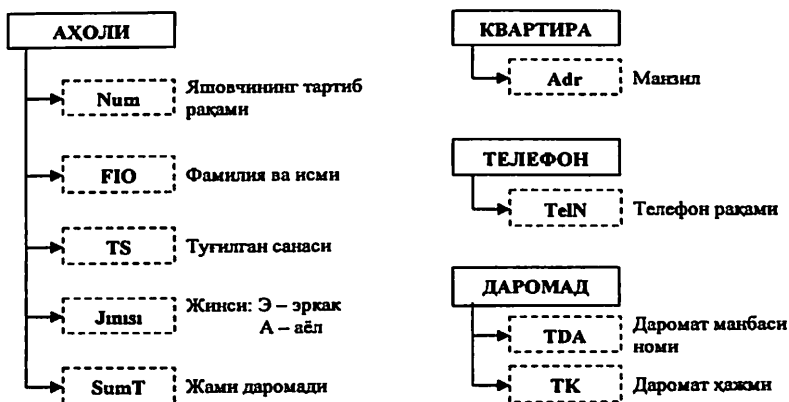
Normallashtirish – bu munosabatni ma'lumotlar kiritishda, yangilashda va o'chirishda qulayliklar tug'diruvchi, ikki yoki unan ko'p munosabatlarga bo'lib chiqish bo'lib hisoblanadi. Normallashtirishning maqsadi har-bir dalil (fakt) faqat bitta joyda bo'ladigan, ya'ni axborot ortiqchaligi yo'qoltilgan MBni olish bo'lib sanaladi. Bu faqat xotirani tejash maqsadida ishlanmasdan, balki saqlanayotgan ma'lumotlarda qarama-qarshiliklarni bo'ldirmaslikga xizmat qiladi.

Relyatsion MBni loyihalash dekompozitsiyalash (bo'lish) yo'li bilan ham amalga oshirilib, bunda MBga kiruvchi dastlabki munosabatlar tshplami, ushbu munosabatlarning sonidan ko'proq va ularning proektsiyalari bo'lgan munosabatlar to'plamiga almashtiriladi. Dekompozitsiyalashni foydalanib loyihalash jarayoni munosabatlarni ketaket normallashtirishdan ibarat bo'lib, bunda navbatdagi munosabatlar to'plami oldingiga taqqoslaganda yaxshiroq ko'rinishga ega va yuqori tartibdagi normal formaga mos bo'ladi. Relyatsion MB nazariyasida ushbu normal formalar alohida ko'rsatib o'tiladi: birinchi normal forma (1NF), ikkinchi normal forma (2NF), uchunchi normal forma (3NF), Boys-Kodd normal forması (BKNF), to'rtinshi normal forma (4NF), beshinchi normal forma (5NF).

loyihalashtirilayotgan MBning har-bir korteji bo'linmaydigan va bo'sh emas yoki atomar qiymatga ega atributlardan turuvchi, barcha kerakli atributlar va kerakli strukturani o'zida saqlovchi munosabat *universal munosabat* deb atash qabul qilingan. Universal munosabat *birinchi normal formaga* mos keladi.

Universal munosabatni foydalanishda uch specifik muammoduch kelinadi: MBga yangi kortejlar qo'shishning qiyinlashishi; MBdagi ma'lumotlarni yangilashga bog'liq muammolar; MBdan kortejlarni o'chirish kerak bo'lgandagi muammolar. Bu muammolar kiritishda, yangilashda va o'chirishda anomallik deb ataladi. Anomallik deganda normalardan chiqish tushuniladi.

Bu muammolar ma'nosini aholi, ularning daromadlari va telefon nomerlari haqidagi ma'lumotlarni saqlovchi MB misolida qarab o'tiladi. Bu MB uchun T-TN-T munosabatni (4.1-jadval), 4.3-rasmda ko'rsatilgan ob'ektning atributlari qiymatlaridan iborat kortejlarni saqlaydi.



2.1.3-rasm. Predmet soha ob'ektlari va ularning atributlari

.4-jadval

Universal munosabat (T-TN-T bir jadvalli MB)

Num	FIO	TS	Jinsi	Adr	TelN	TDA	TK	SumT
1	Risnazarov A.	25.03.75	E	8/11-12	6-35-32	Jumis1	3500	4000
1	Risnazarov A.	25.03.75	E	8/11-12	6-35-32	Jumis2	500	4000
2	Mambetbaeva Q.	07.04.98	H	05-321	Joq	Stipendiya	425	925
2	Mambetbaeva Q.	07.04.98	H	05-321	Joq	Jumis2	500	925
Z	Matirzaeva M.	24.12.78	H	8/11-12	6-35-32	Jumis4	400	1400
Z	Matirzaeva M.	24.12.78	H	8/11-12	6-35-32	Jumis3	1000	1400
4	Xojanova S.	25.07.86	H	1d-7	3-43-00	Jumis3	1000	1000
5	Salilaeva Z.	15.06.46	H	8/11-12	6-35-32	pensiya	1200	1200

Kiritishdagi anomaliya. Agar yangidan daromad manbasiga ega emas (misol uchun tuhilgan farzand) yashovchini T-TN-T munosabatiga kiritish kerak bo'lsa, unda kiritiladigan qatorda TDA va TK maydonlarining qiymatlarini bo'sh tashlab ketishga to'g'ri keladi. Lekin munosabatda (jadvalda) bo'sh (aniq emas) maydonlar bo'lmasligini inobatga olsak, unda bundan yangi yashovchini MBga ro'yxatga kiritishni, u qachon daromad manbasiga ega bo'lginuvchi kutib turish kerak bo'ladi.

Yangilashdagi (o'zgartirishdagi) anomaliya. T-TN-T munosabatni aniq ko'rinadigan va ko'rinmaydigan ortiqchalikga egaligi bilan xarakterlanadi. Aniq ko'rinadigan ortiqchalikga yashovchining familiyasi, uning tug'ilgan sanasi, jinsi, manzili, telefon nomeri va boshqa atributlar qiymatlari munosabatda bir-necha marta takrorlanishi mumkin. Misol Mambetbaeva Q. ning adresi munosabatda ikki marta qaytariladi. Agar u adresini o'zgartirganini ma'lum qilsa, unda munosabatda uning adresini o'zgartirishda ma'lumotlar qarama-qarshiligi kelib chiqishining oldini olish uchun ikki kortejda ham o'zgartirish kiritishni ta'minlash lozim.

T-TN-T munosabatida aniq emas (ko'rinmas) ortiqchalik bu bir xil telefon nomeri bir kvartiradagi barcha yashovchilarga tegishli bo'lishi hisoblanadi. Misol uchun 8/11-12 adresli kvartirada yashovchilarning barchasida qaytalanadi. Agarda Salilaeva Z. MB bilan ishlovchilarga o'zining telefon nomeri o'zgartirganini ma'lum qilsa hamda ular shu asosida faqat uning telefon nomeri o'zgarsa, unda 8/11-12 adresli kvartira uchun munosabatda ikki turli telefon nomeri payda bo'ladi. Bu 8/11-12 adresi uchun aniq telefon nomeri yo'q yoki aniqlab bo'lmaydigan holatga keltiradi.

O'zgartiruvchi anomaliya. T-TN-T munosabatida Xojanova S. ga tegishli faqat bir kortej bor. Misol uchun MB bilan ishlovchi Xojanova S. ning ish joyidan bo'shaganini (daromad manbasi yo'qolganini) bildi hamda ushbu yozuvni o'chirdi. Bu harakat Xojanova S. shu adresda yashasa ham MBdan o'chirilib tashlanishiga olib keladi. Sababi munosabatda bu yashovchi haqida faqatgina bir kortej bor edi.

Funksional qaramlilik

Anomaliyalarning kelip chiqishining oldini olish maqsadida munosabatlarni ajratish (bo'lish) jarayoni *dekompoziciya deb* ataladi. Munosabatni bir nechta yangi munosabatlarga bo'lishda relyatsion algabraning tegishli operatsiyalari ishlatilib, bu munosabat proektsiyalari deb ataladi.

Dastlabki munosabatni «munosabatning proektsiyalari» deb atalishshi bir qatar yangi munosabatlarga bo'lish uchun relyatsion algabraning tegishli operatsiyalari qo'llaniladi. Munosabatning toliq dekompozitsiyasi - bu ularning tabiiy bog'lanuvchi ushbu munosabat bilan bir xil bo'ladigan uning cheklanmagan sondagi proektsiyalari toplami bo'lib sanaladi.

Dekompozitsiyalashni amalga oshirishda qaralayotgan munosabatdagi atributlar orasidagi funksional qaramliliklar kalit bo'lib

hisoblanadi. *Funksional qaramlilik* (FB) quyidagicha aniqlanadi: A va B atributlari berilgan bo'lib, agar A ning har bir ma'nosi uchun barcha vaqtda u bilan bog'liq bo'lgan B ning bir ma'nosi bor bo'lsa, unda B atributi A atributidan funksional qaramli deb ataladi. A va B bir emas ikki yoki undan ko'p atributlarga ega bo'lsa murakkab (sostavnoy) deb ataladi.

Ushbu ta'rifning ma'nosi shundan iboratki, agar B funksional A ga bog'liq bo'lsa, unda A da qandaydir qiymatiga ega bo'lgan kortejlar B da quddi shu qiymatiga ega bo'lishi kerak, A va B qiymatlari vaqti-vaqti bilan o'zgarishi mumkin, ammo bunda A ning har bir noyob qiymati B bilan bog'liq bo'lgan bitta qiymatga ega bo'ladiganday o'zgarishi kerak. Funksional bog'liqlikni tavsiflash uchun eng ko'p ishlatiladigan ikkita usuli 5.4-rasmda ko'rsatilgan.

$$a) A \rightarrow B \quad b) \textcircled{A} \rightarrow \textcircled{B}$$

5.4-rasm. Funksional bog'liqlikni tavsiflash:

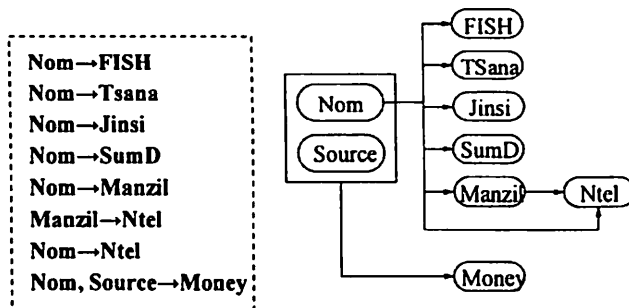
a — matematik; b — grafik

Muayyan vaziyatda, funksional bog'liqlik, munosabatlardagi barcha atributlarning semantik xususiyatlarini tahlil qilish va atributlarning bir-biri bilan qanday bog'liqligi to'g'risida xulosa chiqarish orqali aniqlanadi. Funksional bog'liqlik atributlarning asosiy xususiyatlariga qarab olinishi kerak.

Misol tariqasida, T-TN-T universal munosabatlaridagi atributlarga murojaat qilamiz (4-jadvalga qarang) va bu atributlar qanday bog'liqligini ko'rib chiqamiz. Atributlarning semantikasini o'rganib chiqqandan so'ng, 2.1.4-rasmdagi bog'liqliklarning mavjudligi ko'rinadi.

Ushbu Funksional bog'liqliklarning mavjudligini tushuntiradigan mulohazalar quyidagicha:

1. Turg'unlarning raqamlari noyobdir. Har bir turg'uniga Nom raqami beriladi va barcha raqamlar bir biriga tug'ri kelmaydi. Shunday qilib, agar yashovchining raqami ma'lum bo'lsa, u holda faqat bitta familiya, ism va otaning ismi (FISH) bilan bog'lanishi mumkin: Nom→FISH. Teskarisi tug'ri emas: FISH→Nom to'g'ri funksional bog'liqlik emas, chunki bir nechta yashovchining familiyasi, ismi va otasining ismi bir xil bo'lishi mumkin;



2.1.4-rasm. T-TN-T munosabatlarning atributlari o'rtasida mavjud funksional bog'liqlikni taqdim etishning turli usullari

2. shunga o'xshash mulohazalar TSana, Jinsi va SumD atributlariga nisbatan ham qo'llaniladi. Agar yashovchi raqami ma'lum bo'lsa, u bilan faqat bitta tug'ilgan sana, bitta jinsi va bitta umumiy daromad bog'liq bo'lishi mumkin: **Nom**→**FISH**, **Nom**→**Tsana**, **Nom**→**Jinsi**. Teskarisi funksional bog'liqliklarga to'g'ri kelmaydi, chunki bir nechta aholining tug'ilgan sanasi, jinsi va umumiy daromadi bir xil bo'lishi mumkin emas;

3. har bir fuqaro bitta kvartirada ro'yxatga olingan, ammo bitta kvartirada bittadan ko'proq fuqaro yashashi mumkin. Shunday qilib, **Nom**→**Manzil** to'g'ri funksional bog'liqlik;

4. har bir kvartira uchun bittadan ortiq telefon hisobga olinmaydi, kvartiraning borlig'i yoki yo'qligi telefon bilan aniqlanmaydi, lekin telefon bor bo'lsa u qaysidir bir kvartiraga bog'liq bo'ladi, **Manzil**→**Ntel**;

5. Turg'unning raqamini bilib, uning yashaydigan kvartirasida telefon raqamini bilish yoki telefonning yo'qligini aniqlashingiz mumkin: **Nom**→**Ntel**;

6. Turg'un uchun mavjud bo'lgan Money daromad manbaining hajmi, agar Source daromad manbai bo'lgan Nom turg'unining hfqfxb ma'lum bo'lsa, aniq belgilanadi: **Nom, Source**→**Money**. Ushbu funksional bog'liqlik unga kiritilgan murakkab atributlar to'plamidir (**Nom, Source**).

Munosabatlarning normal formalari

Har bir normal forma ma'lum shartlar to'plamiga mos keladi va xos shartlar to'plamini qondirsa, munosabatlar ba'zi normal formada bo'ladi. Shunday qilib yuqoridagi 2-bo'limda berilgan ta'rifga ko'ra, 1 NFda joylashgan munosabatlar uchun, uning barcha atributlari bo'linmas va bo'sh bo'lmagan qiymatlarga ega bo'lishi kerak.

Barcha normal formalar uchun, har bir keyingi normal forma ma'lum bir tarzda avvalgisining xususiyatlarini yaxshilaydi va keyingi normal formaga o'tishda oldingi normal formalarning xususiyatlari saqlanib qolinishi odatiy hisoblanadi.

1-NF dan keyingi normal formalar qanoatlantiradigan shartlarni ko'rsatish uchun quyida ko'rib chiqilgan bir qator tushunchalar qo'llaniladi.

X va Y bir munosabatning bir yoki bir nechta atributlarini o'z ichiga olgan to'plamlar bo'lsin. Agar X bir nechta atributlardan iborat bo'lsa, unda Y to'plami X ga to'liq funktsional bog'liq deyiladi, agar (a) Y funktsional jihatdan X ($X \rightarrow V$) ga bog'liq bo'lsa va (b) Y har qanday X' dan mustaqil bo'lsa. Bu erda X' tuplami ' X to'plamining atributlari shundayf kuyi to'plamiki, X ning kamida bitta atributi X' ga tegishli bo'lmasligi kerak, ya'ni. $X' \subset X$.

$A \rightarrow C$ funktsional bog'liqligi tranzitiv deyiladi, agar $A \rightarrow B$ va $B \rightarrow C$ bog'liqliklari mavjud bo'lsa. Masalan, T-TN-T munosabatlari uchun, $Nom \rightarrow Ntel$ bog'liqligi tranzitiv, sababi $Nom \rightarrow Manzil$ va $Manzil \rightarrow Ntel$ bog'liqliklari mavjud.

Munosabatning atribut yoki atributlar to'plamini tasvirlaydigan kalit munosabat uchun birlamchi kalit sifatida ishlatilishi mumkin. Birlamchi kalit har doim mumkin bo'lgan kalitdir; ammo, mumkin bo'lgan birlamchi kalit sifatida ishlatilmagan boshqa kalitlar ham bo'lishi mumkin.

Kalit bo'lmagan atribut - bu munosabatning har qanday mumkin bo'lgan kalitlari tarkibiga kirmaydigan atributidir.

O'zaro mustaqil atributlar - bu munosabatning funktsional jihatdan bir-biriga bog'liq bo'lmagan atributlari.

MBni loyihalashda foydalanilgan boshqa bog'liqliklar asosida olinishi mumkin bo'lmagan axborotlarni o'zida jamlamagan funktsional bog'liqlik ortiqcha funktsional bog'liqlik deb ataladi. Ortiqcha funktsional bog'liqliklar o'zida noyob ma'lumotlarni saqlamaydi va natijalarni salbiy ta'sir ko'rsatmasligini ta'minlash uchun ularni funktsional bog'liqliklar to'plamidan olib tashlash mumkin.

Agar munosabat ikkinchi normal formada (2NF) bo'ladi, faqat shunda, agar munosabat 1NFda bo'lsa va birlamchi kalitdan boshqa barcha atributlar to'liq birlamchi kalitga bog'liq bo'lsa. T-TN-T munosabati 2NFda emas, chunki uning birlamchi kaliti juft $\langle Nom, Source \rangle$ atributlari hisoblanadi va o'zi boshlang'ich kalit emas SumD atributi Nom ga bog'liq bo'lganligi sababli munosabatdagi asosiy kalit bilan to'liq funktsional bog'liq emas.

Munosabat uchinchi normal formada (3NF) bo'ladi, agar u 2NF bo'lsa va uning asosiy bo'lmagan atributlaridan biri funktsional ravishda boshqa biron bir kalit atributga bog'liq bo'lmasa.

Relyatsion MB nazariyasida dastlabki va eng muhim natijalardan biri Kodd tomonidan tasdiqlangan: agar har bir munosabatlar Boys-Kodd normal formasiga o'tkazilsa, MBda yuzaga kelishi mumkin bo'lgan anomaliyalarning aksariyati yo'q qilinadi. Munosabat Boys-Kodd normal formasida (NFBK) bo'ladi, agar u 3NFda bo'lsa va munosabatlarning har bir determinanti uning mumkin bo'lgan kaliti bo'lsa. Masalan, T-TN-T munosabati uchun, ko'rsatilgan shartlarning ikkinchisi bajarilmaydi, chunki munosabatlarning har bir determinanti mumkin bo'lgan kalit emas:

5-jadval

Mumkin bo'lgan kalitlar	Determinantlar
<Nom, Source >	<Nom, Source>
	<Nom>
	<Manzil>

Munosabat to'rtinchi normal formada (4NF) bo'ladi, agar u faqat NFBK da bo'lsa va uning to'liq dekompozitsiyalangan ikkala proektsiyada mumkin bo'lgan umumiy kalitga ega emas bo'lsa.

Munosabat beshinchi normal formada (5NF) bo'ladi, agar u faqat 4NF da bo'lsa va uning to'liq dekompozitsiyalarida har bir proektsiyada mumkin bo'lgan kalit mavjud bo'lsa.

Munosabatlarni dekompozitsiyalashga umumiy yondashuv

Oddiy vaziyatda munosabatlarni dekompozitsiyalash usuli bilan relyatsion MBni loyihalash to'rt bosqichdan iborat.

1. MB uchun universal munosabatlarni ishlab chiqish.
2. Munosabatlarning atributlari o'rtasidagi barcha funktsional bog'liqliklarni aniqlash.
3. Munosabatning NFBK da ekanligini aniqlash. Agar shunday bo'lsa, loyihalash tugallandi; agar yo'q bo'lsa, munosabat ikki munosabatlarga bo'linishi kerak.
4. Dekompozitsiya natijasida yuzaga keladigan har bir yangi munosabat uchun 2 va 3-bosqichlarni takrorlash. Loyihalash barcha aloqalar NFBK da bo'lganda tugaydi.

Munosabatni ikkita yangi munosabatlarga bo'lish quyidagicha funktsional bog'liqlik yordamida amalga oshiriladi.

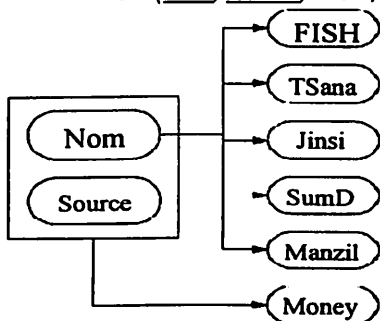
Aytaylik, $R(A, B, C, D, E, \dots)$ munosabati NFBK da emas. Masalan $C \rightarrow D$ funktsional bog'likligi R munosabatining NFBK bo'lmasligi uchun sabab bo'lsin (ya'ni, C determinat, ammo mumkin bo'lgan kalit emas). R

munosabatining proektsiyalari bo'lgan ikkita $R1(A,B,C,E,...)$ va $R2(C,D)$ munosabatlari yaratiladi. Bu erda funktsional bog'liqlikning qaram qismi (ya'ni, D atributi) $R1$ yaratilganda unnan $R2$ munosabati shakllantirishda funktsional bog'liklik bilan ishlatilgan. Shundan so'ng, biz $R1$ va $R2$ munosabatlarining NFBK da mavjudligini tekshirishingiz kerak.

Ushbu turdagi dekompozitsiya, tabiiy ulanishda yo'qotishsiz dekompozitsiya deyiladi.

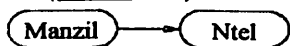
Ta'riflangan usuldan foydalanishning misoli tariqasida T-TN-T munosabatini dekompozitsiyalaymiz. Bu munosabatning mumkin bo'lgan kalitlariga murojaat qilsak, mumkin bo'lgan kalitlar emas ikkita $\langle \text{Nom} \rangle$ va $\langle \text{Manzil} \rangle$ determinantlari borligini ko'ramiz.

$R2$ (*Nom, Source, FISH, Tsana, Jinsi, SumD, Manzil, Money*)



Mumkin bo'lgan kalitlar	Determinantlar
$\langle \text{Nom, Source} \rangle$	$\langle \text{Nom, Source} \rangle$ $\langle \text{Nom} \rangle$

$R2$ (*Manzil, Ntel*)



Mumkin bo'lgan kalitlar	Determinantlar
$\langle \text{Manzil} \rangle$	$\langle \text{Manzil} \rangle$

2.1.5-rasm. T-TN-T munosabati proektsiyalarining natijalari bo'lgan $R1$ va $R2$ munosabatlari

Dekompozitsiyani boshlashdan oldin universal T-TN-T munosabati qayta ko'rib (Nom, Source, FISH, TSana, Jinsi, Manzil, Ntel, Money, SumD) va munosabatlarning atributlari orasidagi barcha funktsional bog'liqliklar aniqlanadi (2.1.5-rasmga qarang).

Funktsional bog'liqliklar orasida proektsiyalash uchun Nom va Manzil determinantlariga bog'liq bo'lgan funktsional bog'liqliklar saylanadi: $\text{Nom} \rightarrow \text{FIO}$, $\text{Nom} \rightarrow \text{TSana}$, $\text{Nom} \rightarrow \text{Jinsi}$, $\text{Nom} \rightarrow \text{SumD}$, $\text{Nom} \rightarrow \text{Manzil}$, $\text{Manzil} \rightarrow \text{Ntel}$, $\text{Nom} \rightarrow \text{Ntel}$.

Keyinchalik, birinchi proektsiyalash uchun qaysi funktsional bog'liqlikni tanlash kerakligini hal qilishimiz kerak. Proektsiyalash uchun

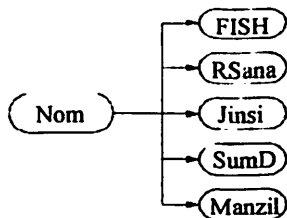
funksional bog'liqlikni tanlashning oddiy qoidasi, misoli $A \rightarrow B \rightarrow C$ shaklidagi «FB zanjiri»dan o'ng tomonidagi oxirgi bog'liqlikni tanlashimiz kerak. Bizning holatda, bunday zanjir $Nom \rightarrow Manzil \rightarrow Ntel$ bo'lib, "zanjirning oxiri" $Manzil \rightarrow Ntel$ birinchi proektsiyani keltirib chiqaradi. Natijada olingan R1 va R2 munosabatlari funksional bog'liqliklari bilan birgalikda 2.1.5-rasmda keltirilgan.

R2(Manzil, Ntel) munosabatining barcha determinantlari mumkin bo'lgan kalitlar bo'lsa ham, u NFBK da joylashgan va dekompozitsiyalashga hojat yo'q. Shu bilan birga, R1(Nom, Source, FISH, TSana, Jinsi, SumD, Manzil, Money) bilan bog'liqlik NFBK da emas, chunki $\langle Nom \rangle$ determinanti mumkin bo'lgan kalit emas. Shuning uchun R1 munosabatini yanada parchalash kerak. Qiyinchilik yuzaga kelgan $\langle Nom \rangle$ determinantiga bog'liq bo'lgan beshta $Nom \rightarrow FISH$, $Nom \rightarrow TSana$, $Nom \rightarrow Jinsi$, $Nom \rightarrow SumD$, $Nom \rightarrow Manzil$ atributlari mavjud bo'lib, ularni bitta FB ning o'ng tomoni sifatida $Nom \rightarrow FISH$, $TSana$, $Jinsi$, $SumD$, $Manzil$ ko'rish mumkin.

R1 munosabatini {Nom, FISH, TSana, Jinsi, SumD, Manzil} va {Nom, Source, Money} atributlari to'plamlariga proektsiyalash bizga 2.1.6-rasmda ko'rsatilgan R3 va R4 munosabatlarini beradi. Ushbu ikki munosabat NFBK ga mos kelib, R2 munosabati bilan birgalikda turg'unlar to'g'risidagi ma'lumotlarga ega bo'lgan MB datalogik modeli sifatida foydalanish mumkin. 4.8-rasmda loyihalashtirilgan MB uchun yakuniy T-TH-T_DB shakli tasvirlangan, va dastlabki T-TN-T munosabati ma'lumotlarga mos keladigan munosabatlar ma'lumotlari bilan ko'rsatilgan (5.1-jadvalga qarang).

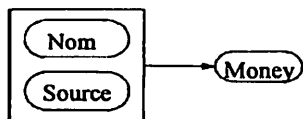
Olingan MBdan ko'rinib turibdiki, dekompozitsiya jarayonida T-TN-T dastlabki munosabati avtomatik ravishda uchta mantiqiy qismga bo'linadi: R2, bu kvartira va telefonlar haqida ma'lumotni o'z ichiga oladi; R3, unda turg'unlar, ularning umumiy daromadi va manzillari to'g'risidagi ma'lumotlar mavjud; R4, unda turg'unlarning daromadlari manbalari va hajmi to'g'risidagi ma'lumotlar mavjud. Bunday mantiqiy bo'linish dastlabki munosabatdagi turli xil atributlar bir-biri bilan qanday bog'liqligi batafsil bayon qilingan funksional bog'liqliklar asosida dekompozitsiya jarayonida foydalanilishini bevosita natijasidir.

R3(Nom, FISH, RSana, Jinsi, SumD, Manzil)



Mumkin bo'lgan kalitlar	Determinantlar
<Nom>	<Nom>

R4(Nom, Source, Money)



Mumkin bo'lgan kalitlar	Determinantlar
<Nom, Source>	<Nom, Source>

2.1.6 -rasm. R1 munosabini proektsiya qilish natijasida olingan R3 va R4 munosabatlari

T-TN-T bitta jadvalli MBda kiritish, yangilash va o'chirish anomaliyalari aniqlanganligi sababli, ishlab chiqilgan T-TN-TDB MB ham tekshirilishi kerak.

Kiritish anomaliyasi. Agar daromad manbalari bo'lmagan yangi turg'un paydo bo'lsa, u holda R3 munosabatida barcha maydonlarning ma'lum bo'lgan (bo'sh bo'lmagan) qiymatlari bilan satr kiritiladi. Bu holatla boshqa munosabatlarga satrlar kiritilmaydi va turg'unning daromad manbalariga ega emasligi T-TN-TDB MBda SumD umumiy daromadning nol qiymati bo'lishi bilan ko'rsatiladi. MBda bo'sh (ya'ni, aniqlanmagan) maydonlar mavjud emasligi sababli, bazada yangi turg'unni ro'yxatga olish muvaffaqiyatli amalga oshirildi va shuning uchun kiritish anomaliyasi bo'lmaydi.

Yangilash anomaliyasi. T-TN-TDB MBda ortiqchaliklar sezilarli darajada kamaydi. Shunday qilib, turg'unning qancha daromad olishidan qat'i nazar, R3 munosabatida har bir turg'un to'g'risidagi ma'lumotlar bitta satrda keltirilgan. Masalan, agar turg'un o'z manzilining o'zgarishini xabar qilsa, unda T-TN-T MBdagi kabi, ko'p yo'nalishlarda emas, balki R3 munosabatining bitta qatorida o'zgarishi kerak.

Loyihalangan MBda, shuningdek, kvartiralarda o'rnatilgan telefon raqamlari tug'risida ma'lumotlarning ortiqcha bo'lishi ham yo'q qilinadi.

Ushbu misollar T-TN-TDB MBda yangilanish anomaliyalari mavjud emasligini ko'rsatadi.

O'chirish anomaliyasi. Aytaylik, MB bilan ishlaydigan xodim, qaysidir bir turg'un qaysidir bir daromad manbaidan mahrum bo'lganligini bilib, R4 munosabatlaridan tegishli kortejni olib tashladi. Ushbu xodim haqidagi barcha boshqa ma'lumotlar MBda saqlanadi. T-TN-TDB MBda olib tashlash anomaliyasi mavjud emas.

Binobarin, BKNF si MBdagi anomaliyalarning ko'pini yo'q qilishni kafolatlaydi.

Nazorat savollari

1. Relyatsion MBni loyihalashning maqsadlari nima?
2. Universal munosabatni foydalanishda qanday muammolar bor?
3. Funktsional bog'liqliklar qanday alohalarni ko'rsatadi?
4. 1NF va 2NF larning farqlarini ko'rsating?

2.2. Ma'lumotlar bazasini kontseptual va datalogik loyihalash

Kontseptual loyihalashning zarurligi

Kontseptual loyihalash butun MBni loyihalash jarayonining asosidir. MB PSSini yetarlicha aks ettirishi uchun MBni tuzuvchi ushbu predmet sohasiga tegishli barcha xususiyatlarni yaxshi bilishi va ularni MBda aks ettirish imkoniyatiga ega bo'lishi kerak. Shuning uchun, MBni loyihalashni boshlashdan oldin, predmet sohasi qanday ishlashini batafsil tushunish kerak. Ushbu bosqichda mijozning (masalan, menejeri yoki iqtisodchisining) ishtiroki muhimdir.

Predmet soha (PS) oldindan tavsiflanishi kerak. Buning uchun tabiiy tildan foydalanish mumkin, ammo uning qo'llanilishida ko'plab ta'rifning noaniqligi va uni izohlashning noaniqligi kabi kamchiliklar mavjud. Shu sababli bu maqsadlar uchun sun'iy formal til vositalaridan foydalaniladi.

Kontseptual loyihalashning maqsadi kelajakda ishlatiladigan dasturlardan mustaqil bo'lgan maxsus til vositalari yordamida yaratilgan dasturiy tavsiflardan iborat bo'lgan PSSining *infologik modelini* (ILM dasturi) yaratishdir.

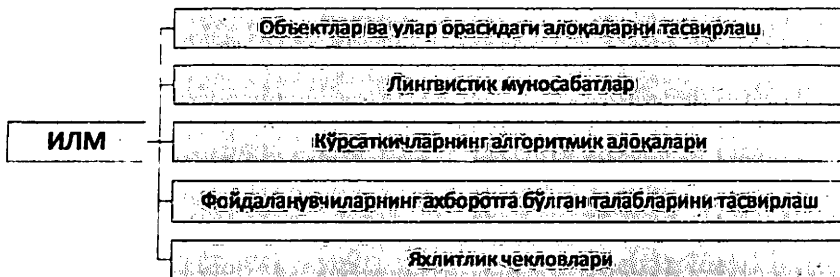
ILM spetsifikatsiyasi tili qo'lda ham, avtomatlashtirilgan loyihalashda ham bir xil qo'llanilishi maqsadga muvofiqdir. Buning uchun u:

- hisoblanadigan bo'lishi mumkin, ya'ni kompyuter tomonidan qabul qilinadi va qayta ishlanadi;
- foydalanuvchilar uchun qulay interfeysdan, xususan, grafik interfeysdan foydalanish;
- tez-tez o'zgarib turadigan asbob-uskunalar va boshqa manbalardan mustaqil bo'lish;
- ILM sinov vositalaridan foydalanish, shuningdek, ILM spetsifikatsiyasini tugatgandan so'ng MB tuzilishini generatsiya qilish uchun moslamaga (apparatga) ega bo'lish.

ILM ni axborot tizimini yaratishda ishtirok etayotgan turli toifadagi mutaxassislar osongina tushunishi kerak. ILM - bu oxirgi foydalanuvchilar va ishlab chiquvchilarning turli xil jamoalari o'rtasidagi aloqa vositasi. ILM axborot tizimini loyihalash uchun zarur va etarli ma'lumotlarni o'z ichiga olishi kerak. ILM bir qator tarkibiy qismlarni o'z ichiga oladi (2.2.1-rasm).

Ob'ektlar va ular orasidagi aloqalarni tasvirlash

Asosiy komponentalar bu PSSi ob'ektlari va ular o'rtasidagi munosabatlarning tavsifi bo'lib hisoblanadi.



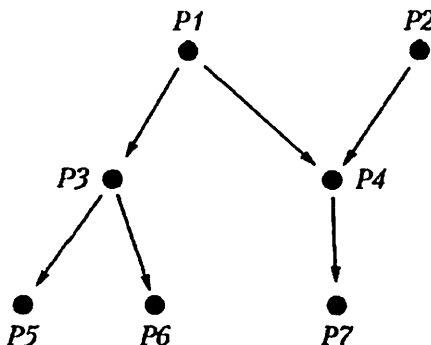
2.2.1-rasm. ILM komponentalari

PSning tavsifi har doim biron bir belgilar tizimidan foydalangan holda taqdim etiladi. Shuning uchun, ILM nafaqat PSsiga xos bo'lgan munosabatlarni, balki til aloqalarini ham aks ettiradi. Shuning uchun "ma'nodoshlik", "shakldoshlik", "izomorfizm" va boshqalar kabi lingvistik toifalarni hisobga olish kerak.

ILMda ko'rsatkichlar orasidagi algoritmik bog'liqliklar ham aks ettirilishi kerak. Ushbu maqsadlar uchun qaysi ko'rsatkichlar boshqalarining hisoblash manbai bo'lib xizmat qilishini aks ettiruvchi ko'rsatkichlarning o'zaro bog'liqlik grafini ishlatishingiz mumkin (2.2.2-rasm). Hisoblash formulalari, shuningdek hisoblash algoritmlari ILM da taqdim etilishi kerak.

Keyingi komponent - bu foydalanuvchilarning axborotga bo'lgan ehtiyojlarini tavsiflash. Ushbu tavsif so'rov turini, ma'lumotlardan foydalanish rejimini va boshqalarni aks ettirishi kerak.

PSning muhim xususiyati yaxlitlik cheklovlari deb ataladi, ular ILMda ham aks etiladi. Yaxlitlik cheklovlari - bu MBda saqlanadigan qiymatlarning ma'noli bo'lishiga shartlar, yoki MBga yozilishida mumkin bo'lgan shartlar.



2.2.2-rasm. Ko'rsatkichlarning o'zaro bog'liqlik grafi

Ma'lumotlar yaxlitligi muammosi har qanday vaqtda MBda ma'lumotlarning to'g'riligini ta'minlashdir. Ma'lumotlar yaxlitligi maxsus shartlar yoki yaxlitlikni cheklash deb nomlangan bayonotlar to'plami bilan ta'minlanadi. Yaxlitlik cheklovlari bu alohida ma'lumot birliklarining ruxsat etilgan qiymatlari va ular o'rtasidagi munosabatlarning maqbulligi to'g'risidagi tasdiqlashlar. Yaxlitlik cheklovlari aksariyat hollarda dasturiy ta'minotning uziga xos xususiyatlari bilan belgilanadi, ammo ular faqat ma'lumot xarakteristikalarini aks ettirishi mumkin.

Yaxlitlik cheklovlari turli xil ma'lumot birliklariga tegishli bo'lishi mumkin: atributlarga (maydonlar), kortejlarga (satrlar, yozuvlar), munosabatlarga (jadvallar, fayllar), munosabatlar o'rtasidagi aloqalarga va boshqalar.

Maydonlar uchun quyidagi cheklash turlari eng ko'p ishlatiladi.

1. *Maydonning turi (tipi) va formati*;
2. *Qiymatlar oralig'ini (diapazonini) belgilash*. Diapazonning qiymati va uning turi PSning xususiyatlariga bog'liq bo'ladi;
3. *Bo'sh bo'lmagan maydonning belgisi*. Bu MBda bo'sh maydon qiymatining mumkin emasligini tavsiflaydi. Masalan, xodimlar to'g'risidagi ma'lumotlarni o'z ichiga olgan munosabatlarda "familiya", "ism", "otasining ismi", "ish haqi" satrlari, albatta, ma'lum ma'noga ega bo'lishi kerak va "ilmiy daraja" maydoni hech qanday qiymatga ega bo'lmasligi mumkin;
4. *Domen berish*. Maydon berilgan qiymatlar to'plamidan qiymat olishi mumkin. Masalan, jins maydoni qiymati erkak yoki ayol bo'lishi mumkin. Professor-o'qituvchilar uchun "lavozim" maydonining qiymati quyidagi qiymatlardan biri bo'lishi mumkin: "assistent", "katta o'qituvchi", "dotsent", "professor". Domen o'z qiymatlarini ro'yxatlash orqali aniqlanishi shart emas.

Har qanday tasniflash kabi, cheklovlar turlarining yuqoridagi tasnifi shartli hisoblanadi. Bundan tashqari, domen algoritmik tarzda aniqlanishi mumkin. Masalan, ko'plab MBBT-lar DATE maydon turini qo'llab-quvvatlaydi va qiymatlarni kiritishda kiritilgan sana haqiqiylikini avtomatik tekshirishni ta'minlaydi. Shuning uchun ma'lumotlar yaxlitligini saqlash uchun MBni boshqarish imkoniyatlari haqida bilish va maydonning to'g'ri turini tanlash muhimdir.

Maydon qiymatining ma'lum bir cheklanishi uning unikalligining belgisidir. Ushbu cheklash ushbu maydon qiymatining haqiqiylikini tekshiradi, ammo butun jadval (fayl) skanerdan o'tkaziladi.

Qiyamatning unikalligining belgisi kalit tushunchasi bilan chambarchas bog'liq. Unikal (noyob) maydon bu munosabatlarning mumkin bo'lgan kalitidir. Agar bir nechta mumkin bo'lgan kalitlar bo'lsa, ulardan biri birlamchi (asosiy) kalit sifatida tanlanishi kerak. Bu maydon bo'sh qiymat olmasligi kerak. Hamma MBBTlar kalit tushunchani qo'llab-quvvatlamaydi va MBni tavsiflashda uni aniqlashni talab qilmaydi.

Ko'rib chiqilayotgan cheklash ko'pincha ba'zi ob'ektlar MBda ko'rsatilganda ro'y beradi va unikal maydon ob'ektning identifikatori hisoblanadi. Shuning uchun u ko'pincha ob'ektning yaxlitligining cheklashi deb ataladi.

Yuqorida ko'rib chiqilgan cheklovlar ushbu qiymat birinchi marta kiritiladimi yoki MBda mavjud bo'lgan qiymatlar to'g'rilanganligidan qat'i nazar, maydon qiymatini tekshirishni aniqlaydi. Faqatgina sozlashlarning to'g'riligini tekshirish uchun foydalaniladigan cheklovlar o'tish cheklovlari deb ataladi. Masalan, agar MBda "xodimlar yoshi" maydoni bo'lsa, o'zgartirish paytida ushbu maydonning qiymati faqat oshishi mumkin. Agar MBda "tug'ilgan yil" maydoni saqlangan bo'lsa, unda ushbu maydonni o'zgartirish imkoniyati taqiqlanishi kerak.

Kortej bilan bog'liq yaxlitlik cheklovlari haqida gap ketganda, bu butun satrning qiymatini cheklashni yoki butun satrdagi alohida maydonlar qiymatlarining nisbatlarini cheklashni tushiniladi. Shunday qilib, tabiiy cheklash bu jadvalning har bir qatorining unikalligi talabi hisoblanadi. Ta'rif bo'yicha, munosabatlarda bir xil kortejlar bo'lmasligi kerak, ammo hamma relyatsion MBBT lar ham ushbu cheklovlarga rioya qilmaydilar.

Butun jadvalga nisbatan qo'llaniladigan cheklovlarga misol sifatida quyidagilarni keltirish mumkin. Ish haqi fondi bitta ishchining o'rtacha ish haqi asosida shakllantiriladi va bu qiymat M so'mga teng deb faraz qilaylik. Unda jadvalning yaxlitligini cheklash uchun, «oklad» maydonining o'rtacha qiymati M dan oshmasligini ko'rsatadigan shartni belgilash kerak. Bir xil jadvalning qatorlari o'rtasidagi munosabatlarni tekshiradigan yaxlitlik cheklovlariga quyidagilar misol bo'ladi: 1) bir odamning ota-onasi va bolasi bo'lish mumkun emas; 2) ota-onaning tug'ilgan yili bolaning tug'ilgan yilidan kam bo'lishi kerak. Yuqoridagi misollardan birinchisi, tsiklning yo'qligiga cheklovning maxsus holati.

Ko'rib chiqilgan barcha cheklovlar bitta jadval ishidagi ma'lumotlar birliklariga ta'sir qiladi. Ushbu turdagi cheklovlarga qo'shimcha ravishda ko'pincha bir nechta bog'liq jadvallarga ta'sir qiladigan cheklovlar qo'llaniladi. Ushbu cheklovlarning eng keng tarqalgani aloqa yaxlitligini cheklashdir. Bu ob'ektlar o'rtasidagi munosabatni aks ettiradigan va

munosabatlarning tashqi (ikkinchi darajali) kaliti bo'lgan atributning qiymati, albatta, mos ob'ektni tavsiflaydigan munosabatlarning birlamchi (asosiy) kaliti bo'lgan atributning qiymatlaridan biri bilan mos kelishi kerakligi bilan izohlanadi.

Aloqa yaxlitligini cheklashning bir turi mavjudlikning aloqasini cheklashdir, bu ob'ektning R4 munosabatida mavjud bo'lishi uchun uchun R2 munosabatidagi ob'ekt bilan bog'liq bo'lishi kerakligidan iborat. Masalan, ishga kirgan har bir xodim qandayda bir bo'limga birlashtirilishi kerak, va "Kadrlar" jadvalidagi "bo'lim" maydonidagi tegishli yozuv "Bo'limlar" jadvalidagi tegishli maydonning qiymatlaridan biriga mos qiymatga ega bo'lishi kerak.

Bundan tashqari, jadvallarning o'zaro bog'liqligini aks ettiruvchi cheklovlar tegishli jadvallardagi ma'lumotlar o'rtasida mantiqiy nomuvofiqliklar yo'qligini tasdiqlaydigan shartlar bo'lishi mumkin. Masalan, agar har bir lavozim uchun ma'lum bir "ish haqi stavaka" o'rnatilgan bo'lsa, unda "Kadrlar" jadvalidagi ish haqi maydoni "lavozimlar" jadvalida qayd etilgan "stavkalar" dan oshmasligi kerak.

Yaxlitlikni cheklashning bir turi yangilanishni taqiqlashdir. Bu ma'lumotni qayta ishlash texnologiyasi yoki PS xususiyatlariga bog'liq bo'lishi mumkin. Agar "Shaxs" ob'ekti tasvirlangan bo'lsa, unda tug'ilgan sana va tug'ilgan joy kabi atributlar doimiy (statik) bo'lib, o'zgarib olmaydi.

Yaxlitlik cheklovlariga rioya qilishni nazorat qilish darhol (bir vaqtning o'zida) va kechiktirilgan yaxlitlik cheklovlari ajratiladi. Kechiktirilgan yaxlitlik cheklovlari bir guruh operatsiyalarni bajarish paytida kuzatilmasligi mumkin, ammo ular ushbu operatsiyalar guruhini tugatgandan so'ng kuzatilishi kerak. Tranzaktsiya tushunchasi kechiktirilgan yaxlitlikni cheklash tushunchasi bilan chambarchas bog'liq.

Yaxlitlik cheklovlarning juda muhim turi bu funktsional *bog'liqliklar*. Ushbu ILMda PSda mavjud bo'lgan funktsional bog'liqliklar to'g'risidagi ma'lumotlar qayd etiladi, MBni loyihalashda va MBning ishlashi davomida yaxlitlikni boshqarish uchun ishlatiladi. MBdagi tegishli maydonlari uchun yangilashni taqiqlash maqsadga muvofiqdir.

Yangilashni taqiqlash nafaqat bitta maydonga, balki butun satrga (yozuvga) va jadvalga ham tegishli bo'lishi mumkin.

Satrnı (yozuvni) yangilashni cheklash misolini ko'rib chiqamiz. Aytaylik, mukofotlar to'g'risidagi ma'lumotlar har bir xodim uchun kadrlar MBda saqlanadi. Ushbu ma'lumot "Rag'batlantirish" jadvalida saqlanadi, unda quyidagi atributlar (maydonlar) mavjud: xodimning tabel

raqami, rag'batlantirish turi, sana. Ushbu jadvalga satrlarni qo'shish mumkin, ammo har bir alohida yozuvni o'zgartirish mumkin emas.

Ushbu misolda, "Rag'batlantirish" va "Xodimlar" jadvallari o'rtasida aloqalar cheklashi mavjud: "Rag'batlantirish" jadvalidagi tabel raqami "Xodimlar" jadvalida bo'lishi kerak; «Xodimlar» jadvalidagi qandayda bir qatorni o'chirishda, «Rag'batlantirish» jadvalidagi unga bog'liq barcha satrlar o'chirilishi kerak.

Ba'zi MBBT lar ma'lumotlarni tariflashda kiritish va kaskadli o'chirish uchun majburiy a'zolik deb nomlanishini belgilashga imkon beradi. Bunday holda, korrektirovkalashda yaxlitlik tizim tomonidan avtomatik ravishda ta'minlanadi va aloqa cheklanishining mavjudligi kafolatlanadi.

Yaxlitlikni avtomatik nazorat qilish uchun ushbu ma'lumot ma'lumotlar lug'atida qayd etilishi kerak. Relyatsion algebra operatsiyalarini bajarishda yaxlitlikni nazorat qilish uchun kamida munosabatlarning kalitlari va mumkin bo'lgan kalitlari to'g'risida ma'lumot yozilishi kerak.

Yaxlitlik cheklovlari aniq va yashirin bo'lishi mumkin. Yashirin cheklovlar ma'lumotlar modelining o'ziga xos xususiyatlari bilan aniqlanadi va avtomatik ravishda MBBT tomonidan tekshiriladi.

Yuqorida ko'rib chiqilgan yaxlitlik cheklovlari misollari foydalanuvchi ma'lumotlariga tegishli. Yaxlitlik tushunchasi yordamchi ma'lumotlarga ham tegishli bo'lishi mumkin. Bu birinchi navbatda indeksli fayllar va ularning tegishli indekslangan MB fayllari o'rtasidagi mustahkamlikni saqlash bilan bog'liq.

MB yaxlitligi tushinchasi bilan bir qatorda, ma'lumotlar bankining axborot yaxlitligi tushunchasini ham kiritilish mumkin, bu uning barcha axborot tarkibiy qismlari (MB fayllari, dastur fayllari, kirish/chiqish shakllarining tavsifi, hisobotlar) to'g'ri bog'lanishini ta'minlashdan iborat. Masalan, agar MB fayli uchun u bilan bog'liq hisobot shakli mavjud bo'lsa, unda berilgan fayldan bir maydon o'chirib tashlaganida, ushbu hisobot formasida xatolik yuz beradi. Agar maydonda saqlanadigan ma'lumot turi o'zgarsa va boshqa ko'p hollarda yaxlitlikni buzilishi yuz berishi mumkin.

Ma'lumotlar bankining yaxlitligini cheklash MB jadvallari tuzilishini tavsiflashda (ya'ni MB sxemasida) yoki ma'lumotlarni qayta ishlash dasturlarida berilishi mumkin. Birinchi yondashuvning afzalligi, yaxlitlig cheklovi topshirig'ining tavsifiy (deklarativ) usuli nazoratning yuqori darajasini berishi tufayli emas, balki belgilangan cheklovlar ma'lumotlar

ustida barcha operatsiyalarni bajarish davomida nazorat qilinish mumkunligi tufayli.

Turli xil MBBTlarda ma'lumotlar yaxlitligini ta'minlash uchun turli xil vositalar to'plami mavjud. Shunday qilib, ba'zi bir relyatsion ma'lumotlar bazalari birlamchi kalit, domen va tashqi kalit kontseptsiyasini qo'llab-quvvatlaydi. Bunday holda, yaxlitlig cheklovini tekshirish avtomatik ravishda amalga oshiriladi. Ba'zi tizimlarda maydon uchun MB tuzilishini tavsiflashda siz bo'sh qiymatni olish taqiqni belgilashingiz mumkin (NOT NULL tushunchasi), bunda siz qabul qilinishi mumkin bo'lgan qiymatlar va boshqa yaxlitlig cheklovlarini aniqlab olishingiz mumkin.

Ma'lumotlar bankini loyihalashda foydalanilayotgan MBBT yaxlitlikni nazorat qilish uchun qanday imkoniyatlarga ega ekanligini o'rganish kerak. Agar MBBT avtomatik ravishda kerakli cheklovni qo'llab-quvvatlamasa, uning bajarilishini ta'minlash loyihalashturuvchining zimmasiga tushadi.

Ma'lumotlarning yaxlitligini buzish ma'lumotlar kiritish yoki boshqa o'zgarishlarni amalga oshirishda xatolar, shuningdek kompyuter tizimidagi nosozliklar natijasida yuzaga keladi.

MBda asosiy (bosh) va quyi (bo'ysunadigan) jadvallarning mavjudligi ma'lumotlarning takrorlanishiga olib keladi va har qanday aniq yoki yashirin takrorlanish ma'lumotlar yaxlitligini ta'minlash uchun maxsus choralarni talab qiladi.

Yaxlitlik chekloviga rioya qilish ustidan nazoratni amalga oshirish ba'zi hollarda katta kuch sarf qilishni talab qiladi. Agar bu mumkin bo'lsa, unda yaxlitlik buzilishi mumkin bo'lgan shartlarni belgilash kerak va ushbu shartlarni faqatgina yaxlitlik cheklovi buzilishiga olib keladigan holatlarda tekshirishi kerak. Masalan, agar rahbarning ish haqi unga bo'ysunadigan xodimning ish haqidan kam bo'lmasligi sharti bo'lsa, unda rahbarning ish haqi oshishi yaxlitlikni buzishi mumkin emas va bu holda MBni tekshirish kerak emas.

Ob'ektlarning tavsifi va ularning xususiyatlari

PSning infologik modelini analitik va grafik vositalar bilan tavsiflash mumkin. Grafik tavsiflash tushunish va tahlil qilish uchun eng sodda usul. Bundan tashqari, grafik tasvirlash MBni loyihalashni avtomatlashtirish tizimlari tomonidan qo'llab-quvvatlanadi. Shuning uchun ob'ektlar va ularning o'zaro munosabatlarini tavsiflash uchun "ob'ekt-xususiyat-aloqa"

deb nomlanadigan modelni yaratishga imkon beradigan grafik usulidan foydalaniladi.

PSni o'rganish va tahlil qilishda, mijoz yoki ishlab chiquvchi ob'ekt sinflarini aniqlaydi. Ob'ektlar sinfi - bu bir xil xususiyatlarga ega bo'lgan ob'ektlar to'plami. Masalan, OTM dagi qabul komissiyasining ishini PSSi deb hisoblasak, unda ob'ektlarning quyidagi sinflarini ajratish mumkin: abituriyentlar, imtihon oluvchilar, auditoriyalar va boshqalar. Ob'ektlar yuqorida sanab o'tilganidek materiallik yoki abituriyentlar imtihon topshiradigan fanlar kabi abstrakt bo'lishi mumkin.

Sinfning har bir ob'ekti qiymati bo'yicha shu sinfning boshqa ob'ektidan farq qilishini taminlaydigan o'zining noyob (unikal) identifikatori bilan ifodalanadi. Ob'ektlarning har bir sinfi ushbu sinf nomi bilan ko'rsatiladi. Shunday qilib, "FAN" deb nomlangan sinf ob'ektlari uchun har bir ob'ektning o'ziga xos identifikatori fan nomi bo'ladi (masalan, fizika).

Har bir ob'ekt ma'lum xususiyatlarga ega. Xuddi shu sinf ob'ektlari uchun ushbu xususiyatlar to'plami bir xil, ammo xususiyatlarning qiymatlari farq qilishi mumkin. Masalan, "ABITURIYENT" sinfidagi barcha ob'ektlar "Ro'yxatdan o'tish raqami", "FISH", "Tug'ilgan yili", "Jinsi" va boshqalar kabi xususiyatlarga ega.

PSni tavsiflashda ob'ektlarning har bir zarur sinfini va ushbu sinf ob'ekti uchun belgilangan xususiyatlar to'plamini tasvirlash kerak. Sinfning umumlashtirilgan vakili bo'lgan abstrakt ob'ekt sinfning nomi yozilgan to'rtburchaklar bilan ifodalanadi (masalan, ABITURIYENT). Ob'ektning xususiyati, xususiyat nomi yozilgan punktir (nuqtalar bilan chizilgan) to'rtburchaklar bilan ifodalanadi (masalan, ro'yxatga olish raqami).

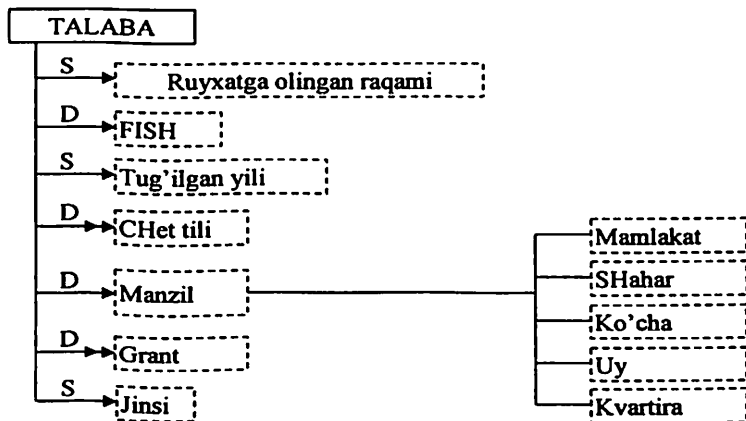
ILMning har bir sinfiga o'ziga xos nom beriladi. Agar ism bir nechta so'zlardan iborat bo'lsa, unda birinchi so'z mohiyatga ega nom bo'lishi kerak. Masalan, xususiyatlar sinfining to'g'ri nomi "OZIQ-OVQAT MAHSULOTI" emas, "MAHSULOT OZIQ-OVQAT" bo'lishi mumkin.

Agar an'anaviy ravishda turli xil nomlar PSlarning har qanday sinfini yoki ob'ekt xususiyatlarining turli nomlarini belgilash uchun ishlatilsa (ya'ni sinonim mavjud bo'lsa), ularning barchasini ILMda lingvistik munosabatlar shaklida o'rnatilishi kerak. Keyin nomlar yoki nomlardan biri asosiy deb tanlanadi va kelajakda ILMda faqat shu narsadan foydalanish kerak.

PSni tavsiflashda ob'ekt va uni tavsiflovchi xususiyatlar o'rtasidagi munosabatni aks ettirish kerak. Ushbu ulanishlar ob'ektning belgilarini va

uning xususiyatlarini bog'laydigan chiziqlar shaklida tasvirlanadi (2.2.3-rasm).

Ob'ekt va uning xususiyatlari o'rtasidagi munosabatlarning tasvirida ushbu xususiyatning o'ziga xos o'zgachaligi hisobga olinadi. Demak, ob'ektning qandayda bir xususiyati faqat bitta qiymatga ega bo'lishi ham mumkin. Masalan, har bir kishida faqat bitta tug'ilgan sana bor. Bunday xususiyatlar bitta deb nomlanadi.



2.2.3-rasm. «ob'ekt — xususiyat» aloqasi

Boshqa ob'ektlar uchun bir vaqtning o'zida ob'ektning bir xususiyati uchun bir nechta qiymatlar mavjud bo'lishi mumkin. Masalan, talaba bir nechta chet tillarni o'rganishi mumkin, shuning uchun "Chet tili" xususiyati ko'plik bo'ladi.

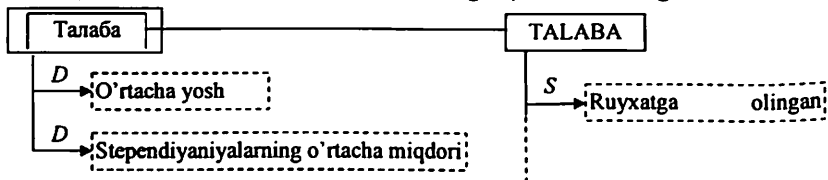
Ob'ekt va bitta qiymatli xususiyat o'rtasidagi munosabatlar *bittalik strelka* bilan belgilanadi, ob'ekt va bir nechta qiymatli xususiyat o'zaro munosabati *ikkilik strelka* bilan ko'rsatilgan.

Bundan tashqari, ba'zi xususiyatlar qiymati doimiy bo'lib, ularning qiymati vaqt o'tishi bilan o'zgarishi mumkin emas (masalan, tug'ilgan yili). Bunday xususiyatlarga *statik* deyiladi. Vaqt o'tishi bilan qiymatlari o'zgarishi mumkin bo'lgan xususiyatlar *dinamik* deyiladi.

Xususiyatning yana bir o'ziga xosligi bu xususiyatning ushbu sinfning barcha ob'ektlarida mavjudligi yoki ba'zi ob'ektlarning yo'qligi. Masalan, alohida talabalarga turli tashkilotlar tomonidan bir yoki bir nechta grant taqdim etilishi mumkin, boshqa talabalar uchun grantlar bo'lmasligi mumkin. Ushbu xususiyat shartli deb nomlanadi.

Shartli xususiyatning ob'ekt bilan o'zaro aloqasi punktir (nuqtali) chiziq bilan ko'rsatiladi va tegishli chiziq ustida D (dinamik) va S (statik) harflari xususiyatlarni ko'rsatish uchun ishlatiladi.

ILMda kompozit xususiyatlar (masalan, manzil) bilan bog'laydigan chiziqlar kvadrat bilan ko'rsatiladi. Agar ILMda nafaqat sinfning alohida ob'ektlariga xos bo'lgan xususiyatlarni, balki butun sinfga tegishli bo'lgan integral xususiyatlarni ham hisobga olish zarur bo'lsa, unda ba'zi bir aniq belgilarni (masalan, 2.2.4-rasmda ko'rsatilgan) ishlatishingiz mumkin.



2.2.4-rasm. Ob'ektlar sinfi va integral xususiyatlar tasviri

Ob'ektlar o'rtasidagi munosabatlarni tavsiflash

Ob'ekt va uning xususiyatlari o'rtasidagi munosabatlarga qo'shimcha ravishda, ILM turli sinf ob'ektlari o'rtasidagi aloqalarni ham qayd etadi. Aloqalarning «birga-bir» (1:1), «birga-ko'p» (1:M), «ko'pga-bir» (M:1) va «ko'pga-ko'p» (M:M) bo'lgan turlari mavjud. Aloqa turini aloqa darajasi ham deyiladi.

ILMda aloqa darajasiga qo'shimcha ravishda, turli sinftagi ob'ektlar o'rtasidagi aloqani tavsiflash uchun, mansublik sinfi ko'rsatilishi kerak. Mansublik sinfi bitta sinf ob'ekti boshqa sinfning har qanday ob'ekti bilan bog'liq bo'lishi kerak yoki kerak emasligini ko'rsatadi. Ob'ektning mansublik sinfi yo majburiy, yo ixtiyoriy bo'lishi kerak.

Agar berilgan sinfning barcha ob'ektlari o'zaro munosabatlarda qatnashishi kerak bo'lsa, unda ushbu ob'ektlar uchun mansublik sinfi majburiy deb bo'ladi. Agar ushbu sinfning ba'zi ob'ektlari munosabatlarda ishtirok etmasa, unda mansublik sinfi ixtiyoriy deb nomlanadi.

ILMda ob'ektlar o'rtasidagi munosabatlar ulanish darajasi va mansublik sinfini hisobga olgan holda 1979 yilda Chen (Chen R.R.S.) tomonidan tavsiya etilgan, "mohiyat-alloqa" yoki "ob'ekt-alloqa" (ER-Entity (mohiyat, ob'ekt) va Relationship (alloqa)) deb nomlangan diagrammalar yordamida tasvirlangan.

Aytaylik, PS OTM ning kafedrası bo'lib, uning o'qituvchilari o'quv dasturida mavjud bo'lgan fanlarni o'qitadilar.

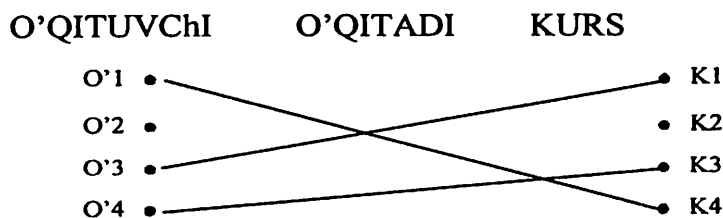
Ob'ektlarning ikkita asosiy sinflari yoki mohiyatlari bular O'QITUVCHI va KURS. Ushbu ikki mohiyat O'QITADI ulanishidan foydalangan holda bog'lanadi, bu quyidagilarni aytishga imkon beradi:

O'QITUVCHI	O'QITADI	KURS
------------	----------	------

Ko'rsatilgan mohiyatlar o'rtasida mavjud bo'lgan O'QITADI alohasini ikki shaklda ko'rsatish mumkin. 2.2.5-rasmda mohiyatlar o'rtasida alohani ER ekzemplarlar diagrammasidan foydalangan holda taqdim etilgan. Diagrammada har bir o'qituvchi qaysi kursni o'qitishi, har bir o'qituvchining identifikatsiya raqami (O'R) va har bir kursning raqami (KR) bo'yicha aniqlanishi ko'rsatilgan.

Diagrammalarning ko'rsatilgan misollari ob'ektlar yoki ob'ektlarning ulanish darajasi va tegishli sinfini aks ettiradi, ya'ni: ob'ektlar yoki mohiyatlarning ulanish darajasi 1:1 va ikkala mohiyatning mansublik sinflari ixtiyoriydir.

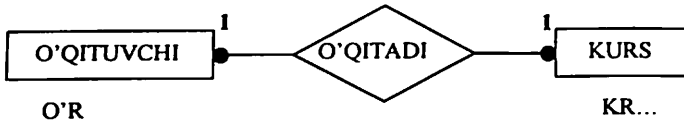
ER-ekzemplarlar diagrammasida (2.2.5-rasmga qarang) barcha mohiyatlarning nomlari ushbu mohiyatlarning namunalari tepasida joylashtirilgan va nomlarda bosh harflar ishlatilgan, mohiyatning namunasi atribut qiymati bilan aniqlangan. Shunday qilib, KURS - bu mohiyat, K1 - bu mohiyatning alohida namunasi.



2.2.5-rasm. ER-ekzemplarlar diagrammasiga misol.

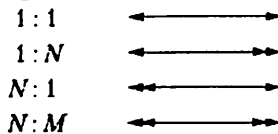
Aloqa ham nomlanadi va uning nomi katta harflardan tashkil topgan bo'lib, aloqalar namunasidan yuqorida joylashtirilgan, bunda har bir individual aloqaning namunasi ushbu o'zaro bog'liqlik o'rnatiladigan sub'ektlarning ikkita namunasi orasiga chiziq bilan o'rnatiladi. Masalan, K1 va O'3 o'rtasidagi aloqalarning namunasi, O'3 raqami o'qituvchi K1 raqamli kursni o'qitadiganin anglatadi. Mohiyat namunasini identifikatsiya qilish uchun foydalaniladigan atribut yoki atributlar to'plami mohiyat kaliti deyiladi. Aloqalarning har bir namunasi ushbu aloqa bilan bog'liq mohiyat kalitlari to'plamida aniq ko'rsatiladi.

ER-tur diagrammasida (2.2.6-rasmga qarang) mohiyatlar to'rtburchaklar shaklida, aloqalar romblar shaklida berilgan. Har bir mohiyatning pastki qismida ushbu mohiyat uchun kalit bo'lgan atribut yoki atributlar to'plami tagiga chizilgan turda berilgan. Mohiyatlarning yonidagi "1" raqamlari 1:1 ulanish darajasini ko'rsatadi va rombni to'rtburchaklar bilan bog'laydigan chiziqlardagi nuxtalar mansublik sinfi ixtiyoriy ekanligini ko'rsatadi.



2.2.6-rasm. ER-tur diagrammasiga misol.

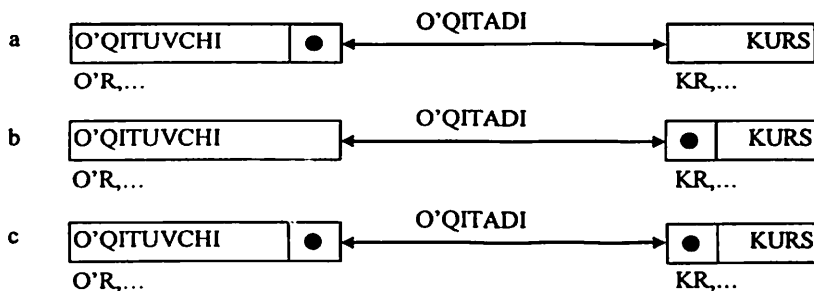
Mohiyatlarning aloqalari darajasini hisobga olgan holda ER-tur diagrammada aloqalarni tasvirlashning sodda usulini tavsiya etishimiz mumkin: romb o'rniga mohiyatlarni bog'laydigan ikki tomonlama strelka ishlatiladi va aloqa nomi strelka tepasida yoziladi yoki ko'rsatilmaydi. 2.2.7-rasmda turli xil aloqa darajalariga mos keladigan ulanishlarni tasvirlashning mumkin bo'lgan usullari ko'rsatilgan.



2.2.7-rasm. Bog'lanish darajalariga qarab mohiyatlarning aloqalarini tasvirlashning alternativ usullari

2.2.5 va 2.2.6 rasmlarda ko'rsatilgan diagrammalarda har bir o'qituvchi bittadan ko'p kurs o'qitmaydigan va har bir kurs bittadan ortiq o'qituvchi tomonidan o'qitilmaydigan holatlarga to'g'ri keladi, ya'ni umuman kurs o'qitmaydigan o'qituvchilarga va o'qitilmaydigan kurslar mavjudligiga ruxsat beriladi.

2.2.8-a-rasmdagi ER-tur diagrammasida har bir o'qituvchi faqat bitta kursni o'qitgan va har bir kursni bittadan ko'p bo'lmagan o'qituvchi o'qitgan holatga mos keladi. Barcha o'qituvchilar qaysidir bir kursni o'qitishda ishtirok etishi sababli, ya'ni qaysidir bir kurs bilan bog'liq holda, O'QITUVCHI mohiyati tegishli sinf mavjudligi talab qilinadi va bu fakt mohiyat joylashgan to'rtburchakta ulanish blokida nuqta bilan belgilanadi.



2.2.8-rasm. 1:1 aloqa darajasi va turli mansuvlik sinflari uchun diagrammalar

2.2.8-b-rasmdagi ER-tur diagrammasida har bir o'qituvchi bittadan ko'p bo'lmagan kursni (ya'ni 1 yoki 0 kurslarni) o'qingan holatga to'g'ri keladi va har bir kurs faqat bitta o'qituvchi tomonidan o'qinbladi.

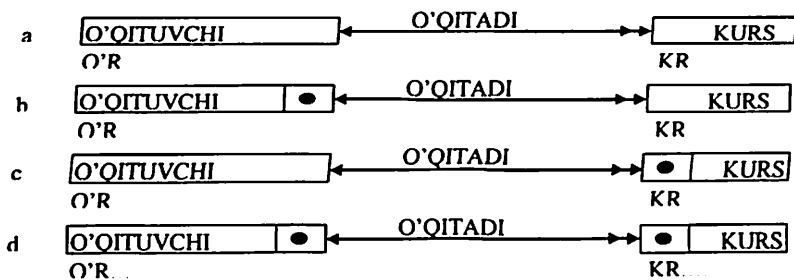
2.2.8-c-rasmdagi ER-tur diagrammasida har bir o'qituvchi faqat bitta kursni o'qitgan va har bir kurs faqat bitta o'qituvchi tomonidan o'qitilgan holatga mos keladi.

Uch holatda ham aloqa darajasi 1:1 ni tashkil qiladi va mohiyatning mansuvlik sinflarining kombinatsiyasi boshqacha.

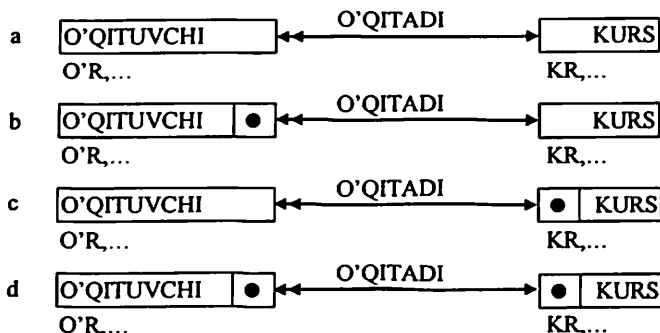
O'qituvchilar tomonidan kurslarni o'qitishni tashkil qilish holatlariga qo'shimcha ravishda, masalan quyidagilar ham bo'lishi mumkin:

1. har bir o'qituvchi bir vaqtning o'zida bir nechta kurslarni o'qitishi mumkin, ammo har bir kursga bittadan ko'p bo'lmagan o'qituvchi dars beradi;
2. har bir o'qituvchi bittadan ko'p bo'lmagan kursni o'qitadi, lekin har bir kurs birdan bir nechta o'qituvchi tomonidan o'qitilishi mumkin;
3. har bir o'qituvchi bir nechta kurslarni (xususan, xesh bir kursni o'qitmasligi) o'qitishi mumkin va har bir kursni bir nechta o'qituvchilar (xesh bir o'qituvchi o'qitmasligi) o'qitishlari mumkin.

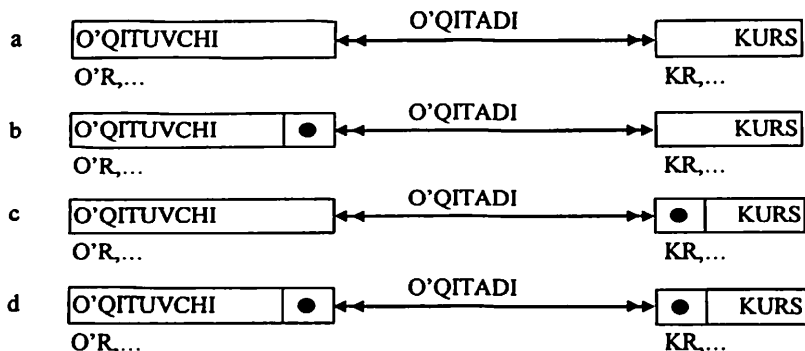
Ulanish darajasi 1:M, M:1,N:M va mohiyatga mansublik sinflarining turli xil birikmalari ushbu holatlarga mos keladi. 2.2.9-2.2.11-rasmlarda o'qituvchilarning darslar o'qitishini tashkil etishning barcha mumkin bo'lgan holatlarini aks ettirgan ER-tur diagrammalari ko'rsatilgan.



2.2.9-rasm. Aloqa darajasi 1:M bo'lgan holatlarga ER-tur diagrammlar



2.2.10-rasm. Aloqa darajasi N:1 bo'lgan holatlarga ER-tur diagrammlar



2.2.11-rasm. Aloqa darajasi N:M bo'lgan holatlarga ER-tur diagrammlar

Murakkab ob'ektlarning tavsifi

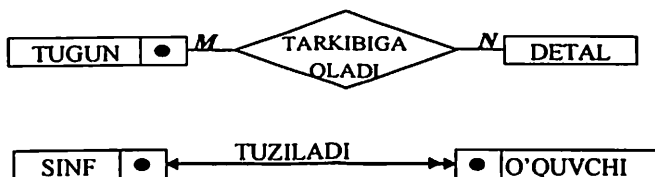
Ilgari ob'ektlar ularning murakkabligini hisobga olmasdan ko'rib chiqildi. Murakkablik darajasiga ko'ra ob'ektlar oddiy va murakkab bo'lishi mumkin. Ob'ekt oddiy bo'ladi, agar u ajralmas deb hisoblanadi.

Murakkab ob'ekt - bu predmet sohalarga ajraladigan boshqa ob'ektlarni kombinatsiyasi.

Murakkab ob'ektlarning bir nechta turlari mavjud: tarkibli (sostavnoy), umumlashtirilgan (obobshenny) va jamlangan (agregirovanny).

Tarkibli ob'ekt "butun-qism" munosabatlarining tassavuriga mos keladi, masalan: Tugunlar-Detallar, Sinf-O'quvchilar va boshqalar.

ILMda tarkibli ob'ektlarni namoyish qilish uchun ER-tipli diagrammalar qo'llaniladi:



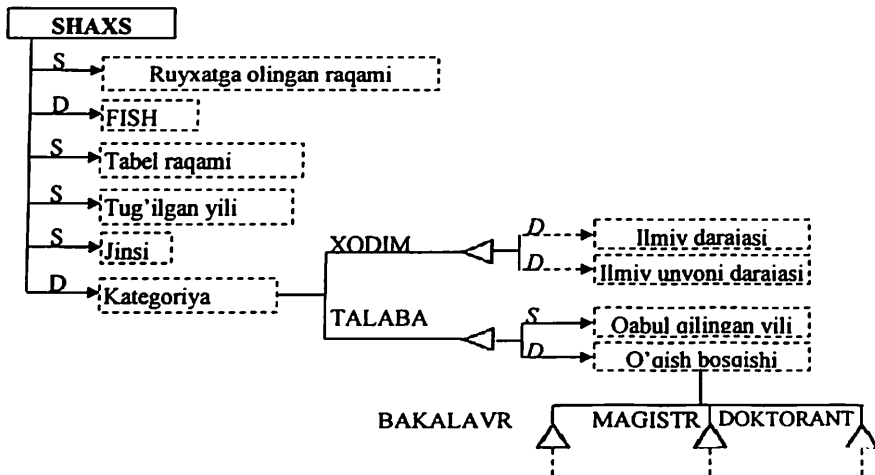
2.2.12

Umumlashtirilgan ob'ekt PS ob'ektlari o'rtasida "jins-tur" aloqalarining mavjudligini aks ettiradi. Masalan, BAKALAVR, MAGISTR, DOKTARANT ob'ektlari, umumlashtirilgan talaba ob'ektini tashkil qiladi.

"Tarkibiy" ob'ekt ham, "tur" ob'ektlari kabi ma'lum xususiyatlarga ega bo'lishi mumkin. Bundan tashqari, mulkka meros qolgan, ya'ni. "Tur" ob'ekti "tarkibiy" ob'ekt ega bo'lgan barcha xususiyatlarga va qo'shimcha xususiyatlarga ega bo'ladi.

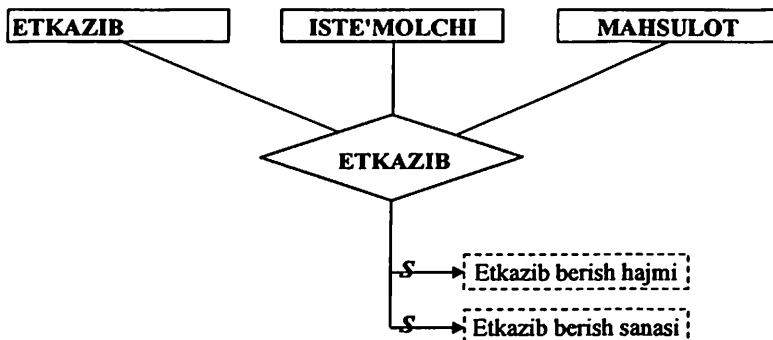
Tarkibiy turdagi aloqalarning ta'rifi PS ob'ektlarning u yoki bu xususiyatlarga ko'ra tasniflanishini anglatadi.

