КӘСІПТІК БІЛІМ БЕРУ

Э. В.ФУФАЕВ, Д.Э.ФУФАЕВ

ДЕРЕКТЕР ҚОРЫ

«Федералдық білім беруді дамыту институты» Федералдық мемлекеттік мекемемен «Ақпараттық жүйелер (салалар бойынша)» кәсіптік орта білім беру бағдарламаларын іске асыратын білім беру мекемелерінің оқу процесінде пайдалану үшін, оқулық ретінде ұсынылды

2014 ж. 11 желтоқсандағы рецензияның тіркеу нөмірі 504.
«ФББДИ» ФММ
10-шы басылым, стереотиптік

ACADEMA

Мәскеу
«Академия» баспа орталығы
2015

ӘОЖ 621.38(075.32)
КБЖ 3281ші72

Ф94 Бұл кітап Қазақстан Республикасының Білім және ғылым министрлігі және «Кәсіпқор» холдингі» КЕАҚ арасында жасалған шартқа сәйкес «ТжКБ жүйесі үшін шетел әдебиетін сатып алуды және аударуды ұйымдастыру жөніндегі қызметтер» мемлекеттік тапсырмасын орындау аясында қазақ тіліне аударылды. Аталған кітаптың орыс тіліндегі нұсқасы Ресей Федерациясының білім беру үдерісіне қойылатын талаптардың ескерілуімен жасалды.

Қазақстан Республикасының техникалық және кәсіптік білім беру жүйесіндегі білім беру ұйымдарының осы жағдайды ескеруі және оқу үдерісінде мазмұнды бөлімді (технология, материалдар және қажетті ақпарат) қолдануы қажет. Аударманы «Delta Consulting Group» ЖШС жүзеге асырды, заңды мекенжайы: Астана қ., Иманов көш., 19, «Алма-Ата» БО, 809С , телефоны: 8 (7172) 78 79 29, эл. поштасы: info@dcg.kz

Пікір беруші -

Мәскеу мемлекеттік электроника және математика институты
«Информатика және телекоммуникациялар» факультеті деканының орынбасары,
техн. ғылым кандидаты, «Лазерлік микротолқынды ақпараттық жүйелер» кафедрасының доценті*Д. П. Николаев*;
Мәскеу мемлекеттік ақпараттық технологиялар колледжінің оқытушысы
И. А. Кумскова

Фуфаев Э. В.

Ф94 Дерекқор: оқулық кәсіптік орта білім беру мекемелерінің студенттеріне арналған / Э. В. Фуфаев, Д. Э. Фуфаев. — 10-шы басылым, стер. — М. :«Академия» баспа орталығы, 2015. — 320 б.

ISBN 978-601-333-173-7 (каз.)
ISBN 978-5-4468-2436-6 (рус.)

Оқу құралы Федералдық мемлекеттік мекемесімен «Ақпараттық жүйелер (салалар бойынша)»; ОП: «Дерекқорды жобалау негізі» мамандықтары бойынша кәсіптік орта білім берустандартына сәйкес жасалды. Дерекқорпды жобалаудың теориялық негізі және өндіріс пен бизнесті басқару кезінде шешім қабылдау процестерінде оларды практикалық қолдану әдістемелері мазмұндалған.

Кәсіптік орта білім беру мекемелерінің студенттеріне арналған

ӘОЖ 621.38(075.32)
КБЖ 3281ші723

Оқулық басылымы

Фуфаев Эдуард Валентинович, Фуфаев Дмитрий Эдуардович

Деректер қоры

Оқу құралы

10-шы басылым, стереотиптік

Редакторы*И. В. Мочалова.* Техникалық редакторы*Е. Ф. Коржуева*
Компьютерлік парақтау: *М. В. Трушина.* Корректоры*Т. В. Кузьмина*

№ 110108111 басылым. Мөрге қол қойылды 07.07.2015. Форматы 60x90/16. «Таймс» гарнитурасы. Офсетті баспа. № 1 офсетті қағаз. Шартты баспа парағы 20,0. Тиражы 1000 дана. Тапсырыс №

«Академия» баспа орталығы» ЖШҚ. www.academia-moscow.ru129085, Мәскеу, Мира даңғылы, 101В, 1-б.
Тел./факс: (495) 648-0507, 616-00-29.

Санитарлық-эпидемиологиялық қорытынды25.05.2015ж. № РОСС RU. АЕ51. Н16679
«Саратовский полиграфкомбинат» БАҚ баспасы ұсынған электрондық тасымалдағыштан. Басып шығарылды. www.sarpk.ru 410004, Саратов қ., Чернышевский көшесі, 59.

ISBN 978-601-333-173-7 (каз.)
ISBN 978-5-4468-2436-6 (рус.)

АВТОРЛАРДАН

Кез-келген фирманың материалдық өндіріс саласында, сондай-ақ халықтың қызмет көрсету саласында да экономикалық қызметінің мақсаты – табыс табу болып табылады.

Осы мақсатқа қол жеткізу – фирманың барлық қызметкерлерінің шығармашылық қабілеттеріне байланысты көп қырлы тапсырма болып табылады. Өндірістік процестің әрбір қатысушысы, яғни басшыдан бастап қатардағы орындаушыға дейін өздерінің өндірістік процесінде оңтайлы шешімдер қабылдай алатын жағдайда ғана алған қойған міндетті шешуге болады. Кез-келген маманның басты міндеті –қате шешімдердің салдарын түзету емес,бірінші қадамнан бастапоңтайлы шешімдер қабылдау. Дерекқор негізінде әзірленген компьютерлік ақпараттық жүйелерді пайдалана отырып, бұл мәселені тиімді шешуге болады.

Деректер базасы ақпараттық теорияның бағыттарының бірі ретінде компьютерлік ақпараттық жүйелерді құрудың әдістері мен құралдарын құрайды, олардың негізі ерекше жолменқолданушыға оңтайлы шешім қабылдау үшін қажетті ақпаратты алу мен талдаудың тиімді әдістерін беретін құрылымдық файлдар мамандығы болып табылады.

Дерекқорды әзірлеу теориясы – ғылымның салыстырмалы түрде жас саласы, алайда дерекқор қазіргі уақытта жасанды интеллект жүйелері, сараптамалық жүйелері, автоматтандырылған құрылымдық және технологиялық жобалау жүйелері сияқты ақпараттарды өңдеудің автоматтандырылған жүйелерін әзірлеудегі осындай бағыттардың негізі болып табылады.

Оқу құралы бес бөліктен тұрады.

I-бөлім дерекқорды жобалаудың және ұйымдастырудың теориялық негіздеріне арналған.

II-бөлім Microsoft Access қолданбалы бағдарлама жүйесін, Visual Basic, SQL бағдарламалау тілдерін қолдана отырып, дерекқорларды әзірлеу технологиясы сипатталады.

III-тараудатаратылған дерекқорларды басқару жүйесін әзірлеудіңтеориялық негіздері қарастырылғанжәне SQL Server 2000 және Oracle бағдарламалау жүйелеріне шолу жасалады.

IV-бөлімде дерекқордың реляциядан кейінгі деп аталатын, дерекқорды жобалау теориясын дамытудың негізгі үрдістері қарастырылады.

V-бөлімде заманауи өндіріс пен бизнесті басқару міндеттерінде дерекқорларды қолданудың нақты мысалдары берілген.

1-5, 7-9-тарауларын1-ші санаттағы бағдарламалық инженер Дмитрий Эдуардов Фуфаев жазған; Ш. 6, 10-18 –тарауларды техн. ғыл. кандидаты. доцент Фуфаев Эдуард Валентинович жазды.

«Раменский құрал жасау зауыты» ААҚ бас технологиялық бөлімінің қызметкерлеріне алғысымызды білдіреміз.

КІРІСПЕ

XX соңы – XXI ғасырдың басы компьютерлік ақпараттық технологиялардың, әсіресе дерекқорды басқару жүйелерінің (ДҚБЖ) адамзат қызметіне белсенді енгізілуімен сипатталады. ДҚБЖ –ол қолданушыға жылдам ақпарат алуды қамтамасыз ететін құрылымдық деректер файлдарын басқаруға арналған бағдарламалық жүйе.

Құрылымдық деректер файлдары немесе дерекқор автоматтандырылған басқару жүйелерінің (АБЖ), жасанды интеллект жүйелерінің және сараптау жүйелерінің, САПР-СД немесе CAD-жүйелерін (Computer Assisted Design) құрылымдық құжаттаманы автоматтандырылған жобалау жүйелерінің, САПР-ТП немесе CAM-жүйелерінің (Computer Aided Manufacturing) бұйымдарын дайындаудың технологиялық процестерін автоматтандырылған жобалау жүйелерінің ажырамас бөлігі болып табылады.

ЭЕТ ақпараттық логикалық тапсырмаларды шешу бойынша алғашқы отандық теориялық және практикалық жұмыстарды, яғни логикалық ережелердің белгілі бір жиынтығы бойынша ақпаратты сақтау, өңдеу және алу міндеттерінзаманауи ДҚБЖ-нің бейнесі депатауға болады.

XX ғасырдың 60-жылдардың басында жүргізілген осы жұмыстарда ЭЕЖ ресурстарын оңтайлы бөлу, ақпаратты іздестіру жылдамдығы, ақпараттың сенімділігін қамтамасыз ету, бағдарламалау тілдерін құру тапсырмалары шешілді. Аталған жұмыстар сериялық шығарылатын«2-Минск»үлгісіндегі отандық электронды-есептеу машиналарына бағытталған. БҰл уақытта осы машина орта өнімділік машиналары сыныбына жатқызылды және келесі сипаттамаларға ие:

Тезәрекеттілік .....секундына 5 - 6 мың операция

ЩЕҚ көлемі .....18 Кбайт

Магниттік дискіде сыртқы жинақтағыш

жадысының көлемі.....2 МБ

«Минск-2» машинасы 40 ... 50 $м^2$ ауданды иеленді.

Ақпараттық-логикалық тапсырмаларды өңдеу алгоритмдері бүгінгі күні реляциялық алгебра деп аталатын логикалық алгебра элементтеріне негізделген. Бағдарламалық тілдер ретінде

АЛГОЛ-60 тілдері және отандық мамандар әзірлеген АЛГЭМ қолданылды.

Осы бағдарламалық және аппараттық құралдарда келесі ақпараттық жүйелер жасалды:

• деректерді сұрыптау және қайта топтастыру процестеріне негізделген жазба алаптарын өңдеу. Кейбір объектіні немесе процесті сипаттайтын нақты белгіленген деректер жинағы жазба деп аталды;

• деректерді толтыру және жою, сұраныстар бойынша мәліметтерді іздестіру операцияларынан тұратын иерархиялық жіктеу жүйелеріндегі деректерді жинақтау және іздестіру;

• оларды жасудың негізгі принципі ақпаратты мағыналық кодтау болып табылатын фактографикалық деректерді жинақтау және талдау.

Мұндай ақпараттық жүйелер мен тиісті бағдарламалау тілдері қазіргі заманғы дерекқордың негізін құрайтын төрт негізгі бөлікті қамтиды:

• білімнің белгілі бір саласының негізгі мәндерін білдіретін терминдердің жиынтығы;

• пәндер арасындағы мағыналық қарым-қатынастардың жиынтығы;

• негізгі терминдер мен тіл қарым-қатынастарынан барынша күрделі терминдер мен қарым-қатынастар жасауға мүмкіндік беретін ресми ережелер жиынтығы (синтаксис);

• ақпаратты түрлендіру және сақтауды жүзеге асыруға мүмкіндік беретін түсініктерді, қарым-қатынастарды және тілдің синтаксистік ережелерінің жүйесін кодтау жүйесі.

Ақпараттық технологияларды дамытудың аталған кезеңінде шетелдік зерттеулердің арасынан, 1968 жылы IBM үлгісіндегі ЭЕҚ өнеркәсіптік ДҚБЖ қолданысқа енгізілген американдық IBM компаниясының атап өтеуге болады.

Дерекқор негізіндегі ақпараттық жүйелерді құру теориясы мен тәжірибесінде нақты серпіліс дербес компьютерлерлер дәуірімен және реляциялық дерекқорлар теориясын құрумен басталды.

Бұл кезеңнің басты ерекшелігі DOS дискілік операциялық жүйесімен дербес компьютерлердің пайда болуы және Dbase, FoxPro, Clipper тілдерінің тұқымдастарын бірегейлендіру элементтері бар дерекқорларды жобалау үшін арнайы тілдердің дамуы болып табылады.

ДҚБЖ-ны одан әрі дамыту SQL (Structured Query Language) сұраныстары құрылымдық тілінің негізіндегі бағдарламалық қамтамасыз етуді стандарттауға байланысты.

Қазіргі уақытта дерекқор бизнес процестерін, сондай-ақ күрделі ғылымды қажет ететін өнімдерді жобалау, өндіру және пайдалану үдерістерін жетілдірудің жаңа бағытының негізін құрайтын - өнімнің барлық тіршіліккезеңін біріктіретін бірыңғай ақпараттық кеңістік құру. Бұл бағыт CALS-технологиялар деп аталады.

CALS-технологиясының мәні, оны жобалаудан және пайдалану мерзіміа яқталғаннан бастап,бұйымның барлық кезеңі бойынша оңтайлы шешімдер қабылдау үшін қажет сенімді ақпарат алу мүмкіндігін беретін бірыңғай ақпараттық кеңістікті құруға алып келеді.

Қызметтің кез-келген саласында оңтайлы шешім қабылдау процесі ең алдымен рұқсат етілген шешімдердің аумағын табу үшін ірі көлемдегі ақпаратты алдын-ала талдау қажеттілігіне, ал содан кейін шектеулі аумақта бірыңғай, оңтайлы шешім табуға байланысты.

Тәжірибе көрсеткендей, ұтымды емес шешімдер қабылдау көбінесе маманда қажетті ақпараттың болмауына байланысты.

Бұл жағдайда дерекқорды және дерекқорды басқару жүйесін өндірісті басқаруды жетілдірудің стратегиялық құралы ретінде қарауға болады.

«Теріс техникалық шешімдердің нәтижелерін және салдарын жою емес, алғашқы қадамнан бастап оңтайлы шешім қабылдау» - заманауи кәсіпорындардың басқарушылық қызметінің ұраны. ДҚБЖ – ол басқарудағы табыс.

І-БӨЛІМ

ДЕРЕККОРДЫ ЖОБАЛАУ ТЕОРИЯСЫ

1-тарау

ДЕРЕККОР НЕГІЗІНДЕГІ АВТОМАТТАНДЫРЫЛҒАН АҚПАРАТТЫҚ ЖҮЙЕЛЕР

1.1. Дерекқор, дерекқорды басқару жүйелері

Заманауи өнеркәсіптік өндіріс пен бизнестің дамуы автоматтандырылған ақпараттық жүйелерді құрусыз мүмкін емес, оның мақсатының бірі – тұтынушыға оңтайлы шешім қабылдауға қажет сенімді ақпарат беру болып табылады.Қазіргі кезде өндіріс пен бизнесті басқарудың бірде-бір міндеті автоматтандырылған ақпараттық жүйелерді пайдаланусыз орындалмауы тиіс. Ол жаңа өнімдерді дайындаудың дизайны мен технологиясының нарығын және оларды жобалау; ол өндіруді, технологиялық процестерді және өнімді дайындау сапасының бақылау жүйесі.

Бүгін біз кез-келген маманның кез-келген қызметін шешім қабылдаудың кейбір жүйесі ретінде қарастыруымыз керек, сондықтан маманға сенімді ақпарат қажет. Осылайша, ақпараттық жүйенің маңызды функцияларының бірі - басқару үдерісін ақпараттық қолдау болып табылады. Бұл жүйелердібасқарушы ақпараттық жүйелер деп атайды (management information systems); олар әдетте ірі және күрделі дерекқорларды қамтиды.

Сонымен, *Дерекқорды* және *Дерекқорды басқару жүйесі* дегеніміз не? Өкінішке орай, осы ақпараттық технологиялардың бағыты бойынша кітаптардың көпшілігінде айқын анықтамалар жоқ. Олардың кейбірін қарастырайық.

Т.С. Қарпованың [6] оқу құралында келесі анықтамалар берілген.

Дерекқор (ДҚ) - объектілердің жай-күйін және олардың қаралатын пәндік саладағы қарым-қатынасын көрсететін белгілі деректер жиынтығы.

Дерекқорды басқару жүйесі - көптеген пайдаланушылармен ДҚ жасау, жүргізу және бірлесіп қолдануға арналған тіл және бағдарламалық құралдар жиынтығы.

Күнделікті ДҚБЖ Mcrosoft Access 2000 нұсқаулығында авторлары И.А. Харитонова және В.Д. Михеев [13] келесі анықтамаларды ұсынады.

Дерекқор - белгілі бір тақырыпқа немесе тапсырмаға жататын мәліметтердің (нақты объектілер, процестер, оқиғалар немесе құбылыстар туралы) жиынтығы, ол жиынтықты тұтастай, сондай-ақ кез-келген бөлігінде ұсынудың қолайлылығын қамтамасыз ететіндей ұйымдастырылған.

Дерекқорды басқару жүйесінің негізгі функциялары– ол деректерді анықтау (дерекқордың құрылымын сипаттау), деректерді өңдеу және деректерді басқару.

ДҚ-ның екінші айқындауышының маңызды айырмашылығына, атап айтқанда ДҚ-да ақпаратты ерекше ұйымдастырудың сипаттамасына назар аударайық.

Дерекқордың және ДҚБЖ тұжырымдамалары біздің көзқарасымыз бойынша ең көп дәрежеде Есептеу жүйелерінің түсіндірме сөздігінде анықталған [11].

Дерекқор, әдетте, сөздің қатаң мағынасында, дерекқорды басқару құралдарын қолданатын деректерді анықтау және қолдануға арналған деректер файлы. Ол, ең алдымен, бұл файл оған қол жеткізетін бағдарламаларға тәуелді емес сызба арқылы анықталатынын, ал екіншіден, ол тікелей қол жеткізетін есте сақтау қондырғысы түрінде іске асырылғанын білдіреді.

Аталған анықтамаларды ескере отырып, *Дерекқор* - тікелей қолжетімділікпен ұйымдастырылған (құрылымдалған) файл ретіндегі файл.

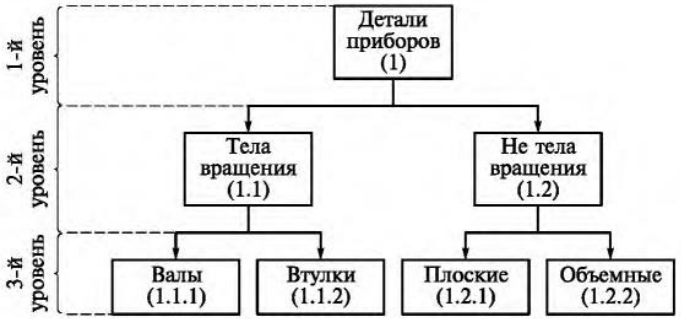
Тікелей қолжетімділікті файлдың түсінігі және оны ұйымдастыру принциптері көп өлшемді алаптарды бағдарламалау теориясынан, соның ішінде Basic, Pascal алгоритмдік тілдерінен белгілі.

Есептеу жүйелерінің түсіндірме сөздігінде [11] келесі ДҚБЖ ұғымдары берілген.

Дерекқорды басқару жүйесі - дерекқор тілінде өңдеу құралдарынан тұратын бағдарламалық қамтамасыз ету жүйесі, ол қолданбалы бағдарламалардан және (немесе) соңғы пайдаланушылардан дерекқорға келетін байланыстарды өңдеуді және дерекқордың тұтастығын қамтамасыз ету мүмкіндігін береді.

Әдетте, дерекқорды ұйымдастырудың үш сыныбы (модельдері) ерекшеленеді: иерархиялық, желілік және реляциялық. Бұл жағдайда «модель» термині деректерді сақтауды және деректерге қолжетімділікті логикалық деңгейде сапалы және сапалы бағалауға мүмкіндік беретін құрылым ретінде қарастырылады (мысалы, деректерді сақтау үшін жадқа деген болжамды қажеттілікті есептеу немесе іздестіру қадамдарының қажетті санын есептеу).

Деректердің иерархиялық моделі. Деректердің иерархиялық моделі, яғни атауынан байқағандай иерархиялық құрылымға ие, яғни әрбір элемент тек бір жоғары элементпен байланысты, бірақ сонымен бірге оған төмендегі бір немесе бірнеше элементтер сілтеме жасай алады.

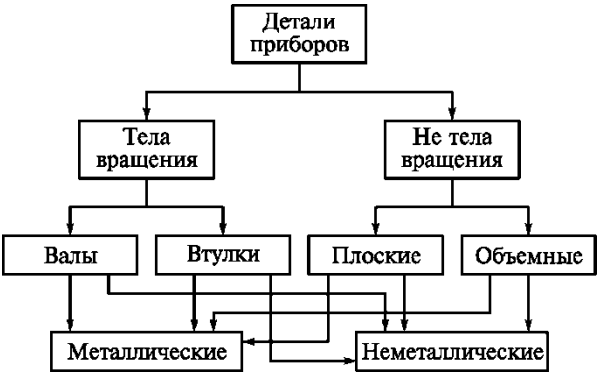


1.1-сур. Деректердің иерархиялық моделі

Иерархиялық модельдің терминологиясында нақты: «элемент» (түйін); «деңгей» және «байланыс» түсініктері қолданылады. *Түйін* көбінесе объекті сипаттайтын атрибут (белгі) болып табылады. Иерархиялық модель граф түрінде жүйелі бейнеленеді, онда әр түйін шың болып табылады. Осы модель өзімен олардың жеке немесе баған - иерархиялық құрылымы бар ағаш түзетін бағыныстылығына байланысты орналасқан элементтер жиынтығын құрайды (1.1-сурет). Мұндай граф кез-келген басқа шыңға бағынбайтын және ең жоғарғы (бірінші) деңгейде орналасқан бірегей шыңға ие. Бірінші деңгейдегі шыңдар саны дерекқордағы ағаштардың санын анықтайды.

Деректердің желілік моделі. Бұл модель иерархиялық модель сияқты: «түйін», «деңгей» және «байланыс» терминологияларын пайдаланады

Деректердің иерархиялық және желілік үлгілері арасындағы жалғыз айырмашылық, бұл деректердің әрбір элементінің (түйін) кез-келген басқа элементпен (түйін) байланыстырылу мүмкіндігінен тұрады (1.2-сур.).



1.2-сур. Деректердің желілік моделі

Кесте 1.1. Құрал бөлшектері

Коды	Беттерінің орналасуы	Қосымша сипаттамасы
1	Айналу денелері	Біліктер
2	Айналу денелері	Тығындар
3	Айналу денелері емес	Жазық
4	Айналу денелері емес	Көлемді

Граф теориясынан мәлім болғандай, желілік граф граф-ағашына түрлендірілуі мүмкін.

Деректердің реляциялық моделі. Деректердің реляциялық моделі негізгі идеясы - екіөлшемді алап түрінде деректердің кез-келген жиынтығын – кестені көрсетуден тұрады. Қарапайым жағдайда реляциялық модель жалғыз екіөлшемді кестені сипаттайды (1.1-кесте), бірақ көбінесе бұл модель бірнеше түрлі кестелер арасындағы құрылым мен қатынастарды сипаттайды. 1.3-суретте көрсетілгендей 1.1-кестеде кестенің екі байланысқан кестеге бөлінуі көрсетілген.

Коды	Беттерінің орналасуы
1	Айналу денелері
2	Айналу денелері емес

Коды	Қосымша сипаттама
1.1	Біліктер
1.2	Тығындар
2.1	Жазық
2.2	Көлемді

1.3-сур. Деректердің реляциялық моделінің екі байланысқан кестелері

Деректердің реляциялық модельдері немесе реляциялық деректерқор қазіргі уақытта өндірістегі және бизнестегі ақпараттық жүйелерді жобалау мен ұйымдастырудың негізгі әдісі болып табылады, сондықтан осы оқулықта біз реляциялық деректер базасын дамытудың теориялық негіздері мен практикалық әдістерін қарастырамыз.

1.2. Реляциялық алгебраның негізі

Реляциялық деректер базасында ақпаратты өңдеу әдістері мен алгоритмдері бұрын логика алгебрасы немесе бульдік алгебра (бұдан әрі осы атауларды қолданатын боламыз) деп аталатын реляциялық алгебраға негізделеді.

Логика алгебрасы өзімен ең алдымен пікір айтудан тұрады. Логика алгебрасы деп шын немесе жалған болып келетін кез-келген ұсыныс түсініледі:

бұл ретте көрсетілген екі мәннің тек біреуі ғана жоғарыдағы мағынаға ие бола алады (мысалы: «Мәскеу – Ресей астанасы» - бұл шынайы сөз; «Қар қара» - жалған мәлімдеме). Жекелеген мәлімдемелерлогика алгебрасында қандай да болмасын әліпби әріптерімен белгіленеді,мысалы: А, В, С. Ақиқат немесе жалған пікірлер,олардың шындық құндылықтары деп аталады. Логика алгебрасында логика қабылданады

1 санымен жасалған пікір шынайылығын анықтау және жалған пікірді 0 санымен қабылдау белгіленген. А = 1 және С = 0 белгілері А-ның шын және С – жалған екенін білдіреді. Әрбір нақты пікір барнақты ақиқат мағынасына ие, ол – тұрақты, 0 немесе 1-ге тең. Нақты (тұрақты) пікірден ауыспалы пікір деп аталатындарды айырған дұрыс.

Ауыспалы пікір – ол нақты мағынасында айтуды емес, айту үшін ауыспалы (ұсынымды ауыспалы), яғни, оның орнына ауыспалы пікірді (немесе олардың атауын) қоюға болатын және «шын» немесе «жалған» немесе сәйкес1 және 0 (екілік ауыспалы) екі мағынаны ғана қабылдай алатынсимвол. Ауыспалы пікірлер (яғни ұсынымды ауыспалы) пікір белгіленетін әріптерден ерекшеленетін әріптермен белгіленеді. Алгебрада ауыспалы пікірлерді қолданулогика алгебрасында жалпылама пікір үшін қызмет етеді. Ол кез-келген пікірлер үшін логика алгебрасының заңдарын тұжырымдауға мүмкіндік береді.

Логика алгебрасындағы зерттеу мәні бинарлық (немесеекі мәнді) функциялар, яғни тек екі мәнді ғана қабылдайтын («шын» немесе «жалған», 0 немесе 1) және бір немесе бірнеше бинарлық ауыспалықтарға байланысты функциялар болып табылады. Ол бульдік функциялар деп аталады.

Қарапайым ретінде қабылданатын бір немесе бірнеше мәлімдемелерден, қарапайым сөйлемдердің бинарлық функциялары болатын күрделі пікірді жасауға болады. Логикада алгебрасында қарапайым сөйлемдерді күрделіге біріктіру осы сөйлемдердің ішкі мазмұнын (мәнін)есепке алмастан жүргізіледі. Белгілі логикалық операциялар (немесе логикалық байланыстар) қолданылады, оларкейбір пікірлерді (тұрақты немесе ауыспалы) барынша күрделіге (тұрақты немесе ауыспалыға) біріктіруге мүмкіндік береді.Негізгі логикалық операцияларға: терістеу, конъюнктура, дизъюнкция, эквиваленттік, импликация жатады. Логикалық операциялар кесте түрінде, сондай-ақ қарапайым сөздердің функциясы ретінде беріледі. 1.2, 1.3-кестелерде терістеу және конъюнкция операциялары үшін маңызы бар мағыналар берілген.

А	А
1	0
0	1

Кесте1.2.

Терістеу операциясының шынайылық мағынасы

Кесте 1.3. Конъюнкция операциясының шынайылық мағынасы

Л	В	А^В
1	1	1
0	1	0
1	0	0
0	0	0

Кесте 1.4. Дизъюнкция операциясының шынайылық мағынасы

Л	В	Л vВ
1	1	1
0	1	1
1	0	1
0	0	0

Пікірді терістеу. А - А жалған болғандағы шын пікір және А дұрыс болғандағы жалған пікір, А деп белгіленеді; оқығанда: «А емес».

Екі пікірдің конъюнкциясы. Екі пікірдің конъюнкциясы – ол күрделі мәлімдеме, оны қалыптастыратын екі сөйлемнің ақиқатында дұрыс және басқа жағдайларда жалған болып табылады. Екі мәлімдеме конъюнкциясы: А^В деп белгіленеді; «А және В» деп оқылады. Конъюнкция белгісі «^» «және» жалғаулығының мағынасынан тұрады. Конъюнкция операциясы логикалық көбейтудің мағынасына ие және «&» белгісімен белгіленуі мүмкін.

Екі пікірдің дизъюнкциясы. Екі пікірдің дизъюнкциясы - оның екі құрамдас сөздерінің жалғандығы жағдайында жалған болып табылатын және басқа жағдайларда ақиқат болып келетін күрделі пікір.

Дизъюнкция операциясы: А v В деп белгіленеді; «А немесе В» деп оқылады (басқа белгісі: А + В, басқа атауы - «логикалық қосу»).

«v» логикалық байланысының белгісі «немесе» жалғаулық мағынасына ие және дизъюнкция белгісі деп аталады. «Немесе» жалғаулығы бірнеше түрлі мағыналарда қолданылады. «v» белгісі пайдаланылатын «немесе» мағынасынан тұруы мүмкін, мысалы, сөз: «Петя немесе Иван қоңырау дауысы кезінде оянады» (мұнда «немесе» сөзі екеуінің оянатынын жоққа шығармайды). Аталған жағдайда дизъюнкция «немесе» бөлінбейтін мағынасына ие. Сондай-ақ, логикалық операциялар түрі ретінде қабылдануы мүмкін, бірақ оны дизъюнкциямен шатастыруға болмайтын, «немесе» жалғаулықты жоятынмағына бар (мысалы: «Оны не мені таңдаңыз»). Дизъюнкция 1.4-кестеде берілген.

Екі пікірдің эквиваленттілігі. Екі пікірдің эквиваленттілігі–ол пікірді білдіретін ақиқат мағынасы бірдей болғанда шындық және кері жағдайда жалған болатын күрделі пікір болып табылады; А = В болып белгіленеді; «АВ-ға барабар» оқылады. Эквиваленттілік үшін келесі жолмен жазуға болатын пікірі әділ:

Кесте 1.5. Эквиваленттілік операциясының шынайылық мағынасы

A	B	A = B
1	1	1
0	1	0
1	0	0
0	0	1

Кесте 1.6. Эквиваленттікті терістеу операциясының шынайылық мағынасы

A	B	A = B
1	1	0
0	1	1
1	0	1
0	0	0

A = 1 = A, ол: A жалған болған кезде A-ның бірлікке барабарлығын білдіреді.

A = 0 = A жазбасы, A жалған болған кезде A-ның нөлге барабарлығын білдіреді. 1.5-кестеде эквиваленттілік операциясының шынайылық мағынасы берілген.

Эквиваленттілікті терістеу. Екі пікірдің эквиваленттілігін білдіретін пікірді терістеу әрекетін қолдана отырып, эквиваленттікті терістеу деп аталатын, A = B жаңа күрделі пікір аламыз. Эквиваленттілікті терістеу A-ның B-ға эквивалентті емес екенін білдіреді. Эквиваленттілікті терістеу 1.6-кестеде берілген.

Осы операция ЭЕМ жобалау теориясында өте маңызды, өйткені ол өзімен екі модуль бойынша күрделі қосарлы санды білдіреді.

Екі пікірдің импликациясы. Екі пікірдің импликациясы: A з B болып белгіленеді; «егерде A, онда B» болып оқылады. Ол A ақиқат, ал B жалған болған жағдайда ғана жалған болып келетін күрделі пікір. Осы пікірдің ақиқаттылығы мағынасы 1.7-кестеде берілген.

Импликация A жағдайы мен B салдары арасындағы мағына бойынша кері байланысты болжамайды, дегенмен мұндай байланысты жоққа шығармайды. Импликация мағынасын: «A жалған немесе B ақиқат» түрінде білдіруге болады. Осы пікірде «немесе» жалғаулығы жойылмайтын мағынаға ие.

1.7-кесте

Импликация операциясының шынайылық мағынасы

A	B	A ⊃ B
1	1	1
0	1	1
1	0	0
0	0	1

Жоғарыда көрсетілген логикалық операциялар арқылы қарапайым пікірден алынғанкез-келген күрделі пікір, *логика алгебрасының формуласы* деп аталатын болады.

Қарапайым пікірлердің екі формуласы A_1A_2, ..., A_n әрбір комбинация үшін пікірдің шынайылығы мағынасы $A_1, A_2,,, A_n$ логика алгебрасының екі формуласы бірдей шынайылық мағынасы ие болған жағдайда барабар деп аталады. Қарайпайым пікірдің п шынайылық мағынасының әртүрлі комбинациясында 2^n дәлдігі бар және осы 2n комбинацияларының әрқайсысы үшін күрделі пікірі шынайы немесе жалған болған жағдайда, онда 2^{2n} қарайпайым пікірдің п деректерінен жасалған логика алгебрасының әр түрлі функциялары болуы мүмкін. А, В (яғни екі ауыспалы пікір) екі қосарлы ауыспалылық жағдайында логика алгебрасының 16 әртүрлі функцияларын жасауға болады, яғни, 16 күрделі логикалық өрнектерді бір-біріне теңдестіру керек. Осы өрнектердің арасында жоғарыда сипатталған барлық логикалық байланыстар (бір ауыспалы функция болып табылатын терістеуден басқа) бар.

Логикадағы алгебрасының негізгі заңдарды білдіретін келесі тең арасалмақтар логика алгебрасында маңызды рөл атқарады:

1) A = A жалған болған кезде А-ның ақиқат екенін А білдіреді;
2) A ^ B = B^A (А және В= В және А) екенін білдіреді;
3) (A ^B) ^C = A ^(B ^C), ((А және В) және С) = (А және (В және С)) немесе, басқаша, логикалық көбейту заңын білдіреді ((A- B) • C) = (A -(B - C));
4) AvB = B vA, A+ B= B+ A—арифметика курсының белгілі заңы екенін: «қосылатын сома орын ауыстырғаннан өзгермейді» білдіреді;
5) (AvB) vC = Av(B vC) немесе (A+ B) + C = A+ (B + C);
6) A^ (B vC) = (A^ B) v (A^ C) оң жақ бөліктің ақиқаттылығы кезінде сол жақ бөліктің ақиқаттылығын білдіреді;
7) Av(B ^C) = (AvB) ^(AvC);
8) (A v B) = A ^ B ;
9) (A ^B) = A v B;
10) A ^A = A;
11) A v A = A;
12) A ^1 = A;
13) A v 0 = A.

Көрсетілген арақатынастардың әділдігі логикалық операциялардың - конъюнкция, дизъюнкция және терістеудің анықтамалары жәнекестелерінің негізінде жүргізілуі мүмкін. 1 - 13 арасалмағы күрделі логикалық өрнектерді қарапайым пішімге түрлендіру үшін пайдаланылады.2 – 5 арақатынастары, конъюнкция және дизъюнкция операциялары үшін ауыстырыдымдылық және үйлесу заңдары әділ, соның арқасында көптеген конъюнкция және дизъюнкцияны жақшасыз жазуға болады. Мысалы: [(A^B) ^C] ^D орнына A^B ^C^D жазуға болады.

Логикалық формулалардағы жақшалардың санын одан әрі азайту үшін біз «V» белгісінің көмегіне қарағанда, «^» белгісінің көмегімен байланысты есептеуді қабылдадық, соңғысы «≡», «≠» және «∩» белгілерінің көмегіне қарағанда тығыз.A ^B ^C ^D... түрінің өрнектерін туынды, ал оның мүшелерін – көбейткіштер деп атайды.
A v B v C v ... формасының өрнектері сома деп аталады, ал оның мүшелері – қосылғыштар деп аталады. 6 арақатынасы логика алгебрасындағы дизъюнкцияға қатысты конъюнкцияның үлестірімділік заңының әділ екенін көрсетеді. Оның A^(B + C) = A^B + A^C (мұнда нүкте логикалық көбейтуді, ал «қосу» белгісі логикалық қосуды білдіреді) түрдегі жазбасы осы заңмен және кәдімгі арифметикадағы қосуға қатысты көбейтудің үлестірімділік заңы арасындағы ұқсастықты айқын көрсетеді. Бірақ логика алгебрасындағы арифметикадан өзгеше, 7 арақатынасы арқылы көрінетін конъюнкцияға қатысты дизъюнкцияның үлестірімділік заңы да орын алады. Екі үлестірімділік заңдар да логика алгебрасы формулалары арқылы жақшалардың ашылуының түрлендіруге және қарапайым алгебрада орындалатындай көбейткіштерді шығаруға (сонымен қатар, жалпы қосылыстарды шығаруға) мүмкіндік береді.

8 және 9 арақатынастар Морган заңдары деп аталады және 1-арақатынаспен бірге теріс белгілері қарапайым пікірге ғана жататын түрге логикалық көріністі түрлендіру мүмкіндігін береді.

1-13 арақатынастардан өзге, келесі тең арақатынастар логикалық өрнектерді түрлендіру үшін өте пайдалы:

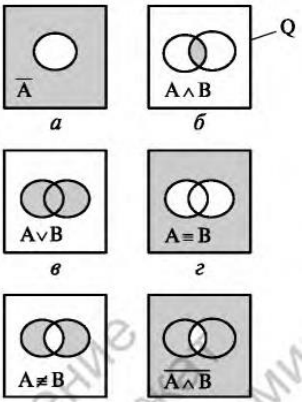
14) A v A ^B = A v B;
15) A ^(A v B) = A v B;
16) A v A ^B = A;
17) A ^B v A ^C= A ^B v A ^C ^B ^C;
18) A ^(A v B) = A;
19) (A v B) ^(A v C) = (A v B) ^(A v C) ^(B v C);
20) A v A ^B = A v B;
21) A v (A v B) = Av B;
22) A ∩B = A v B;
23) A = B = A ^B v A ^B.

Соңғы екі арақатынасы пайдалану тек қана «v», «^», «¯» белгілерінен тұратын «∩» және «≡» белгілерінен тұратын өрнектерге кез-келген өрнекті келтіруге мүмкіндік береді.

Егер A, B, C, ... ауыспалылары ретінде пікірді ғана емес,ол үшін 1-13 арақатынастарды қанағаттандыратын қосу, көбейту және терістеу әрекеттері анықталғанпікірді ғана емес, кез-келген жүйе элементтерін түсінетін боламыз, онда Буль алгебрасы деп аталатын абстрактылы алгебраны аламыз.

Атап айтқанда, пікірлер және негізгі логикалық операциялар «V», «^», « ¯ » өзімен жеке жағдайды немесе интерпретацияны, алгебраны білдіреді. Буль алгебрасының тағы бір мысалы сыныптар алгебрасы болып табылады, ол негізгі логикалық операциялар үшін көрнекі геометриялық түсініктемені береді.

А пікірін қарастырайық, мұнда мәселе *а* кейбір қасиеттерінің қандай да бір сала пәніне тиесілігі туралы. Осы саладағы пәндердің жазықтық бөлігінің нүктелерімен кескінделетінін, кейбір шаршылармен (1.4-сурет) шектелгенін оны біз Q арқылы белгілейтінімізді көзімізге елестетейік.



1.4-сур. Буль функциясының графикалық кескіні:
а — жиынтық А; б — жиынтықтар қиылысы; *в* — жиынтықтардың бірігуі; г — екі жиынтықтың эквиваленттілігі; *д* —эквиваленттілікті терістеу; е —конъюнкцияны терістеу

Q жазықтық нүктелерінің екі сыныпқа (жиынтық) бөлінетіні анық: *а* қасиетіне ие, яғни А = 1 нүкте сыныбы және осы қасиетке ие емес, яғни А = 0 үшін нүктелер сыныбына ие, мұндағы Q жазықтығының әрбір нүктесі осы сыныптардың тек біреуіне ғана міндетті тиесілі.

Бірінші сыныпты А пікірінің геометриялық кескіні немесе А жиынтығы деп есептеуге болады. Бұл ретте, мысалы, суретте көрсетілгендей 1.4-суретте берілгендей *а* көрініс алынуы мүмкін, онда А кескіні тұйық (құйылған) контурмен шектелген белгілі бір аймақ түрінде кескінделген.Әлбетте, А (А емес) өрнегі Q шаршысының қалған нүктелерінің жиынтығымен бейнеленетін болады. Осындай интерпретация кезінде логикалық операциялардың негізін ұсынуға болады.

Екі пікірдің конъюнкциясыекі жиынтықтың қиылысымен ұсынылатын болады (1.4, б-сур). Шындығында, А ^ В = 1 болғанда, А = 1 және В = 1 болады, ол А жиынтығына және В жиынтығына (олардың қиылысына) біруақытта тиесілі нүктелер үшін орын алады.

Екі пікірдің дизъюнкциясы А v В, А және В (1.4, в-сур) жиынтықтарын бірлестіру жолымен алынатын жиынтықпен кескінделетін болады.

Екі пікірдің эквиваленттілігі А = В 1.4, г-суретте көрсетілгендей бейнеленетін болады, өйткені А = В айқындылығы 1 тең немесе А = 1 және В = 1, немесе А = 0 және В = 0 кезіндегідей.

Эквиваленттілікті терістеу 1.4, *д*-суретте көрсетілген $\underline{A =}$В-ның, А = В-ның А = В тең болуын ескерген кезде алынады.

Кесте 1.8.

Логикалық операцияларды белгілеу нұсқалары

Конъюнкция (логикалық көбейту)	Дизъюнкция (логикалық қосу)	Терістеу (логикалық терістеу)	Эквиваленттілік	Импликация
ЖӘНЕ	НЕМЕСЕ	НЕ		Егерде... онда
AND	OR	NOT		If... then
&	|	¬	→	≡
^	v	$\overline{A}$	⊃	↔

1.4, е-сур. көрсетілгендей A ^B конъюнкциясын терістеу.

Венн диаграммалары деп аталатын осындай диаграммалар, оларды талдау және оңтайландыру мақсатында логикалық формаларды көрнекілеп ұсыну үшін қолданылуы мүмкін.

Қаралған логикалық операциялар — конъюнкция, дизъюнкция, терістеу — тәуелсіз болып табылады және бір-бірі арқылы көрсетілуі мүмкін. Атап айтқанда, олардан логикалық операциялардың жүйелерін бөлуге болады және олардың көмегімен логика алгебрасының барлық функцияларын көрсете алады. Логикалық операциялардың мұндай жүйелері (кейде 1 немесе 0 константаларымен бірлесіп) функционалды толық деп аталады.

1.8-кестеде логикалық операцияларды белгілеудің әртүрлі нұсқалары берілген.

Жоғарыда тізімделген логика алгебрасының негізгі операцияларынан басқа, екі операцияы: тиісті пікірлердің үйлесімсіздігін бекітетін конъюнкцияны терістеу және дизъюнкцияны терістеу бар.

Конъюнкцияны терістеу A ^ B байланыстары: A / B және Шеффер операциясы деп аталады.

Дизъюнкцияны терістеу A v B, Шеффердің сызықшасымен көрсетілген A = A / B мағынасына ие.

Шеффер операциясы ЭЕМ процессорларының логикалық сұлбаларын жобалау теориясында маңызды рөл атқарады, өйткені Шефер операциясын іске асыратын электронды сызба әмбебап функционалды элемент болып табылады, оның көмегімен кез-келген функционалды логикалық құрылғылар құрылуы мүмкін. Шеффер операциясының графикалық көрінісі 1.4, е-суретте берілген.

Конъюнкция және дизъюнкция операциялары қосарлы деп аталады және ^-дыv-ғажәне v-ны^-ға басқа ауыстырулардың бірінен алынған логика алгебрасының формулалары қосарлы деп аталады. F және F * қосарлы формулалары үшін пікірлердің теңдігі әліл болып табылады.

$$F(A_1, A_2, \dots A_n) = F^*(A_1, A_2, \dots A_n) \qquad (1)$$

Логика алгебрасында келесі қосарлық принципі белгіленеді: егер F және Ф формулалары тең болатын болса, онда F* және Ф* қосарлы формулаларытең болады.

Логика алгебрасы формулаларының құрылымы қалыпты пішіннің екеуінің біріне дейін келтірілген кезде айқын көрінеді. Алғашқы қалыпты пішін - конъюнктивті (КНФ) – өзімен бірге дизъюнкция конъюнкциясын білдіреді, мұндағы әрбір дизъюнкцияның әрбір жекелеген мүшелері өзімен қарапайым пікірді (яғни, өзімен басқа пікірлерден тұрмайтын пікірлер) немесе қарапайым пікірлерді терістеуден тұрады. Екінші қалыпты пішін - дизъюнктивті (ДНФ) – өзімен бірге конъюнкция дизъюнкциясын білдіреді, мұндағы әрбір конъюнкцияның әрбір жекелеген мүшелері қарапайым пікір немесе оларды терістеу болып табылады.

Қалыпты пішіндер формулалардың екі маңызды сыныбын ажыратуға ыңғайлы: тұрақты-шынайы және тұрақты-жалған.

Тұрақты-шынайы формулалар үнемі 1-ге тең (олар 1-ге сәйкес келеді).

Тұрақты-жалған формулалар үнемі 0-ге тең (0-ге сәйкес келеді).

Формулалардың осы сыныптары логикалық өрнектерді жеңілдетуде маңызды рөл атқарады

Күрделі формулаларды жеңілдету кезінде, A ^A 1 = A және A v 0 = A пікірлерінің теңдігін қолдана отырып, тұрақты-шынайы жәнетұрақты-жалған пікірлерден бас тарта аламыз, алA ^0 = 0 және A v 1 = 1 теңдігін пайдалана отырып тұрақты-жалған пікірге конъюнктивті қосылған және тұрақты-шынайы пікірге дизъюнктивті қосылған пікірлерден бас тартуға болады.

Күрделі формуланың тұрақты-шынайылығы туралы мәлімдеме келесі ережелерді қолдану арқылы алынуы мүмкін:

1) A v A формуласы тұрақты-шынайы;

2) A v B формуласы шынайы, егерде A шынайы болатын болса, ал B — туынды пікір;

3) A ^B формуласы шынайы, егердеA және B шынайы болатын болса.

Осы ережелерді қолдану күрделі формуланың тұрақты шынайылығының келесі өлшем шарттарын шығаруға мүмкіндік береді.

Осындай формулалар әрдайым шындық болып табылады, олардың әрқайсысында КНФ-де кемінде бір негізгі мәлімдеме бар, оны жоққа шығарады.Шынында да, бұл жағдайда әрбір ажырату кем дегенде бір нақты термин болады, яғни КНФ-ке мүше болып табылатын барлық келіспеушіліктер шындықты білдіреді; логика алгебрасының берілген формуласын білдіретін барлық КНФ шын болады.

КНФ-ның әрбір дизъюнкциясына кем дегенде бір негізгі пікір өзінің терістеуімен кіретін осындай формулалар тұрақты-шынайы болып табылады. Ұқсас жолмен ДНФ келтіру арқылы логика алгебрасының берілген формуласының тұрақты-жалған екенін немесе ондай емес екенін анықтауға болады.

Шынында да, бұл жағдайда әрбір дизъюнкцияда кем дегенде бір нақты мүше болады, демек, КНФ мүше болып табылатын барлық дизъюнкциялар шынайы болып табылады, яғни логика алгебрасының берілген формуласын білдіретін барлық КНФ шынай болады.

Егерде осы формуланың ДНФ-інде дизъюнктивті байланысқан әрбір конъюнкциясында, кем дегенде өзінің терістеуімен бір пікір болатын болса формула тұрақты-жалған болады,

Логика алгебрасына кіретін ауыспалы кейбір мағыналары кезінде шынайы болып табылатын формулалары орындалатын деп аталады. Тұрақты-жалған болып табылмайтын кез-келген формулалар орындалатын болады.

Логика алгебрасының кез-келген функциясын ұсынудың әмбебап әдісі $F(A_1, A_2, ..., A_n)$ түрдің барлық конъюнкциясының дизъюнкциясы түрінде болады:

$$F(a_1, a_2, ..., a_n)^ A_1'^ A_2'^ ... ^A_n' \qquad (1.1.)$$

мұндағы $a_1, a_2, ..., a_n$—0 және 1 мағыналарынан жинақ, aA_n' = $A_n a_n$ = 1кезінде; a_n = 0 кезінде $A_n' = A_n$.

Шындығында, кез-келген жинақ үшін $a_1, a_2, ..., a_n$ түрдің бір конъюнкциясы ғана табылады (1.1), ондағы тұрақты көбейткіш $F^a_1, a_2, ..., a_n)$ аталған жинақ кезіндегі аталған функциядағы аталған шынайылық мағынасына ие болады, ал қалған көбейткіштер $A_1'^ A_2'^ ... ^A_n'$ 1-ге тең болады. Осы конъюнкцияда A_n ауыспалысындағы терістеу белгілерін тарату, $a_1, a_2, ..., a_n$ жинағындағы нөлдік таратуға сәйкес келетін болады. Дизъюнкцияда тұрақты көбейткіштері 1-ге тең конъюнкцияларды ғана қалдыра отырып және A^ 1 = A ережесі бойынша конъюнкциядан жоя отырып, аталған функцияны $^A_1'^A_2'^ ... ^A_n'$ конъюнкциясы түріндегі дизъюнкция нысанындағы аталған функцияны білдіретін формуланы аламыз. Осы форма дизъюнктивті жетілген форма (ДСНФ) деп аталады жәнекелесі қасиеттерге ие:

• бірдей қосылғыштарға ие емес;

• әрбір қосылғыш көбейткіштер ретінде немесе негізгі ауыспалылар немесе оларды терістеу ретінде тұрады;

• бірде-бір қосылғыштарда екі бірдей көбейткіштер жоқ және терістеуімен бірге ауыспалылардан тұрмайды.

Осыған ұқсас, ұқсас жолмен қанағаттандыратын дизъюнкция конъюнкциясынан тұратын мінсіз қалыпты пішін (КСНФ) анықталады.

Жетілген қалыпты нысандар логика алгебрасының күрделі формулаларының теңкүштілігі туралы сұрақтарды шешу үшін пайдаланылуы мүмкін: логика алгебрасының екі формуласы бірдей жетілген қалыпты пішіндерге келтірілетін болса теңкүшті болып табылады. Осы нысандарды тәжірибелік қолдану олардың ірілігіне байланысты күрделі, сондықтан іс жүзінде шағын қалыпты пішіндер деп аталатындары жиі қолданылады.

*Шағын дизъюнктивті қалыпты нысан (*МДНФ*)* өзімен конъюнкция дизъюнкциясын білдіреді, онда:

• бірде-бір қосылғышта қайталанатын көбейткіштер жоқ;

• бірдей көбейткіштері болатындай қосылғыштар жұбы жоқ;

• тікелей түрде бір қосылғышқа кіретін, ал басқа қосылғышқа терістеу түрінде кіретін, бір ортақ ауыспалыға ие кез-келген екі қосылғыш үшін, бір жалпы ауыспалы бар, алғашқы екі қосылғыштың қалған көбетйкіштерінің конъюнкциясынан тұратын үшінші қосылғыш бар.

Логика алгебрасының қолдануының маңызды саласы ақпаратты өңдеу процестерін алгоритмдеу және реляциялық дерекқорларды басқару болып табылады.

Бақылау сұрақтары

1. «Дерекқор» және «Дерекқорды басқару жүйесі» ұғымдарына анықтама беріңіз.

2. Дерекқорды ұйымдастырудың иерархиялық, желілік және реляциялық модельдеріне сипаттама беріңіз.

3. *Логика алгебрасы* дегеніміз не?

4. Негізгі логикалық операциялардың анықтамаларын беріңіз: терістеу, конъюнкция, дизъюнкция, эквиваленттік, импликация.

5. Әрбір негізгі логикалық операциялар үшін шынайылық кестелерін жасаңыз.

6. Келесі арасалмақтардытағы қалай жазуға болады:

A ^B = B^A;
(A ^B) ^C = A ^(B ^C);
AvB = B vA;
(AvB) vC = Av(B vC);
A^(B vC) = (A^B) v (A^C).

1. Логика алгебрасының келесі заңдарының оң бөліктерін жазыңыз:

Av(B ^C) =
A vB =
A ^B =
A^A=
A v A =
A ^1 =
A v 0 =

8. Морган заңдары нені белгілейді?

9. Логикалық операцияларды белгілеу нұсқаларын тізіңіз: терістеу, конъюнкция, дизъюнкция, эквиваленттік, импликация.

10. Шеффер операциясы немесе Шеффер штрихы деп не аталады?

РЕЛЯЦИЯЛЫҚ ДЕРЕККОР

2.1. Терминдер және анықтамалар

Реляциялық дерекқорларды дамыту 1960-шы жылдардың соңында басталды, онда кестелер түрінде ресми деректерді ұсынудың әдеттегі әдістерін пайдалану мүмкіндіктері талқыланғаналғашқы жұмыстар пайда болды. Кейбір мамандар ақпаратты ұсынудың осы әдісін шешім кестелері, басқалары - кестелік алгоритмдер деп атады. Реляциялық дерекқорлар теорияшылары ақпаратты ұсынудың кестелік әдісін даталогиялық модельдер деп атады.

Реляциялық дерекқорлар теориясының негізін қалаушы болып IBM қызметкері Э.Ф. Кодд 1970 жылғы 6 маусымда «Ірі коллекторлық деректер банкі үшін реляциялық деректер моделі» («A Relational Model of Data for Large Shared Data Banks»мақаласын жариялады. Бұл мақалада алғаш рет «деректердің реляциялық моделі» пайдаланылып, ол реляциялық дерекқорлар бастамасына жол ашты.

1970 жылдары АҚШ-та доктор Э.Ф. Кодд әзірлеген реляциялық дерекқорлар теориясы, көбейткіштер теориясының математикалық аппаратына сүйенді. Ол деректердің кез-келген жиынтығын математикада қарым-қатынас ретінде белгілі ерекше түрдегі екі өлшемді кестелер түрінде ұсынылуы мүмкін екенін дәлелдеді. Ағылшынша «relation» («қатынас») деген сөзден «реляциялық деректер моделінің» атауы туындады. Қазіргі уақытта дерекқорды (ДҚ) жобалаудың теориялық негізі реляциялық алгебраның математикалық негізінен тұрады (1.2-бөлімді қараңыз).

Осылайша, редяциялық ДҚ белгілі бір байланыстармен біріктірілген екі өлшемді алаптар – кестелер түрінде ұсынылған объектілер туралы ақпараттан (деректерден) тұрады. Дерекқор бір кестеден тұруы мүмкін. Реляциялық дерекқорларды әрі қарай зерттеуге кіріспес бұрын, болашақта тәжірибеде және теорияда қолданылатын терминдер мен анықтамаларды қарастырамыз.

Дерекқор кестесі - нысандардың бірыңғай сыныбы туралы ақпаратты қамтитын екі өлшемді алап. Реляциялық алгебра теориясында екі өлшемді массив (кесте) *қатынастар* деп аталады.

Кесте келесі элементтерден тұрады: өріс, ұяшық, жазба (2.1-сурет).

Өріс дерекқор нысандарын сипаттайтын сипаттамалардың біреуінің мағынасын қамтиды. Кестедегі өрістер саны дерекқор нысандарын сипаттайтын белгілер санына сәйкес келеді.

2.1-сурет. Дерекқор кестесінің құрылымы

Ұяшық тиісті өрістің нақты мәнін (бір нысанның ерекшелігі) қамтиды.

Жазба – кестенің жолы. Ол бір нысанды сипаттайтын барлық белгілердің мағыналарын қамтиды. Жазбалардың (жолдар) саны кестеде көрсетілген деректердің санына сәйкес келеді.

Деректер базасының теориясында *«жазба»* термині *кортеж*ұғымына сәйкес келеді - бір-бірімен AND (И) қатынастарымен байланысты атрибуттардың тізбегі. Графтар теориясында кортеж бағдарланған графтың қарапайым бөлігін - ағашты білдіреді.

2.1-кестеде реляциялық дерекқорларды әзірлеу тәжірибесі мен теориясында қолданылатын терминдер берілген.

Реляциялық дерекқорлардың оңтайлы құрылымын құру үшін қажетті маңызды тұжырымдамалардың бірі – кілт немесе кілтті өріс тұжырымдамасы.

Кілт - кестеде қалған барлық жолдардың мағынасын анықтайтын өріс болып есептеледі. Мысалы, «Паспорт нөмірі» жолы, «Салық төлеушінің сәйкестендіру нөмірі (СТН)» кез-келген жеке тұлғаның сипаттамаларын (компанияның бухгалтерлік немесе кадрлар бөлімдеріндегі тиісті деректерқор кестесін жасау кезінде) бірегей түрде анықтайды.

2.1-кесте

Реляциялық дерекқор терминдері

ДҚ теориясы	Реляциялық ДҚБЖ(FoxPro, Microsoft Access)	SQL Server 7.0
Қарым-қатынас(Relation)	Кесте(Table)	Кесте(Table)
Атрибут (Attribute)	Өріс(Field)	Баған(Column)
Кортеж (Tuple)	Жазба(Record)	Жол(Row)

Кестенің кілті бір емес, бірнеше өрістер болуы мүмкін. Бұл жағдайда, өрістер жиынтығы уақытқа тәуелсіз екі талап қанағаттандырған жағдайда кесте кілті болуы мүмкін: бірегейлік және минималдылық. Бастапқы кілттің бөлігі болып табылмайтын әр өріс кестенің шешуші емес өрісі деп аталады.

Кілттің *бірегейлігі* кез-келген уақытта дерекқор кестесінде кілт өрістерінің бірдей мәндері бар әртүрлі жазбаны қамтуы мүмкіндігін білдіреді. Бірегейлік талаптарын орындау міндетті болып табылады.

Кілт өрістерінің *минималдылығы* талаптары таңдалған өрістердің мәндерінің тек қана дерекқор кестесіндегі жазбалардың бірегейлігі талаптарына сәйкес келетінін білдіреді. Ол сондай-ақ, кілттегі өрістердің біреуі бірегейлікті бұзбастан одан шығарылмайды дегенді білдіреді.

Бірнеше өрістерден тұратын дерекқордың кесте кілтін жасаған кезде, келесі ережелерді басшылыққа алу қажет:

• кестенің құрамына мағыналы өз өзінен кестедегі жазбаны бірегей түрде сәйкестендіретін кестенің өрісін енгізбеген дұрыс. Мысалы, бір мезгілде «паспорттың нөмірінен» және «төлқұжаттың сәйкестендіру нөмірі» өрістерінен тұратын кілтті жапсаған дұрыс емес, өйткені осы атрибуттардың әрқайсысы кестедегі жазбаларды бірегей түрде сәйкестендіре алады;

• кілттің құрамына мәндері кестеде қайталануы мүмкін өрісті, яғни бірегей емес өрісті қосуға болмайды.

Әрбір кестеде кем дегенде бір кілт болуы керек, ол *бастапқы кілт* ретінде таңдалады. Егер кестеде әрқайсысының мағынасын жазбалар анықтайтын өрістер болатын болса, онда осы өрістер *баламалы кілттер* ретінде қабылданады. Мысалы, салық төлеушінің сәйкестендіру нөмірін бастапқы кілт ретінде таңдасаңыз, паспорттың нөмірі баламалы кілт болады.

2.2. Реляциялық дерекқордағы кестелерді қалыпқа келтіру

Реляциялық дерекқор – бір-бірімен байланысты кейбір көптеген кестелер жиынтығы. Бір дерекқордағы немесе бір файлдағы кестелердің саны көптеген факторларға байланысты, олардың негізгілері:

• дерекқор пайдаланушыларының құрамы,

• ақпараттардың тұтастығын қамтамасыз ету (әсіресе көп қолданыстағы ақпараттық жүйелерде маңызды),

• кішігірім өңдеу уақытын және қажетті жадтың ең аз мөлшерін қамтамасыз ету.

Реляциялық дерекқорды жобалау кезінде аталған факторларды есепке алу кестелерді қалыпқа келтіру және олардың арасындағы байланыстарды орнату әдістерін пайдалана отырып жүзеге асырылады.

Кестелерді қалыпқа келтіру–дерекқордің бір кестесін жалпы жоғарыда аталған талаптарға жауап беретін бірнеше кестеге бөлу әдістерінен тұрады.

Кестелерді қалыпқа келтірукестенің құрылымында кезекті өзгерісті, ол соңғы қалыпқа келтірудің соңғы талаптарының қанағаттандырғанша білдіреді. Қалыпқа келтірудің алты түрі бар:

- бірінші қалыпқа келтіру нысаны(First Normal Form — 1NF);
- екінші қалыпқа келтіру нысаны(SecondNormalForm— 2NF);
- үшінші қалыпқа келтіру нысаны(ThirdNormalForm— 3NF);
- Бойс—Кодд қалыпты нысаны(Brice—CoddNormalForm— BCNF);
- төртінші қалыпқа келтіру нысаны(FourthNormalForm — 4NF);
- бесінші қалыпқа келтіру нысаны немесе проекцияның – байланыстың қалыпты нысаны(FifthNormalForm — 5NF, немесеPJ/NF).

Қалыпты нысандарды сипаттаған кезде келесі ұғымдар қолданылады: «өрістер арасындағы функционалдық тәуелділік»; «өрістер арасындағы толық функционалдық тәуелділік»; «өрістер арасындағы көп мағыналы функционалды тәуелділік»; «өрістер арасындағы өтпелі функционалды тәуелділік»; «өрістер арасындағы өзара тәуелсіздік».

А және В өрістерінің арасындағы *функционалдық тәуелділік,* А-ның әрбір мағынасына кез-келген уақытта барлық мүмкін болатын мағынадағы В-ның жалғыз мағынасы сәйкес келетін тәуелділік болып келеді. Функционалды тәуелділікке мысал ретінде, салық төлеушінің сәйкестендіру нөмірі мен оның төлқұжатының нөмірі арасындағы байланысты алуға болады.

А және В өрістері арасындағы *толық функционалдық тәуелділік,* В өрісі функционалды түрде А өрісіне тәуелді келетін және А өрісінің кез-келген көпшіліктеріне функционалды тәуелді емес тәуелділік болып келеді.

Өрістер арасындағы *көпмағыналы функционалды тәуелдік* келесі жолмен анықталады. А өрісі, егерде А өрісінің әрбір мағынасы үшін В өрісінің тиісті мағыналарының «жақсы анықталған көпшілігі» болатын жағдайда, В өрісінің көпмағыналы анықтайды. Мысалы, «Пән» (А өрісі) және «Бағалау» (В өрісі) өрістерінен тұратын мектептегі оқушылардың үлгерімділігі кестесін қарастыратын болсақ, онда В өрісі рұқсат берілетін мағыналардың «жақсы анықталған көпшілігінен» тұрады: 1, 2, 3, 4, 5, яғни «Пән» өрісінің әрбір мағынасы үшін «Бағалау» өрісінің көпбелгілік «жақсы анықталған көпшілігі» бар.

А және С өрістерінің арасында*транзитивтік функционалдық тәуелділік*,С өрісі функционалды В өрісіне байланысты болса және В өрісі функционалды түрде А өрісіне байланысты болғанда кездеседі; бұл реттеА өрісіне В өрісіне функционалдық тәуелділігі жоқ.

Өрістер арасындағы өзара тәуелсіздік келесі түрде анықталады. Бірнеше өріс егерде олардың бірде-бірі функционалды түрде екіншісіне тәуелді болмаған жағдайда,бір-біріне өзара тәуелсіз болады.

Бірінші қалыпты нысан. Кесте өрістердің ешқайсысы бірден артық мағынаға ие болмаса және кез-келген шешуші өріс бос болмаған жағдайда бірінші қалыпты нысанда болады.

Бірінші қалыпты нысан реляциялық деректер үлгісінің негізі болып табылады. Реляциялық дерекқордағы кез-келген кесте автоматты түрде бірінші қалыпты нысанда болады, басқасы анықтау бойынша мүмкін емес. Мұндай кестеде бірнеше өрістерге (белгілерге) бөлуге болатын өрістер (белгілер) болмауы керек.

Қалыпқа келтірілгендерге, әдетте, бастапқыда ондағы ақпаратты компьютерлік өңдеуге арналмаған кестелер жатады. Мысалы, 2.2-кестеде Эксперименттік металл кесуші білдектердің ғылыми-зерттеу институты (ЭНИМС) жариялаған «Әмбебап металл кескіш білдектер» анықтамалығынан алынған кестедегі фрагментті берілген.

Аталған кесте келесі себептер бойынша қалыпқа келтірілмейтін болып табылады.

1. Ол бір ұяшықта бір өрістің бірнеше мағыналары бар жолдардан тұрады: «Өңдеудің жоғары диаметрі, мм» - «Айналдырықтың айналу жиілігі, айн/мин» емес.

2. Бір өріс - «Габариттік өлшемдері (ұзындығы х ені х биіктігі), мм» үш өріске бөлінуі мүмкін: «Ұзындығы, мм», «Ені, мм» және «биіктігі, мм». Осындай бөлудің орындылығы, ауданның немесе алынған көлемді одан әрі есептеу қажеттілігімен негізделуі мүмкін.

Бастапқы кесте бірінші қалыпты нысанға түрлендірілуі мүмкін. Ол үшін:

• «Өңдеудің ең жоғары диаметрі, мм» өрісі «Айналдырықтың айналу жиілігі, айн/мин» өрістерін бір ұяшықтағы мәндер санына сәйкес бірнеше өрістерге бөлу;

2.2-кесте

Токарлық топтың білдектері

р/н №	Білдектің моделі	Өңдеудің жоғары диаметрі, мм	Айналдырықтың айналу жиілігі, айн/мин	Габариттікөлшемдері (ұзындығы х ені х биіктігі), мм
1	1Д12	12 (алты қырлы)	112... 5000 (сол жақ); 56 ...630 (оң жақ)	1630 х 740 х 1410

2.2-кестесінің қалыпты нысаны

р/н №	Білдектің моделі	Цилиндрлік дайындаманың жоғары диаметрі	Цилиндрлік дайындаманың жоғары диаметрі	Айналдырықтың айналу жиілігінің диапазоны (сол жақ), айн/мин	Айналдырықтың айналу жиілігінің диапазоны (оң жақ), айн/мин	Ұзындығы, мм	Ені, мм	Биіктігі, мм
1	1Д12	12	8	112... 5000	56... 630	163	740	141

• «Габариттік өлшемі (ұзындығы х ені х биіктігі), мм» өрісі үш өріске бөлінген: «Ұзындығы, мм», «Ені, мм», «Биіктігі, мм».

Бұл кестенің шешуші өрісі «Білдек моделі» немесе «р/н №» өрісі болуы мүмкін.

2.3-кестеқалыпты нысанның түрінен тұрады.

Тағы бір мысал қарастырайық. 2.2-суретте алдыңғы мысалдағыдай бастапқыда компьютерді өңдеуге арналмаған, сынақ-емтихан тізімдемесі бланкінің фрагменті көрсетілген.

Факультет № . . .					**Топ ...**					
Семестр ...					**Пән бойынша ...**					
Емтихан алушы ...										
р/н №	**Аты-жөні**	**1-Емтихан Сынақ**			**2-1-Емтихан Сынақ**			**3-1-Емтихан Сынақ**		
		Күні, қолы	**Күні, қолы**	**Күні, қолы**	**Күні, қолы**	**Күні, қолы**	**Күні,** қолы	**Күні,** қолы	**Күні,** қолы	**Күні,** қолы
Факультет деканының қолы					**Емтихан алушының қолы**					

2.2-сурет. Сынақ-емтихан тізімдемесінің бланкі

Сынақ-емтихан тізімдемесінің құрамына сәйкес сынақ-емтихан сессиясының нәтижелерін автоматтандырылған өңдеу үшін деректер базасын құруды қалаймыз. Ол үшін бланктің мазмұнын дерекқор кестелеріне түрлендіреміз. Өрістер арасындағы функционалдық тәуелділіктің талаптарын сақтау қажеттілігіне қарай, кемінде екі кестені қалыптастыру қажет (2.3-сурет) (шешуші өрістер әрбір кестеде жартылай майлы шрифтімен айқындалған). Алғашқы кестеде әрбір студенттің нақты пән бойынша сынақты (емтихан) тапсыруының нәтижелері берілген. Екінші кестеде белгілі бір пән бойынша студенттердің белгілі бір тобының сынақ (емтихан) тапсырудың нәтижелеуші қорытындылары бар. Бірінші кестеде «Студенттердің аты-жөні» өрісі, ал екінші кестеде «Пән» өрісі шешуші болып табылады. Кестелер бір-бірімен «Пән» және «Топ шифры» өрістерінде өзара байланысқан болуы керек.

Кестелердің ұсынылған құрылымы бірінші қалыпты нысанның талаптарына толығымен жауап береді, бірақ келесі кемшіліктермен сипатталады:

• кестеге жаңа деректер қосу барлық өрістер үшін мағыналарды енгізуді талап етеді;

• әр кестенің әрбір жолында «Пән», «Оқытушының аты-жөні», «Топ шифры» өрістерінің қайталанатын мағыналарын енгізу керек.

Одан туындайтыны, кестелердің мұндай құрамы және олардың құрылымы кезінде ақпараттың анық артықтығы, әрине қосымша жадты талап етеді.

Жоғарыда айтылған кемшіліктерді болдырмау үшін кестелерді екінші үшінші емес қалыпты нысанға келтіру қажет.

Екінші қалыпты нысан. Кесте бірінші қалыпты нысанның талаптарын және бастапқы кілтке енбейтін оның барлық өрістерін қанағаттандыратын болса, екінші қалыпты нысанда болады, бастапқы кілтпен толық функционалды тәуелділікпен байланысты.

пән	Оқытушының аты-жөні	Топ шифры	Студенттің аты-жөні	Журнал бойынша №;	Тапсыру күні	Бағалау	Оқытушының қолы
↓		↓					
пән	Оқытушының аты-жөні	Топ шифры	Үздік бағалар саны	жақсы бағалар саны	қанағаттанарлық бағалардың саны	қанағанаттарлықсыз бағалардың саны	орташа үлгерім

2.3-сурет. Сынақ-емтихан тізімдемесінің бастапқы кестелері

Егер кестеде тек бір өрістен тұратын қарапайым бастапқы кілтке ие болатын болса, онда ол автоматты түрде екінші қалыпты пішінде болады.

Егер бастапқы кілт құрамдас болса, онда кесте екінші қалыпты нысандаболуы міндетті емес. Онда оны кез-келген өрістегі мәнді бірегей түрде сәйкестендіретіндей, тағы екі немесе көп кестелерге бөлу керек. Егерде кестеде бастапқы кілтке тәуелді емес кемінде бір өріс болатын болса, онда бастапқы кілттіқосымша бағандарға қосу керек. Егер мұндай бағандар болмаса, жаңа бағанды қосу керек.

Екінші қалыпты нысанды анықтайтын осы талаптар негізінде, құрастырылған кестелердің сипаттамалары бойынша келесі қорытындыларды шығаруға болады (2.3-суретті қараңыз).

Бірінші кестеде шешуші өріс пен «Оқытушының аты-жөні» өрісі арасында тікелей байланыс жоқ, себебі әр түрлі мұғалімдер бір пән бойынша сына немесе емтихан қабылдай алады. Кестеде «Пән» шешуші өрісі мен барлық қалған өрістер арасында толық функционалды тәуелділік бар.

Сол сияқты, екінші кестеде шешуші өріс пен «Оқытушының аты-жөні» өрісі арасында тікелей байланыс жоқ.

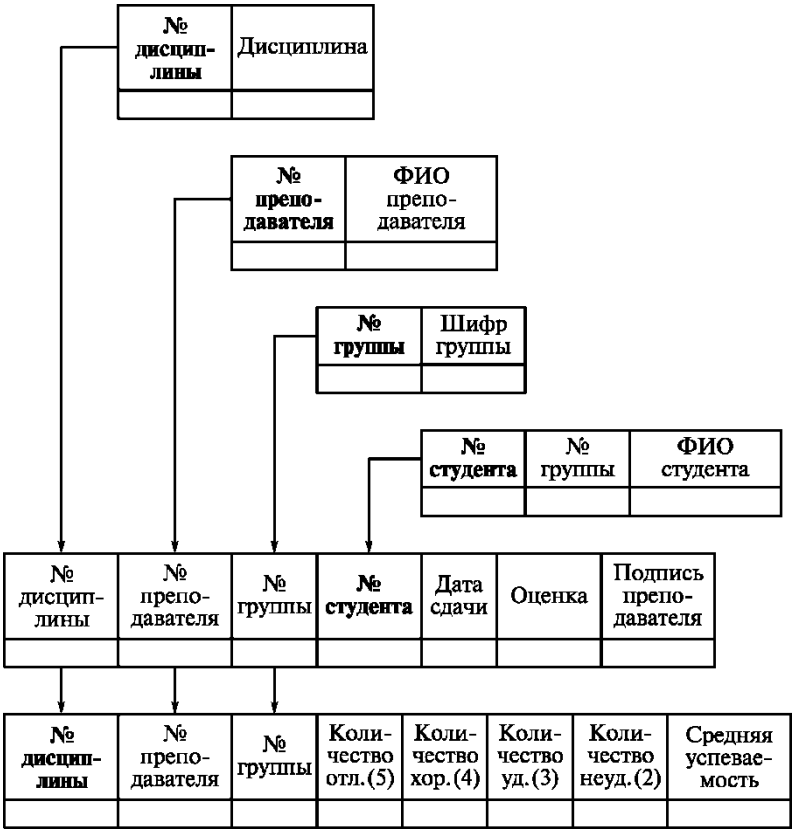
Дерекқорды оңтайландыру үшін, атап айтқанда, әрбір жазбаға «Пән» және «Оқытушының аты-жөні» өрісінің мағыналарын қайталау қажеттілігіне байланысты қажетті жад көлемін азайту үшін, дерекқордың құрылымын өзгерту - бастапқы кестелерді екінші қалыпты нысанға түрлендіру керек. Деректерқор құрылымы кестелерінің құрамы 2.4-суретте берілген.

Деректерқордың түрлендірілген құрылымы алты кестеден тұрады, олардың екеуі бір-бірімен байланысты (әр кестенің шешуші өрістері жартылай майлы шрифтімен белгіленген). Барлық кестелер екінші қалыпты нысанның талаптарын қанағаттандырады.

Бесінші және алтыншы кестелер өрістердегі қайталанатын мағыналарға ие, бірақ осы мағыналар мәтіндік деректердің орнына бүтін сандарды білдіретіндіктен, ақпаратты сақтауға қажетті жадтың жалпы көлемі бастапқы кестелерге қарағанда айтарлықтай аз болады (2.1-суретті қараңыз).

Одан өзге, дерекқордың жаңа құрылымы кестелерді әртүрлі мамандармен (басқару қызметтерінің бөлімшелері) толтыру мүмкіндігін қамтамасыз етеді. Дерекқор кестелерін одан әрі оңтайландыру оларды үшінші қалыпты нысанға жеткізуге алып келеді.

Үшінші қалыпты нысан. Егер кесте екінші қалыпты нысанның анықтамасын қанағаттандыратын болса, кесте үшінші қалыпты нысанда болады және оның бірде-бір шешуші өрістері басқа шешуші емес өріске функционалды тәуелсіз.



2.4-сурет. Екінші нысанға дейінгі қалыпқа келтірілген, сынақ-емтихан тізімдемесінің кестелері

Егер кесте екінші қалыпты нысанда болса және әрбір шешуші емес өріс бастапқы кілтке транзитивті тәуелді емес болатын болса, кесте үшінші қалыпты нысанда екендігін айтуға болады, Үшінші қалыпты нысанның талаптары, шешуші емес барлық өрістер бастапқы кілтке ғана тәуелді болмауына және бір-біріне тәуелді еместігіне алып келеді.

Осы талаптарға сәйкес дерекқор кестелері құрамында (2.4-суретті қараңыз) үшінші қалыпты нысанға бірінші, екінші, үшінші және төртінші кестелер жатады.

Бесінші және алтыншы кестелерді үшінші қалыпты нысанға жеткізу үшін, студенттер топтарында емтихан немесе сынақ өткізілетін пәндердің құрамы туралы ақпараттан тұратын жаңа кесте жасаймыз. Кілті ретінде біз «Есептеуіш» өрісін құрдық, ол кестеде жазба нөмірді орнатады, себебі әрбір жазба бірегей болуы керек.

Нәтижесінде 2.5-суретте көрсетілген дерекқордың жаңа құрылымын аламыз (әр кестенің шешуші өрістері қалың қаріппен белгіленген). Аталған құрылымда үшінші қалыпты нысанның талаптарына сәйкес келетін жеті кесте бар.

Бойс-Кодцтің қалыпты нысаны. Кесте өрістері арасындағы функционалдық тәуелділік ықтимал кілтке толық функционалды тәуелділікке жеткізілетін жағдайда ғана Бойс-Коддтың қалыпты нысанында болады.

Осы анықтамаға сәйкес, дерекқор құрылымындағы (2.4-суретті қараңыз) барлық кестелер Бойс-Коддтың қалыпты нысаны талаптарына сәйкес келеді.

Дерекқор кестелерін одан әрі оңтайландыру кестелердің толық декомпозициясына жеткізілуі тиіс.

Кестенің толық декомпозициясы деп қосылысы кестенің мазмұнына толығымен сәйкес келетін оның проекциясының туынды санының жиынтығын атайды.

Проекция деп жаңа кестенің бір немесе бірнеше бағандарын қамтымайды кестенің көшірмесін атайды.

Төртінші қалыпты нысан. Төртінші қалыпты нысан–толық декомпозиция екі проекцияның қосылуы болуы тиіс бесінші қалыпты нысанның жеке жағдайы болып табылады.

Пәннің №	Пән

Оқытушының №	Оқытушының аты-жөні

Топтың №	Топтың шифры

Студенттің №	Топтың №	Студенттің аты-жөні

Есептеуіш	Топтың №	Пәннің №

Есептеуіш	Студенттің №	Тапсыру күні	Бағалау	Оқытушының қолы

Есептеуіш	Оқытушының №	Үздік бағалар саны(5)	Жақсы бағалар саны (4)	Қанағат. бағалар саны	Қанағаттан арлықсыз бағалар саны (2)	Орташа үлгерім

2.5-кесте. Үшінші нысанға дейінгі қалыпқа келтірілген, сынақ-емтихан тізімдемесінің кестелері

Ол төртінші қалыпты нысанда болатындай, бірақ бесінші қалыпты нысан анықтамасын қанағаттандыра алмайтын кестені табу өте қиын,

Бесінші қалыпты нысан. Кестенің әрбір толық декомпозициясында барлық проекциялар мүмкін болатын кілттен тұрған жағдайда ғана бесінші қалыпты нысанда болады. Бірде-бір толық декомпозицияға ие емес кесте де, бесінші қалыпты нысанда болады.

Іс жүзінде дерекқор кестелерін оңтайландыру үшінші қалыпты нысанмен аяқталады. Кестені төртінші және бесінші қалыпты нысандарға келтіру, біздің ойымызша, таза теориялық қызығушылық болып табылады. Іс жүзінде, бұл мәселе жаңа кестені құру сұраныстарын өңдеу арқылы шешіледі.

2.3. Кестелер арасындағы байланыстарды түзету

Дерекқордың бастапқы кестелерін қалыпқа келтіру үдерісі ақпараттық жүйенің оңтайлы құрылымын жасау - ақпаратқа қысқа мерзімде қолжетімділікті қамтамасыз ететін және жадының кіші ресурстарын қажет ететін дерекқорды әзірлеу мүмкіндігін береді.

Сонымен қатар, бір көзден кестені бірнешеге бөлу ақпараттық жүйелерді жобалаудың маңызды шарттарының бірі - дерекқорды пайдалану процесінде ақпараттың тұтастығын қамтамасыз етуді талап етеді.

Сонымен қатар, бір бастапқы кестені бірнешеге бөлу ақпараттық жүйелерді жобалаудың маңызды талаптарының бірі - дерекқорды пайдалану процесінде ақпараттың тұтастығын қамтамасыз етуді талап етеді.

Жоғарыда келтірілген мысалда бастапқы кестелерді қалыпқа келтіру кезінде (2.3-суретті қараңыз) екі кестеден соңғы қорытындыда үшінші және төртінші қалыпты нысандарға келтірілген жеті кесте алдық.

Тәжірибе көрсеткендей, нақты өндірісте және бизнесте дерекқорлар көп қолданушы жүйелерден тұрады. Ол жекелеген кестелерде деректерді құру мен сақтауға, сонымен қатар шешім қабылдау үшін ақпаратты пайдалануға жатады.

Жоғарыда қаралған мысалда жоғары оқу орнында немесе колледжде оқу үдерісін басқарудың шынайы қызмет ететін жүйесінде, оқу топтарын бастапқы қалыптастыру қабылдау емтихандарының қорытындысы бойынша талапкерлерді қабылдау кезінде қабылдау комиссияларымен жүргізіледі. Жоғары оқу орнында топтардағы студенттердің құрамы туралы ақпараттарды одан әрі қолдау деканаттарға, ал колледждерде оқу бөлімдеріне немесе тиісті құрылымдарға беріледі. Топтар бойынша білім беру пәндерінің құрамы басқа қызметтермен немесе мамандармен анықталады. Оқытушылар құрамы туралы ақпарат кадрлар бөлімінде қалыптастырылады. Сынақ және емтихан сессияларының нәтижелері деканатқа және бөлімше басшыларына қажет, соның ішінде озат студенттерге стипендия беру немесе үлгермейтін студенттерге «стипендиядан алып тастау» туралы шешім қабылдау үшін талап етіледі.

Дерекқор кестелерінің кез-келген өзгерісі барлық басқа кестелерде барабар өзгерістерді анықтауы керек. Бұл дерекқордың тұтастығын қамтамасыз етудің мәні. Іс жүзінде, бұл тапсырма дерекқор кестелері арасындағы байланыстарды орнату арқылы жүзеге асырылады.

Кестелер арасында байланыс орнатудың негізгі ережелерін тұжырымдаймыз.

1. Екі байланыстырылған кестеден негізгі және бағыныстағы кестелерді таңдаңыз.

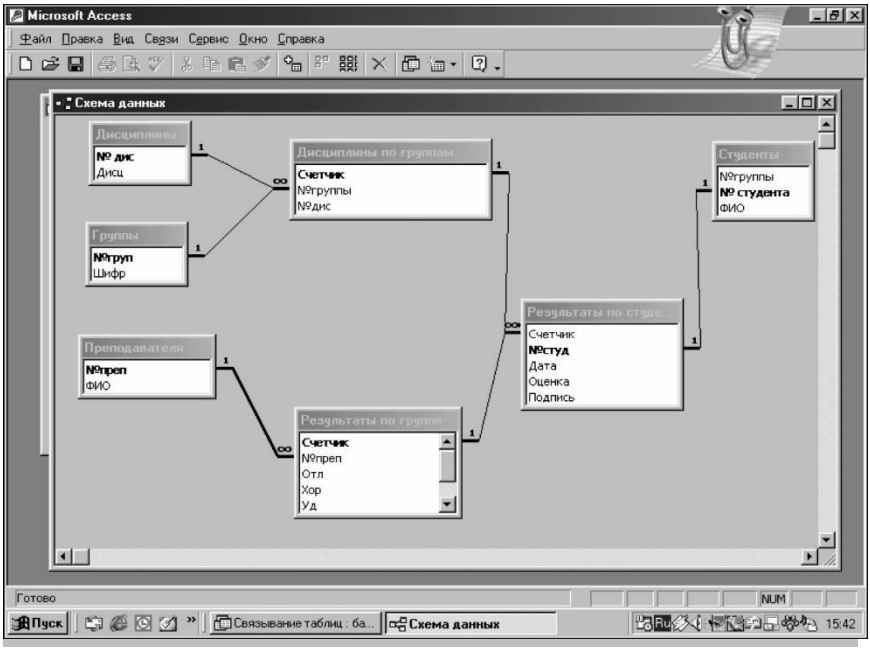
2. Әр кестеде шешуші өрісін таңдаңыз. Негізгі кестенің шешуші өрісі *бастапқы кілт* деп аталады. Бағыныста кестенің негізгі өрісі *сыртқы кілт* деп аталады.

3. Кестенің байланысқан өрістері деректер бірдей үлгісінен тұруы тиіс.

4. Кестелер арасында байланыстың келесі түрлері орнатылады: «бірге бір»; «біреуден көпке»; «көптегендері көптегендерге»:

• «бірге бір» байланысы негізгі кестенің нақты жолыкез-келген уақытта бағыныста кестенің тек бір жолына байланысты болған жағдайда орнатылады;

• «біреуден көпке» байланысы негізгі кестенің нақты жолы кез-келген уақытта бағыныста кестенің бірнеше жолдарымен байланысты болған жағдайда орнатылады;



2.6-сурет. Деректердің логикалық моделі (ДҚ кестелері арасындағы логикалық байланыс)

бұл ретте, бағыныстағы кестенің кез-келген жолы негізгі кестенің бір жолымен ғана байланысты;

• «көптегендері көптегендерге» байланысы негізгі кестенің нақты жолы кез-келген уақытта бағыныстысы кестенің бірнеше жолдарына байланысты болған кезде және сол уақытта бағыныстысы кестенің бір жолы негізгі кестенің бірнеше жолдарымен байланыстырылған жағдайларда орнатылады.

Негізгі кестедегі бастапқы кілттің мағынасы өзгерген келсе, тәуелді кесте әрекеттерінің келесі нұсқалары мүмкін.

Каскадтау(Cascading). Негізгі кестенің бастапқы кілтінің деректерін өзгерткен кезде, тәуелді кестеде сыртқы кілттің тиісті деректерінің өзгеруі орын алады. Барлық қол жетімді байланыстар сақталады.

Шектеу(Restrict). Тәуелді кестедегі жолдар байланысты болып келетін бастапқы кілттің мағынасын өзгертуге тырысқанда, өзгертулер қабылданбайды. Тек тәуелді кестемен байланыс орнатылмаған бастапқы кілттің мағынасын ғана өзгертуге болады.

Орнату(Relation). Бастапқы кілттің деректерін өзгерткен кезде сыртқы кілт анықталмаған мағынаға (NULL) орнатылады. Тәуелді кесте жолдарының тиесілігі туралы ақпарат жоғалады. Егерде бастапқы кілттің бірнеше мағыналарын өзгертетін болсақ, онда тәуелді кестеде бұрын өзгертілген кілттермен байланысты бірнеше жол топтары пайда болады. Осыдан кейін бастапқы кілт қосылған қандай жолды анықтау мүмкін емес.

2.6-суретте 2.5-суретте ұсынылған дерекқор кестелері арасындағы байланыс сызбасы көрсетілген.

Бақылау сұрақтары

1. Дерекқор кестесінің келесі элементтеріне анықтама беріңіз: өріс, ұяшық, жазба.
2. «Кілт», «шешуші өріс» деген түсініктер нені білдіреді?
3. Қандай шешуші өрісті негізгі кілт, ал қайсысын сыртқы кілтдеп атайды?
4. Дерекқордың кестелерін қалыпқа келтіру процесі неден тұрады?
5. Дерекқор кестелерінің қандай бес қалыпты нысанын білесіз?
6. Дерекқор кестелері арасындағы байланыстың келесі түрлеріне анықтама беріңіз:«бірге бір»; «біреуден көпке»; «көптегендері көптегендерге».

3-тарау

РЕЛЯЦИЯЛЫҚ ДЕРЕККОРДЫҢ АҚПАРАТТЫҚ МОДЕЛІ

3.1. Ақпараттық модельдердің үлгілері

Алдыңғы тарауда кестені қалыпқа келтіру қағидаларына негізделген реляциялық дерекқорларда кестелердің оңтайлы құрамын қалыптастырудың теориялық негіздерін қарастырдық. Әрине, кез-келген мәселені, соның ішінде ақпараттарды әртүрлі әдістермен шешуге болады. Жоспарланған дерекқордың нұсқаларын бағалау үшін ақпараттық модельдер әзірленуде. Ақпараттық модельдерді әзірлеу әлі де анық қалыптасқан ережелерге ие емес, сондықтан осы тапсырма, сондай-ақ дәстүрлі бағдарламалау процесі «өнер» болып табылады және әзірлеушілердің біліктілігіне байланысты. Дегенмен, жобаланатын дерекқордың оңтайлылығын бағалау үшін ақпараттық модельдерді құрудың негізгі принциптері бар.

Дерекқордың ақпараттық моделі жүйені сипаттаудың үш деңгейін: тұжырымдамалық, логикалық, физикалық және сәйкесінше - ақпараттық модельдердің үш түрін көздейді.

3.2. Деректердің тұжырымдамалық модельдері

Дерекқорды сипаттаудың *тұжырымдамалық деңгейі* (тұжырымдамалық моделі) деректерді сипаттау және сақтау тәсілдерін көрсетпестен ақпараттық объектілер мен олардың өзара байланысынан тұрады.

Аталған анықтамада ақпараттық объектілер дерекқор кестелерінде сақталатын мәліметтер туралы объектілер сыныбы деп аталады. Әдетте, дерекқор кестесі бір сыныптың нысандары туралы ақпаратты қамтиды.

Сынып деп, белгілердің бірдей жиынтығын сипаттайтын көптеген объектілерді атайды.

Бір сыныптың ақпараттық объектілер туралы деректерді бір немесе бірнеше кестелерден табуға болады.

Әртүрлі сыныптардың ақпараттық объектілері туралы дерек әртүрлі кестелерде болады.

Тұжырымдамалық модельді әзірлеудің түпкі мақсаты деректер базасының кестелерінің оңтайлы құрамын белгілеу болып табылады.

Дерекқордың тұжырымдамалық үлгісін алдыңғы бөлімшелерде мазмұндалған қалыпқа келтіру принциптері негізінде қорытындыда дерекқордың тұжырымдамалық моделін анықтау әдістерін анықтайды.

Осылайша, 1.3-бөлімшеде қойылған мысалда, алға қойылған тапсырмаға сәйкес жеті кестені құруды көздейтін дерекқордың базасының тұжырымдамалық моделі әзірленді.

3.3. Деректердің логикалық моделі

Дерекқор сипаттамасының *логикалық деңгейі* (логикалық моделі) кестелер арасындағы логикалық қатынастарды көрсетеді. 1.4-бөлімшелеріндегі кестелер арасында байланыстарды қалыптастыру ережелерімен таныстық. 2.6-суретте көрсетілген кестелер арасындағы байланыстар сұлбасы тиісті дерекқордың логикалық моделі болып табылады. «Пән», «Топтар» және «Топтар бойынша пәндер» кестелерінің арасында «біреуден көпке» байланыстары орнатылады. «Оқытушылар» және «Нәтижелер бойынша топтар» кестелерінің, сондай-ақ «Топ бойынша нәтижелер» және «Студенттердің нәтижелері» кестелерінің арасында байланыс сипаты орнатылған. «Студенттер» және «Студенттік нәтижелер» кестелері арасында «біреуден көпке» байланысы орнатылған.

Аталған логикалық модельді талдай келе, аталған дерекқор құрылымының білім беру мекемесінің басқару қызметтерінің әр түрлі мамандары жүзеге асыратын кестелердегі кез-келген өзгерістер туралы ақпараттың тұтастығын қамтамасыз етуге мүмкіндік береді деген қорытынды жасауға болады.

3.4. Деректердің физикалық моделі

Реляциялық дерекқордың сипаттамасының *физикалық деңгейі* (физикалық модельдер) ақпаратты өңдеу мен сақтау тәсілдерін сипаттайды. Дерекқорды және дерекқор базасын басқару жүйелерін әзірлеу теориясы мен практикасында деректердің физикалық моделін құрудың екі тәсілі бар.

Бірінші тәсіл нақты ДҚБЖ-мен байланысты емес және әрбір кесте деректерінің физикалық қасиеттерінің сипаттамасын - *дерекқор кестелерінің физикалық үлгілерін қамтиды*.

Физикалық модельді дамытудың екінші тәсілі сәулетті дамытуға, дерекқордың физикалық моделіне – нақты ДҚБЖ-да деректерді ұйымдастыру және сақтауға байланысты.

Дерекқор әзірлеуші өзінің ақпараттық жүйесін жасайтын қолданбалы бағдарламалық жүйесінің сәулетін білмейді, бұл ретте ол әрбір кестеге арналған физикалық үлгілерді жасауы тиіс.

Дерекқор кестелерінің физикалық модельдері. Дерекқор кестесінің физикалық моделі кестедегі әрбір өрістің қасиеттері сипаттамасын болжайды. Өрістердің қасиеттерін сипаттау үшін кесте жобасын 3.1-суретте көрсетілген нысанға сәйкес құру қажет.

р/н №	Өрістің жолы	Өрістің қолы	Деректер үлгісі	Белгілер саны	Дәлділік	Кілт (иә)

3.1-сур. ДҚ кестесінің құрылымын сипаттау жобасы

Осылайша, дерекқордың кесте жобасының физикалық моделін әзірлеу әрбір өрістің қасиетін сипаттауға алып келеді. Дерекқордың кесте өрістерінің міндетті сипаттамаларынкелтіреміз.

Өріс атауы - кестеде деректерді іздестіру үшін алдын-ала тағайындалған таңбалардың кейбір минималды жинағы. Дерекқорды әзірлеу жасау үшін әрбір қолданбалы бағдарламалық жүйеде өрістің атауларын қалыптастыру үшін жеке грамматикалық ережелерге ие. Тұтастай алғанда, өріс атауын бос жердің таңбасымен бастауға, таңбалар ретінде тыныс белгілерін таңдауға жол берілмейді.

Өрістің қолы мағыналары өрістің ұяшықтарында сақталатын объектігінің белгісінің атауымен анықталады. Өрістің қолы кестенің атауында болады. Заманауи ДҚБЖ-да өрістің қолтаңбасын қалыптастыруға қандай да болмасын шектеулер жоқ.

Деректер түрі - нақты бағдарламалық жүйеге сәйкес деректер түрін белгілеу.

Беогілер саны - өрістің ұяшықтарында сақталатын белгілердің болжамды саны.

Дәлділік - сандық өрістердегі үтірден кейінгі таңбалар саны.

Кілт - аталған өрістің шешуші болып табылатындығы туралы нұсқаулық.

Қасиеттердіңталған құрамы кестеде сақталған деректерді сипаттауға қажет минималдылық болып табылады.

Деректерді сақтаудың физикалық моделдері. Деректерді сақтаудың физикалық моделдері компьютердің жадында немесе ақпараттың тиісті тасымалдағыштарында деректерді орналастыру әдістерін, сондай-ақ осы деректерді сақтау және оларға қол жеткізу әдістерін анықтайды. Тарихи тұрғыда алғашқы сақтау және кіру жүйелеріне файлдық құрылымдар мен файлдарды басқару жүйесі (СУФ) жатады. Іс жүзінде, ақпаратты сақтау файлдық құрылымдары операциялық жүйелердің негізі болып табылады және табылуды. Дерекқорды басқару жүйелерінде файлдарды сақтау жүйелерін пайдалану, қолданушыға тұтас файлдың мазмұны емес, жекелеген деректер түрінде ақпаратты қажет болғандықтан тиімді болмады. Сондықтан, заманауи ДҚБЖ файлдық құрылымдардан сыртқы тасымалдағыштарға тікелей орналастырудан - сыртқы жады құрылғыларына өтті. Алайда, файлдық жүйелерде қолданылатын басқару тетіктері көбінесе ақпаратты сақтаудың парақтық жүйелері деп аталатын сыртқы жадыдағы деректерді ұйымдастырудың жаңа жүйелеріне ауысады.

Сондықтан, деректердің физикалық моделдеріне арналған тарауды, біз файлдар мен файл құрылымдарды шолудан бастаймыз.

Дерекқорды ұйымдастырудың файлдық құрылымдары. Әрбір ДҚБЖ-да деректерді сақтау мен қол жетімділік әртүрлі ұйымдастырылған, бірақ барлық ДҚБЖ қолданылатын кейбір файлдық құрылымдар бар.

Сыртқы жадта ақпаратты сақтау үшін пайдаланылатын дерекқор жүйелеріндегі файлдар мен файл құрылымдары жіктелуі мүмкін (3.2-сурет).

Пайдаланушы көзқарасы тұрғысынан, *файл* - өзімен жазбалардың кейбір тізбегі сақталатын дискілік кеңістіктің жаңа саласын білдіреді. Мұндай файлда бірінші және соңғы жазбаны; ағымдағы жазба; ағымдағы және оның соңындағы жазбаныанықтауға болады.

Файлдардағы ақпаратқа қолжетімділікті басқару әдістеріне сәйкес олар туынды адрестеу немесе тікелей қолжетімділік (магниттік және оптикалық дискілер) және кезеңді адрестеу немесе кезеңді қолжетімділік (магнитофондар, стриммерлер) қондырғыларымен сыртқы жады қондырғыларын (ақпаратты жинақтағыштар) айырады.

Туынды адрестеу қондырғыларында жинақтағыштың кез-келген саласына жазбаны оқу үшін бастиекті дереу орнату мүмкін.

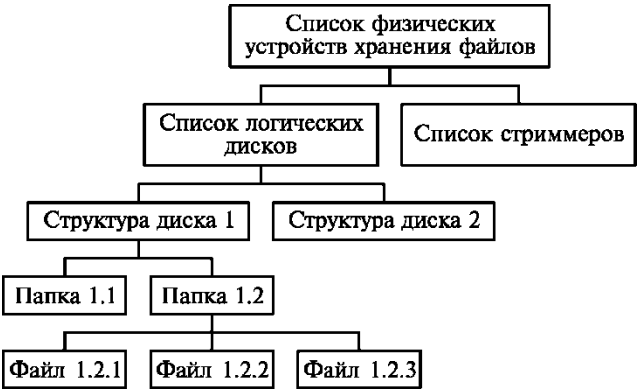
Кезеңді адрестеу қондырғыларында барлық жад ақпарат элементтерінің сызықтық реттілігі ретінде қарастырылады. Сондықтан да, мұндай жинақтағыштарда ақпаратты алу үшін есептеу қондырғысының бастапқы күйінен қажетті жазбаға өту керек.

Тікелей қолжетімді құрылғыларда (ТҚҚ) орналасқан тұрақты жазба ұзындығы бар файлдар *тікелей қолжетімділік файлдары* болып табылады.

Бұл файлдарда қажетті жазбаның орналасуының физикалық мекенжайы жазбаның нөмірі бойынша есептелген (NZ).

Әрбір файлдық жүйе - файлды басқару жүйелері - сыртқы жад көрінісінде жиі иерархия деңгейінің шектеулі санын қамтитын кейбір иерархиялық файлдық құрылымды қолдайды (3.3-сурет).

3.2-сур. Файлдық құрылымдардың жіктелуі



3.3-сур. Иерархиялық файлдық құрылым

Жүйедегі әрбір файл үшін келесі ақпарат сақталады:

- файлдың атауы;
- файлдың түрі (мысалы, кеңеюі немесе басқа сипаттамалар);
- жазудың көлемі;
- бос емес физикалық блоктардың саны;
- негізгі бастапқы мекенжайы;
- кеңейтілген сегментке сілтеме;
- қолжетімділік әдісі (қауіпсіздік коды).

Тұрақты ұзындықтағы жазбалары бар файлдар үшін, K нөмірімен жазбаны орналастыру мекенжайы мына формуламен есептелуі мүмкін

$$BA + (K - 1)\text{-}LZ + 1,$$

мұндағы, BA — базалық мекенжай; LZ — жазбаның ұзындығы.

Жазбаларды оқу тетігін орналастыру қажет мекенжайды анықтауға болатын болса, тікелей кіру құрылғылары мұны дереу жасайды, сондықтан да мұндай файлдар үшін еркін жазбаны оқу оның нөміріне тәуелді емес. Тікелей кіру файлдары ерікті жазбаларға барынша жылдам қолжетімділікті қамтамасыз етеді және оларды пайдалану дерекқор жүйелеріндегі ең келешегі бары болып саналады.

Жүйелі түрде қол жетімді қондырғыларда тек кезеңді кіру файлдары ғана ұйымдастырылуы мүмкін.

Олар екі тәсілмен ұйымдастырылуы мүмкін:

1) жазбаның аяқталуы арнайы маркермен белгіленеді;

2) әрбір жазбаның басында оның ұзындығы жазылады.