Ichki sinflar ILMda aniq turda ajratilishi mumkin, buning uchun grafik belgilash pastki sinfnings maxsus belgilashinan, masalan, uchburchakdan foydalaniladi. 2.2.13-rasmda OTM uchun SHAXS umumlashtirilgan ob'ektini aks ettiruvchi ILM bo'lagi ko'rsatilgan. Buning uchun bir nechta toifalar ajratilgan: Xodim, Bakalavr, Magistr, Doktorant.



2.2.13-rasm. Umumlashtirilgan ob'ektning tasviri

ILMda jamlangan ob'ekt romb bilan ifodalanadi, unda ob'ekt nomi ko'rsatiladi. Bu romb jamlangan ob'ektning tashkil etadigan ob'ektlarning belgilari bilan chiziqlar bilan bog'lanadi. Jamlangan ob'ektning xususiyatlari punktir chiziqli to'rtburchaklar bilan ifodalanadi. Masalan, YeTKAZIB BERISH ning jamlangan ob'ekti (2.2.14-rasm) MAHSULOT, ISTE'MOLCHI va YeTKAZIB BERUVCHI ob'ektlarini jamlaydi.



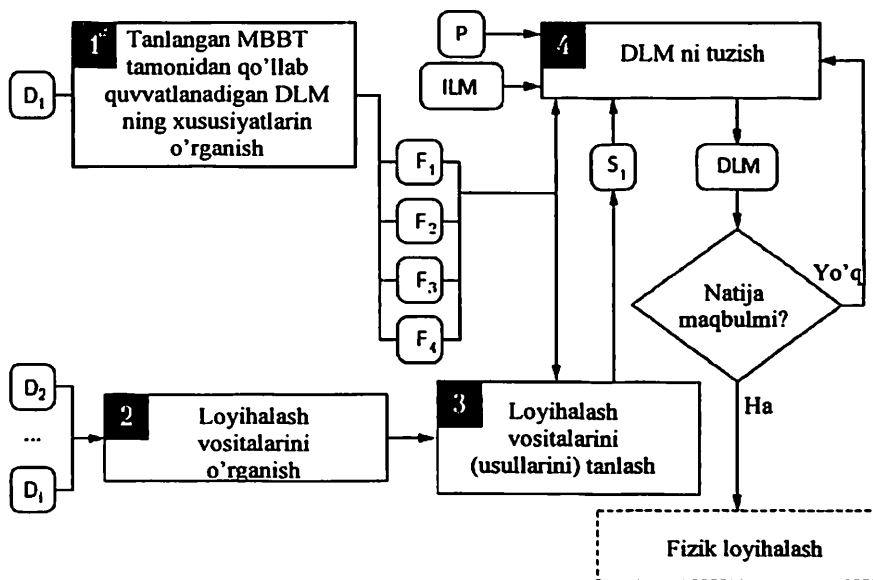
2.2.14-rasm. Jamlangan ob'ekt tasviri

Datalogik loyihalash

Umumiy ma'lumot

Datalogik dizaynning maqsadi ma'lumotlar elementlarining semantik tarkibi va saqlash muhitidan qat'i nazar, mantiqiy aloqalarni aks

ettiradigan *datalogik model* (DLM)ni yaratishdir. Ushbu model ma'lum bir MBBTda taqdim etilgan axborot birliklari nuqtai nazaridan tuzilgan. 5.14-rasmda DLMning rivojlanish ketma-ketligi ko'rsatilgan.



2.2..15-rasm. DLM ishlab chiqish ketma-ketligi

(D_1 – MBBT buyicha hujjatlar; D_2, \dots, D_i – loyiha vositalari bo'yicha hujjatlar; P – saqlanadigan ko'rsatkichlar (atributlar) ro'yxati; DLM – datalogik model; ILM – infologik model; S_1 – loyihalash vositasi; DLMga loyihalash vositalarini tanlashda ta'sir etuvchi omillar: F_1 – amaldagi datalogik tuzilmalar to'plami; F_2 – ma'lumotlarni tariflash tili (MTT) operatorlari; F_3 – MBBT tomonidan DLMga qo'yiladigan cheklovlar; F_4 – ma'lumotlarni fizik tashkil qilish imkoniyatlari)

Har qanday MBBT o'zi uchun maqbul bo'lgan ma'lumotlarning mantiqiy birliklari bilan ishlaydi, shuningdek yuqori darajadagi mantiqiy tuzilmalar va quyi darajadagi tarkibiy qismlarning ma'lumot birliklari uchun qoidalardan foydalanishga imkon beradi. Bundan tashqari, ko'plab MBBTlari MB tarkibiga miqdoriy va boshqa cheklovlar qo'yadi. Shuning uchun, DLM qurishdan oldin, MBBT xususiyatlarini batafsil o'rganish, loyihalash yechimini tanlashga ta'sir qiluvchi omillarni aniqlash, mavjud loyihalash usullari bilan tanishish va avtomatlashtirish vositalarini tahlil qilish va ulardan foydalanishning maqsadga muvofiqligini baholash kerak.

Garchi datalogik loyihalash MBning mantiqiy tuzilishini loyihalashtirsa ham, unga ma'lum bir MBBT tomonidan taqdim etilgan jismoniy ma'lumotlarni tashkil qilish imkoniyatlari ta'sir qiladi. Shuning uchun ma'lumotlarning jismoniy tashkil etilish xususiyatlarini bilish mantiqiy tuzilishni loyihalashda foydali bo'lishi mumkin.

MBning mantiqiy tuzilishi, shuningdek, MB ni ma'lumotlar bilan to'ldirish haqiqiy PSning tasvirlanishidir. Shuning uchun loyihalash qarorlarini tanlash to'g'ridan-to'g'ri ILMda aks ettirilgan predmet sahoning o'ziga xos xususiyatlariga bog'liq.

Datalogik loyihalash natijasi ma'lumotlarni tasvirlash tilida MB ning mantiqiy tuzilishini tavsiflashdir. MBning loyihalashtirilgan mantiqiy tuzilishida barcha axborot birliklari va ularning o'zaro aloqalari, axborot birliklarining nomlari, ularning turi va miqdoriy xususiyatlari (masalan, maydon uzunligi) aniqlanishi kerak.

Datalogik loyihalashga yondashuv

MBni yaratish jarayoni PS ob'ektlarini oldindan tasniflashni, ob'ektlar va ular o'rtasidagi o'zaro aloqalarni ilmda tizimli ravishda taqdim etishni ta'minlaydi.

Datalogik loyihalashning dastlabki bosqichlarida MBning tarkibi aniqlanishi kerak.

MBning mantiqiy tuzilishini loyihalashda, ILM ma'lum bir MBBT tomonidan qo'llab-quvvatlanadigan ma'lumotlar modeliga aylantiriladi va olingan dasturning olingan DLMning muvofiqligi tekshiriladi.

Har qanday PSni DLMda tasvirlash uchun loyiha yechimlarining ko'plab variantlari bo'lishi mumkin. Loyihalash metodikasi eng mos loyiha yechimini tanlashni ta'minlashi kerak.

Barcha MBBT ma'lumotlarining minimal mantiqiy birligi semantik jihatdan bir xil va ob'ekt identifikatoriga yoki ob'ekt yoki jarayonning xususiyatlariga mos keladi.

ILMda aks ettirilgan PS ob'ektlari o'rtasidagi aloqalar DLMda tegishli axborot elementlarini birgalikda tuzish orqali yoki ular orasidagi ulanishni e'lon qilish orqali ko'rsatilishi mumkin. PSdagi barcha munosabatlar to'g'ridan-to'g'ri ma'lum bir DLMda namoyish etilishi mumkin emas. Ko'pgina MBBT ob'ektlar orasidagi N:M aloqasini bevosita qo'llab-quvvatlamaydi. Bunday holda, DLMga ushbu aloqani namoyish etadigan qo'shimcha yordamchi element kiritiladi.

Shuni yodda tutish kerakki, PSda mavjud va ILMda aks ettirilgan ulanishlar nafaqat MB tuzilishi orqali, balki dastur turida ham amalag

oshirilishi mumkin. Masalan, umumlashtirilgan ob'ektlarni namoyish qilganda, MBning mantiqiy tuzilishi darajasida kichik sinflarni tanlay olmaysiz, lekin saqlanadigan ma'lumotlarni qayta ishlashda dasturiyda pastki sinflarni tanlashni ta'minlaysiz.

Qabul qilingan loyihalash qarori nafaqat aks ettirilgan PSning xususiyatlariga, balki MBda saqlanadigan ma'lumotni qayta ishlash xususiyatiga ham bog'liq. Masalan, ko'pincha birgalikda ishlov beriladigan ma'lumotlarni birgalikda saqlash va aksincha, bir vaqtning o'zida ishlatilmaydigan ma'lumotni turli xil fayllarda ajratish tavsiya etiladi. Tez-tez ishlatiladigan ma'lumotlar va ko'p ishlatilmaydigan ma'lumotlar turli xil fayllarda saqlanishi kerak.

MBning tarkibini aniqlash

ILMdan DLMga o'tishda ILM MBni loyihalash uchun zarur bo'lgan va etarli bo'lgan barcha ma'lumotlarni o'z ichiga olganligini yodda tutish kerak. Ammo bu ILMda qayd etilgan barcha xususiyatlar DLMda aniq aks ettirilgan bo'lishi kerak degani emas. DLMni tuzishdan oldin, MBda qanday ma'lumot saqlanishini hal qilish kerak. Masalan, ILM hisoblangan ko'rsatkichlar bor bo'lishi mumkin, ammo ularni MBda saqlash shart emas.

MBda saqlanadigan ko'rsatkichlar tarkibini aniqlashga yondoshuvlardan biri sintez printsiptiga asoslanadi: MBda faqat dastlabki ko'rsatkichlar saqlanishi kerak, barcha qayta ishlanadigan ko'rsatkichlar so'rov vaqtida hisoblanishi kerak.

Ushbu yondashuvning afzalliklari:

1. "nima saqlash kerak" to'g'risida qaror qabul qilishda soddaligi va aniqligi;
2. ma'lumotlarning aniq takrorlanishining yo'qligi;
3. MBda saqlanganlarni emas, balki har qanday hisoblangan ko'rsatkichni olish imkoniyati.

Ushbu afzalliklarga qaramay, har bir holda, hisoblangan ko'rsatkichlarni MBda saqlashning maqsadga muvofiqligini, foydalanish chastotasi va ushbu ko'rsatkichlarni hisoblashning murakkabligini hisobga olgan holda baholash kerak.

Ob'ekt MBda tasvirlanishida, ob'ekt identifikatori atribut bo'ladi, aksariyat hollarda ob'ektni identifikatsiyalash uchun foydalaniladi. Biroq, sun'iy identifikatorlarni yoki kodlarni kiritish kerak bo'lishi mumkin. Bu quyidagi holatlarda zarur:

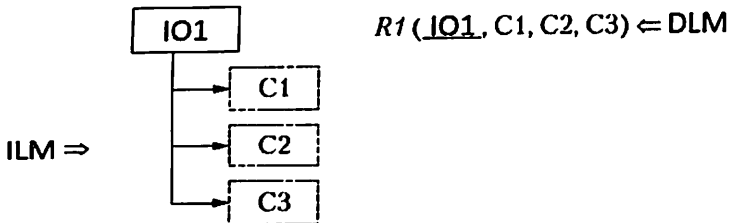
1. PSda shakldoshlik (omonimiya) kuzatilganda, ya'ni ob'ektning tabiiy identifikatori noyob emas. Masalan, talabalar orasida ismdosh va/yoki familiyadoshlari bo'lishi mumkin. Bunday holda, ob'ektlarni aniq identifikatsiyalash uchun kodlardan foydalanish kerak;
2. agar ob'ekt ko'plab bog'lanishlarda ishtirok etsa, unda ob'ektning uzun tabiiy identifikatoriga qaraganda ulanishni aniqlash uchun ob'ektning qisqa kodini ishlatish qulayroq bo'ladi;
3. agar tabiiy identifikator vaqt o'tishi bilan o'zgarishi mumkin bo'lsa, unda kod yo'qligida, bu zarur ma'lumotlarni topishda ko'plab muammolarga olib kelishi mumkin.

ILM asosidagi relyatsion MBni loyihalash usuli

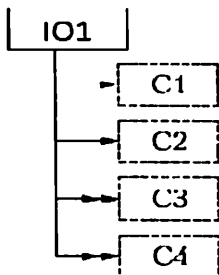
ILM tahlili va undan DLM munosabatlariga o'tish asosida relyatsion MBni loyihalash usulini ko'rib chiqamiz. Ushbu usul juda sodda va tushinarli, va ayni paytda yaxshi natijalar beradi.

ILM dan relyatsion DLMga o'tish uchun quyidagi qoidalar qo'llaniladi, bunda ob'ektlarning identifikatorlari IO1, IO2, IO3 sifatida belgilanadi va ob'ektlarning boshqa xususiyatlari C1, C2 va boshqalar kabi belgilanadi.

1. Har bir oddiy ob'ekt va uning individual xususiyatlari uchun, har bir individual xususiyatga mos keladigan ob'ektning identifikatori va rekviziti atributlar buladigan munosabatlar yaratiladi:



2. Agar ob'ekt bir nechta xususiyatlarga ega bo'lsa, unda ularning har biriga alohida munosabatlar tayinlanadi. Bu munosabatlarning kaliti bir nechta xususiyatlarni aks ettiradigan, mos ob'ekt identifikatori va rekviziti bo'ladi:

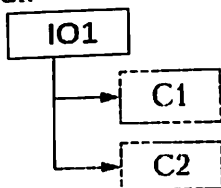


$R1(\underline{IO1}, C1, C2)$
 $R2(\underline{IO1}, C3)$
 $R3(\underline{IO1}, C4)$

3. Agar ob'ekt va uning xususiyatlari o'rtasida shartli bog'liqlik (shartli xususiyat) mavjud bo'lsa, u holda DLMda tasvirlashda quyidagi variantlar bo'lishi mumkin:

a) Agar ob'ektlarning ko'pchiligi shartli xususiyatlarga ega bo'lsa, unda uni MB da oddiy xususiyat turida saqlashga bo'ladi.

b) agar ob'ektlarning ozgina qismi shartli xususiyatga ega bo'lsa, unda munosabatni ob'ekt identifikatorini va shartli xususiyatga mos keladigan atributni o'z ichiga oladigan turda ajratishimiz mumkin. Ushbu munosabat ob'ekt qancha shartli xususiyatga ega bo'lsa, shuncha satrlarni o'z ichiga oladi.



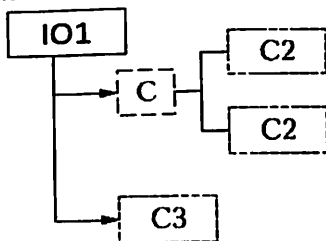
$R1(\underline{IO1}, C1, C2)$

ëки

$R1(\underline{IO1}, C1)$

$R1(\underline{IO1}, C2)$

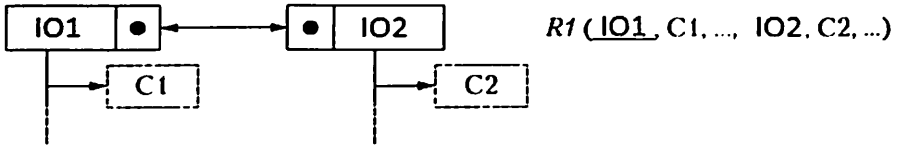
4. Agar ob'ekt kompozit xususiyatga ega bo'lsa, u kompozit xususiyat elementlariga mos keladigan atributlar to'plam turidagi munosabatda ifodalanadi:



$R1(\underline{IO1}, C1, C2, C3)$

?

5. Agar ob'ektlar o'rtasida 1:1 aloqa darajasi mavjud bo'lsa, u holda DLM ob'ektlarning tegishli sinfiga qarab aniqlanadi. Agar ikkala ob'ektning mansublik sinfi majburiy bo'lsa, unda DLM atributlar ob'ektlarning identifikatorlari va ikkala ob'ektning xususiyatlari bo'lgan bitta munosabat bilan belgilanadi:

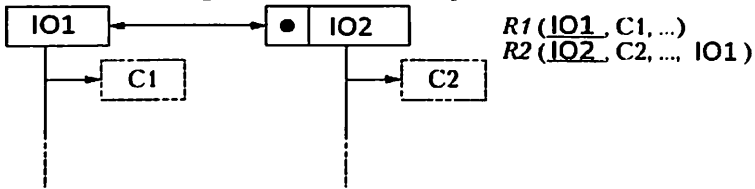


Bunday model eng kam xotirani talab qiladi. Ammo, agar so'rovlarda ko'pincha ob'ektlarning har biridan alohida-alohida ma'lumotlarni talab qilsa, unda ma'lumotni qidirishni tezlashtirish uchun har bir ob'ektni alohida munosabatlar sifatida taqdim etish tavsiya etiladi va DLMda ob'ektlarning o'zaro munosabatini, bir ob'ektning identifikatorini boshqa ob'ektga mos keladigan munosabatda atribut sifatida ko'rsatib berish tavsiya etiladi, masalan:

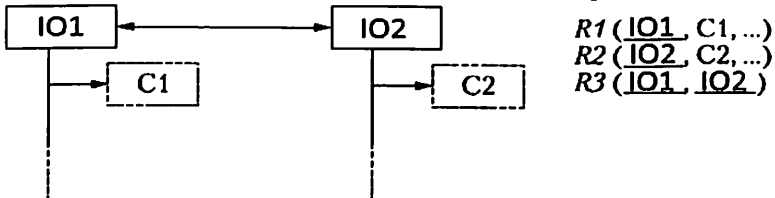
$R1(\underline{IO1}, C1, IO2)$

$R1(\underline{IO2}, C2)$

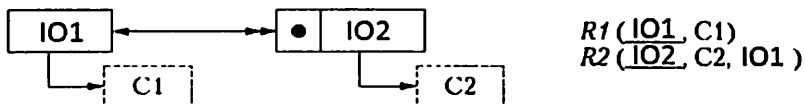
Agar ob'ektlardan birining mansublik sinfi ixtiyoriy bo'lsa, u holda DLM ikki munosabatlar bilan aniqlanadi va mansublik sinfi ixtiyoriy bo'lgan ob'ektning identifikatori zarur mansublik sinfi bilan ob'ektga mos keladigan munosabatlarga atribut sifatida qo'shiladi:



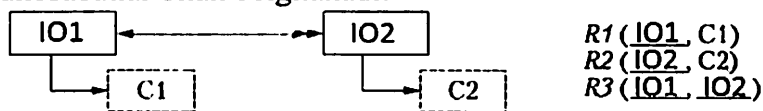
Agar ikkala ob'ektning mansublik sinfi ixtiyoriy bo'lsa, DLM uchta (har bir ob'ekt uchun bittadan va ob'ektlar o'rtasidagi munosabatni namoyish qilish uchun bitta) munosabatlar bilan aniqlanadi:



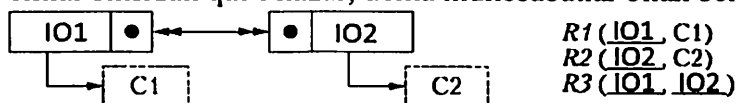
6. Agar ob'ektlar o'rtasida aloqa darajasi 1:M bo'lsa va ko'p ulangan ob'ektning mansublik sinfi majburiy bo'lsa, oddiy ulangan ob'ektning mansublik sinfidan qat'i nazar, DLM ikkita munosabat bilan aniqlanadi. Ko'p ulangan ob'ektga mos keladigan aloqa oddiygina ulangan ob'ektning identifikatori (kaliti) bo'lgan atribut bilan to'ldirilishi kerak:



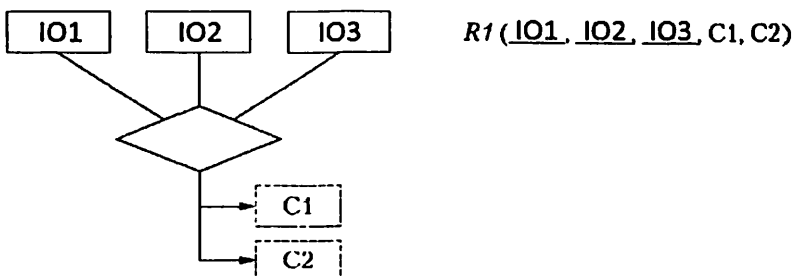
Agar ko'p aloqali ob'ektning mansulik sinfi ixtiyoriy bo'lsa, u holda DLM uchta munosabatlar bilan belgilanadi:



7. Agar PS ob'ektlari o'rtasida M: M aloqa darajasi bo'lsa, u holda DLM ob'ektlar sinfidan qat'i nazar, uchta munosabatlar bilan belgilanadi:

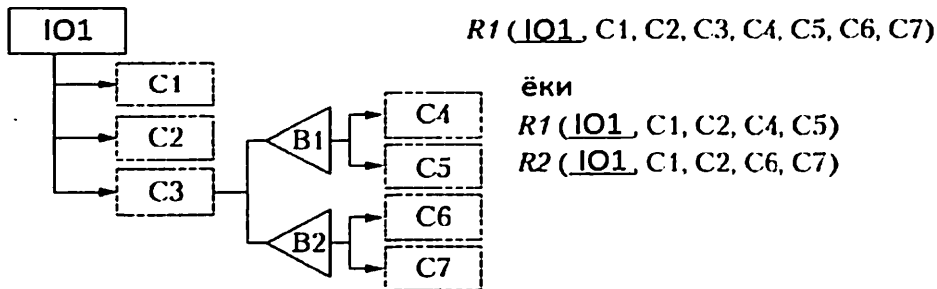


8. DLM-dagi PSda mavjud bo'lgan har bir jamlangan (agregatlangan) ob'ekt alohida munosabat tug'ri keladi. Ushbu munosabatlarning atributlari yig'ilgan ob'ektga kiritilgan barcha ob'ektlarning identifikatorlari, shuningdek ushbu ob'ektning xususiyati bo'ladi:



9. Umumlashtirilgan ob'ektlarni namoyish qilishda turli xil yechimlar bo'lishi mumkin. Birinchidan, umumlashtirilgan ob'ekt bitta munosabat bilan bog'liq bo'lishi mumkin. Ikkinchidan, quyi darjadagi ob'ektlarning har bir toifasi alohida munosabatga tug'ri kelishi mumkun. Birinchi holda, munosabatlarning atributi kamida bitta toifadagi ob'ektlarga xos bo'lgan barcha individual xususiyatlar, shuningdek ob'ektning identifikatori bo'ladi.

Ikkinchi holda, har bir munosabat ob'ektning identifikatorini, ushbu toifadagi ob'ektlarga xos bo'lgan xususiyatlarni, shuningdek, ierarxiyada undan yuqori bo'lgan umumiy ob'ektlarning xususiyatlarini o'z ichiga oladi.



Ushbu oxirg'i ikkita yechimga qo'shimcha ravishda, kombinatsiyalangan variantlarni qo'llash mumkin. Muayyan yechimni tanlash ko'pgina omillarga, shu jumladan ob'ektlarning turli toifalari ma'lumotlarining tez-tez birgalikda qayta ishlanishiga, "turlik" xususiyatlarida qanchalik katta farq borligiga bog'liq bo'ladi.

10. "Butun-qism" munosabatlarining mavjudligi bilan tavsiflanadigan jamlangan (kompozit) ob'ekt DLMda turli yo'llar bilan ko'rsatilishi mumkin. Masalan, TUGUN va DETAL ob'ektlari o'rtasida M:N aloqasi mavjud, chunki bir xil qism turli tugunlarda bo'lishi mumkin, va aksincha, turli qismlar bita tugunda bo'lishi mumkin. Tugunning tarkibi odatda murakkabdir va uni MB strukturasi aniq ifoda etish foydali emas va ba'zan imkonsizdir. Bundan tashqari, ko'rib chiqilayotgan alohalar bir hil ob'ektlar to'plamida amalga oshiriladi. Bunday holda "butun-qism" aloqalarini ko'rsatish uchun ikkita munosabatlardan foydalanish mumkin. Ulardan birinchisida ob'ektlarning o'zi to'g'risida ma'lumotlar, ikkinchisida esa ular o'rtasidagi munosabatlar to'g'risidagi ma'lumotlar, ushbu aloqani tavsiflovchi qo'shimcha ma'lumotlar mavjud bo'ladi. Tugunning tarkibi uchun bu "Nima kiradi", "Qaerga kiradi", "Miqdori" atributlari bo'lishi mumkin.

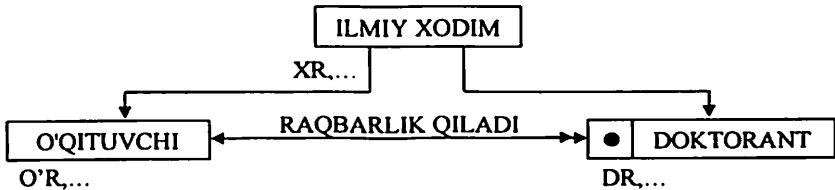
"Butun-qism" aloqasi, masalan biron-bir tashkilot tarkibini aks ettirishi mumkin. Bunday holda, aloqa darajasi 1:M bo'lib va DLMni qurish uchun 6-qoidaning tavsiyalari ishlatilishi mumkin.

11. Ba'zi hollarda, PSni har tomonlama modellashtirish uchun aniq ob'ektlar (mohiyatlar) va munosabatlar etarli bo'lmasligi mumkin. Ushbu holatlardan biri, qandaydir mohiyatning ekzemplarlari tashkilotda turli xil rollarni bajarishi kerak bo'lgan holatlarda yuz beradi.

Misol tariqasida, OTM kafedra uchun ilmiy kadrlar tayyorlash jarayoni to'g'risida ma'lumot to'plash kerak deylik. Ushbu jarayonda ob'ektlarning ikki toifasi mavjud: o'qituvchilar va doktorantlar. Ularning ikkalasi ham ilmiy kadrlardir, ammo ilmiy kadrlar tayyorlash jarayonida

ular turli xil rol o'ynaydi: o'qituvchilar o'z bilimlarini beradilar va aspirantlar bilimga ega bo'ladilar.

ER-tur diagrammasi rollarni hisobga olgan holda quyidagi shaklda bo'ladi.



DLM-ni ishlab chiqishda siz ushbu qoidaga amal qilishingiz kerak. Berilgan mohiyat bir munosabat bilan ifodalanadi, bunda mohiyat kaliti birlamchi kalit bo'lib xizmat qiladi. Rol ob'ektlari va ularni bog'laydigan ulanishlar DLMda ilgari tavsiflangan qoidalar bilan belgilanadigan munosabatlar bilan ifodalanadi va har bir rol oddiy mohiyat sifatida qaraladi.

Ushbu qoidaga ko'ra, bizning misolimiz uchun, DLM uchta munosabatlar bilan ko'rsatiladi:

$$R1(\underline{XR}, \dots)$$

$$R2(\underline{O'R}, \dots)$$

$$R3(\underline{DR}, \dots, O'R)$$

Ushbu munosabatlarda O'R va DR atribut qiymatlari to'plamlari kesishmaydi va XR atribut qiymatlari to'plami O'R va DR atribut qiymatlaridan iborat.

R1 munosabatida XR atributiga o'qituvchilar va aspirantlar uchun umumiy bo'lgan atributlar kiradi, R2 va R3 munosabatlarida, o'qituvchilar va aspirantlarga xos O'R va DR atributlari kiradi.

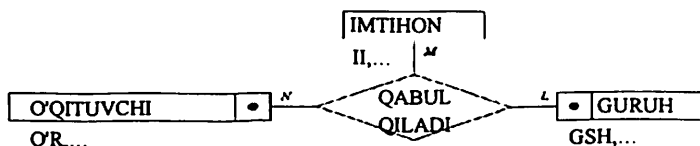
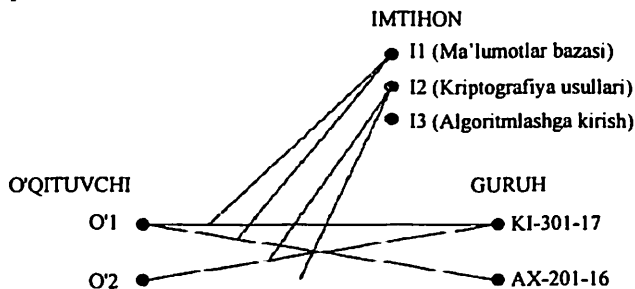
R1-ga oid so'rovlarni amalga oshirishda kirish tezligini oshirish uchun ob'ektning rolini ko'rsatadigan qo'shimcha atribut berilishi mumkin. Ushbu atribut R2 va R3 munosabatlaridan ob'ektning faoliyat turini aniqlash uchun zarur bo'lgan qidiruvni oshirmaslikga imkon beradi.

Shu tarzda olingan DLM 9-qoidani umumlashtirilgan ob'ekt uchun qo'llash natijasiga o'xshaydi. Bir xil PS turli xil ILMlar mos keladi, ammo ushbu turli xil ILMlarni DLMda namoyish etish o'xshash natijalarga olib kelishi mumkin.

12. Hozirgacha ILMda ob'ektlarning faqat binar aloqasi ko'rsatilgan edi, ya'ni bir juft ob'ekt o'rtasidagi aloqa. Garchi PSda ko'p holatlar ikkilik ulanishlar yordamida tasvirlanishi mumkin bo'lsa-da, ob'ektlar

o'rtasida yuqori darajadagi ulanishlar mavjud bo'lgan vaziyatlar muqarrar ravishda yuzaga keladi.

Masalan, ER-ekzemplarlar va ER-tur diagrammalari tasvirida guruhlardan o'qituvchilar tomonidan imtihonlarning qabul qilinishini ko'rib chiqamiz.



(*O'R* - o'qituvchining raqami; *II* - imtihon identifikatori; *GSh* - guruh shifri)

ILMda uch tomonlama aloqalarda, datalogik model to'rtta munosabatlardan iborat bo'ladi: har bir mohiyat uchun bittadan, har bir mohiyatning kaliti mos keladigan munosabatlar uchun asosiy kalit bo'lib xizmat qiladi va ulanish mohiyatlar uchun yana bitta munosabat bo'ladi. Aloqalar natijasida hosil bo'lgan munosabatga uning xususiyatlari orasida har bir mohiyatning kalitlari kiradi.

Ushbu misol uchun, DLM quyidagi munosabatlardan iborat:

$R1(\underline{O'R}, \dots)$

$R2(\underline{II}, \dots)$

$R3(\underline{GSH}, \dots)$

$R4(\underline{O'R}, \underline{II}, \underline{GSH}, \dots)$

Agar n-darajali aloqalar mavjud bo'lsa, unda $(n + 1)$ munosabatlar talab qilinadi: n munosabatlar mohiyatlar uchun va aloqa uchun bitta munosabat.

Shunday qilib olingan DLM, 8-qoidani jamlangan ob'ekt uchun qo'llanilishining natijasiga o'xshash aloqa munosabatini o'z ichiga oladi. Bu tasodif emas, chunki ILM bir nechta ob'ektlar ishtirok etadigan bitta

jarayonni namoyish etadi, ya'ni PS IMTIXON QABULI deb nomlangan agregatlangan ob'ekt sifatida talqin qilinadi.

ILM asosida relyatsion MBni loyihalashga misol

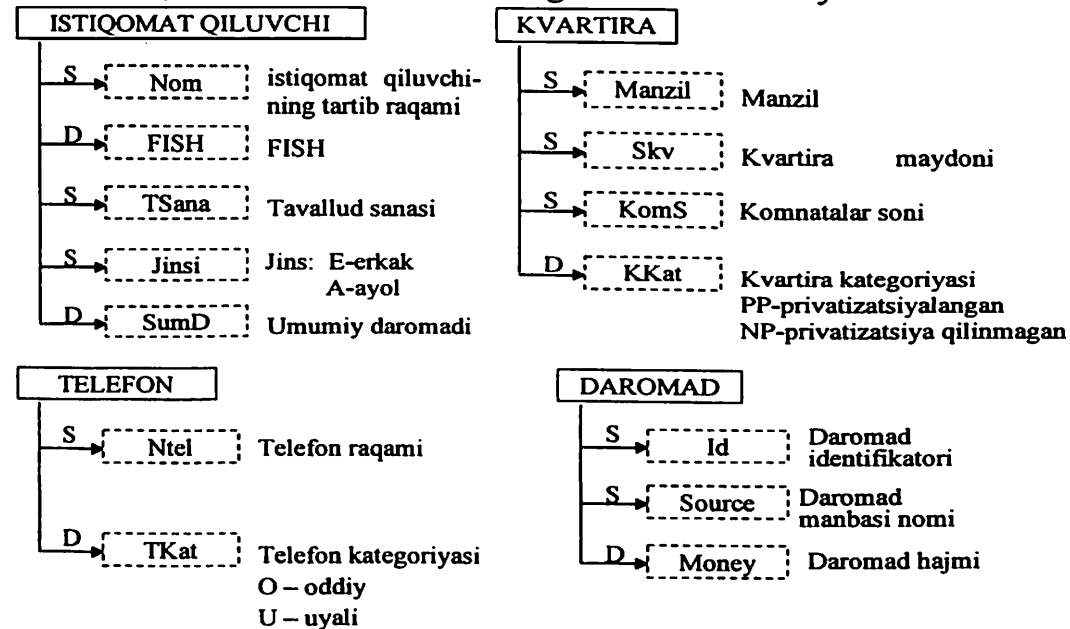
Misol uchun Nukus shahrining aholisini PS sifatida olinib, soliq idorasiga kerakli ma'lumotlar saqlanadigan eng oddiy relyatsion MB ni loyihalashtirish talab etilsin.

Loyihalash quyidagi komponentalarni o'z ichiga oladigan ILMni ishlab chiqish bilan boshlanadi:

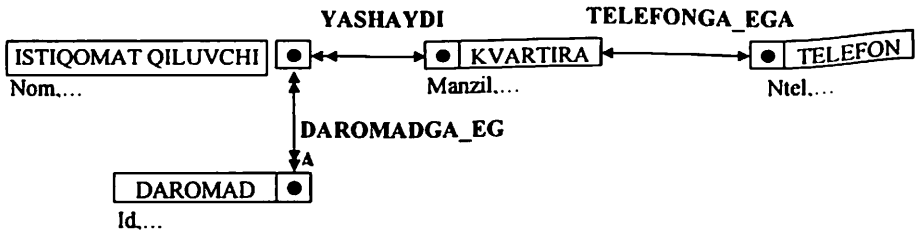
1. PS ob'ektlari va ular o'rtasidagi aloqalarning tavsifi;
2. lingvistik aloqalar;
3. ko'rsatkichlarning algoritmik aloqalari;
4. foydalanuvchilarning axborotga bo'lgan ehtiyojlari tavsifi;
5. yaxlitlikning cheklavlari.

Ob'ektlar va ular o'rtasidagi aloqalar tavsifi

Kerakli ob'ektlar: ISTIQOMAT QILUVCHI, KVARTIRA, TELEFON, DAROMAD. Biz ularning har birini ta'riflaymiz:



Ob'ektlar o'rtasidagi munosabatlar quyidagi ER-tur dagi diagrammada aks etadi:



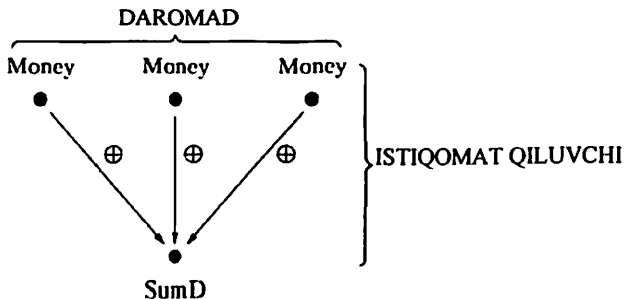
Lingvistik aloqalar

Lingvistik munosabatlarida ILMda ishlatiladigan atamalar va tushunchalarni izohlash kerak, masalan:

- Nom - butun son, Nukus shahrining turg'unining unikal (noyob) raqami;
- Manzil - turg'unning manzili; ko'chani, uyni (bino) va xonadon raqamlarini ko'rsatish orqali beriladi;
- Source - daromad manbaini belgilash:
 - Pensiya1 - keksa yoshdagi nafaqa;
 - nafaqa1 - bolalar uchun nafaqa;
 - stipendiyal - oshirilgan stipendiya;
 - Ish1 - ishlash; va hokazo.;
- Money - so'mdagi daromad miqdori.

Ko'rsatkichlarning algoritmik aloqalari

ILMda aks ettirilgan barcha ko'rsatkichlardan faqat umumiy daromad (SumD) algoritmik aloqaga ega. Uni hisoblash ko'rsatkichlarning o'zaro bog'liqligining quyidagi grafi bilan tavsiflanadi:



Foydalanuvchilarning axborotga bo'lgan ehtiyojlari tavsifi

Bu yerda MB foydalanuvchilaridan keladigan barcha so'rovlar aniqlanishi kerak, masalan:

1. barcha umumiy daromad ko'rsatadigan ro'yxatni olish;

2. umumiy daromadi soliq solinadigan eng kam miqdordan kam bo'lmagan turg'unlar ro'yxatini ko'rsatish;
3. yakka tartibdagi turg'unlarning soliqlarini va soliqlarning umumiy miqdorini hisoblash.

Axborot ehtiyojlarini tavsiflash axborot tizimini ishlab chiqish uchun asos bo'lib xizmat qiladi, bunda foydalanuvchi o'zlarining so'rovlariga javob oladi.

Yaxlitlik cheklovlari

Bu yerda MBda saqlanadigan ma'lumot manoga ega bo'lishi uchun alohida ko'rsatkichlar va ko'rsatkichlar guruhlari tomonidan bajarilishi kerak bo'lgan shartlar shakllantiriladi. Masalan, daromad miqdori salbiy bo'lmasligi kerak, daromad manbai ma'lum ro'yxatidan tanlanishi kerak va hokazo.

Yaxlitlik cheklovlar MBga kiritilgan ma'lumotlarning to'g'riligini va hisob-kitoblarning korrektligini nazorat qilish uchun amaliy tizimni ishlab chiqishda qo'llaniladi.

ILMni yaratgandan so'ng, DLMni yaratishga o'tiladi; bunda MBning tarkibini aniqlanadi va ILM xususiyatlarini hisobga oladigan ilgari shakllangan qoidalardan foydalaniladi.

MBning tarkibini aniqlash

Biz barcha ko'rsatkichlarni, shu jumladan hisoblangan SumD ko'rsatkichini ham MBda saqlanadi, chunki ko'p so'rovlarni bajarishda uning qiymati talab qilinadi.

MBga kiritiladigan munosabatlarni aniqlash

MBga kiritiladigan munosabatlarni aniqlash uchun ER-tur diagrammada ko'rsatilgan har bir tegishli ob'ektlar(mohiyatlar)ga tegishli qoidalarni qo'llanib, keyin ularning umumiy sonini kamaytirish uchun olingan munosabatlarni tahlil qilinadi.

6-qoidaga binoan, ISTIQOMAT_QILUVChI-KVARTIRA aloqadagi ob'ektlari ikkita munosabat bilan ifodalanadi:

PERSON (Nom, ..., Manzil)

PLAT (Manzil, ...)

7-qoida bo'yicha, ISTIQOMAT_QILUVChI-DAROMAD bog'liq ob'ektlari uchta munosabat bilan ifodalanadi:

PERSON (Nom, ...)

PROFIT (Id, ...)

HAVE_D (Nom, Id)

5-qoidaga binoan, KVARTIRA-TELEFON bog'liqli ob'ektlari ikkita munosabat bilan ifodalanadi:

FLAT (Manzil, ...)

TPHONE (Ntel, ..., Manzil)

Ushbu ettita munosabatlarning tahlili DLMda PSning ILM ni namoyish qilish uchun beshta munosabat etarli ekanligini aniqlashga imkon beradi:

PERSON (Nom, FISH, Tsana, Jinsi, SumD, Manzil)

FLAT (Manzil, Skv, KomS, KKat)

HAVE_D (Nom, Id)

TPHONE (Ntel, Tkat, Manzil)

PROFIT (Id, Source, Money)

MBning mantiqiy tuzilishini MBBT tilida tavsiflash

Datalogik loyixalash MBning mantiqiy tuzilishini MBBT tilida tavsiflash bilan yakunlanadi. Ushbu tavsif MBning sxemasi deb ataladi. U boshqa narsalar qatorida, atributning turi va uzunligi (hajmi) kabi munosabatlar atributlarining xususiyatlarini o'z ichiga oladi. MySQL MBBT uchun sxema 5.1-jadvalda keltirilgan.

6-jadval. MBning sxemasi

Jadval	Atribut	Turi (Tipi)	O'lcham	Ruxsat berilgan qiymatlar	Standart qiymatlar
PERSON	Nom	INT (auto_increment)	6	E, A	0
	FISH	VARCHAR	30		
	Tsana	DATE			
	Jinsi	VARCHAR	1		
	SumD	DOUBLE			
FLAT	Manzil	VARCHAR	30	>=0 0..99 PP, NP	0 0 PP
	Skv	DOUBLE			
	KomS	INT	2		
	KKat	VARCHAR	2		
TPHONE	Ntel	VARCHAR	10	##-##### O, U	0
	Tkat	VARCHAR	1		
	Manzil	VARCHAR	30		
PROFIT	Id	INT (auto_increment)	6	>=0	0
	Source	VARCHAR	20		
	Money	DOUBLE			
HAVE D	Nom	INT	6	>0	
	Id	INT	6		

Axborotlarning MB ga kiritilishi va kerakli ma'lumotlarning olinishi to'g'ridan-to'g'ri MBBT qurollari yordamida yoki MBBT buyruqlaridan

foydalangan holda maxsus ishlab chiqilgan amaliy tizim yordamida olinadi.

Nazorat savollari

1. Kontseptual loyihalashning zarurligi nimada?
2. Ob'ektlar o'rtasidagi munosabatlarning turlari?
3. Maydonlar uchun qanday cheklash turlari eng ko'p ishlatiladi?
4. Butun jadvalga nisbatan qo'llaniladigan cheklovlar bu nima?
5. Predmet sohaning infologik modeli nima uchun kerak?
6. Datalogik loyihalashning maqsadi nima?
7. ILM dan relyatsion DLMga o'tish uchun qanday qoidalar qo'llaniladi?
8. Yaxlitlik cheklovlari bu nima?

3-BOB. MA'LUMOT MODYELI VA RYELYASION ALGYEBRA

3.1. Ma'lumot modellari

Ma'lumot modeli tushunchasi va sinflanishi

Ma'lumotlarning modeli — bu ma'lumotlar o'zaro bog'langan tuzilishlari va ular ustida bajariladigan operatsiyalar to'plamidir. Modelning shaklli va unda foydalaniladigan ma'lumotlar tuzilishining turi (ma'lumotlar bazalarini boshqarish tizimida unga asos qilib olingan modelning yoki ma'lumotlarga ishlov berish amaliy dasturi yaratiladigan) dasturlash tizimi tilida foydalangan ma'lumotlarni tashkil etish va ishlov berish kontseptsiyasini aks ettiradi.

Malumotlar bazasi tuzilmasining asosi bu ma'lumot modellaridir. U o'z ichiga ma'lumotlarning o'zaro bog'lanishi, ularning tuzulish ketmaketligi va boshqa bir qancha ma'lumotlarni oladi. Model MBda fizik, mantiqiy va ko'rsatish bosqichlarini tasvirlashni o'ziga xos bir yo'li.

Ma'lumot modellari quyidagi 4 ta turkumga bo'linadi:

- *Munosabat modeli.* Jadvallarni va ulardagi ma'lumotlarni o'zaro bog'lanish munosabatlarini tasvirlash uchun foydalaniladi. Har bir jadvalning bir necha ustunlari va har bir ustunning yagona takrorlanmas nomi bo'ladi. Bu model MBni boshqarish tizimida juda keng foydalaniladi.
- *Mohiyat - aloqa modeli.* Bu modeldan asosan obektlarning yig'indisini va ular o'rtasidagi bog'liqlikni tasvirlaydi. Mohiyat bu biror narsa yoki obekt haqiqiy dunyodagi boshqa obektlardan farqlaydi. Bu modeldan MB tuzilmasida keng foydalaniladi.
- *Ob'ekt-baza ma'lumot modeli.* Ob'ektga yo'naltirilgan dasturlash dasturlash metodlari ichida ustuvor hisoblanadi. Ob'ektga yo'naltirilgan ma'lumot modelida tayyor ob'ektlardan foydalaniladi.
- *Yarimtuzilgan ma'lumot modeli.* Bu model ma'lumotlarning o'ziga xos xususiyatlarini aniqlashga imkon beradi. Bu model yuqorida keltirib o'tilgan modellarga zid ishlaydi. Bunda har bir ma'lumot qismini o'ziga xos xususiyati o'rnatilgan bo'lishi kerak.

Ma'lumki, aynan bir axborotni mashina ichki muhitida joylashtirish uchun ma'lumotlarni turli xil tuzilishlari va modellaridan foydalanish mumkin. Ulardan qaysi birini tanlash axborotlar bazasini yaratayotgan foydalanuvchining zimmasiga yuklatilgan bo'lib, u ko'plab omillardan bog'liq. Bu omillar qatoriga mavjud texnik va dasturiy ta'minot

avtomatlashtirilayotgan masalarning murakkabligi va axborotning xajmi kabilar kiradi

Ma'lumotlar modeli quyidagi tarkibiy qismdan iborat:

Foydalanuvchining MBga munosabatini namoyish etishga mo'ljallangan ma'lumotlar tuzilmasi.

Ma'lumotlar tuzilishida bajarilish mumkin bo'lgan operatsiyalar. Ular ko'rib chiqilayotgan ma'lumotlar modeli uchun ma'lumotlar tilining asosini tashkil etadi. Yaxshi ma'lumotlar tuzilmasining o'zigina yetarli emas. Ma'lumotlarni aniqlash tili (MAT) va ma'lumotlar bilan amallar bajarish tilining (MABT) turli operatsiyalari yordamida bu tuzilma bilan ishlash imkoniga ega bo'lish zarur.

Yaxlitlikni nazorat qilish uchun cheklashlar. Ma'lumotlar modeli uning yaxlitligini saqlash va ximoya qilishga imkon beruvchi vositalar bilan ta'minlangan bo'lishi lozim. Quyida Shunday cheklanishlarning namunalari keltirilgan:

a) har bir "kichik daraxt" tugunga ega bo'lishi kerak. Ma'lumotlarning ierarxik bazalarida birlamchi tugunsiz "tug'ma" tugunlarni saqlash mumkin emas.

b) ma'lumotlarning relyatsion bazasiga nisbatan bir xil kortejlar bo'lmaydi. Fayl uchun bu cheklash barcha yozuvlarning yagonaligini talab etadi.

Ma'lumotlarni ierarxik va tarmoqli modellari

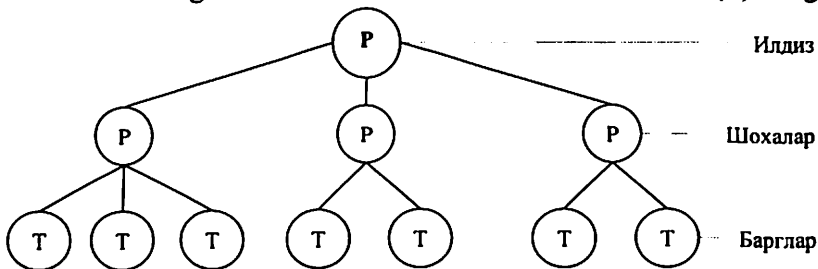
Ierarxik model daraxtsimon graf ko'rinishida berilib, unda ob'ektlar bog'langan satx (ierarxik) ko'rinishida ajratiladi.

Ushbu model ma'lum qoida asosida quriladi. Bu qoidaga muvofiq har bir xosil qilingan tugun (uzel) faqat bir kirishga ega bo'ladi. Strukturada fakat bitta hosil qilinmagan tugun bo'lishi mumkin. Bu tugun ushbu modelning ildizi hisoblanadi. Agar tugunning chiqish strelkasi bo'lmasa u barg deb ataladi. Strukturada bunday barglardan bir nechta bo'lishi mumkin. Tugun yozuv deb interpretatsiya qilinadi. yozuvni qidirish uchun esa ildizdan bargga karab xarakat kilinadi (yuriladi), ya'ni yuqoridan pastga qarab yuriladi. Bunday yondashuv qidirish jarayonini osonlashtiradi (3.1.1-rasmga qaralsin).

Usulning afzallig: strukturani bayon qilishda mantiqiy va fizik satxlardan foydalanish mumkin.

Usulning kamchiligi: ma'lumotlarning elementlari orasidagi va ma'lumotlarning fizik va mantiqiy tashkil qilishdagi o'zaro o'zgarmas bog'likligi tufayli bog'liklikning har qanday o'zgarishi strukturani o'zgarishiga olib keladi.

Yukoridagi birinchi satxda «Etkazib beruvchilar» (P) ning ob'ekti

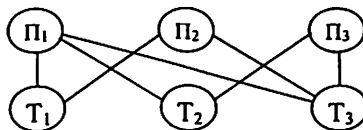


3.1.1-rasm. Ma'lumotlarning ierarxik modeli

to'g'risidagi, ikkinchi satxda har bir P1 , P2 va P3 yetkazib beruvchilar to'g'risidagi, uchinchi satxda esa yetkazib beruvchilarning maxsulotlari (T) to'g'risidagi axborotlar turadi.

Ierarxik model elementlari o'rtasidagi bog'lanishda «bittadan ko'pchilikka» usuli qo'llaniladi. Agar elementlar o'rtasidagi bog'lanishlarda «ko'pchilikdan ko'pchilikka» usuli qo'llanilsa, ma'lumotlarning tarmoqli modeli hosil bo'ladi.

Tarmoqli modelda yozuvlar o'rtasida ixtiyoriy bog'lanish turlari qo'llanishi mumkin bo'lib, «teskari bog'lanish» uchun cheklanish qo'yilmagan (3.1.2-rasm).



3.1.2-rasm. Ma'lumotlarning tarmoqli modeli

Tarmoq MB ni bayon qilishda ikki turdagi guruxlar qo'llaniladi: «yozuv» va «aloqa». Aloqa» turi ikki turdagi «yozuv» larni aniqlaydi: ajdod va avlod. O'zgaruvchi turidagi «aloqa» aloqaning nusxasidir. Tarmoq modelida bitta qoida qo'llaniladi: aloqa asosiy va bog'liqli yozuvlarni o'z tarkibiga oladi.

Usulning afzalligi: ierarxik modelga nisbatan kattagina axborotli moslashuchanligi.

Usulning kamchiligi: MB sxemasining moslanuvchan emasligi va o'ta murakkabligi, hamda oddiy foydalanuvchilar uchun MB dagi axborotlarni ishlash va tushunish qiyinligidadir.

Mashina muhitidagi ma'lumotlarning murakkabroq modellari, fayl modeliga nisbatan, tarmoqli va ierarxik modeli hisoblanadi. Bu modellar ularning o'zlariga xos turdagi MBni boshqarish tizimida ishlatiladi. Mashina "tashuvchisi"dagi MBBTga asos qilib olingan ma'lumotlar modelining turi MBBTni tasnifiy belgilarining eng muhimlaridan biri hisoblanadi. MBBTda ma'lumotlarni mantiqiy tashkil etish usuli ma'lumotlarning tarmoqli yoki ierarxik modeliga mos holda ko'rsatiladi. Bunday model o'zaro bog'liq ob'ektlarning majmuidir. Ikki ob'ektning aloqasi ularning bir-biriga tobeligini aks ettiradi. Tarmoqli yoki ierarxik modelida ob'ekt bo'lib, MBBT kiritilgan ma'lumotlar tuzilmasining asosiy turlari hisoblanadi. Turli MBBTlarda bu turdagi ma'lumotlarning tuzilmasi turlicha aniqlanishi va nomlanishi mumkin (yozuv turi, fayl, segment).

Modellarda ma'lumotlarning tuzilmalari. Ma'lumotlarning namunaviy tuzilmalariga quyidagilar kiradi: ma'lumotlarning elementi, ma'lumotlarning agregati, yozuv, MB va x.k. U elementlari va agregatlari o'zaro aloqada bo'lgan tuzilma bilan tavsiflanadi. Shuning uchun yozuvning tuzilmasi ierarxik xarakterga ega bo'lishi mumkin. Bir xil tuzilmaga ega bo'lgan yozuv nusxalari to'plamining hammasi yozuv turini tashkil etadi.

Ob'ektlarning modellardagi aloqasi. Ma'lumotlar modeli bir necha turidagi yozuvlarni (obektlarni) o'z ichiga olishi mumkin. Ma'lumot modelning ob'ektlar o'rtasida aloqalar o'rnatiladi. Qandaydir bir predmet soxasi uchun modelning o'zaro bog'langan muayyan ob'ektlar to'plami MBni tashkil qiladi.

Ikki turdagi yozuvlarning (model ob'ektlari) o'rtasidagi aloqalar, ularning nusxalari o'rtasidagi gurux munosabatlari bilan aniqlanadi. Gurux munosabati — bu ikki turdagi yozuvlar o'rtasidagi kat'iy ierarxik munosabat bo'lib, ular asosiy yozuvlar to'plami va tobe yozuvlar to'plamidan iborat.

Ierarxik modellarda kalit bo'yicha bevosita kirish odatda, faqat boshqa ob'ektlarga tobe bo'lmagan eng yuqori pog'onadagi ob'ektgagina

mumkin. Boshqa ob'ektlarga kirish modelning eng yuqori pog'onasidagi ob'ektdan aloqalar bo'yicha amalga oshiriladi. Tarmoqli modellarda esa kalit bo'yicha bevosita ixtiyoriy ob'ektga kirish (uning modelda joylashgan pog'onasidan qa'tiy nazar) ta'minlanishi mumkin. Shuningdek, aloqalar bo'yicha har qanday nuqtadan kirish ham mumkin. Tarmoqli modellarda ob'ekt (yozuv, fayl)ning tuzilmasi ko'pincha chizikli va kamroq hollarda esa ierarxik bo'ladi. Quyi pog'onadagi ma'lumotlarning tuzilmasi ham uz xususiyatga va nomiga ega bo'lishi mumkin. Masalan, atribut bu ma'lumotlar elementining analogi. Chizikli tuzilmaga ega bo'lgan ob'ekt fakat oddiy va kalit atributlardan iborat. Ierarxik modellardagi ob'ekt (yozuv, segment) tuzilmasi ierarxik yoki chizikli bo'lishi mumkin.

Turli predmet soxalari uchun ma'lumotlarning tarmoqli modeli ierarxik modeliga nisbatan mashinaning ish muxitida axborot tuzilmalarini aks ettiruvchi umumiy vosita hisoblanadi. Ko'plab predmet soxalarining malumotlari urtasidagi aloqalar tarmoqli ko'rinishga ega. Bu esa ma'lumotlarning ierarxik modeliga ega bo'lgan MBBTdan foydalanishni cheklab qo'yadi. Tarmoqli modellar, ma'lumotlarning ierarxik aloqasini ham aks ettirishga imkon beradi. Bundan tashqari, tarmoqli modellar bilan ishlash texnologiyasi foydalanuvchi uchun kulaydir, chunki ma'lumotlarga kirishni amalga oshirishda xech qanday cheklashlar yo'q va bevosita ixtiyoriy pog'onadagi ob'ektlarga kirish imkoni mavjud.

Ma'lumotlarni modellash

Ma'lumki, ma'lumotlarning talqini bilan birga ular orasidagi munosabatlarni tasvirlash imkoniyatlari ham ta'minlanishi zarur. Ma'lumotlarni berilgan talablarga mos tarzda talqin etilishning intellektual vositalariga *ma'lumotlar modeli* deyiladi. Ma'lumotlar modeli – bu ma'lumotlarni abstraktlash vositasidir. U ma'lumotlarning muayyan qiymatlarini emas, balki ularning axborot mazmunini tasvirlash imkonini beradi. Boshqacha aytganda, ma'lumotlar modeli o'rmonni alohida olingan daraxtlar tarzida emas, balki yaxlit o'rmon tarzida tasvirlash imkoniyatini taqdim etuvchi vositadir.

Mavjud modellar tashqi dunyoning turli qirralarini tasvirlashga mo'ljallangan. Fizik modellar fizik xususiyatlarni o'rganish, matematik modellar matematik ifoda va formulalar yordamida ifodalash, iqtisodiy

modellar iqtisodiyot an'analarini tasvirlash va uning taraqqiyotini bashorat qilish imkonini beradi. Ma'lumotlar modellari esa ma'lumotlarning xususiy semantikasini tasvirlash imkonini beradi hamda bizni tashqi dunyo haqidagi o'ziga xos bilimlar bilan qurollantirishga xizmat qiladi.

Ma'lumotlar modeli tushunchasini oydinlashtirish uchun modellashning elementar ob'ektlariga, taxminan bo'lsada, aniqlik kiritish zarur. Buning uchun, ma'lumotlarning elementar birligi hisoblangan, *<ob'ekt nomi, ob'ekt xususiyati, xususiyat qiymati, vaqt>* tarzida ifodalanadigan kortejni qaraylik. Odatda hodisa (yoki g'oya) biron ob'ekt (*ob'ekt nomi*) va uning qandaydir xususiyatiga (*ob'ekt xususiyati*) bog'liq holda ro'y beradi, bunda ob'ekt xususiyati ma'lum bir vaqtda (*vaqt*) muayyan bir qiymatga (*xususiyat qiymati*) ega bo'ladi.

Ushbu to'rt xususiyatdan «vaqt»ni hisobga olish ko'pincha qiyinchilik tug'diradi. Ko'pgina hollarda ma'lumotlar qayd etilishining vaqtini ko'rsatmasdan, ularni tartibini keltirishning o'zi kifoya yoki eng so'nggi ma'lumotlarnigina taqdim etish yetarli bo'ladi. Shuning uchun ko'pchilik ma'lumotlar modellarida boshqa xususiyatlar kiritilishi yoxud ob'ektlarni tartibga solinishi evaziga vaqt unsuri chiqarib tashlanadi.

Vaqt unsuri e'tiborga olinmagani holda ma'lumotlarning elementar birligini quyidagicha tasvirlash mumkin: *<ob'ekt nomi, ob'ekt xususiyati, xususiyat qiymati>*. Ushbu ifoda turli usullar bilan amalga oshirilishi mumkin. Bu esa ko'plab ma'lumotlar modellarining yaratilishiga sabab bo'lgan. Elementar ma'lumotlar va ular orasidagi aloqalarni tasvirlashning usullaridan biri tarmoqli tuzilma hisoblanadi. Bunda graf cho'qqilari ma'lumotlar elementlariga, yoylari esa ular orasidagi aloqalarga mos keladi.

Ma'lumotlar modellariga doir qo'shimcha tushunchalar

Avtomatlashtirilgan tizimlarda qo'llaniladigan ma'lumotlar modellarining ko'pchiligi kuchli tiplashtirilgan modellar hisoblanadi.

Ma'lumotlar modeli ma'lumotlar tuzilmasi qurilishini belgilab beruvchi qoidalardan iborat. Biroq tuzilma bo'yicha belgilashlar ma'lumotlar semantikasining talqini va ulardan foydalanish usulini to'liq ifodalash imkoniyatini ta'minlab bera olmaydi. Bundan tashqari, ma'lumotlar ustida bajariladigan amallar uchun ham belgilashlar zarur bo'ladi va ular odatda ma'lumotlar tuzilmasi bilan birga keltiriladi, deb hisoblanadi.

Ushbu nuqtai nazardan, MB uchun boshqacharoq ta'rif berish mumkin, ya'ni tuzilmasi muayyan tarx (sxema)ga mos keluvchi

ma'lumotlar jamlanmasiga *ma'lumotlar bazasi* deb ataladi.¹ Ushbu atama berilgan ma'lumotlar jamlanmasining muayyan holatiga hamda bu holatdan kelib chiqadigan boshqa holatlari uchun ham kuchga ega. Gap shundaki, ma'lumotlar modeli tomonidan ko'zda tutilgan amallar MBni bir holatdan boshqa holatga o'zgartiradi. O'zgartirishlar natijasida hosil qilinadigan ma'lumotlar bazalari ketma-ketligini ham MB deb nomlashadi.

Ma'lumotlar modellarining turfaligi ularning qo'llanilish sohalari va foydalanuvchilar kontingentining xilma-xilligiga mos keladi. Ma'lumotlar modellari uchun umumiy bo'lgan qator tushuncha va ta'riflarni qarab chiqaylik.

Xohlagan ma'lumot ob'ekti va u tasvirlaydigan xususiyatlarni olib qaraydigan bo'lsak, ob'ekt xususiyatlari asosan ikki sinfga: *statik va dinamik* xususiyatlarga bo'linadi. *Statik xususiyatlarga* vaqt bo'yicha o'zgarmas, invariant bo'lgan xususiyatlar kiradi. *Dinamik xususiyatlar* esa tashqi dunyoning evolyutsion tabiatiga mos keladi. Bundan kelib chiqib, ma'lumotlar modeli M ni vujudga keltirish qoidalari G va bajariladigan amallar O to'plamlari tarzida belgilash mumkin.

Vujudga keltirish qoidalari to'plami ma'lumotlar modelining statik xususiyatlarini ifodalaydi va *ma'lumotlarni ifodalash tili* sifatida qo'llaniladi.

Ma'lumotlarni ifodalash tili vositalari yordamida ma'lumotlarning tuzilmasi, ob'ektlari, aloqalari hamda ularni amalga oshirishning mumkin bo'lgan variantlari belgilanadi. Ma'lumotlar tuzilmasini aniqlash vujudga keltirish qoidalarini qanoatlantiruvchi, mos toifalar uchun qabul qilingan belgilashlar yordamida amalga oshiriladi. Muayyan ma'lumot toifasi uchun mazkur belgilashlar, ma'lumot toifasining atributlari tarkibi va har bir atributi uchun ma'lumot formatlariga qo'yiladigan belgilashlardan iborat bo'ladi. Ob'ektlar qabul qilishi mumkin bo'lgan qiymatlarini yoki aloqalarning mumkin bo'lgan variantlarini belgilash, ma'lumotlarning har bir toifasi uchun ko'rsatiladigan *butunlik cheklovlari* yordamida amalga oshiriladi. Bunga misol tariqasida soliq to'lovchining qayd raqamini ko'rsatish mumkin. Mazkur raqam har bir xodim uchun betakror (unikal) bo'lishi hamda uni qaytarilmasligini ta'minlash zarur.

Ba'zi ma'lumotlar modellarida vujudga keltirish qoidalari G ni ikkiga bo'lishadi: tuzilmalarni vujudga keltirish qoidalari – G_s va cheklovlarni vujudga keltirish qoidalari – G_c . Bunga mos tarzda S tarx ham ikki qismdan: tuzilmalar S_s bo'yicha belgilashlar va *oshkor ko'rsatilgan cheklovlar* S_c bo'yicha belgilashlardan iborat bo'ladi. «Xodim» tipli mohiyatning «Xodim qayd raqami» atributini kalit (identifikator)

ekanligini ko'rsatish oshkor ko'rsatilgan cheklovga misol bo'ladi. Bu – «Xodim» tipli mohiyatda bironta qayd raqami ikki yoki undan ortiq kortejlarda bir xil qiymatga ega bo'lishi mumkin emasligini belgilaydi.

Ma'lumotlar modellarida oshkor cheklovlar bilan bir qatorda, modelning tuzilma qismi (S_s)da keltiriladigan *ichki cheklovlar* ham qo'llaniladi. Bu cheklovlar ob'ektlar va aloqalarga xos bo'ladi. Masalan, ma'lumot ob'ektlari orasidagi aloqalar shajara ko'rinishidagi tuzilma bilan chegaralanishi mumkin.

G to'plam qoidalari, har biri muayyan ma'lumotlar tuzilmasi va butunlik cheklovlarini belgilab beradigan, S tarxlar to'plami hosil qilinishini ta'minlaydi. G to'plam ko'rinishidagi vujudga keltirish qoidalari ma'lumotlarning statik xususiyatlari uchun belgilashlar kiritish maqsadida qo'llaniladi va ma'lumot ob'ektlarining statik xususiyatlarini ifodalaydi. Ma'lumot ob'ektlarining dinamik xususiyatlari amallar to'plami O bilan ifodalanadi va *ma'lumotlar bilan amallar bajarish tili* sifatida qo'llaniladi. Amallar to'plami MBni bir ko'rinishdan ikkinchisiga o'tkazish maqsadida qo'llanilishi mumkin bo'lgan sa'yi-harakatlarni belgilab beradi.

Hamma amallar ham MBning ko'rinishini o'zgarishiga olib kelmaydi. Chunki, MB dinamikasini joriy yozuv ko'rsatkichi (indikator) va shunga o'xshash boshqaruv elementlari bilan ham bog'lash mumkin. Mazkur elementlar MB ob'ektlari hisoblanmaydi, biroq ular MB ko'rinishi bilan bog'liq va ular ma'lum amallar bajarilishi natijasida o'zgarishi mumkin. Ular MBning muayyan ko'rinishi bilan birgalikda *MBning holatini* belgilab beradi. Ma'lumotlarning dinamik jihatlar MBning holati o'zgarishi orqali tasvirlanadi. Masalan, MBda «keyingi yozuvga o'tish» buyrug'i qo'llanilgandan so'ng, MBning holati o'zgarmaydi, balki uning ko'rinishi o'zgaradi. Bunga joriy yozuv ko'rsatkichining o'zgarishi sabab bo'ladi. O to'plamning har bir amali MBni bir holatdan ikkinchi holatga o'zgartiradi. Bunda MBning tuzilmasi o'zgarishsiz qoladi.

O to'plam amallarini MB holatlari to'plamida aniqlangan funktsiyalar sifatida qarash mumkin. Ba'zida MB tarxi bilan belgilangan tuzilmani saqlab qolish imkonini beruvchi hamda ba'zi oshkor ko'rsatilgan butunlik cheklovlarini buzilishiga olib keluvchi MB ko'rinishlariga o'tishga yo'l beriladi. Bunda bajariladigan amal natijasi noaniq bo'ladi. Masalan, dastur vositalari yordamida kortej qiymati o'zgartirilayotganda oshkor ko'rsatilgan butunlik cheklovi buziladigan bo'lsa, kortej qiymati noaniq yoki bo'm-bo'sh qoldiriladi. Shu bilan birga ichki cheklovlarning

buzilishiga yo'l berilmaydi va ularga rioya qilinishi avtomatik tarzda saqlanadi.

MBBT MBni ifodalash vositalari va MB ob'ektlari uchun belgilangan amallar bajarilishini ta'minlaydi. Boshqacha aytganda, MBBT dasturiy-mantiqiy vositalardan iborat bo'lib, u ma'lumotlarni saqlash tizimini va bazaga ma'lumotlar kiritish, ularni o'zgartirish va saralash vositalarini taqdim etadi. Albatta, MBBT biror ma'lumotlar modeliga tayanadi. Ba'zi MBBTlar esa ma'lum bir ma'lumotlar modeliga moslab ham yaratilgan.

Ma'lumot modelini rivojlanish kontseptsiyasi 5 ta bosqichni ko'rsatishi mumkin:

1. 60- yillarning 2 – yarmida, bunda asosan ierarxik modellarga e'tibor berilgan;
2. 70- yillarning 1 – yarmi, tarmoqli modellar;
3. 70- yillarning 2 – yarmi, relyatsion modellar;
4. 80- yillarning 1 – yarmi, semantik modellar;
5. 80- yillarning 2 – yarmi, ob'ektga mo'ljallangan tizim.

Nazorat savollari

1. Ma'lumotlar modeli va uning turkumlari.
1. Tarmoq modeli amaliyotda qo'llaniladigan o'rinlariga misollar.
2. Ierarxik va tarmoqli modellarini birgalikda qo'llash mumkinmi?
3. Ma'lumotlarni modellashtirish deganda nimani tushunasiz?

3.2. Relyatsion ma'lumotlar modeli va relyatsion algebra

Tushunchalar

Ma'lumotlarning relyatsion modeli 1970 yil IBM firmasining xodimi Edgar Kodd tomonidan taklif etilgan. Relyatsion MBning asosiy g'oyasi MB strukturasi soddalashtirishga qaratilgan. Unda ierarxik modeldagi kabi avlod va ajdodlarga yaqqol ko'rsatgichlar yo'q, barcha ma'lumotlar star va ustunlarga bo'lingan oddiy jadvallarda tasvirlanadi. Relyatsion modeldagi asosiy tushuncha "munosabat" (relation) hisoblanadi.

Ma'lumotlarning shunday modeliga *relyatsion model* deyiladi, unda barcha ma'lumotlar foydalanuvchiga jadval shaklida xavola etiladi va MB ustidagi barcha amallar jadvallar ustiga amallarga olib kelinadi. Har bir jadval MB tarkibida o'z nomiga ega bo'ladi, hamda satr va ustunlardan iborat bo'ladi. Har bir jadval real dunyodagi ob'ektlar (mohiyat) tipini aks ettiradi, har bir satri esa ob'ektning konkret nusxasini bildiradi.

Relyatsion munosabat jadval tarzida ifodalanib, jadvalning har bir satri *kortejni*, ustunlar soni munosabatning *ko'rsatkichi (darajasi)*, satrlari soni *quvvati (kardinal soni)* hisoblanadi. Jadval ustunini – *atribut*, ustun sarlavhasini – *atribut nomi*, sarlavha ostidagi elementini – *atribut qiymati* deb atashadi.

MBda munosabat (jadval) nomi va undagi atribut(ustun)lar o'z nomiga ega bo'ladi hamda bu nomlanishi betakror (qaytarilmas) bo'lishi talab etiladi (atributlarning nomlanishi boshqa munosabatlarda qaytarilishi mumkin, ya'ni bitta jadvaldagi ustunning nom boshqa jadvalning ustun nomi sifatida ishlatilishi mumkin). Ixtiyoriy jadval kamida bitta ustunga ega bo'ladi; ustunlar jadvalda ularni yaratish tartibida joylashadi. Satrlar ustunlardan (atributlar) farqli ravishda nomga va joylashish tartibiga ega emas. Ularning miqdori chegaralanmagan.

Relyatsion MBdagi asosiy tushunchalarga quyidagilar kiradi.

Ma'lumotlar tipi – bu tushuncha dasturlash tillarida xuddi shu ma'noda qo'llaniladigan tushunchaga mos keladi. Hozirda mavjud bo'lgan barcha zamonaviy ma'lumotlar bazalari maxsus ma'lumotlar tiplarini ishlatish imkonini beradi. Bular – butun va suzuvchi vergulli kasr sonlar, simvollar va matn satrlari, mantiqiy ifoda, sana tipidagi ma'lumotlarni saqlashga mo'ljallangan ma'lumotlar tiplaridir. Ko'pgina ma'lumotlar bazalarining serverlarida boshqa ma'lumotlar tiplari ham amalga oshirilgan. Masalan, binar axborotlarning yirik massivini saqlashga qaratilgan (BLOB), joriy etiladigan ob'ektlar uchun (OLE-ob'ekt) mo'ljallangan ma'lumotlar tiplari shular jumlasidandir.

Domen – ma'lumotlar tipi qiymatlarining potentsial to'plamidir. Domen ikki element – ma'lumotlar tipi va ma'lumotlarga nisbatan qo'llaniladigan mantiqiy ifoda bilan belgilanadi. Agarda ushbu mantiqiy ifodaning qiymati «to'g'ri (chin)» natijani taqdim etsa, ma'lumotlar nusxasi domenga taalluqli bo'ladi.

Munosabat – sarlavha va unga mos ma'lumotlardan iborat bo'lgan ikki o'lchamli jadval.

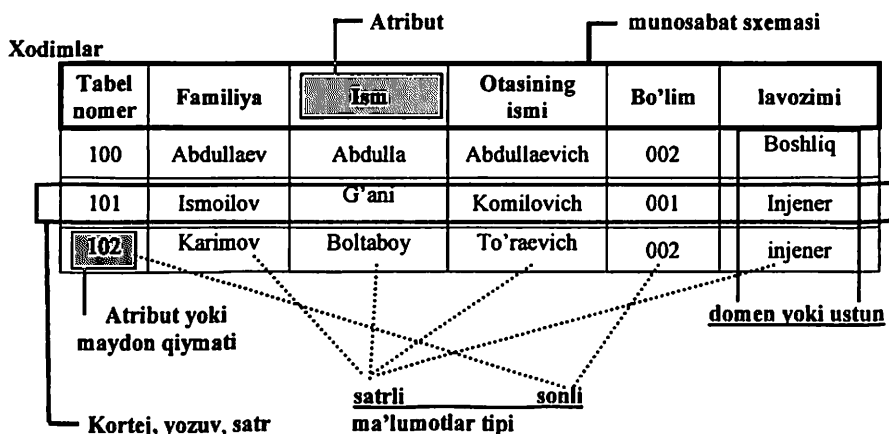
Sarlavha – chekli sondagi atributlar to'plami bo'lib, har bir atribut biror bir domenda aniqlangan hamda atributlar va domenlar orasida qat'iy o'zaro moslik mavjud bo'ladi.

Keltirilgan nazariy tushunchalar relyatsion MBBTning til vositalari va dasturiy tizimlarini ishlab chiqishda qo'llaniladi. Amaliy ish jarayonida esa ular o'rninga ularning quyidagi noformal ekvivalentlari ishlatiladi:

- munosabat – jadval;
- atribut – kolonka, ustun yoki maydon;
- kortej – yozuv, zavis yoki ma'lumotlar qatori (3.2.1-rasm).

Relyatsion yondashuv asoschilaridan biri Deytning ko'rsatishicha, relyatsion model uch qismdan iborat bo'ladi:

- tuzilma qismi;
- bajariladigan amallar (manipulyatsiya) qismi;
- yaxlitlik qismi.



3.2.1-rasm. "Xodimlar" ob'ekti xaqidagi jadval

Modelning *tuzilma qismida* munosabatlar aniqlab olinadi. Relyatsion modelda mazkur munosabatlardan ma'lumotlarning yagona tuzilmasi sifatida foydalaniladi. Relyatsion MB barcha kerakli axborotni o'z ichiga

olgan hamda turli aloqalar bilan birlashtirilgan munosabatlar jamlanmasidan iborat.

Relyatsion modelning *bajariladigan amallar qismida* relyatsion bazalar bilan bajariladigan amallarning ikki asosiy mexanizmi – relyatsion algebra va relyatsion hisob (ischislenie) belgilanadi. Bunda, ma'lumotlarga ishlov berishda to'plamlar nazariyasi elementlari, ya'ni to'plamlarning birlashmasi, kesishmasi, ayirmasi va dekart ko'paytmalaridan foydalanish kabi amallar nazarda tutiladi.

Relyatsion modelning *yaxlitlik qismi* deganda ma'lumotlarni buzilmasligini ta'minlashga qaratilgan biror mexanizm tushuniladi. Yaxlitlik qismi relyatsion MBing butunligiga qo'yiladigan ikki asosiy talabdan iborat bo'ladi, ya'ni *mohiyatlarning butunligi* va *aloqalar bo'yicha butunlik*.

Mohiyatlar butunligi talabi xohlagan munosabatning xohlagan bir korteji uning boshqa xohlagan kortejidan farqli bo'lishi shartligini ifodalovchi talabdan iborat.

Ushbu talab munosabatning tabiatiga mos keladi, ya'ni munosabat to'plam sifatida ifodalanadi. To'plamlar nazariyasida keltiriladigan ta'rifga ko'ra, to'plamda bir-biri bilan mos tushadigan, aynan bir xil elementlar mavjud bo'lmasligi zarur. Demak, munosabatda ham aynan bir xil kortejlar bo'lishi mumkin emas. Shuning uchun mazkur munosabatda har bir kortejni qat'iy belgilab bera oladigan (identifikatsiya qiladigan) atribut yoki atributlar mavjud bo'ladi.