Тікелей қол жетімді файлдар жазуға қол жеткізудің жеткілікті сенімді әдісін қамтамасыз етеді. Тікелей қолжетімділік файлдарының басты кемшілігі, жазбаны іздестіру оның нөмірі бойынша жүргізіледі, ол жазбалар санының көптігі кезінде айтарлықтай уақыт алады.

Дерекқорда көбінесе шешуші өрсітер бойынша жазбаларды іздестіруді ұйымдастыру қажет. Бірақ бұл жағдайда тиісті жазба нөмірі белгісіз. Мұндай жағдайларда хэштеу (рандомизация) деп аталатын әртүрлі әдістер қолданылады.

Хэштеу әдістерінің мәні іздестіруді бастау үшін қолдаланылатын кілттің мәні (немесе оның кейбір сипаттамалары) таңдалады, яғни хэш-функциясы деп аталатын*h (k)* есептеледі, мұндағы *k*–шешуші өрістің мағынасы. Бұл жағдайда іздестіру қадамдарының саны біршама азаяды. Дегенмен, осы тәсілдер кезінде бірнеше әртүрлі кілттерге хэш функциясының бір мағынасы, яғни бір мекенжайы сәйкес келуі мүмкін. Мұндай жағдайлар *коллизия* деп аталады. Хэш-функциясының мағынасына ие кілттердің мағыналары *синонимдер* деп аталады.

Сондықтан да, хэштеуді қол жеткізу әдісі ретінде қолданғанда екі тәуелсіз шешімді қабылдау керек:

• хэш функциясын таңдау;

• коллизияны шешу әдісін таңдау.

Коллизияларды шешудің әртүрлі көптеген стратегиялары бар, олардың барынша көп тарағандары:

• асыра толтыру саласының көмегімен коллизияны шешу;

• еркін ауыстыру әдісімен коллизияны шешу.

Асыра толтыру саласының көмегімен коллизияны шешу. Осы стратегияны таңдаған кезде, сақтау орны екі бөлікке бөлінеді: негізгі сала және толып кету саласы.

Әрбір жаңа жазба үшін хэш-функциясының мағынасы есептеледі, ол оның орналасқан жерінің мекенжайын анықтайды және жазба хэш функциясының алынған мағынасына сәйкес негізгі салаға енгізіледі.

Әрбір жаңа жазба ДҚ-да бар басқа жазбаны қолданған хэштеу функциясының мағынасына ие болатын болса, онда жаңа сала асыра толтыру саласының бірінші босаған орнына енгізіледі, ал негізгі салада орналасқан жазба-синонимге асыра толтыру саласында жаңадан орналастырылған жазба мекенжайына сілтеме жасалады. Егер негізгі салада орналасқан жазба-синонимде сілтеме болса, онда жаңа жазба сілтеме түрінде қосымша ақпарат алады және осы түрдеасыра толтыру саласына енгізіледі.

Осындай алгоритм кезінде кез-келген жаңа жазбаны орналастыру уақыты, асыра толтыру саласындағы бірінші еркін жазба нөмірінің жүйелік ауыспалы түрінде сақталатындығын ескере отырып, дискіге екіден артық емес өтініштен аспайды.

Еркін жазба іздестіру механизмін және хэштеудің осы стратегиясына арналған жазбаны жоюды қарастырамыз.

Жазбаны іздестірген кезде алдымен оның хэш-функциясының мағынасы есептеледі және негізгі салада орналасқан синонимдік тізбектегі бірінші жазба оқылады. Егерде ізделіп отырған жазу синонимдер тізбегіне бірінші сәйкес келмесе, онда іздестіру синонимдер тізбегі бойынша қажетті жазба табылғанға дейін ауыстырыла отырып жүзеге асырылады. Іздестіру жылдамдығы синонимдер тізбегінің ұзындығына байланысты, сондықтан хэш-функциясының сапасы синонимдер тізбегінің максималды ұзындығымен анықталады. Тізбектегі 10-нан артық синонимдердің болуын жақсы нәтиже деп санауға болады.

Еркін жазбаны кетіргенде ең алдымен оның орналасқан орны анықталады. Егер синонимдер тізбегіндегі бірінші жазба кетірілетін болса, кетірілігеннен кейін синонимдер тізбегіндегі екінші (келесі) жазба негізгі саладағы оның орнына орналастырылады; бұл ретте барлық көрсеткіштер (синонимдерге сілтемелер) сақталады.

Егер кетірілетін жазба синонимдер тізбегінің ортасында болатын болса, онда көрсеткішті реттеу қажет: кетірілгеннен кейінгі жазбада, тізбеде кетірілетін жазбада көрсеткіш орналастырылады. Егер осы тізбектегі соңғы жазба болатын болса, онда көрсеткіштерді өзгерту механизмі бірдей, яғни алдыңғы жазбаға бұрын соңғы жазбада сақталған тізбектегі келесі жазбаның жоқтығы белгісі қосылады.

Еркін ауыстыру әдісімен коллизияны шешу. Осы стратегия кезінде файлдық кеңістік салаларға бөлінбейді, бірақ әрбір жазба үшін екі көрсеткіш қосылады: синонимдер тізбегіндегі алдыңғы жазбаға көрсеткіш және синонимдер тізбегіндегі келесі жазбаға көрсеткіш. Тіисті сілтеменің болмауы арнайы белгімен, мысалы нөлмен көрсетіледі. Әрбір жаңа жазба үшін хэш-функциясының мағынасы есептеледі және аталған мекенжай бос болатын болса, онда жазба берілген орынға түседі және синонимдер тізбегінде бірінші болады. Егер хэш-функциясының алынған мағынасына сәйкес келетін мекенжай бос болмаса, онда сілтеменің болуына қарай, аталған мекенжайда орналасқан жазбаның синонимдер тізбегінде бірінші болып табылатыны анықталады. Егер солай болса, онда жаңа жазба бірінші бос кеңістікте орналастырылады және ол үшін оған сәйкес сілтемелер орнатылады: ол синонимдер тізбегінде екінші болады, оған бірінші жазба сілтеме жасайды, ал ол бар болса, келесіге нұсқайды.

Егер талап етілетін орынды алатын жазбалар синонимдер тізбегіндегі бірінші болып табылмаса, онда ол орынды «заңсыз» алып жатыр және «заңды меншік иесі» пайда болған кезде оны «шығарып тастау», яғни жаңа орынға көшіру қажет. Ауыстыру механизмі файлға енгізілген синонимі бар жаңа жазбаны енгізуге ұқсас.

Осы жазба үшін бірінші бос орын іздестіріледі және тиісті сілтемелер реттеледі: ауыстырушы жазба үшін синонимдер тізбегіндегі алдыңғы жазбада ауыстырылатын жазбаның жаңа орнына көрсеткіш енгізіледі, аталған ауыстырлатын жазбадағы көрсеткіштер бұрынғыдай қалады.

«Заңсыз» жазба аусытырылған соң енгізілетін жазба өзінің заңды орнына орналастырылады және синонимдердің жаңа тізбегінде бірінші жазба болып енгізіледі.

Жазбаларды жою тетіктері көбінесе стратегиядағы асыра толтыру саласымен жою тетіктеріне ұқсас және келесі әрекеттерге алып келеді.

Егерде жойылатын жазба синонимдер тізбегіндегі бірінші жазба болатын болса, онда жоюдан кейін оның орнына синонимдер тізбегінің келесі (екінші) жазбасы ауыстырылады және бар болған жағдайда, синонимдер тізбегінің үшінші жазбасындағы көрсеткіштерге тиісті түзету жүргізіледі.

Егерде синонимдер тізбегініңортасында тұратын жазба жойылатын болса, онда көрсеткіштерді түзету ғана жүргізіледі: алдыңғы жазбадағы жойылатын жазбаның көрсеткіші келесі жойылатын жазба көрсеткішіне ауыстырылады, жойылатыннан кейінгі жазбадағы алдыңғы жазба көрсеткіші алдыңғы жойылатын жазба көрсеткішіне ауыстырылады.

Индекс файлдары. Файл құрылымдарында хэш-адрестеудің жоғары тиімділігіне қарамастан, үнемі тиісті функцияны табу мүмкін емес, сондықтан бірінші кілт бойынша қолжетімділік ұйымдастырылған кезде, индекс файлдары кеңінен қолданылады.

Индекс файлдары екі бөліктен тұратын файлдар ретінде ұсынылуы мүмкін. Алдымен блоктардың бүтін сандарын қабылдайтын индекстік салаорналасқан, одан кейін файлдың барлық жазбалары кезеңмен орналасқан негізгі сала орналасады.

Кейбір жүйелерде индекс файлдары қайталама кілтке кіру үшін пайдаланылатын инверттелген тізімдер түріндегі ұйымдастырылған файлдар деп аталады. Индекстің және негізгі салаларының ұйымдастырылуына байланысты файлдардың екі түрі бар: тығыз индексі бар (индексті-тікелей файлдар) және тығыз емес индекспен (индексті-кезеңді файлдар).

Тығыз индексі бар файлдар немесе индекс-тікелей файлдар. Бұл файлдарда негізгі сала еркін тәртіпте орналасқан, бірдей ұзындықтағы жазбалардың тізбегінен тұрады, ал индексті жазба құрылымы келесі түрлерден тұрады:

Кілттің мағынасы	Жазбаның нөмірі

Мұнда *кілттің мағынасы* – ол бірінші кілттің мағынасы, ал жазбаның нөмірі – ол бірінші кілттің аталған мағынасына ие негізгі саладағы жазбаның реттік нөмірі болып табылады.

Индекстік файлдар жазбаны бірегей түрде анықтайтын бастапқы кілттер үшін жасалғандықтан, онда оларда бірінші кілттің бірдей мағынасы ие екі жазбалар болмайды. Негізгі саладағы әрбір жазбаның тығыз индексі бар индекс файлдарында индекс саласынан бір жазба бар. Индекс саласындағы барлық жазбалар кілттің мағынасы бойынша реттеледі, сондықтан реттелген кеңістікте іздестірудің барынша тиімді әдістерін қолдануға болады.

Еркін жазбаға қол жеткізудің ұзақтығы абсолютті мағыналармен бағаланбайды, әдетте дискі болып табылатын сыртқы жад қондырғысына өтініштер саны бойынша бағаланады. Дискіге өтініш шұғыл жадыдағы барлық өңдеулермен салыстырғанда барынша ұзақ операциялар болып табылады.

Реттелген алаптағы барынша тиімді іздестіру алгоритмі логарифмдік немесе бинарлық іздестіру болып табылады. Ықтималдылық теориясында ол жартылай бөліну әдісі деп аталады. Осы әдістің пайда болуы артиллериялық ату теориясына байланысты. Нысанаға жету үшін, нысана орналасқан барлық кеңістік екіге бөлінеді. Содан кейін атыс объектінің орналасуы болжанған жартысында жүргізіледі. Снарядтың түсу нүктесін белгілейді. Егер асып кетсе, онда түзету жасалады – снарядтың түсу нүктесі мен кесіндінің жартысы арасындағы сала қайтадан екі бөлікке бөлінеді және тағы атады.

Бұл жағдайда іздестіру қадамының жоғары саны (нысанаға дәл тию) іздестіру кеңістігіндегі элементтердің (нысаналардың) жалпы санының қосарлы логарифмімен анықталады:

$$T_n = \log_2 N, \qquad (3.1)$$

мұндағы N — элементтер саны.

Жазбаларды іздеген кезде бірінші кілттің берілген мағына бойынша дискіге орналастыру саны ғана маңызды болып табылады. Алдымен индекстік жазбаны іздестірудің қосарлы алгоритмі қолданылатын индекстік салада іздестіру жүргізіледі, ал содан кейін негізгі салада іздеутікелей адрестеу арқылы жазбаның нөмірі бойынша іздестіру жүргізіледі. Жазбаға кірудің максималды уақытын бағалау үшін, іздестіру үдерісінде дискіге өтініштер санын анықтау қажет.

Формулаға сәйкес (3.1) жазбаны іздестіру кезінде дискіге өтініштер саны келесі жолмен анықталады:

$$T_n = \log_2 \cdot N_{инд.\ обл} + 1,$$

мұндағы $N_{инд.обл}$— барлық жазбалар орналастырылатын индексті блоктар саны.

Индексті блоктағы жазбаны іздестіруден кейін негізгі салаға тағы жүгіну қажет екенін ескере отырып, формулаға бірлік қосылды (+ 1).

Тығыз индексті файлды ұйымдастыру сұлбасы

Блок	Кілттер	Жазбаның № сілтеме	Еркін аймақ	салалар
1блок	01-10/01	3		Индексті сала
	02-20/02	4		
	03-20/00	5		
2блок	06-40/00	7		
	07-50/01	8		
	08-30/01	9		
3блок	10-44/01	1		
	11-44/02	2		
	09-35/01	6		
4блок	17-20/03	10		
	18-40/02	11		
	20-35/02	12		

Жазбаның нөмірі	Кілт	Мазмұны	Негізгі сала
1	10-44/01	Математика	
2	11-44/02	Физика	
3	01-10/01	Информатика	
4	02-20/02	Ақпарат теориясы	
5	03-20/00	дерекқор	
6	09-35/01	АСО және У интерфейстері	
7	06-40/00	Ақпаратты қорғау	
8	07-50/01	АСТПП және АЖЖ	
9	08-30/01	Бағдарламалау тілі	
10	17-20/03	Операциялық жүйелер	
11	18-40/02	Интегралды қызмет көрсетудің цифрлық желілері	
12	20-35/02	Бағдарламалау технологиялары	

Тығыз индексі бар файлдардағы жаңа жазбаларды қосу және жою операцияларының қалай жасалатынын қарастырайық. 3.1-кестеде дискілік кеңістікте осындай файлды ұйымдастырудың сұлбасы (еркін аймақтар фонмен бөлінген) берілген.

3.1-кестеде көрсетілгендей, файл екі сала түрінде: негізгі және индекс ұйымдастырылады. Негізгі салада шешуші өрістер, жазбалардың сандары мен мазмұны сақталады. Индекс саласында шешуші өрістердің мәндері және негізгі саладағы жазбалар санына сілтемелер сақталады.

Қосу операциясы кезінде деректер негізгі саланың шетіне жазу жүзеге асырылады. Бұл ретте, индексті салаға тиісті шешуші өрістің мағыналары мен жазбаның нөміріне сілтеме қосылуы керек, мұнда ақпарат жазбалардың тәртібі бұзылмайтын етіп қосылуы тиіс. Сондықтан да, файлдың барлық индекс саласы блоктарға бөлінеді және әрбір блокта бастапқы толтырумен бос аймақ (кеңістік) жасалады. Тиісті блоктың тиісті бос кеңістігінде қосымша жазба туралы ақпарат енгізіледі.

Индексті саланы ұйымдастырудың осындай тәсілі, әрбір деңгейде бұйымның қасиетін сипаттау үшін сандық санаттардың барынша ірі саны қаралатын, ЕСКД жүйелерінде көпдеңгейлі жіктеу (әріптік-сандық кодтау) жүйелеріне ұқсас. Ол жүйенің қателігінсіз өнімнің жаңа түрлерін енгізу және оларға тиісті әріптік-сандық кодтары беруге мүмкіндік береді.

Сондықтан да, деректерді сақтаудың физикалық моделін жобалау кезінде сақталатын ақпараттың көлемін нақыт анықтау, оның өсуін болжау және соған сәйкес сақтау саласындағы тиісті кеңейтуді қамтамасыз етуге болады.

Тығыз индексті файлдар түрінде деректерді сақтауды ұйымдастыру кезінде дискіге өтініштер саны формула бойынша жаңа жазбаны қосу кезінде анықталады

$$T_n = \log_2 N_{инд.\ обл} + 1 + 1 + 1,$$

Формуланың мағынасы төмендегідей: өтініштер саны индекс саласына жіберілген өтініштер санымен, оған негізгі блокқа қосылған бір өтінішпен, индексті блокты өзгерту үшін бір өтінішті қосумен және негізгі салаға жазба енгізу үшін бір өтінішті қосумен белгіленеді.

Осылайша, тығыз индексі бар файлдарда бір жазбаны өңдеу кезінде компьютердің дискілік кеңістігіне екі қосымша өтінішті қажет етеді.

Одан туындайтыны, дерекқор файлдарын ұйымдастыру әдістері және олардың тиісті физикалық үлгілері, оны іздестірген кезде дискілік кеңістікке қол жеткізу уақытын қысқартуға және дерекқордың мазмұнын қосу және түзету уақытын қысқартуға бағытталуы тиіс.

Осыған тағы v з емес индексі бар файлдарды ұйымдастырудың әдісі бағытталған.=

Мұндай файлдардағы деректер жазбаларының құрылымы 3.4-суретте берілген түрге ие.

Файл құрылымының осындай ұйымдастырылуы кезінде жаңа жазбаларды қосу процесі тығыз индексті файлдардағы ұқсас әрекеттерден ерекшеленеді. Әрбір жаңа жазба шешуші өрістің мағынасымен анықталған орындағы тиісті блокқа енгізіледі. Бұл жағдайда келесі әрекеттер тізбегі орындалады:

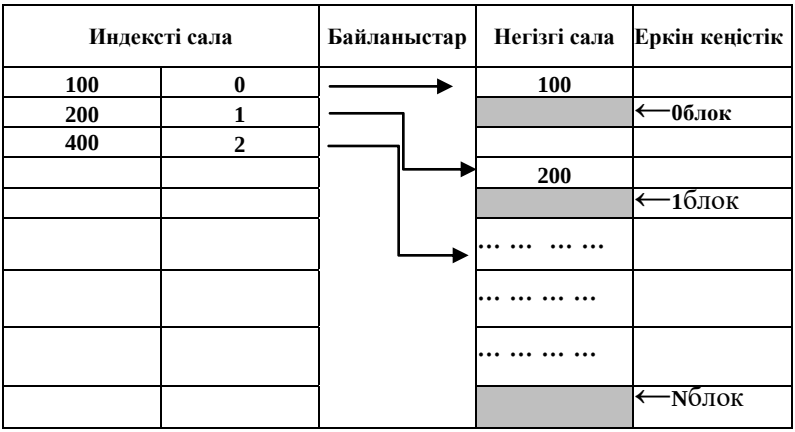
• жаңа жазбаны орналастыру қажет болатын негізгі сала блогының нөмірін белгіленеңді;

• табылған блок негізгі жадыға оқылады;

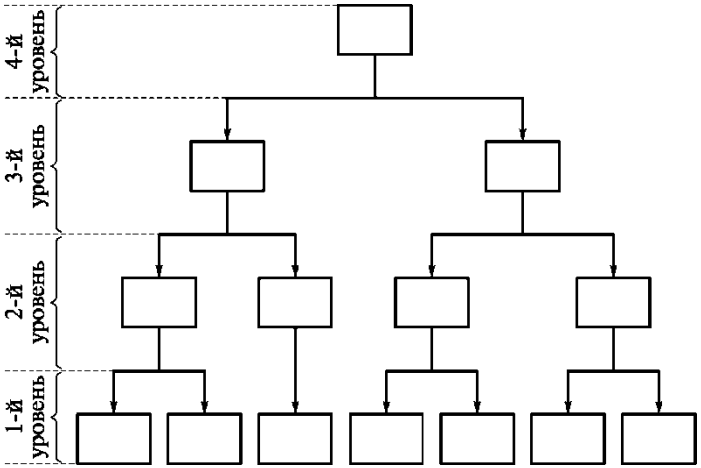
• шұғыл жадыда блокты түзету жүргізіледі;

• түзетілген блок бұрынғы орындағы дискіге жазылады.

Бұл жағдайда жаңа жазба енгізген кезде дискіге өтініштер саны блокты іздестірген кезде бір өтініш қосылған дискіге өтініштер санына тең, оны бұрынғы орынға блогты жазу кезінде орындау қажет. Аталған жағдайда дискіге кіру уақытымен теңдестірілмейтін, оперативті жадыдағы блоктың барлық жазбалары назарға алынбайды.



3.4-сур. Тығыз емес индексті файлдардағы деректер жазбаларының құрылымы



3.5-сур. В-ағаш түріндегі файлдық құрылымды ұйымдастыру мысалы

Демек, мұндай файлдық құрылымды ұйымдастыру кезінде дискілік кеңістікке өтініштер саны әр жазба үшін тығыз индексі бар файлдарға қарағанда бірлікке кем болады, ол жазбалар мағыналы саны кезінде деректерді өңдеу уақытын біршама азайтады, сонымен қатар дискі қондырғысы жұмысының сенімділігін арттырады.

В-ағаш түріндегі индекстерді ұйымдастыру - көп деңгейлі иерархиялық құрылым. Файл құрылымының ұйымдастырылуын жетілдірудің бұл бағыты бастапқыда осы саланың сипаттамасын иерархиялық симметриялы іздестіру ағашының түрінде болжайтын тығыз емес индексі бар файлдардың индексті саласының түрленуіне байланысты. Мұндай ағаштарда әрбір деңгейдегі түйіндердің саны бірдей. Осындай иерархиялық жүйелерді құру кезінде машина жадын ұйымдастырудың теориялық негіздері 1967 жылы АЛГЕМ ассоциативті бағдарламалау тілінің авторы, Мәскеу энергетикалық институтының оқытушысы А.И. Китовпен баяндалған.

Дегенмен, дерекқор теориясы жөніндегі заманауи әдебиетте иерархиялық іздестіру құрылымын В-ағаш деп атау қабылданған («B-tree» ағылш. -теңдестірілген ағаш) («Б-ағаш» деп оқылады).

3.5-суретте В-ағаш түріндегі файл құрылымын ұйымдастыру мысалы көрсетілген.

3.5. Дерекқор үшін жады ұйымдастыру әдістері

Заманауи компьютерлер жадысын ұйымдастыруды іске асыру негізінде екі принцип бар: өтініштердің жергіліктілігі принципі мен құн/өнімділік арақатынасы принципі.

Өтініштердің жергіліктілігі принципі көптеген бағдарламалардың өздерінің барлық командалары мен деректеріне бірдей өтініштерді орындамайтындығы, бірақ өзінің мекенжай кеңістігінің бірқатар бөлігіне басымдылық беруі туралы білдіреді. Деректерді сақтау үшін жадыны ұйымдастырудың келесі аспектілерін қарастырайық:

- жадты иерархиялық ұйымдастыру;
- кэш-жадыны ұйымдастыру;
- негізгі жадты ұйымдастыру;
- виртуалды жад - деректерді қорғауды ұйымдастыру құралы ретінде.

Жадты иерархиялық ұйымдастыру. Заманауи компьютерлердің жадын иерархиялық ұйымдастыру бірнеше деңгейде құрастырылады, мұндабарынша жоғары деңгей көлемі бойынша кіші, шапшаң және және төменгі деңгейге қарағанда байт тұрғысынан қайта есептеудің жоғары құнына ие. Иерархия деңгейлері өзара байланысты: бір деңгейдегі барлық деректерді барынша төменгі деңгейде табылуы мүмкін және барынша төменгі деңгейдегі барлық деректерді келесі төмен жатқан деңгейден табуға болады және одан әрі иерархияның негізіне жеткенге дейін жалғасады.

Жад иерархиясы әдетте көптеген деңгейлерден тұрады, бірақ әр уақытта біз екі жақын деңгейді ғана білеміз. Екі деңгейлі иерархияда болуы немесе болмауы мүмкін ең аз ақпарат бірлігі *блок* деп аталады. Блоктың көлемі тіркелген немесе ауыпалы болуы мүмкін. Егер бұл көлем бекітілген болса, онда жадтың көлемі блоктың өлшемінің көптігі болып табылады.

Барынша жоғары деңгейдегі сәтті немесе сәтсіз өтініш соған сәйкес тигізу (hit) немесе бос жіберу(miss)деп аталады. Тигізу барынша жоғары деңгейден табылған жад объектісіне өтініш, бұл уақытта бос жіберу, оның деңгейде табылмайтынын білдіреді. Тигізу үлесі (hit rate), немесе тиісу коэффициенті (hit ratio) барынша жоғары деңгейден табылған өтініштер үлесі. Кейде ол пайызбен көрсетіледі. Бос жіберулер үлесі (miss rate) барынша жоғары деңгейден табылмаған өтініштер үлесі.

Ақпараттың өңделуі өнімділігін арттырудың жады иерархиясының пайда болуының негізгі себебі болып табылатындықтан, тигізулердің жиілігі мен бос жіберулердің маңызды сипаттамасы болып табылады. Тигізу кезіндегі (hit time) өтініш уақыты – барынша жоғары деңгейгеді иерархияның уақыты, ол келесіден, атап айтқанда, өтініштің тигізу немесе бос жіберу болып табылатынын анықтау үшін қажет уақыттан тұрады. Бос жіберу (miss penalty) жоғалтулары осы блокты талап етілетін қондырғыға (әдетте процессорға) ауыстыру үшін, блокты барынша төменгі деңгейден барынша жоғары деңгейге ауыстыру уақыты болып табылады.

Бос жіберулердегі жоғалтулар екі компоненттен тұрады: қолжетімділік уақыты (access time) – бос жіберу кезінде блоктың бірінші сөзіне өтініш уақыты және жіберілу уақыты (transfer time) – блоктың қалған сөздерін қайта жіберу үшін қосымша уақыт. Қол жеткізу уақыты еске түсіру уақытының төмендеуімен байланысты, ал тасымалдау уақыты екі көршілес жад құрылғысы арасындағы арна өткізу қабілетін байланысты. Қол жетімділік уақыты төменгі деңгейдегі жадының кідірісімен, бұл уақытта жіберілу уақытыекі аралас деңгейлер жады қондырғыларының арасында өткізу арнасы жолағымен барынша байланысты.

Жад иерархиясының кейбір деңгейін сипаттау үшін төрт сұраққа жауап беру керек.

1. Блок иерархияның жоғарғы деңгейінде қайда орналасуы мүмкін (блокты орналастыру)?

2. Жоғарғы деңгейде болған кезде блокты қалай табуға болады (блокты сәйкестендіру)?

3. Бос жіберген уақытта қандай блок ауыстырылуы керек (блоктарды ауыстыру)?

4. Жазба (жазба стратегиясын) кезінде не болады?

Бұл сұрақтардың жауабы кэш-жадыны ұйымдастыруға байланысты.

Кэш-жадыны ұйымдастыру. Кэш жадысының тұжырымдамасы IBM/360 архитектурасына дейін пайда болды. Бүгінгі күні кэш-жады компьютерлердің кез-келген класында, кейбір компьютерлерде - көпше түрінде болады.

Жоғарыдағы барлық терминдер кэш жады үшін пайдаланылуы мүмкін, дегенмен, «жол» («line») сөзі «блок» («block») сөзінің орнына қолданылуы мүмкін.

Кэш-жады сипаттамасы параметрлерінің үлгілік жинағы

Блоктың (жолдың) көлемі....	4... 128 байт
Тигізу уақыты(hit time)	1 — 4 синхрондау тактысы (әдетте 1 такт)
Жоғалту кезіндегі шығындар (miss penalty)	8 — 32 синхрондау тактысы
Қолжетімділік уақыты (access time)	6—18 синхрондау тактысы
Жіберу уақыты (transfer time)	2—22 синхрондау тактысы
Бос жіберулер үлесі(miss	1-20%
	4 Кбайт. 16 Мбайт

Кэш-жадысын ұйымдастыруды барынша түбегейлі қарастырып, жоғарыда қойылған жад иерархиясы туралы сұрақтарға жауап береміз

1. Блок кэш-жадысында қай жерде орналасуы мүмкін? Кэш-жадыда блоктарды орналастыру принциптері олардың ұйымдастырудың үш негізгі түрі бойынша анықталады:

• егер негізгі жады әрбір блогында кэш-жадыкөрсетілуі мүмкін бір ғана тұрақты орын болса, онда бұл кэш-жады *тікелей бейнеленген* (direct mapped) кэш деп аталады.

•. Бұл кэш-жадты барынша қарапайым ұйымдастыру, мұнда кэш-жадының мекенжайына негізгі жадтың мекенжайларын бейнелеу үшін блок мекенжайының кіші санаттары жай ғана қолданылады. Осылайша, негізгі жадтың өзінің мекенжайында бірдей кіші санаттарға ие барлық блоктары бір кэш блогына түседі, яғни,

(кэш-жадының мекенжайы) = (негізгі жады блогының мекенжайы) x mod (кэш-жадыдағы блоктар саны);

• егерде негізгі жадтың кейбір блоктары кэш-жадының кез-келген орнында орналасатын болса, онда кэш *толық ассоциативті* (fully associative) деп аталады;

• егерде негізгі жадтың кейбір блогы кэш-жадыда шектеулі орындарда орналасатын болса, онда кэш *көпше-ассоциативті* (set associative) деп аталады. Әдетте, көпше екі немесе одан көп блок саны тобынан тұрады. Егер көпше *n* блоктарынан тұратын болса, онда мұндай орналасу *n* арналарымен көпше-ассоциативті деп аталады (n-way set associative). Ең алдымен блокты орналастыру үшін көпшені анықтау керек. Көпше жады блогының кіші санаттарымен (индексімен) анықталады:

(кэш-жадтың көпше мекенжайы) = (негізгі жад блогының мекенжайы) ·mod (кэш-жадтың көпше саны).

Блок аталған көпшенің кез-келген жерінде орналастырылуы мүмкін.

Кэш-жадты мүмкін болатын ұйымдастырудың ауқымы өте кең: тікелей бейнелейтін кэш жады - бұл бір арналы көпше-ассоциативті ғана емес, ал *m* блоктарымен толық ассоциативті кэш-жады m-арналы көпше-ассоциативті деп аталуы мүмкін. Заманауи процессорларда, әдетте, тікелей көрсетуі бар кэш-жады немесе екі арналы (төрт арналы) көпше-ассоциативті кэш-жады пайдаланылады.

2. *Кэш-жадтағы блокты қалай табуға болады?* Кэш-жадтағы әрбір блокта негізгі жадтағы қандай блоктың кэш-жадтыбілдіретінін көрсететінін көрсететін мекенжайлық *тег* бар. Бұл тегтер әдетте процессор өңдеген жады блогының мекенжайымен салыстырылады.

Одан өзге, кэш-жады блоктары сенімді немесе қолдануға жарамды ақпараттан тұратынын анықтау әдісі қажет. Осы проблеманы шешудің барынша таралған әдісі–тегке дәлділік битін (valid bit) қосу деп аталады.

Көпше-ассоциативті кэш-жадтағы мекенжайлау процессордан түскен мекенжайды үш бөлікке бөлумен жүзеге асырылады: кэш-жадтың ішкі блогының байтын таңдау үшін пайдаланылатын ығысу өрісі; көпше нөмірін анықтайтын индекс өрісі; салыстыру үшін пайдаланылатын тегтің өрісі.

Егер кэш-жадының жалпы көлемі бекітілген болса, ассоциативтілік дәрежесін ұлғайту блоктар санын көпшеге ұлғайтуға алып келеді; бұл ретте индекстің көлемі азаяды және тегтің көлемі артады;

3. *Мүлт кету кезінде қандай кэш-жады блоктарын ауыстыруға болады?* Мүлт кету кезінде кэш-жабы бақылаушылары ауыстыруға жататын блокты таңдауы керек. Тікелей бейнелейтін ұйымды пайдаланудың артықшылығы, мұнда апараттық шешімдердің барынша қарапайым болып келуінен тұрады. Басқа таңдау жоқ: тек бір блок тексеріледі және тек осы блок ауыстырылуы мүмкін. Толық ассоциативті немесе көпше-ассоциативтілік кезінде кэш-жадыда мүлт кеткен кезінде үміткерді таңдау қажет болатын бірнеше блоктар бар. Әдетте, блоктарды ауыстыру үшін екі негізгі стратегия қолданылады: кездейсоқ және Least-Recently Used (LRU).

Бірінші жағдайда, блок-үімткерлер тегіс таратуға ие болу үшін кездейсоқ таңдалады. Кейбір жүйелерде аппаратураны жөндеу кезінде әсіресе пайдалы болатын туынды әрекетті алу үшін ауыстырудың псевдо-кездейсоқ алгоритмі пайдаланылады.

Екінші жағдайда, жақын арада талап етілуі мүмкін ақпараттың тасталу ықтималдығын азайту үшін блоктарға барлық өтініштер бекітілген. Ең ұзақ пайдаланылмаған блок ауыстырылады.

Кездейсоқ әдіс артықшылығы, оны аппараттық құралдарда іске асу жеңіл болуынан тұрады. Трассаны қолдайтын блоктар саны көбейгенде, LRU алгоритмі барынша қымбатқа түседі және жиі жақын болып келеді. 3.2 -кестедеLRU алмастыру алгоритмін және кездейсоқ алгоритмді пайдаланған кездегі мүлт кетудің әртүрлі үлестерінің айырмашылықтары көрсетіледі.

4. *Жазу кезінде не болады?* Нақты бағдарламаларда кэш-жадыға қол жеткізген кезде, оқуға қол жеткізу басым болады.

3.2-кесте

LRU алмастыру алгоритмін және кездейсоқ алгоритмді (Random) қолданған кездегі мүлт кету үлесі

Кэш-жадының көлемі, Кбайт	Ассоциативтілік, %					
	2-арналы		4-арналы		8-арналы	
	LRU	Rando	LRU	Rando	LRU	Random
16	5,18	5,69	4,67	5,29	4,39	4,96
64	1,88	2,01	1,54	1,66	1,39	1,53
256	1,15	1,17	1,13	1,13	1,12	1,12

Командалардың барлық өтініштері оқу бойынша өтініштер болып табылады және көптеген командалар жадыға жазбайды. Әдетте жазба операциялары жалпы жады трафигінің 10% кем. Жалпы жағдайды барынша шапшаң жасау ниеті оқу операцияларын орындау үшін кэш-жадты оңтайландыруды білдіреді, алайда жоғары өнімді деректерді өңдеу кезінде жазу әрекеттерінің жылдамдығын елемеу мүмкін емес.

Жалпы жағдай да қарапайым болып табылады. Кэш-жадыдағы блокты оқылған және салыстырылған кездегі бір уақытта оқылуы мүмкін. Осылайша, блоктың оқылуы блоктың мекенжайы қол жетімді болғанда бірдден басталады. Егерде оқутигізумен болған жағдайда, блок дереу процессорға жіберіледі. Егер бос жіберу орын алатын болса, онда алдын-ала оқу блогынан қандай да болмасын пайда болмайды, алайда ешқандай зиян да жоқ.

Алайда, жазу операцияларды орындау кезінде жағдай түбегейлі өзгереді. Атап айтқанда процессор (әдетте 1 бастап 8 дейінгі байт) жазу көлемін анықтайды және блоктың тек осы бөлігі өзгертілуі мүмкін. Жалпы, блоктағы оқу-түрлендіру-жазу операцияларының кезеңділігін орындауды (түпнұсқа блогты оқып, оның бөлігін түрлендіру және блоктық жаңа мағынасын жазу) білдіреді. Одан өзге, блокты түрлендіру өтініштің дәл түсуі болып табылғандығына көз жеткізу үшін, тег тексеріліп жатқанда басталмайды. Тегтерді тексеру басқа тапсырмамен қатар орындалмайтындықтан, жазу операциялары оқу операцияларына қарағанда көп уақыт алады.

Әртүрлі машиналарда кэш-жадыны ұйымдастыру жазбаны орындау стратегиясынан ерекшеленеді. Кэш-жадыда жазба орындалған кезде, екі базалық мүмкіндіктер болады:

• өтпелі жазба (write through, store through) — ақпарат екі орынға жазылады (кэш-жады блогына және жадының барынша төмен деңгейдегі блогына);

• кері көшірмеленетін жазба (write back, copy back, store in) — ақпарат тек кэш-жады блогына жазылады. Түрлендірілген кэш-жады блогы негізгі жадқа ауыстырылған кезде ғана жазылады. Ауыстыру кезіндегі блоктарды көшіру жиілігін қысқарту үшін әдетте әрбір кэш-жады блогымен түрлендірілген бит деп аталатын (dirty bit) байланысады. Осы мәртебелік бит кэш-жадыдағы блоктың түрлендірілгенін көрсетеді. Егер ол түрлендірілмесе, онда кері көшірмелеуден бас тартылады, себебі барынша төменгі деңгейде кэш-жадыдағы ақпарат болады.

Жазбаны ұйымдастырудың екі тәсілі де артықшылықтар мен кемшіліктерге ие. Кері көшірмені жазу кезінде операциялар кэш-жады жылдамдығымен орындалады және бір блоктағы бірнеше жазбалар барынша төменгі деңгейдегі жадтағы бір жазбаны ғана талап етеді.

Бұл жағдайда негізгі жадыға өтініш аз жүретіндіктен, одан кейін мультипроцессорлық жүйе үшін өте тартымды болатын жадыны өткізу жолағының аз болуы талап етіледі. Өтпелі жазба кезінде оқу бойынша мүлт кетулер жоғары деңгейдегі жазбаға әсер етпейді. Сонымен қатар, кері көішрмелі жазбаға қарағанда, іске асыру үшін өтпелі жазба жеңілірек. Өтпелі жазбаның тағы бір артықшылығы, негізгі жадының деректердің барынша жаңа көшірмесіне ие болуы болып табылады. Ол мультипроцессорлы жүйелерде маңызды, сондай-ақ кіруді (шығуды) ұйымдастыру үшін маңызды.

Өтпелі жазбаны орындағаннан кейін процессор жазбаның аяқталуын күтетін болса, онда ол жазба үшін (write stall) тоқтатылған деп айтылады. Жазба бойынша тоқтатудың азайтудың жалпы әдісі жадтың мазмұнын жаңарту кезінде командаларды орындауды жалғастыруға мүмкіндік беретін жазу буферді (write buffer) пайдалануға байланысты. Жазба бойынша тоқтаулары жазба буфері кезінде туындауы мүмкін.

Жазба кезінде мүлт кеткен кезде екі қосымша мүмкіндік бар: кэш-жадыға жазбаны орналастыру және кэш-жадыға жазбаны орналастырмау.

Кэш-жадыда жазбаны орналастыру (write allocate) (жазба кезіндегі таңдау деп те аталады (fetch on write)) бұл блок кэш-жадына жүктелетінін, содан кейін тигізе отырып жазбаны орындау кезінде ұқсас қолданылатын әрекеттер орындалады. Бұл оқу кезіндегі мүлт кетулерге ұқсайды.

Кэш-жадыға жазбаны орналастырмау (қоршаған ортаға жазба деп те аталады (write around)), блоктың барынша төменгі деңгейге түрленетінін және кэш-жадыға жүктелмейтінін білдіреді.

Әдетте, кері көішрмелейтін жазбаны іске асыратын кэш-жады, кэш-жадыда жазбаны орналастыру үшін қолданылады (келесі жазба осы блокта ұсталады деп үміттенеді), ал өтпелі жазбадан тұратын кэш-жадыда кэш-жадыға жазбаны орналастыру жиі пайдаланылмайды (өйткені осы блокқа кейінгі жазба бәрібір жадыға кіреді).

Кэш жадының өнімділігін арттыру жолдарын қарастырамыз. Кэш-жадындағы жүйеде жадқа қолжетімділіктің орташа уақыты үшін формула келесідей болып келеді:

Орташа қолжетімділік уақыты = Тигізу кезіндегі өтініш уақыты + Мүлт кетулер саны х Мүлт кету кезіндегі шығындар.

Осы формула кэш-жадының жұмысын оңтайландыру жолдарын айқын көрсетеді: жіберіп алулардың үлесін қысқарту, бос жіберу кезінде шығынды қысқарту, сондай-ақ, тигізу кезінде кэш-жадыға өтініштер уақытын қысқарту.

3.3-кестеде қазіргі уақытта кэш-жадының өнімділігін арттыру үшін қолданылатын әр түрлі әдістер қысқаша ұсынылған.

Кэш-жадының өнімділігін арттыру әдістері

Әдіс	Мүлт кетулер үлесі	Мүлт кетулері кезіндегі шығындар	Тигізу кезіндегі өтініштер уақыты (ескерту)
Блоктың өлшемен ұлғайту	+	-	0
Ассоциативтіліктің дәрежесін арттыру	+		-1
Қосалқы кэші бар кэш-жады	+		2
Псевдоассоциативті кэштер	+		2
Командалар мен деректерді аппаратты алдын ала таңдау	+		2 (деректерді алдын ала таңдау күрделенген)
Компиляторды басқара отырып алдын ала таңдау	+		3 (бұғатталмайтын кэш-жадыны талап етеді)
Мүлт кетулерді азайту үшін арнайы әдістер	+		0 (бағдарламалық қамтамасыз ету мәселесі)
Жазбаларды оқу бойынша мүлт кетулердің басымдылықтарын орнату		+	1 (тек бір процессорлық жүйелер үшін)
Кіші блоктарды қолдану		+	+1 (өтпелі жазба + + бір сөзге кіші блок жазбаларға көмектеседі)
Талап етілетін сөзді бірінші жіберу		+	2
Бұғатталмайтын кэштер		+	3
Екінші деңгейдегі кэштер		+	2 (жеткілікті қымбат жабдық)
Шағын көлемдегі қарапайым кэштер	-		0
Кэш-жадыны индексациялау кезінде мекенжайларды түрлендірулерді аралау			+2
Жазба кезінде шапшаң түсуі үшін жазба операцияларын конвейерлеу			+1

Белгілі бір әдістерді қолдану негізінен әзірлеу мақсаттарымен анықталады; бұл ретте заманауи компьютерлердің құрылымдары жүйенің барлық жағынан теңгерімделгендігін қамтамасыз етеді.

Негізгі жадыны ұйымдастыру. Заманауи компьютерлердегі негізгі жады өзімен жады иерархиясының келесі деңгейін білдіреді. Негізгі жад кэш-жадының сұраныстарын қанағаттандырады және кіріс (шығыс) интерфейсі ретінде қызмет етеді, себебі ол кіріс көзі және шығыс көзі үшін тағайындау орны болып табылады. Негізгі жадтың өнімділігін бағалау үшін екі негізгі параметр пайдаланылады: кідіру және өткізу жолағы қолданылады. Негізгі жадының кідірісі кэш-жадыға байланысқа ие, ал өткізу жолағы немесе өткізу қабілеті кіріс (шығысқа) жатады. Екінші деңгейдегі кэш-жадының танымалдылығының және осындай кэш-жадыдағы блок өлшемінің артуына байланысты, негізгі жадының өткізу жолағы кэш-жады үшін де маңызды болып табылады.

Жадты кешіктіру дәстүрлі түрде екі параметрлермен бағаланады: қол жеткізу уақыты (access time) мен жады циклінің ұзақтығы (cycle time). Қолжеткізу уақыты оқу сұрауын беру және сұралған сөздің жадқа түсу сәті уақыт аралығынан тұрады. Жад циклінің ұзақтығы жадыға екі кезеңде өтініштер арасындағы минималды уақытпен анықталады.

Заманауи компьютерлердің негізгі жадысы статистикалық және динамикалық есте сақтау еркін іріктейтін (ЕІЕҚ) қондырғыларының микросұлбаларында іске асырылады асырылады. Статикалық ЕІЕҚ (СЕІЕҚ) микросұлбалары аз қолжетімділікті уақытқа ие және қалпына келтіру циклін қажет етпейді. Динамикалық ЕІЕҚ (ДЕІЕҚ) микросұлбалары үлкен сыйымдылық пен аз құнмен ерекшеленеді, бірақ олар қалпына келтіру сұлбаларын қажет етеді және оларға қолжеткізудің біршама көп уақытына ие.

ДЕІЕҚ даму процесінде оның сыйымдылығының өсуіне байланысты мұндай микросұлбалардың құнына қатысты басты мәселе, мекенжайлық желілер саны және тиісті тұрқының құны туралы мәселе болды. Мекенжайлар беруге қажет тұрқы байланыстарының санын жартысына қысқартуға мүмкіндік беретін мекенжай желілерін мультиплекстеудің қажеттілігі туралы шешім қабылданды. Сондықтан да, ДЕІЕҚ жүгіну әдетте екі кезеңде жүреді. Бірінші кезең RAS (row-access strobe — жол мекенжайның бағанын) беруден басталады, ол микросұлбада жолдың келіп түскен мекенжайын бекітеді. Екінші кезең бағанның мекенжайын көрсету және осы мекенжайды бекітетін және микросұлбаның шығыс буферлері жұмысына рұқсат беретін мекенжайды қайта қосудан тұрады.Бұл сигналдардың атаулары жолдың мекенжайын және бағанның мекенжайын көрсету арқылы шешуге болатын элементтерге, әдетте тікбұрышты матрица болып табылатын микросұлбаны ішкі ұйымдастыруға байланысты.

ДЕІЕҚ ұйымдастырудың қосымша талаптары оның жай-күйін мерзімді қалпына келтіру қажеттілігі болып табылады. Бұл ретте, жолдағы барлық биттерді бір уақытта қалпына келтіруге болады, мысалы, осы жолды оқу арқылы. Сондықтан да, негізгі компьютерлік жадыдағы ДЕІЕҚ барлық микросұлбаларының барлық кезеңдеріне белгілі бір уақыт аралығында - тәртіппен 8 мс шегінде кезеңді байланысу жүргізілуі тиіс.

Осы талап, компьютердің негізгі жады жүйесінің кейде процессорға қол жетімді емес екенін білдіреді, өйткені ол әрбір микросұлбада қалпына келтіру сигналдарын жіберуге мәжбүр болады. ДЕІЕҚ әзірлеушілері қалпына келтіруге жұмсалған уақытты жалпы уақыттың 5%-нан кем деңгейінде сақтауға тырысады. Әдетте, жад бақылаушылары ДЕІЕҚ кезеңді қалпына келтіруге арналған жабдықтан тұрады.

Динамикалық, статикалық СЕІЕҚ айырмашылығы регенерацияны қажет етпейді және оларға қол жеткізу уақыты циклдің ұзақтығына сәйкес келеді. Шамамен бірдей технологияны қолданатын микросұлбалар үшін ДЕІЕҚ сыйымдылығы СЕІЕҚ сыйымдылығынан 4-8 есе артық, бірақ соңғылары 8-16 есе кіші цикл ұзақтығы мен ірі құнға ие. Осы себептерге байланысты, 1975 жылдан кейін сатылған дерлік кез-келген компьютердің негізгі жадысында жартылай өткізгіш ДЕІЕҚ микросұлбалары қолданылды (бұл жағдайда кэш-жадты құру үшін СЕІЕҚ пайдаланылды). Бірақ ерекше жағдайлар болды. Мысалы, Cray Research компаниясының шұғыл жадысында СЕІЕҚ қолданылды.

Процессордың жылдамдығының өсуіне қарай, теңдестірілген жүйені қамтамасыз ету үшін, негізгі жад сыйымдылықтары сызықты түрде өсуі керек. Соңғы жылдары динамикалық жад микросұлбаларының сыйымдылығы жылына 60%-ға арта отырып, әр үш жыл сайын төрт есеге артты. Өкінішке орай, сол сұлбалардың жылдамдығы аталған кезеңде айтарлықтай баяу қарқынмен өсті (жылына шамамен 7%). Бұл уақытта, 1987 жылдан бастап процессорлар өнімділігі жылына 50% -ға артты.

Негізгі жад есептеу жүйелерінің жылдамдығымен заманауи процессорлар өнімділігін келісу ең маңызды мәселелердің бірі болып қалып отыр. 3.4-бөлімде берілген кэш-жадыныңмөлшемін ұлғайту және кэш-жадыны көпдеңгейлі ұйымдастыруды енгізу есебінен ақпаратты өңдеу өнімділігін арттыру әдістері жүйелер құны көзқарасы тұрғысынан жеткілікті тиімсіз болуы мүмкін. Сондықтан да, уақытылы әзірлемелердің маңызды бағыты ДЕІЕҚ ұйымдастырудың арнайы әдістерін қоса, оны ұйымдастыру есебінен өткізу жолағын немесе жадының өткізушілік қабілетін арттыру әдісі болып табылады.

Кэш жадыны ұйымдастыру үшін, өткізу жолағын жоғарылатудан гөрі жадының кешігуін азайту маңызды. Дегенмен, жадыны өткізу жолағын ұлғайтқан кезде, мүлт кету кезіндегі шығынды елеулі ұлғайпастан, кэш-жады блоктарының өлшемін арттыруға болады.

Жад өткізу жолағын арттырудың негізгі әдістері: жадының санатын немесе «енін» арттыру; жадының қабатталуын пайдалану; жадының тәуелсіз банктерін пайдалану; жады банктеріне конфликтсіз байланысу режимін қамтамасыз ету; жад динамикалық микросұлбасының арнайы жұмыс режимдерін пайдалану.

Негізгі жады санатын арттыру. Көптеген жағдайларда бірінші деңгейдегі кэш-жады сөздегі санаттардың санына сәйкес келетін деректер шинасының физикалық еніне ие, өйткені көптеген компьютерлер осы ақпарат бірліктеріне өтініштер жасайды. Екінші деңгейлі кэш-жадысыз жүйелерде негізгі жады деректер шинасының шина ені кэш-жады деректері шиналарының еніне сәйкес келеді. Кэш-жадының және негізгі жадтың шинасының енін екі есе көбейту немесе төрт есе көбейту жады жүйесін өткізу жолағының енін екі еселейді немесе төрт есе көбейтеді.

Барынша кеңірек шиналарды іске асыру кэш-жады мен процессор арасында деректерді мультиплекстеуді қажет етеді, өйткені процессордағы деректерді өңдеудің негізгі бірлігі сөз болып қалады. Осы мультиплексорлар процессорға ақпараттық ағын түсуінің сынжолында болады. Екінші деңгейдегі кэш-жады бұл мәселені біршама жеңілдетеді, себебі бұл жағдайда мультиплексорлар екі деңгейлі кэш жады арасында орналасуы мүмкін, яғни олармен енгізілетін кідіріс соншалықты шиеленіскен емес. Жад санатының артуына байланысты басқа проблема, жүйені пайдалану орнында қолданушының өзімен орындалатын, жадты кезең-кезеңмен кеңейту үшін шағын көлемді (инкрементті) анықтау қажеттілігімен белгіленеді. Жадтың енін екі еселеу немесе төрт еселеу осы ең төменгі инкрементті екі есе көбейтуге немесе төрт есеге көбейтуге алып келеді. Сонымен қатар, кең жадыдағы жүйелерде қателерді түзетуді ұйымдастыру проблемалары да бар.

Қабаттанатын жады. Көптеген микросұлбалар жүйесінде жадының болуы осындай ұйымға тән әлеуетті параллелизмді пайдалану мүмкіндігін береді. Ол үшін жад микросұлбалары сөздердің бекітілген санын қамтитын банктер немесе модульдергежиі біріктіріледі, мұнда осы банктердің біреуіне ғана кез келген уақытта байланысуға болады. Жоғарыда айтылғандай, нақты жүйелерде мұндай жады банктеріне қолжетімділік жылдамдығы кейде ғана жеткілікті болады. Одан туындайтыны, қолжетімділіктің жоғары жылдамдығын алу үшін көптеген жады банктеріне бір мезгілде қол жеткізуді жүзеге асыру қажет.

Бұл үшін қолданылатын жалпы әдістердің бірі жадыны қабаттау деп аталады. Жады банкінің қабаттауы кезінде, әдетте, жадыдағы банктер *N* жадының кезеңді мекенжайларының *i, i*+ 1, *i* + 2, ..., *i* + (N - 1) әр түрлі N банктеріне келетіндей реттеледі. і-ші жады банкінде тек мекенжайлары келесі түрде болатын сөздер ғана бар:

$$kN + i,$$

мұндағыk = M- 1 (M — бір банктегі сөздер саны).

Әр банктегі деректерге қол жеткізуді қамтамасыз ететін болсақ, оның жекелеген банкіне қарағанда, жалпы қолжетімділік жылдамдығын N жадыға дейін қол жеткізе аласыз. Осындай қабатталған құрылымдарды іске асырудың әр түрлі әдістері бар. Олардың көпшілігі әртүрлі банктерге мекенжайлардың жіберілуін қамтамасыз ететін және банктерден келіп түсетін деректерді мультиплекстеуге арналған конвейерлерді еске түсіреді. Осылайша, қабаттанудың дәрежесі немесе коэффициенті жады банктеріне мекенжайларды бөлуді анықтайды. Мұндай жүйелер оқу кезінде кэш-жадына ақпарат жинаған кезде, сондай-ақ кэш-жадысының кері көшірмелеу механизмдерін пайдаланған жағдайда тән болып табылатын, жадтың кезеңді мекенжайлары бойынша өтініштерді оңтайландырады. Дегенмен, кезеңді орналасқан жадыдағы сөздерге қол жеткізу талап етілетін болса, қабаттасқан жадтың өнімділігі айтарлықтай азайтылуы мүмкін.

Жадтың қабаттануын жинақтау - бірнеше жад бақылаушылары жадыдағы банктерге (немесе жадының қабаттанған топтарына) тәуелсіз жұмыс істеуіне мүмкіндік беретін бірнеше тәуелсіз өтініштерді іске асыру мүмкіндігі.

Егер жады жүйесі көптеген тәуелсіз сұраныстарды (кэш-жадымен жұмыс істеген кезде, көппроцессорлы және векторлы өңдеуді іске асыру кезінде) қолдау үшін әзірленген болса, жүйенің тиімділігі көп дәрежеде әр түрлі банктерге тәуелсіз сұраныстардың түсу жиілігіне байланысты болады. Кезеңді мекенжайларға өтініш немесе тақ саннан өзгешеленетін мекенжайларға қатысты жалпы жағдайда, қабатталған жады дәстүрлі жолдарымен жақсы өңделеді. Мекенжайлардағы кезеңді өтініштер саны айырмашылығы тақ болатын болса, қиындықтар пайда болады. Ірі компьютерлерде пайдаланылатын шешімдердің бірі - ол жады банктерінің санын едәуір арттыру арқылы осындай кері өтініштердің ықтималдығын статистикалық түрде азайту. Мысалы, NEC SX/3 суперкомпьютерінде 128 жады банктері пайдаланылады.

Ұқсас проблемалар бағдарламалық және аппараттық құралдармен шешілуі мүмкін.

ДЕІЕҚ ерекшелігі қасиеттерін пайдалану. ДЕІЕҚ өтініш екі кезеңнен тұрады: дерекқор жолына қол жеткізу және бағанға өтініш.

Бұл ретте, жолдың биттерін буферлеуді жүзеге асыру микросұлбасының ішінде жүреді, ең алдымен бағанға өтініш жасау орындалады. Жолдың өлшемі әдетте жад кристалы сыйымдылығына шаршылы түбір болып табылады: 1 Мбит үшін 1024 бит, 4Мбитке 2048 бит және т.б. Өнімділікті арттыру мақсатында барлық заманауи жад микросұлбалары, жолға өтініш жасау үшін қосымша уақытсыз буферге қосымша өтінішті орындауға мүмкіндік беретін, синхрондау сигналдарын беру мүмкіндігін қамтамасыз етеді. Мұндай оңтайландырудың үш әдісі бар:

- блоктік режим;
- жол (парақты) режимі;
- статикалық бағанның режимі.

Блокты режим (nibblemode) әрбір RAS сигналы үшін төрт кезекті ұяшықтың шығуын қамтамасыз етуі мүмкін.

Парақты режим (page mode) кезінде буфер статикалық ЕІЕҚ ретінде жұмыс істей алады; бағанның мекенжайы өзгерген кезде, буфердің туынды биттеріне қолжетімділік бағанының мекенежайы өзгерген кезде, жолға жаңа өтініш болғанша немесе қалпына келтіру уақыты келгенше дейін мүмкін болады.

Статикалық баған (static column) режимі, тек бағанның мекенжайын өзгерту үшін бағанның мекенжайын әрқашан ауыстырып қосу міндетті еместігін қоспағанда, парақтық режимге өте ұқсас.

Сыйымдылығы 1 Мбит ДЕІЕҚ микросұлбаларынан бастап, көптеген ДЕІЕҚ осы режимдердің кез-келгеніне жол береді, тиісті қосылыстарды таңдау арқылы кристалды тұрқыға орнату сатысында режимді таңдау орындалады. Бұл операциялар ДЕІЕҚ үшін жады циклінің ұзақтығын анықтауды өзгертті. Мұндай оңтайландырудың артықшылығы, ол ішкі ДЕІЕҚ сұлбаларына негізделген және жүйенің өзіндік құнын сәл арттырады, бұл жадының өткізу қабілетін төрт еселеуге мүмкіндігін береді. Мысалы, nibble mode жадының қабаттануына ұқсас қолдау үшін әзірленген. Кристал бір уақытта төрт бит оқиды және оларды төрт оңтайландырылған циклдерге береді. Егер шина бойынша тарату уақыты оңтайландырылған циклдың уақытынан аспайтын болса, төрт есе қабаттаумен жадты ұйымдастырудағы жалғыз қиындық синхронды сигналдарды басқарудың бірнеше күрделі сұлбасынан тұрады. Парақтық режим және статикалық баған режимі, әлдеқайда бірнеше күрделі басқару кезінде қабаттанудың барынша ірі дәрежесін қамтамасыз ете отырып қолданылуы мүмкін. ДЕІЕҚ әзірлеудегі үрдістердің бірі оларда үш жағдайдан тұратын буфердің болуы. Ол жүйеде жадыдағы кристалдардың ірі санының дәстүрлі қабаттануын іске асыру үшін, жадының әрбір банкі үшін буферлік микросұлбалардың болуы көзделуі тиіс.

ДЕІЕҚ жаңа түрлері ДЕІЕҚ және процессор арасындағы интерфейсті одан әрі оңтайландыру мүмкіндігін ескере отырып әзірленген. Мысал ретінде RAMBUS компаниясының өнімдерін атап өтуге болады. Бұл компания стандартты ДЕІЕҚ толтырумен айналысады және оның жеке компоненттерінің жұмысын емес, жады жүйесінің жұмысына ұқсас жекелеген микросұлбаның жұмысын жасайтын жаңа интерфейсті ұсынады. RAMBUS RAS/CAS сигналдарын тастап, оларды мекенжайдың жіберілуі мен деректердің келуі арасындағы басқа өтініштердің орындалуына жол беретін шинамен ауыстырады. Шинаның бұл түрі пакеттік қайта қосылатын шиналар (packet-switched bus) немесе ыдыраған транзакциялары бар шиналар (фрШ-гансасйоп bus) деп аталады. Мұндай шина кристалға жадының жекелеген банкі ретінде жұмыс істеуге мүмкіндік береді. Кристалл деректердің өтпелі санын бір сұранысқа қайтара алады және тіпті регенерацияны өз бетімен орындайды. RAMBUS байтты интерфейс пен синхрондау сигналын ұсынады, сондықтан да микросұлба процессордың тактілік жиілігімен тығыз синхрондалуы мүмкін. Мекенжайлық конвейер толтырылғаннан кейін, жеке кристалдар әрбір 2 нс байт бойынша шығарылуы мүмкін.

Негізгі жадының көптеген жүйелері процессорлар мен жад микросұлбаларының өнімділігіндегі айырмашылықтарды азайту үшін бет режиміндегі ДЕІЕҚ парақтық режиміне ұқсас әдістерді пайдаланады.

Виртуалды жад - деректерді қорғауды ұйымдастыру құралы ретінде. Қазіргі уақытта қабылданған виртуалды жад тұжырымдамасы көптен бері пайда болды. Бұл есептерді ұйымдастырудағы бірқатар өзекті мәселелерді шешуге, соның ішінде мультибағдарламалы жүйелердің сенімді жұмысын қамтамасыз етуге мүмкіндік берді.

Кез-келген уақытта компьютер әр түрлі процестерді немесе тапсырмаларды орындайды, олардың әрқайсысында өз мекенжай кеңістігі бар. Барлық физикалық жадты қандай да болмасын бір тапсырмаға беру тым қымбатқа түседі, әсіресе көптеген тапсырмалар өздерінің мекенжай кеңістігінің тек шағын бөлігін пайдаланғанады. Сондықтан кішігірім физикалық жадты түрлі тапсырмалар арасында бөлу үшін механизм қажет. Виртуалды жады сындай мүмкіндіктерді іске асырудың бір әдісі болып табылады. Ол физикалық жадыларды блоктарға бөледі және оларды түрлі тапсырмалар арасында бөледі. Бұл ретте, ол сонымен қатар оған тиесілі сол блоктармен тапсырмаларды шектейтін кейбір қорғау сұлбасын қамтамасыз етеді. Сонымен қатар, виртуалды жады түрлерінің көпшілігі бағдарламаның бастапқы іске қосу уақытын қысқартады, себебі барлық бағдарламалық код пен деректер орындауды бастау үшін физикалық жадта талап етіледі.

Виртуалды жад тұжырымдамасын іске асырумен тығыз байланысты тағы басқа мәселе, өте үлкен көлемді тапсырмаларды есептеуді ұйымдастыруға қатысты.

Егер бағдарлама физикалық жад үшін тым үлкен болса, онда оның бір бөлігін сыртқы жадта (дискіде) сақтау керек және оны компьютердегі шешімге бейімдеу тапсырмасы бағдарламашыға жүктеледі. Бағдарламашылар бағдарламаларды бөліктерге бөледі және одан кейін негізгі жадқа жүктелетін және оған қолданушының бағдарламасының басқаруымен түсірілетін оверлейлік құрылымды ұйымдастыра отырып, тәуелсіз орындалуы мүмкіндерін анықтады. Бағдарламашы бағдарлама оған бөлінген физикалық жад кеңістігінен тыс жүрмеуін бақылауы тиіс. Бұл жүктемеден виртуалды жады бағдарламашыларды босатты. Ол жады иерархиясының екі деңгейін автоматты түрде басқарады: бастапқы және сыртқы (диск) жады.

Сонымен қатар, бір бағдарламаны физикалық жадтың ерікті жадысында орындауға мүмкіндік беретін, бағдарламаларды автоматты түрде жылжыту механизмін қамтамасыз ететін виртуалды жад бағдарламалардың жүктелуін жеңілдетеді.

Виртуалды жады жүйелерін екі түрге бөлуге болады: парақтар деп аталатын, блоктардың бекітілген өлшемдері бар жүйелері және сегменттер деп аталатын блоктың айнымалы өлшемдері бар жүйелер. Виртуалды жад ұйымдастырудың екі түрін де қарастырайық.

Жадыны парақтық ұйымдастыру. Парақтық ұйымдастыру жүйесіндегі негізгі және сыртқы жад (негізінен дискілік кеңістік) блоктарға немесе бекітілген ұзындықтағы парақтарға бөлінеді. Әрбір пайдаланушыға компьютердің негізгі жадынан жоғары және командалық жүйеге тән мекенжайлар мүмкіндіктері ғана шектелетін мекенжай кеңістігінің кейбір бөліктері беріледі. Мекенжай кеңістігінің бұл бөлігі пайдаланушыныңвиртуалды жады деп аталады. Пайдаланушының виртуалды жадындағы әрбір сөз екі бөліктен тұратын виртуалды мекенжаймен анықталады: мекенжайдың жоғары санаттары парақтың нөмірі ретінде қаралады, ал кіші санаттары сөздің (немесе байттың) парақ ішіндегі нөмірі ретінде қарастырылады.

Жадының әртүрлі деңгейлерін басқару парақтың таратылуын бақылайтын және осы деңгейлер арасындағы алмастыруды оңтайландыратын операциялық жүйе ядросының бағдарламаларымен жүзеге асырылады. Жадты парақтық ұйымдастырған кезде аралас виртуалды парақтар негізгі физикалық жадыныңауысым беттерінде орналасуы міндетті емес. Виртуалды беттер мен негізгі жад беттері арасындағы сәйкестікті көрсету үшін операциялық жүйе әрбір бағдарлама үшін беттер кестесін жасауы және оны машинаның негізгі жадына орналастыруы тиіс. Бұл жағдайда бағдарламаның әрбір беті негізгі жадта болғандығына не болмағандығына қарамастан, бет кестесінің кейбір элементімен сәйкес қойылады. Бет кестесінің әрбір элементі негізгі жадтың физикалық бетінің және арнайы индикатордың нөмірінен тұрады.

Осы индикатордың бірыңғай күйі негізгі жадыда осы парақтың болуын көрсетеді. Индикатордың нөлдік күйі шұғыл жадыдағы парақтың жоқтығын білдіреді.

Осындай сұлбаның тиімділігін арттыру үшін, процессорларда толықтай ассоциативті кэш-жады қолдаланылады, ол сонымен қатар (TLB — translation-lookaside buffer) мекенжайларды түрлендіру буфері деп аталады. TLB-ның болуы парақты ұйымдастырудың сұлбасын жасау принципін өзгертпейді, жадыны қорғау көзқарасы тұрғысынан оны бір бағдарламадан екіншісіне ауыстыру кезінде тазалау мүмкіндігін қамтамасыз ету қажет.

Негізгі жадта орналасқан парақтардың кестелерін іздестіру және TLB-ні жүктеу бағдарламалық әдіспен немесе арнайы аппараттық құралдармен жүзеге асырылуы мүмкін. Соңғы жағдайда пайдаланушы бағдарламасының байланысты болып келетін парақ кестелерімен байланысу мүмкіндігінің алдын алу үшін, арнайы шаралар қарастырылған. Осы мақсатта процессорда парақ кестесінің сипаттағышынан (дескрипторынан) немесе базалық шекаралас жұпты қамтитын қосымша қорғаушы тіркелімі көзделеді. База негізгі жадтағы парақтар кестенің болуын анықтайды, ал шекара тиісті бағдарламаның парақтар кестесінің ұзындығын анықтайды. Осы қорғаныс тіркелімін жүктеу артықшылықты режимде ғана рұқсат етіледі. Әрбір бағдарлама үшін операциялық жүйе парақ кестесінің дескрипторын сақтайды және тиісті бағдарламаны іске қоспас бұрын оны процессорды қорғау тіркеліміне орнатады.

Жадты беттерді ұйымдастырудың қарапайым сұлбаларына тән кейбір ерекшеліктерді атап өтейік. Олардың ішіндегі ең маңыздысы, операциялық жүйенің араласуынсыз бір-бірімен тікелей байланысуы керек барлық бағдарламалар, виртуалды мекенжайлардың ортақ кеңістігін пайдаланулары керек. Ол динамикалық жад бөлу режимінде жұмыс істеуі керек операциялық жүйенің өзіне де қолданылады. Сондықтан да, кейбір жүйелерде пайдаланушының виртуалды мекенжай кеңістігі, пайдаланушы бағдарламаларына кіруді қалайтын жалпы процедуралардың өлшемімен қысқартылады. Жалпы рәсімдерге барлық пайдаланушылар үшін виртуалды мекенжай кеңістігін белгілі бір көлемдерде, олар барлық пайдаланушылардың беттеріндегі кестелерде тұрақты орынға ие болатындай бөлу керек. Бұл жағдайда, орындалатын бағдарламалардың тұтастығын, құпиялылығын және өзара оқшаулануын қамтамасыз ету үшін бет кестесінің элементтерінде арнайы қолжетімділік индикаторларын қолдана отырып, беттерге қол жеткізудің әртүрлі режимдері қамтамасыз етілуі керек.

Осындай пайдаланудың нәтижесі әрбір пайдаланушы беттері кестелерінің біршама өсуі болып табылады. Кестелердің ұзындығын қысқарту мәселесі шешімдерінің бірі кестелерді көп деңгейлі ұйымдастыруды жүргізуге негізделген.

Кестелердің көп деңгейлі ұйымдастырылуының жеке жағдайы жадты парақтық ұйымдастыру арқылы сегменттеу болып табылады. Пайдаланушының мекенжай кеңістігін ұлғайту қажеттілігі мекенжай кеңістігінде бағдарламалар мен деректердің бөліктерін жылжытудың қажеттілігін болдырмау, олар әдетте атауын өзгерту проблемаларына алып келеді және көптеген тапсырмалар арасында ортақ ақпаратты бөлудегі елеулі қиындықтарға алып келеді.

Жадыны сегменттеу. Жадты ұйымдастырудың басқа тәсілдері, бағдарламалардың әдетте жекелеген сала-сегменттеріне бөлінуі дәлеліне сүйенеді. Әрбір сегмент деректердің немесе бағдарламалардың жиынтығынан тұратын және пайдаланушының мекенжай кеңістігінде орналасқан ақпараттың жекелеген логикалық бірлігі болып табылады. Сегменттер оларды символикалық атаумен байланысатын пайдаланушылар арқылы жасалады. Әрбір сегментте нөлден басталатын сөздердің өз нөмірленуі бар.

Әдетте мұндай жүйелерде пайдаланушылар арасында ақпарат алмасу сегменттер негізінде құрылады. Сондықтан да, сегменттер қорғалуы керек ақпараттың жекелеген логикалық бірліктері болып табылады және тап осы деңгейде сегменттер үшін әр түрлі қолжетімділік режимдері енгізіледі. Сегменттердің екі негізгі түрін бөлуге болады: бағдарламалық сегменттер және деректер сегменттері (стек сегменттері деректер сегменттерінің жекелеген жағдайы болып табылады). Жалпы бағдарламалар қайта кіру сипатына ие болғандықтан, онда бағдарлама сегменттерінен тек команданы таңдау және тұрақты оқуға ғана рұқсат етіледі. Бағдарлама сегменттеріне жазба заңсыз деп қаралуы және жүйемен тыйым салынуы мүмкін. Деректер сегменттерінен алынған командаларды таңдауды заңсыз деп санауға болады және деректердің кез-келген сегменті жазбалар немесе оқу бойынша қорғалуы мүмкін.

Сегменттеуді іске асыру үшін іске асырудың тетіктерімен ерекшеленетін, бірақ аталған қағидаларға негізделген бірнеше сұлбалары ұсынылған.

Жад сегменті бар жүйелерде пайдаланушының мекенжай кеңістігіндегі әрбір сөз екі бөліктен тұратын виртуалды мекенжаймен анықталады: мекенжайдың жоғарғы санаттары сегмент нөмірі ретінде қарастырылады, ал төмендегілері сегменттің ішіндегі нөмір ретінде қаралады. Сегменттеумен қатар, жадты парақтық ұйымдастыру да пайдаланылуы мүмкін. Бұл жағдайда сөздің виртуалды мекенжайы үш бөлімнен тұрады: мекенжайдың жоғары санаттары сегменттің санын анықтайды, ортаңғылары – сегмент ішіндегі парақтық нөмірі, ал кішілері - бұл бет ішіндегі сөз нөмірі.

Парақтық ұйымдастыру жағдайындағыдай, виртуалды мекенжайды негізгі жадтың физикалық мекенжайына түрлендіруді қамтамасыз ету қажет.

Осы мақсатта әрбір пайдаланушы үшін операциялық жүйе сегменттер кестесін құруы тиіс. Сегменттік кестенің әрбір элементі сегменттің (қор өрісі, шекаралар және кіру режимдерінің индикаторлары) сипаттамасынан (дескриптор) тұрады. Парақтық ұйымдастыру болмаған кезде, қор өрісі негізгі жадтағы сегменттің басындағы мекенжайды, ал шекара - сегменттің ұзындығын анықтайды. Парақтық ұйымдастыру болған кезде, қор өрісі аталған сегменттің бет кестесінің басталуын анықтайды, ал шекара сегменттегі беттер санын анықтайды. Қолжетімділік режимінің индикатор өрісі – оқу, жазу және орындау әрекеттерінің тіркелімін білдіреді.

Операциялық жүйе әр түрлі пайдаланушылардың сегменттерінің кестелерін негізгі жадында сақтайды. Орындалатын бағдарламадағы кестенің орналасуын анықтау үшін арнайы қорғау тіркелімі қолданылады, ол операциялық жүйемен оны орындамастан бұрын жүктеледі. Осы тіркелім сегмент кестесінің дескрипторынан (қор және шекарадан) тұрады, мұндағы қор орындалудағы бағдарламаның сегмент кестесінің бастапқы мекенжайынан, ал шекара – сегмент кестесінің осы ұзындығынан тұрады. Виртуалды мекенжай сегменті нөмірінің санаты сегмент кестелерінен іздестіру үшін индекс ретінде пайдаланылады. Осылайша, сегменттер кестесінің дескрипторында және сегменттік кесте элементтерінде базалық шекаралас жұптардың болуы пайдаланушының бағдарламасына сегменттердің және олармен байланысты емес беттердің кестелеріне кіруіне жол бермейді. Қолжетімділік режимі индикаторларының сегменттерінің кестесі элементтерінің болуы аталған бағдарлама тарапынан сегментке қол жеткізудің қажетті режимін жүзеге асыруға мүмкіндік береді. ұлбаның тиімділігін арттыру үшін қауымдастырылған кэш-жады пайдаланылады.

Сипатталған сегменттеу сұлбасында қолжетімді индикаторлары бар сегменттік кесте белгілі бір тапсырмалардың бөліктері болып табылатын барлық бағдарламаларды бірдей қолжетімділік мүмкіндігімен қамтамасыз етеді, яғни ол бірыңғай қорғаудың жалғыз саласын (доменді) анықтайды. Дегенмен, қорғалған кіші жүйелерді бір тапсырма аясында құру үшін, орындау нүктесі шешімін басқаратын әртүрлі бағдарламалар арқылы өтетін кезде, көптеген қорғау домендерімен әрбір тапсырманы байланыстыру қажет. Қорғалатын кіші жүйелерді іске асыру кейбір арнайы аппараттық құралдарды талап етеді.

Бақылау сұрақтары

1. Ақпараттық жүйе сипаттамасының қандай деңгейлері (модель түрлері) деректердің ақпараттық моделін ұсынады?
2. Деректердің ақпараттық моделінің әр үлгісіне сипаттама беріңіз.
3. Дерекқорды ұйымдастырудың файлдық құрылымдарының қысқаша сипаттамасын беріңіз.

4. Дерекқорды ұйымдастырудың файлдық құрылымдарында сақталатын әрбір файлды сипаттайтын ақпараттың құрамын көрсетіңіз.

5. Дерекқор кестесінің физикалық моделі қандай?

6. Дерекқордың кесте өрістерінің міндетті сипаттамалары қандай?

7. Дерекқорда жіктелетін файлдар мен файл құрылымдары қандай?

8. Пайдаланушының позициясындағы файл дегеніміз не?

9. Файлдардағы ақпаратқа қол жеткізуді басқару әдістеріне тәуелді сыртқы жады құрылғыларының түрлерін атаңыз.

10. Файлдық жүйеде сақталатын файлдың сипаттамаларды атаңыз.

11. Хеш-функция дегеніміз не және хэштеу әдістерінің мәні қандай?

12. Индекстік-тікелей файлдар мен индекстік-кезеңді файлдар арасындағы айырмашылық қандай?

13. Қандай іздестіру құрылымы ағаш деп атау қабылданған?

14. Жад иерархиясының кейбір деңгейлерін сипаттау үшін қандай төрт сұраққа жауап беру қажет?

15. Негізгі жадтың өнімділігін бағалау үшін қандай екі параметр пайдаланылады?

16. Виртуалды жады дегеніміз не және виртуалды жад ұйымдастыру арқылы көптеген деректерді өңдеуді (есептеулерді) ұйымдастырудың өзекті міндеттері қандай болады?

4-тарау

ДЕРЕККОРДЫҢ БАСҚАРУ ЖҮЙЕСІН ӘЗІРЛЕУ ЖӘНЕ ҰЙЫМДАСТЫРУ

4.1. Дерекқор — заманауи CALS-технологиясының негізі

CALS-технологиялар — бұл кіріктірілген дерекқорға негізделген бірыңғай ақпараттық кеңістікті құруға бағытталған өндіріс және бизнес-процестерді ақпараттық қамтамасыз етуді дамытудың заманауи бағыты болып табылады.

CALS аббревиатурасы екі жолмен түсіндіріледі:

• Continuous Acquisition and Life Support — жеткізу және тіршілік циклін (бұйымның) үздіксіз ақпараттық қолдау;

• CommerceAtLightSpeed — жоғары жылдамдықты (шапшаң) коммерция.

Екі жағдайдада өндіріс пен бизнес-процестерге қатысушылардың арасында үлкен көлемде ақпаратты жасау, түрлендіру және беру туралы айтылады.

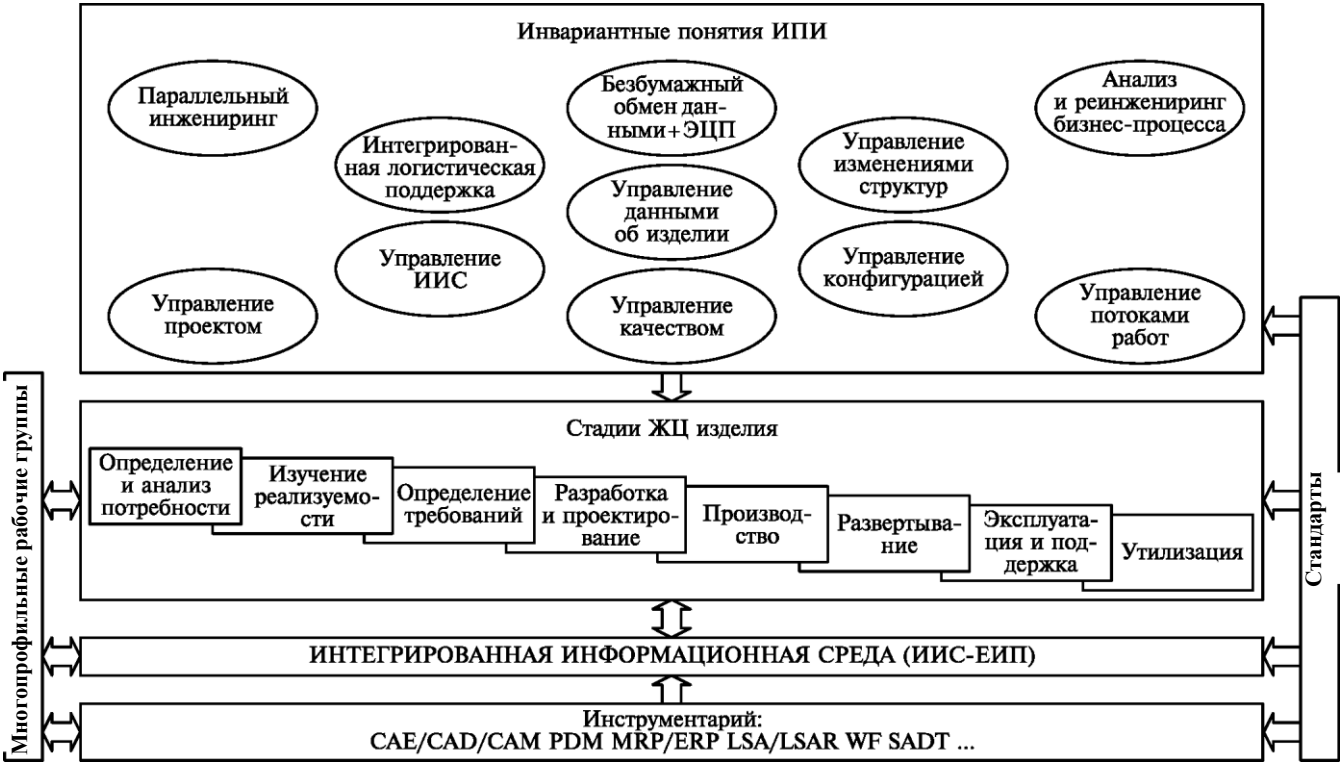
CALS тұжырымдамасы мен идеологиясы АҚШ әскери-өнеркәсіп кешенінің қойнауларында пайда болды және содан кейін барлық НАТО елдерімен қабылданған болатын.

Ресейде CALS барабар аналогы – Бұйымның тіршілік циклін ақпараттық қолдау (ИПИ) үшін ақпараттық қолдау қабылданды. 4.1-суретте Ресейде қабылданған ИПИ сұлбасы ұсынылған.

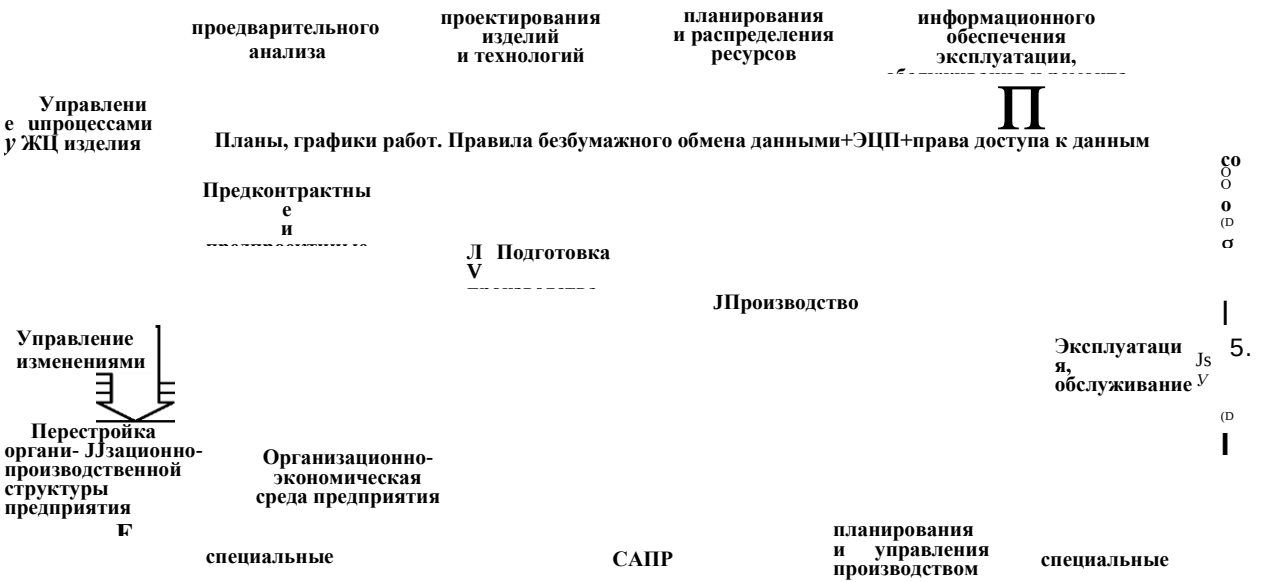
Аталған сұлбаға сәйкес ИТИ негізінк іріктірілген ақпараттық жүйе (КАЖ) немесе бірыңғай ақпараттық кеңістік (БАК) құрайды. Екі терминдер бірдей болып табылады, бірақ Ресей Мемстандарты бекіткен терминологиялық сөздікте бірінші термин қабылданған.

Аталған стандарт КАЖбұйымдар, өндіріс ортасы, кәсіпорын ресурстары мен процестері туралы ақпаратты қамтитын таратылған дерекқорлар жиынтығы ретінде анықталады, ол бұйымға қажет және рұқсат етілетін, бұйымның тіршілік цикліне қатысатын өндірістік-шаруашылық қызмет субъектілерінің деректерінің дұрыстығын, маңыздылығын, қауіпсіздігін және қол жетімділігін қамтамасыз етеді.

КАЖ кәсіпорын құру кезінде ИПИ негізгі қағидаттары іске асырылуы тиіс: *өндірістік процестің кез-келген сатысында туындаған ақпарат осы немесе басқа сатыларға қатысушылардың барлығында сақталады және қол жетімді болады, осы ақпаратты пайдалануда қолданыстағы құқықтарға сәйкес келеді.*



4.1-сур. ИПИ сұлбасы



БАҒДАРЛАМАЛЫҚ ҚҰРАЛДАР
КАЖ

4.2-сур. Өндірістік кәсіпорындар жалпы дерекқорын құру компоненттері

Ақпаратты құру, түрлендіру және тарату процестері заманауи бағдарламалық қамтамасыз ету көмегімен жүзеге асырылады. Мұндай құралдар (4.1-суретті қараңыз) мыналарды қамтиды:

• автоматтандырылған құрылымдық және технологиялық жобалау жүйелері (CAE/CAD/CAM);

• ДҚБЖ-мен қоса, бұйым туралы деректерді басқаруға арналған бағдарламалық құралдар;

• өндірісті жоспарлау және басқарудың автоматтандырылған жүйелері (MRP/ERP);

• дерекқорларды (LSA/LSAR) талдау, сүйемелдеу және қолдау жүйелері;

• жұмыс ағындарын басқарудың бағдарламалық құралдары (WF);

• бизнес-үрдістерді модельдеу және талдаудың бағдарламалық құралдары (SADT).

Одан туындайтыны, дерекқор БАК құру үшін негіз болып табылады.

Осы тұжырымдамаға сүйене отырып, дерекқордың дербес объекті ретінде жобалауды маңызды түрде өзгерту және көпмақсатты, ортақ дерекқорларды жасау стратегиясына көшу керек. 4.2-суретте өнеркәсіптік кәсіпорындардың жалпы дерекқорын қалыптастырудың кейбір компоненттері берілген.

4.2. CALS-технологиялар жағдайында көпмақсатты ақпараттық жүйелерді әзірлеу принциптері

CALS-технологиясының тұжырымдамасынан көріп отырғандай, кәсіпорындарда дамыған ақпараттық жүйелер мен деректер қорлары көпмақсатты болуы керек.

Көпмақсатты дерекқорды әзірлеу қағидалары екі міндетті талаптарды сақтауға алып келеді: жүйелік тәсілдеме және стандарттау.

Жүйелік тәсілдеме. Ақпараттық жүйені әзірлеудің жүйелік тәсілдемесін, яғни мұндай жүйенің бір-бірімен өзара байланысты және өзара әрекет ететін көптеген элементтерден тұратын ірі жүйе ретінде қарастырылуын білдіреді. Ақпараттық жүйелерді жобалау кезінде келесі қағидаларды сақтау қажет:

• жүйелердің барлық әлеуметті пайдаланушыларының мүдделерін есепке алу;

• модульді жобалау және енгізу принципі.

Жүйенің барлық әлеуетті пайдаланушыларының мүдделерін есепке алу. Бұл қағида дерекқорды әзірлеудің келесі кезектілігін білдіреді.

1. Кәсіпорынның қай мамандары мен қандай бөлімшелерінде белгілі бір ақпараттық объекті туралы ақпарат қажет екендігін анықтау.

2. Әртүрлі пайдаланушылар объектілерді сипаттау белгілерін орнатсын.

3. Бір сынып объектілері белгілерінің жалпы құрамын белгілеу.

Жобалаудың мұндай тәсілі дерекқорды әзірлеу мерзімін арттырады, бірақ тұтастай алғанда бүкіл жүйені әзірлеу шығындарын біршама төмендетеді.

Осы принципті түсіндіру үшін, кәсіпорынның бірінде дерекқорды әзірлеудің нақты мысалын келтіруге болады. Дерекқор құру бағдарламаларының пайда болуы қызметкерлермен дәрежесі бойынша бағаланды және олар өздері үшін қажетті дерекқорды әзірлеуге «ұмтылды».

Цех технологтарының алдында тұрған міндеттердің бірі - бөлшектерді механикалық өңдеуге арналған құралды таңдау болып табылады. Олар өздерінің кескіш құрал бойынша цехтық ДҚ әзірледі (оған өздерінің уақыты мен қаражатын жұмсады).

Бұл уақытта зауыттың конструкторлық бөлімінде кескіш құралдарды жобалаумен айналысатын мамандар да өздерінің дерекқорын құрды. Алайда, басшылық кескіш құрал бойынша жалпы зауыттық ақпараттық жүйені құру шешім қабылдаған кезде, онда әртүрлі мамандар кескіш құралдың бірдей ерекшеліктерін түрлі әдістермен сипаттаған. Нәтижесінде әзірленген дерекқорды толығымен қайта жасау қажет болды, бұл қосымша уақытты да, қосымша шығындарды талап етті. Мамандардың арасындағы келісілмеген дерекқорды әзірлеу қаражаты кәсіпорын үшін жоғалды.

Әзірлеу және енгізудің модульдік принципі. Модульдік принцип барлық жүйе түпкілікті әзірленгенге дейін, өндірісте жеке енгізілуі мүмкін, жекелеген өзара байланысты модульдер (кіші жүйелер) түрінде әзірленуі тиіс кез-келген жүйені білдіреді.

Стандарттау. Олардың көпмақсатты сипатын ескере отырып, ақпараттық жүйе әзірлемелерін стандарттау келесі аспектілерге ие:

• ақпарат;

• бағдарламалық қамтамасыз ету;

• жабдық.

Ақпараттық қамтамасыз етуді стандарттау дерекқор нысандарының объектілері ретінде, компьютерлік өңдеу принциптері символды стандарттаумен шартталған, өйткені дерекқордың объектілері компьютермен бірегей мойындалуы тиіс.

ДҚ әзірлеудің осы аспектісі барлық ақпараттық объектілерді сәйкестендірулерге оларды сәйкестендірудің нақты ережелері белгіленуі тиіс (оларды жазудың грамматикалық ережелері) екенін білдіреді. Осылайша, «Жону кескіштің» бөлшегін механикалық өңдеу үшін құралдың атауын белгілей отырып, оның мақсатының басқа әдісіне жол берілмейді (мысалы, «Жону кескіші» атауы «Кескіш жонушы» деген атқа сәйкес емес).

Бағдарламалық қамтамасыз етуді стандарттау қажеттілігі айқын - бірнеше пайдаланушыларды, бір-бірінен қашықтағы жүйелерді әзірлеу кезінде бір жүйенің деректерін басқа жүйенің бағдарламалық етуімен өңделуі тиіс.

Аппараттық қамтамасыз етуді стандарттау компьютерлік техниканы пайдалану шығынын төмендету қажеттілігіне байланысты.

Ресей кәсіпорындарында CALS-технологиясының тұжырымдамасын енгізу, халықаралықты қоса, бірыңғай стандарттарды кеңінен қолдануды қарастырады.

4.3. Жергілікті есептеу желілерде дерекқорды басқарудың көпмақсатты жүйелерін ұйымдастыру

Заманауи кәсіпорындардың компьютерлік ақпараттық жүйелері желілік технологиялар - компьютерлерді жергілікті есептеу желілеріне (ЖЕЖ) біріктіру арқылы әзірленеді. Кәсіпорындардың жергілікті есесептеу желілерінде дерекқорды әзірлеу кезінде оларды ұйымдастырудың екі түрі (екі архитектурасы) пайдаланылады:

• файл-сервер архитектурасы;

• клиент-сервер архитектурасы.

Дерекқорды ұйымдастырудың осы түрлері үшін ортақ белгілері - деректер (файлдар) мен жұмыс станцияларының (қолданушылардың компьютерлерінің) қорларының – клиенттер орналасқан сервердің болуы болып табылады.

Дерекқорды ұйымдастырудың осы екі архитектурасы ақпаратты өңдеу әдістерімен ерекшеленеді.

Файл-сервер архитектурасында ақпаратты өңдеудің барлық процестері клиенттің компьютерінде жүргізіледі. Ол үшін клиент тиісті сұраныс бойынша бүкіл деректер файлын жібереді.

Клиент-сервер архитектурасында ақпаратты өңдеудің барлық процестері клиенттің сұранысы бойынша серверде орындалады, оған тек деректерді өңдеу нәтижелері жіберіледі.

Көпмақсатты желілік дерекқорларды ұйымдастырған кезде клиент-сервер үлгісі бойынша жүйені ұйымдастыруға басымдылық беріледі, ол файл-сервер архитектурасының кемшіліктері мен клиент-сервер архитектурасының артықшылықтарынан туындайды.

Файл-сервер архитектурасы бойынша ДҚ ұйымдастырудың кемшіліктері:

• желілер бойынша ДҚ файлдарынберу кезінде, әсіресе ірі көлемдегі ақпаратпен және бір уақытта бірнеше қолданушылардың файлдарға өтіну мүмкіндігін ескере отырып, жүйемен жұмыс өнімділігі күрт төмендейді;

• желілер бойынша ірі көлемді файлдарды бірнеше қолданушыларға бір уақытта беру кезінде берілетін ақпараттардың сенімділігін жоғалту ықтималдығы артады, ол жүйенің жұмысының сенімділігін төмендетеді.

Клиент-сервер архитектурасы бойынша дерекқорды ұйымдастырудың артықшылықтары:

• егер желі бойынша клиенттердің сұранысымен деректерді өңдеу нәтижелері берілетін болса, онда желіге жүктеме күрт төмендейді және соның салдарынан ДҚ қолдаланушылардың көп санын қосу мүмкіндігі артады. Жүйенің жұмысының өнімділігі файл-сервер архитектурасына қарағанда әлдеқайда жоғары;

• серверде деректерді орталықтандырылған сақтауы және өңдеуі жүйе жұмысының сенімділігін арттырады;

• ДҚБЖ серверлік бөлігін әзірлеуді SQL тілінде немесе басқа жоғары деңгейлі тілдерде орындауға болады, ол деректерді өңдеудің сенімділігі мен өнімділігін арттырады. ДҚБЖ клиенттік бөлігін әзірлеуді қолданбалы бағдарламалық өнімдерді қолданумен орындауға болады, мысалы, Visual Basic, Microsoft Access, ол ақпараттық жүйенің даму уақытын едәуір қысқартады.

4.4. Көпмақсатты дерекқорларды жобалау кезеңдері

Алдыңғы бөлімдерде өндірістің және бизнестің заманауи жағдайында дербес объектілер ретінде дерекқорды жобалау стратегиясынан көпмақсатты ақпараттық жүйелер - жалпы дерекқор жасау стратегиясына көшу қажет екендігін анықтадық. Мұндай көшу көпмақсатты дерекқорларды жобалаудың келесі кезеңдерін көздейді:

1. Көпмақсатты дерекқордың тұжырымдамалық моделін жасау;
2. Техникалық талаптарға сәйкес ДҚБЖ жобасын әзірлеу;
3. Жобаны іске асыру және техникалық құжаттаманы әзірлеу.

Көпмақсатты дерекқордың тұжырымдамалық моделін әзірлеу. Көпмақсатты дерекқорларды жобалаудың осы сатысында келесі қадамдарды орындау қажет:

• ҚАЖ құру мақсатын анықтау;

• ДҚ пайдаланушыларының құрамын белгілеу;

• тұжырымдамалық ДҚ моделін әзірлеу;

• ДҚБЖ жергілікті жобалауға техникалық тапсырма әзірлеу;

• ДҚ әзірлеу үшін қажетті еңбек және материалдық ресурстарды анықтау.

ҚАЖ құру мақсатын анықтау. Кез-келген компьютерлік жүйені құру мақсаты, оны іске асырудан белгілі бір экономикалық тиімділікке қол жеткізу болып табылады, сондықтан белгілі бір кәсіпорын жағдайында ҚАЖ құрудағы басым бағыттарды белгілеу қажет.

Дерекқор өндірісті басқарудың барлық дерлік міндеттері үшін әзірленуі мүмкін, мысалы:

• материалдар мен құрастырушы бұйымдарды жеткізу;

• жаңа бұйымдар конструкциясын жобалау;

• өнімді дайындаудың технологиялық процестерін жобалау;

• технологиялық жабдықтарды жобалау (бейімдеу, құралдар);

• өнімді шығарудышұғыл күнтізбелік жоспарлау және басқару;

• нормативтік қорды әзірлеу (еңбек және материалдық ресурстардың, негізгі және көмекші материалдардың қажеттілігі және т.б.);

• шығарылатын өнім сапасын басқару;

• өнімді өткізу және т.б. басқару.

Дерекқорды әзірлеу бағытын таңдау туралы шешім қабылдау, әрине, кәсіпорын басшыларының құзыреті болып табылады.

ДҚ пайдаланушылары құрамын белгілеу. Өндірістік қызмет саласын таңдай отырып, әзірленетін дерекқор пайдаланушыларының құрамы туралы ақпаратпен қамтамасыз ету қажет. Ол келесі тапсырмаларды шешу үшін қажет:

• ақпараттық объектілердің сыныптарын анықтау, олардың сипаттамалары және, қорытындысында, дерекқор кестесінің құрамын анықтау;

• әлеуетті пайдаланушылардың орналасу орнын анықтау, және қорытындысында, ЖЕЖ архитектурасын анықтау.

ДҚ тұжырымдамалық моделін әзірлеу. Тұжырымдамалық модельді әзірлеудің түпкі мақсаты дерекқор кестелерінің оңтайлы құрамын белгілеу болып табылады (3.2 бөлімін қараңыз). Көпмақсатты пайдаланушылардың дерекқорларын құрудың аталған кезеңінде КАЖ әрбір пайдаланушысының қажеттілігіне негізделеді, ал содан кейін әрбір кесте қалыпқа келтіру процедурасына ұшыратылуы мүмкін.

Жергілікті ДҚБЖ жобалау үшін техникалық тапсырма әзірлеу. Дерекқор кестелерінің құрамын және КАЖ пайдаланушылардың құрамын анықтағаннан кейін, ДҚБЖ жобалаудың техникалық сипаттамасын әзірлеуге кірісуге болады. Техникалық тапсырмада:

• ЖЕЖ архитектурасы мен дерекқор архитектурасын таңдауды негіздеу;

• ДҚБЖ әзірлеу үшін бағдарламалық жүйені таңдауды негіздеу;

• ДҚ әр пайдаланушысы үшін қажетті ақпаратты жеткізетін шығыс құжаттарының нысандарына қойылатын талаптарды әзірлеу;

• әрбір пайдаланушының тапсырмаларын ескере отырып, пайдаланушы интерфейсін құру бойынша талаптарды әзірлеу;

• дерекқорға және оның құрамдас бөліктеріне ақпаратпен, сол сияқты ақпаратты алу процесінде кестелерді толтыру барысында пайдаланушылардың қолжетімділік құқықтарын айқындауды қоса алғанда, ДҚБЖ-ны ұйымдастырушылық қамтамасыз ету талаптарын әзірлеу.

ДҚ әзірлеу үшін еңбек және материалдық ресурстарға қажеттіліктерді анықтау. Жоғарыда тізімделген барлық кезеңдерді орындағаннан кейін техникалық тапсырма міндеттерін орындау үшін еңбек және материалдық ресурстардың қажеттігін бағалау.

Ол үшін, жобаларды басқару бағдарламалық жүйесін қолдану мақсатқа сай, мысалы Microsoft Project.

Осы жүйені пайдалану, ол ресурстарға қажеттілікті анықтау үшін ғана мақсатқа сай емес; ол ДҚБЖ жобалау жөніндегі жұмыстарды орындаудың барлық жүрісін тиімді басқаруға мүмкіндігін береді. Microsoft Project 2002 кеңейтілген өнімдері жобаны басқарудың интуитивті түсінікті құралын, ақпаратқа қолжетімділікті және ұжымдық жұмысты қолдауды көрсетеді, сондай-ақ жобаларды корпоративтік басқарудың қуатты платформасы болып табылады.

Техникалық тапсырманы әзірлей отырып, орындаушылардың құрамын анықтағаннан кейін, жобаны іске асыруға кірісу - кәсіпорынның өндірістік қызметінің таңдаған бағыты бойынша дерекқорды басқару жүйесін құру.

Техникалық тапсырмаларға сәйкес ДҚБЖ жобасын әзірлеу.

Көпмақсатты дерекқорларды жобалаудың аталған кезеңінде келесі тапсырмаларды орындау қажет:

1. Дерекқор кестелеріне түрлендіру үшін нақты пәндік саланың объектілері туралы бастапқы ақпаратты жинау, талдау және дайындау;
2. Дерекқор кестелерінің оңтайлы құрамы мен құрылымын әзірлеу;
3. Кестелер арасындағы логикалық байланыстарды белгілеу;
4. Тапсырманы орындау үшін сұраныстардың қажетті санын әзірлеу;
5. Техникалық тапсырмада белгіленген шығыс құжаттарының талаптарына жауап беретін есептердің қажетті санын әзірлеу;
6. Қолданушылық интерфейстің нысандарын әзірлеу;
7. Пайдаланушының жүйемен жұмысын автоматтандыратын басқару модульдерін әзірлеу.

Жобаны іске асыру және техникалық құжаттаманы әзірлеу. Әзірленген ДҚБЖ жобасын іске асыру келесі міндеттерге алып келеді:

• объектілер туралы мәліметтермен дерекқор кестелерін толтыру;
• алға қойған тапсырмаларды орындау кезінде ДҚБЖ жұмысын тексеру;
• пайдаланушыларға арналған нұсқаулықтарды әзірлеу;
• жүйені тапсырыс берушіге тапсыру.

4.5. Реляциялық дерекқорды әзірлеу жүйесінің негізгі компоненттері

ДҚБЖ жобасында кемінде келесі негізгі компоненттер болуы тиіс:
- кестелер;
- сұраныстар;
- нысандар;
- есептер;
- басқару бағдарламалары.