Bu atribut yoki atributlar *kalit* deb nomlanadi.

Kalit quyidagi talablarga javob berishi zarur:

- *qaytarilmas* (betakror) bo'lishi;
- *minimal* bo'lishi, ya'ni kalitdan xohlagan bir atributning chiqarib tashlanishi kalitning betakrorligini buzilishiga olib kelishi lozim.

Kalitdagi atributlar soni mazkur munosabatning ko'rsatkichidan kichik bo'ladi. Biroq, munosabatning barcha atributlarini kombinatsiyasigina betakrorlikni ta'minlaydigan bo'lsa, kalit munosabatning barcha atributlaridan ham tashkil topishi mumkin. Odatda munosabat bir necha kalitlarga ega bo'ladi. Bular ehtimoldagi kalitlar deb ham ataladi. Munosabatning barcha kalitlari orasidan bittasi birlamchi kalit sifatida tanlab olinadi. Birlamchi kalitni tanlashda kamroq sonli atributlarga ega kalitga ahamiyat qaratiladi yoki uzun matn qatorlaridan tashkil topgan kalitni birlamchi qilib olish maqsadga muvofiq hisoblanmaydi.

Kalitlar ham bir nechta turlarga ajratiladi (3.2.2-rasm):

- *birlamchi kalit* – jadvaldagi biror noyob qiymatli atributga o'rnatiladi;
- *ikkilamchi kalit* (tashqi kalit) – boshqa jadvalning birlamchi kalit o'rnatilgan atributidagi qiymatlaridan iborat atribut. Bunda qaralayotgan jadvalda kalit bo'lmagan atribut boshqa jadvalda kalit bo'ladi. Agar jadval tashqi kalitga ega bo'lsa u xolda: *a)* mos birlamchi kalitga ega bo'lgan jadval bilan mantiqan bog'langan; *b)* bu bog'lanish birga ko'p xarakterga bo'ladi.;
- *rekursiv kalit* – jadvalning birlamchi kaliti o'rnatilgan atributi qiymatini shu jadvalning boshqa atributiga qabul qilishi. Bu xuddi ikkilamchi kalitga o'xshaydi. Bunday jadvalda birinchi yozuvda rekursiv kalit o'rnatilgan maydon qiymati NULL qiymat oladi.

Amaliyotda birlamchi kalit sifatida ko'pincha raqamli maxsus atribut qo'llaniladi. Bu atribut avtoinkrement maydon yoki hisoblagich (schyotchik) kabi nomlar bilan ataladi. Uning qiymati trigger yoki MBBT mexanizmi belgilab beradigan boshqa maxsus vositalar tomonidan hosil qilinadi. Jadvallar o'rtasidagi aloqa relyatsion ma'lumotlar modelining asosiy elementi hisoblanadi. Bu aloqa tashqi kalitlar yordamida qo'llab – quvvatlanadi.

Mohiyatlar butunligi talabi birlamchi kalit yordamida amalga oshiriladi hamda zarur ma'lumotlarga yetishishda ushbu kalit qiymatidan foydalaniladi. Barcha MBBTlarida birlamchi kalit orqali ma'lumotlarga yetishishni tezlashtirish uchun maxsus, *indekslash* deb nomlanuvchi mexanizm mavjud. Indeks, qo'pol qilib aytganda, daraxt ko'rinishiga ega invert (teskari) ro'yxat tarzida ifodalanadi va u har bir birlamchi kalit uchun yozuvning joylashgan haqiqiy o'rnini ko'rsatib beradi. Invert ro'yxatda indeks sifatida ko'rsatilgan atribut yoki atributlar guruhining har bir qiymati uchun mos birlamchi kalit qiymati (yoki bir necha mos qiymatlar) keltiriladi. Tabiiyki, turli MBBTlarda indekslar turlicha (masalan, lokal MBBTlarida alohida fayllar sifatida) amalga oshirilgan, biroq indekslash tamoyillari barcha MBBTlar uchun bir xil.

Munosabatni birlamchi kalitdan farq qiluvchi atributlar tarkibidan foydalanib indekslash ham mumkin. Indeksning mazkur tipi ikkilamchi indeks deb nomlanadi va munosabatdagi ma'lumotlarga yetishish vaqtini kamaytirish hamda yozuvlarni tartiblash maqsadlarida qo'llaniladi. Shunday qilib, munosabat biror tarzda tartiblanmagan bo'lsada, indeks (lokal MBBTlarda indeks fayli) doimo tartiblangan yozuvlardan iborat bo'ladi.

Boshliqlar

№	Familiya	Bo'lim	Staj
5742	Ivanov	SAPR	12
6931	Petrov	№5	25
2345	Sidorov	laboratoriya	21

Бирламчи**Xodimlar**

Xodim kod	Familiya	№ boshliq	Lavozim
4781	Ivanov	5742	M.n.s
5325	Petrov	6931	S.p.s
3120	Sidorov	5742	P.s
1230	Yashin	2345	Inj
2138	Yudin	6931	Bosh inj

Иккиламчи**3.2.2-rasm. Munosabat kalitlariga namuna**

Aloqalar bo'yicha butunlik talabi – bu MBdagi munosabatlar bir-biri bilan aloqalar orqali bog'langanligi oqibatida vujudga keladigan hamda relyatsion MBga qo'yiladigan cheklovdan iborat.

Aloqalar bo'yicha butunlik talabining mohiyati quyidagicha. Aloqadagi munosabatda yuzaga keladigan tashqi kalitning har bir qiymati uchun aloqa ko'rsatkichi olib boradigan munosabatda xuddi shunday qiymatli birlamchi kalitga ega kortej mavjud bo'lishi shart.

Ma'lumotlarning aloqalari butunligini ta'minlash uchun ko'pgina MBBTlarda tashqi kalitlar deb nomlanuvchi mexanizm mavjud. Bu mexanizmning mohiyati shundan iboratki, biror munosabatning bir atributiga (yoki atributlari guruhiga) boshqa munosabatning birlamchi kaliti bilan bog'liqlik (ko'rsatkich) belgilab qo'yiladi hamda bu holat ushbu munosabatlar orasida tobelik aloqalarini vujudga keltiradi. Bunda, birlamchi kalitiga boshqa munosabatning tashqi kalitidan ko'rsatkich keladigan munosabatga asosiy yoki master-munosabat, ko'rsatkich chiqib ketadigan munosabatga tobe yoki detail-munosabat deyiladi. Bunday ko'rsatkich belgilab qo'yilgandan so'ng MBBT aloqalarning «buzilmasligi»ni avtomatik tarzda nazorat qilish imkoniga ega bo'ladi. Bunda quyidagi amallar nazorat qilinadi:

- tobe jadvalga asosiy jadvalda mavjud bo'lmagan birlamchi kalit qiymatiga mos keluvchi yozuvni kiritishga urinish;
- asosiy jadvaldagi, tobe jadvalning hech bo'lmaganda bitta yozuvidan ko'rsatkich mavjud bo'lgan yozuvini yo'qotishga harakat qilish;
- asosiy jadvaldagi, tobe jadvalning hech bo'lmaganda bitta yozuvidan ko'rsatkich mavjud bo'lgan yozuvi birlamchi kaliti qiymatini o'zgartirishga urinish.

Amalda asosiy jadval yozuvlarini yo'qotish va o'zgartirish uchun quyidagi ikki yondashuvning biridan foydalaniladi:

- asosiy jadvaldagi, tobe jadvalning hech bo'lmaganda bitta yozuvidan ko'rsatkich mavjud bo'lgan barcha yozuvlarini yo'qotilishi hamda asosiy jadval birlamchi kaliti qiymatlarini o'zgartirilishini ma'n qilish;
- asosiy jadval birlamchi kaliti qiymatidagi barcha o'zgarishlarni tobe jadval yozuvlariga ham joriy etish, ya'ni:
 - o agarda asosiy jadvaldagi biror yozuv yo'qotiladigan bo'lsa, tobe jadvaldagi ushbu yozuvga ko'rsatkich hosil qilgan barcha yozuvlar ham yo'qotilishi lozim;
 - o agarda asosiy jadvaldagi yozuvning birlamchi kaliti qiymati o'zgartiriladigan bo'lsa, tobe jadvaldagi ushbu yozuvga ko'rsatkich hosil qilgan barcha yozuvlarning tashqi kalitlari ham mos tarzda o'zgartirilishi lozim.

Jadvalli munosabatni ba'zan sxematik ravishda ko'rsatiladi. Bunda munosabat nomi yokiga qavs ichida atributlarni nomi yoziladi.:

Talaba (shaxsiy nomer, FIO, to'g'ilgan sana, kurs mutaxassislik)

Relyatsion MB munosabatlarni o'zaro bog'langan to'plamidir. Har bir munosabat kompyuterda fayl ko'rinishida tasvirlanadi.

7-jadval

Fayl	Jadval	Munosabat	Mohiyat
Yozuv	Satr	Kortej	Mohiyat nushasi
Maydon	Ustun	Atribut	Atribut

Munosabatlar jadval ko'rinishida berilganda quyidagi xossalarga ega bo'ladi:

1. Jadvalni har bir elementi ma'lumot elementi hisoblanadi va takrorlanuvchi guruxlar bo'lmaydi;
2. Barcha ustunlar jadvalda bir jinlidir;
3. Har bir ustunda nom tayinlangan;
4. Jadvalda bir xil satr ikki marta uchramaydi;
5. Bunday jadvallarda satr va ustunlar ixtiyoriy tartibda qaraladi va ixtiyoriy ketma – ketlikda ishlatilishi mumkin.

Bunday xususiyatli jadvallar munosabat deb kelishilgan (otnosheniya - relation). Munosabat asosida kurilgan MB relyatsion MB deyiladi. Misol uchun jadvalni sxematik kiskartirilgan kurinishi (sxemasi) ***Xizmatchi*** (*xiz.nom F.I,Sh, unvoni, tugilgan yili, bulim, mut.kodi. mansab, maosh*); Bunday yozuv ma'lumot ***baza sxemasi*** deyiladi Shunday kilib, relyatsion MB ma'lumot elementlar tuplami asosida kuriladi. Munosabat yoki

jadvalni kortejlar tuplami deb karash mumkin. Agar jadvalda n ta ustun balsa, u n tartibli kortejdan iborat deyiladi va munosabat xam n -darajali deyiladi. Xar bir atribut qiymatlari tuplami domen deyiladi.

Relyatsion algebra va uning amallari

Relyatsion algebra qo'shish, ayirish yoki ko'paytirish kabi odatiy algebraik operatsiyalarni to'plamlar ustida amalga oshirish imkonini beradi. Sonlar ustida algebraik operatsiyalar bir va undan ortiq sonlarni qabul qilib son qiymatini qaytaradi, relyatsion algebra bir yoki undan ortiq to'plamlarni qabul qilib yagona to'plamni qaytaradi. Relyatsion MBBTda ma'lumotlar bilan ishlash uchun bir qancha tillar yaratilgan. Ba'zi hollarda bu tillarni ma'lumotlarni qism tillari deb ataladi. MB bilan ishlovchilar bu tillarda avtomatlashtirishni 3 bosqichga bo'lishadi:

- 1) Eng pastki bosqich – kortej deb ataladi. Bunda dasturchi yozuvlar yoki kartijlar bilan ishlaydi.
- 2) Relyatsion algebra deyiladi. Bunda foydalanuvchi munosabatlar ustida yuqori bosqichli amallar to'plamini kiritadi.
- 3) Eng yuqori bosqich – hisoblash bosqichi. Bunda foydalanuvchi bevosita kompyuterga maxsus tillarda murojaat qiladi va mashina bu murojaatni qabul qiladi.

Relyatsion MBBT da ma'lumotlar bilan ishlashda ishlatiladigan 2ta katta gurux tillari relyatsion hisoblash deyiladi. Relyatsion hisoblash predikatlarni hisoblashga asoslangan bo'lib ifodalarni yozishga mo'ljallangan qoidalar to'plamidan iboratdir. Ular yordamida biz mavjud munosabatlardan yangi munosabatlar yaratishni ta'minlaymiz. Bunday ifodaalrni yozishda solishtirish amallari, mantiqiy amallar va mavjudlik kvanteri va umumiylik kvanteri ishlatiladi.

Relyatsion algebra Kodd tomonidan aniqlangan ikkita gurux amallari 8 ta operatoridan iborat.

Биринчи гурух. Туплар устида бажарилувчи аъъанавий амаллар

- бирлаштириш - Union (\cup)
- кесишма - Intersection (\cap)
- айириш ($-$)

Иккинчи гурух. Махсус релацион амаллар

- танлаш - Selectrion (σ)
- проекция - Projection (π)
- улаш - Join (\bowtie)
- бўлиш (\div)

Relyatsion algebra amallarini operandlari sifatida doimiy yoki o'zgarmas va o'zgaruvchan munosabatlar ishlatiladi. Ushbu amallarning munosabatlarda qo'llagandagi natijalarni batafsil ko'rib chiqamiz.

1) *Birlashtirish* (\cup). Berilgan ikkita munosabatdan biriga va ikkalasiga ham tegishli bo'lgan kartejlardan iborat yangi munosabatni qaytaradi R_1 va R_2 munosabatlarni birlashtirish $R_1 \cup R_2$ ko'rinishida berilib, bu amalning natijasi R_1 munosabatga tegishli bo'lgan yoki R_2 munosabatga tegishli bo'lgan yoki ikkalasiga ham tegishli bo'lgan kartejlar to'plamidir. Bu amallarni bajarayotganda bir xil tartibda bo'lishi kerak. Natijani tartibi ham operandlar tartibiga teng bo'ladi (7.3, a).

$$R_1 \text{ UNION } R_2$$

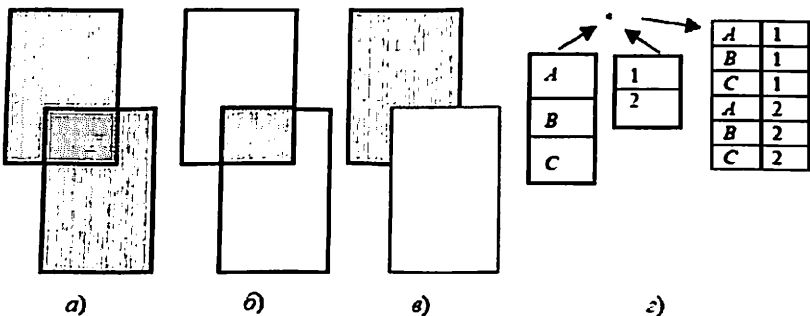
2) *Kesishma* (\cap). Bir vaqtning o'zida ikkala munosabatga ham tegishli bo'lgan barcha kartejlardan iborat yangi munosabatni qaytaradi. (7.3, b).

$$R_1 \text{ INTERSECT } R_2$$

3) *Ayirish* ($-$). Berilgan ikkala munosabatdan faqat birinchisiga tegishli bo'lgan kartejlardan iborat yangi munosabatni qaytaradi. Ayirma R_1 va R_2 munosabatlarni ayirmasi $R_1 - R_2$ ko'rinishida yoziladi va undagi kartejlar to'plami R_1 munosabatga tegishli, lekin R_2 munosabatga tegishli bo'lmagan kartejlardir. Bu amalning bajarilishida ham operandlarni tartibi bir xil bulishi kerak (7.3, v).

$$R_1 \text{ EXCEPT } R_2$$

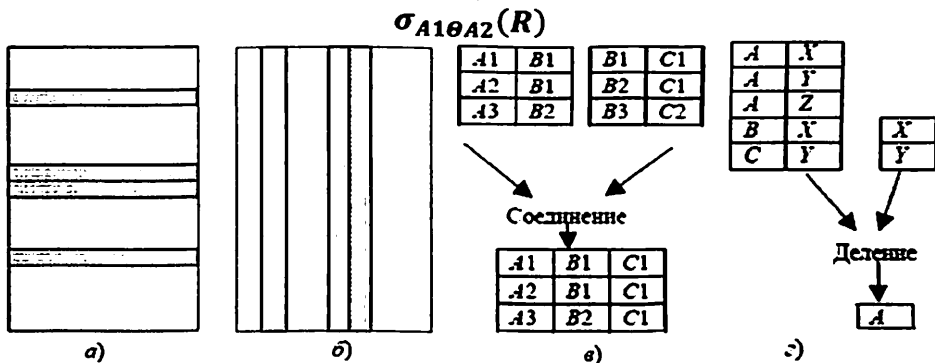
4) *Dekart ko'paytma* ($*$). Berilgan ikkala jadvaldagi kartejlarning kombinatsiyalaridan iborat yangi jadval xosil qiladi. Bizda R_1 va R_2 munosabat berilgan bo'lsin. R_1 munosabatni tartibi $R_1 - P$ va R_2 munosabatniki $R_2 - q$ ga teng bo'lsin. Unda dekart ko'paytma $R_1 * R_2$ ko'rinishida yozilib, uning natijasi uzunligi $P + q$ ga teng bo'lgan kartejlar to'plamidan iborat bo'lib, bu kartejlarni birinchi P komponentasi R_1 kartejga teng bo'ladi, qolgan q komponentasi R_2 kartejga teng (3.2.3-rasm).



3.2.3-rasm. To'plamlar ustida amallar

a-birlashtirish, *b* – kesishma, *v*- ayirish, *g* – dekart ko'paytirish

- 5) *Seleksiya tanlash* (σ) – berilgan jadvaldagi ma'lum shartni qanoatlantiruvchi barcha kortejlardan iborat yangi jadval xosil qiladi. Bu algebraik cheklanish hisoblanadi. Bu amal bajarilganda operandlar sifatida munosabat atributlari ishtirok etadi va solishtirish arifmetik amallari: =, \neq , \leq , \geq , $<$, $>$ va mantiqiy amallar: va (U), yoki (V), not amallari ishlatiladi (3.2.4(a)-rasm).



3.2.4-rasm. Maxsus relyatsion munosabatlar

a-tanlash, *b* – proektsiya, *v*- ulash, *g* – bulish

- 6) *Proektsiya* (π) – berilgan jadvaldagi ba'zi kortejlarni istisno (chiqarib tashlash) qilib qolgan kortejlardan (podkortej) yangi munosabat xosil qiladi. Proektsiya, R munosabatga bu amal tadbiiq etilganda, R munosabatdan ba'zi bir komponentalar olib tashlanadi. Qolganlari esa qaytadan tartiblanadi (3.2.4(b) -rasm).

$$\pi_{A_1, A_2, \dots, A_N}(R)$$

- 7) *Ulash* (\bowtie) – berilgan ikkala jadvalda umumiy qiymatga ega bo'lgan kortejlarning ulanishidan iborat yangi jadval xosil qiladi. Natijaviy jadvalda umumiy qiymat faqat bir marta qatnashadi. Bunday ulash tabiiy ulash deb ataladi (3.2.4(v) -rasm).

$$R_1 \bowtie R_2$$

- 8) *Bo'lish* – berilgan binar va unar ikkita jadval uchun unar jadvalning barcha qiymatlari bilan moslashgan binar jadvaldagi bitta atributning qiymatlaridan iborat jadval (3.2.4(g) -rasm).

$$R_1 \div R_2$$

Jadval ustidagi har amal natijasi jadvaldan iborat bo'ladi. Ushbu relyatsion xususiyat yopiqlik xossasi deb ataladi. Biror amal natijasi boshqa amal uchun boshlang'ich ma'lumot sifatida qo'llanilishi mumkin. Shuning uchun, masalan, birlashtirish proektsiyasini olish yoki ikkita

Munosabatlar ustida amallarga misollar

Birlashtirish

R1

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P2	Гайка	Челябинск	20	24

R2

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P3	Шуруп	Одесса	14	33

R1 ∪ R2

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P2	Гайка	Челябинск	20	24
P3	Шуруп	Одесса	14	33

Kesishma

R1

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P2	Гайка	Челябинск	20	24

R2

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P3	Шуруп	Одесса	14	33

R1 ∩ R2

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40

Seleksiya tanlash

R

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P2	Гайка	Челябинск	20	24
P3	Шуруп	Одесса	14	33

 $\sigma_{Price > 30}(R)$

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P3	Шуруп	Одесса	14	33

Ayirish

R1

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P2	Гайка	Челябинск	20	24

R2

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P3	Шуруп	Одесса	14	33

R1 - R2

PID	Name	City	Weight	Price
P2	Гайка	Челябинск	20	24

R2 - R1

PID	Name	City	Weight	Price
P3	Шуруп	Одесса	14	33

Dekart ko'paytma

R1

A	B
a1	b1
a2	b2

R2

C	D	E
c1	d1	e1
c2	d2	e2
c3	d3	e3

R1 × R2

A	B	C	D	E
a1	b1	c1	d1	e1
a1	b1	c2	d2	e2
a1	b1	c3	d3	e3
a2	b2	c1	d1	e1
a2	b2	c2	d2	e2
a2	b2	c3	d3	e3

Proektsiya

R

PID	Name	City	Weight	Price
P1	Болт	Париж	15	40
P2	Гайка	Челябинск	20	24
P3	Болт	Одесса	15	33

 $\pi_{Name, Weight}(R)$

Name	Weight
Гайка	20
Болт	15

Ulash

R1		R2		R1 ⋈ R2		
A	B	B	G	A	B	G
a1	b1	b1	c1	a1	b1	c1
a2	b2	b2	c2	a1	b1	c4
		b3	c3	a2	b2	c2
		b1	c4			

Bo'lish

SP		P		SP		P	
SID	PID	PID		SID	PID	PID	SID
S1	P1	P1		S1	P1	P2	S1
S1	P2			S1	P2	P4	S4
S1	P3			S1	P3		
S1	P4			S1	P4		
S1	P5			S1	P5		
S2	P1			S2	P1		
S2	P2			S2	P2		
S3	P2			S3	P2		
S4	P2			S4	P2		
S4	P4			S4	P4		
S5	P5			S5	P5		

Nazorat savollari

1. Munosabat va mohiyat-aloqa tushunchalari va ularning farqi nimadi?
2. Relyatsion model qismlari va ularning vazifalari qanday?
3. Kalit turlari va ularga amaliy misollar keltiring.
4. Relyatsion algebra nima va qaeda qo'llaniladi?
5. Maxsus relyatsion amallar bilan ananaviy amallarning farqi nimadi?
6. Kombinatsiyalangan relyatsion amallar bajarish mumkinmi?

4-BOB. SQL TILIDA SO'ROVLAR

4.1. SQL tili va uning tarkibi

SQL tili tarixi

SQL tili relyatsion algebra paydo bo'lgandan keyin paydo bo'ldi va uning birinchi prototipi IBM Research kompaniyasi tomonidan 70 yillar oxirida yaratilgan. 1974 yil IBMda ishlovchi D.Chemberlen "Structured English Query Language" yoki SEQUEL deb nomlangan tilni e'lon qiladi. 1976 yil bu tilning qayta ishlangan SEQUEL/2 versiyasi yaratildi va u rasmiy ravishda SQL (Structured Query Language — Strukturalashgan so'rovlar tili) deb atalgan. SQL qisqartmasi "es-kyu-el" deb talaffuz etadi.

Bu til dastlab IBM System R nomli MBBT tarkibiga kiritilgan. Keyinchalik bu til ko'pgina tijorat MBBT tarkibida qo'llanilgan va keng tarqalganligi sababli vaqt o'tishi bilan relyatsion MBBT larda ma'lumotlar ustida amallar bajaruvchi tillarning norasmiy standarti bo'lib qoldi.

1980 yillarda MBBTlarining bir nechta variantlari mavjud bo'lgan va ularning har bir o'z so'rovlar tilidan foydalangan. Dasturiy maxsulotlarning bir MBBTdan boshqasiga o'tkazishda kelib chiqadigan muammolarni bartaraf qilish uchun yagona standart asosida ishlovchi til yaratish g'oyasi ilgari surilgan/

1986 yilda Amerika Milliy Standartlar Instituti (American National Standards Institute - ANSI) va Xalqaro Standartlash Tashkiloti (International Organization for Standardization - ISO) SQL-86 deb nomlanuvchi SQL standartini chop etdi. 1989 yilda SQL tilining birinchi ramiy standarti SQL-89 ANSI tomonidan qabul qilingan. Ko'pgina MBBTlar ushbu standartni qo'llab – quvvatlaydi. Biroq MB bilan bog'liq axborot texnologiyalarining rivojlanishi va ba'zi talablarning paydo bo'lishi birinchi SQL standartini qayta ishlash va kengaytirishni taqoza etdi.

1992 yil oxirida SQL tilining yangi xalqaro standarti (SQL/92 yoki SQL2) qabul qilindi unda ham ba'zi kamchiliklar aniqlangan, biroq shunga qaramasdan SQL/89 ga nisbatan aniq va to'liqroq hisoblanadi. Hozirgi paytda ko'pgina MBBT ishlab chiqaruvchilar o'z maxsulotlarini SQL2 standartini qanoatlantiradigan qilib o'zgartirdilar.

1999 yil SQL3 deb atalgan yangi standart paydo bo'ldi. Agar SQL1 va SQL2 standartlari biri –biridan miqdor jixati bilan farq qilgan bo'lsa, SQL3 standarti sifat jixatlari bilan farqlanadi. SQL3 ga murakkab strukturaga ega ma'lumotlar tipini ishlatish imkonini beradigan yangi

ma'lumotlar tipi kiritilgan. Bu tipni ob'ektga mo'ljallanganlik darajasi yuqori hisoblanadi.

Standartning keying ko'rinishlari SQL:1999, SQL:2003, SQL:2006 va so'ngida SQL:2008 bo'lib hisoblanadi. Bu tilni yaratishdan asosiy maqsad oddiy protsedurali emas til yaratish orqali, programmalash ko'nikmalariga ega bo'lmagan foydalanuvchilar ham undan foydalanishini ta'minlash bo'lgan.

8-jadval

Version	Short Name	Standard	Release Date
1986	SQL-86	ANSI X3.135-1986, ISO 9075:1987	1986-01-01
1989	SQL-89	ANSI X3.135-1989, ISO/IEC 9075:1989	1989-01-01
1992	SQL-92	ISO/IEC 9075:1992	1992-01-01
1999	SQL:1999	ISO/IEC 9075:1999	1999-12-16
2003	SQL:2003	ISO/IEC 9075:2003	2003-12-15
2006	SQL:2006	ISO/IEC 9075:2006	2006-06-01
2008	SQL:2008	ISO/IEC 9075:2008	2008-07-15
2011	SQL:2011	ISO/IEC 9075:2011	2011-12-15
2016	SQL:2016	ISO/IEC 9075:2016	2016-12-01

SQL tilini «so'rovlar tili» deb aytilgan bo'lsa ham, u MB so'rovidan ko'ra ko'proq ishlarni amalga oshirishi mumkin. Misol uchun, u ma'lumotlarning tuzilishini aniqlaydi, MBdagi ma'lumotlarni o'zgartiradi, xavfsizlik cheklashlarini o'rnatadi. SQL tilini to'la qonli an'anaviy dasturlash tillari tarkibiga kiritib bo'lmaydi. Chunki unda dastur bajarilishini boshqaruvchi va boshqa ko'pgina an'anaviy operatorlar yo'q. Unda faqat MBda saqlanayotgan ma'lumotlarga murojaat qiluvchi operatorlar mavjud. SQL tili o'rganish uchun juda oson, ya'ni:

- 1) *bu noprotsedura til.* Shuning uchun unda ma'lumotni qanday olish emas, balki qanday ma'lumot olish kerakligi ko'rsatiladi. Boshqacha aytganda, SQL tili ma'lumotlarga murojaat usulini ko'rsatishni talab etmaydi. Boshqa zamonaviy tillar kabi SQL tili operatorlarning mustaqil formatiga ega. Ya'ni operatorlarni yozishda operatorlarni aloxida elementlari ekrandagi ma'lum o'rinlarda joylashishi bilan bog'liq emas.
- 2) *buyruqlar ingliz tilining* odatdagi so'zlaridan iborat kalit so'zlardan tashkil topgan, masalan, CREATE TABLE (jadval yaratish), INSERT (kiritish), SELECT (tanlash)

SQL operatorlari *xizmatchi so'zlar* va *foydalanuvchi qo'llaydigan so'zlardan* tashkil topadi. Xizmatchi so'zlar SQL tili doimiy qismi bo'lib, ular aniq qiymatga ega. Ularni standartda ko'rsatilganday yozish kerak va ularni bir satrdan ikkinchisiga ko'chirish uchun bo'linmaydi. *Foydalanuvchi qo'llaydigan so'zlar* foydalanuvchi tomonidan ma'lum sintaksis qoidalari asosida beriladi. Ular o'z navbatida MB ob'ektlarining

har xil nomlaridan iborat bo'ladi (jadval, ustun, tasvirlar, indekslar va x.k.). Operatorlarda so'zlar o'rnatilgan sintaksis qoidalariga moslab joylashtiriladi. Til standartida bu ko'rsatilmagan bo'lsa ham SQL tilining dialektlarida (ko'rinishida) matn tugallanganini bildiruvchi belgi (ko'pgina nuqtali vergul) ishlatiladi.

SQL operator komponentalarini ko'pchiligi reestriga bog'liq emas, ya'ni ixtiyoriy har qanday katta va kichik xarflar ishlatishi mumkin.

Bularda bitta istisno bor. Bu istisno simvulli literallarga tegishli. Ularda litera ma'lumotlar, ularga mos bo'lgan MBdagi qiymatlar qanday saqlansa shunday yozilishi kerak. Masalan, agar MBda familiyaning qiymati «SWITH» ko'rinishida bo'lsa, qidirish shartida «swith» simvol literal ko'rinishida berilsa, bunga tegishli yozuv xech kachon topilmaydi.

SQL tili erkin formatga ega bo'lgani uchun SQL alohida operatorlarni va ularning ketma-ketligini alohida ajratib va tekislab yozish mumkin hamda quyidagi qoidalarga bo'ysunish talab etiladi:

- operatoridagi har bir konstruktsiya yangi satrdan boshlanishi kerak;
- har bir konstruktsiya boshlanishida tashlab ketladigan bo'sh pozitsiyalar, boshqa operator konstruktsiyalari bo'lishi kerak
- agar konstruktsiya bir necha qismdan iborat bo'lsa, ularning har biri qism yangi satrlar bo'sh o'rinlar oldingi konstruktsiyaga nisbatan sijitib yoziladi.

SQL tilining afzalliklari va imkoniyatlari

MB bilan ishlovchi ixtiyoriy til foydalanuvchiga quyidagi imkoniyatlarni yaratishi lozim:

- strukturasi to'la tavsiflagan xolda MB va jadvallarini yaratish;
- ma'lumotlar ustida manipulyatsiya amallarini bajarish, masalan, jadvallardan ma'lumotlarni kiritish, taxrirlash va o'chirish;
- oddiy va murakkab so'rovlarni bajarish.

Bugungi kunga kelib, SQL oddiy so'rovlar tuzuvchi instrument bo'lib qolmasdan, balki ma'lumotlar tuzilmasini yaratish, ulardagi ma'lumotlarni o'zgartirish, ma'lumotlarni himoyalash kabi qator imkoniyatlarga ham egadir. Shunga qaramasdan, ma'lumotlarni o'qish hozirda ham SQL ni muhim bir funktsiyasi hisoblanadi. Hozirda bu til MBBT foydalanuvchilariga taqdim etayotgan barcha funktsional *imkoniyatlari* qo'llanilmoqda. Jumladan:

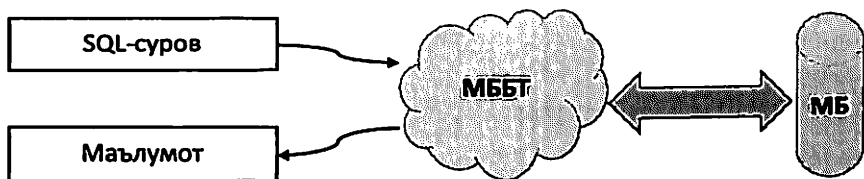
- *Ma'lumotlar strukturasi tuzish* - SQL foydalanuvchilarga ma'lumotlar strukturasi tuzish, o'zgartirish hamda MB elementlari o'rtasida aloqalarni o'rnatish imkoniyatini beradi;

- *Ma'lumotlarni o'qish* – SQL foydalanuvchi yoki dasturga MBda saqlanuvchi ma'lumotlarni o'qish va ulardan foydalanish imkonini beradi;
- *Ma'lumotlarni qayta ishlash* – SQL foydalanuvchiga yoki dasturga MBni o'zgartirish, ya'ni unga yangi ma'lumotlar qo'shish, mavjud ma'lumotlarni o'zgartirish va o'chirish imkonini beradi;
- *MBni himoyalash* - SQL yordamida MB foydalanuvchilarini undagi ma'lumotlarni o'qish va o'zgartirish imkoniyatlarini chegaralab qo'yish mumkin. Bu orqali ruxsat berilmagan foydalanuvchilardan axborotlar himoyalanaadi
- *Ma'lumotlardan birgalikda foydalanish* - SQL ma'lumotlardan birgalikda foydalanishni koordinatsiya qiladi, bu esa paralel ishlayotgan foydalanuvchilar bir-birlariga xalaqit bermasdan MBdagi ma'lumotlardan foydalanishlari imkonini beradi;
- *Ma'lumotlar yaxlitligini ta'minlash* - SQL MBni yaxlitligini ta'minlashga imkon beradi va unda noo'rin o'zgartirishlar qilishni oldini oladi
SQL tilining afzalliklariga quyidagilar kiradi:
- *Qandaydir MBBTga bog'liq emasligi* – ayrim atamalar (yoki dialekt) va sintaksisdagi o'zgachaliklarga qaramasdan o'z tarkibida DDL va DML larga ega SQL so'rovlar bir MBBTdan ikkinchisiga oson ko'chirib o'tkazilishi mumkin. Shunday tizimlar mavjudki, ularning yaratuvchilari dastlab iloji boricha kamroq MBBTlarni foydalanishga harakat qilishgan.
- *Standartlarning mavjudligi* – til uchun standartlar va muvofiqligi uchun testlar mavjudligi hamda SQLning umumiy standartga rioya qilgan holda qo'llanilishi tilning keng qo'llanilishini ta'minlab bermoqda
- *Deklarativlik* – Ibcnehx SQL yordamida faqatgina qanday ma'lumotlarni ajratib olish va modifikatsiya qilishni bayon qiladi. Buni qanday bajarishni esa SQL so'rovni qayta ishlashda MBBTning o'zi hal qiladi.

SQL turlari yoki rejimlari

Yuqorida ta'kidlib o'tilganidek, SQL3 tili tarkibida hisoblash jarayonini boshqarish imkonini beruvchi IF...THEN...ELSE, GO TO, DO ... WHILE kabi buyruqlar mavjud emas. Bunday masalalar dasturiy yo'l bilan (dasturlash tili yoki masalalarni boshqarish tili) yoki interaktiv holda

(foydalanuvchining so'rovlari asosida) amalga oshiriladi. Imkoniyati cheklanganligi sababli (hisoblash jarayonini boshqarish imkoniyati) SQL tili 2 ta usulda qo'llanilish mumkin. Birinchi usulda *interaktiv* ishlash nazarda tutiladi. Bunda foydalanuvchi SQL operatorlarini terminaldan beradi. Ikkinchi usulda *protsedurali tildagi dasturga* SQL tili operatorlari kiritiladi. Bu usullar ba'zi adabiyotlarda MB bilan ishlash texnologiyasi yoki rejimi yoki SQLturlari deb ataladi. Ba'zi adabiyotlarda bu rejimlar *o'zgarmas SQL* va *dinamik SQL* (Embedded SQL and dynamic SQL) deb ham yuritiladi.



Interaktiv rejimda MB bilan ishlashda foydalanuvchi muloqat rejimida ishlaydi, ya'ni SQL tilidagi so'rovni kiritadi va natijani oladi, yani so'rovni kiritadi va natijaga ega bo'ladi va h.k. Interaktiv SQLda foydalanuvchi SQL-so'rovlar va natija interaktiv rejimda olinadi.

O'rnatilgan SQL rejimidp SQL tilidagi so'rovlar boshqa dasturlash tillarida yaratigan dastur kodining tarkibiga kiritiladi. Bu MB bilan boshqa algoritmik tillarda yaratilgan amaliy dasturlar orqali ishlashni ta'minlaydi. Biroq bu yerda MB bilan dastur kompyutorini bog'lash uchun qo'shimcha dasturiy vosita kerak bo'ladi. U dasturlash tili bilan SQL operatorlari o'rtasidagi interfeysni ta'minlab beradi. Bu rejimda SQL buyuruqlari S++, S, C#, Delphi, Java, PHP kabi dasturlash tilidagi kodlar orasiga oldindan yoziladi. Bu shunday tillarni ishlatadigan dasturlarni samarador, quvvatli qiladi. Ular relyatsion MB bilan ishlash imkonini beradi.

Rejimlarning avfzallik va kamchiliklariga to'xtalib o'tiladigan bo'lsa, interaktiv rejimda SQL so'rovlarni tezkorlik bilan yozish, o'zgartirish va natija olish mumkin. Lekin olingan natija vizual jadval bo'lganligi sababli va uning SQL so'rov kodlari kompyuterning xotirasida saqlanmaydi. O'rnatilgan rejimda esa aniq natija beruvchi SQL so'rov kodlari dastur tarkibiga joylashtirilib, dastur orqali murojaat qilinganda natijaviy ma'lumotlarni olish mumkin. Lekin, agar natija olinayotgan SQL so'rov kodini o'zgartirish kerak bo'lsa, u holda ushbu ishlab chiqilgan dasturni dasturlash tilida qayta kodlashtirish keark bo'ladi. Aslini olganda MBBT ham MB bilan ishlovchi dasturlash tillari orqali yaratilgan interfeys

bshlib, uning tarkibida foydalanuvchi buyuruqlarini bajaruvchi SQL kodlaridan iborat.

MB bilan ishlovchi til yuqoridagi amallarni bajarish uchun foydalanuvchilardan kam urinishlarini talab qilishi hamda komandalarining sintaksisi va tuzilishi o'rganish uchun asosan tushunarli bo'lishi kerak. Nixoyat bu til universal bo'lishi kerak. Bu bir MBBT dan boshqasiga o'tganda komandalarni bir xil strukturasi va sintaksisidan foydalanishni ta'xminlaydi. SQL tili bu talablarni barchasini qanoatlantiradi.

SQL tili bir qator murakkabliklarga ega ob'ektlardan iborat. Bular:

- operatorlar
- ko'rsatmalar
- hisoblanadagan funktsiyalar

SQL tilining vazifasi

SQL operatorlari vazifasiga ko'ra ular bir necha guruhlariga bo'lingan:

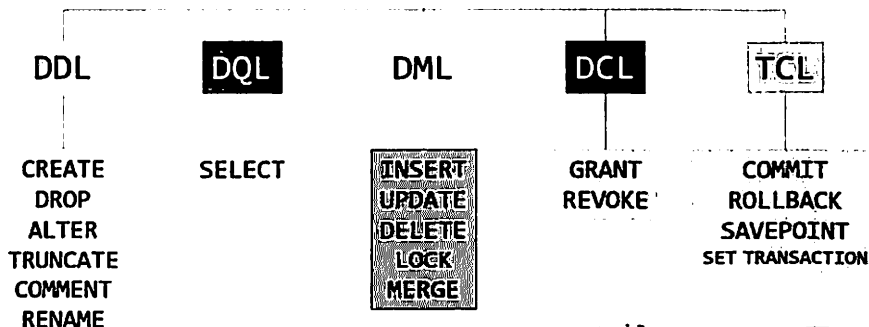
DDL (Data Definition Language) – ma'lumotlarni tavsiflash (yoki hosil qilish) operatorlari. Bu guruhga MBda ob'ektlar (jadval, sxema, indeks)ni yaratish, taxrirlash va o'chirishda ishlatiladigan operatorlar kiradi. ANSI bu guruhni SDL(Schema Definition Language) deb ataydi.

DML (Data Manipulation Language) – ma'lumotlarni manipulyatsiya qiluvchi (qayta ishlovchi) operatorlari. Bunga faqat istalgan vaqtda jadval ichidagi ma'lumotlar bilan ishlash (jadvallarga ma'lumot kiritish, yangilash va o'chirish) operatorlari bilan chegaralanadi.

DCL (Data Control Language) – ma'lumotlarni boshqarish (murojaatlarni tavsiflash) operatorlar. ANSI ning ruxsati bilan, DCL DDL ning bir qismi sifatida qaraladi.

TCL (Transaction Control Language) – tranzaktsiyalarni boshqarish operatorlari. Bu SQL tranzaktsiyalarning boshlanishi va tugashini aniqlovchi buyruqlarni o'z ichiga oladi.

SQL Command Types



Bu guruhlarni aralashtirmaslik zarur. Bular alohida tillar emas, balki SQL operatorlarining guruhlaridir. Bu gurhlarga qo'shimcha quyidagilarni keltirish mumkin:

Integrity (umumiylik). DDL kalit maydonini o'rnatuvchi buyruqlarni o'z ichiga oladi. U kalit maydonini mantiqiy tuzilishini buzish va yangilanishlarga yo'l qo'ymaydi.

View definition (ko'rinishni aniqlash). DDL ko'rinish(view)larni aniqlovchi buyruqlarni o'z ichiga oladi.

Authorization (avtorizatsiya). DDL jadvallar va ko'rinish(view)lardan erkin foydalanish huquqini belgilovchi buyruqlarini o'z ichiga oladi.

Bu tillar ma'lumotlarni qism tillari yuqori darajali dasturlash tillari deyiladi, chunki ularni tarkibida barcha xisoblarni bajarish uchun zarur bo'lganda bo'ladigan til konstruktsiyalari bo'lmaydi (shartli o'tish amallari, yoki tsikl operatori).

9- jadval. Ma'lumotlarni aniqlash operatorlari - DDL

Operator	Ma'nosi
CREATE SCHEMA	MB sxemasini yaratish
DROP SSHEMA	MB sxemasini o'chirish
CREATE TABLE	MB da yangi jadval yaratish
DROP TABLE	MB dan jadvalni o'chirish
ALTER TABLE	Mavjud jadval strukturasi o'zgartirish yoki joriy jadval uchun o'rnatilgan butunlik cheklanishlarini o'zgartirish
CREATE VIEW	Tasavvur yaratish - biror SQL -so'rovga mos virtual jadvalni yaratish
DROP VIEW	Tasavvurni o'chirish

10- jadval. Ma'lumotlarni manipulyatsiyalash operatorlari -DML

<i>Operator</i>	<i>Ma'nosi</i>
DELETE	Satrni o'chirish. Jadvaldagi bitta yoki bir nechta sharni qanoatlandiruvchi yozuvlarni (barcha ustunlaridagi qiymatlarini) o'chiradi.
INSERT	Satrni qo'yish. Jadvalga bitta satrni qo'yish. Bu operatorning boshqa modifikatsiyalarida biror jadvalning bir necha stari yoki so'rov natijasi jadvalga qo'yilishi mumkin.
UPDATE	Satrni yangilash. Bitta yoki filtratsiya shartini qanoatlandiruvchi bir necha ustunlaridagi bitta yoki ko'proq ustunlardagi qiymatlarni almashtirish

11- jadval. so'rov tili yoki ma'lumotlarni tanlash operatori - DQL

<i>Operator</i>	<i>Ma'nosi</i>
SELECT	Satrni tanlash. Relyatsion algebraning barcha amallarini bajaruvchi operator bo'lib so'rovga mos natijaviy jadvalni xosil qiladi.

12- jadval. Tranzaktsiyalarni boshqarish operatorlari - TCL

<i>Operator</i>	<i>Ma'nosi</i>
COMMIT	Tranzaktsiyani tugatish. Tranzaktsiyani tashkil qiluvchi ma'lumotlarni qayta ishlovchi murakkab va o'zaro bog'langan amallarni tugatish
ROLLBACK	Tranzaktsiyani bekor qilish. Tranzaktsiya bajarilishi natijasida yuz bergan o'zgarishlarni bekor qilish
SAVEPOINT	Tranzaktsiya bajarilishida oraliq nuqta saqlash. MB ni oraliq xolatini saqlash. Bu keyinchalik shu xolatga qaytish uchun zarur bo'ladi.

13-jadval. Ma'lumotlarni boshqarish operatorlari - DCL

<i>Operator</i>	<i>Ma'nosi</i>
ALTER DATABASE	MBni o'zgartirish. Butun MBga tegishli bo'lgan ob'ektlar va cheklanishlar to'plamini o'zgartirish
ALTER DBAREA	MB saqlash soxasini o'zgartirish. Avval yaratilgan saqlash soxasini o'zgartirish
ALTER PASSWORD	Parolni o'zgartirish. Butun MB parolini o'zgartirish
CREATE DATABASE	MB yaratish. Barcha parametrlari ko'rsatilib yangi MB yaratiladi
CREATE DBAREA	Saqlash soxasini yaratish. Yangi saqlash soxasini yaratish va unda ma'lumotlar joylashtirishga ruxsat etishi
DROP DATABASE	MB ni o'chirish. Mavjud MBni o'chirish (bunday amal bajarish vakolatiga ega bo'lgan foydalanuvchilar uchun)
DROP DBAREA	MB saqlash soxasini o'chirish. Mavjud saqlash soxasini o'chirish (agar unda joriy vaqtda faol ma'lumotlar joylashmagan bo'lsa)
GRANT	Vakolat berish. MB ba'zi ob'ektlari ustida bir qator amallarni bajarish xuquqini berish
REVOKE	Xuquqdan maxrum qilish. Biror yeb'ektga yoki ob'ekt

ustida ba'zi amallarni bajarish xuquqidan maxrum qilish

SQL ma'lumot toifalari

Asosiy maydon tiplari ishlatiladi:

- *char(n)*: foydalanuvchi tomonidan o'rnatilgan n o'zgarmas uzunlikdagi belgili qator. Uning o'rniga to'liq ko'rinishi - character ishlatilishi ham mumkin.
- *varchar(n)*: foydalanuvchi tomonidan o'rnatilgan maksimal uzunligi n ta bo'lgan maydon uzunlikdagi belgili qator. Bu maydonning to'liq ko'rinishi character varying.
- *int*: butun son (ushbu tip uzunligi masha tomonidan o'rnatiladi). To'liq ko'rinishi integer.
- *smallint*: kichik o'lchamli butun son.
- *numeric(p, d)*: o'zgarmas nuqtali sonlarni berishda ishlatiladi. Bu yerda p - raqamlar (musbat ishorali) va d - o'ng tomondan sonning butun qismiga beriluvchi xonalar soni. Masalan numeric(3, 1) - 44.5; Bunda numeric(3, 1) ga 444.5 yoki 0.32 kabi sonlarni berib bo'lmaydi.
- *real, double precision*: qo'zg'a'luvchi nuqtali va juft-achegarali qo'zg'aluvchan nuqtali sonlar. Chegaralari mashina tomonidan beriladi.
- *float(n)*: qo'g'aluvchan nuqtali juda kichik n xonali son.

Har bir tip maxsus NULL deb nomlanuvchi qiymatni ham o'z ichiga olishi mumkin. SQLda atribut qiymatlari noma'lum bo'lgan o'tkazibi yuborilgan yoki mavjud bo'lmaganlarini NULL bilan yoziladi. NULL qiymat oddiy tushunchada qiymat qiymat hisoblanmaydi. U faqat atributni haqiqiy qiymati tushib qoldirilgan yoki noma'lumligini anglatadi. NULLni ishlatishda quyidagilarga e'tibor berish kerak:

- Agregat funktsiyalar ishlatilganda birorta atributni qiymatlar to'plami bo'yicha hisoblashlar bajarilganda aniqlikni ta'minlash maqsadida NULL qiymat hisobga olinmaydi.
- shartli operatorlarda TRUE, FALSE dan tashqari UNKNOWN paydo bo'lsa natija NULL qiymatda chiqadi.
- bu qiymatni tekshirishda IS NULL yoki IS NOT NULL lar foydalaniladi.
- almashtirish funktsiyalari ham argument sifatida NULL bo'lsa natija NULL ga teng bo'ladi.

Misol uchun A maydon char(10) tipidagi o'zgarmas uzunlikli qatorli maydon bo'lsin. Agar bu maydonga «Avi» qatori berilsa, 10 ta belgi qilish

uchun yana 7 ta oraliq (probel) avtomatik kompyuter tomonidan qo'shiladi. Taqqoslanadigan bo'lsa, varchar(10) tipli B maydonga «Avi» qatori saqlansa, unda bu maydonga hech qanday oraliq (probel) qo'shilmaydi. Agar char tipidagi bu ikkala maydonni solishtiradigan bo'lsak, ularning uzunliklari bir xil bo'lishi uchun solishtirishdan oldin qo'shimcha probellar avtomatik qo'shiladi va keyin solishtiriladi.

Char va varchar tiplarini solishtirish paytida esa, ularning uzunliklarini tenglashtirish uchun solishtirishdan oldin varchar tipidagi maydonga qo'shimcha probellar qo'shiladi. Lekin shunga qaramay, solishtirish paytida bu amallarning bajarilishi ma'lumotlar ombori tizimiga bog'liq. Natija sifatida esa A va B maydonlarida bir xil «Avi» qatori saqlansa ham solishtirishda $A=B$ false qiymatini qaytarishi mumkin. Shuning uchun ham yuqoridagi muammoni chetlab o'tish uchun char tipi o'rniga varchar tipidan foydalangan yaxshi.

Bundan tashqari SQL da yana Unicode belgilaridan foydalangan holda turli xil tildagi ma'lumotlarni saqlovchi nvarchar tipi ham ishlatiladi. Ammo, ko'pgina MBlarda Unicode (UTF-8 ko'rinishidagi) belgilarini varchar tipida saqlash imkoniyatiga ega.

Nazorat savollari

1. SQL standartlari va uning rivojlanish bosqichlari
2. SQL tili operatorlarning formatiga bo'lgan talablar
3. SQL tilining afzalliklariga nimalar kiradi?
4. SQL rejimlarini tushuntirib bering
5. SQL operatorlari guruhlarini vazifasini tushuntiring
6. SQL tilidagi asosiy ma'lumot toifalari qanday?
7. SQL tilida NULL qiymat nima?

4.2. DDL – ma'lumotlarni ta'riflash operatorlari

Endi bizga ma'lumki, relyatsion MBning asosini jadvallar tashkil qiladi. Jadvallarning tuzilishi sxema deb qaralib, bu sxema ustunlarning (maydon) o'zaro munosabatidan iborat. Relyatsion munosabat shartlaridan kelib chiqqan holda sxemada ustunlar soni cheklangan. Chunki ustunlar qaralayotgan ob'ektlarni ifodalovchi bir turdagi chekli xususiyatlarini o'z ichiga oladi. Misol uchun, odam ob'ektini biologik xususiyatlarini qaraladigan bo'lsa, bu xususiyatlarga odamning maxsus kodi, ismi sharifi, jinsi, yoshi, bo'yining uzunligi, og'irligi, qon bosimi, irqi kabilar kiradi. Odamlar ob'ektini quyidagi jadval ko'rinishida ifodalanadi.

14-jadval

Kod	Fio	Jinsi	Yoshi	Bo'yi	Og'irligi	Qon bosimi	Irqi

Ko'rinib turganidek, relyatsion turidagi jadval ustunlari soni oldindan ma'lum va bu ustunlar cheklangan holdagi ma'lumot turiga ega. Misol uchun "kod" ustuni 10 xonali sonli, "FIO" ustuni maksimum 50 simvulli satri, "jinsi" ustuni erkak yoki ayol qiymatini qabul qiluvchi mantiqiy, "yoshi" ustuni esa 1 dan 120 gacha bo'lgan natural sonli ma'lumot turlariga ega.

Keltirilgan jadvallarni MBda xususiyatlari bo'yicha yaratish, taxrirlash va o'chirish uchun SQLning DDL guruh operatorlari orqali amalga oshiriladi. Ya'ni, MBdagi jadvallarni ta'riflash tizimi DDL hisoblanadi. DDL faqat jadvallarni ta'riflamasdan, yana har bir jadval haqidagi ma'lumotni tavsiflashga imkon beruvchi quyidagilarni o'z ichiga oladi:

- Har bir jadvalning sxemasini;
- Qiymat turlarining har bir atribut bilan birlashtirilishini.
- Kalit maydonlarini (umumiylikni cheklashni).
- Har bir jadval uchun mavjud bo'lgan ko'rsatkich(indeks)lar o'rnatishni.
- Har bir jadval uchun xavfsizlik va avtorizatsiya axborotini.
- Har bir jadvalning fizik tasnifini diskda saqlashni.
- SQL standartida turli xildagi tiplar

Asosiy sxema ta'rifi. SQLda jadvalni yaratishda **CREATE TABLE** buyrug'idan foydalaniladi. Create table buyrug'ining umumiy sodda ko'rinishi quyidagicha:

CREATE TABLE R**(A₁ D₁,**

.....,

A_n D_n,**<kalit maydoni 1>,**

.....,

<kalit maydoni n>);

Bu yerda R - jadval nomi, har bir $A_i - R$ jadvalni tashkil etuvchi ustunlar, ya'ni maydon nomi, $D_i - A_i$ maydonning ma'lumot tipi. Ustun nomi A_i bilan uning ma'lumot tipi o'rtasiga pobel qo'yiladi. Agar jadval maydonlari bir nechta bo'lsa, yangi ustun nomi oldiga vergul qo'yiladi. SQLda biror buyuruq nixoyasi nuqtali vergul bilan yakunlanadi. Create table buyruqidi bo'sh jadval yaratadi, ya'ni jadvalda satrlar bo'lmaydi.

Quyidagi buyuruq MBda *xodimlar* nomli jadvalni yaratadi:

CREATE TABLE xodimlar**(fio varchar(20),****bulim varchar(15),****maoshi numeric(8,2),****primary key (fio));**

Yuqorida yaratilgan jadvalda 3 ta maydon mavjud va ular quyidagilar:

fio - maksimal belgilar soni 20 ta bo'lgan belgili qator maydoni,

bulim - maksimal belgilar soni 15 ta bo'lgan belgili qator maydoni va

maoshi - umumiy xonalari soni 8 ta, ularning 2 tasi sonning butun qismini ifodalash uchun ketgan son bo'lgan butun qiymatli son maydoni.

Shu bilan birga create table buyrug'i ichiqda umumiylikni cheklash buyrug'i *fio* maydoni *xodimlar* jadvalida asosiy kalit maydon ekanligini ko'rsatyapti.

SQLda turli xildagi kalit maydonlardan mavjud. Jumladan:

- ✓ *primary key* ($A_{j_1}, A_{j_2}, \dots, A_{j_m}$): primary key $A_{j_1}, A_{j_2}, \dots, A_{j_m}$ lar jadval uchun asosiy kalit so'z ekanligini bildiradi. Primary key maydon qiymatlari hech qachon null bo'lmasligini va takrorlanmas bo'lishi lozimligini ta'minlaydi.
- ✓ *foreign key* ($A_{k_1}, A_{k_2}, \dots, A_{k_n}$) *references s*: Bu kalit maydoni jadvaldagi har bir ma'lumotning $A_{k_1}, A_{k_2}, \dots, A_{k_n}$ maydonlarining qiymati s jadvalning ba'zi ma'lumot maydonlarining qiymatlari kalit maydonlarining qiymatiga mos kelishi kerakligini bildiradi. Bu kalit maydon jadvalni boshqa javdallar bilan bog'lashda xizmat qiladi.
- ✓ *not null*: ushbu kalit maydoni o'zi o'rnatilgan maydon qiymatining null qiymat olishi mumkin emasligini bildiradi.

SQL kalit maydonning mantiqiy tuzilishini buzuvchi har qanday yangilanishga yo'l qo'ymaydi. Misol uchun, jadvalning *primary key* maydoniga ma'lumot kiritish qismiga *null* qiymatli ma'lumot kiritilsa yoki mavjud ma'lumot *null* qiymati bilan o'zgartirilsa va yoki *primary key*

maydonning ma'lumot kiritish qismiga kiritilgan qiymat shu jadvalning boshqa maydoniga kiritilgan qiymat bilan bir xil bo'lsa, SQL xatolik haqida ma'lumot chiqarib, yangilashni bekor qiladi. MBda yaratilgan yangi jadval bo'sh bo'ladi.

SQL MBdan jadvalni olib tashlash (o'chirish) uchun *DROP TABLE* buyrug'i ishlatiladi. *DROP TABLE* buyrug'i MBdan o'chirilayotgan jadvalning barcha ma'lumotlari bilan birga o'chirib tashlaydi.

DROP TABLE R;

E'tibor beradigan bo'lsak, jadvalni yaratishda jadval nomi, jadvalni tashkil etuvchi ustunlar ta'riflangan va cheklashlar o'rnatilgan edi. O'chirishda esa MBda mavjud jadvalning nomi ko'rsatilishi kifoya. Tafsiya o'rnida: MB ishlayotgan xodim bu buyuruqdan extiyotkorlik bilan foydalanishi lozim. Chunki bunda o'chirilgan ma'lumotlarni qayta tiklash muammolari kelib chiqadi.

Ma'lumot o'rnida. SQL buyruqlarining sinaksisida ko'p uchraydigan belgilashlar: [] – kvadrat qavs ichidagi ifodani zarrur bo'lgan hollarda ishlatiladi, | – tik chiziq ushbu buyruqda faqat yoki o'ng, yoki chap tomonidagi ifoda(kalit so'z)dan foydalanishni bildiradi.

CREATE TABLE buyrug'ining kengaytirilgan ko'rinishi quyidagicha:

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
[(create_definition,...)]
[table_options] [select_statement]
```

Bu yerda *tbl_name* – yaratilayotgan jadval nomi va u majburiy, *TEMPORARY* kalit so'zi yaratilayotgan jadvalning vaqtinchalik bo'lib, ushbu jadval SQLning keyingi seansigacha saqlanib qolinadi.

IF NOT EXISTS – kalit so'zi, yangi jadval ushbu MBda xuddi shu nomli jadval bo'lmasagina yaratilishi lozirligini bildiradi.

create_definition qismida jadval maydonlari ta'riflanadi, ya'ni:

```
maydon type [NOT NULL | NULL]
[DEFAULT default_value] [AUTO_INCREMENT]
[CHECK] [PRIMARY KEY] [FOREIGN KEY] [KEY] [UNIQUE] [INDEX]
```

maydon – jadvalda ta'riflanayotgan maydon nomi;

type – *maydon* nomli ustun ma'lumot turi (misol uchun MySQL uchun: Int[(length)], Real[(length,decimals)], Numeric(length,decimals), Char(length), Varchar(length), Date, Time, Longblob, Text, longtext, SET(value1,value2,value3,...))

NOT NULL | NULL – maydon qiymati bo'sh yoki bo'sh qiymatga ega bo'lishi;

DEFAULT default_value – maydon uchun dastlabki qiymat o'rnatiladi. Bu yerda *DEFAULT* kalit so'zdan keyin *default_value* o'rniga

qiymat yoziladi. Agar qiymat matnli ko'rishda bo'lsa, qo'shtirnoq ichida yoziladi.

CHECK (*expr*) – maydon qiymatlariga chegara qo'yishda ishlatiladi. Misol uchun sonli maydon 100dan kichik qiymat olsa, **CHECK** (*maydon < 100*).

Ta'riflanayotgan maydonga zaruriyat bo'lsa quyidagilardan faqat bitta orqali kalit maydon qilinadi:

AUTO_INCREMENT – faqat int tipidagi maydonga avtohisoblagich o'rnatish. Bu qaytarilmas raqam ob'ektni identifikatsiya qiliadi;

PRIMARY KEY (*index_col_name,...*) – ko'rsatilgan maydonini birlamchi (asosiy) kalit maydon qiladi;

KEY [*index_name*] (*index_col_name,...*) – ko'rsatilgan maydonni alohida ko'rsatilgan nomga ega kalit maydon qilish;

INDEX [*index_name*] (*index_col_name,...*) – maydonni index maydon qilish;

UNIQUE [INDEX] [ind_name] (ind_col_name,...) – maydonni unikal qilish. Ba'zi xollarda biror maydonga kiritilayotgan barcha qiymatlar bir biridan farq qilishi kerak. Bunda shu maydon uchun **UNIQUE (*yagona*) so'z ishlatiladi;**

FOREIGN KEY – ikkilamchi kalit yaratish. To'liq yozilishi quyidagicha:

[CONSTRAINT symbol]

FOREIGN KEY (maydon)

REFERENCES table2 [(name2)]

Bu yerda yaratilayotgan *maydon* nomli ustun ma'lumot tipi sonli va uning qabul qiladigan qiymatlari boshqa *table2* jadvaldan *name2* nomli va u **PRIMARY KEY** turidagi maydon domenidan qiymatlar qabul qiladi;

table_options qismida yaratilayotgan jadvalning MB bo'yicha xususiyatlari ko'rsatiladi, jumladan:

TYPE = { *InnoDB* | *MERGE* | *MYISAM* | ... } – jadvalning yozuv formati

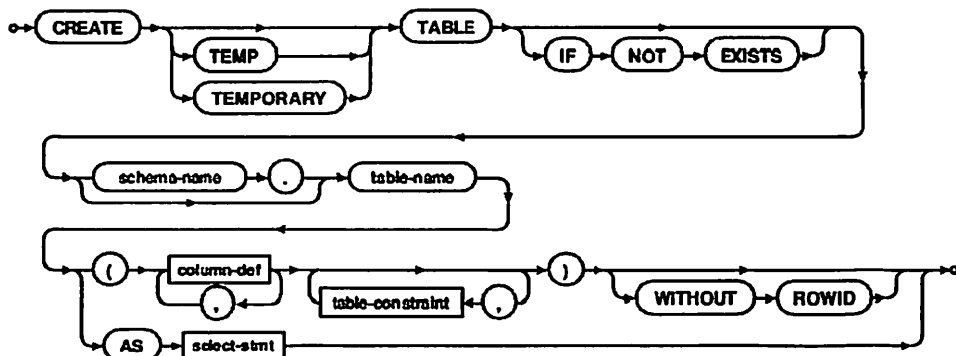
AUTO_INCREMENT = # – avtohisoblagich davom qiladigan raqam;

COMMENT = "string" – jadval uchun izoh ko'rsatilishi;

MAX_ROWS = #, **MIN_ROWS** = # - eng katta va eng kichi yozuvlari soni.

select_statement qismi *create_definition* bo'lmagan holda ishlatiladi:

[IGNORE | REPLACE] SELECT ... (har qanday to'g'ri SELECT natijasi)



To'g'ri tuzilgan SELECT so'rovi natijasidagi yozuvlaridagi maydon nomlari bo'yicha jadval yaratiladi va avtomat aniqlangan ma'lumt turlari jadvalning mos maydon turlari qilib belgilanadi.

```

CREATE TABLE IF NOT EXISTS newbook_mast(
  book_id varchar(15) NOT NULL AUTO_INCREMENT PRIMARY KEY,
  book_name varchar(50),
  isbn_no varchar(15) NOT NULL UNIQUE,
  cate_id varchar(8),
  aut_id varchar(8),
  pub_id varchar(8),
  dt_of_pub date,
  pub_lang varchar(15) DEFAULT "O'zbek adabiyoti"
  no_page decimal(5,0) CHECK(no_page>0),
  book_price decimal(8,2),
  FOREIGN KEY (aut_id) REFERENCES newauthor(aut_id));
  
```

Bu jadvalda *book_id* maydoniga avtohisoblagich o'rnatilgan birlamchi kalit maydon, *pub_lang* maydoniga dastlabki qiymat va *no_page* maydoniga cheklov o'rnatilgan. *FOREIGN KEY* orqali *aut_id* maydoni *newauthor* jadvalining *aut_id* maydoni orqali ikkilamchi kalit qilingan.

Jadvaldagi ma'lumotlarni maydonlar bo'yicha qidirish-tanlash amalini yetarli darajada tezlatish uchun ma'lumotlarni berilgan maydon bo'yicha indeksatsiya qilish ishlatiladi. Indeksni bitta yoki bir nechta maydon bo'yicha bajarish mumkin. Indeks buyuruqining sintaksisi quyidagicha:

```

CREATE INDEX <indeks nomi> ON
<jadval nomi> (<ustun nomi>[,<ustun nomi>]);
  
```

Bu komanda bajarilishi uchun jadval yaratilgan va indeksda ko'rsatilgan ustunlar unda bo'lishi kerak. Masalan, *xodimlar* jadvalidan talabani *fio* maydoni qiymati bo'yicha maoshini qidirish tez-tez talab etilsa, unda shu maydon bo'icha indekslash bajariladi

```

CREATE INDEX fio_index ON xodimlar (fio);
  
```

Indeksni olib tashlash uchun (bunda uni nomini albatta bilish kerak) quyidagi komanda ishlatiladi.

```

DROP INDEX <indeks nomi>;
  
```

Masalan,

DROP INDEX <fi_o_index>;

Yaratilib bo'lingan va ko'p hollarda unga ma'lumot kiritilgan jadvallarning tuzilmasini, ya'ni maydonlar bo'yicha tahrirlash ishlarini olib borishga to'g'ri keladi. Bunday amallarni DDL guruhning *ALTER TABLE* operatori bajaradi. Sodda ko'rinishda qaraladigan bo'lsa, mavjud jadvalga maydon qo'shish (*Add*) va olib tashlashda (*Drop*) quyidagi buyruqlardan foydalaniladi. alter table buyrug'ining umumiy ko'rinishi quyidagicha:

ALTER TABLE r **ADD** A T;

ALTER TABLE r **DROP** D;

Bu yerda r - mavjud jadval nomi, A - yangi qo'shilgan maydon nomi, T - maydonning tipi, D - o'chiriladigan maydon nomi. Bunda jadvalga yangi kiritilgan maydonning barcha maydonlari null qiymati bilan to'ldirib chiqiladi. Umumiy holda *ALTER TABLE* operatori sintaksisi quyidagicha:

ALTER [IGNORE] **TABLE** jadval_nomi [, ko'rsatma ...]

Quyida ushbu operator ko'rsatmalar va ular parametrlari keltiriladi:

ADD – ko'rsatilgan jadvalga yangidan maydon (ma'lumot tipi bilan) va kalitlar qo'shish imkonini beradi. Bu buyuruq bo'yicha mavjud jadval satrlariga yangi ustun qo'shiladi va unga NULL qiymati yoziladi. Jadvalga bir nechta ustun ham ko'shsa, ular bir biridan vergul bilan ajratiladi.

ADD [COLUMN] create_definition [FIRST | AFTER column_name]

ADD [COLUMN] (create_definition, create_definition,...)

ADD INDEX [index_name] (index_col_name,...)

ADD PRIMARY KEY (index_col_name,...)

ADD UNIQUE [index_name] (index_col_name,...)

ADD FULLTEXT [index_name] (index_col_name,...)

ADD [CONSTRAINT symb] **FOREIGN KEY** index_name (ind_col_name,...) [reference_def]

Alter, Change, Modify – ko'rsatilgan maydonni taxrirlaydi. Ustun tasniflarini modifikatsiyalashda ushbu cheklanishlarni xisobga olish kerak:

- Not Null o'rnatilishi uchun ustunda birorta ham Null qiymat bulmasligi;
- Ma'lumot toifasini o'zgartirishni, faqat ustun bo'sh bo'lsa bajarish;
- To'ldirilmagan ustun uchun o'lcham/aniqlik uzunliklarini o'zgartirish;
- To'ldirilgan ustun uchun o'lcham/aniqlik faqat kattalashtirish;
- Sukut bilan o'rnatilgan qiymatni har doim uzunligini o'zgartirish

mumkin

ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}

CHANGE [COLUMN] old_col_name create_def [FIRST | AFTER col_name]

MODIFY [COLUMN] create_definition [FIRST | AFTER column_name]

DROP – ko'rsatilgan maydon va kalitlarni o'chiradi.

DROP [COLUMN] col_name

DROP PRIMARY KEY

DROP INDEX *index_name*

RENAME [TO] new_tbl_name – nomini o'zgartiradi.

ORDER BY col – jadvaldagi ustunlar tartibini belgilaydi.

table_options –jadval xususiyatlarini o'zgartiradi.

Eslatma. Alter table operatorida faqat bitta ko'rsatma bajariladi va/yoki ko'rsatmalar vergul orqali ajratiladi. Misollar.

CREATE TABLE testtable (c1 INT(11), c2 VARCHAR(15));

ALTER TABLE testtable **RENAME** W;

ALTER TABLE W **MODIFY** c1 TINYINT NOT NULL, **CHANGE** c2 c3 VARCHAR(25);

ALTER TABLE W **ADD** c4 TIMESTAMP;

ALTER TABLE W **ADD INDEX** (c4), **ADD UNIQUE** (c1);

ALTER TABLE W **DROP COLUMN** c3;

ALTER TABLE W **ADD** c3 INT NOT NULL AUTO_INCREMENT,

ADD PRIMARY KEY (c3);

ALTER TABLE W **MODIFY** c1 BIGINT;

ALTER TABLE W **MODIFY** c1 BIGINT UNSIGNED DEFAULT 1 COMMENT 'test column';

ALTER TABLE W **CONVERT TO CHARACTER SET** latin1;

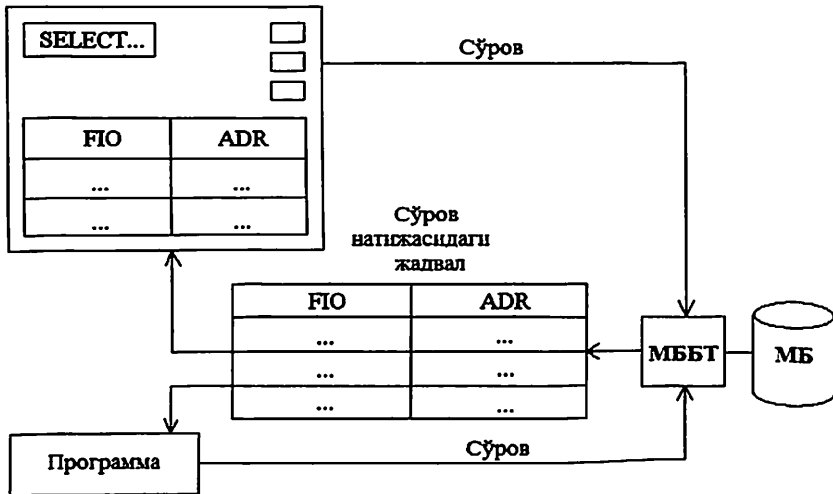
Nazorat savollari

1. Jadvallarning sxemasi deganda nimani tushuniasiz?
2. Jadval yaratishdagi birlamchi va ikkilamchi kalitlarning mohiyati va farqi nimadi?
3. Jadval attributlariga cheklov va izox qanday o'rnatiladi?
4. INDEX nima va ular bilan ishlash operatorlari qanday?
5. Mavjud jadvalni tahrirlash buyruqlari sintaksisini tushuntiring.

4.3. DQL sinfi operatori

SQL tilida ma'lumotlar bilan manipulyatsiya amallari uchun to'rtta kalit so'z ishlatiladi: SELECT, INSERT, UPDATE, DELETE. SELECT operatorli ifodalar alohida o'rin egallaydi, chunki u ma'lumotlar ajratib olish uchun mo'ljallangan va bu foydalanuvchilar yechadigan masalalarni ko'p qismini tashkil qiladi.

SQL tili birinchi navbatda so'rovlarni bajaradi. SQL so'rovlari orqali ma'lumotlarni tanlashda SELECT (tanlash) so'zidan foydalanilib, bu SQL operatorlarining eng funktsionali hisoblanadi. U MB jadvalidan axborotlarni tanlab olish uchun muljallangan. SELECT operatori MBdan ma'lumotlarni o'qiydi va so'rov natijasini jadval shaklida qaytaradi. (4.3.1-rasmga qarang)



4.3.1-rasm. SQL so'rovining bajarilish

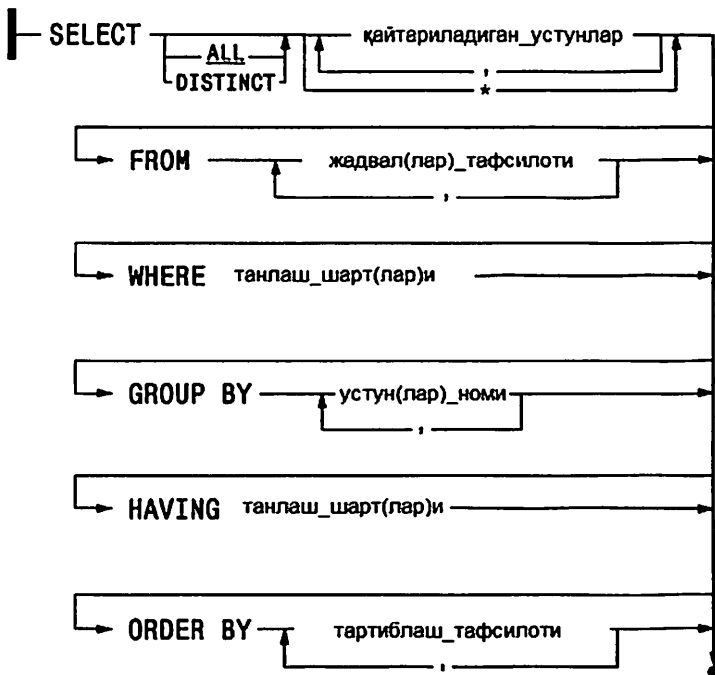
SELECT operatorining sintaksisi quyidagicha va diagrammasi 2-rasmdagi kabi keltiriladi.