Кестелер. Дерекқор кестелері басқаша мақсатта болуы мүмкін (мысалы, тұрақты ақпарат кестелері, ауыспалы кестелер ақпараты). *Тұрақты ақпарат кестелері* (шартты түрде тұрақты) деректер ұзақ уақыт барысында өзгермейтін деректерден тұруы мүмкін (мысалы, ұйым қызметкерлерінің тізімдері, өнімді дайындау кезінде қолданылатын технологиялық операциялардың атаулары және т.б.).

Ауыспалы ақпараттар кестелері – қолданушымен үнемі толықтырылып немесе өзгертілетін объектілер туралы ақпарат кестелері.

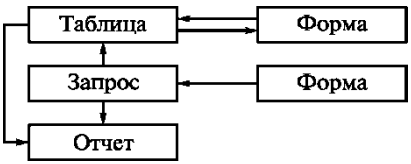
Сұраныстар. Дерекқор сұраныстары қолданушы берген талаптар бойынша (жол мағыналарымен) ақпарат іздестіру және өңдеуге арналған командалар жинағынан тұрады. Заманауи ДҚБЖ сұраныстары:
- үлгі үшін;
- жаңарту;
- қосымша;
- жою;
- кестелер жасау сұраныстарын құру мүмкіндігін береді.

Таңдау сұранысы дерекқордың нақты кестесінде (кестелерінде) ақпаратты іздестіруге (таңдауға) арналған.

Жаңарту сұраныстары кестенің жекелеген ұяшықтарында деректерді автоматты жаңартуға арналған.

Қосуға немесе жоюсұраныстары ДҚ кестесіндегі жазбаларды кестеге автоматты қосу немесе жоюға арналған.

Кестелерді құру сұраныстары қазіргі ДҚ негізінде жаңа кестелер құруға арналған. Бұл жағдайда,жаңа кестенің құрылымы автоматты қалыптасады.



4.3-сур. ДҚБЖ элементтердің байланысы сұлбасы

Нысандар. Ақпараттық жүйелерді әзірлеу кезіндегі нысандар қолданушысы мен компьютер арасындағы «достық» интерфейсті ұйымдастыруға арналған. Мақсаты бойынша нысандарды келесі топтарға бөлуге болады:

• кестелерге деректерді енгізудің нысандары;

• өтініштерді орындау үшін жағдайлар жасау нысандары;

• жүйенің жұмысын автоматты түрде басқаруға арналған нысандар (түймешекі нысандары, нысан-мәзір және т.б.)

Есептер. Есептер – ақпаратты өңдеу нәтижелерін шығару үшін құжаттардың түрлері. Әдетте, есептер кәсіпорында қабылданған есептілік нысандарына сәйкес болуы мүмкін. Ол бухгалтерлік есеп формалары немесе технологиялық құжаттардың нысандары және т.с. болуы мүмкін.

Есептер ДҚ кестелеріндегі немесе сұраныс нәтижесінде қалыптасқан ақпарат негізінде жасалады.

ДҚБЖ әзірлеу кезінде оның элементтері бір-бірімен 4.3-сур. ұсынылған сұлбаға сәйкес байланысуы мүмкін.

Басқару бағдарламалары. Басқару бағдарламалары дерекқор компоненттерімен жұмысты автоматтандыруға арналған. Олар макрокомандалар (макрос) немесе бағдарламалау тілінде, мысалы VBA жазылады.

Бақылау сұрақтары

1. Өндіріс пен бизнесті заманауи жетілдірудің негізгі бағыты - CALS-технологиясын анықтаңыз.
2. Дерекқорларды басқару жүйесін көпмақсатты әзірлеудің келесі қағидалары: жүйенің барлық әлеуетті пайдаланушыларының мүдделерін және жүйені әзірлеу мен енгізудің модульдік қағидаларын ескере ме?
3. Көпмақсатты дерекқорды жобалаудың негізгі кезеңдерін атаңыз.
4. Дерекқор жобасын әзірлеу кезінде қандай тапсырмаларды шешу қажет?
5. Реляциялық дерекқорды басқару жүйелерінің негізгі компоненттерін атаңыз?
6. Көпмақсатты дерекқорларды ұйымдастырудың негізгі формаларының негізгі сипаттамаларын, артықшылықтарымен кемшіліктерін атаңыз: файл-сервер архитектурасы және клиент-сервер архитектурасы.

ДЕРЕКҚОРДЫ БАСҚАРУ ЖҮЙЕЛЕРІН ӘЗІРЛЕУ ҮШІН БАҒДАРЛАМАЛЫҚ ӨНІМДЕРДІ ШОЛУ

5.1. Дерекқорды әзірлеу бағдарламалық құралдарының даму тарихы

Ақпараттық-іздестіру жүйелерін дамытудың бастапқы кезеңдерінде деректерді басқарудыңайла-шарғы жасауарнайы тілдері (АШЖС) – сұраныстар тілі әзірленді. Олар иерархиялық байланыстырылған файл түрінде ұсынылған деректермен жұмыс істеуге бағдарланған және ақпарат іздестіруге арналған тиісті алгоритмдерге ие.

Реляциялық дерекқорлардың пайда болуы ақпаратты іздестірудің басқа, барынша шапшаң алгоритмдері үшін алғышарттар жасады.

Кесте түрінде құрылымдалған ақпараттыөңдеу үшін - екі өлшемді алаптар, XX ғасыр. 70-жылдарының IBM фирмасымен кейіннен Structured Query Language (SQL) деген атауға ие болған тиісті тіл - құрылымдық сұраныстар тілі әзірледі Қазіргі уақытта SQL – аталғанреляциялық ДҚБЖ деректерін өңдеу тілінің халықаралық стандарты болып табылады. SQL тілі – барлық ДҚБЖ әзірлеу бағдарламалық өнімдерінің ядросы болып табылады.

ДББЖ қолдаланушылар мен әзірлеушілер арасында кеңінен таралған бағдарламалық өнімдер:

• арнайы бағдарламалау тілдері - Visual FoxPro, SQL, MS SQL-Server;

• қолданбалы бағдарламалық жүйелер - Microsoft Access, Oracle және т.б.

Осы бағдарламалық құралдардың кейбір сипаттамаларын қарастырайық.

Visual FoxPro. Бағдарламалау тілі FoxPro дерекқорды дамыту үшін ең танымал тілдердің бірін одан әрі дамытуды білдіреді. Visual FoxPro «ата тегінен» принципиалды ерекшелігі оның «визуалды» - ДҚБЖ дерлік барлық компоненттерінің объектілі-бағдарланған бағдарламалау мүмкіндігі. Visual FoxPro интерфейсі Windows операциялық жүйелерінің графикалық қабығына толығымен сәйкес келеді, ол олардың компьютерлік операциялық жүйелерінде деректері барлар үшін өте түсінікті болып табылатын ДҚБЖ құру жұмысын жасайды.

«Визуалды бағдарламалау» құралдарының болуына қарамастан, біз бағдарламашылар үшін осы бағдарламалық жүйені ұсынамыз. Visual FoxPro мүмкіндіктері бір кәсіпорын ішінде жергілікті немесе көпмақсатты дерекқорларды дамытуды қамтиды.

MS SQL-Server. Аталған бағдарламалық жүйе негізінен қолданушылық қосымшаларды әзірлеуге емес, бірақ клиент-сервер архитектурасы бойынша әзірленген көпмақсатты дерекқорларды басқаруға арналған. Осы жүйе дерекқорларды басқаруға мүмкіндік береді (деректерді көшіру, оларды қатарлас өңдеуді жүргізу, кәсіпорынның жергілікті компьютерлік желісінде, сондай-ақ Интернетте деректерді қабылдау және беру және т.б.), техникалық сипаттамалары бойынша әртүрлі аппараттық құралдарға ие клиенттік компьютерлермен өзара әрекеттеседі. SQL-Server елеулі көлемдегі ақпаратты өңдеуге арналған, бірақ, әдетте, жекелеген кәсіпорындар үшін жеткілікті болатын терабайттан жоғары емес.

Microsoft ACCESS. Ол дерекқорды әзірлеу үшін қолданылатын танымал бағдарламалық жүйелердің бірі.

Microsoft ACCESS — ол Microsoft фирмасы әзірлеген бағдарламалық орта. Ол жеткілікті ірі көлемдегі ақпаратпен (жүздеген мегабайт) реляциялық дерекқорларды басқару жүйелерін жасауға арналған. Microsoft ACCESS компаниясы пайдаланушыға деректерді жасау мен өңдеуді автоматтандыруға, сондай-ақ жұмыс барысында деректерді басқаруға қажетті барлық құралдарды ұсынады.

Бұл жүйенің басты артықшылығы - оның бағдарламашыға емес, түпкілікті пайдаланушыға бағытталуы болып табылады.

Microsoft Access бағдарламасының ең соңғы үлгілері көпмақсатты дерекқорларды құру үшін оны қолдануға мүмкіндік береді. Бұл жағдайда дерекқор кестелері серверге жіберілуі мүмкін, ал қолданушылық интерфейс клиенттің компьютерінде сақталады. Бұл жағдайда барлық ДҚБЖ компоненттерін әзірлеу қолайлығын Microsoft Access-ді пайдаланумен үйлестіруге, ал көпмақсатты дерекқорларын басқару міндеттерін MS SQL- Server жүктеумен үйлестіруге болады.

Microsoft Access-дың басқа артықшылығы – дерекқордың әзірлеуді үйрету құралы ретінде барлық басқа бағдарламалық өнімдеріне қарағанда оның теңдесіз артықшылығы болып табылады.

Oracle. Осы жүйе корпоративті реляциялық дерекқорды дамытуға арналған, ондағы ақпараттың көлемі терабайттан жоғары. Жүйенің негізін - SQL тілі құрайды. Oracle деректерді қорғаудың жоғары деңгейін қамтамасыз ету мүмкіндігімен ерекшеленеді.

5.2. SQL сұраныстарының құрылымдалған тілі

SQL сұраныстарының құрылымдалған тілі операторлар мен грамматикалық ережелерден тұратын қарапайым бағдарламалау тілі болып табылады. SQL тілінде дерекқор кестесіне қойылатын сұраныс SELECT нұсқаулығынан тұрады және келесі түрде сипатталуы мүмкін.

Деректерді анықтау операторлары

Оператор	Әрекеті
CREATE TABLE	ДҚ жаңа кестесін жасайды
DROP TABLE	ДҚ кестені жояды
ALTER TABLE	Қазіргі кестенің құрылымын немесе аталған кесте үшін көрсетілген тұтастықтың шектеулерін өзгертеді
CREATE VIEW	Кейбір SQL сұранысына сәйкес келетін виртуалды кесте жасайды
ALTER VIEW	Бұрын жасалған көріністі өзгертеді
DROP VIEW	Бұрын жасалған көріністі жояды
CREATE INDEX	Индекске кіретін атрибуттарға шапшаң қол жеткізуді қамтамасыз ету үшін кесте үшін индекс жасайды
DROP INDEX	Бұрын құрылған индексті жояды

SELECT [ALL|] (Кестелер немесе сұраныс жолдарының тізімі)
FROM (Сұранысқа негіз болатын кестелер немесе сұраныстардың тізімі)

5.2-кесте

Деректерді манипуляторлау операторлары

Оператор	Әрекет
DELETE	Негізгі кестеден сүзгілеу талаптарына сәйкес келетін бір немесе бірнеше жолдарды жояды. Операторды пайдалану тұтастықты сақтау принциптеріне сәйкес келеді, сондықтан осы оператор ол дұрыс синтаксиста дұрыс жазылған болса да, әрқашан дұрыс орындалмайды.
INSERT	Негізгі кестеге бір жол қосады. Операторды рұқсат берілген түрлендірулері, онда бірнеше жолдар бір кестеден немесе сұраныстан базалық кестеге ауыстырылады.
UPDATE	Бір немесе бірнеше бағанның мәндерін сүзгілеу талаптарына сәйкес келетін бір немесе бірнеше жолдарды жаңартады.

Сұраныстар операторы

Оператор	Әрекет
SELECT	Реляциялық алгебраның барлық операторларын ауыстыратын және сұранысқа сәйкес нәтижелі қарым-қатынасты жасауға мүмкіндік беретін оператор.

5.4-кесте

Әрекеттерді (транзакцияларды) басқару операторлары

Оператор	Әрекет
CCOMMIT	Транзакцияға біріктірілген кешенді, өзара байланысты ақпаратты өңдеуді аяқтайды
ROLLBACK	Транзакцияны орындау кезінде жүргізілген өзгерістерден бас тартады
SAVEPOINT	ДҚ аралық жағдайын сақтайды, оны бұдан әрі қайтып келетіндей белгілейді

5.5-кесте

Деректерді әкімшіліктендіру операторлары

Оператор	Әрекет
ALTER DATABASE	Дерекқордың негізгі объектілеріндегі жинақты, барлық дерекқорға қатысты шектеулерді өзгертеді
ALTER DBAREA	Бұрын жасалған сақтау қорын өзгертеді
ALTER PASSWORD	Барлық деректердің қорын базасын
CREATE DATABASE	Жаңа дерекқорды жасайды
CREATE DBAREA	Деректерді сақтаудың жаңа қорын құрады
DROP DATABASE	Дерекқорды жояды
DROP DBAREA	Дерекқорды сақтау саласын жояды
GRANT	Дерекқорға немесе оның жекелеген элементтеріне қолжетімділік құқығын береді
REVOKE	Дерекқорға немесе оның жекелеген элементтеріне қолжетімділіктен айырады

Кесте 5.6. Меңзерді басқару операторлары

Оператор	Әрекет
DECLARE	Сұраныс үшін меңзерді анықтайды. Атауын береді және оған байланысты ДҚ сұранысты анықтайды.
OPEN	Меңзерді ашады. Дерекқор объектісін ашады
FETH	Белгілі жазбаға меңзер белгілейді және оны оқиды
CLOSE	Меңзерді жабады. Дерекқор объектісін жабады
PREPARE	SELECT нұсқаулығына сәйкес сұранысты орындау жоспарын түрлендіреді
EXECUTE	Бұрын түрлендірілген сұранысты орындайды

[WHERE (Деректерді іріктеу шарты)]

[GROUPBY (Сұранысты орындау нәтижесінде шығарылатын жолдардың тізімі)]

[HAVING (Сұраныстағы деректерді топтастыру талаптары)]

[ORDERBY (Сұраныстағы деректерді шығару реттелетін жолдардың тізімі)]

SELECTALL нұсқаулығы құрылымында қарастырылған шешуші сөз, сұраныс талаптарын қанағаттандыратын кестенің немесе сұраныстың барлық жазбаларынан тұратын қорытындылаушы жинақты білдіреді.

Шешуші сөздер сұраныста болмауы мүмкін.

Орындалатын іс әрекеттер сипаттарына қарай SQL операторларын келесі топтарға бөлуге болады:

• деректерді анықтау операторлары;
• деректерді манипуляциялау операторлары;
• сұраныстар операторлары (тілі);
• әрекеттерді (транзакцияларды) басқару операторлары;
• деректерді әкімшіліктендіру операторлары;
• басқарудың операторлары (меңзерді басқару).

5.1- 5.6-кестеде SQL тілі операторларының тиісті топтары және олармен орындалатын әрекеттер ұсынылған.

5.3. MS SQL Server 7.0 туралы жалпы мәлімет

Бағдарламалық қамтамасыз етуді әзірлеуде әлемдегі ең танымал көшбасшы болып табылатын Microsoft корпорациясы дерекқор нарығында ауқымды зерттеулер мен заманауи дерекқорды басқару жүйелеріне қойылатын талаптар жүргізеді.

Осы ақпарат негізінде SQL Server тұқымдасы корпорациясының жаңа өнімін құру стратегиясы әзірленді. SQL Server 7.0 дерекқорды басқару жүйесінің жаңа нұсқасы шын мәнінде, көптеген жаңа технологиялардан тұратын заманауи өнім болып табылады. SQL Server ерекшелігі, оның жергілікті дерекқорларғағана емес, ондаған кестелерге, жүздеген қолданушыларға және деректердің миллион деректеріне ие кәсіпорындар ауқымының дерекқорына арналғандығы болып табылады. Екі жағдайда да сол бағдарламалық код қолданылады. SQL Server 7.0 Windows NT операциялық жүйесінің басқаруымен және Windows 95/98 жұмыс істей алады.

SQL Server жетінші нұсқасында дерекқорды басқарудың жаңа әдісін қоса, көптеген жаңа функционалдық мүмкіндіктер, OLE DB 2.0, ActiveX, COM технологияларына қолдау көрсету, Microsoft Search, OLAP, Microsoft Repository және басқа өзгерістер іске асырылған. Дерекқорды басқарудың заманауи жүйелерінің көпшілігі әкімшіліктендіруге күрделі және арнайы дайындық және жұмыс тәжірибесіне ие пайдаланушыларға арналған. SQL Server 7.0 - ноутбуктан бірнеше процессорлық кластерге дейін масштабталатын өнім болып табылады. Ол көптеген әкімшіліктерді жеңілдететін және дерекқорларды өңдеу мен жүргізу процесін жеңілдететін көптеген жаңа шешімдер мен технологиялардан тұрады.

Әр жолдағы байттардың ең көп саны – 8060 жетеді, ал кесте 1024 дейінгі бағаннан тұруы мүмкін. Деректердің жаңа түрлері пайда болды, Unicode қолдауы іске асырылды, бұғаттау жүйесі жақсарды. Бұғаттау жүйесі оңтайландырылған, енді ол тезірек бұғаттауды және синхрондау үшін аздаған ішкі жоғалуды орындайды.

SQL Server динамикалық өздігінен басқару. SQL Server 7.0 конфигурацияның кейбір параметрлері үшін автоматты конфигурация режимін қамтамасыз ете отырып, серверді әкімшіліктендіруді жеңілдетеді. Сервер қандай да болмасын ресурстарға қажеттілікті үнемі қадағалап отырады және өзінің күйге келтіру параметрлерін динамикалық түрде өзгертеді. Мысалы, егер дерекқорлардың біреуі бұдан әрі пайдаланылмаса және сервермен автоматты түрде жабылса, онда шұғыл жады мен процессорлық уақытқа қойылатын талаптар төмендейді. Серверді конфигурациялау үшін статикалық мағыналарды пайдалану кезінде қолданылмайтын ресурстар әлі де SQL Server операциялық жүйемен резервтелетінбасқа қосымшалармен қолданылмауы мүмкін. Автоматты конфигурация режимінде SQL Server операциялық жүйенің қолдаланылмайтын ресурстарын қайтаратын болады, ол жүйенің өзінің, сол сияқты қолданбалы бағдарламалардың өнімділігін арттырады.

Дерекқормен айналысатын жады көлемін ендіру және операцияларын орындау нәтижесінде үнемі өзгереді. SQL Server 7.0 бұғаттау диспетчері ірі дерекқорлармен жұмыс істегенде пайдаланатын ресурстардың көлемін динамикалық түрде басқарады, ол әкімшінің сервер конфигурациясының параметрлерін қолмен өзгерту қажеттілігінен босатады.

Автоматты конфигурациялау мүмкіндігі әсіресе ноутбук сияқты шағын жүйелерде SQL Server іске асыру әсіресе пайдалы. Бұл ретте, пайдаланушы серверді конфигурациялау туралы алаңдамастан, өніммен өнімді жұмыс істей алады. Егер сіз қиындықтардан қорықпайтын тәжірибелі әкімші екеніңізді сезсеңіз, автоматты SQL Server басқаруынан бас тартуға болады. Бұл жағдайда сіз жүйенің конфигурациясын толығымен басқарасыз.

Сұраныстарды өңдеуші. SQL Server 7.0 сұраныстарын өңдеу кешенді сұраныстарды өңдеу жылдамдығын арттыратын жаңа іздестіру әдістері іске асырылады. Енгізілген циклдар көмегімен жалғыз қосылу әдісі қолдаланатын алдыңғы нұсқасынан айырмашылығы, SQL Server сұранысын өңдеу бірдей хэштелген қатынастарды (hash join), реляциялық қосылыс әдістерін, біріктіретін сұрыпталған қосылыстарының реляциялық қосылыстарын (merge join) және хэштеу негізінде агрегациялауды (hash aggregation) қолдануы. Мұнда бір сұраныстың ішінде әртүрлі қосылыс әдістері қолданылуы мүмкін. Кестелерден деректерді алуды бастамас бұрын, оларды алдын-ала сүзуге болады. Ол SQL Server бірнеше индекстермен кестелерге индекстерді қиылысу және біріктіру техникасы қолдануымен мүмкін болатын болды. Бір кестеге арналған барлық индекстер бір мезгілде сақталады, ал шектеулерді тіркеу сұраныстарды орындау жоспарына қосылады.

SQL Server сұраныстарды параллель орындауды қолдайды. Сервер бірнеше процессорларға ие болатын болса, онда сұраныстарды орындау олардың арасында біркелкі бөлінеді. SQL Server сұраныстардың параллель орындалуы оны өңдеу жылдамдығының артуына әкелетін және сұранысты орындауға арналған жоспар жасайтын кезді өзі шешеді.

SQL Server жаңа нұсқасында таратылған транзакцияларды қолдау іске асырылған, ол бір сұраныстың қашықтағы серверлерді қосуына мүмкіндік береді. Сұраныстарды өңдеу деректердің басқа көздерімен өзара әрекеттесу үшін OLE DB технологиясын пайдаланады. Ол басқа Microsoft өнімдерімен барынша тығыз интеграциялауға мүмкіндік береді және SQL Server үшін бағдарламаларды әзірлеу процесін жеңілдетеді.

OLE DB технологиясын пайдаланудың тағы бір артықшылығы – оны реляциялық емес деректер көздеріне қол жеткізу үшін пайдалану болып табылады. Бұл жағдайда, бір сұраныс контексіндегі деректерге қол жеткізуге мүмкіндік беретін жаңа Universal Access технологиясы (әмбебап қолжетімділік) іске қосылады.

Ол экспортталған деректерді аралық сақтау үшін әмбебап серверлер деп аталатын серверлерді пайдалануды қажетсіз етеді.

Ірі көлемді дерекқорды қолдау. SQL Server бұрынғы нұсқалары 300 Мб дейінгі көлемді дерекқорды қолдаған болатын. SQL Server 7.0 бірнеше тэрабайт көлемді базалармен жұмыс істей алады. Осы нұсқадаMicrosoft Tape Format магнит таспасының жазба стандартын қолдау іске асырылды. Ол SQL Server мұрағаттарын және операциялық жүйе мұрағаттарын сақтау үшін жалпы резервтік көшірмелеу қондырғысын пайдалануға мүмкіндік береді.Өзгерістер дерекқордың резервтік көшірмелеу процесінің өзіне де әсер етті. Біріншіден, резервтік көшірме жасау және деректерді қалпына келтіру әрекеттері әлдеқайда шапшаң орындалады. Екіншіден, резервтік көшірмелеу тек соңғы толық сақтық көшірме орындалған сәттен бастап құрылған немесе өзгертілген парақтарды ғана мұрағаттайды. Одан өзге, жекелеген деректер мен файл топтарын резервттік көшірмелеуді орындауға болады, ол бүкіл дерекқорды мұрағаттау процесін бірнеше кезеңге бөлуге мүмкіндік береді.

SQL Server қауіпсіздік жүйесі. SQL Server жаңа нұсқасы Windows NT тығыз интеграциясы іске асырылады. Ол үшін Windows NT есеп жазбаларын ғана емес, сонымен қатар өзіндік SQL Server есеп жазбаларын қоса, рөлдерді жасауға болады. Пайдаланушы жекелеген дерекқор объектілеріне қолжетімділікті барынша тиімді басқаруға мүмкіндік беретін көптеген рөлдердің қатысушысы бола алады. Серверге қызмет көрсететін персонал арасында өкілдіктерді неғұрлым икемді бөлуді қамтамасыз ететін жаңа стандартты рөлдер пайда болды.

SQL Server аспаптық әкімшіліктендіру. SQL Server 7.0 аспаптық әкімшіліктендіру дерекқор серверлерін басқаруға, сұраныстарды оңтайландыруға және туындаған мәселелерді шешуге кең мүмкіндіктерді береді.

SQL Server Enterprise Manager. SQL Server жаңа нұсқасында Enterprise Manager әкімшіліктендірілген аспап Microsoft Management Console жүктелетін модулі түрінде іске асырылады. Ол иерархиялық ағаш түріндегі барлық SQL Server объектілерін білдіре отырып, кәсіпорын желісіндегі барлық дерекқор серверлерін басқаруға мүмкіндік береді.

Enterprise Manager пайдалану арқылы әр түрлі серверлер мен дерекқордың күйге келтірулерін басқаруға, қауіпсіздік жүйесін конфигурациялауға болады. Кестелерді, көріністерді, толық мәтінді индекстерді, сақталған процедураларды жасай және өзгерте отырып, операторларды тағайындауға, ескертулерді басқаруғам болады, ол CALS-технологиялары жүйесінің кәсіпорындарында ұйымдастыру кезінде өте маңызды. Сонымен қатар, Enterprise Manager тапсырмаларды жасауға мүмкіндік береді.

SQL Server тапсырмасы көптеген қадамдарды қамтуы мүмкін, олардың әрқайсысы басқарылады:

•Transact-SQL құралдарымен;

• Microsoft Visual Basic Scripting Edition көмегмен;

• Microsoft JScript қолдана отырып;

• операциялық жүйедегі командалар беріледі.

SQL Server қызмет көрсету әрекеттерінің көпшілігін Enterprise Manager қолдана отырып орындауға қолайлы. Сондай-ақ, сақталатан процедураларды пайдалана отырып немесе жүйе кестелеріне тікелей қатынасу арқылы сервер конфигурациясын басқаруға болады. Enterprise Manager кез келген әрекеттерді екі жолмен орындауға болады:

• объектіге меңзеп, тінтуірдің оң жақ батырмасын басу арқылы кіруге болатын мәтінмәндік мәзірдің командаларын;

• шеберлерді (wizards) пайдалану.

SQL Server Service Manager. Service Manager утилитасы келесі SQL Server қызметтерінің жұмысын басқаруға арналған:

• MSSQLServer - SQL Server іске қосатын қызмет;

• SQLServerAgent - әкімшіліктендіру тапсырмаларын автоматты түрде орындауға жауап беретін қызмет;

• MSDTC - таратылған транзакциялардың орындалуын басқаратын қызмет;

• MSSearch - толық мәтінді іздестіруді іске асырылатын қызмет.

Service Manager көмегімен желідегі кез келген SQL Server тізімделген қызметін іске қосуға, кідіртуге және тоқтатуға болады, сондай-ақ оларды операциялық жүйе басталған кезде автоматты түрде іске қосу мүмкіндігін береді немесе тыйм салуға болады.

SQL Server Performance Monitor. SQL Server 7.0 орнатқаннан кейін, стандартты Windows NT — Performance Monitor диагностикалық утилитасын ДҚБЖ жұмысы туралы ақпаратты жинау үшін пайдалануға болады. Performance Monitor утилитасы көбінесе уақыт бірлігіне транзакциялар саны, ашық байланыстар саны немесе орындалған сұраныстардың саны сияқты негізінен статистикалық ақпаратты жинайды. Барынша түбегейлі бақылау үшін SQL Server 7.0 жеткізілетін Profiler пайдаланыңыз.

SQL Server Profiler. Profiler утилитасы SQL Server жұмысын егжей-тегжейлі талдауға арналған. Бұл утилитаны пайдалана отырып, сұраныстар мен сақталатын рәсімдерді орындау уақыты, орнатылған қосылыстар туралы, орнатылған бұғаттаулар, белсенді транзакциялар және т.б. туралы ақпаратты жинаға болады.

SQL Server Query Analyzer. SQL Server 7.0 Query Analyzer графикалық утилитасын пайдалана отырып, сұраныстарды оңтайландыру және түзету. Ол пайдаланушыға келесі параметрлерді ұсынады:

SQL Server шеберлері

Атауы	Сипаттама
Backup Wizard	Дерекқорды резервтік көшірмелеуді орындайды
Failover Setup Wizard	SQLServer негізінде кластер ұйымдастыруға көмектеседі
Configuring Publishing and Distribution Wizard	Репликация кезінде баспа мен дистрибьютордың конфигурациялау процесін жеңілдетеді
Create Alert Wizard	Хабарлама жасайды
Create Database Wizard	Дерекқор жасайды
Create Diagram Wizard	Дерекқор диаграммасын жасайды
Create IndexWizard	Индекс құрады
Create Job Wizard	Тапсырма жасайды
Create New DataSource Wizard	ODBC-драйвер жәнеODBC— деректер көзін инсталляциялайды
Create Login Wizard	Қолданушы үшін SQLServereceп жазбасын жасайды
Create Publication Wizard	Келесі репликациялар үшін публикация жасйады
Create Stored Procedure Wizard	Сақталатын рәсімдер жасайды
Create Trace Wizard	Profiler үшін трассалауды жасайды
Create View Wizard	Көрініс құрады
Create Maintenance Plan Wizard	Қолдау файлын жасайды
Disable Publishing and Distribution Wizard	Репликация үшін баспа мен дистрибьюторды жояды
Full-text Indexing Wizard	Толық мәтінді индекстерді анықтайды
Index Tuning Wizard	Индекстерді оңтайландырады
Make Master Server Wizard	Шебер-сервер орнатады
Make Target Server Wizard	Сервер-қабылдағыш орнатады
Register Server Wizard	EnterpriseManager серверлерді тіркеу процесін жеңілдетеді
Pull Subscription izard	Деректер алу үшін жазылушыны конфигурациялайды

Атауы	Сипаттама
Push Subscription Wizard	Шығарушы баспамен жазылушыны конфигурациялайды
SQL Server Upgrade Wizard	Дерекқорды жаңарту мүмкіндігін береді
Web Assistant Wizard	Web-тапсырмалар жасайды
DTS Import Wizard	SQLServer деректерді импорттау үшін DTS— пакет жасайды

• Transact-SQL командаларын теруге болатын кіріктірілген мәтіндік редактор;

• түрлі Transact-SQL командалары әр түрлі түстерде көрсетіледі, бұл күрделі сұраныстарға жеңіл бағдарлануға мүмкіндік береді;

• сұраныс орындалған кезде логикалық қадамдар графикалық диаграмма түрінде көрсетіледі. Бұл сұраныстардың қай бөлігін ресурстарды оңтайлы түрде пайдаланатынын анықтауға мүмкіндік береді;

• индексті оңтайландыру шебері қосымша индекстерді енгізу сұраныстарының өнімділігін арттыра алатынын анықтауға көмектеседі.

SQL Server Upgrade Wizard. Upgrade Wizard шеберінің көмегімен SQL Server 7.0 дерекқорларын жаңа нұсқада пайдалану үшін түрлендіруге болады. Бұл жағдайда дерекқордың өзінің жаңартылуы ғана емес, сондай-ақ репликалау параметрлері сияқты көптеген сервер параметрлерінің жаңаруы жүреді.

SQL Server әкімшіге күнделікті тапсырмаларды орындауды жеңілдету үшін жасалған күнделікті тапсырмаларды қамтиды. 5.7-кестеде қазіргі шеберлердің тізімі берілген.

5.4. Microsoft Access ДҚБЖ

MicrosoftACCESS — ол Microsoft фирмасы әзірлеген бағдарламалық орта. Ол жеткілікті өте ірі көлемдегі ақпаратпен (жүздеген мегабайт) реляциялық дерекқорларды басқару жүйелерін жасауға арналған. Microsoft ACCESS қолданушыға деректерді жасау мен өңдеуді автоматтандыру үшін, сондай-ақ жұмыс кезінде деректерді басқару үшін барлық қажетті құралдармен қамтамасыз етеді.

Microsoft Access жүйесі өндіріс пен бизнестің түрлі салаларында ақпараттық жүйелерді дамытуға жеткілікті мүмкіндіктерге ие. Ол жұмыс үстелі дерекқоры деп аталғанымен, ол оны «үстелдік» ақпараттық жүйелерді дамыту үшін ғана емес, сондай-ақ автоматтандырылған сараптамалық жүйелерді әзірлеу немесе автоматтандырылған жобалау жүйелерін дамыту сияқты күрделі көп қолданушылық тапсырмалар үшін де пайдалануға мүмкіндік беретін сипаттамаларға ие.

(Access 2000-де әзірленген кестеде 2 миллиардқа дейін жазбалар болуы мүмкін.)

Мысалы, ТОП-ЖҮЙЕЛЕР фирмасы Microsoft Access жүйесін қолдана отырып Tflex-CAD параметрлік сызу жүйесіне кіріктірілген - TECHNO-PRO бөліктерін өндіруге арналған технологиялық процестерді автоматтандырылған жобалау жүйесі әзірледі.

К.Э. Циолковский атындағы МАТИ-РМТУ «Ұшу аппараттарын басқару аспаптар мен жүйелерін өндіру технологиясы» кафедрасындажаңа құралдарды шығарудың күтілетін шығындарын бағалау үшін «КЛАСС-ЭКСПЕРТ» сараптама жүйесі, «ЛАЗЕР-2000» құрастырудың технологиялық процестері үшін АЖЖ (САПР) әзірленді.

ACCESS ДҚБЖ дерекқор негізінде ақпараттық жүйелерді әзірлеу әдістемесін оқыту үшін басқа жүйелермен бәсекелесуден тыс қойылатын келесі сипаттамаларға ие:

• бағдарламалау тілін білмейтін мамандар тарапынан игерудің қарапайымдылығы, ол жобалау уақытын қысқартады және жүйені әзірлеу шығындарын азайтады;

• Windows қосымшаларымен үйлесімділік;

• графикалық және көп мультимедиалық объектілерді енгізу арқылы дерекқор құру мүмкіндігі;

• жергілікті және жаһандық желілерде жұмыс істеу мүмкіндігі;

• басқа бағдарламалық қамтамасыз ету жүйелерімен әзірленген ДҚ кестелерін пайдалану мүмкіндігі.

ACCESS ДҚБЖ артықшылықтарының бірі - басқа бағдарламалық өнімдерді пайдалана отырып әзірленген деректермен жұмыс істеу мүмкіндігі, деректерді басқа дерекқорларға импорттау немесе басқа дерекқордан деректерді экспорттау мүмкіндігі.

ACCESS ашық деректерге қолжетімділікті стандарттауды қолдайтын ODBC (Open Database Connectivity) кез-келген дерекқорлармен өзара әрекеттеседі.

ACCESS ДҚБЖ ақпаратпен барлық әрекеттер - деректерді оқу, кірістіру, жою және т.б. - тіл командалары арқылы орындалады SQL.

Бақылау сұрақтары

1. Дерекқорды жасау үшін бағдарламалық өнімдерді атаңыз және оларға салыстырмалы сипаттамалар беріңіз.
2. SQL тілі дерекқорды әзірлеу бағдарламалық жүйелерінде қандай рөл атқарады?
3. Тілдердің келесі сөздерінің мақсаттарын атаңыз:
   - FROM;
   - WHERE;

- GROUP BY;
- HAVING;
- ORDER BY.

4. SQL тілі қандай операторлар тобынан тұрады?
5. Алдыңғы нұсқалармен салыстырғанда SQLServer7.0. жүйесінің негізгі айрықша ерекшелігі не болып табылады?
6. SQLServer7.0. келесі құралдарының көмегімен шешілетін негізгі тапсырмаларды атаңыз:
   - SQL Server Enterprise Manager;
   - SQL Server Query Analyzer;
   - SQL Server Upgrade Wizard.
7. SQLServer7.0 шеберлері келесілері қандай функцияларды орындайды:
   - Create Database Wizard;
   - Web Assistant Wizard;
   - Create Alert Wizard.
8. MicrosoftAccess жүйенің қолдану салаларын атаңыз.

MICROSOFT ACCESS ҚҰРАЛДАРЫМЕН ДЕРЕККОРДЫ ӘЗІРЛЕУ ТЕХНОЛОГИЯСЫ

6-тарау

КЕСТЕЛЕР МЕН СҰРАНЫСТАРДЫ ӘЗІРЛЕУ

6.1. Дерекқор кестелерін әзірлеу технологиясы

Дерекқор кестелерін жасау процесін келесі кезеңдерге бөлуге болады:

• деректердің физикалық моделін жасау;
• *кесте құрастырушысының* көмегімен кесте құру;
• кестелер арасындағы байланыстарды орнату;
• кестелерді деректермен толтыру.

Деректердің физикалық моделін жасау. Компьютерді қосу мен ACCESS-ті іске қоспас бұрын, ДҚ нысандарының міндетті сипаттамаларын, яғни деректердің физикалық моделін қолдағы қарындашпен құрастыруды ұсынамыз:

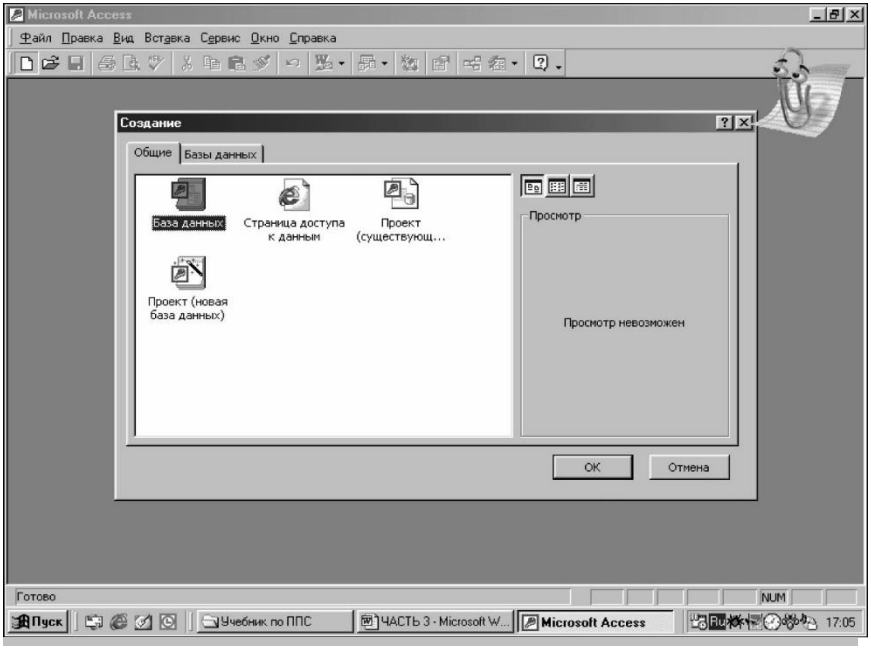
• объектінің сипаттамасын белгілерінің номенклатурасын (жолдардың құрамы мен саны) орнатуды;
• кестедегі әрбір жолдың сипаттамаларын белгілеуді;
• нәтижелерді кесте түрінде рәсімдеуді (6.1-кесте).

Объектілерді сипаттау белгілерінің құрамы мен оларға сәйкес жолдардың сипаттамалары ойластырылғаннан кейін, ACCESS ортасында кесте құруға кірісуге болады. Осы жүйенің қазіргі нұсқаларында әрекеттердің кезеңділігі іс жүзінде бірдей. Арасындағы айырмашылық тек қана диалогтық терезелерді ресімдеудегі бірнеше айырмашылықтардан тұрады. Барлық келесі мысалдар Microsoft Access 2000 нұсқасына қатысты болады.

6.1-кесте

ДҚ жолдарының сипаттамаларын сипаттау кестелері

ДҚ объектілері белгілерінің құрамы		ДҚ жолдарының сипаттамасы			
р/н №	Белгісі	Жолдың атауы	Деректердің үлгісі	Белгілер саны	Дәлділік



6.1-сурет. *Дерекқор құру* диалогтық терезесі

Кесте құрастырушысының **көмегімен кесте құру**. Кесте құрылымдары арқылы кесте жасау үшін келесі әрекеттерді орындау қажет:

• компьютерді қосып, Windows және Access бағдарламаларын қамтамасыз етуді жүктеу;

• пайда болған диалогтық терезесіне *Access* жүктелгеннен кейін, *файл* мәзіріндегі батырманы екі рет шертіп қалыңыз және *жасау* бұйрығын таңдаңыз;

• *Жасау* пайда болған диалогтық терезесіне, *дерекқор* ауыстырып қосқышты белсендіру, содан кейін OK батырмасын шертіп қалу;

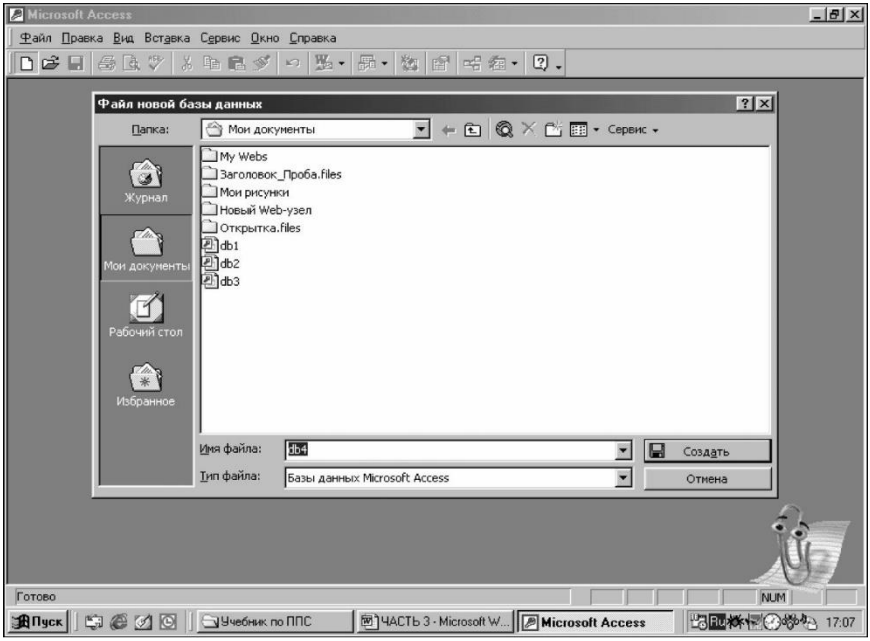
• жаңа дерекқордың *Файл* пайда болған диалогтық терезесінде, ДҚ сақталатын директорияда (папкада) атауын көрсете отырып, жаңа дерекқорға *Файл* атауын беру; *Жасау* батырмасы бойынша тінтуірді шертіп қалу.

Диалогтық терезелер 6.1, 6.2 сур. көрсетілген.

Келесі пайда болған *Дерекқордың* келесі диалогтық терезесінде *Кесте* қосымша бетін белсендіріңіз және *Құрастырушы режимінде Құру* командасын таңдау.

Пайда болған кесте *Құрастырушы* (6.3-сурет) терезесінде белгіленген құрылымы мен сипаттамаларына сәйкес кесте құрылымын жасау.

Кесте *құрастырушысы* төрт ақпараттық блоктан тұрады:



*6.2-сурет.*Жаңа ДҚ файлының атауы мен жаңа орны көрсетілген диалогтық терезе

- өрістің аталуы;
- деректер үлгісі;
- сипаттама;
- жолдың қасиеті.

Жолдың қасиеті блогында екі терезе (бетбелгілер) бар: жалпы және ауыстыру.

Жалпы жолдың қасиеті міндетті түрде толтырылады. Ауыстыру терезесінде деректерді тікелей кестеге енгізген кезде көрсетілетін мәндер тізімін көрсетуге болады. Бұл жағдайда қолданушы тінтуірді керекті мән бойынша басу керек болады. Осы жолдар тізімі бар жолдар деп аталады.

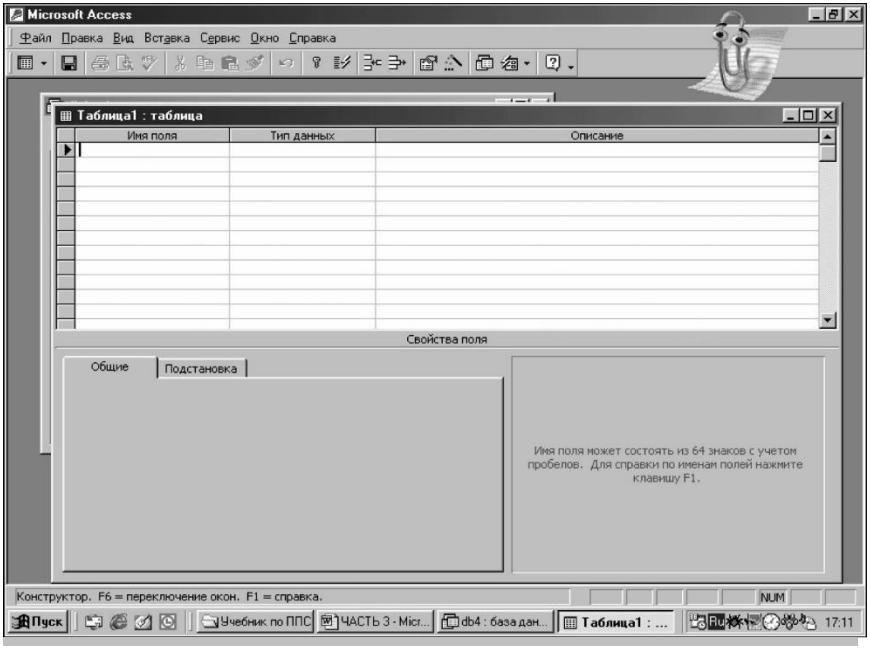
Кесте жолдарының атауларын көрсете отырып, төмендегі ұсыныстарды басшылыққа алу керек:

• жол атауы бос орыннан басталмауы керек;

• жол атауында тыныс белгілері, жақшалар, леп белгісі болмауы керек;

• кестедегі атауларды қайталауға жол берілмейді;

• жол атаулары 255 таңбадан тұруы мүмкін. Атауға ең аз таңбалар саны бар атау берілуі керек (бұл жады көлемін және ақпаратты іздеу уақытын азайту үшін қажет). Жол атауы жол ұяшықтарына енгізілетін нысанның қысқартылған атауын білдіреді.



6.3-сурет. *Кесте құрылымдары* терезесі

Кесте құрастырушысының ақпараттық блоктары жолдарына деректер толтыру технологиясы Word мәтіндік редакторындағы кестелермен жұмыс істеу технологиясына ұқсас.

Ақпараттық блоктарды толтыру әрбір жол үшін кезекті түрде жасалуы керек. Ақпараттық блоктарды толтырудың келесі тәртібін ұсынамыз:

• жолдың атауын енгізіңіз;

• деректер түрін таңдау;

• аталған жолдың ұяшығына енгізілген мәндердің сипатын түсіндіретін *Түсініктеме сипаттау* блогының жолына енгізу (бұдан әрі кестені толтырған кезде, осы түсініктеме экранның төменгі жағындағы көмекші жолға шығарылады);

• жолдың қасиеттерін белгілеу;

• жоғарыдағы әрекеттерді кестедегі барлық қалған жолдар үшін қайталау.

Жолдың атауы жоғарыда көрсетілген ұсынымдарға сәйкес енгізілгеннен кейін оған арналған деректер түрін таңдаймыз. Microsoft Access кесте *құрастырушысы кестесінен* деректердің түрін таңдауды тізімнен таңдау арқылы жүзеге асыруға болады. Тізімде келесі деректер түрлері беріледі.

Мәтіндік. Осы түрдегі жолда 255 дейін таңба болуы мүмкін. Соның ішінде сандарды қоса, кез келген таңбалар болуы мүмкін.

Мәтін жолына, егер олар онымен есептеулер жүргізбесе, бір нөмірлерді енгізуге болады.
МЕМО. МЕМО жолы мәтіндік ескертулер жолы деп аталады. Бұл жол ұзындығы 255-тен астам таңбадан тұратын мәтіндік ақпаратты енгізу үшін арналған (Access 2000 - 65 535 таңбаға дейін). Деректердің осы түрі кестеде деректердің өзі сақтамайтын мәтіннен емес, бірақ жеке сақталатын деректер блоктарына сілтеме жасайтынымен ерекшеленеді. Ол кестелерді өңдеуді айтарлықтай жылдамдатады. МЕМО жолдары кілт немесе индекстік бола алмайды.

Сандық. Бұл деректер түрі математикалық есептерге қатыса алатын дерекқор нысандарының сипаттамаларына арналған.

Күні/уақыты. Деректердің бұл түрі кестенің нақты жазбасын сипаттайтын күнді немесе уақытты көрсетуге арналған (мысалы, қоймадағы тауарды алу күні немесе Интернетте қолданушының жұмысының басталу және аяқталу уақыты). Аталған жолға 100-ден бастап 9999-ға дейінгі күндерді енгізуге болады.

Ақшалық. Деректердің осы түрі сандыққа ұқсас. Ол тек енгізілген сандардың ерекшеліктерімен ғана ерекшеленеді. Нөмірдің дәлдігі төрт ондық саннан аспайды. Толық бөлік 15 таңбалы саннан тұруы мүмкін. Санның соңында валюталық белгілер (р. немесе $) көрсетілуі мүмкін.

Есептеуіш. Жол дерекқор кестесінің бірегей (қайталанбайтын), жазбасынан тұрады. Осы жолдың мәндері жаңартылмайды.

Логикалық. Параметрлері тек Иә немесе Жоқ (Иә/Жоқ), Айқын/Жалған, Қосылған/Өшірілген ретінде параметрлері түсіндірілген тек екі мәнді қабылдайтын жолдың түрі. Логикалық түрдегі жолдарда кілт болмайды, бірақ индекстік жолдар болуы мүмкін.

OLE (OLE нысаны). Бұл жол ұяшықтарында Windows үшін әзірленген қосымшаларға сілтемелер енгізіледі. Бұл мәтін, графикалық және мультимедиалық файлдар болуы мүмкін. Бұл жолдағы ұяшықтарда сақталған деректер көлемі тек компьютердің дискілік кеңістік арқылы шектеледі.

Гиперсілтеме (*Hyperlink).* Бұл деректер түрі жол гиперсілтемесін кіргізуге мүмкіндік береді, оның көмегімен дерекқор кестесі орналасқан компьютерде немесе жергілікті желідегі немесе Интернеттегі кез-келген компьютерде орналасқан кез-келген файлға немесе файлдың үзіндісіне сілтеме жасай аласыз. Гиперсілтеме үш бөліктен тұрады: файлға жолды көрсететін мекенжайы; фрагменттің орналасқан жерін көрсететін қосымша мекенжай мәтіннің немесе парақтың бетінде; көрсетілетін мәтін. Гиперсілтеменің әрбір бөлігінде 2048 дейін таңба болуы мүмкін.

Алмасу шебері. Бұл түр таңдалғанда, жол ұяшықтарына енгізілген деректермен қабылдануы мүмкін тұрақты мәндер тізімін жасауға болады.

Деректердің атауын және түрін орнатқаннан кейін, меңзерді *Сипаттама* блогындағы тиісті жолға қойып, кестені толтырған кезде пайдаланушыға ақпаратты дұрыс енгізуге мүмкіндік беретін түсініктеме енгізу.

Түсініктемені енгізуді ұсынамыз, әсіресе дұрыс деректерді енгізу үшін жолдың атауын немесе қолын көрсету үшін жеткілікті ақпарат болмаған кезде.

Түсініктемені енгізгеннен кейін, *Жолдың сипаты* блогына өтіңіз, *Жалпы* бөлім және қажетті сипаттарды жолға орнатыңыз. *Кесте құрастырушысында* деректер түріне байланысты әрбір жол автоматты түрде (бастапқы күй бойынша) белгілі бір сипат жиынына беріледі. Кестені құрылымдай отырып, бұл сипаттарды нақты деректер талаптарына сәйкес өзгертуге болады.

6.2-кестеде *Жолдың сипаттары, Жалпы* ақпараттық блогында берілетін жол сипаттамаларының қасиеттері тізімделген.

Кестедегі барлық өрістердің сипаттамаларын (қасиеттерін) сипаттағаннан кейін, *Құрастырушы* жабылады; бұл ретте, диалогтық терезелер ашылады, онда сізден кестенің атауын көрсету сұралады және олар көр сетілмесе, кілт жолдары орнатылады.

Кестенің атауы. Кесте атауын көрсету кезінде келесі ұсыныстарды қарастыру керек:

• жолдың атауы кестедегі деректердің мазмұнын көрсетуі тиіс (объектілерсыныбын);

• кестенің атауында тыныс белгілері, жақшалар, леп белгісі болмауы керек;

• кесте атауы бос орыннан басталмауы керек;

• бір дерекқор файлында бірдей атаумен кестелер болмауы керек.

6.2-кесте

ДҚ кестелері жолдарының сипаттамалары

Жолдың қасиеті	Сипаттамасы
Жолдың өлшемі	Аталған жолдың ұяшықтарда енгізілген деректердің ең үлкен өлшемін орнатады. Мәтін (таңба) өрістерінің деректер мөлшері 255 таңбадан аспауы керек. Сандық өрістер үшін деректердің өлшемі санның түріне қарай автоматты түрде орнатылады: байт - 0-ден 255-ге дейінгі тұтас сандар - 1 байт; бүтін сан - -32 768-ден +32 767 -ге дейінгі бүтін сандар - 2 байт; 2 147 483 648-ден +2 147483 648-ке дейінгі ұзын бүтін сандар;алты таңбаға дейінгі дәлдікпен өзгермелі нүктелі бүтін сан -3,4 х 1038-ден +3,4 х 1038-ге дейінгі сандар - 4 байт; сегіз таңбаға дейінгі дәлдікпен өзгермелі нүктелі бүтін сан - 1.797 х 10 308 - + 1.797 х 10 308 - 8 байт сандар

6.2-кесте жалғасы

Жолдың қасиеті	Сипаттамасы
Жолдың форматы	*Мәтін және МЕМО* сияқты мәтін жолдары үшін деректерді енгізу форматын дисплей экранында көрсетілетін деректерге сәйкес орнатуға болады. *Сандық, Ақшалық* түрлердің жолдары үшін келесі форматтарды таңдауға болады: стандартты – бастапқы күй бойынша орнатылатын формат (мыңдаған бөлгіштер, валюталық белгілер, ондық сандардың саны санның дәлдігіне сәйкес келеді); ақшалық - үтірден кейін екі белгі белгіленеді және валюта белгісі бейнеленеді; бекітілген– үтірге дейінгі кемінде бір белгі және үтірден кейінгі екі белгі; мыңдаған бөлгіштермен –үтірден кейін екібелгі және мыңдаған бөлгіштермен; пайызбен - санның соңында пайыз белгісі шығарылады; экспоненталық сандар экспоненталық түрде көрсетіледі (мысалы, 1,10 x 103). Күн/уақыт түріндегі жолдар үшін келесі форматтар бар: толық күн форматы – бастапқы күй бойынша орнатылады және мыналарды қамтиды: 15.04.97.05:3:10 PM; ұзын күн форматы, мысалы:1997 жылғы 13 сәуір, жұма; орташа күн форматы, мысалы: 13-сәуір-97; қысқа мерзімді формат, мысалы: 13.04.97; ұзақ уақыт форматы, мысалы: 14:33:10; орташа уақыт форматы, мысалы: 14:33 PM; қысқа уақыт форматы, мысалы: 14:33. Логикалық типті жолдар үшін келесі форматтарды пайдалануға болады: Иә/Жоқ; Шындық / өтірік; Қосу / өшіру
Ондық белгілер саны (жолдың дәлдігі)	*Сандық және ақшалық* түрдегі жолдар үшінберіледі. Бастап дейінгі белгілер саны
Енгізу маскасы	Маска деректерді *Мәтіндік, Сандық, Ақшалық, Күн/Уақыт* түрлеріне енгізу үшін үлгісін орнатады. Күн/Уақыт түріндегі жолдарға енгізу маскасы таңдалған форматқа сәйкес келеді

6.2-кесте жалғасы

Жолдың	Сипаттамасы
Жолдың қолы	Ол кестелердің және басқа нысандардың элементтерін, есептердің атауларына («тақырыптарына») енгізілетін жолдың барынша сипаттамалық атауына арналған. Егер өрістер енгізілмесе, кестелердің, пішіндердің форматтардың және есептердің тиісті элементтері жол атауларына енгізіледі
Мағынаға қойылатын талап	Кіріс деректерінің мәндері бойынша шектеулерді орнатады. Мысалы, сандық жол үшін «<100» шартын көрсету осы жолдағы 100-ден көп деректерді енгізе алмайтыныңызды білдіреді. «Мәскеу» OR «Вологда» OR «Новосччирск» типіндегі талаптар, енгізілетін қала атауларының Мәскеу, Вологда немесе Новосибирск қана болатындығын білдіреді. Кіріс деректерінің мәндері салыстыру үшін пайдаланылатын салыстыру операторларынан және мәндерден тұратын өрнектермен анықталады. Шарттарды анықтаған кезде белгілі операторлар пайдаланылады: <(аз); <= (аз немесе тең); > (көп); > = (үлкен немесе тең) Шарттарды анықтаған кезде белгілі операторлар пайдаланылады: = (тең); <> (тең емес). Өрнектерде OR (немесе), AND (және) логикалық операторлары, сондай-ақ BETWEEN, IN, LIKE салыстыру операторлары пайдаланылуы мүмкін: BETWEEN - енгізілген жол мәнінің көрсетілген ауқымның ішінде екенін тексереді. Ауқымның жоғарғы және төменгі шекаралары логикалық оператор мен AND бөлінеді. Мысалы, BETWEEN 20 AND 45 өрнегі кіріс мәні 20-дан 45-ке дейін болуы керек дегенді білдіреді. Бұл өрнек келесідей жазылуы мүмкін:> 50 AND<100; IN - енгізілген жол мәнінің теңдігі көрсетілген тізімнен келген мәнге тексереді. Мысалы,IN («Мәскеу», «Вологда», «Новосибирск») бұл өрнек «Мәскеу» немесе «Вологда» немесе «Новосибирск» өрнегіне сәйкес келеді; LIKE - *Мәтін немесе Мемо*өрістері көрсетілген белгі үлгісіне сәйкес келетінін тексереді. Мысалы, LIKE «Тех*» өрнегі енгізілетін таңба жолының «Тех» белгісінен басталуы тиіс екенін білдіреді.

6.2-кесте соңы

Жолдың	Сипаттамасы
Қате туралы хабар	Кіріс деректерінің мәндері үшін берілген шарттарда сәйкессіздіктер болған жағдайда экранға шығарылатын мәтін
Міндетті жол	Егер өріс міндетті түрде таңдалса, бұл кесте осы өрістің ұяшықтарында толтырылған кезде деректер міндетті түрде енгізілуі керек дегенді білдіреді
Бос жолдар	*Мәтін* және *Жазба* өрістеріне бос жолдарды енгізуге берілген рұқсат
Индекстелген жол	Деректерді кестелерде іздеуге болатын жолдар үшін осы мағынаны орнату ұсынылады. Индексті орнату деректерді жылдам іздеуді жылдамдатады

Шешуші өріс. Шешуші өріс дерекқордың кестесінің деректері басқа кестелердің деректерімен байланыстырылуы керек жағдайларда орнатылады. Шешуші өріс кестеде әр жазбаны бірегей түрде анықтауы керек. Осы шешуші өрістердің мәндері қайталанбайды (қайталанбауы керек).

Егер осы өрістің деректер мәндері бүкіл жазбаны анықтай алса, шешуші өріскестенің кез келген өрісі болуы мүмкін.

Жазба бір өрістегі деректердің мәнімен анықталмаса, онда бірнеше шешуші өрістер орнатылады.

Шешуші өріс ретінде, кестеде әрбір жазбаны бірегей түрде анықтайтын Есептегіш типті өрісті таңдауға болады.

Шешуші өріс *Кесте құрастырушысының* өрістер сипаттарын сипаттаған кезде құрылады. Ол үшін қажетті жолды таңдап, құралдар тақтасындағы сәйкес тиісті батырманы басыңыз.

6.4-суретте кестенің *Кесте Құрастырушысының* кесте құрылымының үзіндісі көрсетілген.

Кестелерді әзірлеу технологиясына қатысты кейбір ескертулер жасайық. *Кесте құрастырушысының* жұмыс істеу технологиясы толық Word мәтіндік редакторындағы кестелерге ұқсас жұмыс істейді.

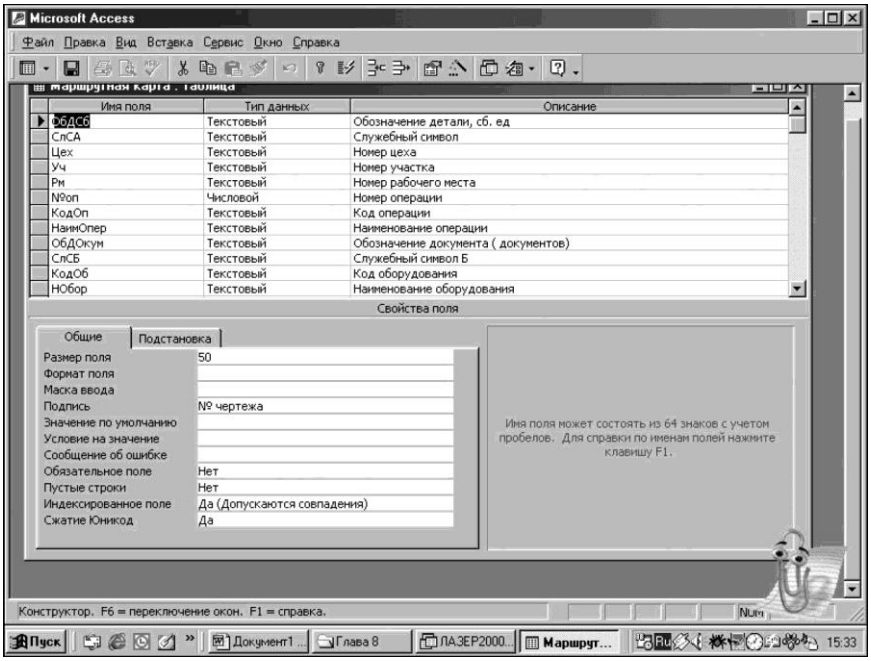
Нысандардың бірдей сипаттамаларын қамтитын бірнеше кестелерді жасаған кезде деректерді көшіру технологиясын қолдануыңыз қажет. Ол үшін:

1) бұрын құрылған кестені *құрастырушы* режимінде ашу;

2) басқа кестеде қайталанатын өрісті бөлу;

3) таңдалған өрісті (оның барлық қасиеттері бар) алмастыру буферіне көшіру;

4) басқа кестені құрастырғанда, өрістің сипаттамаларын айырбастау буферінен кестенің конструкторының тиісті жолына қою қажет.



6.4-сурет. Құрастырушы терезесіндегі кестенің толтырылған сипаттамаларының мысалы

Дерекқор кестелерінің құрылымы жасалғаннан кейін, кестелер арасында логикалық байланыстарды орнату қажет.

Кестелер арасында қатынастар орнату. Кестелер арасындағы байланыстарды орнату деректердің тұтастығын қамтамасыз ету үшін қажет.

Деректердің тұтастығы ақпараттың байланыстырылған кестелердегі кездейсоқ өзгерістерден қорғалғанын қамтамасыз етеді.

Байланыстырылған кестелерде бір кесте бастапқы, екіншісі бағынышты болып табылады. Негізгі кесте міндетті түрде шешуші өрісті сақтауы керек. Бағынысты кестеде шешуші өріс емес ұқсас өріс болуы керек.

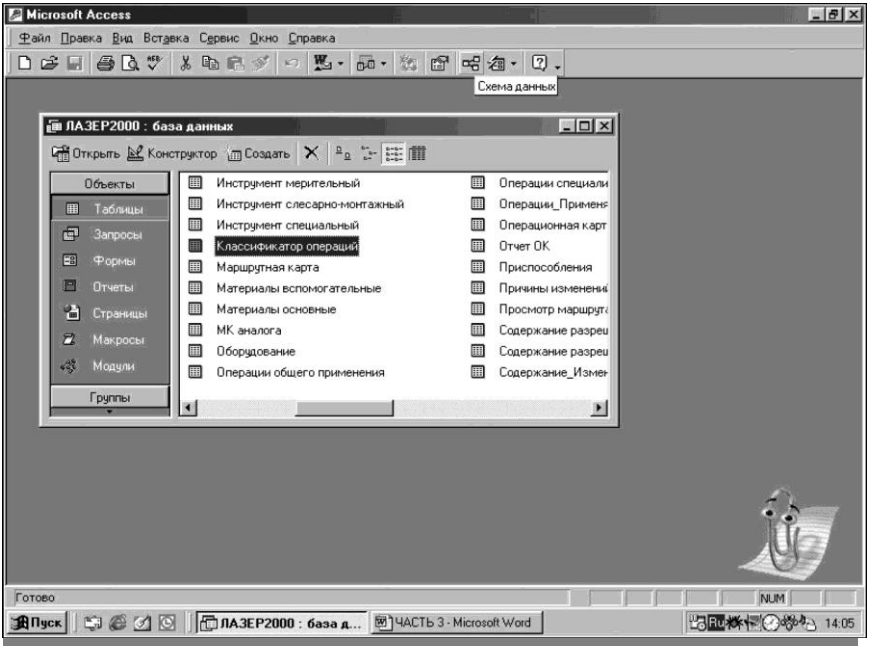
Кестелер арасындағы қатынастарды орнату үшін келесі әрекеттерді орындау керек.

• Дерекқор терезесінің құралдар тақтасындағы команданы (белгіше) белсендіру. *Деректер сұлбасын* іске қосыңыз (6.5-сурет);

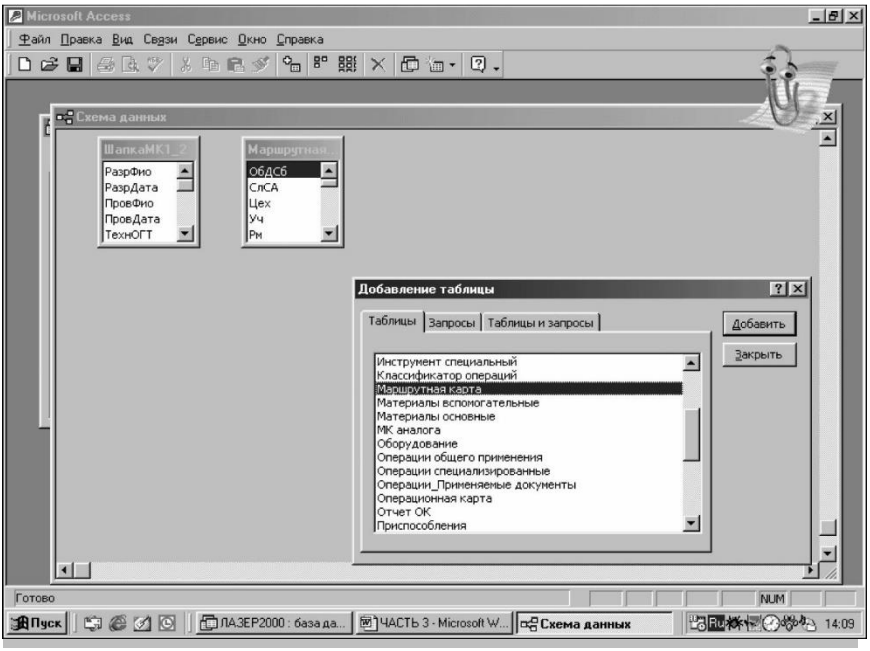
• ашылған деректер сызбасын құрастырушы терезесінде (6.6-сурет) негізгі және бағынды кестелерді енгізіңіз;

• кестелерді бірдей өрістер бойынша байланыстырыңыз.

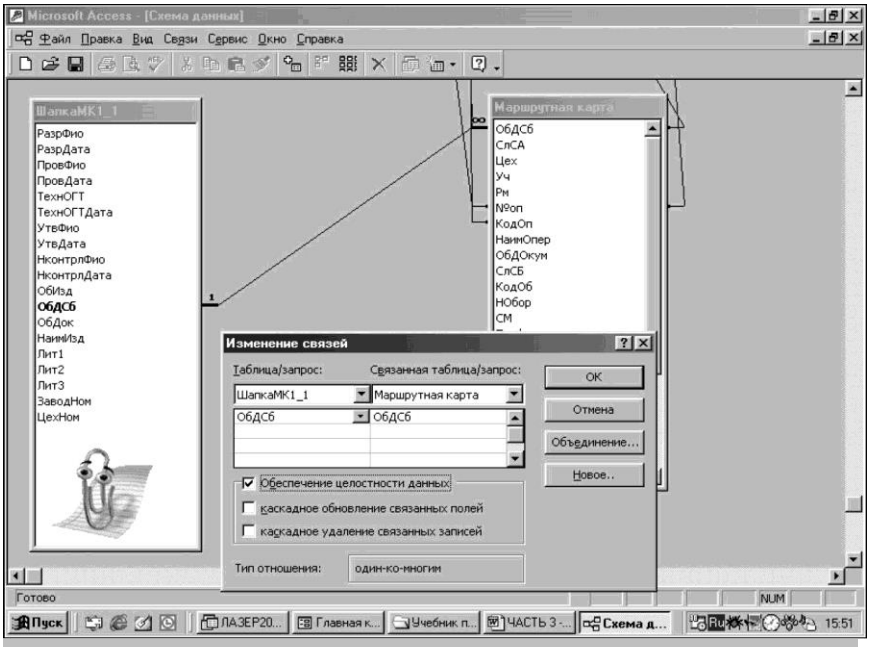
6.7-суретте *Деректер сұлбасы* терезесі көрсетіледі, онда САПР ТП «ЛАЗЕР 2000» екі кестесінің арасында «біреуі көпшілігіне» қатынас орнатылған. Байланысты жасаған кезде *Деректер тұтастығы параметрі* қосылады.



6.5-сурет. *Дерекқор* терезесі



6.6-сурет. «Біреуі көпшілігіне»деректер сұлбасы



6.7-сурет. «Біреуі көпшілігіне»деректер сұлбасы

Бұл параметр негізгі кестедегі жазбаларды ерікті түрде жоюға немесе өзгертуге мүмкіндік бермейді.

Егерде байланыстыру параметрлерін кестелер арасында орнатсаңыз (қоссаңыз) *Байланыстырылған өрістерді каскадты жаңарту* және *Байланыстырылған жазбаларды каскадты жою*, онда негізгі кестедегі деректерге кез-келген өзгерістер кезінде автоматты түрде бағынатын кестедегі байланысты деректерді өзгереді.

Деректер базасының кестелерін құрастырғаннан кейін әрбір кестенің құрылымы жасалады, кестелер арасындағы қатынастар анықталады және орнатылады, ал деректер жиыны кестелермен толтырылады.

Деректермен кестелерді толтыру. Кестелердегі деректерді енгізу технологиясы екі жолмен жасалады:

• деректерді кесте ұяшықтарына тікелей енгізу;

• нысандар арқылы деректерді енгізуді ұйымдастыру.

Деректерді енгізудің бірінші әдісін таңдағанда, оны басқару керек:

• операторлық қателері ықтималдығын төмендету;

• деректерді енгізу процесінің өзін ұйымдастырудың ыңғайлылығы.

Егер дерекқор кестесінде монитор экранында орналастырылған және басқа кестелермен байланысы жоқ өрістер аз болса немесе коммерциялық емес жүйе жасасаңыз, деректерді енгізу үшін сәйкес форманы жасамауға болады.

Басқа жағдайларда кестелерге деректерді енгізу үшін формаларды әзірлеуді ұсынамыз (7-тарауды қараңыз).

6.2. Сұраныстарды әзірлеу технологиясы

Кез-келген ақпараттық жүйенің негізгі мақсаты - пайдаланушыны қажетті және сенімді ақпаратпен қамтамасыз ету.

ДҚ кестелерінде қамтылған ақпаратты өңдеу сұраныстар көмегіменжүзеге асырылады.

Сұраныстар - пайдаланушы анықтаған шарттарға (өріс мәндері) сәйкес кестелерде ақпаратты іздестіруге және өңдеуге алдын-ала тағайындалған кейбір командалардың жиынтығы. ACCESS жүйесінде орындалатын әрекеттерге байланысты келесі сұраныс түрлерін жасауға болады:

- әрекеттерді орындау (іріктеу үшін);
- жаңарту;
- қосымша;
- жою;
- кестелер жасау.

Сұраныстардың әрқайсысы оларды құру технологиясы мен ақпарат нысанын әр түрлі болуымен ерекшеленуі мүмкін. Жасау технологиясына байланысты сұраныстарды тұрақты және параметрлік бөліктерге бөлуге болады.

Тұрақты сұраныстар - бұл сұраныстар, ұзақ уақыт бойы өзгермейтін ақпаратты таңдау шарты.

Параметрлік сұраныстар - ақпаратты таңдау параметрлері өзгеретін сұраныстар.

Сұраныстарды орындау нәтижесі динамикалық кестелер болып табылады. Нысаны бойынша динамикалық кестелердің екі түрі болуы мүмкін:

- құрылымы ДҚ бастапқы кестесіне (кестелеріне) сәйкес келеді;
- кестелер құрылымы ДҚ бастапқы кестесінен (кестелерінен) ерекшеленетін, яғни *кросс-кестелер* деп аталады.

Кросс-кестенің құрылымы дерекқордың түпнұсқа кестесінің құрылымынан ерекшеленеді, себебі кестеде көрсетілген баған тақырыптары атаулар емес, таңдалған өрістердің мәндері болып табылады. Мұндай кестелер сұраныстардың арнайы түрлерімен – қиылысқан сұраныстармен жасалады.

Қиылысқан сұраныстар - бір мезгілде деректерді топтастыруды жеке өрістердің мәндері бойынша таңдау үшін пайдаланылатын сұраныстар. Осындай сұраныс арқылы деректер кросс-кесте түрінде қалыптастырады.

Сұраныстарды жасау жолдары. ACCESS-те пайдаланушыға сұраныстарды жасаудың екі жолы ұсынылады:

1) Шеберлер көмегімен *құрастырушы* режимінде жобалау;

2) бағдарламалау —SQL режимінде.

Құрастырушы(Конструктор) режимінде сұраныс жасау шебердің көмегімен орындалады. Бұл жағдайда, пайдаланушы шебердің мүмкіндіктерін пайдаланып, құрылымдау терезесінде сұраныс параметрлерін көрсетуі керек. ACCESS бұл жағдайда автоматты түрде бағдарлама кодын SQL тілінің командаларын арнайы ретпен жасайды.

Сұранысты SQL режимінде бағдарламалаған кезде, пайдаланушы SQL режимін пайдаланып, сұраныс арқылы орындалатын барлық әрекеттерді сипаттауы керек.

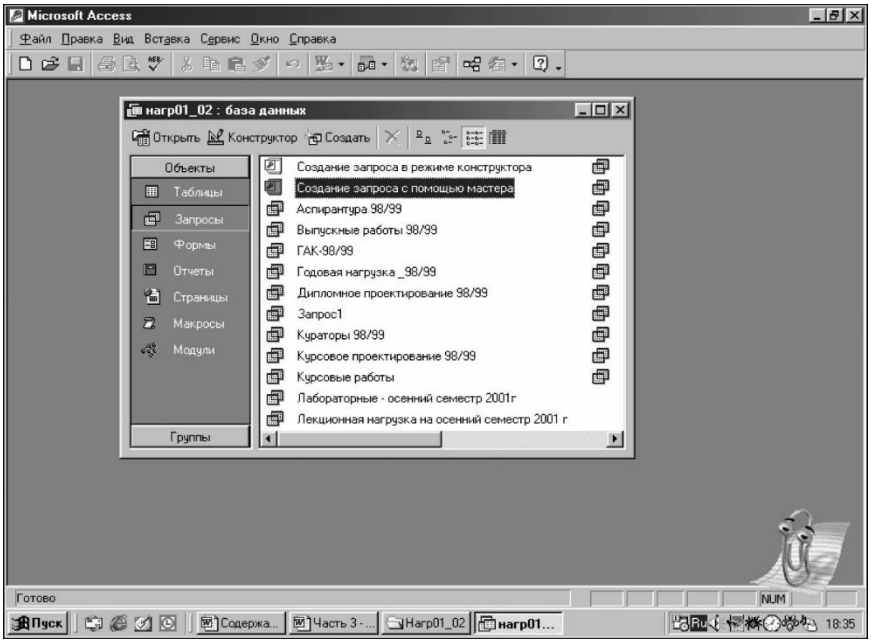
Құрастырушы режимінде сұраныстарды жасау мүмкіндіктері дерекқор кестелерінде ақпаратты өңдеудің кез келген міндетін дерлік жасау үшін жеткілікті. Сұранысты рәсімдеу технологиясы ACCESS ДҚБЖ 2000 мысалында талқыланады.

Құрастырушы режимінде сұраныс әзірлеу. ACCESS ДҚБЖ 2000 сұранысты екі әдіспен орындауға болады:

1) жаңа сұранысты дербес жасау;

2) шеберлерді пайдаланып сұраныс жасау.

Кез-келген әдіс үшін ДҚ элементтерінің терезесін ашу қажет (6.8-сурет). *Дерекқордың объектінің терезесі сұранысы* батырмасы белсендірілген кезде, барлық дерекқор сұраныстары бар терезе ашылады.



6.8-сурет. ДҚ элементтерінің (сұраныстар) терезесі

(Бұдан кейінгі барлық мысалдар К.Э. Циолковский атындағы ТПП және СУДА МАТИ-РМТУ кафедрасында әзірленген деректер негізінде оқытушылардың жүктемесі мен талдаулары үшін қаралатын болады.)

Сұраныс жасаған кезде қолданушы сұранысты құрылмдаудың келесі нұсқаларын таңдай алады:

1) *Құрастырушы* режимінде сұраныс жасау;

2) шебердің көмегімен сұраныс жасау.

Бірінші әдісті таңдаған кезде *Сұраныс жасау* терезесі ашылады (6.9-сурет), онда қолданушыға сұранысты әзірлеудің келесі режимдері беріледі:

1) сұранысты дербес құрылмдау (*Құрастырушы режимі*);

2) шебер-режим көмегімен сұранысты құрылмдау:

• қарапайым сұраныс;

• қиылысқан сұраныс;

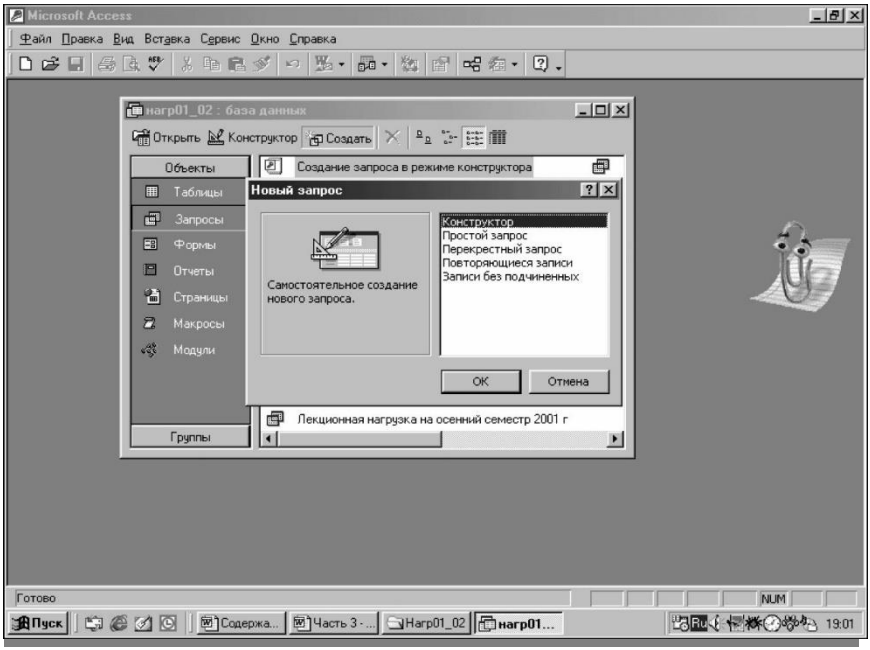
• қайталанатын жазбалар;

• бағынышты емес жазбалар.

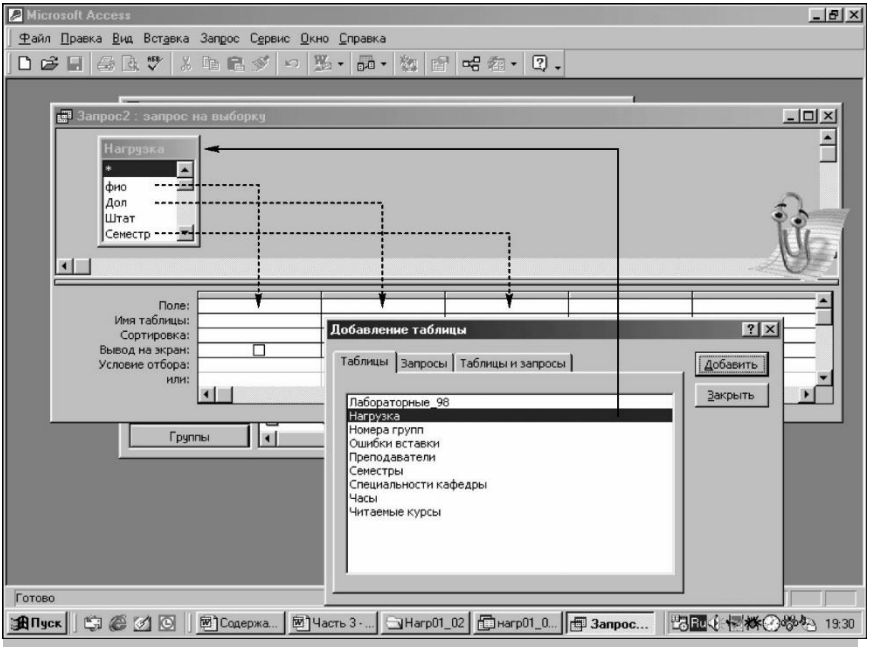
Сұранысты Құрастырушы **көмегімен сұраныс жасау.** *Сұранысты Құрастырушы* көмегімен сұраныс жасаған кезде (6.10-сурет) төмендегі әрекеттерді орындау қажет:

1. *Сұранысты Құрастырушы* ашу.

2. *Кестені қосу* ашылатын терезесінде кестені немесе кесте негізінде сұранысты жасау құрылады (*Қосу* командасы).



6.9-сурет. Сұранысты құру кезіндегі бастапқы диалогтың терезесі



6.10-сур. *Сұраныстарды* Құрастырушы терезесі

Тиісті кестенің атауы бойынша тінтуірді басу арқылы таңдау жүзеге асырылады. Дегенмен, *Құрастырушы* терезесінде өріс атаулары бар кесте пайда болады. (6.10-суретте осы әрекеттер орындау тұтас бағыттармен көрсетілген).

1. Кестеге (лерге) кіргеннен кейін *Жабу* батырмасын басыңыз.

2. Сұраныс үшін қажетті кестелерді *Сұраныстар құрастырушы* терезесінің өрісі жолдарына жылжытыңыз (6.10-суреттеосы әрекеттерді орындау нүктелі бағыттармен көрсетіледі).

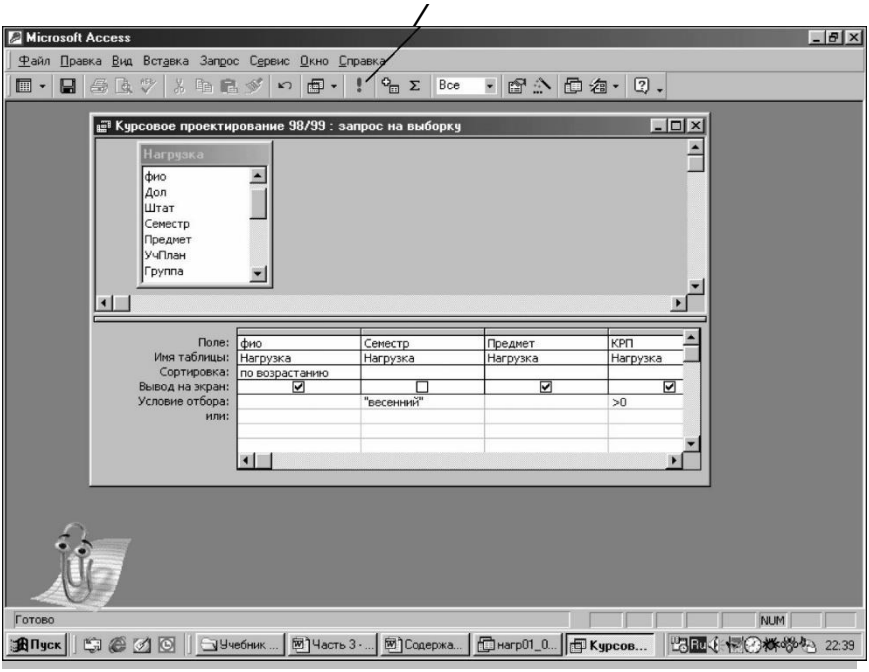
1. Жазбаларды қандай да болмасын өрістердің мәндері бойынша сұрыптау тәртібі орнатылсын (мысалы, жазбаларды *Аты-жөні* жолы бойынша әліпбилік ретпен сұрыптау.

2. Жолдың мәндерді экранға шығару қажеттілігін анықтаңыз. Егер «Ия» болса, онда шаршы бойынша тінтуірді басыңыз.

3. «Таңдау критерийлері» жолына кестелерден деректерді іріктеу жүргізлетін жолдардың тиісті мағыналары енгізілсін.

6.11-суретте орындалатын әрекеттерден кейін әзірленген сұраныс көрсетілген.

Сұраныс оқытушыларды – *Аты-жөні* жолынан таңдау үшін «Жүктеме» кестесінен жазбаларды іріктеп алуы тиіс, онда көктемгі семестрде – *Семестрлік* жол - пәндер бойынша алдын-ала өткізілген *Пәндер* алаңы бар - ХҚК жолындағы курстық жобалар.



6.11-сур. *Құрастырушы* режиміндегі сұраныстың түрі

Бұл жағдайда оқытушылардың тегі әліпби тәртіппен көрсетілуі керек, ал *Семестрдің* жолының мәні экранға шығарылмауы керек.

Құрылымдалған кезде көрсетілген шарттарды орындау үшін сұраныстың келесі параметрлері енгізілді:

• «Сұрыптау» жолында *Аты-жөні* жолы үшін сұрыптаудың тиісті тәртібі - өсу тәртібінде беріледі;

• «Іріктеу шарттары» жолында «Семестр» жолы үшін «көктем» шартыенгізілген;

• «Іріктеу критерийлері» жолында КРП жолы үшін «<> 0» шарты енгізілген, яғни осы жолда курстық жұмысты орындау үшін бөлінген сағаттар саны нөлден жоғары болуы тиіс;

• «Экранға шығару» жолында *Семестр* жолының мәндерін шығарудан бас тартылады.

Сұранысты құрылымдау процесінде оның орындалуын тексеруге болады. Ол үшін құралдар панеліндегі Сұранысты орындау батырмасы бойынша тінтуірмен жеткілікті басыңыз.

Біз таңдауға сұраныс жасау технологиясын қарастырдық. Сұраныстардың басқа түрлерінің жобалау технологиясы ұқсас, бірақ сұранысты жабудан бұрын тізімнен қажетті түрді таңдаған дұрыс (6.12-сур.).

Сұранысты құрылымдау аяқталған соң оны жабу керек (*Құрастырушы* терезесіндегі тиісті батырманы басыңыз). Бұл ретте сұраныстың атауын енгізу ұсынылатын диалогтық терезе ашылады.

Деректерді таңдау үшін шарттарды құрастыру ережелері. Деректерді іріктеу шарттарын көрсете отырып келесі ережелер сақталуы керек.

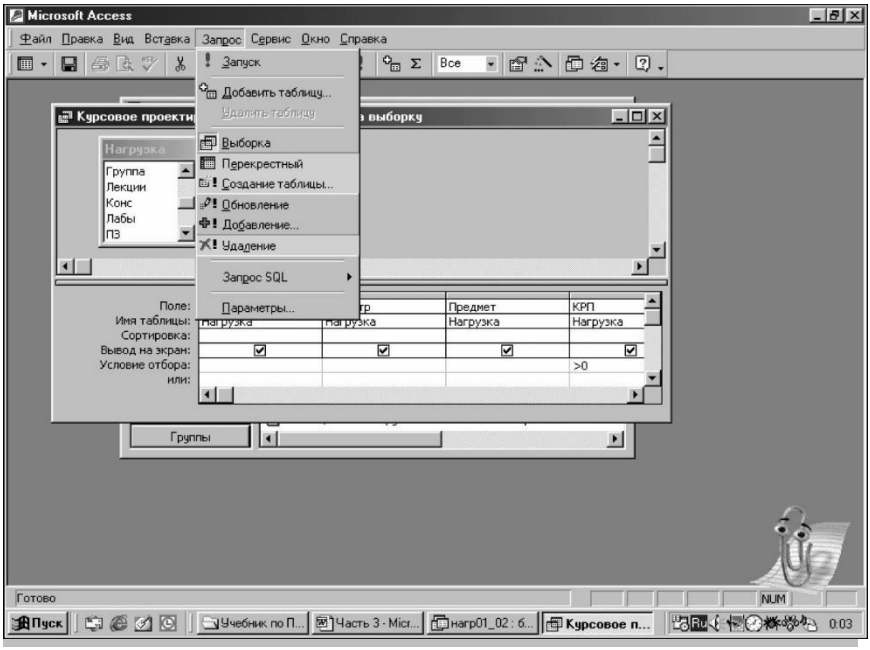
1. Мәтіннің (таңба) өрісінің мәнін енгізбес бұрын «=» белгісі болуы керек.

2. Енгізілетін таңбалар жолы тырнақшаға алынуы керек. Бұл ереже, егер оны сақтауды ұмытып қалсаңыз, ACCESS өзі орындайды.

3. Үлгідегі математикалық шарттар мәлім салыстыру операторлармен анықталады (=, <>, <,>, <=,> =).

Осы операторлардан басқа BETWEEN, IN, LIKE арнайы салыстыру операторларын пайдалануға болады (6.2-кестені қараңыз).

Күн/Уақыт өрістеріне арналған іріктеу шарттарын жасаған кезде, сұранысты құрастырған кезде есептелген өрістер ретінде енгізілетін шарттар (функциялар) қолданылуы мүмкін (6.3-кесте).



6.12-сурет. Сұраныс түрін таңдау терезесі

***Күн/Уақыт* үлгісіндегі өрістерді таңдау кезінде немесе сипаттарын беру үшін қолданылатын функциялар**

Функциясы	Мағынасы
Day	1-ден 31-ге дейінгі ауқымдағы ай сандарын таңдау үлгісін белгілейді
Month	1-ден 12-ге дейінгі ауқымдағы ай сандарын таңдау үлгісін белгілейді
Year	100-ден 9999-ға дейінгі диапазонда жылдар бойынша таңдау талаптарын белгілейді
Weekday	Аптаның күндері бойынша 1 (жексенбі) күнінен 7 (сенбі) дейін таңдау талаптарын белгілейді
Hour	Тәулік сағаттары бойынша 0-ден 23-ке дейін таңдау талаптарын белгілейді
Datepart «q» или«ww»	Уақыт ауқымдары бойынша (апта нөмірі, тоқсан саны) таңдау талаптарын белгілейді. Мынадай жазылады: Datepart «q» - тоқсан бойынша таңдау үшін; Datepart «ww» - апта бойынша таңдау үшін (q 1-ден 4-ке дейінгі мәндерді қабылдайды, ww 1-ден 53-ке дейінгі мәндерді қабылдайды).
Date ()	Ағымдағы күнді таңдау талаптарын белгілейді, мысалы, «<Date () - 15» шарты 15 күн ішінде ағымдағы күнінен аз болатын барлық жазбаларды таңдайтынын білдіреді.

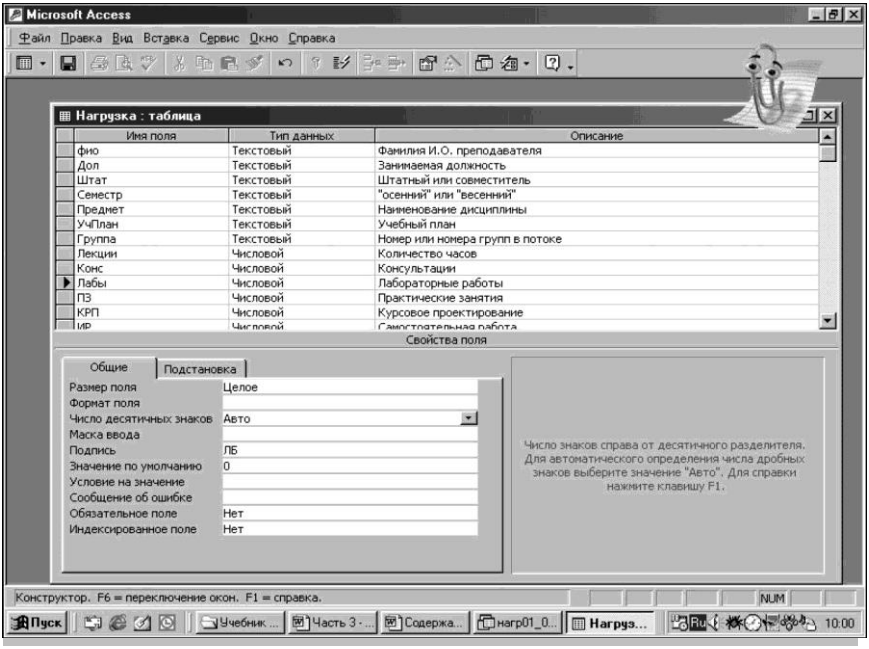
Егер бір мезгілде бірнеше параметрлерге, бірнеше жолдар мағынасына жауап беретін жазбаларды таңдау қажет болатын болса, ол И (AND) логикалық талаптарына сәйкес, онда «*Таңдау талаптары*» жолына жолдардың тиісті мағыналары енгізіледі

Егер жазбалар таңдау НЕМЕСЕ (OR) жол бермейтін қарым-қатынасқа байланысты болатын болса, онда тиісті мағыналар «НЕМЕСЕ» жолының ұяшығына енгізіледі енгізіледі.

Сұраныстар Құрастырушы жолына тікелей деректерді іріктеу талаптарын енгізуді, тұрақты сұраныстарды құрылымдау кезінде ұсынамыз.

Параметрлік сұраныстарды енгізу талаптарын құрылымдау кезінде деректерді іріктеуді нысан арқылы жүргізуді ұсынамыз.

Қиылысқан сұраныстарды құрылымдау. Нәтижесі кросс-кестелер болып табылатын қиылысқан сұраныстар, кестелерде немесе сұраныстарда сандық өрістерді талдау үшін мақсатқа сай болады. Мысалы, төменде келтірілген мысалда «Жүктеу» кестесінің негізінде, әрбір оқытушыға олар үшін әртүрлі топтарда жүргізілетін зертханалық жұмыстардың сағаттарының санын қарау қажет. «Жүктеме» кестесінің құрылымы 6.13- сур. берілген.



6.13-сурет. Оқытушылар жүктемелерін талдау үшін кестені құрылымдаудың мысалы

Сұрау орындау нәтижесі бойынша «Оқу топтары бойынша зертханалық жұмыс сағаттарын бөлу» (6.4-кесте) динамикалық кестені (кросс-кестені) алу қажет.

Осы кестеде оқытушының тегі, аты және әкесінің аты бағандағы ұяшықтарда жазылуы тиіс.

«1-топ»... «N тобы» бағандарының саны кестедегі «Жүктеме» кестесіндегі топтардың санына сәйкес келуі керек, ал әрбір бағанның қолы топтың атауына сәйкес келуі керек. Осылайша, бастапқы және қажетті кестелердің құрылымы арасындағы айырмашылық келесіден тұрады: Екі кестедегі Аты-жөні жолы бірдей мақсатқа ие. Осы жолдың мағынасы кросс-кестеде жол тақырыбы деп аталады.

6.4-кесте

Оқу топтары бойынша зертханалық жұмыс сағаттарын бөлу

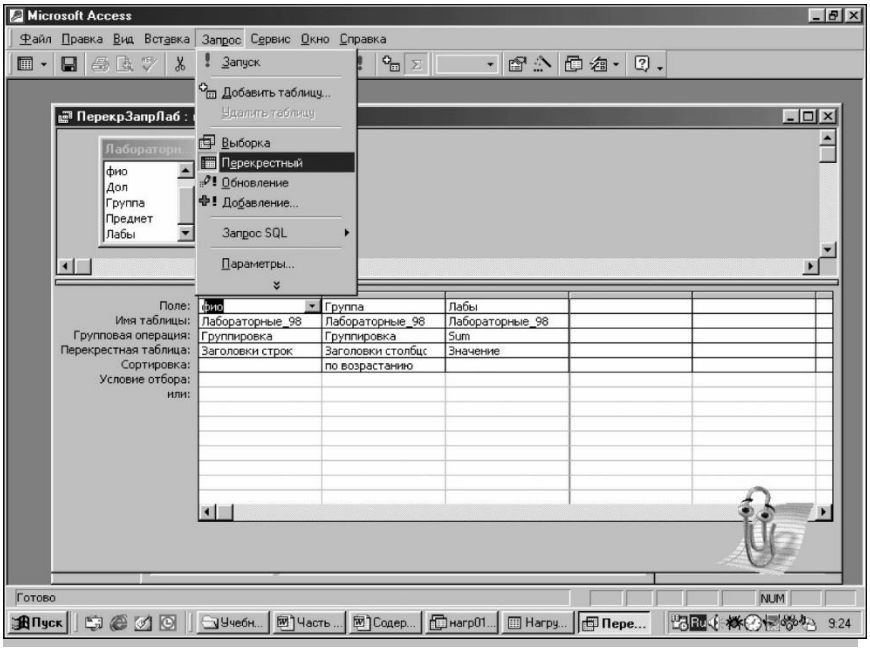
Аты-жөні	1-топ	2-топ		N-тобы
Сидоров С. С.	10	20		30
Петров П. П.	0	0		10

Кесте-кросс бағандарының атауы «Жүктеу» кестесіндегі *Топ* жолының мәндері болып табылады. Баған ұяшықтарда *Лаба* жолындағы әрбір оқытушыға мән берілетін сағат (сағат саны) жазылуы керек. «Жүктеме» кестесіндегі әрбір оқытушыға арналған жазбалардың саны оқып жатқан пәндердің санына сәйкес келетінін ескере отырып, кросс-кесте ұяшықтарына барлық пәндер бойынша барлық топтарда өткізілген зертханалық жаттығулардың жалпы санын жазу қажет.

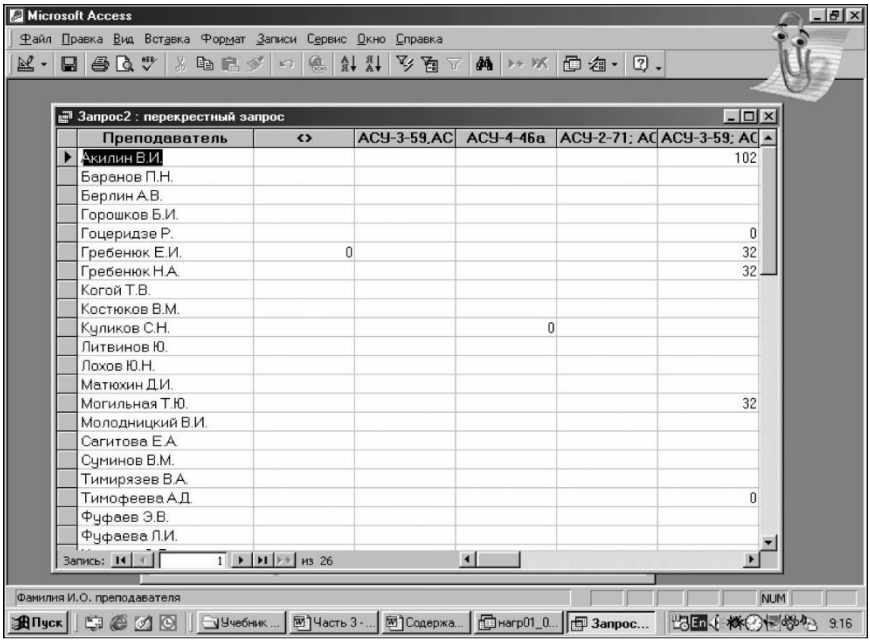
Қиылысқан сұраныста құру кезеңділігі іс жүзінде жоғарыда сипатталған сұраныстарды құрылымдау технологиясымен бірдей. Осындай сұранысты жобалауды бастау үшін оны жасаудың режимін таңдау керек, мысалы, *Құрастырушы* немесе *Қиылысқан сұраныс*.