```
SELECT [ALL|DISTINCT] <atributlar ruy. > [AS<atributlar aliasi >]
FROM <jadvallar ruyxati>
[{{INNER | LEFT | RIGTH [OUTER}} JOIN <jadval> ON <bog'lanish sharti>]
[WHERE <tanlash sharti>]
[ORDER BY <atributlar ruyxati>]
[GROUP BY <atributlar ruyxati>]
[HAVING <shart>]
[UNION <SELECT operatori ifoda>];
[LIMIT <oraliq chegara>]
```

Kvadrat qavslrda operatorni yozishda qatnashishi shart bo'lmagan elementlar ko'rsatilgan.

SELECT operatori ishlatilishi jihatidan uchta variantga ega:

- 1) funktsiyallar, aniq qiymat va hisoblash natijasini chiqarishda faqat SELECT kalit so'zini ishlatilishi;
- 2) jadval(lar)dan zarur ma'lumotlarni chiqarishda yuqoridagi kabi (SELECT ... FROM ...) ishlatish;
- 3) aralash xolatida ishlatish.



4.3.2-rasm. SELECT operatorining sintaksis diagrammasi

ALL kalit so'zi natijaga shartni qanoatlantiruvchi barcha satrlar, shuningdek takrorlanuvchi satrlar ham kirishini bildiradi. Ko'pchilik xolatlarda bu *ALL* kalit so'zini yozish shart emas.

DISTINCT kalit so'zi natijaga takrorlanuvchi satrlar kiritilmasligini bildiradi. Keyin boshlang'ich jadvaldagi atributlar ro'yxati ko'rsatiladi. Bu atributlar natijaviy jadvalga kiritiladi. *ALL* va *DISTINCT* kalit so'zlari qaytariluvchi, ya'ni ekranga chiqariluvchi jadval maydonlari oldiga yoziladi.

* simvoli boshlang'ich jadvalning barcha atributlari natijaga qaytarilishini ta'minlaydi. Agar uning o'miga jadvaldagi qaytariluvchi ustun nomlari vergul bilan ajratilgan holda kiritilsa, natija faqat ushbu ustunlarni chiqaradi. Agar tuzilayotgan so'rovda bir nechta jadval ishtirok etsa, u holda ustun nomi va u qaysi jadvaldan ekanligi ko'rsatilishi lozim.

Misol uchun $A(a1, a2, a3)$ va $B(b1, b2, a1, b3, b4)$ jadvallar berilgan bo'lsin.

- ✓ $A.*$, $B.*$ – A va V jadvallardagi barcha ustunlar chiqaradi;
- ✓ $A.*$, $B.b1$ – A jadvalning barcha ustuni, V jadvalning $b1$ ustuni;
- ✓ $A.a1$, $B.b2$, $B.a1$ - A jadvaldan $a1$ ustun, V jadvaldan $b2$ va $a1$ ustunlari.

AS alias ustun (va/yoki jadval) nomlarini ekranga chiqishini maskalaydi, ya'ni boshqa nomga o'zgartiradi.

$FROM$ so'zi operatorida qatnashishi shart bo'lgan so'zlardan hisoblanadi. Bu so'zdan keyin tanlov bajariladigan jadvallar nomi ko'rsatiladi. Jadval nomlari alias bilan yozilishi mumkin. $ALIAS$ bu jadvalga beriladigan taxallus. Alias orqali so'rovda ko'p uchraydigan jadval nomlarini qisqacha ishlatish imkonini beradi. Agar jadvallar bir nechta bo'lsa ular o'rtasi vergul bilan ajratiladi.

$JOIN$ va undagi kalit so'zlar so'rovda ko'p jadval (va/yoki boshqa $SELECT$ so'rovi) ishtirok etganda ularni mos atributlarining o'zaro bog'lanishini ta'minlaydi. Bu yerda bog'lanish to'g'ridan-to'g'ri ($INNER$), o'ngdan ($RIGHT$) va chapdan ($LEFT$) kabi turlarga ajratilib, bog'lanish sharti ON so'zidan keyin ifodalanadi. Bog'lanishlarni $WHERE$ orqali ham berish mumkin, lekin srov tezkorligi uchun $JOIN$ standartini qo'llash tavsiya etiladi.

$WHERE$ kalit so'zidan keyin jadval satrlarini tanlab olish sharti ko'rsatiladi. Bunda natijaviy jadvalga $WHERE$ ifodasidagi shart rost qiymat qabul qiladigan satrlar kiritiladi. Shartlar maydonga nisbatan bir qiymatli, qo'p qiymatli, oraliqli va tegishlilik turlarga ajratiladi. Agar so'rovdagi maydonlarga bir nechta shart ishtirok etadigan bo'lsa ular orasida AND va OR kabi mantiqiy bog'lanishlar qo'yiladi. Yana shuni bilish muhimki $WHERE$ ifodasidagi shartlar so'rovdagi jadvallarning mos maydonlari bo'yicha har bir yozuv tekshiriladi va shartlarning hammasi rost bo'lganda natijaga chiqariladi.

$ORDER BY$ kalit so'zi natijaviy jadval satrlarini ko'rsatilgan ustunlar ro'yxatini o'sish yoki kamayish bo'yicha tartiblaydi. Tartiblanuvchi atributlar ko'p bo'lsa, oralari vergul bilan ajratiladi

$GROUP BY$ kalit so'zidan keyin guruhlanadigan atributlar ro'yxati ko'rsatiladi. Guruhlanadigan atributlar ko'p bo'lsa, ular orasi vergul bilan ajratiladi. Guruhlar ko'rsatilgan atributning bir hil bo'lgan qiymatlari asosida amalga oshiriladi.

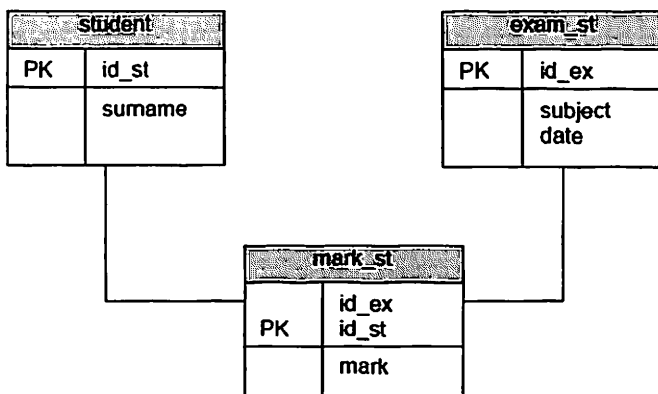
$HAVING$ ifodasida har bir gruppaga qo'yiladigan shartlar ko'rsatiladi $GROUP BY$ va $HAVING$ kalit so'zlari murakkabroq va guruhlanuvchi

so'rovlarda ishlatiladi. WHERE so'zida shartlar har bir yozuv bo'yicha olib boriladigan bo'lsa, HAVING da maydonni guruhlanishida hosil bo'lgan maydon elementlari to'plami ustida bajarilgan amallarning natijasi bo'yicha amalga oshiriladi. Ya'ni bunda oldin GROUP BY kalit so'zi orqali guruhlarga ajratiladi, keyin ushbu guruh qiymati aniqlanadi va oxirida shart tekshiriladi.

LIMIT tanlash natijalarini son jihatidan chiqarish oraliq chegarasi. Misol uchun natija jadvalida N ta yozuv bo'lsa LIMIT so'zida N dan kichik bo'lgan ikkita butun son oraliq'i beriladi. Misol uchun LIMIT 10, 25 natija sifatida qaytarilgan jadvaldan 10, 25 oraliqdagi yozuvlarni chiqaradi.

FROM, WHERE va ORDER BY kalit so'zlari SQL tilining qolgan ma'lumotlarni manipulyatsiyalash operatorlarida ham shu tarzda ishlatiladi.

Endi so'rovlar yaratishni aniq misol uchun ko'rib o'tiladi (4.3.3-rasm).



4.3.3-rasm

student	
id_st	Surname
1	Xasanov
2	Karimov
3	Jabborov

exam_st		
id_ex	Subject	Date
1	Matem	10.05.2020
2	Fizika	
3	Informatika	25.05.2020

mark_st		
id_ex	id_st	mark
2	2	3
3	1	2
1	1	4
3	2	5
2	1	3
1	2	2
2	3	4
1	3	5
3	3	3

Endi SQL so'rovlari oddiydan boshlab tuzishga misollar qaraladi.

student jadvalidan barcha studentlar ro'yxatini tasvirlash.

```
SELECT * FROM student
```

yoki

```
SELECT id_st, surname FROM student
```

Yuqoridagi so'rovga ORDER BY surname ifodasi qo'shilsa, u holda ro'yxat familiya bo'yicha tartiblanadi. Jimlikka ko'ra tartiblash o'sish bo'yicha bajariladi. Jimlik o'rniga ASC so'zi ishlatiladi.

Ms: student jadvalidan barcha studentlar ro'yxatini tartiblab tasvirlash.

```
SELECT * FROM student
```

```
ORDER BY surname
```

Agar kamayish bo'yicha tartiblash kerak bo'lsa, u xolda oxirgi ifodadagi atribut nomidan keyin DESC so'zi qo'shiladi.

```
SELECT * FROM student
```

```
ORDER BY DESC surname
```

Alias qo'yish. Natijaviy jadval maydon nomlarini o'zgartirishda AS so'zidan foydalaniladi.

```
SELECT id_st AS KOD, surname AS 'FIO talaba'
```

```
FROM student
```

Eslatma. Agar aliasdagi maskalanuvchi matn bir nechta so'zdan iborat bo'lsa, ya'ni ular orasida probel bo'lsa, bu maska matni tirnoqchaga olinadi. Aks holda tirnoqchaga olish shart emas. So'rov natijasida qo'yidagicha bo'ladi.

KOD	FIO talaba
1	Xasanov
2	Karimov
3	Jabborov

WHERE so'zidan keyin solishtirish amallarini (<, >, =, <> va x.k.) va mantiqiy operatorlar qatnashgan ifodalarni joylashtirish mumkin.

«1» kodli student olgan baxolar ro'yxatini tanlab olish va tasvirlash

```
SELECT id_st, mark
```

```
FROM mark_st
```

```
WHERE id_st = 1
```

SQL tilida bir nechta shartlarni birgalikda bajarilishi uchun AND, OR va NOT kabi mantiqiy operatorlar ishlatiladi.

AND – mantiqiy “hamda”, “va” ma'nosini beradi. Ikki shart orasiga qo'yiladi (shart1 AND shart2). Ifoda True bo'lishi uchun ikkala shart ham rost bo'lishi kerak.

OR – mantiqiy “yoki” ma'nosini beradi. Ikki shart orasiga qo'yiladi (shart1 AND shart2). Ifoda True bo'lishi uchun ikki shartdan biortasi rost bo'lishi kerak.

NOT – bekorlash, emas ma'nosini beradi (invert shart). Mantiqiy ifoda True (False) bo'lganda unga teskari bo'lgan False (True) qiymatlarni

qaytaradi (NOT mantiqiy ifoda). Har bir mantiqiy operator uchun “haqiqat” jadvalini ishlab chiqish mumkin. Bunda qiymat NULL boʻlganda natija qanday boʻlishiga qarab qoʻshimcha shart beriladi.

		AND		
		True	False	NULL
True	True	True	False	NULL
False	False	False	False	False
NULL	NULL	False	False	NULL

		OR		
		True	False	NULL
True	True	True	True	True
False	True	True	False	False
NULL	True	True	False	NULL

		NOT		
		True	False	NULL
True	True	False	True	NULL
False	True	True	False	NULL

Shuningdek, SQL soʻrovida atributning NULL qiymatlari va NULL ifodalar bilan ham ishlash mumkin. Bunda *IS NOT NULL* (NULL ekanligi) va *NOT NULL* (NULL emasligi) operatorlari mavjud.

Mantiqiy operatorlarning bajarilishi jihatdan ularning ustivoligi quyidagicha belgilanadi:

- 1) barcha mantiqiy ifoda qiymati hisoblanadi (True yoki False);
- 2) NOT;
- 3) AND;
- 4) OR.

Imtixonlarda 2 yoki 3 baxo olgan studentlar kodini tanlab olish.

```
SELECT id_st, mark
FROM mark_st
WHERE mark >= 2 AND mark <= 3
```

Imtixonlarda 2 yoki 3 baxo olmagan studentlar kodini tanlab olish.

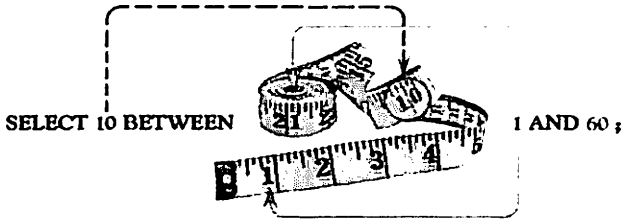
```
SELECT id_st, mark
FROM mark_st
WHERE NOT(mark = 2 OR mark = 3)
```

SQL tilida shart ifodalarini tuzish uchun solishtirish va mantiqiy operatorlardan tashqari yana bir qator maxsus operatorlar qoʻllaniladi. Bu operatorlar dasturlash tillarida mavjud emas. Bu operatorlar:

- *BETWEEN* – qiymatlar diapozoniga tegishligini tekshirish;
- *LIKE* – namuna bilan mosligini tekshirish;
- *IS NULL* – qiymat mavjudmasligini tekshirish;
- *IN* – biror qiymatlar toʻplamiga tegishligini tekshirish;
- *ANY* – qiymatlar toʻplamiga tegishligini shart asosida tekshirish;
- *ALL* – qiymatlar toʻplamiga toʻliq tegishligini tekshirish;
- *EXISTS* – qism soʻrovdan natija qaytarilishini tekshiradi.

Endi ushbu peredikatlarga batafsil toʻxtalib oʻtiladi.

BERWEEN peredikati maydon kiymatini berilgan intervalga kirganligini tekshirish uchun ishlatiladi.

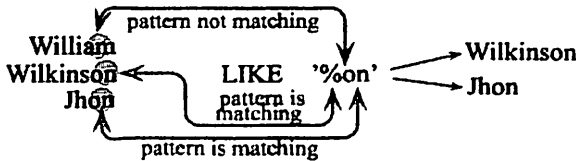


Yukoridagi natijani *BETWEEN* operatoridan foydalanib olish mumkin:

```
SELECT id_st, mark
FROM mark_st
WHERE mark BETWEEN 2 AND 3
```

Birorta xam 2, 3 olmagan talaba haqidagi ma'lumotlar olish uchun *NOT IN* yoki *NOT BERWEEN* orqali yoziladi.

LIKE peridikati faqat simvulli maydonlar uchun qo'llaniladi va maydon qiymati operatorida ko'rsatilgan namunaga mosligini tekshirish imkonini yaratadi.



Namuna quyidagi maxsus simvollaridan tashkil topadi:

- `_` (tagiga chizish belgisi) – bitta ixtiyoriy simvolni bildiradi;
- `%` (foiz belgisi) – ixtiyoriy miqdordagi simvollar ketma – ketligini bildiradi.

9-jadval

Shablon	Tasnif
'abc%'	"abc" xarflar bilan boshlanuvchi ixtiyoriy kator.
'abc_'	"abc" xarflar bilan boshlanuvchi va uzunligi 4 simvulli ixtiyoriy kator
'%z'	"z" xarfi bilan yakunlanuvchi ixtiyoriy kator.
'%mos%'	"mos" so'zi qatorning ixtiyoriy joyida keluvchiqator.

Misol: Familialari A xarfi bilan boshlanuvchi studentlar ro'yxatini tanlab olish.

```
SELECT id_st, surname
FROM student
WHERE surname LIKE 'A%'
```

Jadval atribut(lar)i qiymati mavjud (emas)ligini tekshirishda *NOT NULL* va *IS NOT NULL* operatorlar mavjud.

Misol. Imtixon sanasi ma'lum fanlar ro'yxatini chiqarish.

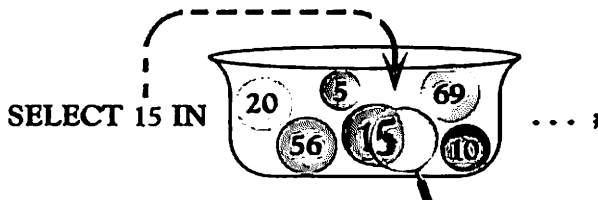
```
SELECT id_st, mark
FROM mark_st
```

WHERE Date NOT NULL

Misol. Imtixon sanasi noma 'lum fanlar ro'yxatini chiqarish.

```
SELECT id_st, mark
FROM mark_st
WHERE Date IS NOT NULL
```

IN peridikati biror qiymatlar to'plamiga tegishlilikni tekshirish uchun ishlatiladi.



IN peridikatining sintaksisi:

```
SELECT [ustunlar... | hisoblashlar ]
FROM [jadval_nomi]
{WHERE | HAVING | {AND | OR}} value [NOT]
```

IN ((a1, a2 [, ...] | qism so'rov));

IN peridikatini oldida NOT kalit so'zini qo'llanilishi uning shartni tekshirilishin teskarisini beradi.

Ekzamenlarda kamida bitta 2 yoki 3 baho olgan studentlarni identifikatorini tanlab chikarish so'rovi.

```
SELECT id_st, mark
FROM mark_st
WHERE mark IN (2,3)
```

ANY (SOME), ALL peridikatlari ham IN kabi vazifani bajarib, lekin ularning imkoniyati IN dan kattaroq hisoblanadi. Bu peridikatlar WHERE yoki HAVING bo'limida berilgan qiymat(lar) bilan predikatdagi to'plam orasida taqqoslash operatori (=, <, !=, >, >=, <=, yoki <=>) orqali ishlatiladi. Shuningdek, peridikatlar TRUE yoki FALSE (rost yoki yolg'on) qiymat qaytaradi.

ANY, ALL peridikatlari qism so'rovlarda ham ishlatiladi.

ANY, ALL, SOME peridikatlarini sintaksislari o'xsh:

```
SELECT [ustunlar... | hisoblashlar ]
FROM [jadval_nomi]
WHERE ifoda shart_ operatori {ALL | ANY | SOME} ( qism so'rov)
```

Misol uchun bizga (20, 56, 5, 15, 69, 10) to'plam berilgan bo'lib, uning eng katta elementi 5, eng kichik elementi 69 ga teng.

ANY peredikatida shartni qanoatlandiruvchi xech bo'lmaganda bitta element bor bo'lsa, rost hisoblanadi.

```
... WHERE 70 > ANY(20, 56, 5, 15, 69, 10) // Natija TRUE
```

```
... WHERE 70 < ANY(20, 56, 5, 15, 69, 10) // Natija FALSE
```

... WHERE 4 > ANY(20, 56, 5, 15, 69, 10) // Natija FALSE

... WHERE 4 < ANY(20, 56, 5, 15, 69, 10) // Natija TRUE

ALL peridikati esa **ANY** perikatiga o'xshash bo'lib, unda to'plamga tegishli barcha element peridikat bilan ifodani o'rtasidagi shartni qanoatlandirishi lozim.

... WHERE 70 > ALL(20, 56, 5, 15, 69, 10) // Natija TRUE

... WHERE 70 < ALL(20, 56, 5, 15, 69, 10) // Natija FALSE

... WHERE 4 > ALL(20, 56, 5, 15, 69, 10) // Natija FALSE

... WHERE 4 < ALL(20, 56, 5, 15, 69, 10) // Natija TRUE

ANY (SOME) va **ALL** peridikatlari oldida **NOT** kalit so'zini foydalanish mumkin.

EXISTS peridikati faqat qism so'rovlarda qo'llaniladi. Uning oldida taqqoslash operatorlari ishlatilmaydi.

Quyida so'rov hech bo'lmaganda bitta natija qaytarsa **EXISTS** operatori **TRUE** qiymatni qaytaradi aks holda ya'ni quyida so'rov natija qaytarmasa **FALSE** qiymatni qaytaradi.

EXISTS peridikatini sintaksisi quyidagicha:

```
SELECT [column_name... | expression1 ]
```

```
FROM [table_name]
```

```
WHERE [NOT] EXISTS (subquery)
```

Amaliyotda **IN**, **ANY (SOME)**, **ALL** va **EXISTS** peridikatlari ko'pchilik hollarda qism so'rovlar yaratishda ishlatilishi sababli, bu peridikatlarga qism so'rovlar paragrafida batafsil misollar orqali to'xtalib ztiladi.

Nazorat savollari

1. SQL tilida ma'lumotlarni manipulyatsiya qilish operatorlari?
2. **SELECT** operatori sintaksisidagi kalit so'zlar ma'nosi?
3. SQL so'rovida ishtiroki shart bo'lgan kalit so'zlar qaysilar?
4. SQL tilida mantiqiy operatorlar qaysilar va qaerda qo'llaniladi?
5. Mantiqiy operatorlar bajarilishidagi bir-biridan ustuvorligini asoslang?
6. Maxsus mantiqiy operatorlarni sanang.
7. So'rovda **BERWEEN** peridikati ekvivalent ifoda qanday?
8. **ANY** va **ALL** peridikatlarini **IN** peridikatidan farqi nimada?
9. Qism so'rovlardada qo'llaniladigan asosiy peridikatlar qaysilar?

4.4. Ko'p jadvalli SQL so'rovlar

Shu paytgacha faqat bitta jadvaldan ma'lumotlarni tanlash misollari keltirildi. Relyatsion amallarga mos bo'lgan bir nechta jadvallardan ma'lumotlar tanlab olishni ham bajarish mumkin. Buning uchun jadvallarning qaysidir atribut qiymatlari bir-biriga mutanosib bo'lishi talab etiladi. Ya'ni atributlarning nomi turlicha bo'lishi mumkin, lekin ulardagi ma'lumot turlari mos bo'lishi kerak. Qoidaga ko'ra, ma'lumotlar tanlab olinadigan jadvallar u yoki bu shaklda bir biri bilan bog'langan. Masalan, birga ko'p va x.k.

Agar jadvallar to'plam deb qaraladigan bo'lsa, umumiy holda to'plamlar ustida asosan quyidagi 3 amal bajariladi.

- ✓ *Kesishma* – ikki yoki undan ko'p turlicha bo'lgan to'plamlarning umumiy elementlarini belgilash uchun foydalaniladi.
- ✓ *Ayirma* – birinchi to'plamda mavjud, lekin ikkinchi to'plamda shu elementlar yo'q bo'lgan boshqa elementlarni tanlashda foydalaniladi
- ✓ *Birlashma* – ikki yoki undan ko'p o'xshash bo'lgan to'plamlarni birlashtirishda ishlatiladi.

Bundan ko'pjadvalli so'rovlarni uch guruhga ajratish mumkin:

- *WHERE* - shartlar orqali jadvallarni bog'lash
- *JOIN* – jadvallarni gorizontal birlashtirish bo'yicha amallar
- *UNION* – so'rov natijalarini vertikal birlashtirish amallari.

Endi jadvallarni o'zaro birlashtirish yoki bog'lash guruhlari bo'yicha amallarni misollar orqali ko'rib o'tiladi.

WHERE orqali jadvallarni bog'lash

Misol tariqasida yuqoridagi kabi kitoblar to'g'risida ma'lumot beruvchi ER-diagramma va uning bog'langan jadvallari berilgan bo'lsin. Diagrammadan ko'rinib turibdiki, Books va Avtor jadvallari id_avtor atributlari orqali, Books bilan Categoriya jadvallari esa id_cat atributlari orqali bog'langan.

1-masala. SQL so'rov orqali barcha kitob nomi, muallifi va yilini matn ko'rinishda chiqarish talab etilgan bo'lsin.

So'rovda bog'lanishni WHERE orqali ifodalash quyidagicha bo'ladi.

```
SELECT Books.id_book, Books.name, Avtor.fio, Books.yil
FROM Books, Avtor
WHERE Books.id_avtor = Avtor.id_avtor
```

Ko'rinib turganidek jadval nomidan iborat prefiksdan foydalanish so'rovni murakkablashtiradi. Bunday murakkablikni bartaraf etish uchun alias (pseudonim) ishlatiladi. Yuqoridagi so'rovni quyidagicha yozish mumkin:

```
SELECT B.id_book, B.name, A.fio, B.yil
```

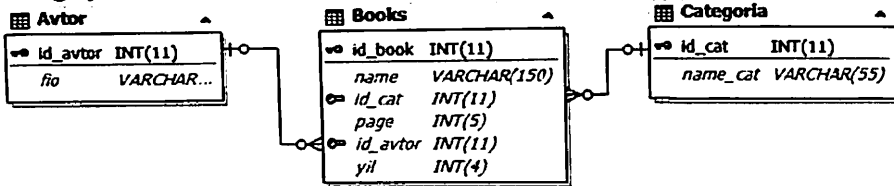
FROM Books B, Avtor A
WHERE B.id_avtor = A.id_avtor

Keltirilgan ko'p jadvali so'rov bir jadvali so'rovdan quyidagilar bilan farq qiladi.

1. FROM sektsiyasida ikkita jadval ko'rsatilgan.
2. jadvallr soni bitta ko'p, shuning uchun ko'rsatilgan maydonlar nomining bir qiymatligi yo'qoladi. Masalan, ko'p xollarda maydonni FROM da ko'rsatilgan jadvallar ro'yxatidagi qaysi jadvaldan olish noma'lum bo'lib qoladi. Maydon nomlarining ko'p qiymatligini bataraf etish uchun maydon nomida perefiks - jadval nomi qo'shimcha qilinadi. Jadval nomi maydon nomidan nuqta bilan ajratiladi.
3. WHERE ifodasida jadvallarni birlashtirish sharti ko'rsatiladi.

Yuqoridagi ikki yoki undan ko'p amallarni o'zaro bog'lashda, jadvallardagi mos quyilayotgan atributlarning qiymati bittalab tekshiriladi. Bu esa so'rov bajarilishi uchun ortiqcha resurs zaruriyatini keltirib chiqaradi.

2-masala. 2005 yildan keyin chop etilgan adabiyotlarni muallifi, kategoriyasi va saxifalari bilan chiqarish talab etilgan bo'lsin.



id_book	name	id_cat	page	id_avtor	yil
1	Odamiylik Mulki	4	980	1	2016
2	Ikki eshik orasi	1	450	2	1996
3	Ikki eshik orasi	1	460	2	2000
4	Ikki eshik orasi	1	380	2	2012
5	Sariq devni minib	5	220	3	1990
6	Sariq devni minib	5	180	3	2005
7	Shaytanat	2	1250	1	2018
8	Dexqon va ayiq	5	100	5	2019
9	Eng katta seor	5	230	5	2020
10	Nafs kishanlari	4	240	1	2015
11	Samum	2	380	1	2012
12	Jannati odamlar	4	210	3	1996
13	Shirin qovunlar mamlakati	5	180	3	1986
14	Qora tol	1	350	4	1988
15	Dala armonlari	1	450	4	1967
16	Mangu buloq	4	360	4	1970
17	Sehrlri qalpoqcha	3	250	3	1964
18	Dunyoning ishlari"	4	280	2	2005
19	Shaytanat	2	960	1	2020

id_avtor	fio
1	Toxir Malik
2	O'tkir Hoshimov
3	Xudayberdi To'xtaboyev
4	Ibraym Yusupov
5	Xalq ogzaki ijodi

id_cat	name_cat
1	Tarixiy
2	Detektiv
3	Fantastik
4	Tarbiyaviy
5	Ertak

SELECT B.name, A.fio, C.name_cat, B.page
FROM Books B, Avtor A, Categoria C

WHERE B.id_avtor = A.id_avtor AND

B.id_cat = C.id_cat AND B.yil > 2005

So'rov natijasi quyidagicha bo'ladi.

name	fito	name_cat	page
Ikki eshik orasi	Utkir Hoshimov	Tarixiy	380
Shaytanat	Toxir Malik	Detektiv	1250
Samum	Toxir Malik	Detektiv	380
Shaytanat	Toxir Malik	Detektiv	960
Odamiylik Mulki	Toxir Malik	Tarbiyaviy	980
Nafs kishanlari	Toxir Malik	Tarbiyaviy	240
Sariq devni minib	Xudayberdi To'xtaboyev	Ertak	180
Dexqon va ayiq	Xalq ogzaki ijodi	Ertak	100
Eng katta sexr	Xalq ogzaki ijodi	Ertak	230

JOIN orqali jadvallarni gorizontaal birlashtirish

Ko'pchilik holatlarda butun MBdan amaliy ahamiyatli natijalarni olishda ko'p jadvalli murakkab so'rovlarni yaratishga to'g'ri keladi. Hozirgi kunda MBBTlarida bir nachta jadvallarning mos atributlari bo'yicha o'zaro bog'lanishni ta'minlovchi standartlar mavjud. Bular zarur ma'lumotlarni turli jadvallardan olishda jadvallarni o'zaro gorizontaal bog'lanishni (JOIN) ta'minlaydi. Bu yerda o'ng tomon va chap tomon jadvallar tegishli atributlari biror shart asosida birlashtiriladi. Gorizontaal bog'lanish bir nechta turga ajratiladi: *Join, Left Join, Right Join, Full Join, Cross Join, Natural Join*.

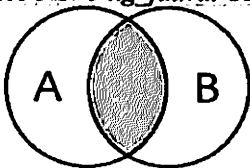
Bizga Authors va Books jadvallari berilib bu jadvallarni vertikal birlashtirishda ularning mos AuthorID va BookID atributlari qaraladi.

Authors	
AuthorID	AuthorName
1	Bruce Eckel
2	Robert Lafore
3	Andrew Tanenbaum

Books	
BookID	BookName
3	Modern Operating System
1	Thinking in Java
3	Computer Architecture
4	Programming in Scala

JOIN – chap va o'ng tomondagi jadvallar birlashtiriladi va mos qo'yilgan atributlar orasidagi shart bajarilganda javob (yozuv) qaytariladi

chap_jadval INNER JOIN o'ng_jadval ON bog'lanish_sharti



INNER JOIN sintaksisi:

```
SELECT maydon nomlari [... n]
```

```
FROM Jadval_1
```

```
INNER JOIN Jadval_2 ON bog'liqlik_sharti
```

Misol:

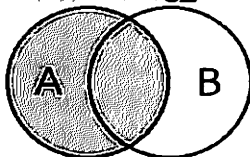
```
SELECT * FROM Authors A
INNER JOIN Books B ON A.AuthorID = B.BookID
```

Natija

A.AuthorID	A.AuthorName	B.BookID	B.BookName
3	Andrew Tanenbaum	3	Modern Operating System
1	Bruce Eckel	1	Thinking in Java
3	Andrew Tanenbaum	3	Computer Architecture

LEFT JOIN – chap tomon jadvalining barcha qatori qaytariladi. Bu qaytarilayotgan qiymatlarga mos o'ng tomon jadvalidan mos qiymatlar olinadi, agar mos qiymati bo'lmasa bo'sh qiymat (NULL) qaytariladi

chap_jadval LEFT OUTER JOIN o'ng_jadval ON bog'lanish_sharti



LEFT JOIN sintaksisi:

```
SELECT maydon nomlari [... n] FROM Jadval_1
LEFT OUTER JOIN Jadval_2 ON bog'liqlik_sharti
```

Misol:

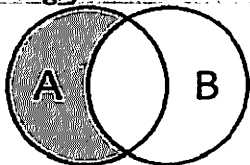
```
SELECT * FROM Authors A
LEFT OUTER JOIN Books B ON A.AuthorID = B.BookID
```

Natija

A.AuthorID	A.AuthorName	B.BookID	B.BookName
1	Bruce Eckel	1	Thinking in Java
2	Robert Lafore	NULL	NULL
3	Andrew Tanenbaum	3	Modern Operating System
3	Andrew Tanenbaum	3	Computer Architecture

LEFT JOIN orqali chap tomon jadvalining o'ng tomon jadvaliga tegishli bo'lmagan qatorlaridan holi qatorlarini ajratib olish mumkin. Ya'ni chap tomon jadvalidan o'ng tomon jadval elementlarini ayirib chap tomon jadval qatorlarini chiqarish mumkin.

chap_jadval LEFT OUTER JOIN o'ng_jadval ON bog'lanish_sharti WHERE o'ng_jadval IS NULL



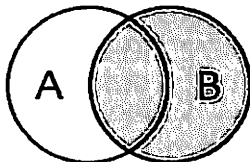
Misol:

```
SELECT * FROM Authors A
```

LEFT OUTER JOIN Books B ON A.AuthorID = B.BookID
WHERE B.BookID IS NULL

RIGHT JOIN – o'ng tomon jadvalining barcha qatori qaytariladi. Bu qaytarilayotgan qiymatlarga mos chap tomon jadvalidan mos qiymatlar olinadi, agar mos qiymati bo'lmasa bo'sh qiymat (NULL) qaytariladi

chap_jadval RIGHT OUTER JOIN o'ng_jadval ON bog'lanish_sharti



shaxlo373@list.ru

RIGHT JOIN sintaksisi:

```
SELECT maydon nomlari [... n] FROM Jadval_1
RIGHT OUTER JOIN Jadval_2 ON bog'liqlik_sharti
```

Misol:

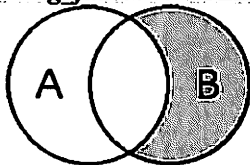
```
SELECT * FROM Authors A
RIGHT OUTER JOIN Books B ON A.AuthorID = B.BookID
```

Natija

A.AuthorID	A.AuthorName	B.BookID	B.BookName
3	Andrew Tanenbaum	3	Modern Operating System
1	Bruce Eckel	1	Thinking in Java
3	Andrew Tanenbaum	3	Computer Architecture
NULL	NULL	4	Programming in Scala

RIGHT JOIN orqali o'ng tomon jadvalining chap tomon jadvaliga tegishli bo'lmagan qatorlaridan holi qatorlarini ajratib olish mumkin. Ya'ni o'ng tomon jadvalidan chap tomon jadval elementlarini ayirib o'ng tomon jadval qatorlarini chiqarish mumkin.

chap_jadval RIGHT OUTER JOIN o'ng_jadval ON bog'lanish_sharti WHERE o'ng_jadval IS NULL



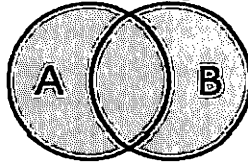
Misol:

```
SELECT * FROM Authors A
RIGHT OUTER JOIN Books B ON A.AuthorID = B.BookID
WHERE A.AuthorID IS NULL
```

FULL JOIN – chap va o'ng tomon jadvallarining barcha qatori qaytariladi. Agar bog'lanish sharti chap va o'ng tomon jadvallarni qanoatlantirsa, ular bir qatorga birlashtiriladi. Bog'lanish sharti

qanoatlantirilmasa, NULL qiymati bog'liq bo'lmagan qatorlarga ko'ra chap yoki o'ng tomon jadvallari o'rniga qo'yiladi

chap_jadval FULL OUTER JOIN o'ng_jadval ON bog'lanish_sharti



FULL JOIN sintaksisi:

```
SELECT maydon nomlari [... n] FROM Jadval_1
FULL OUTER JOIN Jadval_2 ON bog'liqlik_sharti
```

Misol:

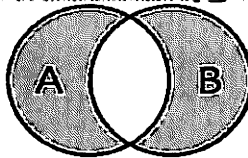
```
SELECT * FROM Authors A
FULL OUTER JOIN Books B ON A.AuthorID = B.BookID
```

Natija

A.AuthorID	A.AuthorName	B.BookID	B.BookName
1	Bruce Eckel	1	Thinking in Java
2	Robert Lafore	NULL	NULL
3	Andrew Tanenbaum	3	Modern Operating System
3	Andrew Tanenbaum	3	Computer Architecture
NULL	NULL	4	Programming in Scala

FULL JOIN orqali chap va o'ng tomondagi jadvallar birlashtirilishi va ularda mos qo'yilgan atributlar orasidagi shart bajarilgandan tashqari (teskari) yozuvlarini chiqarish mumkin.

chap_jadval FULL OUTER JOIN o'ng_jadval ON bog'lanish_sharti WHERE o'ng_jadval IS NULL OR chap_jadval IS NULL

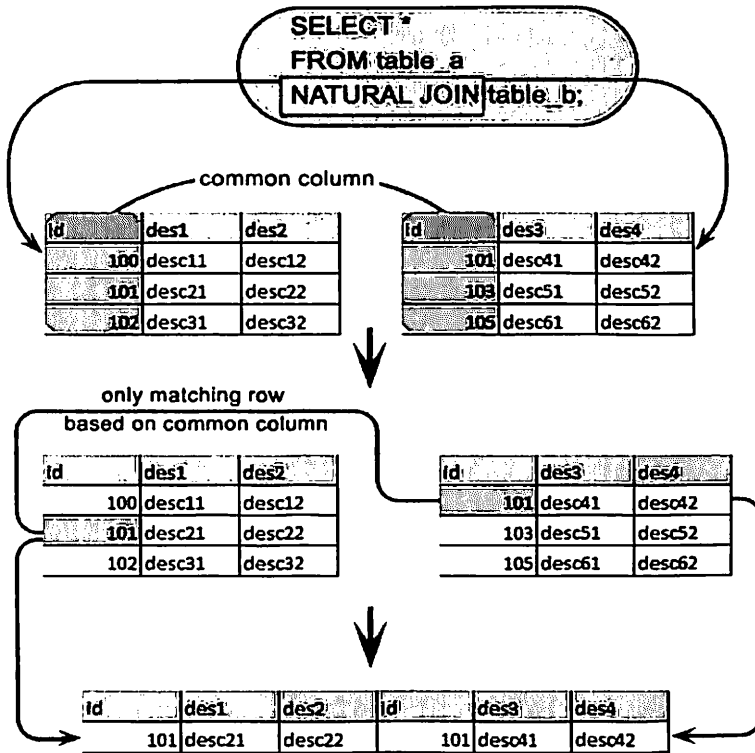


Misol:

```
SELECT * FROM Authors A
FULL OUTER JOIN Books B ON A.AuthorID = B.BookID
WHERE A.AuthorID IS NULL OR B.BookID IS NULL
```

NATURAL JOIN – chap tomon jadvali bilan o'ng tomon jadvalining mos atributlari avtomatik aniqlanib, bu atributlar orasida shart bajarilsa javob qaytariladi. Xuddi INNER JOIN kabi

chap_jadval NATURAL JOIN o'ng_jadval



CROSS JOIN – chap tomon jadvalining har bir yozuvi o'ng tomon jadvalining har bir yozuviga mos quyiladi, ya'ni dekart ko'paytma bo'ladi

chap_jadval CROSS JOIN o'ng_jadval

CROSS JOIN sintaksisi:

```
SELECT maydon nomlari [... n] FROM Jadval_1
CROSS JOIN Jadval_2 ON bog'liqlik_sharti
```

Misol:

```
SELECT *
FROM Authors A
CROSS JOIN Books B
```

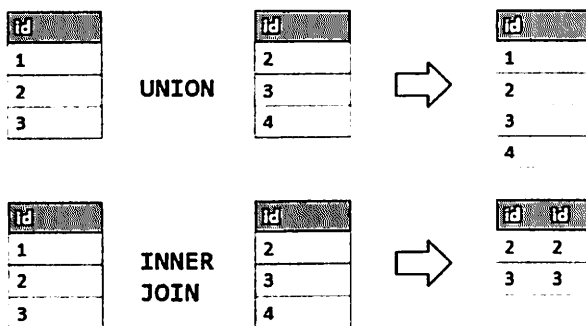
Natija

A.AuthorID	A.AuthorName	B.BookID	B.BookName
1	Bruce Eckel	3	Modern Operating System
1	Bruce Eckel	1	Thinking in Java
1	Bruce Eckel	3	Computer Architecture
1	Bruce Eckel	4	Programming in Scala

2	Robert Lafore	3	Modern Operating System
2	Robert Lafore	1	Thinking in Java
2	Robert Lafore	3	Computer Architecture
2	Robert Lafore	4	Programming in Scala
3	Andrew Tanenbaum	3	Modern Operating System
3	Andrew Tanenbaum	1	Thinking in Java
3	Andrew Tanenbaum	3	Computer Architecture
3	Andrew Tanenbaum	4	Programming in Scala

UNION orqali jadvallarni vertikal birlashtirish

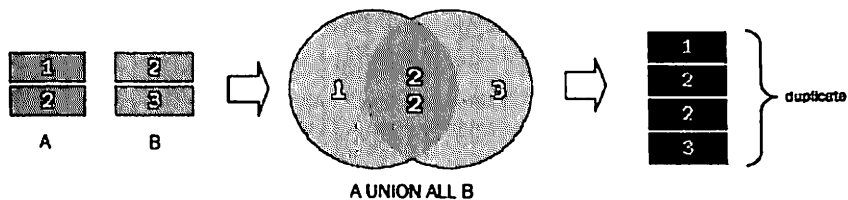
Biz JOIN orqali so'rovda jadvallarni gorizontol birlashtirib natijalarini olishni qarab chiqdik. Endi so'rov natijalarini UNION orqali vertikal birlashtirish amallarini qaraladi. Qisqacha aytganda jadvallarni JOIN yonma-yon, UNION ustma-ust birlashtiradi.



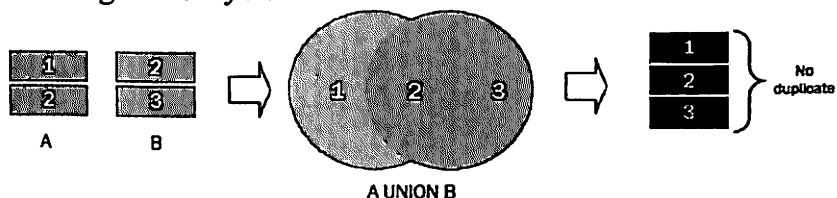
UNION – turlicha jadvallardan qayta ishlangan natija beruvchi, lekin bir xil tuzulmga ega bo'lgan so'rovlarning natijalarini umumlashtiradi. Shunday ekan bunda har bir so'rovning ustunlari soni, ma'lumot turlariga ko'ra ustunlarning joylashuv tartibi mos bo'lishi kerak. Misol uchun, yuqori va ostki so'rovlar uchun sonli ustun ostida son, matnli ustun ostida matn va h.k.

Vertikal jadvallarni birlashtirish amallari MBBT xususiyatlaridan kelib chiqqan holda amalga oshiriladi. Misol uchun MS SQL da quyidagi turlari mavjud: UNION ALL, UNION, EXCEPT, INTERSECT.

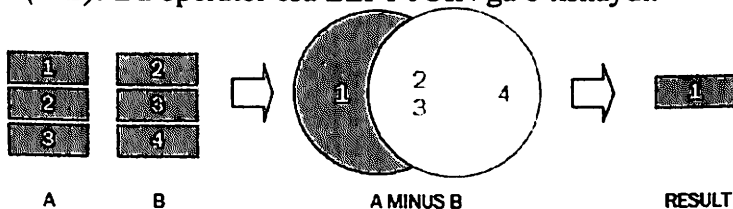
UNION ALL – ikki to'plamdagi barcha qatorlarni birlashtiradi ($A+B$). Bu operator xuddi gorizontol birlashtirishdagi FULL JOIN ga o'xshaydi.



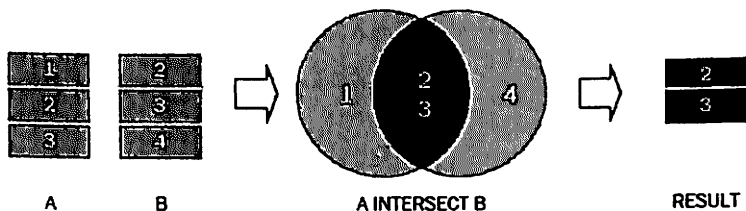
UNION – ikki to'plamdan takrorlanmas (unikal) qatorlarni birlashtiradi. $DISTINCT(A+B)$. Bu operator gorizontaal birlashtirishdagi LEFT JOIN ga o'xshaydi.



EXCEPT \ MINUS – yuqori jadvalning unikal qatorlaridan quyi jadvalning unikal qatorlarini ayirib, yuqori jadval qatorlarini chiqaradi $DISTINCT(A-B)$. Bu operator esa LEFT JOIN ga o'xshaydi.



INTERSECT – yuqori va quyi jadvallarda mavjud unikal qatorlarni natija sifatida qaytaradi $DISTINCT(A\&B)$. Bu esa INNER JOIN ga o'xshaydi.



Misollar.

```
SELECT *
FROM TopTable
```

T1	T2
1	Text 1
1	Text 1
2	Text 2
3	Text 3
4	Text 4
5	Text 5

```
SELECT T1 x,T2 y
FROM TopTable
UNION ALL
SELECT B1,B2
FROM BottomTable
```

x	y
1	Text 1
1	Text 1
2	Text 2
3	Text 3
4	Text 4
5	Text 5
2	Text 2
3	Text 3
6	Text 6
6	Text 6

```
SELECT T1 x,T2 y
FROM TopTable
UNION
SELECT B1,B2
FROM BottomTable
```

x	y
1	Text 1
2	Text 2
3	Text 3
4	Text 4
5	Text 5
6	Text 6

```
SELECT *
FROM BottomTable
```

B1	B2
2	Text 2
3	Text 3
6	Text 6
6	Text 6

```
SELECT DISTINCT *
FROM TopTable
```

T1	T2
1	Text 1
2	Text 2
3	Text 3
4	Text 4
5	Text 5

```
SELECT T1 x,T2 y
FROM TopTable
INTERSECT
SELECT B1,B2
FROM BottomTable
```

x	y
2	Text 2
3	Text 3

```
SELECT T1 x,T2 y
FROM TopTable
EXCEPT
SELECT B1,B2
FROM BottomTable
```

x	y
1	Text 1
4	Text 4
5	Text 5

```
SELECT DISTINCT *
FROM BottomTable
```

B1	B2
2	Text 2
3	Text 3
6	Text 6

Nazorat savollari

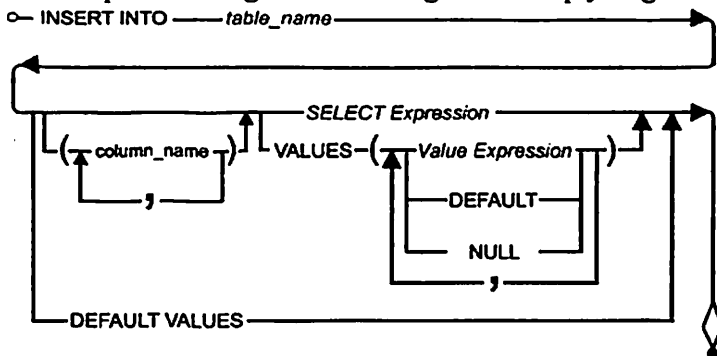
1. SQLda ko'p jadvalli so'rovlar yaratish turlari qanday?
2. Jadvallarni gorizontol birlashtirish buyruqlari qanday?
3. Jadvallarni vertikal birlashtirish buyruqlari qanday?
4. Jadvallarni birlashtirish farqlari va birga ishlatish mumkinmi?
5. Ikkita jadval elementlarining kesishmasidan tashqari elementlar qanday olinadi?

4.5. DML sinfi operatorlari

MBdagi jadvallardagi ma'lumotlarni turli usullar bilan tanlab olish operatorlari bilan yukorida tanishib chikdik. Endi biz SQL tilida jadvallarga ma'lumotlarni kiritish (*INSERT*), yangilash (*UPDATE*) va uchirish (*DELETE*) operatorlarini urganib chikamiz.

INSERT ma'lumotlarni kiritish operatori

INSERT operatori berilgan jadvaldagi ko'rsatilgan mos ustunlariga tashqaridan ma'lumot yozuvlarini qo'shishi amalini bajaradi. Umumiy holda INSERT operatorining sintaksis diagrammasi quyidagicha beriladi:



INSERT operator jadvalga faqat bitta star, oldindan ma'lum sondagi satr va qandaydir so'rov natijasida olingan jadval satrlarini (noma'lum va chekli sondagi) kiritish imkonini beradi.

Jadvalga faqat bitta star kiritish imkonini beradigan INSERT operator sintaksisi quyidagicha yoziladi:

```

INSERT INTO jadval_nomi [(<ustun1, ustun2, ..., ustun n>)]
VALUES (<qiymat1, qiymat2, ..., qiymat n>);
  
```

Bunday sintaksis jadvalga faqat bitta star kiritish imkonini beradi. Agar satrdagi barcha ustunlarga qiymat kiritilayotgan va bu qiymatlar ustunlar ketma-ketligi asosida amalga oshirilayotgan bo'lsa, so'rovda barcha ustunlar nomini ko'rsatish zarur emas.

```

INSERT INTO jadval_nomi
VALUES (<qiymat1, qiymat2, ..., qiymat n>);
  
```

Jadvalga birdan ortiq satr kiritiladigan bo'lsa, VALUES qismidagi har bir satr alohida kavslarda yozilib, ular o'rtasiga vergul qo'yiladi.

```

INSERT INTO jadval_nomi
VALUES (<qiymat1, qiymat2, ..., qiymat n>),
(<qiymat1, qiymat2, ..., qiymat n>);
  
```

Endi INSERT operatoriga misollar qaraladi. Bizga fuqarolar (fuqaro) haqidagi to'ldirilgan jadval, talabalar (student) haqidagi birorta ma'lumot

to'ldirilmagan jadval va stipendiya (STIPENDIA) jadvallari berilgan bo'lsin. Bu jadvallarning tuzilmasi quyidagicha.

FUQARO

ID_FUKARO (AI)	FAMILIYA varchar(20)	ISMI varchar(20)	YOSHI Int(1)	FAOLIYAT varchar(20)	MAXALLA varchar(50)
1	Allayarov	Jasurbek	19	Ishsiz	Nur
2	Axmedov	Shaxzod	21	Xodim	Jaslik
3	Berdanov	Jamshid	20	Talaba	Jaslik
4	Bozorboyev	Sarvarbek	35	Talaba	Nukus
5	Nomozov	Lazizjon	60	Nafaka	Nur

STIPENDIA

BAXO Int(1)	PUL Int(6)
3	350000
4	470000
5	600000

STUDENT

ID_ST (AutoIncrement, PK)	FIO varchar(50)	SPES varchar(70)	KURS Int(1)	BAXO Int(1) Default 3	YOSHI Int(2)	STIPENDIA Int(6)
------------------------------	--------------------	---------------------	----------------	--------------------------	-----------------	---------------------

STUDENT jadvaliga fakat bitta talabani INSERT operatori bilan kiritish: INSERT INTO STUDENT (FIO, SPES, KURS, BAXO, YOSHI, STIPENDIA) VALUES ('Pardabaev Islam', 'DI', 2, 4, 18, 0);

Eslatma. Jadvalga ma'lumot kiritishda avtohisoblagich (AutoIncrement) o'rnatilgan maydon ko'rsatilmaydi va unga qiymat ham berish mumkin emas.

Agar kiritilayotgan ma'lumotlar jadvaldagi atributlar ketma-ketligi bo'yicha bo'lsa, u holda atributlar ko'rsatilishi shart emas. Ma'lumotlarni kiritish operatori birdaniga bir necha satrlarni kiritish imkoniga ham ega. INSERT INTO STUDENT VALUES ('Alimov N, 'KI', 3, 3, 21), ('Azizov B', 'DI', 2, 5, 22, 0);

Agar jadval atributlari o'rin almashib kelsa, u holda kiritilayotgan qiymatlar ham mos o'rin almashishi shart. Shuningdek, atributlarning qabul qiladigan aniqlanmagan qiymat (NULL) va bo'sh qiymat ('') lari farqlanadi.

INSERT INTO STUDENT (FIO, YOSHI, BAXO, SPES, KURS) VALUES ('Zaripov F', 32, 3, '', 4);

Jadvalga ma'lumot kiritishda atribut xossasiga NOT NULL o'rnatilmagan bo'lmaganlaridan tashqari boshqa atributlarga qiymat berish shart emas. Bunday xolatda agar atributga Default xossasiga qiymat berilgan bo'lsa, avtomatik shu qiymat, aks holda NULL qiymat yoziladi. INSERT INTO STUDENT (FIO, YOSHI)

VALUES ('Rajapov Sadulla', 19), ('Sadikov M', 24);

Yuqoridagi so'rov natijalari quyidagicha bo'ladi.

ID_ST	FIO	SPES	KURS	BAXO	YOSHI	STIPENDIA
1	Pardabaev Islam	DI	2	4	18	0
2	Alimov N	KI	3	3	21	0
3	Azizov B	''	2	5	22	0
4	Zaripov F	''	4	3	32	
5	Rajapov Sadulla'			3	19	
6	Sadikov M			3	24	

Agar jadvalni yaratishda ustun yoki atributga majburiy qiymat (NOT NULL) belgisi qo'yilgan bo'lsa, u holda INSERT operatorida joriy

ustunning har bir satriga kiritiladigan qiymatg ko'rsatilishi kerak. Shuning uchun, agar jadvalning hamma ustuni majburiy qiymatli bo'lsa, u holda har bir yangi kiritiladigan satrda barcha ustun uchun qiymat mavjud bo'lishi kerak va bunda ustunlar ro'yxatini ko'rsatish shart emas. Aks holda jadvalda kamida bitta majburiy qiymatli bo'lmagan ustun bo'lsa, u holda albatta ustunlar ro'yxatini ko'rsatish shart bo'ladi.

Qiymatlar ro'yxatida maxsus funktsiyalar va ifodalar ko'rsatilish ham mumkin. Bunda ushbu funktsiyalarning qiymatlari ma'lumotlarni kiritish momentida xisoblangan bo'lishi zarur.

INSERT operatorida ko'plab satrlar kiritilishi SQLda so'rov yaratuvchi va dasturchilarga yaxshi imkoniyat yaratadi. Bunda qiymatlar satri boshqa bir jadvaldan (ma'lum shartlar asosida) SELECT operatori orqali tanlab olinadi. Bu natijani esa to'g'ridan-to'g'ri INSERT operatori bilan boshqa jadvalga kiritish mumkin. Ammo faqat bitta jadvaldan olib yana shu jadvalga yozish mumkin emas.

Masalan FUQARO jadvalidan FAOLIYAT atributi "talaba" bo'lgan yozuvlardan faqat Familiya, ismi va yoshi maydonlarini STUDENT jadvaliga yozish qaralsin. Buni soddalashtirish uchun dastlab lozim bo'lgan tanlash so'rovi tuziladi. Ma'lumki, FUQARO jadvalida FAMILIYA va ISMI alohida atribut, lekin bu STUDENT jadvalida bitta atribut ekanligini nazarda tutib, ular so'rovda birlashtirib yuboriladi.

```
SELECT FAMILIYA || ISMI AS FIO, YOSHI
FROM FUQARO WHERE FAOLIYAT='Talaba'
```

So'rov natijasi quyidagicha bo'ladi:

FIO	YOSHI
Berdanov Jamshid	20
Bozorboyev Sarvarbek	35

Bu so'rov natijasini STUDENT jadvaliga kiritish quyidagicha bo'ladi:

```
INSERT INTO STUDENT (FIO, YOSHI)
```

```
SELECT FAMILIYA || ISMI AS FIO, YOSHI
FROM FUQARO WHERE FAOLIYAT='Talaba';
```

Bu so'rovning natijasi quyidagicha bo'ladi.

ID	ST	FIO	SPES	KURS	BAXO	YOSHI	STIPENDIA
1		Pardabaev Islam	DI	2	4	18	0
2		Alimov N	KI	3	3	21	0
3		Azizov B	"	2	5	22	0
4		Zaripov F	"	4	3	32	
5		Rajapov Sadulla'			3	19	
6		Sadikov M			3	24	
7		Berdanov Jamshid			3	20	
8		Bozorboyev Sarvarbek			3	35	

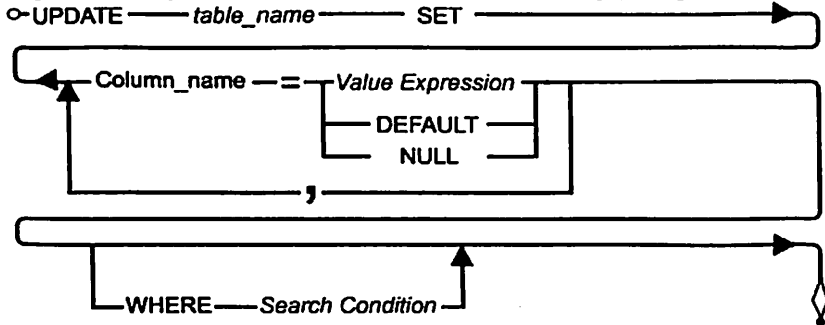
UPDATE ma'lumotlarni yangilash

Ma'lumotlarni yangilash operatori o'zgarish yuz berganda, ya'ni MBning jadvalidagi aniq atributlarni shartsiz yoki ma'lum shartlar bajarilganda o'zgarmas qiymatlarga yangilash vazifasini bajaradi.

Shart(lar)ni qanoatlantirgan xolatda UPDATE operatori bajarilishi jihatidan uch guruhga ajratiladi:

- Ko'rsatilgan atributga o'zgarmas qiymat bilan yangilash;
- Atributni oldingi qiymatini qayta ishlab yangilash;
- To'plamlararo qiymatlaridan moslarini yangilash.

Operatorning umumiy sintaksis diagrammasi quyidagicha beriladi



UPDATE jadval_nomi
 SET ustun_nomi=yangi_qiymat
 [WHERE tanlash_sharti]

Endi bularni aniq misollarda qo'rib chiqiladi.

1) Atributni o'zgarmas qiymat bilan yangilash. Student jadvalidagi kurs atributi bo'sh qiymatga ega va yoshi 22 dan kam bo'lgan talabalarni kursini 1 kurs deb o'zgartirish talab etilsin.

```

UPDATE STUDENT
SET KURS=1 WHERE KURS IS NULL AND YOSHI<=22
  
```

Agar yuqoridagi misolda yangilash so'rovida shart berilmasa, Student jadvalidagi barcha yozuvdagi Kurs atributini 1 ga o'zgartiradi.

Yangilash so'rovida birdaniga bir nechta ustunni yangilash mumkin. Bizdan Yoshi=22 dan katta va Spes atributi mavjud bo'lmagan yozuvlarning Spes="AX", Kurs=0 qilib o'zgartirilsin.

```

UPDATE STUDENT
SET KURS=0, SPES="AX"
WHERE SPES IS NULL AND YOSHI>22
  
```

2) Atribut qiymatini qayta ishlab yangilash. Barcha talabalar kursini bittaga oshirish talab etilgan bo'lsin.

```

UPDATE STUDENT
SET KURS= KURS+1
  
```

Yuqoridagi 3 ta yangilash so'rovining natijasi quyidagicha bo'ladi:

ID ST	FIO	SPES	KURS	BAXO	YOSHI	STIPENDIA
1	Pardabaev Islam	DI	2	4	18	0
2	Alimov N	KI	4	3	21	0
3	Azizov B	''	3	5	22	0
4	Zaripov F	''	5	3	32	
5	Rajapov Sadulla'		2	3	19	
6	Sadikov M	AX	1	3	24	
7	Berdanov Jamshid		2	3	20	
8	Bozorboyev Sarvarbek	AX	1	3	35	

3) To'plamlararo qiymatlaridan moslarini yangilash murakab so'rovdur kiradi. Misol uchun Student jadvalidagi Stipendia atributini Stipendia jadvalidagi Pul qiymati bo'yicha yangilash talab etilsin. Bunda dastlab jadvallardagi mos Baxo atributlari shartda taqqoslanadi va yangilashga Stipendia jadvalini Student atributiga Stipendia jadvalidagi Pul atribut qiymatlari beriladi.

```
UPDATE STUDENT, STIPENDIA
SET STUDENT.STIPENDIA= STIPENDIA.PUL
WHERE STUDENT.BAXO=STIPENDIA.BAXO
```

Bu murakkab yangilash so'rovining natijasi quyidagicha bo'ladi:

ID ST	FIO	SPES	KURS	BAXO	YOSHI	STIPENDIA
1	Pardabaev Islam	DI	2	4	18	470000
2	Alimov N	KI	4	3	21	350000
3	Azizov B	''	3	5	22	600000
4	Zaripov F	''	5	3	32	350000
5	Rajapov Sadulla'		2	3	19	350000
6	Sadikov M	AX	1	3	24	350000
7	Berdanov Jamshid		2	3	20	350000
8	Bozorboyev Sarvarbek	AX	1	3	35	350000

Maxsus shartlar asosida yangilash UPDATE operatorini bitta amal uchun takror shartlarini o'zgartirib bajarishni optimallashtiradi. Misol uchun Student jadvalidagi Stipendia atributini bahoga nisbatan turlicha o'zgartirish, jumladan, 3 baxo 10% kamaytirish, 4 baxo 50% oshirish va 5 baxo 2,5 karra ortishi talab etilgan bo'lsin. Odatda bunga ushbu ko'rinishda so'rov tuziladi:

```
UPDATE STUDENT SET STIPENDIA=0,9*STIPENDIA WHERE BAXO=3
UPDATE STUDENT SET STIPENDIA=1,5*STIPENDIA WHERE BAXO=4
UPDATE STUDENT SET STIPENDIA=2,5*STIPENDIA WHERE BAXO=5
```

Yuqoridagi yangilash so'rovlarni bitta so'rov orqali tuzishda dasturlash tillaridagi kabi IF yoki CASE kalit so'zlaridan foydalanish mumkin.¹

```
UPDATE STUDENT
SET STIPENDIA=
CASE BAXO
```

¹ <https://andreyex.ru/bazy-dannyyx/baza-dannyyx-mysql/11-osnovnyx-primerox-komandy-update-v-mysql/>

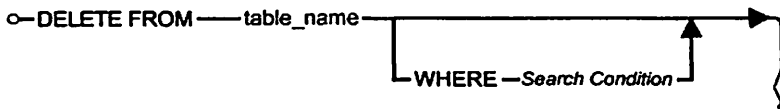
```
WHEN 3 THEN 0,9*STIPENDIA
WHEN 4 THEN 1,5*STIPENDIA
WHEN 5 THEN 2,5*STIPENDIA
ELSE STIPENDIA
```

END

Eslatma. UPDATE operatori bilan jadval yozuvlarni o'zgartirish va DELETE operatori bilan o'chirish amallarini bajarishda ehtiyotkorlik talab etiladi. Ya'ni bu operatorlarda WHERE qismi to'g'ri qo'yishni unutmang. Chunki, bu operatorlar jadval yozuvlariga ta'sir qilib, ularni qayta tiklash muammosini keltirib chiqarishi mumkin. Agar tanlash sharti (WHERE qismi) ko'rsatilmagan bo'lsa, u xolda o'zgartirish amali jadvalning barcha satrlari uchun bajariladi.

DELETE o'chirish operatori

Ma'lumotlarni o'chirish operatori jadvaldan shart bo'lsama hammasini, aks holda shartni qanoatlantiruvchi satr(lar)ni o'chirishi mumkin. Operatorning sintaksisi va diagrammasi quyidagicha beriladi.



```
DELETE FROM jadval_nomi
[WHERE tanlash_sharti]
```

Agar satrlarni tanlash sharti ko'rsatilmasa, u xolda jadvaldagi barcha satrlar o'chiriladi. Natijada ma'lumotlarga ega bo'lmagan bo'sh bo'lgan jadval hosil bo'ladi.

DELETE operatorining WHERE qismida turlicha murakkablikdagi shartlarni, jumladan Like, Beetwen, In kabi peredikatlarni ishlatish mumkin. Endi bunga qaralayotgan jadvallar asosida misollar keltiriladi.

Fukaro jadvalidan ID_FUKARO atributi 2 ga teng yozuvni o'chirish.

```
DELETE FROM Fukaro
WHERE ID_FUK=2
```

Fukaro jadvalidan FAOLIYAT atributi "Nafaka" yoki "Ishsiz" va MAXALLA atributi "Nur" yoki "N" xarfdan boshlanuvchi yozuvlarni o'chirish

```
DELETE FROM Fukaro
WHERE (MAXALLA LIKE "Nur" OR MAXALLA LIKE "N%") AND
(FAOLIYAT IN("Nafaka", "Ishsiz"))
```

Yuqoridagi ikkita so'rov natijasi quyidagicha bo'ladi.

ID FUK	FAMILIYA	ISMI	FAOLIYAT	MAXALLA
1	Allayarov	Jasurbek	Ishsiz	Nur
2	Axmedov	Shaxzod	Xodim	Jaslik
3	Berdanov	Jamshid	Talaba	Jaslik
4	Bozorboyev	Sarvarbek	Talaba	Nukus

X MAXALLA LIKE "Nur" va FAOLIYAT="Ishsiz"

X ID_FUKARO=2

5	Nomozov	Lazizjon	Nafaka	Nukus	X	MAXALLA	LIKE	"N%"	va
↓	↓	↓	↓	↓	↓	FAOLIYAT="Nafaka"			
ID FUK	FAMILIYA	ISMI	FAOLIYAT	MAXALLA					
3	Berdanov	Jamshid	Talaba	Jaslik					
4	Bozorboev	Sarvarbek	Talaba	Nukus					

Endi DELETE operatori orqali boshqa jadval ma'lumotlariga bog'lab o'chirish (murakkab) so'rovini tuzish qaraladi. Misol uchun Fukaro jadvalidagi Familiya va Ismi atributlari birgalikda Student jadvalidagi FIO atributiga mos bo'lgan xolatda yoki Student jadvalidagi Spes atributi " (bo'sh, lekin NULL emas) qiymatga ega bo'lgan yozuvlarni Student jadvalidan o'chirish talab etilsin.

DELETE FROM STUDENT

WHERE FIO IN(

SELECT FAMILIYA || ISMI AS FIO

FROM FUQARO)

OR FIO IS NULL

Bu murakkab o'chirish so'rovining natijasi quyidagicha bo'ladi:

ID ST	FIO	SPES	KURS	BAXO	YOSHI	STIPENDIA
1	Pardabaev Islam	DI	2	4	18	470000
2	Alimov N	KI	4	3	21	350000
5	Rajapov Sadulla'		2	3	19	350000
6	Sadikov M	AX	1	3	24	350000

DELETE operatsiyasini bajarishda unda qatnashgan qism so'rovda satrlar o'chiriladigan jadval ko'rsatilmasligi kerak. Ma'lumotlarni manipulyatsiyalash operatsiyalarining barchasi MBning butunligi tushunchasi bilan bog'langan. Manipulyatsiyalash amallari sintaktik jixatdan to'g'ri bo'lsada butunlik talablari tufayli bajarilmasligi mumkin.

Nazorat savollari

1. SQL tilida ma'lumotlarni manipulyatsiya qilish operatorlari qanday?
2. Jadvallarga ma'lumot kiritish operatori shakllari qanday?
3. Yangilash operatori bajarilishi jihatidan turlari qanday?
4. DELETE operatori sintaksini bayon qiling.
5. DML guruh operatorlarining qaysi birlarida WHERE ishlatish mumkin emas?
6. DML guruh operatorlari tarkibida qaysi peridaktlarni ishlatish mumkin?
7. Jadvaldagi biror usutun ma'lumotlarini o'chirib tashlashda qaysi operator foydalaniladi?

4.6. SQL funksiyalari

Funksiya deganda shunday kiruvchi ma'lumotlarni qayta ishlovchi apparat nazarda tutiladi. Demak, funksiya bir yoki bir nechta argumentdan, ya'ni kiruvchi bir nechta parametrlardan iborat bo'lib, bu argumentlar qayta ishlanib bitta argumentli natija qaytariladi.

SQL tilida funksiyalar asosan 3 toifaga ajratiladi: standart funksiyalar, agregat funksiyalar va saqlanuvchi funksiyalar. Bu yerda standart va agregat funksiyalar SQL tili tarkibiga kirsam, saqlanuvchi funksiyalar foydalanuvchi tomonidan ishlab chiqilib, ayrim hollarda foydalanuvchi funksiyasi ham deyiladi. Shuningdek, standart funksiyalar argument va parametrlari soni oldindan ma'lum bo'lsa, agregat funksiyalarda argumentlar noma'lum sonda bo'ladi. Endi ushbu funksiyalar bilan batafsil tanishib chiqamiz.

Standart funksiyalar ham bir nechta guruhlarga ajratiladi: sana/vaqt va satrli funksiyalar, o'zgartirish va matematik kabi funksiyalar. Bu funksiyalar juda ko'p bo'lib, ularning ayrimlari bilan tanishamiz.

Satrli funksiyalar

Yozuv ko'rinishidagi ustunlarni, biror qiymat bilan solishtirishda yozuvning katta yoki kichik harfda ekanligini bilish muhim, aks holda kerakli natijaga erishish qiyin. Sababi, katta-kichik yozuvlar qo'shtirnoq ichida farqlanadi. LOWER va UPPER funksiyalari orqali solishtirilayotgan yozuvning reestrini kichik yoki katta harflarga o'tkazib, keyin solishtiradi.

LOWER (str) yoki *LCASE(str)* – berilgan satrni kichik xarflarga almashtiradi.

```
SELECT LCASE('QUADRATICALLY'); // Natija: 'quadratically'
```

UPPER (str) yoki *UCASE(str)* – xarflarni katta xarflarga almashtiradi.

```
SELECT UCASE('Hej'); // Natija: 'HEJ'
```

INITCAP (str) – satrdagi har bir so'zning 1-xarfini bosh xarf qiladi.

```
SELECT INITCAP ('tatu nukus'); // Natija: 'Tatu Nukus'
```

CONCAT(str1, str2,...) – argumentda berilgan o'zgaruvchilarni ketma-ketligi bo'yicha birlashtiradi. Kamida bitta argument bo'lishi lozim.

```
SELECT CONCAT('My', 'S', 'QL'); // Natija: 'MySQL'
```

```
SELECT CONCAT('My', NULL, 'QL'); // Natija: NULL
```

```
SELECT CONCAT(14.3); // Natija: '14.3'
```

CONCAT_WS(separator, str1, str2,...) – argumentlar o'rtasiga separator da ko'rsatilgan simvolni qo'yib birlashtiradi.

```
SELECT CONCAT_WS(", ", "First", "Second", "Last");
```


// Natija: 'First,Second,Last'

SELECT CONCAT_WS(",","First",NULL,"Last");// Natija: 'First,Last'

LENGTH(str) – yozuvdagi barcha simollarning sonini aniqlab beradi, bo'sh joy (probel) ham bitta simvol shaklida hisoblanadi.

SELECT LENGTH('text');// Natija: 4

REPLACE(str, from_str, to_str) – yozuvlarda ko'rsatilgan simvollarni boshqasiga almashtirib beradi. Office programmalaridagi "zamenit" buyrug'i vazifasini bajaradi. Uch parametrlilik funktsiyada: *str* – ustun nomi, *from_str* – qaysi almashtiriladigan simvollar va *to_str* – o'rniga qo'yiladigan simvollarni bildiradi.

SELECT REPLACE('www.mysql.com', 'w', 'Ww');

// Natija: 'WwWwWw.mysql.com'

REPEAT(str, count) – *str* yozuvni *count* marta takrorlab ekranga chiqaradi.

SELECT REPEAT('SQL', 3);// Natija: 'SQLSQLSQL'

INSERT(str, pos, len, newstr) – *str* yozuvni *pos* o'ringdan boshlab *len* uzunlikdagi qismini *newstr* yangi satrga almashtiradi.

SELECT INSERT('Quadratic', 3, 4, 'What');// Natija: 'QuWhattic'

ELT(N, str1, str2, str3, ...) – argumentlardan N-indeksdagi soni ko'rsatadi. Agar berilgan N soni 1 dan kichik yoki argumentlar sonidan katta bo'lsa, NULL qiymat qaytariladi.

SELECT ELT(1, 'ej', 'Heja', 'hej', 'foo');// Natija: 'ej'

SELECT ELT(4, 'ej', 'Heja', 'hej', 'foo');// Natija: 'foo'

FIND_IN_SET(str, strlist) – berilgan *str* yozuvni ro'yxat yoki massiv shakliga ega bo'lgan *strlist* to'plamdan qidiradi va uning indeksini aniqlaydi. Bunda *strlist* dagi elementlar vergel bilan ajratilishi lozim.

SELECT FIND_IN_SET('b','a,b,c,d');// Natija: 2

REVERSE(str) – yozuvdagi simvollarning pozitsiyasini teskari tartibga aylantirib ekranga chiqaradi.

SELECT REVERSE('abc');// Natija: 'cba'

Eslatma. Satrli funktsiyalar *SELECT* operatori bilan ishlatilsa faqat ekranga chiqarilayotganda ma'lumotlar o'zgartiriladi, lekin bazadagi ma'lumotlar o'z holida qoladi. Lekin *UPDATE* operatori bilan ishlatilsa, bazadagi ma'lumotlar o'zgartiriladi.

Keyingi keltiriladigan satrli funktsiyalar tahrirlash funktsiyalari deb ham yuritiladi.

INSTR(str, substr) – kerakli simvolni yozuvlar ichidan qidirishga mo'ljallangan. Bunda *str* yozuv tarkibidan *substr* qismiyozuv qidiriladi va natijada topilgan simvolning nechanchi tartibda turgani son ko'rinishida ekranga chiqadi. Natija faqat dastlabki uchragan simvol pozitsiyasini aniqlaydi, keyingilarni qaramaydi.

SELECT INSTR('foobarbar', 'bar');// Natija: 4

```
SELECT INSTR('xbar', 'foobar'); // Natija: 0
```

Hisobotlar tayorlashda uni o'qish qulay bo'lishi juda muhim. Shuning uchun har bir chiqariladigan simvol yoki sonlar bir hil tartibda chiqishi lozim. Ya'ni uzun so'zlar ham qisqa so'zlar ham bir hil uzunlikda chiqishi maqsadga muvofiqdir. SQL da bu vazifani "PAD" funktsiyasi bajaradi, faqat bu funktsiyani oldiga qisqa bo'lgan so'zlar to'ldirilishi uchun qaysi tomondan boshlab to'ldirilishini ko'rsatuvchi L(left) yoki R(right) harflar qo'shiladi. To'ldirish deganda berilgan yozuvni chap yoki o'ng tomoniga ko'rsatilgan simvol soniga yetmagan simollarni birlashtirish tushuniladi. Ko'pchilik standart hollarda to'ldirish lozim bo'lgan simvolga probel yoki pastki chiziq qo'yiladi.

LPAD(str, len, padstr) va *RPAD(str, len, padstr)* - bu yerda *str* yozuvni *len* uzunlikda *padstr* simvollar bilan to'ldiradi

```
SELECT LPAD('hi',4,'?'); // Natija: '??hi'
```

```
SELECT RPAD('hi',5,'?'); // Natija: 'hi???'
```

LEFT(str, len) va *RIGHT(str, len)* funktsiyalar berilgan *str* yozuvni chap va o'ng tomonidan *len* uzunlikdagi simvollarni kesib oladi.

```
SELECT LEFT('foobarbar', 5); // Natija: 'fooba'
```

```
SELECT RIGHT('foobarbar', 4); // Natija: 'rbar'
```

SUBSTRING(str, pos, len) yoki *SUBSTR (str, pos, len)* – berilgan *str* yozuvning *pos* o'rindagi simvoldan boshlab *len* uzunlikda kesib oladi.

SUBSTRING(str FROM pos FOR len) – yuqoridagiga ekvivalent funktsiya.

```
SELECT SUBSTRING('Quadratically',5,6); // Natija: 'ratica'
```

```
SELECT SUBSTRING('Quadratically',5); // Natija: 'ratically'
```

```
SELECT SUBSTRING('foobarbar' FROM 4); // Natija: 'barbar'
```

SUBSTRING_INDEX(str, delim, count) – *str* yozuvning *delim* ajratuvchi qism simvollarini *count* marta kelishidan keyingi (oldingi) simvollarini o'chiradi. Agar *count* musbat son bo'lsa, keyin aks holda manfiy son bo'lsa oldin kelishini bildiradi.

```
SELECT SUBSTRING_INDEX('www.mysql.com', '.', 2);
```

```
// Natija: 'www.mysql'
```

```
SELECT SUBSTRING_INDEX('www.mysql.com', '.', -2);
```

```
// Natija: 'mysql.com'
```

LTRIM (str, [<substr >]) va *RTRIM (str, [<substr >])* – berilgan *str* yozuvni chap va o'ng tomonidan *substr* qisimsatrni o'chirib tashlaydi yoki tozalaydi. Agar *substr* qisimsatr ko'rsatilmasa probellardan tozalaydi.

```
SELECT LTRIM(' barbar'); // Natija: 'barbar'
```

```
SELECT RTRIM('barbar '); // Natija: 'barbar'
```

TRIM (*str*, [<*substr* >]) – yuqoridagi funktsiyalar kabi, lekin berilgan *str* yozuvning o'ng va chap tomonidan *substr* qismatni o'chirib tashlaydi. Bu funktsiyaning to'liq yozilishi quyidagicha:

TRIM([[**BOTH** | **LEADING** | **TRAILING**] [*remstr*] **FROM**] *str*)

```
SELECT TRIM(' bar '); // Natija: 'bar'
SELECT TRIM(LEADING 'x' FROM 'xxxbarxxx'); // Natija: 'barxxx'
SELECT TRIM(BOTH 'x' FROM 'xxxbarxxx'); // Natija: 'bar'
SELECT TRIM(TRAILING 'xyz' FROM 'barxyz'); // Natija:
```

CONV(*N*, *from_base*, *to_base*) – berilgan *N* sonni *from_base* sanoq sistemasidan *to_base* sanoq sistemasiga o'tkazadi (konvertatsiya qiladi).

```
SELECT CONV("a",16,2); // Natija: '1010'
SELECT CONV("6E",18,8); // Natija: '172'
SELECT CONV(-17,10,-18); // Natija: '-H'
SELECT CONV(10+"10"+"10"+0xa,10,10); // Natija: '40'
```

BIN(*N*) yoki **CONV**(*N*, 10, 2) – o'nlikdan ikkilik sanoq sistemasiga o'tkazadi.

```
SELECT BIN(12); // Natija: '1100'
```

OCT(*N*) yoki **CONV**(*N*, 10, 2) – o'nlikdan sakkizlik sanoq sistemasiga o'tkazadi.

```
SELECT OCT(12); // Natija: '14'
```

HEX(*N_or_S*) – agar *N_or_S* arument son shaklida berilgan bo'lsa, to'g'ridan-to'g'ri 16 lik sanoq sistemasiga o'tkazadi. Agar *N_or_S* satr shaklida berilsa, uning har bir simvolini oldin ikkilik sanoq sistemaga o'tkazib, keyin uni 16likga aylantiradi.

```
SELECT HEX(255); // Natija: 'FF'
SELECT HEX("abc"); // Natija: 616263
SELECT 0x616263; // Natija: "abc"
```

Matematik funktsiyalar

ABS(*X*) – sonning absalyut qiymati.

```
SELECT ABS(2); // Natija: 2
SELECT ABS(-32); // Natija: 32
```

SIGN(*X*) – argumentning ishorasiga ko'ra (-1,0,1) qiymatlardan birtasini qaytaradi.

```
SELECT SIGN(-32); // Natija: -1
SELECT SIGN(0); // Natija: 0
SELECT SIGN(234); // Natija: 1
```

MOD(*N*, *M*) yoki % – *N* sonini *M* soniga bo'lgandagi qoldiqni qaytaradi.

```
SELECT MOD(234, 10); // Natija: 4
SELECT 253 % 7; // Natija: 1
SELECT MOD(29,9); // Natija: 2
```

FLOOR(*X*) – *X* dan oshmaydigan eng katta butun sonni qaytaradi.

SELECT CEILING(1.23); // Natija: 2

SELECT CEILING(-1.23); // Natija: -1

CEILING(X) – X dan kam bo'lmagan eng kichik butun sonni qaytaradi.

SELECT CEILING(1.23); // Natija: 2

SELECT CEILING(-1.23); // Natija: -1

ROUND(X) – argument yaqin bo'lgan butun songa yaxlitlaydi.

SELECT ROUND(-1.23); // Natija: -1

SELECT ROUND(-1.58); // Natija: -2

SELECT ROUND(1.58); // Natija: 2

ROUND(X, D) – X kasr sonni D-chi kars songa yaxlitlaydi. Agar $D=0$ bo'lsa **ROUND(X)** funktsiya bilan ekvalent bo'ladi

SELECT ROUND(1.298, 1); // Natija: 1.3

SELECT ROUND(1.298, 0); // Natija: 1

POW(X, Y) yoki **POWER(X, Y)** – X argumentni Y darajasini qaytaradi.

SELECT POW(2,2); // Natija: 4.000000

SELECT POW(2,-2); // Natija: 0.250000

SQRT(X) – X argumentning kvadrat ildizini hisoblaydi.

SELECT SQRT(4); // Natija: 2.000000

SELECT SQRT(20); // Natija: 4.472136

RAND() yoki **RAND(N)** – 0 va 1 oralig'idagi tasodifiy son. Agar N berilsa 0 bilan 0.N oralig'idagi tasodifiy son qaytariladi.

SELECT RAND(); // Natija: 0.9233482386203

SELECT RAND(20); // Natija: 0.15888261251047

SELECT RAND(20); // Natija: 0.15888261251047

SELECT RAND(); // Natija: 0.63553050033332

SELECT RAND(); // Natija: 0.70100469486881

LEAST(X,Y,...) – kamida ikkita argument qatnashishi lozim bo'lib, funktsiya eng kichik argumentni qaytaradi.

SELECT LEAST(2,0); // Natija: 0

SELECT LEAST(34.0,3.0,5.0,767.0); // Natija: 3.0

SELECT LEAST("B","A","C"); // Natija: "A"

GREATEST(X,Y,...) – eng kichik argument qaytariladi.

SELECT GREATEST(2,0); // Natija: 2

SELECT GREATEST(34.0,3.0,5.0,767.0); // Natija: 767.0

SELECT GREATEST("B","A","C"); // Natija: "C"

Sanali (date) funktsiyalar

Ayrim sanali funktsiyalarda sanaga bog'liq atamalar (Datepart)ni maxsus qisqartilgan simvollar orqali foydalaniladi (jadvalga qarang).

Sanali funktsiyalari davr yoki vaqt bilan ishlashga mo'ljallangan bo'lib, ularning qatoriga quyidagilar kiradi:

Datepart	Qisqartmasi	Interval
Year — yil	yy, yyyy	1753 -9999
Quarter — kvartal	qq, q	1-4

Month — oy	mm, m	1-12
Dayofyear — yil kuni	dy, y	1-366
Day — kun	dd, d	1-31
Week — hafta (yilning xaftasi)	wk, ww	1-7
Hour — soat	hh	0-23
Minute — minut	mi, n	0-59
Second — sekund	ss, s	0-59
Millisecond - millisekund	ms	0-999

DATEADD(*datepart*, *number*, *date*) – sanali ma'lumot qaytaradi. Funktsiya date parametridagi sananing mos pozitsiyasiga datepart da ko'rsatilganga mos number sonini qo'shadi.

```
SELECT DATEADD(day, 7, current_timestamp);
// Natija: joriy vaqtga 7 kun qo'shadi.
```

DATEDIFF(*datepart*, *startdate*, *enddate*) – funktsiya ikkita startdate va enddate vaqt oralig'ini datepart bo'yicha hisoblab qaytaradi.

```
SELECT DATEDIFF(wk, '2020-05-24', '2020-05-30');// shanba kuni (0) chiqaradi
```

DATENAME(*datepart*, *date*) – funktsiya datepart da ko'rsatilgan simvolga mos berilgan sananing nomini chiqaradi.

```
SELECT DATENAME(weekday, '2020-05-27')+' '+
DATENAME(day, '2020-05-27')+' '+
DATENAME(month, '2020-05-27')+' '+
DATENAME(year, '2020-05-27');
// Natija: Wednesday, 27 May 2020
```

DATEPART(*datepart*, *date*) – funktsiya datepart da ko'rsatilgan simvolga mos berilgan sananing qiymatini chiqaradi.

```
SELECT 'Haftaning ' + DATEPART(weekday, '2020-05-27')+' kuni. ' +
DATEPART(year, '2020-05-27')+' yil ' +
DATEPART(month, '2020-05-27')+' oyining ' +
DATEPART(day, '2020-05-27')+' -kuni';
// Natija: Haftaning 3 kuni. 2020 yil 05 oyning 27-kuni
```

DAY(*date*) – berilgan sanadagi kunni chiqaradi;

MONTH(*date*) – berilgan sanadagi oyni chiqaradi;

YEAR(*date*) – berilgan sanadagi yilni chiqaradi;

GETDATE() – kompyuterdagi joriy vaqtni chiqaradi.

Eslatma: sana andozasi Amerika formati uchun ko'rsatilgan: oy/kun/yil. Keltirilgan sanali funktsiyalarni SELECT operator orqali bajarish, shuningdek, ular ustida taqqoslash amallarini, ya'ni =, <, >, >=, <=, <>, !<(kichik emas), !>(katta emas), !=(teng emas). Albatta amallar mantiqiy bajarilishining ustivorligi qavslar orqali belgilanadi.

Tizimli funktsiyalar

Bu turdagi funktsiyalar MB va uning tarkibi haqida ma'lumotlarni olish uchun qo'llaniladi. SQL Server ushbu tizimli funktsiyalar mavjud.

- *COL_LENGTH*(jadval, maydon) – maydon uzunligini chiqaradi;
- *DATALENGTH*(ifoda) – ifoda uzunligini qaytaradi;
- *GETAHSINULL*(MB nomi) – MBda NULL qiymati foydalanilgan yoki yo'qligini chiqaradi;
- *IDENTINCR*(jadval) – jadvaldagi hisoblagich (AI) maydonni oxirgi qiymatini ko'rsatadi;
- *IDENT_SEED*(tablitsa) – jadvaldagi hisoblagich maydonning boshlang'ich qiymatini ko'rsatadi;
- *ISDATE*(ifoda) – agar ifoda atama bo'lsa, atamani, aks holda nolni qaytaradi;
- *ISNUMERIC*(ifoda) – agar ifoda sonli bo'lsa, uni, aks holda nolni qaytaradi;
- *NULIFF*(ifoda1, ifoda2) – agar ifoda1 bilan ifoda2 teng bo'lsa nolni qaytaradi.

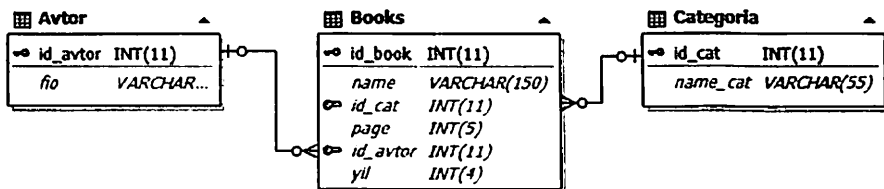
Agregat funktsiyalar

Yuqorida standart funktsiyaning ayrimlari bilan tanishib chiqildi. Bundan xulosa qilish mumkinki, SQL tilidagi standart funktsiyalar jadvaldagi biror atribut(lar)ga nisbatan ishlatilib, u har bir qator uchun bajariladi. Ma'lumki jadvalda yozuvlar ko'p sonli bo'lishi bilan birga ular chegaralangan bo'ladi. Statistik ma'lumotlarni ishlab chiqarishda, ayniqsa sonli toifadagi ma'lumotlarni qayta ishlaganda berilgan jadvaldagi tanlangan yozuvlardagi biror maydon qiymatlari ustida amallar bajarishga to'g'ri keladi. Ya'ni jadval bo'yicha biror maydon ma'lumotlarini umumlashtirish orqali natijalar olinadi. Bunday amallarni bajaruvchi funktsiyalar SQL tilida agregat yoki statik funktsiyalar, guruhli yoki jamlovchi natijaviy funktsiyalar deb yuritiladi.

Agregat funktsiyalar deb shunday funktsiyaga aytiladi ekanki, unga kiruvchi chekli, ammo noma'lum sonli elementlardan tashkil topgan massiv shaklida argumentlarni umumlashtirib qayta ishlovchi funktsiyalarga aytiladi. Misol uchun bularga ko'rsatilgan maydondagi barcha sonlar yig'indisi, o'rtacha qiymati, eng katta yoki eng kichik

qiymatlarni topish funktsiyalar kiradi. Endi agregrat funktsiyalar bilan misollar orqali tanishib chiqamiz.

Bizga KITOB MB va undagi jadval ma'lumotlari berilgan bo'lsin.



Books

id_book	name	id_cat	page	id_avtor	yil
1	Odamiylik Mulki	4	980	1	2016
2	Ikki eshik orasi	1	450	2	1996
3	Ikki eshik orasi	1	460	2	2000
4	Ikki eshik orasi	1	380	2	2012
5	Sariq devni minib	5	220	3	1990
6	Sariq devni minib	5	180	3	2005
7	Shaytanat	2	1250	1	2018
8	Dexqon va aviq	5	100	5	2019
9	Eng katta sexr	5	230	5	2020
10	Nafs kishanlari	4	240	1	2015
11	Samum	2	380	1	2012
12	Jannati odamlar	4	210	3	1996
13	Shirin qovunlar mamlakati	5	180	3	1986
14	Qora tol	1	350	4	1988
15	Dala armonlari	1	450	4	1967
16	Mangu buloq	4	360	4	1970
17	Sehrli qalpoqcha	3	250	3	1964
18	Dunyoning ishlari"	4	280	2	2005
19	Shaytanat	2	960	1	2020

Avtor

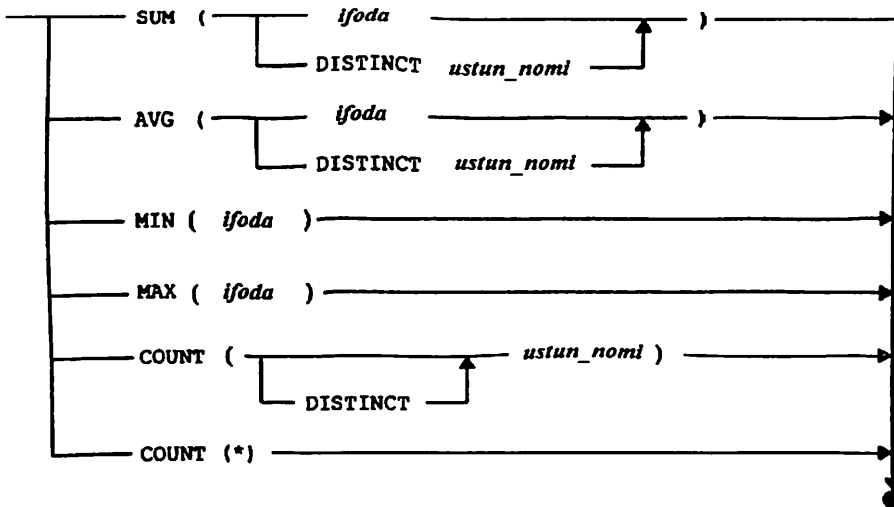
id_avtor	fio
1	Toxir Malik
2	O'tkir Hoshimov
3	Xudayberdi To'xtaboyev
4	Ibraym Yusupov
5	Xalq ogzaki ijodi

Categoriya

id_cat	name_cat
1	Tarixiy
2	Detektiv
3	Fantastik
4	Tarbiyaviy
5	Ertak

Agregat funktsiyalarni *Function([ALL | DISTINCT] atribut)* kabi umumiy shaklida ifodalash mumkin. Agregat funktsiya ushbu talablarga javob beradi.

- Agar so'rovda shart qo'yilmasa (WHERE) ustundagi barcha qiymatlarni qabul qiladi.
- Agar so'rovda shart qo'yilsa (WHERE) sharni qanoatlandiruvchi qatorlardagi ustunlarning qiymatlari qabul qilinadi.
- ALL kalit so'zi (odatda yozilmaydi) - hisoblashda barcha qatorlardagi qiymatlar ishtirok qiladigan xolatda ishlatiladi.



DISTINCT kalit so'zi ustun qabul qiladigan qiymatlardan faqat takrorlanmas bo'lgan qiymatlari ustida hisoblash yuritiladi.

SUM([ALL | DISTINCT] atribut) – ko'rsatilgan ustundagi sonlarning yig'indisini hisoblaydi.

Ms 1: Books da berilgan kitoblarning saxifalari yig'indisini topish.

```
SELECT SUM(page) FROM Books // Natija: 7910
```

Ms 2: Faqat 3 muallif kitoblarining saxifalari yig'indisini topish.

```
SELECT SUM(page) FROM Books WHERE id_avtor=3 // Natija: 1040
```

MIN([ALL | DISTINCT] atribut) – ustundagi sonlarning eng kichik qiymati topiladi.

Ms 3: 1 muallif kitoblaridan eng kichik bo'gan kitob saxifasini topish.

```
SELECT MIN(page) FROM Books WHERE id_avtor=1 // Natija: 240
```

MAX([ALL | DISTINCT] atribut) – ustundagi sonlarning eng katta qiymati topiladi.

Ms 4: 1 muallif kitoblaridan eng katta bo'gan kitob saxifasini topish.

```
SELECT MAX(page) FROM Books WHERE id_avtor=1 // Natija: 1250
```

AVG([ALL | DISTINCT] atribut) – ustundagi sonlarning o'rtacha qiymati hisoblanadi.

Ms 5: 1 muallif kitoblarining o'rtacha saxifasini topish.

```
SELECT AVG(page) FROM Books WHERE id_avtor=1 // Natija: 762
```

COUNT([ALL | DISTINCT] atribut) ()* – jadvaldagi qatorlar soni.

Agar ustun nomi (*atribut*) ko'rsatilsa, ushbu ustunning NULL bo'lmagan

qiymatli qatorlar soni topiladi. Agar ustun nomi o'rniga "*" ishlatilsa, jadvalning hajmini, ya'ni qatorlari sonini topadi.

Ms 6: 1 va 3 mualliflarning barcha kitoblari sonini topish.

```
SELECT COUNT(*) FROM Books WHERE id_avtor IN(1,3) // Natija: 10
```

Statistik funktsiyalarning sintaksis diagrammasi quyidagicha bo'ladi.

Keltirilgan agregat funktsiyalar barcha MBBT uchun umumiy hisoblanadi. MBBTning turiga qarab boshqa agregat funktsiyalar ham mavjud. Misol uchun MS ACCESS da VAR, VARP, STDEV, STDEVP funktsiyalari bor. Mazkur agregat funktsiyalar ko'pchilik hollarda guruhli amallar bilan qo'llaniladi.

Nazorat savollari

1. SQL funktsiyalari qaysi jixatlari bo'yicha turlarga ajratiladi?
2. Satrli ma'lumotlar bilan ishlovchi funktsiyalarni vazifalari bo'yicha guruhlariga ajrating?
3. Matematik funktsiyalar qanday maqsadlarda qo'llaniladi?
4. Sanali funktsiyalarda asosiy parametrlari nimalardan iborat?
5. Tizimli funktsiyalar qaysi tomon (mijoz-server) kompyuteri ma'lumotlari bo'yicha ishlaydi?
6. Agregat funktsiyalarning mohiyati va turlari?
7. Statik funktsiyalar bilan agregat funktsiyalarning farqi nimadi?

4.7. SQL da guruhli operatorlar

GROUP BY - guruhlash operatori

Oldin keltirib o'tilganidek, SQL so'rosidagi SELECT operatorining kengaytirilgan sintaksisida *GROUP BY* va *HAVING* parametrlari mavjud.

GROUP BY tanlangan atributdagi qiymatlarning takrorlanmasligini ta'minlaydi, yoki agregat funktsiyalar orqali tanlangan ustun qiymatlari bo'yicha guruhlariga ajratib beradi. Guruhlar ko'rsatilgan atributning bir hil bo'lgan qiymatlari asosida amalga oshiriladi. Agar guruhlashda agregat funktsiyalar ishlatilsa, ular shu guruxga tegishli qiymatlar uchun o'rinli buladi. *GROUP BY* kalit so'zidan keyin guruhlanadigan atributlar ro'yxati vergul bilan ajratilgan holda ko'rsatiladi.

here is some columns of 'book_mast' table

pub_id	no_page
P003	201
P001	300
P002	510
P004	600
P006	345
P005	247
P002	165
P006	88
P007	350
P007	400
P005	225
P008	350
P001	165

average of 'P002' group
= (510+165)/2=337.5

pub_id	AVG(no_page)
P001	232.5000
P002	337.5000
P003	148.0000
P004	460.0000
P005	236.0000
P006	216.5000
P007	375.0000
P008	267.5000

average of 'P001' group
= (300+165)/2=232.5

the mention process will be repeat for rest of 'pub_id'

© w3resource.com

Ms 7: Barcha kategoriyadagi kitoblarning sonini topish.

```
SELECT Categoria.id_cat, Categoria.name_cat, COUNT(*) AS SONI
FROM Books
INNER JOIN Categoria ON Books.id_cat = Categoria.id_cat
```

Natija:

id_cat	name_cat	SONI
1	Tarixiy	5
2	Detektiv	3
3	Fantastik	1
4	Tarbiyaviy	5
5	Ertak	5

Ms 8: "Toxir Malik" muallifligidagi kitoblarning saxifalari yig'indisini kategoriyalar bo'yicha hisoblash.

```
SELECT Categoria.id_cat,
       Categoria.name_cat,
       SUM(Books.page) AS SAXIFALAR
FROM Books
INNER JOIN Categoria ON Books.id_cat = Categoria.id_cat
INNER JOIN Avtor ON Books.id_avtor = Avtor.id_avtor
WHERE Avtor.fio = 'Toxir Malik'
```

Natija:

id_cat	name_cat	SAXIFALAR
2	Detektiv	2590
4	Tarbiyaviy	1220

Ms 8: Tarbiyaviy kategoriyada ijod qilgan muallflarning eng katta saxifali kitoblari ro'yxatini chiqarish.

```
SELECT Avtor.id_avtor, Avtor.fio, MAX(Books.page) AS SAXIFALAR
FROM Books
```

```
INNER JOIN Categoria ON Books.id_cat = Categoria.id_cat
INNER JOIN Avtor ON Books.id_avtor = Avtor.id_avtor
WHERE Categoria.name_cat = 'Tarbiyaviy'
GROUP BY Avtor.id_avtor
```

Natija:

id_avtor	fi	SAXIFALAR
1	Toxir Malik	980
2	Utkir Hoshimov	280
3	Xudayberdi To'xtaboyev	210
4	Ibraym Yusupov	360

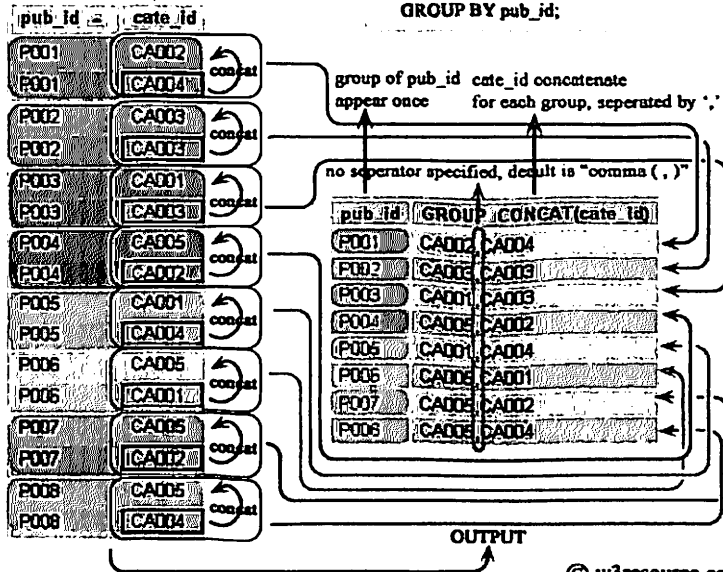
GROUP_CONCAT funksiyasi

Guruhli amallardagi SUM, AVG, MIN, MAX agregat funksiyalar sonli ma'lumotlar uchun qo'llaniladi. Guruhli amallarda faqat sonli emas, balki barcha turdagi ma'lumot toifalari uchun mo'ljallangan maxsus GROUP_CONCAT funksiyasi mavjud. Bu funksiya xuddi SQL tilidagi CONCAT funksiyaga o'xshab, u guruhga tegishli bo'lgan ma'lumotlarni birliktiradi.

here is some columns of book_mast table have displayed with order by on pub_id

code executed on that table

```
SELECT pub_id,
GROUP_CONCAT(cate_id)
FROM book_mast
GROUP BY pub_id;
```



Ms 9: Mualliflarning ijud qilgan kategoriyalarini bir qiymat sifatida chiqarish.

```
SELECT Avtor.id_avtor, Avtor.fi,
GROUP_CONCAT(DISTINCT Categoria.name_cat) AS KATEGORIYALARI
FROM Books
```

```
INNER JOIN Avtor ON Books.id_avtor = Avtor.id_avtor
INNER JOIN Categoria ON Books.id_cat = Categoria.id_cat
GROUP BY Avtor.id_avtor
```

Natija:

id_avtor	fio	KATEGORIYALARI
1	Toxir Malik	Detektiv, Tarbiyaviy
2	Utkir Hoshimov	Tarixiy, Tarbiyaviy
3	Xudayberdi To'xtaboyev	Tarbiyaviy, Ertak, Fantastik
4	Ibraym Yusupov	Tarixiy, Tarbiyaviy
5	Xalq ogzaki ijodi	Ertak

HAVING – guruhlarni tanlash sharti

HAVING ifodasida *GROUP BY* orqali shakllantirilgan har bir gruppaga qo'yiladigan shartlarni ifodalaydi. *WHERE* so'zida shartlar har bir yozuv bo'yicha olib boriladigan bo'lsa, *HAVING* da maydonni guruhlanishida hosil bo'lgan maydon elementlari to'plami ustida bajarilgan amallarning natijasi bo'yicha amalga oshiriladi. Ya'ni bunda oldin *GROUP BY* kalit so'zi orqali guruhlariga ajratiladi, keyin ushbu guruh qiymati aniqlanadi va oxirida shart tekshiriladi. *HAVING* parametri ishlatilgan so'rovlar murakkab so'rovlar turkumiga kiradi.

Ms 10: Tarbiyaviy kategoriyada ijod qilgan muallflarning 300 sahifadan kam bo'lmagan eng katta saxifali kitoblari ro'yxatini chiqarish.

```
SELECT Avtor.id_avtor, Avtor.fio,
       MAX(Books.page) AS SAXIFALAR
FROM Books
INNER JOIN Categoria ON Books.id_cat = Categoria.id_cat
INNER JOIN Avtor ON Books.id_avtor = Avtor.id_avtor
WHERE Categoria.name_cat = 'Tarbiyaviy'
GROUP BY Avtor.id_avtor
HAVING MAX(Books.page)>300
```

Natija:

id_avtor	fio	SAXIFALAR
1	Toxir Malik	980
4	Ibraym Yusupov	360

Ms 11: Uchta kitobdan kam bo'lmagan muallflarning kitoblar soni, saxifalar summasi va saxifalarning o'rtacha qiymatini mualliflar kesimida topish.

```
SELECT A.id_avtor,
       A.fio,
       COUNT(B.id_book) AS SONI,
       SUM(B.page) AS SAXIFALAR,
       AVG(B.page) AS URTACHA
FROM Books B
```

```
INNER JOIN Avtor A ON B.id_avtor = A.id_avtor
GROUP BY A.id_avtor
HAVING COUNT(B.id_book) >= 3
```

Natija:

id_avtor	fio	SONI	SAXIFALAR	URTACHA
1	Toxir Malik	5	3810	762,0000
2	Utkir Hoshimov	4	1570	392,5000
3	Xudayberdi To'xtaboyev	5	1040	208,0000
4	Ibraym Yusupov	3	1160	386,6667

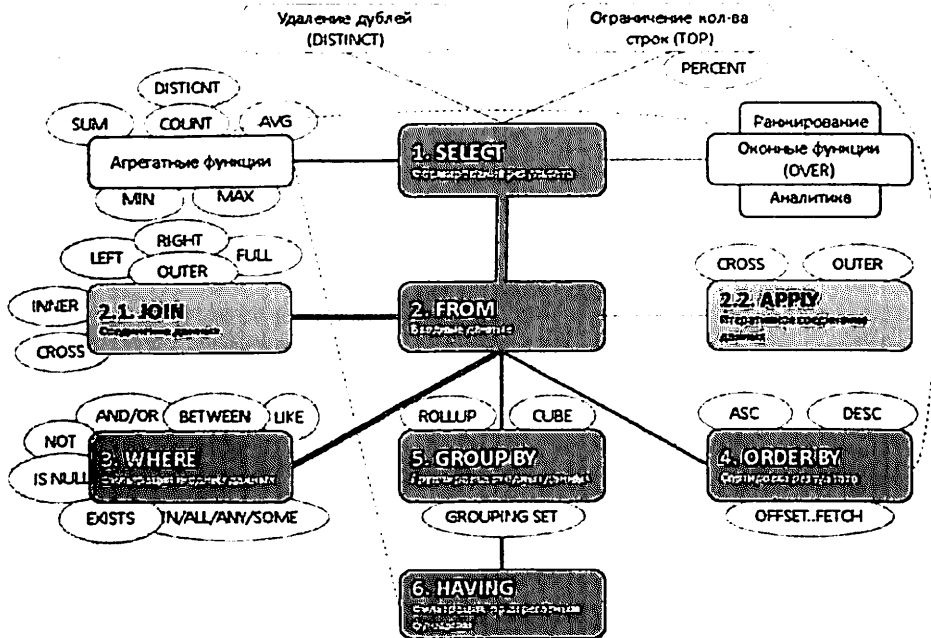
HAVING shartini GROUP VY parametrsiz ham ishlatish mumkin. Bunda natija faqat bitta qatorga chiqariladi. Tajribada bu kam qo'llaniladi.
Ms 12: Uchta kitobdan kam bo'lmagan mualliflarning kitoblar soni, saxifalar summasi va saxifalarning o'rtacha qiymatini topish.

```
SELECT A.id_avtor, A.fio,
COUNT(B.id_book) AS SONI,
SUM(B.page) AS SAXIFALAR,
AVG(B.page) AS URTACHA
FROM Books B
INNER JOIN Avtor A ON B.id_avtor = A.id_avtor
GROUP BY A.id_avtor
HAVING COUNT(B.id_book) >= 3
```

Natija:

id_avtor	fio	SONI	SAXIFALAR	URTACHA
1	Toxir Malik	19	7910	416,3158

So'rovni qayta ishlash tartibi va tavsifi



<https://minyurov.com/2015/03/15/talking-sql/>
 /*(8)*/ SELECT /*(9)*/ DISTINCT /*(11)*/ TOP
 /*(1)*/ FROM
 /*(3)*/ JOIN
 /*(2)*/ ON
 /*(4)*/ WHERE
 /*(5)*/ GROUP BY
 /*(6)*/ WITH {CUBE | ROLLUP}
 /*(7)*/ HAVING
 /*(10)*/ ORDER BY
 /*(11)*/ LIMIT

VT "Virtual Table" degan ma'noni anglatadi va so'rovni qayta ishlash paytida turli xil ma'lumotlar qanday ishlab chiqarilganligini ko'rsatadi.

1. FROM: FROM bo'limidagi birinchi ikkita jadval o'rtasida dekart ko'paytma (cross join) amalga oshiriladi va natijada VT1 virtual jadvali shakllantiriladi.

2. ON: ON filtri VT1 ga qo'llaniladi. VT2 ga faqat TRUE bo'lgan satrlar qo'shiladi.

3. OUTER (qo'shilish): agar OUTER JOIN ko'rsatilgan bo'lsa (CROSS JOIN yoki INNER JOIN ga qarama-qarshi), saqlangan jadval yoki jadvallardagi mos bo'lmagan satrlar VT2 satrlariga tashqi satrlar sifatida qo'shiladi va VT3 shakllantiriladi. Agar FROM satrida ikkitadan ko'p jadvallar paydo bo'lsa, barcha jadvallar qayta ishlangunga qadar, oxirgi jadval natijasi va FROM paragrafidagi keyingi jadval o'rtasida 1 dan 3 gacha bo'lgan qadamlar takrorlanadi.

4. WHERE: WHERE filtri VT3 jadvaliga qo'llaniladi faqat TRUE bo'lgan satrlar VT4 jadvaliga kiritiladi.

5. GROUP BY: VT4 jadvaling satrlari GROUP BY bandida ko'rsatilgan ustunlar ro'yxati asosida guruhlanadi va natijada VT5 jadvali yaratiladi.

6. CUBE | ROLLUP: Superguruhlar (guruhlarning guruhi) VT5 qatorlaridan qo'shiladi, VT6 yaratiladi.

7. HAVING: HAVING filtri VT6 ga qo'llaniladi va faqat TRUE bo'lgan guruhlar VT7 ga kiritiladi.

8. SELECT: SELECT ro'yxati bajariladi va VT8 yaratiladi.

9. DISTINCT: VT8 dan takrorlanuvchi qatorlar olib tashlanadi va VT9 hosil bo'ladi.

10. ORDER BY: VT9 satrlari ORDER BY da ko'rsatilgan ustunlar ro'yxati bo'yicha tartiblanadi kursor hosil bo'ladi (VT10).

11. TOP: Belgilangan qatorlar soni yoki foizlari VT10 boshidan tanlanadi. VT11 jadvali hosil qilinadi va murojat qiluvchiga qaytariladi. LIMIT operatori ham SQL lahjalaridagi (shevalaridagi) Postgres va Netezza kabi TOP operatori bajargan funktsiyani bajaradi.

Nazorat savollari

1. SQLda guruhli amallar mohiyatini tushuntiring?
2. Guruhli amallarda qaysi toifadagi funktsiyalar qo'llaniladi?
3. HAVING kalit so'zini GROUP BY bilan ishlatig sharmi va nima uchun?
4. HAVING sharti bilan WHERE shartining farqini izohlang?

5-BOB. MURAKKAB SQL SO'ROVLAR VA TASAVVURLAR

5.1. SQL da murakkab va ichki so'rovlar

Quyi so'rovlar bo'yicha tushunchalar

Amaliyotda oddiy SQL so'rovlari orqali MB ma'lumotlarini qayta ishlab, yetarli darajadagi hisobotlarni chiqarish imkoniyati kam. MBda bir nechta jadvallarni birgalikda qayta ishlash, guruhli amallarni bajarish, to'plamlar ustida natijalar olish SQL da murakkab so'rovlar hisoblanadi. Murakkab so'rovlar qatoriga asosan so'rov ichida so'rovlar yoki so'rov osti so'rovlar, quyi yoki qism so'rovlar ham ishlatiladi..

Quyi so'rovlar (podzapro, subquery) asosan SELECT konstruktsiyasining *SELECT*, *FROM*, *WHERE* va *HAVING* bo'limlarda ishlatilib, *ORDER BY* va *GROUP BY* bo'limlarida qo'llanish mumkin emas. Shuningdek, quyi so'rovlar *SELECT*, *INSERT*, *UPDATE*, *DELETE* operatorlari ichida yoki boshqa so'rov ichida joylashishi mumkin. Ko'pchilik hollarda quyi so'rovlar SELECT operatorini WHERE bo'limida ishlatiladi.

Quyi so'rovlar bilan solishtirish amallari ($=$ | $<>$ | $<$ | $<=$ | $>$ | $>=$) qo'llanish mumkin bo'lib, u albatta taqqoslash amalining o'ng tomonida joylashadi. Taqqoslash operatorlari o'z o'rnida *IN*, *ANY*, *ALL* yoki *EXISTS* kabi ko'p qatorli operatorlar bo'lishi ham mumkin.

<pre>SELECT select_list FROM table WHERE expr operator</pre>	<pre>(SELECT select_list FROM table)</pre>
--	--

Qism so'rovlar aytilishi jixatdan ikki qismdan iborat: ichki (*inner query*) va tashqi so'rov (*outer query*). Ichki so'rovlar asosiy SQL so'rovi tarkibida qavs ichiga yozilgan boshqa SQL so'rovdur. Ichki so'rov o'z navbatida tartib bo'yicha yana ichki so'rovlardan iborat bo'lishi mumkin. Tashqi so'rov bir yoki bir nechta ichki so'rovlarni qamrab oladi. Ichki so'rov tashqi so'rovdan oldin bajarilib, uning natijasi tashqi so'rovga uzatiladi. Quyi so'rovlarning namunaviy sintaksis ko'rinishi quyidagicha

Bizga juda oddiy bzlgan talaba va uning baholarini to'g'risidagi jadvallari berilgan bo'lsin:

student		marks	
StudentID	Name	StudentID	Total marks
V001	Abe	V001	95
V002	Abhay	V002	80
V003	Acelin	V003	74
V004	Adelphos	V004	81

Ms 1. Studentid = V002 talabaga qaraganda yaxshiroq bahoga ega bo'lgan talabalar ro'yxatini chiqarish talab etilsin.

Ma'lumki, biz albatta V002 talabaning bahosini bilmaymiz. Shuning uchun mazkur masalani yechishda ikkita so'rov yaratish lozim. Birinchi so'rov Studentid = V002 talabaning bahosini aniqlasa, ikkinchi so'rov birinchi so'rov natijasidan yuqori bo'lgan natijali talabalar ro'yxatini chiqaradi.

Birinchi so'rov:

```
SELECT * FROM marks
WHERE studentid = 'V002';
```

Natija

StudentID	Total marks
V002	80

Ushbu so'rov natijasi 80 ga teng bo'lib, undan foydalangan holda 80 bahodan yuqori baho olgan talabalar ro'yxatini chiqaramiz:

Ikkinchi so'rov:

```
SELECT a.studentid, a.name, b.total_marks
FROM student a, marks b
WHERE a.studentid = b.studentid AND b.total_marks > 80;
```

Natija:

studentid	name	total marks
V001	Abe	95
V004	Adelphos	81

Qo'yilgan masala yuqoridagi ikkita so'rov orqali yechildi. Ko'rinib turganidek, birinchi so'rovni natijasini olib boshqa so'rovga qo'yish yechim yo'lining unchalik ham to'g'ri emasligini bildiradi. Shuning uchun bunday xolatlarda qism so'rovlardan foydalanish maqbul hisoblanadi. Demak, xuddi shu masaladagi ikkita so'rovni bir birining ichiga joylashtirgan holda qarash to'g'ri bo'ladi. Bu qism so'rov quyidagicha:

```
SELECT a.studentid, a.name, b.total_marks
FROM student a, marks b
WHERE a.studentid = b.studentid AND
      b.total_marks >
```

*(SELECT total_marks FROM marks
WHERE studentid = 'V002');*

5-BOB. MURAKKAB SQL SO'ROVLAR VA TASAVVURLAR

5.1. SQL da murakkab va ichki so'rovlar

Quyida so'rovlar bo'yicha tushunchalar

Amaliyotda oddiy SQL so'rovlari orqali MB ma'lumotlarini qayta ishlab, yetarli darajadagi hisobotlarni chiqarish imkoniyati kam. MBda bir nechta jadvallarni birgalikda qayta ishlash, guruhli amallarni bajarish, to'plamlar ustida natijalar olish SQL da murakkab so'rovlar hisoblanadi. Murakkab so'rovlar qatoriga asosan so'rov ichida so'rovlar yoki so'rov osti so'rovlar, quyi yoki qism so'rovlar ham ishlatiladi..

Quyida so'rovlar (podzapro, subquery) asosan SELECT konstruktsiyasining *SELECT*, *FROM*, *WHERE* va *HAVING* bo'limlarda ishlatilib, *ORDER BY* va *GROUP BY* bo'limlarida qo'llanish mumkin emas. Shuningdek, quyi so'rovlar *SELECT*, *INSERT*, *UPDATE*, *DELETE* operatorlari ichida yoki boshqa so'rov ichida joylashishi mumkin. Ko'pchilik hollarda quyi so'rovlar *SELECT* operatorini *WHERE* bo'limida ishlatiladi.

Quyida so'rovlar bilan solishtirish amallari (= | <> | < | <= | > | >=) qo'llanish mumkin bo'lib, u albatta taqqoslash amalining o'ng tomonida joylashadi. Taqqoslash operatorlari o'z o'rnida *IN*, *ANY*, *ALL* yoki *EXISTS* kabi ko'p qatorli operatorlar bo'lishi ham mumkin.

```
SELECT select_list
FROM table
WHERE      expr operator (SELECT select_list
                          FROM table)
```

Qism so'rovlar aytilishi jixatdan ikki qismdan iborat: ichki (*inner query*) va tashqi so'rov (*outer query*). Ichki so'rovlar asosiy SQL so'rovi tarkibida qavs ichiga yozilgan boshqa SQL so'rovdur. Ichki so'rov o'z navbatida tartib bo'yicha yana ichki so'rovlardan iborat bo'lishi mumkin. Tashqi so'rov bir yoki bir nechta ichki so'rovlarni qamrab oladi. Ichki so'rov tashqi so'rovdan oldin bajarilib, uning natijasi tashqi so'rovga uzatiladi. Quyida so'rovlarning namunaviy sintaksis ko'rinishi quyidagicha

Bizga juda oddiy bzlgan talaba va uning baholarini to'g'risidagi jadvallari berilgan bo'lsin:

student	
StudentID	Name
V001	Abe
V002	Abhay
V003	Acelin
V004	Adelphos

marks	
StudentID	Total marks
V001	95
V002	80
V003	74
V004	81

Ms 1. Studentid = V002 talabaga qaraganda yaxshiroq bahoga ega bo'lgan talabalar ro'yxatini chiqarish talab etilsin.

Ma'lumki, biz albatta V002 talabani bahosini bilmaymiz. Shuning uchun mazkur masalani yechishda ikkita so'rov yaratish lozim. Birinchi so'rov Studentid = V002 talabani bahosini aniqlasa, ikkinchi so'rov birinchi so'rov natijasidan yuqori bo'lgan natijali talabalar ro'yxatini chiqaradi.

Birinchi so'rov:

```
SELECT * FROM marks
WHERE studentid = 'V002';
```

Natija

StudentID	Total marks
V002	80

Ushbu so'rov natijasi 80 ga teng bo'lib, undan foydalangan holda 80 bahodan yuqori baho olgan talabalar ro'yxatini chiqaramiz:

Ikkinchi so'rov:

```
SELECT a.studentid, a.name, b.total_marks
FROM student a, marks b
WHERE a.studentid = b.studentid AND b.total_marks > 80;
```

Natija:

studentid	name	total marks
V001	Abe	95
V004	Adelphos	81

Qo'yilgan masala yuqoridagi ikkita so'rov orqali yechildi. Ko'rinib turganidek, birinchi so'rovni natijasini olib boshqa so'rovga qo'yish yechim yo'lining unchalik ham to'g'ri emasligini bildiradi. Shuning uchun bunday xolatlarda qism so'rovlardan foydalanish maqbul hisoblanadi. Demak, xuddi shu masaladagi ikkita so'rovni bir birining ichiga joylashtirgan holda qarash to'g'ri bo'ladi. Bu qism so'rov quyidagicha:

```
SELECT a.studentid, a.name, b.total_marks
FROM student a, marks b
WHERE a.studentid = b.studentid AND
      b.total_marks >
```

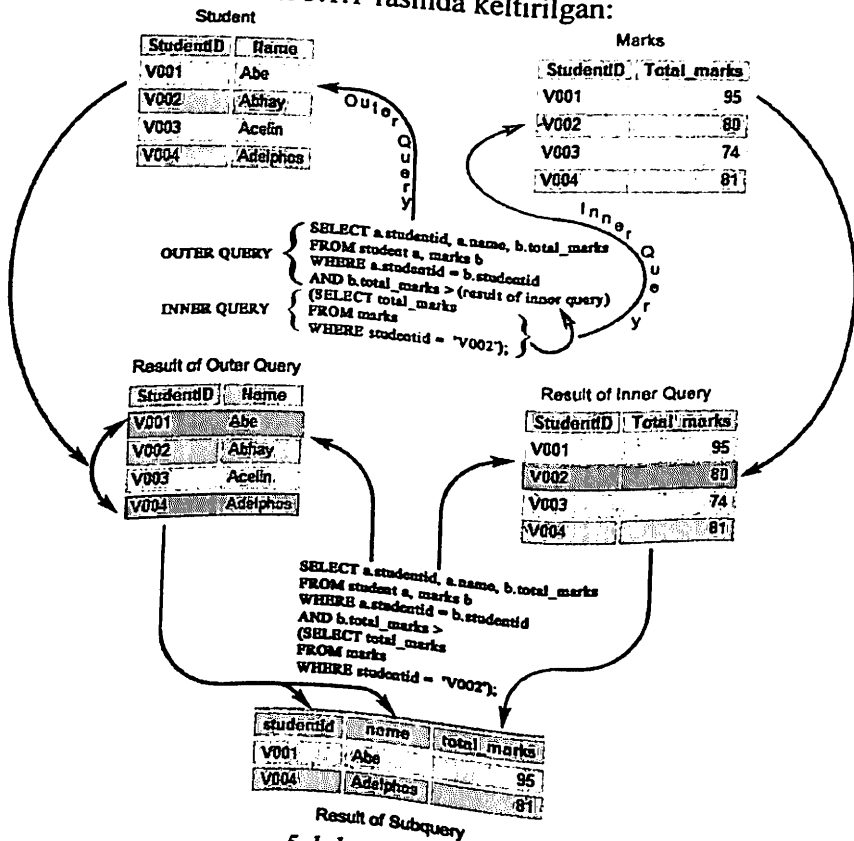
*(SELECT total_marks FROM marks
WHERE studentid = 'V002');*

Ko'rinib turibdiki natija ikkala holatda ham bir xil. Ushbu masalaga tugilgan qism so'rov sxemasi 2-rasmda keltirilgan.

SELECT operatori odatdagi (oddiy) so'rovlarni, ichki va tashqi so'rovlarni boshlash uchun ishlatiladi. Ichki so'rovdagi SELECT operatori ham odatdagi SELECT operatori kabi bo'ladi. Ichki so'rovning umumiy sintaksisi quyidagicha:

```
(SELECT [DISTINCT] qism_so'rovdagi_tanllangan_atributlar
FROM {jadval_nomi | tasavvur_nomi}
    {jadval_nomi | tasavvur_nomi} ...
[WHERE qidirish_sharti]
[GROUP BY guruhlanuvchi_ustun [, guruhlanuvchi_ustun] ...]
[HAVING qidirish_sharti])
```

So'rov sxemasi 5.1.1-rasmda keltirilgan:



5.1.1-rasm. So'rov sxemasi

Quyi so'rovlarga talablar va turlari

Qism so'rovlar tuzishda quyidagi ko'rsatmalarga amal qilish lozim:

- Qism so'rovlar qavs ichida berilishi shart;
- Qism so'rov taqqoslash operatoring o'ng tomonida joylashadi;
- Qism so'rov ichida natijalarni manipulyatsiya (o'zgartirib) bo'lmaydi. Shuning uchun quyi so'rov tarkibida ORDER BY ishlatilmaydi;
- Asosiy (tashqi) so'rovda ORDER BY ishlatish mumkin;
- Bir qatorli qism so'rovlarda bittalik operatorlar ishlatiladi. Ya'ni, bitta qiymat qaytaruvchi qism so'rov oldiga bittalik taqqoslash operatorlari qo'llaniladi;
- Agar ichki so'rov tashqi so'rovga null qiymatini qaytarsa, tashqi so'rovdagi WHERE kalit so'zida taqqoslash operatori ishlatilganda birorta qator qaytarmaydi.

Quyida so'rovlarning bir nechta turlari mavjud:

- Bir qatorli quyi so'rovlar: 0 yoki 1 qator qiymati qaytariladi.
- Ko'p qatorli quyi so'rovlar: 1 yoki undan ko'p qator qiymatini qaytaradi.
- Korrelyatsion qism so'rovlar: tashqi so'rovning bir yoki bir nechta ustunlariga murojatlar amalga oshiriladi.
- Ichki quyi so'rovlar: quyi so'rovlar boshqa bir quyi so'rov ichida qo'llaniladi.

Odatda INSERT operatori konstruktsiyasida ma'lumot kiritiluvchi jadvalga (ustunlari ko'rsatilgan bo'lishi mumkin) VALUES kalit so'zidan keyin mos qiymatlar to'plami kiritiladi. Qism so'rovlar bilan jadvalga ma'lumot kiritishda VALUES kalit so'zi ishlatilmaydi.

INSERT operatorida kism so'rovning umumiy sintaksisi quyidagicha:

```
INSERT INTO jadval_nomi [(ustun1 [, ustun2])]
SELECT [ * ] ustun1 [, ustun2]
FROM jadval1 [, jadval2 ]
[WHERE shart operatori];
```

Bunda mantiq shundan iboratki, oldin kiritiladigan ma'lumotlar SELECT operatori konstruktsiyasiga ko'ra boshqa jadvallardan tanlab olinadi. Keyin aniqlangan ma'lumotlar to'plami ko'rsatilgan jadvalga INSERT qilinadi. Misol uchun:

```
INSERT INTO neworder
SELECT * FROM orders
WHERE advance_amount IN(2000,5000);
```

UPDATE operatorida bir qatorli qism so'rovlardan qaytarilgan natijani ma'lum shartlar yangilanuvchi ustunga berish mumkin. UPDATE operatorida kism so'rovning umumiy sintaksisi quyidagicha:

```
UPDATE jadval_nomi SET yangilanuvchi_ustun = yangi_qiymat
[WHERE taqqoslash operator]
```

```
(SELECT ustun_nomi
FROM jadval_nomi
[WHERE ...])
```

Misol:

```
UPDATE neworder
SET ord_date='15-JAN-10'
WHERE ord_amount<
```

```
(SELECT MIN(ord_amount) FROM orders);
```

Agar qaytariluvchi natija ko'p ko'p qatorli bo'lgan xolatda, albatta shart ishlatiladi. Amaliyotda shunday masalalar uchraydiki, unda boshqa jadvallardan olingan ko'p qiymatli natijani mos maydonlar orqali yangilashdir. Buning sintaksisi quyidagicha:

```
UPDATE jadval1 SET
jadval1.ustun1 = jadval2.ustun1,
jadval1.ustun2 = qiymat,
```

...

```
FROM jadval1
[INNER | LEFT] JOIN jadval2 ON jadvallar_bog'lanishi
WHERE shart_ifodasi;
```

DELETE operatorida qism so'rovlarni sintaksisi quyidagicha:

```
DELETE FROM jadval_nomi
[WHERE taqqoslash operator]
(SELECT ustun_nomi
FROM jadval_nomi
[WHERE ...])
```

Misol:

```
DELETE FROM neworder
WHERE advance_amount<
```

```
(SELECT MAX(advance_amount)
FROM orders);
```

Yuqorida select, insert, update, delete operatorlarida ichki so'rovlarni sintaksis tuzilishini ko'rib o'tildi. Murakkab so'rovlarda qism so'rovlarni ishlatish, ayniqsa, ob'ekti bitta jadvaldan iborat bo'lgan oddiy so'rovlarni ko'p jadvallilik shaklida tuzish, jumladan, bir vaqtda ko'p jadvaldan ma'lumot o'chirish, boshqa jadvalning ma'lumotini ikkinchi jadvalga ko'chirish yoki ma'lumot kiritish kabi so'rovlarni tuzish talab etiladi. Albatta murakkab so'rovlarni tuzishning umumlashgan sintaksisi yoki algoritmi mavjud bo'lmasdan, bu ko'proq MB foydalanuvchisining malakasiga bog'liq bo'ladi. Endi qism so'rovlarning turlari bilan batafsil tanishib chikiladi.

Bir qatorli quyi so'rovlar

Bir qatorli quyi so'rovlar tashqi so'rovga 0 yoki 1 qator qiymatni qaytaradi. Bunday qism so'rovlar tashqi so'rovlarning WHERE, HAVING yoki FROM bo'limlarida ishlatilishi mumkin.

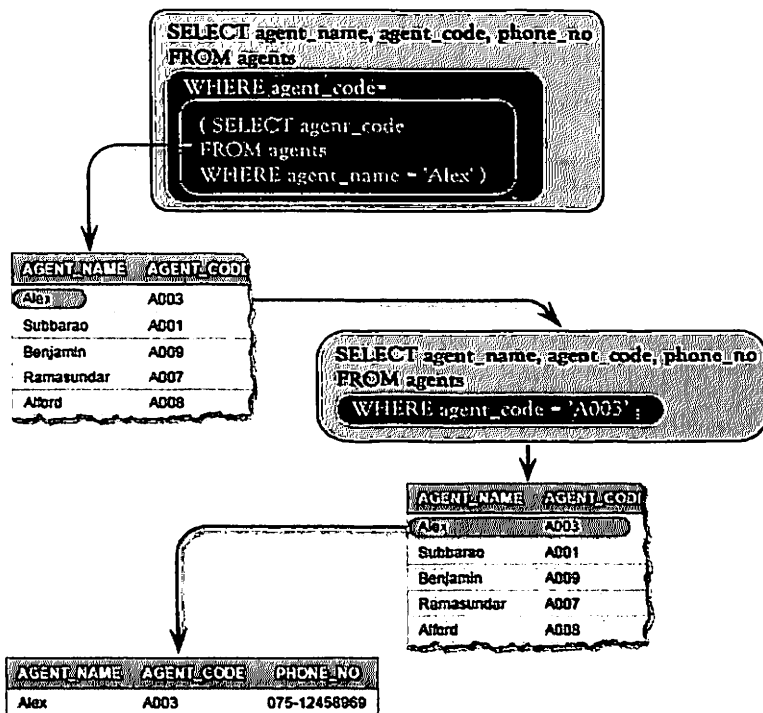
A) *WHERE bo'limida quyi so'rovni ishlatish:* Quyi so'rovning *WHERE bo'limida* ishlatilishiga quyidagi jadval asosida misol ko'ramiz:

```
SELECT agent_name, agent_code, phone_no
FROM agents
WHERE agent_code =
  (SELECT agent_code
   FROM agents
   WHERE agent_name = 'Alex');
```

Yuqoridagi SQL kod agents jadvalidagi agent_name = 'Alex' bo'lgan qatorlarning agent_name, agent_code, phone_no ustunlarini ko'rsatadi. Natija:

AGENT_NAME	AGENT_CODE	PHONE_NO
Alex	A003	075-12458969

SQL kod sxemasi 5.1.2-rasmda.



5.1.2-rasm. So'rov sxemasi

Bir qatorli qism so'rovlarda solishtirish amallarini *WHERE bo'limida* =, <>, <, >, <=, => amallari qo'llash mumkin.

Quyi so'rovni HAVING bo'limida ishlatish: *HAVING* operatori guruhli qatorlarni filtrlash uchun ishlatiladi. Tashqi so'rovning *HAVING*

bo'limida quyi so'rovlardan foydalanish mumkin. Bu quyi so'rov natijalari asosida guruhli qatorlarni filtrlash imkonini beradi.

Quyidagi orders jadvalidan ord_amount, agent_code soni va agent_code ustunlarini ushbu shartlar asosida tanlab olish talab etilsin:

- agent_code qaytalanmasligi kerak;
- agent_code bo'yicha guruhlangan ord_amount ustunining o'rtacha qiymati ord_amount ustunining o'rtacha qiymatiga teng bo'lishi kerak;
- agent_code=A008 bo'lishi kerak;

Masala SQL kodi:

```
SELECT ord_num,ord_amount,ord_date,cust_code, agent_code
FROM orders
WHERE ord_amount>
      (SELECT AVG(ord_amount)
       FROM orders
       WHERE ord_date='20-APR-08');
```

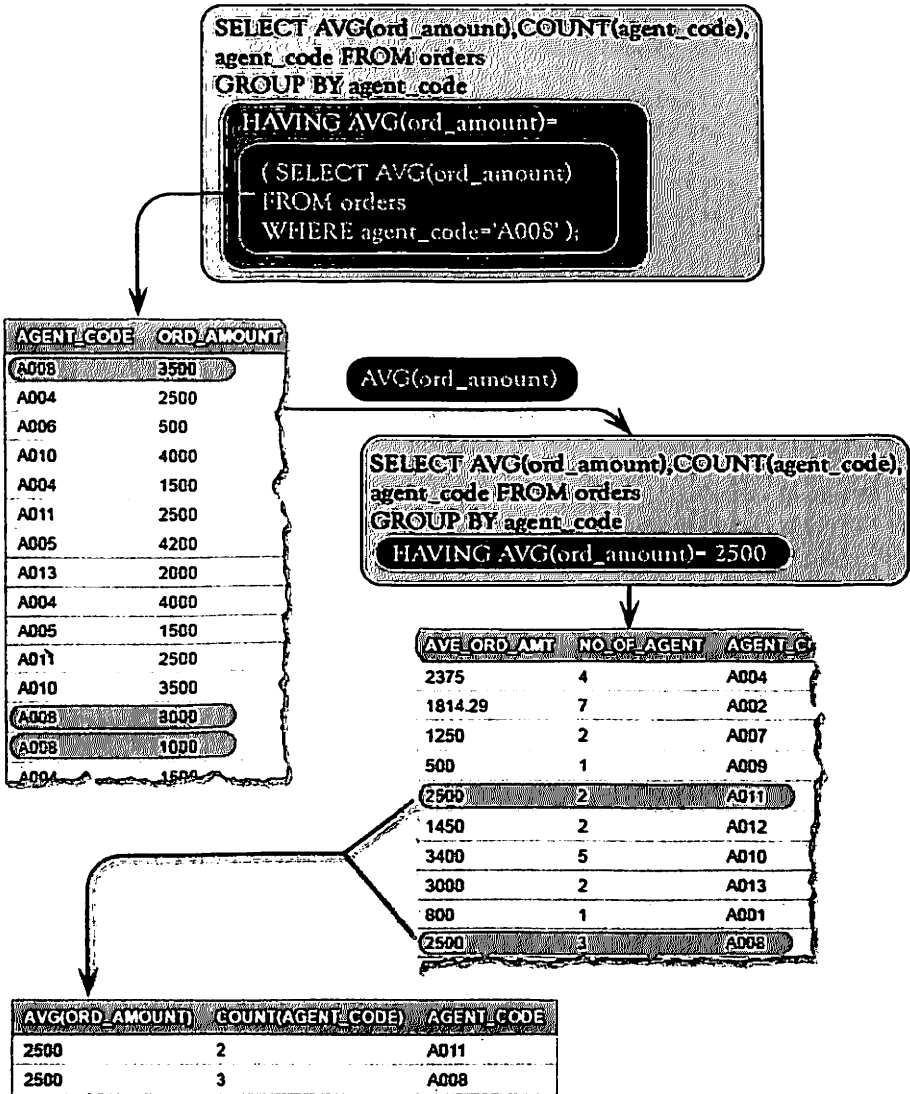
Natija:

AVG(ORD_AMOUNT)	COUNT(AGENT_CODE)	AGENT_CODE
2500	2	A011
2500	3	A008

So'rov bajarilish sxema 5.1.3-rasmda keltirilgan.

Quyi so'rovlarni FROM bo'limida ishlatish: Quyi so'rovni tashqi so'rovning FROM bo'limida ham ishlatish mumkin. Bunday quyi so'rovlar ichki tasavvurlar (VIEW) nomi bilan ham ma'lum. Ma'lumki, SELECT opkratoridagi FROM bo'limida jadval va tasavvurlarning nomlari keltiriladi. Shuningdek, FROM bo'limining davomi sifatida JOIN operatori ham ishlatiladi.

Demak, FROM da qaralayotgan ob'ekt jadval, ya'ni to'plam bo'lganligi sababli ushbu to'plam elementlari tanlovini albatta SELECT operatori orqali chiqarish imkoniyati mavjud.



5.1.3-rasm. So'rov sxemasi

Bu usul murakkab SQL so'rovlarini yaratishda oddiyligidan proffersionallikka qadar maqbullashtirish, ayniqsa JOIN operatorida to'plamlarning o'ng va tomon bog'lanishlarini farqlashda juda avfzal hisoblanadi. Umumiy holda FROM bo'limida quyi so'rovlarni ishlatish sintaksisi quyidagicha:

```

SELECT jadvallardan chiqariluvchi ustunlar
FROM (//qism so'rov
      SELECT qism_so'rovdagi_tanllangan_atributlar
      FROM {jadval_nomi | tasavvur_nomi}
      { jadval_nomi | tasavvur_nomi } ...
      [WHERE qidirish_sharti]
      [GROUP BY guruhlanuvchi_ustun [, guruhlanuvchi_ustun] ...]
      [HAVING qidirish_sharti]
      ) to'planning alias nomi

```

```

.....
FROM [jadval_nomi | tasavvur_nomi, ]
      (//1-qism so'rov
      SELECT qism_so'rovdagi_tanllangan_atributlar
      FROM {jadval_nomi | tasavvur_nomi}
      { jadval_nomi | tasavvur_nomi } ...
      [WHERE qidirish_sharti]
      [GROUP BY guruhlanuvchi_ustun [, guruhlanuvchi_ustun] ...]
      [HAVING qidirish_sharti]
      ) 1-to'planning alias nomi

```

```

[INNER | LEFT] JOIN
      (//1-qism so'rov
      SELECT qism_so'rovdagi_tanllangan_atributlar
      FROM {jadval_nomi | tasavvur_nomi}
      { jadval_nomi | tasavvur_nomi } ...
      [WHERE qidirish_sharti]
      [GROUP BY guruhlanuvchi_ustun [, guruhlanuvchi_ustun] ...]
      [HAVING qidirish_sharti]
      ) 2-to'planning alias nomi
      ON to'plamlar_bog'lanishi

```

FROM bo'limida quyi so'rovlarning turlari unchalik ham rol o'ynamaydi. Asosiysi, kiritilayotgan qism to'plam NULL bo'lmagan natija qaytarishida. Shuni e'tiborga olish kerakki FROM bo'limida bittadan ko'p jadval yoki boshqa qism so'rovlar mavjud bo'lgan xolatda qism so'rovlar qo'llanilsa, qavsdan tashqarida albatta *alias nom* berish shart. Alias nom boshqa bo'limlarda berilmaydi, faqat FROM bo'limida beriladi.

Quyida biz foods jadvalidagi `item_id < 4` bo'lgan `item_id` larni chiqarish so'rovini ko'ramiz:

So'rov:

```

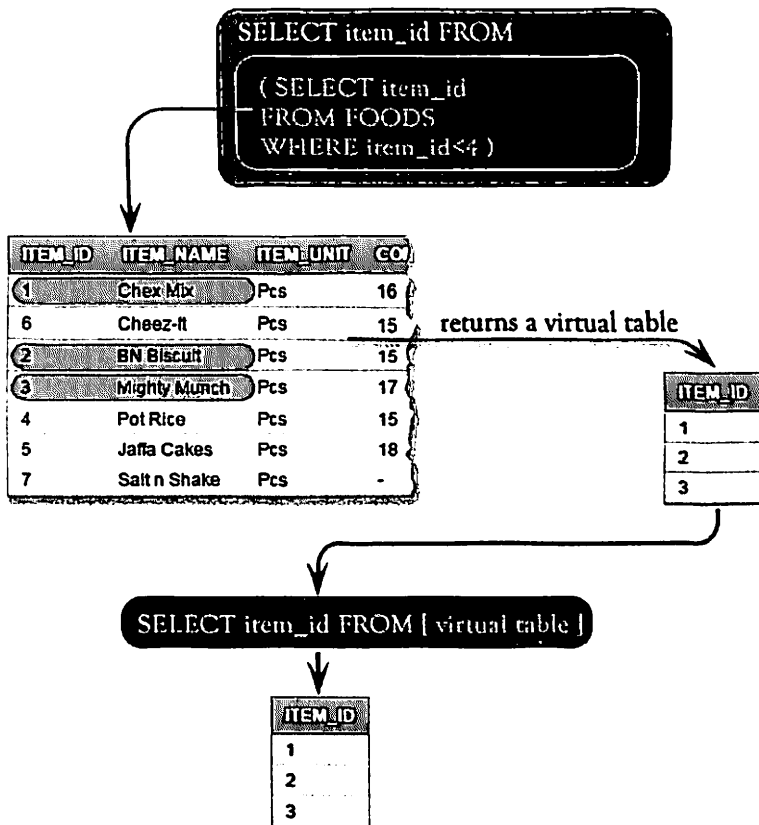
SELECT item_id
FROM
      (SELECT item_id
      FROM FOODS
      WHERE item_id<4)

```

Natija:

ITEM_ID
1
2
3

So'rov sxemasi:



5.1.4-rasm. So'rov sxemasi

Ko'p qatorli quyi so'rovlar

Ko'p qatorli quyi so'rovlar bir yoki undan ko'p qatorlar qiymatini tashqi so'rovga yuboradi. Bir nechta qator qiymatlarini qaytaradigan quyi so'rovlarni bajarishda tashqi so'rovda IN, ANY yoki ALL operatorlaridan foydalanish mumkin.

Quyi so'rovlarda IN operatorini qo'llash

IN operatori qiymatlar to'plami ichidagi qiymatni tekshirish uchun ishlatiladi. Bu qiymatlar to'plami quyi so'rov natijasida shakllantiriladi. Misol tariqasida agents va orders jadvalar berilgan.

Ms 1. orders jadvalidan ord_num, ord_amount, ord_date, cust_code va agent_code ustunlarini ushbu shartlar asosida tanlab olish talab etilsin:

- Tashqi so'rov: orders jadvalidagi agent_code ustuni IN operatori ichidagi quyi so'rov natijasidagi to'plamda bo'lishi kerak
- Quyi so'rov: agents jadvalining working_area = 'Bangalore' bo'lishi kerak.

SQL kod quyidagicha:

```
SELECT ord_num,
       ord_amount,
       ord_date,
       cust_code,
       agent_code
FROM orders
WHERE agent_code IN(
                SELECT agent_code FROM agents
                WHERE working_area='Bangalore'
                );
```

Natija:

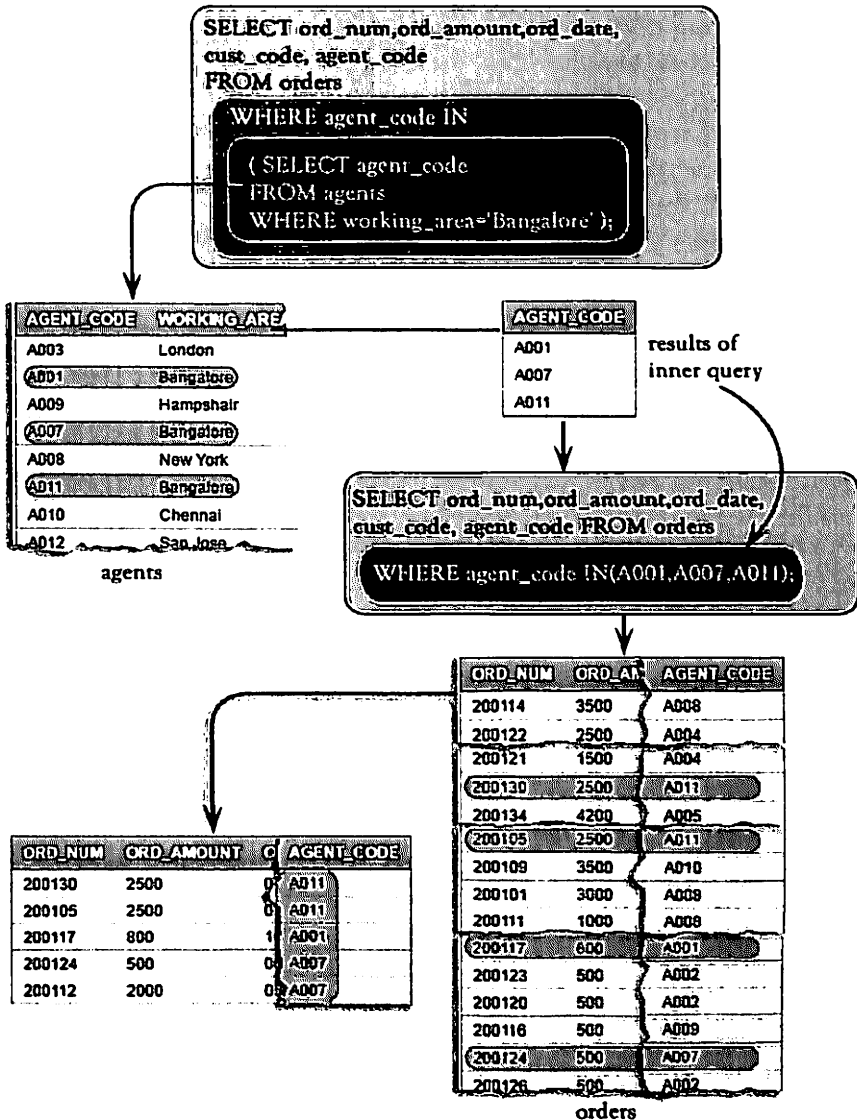
ORD_NUM	ORD_AMOUNT	ORD_DATE	CUST_CODE	AGENT_CODE
200130	2500	30-JUL-08	C00025	A011
200105	2500	18-JUL-08	C00025	A011
200117	800	20-OCT-08	C00014	A001
200124	500	20-JUN-08	C00017	A007
200112	2000	30-MAY-08	C00016	A007

Keling quyi so'rovning o'zi qanday vazifa bajarayotganini va u hosil qilgan to'plam elementlarini ko'ramiz:

```
SELECT agent_code
FROM agents
WHERE working_area='Bangalore';
```

Natija quyidagicha bo'lib, uning SQL kod sxemasi 5.1.5-rasmda keltirilgan

```
AGENT_CODE
-----
A001
A007
A011
```



5.1.5-rasm. So'rov sxemasi

IN operatoridan tashqari NOT IN operatori ham ishlatiladi. Bunda IN dan farqli jihatida NOT IN operatori to'plam elementlaridan boshqa elementlarni olish uchun ishlatiladi.

Ko'p qatorli quyida so'rovlarda ANY va ALL operatorlar.

IN operatoriga o'xshash bo'lgan ANY va ALL operatorlari WHERE yoki HAVING kalit so'zlari bilan ishlatiladi. Bu operatorlar ichki

so'rovlardan mantiqiy qiymat qaytaradi. Agar ichki so'rovdagi qiymatlar to'plamidan birortasi shartga mos kelsa, *ANY* operator TRUE qiymat qaytaradi. Agar ichki so'rovdagi qiymatlar to'plamidan barchasi shartni qanoatlandirsa, *ALL* operator TRUE qiymat qaytaradi. Ya'ni *ANY* va *ALL* operatorlarining farqi ikki so'rov natijalarini *sharni qisman va to'la* qanoatlandirishida bo'lib hisoblanadi.

IN operatorining *ANY* va *ALL* operatorlaridan unda qism so'rov natijalariga ko'rsatilgan elementlarning tegishligida. Ichki so'rov natijalari bilan taqqoslash uchun *ANY* va *ALL* operatorlaridan (=, < >, <, >, <= , >=) oldin amali qo'yiladi.

ANY operatorining sintaksisi:

```
SELECT ustun(lar, )
FROM jadval_nomi(lar bog'lanishlar)
WHERE ustun_nomi taqqoslash_amali ANY
                                (SELECT ustun_nomi
                                FROM jadval_nomi
                                WHERE shart);
```

ALL operatorining sintaksisi:

```
SELECT ustun(lar, )
FROM jadval_nomi(lar bog'lanishlar)
WHERE ustun_nomi taqqoslash_amali ALL
                                (SELECT ustun_nomi
                                FROM jadval_nomi
                                WHERE shart);
```

ALL va *ANY* orasidagi yana bir asosiy farq, ichki so'rov hech qanday natija qaytarmagan xolatda ko'rinadi. Bu holda *ALL* avtomatik TRUE, *ANY* esa avtomatik FALSE ga teng bo'ladi. *ANY* operatorining ekvivalenti sifatida *SOME* operatori ham qo'llaniladi.

Misol. Berilgan customer va agents jadvallarijan *ANY* operatori orqali country = 'UK' bo'lgan istalgan agentlardan birini tekshirish talab etilsin. Bunda agents jadvalidan agent_code, agent_name, working_area, commission ustunlarini ushbu shartlar orqali tanlab olinadi:

- Tashqi so'rov: agents jadvalidagi agent_code ustuni customer jadvalidagi biror (*ANY*) agent_code ga teng bo'lishi kerak.
- Ichki so'rov: customer jadvalidagi cust_country='UK' bo'lishi kerak.

To'liq SQL kod quyidagicha:

```
SELECT agent_code,agent_name,working_area,commission
FROM agents
WHERE agent_code=ANY(
                                SELECT agent_code FROM customer
                                WHERE cust_country='UK');
```

Natija:

AGENT CODE	AGENT NAME	WORKING AREA	COMMISSION
A009	Benjamin	Hampshair	.11
A003	Alex	London	.13
A006	McDen	London	.15

So'rov sxemasi 5.1.6-rasmda berilgan.

Ko'p ustunli quyi so'rovlar.

SQL orqali ko'p ustunlarni natijasi sifatida qaytaruvchi so'zrovlarni ifodalash ham mumkin. Misol uchun orders jadvalidan eng past narxda buyurtma qilgan agentliklarning kodi bo'yicha guhlab chiqaruvchi so'rov sxemasi 5.1.7-rasmda kerilgan.

Korrelyatsion (mutanosib) quyi so'rovlar

SQL da korrelyatsion quyi so'rovlar tashqi so'rovda berilgan jadvaldan ma'lumotlarni tanlash uchun ishlatiladi. Bu turdagi quyi so'rovlarning korrelyatsion deb yuritilishiga sabab quyi so'rov tashqi so'rov jadvaliga murojat qiladi. Yuqorida qarab o'tilgan qism so'rov turlarida faqat ichki so'rov natijalari tashqi so'rov uchun qaytarilgan edi. Ma'lumki, qism so'rovlarda bir nomdagi jadvallardan foydalanish mumkin. Ushbu turdagi so'rovlarda qaysi jadvaldan foydalanilayotganini belgilash uchun ALIASlardan (taxallus) foydalaniladi.

Misol. orders jadvalidan ord_num, ord_amount, cust_code va agent_code ustunlarini quyidagi shart asosida tanlash kerak: orders.agent_code=agents.agent_code va agent_name='Alex' bo'lishi kerak.

SQL kod sxemasi 5.1.8-rasmda berilgan.

Korrelyatsion quyi so'rovlarda EXISTS operatorini qo'llash

EXISTS operatori quyi so'rov natijalari mavjudligini tekshiradi. EXISTS - bu "TRUE" yoki "FALSE" qaytaruvchi operatoridir. Bu shuni bildiradiki, u predikatda avtonom yoki mantiqiy operatorlar AND, OR va NOT yordamida tuzilgan mantiqiy ifodalar bilan kombinatsiya qilingan xolda ishlatilishi mumkin. Quyi so'rov hech bo'lmaganda bitta natija qaytarsa EXISTS operatori TRUE qiymatni qaytaradi aks holda ya'ni quyi so'rov natija qaytarmasa FALSE qiymatni qaytaradi. EXISTS operatori korrelyatsion quyi so'rovlarda ko'p ishlatiladi.

Misol uchun employees jadval berilgan bo'lsin (ma'ruza oxirida).

Quyida so'rovda boshqa xodimlarni boshqaradigan (manager_id) xodimlarning employee_id, manager_id, first_name, last_name ustunlarini chiqaramiz:

```
SELECT employee_id, manager_id, first_name, last_name
FROM employees a
WHERE EXISTS
    (SELECT employee_id
     FROM employees b
     WHERE b.manager_id = a.employee_id)
```

So'rov sxema 5.1.9-rasmda berilgan:

EXISTS operatoridan tashqari NOT EXISTS operatori ham ishlatiladi. Bunda EXISTS dan farqli jihati NOT EXISTS operatori rost bo'lgan to'plam elementlaridan boshqa elementlaridan boshqasini qaytarish uchun ishlatiladi.

Xulosa

Qism so'rovlar bo'yicha quyidagi umumiy xulosalar keltiriladi:

- Qism so'rovlarda ikki qismdan iborat: tashqi so'rov va ichki so'rov.
- Ichki so'rovlar oldin bajarilib, tashqi so'rovga natija qaytaradi.
- Ichki so'rov natija qaytarishi bo'yicha asosan 3 turga ajratiladi.



- Qism so'rovlar FROM, WHERE va HAVING bo'limlarda ishlatiladi;
- Ichki so'rovlar ORDER BY va GROUP BY bo'limlarida qo'llanish mumkin emas;

- Ichki so'rovda ORDER BY kalit so'zi ishlatilmaydi;
- Ichki so'rovlar SELECT, INSERT, UPDATE, DELETE tarkibida joylashadi;
- Quyi so'rovlar SELECT bilan boshlanadi;
- Quyi so'rovlar WHERE va HAVING kalit so'zlardan keyin taqqoslash operatorlaridan so'ng IN, ANY, ALL, EXISTS peredikatlari orqali yoki shunchaki qavs ichida yoziladi;
- IN peredikati oldida taqqoslash operatorlar bo'lmaydi;
- FROM qismida kelgan ichki so'rovga alias nomi beriladi;
- Ichki so'rovlar orqali murakkab so'rovlar yaratiladi;
- Ichki so'rovdan NULL natija qaytsa, tashqi so'rov bajarilmaydi;
- Ichki so'pr tarkibida boshqa ko'plab ichki so'rovlar bo'lishi mumkin.
- Bitta tashqi so'rovda ko'plab ichki so'rovlar ishlatish mumkin.

Nazorat savollari

1. SQLda goddiy va murakkab so'rovlarni farqi nimada?
2. Quyi so'rovlar maxsus qatorli operator turlari qanday?
3. Quyi so'rovlarga yaratishda qanday talablarni bajarish lozim?
4. Quyi so'rovlarning turlari va ularning farqi qanday?
5. Quyi so'rovlar SELECT operatorini qaysi bo'limlarida ishlatiladi?
6. Quyi so'rovlarga alias nom qaerda ishlatiladi vanima uchun kerak??
7. Ko'p qatorli quyi so'rovlarda qo'llaniladigan operatorlar qaysilar?
8. ANY va ALL operatorlarining vazifasi va farqi nimada?
9. Korrelyatsion quyi so'rovlar axamiyati nimada?

Ma'ruzada qo'llaniladigan sxema va jadvallar

```
SELECT agent_code,agent_name,working_area,
commission
FROM agents
WHERE agent_code = ANY
( SELECT agent_code
FROM customer
WHERE cust_country='UK');
```

AGENT_CODE	CUST_COUNTRY
A007	India
A003	UK
A008	USA
A008	USA
A011	India
A006	UK
A003	UK
A008	USA
A005	Australia

customer

AGENT_CODE
A008
A003
A006
A003
A006

results of inner query