Құрастырушы режимін таңдаған кезде, *Сұраныс* мәзірінде *Қиылысқан* батырмасын басыңыз; бұл жағдайда *Сұраныс Құрастырушы* терезесі 6.14-суретте көрсетілген түрді қабылдайды.

Қиылысқан сұраныста үш өріс болуы тиіс. Бір өріс жол тақырыбын, екінші өріс баған тақырыптарын анықтайды, ал үшінші өріс бастапқы кестенің тиісті өрісінің мәнін қамтуы керек. Жоғарыда келтірілген мысалда, *Топтау* параметрі *Құрастырушы* жолындағы «Топтық операция» жолындағы бірінші екі өріс үшін таңдалады, ал үшінші Лаба жолы үшін *Sum* (Сомалау) параметрі берілген.



6.14-сурет. Кросс-сұранысты құрылымдау терезесі



6.15-сурет. Қиылысқан сұранысты орындау нәтижесі

6.15-суреттеосы сұраныстың нәтижесі ұсынылған. Сұраныстағы жолдардың саны *Аты-жөні* өріс атауының мәндерінің санына тең. «Жүктеме» алғашқы кестесінде бір оқытушы үшін жазбалардың саны 1-ден асуы мүмкін (оқылатын пәндердің санына байланысты). Топтық операцияларды орындау кезінде барлық жазбаларды *Топтастыру* оқытушылардың атаулары бойынша топтастырылады және бір жолда қалыптасады. Сол сияқты, Топтық жолмен әрекет орындалады.

6.3. Сұраныс көмегімен есеп айырысуларды автоматтандыру

ACCESS ДҚБЖ маңызды ерекшеліктерінің бірі – есептеудің әр түрлі түрлерін автоматтандыру мүмкіндігі. Мысалы, шетел валютасындағы ақшалардың баламалары үшін тауарлардың бағасын рубльдермен қайта есептеу үдерісі. Мұндай есеп айырысу кәсіпорындардың сауда және қаржылық қызметінде қажет. Сұраныстарды қолдану арқылы есептеулер процесі сұранысты құрылымдау кезінде арнайы есептік өрісті құруға негізделеді. Осы әдіс бастапқы кестеде есептелетін өрісті көзделмеген жағдайларда пайдаланылуы керек.

Мұндай өріс сұранысты *Құрастырушы* режимінде құру кезінде жасалуы мүмкін. Ол үшін келесі әрекеттерді орындау қажет:

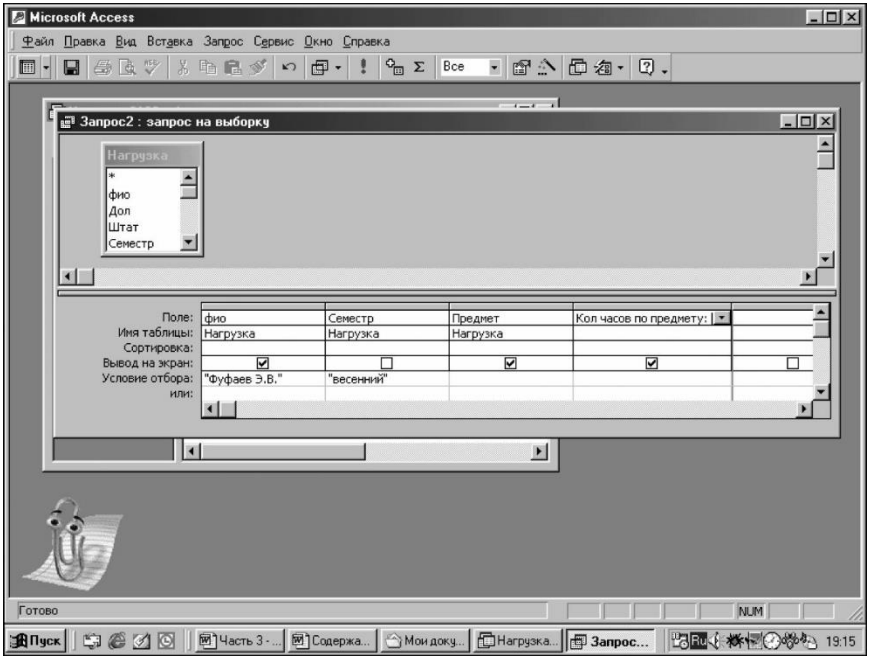
• *Құрастырушы* режимінде сұраныс жасау;

• меңзерді *Өріс* жолының соңғы ұяшығына орналастыру және тінтуірдің оң жағын басу арқылы мәтінмәндік мәзірді белсендіру;

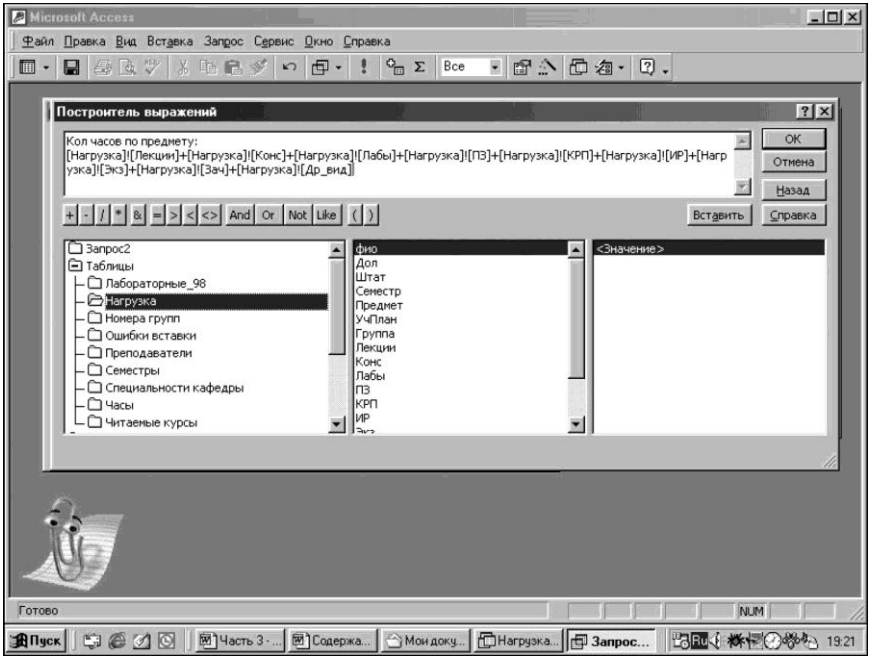
• пайда болған терезеде *Салу* командасын белсендіру. Осы команданы орындау нәтижесінде *Сөйлем құрау* терезесі ашылады, оның нұсқаулығын орындай отырып есеп айырысу сөйлемі жасалады.

Мысалы, оқытушының көктемгі семестрдегі пәндер бойынша жоспарланған жүктемесін есептеу керек.

6.16-суретте осындай сұраныс көрсетіледі. Бұл сұраныста кестеден тек үш өріс таңдалған: *Аты-жөні, Семестр, Пән*. Меңзерді *Құрастырушының* төртінші бағанының *«Өріс»* жолына қою үшін есептелетін өрістің атауы беріледі: «Пән бойынша сағаттар саны». Содан кейін *Сөйлем құрау* шақырылады (6.17-сурет), онда барлық сандар өрістерінің мәндерін сомалайтын есептеу формуласы жинақталады. Бұл формула сұраныстың есептеу өрісіне орналастырылды.



6.16-сурет.Есептеу өрісімен сұраныс *Құрастырушысы*



6.17-сурет. Есеп айрысу өрісін құру үшін *Сөйлем құрау* терезесі

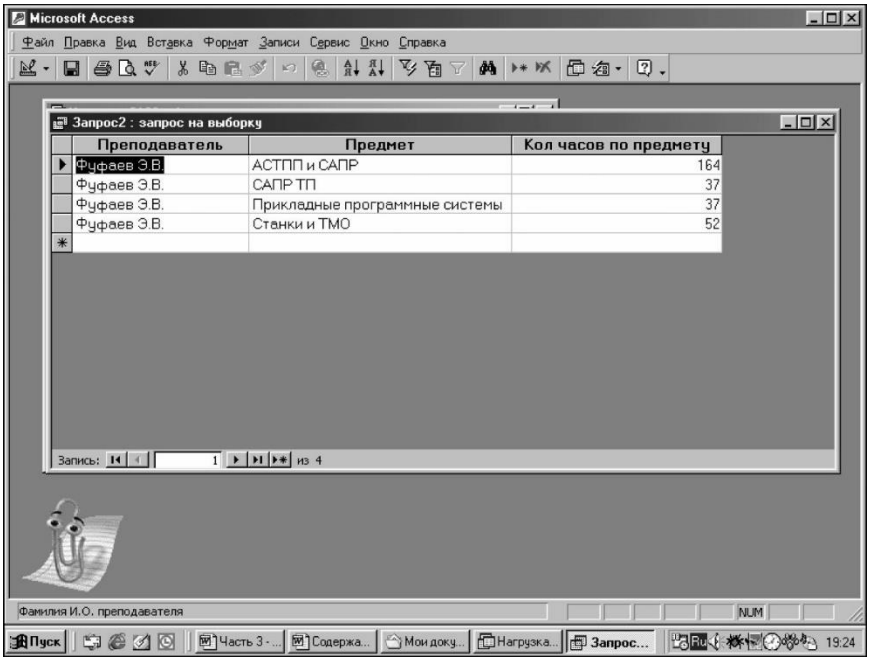
6.18-суретте сұранысты орындау нәтижесі берілген.

Осылайша, сұранысты орындау нәтижесінде нақты оқтыушы жүргізетін әрбір пән бойынша сағаттарының жинақ санының есебі орындалады. Экранға шығаратын динамикалық кестеде *Семестр* өрісінің жоқ екеніне назар аударыңыз. *Сұраныс құрастырушы* терезесінде (6.16-суретті қараңыз) осы өрістің экранына шығарудан бас тарту орындалды.

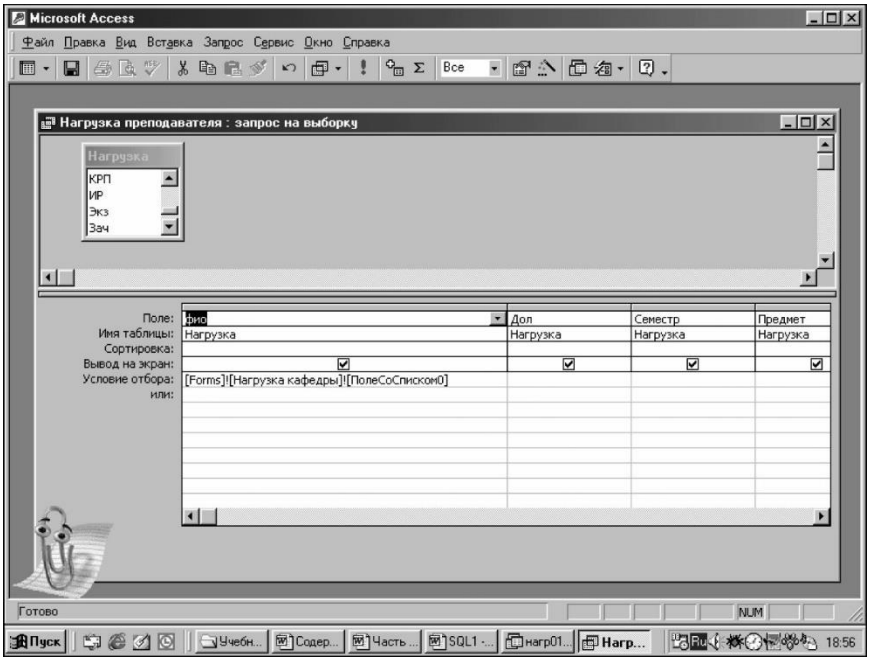
Алдыңғы бөлімдерде *Сұраныс құрастырушысын* қолдана отырып, әртүрлі сұраныстарды жасау технологиясы қарастырылған болатын. Дегенмен, кез-келген сұраныссұранысты құрылымдау процесінде автоматты түрде жасалатын SQL-бағдарламасымен орындалады. Бұл бағдарламаның не екенін түсіну үшін, *Құрастырушы* режимінде емес, SQL режимінде сұранысты қарау жеткілікті.

Мысал ретінде, *нәтижелер бойынша және мөлшерлеме бойынша* есептелетін есеп айырысу өрістерімен оқытушының жүктемесін қалыптастыру сұранысын қарастырайық (6.19, 6.20-сур.). 6.19-суретте сұраныстың бөлігіформадан *Аты-жөні* өрісінің мәнінен деректерді іріктеу шартымен көрсетіледі, ал 6.20-сур.есеп айырылысатын өрістермен сұрансытың бөлігі көрсетіледі.

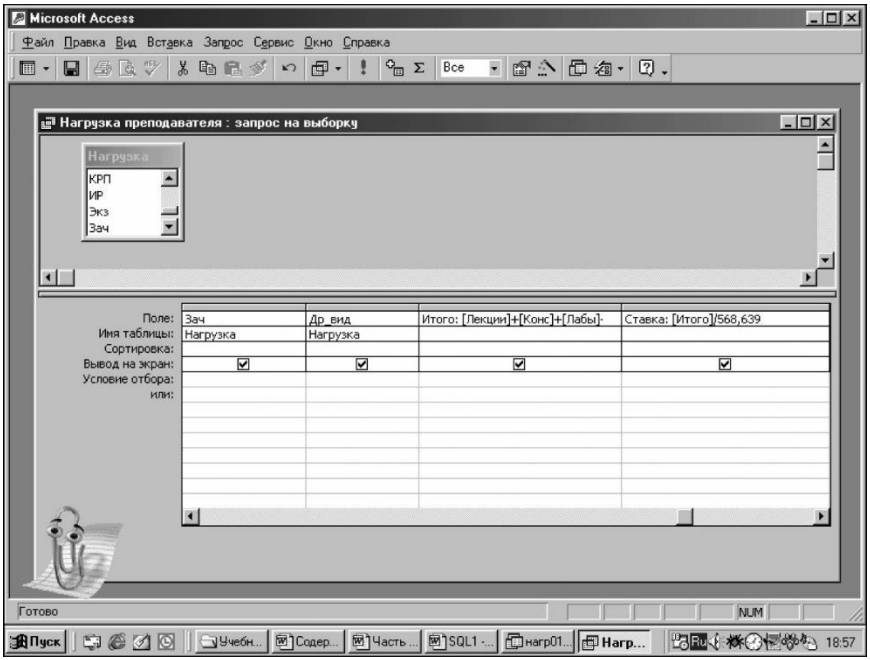
Егер осы сұранысты SQL режимінде қарайтын болсақ, SQL бағдарламасының мәтіні *Сұраныс Құрастырушы* терезесінде пайда болады (6.21-сурет).



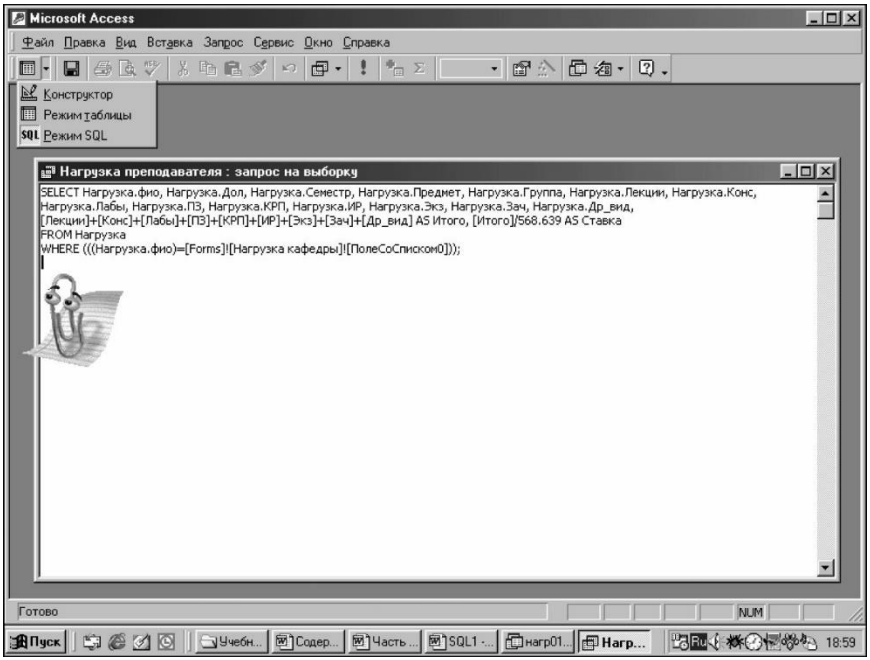
6.18-сурет. Сұранысты орындау нәтижесі



6.19-сурет. Іріктеуге І-ші сұраныстың фрагменті



6.20-сур. Таңдауға берілетін 2-ші сұраныстың фрагменті



6.21-сурет. SQL режиміндегі сұраныс *Құрастырушысының* терезесі

SQL тіліндегі сұранысты сипаттау құрылымын келесі мысалда қарастырайық.

SELECTЖүктеме. Аты-жөні, Жүктеме.үлес, Жүктеме. Семестр, Жүктеме. Пән, Жүктеме. Топ, Жүктеме. дәріс, Жүктеме. Констр., Жүктеме. Лабы, Жүктеме. ПЗ, Жүктеме. КРП, Жүктеме. ИР, Жүктеме.Емтихан, Жүктеме.Сынақ, Жүктеме. Басқа түрлері, [Дәрістер] + [Конс] + + [Лабы] + [ПЗ] + [КРП] + [ИР] + [Емтих] + [Сынақ] + [басқа түрлері] AS Жиыны, [Жиыны]/5б8.639 AS Мөлшерлеме

FROM Жүктеме.
WHERE (Жүктеме. Аты-жөні) = [Forms]![Кафедра жүктемесі]![Тізіммен өріс]));

Нұсқаулық (оператор) SELECT тіл ядросы болып табылады. Ол дерекқор кестесінен өрістерді таңдау үшін қолданылады. Аталған мысалда сұранысқа енгізілген «Жүктеме» кестесінің барлық өрістері тізімделген.

FROM сөйлемі нұсқаулықтың бөлігі болып табылады және сұраныс деректерінің (кесте немесе сұраныс) көзін анықтау үшін қызмет етеді. Бұл жағдайда ол «Жүктеме» кестесі.

WHERE сөйлемі сұранысты орындау кезінде деректерді таңдау шарттарын анықтайды. Аталған сөйлемде деректерді іріктеу шарты *Кафедра жүктемесінің Тізіммен өрісіне* енгізілген мағынасына тең, Аты-жөні өрісінің мағынасы болып табылатынын көрсетіледі.

Тұтастай алғанда, SELECT нұсқаулығының синтаксисі келесі жолмен сипатталуы мүмкін.

SELECT [ALL|] (Кестедегі немесе сұраныстағы өрістердің тізімі)
FROM (Сұраныстардың құрылуы негізделген кестелердің немесе сұраныстардың тізімі)
[WHERE (Деректерді таңдау талаптары)]
[GROUP BY (Сұранысты орындау нәтижесінің шығарылатын өрістерінің тізімі]
[HAVING (Сұраныстағы деректерді топтау шарттары]
[ORDER BY (Сұраныстағы деректердің шығарылуы реттелетін өрістер тізімі]

Жоғарыдағы құрылымда SELECT ALL нұсқаулығы - шешуші сөз, яғни жазбалардың нәтиже жиынына сұраныс талаптарын қанағаттандыратын кестенің немесе сұраныстың барлық жазбаларын қамтитынын білдіреді.

Шешуші сөздер сұранымда болмауы мүмкін.

Бақылау сұрақтары

1. *Кесте құрастырушы* қандай ақпарат блоктарынан тұрады және олар қандай тәртіпте толтырылуы керек?
2. Өріс атауы қанша таңбадан тұруы мүмкін?
3. Өріс атауы бос орындардан бастала ала ма?

1. Өріс атауын белгілегенде қандай белгілерге рұқсат етілмейді?
2. Мәтіндік деректер түрінің МЕМО-дан айырмашылығы қандай?
3. Сандық деректер түрі мен ақшалық түрінің арасындағы айырмашылық қандай?
4. OLE деректерін қандай жағдайда пайдалану керек?
5. *Гиперсілтеме* деректерінің түрін қай уақытта қолдану керек?
6. Қандай жағдайларда *Шешуші өріс* сипаты беріледі?
7. Шешуші өріс ДҚ кестесіндегі деректер мәндерінің қайталануы мүмкін бе?
8. Қандай жағдайларда өріске *Міндетті* сипаттамасы беріледі?
9. Қандай кестелер негізгі деп аталады, ал бағынышты деп қандайлары аталады?
10. «Деректердің тұтастығын қамтамасыз ету» терминінің мәні қандай?
11. ACCESS ДҚБЖ-де әзірленген сұраныстардың мақсаты мен түрлерін атаңыз.
12. Тұрақты сұраныстың параметрлік сұраныстан айырмашылығы қандай?
13. Қиылысқан сұраныстың мақсаты қандай?
14. Орындалатын әрекеттер бойынша сұраныстардың түрлерін атаңыз.
15. Мәтіндік өрістерде деректерді таңдау үшін жағдайларды енгізу ережелерін атаңыз.
16. Деректерді іріктеу шарттарымен, AND және OR байланысты қатынастармен арасындағы айырмашылық қандай?
17. Келесі функциялардың мақсаты қандай: Day; Month; Year; Date()?
18. Сұраныстарда есептелетін өріс қандай жағдайларда пайда болады?
19. Өрнек құрастырушыны пайдаланып сұраныста есептелетін өрісті жасағандағы әрекеттердің реті қандай?

7 тарау

ДЕРЕКТЕРМЕН ЖҰМЫСТЫ АВТОМАТТАНДЫРУ

7.1. Нысандар арқылы деректерді енгізу және талдау

Біз 6 тарауда кез-келген дерекқордың негізі - кестелер мен сұратуларды жасап шығарудың негізгі әдістемелік тәсілдерін қарастырдық.

Алайда қазіргі заманғы қолданбалы бағдарламаларды немесе пайдаланушылық қосымшаларды жасау пайдаланушының достық интерфейсін жасап шығаруды, яғни пайдаланушы мен компьютер арасындағы сұхбатты ұйымдастырудың тиімді тәсілдерін жасап шығаруды талап етеді.

Сұхбатты ұйымдастыру тәсілдерінің бірі нысандарды жасап шығару болып табылады.

Microsoft ACCESS жүйесі қосымшаны жасап шығарушыға төмендегі мақсатты сұхбаттық нысандарды жасаудың қуатты құралдарын ұсынады:

- деректерді кестеге енгізу үшін;
- ақпаратты өңдеу шарттарын сұраныстарға енгізу;
- пайдаланушы интерфейсін ұйымдастыру.

Деректерді кестеге енгізу нысандары оператордың қателесу ықтималдығына жол бермейтіндей, ақпаратты енгізу рәсімін ұйымдастыруға арналған. Бұдан басқа, мұндай нысандар кестедегі деректерді талдауға арналуы мүмкін.

Ақпаратты өңдеу шарттарын сұратуларға енгізу нысандарының мәні баламалы және бұдан басқа, SQL тілін қолданбай сұраныстарды құру мүмкіндігін береді.

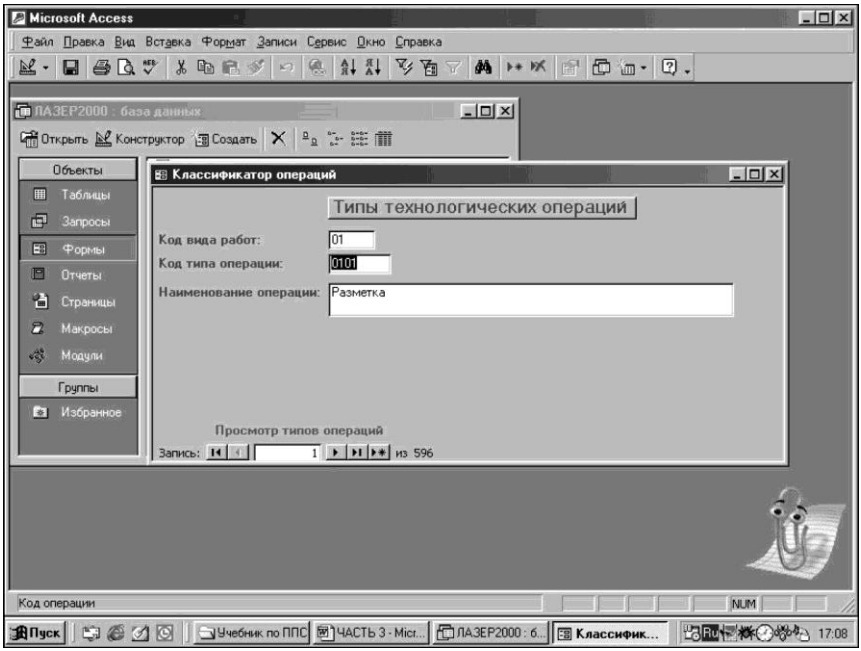
Пайдаланушы интерфейсін ұйымдастыруға арналған нысандар жасап шығарылған қолданбалы бағдарламаны тиімді рәсімдеуге арналған. Бұл тағайындау нысандары, мысалы, бет басы нысандары, мәзір нысандары, пернелі нысандар және т.б.

Деректерді кестеге енгізу нысандарын жасап шығару технологиясы. Деректер енгізу нысандары кестемен жұмыс істеу кезінде пайдаланушының қолайлы және интуитивті түсінікті интерфейсі болып табылады.

Деректерді енгізу нысандары мыналарды қамтамасыз етеді:

- деректерді енгізу және қосу;
- кез-келген жазбаны қарау;
- деректерді түзету.

7.1 - суретте деректерді САПР ТП «ЛАЗЕР 2000»«Операцияларды жіктегіш» кестесіне енгізу нысаны көрсетілген..



7.1 - сурет. Деректерді САПР ТП «ЛАЗЕР 2000»«Операцияларды жіктегіш» кестесіне енгізу нысаны

Деректерді кестеге енгізу нысандары мына реттілікпен құрылады:

• *Дерекқор терезесінде Нысан* объектісін ерекшелеу (активтендіру);

• *Құру командасын таңдау;*

• ашылған *Жаңа нысан* сұхбат терезесінде нысан құрылатын кестені (тізімнен) таңдау керек;

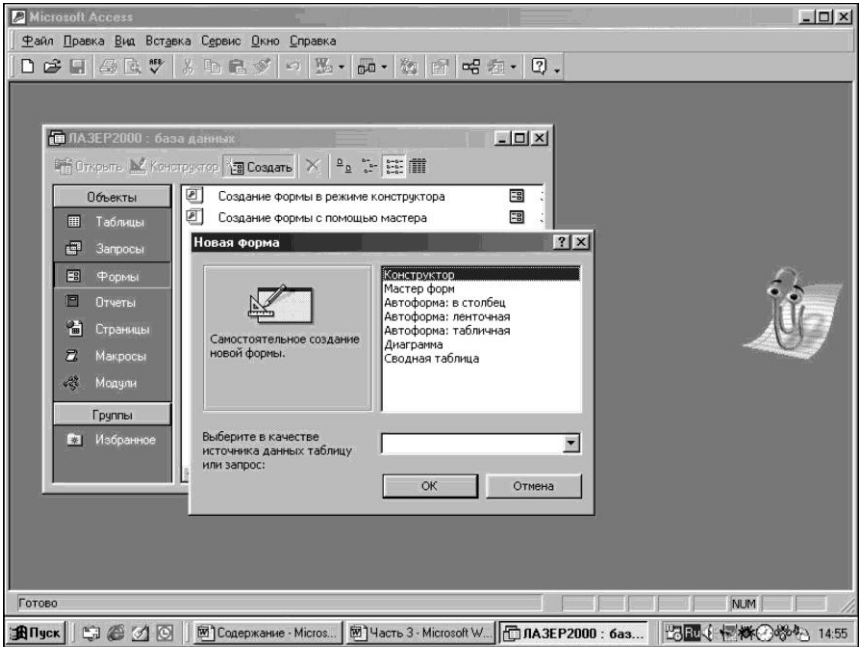
• нысан құру тәсілін таңдау.

ACCESSжүйесі дерекқорды жасап шығарушыға нысандарды жобалаудың жеті тәсілін ұсынады (7.2-сур).

• Конструктор;
• Нысандар шебері;
• Автонысан: бағанмен;
• Автонысан: таспалы;
• Автонысан: кестелі;
• Диаграмма;
• Жиынтық кесте.

Біз деректер енгізу нысанын жасау үшін «Автонысан: бағанмен» нысандарды автоматты жобалау әдісін қолдануды ұсынамыз.

Осы нысандарды жасау тәсілін пайдаланушы интерфейсін жасау кезінде кеңінен тараған тәсіл деп есептеуге болады.



7.2 - сурет. *Нысандарды жасау тәсілін таңдау* терезесі

Нысандарды жасаудың осындай тәсілін таңдау кезінде кестенің барлық өрістері бір бағанда орналасады - әрбір өріс бір жолда болады.

Өрістің қолтаңбасы кестеде белгіленген атқа сәйкес келеді. Осы парақта (экранда) бір жазбаның деректерін енгізуге арналған өрістер орналасады. Осындай тәсілмен алынған нысанды *Құрастырушы* режимін ашып, толық өңдеуге болады. (*Нысандар Құрастырушысы* сондай-ақ нысандарды өздігінен жасау үшін қолданылады).

“Кестелі” нысаны кесте нысанына сәйкес келеді. Бір парақта монитор экранына сиятын жазба ұсынылған (7.3 - сур).

Әр жазба бір жолға орналасады.

“Диаграмма” нысанын кестедегі жазбаларды диаграммалар немесе графикалар түрінде қарау үшін жасап шығару ұсынылады.

1 ая	2 ая	3 ая	4 ая

7.3- сурет. Кестелі нысанның түрі

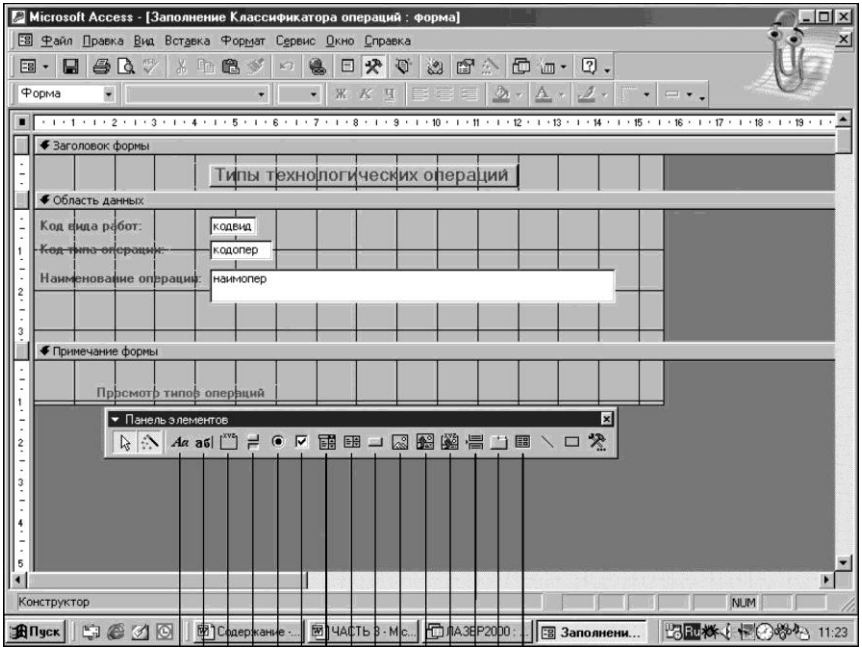
Мұндай нысандар фирманың экономикалық қызметінің нәтижелерінің талдауын немесе ғылыми эксперимент нәтижелерін өңдеу үшін қажет болатыны анық.

Пайдаланушыға осы режимде графиктер мен диаграммалардың әр түрі ұсынылады.

“Жиынтық кесте” нысанында бір уақытта екі байланысты кестеден ақпарат ұсынылуы мүмкін. Бір кесте басты, екіншісі - бағынышты болып есептеледі. Бағынышты кесте басты кестенің нысанына кіріктірілген.

7.4 - суретте *Нысандар Құрастырушысы терезесі көрсетілген. Нысандар Құрастырушысы келесі блоктардан тұрады*:

- нысанның тақырыпшасы,
- деректер саласы,
- нысанның ескертпесі.



Жазу

Өріс

Ауыстырып қосқыштар

тобы

Сөндіргіш

Қосқыш

Жалауша

Тізіммен өріс

Бағынышты нысан (есеп)

Қосымша беттер жинағы Беттің үзілуі

Объекттің қосылған жиегі Объекттің бос жиегі

Сурет Перне Тізім

7.4- сурет. *Нысандар* құрастырушысы *терезесі*

Осы блоктардың тағайындалуы олардың атауларымен айқындалады.

Әр команданың тағайындалуын атап өтейік.

Жазу.Бұл команда *Нысандар Құрастырушының* кез-келген блогында жазу (мәтін) енгізуге арналған. Мәтінді енгізу үшін:

• *Жазу* пернесіне басып (тінтуірдің сол жақ пернесі), тінтуірдің пернесін басылған күйінде ұстап меңзерді енгізілетін мәтіннің басына орналастыру керек;

• Тінтуірдің пернесін жіберіп, мәтінді енгізу керек. Мәтінді енгізу және оны рәсімдеу технологиясы толығымен Word редакторындағы мәтінмен жұмыс істеу технологиясына баламалы.

Ескертпе.Жазу мәтінін бірнеше жолға орналастыру үшін мәтіннің бірінші жолының соңында күймешені қайтару символын енгізу үшін [CTRL] + + [ENTER] пернелеріне басыңыз. Бұл жағдайда мәтінді енгізу шамасына қарай кейінгі жолдар автоматты түрде орын ауыстырады, ал жазуның ең жоғарғы ені мәтіннің бірінші жолының ұзындығымен айқындалады.

Перне.Бұл нысандармен жұмыс істеу кезінде басқару әрекеттерін құруға арналған командалар жинағын құрайтын басқару элементі.

Өріс, Тізімі бар өріс, Тізім пернелері деректерді енгізудің тиісті өрістерін құруға арналған. Деректерді сұратуларға енгізу нысандарын жобалау кезінде оларды пайдалану қажет болады.

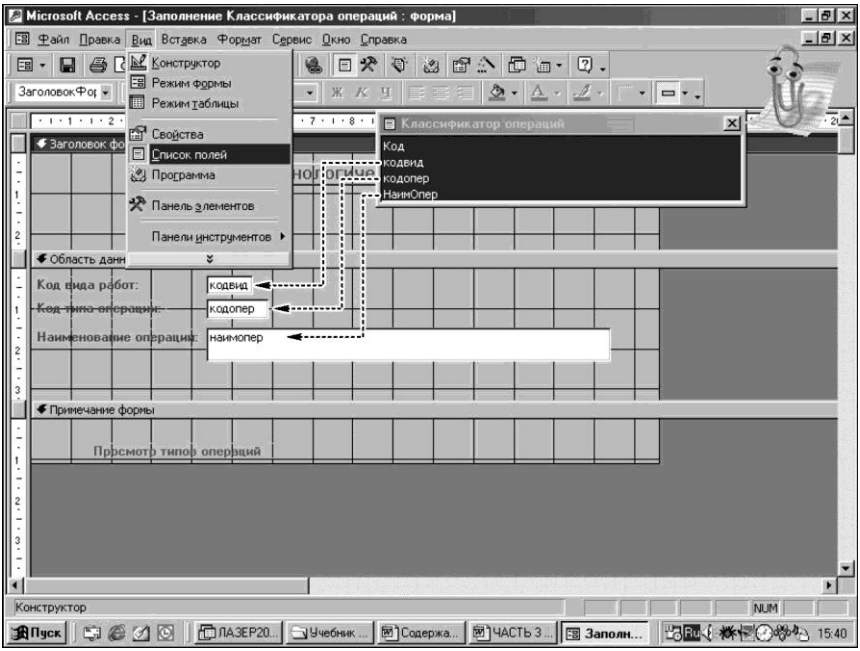
Егер деректерді кестеге енгізу нысаны бір автоматты әдіспен жасап шығарылса, онда аялар автоматты түрде *Нысандар Құрастырушының* деректер саласына орналастырылады және пайдаланушыға осы командаларға жүгіну қажет болмайды. .

Деректерді кестеге енгізу нысандарын өздігінен конструкциялау кезінде өрістер өрісті кестенің өрістер тізімінен көшіру арқылы орналастырылады. Ол үшін *Құрастырушы* режимінде мәзірдің *Түрі* командасын активтендіріп, *Өрістер тізімі* командасын таңдап, пайда болған кестенің өрістері тізімінен өрістерді *Кестелер Құрастырушының* деректер саласына орналастыра отырып, ретімен көшіру керек (7.5 - сур). Өрістерді кқшіру және орналастыру реті болжалды деректер енгізу ретіне сәйкес келу керек.

Ауыстырып қосқыштар тобы, Сөндіргіш, Қосқыш, Жалауша пернелері деректердің логикалық типті өрістерге енгізілуін ұйымдастыруға арналған.

Сурет.Бұл команда суреттерді нысанға енгізуге арналған. Суреттерді енгізу технологиясы суреттерді Word құжатына енгізу технологиясына баламалы.

Объекттің бос жиегі. БұлOLE-объекттер типті кесте өрістерінің деректері көрсетілетін терезе.



7.5 - сурет. Нысанды өздігінен жобалау

Кестелерді автоматты түрде жасау кезінде нысандағы терезелер де автоматты жасалады.

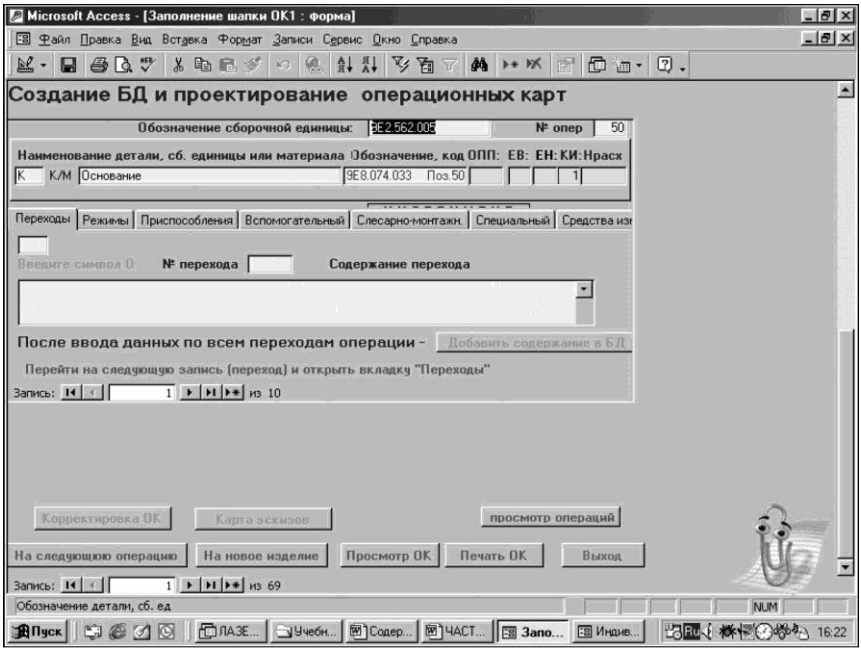
Объекттің қосылған жиегі. Бұл басқа файлда немесе басқа ДҚ-да орналасқан OLE-объектті орналастыруға болатын терезе.

Беттің үзілуі. Егер деректер енгізу аялары бір бетке (дисплейдің экранына) сыймаса, осы команда қолданылады.

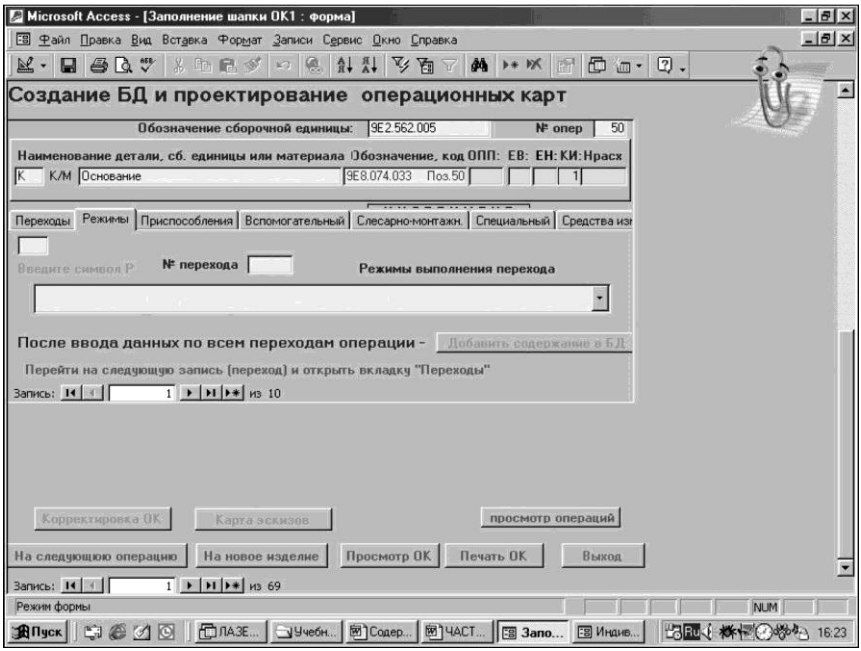
Қосымша беттер жинағы. Егер деректерді енгізу өрістері бір бетке (дисплей экранына) сыймаса, осы команда қолданылады. Қосымша беттер жинағын пайдалану кезінде өрістерді қандай да бір белгілері бойынша топтастыруды және әрбір топ үшін тиісті қосымша бетті жасауды ұсынамыз. Нысанды конструкциялау барысында аяларды қосымша бетте орналастыру технологиясы аяны тізімнен көшіруге негізделген.

7.6 суретте *Өтпелер* ашық қосымша бетімен, ал 7.7 суретте *Режимдер* ашық қосымша бетімен деректер енгізу нысаны көрсетілген.

Бағынышты нысан (есеп). Бұл команда құрамдас нысандарды жасап шығару кезінде қолданылады. Құрамдас нысандар, әдетте "біреуі көпшілікке" қатынастарымен байланысты кестелер үшін жасап шығарылады. Бұл жағдайда бір кесте басты, ал екіншісі - бағынышты болып табылады.



7.6 - сурет. *Өтпелер* ашық қосымша беті бар нысан



7.7 - сурет. *Режимдер* ашық қосалқы беті бар нысан

Осы сияқты деректерді енгізудің құрамдас нысандарын жобалау кезінде нысанның біреуін басты, екіншісін - бағынышты деп атаймыз. Құрамдас нысандарды жасап шығару сұлбасын келесі әрекеттер ретімен ұсынуға болады:

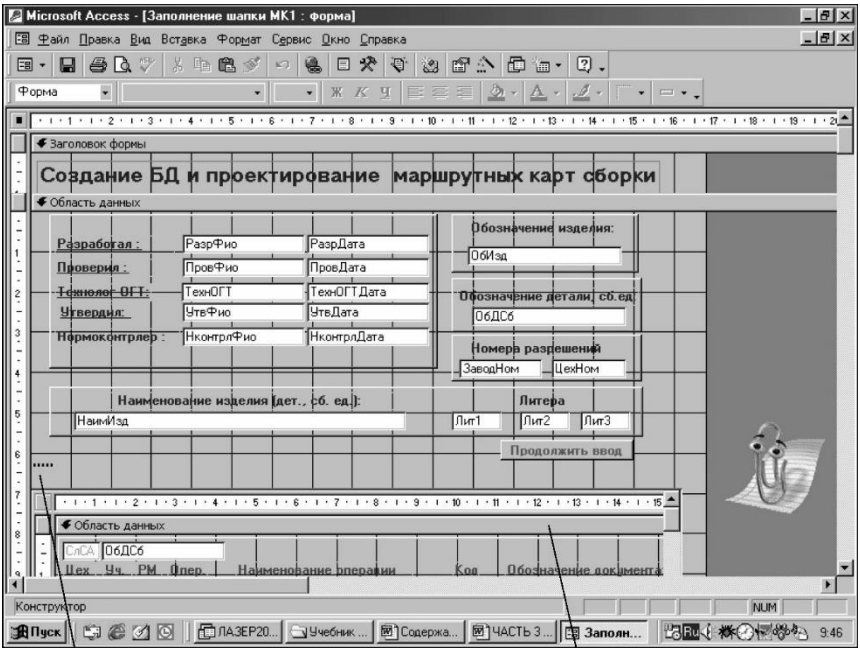
- деректерді бағынышты кестеге енгізу нысанын жасап шығару;
- *Бағынышты нысан* командасын (пернесін) пайдалана отырып, бағынышты нысанды енгізуге арналған саланы қарастыра отырып, басты кестеге деректер енгізу нысанын жасап шығару.

Төменде САПР «ЛАЗЕР 2000» технологиялық құрастыру үрдістерінің маршруттық карталарын автоматты жобалау үшін дерекқорды жасау кезінде ақпаратты енгізудің құрамдас нысандарын конструкциялау үлгілері келтірілген (*Құрастырушы* режимінде*).*

Басты нысан екі беттен тұрады (7.6, 7.7 - суретке қараңыз). Бірінші бетте деректерді басты кестеге енгізу өрістері, ал екінші бетте - деректерді бағышынты кестеге енгізу өрістері орналастырылған.

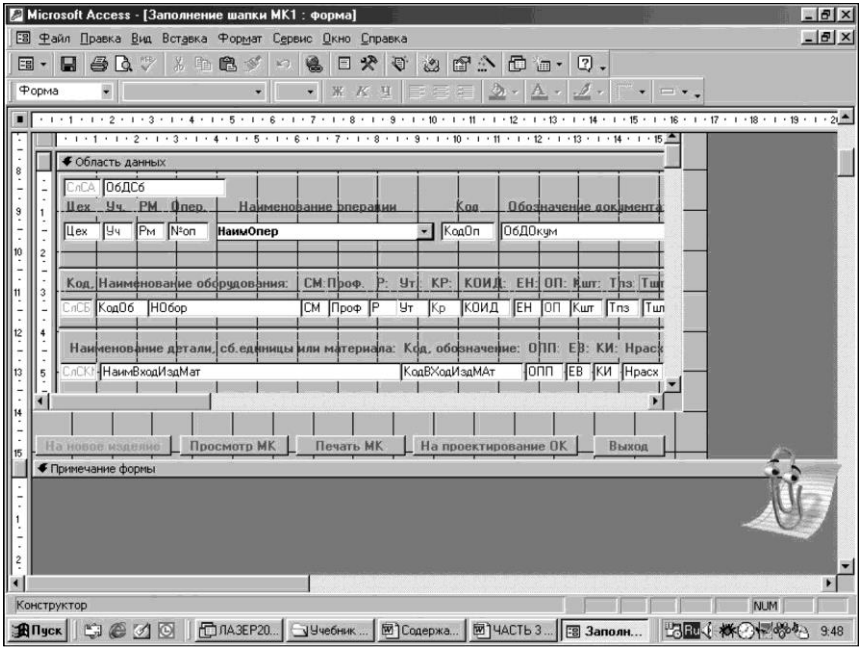
7.8, 7.9 - суретте деректерді сәйкесінше басты және бағынышты кестеге енгізу нысандарының конструкциялары келтірілген.

Деректерді суратуларға енгізу нысандарын жасап шығару технологиясы. Параметрлі сұраныстарды және оларға байланысты деректерді іріктеу шарттарын енгізу нысандарын жасау технологиясын қарастырамыз.



Беттің үзілуін белгілеу **Бағынышты нысан**

7.8- сурет. Құрамдас нысанның бірінші беті



7.9 - сур. Құрамдас нысанның екінші беті

Деректерді сұраныстарға енгізу үшін арнайы нысандарды жасап шығару қажеттігі келесі факторлармен шартталған:

• деректерді іріктеу шарттарын енгізу қателігіне жол бермеу немесе болдырмау;

• деректерді енгізу шарттарының жиі өзгермелі мәндері кесінде сұраныс жасау қажеттігі;

• клиент-сервер архитектурасы бойынша ұйымдастырылған желілік дерекқорларды жасап шығару.

Осындай нысандар мен сұраныстарды жасап шығару технологиясы толығымен 7.3 және 7.5 бөлімшелерінде баяндалған әдістерге сәйкес келеді. Ерекшелігі тек оларды жобалау тәртібінде. Деректерді іріктеу шарттарын енгізу нысандарымен сұраныстарды мына ретпен жобалау керек.

1. Деректерді енгізу шарттарын енгізбей сұранысты жасап шығару.
2. Деректерді іріктеу шарттарын енгізу нысанын жасап шығару.
3. *Құрастырушы режимінде сұранысты ашу.*
4. Тиісті ая үшін “Іріктеу шарты” жолының ұяшығына меңзерді орнату.
5. Нысанның тиісті аясының мәнімен деректерді іріктеу шарттарының байланысын орнататын тіркес құру.

Мысал ретінде оқытушылардың жүктемесін жасау және талдау үшін дерекқорды қарастырайық.

Дерекқормен жұмыс істеу барысында оқытушының тегінетәуелді “Жүктеме” кестесінен жазбаларды жиі таңдауға тура келеді. Осындай таңдау үшін тиісті сұраныс жасап шығарылды (7.10 - сур).

Аты-жөні аясының ұяшығына сұраныс жасау үшін “Іріктеу шарттары” жолында оқытушының аты-жөнін енгізу керек.

Тегін енгізу үшін тиісті нысан жасап шығарылған (7.11 - сур).

Осы нысанда оқытушылардың тегін енгізуге арналған тізіммен арнайы өріс көзделген.