```
SELECT agent_code,agent name,working_area,
commission FROM agents
WHERE agent_code=ANY(any row from inner
query);
```

AGENT_CODE	AGENT_NAME	COMMISSION
A003	Alex	.13
A001	Subbarao	.14
A009	Benjamin	.11
A007	Ramasunda	.15
A008	Alford	.12
A011	Ravi Kumar	.15
A010	Santakumar	.14
A012	Lucida	.12

agents

AGENT_CODE	AGENT NAME	COMMISSION
A009	Benjamin	.11
A003	Alex	.13
A006	McDen	.15

5.1.6-rasm. So'rov sxemasi

```
SELECT ord_num, agent_code, ord_date,
ord_amount FROM orders
```

```
WHERE (agent_code, ord_amount) IN
```

```
(SELECT agent_code,
MIN(ord_amount) FROM orders
GROUP BY agent_code);
```

ORD_NUM	ORD_AMOUNT	AGENT_CODE
200114	3500	A008
200122	2500	A004
200118	500	A006
200119	4000	A010
200121	1500	A004
200130	2500	A011
200134	4200	A005
200115	2000	A013
200108	4000	A004
200103	1500	A005
200105	2500	A011
200109	3500	A010

orders

results of inner query

AGENT_CODE	MIN(ORD_AM)
A004	1500
A002	500
A007	500
A009	500
A011	2500
A012	900
A010	2000
A013	2000
A001	800
A008	1000

```
SELECT ord_num, agent_code, ord_date,
ord_amount FROM orders
```

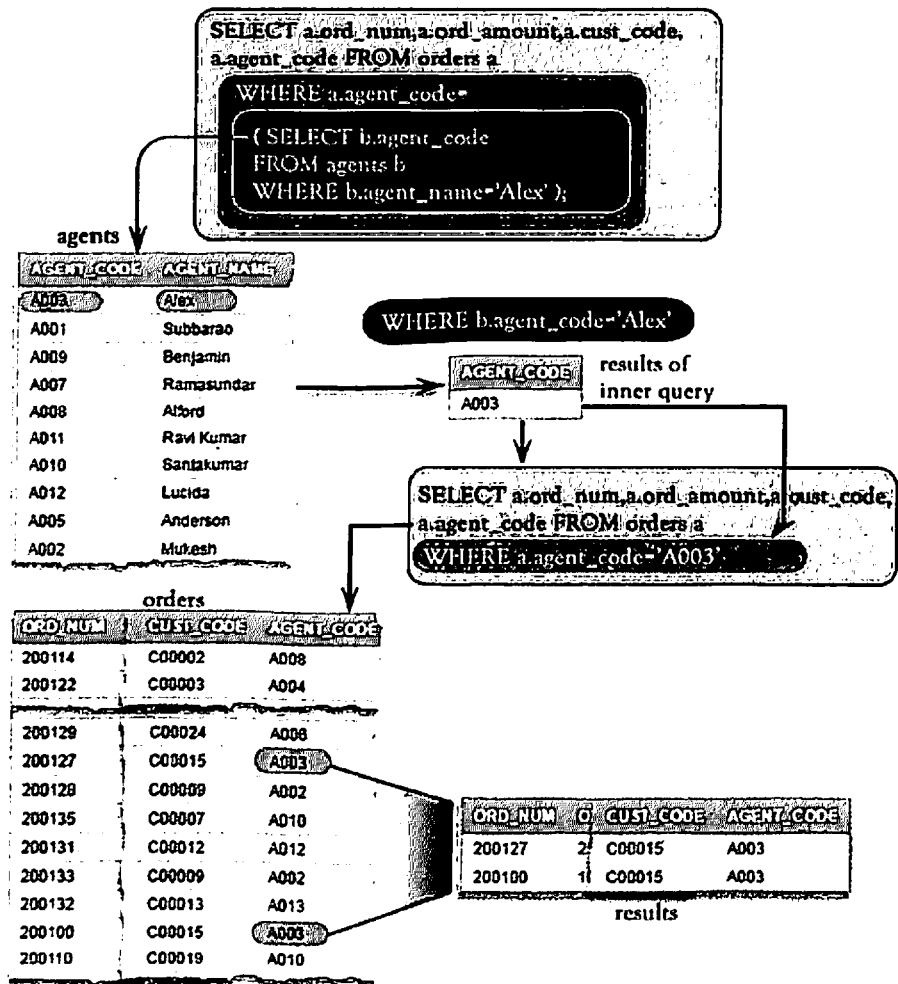
```
WHERE (agent_code, ord_amount) IN ( results
from inner query );
```

ORD_NUM	AGENT_CODE	ORD_AMOUNT
200114	A008	3500
200122	A004	2500
200118	A010	4000
200121	A004	1500
200130	A011	2500
200134	A005	4200
200115	A013	2000
200105	A011	2500
200109	A010	3500
200101	A008	3000
200111	A008	1000
200104	A004	1500

ORD_NUM	AGENT_CODE	ORD_AMOUNT
200104	A004	1500
200121	A004	1500
200126	A002	500
200120	A002	500
200123	A002	500
200124	A007	500
200116	A009	500
200105	A011	2500
200130	A011	2500
200131	A012	900
200135	A010	2000
200115	A013	2000

results

5.1.7-rasm. So'rov sxemasi



5.1.8-rasm. So'rov sxemasi

EXISTS operatorini qo'llashsha misol

```
SELECT employee_id, manager_id, first_name,
last_name FROM employees a
```

WHERE EXISTS

```
(SELECT employee_id
FROM employees b
WHERE b.manager_id = a.employee_id),
```

FIRST_NAME	EMPLOYEE_ID	MANAGER_ID
Steven	100	
Neena	101	100
Alex	102	100
Alexander	103	102
Bruce	104	103
David	105	103
Valli	106	103
Diana	107	103
Nancy	108	101
Daniel	109	108

employees alias b

EMPLOYEE_ID	MANAGER_ID	FIRST_NAME
100		Steve
101	100	Neer
102	100	Lex
103	102	Alexa
104	103	Bruce
105	103	David
106	103	Valli
107	103	Diana
108	101	Nancy
109	108	Daniel

employees alias a

record matches in more than one row

EXISTS [TRUE]

```
SELECT employee_id, manager_id, first_name,
last_name FROM employees a
```

WHERE EXISTS [TRUE]

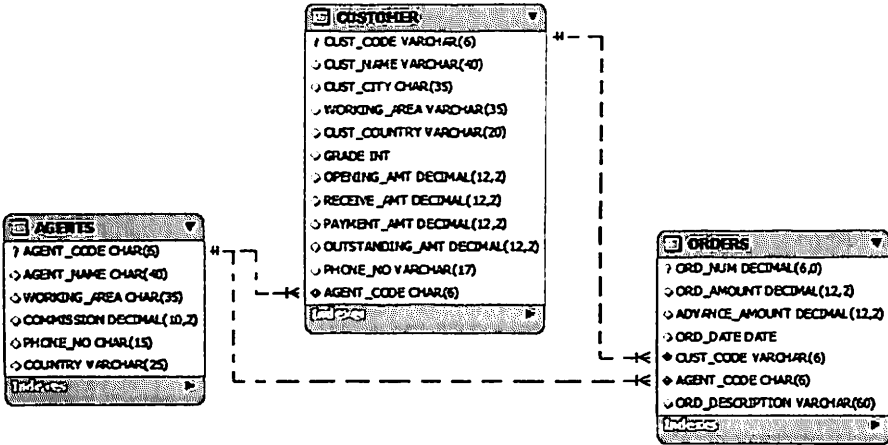
EMPLOYEE_ID	MANAGER_ID	FIRST_NAME
100		Steve
101	100	Neer
102	100	Lex
103	102	Alexa
104	103	Bruce
105	103	David
106	103	Valli
107	103	Diana
108	101	Nancy
109	108	Daniel

employees alias a

EMPLOYEE_ID	MANAGER_ID	FIRST_NAME
100		Steve
101	100	Neer
102	100	Lex
103	102	Alexa
108	101	Nancy
114	100	Dan
120	100	Math
121	100	Adam

results

5.1.9-rasm. So'rov sxemasi



orders

ORD_NUM	ORD_AMOUNT	ADVANCE_AMOUNT	ORD_DATE	CUST_CODE	AGENT_CODE	ORD_DESCRIPTION
200114	3500	2000	15-AUG-08	C00002	A008	
200122	2500	400	16-SEP-08	C00003	A004	
200118	500	100	20-JUL-08	C00023	A006	
200119	4000	700	16-SEP-08	C00007	A010	
200121	1500	600	23-SEP-08	C00008	A004	
200130	2500	400	30-JUL-08	C00025	A011	
200134	4200	1800	25-SEP-08	C00004	A005	
200108	4000	600	15-FEB-08	C00008	A004	
200103	1500	700	15-MAY-08	C00021	A005	
200105	2500	500	18-JUL-08	C00025	A011	
200109	3500	800	30-JUL-08	C00011	A010	
200101	3000	1000	15-JUL-08	C00001	A008	
200111	1000	300	10-JUL-08	C00020	A008	
200104	1500	500	13-MAR-08	C00006	A004	
200106	2500	700	20-APR-08	C00005	A002	
200125	2000	600	10-OCT-08	C00018	A005	
200117	800	200	20-OCT-08	C00014	A001	
200123	500	100	16-SEP-08	C00022	A002	
200120	500	100	20-JUL-08	C00009	A002	
200116	500	100	13-JUL-08	C00010	A009	
200124	500	100	20-JUN-08	C00017	A007	
200126	500	100	24-JUN-08	C00022	A002	
200129	2500	500	20-JUL-08	C00024	A006	
200127	2500	400	20-JUL-08	C00015	A003	
200128	3500	1500	20-JUL-08	C00009	A002	
200135	2000	800	16-SEP-08	C00007	A010	
200131	900	150	26-AUG-08	C00012	A012	
200133	1200	400	29-JUN-08	C00009	A002	
200100	1000	600	08-JAN-08	C00015	A003	
200110	3000	500	15-APR-08	C00019	A010	
200107	4500	900	30-AUG-08	C00007	A010	
200112	2000	400	30-MAY-08	C00016	A007	
200113	4000	600	10-JUN-08	C00022	A002	
200102	2000	300	25-MAY-08	C00012	A012	

agents

AGENT_CODE	AGENT_NAME	WORKING_AREA	COMMISSION	PHONE_NO	COUNTRY
A007	Ramasundar	Bangalore	0.15	077-25814763	
A003	Alex	London	0.13	075-12458969	
A008	Alford	New York	0.12	044-25874365	
A011	Ravi Kumar	Bangalore	0.15	077-45625874	
A010	Santakumar	Chennai	0.14	007-22388644	
A012	Lucida	San Jose	0.12	044-52981425	
A005	Anderson	Brisban	0.13	045-21447739	
A001	Subbarao	Bangalore	0.14	077-12346674	
A002	Mukesh	Mumbai	0.11	029-12358964	
A006	McDen	London	0.15	078-22255588	
A004	Ivan	Torento	0.15	008-22544166	
A009	Benjamin	Hampshair	0.11	008-22536178	

foods

ITEM ID	ITEM NAME	ITEM UNIT	COMPANY ID
1	Chex Mix	Pcs	16
6	Cheez-It	Pcs	15
2	BN Biscuit	Pcs	15
3	Mighty Munch	Pcs	17
4	Pot Rice	Pcs	15
5	Jaffa Cakes	Pcs	18
7	Salt n Shake	Pcs	

customer

CUST_CODE	CUST_NAME	CUST_CITY	WORKING_AREA	CUST_COUNTRY	GRADE	OPENING_AMT	PHONE_NO	AGENT_CODE
C00013	Holmes	London	London	UK	2	6000	BBBBBBB	A003
C00001	Micheal	New York	New York	USA	2	3000	CCCCCCC	A008
C00020	Albert	New York	New York	USA	3	5000	BBBBBBS	A008
C00025	Ravindran	Bangalore	Bangalore	India	2	5000	AVAVAVA	A011
C00024	Cook	London	London	UK	2	4000	FSDSDSF	A006
C00015	Stuart	London	London	UK	1	6000	GFSGERS	A003
C00002	Bolt	New York	New York	USA	3	5000	DDNRDRH	A008
C00018	Fleming	Brisban	Brisban	Aust.	2	7000	NHBGVFC	A005
C00021	Jacks	Brisban	Brisban	Aust.	1	7000	WERTGDF	A005
C00019	Yearannaidu	Chennai	Chennai	India	1	8000	ZZZBVFV	A010
C00005	Sasikant	Mumbai	Mumbai	India	1	7000	147-25896312	A002
C00007	Ramanathan	Chennai	Chennai	India	1	7000	GHRDWS	A010
C00022	Avinash	Mumbai	Mumbai	India	2	7000	113-12345678	A002
C00004	Winston	Brisban	Brisban	Aust.	1	5000	AAAAAAA	A005
C00023	Karl	London	London	UK	0	4000	AAAABAA	A006

C00006	Shilton	Torento	Torento	Canada	1	10000	DDDDDDD	A004
C00010	Charles	Hampshair	Hampshair	UK	3	6000	MMMMMMM	A009
C00017	Srinivas	Bangalore	Bangalore	India	2	8000	AAAAAAB	A007
C00012	Steven	San Jose	San Jose	USA	1	5000	KRFYGJK	A012
C00008	Karolina	Torento	Torento	Canada	1	7000	HJKORED	A004
C00003	Martin	Torento	Torento	Canada	2	8000	MJYURFD	A004
C00009	Ramesh	Mumbai	Mumbai	India	3	8000	Phone No	A002
C00014	Rangarappa	Bangalore	Bangalore	India	2	8000	AAAATGF	A001
C00016	Venkatpati	Bangalore	Bangalore	India	2	8000	JRTVFDD	A007
C00011	Sundariya	Chennai	Chennai	India	3	7000	PPHGRTS	A010

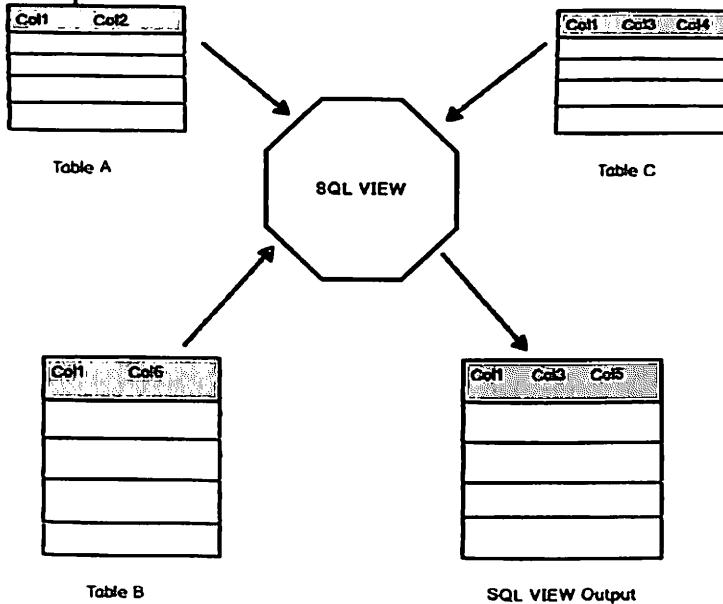
Employees

employee_id	first_name	last_name	phone_number	salary	manager_id	department_id
100	Steven	King	515.123.4567	24000		90
101	Neena	Kochhar	515.123.4568	17000	100	90
102	Lex	De Haan	515.123.4569	17000	100	90
103	Alexander	Hunold	590.423.4567	9000	102	60
107	Diana	Lorentz	590.423.5567	4200	103	60
108	Nancy	Greenberg	515.124.4569	12000	101	100
113	Luis	Popp	515.124.4567	6900	108	100
114	Den	Raphaely	515.127.4561	11800	100	30
118	Guy	Himuro	515.127.4565	2600	114	30
119	Karen	Colmenares	515.127.4566	2500	114	30
120	Matthew	Weiss	650.123.1234	8000	100	50
121	Adam	Fripp	650.123.2234	8200	100	50
122	Payam	Kautling	650.123.3234	7900	100	50
123	Shanta	Vollman	650.123.4234	6500	100	50
124	Kevin	Mourgos	650.123.5234	5800	100	50
128	Steven	Markle	650.124.1434	2200	120	50
132	TJ	Olson	650.124.8234	2100	121	50
136	Hazel	Philtanker	650.127.1634	2200	122	50
139	John	Seo	650.121.2019	2700	123	50
144	Peter	Vargas	650.121.2004	2500	124	50
145	John	Russell	011.44.1344.	14000	100	80
146	Karen	Partners	011.44.1344.	13500	100	80
147	Alberto	Erazuriz	011.44.1344.	12000	100	80
148	Gerald	Cambault	011.44.1344.	11000	100	80
149	Fleni	Lotkey	011.44.1344.	10500	100	80
155	Oliver	Tuvault	011.44.1344.	7000	145	80
167	Amit	Banda	011.44.1346.	6200	147	80
173	Sundita	Kumar	011.44.1343.	6100	148	80
179	Charles	Johnson	011.44.1644.	6200	149	80
201	Michael	Hartstein	515.123.5555	13000	100	20
202	Pat	Fay	603.123.6666	6000	201	20
205	Shelley	Higgins	515.123.8080	12000	101	110
206	William	Gietz	515.123.8181	8300	205	110

5.2. Tasavvurlar

Tasavvurlar haqida tushuncha va afzalliklari

Jadval MB tuzilmasi va undagi ma'lumotlarning tasnifini belgilaydi. Shuningdek, ma'lumotlarni xar tomonlama taqdim etish yoki tasvirlash SQL yordamida amalga oshiriladi. MBda ko'rinish, tasvirlash, taqdim etish atamalari ingliz tilidagi "VIEW" (rus tilida "predstavlenie") so'zi bilan bir xil ma'noda ishlatilib, ko'pincha o'zbek tilida "TASAVVUR" atamasi bilan qo'llaniladi.



5.2.1-rasm. Tasavurning namunaviy ko'rinishi.

Tasavvur bu MBda o'z nomiga ega va doimiy saqlanadi ma'lumotlarni tanlashdagi SQL so'rovidir. Qisqacha tasavvur MBda saqlanuvchi SQL so'rov deb aytiladi. Tasavvur foydalanuvchiga saqlanuvchi so'rov natijalarini taqdim etadi. Bunda SQL xuddi MBdagi real xolatda jadvallarga kirishni ta'minlaydi. Ya'ni tasavvurlar MBdagi "virtual jadvallar" bo'lib, bu virtuala jadvallarga xuddi real jadvaldagi kabi SQL so'rovlari bajariladi. Ayrim MBBTda (ms: NoSQL) MBga murojaatlar faqat tasavvutrlar orqali amalga oshiriladi. Tasavvur xuddi MBBTning eng soddasi hisoblangan MS ACCESS paketida "Zapros" ob'ekti bilan bir xil.

Tasavvurlar MBning murakkabligini yashirib uni fotsdalanuvchilar uchun soddalashtiradi va ma'lumotlar modelini abstraklashtirish, MB

xavfsizligini ta'minlash va normallashtirish uchun qulay. Tasavvurlar asosan biznet logikalarni amalga oshirishda ishlatiladi. Bazaviy jadvallarga yaratilgan interaktiv SQL so'rovi natija chiqarilgandan keyin xotirada saqlanmaydi. Tasavurning esa foydali tomoni ko'p ishlatiladigan SQL so'rov bir marta yaratilib, undan ko'p marta foydalaniladi.

Tasavvurlar quyidagi sabablarga ko'ra amaliyotda keng qo'llaniladi:

- Ma'lumotlarni markazlashtirish yoki birlashtirishdagi amallarning, jumladan JOIN orqali bog'lanishlari kabi takrorlanishlarning oldini oladi;
- turli foydalanuvchilarga MBni turlicha ifodalashga (ko'rinish) imkon beradi (ms: jadval parametrlarini o'zgargan va cheklangan holi);
- foydalanuvchiga jadval(lar)ning ustun va qatorlarini to'liq ko'rishga ruxsatlarni cheklaydi;
- jadvallarga to'g'ridan-to'g'ri murojaatlarni oldini oluvchi mavxum qatlamni ta'minlaydi;
- MBga ruxsatlarni soddalashtirish, ya'ni ma'lumotlar tuzilmalaridan har bir foydalanuvchiga aynan mos keladigan shakllarda ruxsat etish. Mazkur mavzuda tasavvurlarni yaratish, MB bilan ishlashni soddalashtirish va xavfsizlikni oshirish masalalari qarab o'tiladi.

Tasavvurdagi virtual jadvalning tarkibi SQL so'rov orqali aniqlanadi.

Bizga ma'lumki, SQL so'rovlari maxsus shartlar asosida bir yoki bir nechta ulangan va/yoki birlashtirilgan jadvallardan ma'lumotlarni tanlab olish yoki qisqacha MB jadvallari ga MB foydalanuvchisi uchun tasavvur satr va ustunlardan iborat haqiqiy jadvalga o'xshaydi. Ammo, tasavurning jadvaldan farqli tomoni shundaki, u MBda qiymatlar to'plami sifatida mavjud emasligida. MBda saqlanishi va nomlanishiga ko'ra tasavvur ikkilamchi qatlam bo'lib, undan foydalanuvchining SQL so'rovi orqali oladigan ustun va qatorli natijalari ostida birlamchi SQL so'rov turadi.

Tasavvurlarning afzalliklari

Har xil xarakterdagi MBda tasavvurlardan foydalanish turli vaziyatlarga ko'ra samarali bo'lishi mumkin. Misol uchun tasavvurlar shaxsiy kompyuterdagi MBda qulaylik yaratish va so'rovlarni soddalashtirishda, ishlab chiqarish kabi katta MBlarida esa xar bir foydalanuvchining shaxsiy MB tarkibini yaratish va xavfsizligini ta'minlashda qo'llaniladi. Tasavvurlarning asosiy afzalliklari quyida keltirilgan.

Xavfsizlik. Bunda har bir foydalanuvchiga faqat o'ziga taulliqi bo'lgan ma'lumotlarni qamrab olgan kam sonli tasavvurlarga kirish huquqi

beriladi. Shunday bo'lgan xolatda saqlanayotgan ma'lumotlarga foydalanuvchining foydalanish ruxsatlarini cheklash mumkin.

So'rov soddaligi. Tasavvurni ishlatish orqali bir nechta jadvallardagi ma'lumotlarni tanlab bitta jadval sifatida taqdim etish mumkin. Shunda ko'p jadvallar uchun yozilgan so'rovlarning ko'rinishni bitta jadvali so'rovga aylantiriladi. Shuningdek, foydalanuvchi jadvallarning tegishli qismini ko'ra oladi.

Tarkibiy soddalik. Tasavvurni qo'llash orqali har bir foydalanuvchi o'zining MB tuzilmasini yaratishi va uni foydalanuvchilarning mavjud virtual jadvallari to'plami sifatida belgilash mumkin.

O'zgarishlardan himoyalash. Tasavvur MB tuzilishining izchil va o'zgarmas tasvirini qaytarishi mumkin (xatto manbadagi jadvallar bo'linsa, qayta tuzilmalangan yoki o'zgartirilgan bo'lsa ham). Shunday bo'lganda jadval yoki ustunlarining nomlari o'zgartirilganda tasavvurni ham yangilash lozim.

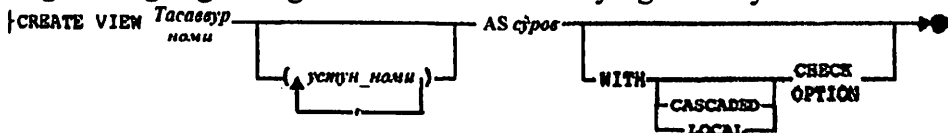
Ma'lumotlar yaxlitligi. Tasavvur orqali MBga kirish yoki ma'lumotlar chiqarishdagi yaxlitlik shartlarining bajarilishi tekshiriladi.

Kamchiliklariga to'xtalib o'tiladigan bo'lsa, tasavvurlar tarkibidagi SQL so'rov murakkabligiga qarab qiyinchiliklar va vaqt talab qiladi.

Tasavvurlarni yaratishda oldin undagi SQL so'rovining to'g'riligi, ortiqcha ustun va qatorlardan tozalanganligi, so'rovning samaradorligi va optimalligini tekshirish shart hisoblanadi. Shuningdek, tasavvur natijasini o'zgartirishda albatta uning manbasi bo'layotgan bazaviy jadvallar va/yoki boshqa tasavvurlarga qilingan SQL so'rovini qayta ko'rib chiqish talab etiladi.

CREATE VIEW – tasavvur yaratish va ularning turlari

View - SQL tilining DDL operatorlar guruhiga mansub bo'lib, ular bilan ishlash amallari Table ob'ekti kabi. Ya'ni, Create View, Alter View, Drop View amallar orqali tasavvurlarni yaratish, o'zgartirish va o'chirish amalga oshiriladi. Tasavvurlarni yaratishda quyidagi sintaksis diagrammaga ega bo'lgan CREATE VIEW buyrug'idan foydalaniladi.



Sintaksisini asosini yaratilayotgan "tasavvur nomi" va uning manbasi bo'lgan "so'rov" tashkil qiladi. Tasavvur muvaffaqiyatli yaratish uchun undagi so'rovda ishlatilayotgan barcha jadvallarga oldindan ruxsat bo'lishi

kerak. Bundan tashqari ayrim MBBTlarida (ms: Oracle) tasavvurlarni yaratish uchun ham ruxsat berilgan lozim. Ma'lumki, Select operatori natijasidagi jadvalli ma'lumotlar ustun nomlari orqali chiqariladi. Bundan agar so'rov natijasida olingan ustunlarning nomlarini o'gartirmasdan tasavvurga berish uchun "tasavvur nomi"dan keyin maydonlar ko'rsatilmaydi. Aks holda, ustunlarni boshqa nom bilan tasavvurda saqlash sintaxisidagi "ustun_nomi" orqali ustunlarning tartibi bo'yicha qayta nomlanadi hamda ularning soni mos bo'lishi shart. Agar so'rovda bitta nomli ikkita ustun mavjud bo'lsa va/yoki qandaydir funktsiyalar (ms: sum) orqali hisoblashlar yuritilgan bo'lsa, albatta ularga tasavvurda nom berish, ya'ni barcha ustunlarning ro'yxati ko'rsatilishi shart. Bu yerda e'tibor berish lozimki, tasavvurda virtual jadvalning faqat ustun nomi ko'rsatilib, uning ma'lumot toifasi, uzunligi keltirilmaydi. Chunki, bu tasniflar avtomatik tasavvurga berilayotgan manbadan aniqlanadi.

Amaliyotda qo'llanilish maqsadi va qoidalariga ko'ra tasavvurlar bir nechta turlarga ajratiladi, lekin ularning barchasi bir xil xarakterda bo'ladi. Tasavvurlarning turlariga quyidagilar kiradi: gorizontal, vertikal, aralash, guruhli, ulangan tasavvurlar.

Gorizontal tasavvurlar. Odatda bunday turdagi tasavvurlarda SQL so'rovi uchun relyatsion algebraning faqat *tanlash* (selektsiya) amali bajariladi va natija sifatida Select operatori orqali barcha ustunlar chiqariladi. Ya'ni undagi so'rovda * belgisi ishlatiladi. Bu tasavurning sintaksis berilishi quyidagicha bo'ladi.

```
CREATE VIEW <tasavvur nomi> AS
SELECT *
FROM Table(s) /Jadval(lar va bog'lanishlari)
WHERE row(s) /Ustun(lar) qo'yilgan shartlar
```

Ms 1: Books jadvaldan 2018 yildan keyin chiqqan kitob yozuvlarini chiqaruvchi tasavvurni yaratish

```
CREATE VIEW Books2018 AS
SELECT * FROM Books WHERE yil>=2018
```

Tasavvurni bajarish yoki unga murojaat SQL so'rov kabi amalga oshiriladi.

```
SELECT * FROM Books2018
```

Natija:

id\book	name	id\cat	page	id\avtor	yil
7	Shaytanat	2	1250	1	2018
8	Dexqon va ayiq	5	100	5	2019
9	Eng katta sexr	5	230	5	2020
19	Shaytanat	2	960	1	2020

Vertikal tasavvurlar. Bu turdagi tasavvurlarda SQL so'rovi uchun relyatsion algebraning faqat *proeksiya* amali bajariladi. natijada Select operatori orqali kerakli ustunlar ro'yxati ko'rsatiladi. Ya'ni so'rovda * belgisi ishlatilmaydi. Vertikal tasavvurning sintaksisi quyidagicha:

```
CREATE VIEW <tasavvur nomi> AS
SELECT ustun1, ustun2, ...
FROM Table(s) /Jadval(lar va bog'lanishlari)
WHERE row(s) /Ustun(lar) qo'yilgan shartlar
```

Ms 2: Books jadvaldan kitoblarning eng birinchi chiqarilgan yili bo'yicha nomi, saxifasi va ma'lumotlarini chiqaruvchi tasavvurni yaratish

```
CREATE VIEW kitob_yili AS
SELECT name, page, min(yil) AS ilk_yil
FROM Books
```

Tasavvurni bajarish yoki unga murojaat SQL so'rov kabi amalga oshiriladi.

```
SELECT * FROM kitob_yili
```

Natija:

name	page	yil
Odamiylik Mulki	980	2016
Ikki eshik orasi	450	1996
Sariq devni minib	220	1990
Shaytanat	1250	2018
Dexqon va ayiq	100	2019
...

Ms 3: Oldingi misol bo'yicha tasavvurlarda ustunlar ro'yxatini chiqarish

E'tibor beriladigan bo'lsa 2 misol so'rovida agregat funktsiya ishlatildi va budagi hisoblash natijasi ilk_yili nomli aliasga qo'yildi. Agarda, unga alias qo'yilmaganda, yaratilayotgan tasavvurda xatolik ro'y beradi. Buni kengroq tushunish uchun ushbu so'rovni yana bir marta faqat o'zini bajarib ko'raylik.

```
SELECT name, page, min(yil)
FROM Books
```

Natija:

name	page	min(yil)
Odamiylik Mulki	980	2016
...

Tasavvurga berishda kelib chiqadigan xatolik shundan iboratki, so'rovda ishlatilgan agregat funktsiyaga o'xshash barcha turdagi hisoblash natijalariga alias berilmaganida. To'g'ri alias berilmasa ham SQL so'rov natija beradi, ammo, tasavvurda emas. Buday bo'lgan xolatlarda natijali to'g'ri bo'lgan SQL so'rovlar qatnashgan tasavvurning o'zida ustunlar

ro'yxatini ketma-ketlik bo'yicha keltirish shart. Yuqoridagi so'rov uchun tasavvur quyidagicha bo'ladi.

```
CREATE VIEW kitob_yili (kitob_nomi, saxifalar, ilk_bosma_yili) AS
SELECT name, page, min(yil)
FROM Books
```

Tasavvurni bajarish.

```
SELECT * FROM kitob_yili
```

Natija:

kitob_nomi	saxifalar	ilk_bosma_yili
Odamiylik Mulki	980	2016
...

Aralash tasavvurlar. Bunday tasavvurlardagi SQL so'rovda relyatsion algebraning ham *proektsiya*, ham *tanlash* amallari birgalikda bajariladi. natijada Select operatorida zarur ustunlar ro'yxati ko'rsatilishi bilan birga ustunlarga shartlar qo'yiladi. Aralash tasavvurning sintaksisi quyidagicha:

```
CREATE VIEW (col1, col2, ...) AS
SELECT ustun1, ustun2, ...
FROM Table(s) /Jadval(lar va bog'lanishlari)
WHERE row(s) /Ustun(lar) qo'yilgan shartlar
```

Ms 4: Books jadvaldan 2019 yildan keyin chiqqan kitoblarning nomi, saxifasi va yili ma'lumotlarini chiqaruvchi tasavvurni yaratish

```
CREATE VIEW kitob_2019 (kitob_nomi, saxifalar, bosma_yili) AS
SELECT name, page, yil
FROM Books
WHERE yil >= 2019
```

Tasavvurni bajarish.

```
SELECT * FROM kitob_2019
```

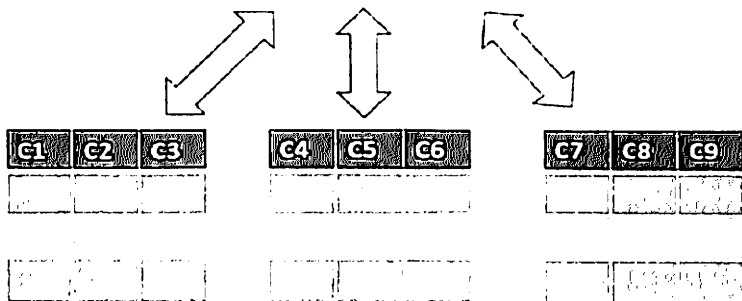
Natija:

kitob_nomi	saxifalar	bosma_yili
Dexqon va ayiq	100	2019
Eng katta sexr	230	2020
Shaytanat	960	2020

Guruhli va ulangan tasavvurlar. Amaliyotda ko'pincha tasavvurlar orqali bir nechta jadvallar bog'lanib, ulardan zarur ustunlarga natija sifati olinadi. Shuningdek, MBdan qarorlar qabul qilishda tayyorlanadigan hisobotlarda guruhli amallar keng ishlatilishini oldingi mavzuda tanishib o'tilgan edi. Demak guruhli amallar (Group By, Having) bajarilgan SQL so'rovlari orqali tuzilgan tasavvurlar guruhli tasavvurlar hisoblanadi.



```
CREATE VIEW V AS
SELECT C2, C3, C5, C7
FROM T1
INNER JOIN T2...
INNER JOIN T3...
```



5.2.2-rasm. Tasavvurlarning tuzilmasiga namuna

Ms 5: MBdan 2005 yildan oldin va bosma yillaridan eng oxirgi marta chiqqan kitoblar nomi bo'yicha tartiblangan holda kitob nomi, muallifi, kategoriyasi, saxifasi va yili ma'lumotlarini bazaviy jadvallardagi nomiga o'xshagan nomlar bilan chiqaruvchi tasavvurni yaratish

Buning uchun oldin Books, Avtor va Categoriya jadvallari sxema bo'yicha Inner Join orqali bog'lanadi va Books jadvalidagi yil ustuniga shart beriladi. Bu tasavvurda ozroq murakkablik jixati bo'lganligi sababli, dastlab berilgan shartlarni qanoatlandiruvchi SQL so'rov tuzib olinadi.

```
SELECT DISTINCT C.name_cat, A.fio,
B.id_book, B.name, B.page,
MAX(B.yil) AS yil
FROM Books B
INNER JOIN Avtor A ON B.id_avtor = A.id_avtor
INNER JOIN Categoria C ON B.id_cat = C.id_cat
WHERE B.yil <= 2005 GROUP BY B.name
```

So'rov natijasi

name_cat	fio	id_book	name	page	yil
Tarixiy	Ibraym Yusupov	15	Dala armonlari	450	1967
Tarbiyaviy	Utkir Hoshimov	18	Dunyoning ishlari"	280	2005
Tarixiy	Utkir Hoshimov	2	Ikki eshik orasi	450	2000
Tarbiyaviy	Xudayberdi To'xtaboyev	12	Jannati odamlar	210	1996
Tarbiyaviy	Ibraym Yusupov	16	Mangu buloq	360	1970
Tarixiy	Ibraym Yusupov	14	Qora tol	350	1988
Ertak	Xudayberdi To'xtaboyev	5	Sariq devni minib	220	2005
Fantastik	Xudayberdi To'xtaboyev	17	Shrli qalpoqcha	250	1964
Ertak	Xudayberdi To'xtaboyev	13	Shirin qovunlar mamlakati	180	1986

Natijaga erishilgan SQL so'rovdan tasavvur yaratish mumkin. Shartga ko'ra, bu yaratiladigan tasaffurda ustunlar ro'yxati o'zgartirilishi lozim.

```
CREATE VIEW Kitoblar (toifa, muallif, id_asar, asar, bet, nashr_yili) AS
SELECT DISTINCT C.name_cat, A.fio,
      B.id_book, B.name, B.page, MAX(B.yil) AS yil
FROM Books B
INNER JOIN Avtor A ON B.id_avtor = A.id_avtor
INNER JOIN Categoria C ON B.id_cat = C.id_cat
WHERE B.yil <= 2005
GROUP BY B.name
```

Yuqoridagi kabi yaratilgan tasavvurlar faqatgina kerakli natijani olish bilan birga albatta MBda xavfsizlikni ta'minlashga xizmat qiladi. Chunki, foydalanuvchi endi faqat tasavvurdagi berilgan va cheklangan ustunlarga murojaat qiladi. Ya'ni u MBdagi asosiy manba bo'lgan jadvallarning nomi, ustunlarning nomlarini hamda sharni qanoatlandirmaydigan boshqa yozuvlarni ko'rishi taqiqlangan. Nima uchun deganda keltirilgan tasavvur so'rovida Books jadvalidan name, page va MAX(yil) maydonlari, Avtor jadvalidan fio va Categoriya jadvalidan name_cat maydonlari olindi va ular mos ravishda asar, bet, nashr_yili, muallif, muallif va toifa nomlarga o'zgartirildi. Shuningdek, so'rovda 2005 yildan oldin chiqqan kitoblar olindi. Demak foydalanuvchi endi yashiringan maydonlar va 2005 yildan keyingi kitoblarni ko'ra olmaydi.

Tasavvurlarni bajarishda, odatdagi kabi ularga murojaat qiluvchi SQL so'rovlarida proektsiya, tanlash, ulash va bo'lish kabi relyatsion amallarning barchasini bajarish, ya'ni tasavvurdan olinadigan natijalarni o'zgartirish yoki murakkablik darajasini oshirish mumkin.

Ms 6: Kitoblar nomli tasavvurdan 2000 yildan keyin chiqqan kitoblarning nomi va muallifini chiqarish

```
SELECT muallif, asar FROM Kitoblar WHERE nashr_yili >= 2000
```

Natija

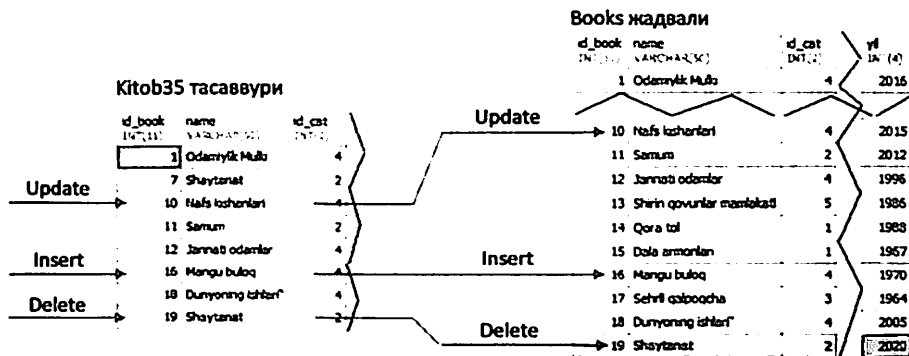
muallif	asar
Utkir Hoshimov	Dunyoning ishlari"
Utkir Hoshimov	Ikki eshik orasi
Xudayberdi To'xtaboyev	Sariq devni minib

Shuni ham unitmaslik kerakki, tasavvurlarga murojaat qiluvchi SQL so'rovlarning o'zida ham guruhli amallarni bajarish mumkin.

Tasavvurlarni yangilash yoki tasvurlarda DML amallar

MBdagi jadval ma'lumotlarini yangilashda, ya'ni DML sinfiga tegishli bo'lgan operatorlar orqali jadvalga yangi yozuv kiritish (Insert),

shartlar asosida yangilash (Update) va o'chirish (Delete) amallari bajariladi.



5.2.3-rasm. Tasavvurlarni yangilash namunasiga

Endi ushbu amallar tasavvurlarda bajarilishini aniq misollar orqali o'rganamiz. Bizga 1 (ertak) va 2 (fantastika) kategoriyalariga tegishli kitoblarni chiqaruvchi gorizontol tasavvur berilgan bo'lsin.

```
CREATE VIEW Kitob35 AS
SELECT * FROM Books WHERE id_cat IN (2, 4)
```

Tasavvurlarni yangilash namunasiga ko'ra turganidek, Kitob35 tasavvuriga mantiqiy ravishda yangi yozuv kiritish (Insert), yangilash (Update) va o'chirish (Delete) deganda aslida Books jadvaliga yangi yozuv kiritish, yangilash va o'chirish bo'lib hisoblanadi.

Manbasi bitta jadvaldan iborat bo'lgan gorizontol tasavvurlarda ma'lumotlarni yangilashda muammolar yo'q. Ammo vertikal yoki aralash tasavvurlarda muammolar payd bo'lishi mumkin. Nima uchun deganda bu turdagi tasavvurlarda manbasha murojaat qiluvchi so'rovida ustunlar ro'yxati cheklangan (to'liq emas) bo'lib, aynan ushbu ro'yxatda mavjud bo'lmagan jadval ustunlariga NULL ma'lumot qabul qilmaslik shartlari o'rnatilgan bo'lishi mumkin. Shuningdek, ulangan va guruhli tasavvurlarda yangilash amallarini bajarish mumkin emas. Chunki, bu turdagi tasavvurlarning SQL so'rovida bir nechta jadvallar va hisoblashlar ishtirok qiladi. Xulosa shundan iborat bo'ladiki, tasavvurlar ustida xatosiz yangilash amallarini bajarish uchun ANSI/ISO ning SQL1 standart shartlar bajarilishi talab etiladi. Tasavvur SQL so'rovida (manbasi) SQL1 standart shartlari quyidagilardan iborat:

- DISTINCT peredikati ishlatilmagan bo'lishi;
- FROM da faqat bitta jadval ishtirok etishi;
- manbada mavjud, lekin tasavvurda mavjud bo'lmagan usutunlar ro'yxatida NULL ma'lumot qabul qilmaslik shartlari quyilmasligi;

- ustunlarda agregat funktsiyalar, hisoblashlar bo'lmisligi;
- WHERE da oddiy shartlar ishtirok etishi;
- GROUP VU va NAVING mavjud bo'lmisligi lozim.

Ms 7: Kitob35 nomli tasavvurga yangi ma'lumot kiritish

INSERT INTO Kitob35(name, id_cat, page, id_avtor, yil) VALUES

('Sherlar guldastasi', 4, 250, 4, ''),

('Gulliverni sarguzashtlari', 5, 180, 3, 2020)

Kitob35 tasavvurga ikkita yozuvni kiritish xatoliksiz bajarildi. Tekshirish uchun yangilangan Kitob35 tasavvuri va Kitob jadvalini chaqiriladi:

SELECT * FROM Kitob35

id_book	name	id_cat	page	id_avtor	yil
1	Odamiylik Mulki	4	980	1	2016
...
19	Shaytanat	2	960	1	2020
20	Sherlar guldastasi	4	250	4	0

SELECT * FROM Kitob

id_book	name	id_cat	page	id_avtor	yil
1	Odamiylik Mulki	4	980	1	2016
...
19	Shaytanat	2	960	1	2020
20	Sherlar guldastasi	4	250	4	0
21	Gulliverni sarguzashtlari	5	180	3	2020

Natijalarga e'tibor beriladigan bo'linsa, "Gulliverni sarguzashtlari" nomli kitob kategoriyasi id_cat=5 bo'lganligi, ya'ni Kitob35 tasavvurining SQL so'rovidagi id_cat IN (2, 4) shartini bajarmaganligi sababli ko'rsatilmaydi.

Aslini olganda 7-misoldagi kabi tasavvurdagi shart bajarilmaydigan holatda undagi ma'lumotlarni yangilash xavfsizlik jixatidan noto'g'ri. Chunki, foydalanuvchiga shart orqali tasavvurda zarur ma'lumotlar taqdim etilgandi. Demak qaralayotgan misolda foydalanuvchi faqat 2 va 4 kategoriyadagi kitoblarni kiritishga ruxsat bo'ladi. Tasavvurlarda bu muammoni hal qilish uchun uning konstruksiyasidagi SNYeSK OPTION kalit so'zi qo'llaniladi.

Tasavvurlarni yangilashda SNYeSK OPTION parametri

SNYeSK OPTION – tasavvurlardagi yangilanishlarni boshqarishga yoki nazorat qilish xizmat qiladi. Bu orqali agar tasavvur SQL so'rovda WHERE sharti ishlatilgan bo'lsa, ushbu shartni qanoatlandiruvchi yangilanishlarga ruxsat beradi. Buning uchun tasavvurni yaratish vaqtida ushbu SNYeSK OPTION o'rnatilgan bo'lishi lozim.

Misolga qaytadigan bo'lsak, oldin Kitob35 tasavvurini o'chiramiz:

DROP VIEW Kitob35

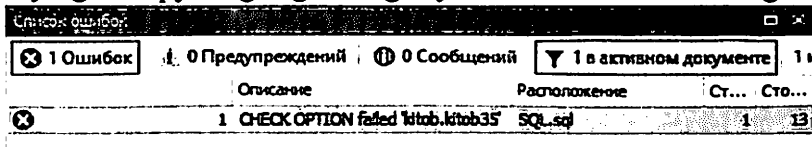
Kitob35 tasavvurini qayta yaratamiz:

```
CREATE VIEW Kitob35 AS
SELECT *
FROM Books
WHERE id_cat IN (2, 4)
WITH CHECK OPTION;
```

Tekshirish uchun Kitob35 tasavvuriga yangi ma'lumotlar kiritamiz.

```
INSERT INTO Kitob35(name, id_cat, page, id_avtor, yil) VALUES
('Talabalik sarguzashtlari', 1, 250, 3, '2019')
```

So'rov bajarilsa, MBBT xatolik to'g'risida ogoxlantiradi. Chunki, INSERT operatorida Kitob35 tasavvuridagi id_cat IN (2, 4) shartni bajarmaydigan 1 qiymatga ega kategoriyali kitobni kiritish so'ralgan.



Yuqorida keltirilgan misol kabi tasavvurlarni yaratishda *SNYeSK OPTION* kalit so'zini o'rnatish nafaqat tasavvur orqali bazaviy jadvalga ma'lumot kiritishda (Insert), balki ma'lumotlarni yangilash (Update) va o'chirishda (Delete) ham ishlatish mumkin. Demak, xulosa qiladigan bo'lsak, eng muhimi tasavvurdagi *SNYeSK OPTION* orqali MBning xavfsizligi ta'minlanadi.

Ma'lumki, tasavvurlar manbasi faqat bazaviy jadvallar bo'lmasdan, balki oldin yaratilgan boshqa tasavvurlar bo'lishi mumkin. *SNYeSK OPTION* parametrlaridagi *LOCAL* va *CASCADED* aynan tasavvur manbasi boshqa tasavvur bo'lgan xolatlarda manba ma'lumotlarni yangilash jarayonlarida qo'llaniladi.

WITH CASCADED CHECK OPTION (Kaskadli nazoratlash)

Kaskadli nazoratlash parametrining samarasini aniq misollar orqali qarab, keyin xulosa qilamiz².

Birinchi. Bitta sonli toifali maydonli t1 jadval yaratiladi:

```
CREATE TABLE T1 (C int);
```

Ikkinchi. T1 manbasi bo'lgan unda C > 10 sharni qanoatlandiruvchi

V1 nomli tasavvur yaratiladi.

```
CREATE VIEW V1 AS
SELECT C FROM T1
WHERE C > 10;
```

Uchinchi. V1 tasavvur orqali T1 jadvalga ma'lumot kiritiladi.

```
INSERT INTO V1(C) VALUES (5);
```

² <https://www.mysqltutorial.org/mysql-view-local-cascaded-in-with-check-option/>

V1 tasavvurda yangilash amali xatosiz bajarilishi sababi unga WITH CHECK OPTION parametrl o'rnatilmagan.

To'rtinchi. Manbasi V1 tasavvur bo'lgan va CHECK OPTION parametrl o'rnatilgan V2 nomli tasavvur yaratiladi.

```
CREATE OR REPLACE VIEW V2 AS  
SELECT C FROM V1  
WITH CASCADED CHECK OPTION;
```

Beshinchi. V2 tasavvur orqali T1 jadvalga ma'lumot kiritiladi.
INSERT INTO V2(C) VALUES (5);

V2 tasavvurda shart berilmagan bo'lsa ham unga ma'lumot kiritish so'rovi bajarilishida MBBT tomonidan xatolik ro'y beradi.

Error Code: 1369. CHECK OPTION failed 'V2'

Oldinchi. V2 tasavvur manbasi bo'lgan V3 nomli tasavvur yaratiladi.

```
CREATE VIEW V3 AS  
SELECT C FROM V2  
WHERE C < 20;
```

Yettinchi. V3 tasavvur orqali T1 jadvalga 8 qiymati kiritiladi.
INSERT INTO V3(C) VALUES (8);

V3 tasavvur tasnifiga ko'ra ma'lumot kiritish to'g'ri bo'lsa ham unga ma'lumot kiritishda xatolik ro'y beradi.

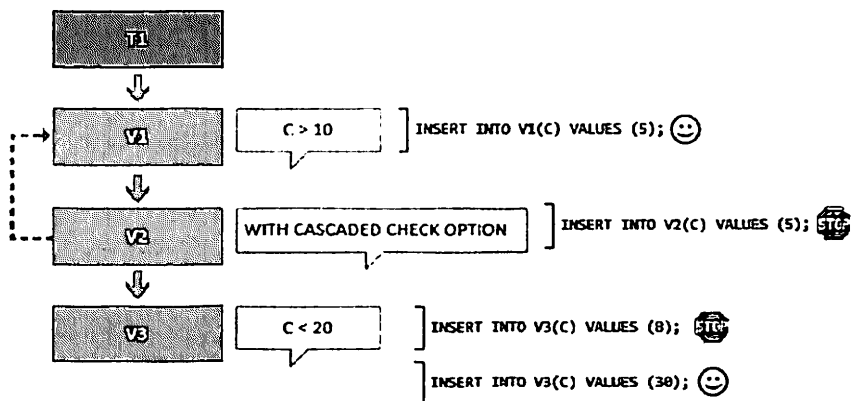
Error Code: 1369. CHECK OPTION failed 'V3'

Sababi V3 tasavvur manbasi V2 tasavvur bo'lib, V2 tasavvur tasnifida CHECK OPTION parametrl o'rnatilgan.

Sakkizinchi. V3 tasavvur orqali T1 jadvalga 30 qiymati kiritiladi.
INSERT INTO V3(C) VALUES (30);

Bu so'rov xatosiz bajariladi. Chunki V3 tasavvur tasnifidagi shartni bajarilishi uchun CHECK OPTION parametri o'rnatilmagan hamda V3 tasavvur manbasi bo'lgan V2 tasavvur tasnifidagi shart bajarilmoqda.

Xulosa qilib aytiladigan bo'linsa, tasavvurda WITH CASCADED CHECK OPTION parametri mavjud bo'lsa, u holda tasavvur manbalari bo'lgan shartlar rekursiv qanoatlandirilgan xolda bazaviy jadvallar yangilanadi CASCADED parametri bajarilishi 5.2.4-rasmga keltirilgan.



5.2.4-rasm.. Tasavvurda *CASCADED* parametri bajarilishi

WITH LOCAL CHECK OPTION (Lokal nazoratlash)

LOCAL parametrining *CASCADED* parametridan farqini misollar orqali qarab chiqamiz.

Birinchi. Oldin yaratilgan V2 tasavvur tasnifidagi *WITH CASCADED CHECK OPTION* parametri *ALTER VIEW* orqali *WITH LOCAL CHECK OPTION* parametriga o'zgartiriladi.

```
ALTER VIEW V2 AS SELECT C FROM V1
WITH LOCAL CHECK OPTION;
```

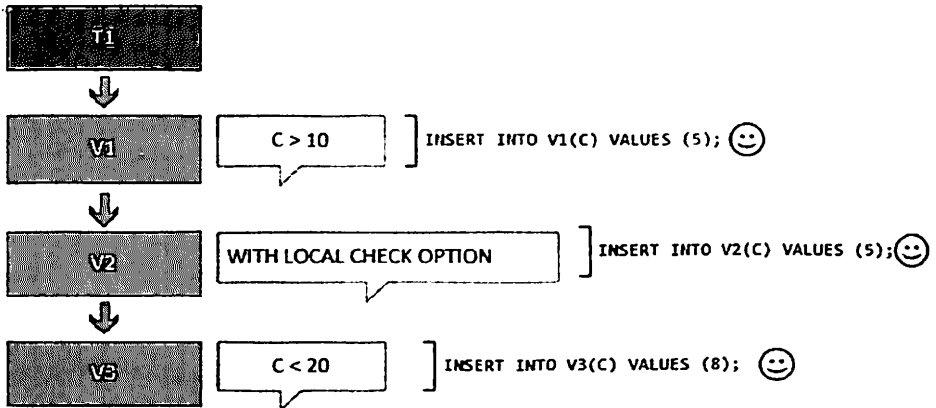
Ikkinchi. V2 tasavvur orqali T1 jadvalga ma'lumot kiritiladi.
`INSERT INTO V2(C) VALUES (5);`

V2 tasavvurdagi yangilanish xatosiz bajariladi. Chunki, manbasi V1 bo'lgan V2 tasavvurda qo'shimcha shartlar yo'q hamda V1 tasavvur tasnifidagi shartlarni o'tkazib nazorat o'rnatilmaganligi uchun yuborish mumkin. E'tibor beriladigan bo'lsak, oldingi misoldagi *WITH CASCADED CHECK OPTION* parametri o'rnatilgan V2 tasavvurda yangilash amali ishlamagan edi.

Uchinchi. V3 tasavvur orqali T1 jadvalga 8 qiymati kiritiladi.
`INSERT INTO V3(C) VALUES (8);`

Bu amal ham muvaffaqiyatli tugatiladi. Sabab, *WITH LOCAL CHECK OPTION* parametri o'rnatilganligi uchun V1 tasavvur tasnifidagi shartlar o'tkazib yuboriladi. Ammo oldingi misolda ushbu qadam bajarilmagan edi.

Xulosa. *WITH LOCAL CHECK OPTION* parametri o'rnatilgan tasavvurlar manbasi bog'liq bo'lgan boshqa tasavvurlardagi shartlarni o'tkazib yuboradi. *LOCAL* parametri bajarilishi 5.2.5-rasmga keltirilib, bu parametri odatda yozilmaydi.



5.2.5-rasm. Tasavvurda LOCAL parametri bajarilishi

Nazorat savollari

1. Tasavvur nima va uning ahamiyati nimada?
2. Tasavvurlar bilan odatiy SQL so'rovlar farqi qanday?
3. Tasavvurlarning afzalliklarini sanang?
4. Tasavvurlarni yaratish sintaksisini izohlang.
5. Necha turdagi tasavvurlar bor va ular bir-biridan qanday ajratiladi?
6. Tasavvurlarni yangilash deganda nimani tushunasiz?
7. Tasavvurdagi SQL so'rovi uchun standart shartlarga nimalar kiradi?
8. Tasavvurlardagi SNYeSK OPTION parametrining vazifasi qanday?
9. Tasavvurlarda nazoratlash parametrini qanday?

6-BOB. SQL TILIDA DASTURLASHTIRISH

6.1. Tranzaktsiyalar

Tranzaktsiya tushunchalari

Tranzaktsiya bu MB bilan bajariladigan ishlar(xarakatlar) ketma-ketligi bo'lib unda barcha ishlar to'raligicha bajariladi yoki birortasi xam bajarilmaydi. Oxirgi xolda MB o'zgarishsiz qoladi. Bunday tranzaktsiya ba'zan atomar deb ataladi. Chunki u bir butun ko'rinishida bajariladi. Masalan, yangi buyurtmani ro'yxatga olishda MB bilan bajarilgan ketma-ket ishlarni ko'rib chiqamiz. Bu quyidagi ketma ketlikdan iborat:

1. xaridorni to'lov puli oshishi bilan bog'liq bo'lgan yozuvni o'zgartirish kerak
2. sotuvchini komission puli oshishi bilan bog'liq bo'lgan yozuvni xam o'zgartirish kerak
3. MBga yangi buyurtma xaqidagi yangi yozuv qo'shish kerak.

Masalan bu ishlarni bajarishni oxirgi bosqichida fayl soxasida hajmi yetmaganligi munosabati bilan uchinchi qadam bajarilmadi. Bunda xaridor uchun bajarilmagan buyurtmaga xaq to'lash bajariladi. Sotuvchi xaridorga yuborgan tovar uchun ustma haq oladi. Masaladan ko'rinib turibdiki, bu uch qadam bir butun bo'lib barchasi bajarilishi kerak yoki birortasi xam bajarilmasligi kerak. Atomar tranzaktsiyada ko'rsatilgan xarakatlarda birortasi bajarilmasa MBda o'zgarish amalga oshmaydi. Tranzaktsiyani bajarish chegaralarini ko'rsatish uchun tranzaktsiyani boshlanishiga tranzaktsiyani saqlashga va tranzaktsiyani qatarishga buyruq berishi zarur. Xar xil MBBT da bu buyruqlar shakli bo'yicha farqlanadi. Agar bir vaqtda ikkita tranzaktsiya bajarilsa, bu tranzaktsiya parallel tranzaktsiya deyiladi.

Tranzatsiya dasturlari to'rt xususiyatga ega: (Atomicity) atomarlik, (Consistency) moslashuvchanlik, (Isolation) ajrala olishlik, (Durability) uzoq vaqtga chidamlilik. Dasturli tranzatsiyalar ACID-tranzaktsiya deb yuritiladi:

Atomarlik xossasi. Tranzatsiya bir butun bajarilishi yoki umuman bajarilmasligi lozim.

Moslashuvchanlik xossasi. Tranzaktsiyaning bajarilishiga bog'liq ma'lumotlar bir kelishilgan xolatdan boshqasiga qo'shimcha qo'llab-quvvatlashsiz tranziktsiya operatorlari orqali o'tadi.

Ajrala olishlik xususiyati. MBdagi barcha ma'lumotlar fizik jixatdan bir-biridan alohida bo'lgan holda parallel yuradi.

Uzoq vaqtga chidamlilik. Agar tranzaktsiya butunligi bilan to'g'ri bajarilsa, u yerda yuajarilgan barcha o'zgarishlar uzoq vaqt saqlanadi.

Tranzaktsiyani boshqarish – relyatsion MBBT da bajarilayotgan turli amallarni boshqarish bo'lib hisoblanadi Tranzaktsiya ichida INSERT, UPDATE va DELETE operatorlarning bajarilishi turadi. Buyruqlar to'g'ri bajarilgandan keyin tranzaktsiyaga yig'ilgan ma'lumotlar o'zgar olmaydi. Tranzaktsiya to'liq yakunlanishi uchun tranzaktsiyani boshqaradigan buyruqlar bor. U orqali MBdagi barcha o'zgarishlarni saqlash va o'chirish mumkin. Tranzaktsiyani boshlash va tamomlashning uchun ikki buyruqi yoki usuli mavjud: COMMIT (fiksatsiya) va ROLLBACK (otkat).

COMMIT – tranzaktsiya bajarilish davomida MB bo'lgan o'zgarishlarni diskge yozish, tranzaktsiya tasdiqlanmaguncha o'zgarishlarni o'chirib, oldingi MBga qaytish mumkin. *COMMIT* vaqtida MB barcha o'zgarishlar doimiy bo'ladi. Agar tranzaktsiya bajarilish vaqtida qandaydir sabablarga ko'ra to'liq yakunlanmasa, MB dastlabki xolatiga qaytadi. COMMIT operatori tomonidan tranzaktsiyani tamomlashdagi natijalar tashqi xotirada kafolatli saqlanadi (commit so'zining ma'nosi tranzaktsiya natijalarini "fiksirlash").

ROLLBACK – SQL operatorlari bilan yakunlanmagan tranzaktsiya tanasidagi barcha o'zgarishlarni bekorlaydi. ROLLBACK operatori tomonidan tranzaktsiyani tamomlashdagi natijalar tashqi xotiradan kafolatli turda o'chiriladi (rollback so'zining ma'nosi tranzaktsiya natijalarini yo'qotish).

Yangi *tranzaktsiya* MB ish seansidan boshlanadi. Undash tashqari agar COMMIT WORK yoki ROLLBACK WORK operatorlari bajarilmasa SQL-operator bajarayotgan barcha ketma-ketlik bir tranzaktsiyaga kiradi.

Tranzaktsiya modellari

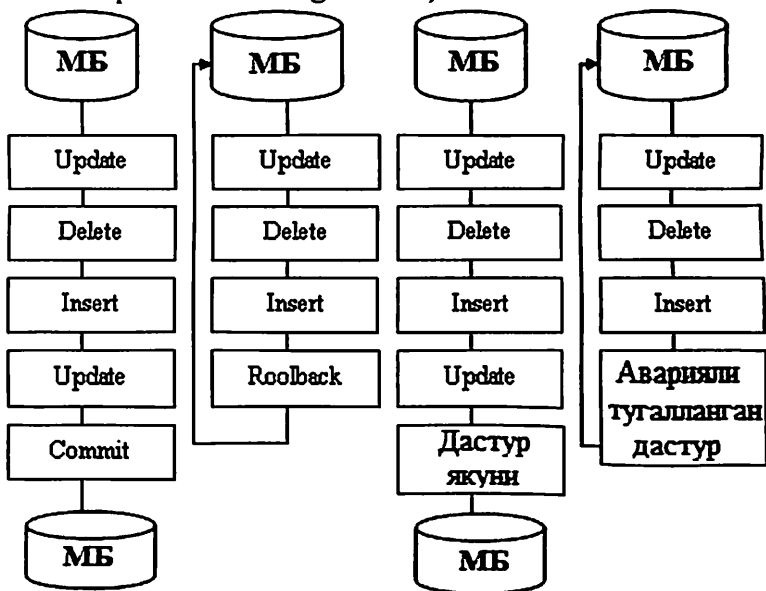
ANSI/SQL standartida tranzaktsiyalar modeli hamda Commit va Rollback operatorlarining vazifaviy namunasi belgilangan. Standartga ko'ra tranzaktsiya joriy MB xolatini o'zgartiradigan foydalanuvchi tomonidan yoki dastur tanasida mavjud bo'lgan SQL operatorlari bilan boshlanadi. Keyingi barcha SQL operatorlar ketma-ketligi tranzatsiya tanasini tashkil etadi. Tanzaktsiya uch rejimdan bittasi orqali boshlanishi kerak: avtofiksatsiya (autocommit), aniq (explicit) yoki aniq emas rejim (implicit). Tijorat MBBTlaring ko'pchiligi, jumladan, SQL Server uchun avtofiksatsiya rejimi qabul etilgan. Shuningdek, tranzaktsiya mumkin bo'lgan quyidagi to'rt yo'ldan bittasi orqali yakunlanadi:

1. Commit operatori tranzaksiyaning muvaffaqiyatli yakunlanganini bildiradi; undan joriy tranzaksiya doirasida MBga doimiy o'zgarishlarni kiritishda foydalaniladi.

2. Rollback operator tranzaksiyani to'xtatadi, o'zgarishlarni bekorlaydi; Rollbackdan foydalangandan keyin yangi tranzaksiya boshlanadi;

3. joriy tranzaksiya boshlangandan keyin dasturni muvaffaqiyatli yakunlangani tranzaksiyaning muvaffaqiyatli yakunlanganini anglatadi (Commit operatori ishlatilgani kabi);

4. dasturning muvaffaqiyatsiz yakunlanishi tranzaksiyani to'xtatadi (ROLLBACK operatori ishlatilgani kabi).



6.1.1-rasm. ANSI/ISO tranzaksiyaning modeli

Dastlab tranzaksiyaning ANSI/ISO modelini tijorat MBBTlari amalga oshirgan bo'lsa, keyinchalik SYBASE MBBTda qo'shimcha amallar kiritilgan kengaytirilgan tranzaksiya modelini ishlab chiqdi. Kengaytirilgan modelda quyidagi to'rt amal qo'llaniladi:

- BEGIN TRANSACTION operatori tranzaksiyag boshlanganini e'lon qiladi;
- COMMIT TRANSACTION operator tranzaksiyaning muvaffaqiyatli yakunlanganini e'lon qiladi;

- SAVE TRANSACTION operator tranzaksiya ichiga saqlash, ya'ni ushbu operator bajarilayotgan vaqtda MBning oraliq holatiga mos joyiga saqlash nuqtasini yaratadi. Tranzaksiyada saqlash nuqtalari ko'p bo'lishiga bog'liq Save Transaction operatori nomga ega bo'lishi mumkin;
- ROLLBACK operatorida ikkita modifikatsiya bor. Agar bu operator qo'shimcha parametrsiz ishlatilsa, butun tranzaksiyani orqaga qaytaradi (bu xolat ANSI/ISO modelidagi Rollback operatori bilan bir hil bo'ladi). Agar operator qo'shimcha parametrlil va ROLLBACK B kabi yozilgan bo'lsa, tranzaksiyadagi V saqlash nuqtasigacha amallarni qisman qaytaradi.

Tranzatsiyaga misol

MBdagi tranzaksiyalarning sintaksis berilishi MBBT turlariga nisbatan turlicha bo'ladi. Misol uchun MySQLda quyidagicha keltiriladi.

START TRANSACTION

... tranzaksiya xususiyatlari ...

WITH CONSISTENT SNAPSHOT

| **READ WRITE**

| **READ ONLY**

BEGIN [WORK]

COMMIT [WORK] [AND [NO] CHAIN] [[NO] RELEASE]

ROLLBACK [WORK] [AND [NO] CHAIN] [[NO] RELEASE]

SET autocommit = {1 | 0} or {no | off}

Misol

KITOB MBdangi Books jadvali ma'lumotlari bilan berilgan bo'lsin. Jadvalda bitta muallifga tegishli bir nomdagi kitoblar turli yillarda chop etilgan. Jadvalga "nashr" nomli yangi ustun qo'shib, unga kitoblarning nashr qilinishni raqamlashtirishni tranzaksiyadan foydalangan holda yozilishi talab etilsin.

1-qadam. Books jadvaliga "nashr" nomli yangi ustun qo'shiladi:

ALTER TABLE Books

ADD COLUMN nashr INT(3) NULL AFTER yil;

Ikkinchi qadam. "nashr" ustunni 1 raqamiga yangilaymiz.

UPDATE Books SET nashr=1

WHERE nashr IS NULL

2-qadam. **START TRANSACTION** orqadi tranzaksiyani qo'shiladi.

START TRANSACTION;

3-qadam. Odatdagi o'rnatilgan holatda MySQL MBga kiritilgan o'zgartirishlarni avtomatik kiritib (fiksirlab) boradi. Dasturlashtirish, jumladan tranzaksiyadagi ko'rsatmalarni amalga oshirish uchun MySQL

da o'zgarishlarni MBga avtomatik kiritmaslik lozim. Buning uchun quyidagi buyruq kiritiladi:

```
SET AUTOCOMMIT=0;
```

yoki

```
SET AUTOCOMMIT=OFF;
```

AUTOCOMMIT=OFF natijasida endi kiritilgan boshqa SQL amallar bajarilmay, vaqtincha ushlab turiladi.

4-qadam. Misol uchun quyidagi ikkita buyruq:

```
DELETE FROM Books
```

```
WHERE id_avtor=1;
```

```
//5 Stroki udalena [0,001c]
```

```
UPDATE Books SET nashr=0;
```

```
17 Stroki obnovlena [< 0,001c]
```

5-qadam. Books jadvalida o'chirish va yangilash amallar MBda virtual bajarilib, haqiqiy MB o'zgarishlar qilmaydi:

```
SELECT *
```

```
FROM Books
```

```
WHERE id_avtor<=2;
```

Natija:

id_book	name	id_cat	page	id_avtor	ylil	nashr
2	Ikki eshik orasi	1	450	2	1996	100
3	Ikki eshik orasi	1	460	2	2000	100
4	Ikki eshik orasi	1	380	2	2012	100
18	Dunyoning ishlari"	4	280	2	2005	100

6-qadam. ROLLBACK buyruqi orqali yuqoridagi tranzaktsiyada keltirilgan ikkita amallarning bajarilishi bekorlanadi:

```
ROLLBACK;
```

Tekshirish uchun yuqoridagi so'rov qayta yuklanadi:

```
SELECT *
```

```
FROM Books
```

```
WHERE id_avtor<=2;
```

Natija:

id_book	name	id_cat	page	id_avtor	ylil	nashr
1	Odamiylik Mulki	4	980	1	2016	1
2	Ikki eshik orasi	1	450	2	1996	1
3	Ikki eshik orasi	1	460	2	2000	1
4	Ikki eshik orasi	1	380	2	2012	1
7	Shaytanat	2	1250	1	2018	1
10	Nafs kishanlari	4	240	1	2015	1
11	Samum	2	380	1	2012	1
18	Dunyoning ishlari"	4	280	2	2005	1
19	Shaytanat	2	960	1	2020	1

Natijadan ko'rinadiki, tranzaktsiya tarkibida keltirilgan delete va update buyruqlarining MBdagi o'zgartirishlari bekorlandi.

7-qadam. Endi masala shartida berilishi bo'yicha kitoblarning nashr qilinishni raqamlashtirish kerak. Buning uchun murakkab xususiyatga ega update so'rovi ishlab chiqiladi. Bu so'rov nashr=1 hamda bir xil nomga,

muallif va kategoriyaga ega bo'lgan kitoblarning soni birdan katta bo'lganlarini topish va ulardagi nashr maydonini orttirish vazifasini bajaradi. So'rov bir necha marta qaytarib, birinchi bajarilganda 2 ga keyin bajarilganda 3 va h.k. nashr maydonini yangilaydi. Yuqoridagi tranzaktsiya yakunlanmadi. Shuning uchun ish davom qildiriladi:

UPDATE Books AS B,

```
(SELECT DISTINCT id_book FROM Books WHERE nashr=1
GROUP BY name, id_cat, id_avtor
HAVING COUNT(*) > 1 ORDER BY yil) AS A
```

```
SET B.nashr = 2 WHERE B.id_book=A.id_book;
```

```
//3 Stroki obnovlena [0,001c]
```

UPDATE Books AS B,

```
(SELECT DISTINCT id_book FROM Books WHERE nashr=1
GROUP BY name, id_cat, id_avtor
HAVING COUNT(*) > 1 ORDER BY yil) AS A
```

```
SET B.nashr = 3 WHERE B.id_book=A.id_book;
```

```
//1 Stroki obnovlena [0,001c]
```

8-qadam. SAVEPOINT orqali tranzaktsiyaga belgi qo'yish:

SAVEPOINT belgi;

9-qadam. SET orqali o'zgaruvchilarni belgilash:

```
SET @name:='Ikki eshik orasi';
```

```
SET @id_cat=1; SET @id_avtor=2;
```

```
SET @page=400; SET @yil=2020; SET @nashr=1;
```

10-qadam. Kiritilgan o'zgaruvchilarni shartda qanoatlandiruvchi, ya'ni berilgan kitobning eng katta nashr raqamini topish:

```
SELECT @nashr:=MAX(nashr)+1
```

```
FROM Books
```

```
WHERE name=@name AND id_cat=@id_cat AND id_avtor=@id_avtor;
```

Natija quyidagi shaklda chiqadi

```
@nashr:=MAX(nashr)+1
```

```
4
```

11-qadam. SET ko'rsatilgan o'zgaruvchilarni va aniqlangan nashr raqamini Books jadvaliga yozish:

```
INSERT INTO Books (name, id_cat, page, id_avtor, yil, nashr)
```

```
VALUE (@name, @id_cat, @page, @id_avtor, @yil, @nashr);
```

12-qadam. Natijani ko'rish:

```
SELECT * FROM Books WHERE id_avtor<=2;
```

Natija:

id_book	name	id_cat	page	id_avtor	yil	nashr
1	Odamiylik Mulki	4	980	1	2016	1
2	Ikki eshik orasi	1	450	2	1996	2
3	Ikki eshik orasi	1	460	2	2000	3
4	Ikki eshik orasi	1	380	2	2012	1
7	Shaytanat	2	1250	1	2018	2

10	Nafs kishanlari	4	240	1	2015	1
11	Samum	2	380	1	2012	1
18	Dunyoning ishlari"	4	280	2	2005	1
19	Shaytanat	2	960	1	2020	1
57	Ikki eshik orasi	1	400	2	2020	4

13-qadam. Tranzaktsiyadagi amallarni SAVEPOINT orqali tranzaktsiyada ko'rsatilgan belgi gacha bekorlash:
ROLLBACK TO SAVEPOINT belgi;

14-qadam. COMMIT orqali tranzaktsiyai amallarini bajarib, yakunlash:
COMMIT;

// Vipolnenie zaversheno uspeshno [$< 0,001c$]

Tranzaktsiyaga sharx. Agar 13-qadam amali bajarilmaganda 12-qadamdagi natija MBda saqlangan bo'lar edi. 10-qadamda Books jadvaliga kiritiladigan kitoblarning nashr raqamlarini avtomatik davom qilish uchun aynan oldingi nashr raqamini aniqlash so'rovi bajarildi.

Xulosa. MBda tranzaktsiyani amalga oshirishda ushbular bajariladi:

- 1) START TRANSACTION – tranzaktsiyani qo'shish;
- 2) AUTOCOMMIT=0 rejimini o'rnatish;
- 3) O'zgatuvchilarni SET orqali e'lon qilish;
- 4) SAVEPOINT – tranzaktsiyaga belgi qo'yish;
- 5) ROLLBACK yoki ROLLBACK belgi orqali buyruqlarni bekorlash;
- 6) COMMIT –tranzaktsiyadagi amallarni yakunlash.

Nazorat savollari

1. MB da tranzaktsiyalar va uning ahamiyati nimada?
2. Tranzatsiya xossalari qanday?
3. Tranzaktsiya tarkibida qaysi operatorlardan iborat bo'lishi mumkin?
4. Tranzaktsiyani boshlash va yakunlash buyruqlari qanday?.
5. Tranzaktsiya modellarini tasniflang.
6. Tranzaktsiyalarda kengaytirilgan modelda qanday amal qo'llaniladi?

6.2. Saqlanadigan protsedura va funktsiyalar

Delimiter – ajratuvchi

MB bozorining uzoq vaqt davom etuvchi tendentsiyalardan biri bu ma'lumotlarni qayta ishlash arxitekturalari axamiyatining o'sib borishi bo'lib hisoblanadi. Dorelyatsion tizimlar asosan ma'lumotlarni saqlash va tanlashga, amaliy dasturlar navigatsiya, ma'lumotlarni tartiblash, saralash va qayta ishlash uchun javob beradi. Ya'ni dastlabki SQL MBdan zurur ma'lumotlarni chiqarish, saralash va tartiblash amallarini bajarib kelgan. Hozirgi kunda barcha axborot tizimlarining asosini MB tashkil qilinayotgani, ma'lumotlar oqimini aloqa tizimlari orqali tranzaktsiyalar bilan olib borilishi natijasida SQL dasturlash imkoniyatlari paydo bo'ldi.

MBBTlarida biror maqsadni amalga oshirish uchun dasturlash tillaridagi kabi amallarni bajarish lozim bo'lib qoladi. Ma'lumki, C, C++, C#, Java, PHP kabi dasturlash tillarida buyruqlar tugallanganligi ham nuqtali vergul orqali amalga oshiriladi. Bu qoida SQL dasturlashtirishda qo'llaniladigan SQL dasturlarni (saqlanuvchi protsedura va funktsiyalar, trigger va kursorlar) yaratishga tegishli hisoblanadi. SQL dasturlar xuddi boshqa dasturlashtirish tillaridagi kabi biror maqsadni amalga oshirish uchun ko'plab operator va buyruqlar ketma-ketligidan iborat koddan tashkil topadi. SQL dasturni bajarish vaqtida undaga kodlar bir butun kompiyatsiya qilinishi lozim. Demak, SQL dasturini bajarishning o'zi bitta SQL buyruqi o'rnida bo'ladi.

SQLda biror buyruq tugallanishi nuqtali vergul orqali amalga oshiriladi. Agar bitta SQL ishlatilsa, odatda bu qo'yilmaydi. Agar bir nechta buyruqlar ket-ketlikda bajariladigan bo'lsa, ularning orasi albatta nuqtali vergul orqali ajratiladi.

```
SELECT * FROM products;  
SELECT * FROM customers;
```

Shuning uchun ham bu dasturda qo'llaniladigan qoidaviy nuqtali vergullar muammo keltirib chiqaradi. Bunda yoki dastur kapsula olinishi yoki SQL buyruqlarini boshqa simvoldagi ajratuvchiga o'zgartirish lozim.

SQL yuqoridagi muammoni hal qilishda maxsus **Delimiter** (ajratuvchi) kalit so'zi mavjud. Delimiter SQL dasturini serverga bitta operator o'rnida yuborish uchun vaqtinchalik ajratuvchi belgilash vazifasini bajaradi. SQLda Delimiter quyidagicha amalga oshiriladi.

```
DELIMITER delimiter_character
```

bu yerda `delimiter_character` bitta yoki ko'plab belgili so'z (ms: //, \$, \$\$, %%).

```
DELIMITER //
SELECT * FROM customers //
SELECT * FROM products //
```

Operatorlar jamlanmasi bajarilib bo'lgandan keyin yana Delimiter nuqtali vergulga aylantirish lozim.

```
DELIMITER ;
```

Aks holda bu ish MBBT qayta yuklanganda avtomatik bajariladi. Misol uchun MySQLda saqlanuvchi protseduralarni yaratish uchun Delimiter ishlatilishi quyidagicha bo'lishi mumkin:

```
DELIMITER $$
CREATE PROCEDURE sp_name()
BEGIN
  - statements
END
$$ DELIMITER ;
```

Saqlanuvchi protsedura misoli o'rnida keltirilgan SQL dasturida quyidagi vazifalar bajarilishi lozim:

1. standart ajratuvchi \$\$ ga o'zgartiriladi;
2. dastur tanasida (;) ishlatilib, dastur nixoyasi END bo'ladi;
3. tanlangan ajratuvchi (\$\$) standart ajratuvchiga o'zgartiriladi

Eslatma. Ko'pchilik MBBTlari bilan ishlovchi amaliy dasturiy majualarda (dbForge Studio, MySQL manager) Delimiter o'rnatish talab etilmaydi. Chunki, bu dasturlarda foydalanuvchi uchun interfeys orqali tayyor andoza yaratilib beriladi.

Saqlanadigan protseduralar

Saqlanadigan protseduralar (SP, ing.: Stored Procedure - SP) - bog'langan SQL operatorlarining guruhi. SP SQL operatorlarining tegishli guruhlarini anglatadi. Ulardan foydalanish dasturchining ishini soddalashtiradi va yanada moslashuvchan qiladi. Shuning uchun alohida ketma-ket SQL operatorlariga qaraganda SPlarni bajarish samaraliroq bo'ladi. SP bir yoki bir nechta SQL ko'rsatmalari yoki funktsiyalaridan iborat bo'lgan va MBda mavjud buyruqlar to'plamini ta'minlaydi.

MBa alohida SQL operatorlari o'rniga SPni qo'llash quyidagi afzalliklarni ta'minlaydi:

- SDda zarur bo'lgan operatorlar mavjud;
- SP *sintaktik tahlil bosqichidan o'tadi va bajarilishga tayyor bo'ladi.* Ya'ni SP kompilyatsiya qilinib, serverda saqlanadi. Shuning uchun SP SQL operatorlar ketma-ketligiga qaraganda tezrkordir;
- SP dasturlarning bajarilishini osonlashtiradi: protseduralar yangilaganishlari barcha ilovalar uchun avtomatik aks etadi;

- SPni bajarishidan oldin SQL Server SP bajarilish rejasini tuzadi, uni optimallashtirish va tuzishni amalga oshiradi;
- Katta vazifalarni boshqarish qismida kichik va samarali vazifalarga bo'lish mumkinligi sababli, SP modulli dasturlashni qo'llab-quvvatlaydi. Shuning uchun ular yagona MBga murojlat qilingan mijoz dasturlari uchun umumiy bo'lib, kodlarning takrorlanishi oldini oladi va dastur hajmini kamaytiradi;
- SR boshqa SP va funktsiyalarini chaqirishlari hamda SPni o'zi ham boshqa turdagi dasturlarni chaqirishi mumkin;
- SPdan foydalanish samaraliroq yoki axborot tizimining samaradorligini oshiradi: ular o'nlab yoki yuzlab buyruqlardan iborat bo'lishi mumkin, ammo ularni bajarish uchun zarur bo'lgan SPLarning nomlarini ko'rsatish kifoya. Bu mijozdan serverga yuborilgan so'rovlar sonini kamaytiradi va bu o'z navbatida tarmoq trafiginu kamaytiradi.
- SP MB serveriga mantiqiy ma'lumotni chiqarishga imkon beradi. Bu axborot tizim MBning mijoz qismiga bog'liqligini kamaytiradi;

Umumiy holda SP tarmoq trafiginu kaytiradi va tizimning samarali ishlashini kafolatlaydi. SP jadvallar yoki MBning ob'ektlariga bog'liq emas. Ular mijoz dasturi, boshqa SP yoki trigger tomonidan chaqiriladi. Ishlab chiquvchi SP kirish huquqlari va vakolatlarni boshqarishi, bajarishini cheklashi mumkin. SPni modifikatsiya qilishni faqat uning egasi yoki MBda fiksirlangan rolga ega ishtirokchilar olib bora oladi. Zaruriyat bo'lsa, uni takomillashtirish huquqi bir foydalanuvchidan boshqasiga o'tkazishi ham mumkin.

SQL Server bilan ishlashda foydalanuvchilar turli xil amallarni bajaradigan o'z protseduralarini yaratishlari mumkin. SP MBning eng ahamiyatli ob'ekti bo'lib, ularning har biri ma'lum bir MBda saqlanadi.

SP yuqori tezlikda ishlashni ta'minlaydigan MBda va saqlanadi. SPga kiritilgan parametrlarni olish, dasturga qaytarish va dasturdan chaqirish mumkin. SPLarning ikki turi mavjud:

- Tanlash protseduralari. Dastur Select operatoridagi jadvallar yoki tasavvurlar o'rniga ishlatilishi mumkin. PS bir yoki bir nechta qiymat qaytarishi lozim, aks holda SP natijasi noto'g'ri bo'ladi;
 - Bajariluvchi protseduralar maxsus operatorlar yordamida aniq chaqiriladi. Bajariluvchi protsedura chaqirilayotgan dasturga natija qaytarmasligi mumkin.
- SQL Serverda SPning bir necha turlari mavjud:
- turli ma'muriy harakatlar bajarishga mo'ljallangan *tizimli SP*;

- turli xil harakatlarni amalga oshiruvchi *foydalanuvchi SP*;
- *Vaqtinchalik so'rovlar jarayoni qisqa vaqt davomida amalga oshiriladi*, so'ngra avtomatik ravishda server tomonidan o'chiriladi. Ular lokal va global bo'linadi. Lokal vaqtincha SRLar ulangan joydan chaqiriladi. Shlobal vaqtincha SRLar har qanday serverlarga ulanishda talab etiladi.

Saqlanadigan protseduralarni yaratish

SP yaratish uchun ushbu vazifalarni bajarish lozim:

- yaratiladigan SPning turini aniqlash (vaqtinchalik/maxsus);
- foydalanish huquqlarini rejalashtirish;
- SP parametrlarini aniqlash;

Protsedura tili SQL o'rinlanishiga juda zich bog'liq. Ma'lumotlar manipulyatsiyasi va kengaymalari uchun SQL barcha ko'rsatmalarni qamrab oladi, jumladan:

- shartli operatorlar;
- turli xil takrorlash (tsikl) operatorlari;
- istisnoli vaziyatlarga ishlov berish imkoniyatlari.

SP sarlavha va tanadan iborat. *Sarlavha* qismiga quyidagilar kiradi:

- *Nomi*. Protsedura nomi MBdagi jadval va protsedura nomlari bo'yicha unikal bo'lishi kerak;
- *Kirish parametrlar ro'yxati va ularning ma'lumot turlari*. Protsedura parametrlarni (mavjud bo'lmasligi mumkin) chaqirayotgan dasturdan qabullaydi;
- *Chiqish parametrlar ro'yxati va ularning ma'lumot turlari*. Agar protsedura chaqirilayotgan dasturga qiymatlar qaytarsa ishlatiladi;

Protseduraning *tana* qismi quyidagilardan iborat:

- Lokal o'zgaruvchilar ro'yxati va ularning ma'lumot toifalari (agar ular protsedura kodida ishlatilsa);
- Protsedura va trigger tilidagi ko'rsatmalar bloki BEGIN va END kalit so'zlari orasida bo'ladi. Blok ichiga bir nechta darajali joylashuvchi boshqa bloklar ham kirishi mumkin.

SPni yaratishda CREATE PROCEDURE operatoridan foydalaniladi.

Create procedure operatori MBda yangi SPni belgilaydi.

Yangi SP yaratish va mavjud SPni o'zgartirish buyruqlari MBBT turiga qarab turli sintaksis ko'rinishga ega bo'ladi. Jumladan, Transact-SQL dagi sintaksis quyidagicha bo'ladi:

```
CREATE [ OR ALTER ] { PROC | PROCEDURE }
[ schema_name. ] procedure_name [ ; number ]
[ { @parameter [ type_schema_name. ] data_type }
```

```

    [ VARYING ] [ = default ] [ OUT | OUTPUT | [ READONLY ]
    ] [ ,...n ]
[ WITH <procedure_option> [ ,...n ] ]
[ FOR REPLICATION ]
AS { [ BEGIN ] sql_statement [ ; ] [ ...n ] [ END ] }
[ ; ]
<procedure_option> ::=
    [ ENCRYPTION ]
    [ RECOMPILE ]
    [ EXECUTE AS Clause ]

```

MySQL dagi sintaksis:

```

CREATE [ DEFINER = { user | CURRENT_USER } ]
PROCEDURE sp_name ((proc_parameter[,...]))
[characteristic... ] routine_body
proc_parameter: [ IN | OUT | INOUT ]
param_name type: Any valid MySQL data type
characteristic: COMMENT 'string'
| LANGUAGE SQL
| [ NOT ] DETERMINISTIC
| { CONTAINS SQL | NO SQL | READS SQL DATA
| MODIFIES SQL DATA }
| SQL SECURITY { DEFINER | INVOKER }
routine_body:
Valid SQL routine statement

```

Tizimli yoki vaqtinchalik SPni yaratishda *sp_*, *#*, *##* prefiksalaridan foydalanish mumkin. Nomdagi raqam protsedura guruhidagi SPning identifikatsiya raqamini aniqlaydi. Kirish va chiqish parametrlari ma'lumotlarini qabullash va uzatish uchun qo'llaniladi. Ularning nomlari lokal o'zgaruvchilarning nomlari kabi *@* belgisi bilan boshlanishi kerak.

SPning sifatli kirish parametrlarida *OUTPUT* kalit so'zi *CURSOR* ma'lumot turidan foydalaniladi.

VARYING kalit so'zi *CURSOR* turidagi *OUTPUT* parametr bilan birgalikda ishlatiladi. U chiqish parametri natijalar to'plamini aniqlaydi. *DEFAULT* kalit so'z parametrga mos keladigan o'rnatilgan qiymatni bildiradi.

RECOMPILE - tizimda har bir protsedurani bajarish rejasi tuziladi.

FOR REPLICATION parametri chiqariluvchi ma'lumotlar sifatini ta'minlashdagi ma'lumotlar replikatsiyasi uchun talab qilinadi.

ENCRYPTION kalit so'zi serverga SP kodini shiflashni buyuradi. U SPdagi mualliflik huquqili algoritmlardan foydalanishni himoyalaydi. *RETURN* buyrug'i SP bajarilishini yakunlaydi.

Saqlangan protsedurani bajarish

SPni bajarishga yuboradigan operator protseduraga bog'liq. Tanlash jarayoni *SELECT* ma'lumotlarini tanlash operatoridan foydalangan holda

ular bilan bog'lanishda amalga oshiriladi. Bajariladigan SP chaqirish uchun maxsus operatorlardan foydalanib, bu operatorlar MBBT turlari bo'yicha bir-biridan farqlanadi. Misol uchun SQL Serverda *EXECUTE* operatoridan foydalanish tavsiya etiladi.

SPni bajarish uchun quyidagi buyruq ishlatiladi:

```
{ { EXEC | EXECUTE } }
  procedure_name
  [ { value | @variable [ OUT | OUTPUT ] } ] [ ...n ] }
[;]
```

MySQLda *CALL* operatori orqali SP chaqiriladi.

```
CALL procedure_name
  [ { value | @variable [ OUT | OUTPUT ] } ] [ ...n ] }
[;]
```

Saqlangan protseduralar ustida amallar

Buyruq nomi	Tavsifi
<i>Create Alter Drop Procedure</i>	Protsedura yaratish taxrirlash o'chirish
<i>Show create procedure p name</i>	P name protsedura matnini ko'rsatish
<i>Show procedure status like 'p name'</i>	p_name protsedura tasnifini ko'rsatish
<i>Call proc name()</i>	proc name nomli protsedurani chaqirish
<i>Declare</i>	lokal o'zgaruvchini aniqlash
<i>Set</i>	lokal va global o'zgaruvchiga qiymat berish
<i>Select ... into</i>	Ko'rsatilgan ustundagi o'zgaruvchiga qiymatni saqlash
<i>If..Then..Elseif..Else...End If</i>	if-then-else-end shart operatori
<i>Case ... when</i>	Tanlash operatori
<i>Loop, repeat, while</i>	Sikl
<i>Returns</i>	Funksidagi qiymatni qaytarish

Misollar

Books

id_book	name	id_cat	page	id_avtor	yl	nashr
1	Odamiylik Mulki	4	980	1	2016	1
2	Ikki eshik orasi	1	450	2	1996	2
3	Ikki eshik orasi	1	460	2	2000	3
4	Ikki eshik orasi	1	380	2	2012	1
5	Sariq devni minib	5	220	3	1990	2
6	Sariq devni minib	5	180	3	2005	1
7	Shaytanat	2	1250	1	2018	2
8	Dexqon va ayiq	5	100	5	2019	1
9	Eng katta sexr	5	230	5	2020	1
10	Nafs kishanlari	4	240	1	2015	1
11	Samum	2	380	1	2012	1
12	Jannati odamlar	4	210	3	1996	1
13	Shirin qovunlar mam	5	180	3	1986	1
14	Qora tol	1	350	4	1988	1
15	Dala armonlari	1	450	4	1967	1
16	Mangu buloq	4	360	4	1970	1
17	Sehrli qalpoqcha	3	250	3	1964	1

Avtor

id_avtor	fla
1	Toxir Malik
2	O'tkir Hoshimov
3	Xudayberdi To'xtaboyev
4	Ibraym Yusupov
5	Xalq ogzaki ijodi

Categoriya

id_cat	name	cat
1	Tarixiy	
2	Detektiv	
3	Fantastik	
4	Tarbiyaviy	
5	Ertak	

18	Dunyoning ishlari"	4	280	2	2005	1
19	Shaytanat	2	960	1	2020	1
20	Sherlar guldastasi	4	250	4	0	1
21	Gulliverni sarguzashtlari	5	180	3	2020	1

Ms 1. Oddiy protsedura. Books jadvalidan kiritilgan muallif kodiga mos kitobni chiqaruvchi oddiy SP tuzish

DELIMITER \$\$

CREATE PROCEDURE P_kitob(IN avtor INT)

BEGIN

SELECT *

FROM Books

WHERE id_avtor = avtor,

END \$\$

DELIMITER ;

Yaratilgan P_kitob protsedurasi endilikda joriy MBda doimiy ravishda saqlanadi. Undan ixtiyoriy vaqtda chaqirib foydalanish mumkin. P_kitob protsedurasini chaqirish quyidagicha bo'ladi:

CALL KITOB.P_kitob(4);

Natija:

id_book	name	id_cat	page	id_avtor	yl	nashr
14	Qora tol	1	350	4	1988	1
15	Dala armonlari	1	450	4	1967	1
16	Mangu buloq	4	360	4	1970	1
20	Sherlar guldastasi	4	250	4	0	1

Ms 2. Ko'p amalli protsedura. Books jadvaliga yangi ma'lumot kirituvchi SP yaratish. SP quyidagi shartlarni qanoatlandirsin:

- 1) yangi yozuvdagi muallif kodi Avtor jadvalida bo'lmasa, oldin Avtor jadvaliga "Muallif+kiritilgan avtor kodi" qiymatli muallif qayd etilsin va uning kodi olinsin;
- 2) yangi yozuvdagi kategoriya kodi Categoria jadvalida bo'lmasa, Categoria jadvaliga "Kategoriya+kiritilgan kategoriya kodi" qiymatli kategoriya qayd etilsin va uning kodi olinsin;
- 3) Kitobdagi muallfi, kategoriyasi va nomi bir xil kitoblar nashr maydoniga qayd etilsin
- 4) oxirgi yozuv muallif va kategoriyasi bilan chiqarilsin

DELIMITER \$\$

CREATE DEFINER = 'root'@'%'

PROCEDURE B_kitob(IN names VARCHAR(255), IN cat INT, IN avtor INT)

BEGIN

```

/*O'zgaruvchilarni tariflash */
DECLARE cavor, ccat, cnashr INT;
DECLARE hh varchar(250);
/*O'zgaruvchiga qiymat berish */
SET cavor=0;
/*Muallif ID kiritilgan avtor bilan tekshirish */
SET cavor=(SELECT COUNT(*) FROM Avtor WHERE id_avtor=avtor);
/*Muallif jadvalda yo'q bo'lsa */
IF cavor = 0 THEN
    /*Avtor jadvaliga yangidan qayd qilish */
    INSERT INTO Avtor (fio) VALUES (CONCAT("New Avtor ",avtor));
    /*Avtor jadvalidagi oxirgi ID ni aniqlash */
    SET avtor=(SELECT MAX(id_avtor) FROM Avtor);
END IF;
/*Kiritilgan kategoriyani tekshirish */
SET ccat=0;
SET ccat=(SELECT COUNT(*) FROM Categoria WHERE id_cat=cat);
IF ccat = 0 THEN
    INSERT INTO Categoria (name_cat)
    VALUES (CONCAT("New Categoria ", cat));
    SET cat=(SELECT MAX(id_cat) FROM Categoria);
END IF;
/*Books jadvaldan nashr raqamini aniqlashtirish */
SET cnashr=(SELECT MAX(nashr)
FROM Books WHERE name=names AND id_avtor=avtor AND id_cat=cat);
IF cnashr > 0 THEN SET cnashr = cnashr + 1;
ELSE SET cnashr = 1;
END IF;
/*Books jadvalga ma'lumot yozish */
INSERT INTO Books (name, id_cat, page, id_avtor, yil, nashr)
VALUE (names, cat, 100, avtor, 2020, cnashr);
/*Books jadvaldan yozilgan ma'lumotlarni chiqarish */
SELECT A.id_avtor, A.fio, C.name_cat, C.id_cat,
B.id_book, B.name, B.nashr, B.page, B.yil FROM Books B
INNER JOIN Categoria C ON B.id_cat = C.id_cat
INNER JOIN Avtor A ON B.id_avtor = A.id_avtor
WHERE A.id_avtor=avtor AND B.id_cat=cat;
END $$
DELIMITER ;

```

Berilgan Avtor va Categoria jadvallarida 5 tadan yozuv mavjud. Endi ushbu SPni turli qiymatlar bergan holda tekshirib ko'ramiz.

CALL B_kitob("Giybat ozi nima", 6, 1);

Natija:

id_avtor	fio	id_cat	name_cat	id_book	name	nashr
1	Toxir Malik	6	New Categoria	6	Giybat ozi nima	1

CALL B_kitob("Giybat ozi nima", 6, 1);

Natija:

id_avtor	fi	id_cat	name_cat	id_book	name	nashr
1	Toxir Malik	6	New Categoria 6	23	Giybat ozi nima	1

CALL B_kitob("Giybat ozi nima", 6, 1);

Natija:

id_avtor	fi	id_cat	name_cat	id_book	name	nashr
1	Toxir Malik	6	New Categoria 6	24	Giybat ozi nima	1

CALL B_kitob("Ikki eshik orasi", 1, 2);

Natija:

id_avtor	fi	name_cat	id_cat	id_book	name	nashr	page	yl
2	Utkir Hoshimov	Tarixiy	1	2	Ikki eshik orasi	2	450	1996
2	Utkir Hoshimov	Tarixiy	1	3	Ikki eshik orasi	3	460	2000
2	Utkir Hoshimov	Tarixiy	1	4	Ikki eshik orasi	1	380	2012
2	Utkir Hoshimov	Tarixiy	1	25	Ikki eshik orasi	4	100	2020

CALL B_kitob("Nafs kishanlari", 4, 1);

Natija:

id_avtor	fi	name_cat	id_cat	id_book	name	nashr	page	yl
1	Toxir Malik	Tarbiyaviy	4	1	Odamiylik Mulki	1	980	2016
1	Toxir Malik	Tarbiyaviy	4	10	Nafs kishanlari	1	240	2015
1	Toxir Malik	Tarbiyaviy	4	26	Nafs kishanlari	2	100	2020

Saqlanadigan funktsiyalar

Ma'lumki, SQLda matematik va sanali ma'lumotlar bilan ishlovchi ko'plab funktsiyalar mavjud. Lekin, amaliyotda qo'yilgan masalani yechishda mavjud standart funktsiyalar yetarli bo'lmaydi. Hozirgi zamonaviy MBBTlarida bu muammolar yechilgan bo'lib, SQL da dasturlashning bir yo'nalishi bo'lgan maxsus saqlanuvchi funktsiyalarni yaratish va undan doimiy ravishda foydalanish mumkin.

Saqlanadigan funktsiyalar (SF - Stored Function) – bu protsedura kabi foydalanuvchi tomonidan ishlab chiqilgan biror maqsadni amalga oshiruvchi amallar ketma-ketligidan iborat dastur SF MBda saqlanib, undan natija qaytadi. Natijalar kiritilgan yoki bo'sh argumentlar bo'yicha nafaqat MBdagi jadval, tasavvurlar bilan ishlash, balki ixtiyoriy algoritmik hisoblashlarni bajaradi.

Oldingi mavzularda SQL funktsiyalari standart, agregat va saqlanuvchi guruhlarga ajratilgan edi. SF ham standart funktsiyalarning xarakterga ega. Ya'ni bu funktsiyalar dasturiy jixatdan tuzilishidan qat'iy nazar SQL so'rovlari orqali chaqirilganda har bir qator bo'yicha alohida bajariladi. Ya'ni bu funktsiyalarga kiruvchi argumentlar soni va toifasi aniq bo'lib, statik funktsiyalar kabi ko'p argumentlar uchun bajarilmaydi. SF va SPlar MBBTning jadval, tasavvur kibi ob'ektlari qatoriga kiradi.

Shuning uchun bular bilan ishlashda SQL ning DDL guruh (create, alter, drop, show) operatorlari qo'llaniladi.

SFlar bilan ishlash SPlar bilan bir xil. Ya'ni, SF yaratish - *CREATE Function*, qayta o'zgartirish *ALTER Function* va o'chirish *DROP Function*, SF dastur kodini ko'rish - *SHOW Create Function func_name* kabi operatorlar orqali amalga oshiriladi.

Shuningdek, funktsiya bilan ishlash ham *DELIMITER* – ajratuvchini o'zgartirish orqali olib boriladi.

Funktsiyaning dastur kodida algoritmik operatorlarni, ya'ni *Declare* – o'zgaruvchilarni ta'riflash, *Set* – o'zgaruvchilarga qiymat berish, *Select ... Into* – *Select* operatorida ko'rsatilgan ustunlardagi qiymatlarni saqlash, *If-Then-Else-Else-End* shart, *Case ... When* – tanlash, *Loop, Repeat, While* – tsikl operatorlari bilan ishlash mumkin.

Funktsiyaning natijaviy qiymatini qaytarishda *RETURNS* operatoridan foydalaniladi.

SF bilan ishlash SPlar bilan farqi birinchidan, funktsiya faqat bitta natija qaytaradi, ikkinchidan, undan nomi ko'rsatilgan holda to'g'ridan-to'g'ri foydalanish mumkin.

SFlar MBBTlarining turlariga qarab ular bilan ishlashda ozroq stilistik farq qiladi. Quyida SFni MySQLdagi sintaksisi keltirilgan.

CREATE

[DEFINER = user]

FUNCTION sp_name ((func_parameter{...}))

RETURNS type

[characteristic ...] routine_body

func_parameter:

param_name type

type:

Any valid MySQL data type

characteristic: {

COMMENT 'string'

| LANGUAGE SQL

| [NOT] DETERMINISTIC

| {CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA}

| SQL SECURITY {DEFINER | INVOKER}

}

routine_body:

Valid SQL routine statement

Misollar

Ms 1. O'zlashtirish reytingini anglatuvchi 100 gacha bo'lgan butun son berilgan. Ushbu sonni mezon bo'yicha (2 baxo 0-55; 3 baxo 56-70; 4 baxo 71-85; 5 baxo 86-100) baxoga aylantiradigan SF tuzish talab etilsin.

Masalani yechishda yaratiladigan SF dasturida IF-THEN shart operatoridan foydamiz:

```

DELIMITER bes
CREATE FUNCTION Rey_func(reyting INT)
RETURNS int(1)
COMMENT 'Reyting ballni bahoga aylantirish'
BEGIN
    DECLARE baxo INT(1);
    /* Reyting shartlarini tekshirish */
    IF reyting<=55 THEN SET baxo=2;
    ELSEIF reyting>55 AND reyting<=70 THEN SET baxo=3;
    ELSEIF reyting>70 AND reyting<=85 THEN SET baxo=4;
    ELSEIF reyting>85 AND reyting<=100 THEN SET baxo=5;
    ELSE SET baxo=0;
    END IF;
    /* Funktsiya natijasini qaytarish */
    RETURN baxo;
END bes
DELIMITER ;

```

SFni yaratishda DELIMITER ni bes so'zi bilan belgilab olindi. Funktsiya tanasidagi RETURNS - funktsiya qaytaradigan ma'lumot toifasi. COMMENT – SFga berilgan izox. SF tanasida ko'p o'zgaruvchi ishlatilish mumkin bo'lgani uchun ulardan qaysi biri funktsiya natijasi sifatida qaytarilishini RETURN belgilaydi.

Saqlangan Rey_func() funktsiyani tekshirish:

```

SELECT Rey_func(40) // Natija: 2
SELECT Rey_func(78) // Natija: 4

```

Ms 2. Kiritilgan tug'ilgan kun sana bo'yicha uning burjini chiqaruvchi SP tuzish talab etilsin.

SP algoritmi quyidagicha: 1) sanadan oy va kun ajratib olinadi; 2) Case ... When tanlash operatori yordamida burj topiladi.

```

DELIMITER bes
CREATE DEFINER = 'root'@'%'
FUNCTION burj(sana date)
RETURNS varchar(100) CHARSET utf8
BEGIN
    DECLARE yil, ay, kun int;
    DECLARE nomi varchar(20);
    /* kiruvchi sana o'zgaruvchidag oy va kunni ajratib olish */
    SET ay = MONTH(sana);
    SET kun = DAY(sana);
    /* CASE operatori orqali tanlash */
    CASE ay
        WHEN 1 THEN

```



```

IF kun >= 22 THEN SET nomi = "Dalv";
ELSE SET nomi = "Tog' echkisi "; END IF;
WHEN 2 THEN
IF kun >= 22 THEN SET nomi = "Baliq";
ELSE SET nomi = "Dalv"; END IF;
WHEN 3 THEN
IF kun >= 22 THEN SET nomi = "Hamal";
ELSE SET nomi = "Baliq"; END IF;
WHEN 4 THEN
IF kun >= 22 THEN SET nomi = "Savr";
ELSE SET nomi = "Hamal"; END IF;
WHEN 5 THEN
IF kun >= 21 THEN SET nomi = "Ekizaklar";
ELSE SET nomi = "Savr"; END IF;
WHEN 6 THEN
IF kun >= 22 THEN SET nomi = "Qisqichbaqa";
ELSE SET nomi = "Ekizaklar"; END IF;
WHEN 7 THEN
IF kun >= 23 THEN SET nomi = "Arslon";
ELSE SET nomi = "Qisqichbaqa"; END IF;
WHEN 8 THEN
IF kun >= 24 THEN SET nomi = "Parizod";
ELSE SET nomi = "Arslon"; END IF;
WHEN 9 THEN
IF kun >= 24 THEN SET nomi = "Tarozu";
ELSE SET nomi = "Parizod"; END IF;
WHEN 10 THEN
IF kun >= 24 THEN SET nomi = "Chayon";
ELSE SET nomi = "Tarozu"; END IF;
WHEN 11 THEN
IF kun >= 23 THEN SET nomi = "Qavs";
ELSE SET nomi = "Chayon"; END IF;
WHEN 12 THEN
IF kun >= 22 THEN SET nomi = "Tog' echkisi ";
ELSE SET nomi = "Qavs"; END IF;
ELSE SET nomi = "Noaniq";
END CASE;
RETURN nomi;

```

END bes

DELIMITER ;

Saqlangan *burj()* funktsiyani tekshirish:

SELECT burj('01-07-2020') // Natija: → Qisqichbaqa

SELECT burj('12-12-2020') // Natija: → Qavs

Ms 3. Kiritilgan sanadan xozirgi kungacha bo'lgan yil oy va kunni hisoblovchi SP tuzish talab etilsin.

```

CREATE FUNCTION yiloy(sana date)
RETURNS varchar(100) CHARSET utf8

```

BEGIN

```
DECLARE joriy DATE; DECLARE yil, ay, kun int;
/* Joriy sanani aniqlash */
Select current_date() into joriy;
SET ay = 11- MONTH(sana) + MONTH(joriy) - 1;
SET kun = 30 - DAY(sana) + DAY(joriy);
IF kun>=30 THEN SET ay = ay +1; SET kun = 0; END IF;
SET yil = year(joriy)-year(sana);
IF ay>=12 THEN SET yil = yil +1; SET ay = 0; END IF;
RETURN CONCAT(yil, " yil ", ay, " oy ", kun, " kun");
```

END

Saqlangan *yiloy ()* funksiyani tekshirish (joriy sana 02.06.2020):

```
SELECT yiloy ('01-07-1979') // Natija: → 41 yil 11 oy 1 kun
```

```
SELECT yiloy ('05.09.2005') // Natija: → 15 yil 7 oy 27 kun
```

Nazorat savollari

1. SQLda dasturlash deganda nimani tushunasiz?
2. Delimiter vazifasi nimadan iborat?
3. Saqlanuvchi protsedura axamiyati va u qaerda saqlanadi?
4. Saqlanuvchi protsedura turlarini sanang.
5. Saqlanuvchi protsedura tuzilishi nimalardan iborat?
6. Saqlangan protseduralar ustida qanday amallar bajariladi?
7. Saqlanadigan funksiyalar bilan odatiy funksiyalarning farqi?
8. Saqlangan protsedura va funksiyalarning umumiy jixatlari nimada?

6.3. Triggerlar

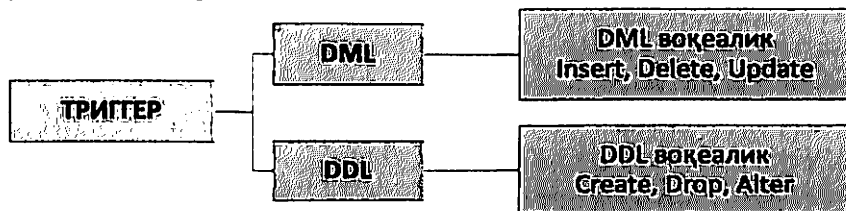
Zamonaviy MBBT serverda saqlanuvchi protseduralar va triggerlar ham ishga tushirilishi mumkin. Saqlovchi protsedura va triggerlar MBBT yadrosi bilan birgalikda MBBT serverini tashkil qiladi. Saqlanuvchi protseduralarga ishchi stantsiyalardagi ilovalardan ham murojaat qilsa bo'ladi. Bu amaliy dastur kodi hajmini kamaytirishga imkon beradi va ishchi stantsiyalardan SQL – operatorlar oqimini kamaytirish imkonini beradi, bu esa kerakli SQL – operatorlar guruhini saqlovchi protseduralarda kodlash mumkin.

Triggerlar saqlanadigan protseduralarning bir turi. Ya'ni uni dasturlashtirish jixatlari bir hisobblanadi. SP chaqirish natijasida bajarilsa, triggerlar MBdagi ob'ektlari va ma'lumotlari bo'yicha qayndaydir voqealik sodir bo'lgan xolatda avtomatik bajariladi. Misol uchun triggerga bog'liq jadval yoki tasavvurlarda o'zgarishlar (o'chirish, yangilash va kiritish) bo'lgan vaqtda trigger avtomatik chaqiriladi. .

Trigger MBBT yadro tomonidan bajariladigan dasturlar bo'lib ular MB jadvallarini o'zgartirishdan (UPDATE, INSERT, DELETE) oldin va keyin MBBT yadrosi tomonida bajariladigan dasturlardir. Triggerning faollashishi vaqti – operator bajarilguncha (BEFORE) yoki keyin (AFTER). Ular MBni butunligini avtomatik ravishda ta'minlab beradi.

MBda ma'lumotlarning ma'lumotlarning yaxlitligini ta'minlash uchun triggerlar qo'llaniladi. Ular quyidagi xususiyatlarga ega:

- foydalanuvchi ruxsat etilgan jadval maydonlariga qiymatlar kiritishini boshqarish imkoniyati;
- Triggerlar bilan bog'liq jadvallardan foydalanadigan ilovalarda triggerlar o'zgarishini kuzatuvchi dasturlarni soddalashtirish;
- Jadval o'zgarishini avtomatik hujjatlashtirish Jadval o'zgarganda har safar bajariladigan triggerlar yordamida qo'shimcha o'zgarishlar jurnalini boshqarish.



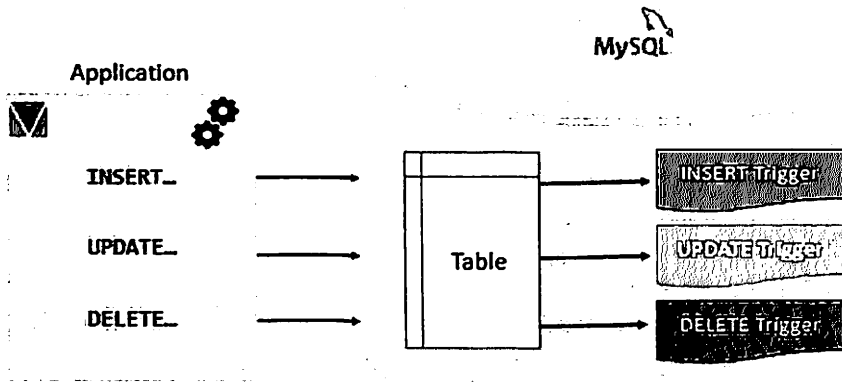
SQL standarti triggerlarning ikki darajasini aniqlaydi: qatorlar darajasi va operatorlar darajasi.

Qatorlar darajasidagi triggerlar har bir qatorda o'zgarish bo'lganda faollashadi. Misol uchun jadvalga 100 ta ma'lumot o'zgartirishi voqealigi

yuz bersa (kiritish, yangilash va o'chirish amallari), trigger ta'sirlangan 100 qator uchun 100 marta faollashadi.

Operatorlar darajasidagi triggerlar esa nechta qatorda o'zgarish bo'lishidan qat'iy nazar, har bir tranzaksiyada faollashadi.

Bu trigger darajalarini qo'llab-quvvatlash MBBTga bog'liq bo'lib, MySQL da faqat qator darajasidagi triggerlar foydalanish mumkin.



Triggerlar MB uchun yashirin qoidalar orqali xavfsizlik va butunlikni ta'minlash kabi ushbu avfzallarga ega:

- Triggerlar MB butunligini tekshirishning boshqa usullarini taqdim etadi;
- Triggerda MB qatlamidagi xatoliklarni qayta ishlaydi;
- Trigger rejalashtirilgan vazifalarni bajarishni alternativ usulini beradi. Ya'ni triggerdan foydalanish natijasida rejalashtirilgan vazifalarni bajarilishini kutish shart emas. Chunki, trigger jadval ma'lumotlarida o'zgarishdan oldin yoki keyin avtomatik bajariladi.
- Trigger jadval ma'lumotlarini o'zgartirishni auditlash uchun foydali bo'lishi mumkin.

Trigger kamchiliklari:

- Triggerlar barcha tekshirishlarni amalga oshira olmasligi, faqat kengaytirilgan tekshirishlarni taqdim etishi mumkin. Oddiy tekshiruvlar uchun NOT NULL, UNIQUE, CHECK i FOREIGN KEY cheklovlaridan foydalanish mumkin.
- Triggerlar muammolarni bartaraf qilishi qiyin bo'lishi mumkin. Chunki u avtomatik ravishda MBda bajarilib, bu mijoz dasturida ko'rinmasligi mumkin.

- Triggerlar MBBT server, jumladan MySQL server harajatlarini o'rtirishi mumkin.

Triggerlar yaratish.

Triggerlar MBning ob'ektlari qatoriga kirganligi uchun ular bilan ishlovchi SQL operatorlar DDL guruxiga mansubdir. Ya'ni triggerlar *CREATE TRIGGER* operatori bilan yaratiladi, *DROP TRIGGER* operatori bilan o'chiriladi va *SHOW TRIGGERS* orqali MB triggerlari tasnifi ko'riladi. SPlar kabi triggerlar ham sarlavha va tana qismidan iborat. Trigger bilan SPning yagona farqi sarlavhada.

Trigger sarlavhasi:

- MBda takrorlanmas bo'lgan trigger nomi;
 - Trigger bilan bog'liq jadval nomi;
 - Trigger qachon ishga tushirilishini aniqlash bo'yicha ko'rsatmalar;
- Trigger tanasi quyidagilardan iborat:
- Lokal o'zgaruvchilar va ma'lumotlar turlari ro'yxatidan (agar ular trigger kodda ishlatilmasa);
 - BEGIN va END kalit so'zlari o'rtasida umumlashtirilgan protsedura va triggerlar tilidagi ko'rsatmalar (kodlar) blok. Blok ichida boshqa ichma-ich bloklar bo'lishi mumkin.

Triggerni yaratish (create trigger) sintaksisi MySQL uchun quyidagicha:

```
CREATE TRIGGER trigger_nomi
{BEFORE | AFTER} {INSERT | UPDATE | DELETE}
ON jadval_nomi FOR EACH ROW
trigger_tanasi;
```

Sintaksis bo'yicha dastlab CREATE TRIGGER kalit so'zidan keyin yaratilmoqchi bo'lgan triggerning noyob nomi ko'rsatiladi. So'ng ro'y beradigan voqealikdan oldin yoki keyin trigger bajarilishini bildiruvchi BEFORE yoki AFTER rejimi belgilanadi. Triggerni faollashishi lozim bo'lgan INSERT, UPDATE yoki DELETE operatorlaridan bittasi tanlanadi. Keyingi qadamda ON kalit so'zidan keyin triggerga tegishli (bog'lanayotgan) jadval nomi ko'rsatiladi. Agar trigger tanasi ko'p amallardan iborat bo'lsa BEGIN END kalit so'zlardan foydalaniladi.

Ma'lumki, DML operatorlari jadval ma'lumotlarini ustunlar kesimida o'zgartiradi. E'tibor berish kerakki, ushbu ustunda oldingi va keyingi qiymat degan tushuncha bor. Misol uchun, UPDATE operatori bilan biror ustun yangilanadigan bo'lsa, bu ustunning oldingi qiymati va

yangilangandan keyinki qiymati bor bo'ladi. Triggerlarda DML operatorlari bajarilishidan oldin (BEFORE) va bajarilgandan keyin (AFTER) ustunlarning qiymatlarini aniqlashda NEW va OLD modifikatoridan foydalaniladi. Triggerlarni NEW va OLD modifikatori mavjudligini jadval ko'rinishda ifodalanishi quyida keltirilgan.

Xodisalar	OLD (oldingi qiymati)	NEW (yangi qiymati)
INSERT	yo'q	bor
UPDATE	bor	bor
DELETE	bor	yo'q

Triggerlarni yaratishga misollar.

Avtor, Categoria va Books jadvalidagi ma'lumotlari o'zgarishini audit qiluvchi trigger yaratish. Shartlari: voqealik sodir bo'lgan jadval nomi, voqealik turi va joriy vaqt bilan birga:

- 1) *Biror yozuv ma'lumoti yangilansa, ushbu yozuv kodi va yangilangan ustunning oldingi va keyingi qiymatlarini qayd qilish;*
- 2) *Ma'lumot o'chirilsa, o'chirilgan yozuv kodi va muallif-kategoriya-kitob nomini qayd qilish;*
- 3) *Ma'lumot qo'shilsa, yangi ma'lumot kodi va muallif-kategoriya-kitob nomini qayd qilish.*

Bu shartlarni to'liq qanoatlandirishda Avtor, Categoria va Books jadvallardagi INSERT, UPDATE va DELETE voqealıkları uchun alohida-alohida $3 \times 3 = 9$ ta triggerlar yaratilishi lozim.

Shuningdek, MBda voqealıklarıni audit qiluvchi bes_audit nomli jadval ham yaratiladi:

```
CREATE TABLE bes_audit (
  id int(11) NOT NULL AUTO_INCREMENT,
  table_name varchar(20) DEFAULT NULL,
  voqea_turi varchar(20) DEFAULT NULL,
  id_row int(11) DEFAULT NULL,
  old_name varchar(255) DEFAULT NULL,
  new_name varchar(255) DEFAULT NULL,
  sana datetime DEFAULT NULL,
  PRIMARY KEY (id)
);
```

Birinchi trigger. Books jadvaliga yangi ma'lumot kiritilishini audit qiluvchi books_insert nomli trigger dastur kodi quyidagicha bo'ladi:

DELIMITER bes

CREATE

DEFINER = 'root'@'%'

TRIGGER books_insert

AFTER INSERT

ON Books

FOR EACH ROW

BEGIN

DECLARE joriy datetime;

SET joriy = **NOW**();

INSERT INTO bes_audit

SET table_name = 'Books', voqea_turi = 'Insert',

old_name = '', new_name = **NEW.name**,

id_row = **NEW.id_book**, sana = joriy;

END bes

DELIMITER ;

Books jadvaliga yangi ma'lumot kiritiladi:

INSERT INTO Books (name, id_cat, page, id_avtor, yil, nashr)

VALUE ('Tanlangan asarlar', 1, 350, 1, 2020, 1);

SELECT * FROM Books **ORDER BY** id_book **DESC LIMIT** 1;

Natija

id_book	name	id_cat	page	id_avtor	yil	nashr
27	Tanlangan asarlar	1	350	1	2020	1

Books jadvaliga yangi ma'lumot kiritish (**INSERT**) voqealik ro'y bergandan so'ng (**AFTER**) shan vaqtda **books_insert** triggerini avtomatik faollashadi va natijada **bes_audit** jadvaliga voqealikni qayd qiladi. Natija **bes_audit** jadvalini ko'rish orqali tekshiriladi:

bes_audit

id	table_name	voqea_turi	id_row	old_name	new_name	sana
1	Books	Insert	27		Tanlangan asarlar	04.06.2020 11:14:15

Ikkinchi trigger. Books jadvalidagi ustunlardagi ma'lumotlar o'zgartirilganligini audit qiluvchi **books_update** trigger. Triggerini shunday yaratish lozimki, unda qaysi ustun ma'lumoti o'zgarsa, ushbu ustunning oldingi va keyingi qiymatlari qayd etilsin. Ma'lumki, Books jadvalida name, id_cat, page, id_avtor, yil, nashr kabi ustunlar mavjud. Trigger Books jadval yangilanishida avtomatik faollashishidan kelib chiqib, Books jadvalidan faqat name, id_cat, id_avtor ustunlarda o'zgarish ro'y bersa, uni **bes_audit** jadvaliga qayd qilishi lozim. Shuningdek, jadvaldagi id_cat va id_avtor ustunlari sonli ma'lumot bo'lib, ular mos holda Avtor va Categoria jadvallariga bog'langan. Trigger **bes_audit** jadvaligi o'zgarish qiymatlarini sonli ko'rinishda emas, balki o'zgartirilayotgan muallifning (kategoriya) oldingi va keyingi ismini qayd qilsin. Ushbu talablarni bajaruvchi **books_update** nomli trigger biroz murakkab bo'lib, uning dastur kodi quyidagicha:

DELIMITER bes

CREATE DEFINER = 'root'@'%'

TRIGGER books_update

BEFORE UPDATE

ON Books

FOR EACH ROW

BEGIN

DECLARE joriy datetime;

DECLARE a1, a2 varchar(100);

SET joriy = NOW();

/ Kitob nomida o'zgarish borligini tekshirish */*

IF OLD.name <> NEW.name THEN

INSERT INTO bes_audit

SET table_name = 'Books', voqea_turi = 'Update (name)',

old_name = OLD.name, new_name = NEW.name,

id_row = OLD.id_book, sana = joriy;

END IF;

/ Avtor kodini o'zgartirilganligini tekshirish */*

IF OLD.id_avtor <> NEW.id_avtor THEN

/ Avtoring oldingi va keyingi ismlarini aniqlash */*

SELECT fio INTO a1

FROM Avtor WHERE id_avtor = OLD.id_avtor;

SELECT fio INTO a2

FROM Avtor WHERE id_avtor = NEW.id_avtor;

INSERT INTO bes_audit

SET table_name = 'Books', voqea_turi = 'Update (avtor)',

old_name = a1, new_name = a2,

id_row = OLD.id_book, sana = joriy;

END IF;

/ Kategoriya kodini o'zgartirilganligini tekshirish */*

IF OLD.id_cat <> NEW.id_cat THEN

Select name_cat INTO a1

From Categoria Where id_cat = OLD.id_cat;

Select name_cat INTO a2

From Categoria Where id_cat = NEW.id_cat;

INSERT INTO bes_audit

SET table_name = 'Books',

voqea_turi = 'Update (Categoria)',

old_name = a1, new_name = a2,

id_row = OLD.id_book, sana = joriy;

END IF;

END bes

DELIMITER ;

Books jadvalidan bir vaqtda name, page, id_avtor, yil ustun ma'lumotlarini yangilanadi.

UPDATE Books

SET name = 'Tanlangan tarixiy romanlar',

page = 1440,

id_avtor = 5,

yil = 2018

WHERE id_book = 27;

SELECT * FROM Books ORDER BY id_book DESC LIMIT 1;

Natija

id_book	name	id_cat	page	id_avtor	yil	nashr
27	Tanlangan tarixiy romanlar	1	1440	5	2018	1

Trigger dasturida ma'lum shartlar ko'rsatilganligi sababli Books jadvalining name, page, id_avtor, yil ustunlarida ro'y bergan yangilanishlardan faqat kitob nomi muallif qiymatlarini bes_audit jadvaliga qayd etadi. Lekin, triggerda ko'rsatilgan bo'lsa ham kategoriyani bes_audit jadvaliga qayd etmaydi. Chunki, bu ustunda yangilanish bo'lmadi. Trigger natijasi bes_audit jadvalida o'z aksini topadi.

bes_audit

id	table_name	voqea turi	id_row	old_name	new_name	sana
1	Books	Insert	27		Tanlangan asarlar	04.06.2020 11:14:15
2	Books	Update (name)	27	Tanlangan asarlar	Tanlangan tarixiy romanlar	04.06.2020 12:46:10
3	Books	Update (avtor)	27	Toxir Malik	Xalq ogzaki ijodi	04.06.2020 12:46:10

Agar *books_update* triggerida shartlar ko'rsatilmaganda yangilanish ro'y bergan ustunni emas, balki barcha ustun ma'lumotlarini takroran bes_audit jadvaliga qayd etgan bo'lar edi.

Uchinchi trigger. Books jadvalidan ma'lumot o'chirilganligini audit qiluvchi *books_delete* trigger. MBda ma'lumotlarni o'chirish xatarli bo'lgan amallardan hisoblanib, keyinchalik o'chirilgan ma'lumotlarni qayta tiklash muammosi paydo bo'lishi mumkin. Shuning uchun ham o'chirilayotgan ma'lumotlarning xech bo'lmaganda bir qismidan nusxa ko'chirib qo'yish muhim.

Aynan hozir yaratiladigan trigger ham xavfsizlik jixatidan ham qayta tiklash ham tarxiy voqealikni saqlash borasidagi funktsiyalarni bajaradi. Trigger Books jadvalidan o'chirilayotgan kitob kodi va nomini bes_audit jadvaliga qayd etadi. Uning dastur kodi quyidagicha:

DELIMITER bes

CREATE

DEFINER = 'root'@'%'

TRIGGER books_delete

AFTER DELETE

ON Books

FOR EACH ROW

BEGIN

DECLARE joriy datetime;

SET joriy = **NOW**();

INSERT INTO bes_audit

SET table_name = 'Books', voqea_turi = 'Delete',

```
old_name = '', new_name = NEW.name,
id_row = NEW.id_book, sana = joriy;
```

```
END bes
DELIMITER ;
```

Endi Books jadvalidan id_book=27 shartni qanoatlandiruvchi qator o'chiriladi.

```
DELETE FROM Books
WHERE id_book = 27
SELECT * FROM Books WHERE id_book=27;
```

Natija

id_book	name	id_cat	page	id_avtor	yil	nashr

bes_audit jadvali chaqirish orqali books_delete trigger natijasi ko'riladi.

bes_audit

id	table_name	voqea turi	id_row	old name	new name	sana
1	Books	Insert	27		Tanlangan asarlar	04.06.2020 11:14:15
2	Books	Update (name)	27	Tanlangan asarlar	Tanlangan tarixiy romanlar	04.06.2020 12:46:10
3	Books	Update (avtor)	27	Toxir Malik	Xalq oqzaki ijodi	04.06.2020 12:46:10
4	Books	Delete	27	Tanlangan tarixiy romanlar		04.06.2020 13:15:40

Keltirilgan misolga xulosa qilinadigan bo'lsak, Books jadvaliga ma'lumot kiritildi, yangilandi va o'chirildi. Natijada esa Books jadvaliga kiritilgan ma'lumot yo'q.

MBga kirgan foydalanuvchi bu amallarni bo'lib o'tganligini bilmaydi. Ammo yaratilgan books_insert, books_update va books_delete nomli triggerlar natijasida bu voqaliklar qayd etildi. Boshqa Avtor va Categoria jadvalariga ham o'zgarishlar kiritilganini qayd qiluvchi triggerlar xuddi yuqoridagi kabi kabi ishlab chiqiladi.


















DROP TRIGGER operatori bilan triggerlarni olib tashlash uchun ishlatilgan. Ushbu operatorning umumiy sintaksisi quyidagicha:

```
DROP TRIGGER trigger_nomi
```

Triggerni yaratishda *NOT FOR REPLICATION* argumenti jadvalni o'zgartirish paytida uni faollashishiga imkon bermaydi.

Triggerlarni eng samarali jixati shundaki, u SP kabi chaqirilmay, balki ko'rsatilgan ob'ekt ustida o'zgarish voqealigi sodir bo'lganda avtomat turda MBBT yadrosida bajariladi.

SQL tili dasturlashtirish xarakteriga ega saqlanuvchi protseduralar, funktsiyalar va triggerlar, shuningdek tasavvurlar MBning ob'ekti sifatida doimiy saqlanib turadi.

Имя	...	Размер	Создано	Движ...	Комментарий	Тип
 burj			2020-06-02 ...			Function
 Rey_func			2020-06-02 ...		Reyting ballni bahoga aylantirish	Function
 yiloy			2020-06-02 ...			Function
 B_kitob			2020-06-02 ...			Procedure
 P_kitob			2020-06-02 ...			Procedure
 Avtor	5	16,0 KiB	2020-03-24 ...	InnoDB		Table
 bes_audit	4	16,0 KiB	2020-06-04 ...	InnoDB		Table
 Books	26	48,0 KiB	2020-03-24 ...	InnoDB		Table
 Categoria	6	16,0 KiB	2020-03-24 ...	InnoDB		Table
 books_delete					AFTER DELETE in table 'Books'	Trigger
 books_insert					AFTER INSERT in table 'Books'	Trigger
 books_update					BEFORE UPDATE in table 'Books'	Trigger
 books2018					VIEW	View
 kitob35					VIEW	View
 kitoblar					VIEW	View
 kitob_2019					VIEW	View
 kitob_yili					VIEW	View

Nazorat savollari

1. Triggerlar ahamiyati, qachon va qerda qo'llaniladi?
2. Triggerlarning saqlanadigan protseduralardan farqi nimada?
3. Triggerlarning turlari darajalari qanday?
4. Triggerlar yaratish tartibi va tarkibi qanday?
5. Triggerlarda oldingi va keyingi qiymat ma'nolarini tushuntiring.

7-BOB. XAVFSIZLIK VA KIRISH TYEXNOLOGIYALARI

7.1. Ma'lumotlar bazasini ishlatish

Ko'p foydalaniluvchi MBlari tashkilot va korxonalarni muxim axborot resurslari xisoblanadi. Shu bilan birga bunday resurslardan foydalanish bir qancha muammolar keltirib chiqaradi. Birinchidan, bunday MBni ishlab chiqish va loyixalash murakkab, chunki bunda ko'p sondagi foydalanuvchilarni talablarini xisobga olish kerak. Shu bilan birga vaqt o'tishi bilan barcha talablar o'zgaradi. Bu esa o'z navbatida ma'lumot baza strukturasi o'zgartirish zaruratiga olib keladi. Bunday strukturalarni o'zgartirish aniq rejalashtirishni va nazorat qilishni talab etadi. Chunki bir gurux foydalanuvchilar uchun bajarilgan o'zgartirish boshqa gurux foydalanuvchilari uchun muammolar tug'dirmasligi talab etiladi. Bundan tashqari bir necha foydalanuvchilarni so'rovlarini parallel qayta ishlashda maxsus tadbirlar qabul qilish kerak.



7.1.1-rasm. MBga ixtiyoriy nuqtadan murojaat qilish

MB tashkilot faoliyatini muxim komponenti bo'lib xizmat qiladi. Uni tarkibi buzunlishi, axborotlarni yo'qolishi kabi nuqsonlar paydo bo'ladi. Shu bilan birga operatsion tizimlardagi takomillashtirishlar va dasturiy ta'minotni yangi versiyalarini paydo bo'lishi MBni takomillashtirishni talab etadi. Bunday masalalarni yechish uchun tashkilot va korxonalarda MBni ma'murlash bo'limlari tuzilgan. Bu bo'limni asosiy vazifasi MB va uni ishlatadigan ilovani yaratishni, ishlatishni MBga xizmat ko'rsatishni amalga oshirishdan iborat. Jumladan:

- MB strukturasi boshqarish;
- Parallel qayta ishlashni boshqarish;
- Qayta ishlash bo'yicha xuquq va vazifalarni(burchlarni) belgilash
- MB xavfsizligini ta'minlash va h.k.

Parallel qayta ishlashni boshqarish. Parallel qayta ishlashni boshqarish bo'yicha asosiy tadbirlar birorta foydalanuvchini boshqasiga kutilmagan xolda ta'sir qilishini hisobga olishga qaratilgan. Parallel qayta ishlash sharoitida ba'zan foydalanuvchilar MBga yakka holda foydalanish sharoitidagi natijalarni olish maqsadiga qaratilgan. Boshqa hollarda foydalanuvchilarning kutilgan tarzda bir biriga ta'siri tushuniladi. Masalan, kompyuter tizimiga o'z ma'lumotlarni kiritgan foydalanuvchi MBdan bir qancha foydalanuvchi foydalanishiga qaramasdan bir xil natija olishi kerak. Shu bilan birga foydalanuvchi masalan omborxonada holati haqidagi axborotni oxirgi vaqtdagi hisobotini olishga harakat qilishi mumkin. Bunda shu paytda boshqa foydalanuvchi ma'lumotlarni o'zgartirayotgan bo'lsa, amalga oshirayotganligini tugallanmagan o'zgarishlarni hisobga olib ko'rishga harakat qiladi. Hozirgi kunda barcha parallel qayta ishlash xollari uchun ma'qul bo'ladigan usullar mavjud emas. Masalan foydalanuvchi parallel qayta ishlashni qattiq boshqarishi mumkin. Bunda u MBni boshqa foydalanuvchilar uchun blokirovka qilib qo'yishi mumkin, o'zini ishi tugamaguncha boshqa mijozlar MB bilan xech qanday ish bajara olmaydi. Bu usul ishonchli bo'lishi bilan birga qimmat hisoblanadi. Ishlab chiqarish samaradorligini oshiradigan, lekin dasturlash va amalga oshirish qiyin usullar xam mavjud. MB bilan ishlaydigan ko'pgina ilovalarda foydalanuvchi ishini tranzaktsiya ko'rinishida amalga oshiriladi. Bir necha SQL operatorlari tranzaktsiya deb nomlanuvchi bitta bloka birlashtiriladi. Tranzaktsiya ba'zan ishni mantiqiy birligi hisoblanadi.

Himoya faqat MB ma'lumotlarigina tegishli emas. Himoyalash hisoblash tizimlarining boshqa qismlarini xam o'z ichiga oladi. Shuning uchun MB ximoyasi hisoblash qurilmalariga, dasturiy ta'minotga va bevosita ma'lumotlarga tegishli bo'lishi mumkin. Ma'lumotlarni samarali himoya qilish uchun mos nazorat vositalari talab etiladi.

MB muxim korporativ resurs hisoblanadi, shuning uchun uni himoya qilish katta ahamiyatga ega. MB kuyidagi xavflar mavjud:

- Ma'lumotlar o'g'irlash va almashtirish
- Ma'lumotlarni maxfiyligini saqlash (sirni oshkor qilmaslik);
- MB butunligini saqlash;
- MBga murojat qilishdan maxrum bo'lish.

MB bo'ladigan xavflarni bartaraf qilish fizik nazoratdan boshlanib, administrativ tashkiliy ishlar bilan tugaydi. Hozirgi kunda kompyuter tizimlarini nazorat qilishni xilma xil vositalari mavjud. Shu bilan birga MB

bilan ishlaydigan MBBTni xavfsizligi xam muxim. Bu esa uzunlik navbvtida MBBT ishlaydigan operatsion tizim xavfsizligi bilan bog'liq

·MBni ximoyalashda quyidagi usullardan foydalanish mumkin:

- Foydalanuvchilarni mualliflashtirish;
- Rezerv nusxalar yaratish va tiklash;
- Bir butunlikni ta'minlash;
- MBga ximoyalash parollarini qo'yish.

Nazorat savollari

1. MBga murojaat qilish nuqtalariga misollar keltiring.
2. Parallel qayta ishlashni boshqarish nimalardan iborat?
3. MBga bo'lgan xavflar qanday?

7.2. Ma'lumotlar bazasini ma'murlash va SQL da xavfsizlik

MBni himoyalash

Kompyuter axborotini himoyalash muammolari bo'yicha 70 – 80 yillarda o'tkazilgan, keyinchalik turli ilovalarda rivojlantirilgan va mos standartlarda qayd etilgan tadqiqotlar *axborot xavfsizligi* tushunchasining tarkibiy elementlari sifatida quyidagilarni belgilaydi:

- *konfidentsiallik* (ruxsatsiz foydalanishdan himoyalash);
- *yaxlitlik*(axborotni ruxsatsiz o'zgartirishdan himoyalash);
- *foydalanuvchanlik* (axborotni va resurslarni ushlab qolinishidan himoyalash, buzilishdan himoyalash, ishga layokatlikni himoyalash).

Axborot xavfsizligi tarkibiy elementlariga mos *tahdidlar* qarshi turadi. Axborot xavfsizligiga tahdid deganda axborot xavfsizligiga bevosita yoki bilvosita zarar yetkazishi mumkin bo'lgan kompyuter tizimiga amalga oshirilgan yoki oshiriluvchi ta'sir tushuniladi. Taxdidlarni axborot xavfsizligini *buzuvchi (bug'unchi)* amalga oshiradi yoki amalga oshirishga urinadi.

Axborot xavfsizligiga u yoki bu tahdidlarni amalga oshirish bo'yicha *buzg'unchi* imkoniyatlari komplekcing formallashtirilgan tavsifi yoki ifodasi *buzg'unchining (niyati buzuvchi) modeli* deb ataladi.

Kompyuter tizimida axborotning himoyalanganligini ta'minlash bo'yicha tashkiliy-texnologik va dasturiy-texnik choralar kompleksining sifatiiy tavsifi *xavfsizlik siyosati* deb ataladi. Xavfsizlik siyosatining formal (matematik, algoritmik, sxemotexnik) ifodasi va ta'rifi *xavfsizlik modeli* deb ataladi.

MB xavfsizligini ta'minlashga taalluqli ba'zi atamalar quyida keltirilgan:

- axborotdan foydalanish (access to infopmation) – axborot bilan tanishish, uni ishlash(xususan, nusxalash), modifikatsiyalash, yo'q qilish;
- foydalanish sub'ekti (access subject)–xarakatlari foydalanishni cheklash qoidalari orqali qat'iy belgilanuvchi shaxs yoki jarayon;
- foydalanish ob'ekti (access object) – avtomatlashtirilgan tizimning axborot birligi bo'lib, undan foydalanish foydalanishning cheklash qoidalari orqali qat'iy belgilanadi;
- foydalanishni cheklash qoidalari (security policy) – sub'ektlarning ob'ektlardan foydalanish xuquqini qat'iy belgilovchi qoidalar majmui;
- ruxsatli foydalanish (authorized access to information) – foydalanishni cheklash qoidalarini buzmasdan axborotdan foydalanish;
- ruxsatsiz foydalanish (unauthorized access to information) – axborotdan foydalanishni cheklash qoidalarini buzib foydalanish;
- foydalanish sub'ekting vakolat darajasi (subject privilege) – foydalanish sub'ekting foydalanish xuquqlari majmui (“imtiyozlar”);
- foydalanishni cheklash qoidalarini buzuvchi (security policy violator)– axborotdan ruxsatsiz foydalanuvchi foydalanish sub'ekti;
- foydalanishni cheklash qoidalarini buzuvchining modeli (security policy violator model) – foydalanishni cheklash qoidalarini buzuvchining abstrakt (formallashgan yoki formallashmagan) tavsifi;
- axborot yaxlitligi (information integrite) – axborot tizimining tasodifiy va(yoki) atain buzish sharoitlarida axborotning o'zgarmasligini ta'minlash qobiliyati;
- konfidentsiallik belgisi (sensitivity label) – ob'ekt konfidentsialligini xarakterlovchi axborot birligi;
- ko'p sathli himoya (multilevel secure) – turli sathli konfidentsiallikga ega ob'ektlardan foydalanishning turli xuquqlariga ega sub'ektlarning foydalanishlarini cheklashni ta'minlovchi himoya.

Kompyuter tizimida ro'yxatga olingan har bir sub'ekt (foydalanuvchi yoki foydalanuvchi nomidan harakatlanuvchi jarayon) bilan uni bir ma'noda indentifikatsiyalovchi axborot bog'liq.

Bu ushbu sub'ektga nom beruvchi son yoki simvollar satri bo'lishi mumkin. Bu axborot sub'ekt identifikatori deb yuritiladi. Agar foydalanuvchi tarmoqda ro'yxatga olingan identifikatorga ega bo'lsa, u legal (qonuniy), aks holda legal bo'lmagan (noqonuniy) foydalanuvchi hisoblanadi. Kompyuter resurslaridan foydalanishdan avval foydalanuvchi kompyuter tizimining identifikatsiya va autentifikatsiya jarayonidan o'tishi lozim.

Identifikatsiya (Identification) - foydalanuvchini uning identifikatori (nomi) bo'yicha aniqlash jarayoni. Bu foydalanuvchi tarmoqdan foydalanishga uringanida birinchi galda bajariladigan funktsiyadir. Foydalanuvchi tizimga uning so'rovi bo'yicha o'zining identifikatorini bildiradi, tizim esa o'zining MBda uning borligini tekshiradi.

Autentifikatsiya (Authentication) – ma'lum qilingan foydalanuvchi, jarayon yoki qurilmaning haqiqiy ekanligini tekshirish muolajasi. Bu tekshirish foydalanuvchi (jarayon yoki qurilma) haqiqatan aynan o'zi ekanligiga ishonch hosil qilishiga imkon beradi. Autentifikatsiya o'tqazishda tekshiruvchi taraf tekshiriluvchi tarafning haqiqiy ekanligiga ishonch hosil qilishi bilan bir qatorda tekshiriluvchi taraf ham axborot almashinuv jarayonida faol qatnashadi. Odatda foydalanuvchi tizimga o'z hususidagi noyob, boshqalarga ma'lum bo'lmagan axborotni (masalan, parol yoki sertifikat) kiritishi orqali identifikatsiyani tasdiqlaydi.

Identifikatsiya va autentifikatsiya sub'ektlarning (foydalanuvchilarning) haqiqiy ekanligini aniqlash va tekshirishning o'zaro bog'langan jarayonidir. Muayyan foydalanuvchi yoki jarayonning tizim resurslaridan foydalanishiga tizimning ruxsati aynan shularga bog'liq. Sub'ektni identifikatsiyalash va autentifikatsiyalashdan so'ng uni avtorizatsiyalash boshlanadi.

Avtorizatsiya (Authorization) – sub'ektga tizimda ma'lum vakolat va resurslarni berish muolajasi, ya'ni avtorizatsiya sub'ekt harakati doirasini va u foydalanadigan resurslarni belgilaydi. Agar tizim avtorizatsiyalangan shaxsni avtorizatsiyalanmagan shaxsdan ishonchli ajrata olmasa bu tizimda axborotning konfidentsialligi va yaxlitligi buzilishi mumkin. Autentifikatsiya va avtorizatsiya muolajalari bilan foydalanuvchi harakatini ma'murlash muolajasi uzviy bog'langan.

Ma'murlash (Accounting) – foydalanuvchining tarmoqdagi harakatini, shu jumladan, uning resurslardan foydalanishga urinishini qayd etish. Ushbu hisobot axboroti xavfsizlik nuqtai nazaridan tarmoqdagi xavfsizlik xodisalarini oshkor qilish, taxlillash va ularga mos reaksiya ko'rsatish uchun juda muhimdir.

Xavfsizlik modellari

Xavfsizlik modeli quyidagilarni o'z ichiga oladi:

- kompyuter (axborot) tizimining modeli;
- axborotning tahdidlardan himoyalanganlik mezonlari, printsiplari, cheklanishlari va maqsad funksiyalari;
- tizimning xavfsiz ishlashining formallashtirilgan qoidalari, cheklanishlari, algoritmlari, sxemalari va mexanizimlari.

Aksariyat xavfsizlik modellari asosida kompyuter tizimlarini sub'ekt-ob'ekt modeli yotadi, xususan, avtomatlashtirilgan axborot tizimlarining yadrosi sifatidagi MB ham. Kompyuter tizimlarining MB MBning sub'ektiga, (mohiyatan aktiv) MBning ob'ektiga (mohiyatan passiv) va sub'ektlar xarakati natijasidagi ob'ektlar ustidagi jarayonlarga ajratiladi (7.2.1-rasm).

Axborot tizimlari ishlashi xavfsizligining ikkita eng muhim printsiplari ma'lum:

- ob'ektga nisbatan barcha sub'ektlar va jarayonlarning identifikatsiyasi va autentifikatsiyasi;
- ob'ektga nisbatan sub'ektlar vakolatlarini cheklash va ma'lumotlar ustidagi har qanday vakolatlarni tekshirish shartligi.



7.2.1-rasm. Ma'lumotlar xavfsizligi modellariidagi kompyuter tizimlarining MB

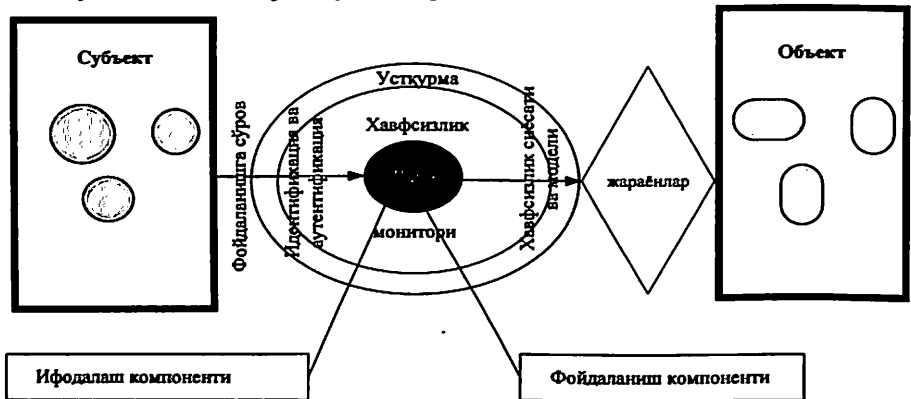
Mos holda MBBT yadrosi strukturasiida ma'lumotlarni ishlashning barcha jarayonlarida belgilangan xavfsizlik siyosatini amalga oshiruvchi, xavfsizlik monitori (serveri, menedjeri, yadrosi) deb ataluvchi qo'shimcha komponent ajratiladi (Trusted Computing Base – TCB). Ushbu komponent ma'lumotlarni ishlashning barcha jarayonlarida xavfsizlikning ma'lum siyosatini amalga oshiradi. Sxematexnika nuqtai nazaridan, kompyuter tizimini ma'lumotlarni ifodalash va ulardan foydalanish

(manipulyatsiyalash) komponentlarini, hamda interfeys va tatbiqiy funksiyalarni amalga oshiruvchi ustqurmani o'z ichiga oluvchi yadro majmui sifatida tasavvur etilsa, xavfsizlik monitoringning roli va o'rini 7.2.2-rasmda keltirilgan sxema orqali izohlash mumkin.

Tor ma'noda kompyuter tizimi monitori amalga oshiruvchi xavfsizlik siyosatining o'zi xavfsizlik modelini aniqlaydi (ikkinchi va uchinchi komponentlar).

Foydalanishni boshqarish xavfsizlik modelining muhim jihati hisoblanadi. Ikkita yondashish mavjud:

- foydalanishni ixtiyoriy boshqarish;
- foydalanishni majburiy boshqarish;









7.2.2-rasm. Kompyuter tizimlarida axborotni himoyalashning sxematik jihati

Xavfsizlik modeli quyidagi bazaviy tasavvurlarga asoslangan: Tizim o'zaro xarakterdagi «sub'ektlar» va «ob'ektlar» majmuasidan iborat. Tizimdagi barcha o'zaro xarakterlar sub'ektlar va ob'ektlar orasida ma'lum hildagi munosabatlarni o'rnatish orqali modellashtiriladi. Barcha amallar o'zaro xarakterdagi yordamida nazoratlanadi va xavfsizlik siyosati qoidalariga muvofiq ma'n etiladi yoki ruxsat beriladi. Xavfsizlik siyosati qoidalar ko'rinishida beriladi, bu qoidalarga mos holda sub'ektlar va ob'ektlar orasida barcha o'zaro xarakterlar amalga oshirilishi shart. Ushbu qoidalarni buzilishiga olib keluvchi o'zaro xarakterlar foydalanishni nazoratlovchi vositalar yordamida to'sib qo'yiladi va amalga oshirishi mumkin emas.

Sub'ektlar, ob'ektlar va ular orasidagi munosabatlar (o'rnatilgan o'zaro xarakter) to'plami tizim «holatini» belgilaydi. Tizimning har bir holati modelda taklif etilgan tizimiga muvofiq xavfsiz yoki taxlikali (xavfli) bo'ladi. Xavfsizlik modelining asosiy elementi – xavfsiz

holatidagi tizim barcha o'rnatilgan qoida va cheklashlarga rioya qilinganda taxlikali (xavfli) holatiga o'tishi mumkin emasligi tasdig'ining isboti.

Xavfsizlikning diskretsiyon modeli. Ma'lumotlar xavfsizligining eng sodda (bir sathli) modeli foydalanishni cheklashning diskretsiyon (tanlash) printsiyiga asosan quriladi. Unga binoan ob'ektdan foydalanish "foydalanish sub'ekti – foydalanish turi – foydalanish ob'ekti" uchlik ko'rinishidagi foydalanishning ruxsat etilgan to'plami asosida amalga oshiriladi. Ushbu model doirasida axborotni ishlash tizimi axborotdan foydalanuvchi sub'ektlar (*S to'plam*), himoyalovchi axborotga ega bo'lgan ob'ektlar (*O to'plam*) va bajarishga vakolatni anglatuvchi foydalanish huquqlarining chekli to'plami *R* majmui ko'rinishida ifodalanadi. Shu bilan birga model ta'siri doirasiga sub'ektlar orasidagi munosabatlarni kiritish uchun barcha sub'ektlar bir vaqtning o'zida ob'ektlar hisoblanadi – $S \subseteq O$.

	O_1 	O_2 	O_3 
S_1 	R W X	R X	R W X
S_2 	R X	R W	W X
S_3 	R	X	W

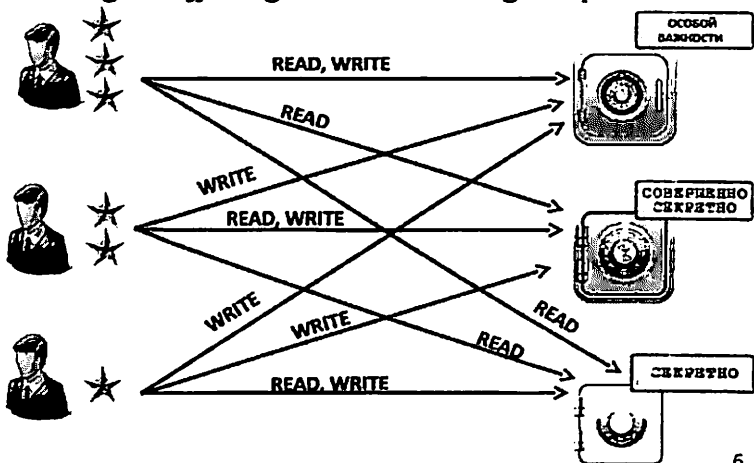
Tizim holati makoni uchun tashkil etuvchi ob'ektlar, sub'ektlar va huquqlar to'plamlarning dekart ko'paytmasi sifatida shakllantiradi. Bu makonda tizimning joriy holati uchlik orqali aniqlanadi. Bu uchlikka sub'ektlar to'plami, ob'ektlar to'plashi va sub'ektlarning ob'ektlardan foydalanish xuquqlarini tavsiflovchi foydalanish xuquqlari matritsasi kiradi – $q=(S,O,M)$. Matritsa qatorlari sub'ektlarga, ustunlari esa ob'ektlarga mos keladi. Ob'ektlar to'plami o'z ichiga sub'ektlar to'plamini olganligi sababli matritsa to'g'ri to'rtburchak ko'rinishida bo'ladi.

Matritsani ixtiyoriy yacheykasi $M[S,O]$ "S" ning "O" dan foydalanish xuquqlari to'plami R ga tegishli foydalanish xuquqiga yig'indisiga ega. Tizimning vaqt bo'yicha ahvoli turli holatlar orasidagi o'tishlar yordamida modellashtiriladi.

Bella LaPadulaning mandatli (muxtor huquqli) modeli.

Foydalanishni boshqarishning mandatli modeli ko'pgina mamlakatlarning davlat va xukumat muassasalarda qabul qilingan manfiy xujjat almashish qoidalariga asoslangan. Bella LaPadula siyosatining asosiy mazmuni amaliy hayotdan olingan bo'lib, himoyaluvchi axborotni ishlashda qatnashuvchilarga xavfsizlik satxi nomini olgan maxsus belgi, masalan, "maxfiy" mutlaqo maxfiy va x.k. kabilarni tayinlashdan iborat. Xavfsizlikning barcha satxlari o'rnatilgan ustunlik munosabati asosida tartiblanadi, masalan, "mutlaqo maxfiy satxi" maxfiy satxidan yuqori yoki undan ustun turadi. Foydalanishni nazoratlash o'zaro xarakatdagi tomonlarning xavfsizlik satxlariga bog'liq holda quyidagi ikkita oddiy qoidalar asosida amalga oshiriladi:

- Vakolatli shaxs (sub'ekt) faqat xavfsizlik satxi o'zining xavfsizlik satxidan yuqori bo'lmagan xujjatlardan axborotni o'qishga haqli;
- Vakolatli shaxs (sub'ekt) xavfsizlik satxi o'zining xavfsizlik satxidan past bo'lmagan xujjatlarga axborot kiritishga xaqli.



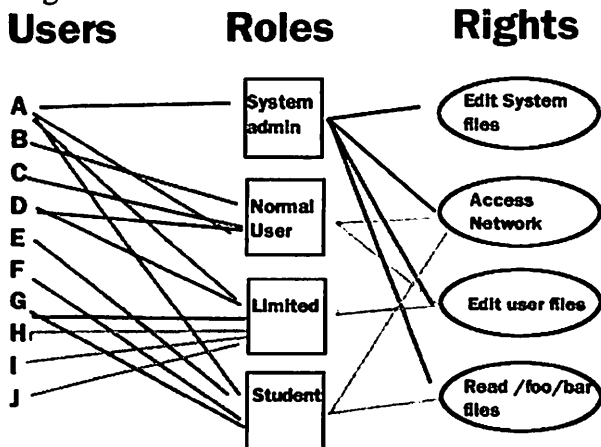
Birinchi qoida yuqori sathli shaxslar tomondan ishlanadigan axborotdan past satxli shaxslar tomonidan foydalanishdan himoyalashni ta'minlaydi. Ikkinchi qoida (juda muhim qoida) axborotni ishlash jarayonning yuqori satxli ishtirokchilariga axborotning sirqib chiqishini (bilib yoki bilmasdan) bartaraf etadi.

Xavfsizlikning Bella LaPadulla modelida tizim Xarrison – Ruzzo – Ulman modeliga o'xshash sub'ektlar S , ob'ektlar O va foydalanish huquqlari to'plami ko'rinishida ifodalanadi. Ob'ektlar to'plami ko'rinishida ifodalanadi. Ob'ektlar to'plami sub'ektlar to'plamini o'z ichiga oladi $S \subseteq O$ va foydalanishning faqat ikkita xili read (o'qish) write

(yozish) ko'riladi. Ammo ushbu model qo'shimcha huquqlarni (masalan, axborotni qo'shish, programmani bajarish va x.k.) kiritish bilan kengaytirilishi mumkin bo'lsada, ular bazaviy (o'qish va yozish) huquqlar orqali akslantiriladi. Foydalanishni moslanuvchan boshqarishni ta'minlashga imkon bermaydigan bunday qat'iy yondashishning ishlatilishi mandatli modelda sub'ektning ob'ekt ustida bajariladigan amal nazoratlanmasligi, balki axborot oqimi nazoratlanishi bilan izohlanadi. Axborot oqimi faqat ikki xil bo'lishi mumkin: sub'ektdan ob'ektga (yozish), yoki ob'ektdan sub'ektga (o'qish).

Xavfsizlikning rolli modeli. Rolli model xavfsizlikning, bu siyosat diskretion modelga xos foydalanishni boshqarishdagi moslanuvchanlik bilan mandatli modelga xos foydalanishni nazoratlash qoidalarining qat'iyligi orasidagi murosaga asoslangan.

Rolli modelda "sub'ekt" tushunchasi "foydalanuvchi" va "rol" tushunchalari bilan almashtiriladi. Foydalanuvchi – tizim bilan ishlovchi va ma'lum xizmat vazifalarini bajaruvchi odam. Rol tizimda faol ishtirok etuvchi obstrakt tushunchasi bo'lib, u bilan ma'lum faolligini amalga oshirish uchun zarur vakolatlarining chegaralangan, mantiqiy bog'liq yig'indi bog'langan.



Rolli siyosat ishlatilganida foydalanishni boshqarish ikki bosqichda amalga oshiriladi: birinchi bosqichda har bir rol uchun ob'ektdan foydalanish huquqlari naboridan iborat vakolatlar nabori ko'rsatiladi, ikkinchi bosqichda har bir foydalanuvchiga uning qo'lidan keladigan rollar ro'yxati tayinlanadi. Rollarga vakolatlar eng kichik imtiyoz printsipida tayinlanadi, ya'ni har bir foydalanuvchi o'zining ishini bajarish uchun faqat minimal zarur vakolatlar naboriga ega bo'lishi shart.

Foydalanishni boshqarish

Foydalanishni boshqarish bilan bog'liq masalalar uchun *INGRES* MBBT ishlatiladi. Odatda MBBTda foydalanishni ixtiyoriy boshqarish qo'llaniladi. Bunda ob'ektdan foydalanish xuquqi ko'pincha imtiyoz deb yuritiladi.

Odatda MBBTda foydalanuvchilarni identifikatsiyalash va ularni haqiqiylikini tekshirish uchun operatsion tizimning mos mexanizmlari, yoki SQL-CONNECT operatori qo'llaniladi. Masalan, *ORACLE* MBBT xolida CONNECT operatori quyidagi ko'rinishga ega bo'ladi:
CONNECT foydalanuvchi [parol] [@ma'lumotlar_bazasi].

Har xolda, MB serveri bilan ishlash seansining boshlanishi onida foydalanuvchi o'zining nomi bilan identifikatsiyalanadi, autentifikatsiya vositasi sifatida esa parol ishlatiladi. Ushbu jarayonning tafsilotlari ilovaning mijoz qismining amalga oshirilishi orqali aniqlanadi.

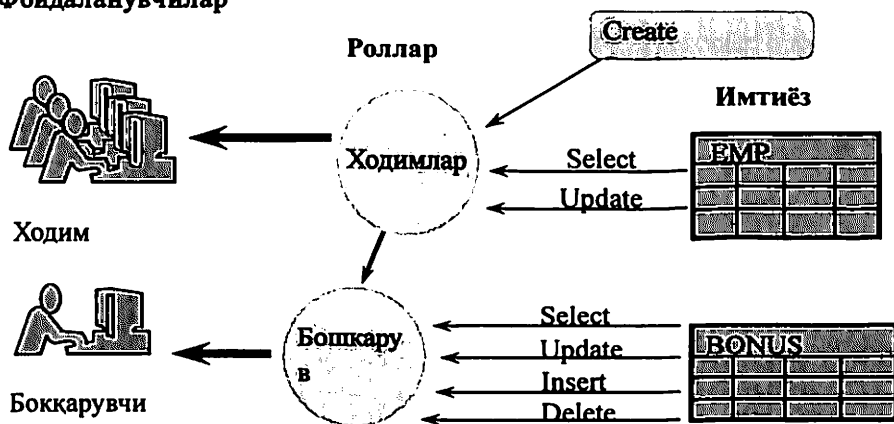
IMTIYOZLAR (PRIVILEGE). Har bir foydalanuvchi MBda nima qilish mumkinligini ko'rsatuvchi imtiyozlarga egadir. Bu imtiyozlar vaqt o'tishi bilan o'zgarishi, ya'ni, eskilari o'chirilib, yangilari qo'shilishi mumkin. Imtiyozlar bu ob'ektlar bilan ishlash huquqini ko'rsatadi. Ob'ekt imtiyozlari bir vaqtning o'zida foydalanuvchilar va jadvallar (tasavvurlar) bilan bog'liq. Ya'ni imtiyoz ma'lum foydalanuvchiga ko'rsatilgan jadvalda, asosiy jadvalda yoki tasavvurda beriladi. Ixtiyoriy turdagi jadvalni yaratgan foydalanuvchi shu jadval egasidir. Bu shuni bildiradiki foydalanuvchi bu jadvalda hamma imtiyozlarga ega va imtiyozlarini shu jadvalning boshqa foydalanuvchilariga uzatishi mumkin.

Imtiyozlar sub'ektlarga (alohida foydalanuvchilarga), guruhlariga, rollarga yoki barcha foydalanuvchilarga ob'ektlardan foydalanish uchun berilishi mumkin.

ROL (ROLE) – imtiyozlar to'plami. Odatda MB ko'p foydalanuvchili bo'lib, ularning ko'pchiligi MBda bir xil imtiyozlarga ega bo'ladi. Foydalanuvchilarga imtiyozlar berish va bekorlash uchun har biriga alohida ravishda olib borish administrator uchun ortiqcha ish va vaqt talab qilishi mumkin. Shuning uchun mazkur muammoni soddalashtirish maqsadida MBBTlarning (ms: MySQL, Oracle) ko'pchiligida rol deb nomlanuvchi yangi ob'ekt taqdim etilgan. Bunda xuddi foydalanuvchilarni ro'yxatga olish singari, endi rollarda imtiyozlar berish va bekorlash mumkin. Rollar imtiyozlari foydalanuvchilar va guruhlar imtiyozlaridan ustun turadi. Bunday rollar imtiyozlarni strukturalash vositasi sifatida xizmat qiladi va ularning modifikatsiyalanishini osonlashtiradi.

FOYDALANUVCHI (PUBLIC) – MBBTga kirgan shaxlar majmuasi. Foydalanuvchiga imtiyozlar berilishi – foydalanishning ko'zda tutilgan xuquqlarini berishning qulay usuli. Turli foydalanuvchilar zimmasiga turli MBni ma'murlashni yuklash ma'noga ega bo'ladi, qachonki ushbu bazalar mustaqil va ularga nisbatan imtiyozlarni ajratishning kelishilgan siyosatini yoki rezervli nusxalashni o'tkazishga to'g'ri kelmasa. Bu xolda har bir ma'mur qancha zarur bo'lsa, shuncha biladi. Bir tomondan, INGRES foydalanuvchisi va ma'lumot bazasi, ikkinchi tomondan operatsion tizim superfoydalanuvchi (OS UNIX xolida root) va xizmatchi foydalanuvchilar (OS UNIXda bu bin, Ip, IJUCP va h. bo'lishi mumkin) orasidagi o'xshashlikni kuzatish mumkin. Xizmatchi foydalanuvchilarning kiritilishi superfoydalanuvchi imtiyozlarini olmasdan funktsional qismtizimlarni ma'murlashga imkon beradi. Xuddi shu tarzda serverda saqlanuvchi ma'lumotlarni bo'lmalarga ajratish mumkin. Bitta bo'lma ma'murining obro'sizlantirilishi albatta boshqa bo'lma ma'murining obro'sizlantirilishi degani emas.

Фойдаланувчелар



Имтиёз турлари. MBBTda imtiyozlarni ikkita kategoriyaga ajratish mumkin: xavfsizlik imtiyozlari va foydalanish imtiyozlari.

Xavfsizlik imtiyozlari doim muayyan foydalanuvchiga uning yaratilishi (CREATE USER operatori yordamida) yoki xarakteristikalarini o'zgartirish (ALTER USER operatori yordamida) vaqtida ajratiladi. Xavfsizlik imtiyozlari ma'muriy harakatlar bajarishga imkon beradi. Bunday imtiyozlar beshta:

✓ **SECURITY** – MBBT xavfsizligini boshqarish va foydalanuvchi harakatlarini kuzatish xuquqi. Foydalanuvchi ushbu imtiyoz bilan har qanday MBga ulanishi, foydalanuvchilar, guruhlar va rollar

xarakteristikalarini yo'q qilishi va o'zgartirishi, MBdan foydalanish xuquqini boshqa foydalanuvchiga berishi, qayd qilinuvchi axborotning yozilishini boshqarishi, boshqa foydalanuvchilar so'rovini kuzatishi va, nihoyat, boshqa foydalanuvchilar nomidan INGRES-komandalarni ishga tushirishi mumkin. Security imtiyozi MB serverining ma'muriga, hamda axborot xavfsizligiga shaxsan javobgar shaxsga zarur. Ushbu imtiyozni boshqa foydalanuvchilarga berish (masalan, MB ma'muri tomonidan) MB serverining himoyasidagi bo'lishi mumkin bo'lgan zaif joylarni ko'paytiradi;

✓ *CREATEDB* – MB yaratish va yo'q qilish xuquqi. Ushbu imtiyozga server ma'muridan tashqari foydalanuvchilar ega bo'lishlari lozim. Foydalanuvchilar ixtiyoriga alohida MBning ma'murlari roli taqdim etiladi;

✓ *OPERATOR* – odatda operator ixtiyoridagi harakatlarni bajarish xuquqi. Serverni ishga tushirish va to'xtatish, axborotni saqlash va tiklash ko'zda tutiladi. Ushbu imtiyozni server va MB ma'muridan tashqari operatsion tizim ma'muriga ham berish maqsadga muvofiq hisoblanadi;

✓ *MAINTAIN LOCATIONS* – MB serveri ma'murining bazasi va operatsion tizim o'r mashgan joyni boshqarish xuquqi;

✓ *TRACE* – sozlovchi trassirovka flaglari xolatlarini o'zgartirish xuquqi. Ushbu imtiyoz murakkab, tushunarsiz vaziyatlarni taxlillashda MB serveri ma'muriga va boshqa tajribali foydalanuvchilarga foydali.

Foydalanish imtiyozlari, nomiga muvofiq, sub'ektlarning ma'lum ob'ektlardan foydalanish huquqini belgilaydi va foydalanuvchilarga, guruhlariga, rollarga yoki barchaga *GRANT* operatori yordamida ajratiladi va *REVOKE* operatori yordamida olib quyiladi. Ushbu imtiyozlar, odatda, *mos ob'ekt egasi* (MB ma'muri) yoki *security* imtiyoziga ega shaxs (odatda MB serveri) tomonidan beriladi. Grant va revoke operatorlari SQL operatorlarining *DCL* – ma'lumotlarni boshqarish guruhiga mansub.

Guruhlariga va rollarga imtiyozlarni berishdan oldin ular *CREATE GROUP* va *CREATE ROLE* operatorlari yordamida yaratish lozim.

Guruh tarkibini o'zgartirish va rollardagi parollarini o'zgartirishga uchun *ALTER GROUP* operatori xizmat qiladi. *DROP GROUP* operatori guruhlarni va rollarni yo'q qilishga imkon beradi (faqat guruh va rol a'zolari ro'yxati yo'q qilinganidan so'ng).

Foydalanish imtiyozlarni ular taalluqli ob'ektlar turi bo'yicha ajratish mumkin. INGRES MBBTda bunday turlar beshta: *jadvallar va tasavvurlar; muolajalar; MB; MB serveri; hodisalar.*

MBBTga ro'yxatdan o'tgan foydalanuvchi dastlab jadvallardan va tasavvurlardan foydalanishning hech qanday xuquqiga ega emas. Bu xuquqlarni GRANT operatorlari yordamida berish mumkin.

Foydalanish imtiyozlarini berish GRANT operatori yordamida amalga oshirilib, uning umumiy format ko'rinishi quyidagicha:

GRANT imtiyoz(lar);

ON ob'ekt(lar);

TO kim(lar)ga.

Jadvallar va tasavvurlarga muvofiq quyidagi foydalanish imtiyoz(xuquq)larini boshqarish mumkin:

- *SELECT* - ma'lumotlarni tanlash xuquqi (ma'lum ustunlarni ko'rsatish mumkin);
- *INSERT* - ma'lumotlarni qo'shish xuquqi;
- *DELETE* - ma'lumotlarni yo'q qilish xuquqi;
- *UPDATE* - ma'lumotlarni yangilash xuquqi (yangilanishga ruxsat bo'lgan ma'lum ustunlarni ko'rsatish mumkin);
- *REFERENCES* - berilgan jadvalga (ma'lum ustunlarni ko'rsatish mumkin) havola qiluvchi tashqi kalitlardan foydalanish xuquqi;
- *INDEX* – jadvalda indeks yaratish huquqi;
- *SYNONYM* -ob'ekt uchun sinonim yaratish huquqi;
- *ALTER* - jadvalda ALTER TABLE komandasini bajarish xuquqi.

Muolajalarga nisbatan bajarish xuquqi berilishi mumkin. Bunda muolajalar ishlov beruvchi ob'ektlardan foydalanish xuquqlarining ajratilishi xususida o'ylash kerak emas, ularning mavjudligi shart emas. Shunday qilib, MB muolajalari ma'lumotlar ustida qat'iy belgilangan harakatlarni bajarish uchun nazoratli foydalanishni taqdim etishning qulay vositasi hisoblanadi.

MBdan foydalanish xuquqlarini uning *ma'muri* yoki *security* imtiyoziga ega foydalanuvchi taqdim etishi mumkin. Ushbu "xuquqlar" aslida MBdan foydalanishga qator cheklashlar o'rnatadi, ya'ni mohiyatan taqiqlovchi hisoblanadi. Kiritish/chiqarish amallar soniga yoki bitta so'rov bilan qaytariluvchi qator soniga cheklash, jadvallar va muolajalar va h. yaratish xuquqiga cheklash ko'zda tutiladi. Odatda foydalanuvchi miqdoriy limitlar bilan qoniqmaydi va bazada ob'ektlar yaratish xuquqini oladi.

Ta'kidlash lozimki, MBni yaratishda uning maqomi (umumiy yoki shaxsiy) ko'rsatiladi. Bu bazadan nazarda tutilgan foydalanish xuquqiga ta'sir etadi. Odatda umumiy bazaga ulanish xuquqi barchaga beriladi. Shaxsiy bazaga ulanish xuquqi oshkora ravishda berilishi lozim. Ulanishga

xuquq baza va undagi ob'ektlar bilan boshqa barcha amallarni bajarish uchun kerak.

QUERY_IO_LIMIT va QUERY_ROW_LIMIT imtiyozlar (bu holda cheklashlar deb atash to'g'riroq bo'lar edi) so'rovlar optimizatori bergan baho asosida tekshiriladi. Agar optimizator oldindan so'rov kiritish/chiqarish amaliga yoki qaytariluvchi qatorga ajratilgan limit sonidan oshganini aytsa, so'rov rad etiladi. Bu xildagi miqdoriy cheklashlarning qo'yilishi serverning bitta mijoz tomonidan monopoliya qilinishiga to'sqinlik qiladi va yuqori tayyorlikni madadlash instrumentining biri sifatida ishlatilishi mumkin.

Foydalanishni boshqarishda tasavvurlardan foydalanish. MBBT foydalanishni boshqaruvchi o'ziga xos vosita – tasavvurlarni taqdim etadi. Tasavvurlar sub'ektlar uchun bazaviy jadvallarning ma'lum qatorlarining ko'rinarli bo'lishiga (proektsiyani amalga oshirishga) yoki ma'lum qatorlarni tanlashga (selektsiyani amalga oshirishga) imkon beradi. MBning ma'muri sub'ektlarga bazaviy jadvallardan foydalanish xuquqini bermasdan va munosib tasavvurlarni tuzib jadvallarni ruxsatsiz foydalanishdan himoyalaydi va har bir foydalanuvchini o'zining MBga qarashi bilan ta'minlaydi.

Foydalanish xuquqlarining ierarxiyasi. GRANT operatori va MBBTdan foydalanishni boshqaruvchi boshqa vositalar foydalanishning quyidagi cheklashlar turini amalga oshirishga imkon beradi:

- amaliy cheklashlar (jadvalning barcha yoki faqat ba'zi ustunlariga qo'llaniluvchi select, insert, update, delete xuquqlari hisobiga);
- muhimligi bo'yicha cheklashlar (tasavvurlar mexanizmi hisobiga);
- resurslarga cheklashlar (MBdan foydalanish imtiyozlari bo'yicha).

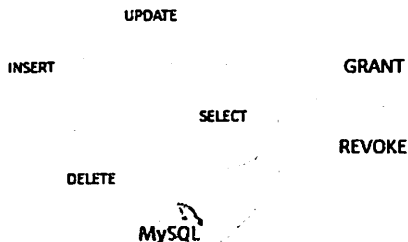
So'rovlarni ishlashda MBBT avval ob'ektlardan foydalanish xuquqini tekshiradi. Agar amaliy cheklashlar buzilgan bo'lsa so'rov, mos tashxis chiqarilib, rad etiladi. Muhimligi bo'yicha cheklashlarning buzilishi faqat natijaviy qatorlarning soniga ta'sir etadi; bunda hech qanday tashhis chiqarilmaydi. Nihoyat, oldingi ikkita cheklashlar hisobga olinganidan so'ng, so'rov ishlanish uchun optimizatorga beriladi. Agar optimizator resurslarga cheklashlar oshirilganini aniqlasa, so'rov mos tashhis chiqarilib, rad etiladi.

Imtiyozlar ierarxiyasiga boshqa nuqtai nazardan qarash mumkin. Har bir foydalanuvchi, o'zining tashqari, PUBLIC imtiyoziga ega. Undan tashqari, u turli guruhlarda bo'lib, ma'lum rollar bilan ilovalarni ishga tushirishi mumkin. Quyida imtiyozlarning nomlangan turli eltuvchilari taqdim etgan xuquqlarning o'zaro munosabati xususida so'z ketadi.

INGRES MBBT uchun avtorizatsiya ierarxiyasi ushbu ko'rinishga ega:

- rol (yuqori ustuvorlik);
- foydalanuvchi;
- guruh;
- PUBLIC (past ustuvorlik).

Foydalanuvchi, imtiyoz va rollar bilan ishlash operatorlari



Foydalanuvchilarni yaratish va o'chirish.

1. Foydalanuvchilarni yaratish

SQL standartida foydalanuvchilarni yaratish **CREATE USER** (ayrim MBBTlarda **GRANT CONNECT**) operatori orqali amalga oshiriladi. Bu operatorning sintaksisi turli MBBTda (DB2, Oracle, MySQL, PostgreSQL, Microsoft SQL Server) bir-biridan farq qiladi. Lekin barchasida bitta umumiylik mavjud.

Misol uchun MySQLda foydalanuvchilarni yaratish sinaksisi:

```
CREATE USER [IF NOT EXISTS] account_name  
IDENTIFIED BY 'password';
```

Bunda `account_name` ikki qismdan, ya'ni foydalanuvchi nomi (lotin xarf va raqamlar) va xost nomidan iborat bo'lib ular `@` simvoli orqali ajratiladi (`username@hostname`). Bu yerda xost nomi majburiy bo'lmasdan, u yozilmasa, ixtiyoriy xostga ulanish mumkin.

Foydalanuvchilarga yangi parollarini o'rnatish sintaksisi:

```
GRANT CONNECT TO username IDENTIFIED BY password
```

Ammo, mavjud parollarini o'zgartirish MBBTlar uchun turlicha:

DB2:

```
CONNECT TO DatabaseName USER userid  
USING password NEW new_password  
CONFIRM confirm_password
```

Oracle:

ALTER USER username IDENTIFIED BY password

MySQL:

SET PASSWORD FOR user = PASSWORD('some password') |

OLD_PASSWORD('some password') | 'encrypted password'

PostgreSQL:

ALTER USER username WITH PASSWORD password

SQL Server:

ALTER LOGIN loginName WITH PASSWORD = password

Parolni maxsus imtiyozlarga ega foydalanuvchi boshqalarnikini va o'zi kirgan foydalanuvchi esa faqat o'zini parolini o'zgartira oladi

2. Rol yaratish va o'chirish

Rollarni yaratish **CREATE ROLE** operatori orqali amalga oshirilib, uning sintaksis quyidagicha:

CREATE ROLE role_name [WITH ADMIN {CURRENT_USER | CURRENT_ROLE}]

Rol nomlar ikki qismdan iborat: nomi va xot nomi.

role_name@host_name

Agar rolda host_name kiritilmasa, u o'rnatilgan “%” qiymatini olib, u har qanday xost bo'lishi mumkinligini anglatadi.

Bitta yoki bir nechta rolni o'chirish uchun **DROP ROLE** operatori ishlatiladi:

DROP ROLE role_name[, role_name, ...];

Imtiyozlar berish (GRANT)

GRANT operatori sintaksisi:

GRANT privilege[(col1,...)] [,privilege]..

ON privilege_level

TO user_name [, | role_name];

Bu yerda privilege – tayinlanayotgan imtiyozlar ro'yxati, privilege_level - jadval va/yoki tasavvur(lar) nomi, user_name [, | role_name] – imtiyozlar olgan foydalanuvchilar yoki rollar ro'yxati.

Masalan:

GRANT SELECT, INSERT ON table_student, table_baxo TO Diane, Kurator;

Ma'lum ustunlarga imtiyozlarni cheklanish. Bu cheklanish **UPDATE** va **REFERENCES** imtiyozlarida ishlatilishi mumkin. Bu xolda imtiyoz ko'rsatilgandan so'ng qavs ichida shu imtiyoz qo'llaniluvchi ustunlar ko'rsatiladi (agar ustunlar ko'rsatilmagan bo'lsa, imtiyoz butun jadvalga ta'sir o'tkazadi). Masalan: Diane ga Salepeople jadvalining City va Comm ustunlari qiymatlarini o'zgartirish huquqini berish:

GRANT UPDATE (City, Comm) ON Salespeople TO Diane;

yoki Stephen ga Cnum va CName ustunlarini o'zining jadvallaridagi

ixtiyoriy tashqi kalitlarga nisbatan ajdod kalit sifatida ishlatish huquqini berish:

GRANT REFERENCES (CName, CNum) ON Customers TO Stephen;

bunda Stephen (CName, CNum) yoki (CNum, CName) usutunlarni, jadvalarining ikki ustuni bilan tashqi kalit yordamida mos kelgan ikki – ustunli ajdod kalit sifatida aniqlashi mumkin. Yoki u maydonga individual murojaat qilish uchun ajratilgan tashqi kalitlar yaratishi mumkin.

ALL va PUBLIC argumentlaridan foydalanish. ALL jadvalda hamma imtiyozlarni berish uchun ishlatiladi. Masalan:

GRANT ALL ON Customers TO Stephen;

Agar siz imtiyozlarni publikatsiya (PUBLIC) uchun uzatsangiz, hamma foydalanuvchilar avtomatik ravishda ularni qabul qiladi. Odatda bu ma'lum asos jadvallarda yoki tasavvurlarda (VIEW) imtiyozi uchun qo'llanadi. Ixtiyoriy foydalanuvchiga Buyurtmalar jadvalini ko'rish imkonini berish uchun, siz quyidagini kiritishingiz mumkin:

GRANT SELECT ON Orders TO PUBLIC;

WITH GRANT OPTIONS yordamida imtiyozlar berish. Ba'zida jadval yaratuvchisiga boshqa foydalanuvchilar uning jadvalitda imtiyozlarni uzatish imkoniga ega bo'lishlari kerak. Odatda bu bir yoki bir necha xodimlar bir necha yoki hamma asos jadvallarni yaratib, ularni shu jadvallar bilan ishlaydigan xodimlarga topshiradigan tizimlarda zarurdir. SQL da buning uchun WITH GRANT OPTION ifodasidan foydalaniladi.

Imtiyozlar boshqa turlari (tizim imtiyozlari). Ma'lumotlar maxsus ob'ektlari terminlaridaaniqlanmaydigan imtiyozlar tizim imtiyozlari yoki ma'lumotlar bazalari qoidalari deb ataladi. Umumiy yondoshishda uchta asosiy tizim imtiyozlari mavjud:

- **CONNECT** (Ulash),
- **RESOURCE** (Resurs),
- **DBA** (MB Administratori).

Soddaroq qilib aytish mumkinki **CONNECT** agar ob'ekt imtiyozlari uzatilgan bo'lsa registratsiya qilinish, tasavvurlar va sinonimlar yaratish huquqidan iborat. **RESOURCE** asos jadvallari yaratish huquqidan iborat. **DBA** bu **MB**da foydalanuvchiga eng yuqori imkoniyatlar beruvchi super foydalanuvchi imtiyozidir. **MB** administratori funksiyasiga ega bir yoki bir necha foydalanuvchi shu imtiyozga ega bo'lishi mumkin. Faqat **DBA** identifikatorli foydalanuvchi **CONNECT**, **RESOURCE** va **DBA** imtiyozlarini berishi mumkin.

Foydalanuvchiga resurs yoki administrator imtiyozini berish uchun quyidagi komandalarni bajarish yetarli

GRANT RESOURCE TO userid;

yoki mos ravishda
GRANT DBA TO userid.

SQL Central da RESOURCE va DBA imtiyozlarini berish uchun Users&Groupe papkasida foydalanuvchi xossalarini chaqirib, Authoritnes qo'shimcha sahifasida mos imtiyoz yoniga belgi qo'yish lozim.

Imtiyozlarni bekor qilish (REVOKE)

Oldin berilgan imtiyozlarni (ruxsat beruvchi va taqiqlovchi) foydalanuvchi yoki rollardan bekor qilishda *REVOKE* operatori xizmat qiladi. Imtiyozlarni *REVOKE* operatori yordamida rad etish mumkin, uning sintaksisi GRANT ga o'xshash, lekin teskari ta'sirga ega.

REVOKE operatori sintaksisi:

```
REVOKE privilege[(col1,...)] [,privilege],...  
ON privilege_level  
TO user_name [, | role_name];
```

Bu yerda esa privilege – bekor qilinayotgan imtiyozlar ro'yxati.

Masalan Adrian va Stephen uchun sotib oluvchilar jadvalida INSERT va DELETE imtiyozlarini rad etish uchun quyidagi komandadan foydalanish lozim:

```
REVOKE INSERT, DELETE ON Customers FROM Adrian, Stephen;
```

Imtiyozlarni rad etishda quyidagi qoidalarga rioya qilinadi: imtiyozlar ularni bergan foydalanuvchi tomonidan rad etiladi va rad etish kasakadlanadi, Ya'ni undan shu imtiyozlarni olgan hamma foydalanuvchilarga tarqaladi.

Imtiyozlarni bekorlashda shunga axamiyat berish lozimki, agar rol yoki foydalanuvchiga ko'plagan imtiyozlar berilgan bo'lsa, ulardan qisman imtiyozlarni *REVOKE* bilan bekorlash mumkin.

Amaliy masala

Xuddi foydalanuvchilarni ro'yxatga olish singari, endi rollarda imtiyozlar berish va bekorlash mumkin.

Ko'p foydalanuvchiga bir xil imtiyozlar berish quyidagi amallar bajarilishi lozim:

1. Yangi rol yaratiladi
2. Rolga imtiyoz beriladi
3. Foydalanuvchilarga rol beriladi

Agar foydalanuvchining imtiyozlari o'zgartirilishi zarur bo'lsa, u holda faqat unga berilgan roldagi imtiyozlar o'zgartiriladi. Bunda ushbu o'zgartirilgan roldagi foydalanuvchilarning barchasini imtiyozlari avtomatik o'zgaradi.

MySQLda rollarga misol.

CRM nomli MBda customer nomli jadval berilgan bo'lsin.

id	first_name	last_name	phone	email
1	John	Doe	(408)-987-7654	john.doe@mysqftutorial.org
2	Lily	Bush	(408)-987-7985	Lily.bush@mysqftutorial.org

1. Rol yaratish

Berilgan MBdan to'liq foydalanish, faqat o'qish va faqat yozish/o'chirish/yangilash huquqiga ega imtiyozlarni beruvchi rollarni yaratish talab etilsin.

```
CREATE ROLE crm_dev, crm_read, crm_write;
```

2. Rolga imtiyoz berish

Endi Yuqorida yaratilgan rollarga imtiyozlar beriladi.

A) crm_dev nomli rolga barcha imtiyozlarni berish

```
GRANT ALL ON crm.* TO crm_dev;
```

V) crm_read nomli rolga faqat o'qish (SELECT) imtiyozini berish:

```
GRANT SELECT ON crm.* TO crm_read;
```

S) crm_write role nomli rolga insert, update va delete imtiyozini berish:

```
GRANT INSERT, UPDATE, DELETE ON crm.* TO crm_write;
```

3. Foydalanuvchilarni yaratish.

Yangi foydalanuvchilarni yaratishda CREATE USER operatori foydalaniladi:

– dasturchi imtiyoziga ega foydalanuvchi

```
CREATE USER crm_dev1@localhost IDENTIFIED BY 'Secure$1782';
```

– o'qish imtiyoziga ega foydalanuvchi

```
CREATE USER crm_read1@localhost IDENTIFIED BY 'Secure$5432';
```

– o'qish/yozish imtiyoziga ega foydalanuvchi

```
CREATE USER crm_write1@localhost IDENTIFIED BY 'Secure$9075';
```

```
CREATE USER crm_write2@localhost IDENTIFIED BY 'Secure$3452';
```

4. Foydalanuvchilarga rol berish.

Foydalanuvchiga rollar tayinlashda GRANT operatori ishlatiladi.

A) crm_rev rolini bes nomli foydalanuvchiga berish:

```
GRANT crm_dev TO crm_dev1@localhost;
```

V) crm_read rolini bes nomli foydalanuvchiga berish:

```
GRANT crm_read TO crm_read1@localhost;
```

C) crm_write rolini bes nomli foydalanuvchiga berish:

```
GRANT crm_read, crm_write TO crm_write1@localhost, crm_write2@localhost;
```

Rol tayinlashni tekshirish uchun SHOW GRANTS operatori ishlatiladi.

```
SHOW GRANTS FOR crm_dev1@localhost;
```

Natija:

```

Grants for crm_dev1@localhost
▶ GRANT USAGE ON *.* TO 'crm_dev1'@'localhost'
GRANT 'crm_dev'@'%' TO 'crm_dev1'@'localhost'

```

Ko'rinib turganidek, bu yerda faqat crm_dev1 foydalanuvchisiga taynlangan rollar ro'yxati natija sifatida ko'rsatilmoqda. Agar rollar bilan birga ularga berilgan imtiyozlarni ham ko'rish kerak bo'lsa, u holda USING kalit so'zi ishlatiladi

```
SHOW GRANTS FOR crm_write1@localhost USING crm_write;
```

Natija:

```

Grants for crm_write1@localhost
▶ GRANT USAGE ON *.* TO 'crm_write1'@'localhost'
GRANT INSERT, UPDATE, DELETE ON 'crm'.* TO 'crm_write1'@'localhost'
GRANT 'crm_read'@'%', 'crm_write'@'%' TO 'crm_write1'@'localhost'

```

5. Rollardagi imtiyozlarni bekor qilish

Misol uchun o'qish/yozish imtiyoziga ega crm_write rolidagi foydalanuvchilardan vaqtinchalik o'qishdan boshqa imtiyozlarni olib tashlash quyidagicha bo'ladi:

```
REVOKE INSERT, UPDATE, DELETE ON crm.* FROM crm_write;
```

Bunda crm_write rolidan SELECT imtiyozi qoldirilib, faqat INSERT, UPDATE, DELETE amallari bekor qilindi.

Shuningdek, o'qish/yozish imtiyozini qayta belgilash esa quyidagicha

```
GRANT INSERT, UPDATE, DELETE ON crm.* FOR crm_write;
```

6. Rollarni o'chirish

crm_read, crm_write roles rollarini o'chirish:

```
DROP ROLE crm_read, crm_write;
```

SQLda bir foydalanuvchining imtiyozlarni boshqa foydalanuvchiga nusxalash imkoniyati mavjud. Misol uchun CRM MBda crm_dev2 nomli yangi foydalanuvchi yaratilgan bo'lib, unga crm_dev1 foydalanuvchining imtiyozlarini nusxalash talab etilsin.

Birinchi qadamda crm_dev2 nomli foydalanuvchi yaratiladi:

```
CREATE USER crm_dev2@localhost IDENTIFIED BY 'Secure$6275';
```

Ikkinchi qadamda crm_dev1 imtiyozlari crm_dev2 foydalanuvchiga quyidagicha nusxalanadi:

```
GRANT crm_dev1@localhost TO crm_dev2@localhost;
```

Xulosalar.

- MBBTda foydalanuvchining ob'ektlarga ta'sirida uchta omil ishlatiladi: foydalanuvchi (USER), imtiyoz (SELECT, INSERT DELETE, UPDATE), rollar (ROLE).
- Foydalanuvchilar (User) va rollarni (Role) yaratish CREATE Users va CREATE ROLE operatori orqali amalga oshiriladi.

- Foydalanuvchilar (User) va rollarni (Role) o'chirish DROP Users va DROP ROLE operatori orqali amalga oshiriladi.
 - Rollar – imtiyozlar jamlanmasi;
 - GRANT - foydalanuvchi va rollarga imtiyozlar berish vazifasini bajaradi;
 - REVOKE - foydalanuvchi va rollarga berilgan imtiyozlarni bekorlash vazifasini bajaradi;
 - REVOKE faqat ko'rsatilgan imtiyozlarni imtiyozlarni bekorlab, u foydalanuvchi va rollarga berilgan imtiyozlarni barchasini o'chirmaydi:
 - SHOW GRANTS operatori bilan mavjud imtiyozlar tasnifi ko'riladi
 - MBning ob'ektlariga sxema, jadval, tasavvur va protseduralar kiradi;
 - MBBTlarida foydalanuvchiga qaysidir imtiyoz uchun ruxsat qilingan jadval va tasavvurlarning ustunlarini va qatorlarini ham chegaralab kursatish mumkin.
 - Jadval va tasavvurlarda ustunlarini chegaralab qo'yish uchun GRANT konstruktsiyasining imtiyoz berish bo'limiga qavs orqali ko'rsatiladi;
- GRANT [SELECT,] | [INSERT,] | [DELETE,] | [UPDATE] (ustunlar);
- Jadval ustunlari va qatorlarini chegaralab qo'yishda tasavvurlardan foydalanish avfzal yo'l hisoblanadi.
 - Maxsus imtiyozlarni belgilashda odatda bitta jadval va tafovut tug'diradigan bo'lsa bitta imtiyoz belgilash lozim.

Nazorat savollari

1. MB xavfsizligini ta'minlash dolzarbligini tushuntiring?
2. Identifikatsiya va autentifikatsiya vazifalari nimalardan iborat?
3. Xavfsizlik modeli nimalardan tashkil topadi?
4. MB xavfsizligini ta'minlashda ob'ekt va sub'ektlar.
5. Xavfsizlikning diskretion modelining asosiy g'oyasi nima?
6. Mandatli modeli qanday?
7. Xavfsizlikni ta'minlashda qaysi modellar ko'proq qo'llaniladi?
8. Foydalanishni boshqarish nima?
9. Foydalanish imtiyozlari turlarini sanang.
10. Imtiyozlar bilan ishlash buyruqlari qanday?
11. MB xavfsizligida tasavvurlarning ahamiyati.
12. Imtiyozlarni cheklanish qanday amalga oshiriladi?

7.3. MBga kirish texnologiyalari

MBga kirish texnologiyalari haqida

Ilovalar bilan MBning o'zaro aloqasini turli usullar bilan tashkil qilish mumkin. Bunda asosan foydalanuvchi kompyuteriga o'rnatilgan, mijoz dasturi tarkibiga kiritilgan MBBT kutubxonasiidagi API (Application Programming Interface) usullar ustunlik qiladi. Dasturchi o'z muhitida amaliy dasturlarni ishlab chiqish vaqtida dasturiga tizimli o'zgartirish va qo'shimchalar kiritmasligi uchun universal usullardan foydalanadi.

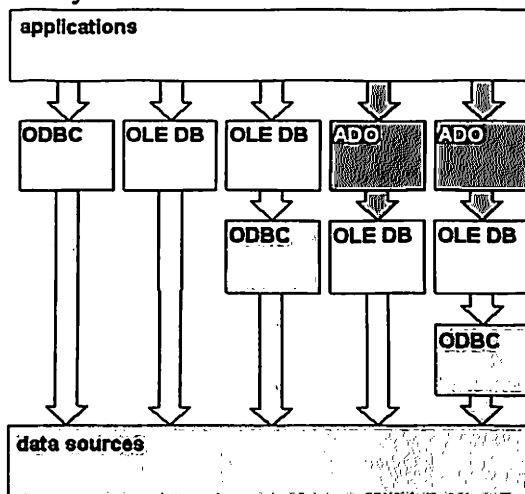
Ma'lumotlarga kirishning universal mexanizmlari *odatda kutubxona va qo'shimcha modullar* (drayver yoki provayder) ko'rinishida qo'llaniladi. *Kutubxonalarda* ma'lum strandart sinflar, usullar va parametrlar to'plami mujassam hamda ma'lumotlarga kirishning standart interfeysi bilan ta'minlangan. *Qo'shimcha modullar* aniq MBBT uchun mijoz ARI funktsiyalariga to'g'ridan-to'g'ri murojaatini ta'minlaydi. Bundan tashqari bu qo'shimcha modullar joriy ehtiyojlardan kelib chiqib o'rnatiladi. Misol uchun agarda ilova ADO ning Ado Table vizual komponentasidan foydalangan uni MBni MySQL dan Postgres ga o'tkazish lozim bo'lsa, u holda kerakli drayverlar o'rnatish va parametrlar komponentasidan Connection qatorini o'zgartirish yetarli bo'ladi. Bunda ilova kodini o'zgartirish shart emas.

Universal usullar kamchiliklarsiz (ishlashning yomonlashuvi, mos drayverlarni yetkazish va sozlash va h.k.) bo'lsa ham bu barcha asosiy afzalliklar universallik bilan taqqoslanmaydi. Shuni unitmaslik lozimki, ular dasturchilarning ish samaradorligini oshiradi, chunki spetsifik interfeys va vositalarni o'rganishning keragi yo'q.

Hozirgi kunda ma'lumotlarga kirishning keng tarqalgan universal usullari qatoriga quyidagilar kiradi

- *ODBC* – Open Database Connectivity (ochiq ma'lumotlar bazasiga ulanish).
- *OLE DB* – Object Linking and Embedding Database (ob'ektlarni bog'lash va ichki MB).
- *ADO* – ActiveX Data Objects (ActiveX ma'lumotlar ob'ekti).
- *BDE* – Borland Database Engine (Borland MYu mexanizmi).
- *ADO.NET* – ActiveX Data Objects texnologii NET (NET texnologiyasi ActiveX ma'lumotlar ob'ekti)
- *JDBC* – Java Database Connectivity (Java da MBga ulanish)

ODBC (Microsoft) – relyatsion MB bilan o‘zaro ta’sir mexanizmi. MBga kirishni ta’minlash uchun MBBTning mijoz qismi talab qilinadi³. Ushbu MBBTga kirish uchun *ODBC* drayveri va uni kompyuterga mos *ODBC* sozlash (odatda drayver, foydalanuvchi, MB nomi, paroli va ba’zi drayverlar) zarur. Drayver ilova bajarilganda ilova bajariluvchi manzilli muhitga yuklanadi va MBga kirish uchun ishlatiladi. Har bir MB o‘zining *ODBC* drayveridan foydalanadi. *ODBC* API standartlashtirilgan.



7.3.1-rasm.

OLE DB (Microsoft) - SOM-interfeysi asosida turli xil ma’lumot manbalari bilan ishlash mexanizmi (*ODBC* faqat relyatsion MB bilan ishlaydi). *OLE DB* SOM-interfeyslar (Component Object Model) to’plamini belgilaydi, jumladan, turli xizmatlarning turli xil ma’lumotlarga (norelyatsion MB kabi elektron pochta tizimi papkalari yoki oddiy fayllar) bir turdagi kirishlarni, ierarxik yozuvlar to’plami va ma’lumotlar to’plami bilan ishlashni qo’llab-quvvatlashni va tarmoqqa beqaror ulanishni ta’minlaydi. MBga kirishda MBBT uchun *OLE DB* provayderini o’rnatish talab etiladi (ilova bajarilayotgan manzilli muhitga MBBT DLL yuklanadi). Bundan tashqari firma MBBTning mijoz qismi API orqali emas, balki *ODBC* API interfeysi orqali ishlaydigan maxsus provayder (Microsoft *OLE DB* Provider for *ODBC* Drivers) ishlab chiqqan.

ADO (Microsoft) - *OLE DB* ning takomillashtirilgani (*OLE DB* kutubxonasidan foydalanadi), ma’lumotlarga kirish interfeysni amalga oshiruvchi COM-ob’ektlarini qamrab oluvchi qo’shimcha kutubxonalar to’plami. Bu kutubxona to’plami dastlab Microsoft ning ikki ob’ektlari

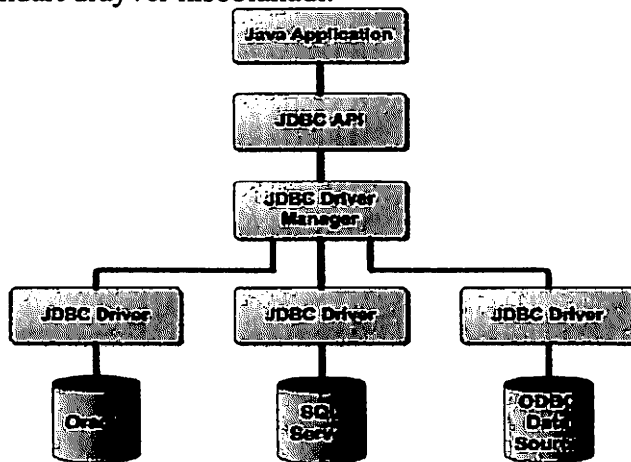
³ https://studopedia.su/16_50867_obzor-osnovnih-tehnologiy-dostupa-k-bazam-dannih.html

texnologiyasni o'z ichiga oladi: Data Access Objects (DAO) va Remote Data Objects (RDO), ya'ni mos ravishda MBga lokal va uzoqdan kirish mexanizmi. Yagona texnologiyani yaratish talabidan kelib chiqib, MB bilan ishlashning yagona yondoshuvini hamda ma'lumotlarga lokal va uzoqdan kirishning yagona interfeysi ta'minlash maqsadida ADO texnologiyasi paydo bo'lgan. Shuningdek, OLE DB texnologiyasi asosida ADO har tomonlama qo'llab-quvvatlovchi va har qanday MB bilan ishlashga imkon beradi.

BDE (Borland) – ma'lumotlarga kirishning universal mexanizmi bo'lib, u ikkita drayver-kutubxonaga guruhiga asoslangan: SQL Links – MBBT serveri uchun hamda ODBS Links – serverli va avtonom MBBT uchun). Bu kutubxonalar dBase, Access, FoxPro, ODBC-manbalari, shuningdek, ko'pchilik MBBT serverlaridan ma'lumotlarga kirish uchun standart funktsiyalar to'plamidan foydalanishgan imkon beradi. BDE vizual muhitning komponentalar darajasida

ADO.NET (Microsoft) - three-tier (ko'pyarusli arxitektura) MB bilan ishlash texnologiyasi. Bunda MBga ulanish MB bilan ishlash davrida amalga oshiriladi. Bu texnologiyaning paydo bo'lishi Internet server MB kabi MBga kirishlar sonining ortishi vako'p hajmdagi faol kirishlarning qo'llab-quvvatlanmaslik muammolaridan kelib chiqqan. ADO.NET texnologiyasi shu kabi muammolarni hal qilish hamda dasturlashning soddaligi va qulayligini ta'minlashga qaratilgan.

JDBC – Java dasturini MB bilan bog'lash uchun foydalaniladigan drayver. JDBC Driver platformaga bog'liq bo'lmagan, turli MB bilan bog'lanish imkonini beradigan, JavaSE da java.sql paketi tarkibiga kiritilgan standart drayver hisoblanadi.



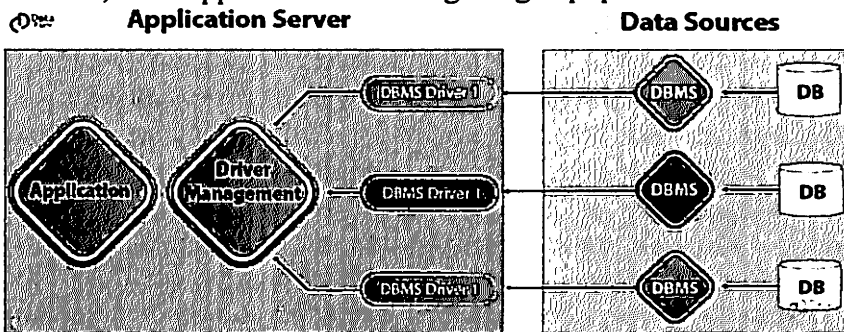
7.3.2-rasm.

ODBC texnologiyasi

Hamma API lar kabi ODBC birgalikda ma'lum funktsiyalar to'plamini ta'minlovchi sinflar va interfeyslar to'plamidir.

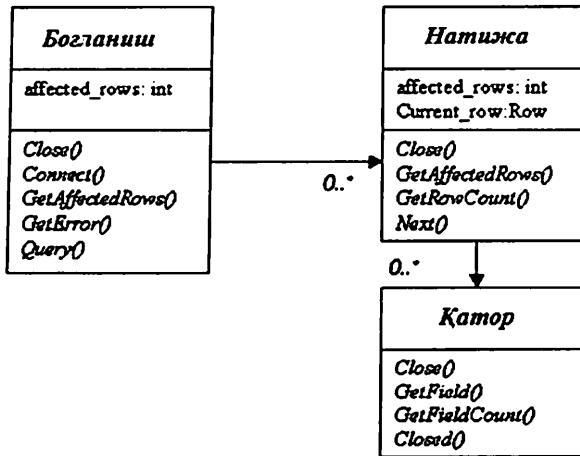
ODBC arxitekturasi to'rtta komponentdan iborat:

1. Application – SQL so'rovlarini yuborish va natijalar olish uchun ODBC funktsiyalarini chaqiradi va qayta ishlaydi;
2. ODBC Driver – ilova nomidan drayverni yuklaydi va bajaradi;
3. Driver Manager – ODBC funktsiyalarini chaqirishni boshqaradi, SQL so'rovlarini aniq MBga yuboradi va natijalarni ilovaga qaytaradi. Agar lozim bo'lsa, bog'langan MBBT tomonidan qo'llab-quvvatlanadigan sintaksisga mos keladigan qilib ilova so'rov kodini o'zgartiradi;
4. Data Source – ma'lumot manbasi, bunda foydalanuvchi kirishi lozim MBBT, tarmoq platformasi va unga bog'liq operatsion tizim.



7.3.4-rasm.

ODBC funktsiyalari MBga murojaatni ta'minlaydi. ODBC API ni tashkil qiluvchi sinflar va interfeyslar ixtiyoriy turdagi ma'lumotlar murojaat qilishdagi umumiy tushunchalar abstraktsiyasidir. Masalan, Connection MB bilan bog'lanishni tasvirlovchi interfeysdir. Shunga o'xshab Result Set SQL SELECT buyruqi qaytaruvchi natijaviy to'plamni tasvirlaydi. Tabiiyki MBga murojaat konkret detallari uning yaratuvchisiga bog'liq. ODBC bu detallar bilan ishlamaydi. ODBC sinflarini MBni dasturlashga ob'ektga-yo'naltirilgan usullar nuqtai nazaridan ko'rib chiqiladi.



7.3.5-rasm. MBga ob'ektga-yo'naltirilgan murojaat bibliotekasi

MBBT bilan ishlashni uchta asosiy tushuncha tasvirlaydi: ulanish, natijaviy to'plam va natijaviy to'plam satrlari. 7.3.5-rasm bu ob'ektlarni UML-diagrammada ko'rsatadi. UML-bu yangi Unifikatsiyalangan Modellashtirish Tili bo'lib, Grady Booch, Ivar Jacobson, James Rumbaugh tomonidan ob'ektga-yo'naltirilgan loyihalash va taxlilni xujjatlash yangi standarti sifatida taklif qilingan.

MBga ulanish. Ixtiyoriy muhitda MBa murojaat ulanishdan boshlanadi. Bizning ob'ektga yo'naltirilgan biblioteka yaratish Connection ob'ektini yaratishdan boshlanadi. Ob'ekt Connection server bilan bog'lanishni o'rnatish, zarur MBni tanlash, so'rovlarni uzatish va natija olishni bilishi kerak. Connection sinfi usullari hamma MBBT uchun bir xildir. Lekin sinf ichida, kompilyatsiya qilinayotgan biblioteka uchun xos bo'lgan yopiq qismlar yashiringandir. Bog'lanishni o'rnatishda ma'lumotlar bilan bog'lanishni ta'minlaydigan sinf qismlari farqli bo'lib qoladi.

MB bilan ulanishni o'rnatish. Bu API yordamida yaratiladigan hamma amaliy dasturlarga MBga ulanish uchun Connection sinfi nusxasini uning konstruktorlaridan biri yordamida yaratish kerak bo'ladi. U kabi uzilish uchun Amaliy dastur Connection nusxasini o'chirishi kerak. U to'g'ridan to'g'ri `Close()` va `Sonnect()` usullariga murojaat qilib Connection nusxasini qaytadan ishlatishi mumkin.

MBdan uzilish. Connection yana bir mantiqiy funksiyasi MB bilan aloqani uzish va dasturdan berkitilgan resurslarni ozod qilishdir. Bu funksiyani `Close()` usuli amalga oshiradi.

MBga murojaatlarni bajarish. Bog'lanishni ochish va yopish odatda MBga komandalar yuboriladi. Connection sinfi argument sifatida SQL komanda oluvchi Query() usuli yordamida bajaradi. Agar komanda so'rov bo'lsa 2-5.rasmda ko'rsatilgan ob'ekt modelidan Result sinfi nusxasini qaytaradi. Agar komanda ma'lumotlarni yangilayotgan bo'lsa, usul NULL qaytaradi va affected_rows qiymatini o'zgartirilgan satrlar soniga teng qiladi.

Natijaviy to'plamlar. Result sinfi natijaviy to'plam ma'lumotlariga hamda shu natijaviy to'plam bilan bog'liq metama'lumotlarga murojaatni ta'minlashi kerak. 2-5 rasmda ko'rsatilgan ob'ektli modelga asosan bizning Result sinfimiz natijaviy to'plam satrlarini tsikl bo'yicha o'qish va undagi satrlar sonini aniqlashni ta'minlaydi.

Natijalar bo'yicha ko'chish. Result sinfi natijaviy to'plam bilan qatorma qator ishlaydi. Result sinfi nusxasini Query() usuli yordamida olgandan so'ng amaliy dastur to'navbatdagi Next() usuli 0 qaytarmaguncha, ketma ket Next() va GetCurrenRow() usullarini chaqirishi lozim.

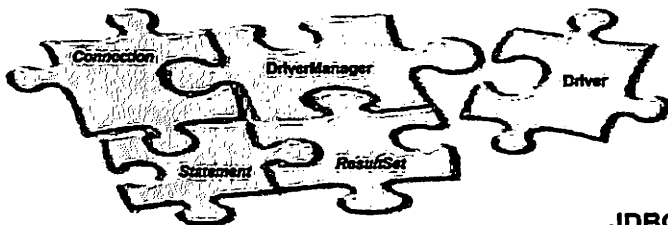
Satrlar. Natijaviy to'plamning aloxida satri ob'ektli modelda Row sinfi bilan tasvirlanadi. Ma'lumotlarga murojaat massiv indeksi bo'yicha so'rov tomonidan berilgan tartibda amalga oshiriladi. Masalan, SELECT user_id, password FROM users so'rov uchun indeks 0 foydalanuvchi nomini va indeks 1 parolni ko'rsatadi. C++ API bu indeklashni foydalanuvchi uchun xamkorlik qiladi. GetField(1) yoki fields[0] birinchi maydonni qaytaradi.

JDBC texnologiyasi

JDBC standart API bo'lib, MBga ulanishda SQL so'rovlarning bajarilishini ta'minlaydi. JDBC bir qator klass va interfeyslar to'plamidan iborat bo'lib, u Java ilovalari va MB uchun foydalaniladi. Uning asosiy komponentalari quyidagilardan iborat⁴:

DriverManager – bu klass, Driver (database drivers) ro'yxatini boshqarish uchun foydalaniladi.

Driver – bu interfeys, MB bilan aloqani o'rnatish va ushbu aloqani boshqarishda foydalaniladi. Bu yerda Driver yuklanganda, dasturchi uni chaqirishi shart emas.



JDBC API

Connection – MBga bog'lanishning barcha usullarini interfeysi. U aloqa kontekstini tavsiflaydi. MBga barcha murojaatlar faqat ushbu ob'ekt bilan amalga oshiriladi.

Statement – bu interfeys, MBga jo'natilgan SQL buyruqlarni taxlil qilish, umumlashtirish, rejalashtirish va bajarish vazifalari kamrab oladi.

ResultSet – so'rov natijasida olingan qatorlar to'plamini beradi.

Odatda JDBC API da `java.sql.Driver` interfeysi mavjud bo'ladi. Shuning uchun MB turiga mos bo'lgan Driver kutubxonalarni yuklab olinadi. Buning uchun quyidagi Internet manzillaridan MBga tegishli drayverlarning jar paketini ko'chirib olish zarur.

- *MySQL uchun:* <http://dev.mysql.com/downloads/connector/j/>
- *PostgreSQL uchun:* <http://jdbc.postgresql.org/download.html>
- *Oracle uchun*

<http://www.oracle.com/technetwork/database/features/jdbc/index091264.html>

MB bilan ishlashda asosan quyidagi ketma-ketlik bajarilishi lozim.

1. *Class.forName("klasning to'liq nomi")* – operatori orqali MBga ulanish drayverining nomi ko'rsatiladi.
2. *Connection conn = DriverManager.getConnection("jdbc:<subprotocol>:<subname>", "name", "password")* – ulanayotgan MB tasnifi ko'rsatiladi. Bu yerda *conn* – tasnifni o'zgaruvchiga berish, *subprotocol* – drayver nomi yoki ulanish mexanizmi nomi (MBBT turi), *subname* – xost, port va MB nomini bildiradi. JDBC Manager MB bog'lanishni bajarib, uning asosida MB bilan ishlashga imkoni yaratiladi.
3. *statement.execute("SQL so'rov qatori")* bilan natijasi massivga olinmaydigan so'rovlar (Insert, Update, Delete) bajariladi
4. *ResultSet rs = statement.execute("SQL so'rov qatori")* bilan bajarilgan so'rov natijasi ko'rsatilgan massivga (rs) yoziladi.
5. *conn.close()* – *conn* nomli bog'lanishni yopadi.

Turli MBga bog'lanishni amalga oshirish namunalari.

MySQLga ulanish:


```

Class.forName("com.mysql.jdbc.Driver");
Connection conn =
DriverManager.getConnection("jdbc:mysql://hostname:port/dbname",
"username", "password");
conn.close();
PostgreSQLga ulanish:
Class.forName("org.postgresql.Driver");
Connection connection =
DriverManager.getConnection("jdbc:postgresql://hostname:port/dbname",
"username", "password");
connection.close();
ORACLEga ulanish:
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:mkyong",
"username", "password");
connection.close();

```

Iloji boricha JDBC drayverini try {} va catch {} ichiga joylashtirish lozim. Bu esa drayverning kompyuterda borligini va ishlashini nazorat qiladi.

```

try
{
Class.forName("com.mysql.jdbc.Driver");
}
catch (ClassNotFoundException e)
{
System.out.println("Where is your MySQL JDBC Driver?");
e.printStackTrace();
return;
}

```

Javada MBga ulanish uchun klass hosil qilinadi:

```

private static Connection getDBConnection()
{
Connection dbConnection = null;
try {Class.forName(DB_DRIVER);}
catch (ClassNotFoundException e) {System.out.println(e.getMessage());}
try {
dbConnection = DriverManager.getConnection(DB_CONNECTION,
DB_USER,DB_PASSWORD);
return dbConnection;
}
catch (SQLException e) {System.out.println(e.getMessage());}
return dbConnection;
}

```

Quyida ko'rsatilgan MBda jadval yaratish metodi keltirilgan:

```

private static void createDbUserTable() throws SQLException{
Connection dbConnection = null;
Statement statement = null;
String createTableSQL = "

```

```

CREATE TABLE DBUSER("
    + "USER_ID NUMBER(5) NOT NULL, "
    + "USERNAME VARCHAR(20) NOT NULL, "
    + "CREATED_BY VARCHAR(20) NOT NULL, "
    + "CREATED_DATE DATE NOT NULL, "
    + "PRIMARY KEY (USER_ID) " + ");

try {
    dbConnection = getDBConnection();
    statement = dbConnection.createStatement();
    // SQL so'rovini bajarish
    statement.execute(createTableSQL);
    System.out.println("Table 'dbuser' is created!");
}
catch (SQLException e) {System.out.println(e.getMessage());}
finally {
    if (statement != null) {statement.close();}
    if (dbConnection != null) {dbConnection.close();}
}
}

//main funksiyasi ichiga createDbTable() metodini chaqirish
public static void main(String[] argv) {
    try {createDbUserTable();}
    catch (SQLException e) {System.out.println(e.getMessage());}
}

MBga ma'lumot kiritish va o'qish:
String insertTableSQL = "INSERT INTO DBUSER"
    + "(USER_ID, USERNAME, CREATED_BY, CREATED_DATE) "
    + "VALUES"+ "(1,'mkyong','system', "
    + "to_date(" + getcurrentTimeStamp() + ", 'yyyy/mm/dd hh24:mi:ss'))";

private static String getcurrentTimeStamp()
    {Date today = new Date();
    return dateFormat.format(today.getTime()); }

statement.executeUpdate(insertTableSQL);
//MBdan ma'lumotlarni o'qish:
String selectTableSQL = "SELECT USER_ID, USERNAME from DBUSER";
try {
    dbConnection = getDBConnection();
    statement = dbConnection.createStatement();
    // MBdan ma'lumotlarni tanlash
    ResultSet rs = statement.executeQuery(selectTableSQL);
    // Ma'lumot olingan bo'lsa while tsikliga tashlash.
    while (rs.next()) {
        String userid = rs.getString("USER_ID");
        String username = rs.getString("USERNAME");
        System.out.println("userid: " + userid);
        System.out.println("username: " + username); }
}
catch (SQLException e) {System.out.println(e.getMessage());}

```

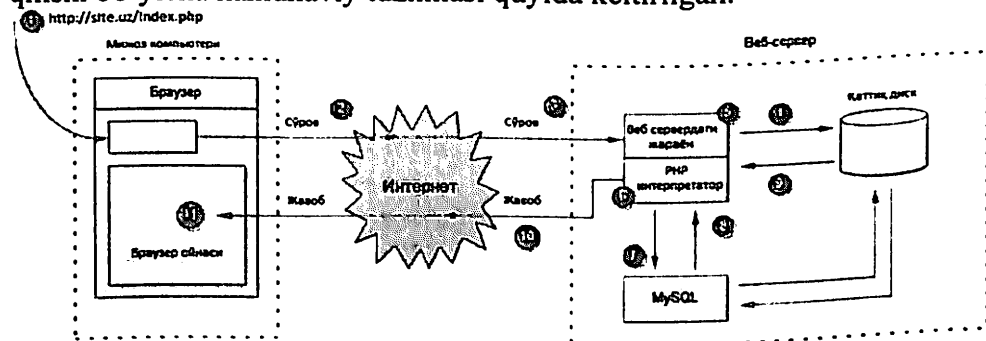
Web dasturlashda MB murojaat

Ma'lumki, hozirgi kunda ko'pgina axborot tizimlar, platformalar, shuningdek, veb saytlar veb dasturlashtir va texnologiyalar bilan ishlab chiqiladi va Internet tarmog'iga qo'yiladi. Odatda veb dasturchilar Internetga veb axborot tizimlarini joylashtirishdan oldin, uni loyihasini, MB tuzilmalarini va saxifalovchi kodlarini lokal serverida ishlab, testlan o'tkazib olishadi.

Lokal serverni o'rnatish uchun albatta server texnologiyalari: *Apache, PHP, MySQL* kerak bo'ladi. Ularni alohida olib o'rnatib chiqish mumkin, lekin bu dasturchi uchun ortiqcha qiyinchilik va vaqtni oladi. Bularning tayyor yechimlari orasida eng ommalashgan bopi DENWER hisoblanadi. Lekin, DENWER zamonaviy talablarga uncha mos kelmaydi. Shuning uchun Open Serverdan foydalanish tavsiya etiladi. Open Serverda veb texnologiyalarning eng so'nggi versiyalari o'rnatilgan bo'ladi va unda tashqari qo'shimcha imkoniyatlar, kengaytirilgan sozlamalar, qo'shimcha modullar ham mavjud.

Shuni esda tutish lozimki, har qanday veb axborot tizimida MB bilan ishlash uchun oldindan veb senariy (PHP tilida) ishlab chiqiladi. Bunda veb sahifadagi havola bosilganda mijoz brauzeri veb serverga so'rov yuboradi. So'rov yuborish POST yoki GET metodlaridan birortasi orqali amalga oshiriladi. Mizojdan yuborilgan so'rovni server tomonida qayta ishlab, yana mijozga qaytarish qo'p bosqichli

Umumiy holda veb sahifalarning veb serverlar orqali MBga murojaat qilishi bo'yicha namunaviy tuzilmasi quyida keltirilgan:



1. Brauzer adres qatoriga veb sahifa manzilini kiritish.
2. Brauzer manzilni ajratadi va sahifa nomini veb serverga jo'natadi
3. Xostdagi veb server index.php sahifasini so'rov uchun qabul qiladi
4. Veb-server xostning qattiq diskidan index.php faylini o'qiydi

5. Veb-server uni oddiy HTML fayl emas, balki PHP stsenariy ekanligini aniqlaydi va uni boshqa jarayon – PHP interpretatorga qayta ishlash uchun jo'natadi
6. PHP interpretator veb server jarayonidan qabul qilingan matnda topilgan PHP-kodni bajaradi. Bu kod MySQL MBga murojaatni o'z ichiga oladi.
7. PHP interpretatori MySQL MBga so'rov yuboradi
8. MySQL MB so'rov natijasini qaytaradi.
9. PHP interpretator PHP-kodni bajarishni yakunlaydi, MBdan qabul qilingan ma'lumotni qo'shadi va natijani veb server jarayoniga qaytaradi.
10. Veb-server brauzerga natijani HTML matn ko'rinishida qaytaradi.
11. Veb-brauzer kompyuteringiz ekranida veb sahifaning tashqi ko'rinishini qabul qilingan HTML matnga mos ravishda shakllantiradi

Quyida PHP interpretatori MySQL MB bilan ishlash buyruqlari batafsil keltiriladi:

- *mysqli_connect()* – MB tasnifi bo'yicha ulanish
- *mysqli_query()* – SQL so'rovni bajarish
- *mysqli_stmt_num_rows()* – bajarilgan so'rov natijasidagi yozuvlar soni
- *mysqli_stmt_field_count()* – bajarilgan so'rovdagi ustunlar soni
- *mysqli_fetch_array()* – so'rov natijalarini qatorli ko'rinishda chiqarish
- *mysqli_insert_id()* – jadvalga eng oxirgi kiritilgan yozuv ID raqami
- *mysqli_store_result()* – oxirgi so'rov natijalarini lokal saqlash
- *mysqli_stat()* – tizimning joriy xolati
- *mysqli_close()* – MBga ulanishni yakunlash.

Misol. Adabiyotlar ro'yxatidan eng keyingi yilda chiqarilganlarini mualliflari bilan birga chiqaruvchi oddiy adabiyot.php sahifasini yaratish.

Bunda PHP tili operatorlari va HTML teglaridan iborat adabiyot.php fayli quyidagi koddan iborat bo'ladi.

```
<html>
  <head>
    <meta content="text/html; charset=utf-8">
    <title>Adabiyotlar ro'yxati</title>
  </head>
</body>
<?php
//Lokal serverdagi KITOB nomli MBga
//root foydalanuvchi bo'lib ulanish
```

```

$dtcon = mysqli_connect("localhost", "root", "", "KITOB");
//SQL so'rov matnini $sql o'zgaruvchisiga berish
$sql="SELECT DISTINCT C.name_cat AS toifa,
        A.fio AS muallif,
        B.id_book AS id_asar,
        B.name AS asar,
        B.page AS bet,
        MAX(B.yil) AS nashr_yili
FROM Books B
INNER JOIN Avtor A ON B.id_avtor = A.id_avtor
INNER JOIN Categoria C ON B.id_cat = C.id_cat
GROUP BY B.name
ORDER BY asar";

```

```
//HTML da jadval shakli
```

```

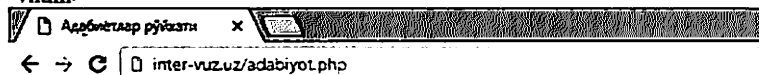
?>
<table>
<tr>
<th>#</th>
<th>Kategoriya</th>
<th>Muallif</th>
<th>Adabiyot nomi</th>
<th>Saxifa</th>
<th>Yili</th>
</tr>
<?
//$sql o'zgaruvchidagi SQL so'rovni serverga jo'natish
//va natijani $r massiviga saqlash
$r= mysqli_query($dtcon, $sql);$sus=0;

//ikki o'lchamli $r massivning har bir qatorini
//tsikl orqali bir o'lchamli $e massivga yozish
while($e = mysqli_fetch_array($res))
{
    $sus++;
    //jadval yacheykalariga $e massivdagi ma'lumotlarni chiqarish
    ?>
    <tr>
        <td><?=$sus?></td>
        <td><?=$e[toifa]?></td>
        <td><?=$e[muallif]?></td>
        <td><?=$e[asar]?></td>
        <td><?=$e[bet]?></td>
        <td><?=$e[nashr_yili]?></td>
        <td>
            <? echo "<a href='?st=$e[id_asar]?'>Edit</a>"?>
        </td>
    </tr>
    <?
}
</table>
?>

```

</body>

</html>



#	Категория	Муаллиф	Адабиёт номи	Саҳифа	Йил	
1	Tarixiy	Ibraym Yusupov	Dala armonlan	450	1967	Edin
2	Ertak	Xalq ogzaki ijodi	Dexqon va ayiq	100	2019	Edin
3	Tarbiyaviy	Utkir Hoshimov	Dunyoning ishlari	280	2005	Edin
4	Ertak	Xalq ogzaki ijodi	Eng katta setr	230	2020	Edin
5	New Categoria 6	Toxir Malik	Giybat ozi nima	100	2020	Edin
6	Ertak	Xudayberdi To'xtaboyev	Gulliverni sarguzastlari	180	2020	Edin
7	Tarixiy	Utkir Hoshimov	Ilki eshik orasi	450	2020	Edin
8	Tarbiyaviy	Xudayberdi To'xtaboyev	Jannati odamlar	210	1996	Edin
9	Tarbiyaviy	Ibraym Yusupov	Mangu buloq	360	1970	Edin
10	Tarbiyaviy	Toxir Malik	Nafs kishanlari	240	2020	Edin
11	Tarbiyaviy	Toxir Malik	Odamiylik Mulki	950	2016	Edin
12	Tarixiy	Ibraym Yusupov	Qora tol	350	1988	Edin
13	Detektiv	Toxir Malik	Samum	380	2012	Edin
14	Ertak	Xudayberdi To'xtaboyev	Sanq devmi minab	220	2005	Edin
15	Fantastik	Xudayberdi To'xtaboyev	Sehrlil qalpoqcha	250	1964	Edin
16	Detektiv	Toxir Malik	Shaytanat	1250	2020	Edin
17	Tarbiyaviy	Ibraym Yusupov	Sherlar guldastasi	250	0	Edin
18	Ertak	Xudayberdi To'xtaboyev	Shirin qovunlar mamlakati	180	1986	Edin

Nazorat savollari

1. Ma'lumotlarga kirishning universal mexanizmlarining zaruriyati nimadi va turlarini sanang?
2. ODBC texnologiyasidagi komponentalarning vazifalari nimadi?
3. JDBC bilan ODBC farqi nimda?
4. Veb dasturlashtirishda stsenariy deb nimaga aytiladi?
5. Mijoz brauzeridan MBga murojaat bosqichlari qanday?

FOYDALANILGAN ADABIYOTLAR

1. Agaltsov V.P. Bazi dannix. V 2-x kn. Kniga 2. Raspredelennie i udalennie bazi dannix: uchebnik // Moskva. Izd. «Forum, Infra-M» - 2018. S.261
2. Golitsina O.L., Maksimov N.V., Popov I.I. Bazi dannix: Uchebnoe posobie. 2-e izd. // Moskva. Izd. «Forum, Infra-M» - 2009. S.4001
3. Martishin S. Bazi dannix. Prakticheskoe primeneniye SUBD SQL i NoSQL-tipa dlya proektirovaniya informatsionnix sistem. Uchebnoe posobie // Moskva. Izd. «Forum, Infra-M» - 2019. S.368
4. Rahul Batra. SQL Primer - An Accelerated Introduction to SQL Basics // Apress; 1st ed. Edition. 2018. Gurgaon, India. 212 p.
5. Luc Perkins, Eric Redmond, Jim Wilson. Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement. // Pragmatic Bookshelf, 2018. 360 p.
6. Elmasri, Ramez; Navathe, Shamkant B. Fundamentals of Database Systems (7th Edition) // Pearson, 2015. 671 p.
7. Groff, Djeys R., Vaynberg, Pol N., Oppel', Endryu Dj. SQL: polnoe rukovodstvo, 3-e izd. : Per. s angl. - M.: OOO "I.D. Vilyame", 2015. - 960 s.
8. Abraham Silberschatz, Henry F. Korth, S. Sudarshan. Database system concepts // 6th ed. Published by McGraw-Hill. New York, NY 10020. Copyright © 2011/ P.1376
9. Ilyushechkin V. M. Osnovi ispolzovaniya i proektirovaniya baz dannix: uchebnik dlya SPO /. Seriya : Professionalnoe obrazovanie. M.: Izdatelstvo Yurayt, 2019. 213 s.
10. Fiayli K. SQL: Per. s angl. – M.: DMK Press. – 456 s. Moskva. 2013 g.
11. Jeffrey A. Hoffer, Mary B. Prescott, and Fred R. McFadden. Modern Database Management (8th Ed.) – 557 r. Prentice-Hall, 2007.
12. For those seeking a stronger technical treatment of database systems: Elmasri, R. and S. B. Navathe: Fundamentals of Database Systems (5th Ed.) – 671 r. Addison Wesley, 2015.
13. Fundamentals of database systems sixth edition. Ramez Elmasri. Department of Computer Science and Engineering The University of Texas at Arlington. 2011
14. Vvedenie v Oracle 10g. Perri Djeys, Post Djerald. 697 str 2013
15. Grigorev Yu.A., Plutenko A.D.. Jiznenniy tsikl proektov raspredelennix baz dannix. Blagovещesnk AmGU, 1999.

16. Digo S.M. Bazi dannix Proektirovanie i ispolzovanie. izdatelstvo "Finansi i statistika" 592 str, 2005 g.
17. Alov R.D., Ne'matova D.E. Berilganlar bazalari tizimlarida zamonaviy texnologiyalari fanidan ma'ruzalar matni. Toshkent-2018. 345 b.
18. Muradova N.K., Sharipov B.A., Majidov R.R., Tulyaev D. Ma'lumotlar va bilimlar bazasi. Oliy o'quv yurtlari talabalari uchun o'quv qo'llanma. - T.: TDIU, 2007. 355 – bet.
19. Nazirov Sh., Ne'matov A., Qobulov R. Ma'lumotlar bazasini dasturlash chuqurlashtirilgan kursi // — T.: Sharq, 2007. — 136 b.
20. <https://www.w3resource.com/>
21. www.w3school.com;
22. <https://www.mysqltutorial.org/>
23. <http://2sql.ru/>
24. <http://www.sql-ex.ru/>
25. www.intiut.ru;
26. www.oracle.com
27. www.ziyonet.uz;

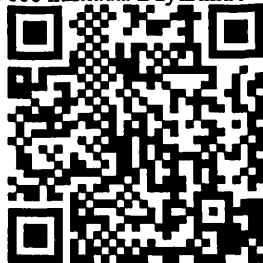
**NISHANOV A.X., BABADJANOV E.S.,
ERGASHEV A.Q., RISNAZAROV A.M.**

MA'LUMOTLAR BAZASI

«Mahalla va oila nashriyoti»
Toshkent – 2021

Nashr uchun mas'ul: B. Mavlonov
Muharrir: U. Yunusov
Badiiy muharrir: F. Sobirov
Dizayner-sahifalovchi: L. Abdullayev

Nashriyot ro'yxat raqami № 1043191. 24.09.2021-y.
Bichimi 60x84 1/16 Offset qog'ozi.
Times New Roman garniturası.
Shartli bosma tabog'i 16,75. Nashr hisob tabog'i 10,6.
Adadi 100 nusxada. Buyurtma № 10-12.



1940

100000, Toshkent shahri, Mirzo Ulug'bek tumani,
M.Ismoilıy ko'chasi 1-G uy.

«ZUXRA BARAKA BIZNES» MChJ bosmaxonasida chop etildi.
Toshkent shahri Bunyodkor shoh ko'chasi 27 A–uy.