Нысанды жасап шығарғаннан кейін Аты-жөні аясының “Іріктеу шарттары” жолында “Оқытушының жүктемесі” тиісті сұратуында келесі түрдегі тіркес құрылған:

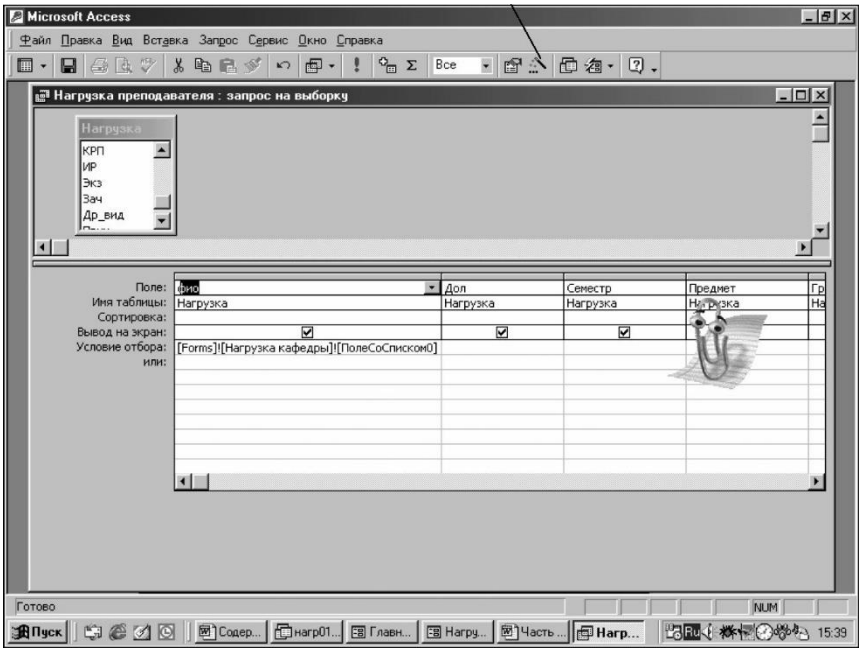
[Forms] ! [Кафедра жүктемесі] ! [Тізімі бар аяО],

мұндағы Forms — ДҚ объектісін білдіретін өзекті сөз (бұл жағдайда - нысан);

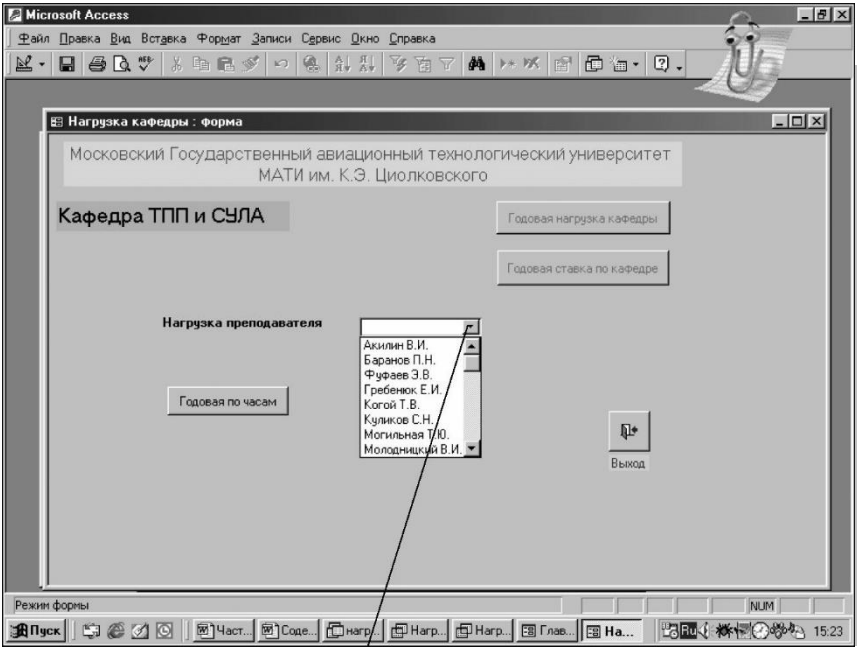
Кафедра жүктемесі — дерекқор объектісінің (нысанының) аты;

Тізімі бар өріс — нысандағы аяның аты, оның мәні “Оқытушының жүктемесі” сұратуындағы Аты-жөні өрісі үшін деректерді іріктеу шарттары болып табылады;

Тіркестерді құрушы терезесін ашу пернесі



7.10 - сурет. *Аты-жөні* өрісінің мәні бойынша деректерді таңдауға



Аты-жөнін таңдауға арналған тізімі бар ая

7.11 - сурет. Оқытушының тегін енгізу өрісі бар нысан

Жақшалар [ ] және «!» белгісі — тіркестерді құру грамматикасының элементтері.

Тіркестерді *Тіркестерді құрушы* шеберін қолдана отырып құрған жөн (7.12 - сур). Ол үшін нысанды жасап шығарғаннан кейін:

- *Құрастырушы режимінде сұраныстыашу*;
- «Іріктеу шарттары» жолының ұяшығына меңзерді орнату;
- *Тіркестерді құрушыны* ашу;
- қажетті тіркесті құру керек.

Тіркестерді құрушы терезесін ашқаннан кейін келесі әрекеттерді орындау керек:

• тиісті белгі бойынша тінтуірмен басып, ДҚ объектісін таңдау керек (осы мысалда — «Forms»). Осыдан кейін осындай типті барлық объекттің тізімі ашылады;

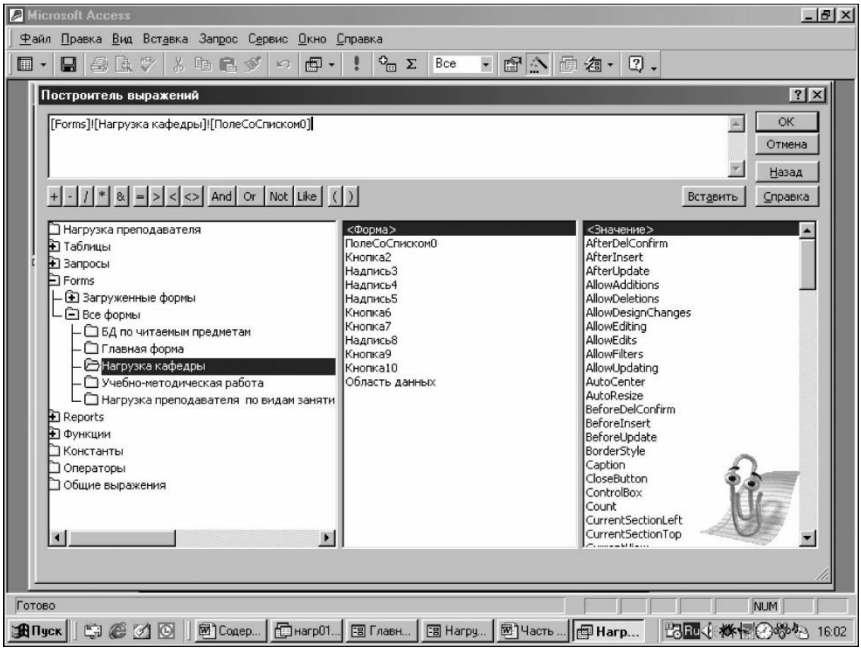
• объекттің (нысанның) аты тізімінен таңдау керек. Элементтер терезесінде ДҚ объектісін таңдау нәтижесінде барлық элементтің тізімі шығады (өрістер, қолтаңбалар, пернелер және т.б.);

• ДҚ объектісінің элементін таңдау (Тізімі бар өріс).

Осы әрекеттер нәтижесінде нысан аясында енгізілетін мәндері бар сұраныстағы деректерді іріктеу шарттарын байланыстыратын тіркес қалыптасады.

Пайдаланушы интерфейсін ұйымдастыруға арналған нысанды жасап шығару технологиясы. Кез-келген қосымшаны жасап шығарудың қорытынды міндеті қолайлы пайдаланушы интерфейсін жасау болып табылады. Пайдаланушы интерфейсін жасап шығару “фильм” сценариін жасап шығару және оны дисплейдегі “экранды нысандардың” пайда болу реті түрінде іске асыру болып табылады. ACCESS жүйесі кіріктірілген бағдарламалық құралдармен, атап айтқанда VISUAL BASIC бағдарламалау тілімен жасап шығарушыға минималды шығынмен пайдаланушы интерфейсін құру мүмкіндігін береді. Бұл ACCESS ортасында пайдаланушы интерфейсінің нысандарын жасап шығару кезінде объектті-бағдарлы бағдарламалау әдісі іске асырылған. Осы әдістің мәні жасап шығарушының объект қасиеттерін сипаттайтынында, ал бағдарлама осы қасиеттерге сәйкес экранда сипатталған объекттерді қалыптастыратынында болып табылады.

Қолда бар құрал-саймандық құралдарға қарамастан пайдаланушы интерфейсін жасап шығару өнер болып табылатыны анық. Сондықтан пайдаланушы интерфейсін құру үшін коммерциялық жүйелерді жасап шығару кезінде дизайнерлердің қызметіне жүгіну керек. Қосымша сценариі - нысандар құрамы мен реті - жасап шығарушымен нақты міндетті шешу алгоритміне сәйкес айқындалады.



7.12 - сурет. *Тіркестерді құрушы* терезесі

Пайдаланушы интерфейсін жасап шығарудың кейбір тәсілдері мен әдістерін қарастырайық.

Пайдаланушы интерфейсінің нысандарын жасап шығару үшін деректерді кестелер мен сұраныстарға енгізу нысандарын жасау технологиясын сипаттау кезінде қарастырылған әдістер мен құралдар қолданылады. Деректерді кестелер мен сұраныстарға енгізу нысандары пайдаланушы интерфейсінің бір бөлігі болып табылатынын атап өткен жөн.

Бұл жағдайда нысандарды құрудың негізгі әдісі оларды *Құрастырушы режимінде жасап шығару болып табылады*. Дерекқормен жұмысты автоматтандыру мүмкіндігін беретін басқарушы элементтеді құру технологиясын қарастырайық.

Пайдаланушы интерфейсінің нысандарын құрудың кеңінен тараған түрі пернелі нысандарды құру болып табылады. Мұндай нысандардағы басқару объектілерінің бірі пернелер болып табылады. Пернеге "Басу" қандай да бір әрекеттерді орындауға әкелуі керек (мысалы, қандай да бір нысанды немесе сұранысты ашу, есепті басып шығару, қосымшадан шығу және т.б.).

Сонымен, пайдаланушы интерфейсін жүйемен жұмыс істеу кезіндегі әрекеттер алгоритміне сәйкес келетін экранды нысандарды реті деп қарастырамыз.

Экран нысандарының кестелерді, сұраныстарды, есептерді қоса алғанда дерекқордың барлық объекттерін басқару элементтері болу керектігі анық.

Дерекқормен жұмыс істеу кезінде орындалатын әрекеттер макростар мен бағдарламалық модульдер (бағдарламалар) түрінде бағдарламалануы мүмкін. Макростар дерекқордың элементтерімен жұмысты автоматтандыратын макрокомандалар жинағы болып табылады. Модульдер - бұл VISUAL BASIC тілінде жазылған бағдарламалар (8 тарауды қараңыз).

Жасап шығарылған деректерді кестеге енгізу нысаны пайдаланушыға келесі әрекеттерді орындау мүмкіндігін береді:

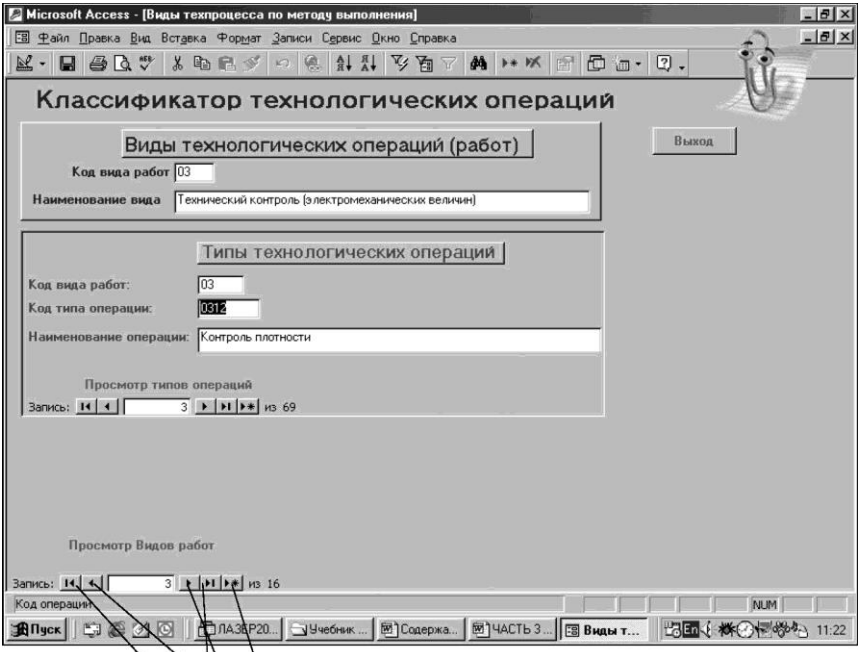
- деректерді енгізу және қосу;
- кез-келген жазбаны қарау;
- деректерді түзету.

Деректерді енгізу және қосу. Деректерді енгізудің технологиялық тәсілдерін 7.13 суретте ұсынылған нысан үлгісінде қарастырайық.

Ескертпе. Ақпаратты нысандар өрісіне енгізу технологиясы ақпаратты Word кестесіне енгізу технологиясына баламалы.

Осы нысан құрамдас болып табылады және деректерді екі байланысты кестеге енгізуге арналған: «Технологиялық операциялар түрі» және «Технологиялық операциялар типтері".

Кез-келген жазбаны қарау және деректерді түзету. Кестелермен жұмыс істеу барысында қандай да бір жазбаның мазмұнын қарау қажет болатын жағдай туындауы мүмкін, ал түзету енгізу қажет болғанда — қандай да өрістегі деректер мәнін өзгерту керек. Ол үшін:



7.13 - сур. Деректерді кестеге енгізу нысанының үлгісі

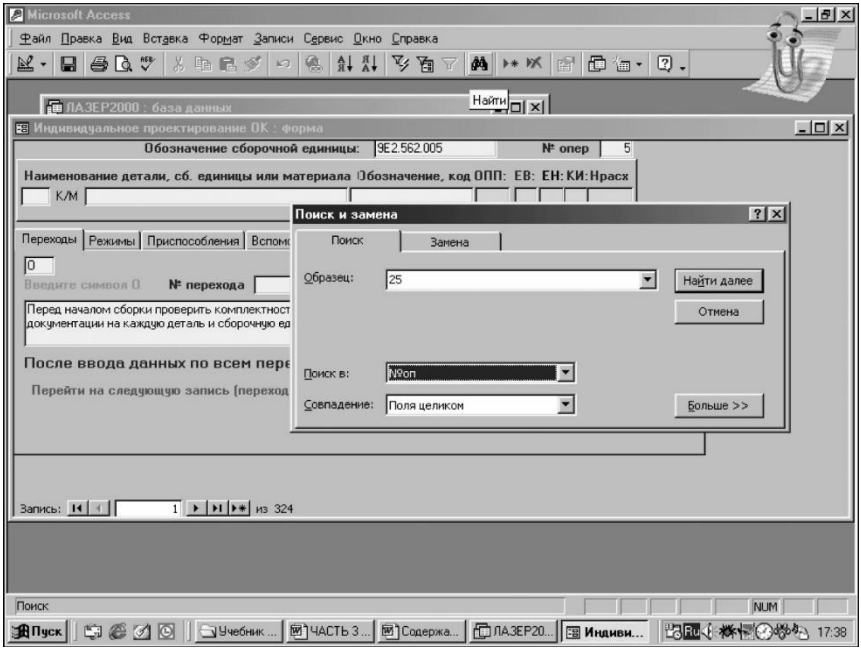
1. Кестені ашу.
2. Мәні бойынша іздеу салынатын жазба табылатын, өрістің терезесіне меңзерді орнату керек.

Ескертпе. Іздеу жасалатын өрістің атын тікелей *Іздеу және ауыстыру* сұхбат терезесінде енгізуге болады.

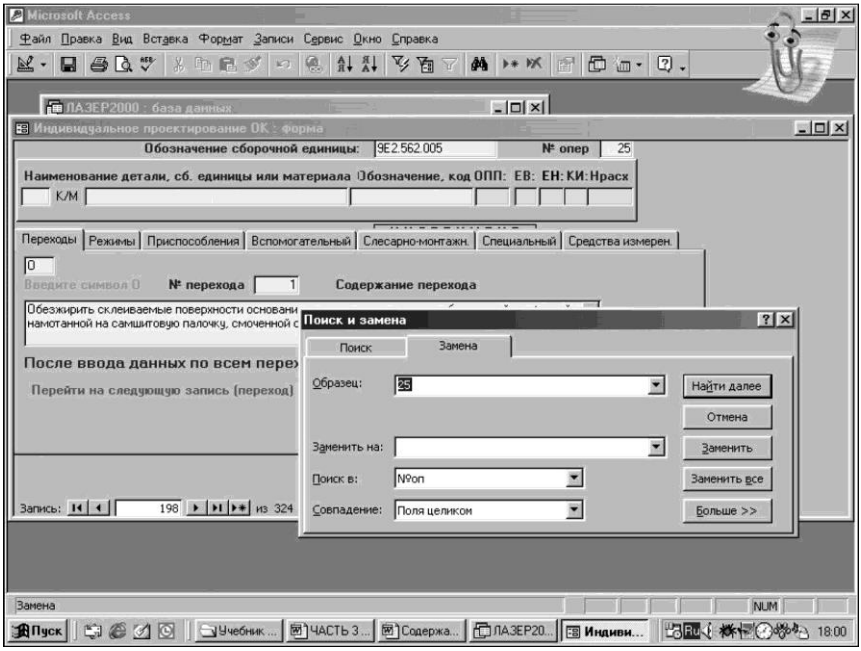
3. *Түзету* мәзірін *Табу* командасын таңдау немесе құрал-саймандар панелінде тиісті пернені белсендіру.
4. Ашылған *Іздеу және ауыстыру* сұхбат терезесінде қажетті жазбаның өрістерінің бір мәнін енгізу керек.
5. *Одан әрі табу* пернесіне басу керек.

Орындалған әрекеттер нәтижесінде деректерді енгізу нысанының терезесінде ізделетін жазбаның барлық өрістерінің мәні шығады.

7.14 суретте 9Е2.562.005 бұйымын құрастырудың технологиялық операцияларының сипаттамасын құрайтын, кестеде №25 операция үшін жазбаны іздеу мысалы көрсетілген. Нысанның деректер саласында кестені ашу кезінде №5 операцияның мазмұны көрсетілген.



7.14 - сур. ДҚ кестесінде жазбаларды іздеу *Параметрлерін таңдау* терезесі



7.15 – сурет. Жазбаны іздеу нәтижесінің шығару үлгісі

Табу командасын белсендіргеннен кейін құрал-саймандар панелінде *Іздеу және ауыстыру* сұхбат терезесі ашылды.

Сұхбат терезесінің *Үлгі-нұсқа* аясында 25-ке тең *№ оп өрісінін мәні енгізілді. Одан әрі табу пернесіне "басқаннан" кейін жүйе жазбаны іздеді және 25 операциясы үшін барлық өрісінің мәндерін шығарды* (7.15 - сур).

Деректерді түзету. *Іздеу және ауыстыру* сұхбат терезесінде аяның мәнін жаңасына өзгерту қажет болғанда *Ауыстыру* қосымша бетін ашып, *№ оп* өрісіндегі деректердің мәнін түзету немесе ауыстыру керек.

Сонымен, осы бөлімшеде біз ДҚ кестелерді жазбаларды іздеудің бір тәсілімен - бір өрістің мәні бойынша іздеу тәсілімен таныстық.

7.2. Деректерді өңдеу нәтижелерін есептер түрінде шығару

Есептер, әдетте, ақпаратты өңдеу нәтижелерін басып шығару үшін жасап шығарылады. Есептер нысандар сияқты кестелермен және сұратулармен байланысты болуы мүмкін. Бірінші жағдайда есепте бүкіл кестенің ақпараты орналастырылады, ал екіншісінде - тиісті сұраныс шарттарына сәйкес келетін жазбалар ғана орналастырылады.

Ақпаратты немесе сұраныс нәтижесін тікелей кестеден басып шығарумен салыстырғанда есептердің басымдылықтары мынадай:

• есептер деректерді жекелеген өрістер бойынша топтастыру мүмкіндігін береді, бұл ретте деректерді топтастырудың иерархиялық сипаты (10 деңгейге дейін) болуы мүмкін.

• есептер деректерді аралықты қоса алғанда санды өрістер бойынша, жазба топтары бойынша немесе бүкіл санды өріс бойынша қорытынды есептер бойынша есептей отырып шығару мүмкіндігін береді;

• есептер кесте түрінде рәсімделуі мүмкін, олардың құрылымы нақты кәсіпорында қабылданған (мысалы, тауарды қоймаға түсіру немесе қабылдау жүккұжаттары, автоматтандырылған жобалау жүйелеріндегі технологиялық үрдістердің маршрутты немесе операциялық карталары);

• есептерге нысандарға сияқты дерекқордың кесте аяларымен байланысқан және байланыспаған суреттерді, диаграммалар мен OLE басқа объекттерін қоюға болады;

• есептерге мәтіндік ескертпелерді қоюға болады;

• есептер фирманың қызметін жарнамалайтын тұсаукесерлер үшін пайдаланылуы мүмкін;

• есептерге обағынышты есептер енуі мүмкін.

Есепті басып шығарудан бұрын алдын ала экранда қарауға болады.

Есепті жасау бойынша жұмыстың басы нысанды жасау бойынша жұмыс технологиясына баламалы:

- дерекқорды ашу;
- дерекқор терезесінде *Есептер* объектісін ерекшелеп алу керек.

Содан кейін ДҚ объектілерінің терезесі *Есептер* ашылады (7.16 сур), ол жаңа есептер құру тәсілдерін және бұдан бұрын жасап шығарылған есептер тізбесін құрайды.

ACCESS 2000 жүйесінде жаңа есепті жасау үшін екі тәсіл ұсынылады:

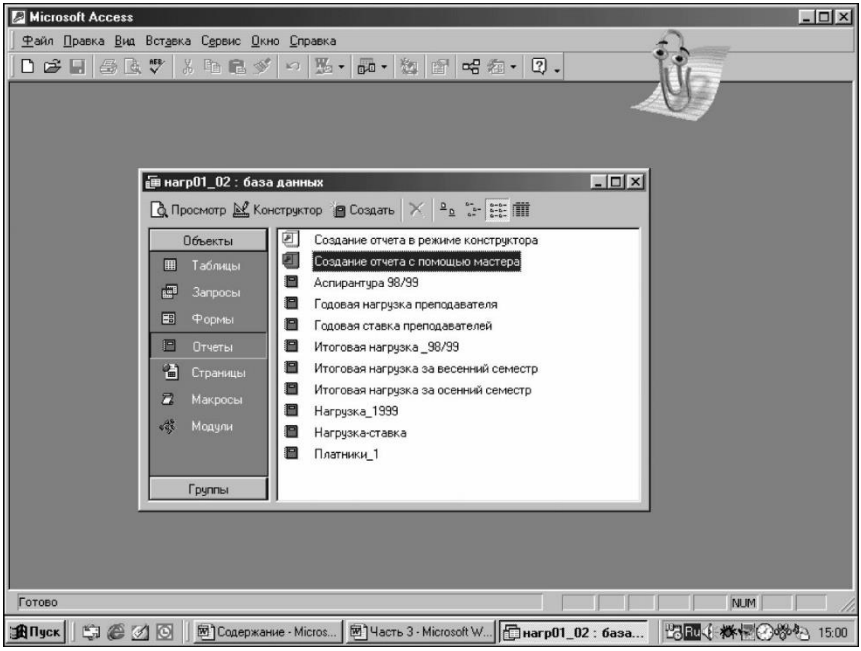
- есепті *Құрастырушы режимінде жасау*;
- есепті шебер арқылы жасау.

Есепті *Құрастырушы режимінде жасау*. Есептерді жасаудың осы тәсілін шебер көмегімен есептерді жасау дағдыларын алғаннан кейін жүргізу ұсынылады.

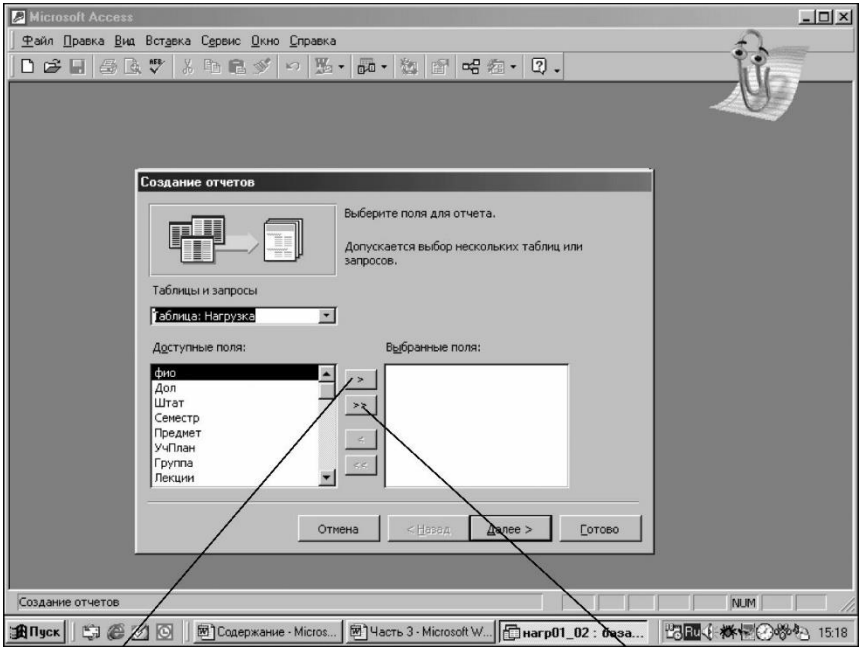
Шебер көмегімен есеп жасау. Есептерді жасаудың осы тәсілі ретімен ашылатын сұхбат нысандарында сұрақтарға жауап берудің белгілі бір реті болып табылады.

Бірінші нысанда (7.17 - сур):

- негізінде есеп жасалатын, кестені немесе сұранысты таңдау;
- мәндері есепке шығарылатын аяларды таңдау;



7.16 – сурет. ДҚ есептетері терезесі



/ Таңдалған сағ. аясын “көшіру” пернесі.
Кестенің немесе сұраныстың барлық аясын “көшіру” пернесі \

7.17 - сур. Кестенің немесе сұраныстың *Аяларды таңдау* терезесі

• барлық әрекетті орындағаннан кейін *Одан әрі* пернесіне басу керек.

Келесі ашылған терезе (7.18 - сур), жекелеген аялар бойынша топтастыру деңгейлерін қосу керек. Ұсынылған мысалда есептегі деректерді топтастырудың екі деңгейі көзделген. Бірінші деңгейге сәйкес барлық деректер оқытушылардың тегі - *Аты-жөні* аясы бойынша топтастырылады. Топтастырудың екінші деңгейінде барлық жазбалар әрбір оқытушы үшін семестрлер бойынша (көктемгі және күзгі) - Семестр аясы бойынша топтастырылады.

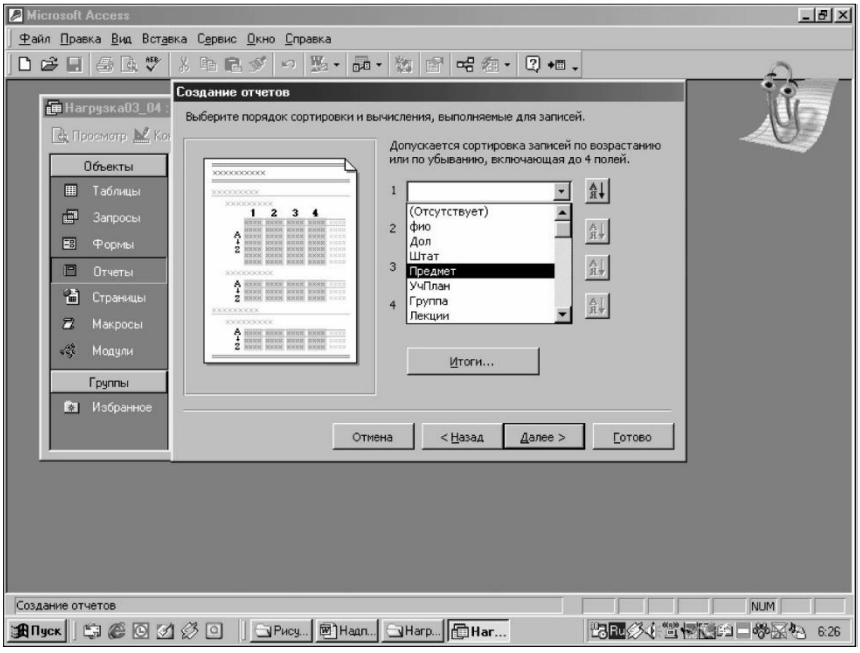
Келесі сұхбат терезесінде (7.19 - сур):

• жазбаларды — «өсуі» немесе «кемуі» бойынша сұрыптау тәртібі;

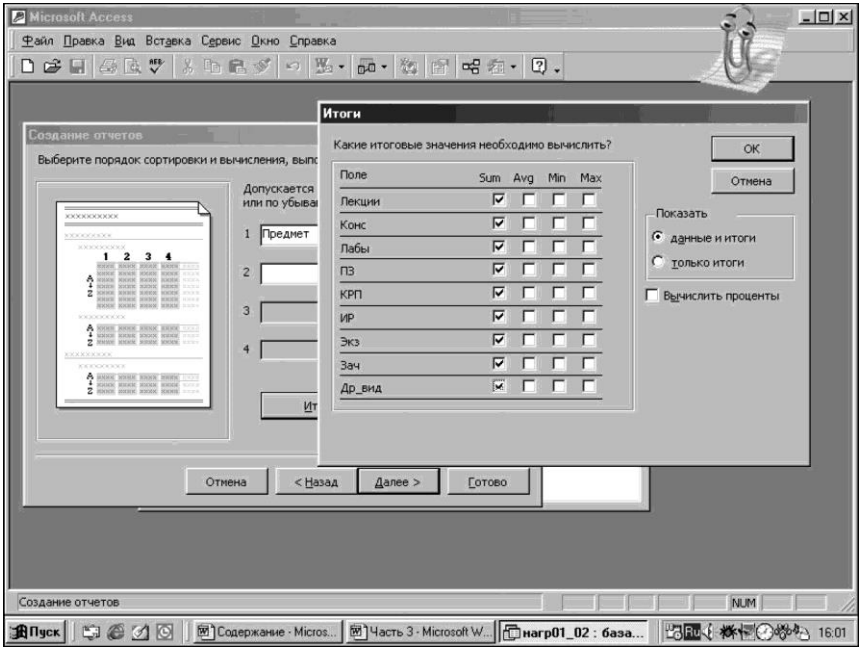
• қорытынды есеп жасау қажет болатын санды аяларды таңдау керек.

Келтірілген мысалда деректерді сұрыптау үшін *Мәні* аясы таңдалған. Бұл оқытушы лекция оқитын, зертханалық жұмыстар жасайтын немесе курстық жұмыстардың орындалуын басқаратын барлық пәндердің атауы есепті рәсімдеу кезінде әрбір оқытушы үшін әрбір секторда алфавит тәртібінде - “өсуі” бойынша орналасады.

7.18 - сур. Ерекшеленген аялар бойынша *Топтастыру деңгейлерін анықтау* терез



7.19 – сурет. Сұрыптау деңгейлерін анықтау



7.20 - сур. Есеп жүргізуге арналған *Аяларды таңдау* терезесі

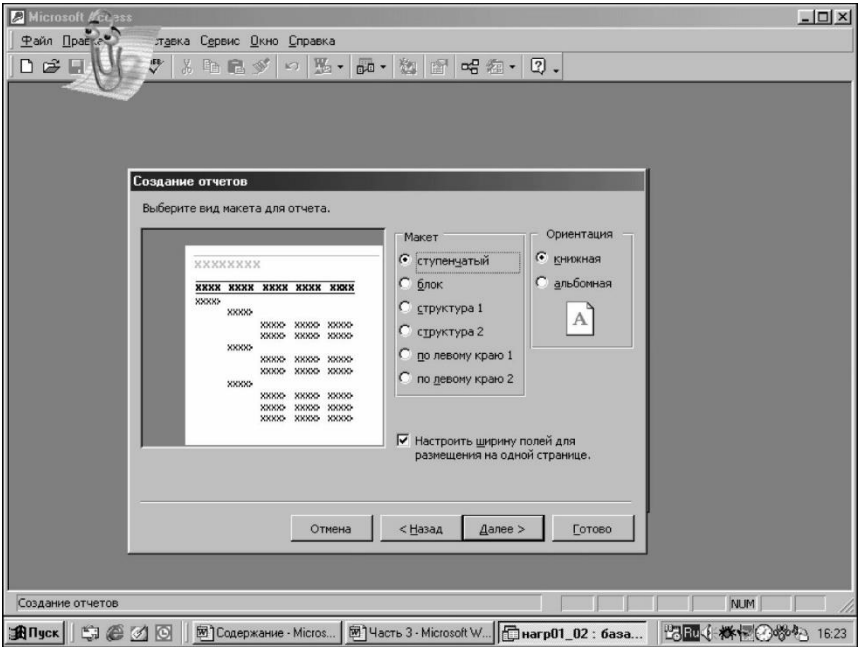
Қорытынды жасау аяларын таңдау үшін сұрыптау тәртібін орнатқаннан кейін *Қорытынды* пернесіне басу керек. Нәтижесінде жаңа сұхбат терезесі ашылады (7.20-сур.), онда қорытынды есептер көзделетін, санды аяларды ерекшелеу керек. Келтірілген суретте барлық санды аялар бойынша сағат санын есептеу көзделген.

Қорытынды терезесінің мына нұсқауына назар аударайық - қорытынды есептеулердің сипатын таңдау: “деректер мен қорытындылар” және “тек қорытындылар”.

Бірінші жағдайда есепке әрбір ая бойынша сағат саны көрсетілген барлық пәндер бойынша жазбалар және барлық ерекшеленген санды аялар бойынша есептеулер нәтижелері (сомалау) келтіріледі. Осындай ақпаратты шығару нысанын *деректер топтастырылған есептер және қорытынды жасау* деп атайды.

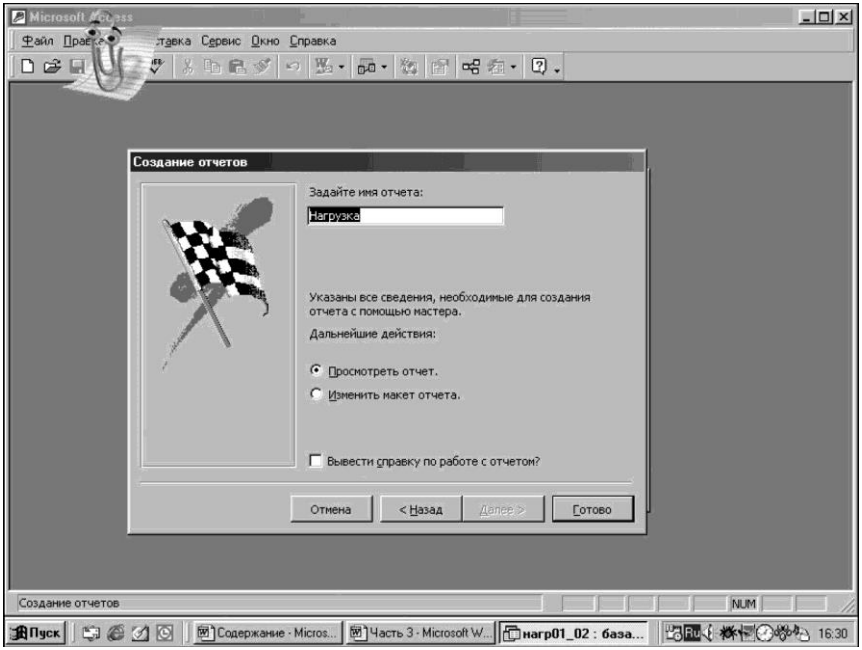
Екінші жағдайда әрбір оқытушы үшін пән атауы көрсетілмей әрбір семестр бойынша есептеулер нәтижелері ғана келтіріледі (әрбір ая бойынша семестрдегі сағаттың жалпы саны).

Барлық орындалған әрекеттен кейін деректерді есепте көрсететін макет түрлері ұсынылған терезе шығады (7.21-сур). Деректерді көрсету макетін таңдау аясында *Есептерді рәсімдеу стильдері* терезесі ашылады (7.22-сур.).

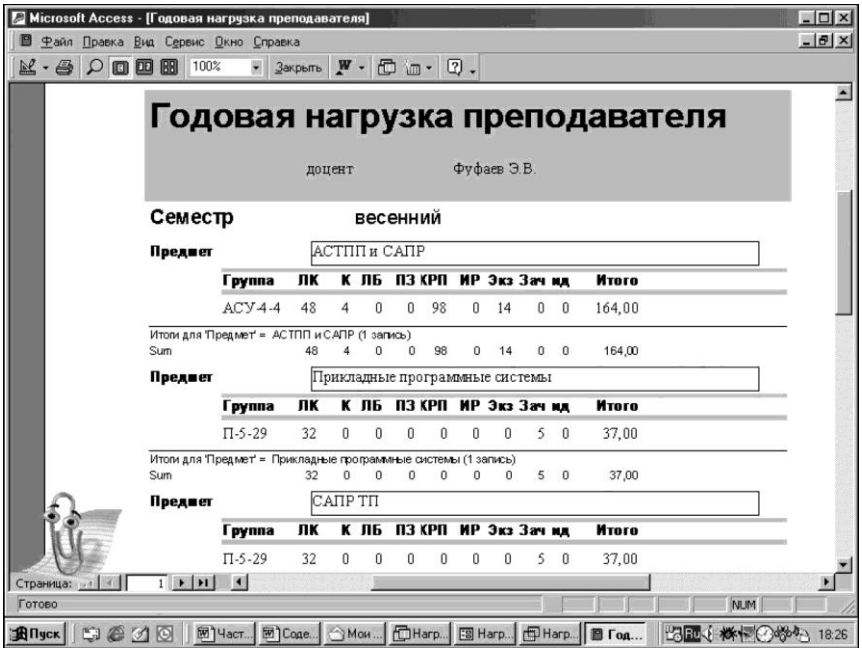


7.21-сурет. *Есепті рәсімдеу түрлерін таңдау* терезесі

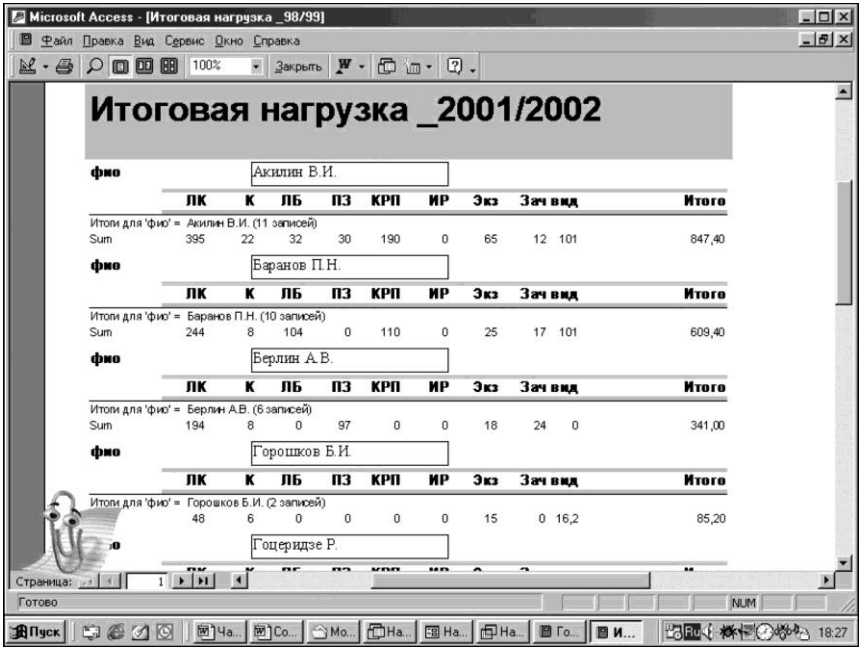
7.22 – сурет. Есептерді безендіру стилі терезесі



7.23 - сурет. *Есеп жасау шебері* соңғы сұхбат терезесі



7.24 - сурет. Деректер топтастырылған және қорытынды шығарылған есеп



7.25 - сурет. Қорытынды есеп

Рәсімдеудің бір стилін таңдағаннан кейін есепті жасап шығару аяқталады. Есепке ат тағайындау ғана қалады. Бұл келесі - соңғы - *Есептерді жасау шебері* сұхбат терезесінде жүргізіледі (7.23 - сур).

Есептің екі түріне мысал келтірейік: деректер топтастырылған және қорытынды жасалған есеп (7.24 - сур.) және қорытынды есеп (7.25 - сур.).

7.3. Макростар көмегімен дерекқор объектілерін басқару

Макрос жиі қайталанатын әрекеттердің орындалуын автоматтандыру үшін қолданылатын MicrosoftAccess нұсқаулықтарының, макрокомандалардың реті болып табылады.

ACCESS-тегі макрокомандалар функционалды тағайындалуы бойынша келесі кластарға бөлінеді:

- кестелерді, сұраныстарды, нысандарды, есептерді ашу және жабу;
- деректерді басып шығару;
- сұранысты орындау;
- шарттардың дұрыстығын тексеру және макрокомандаларды басқару;
- мәндерді орнату;
- деректерді іздеу;

- пайдаланушы мәзірін құру және мәзір командаларын орындау;
- ақпараттың экранға шығарылуын басқару;
- пайдаланушыға орындалатын әрекеттер туралы хабарлау;
- объекттердің атауын өзгерту, көшірме жасау, жою, импорт және экспорт;
- Windows басқа қосымшаларын іске қосу.

7.1-кестеде макростарды құруға арналған макрокомандалар аталған.

7.1-кесте

Басқарушы макростарды құруға арналған макрокомандалар тізбесі

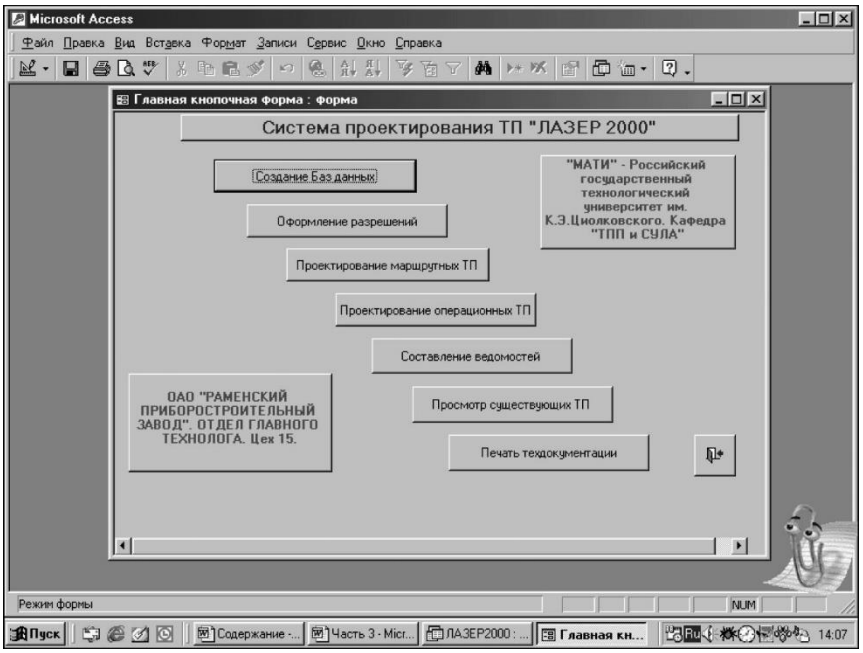
Макрокоманда	Тағайындалуы
Кестелерді, сұраныстарды, нысандарды, есептерді ашу және	
Жабу	Белсенді (белгіленген) терезені жабады
НысандыАшу	*Кесте, Құрастырушы* немесе *Нысан режимінде нысанды ашады*
МодулдіАшу	*Құрастырушы* режимінде модулді ашады
СұратудыАшу	*Кесте, Құрастырушы* режимінде сұранысты ашады
ЕсептіАшу	*Қарау режимінде есепті ашады*
КестелердіАшу	*Кесте, Құрастырушы режимінде кестені*
Деректерді басып шығару	
НысандыАшу	*Алдын ала қарау* режимінде нысанды ашады
ЕсептіАшу	*Қарау* режимінде есепті ашады
КестелердіАшу	*Кесте* режимінде кестені ашады
НысанғаШығару	XLS — Excel үшін немесе RTF — Word үшін
БасыпШығару	Деректерді (кестелерді, сұраныстарды немесе есепті) басып шығарады
Сұранысты орындау	
СұратудыАшу	Сұранысты іске қосады және таңдалған жазбаларды *Кестелер* режимінде шығарады
Шарттардың нақтылығын тексеру макрокомандалардың	
ОқиғаныңКүшінЖою	Оқиғаның күшін жояды (орындалатын әрекет)
МәзірКомандасы	Стандартты мәзір командасын орындайды. Меншік мәзірді құру үшін қолданылуы мүмкін
Шығу	Access барлық терезелерін жабады және одан шығады

Макрокоманда	Тағайындалуы
БағдарламаныІске Қ	РәсімдіAccess BASIC-те орындайды
МакростыІскеҚос	Басқа макросты іске қосады
БарлықМакростыТоқтату	Ағымдағыны қоса алғанда, барлық макростың жұмысын тоқтатады
МакростыТоқтату	Ағымдағы макростың жұмысын тоқтатады
	Мәндерді орнату
Жаңарту	Бұл команда белсенді объекттегі деректерді жаңарту үшін пайдаланылуы мүмкін (*Нысан немесе Кесте режимінде)*
ПернетақтаКомандалары	Басылған пернелердің ретін алмасу буферінде есте сақтайды
МәнінБелгілеу	Кез-келген басқару элементінің мәндерін жаңартады немесе өзгертеді
СүзгіҚолдану	Есептердегі немесе нысандардағы шығарылатын ақпаратты шектейді
КелесіЖазба	Ағымдағыдан кейінгі жазбаны іздейді
ЖазбаныТабу	Белгіленетін іздеу шартын қанағаттандыратын жазбаны іздейді
Жазбаға	Нөмірі көрсетілген, сонымен қатар бірінші, соңғы, келесі, бұдан бұрынғы жазбаны ағымдағы етеді

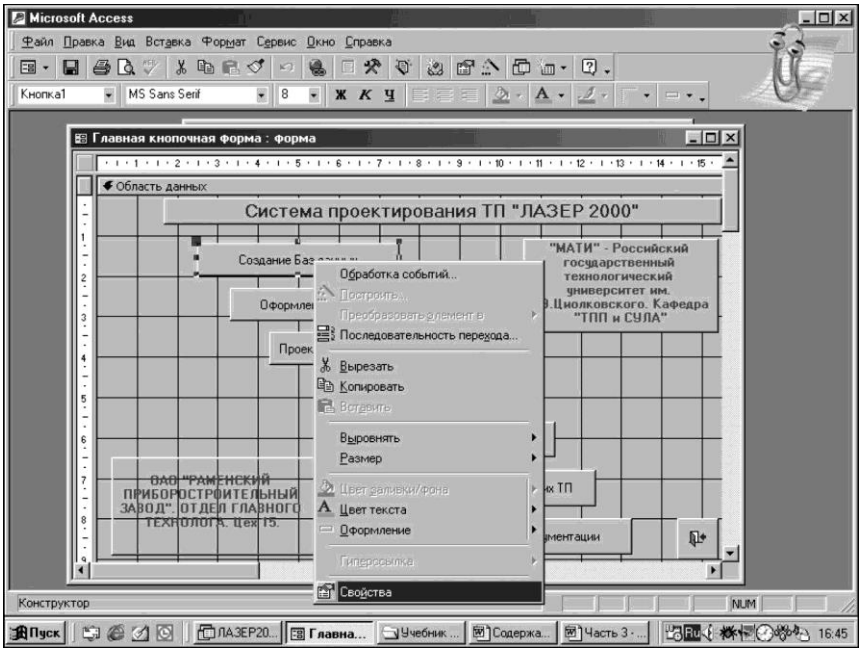
Пайдаланушы мәзірін құру және мәзір командаларын орындау

МәзірдіҚосу	Ашылатын мәзірді жасайды
МәзірКомандасы	Access стандартты мәзірінен бір әрекетті орындайды
	Экранға шығарылуын басқару
ЭкранғаШығару	Макростың жұмыс істеуі кезінде орындалатын, аралық әрекеттер туралы ақпаратты экранға шығарады
Бетке	Нысандағы немесе есептегі белгіленген бетке өтеді
Құмсағат	Меңзердің (нұсқаудың) пішінін макростың жұмыс істеу уақытына ауыстырады (мысалы, шағын кестеден таңдауға сұраныс жасау кезінде)
Ашу	Белсенді терезенің көлемін толық көлемге дейін арттырады

Макрокоманда	Тағайындалуы
Бүктеу	Белсенді терезені белгіге бүктейді
ЖылжытуКөлем	Белсенді терезенің көлемін жылжытады
ҚалпынаКелтіру	Объекттің бұрынғы көлемін қалпына
Объектті ерекшелеу	Белгіленген объекттің терезесін ерекшелейді
Жаңарту	Сұранысқа байланысты басқару элементтерінің деректерін жаңартады. Команда кестедегі немесе нысандарғы деректерді жаңартады
ХабарламаныОрнату	Жүйелік хабарламаларды немесе ескертулерді тағайындайды (күшін жояды). Қателік туралы хабарламалардың экранға шығарылуын тоқтатпайды
БарлықЖазбаныКөрсету	Бұдан бұрын орнатылған сүзгілерді елемей, кестенің барлық жазбасын көрсетеді
Құрал-саймандарПанелі	Стандартты немесе пайдаланушы құрал-саймандардың кез-келген панелін шығарады (алып тастайды)
	Пайдаланушыға хабарламалар
Сигнал	Дыбыстық сигнал береді
Хабарлама	Ескерту немесе ақпараттық хабарлама шығарады
	Объекттермен жұмыс істеу
ОбъекттіңКөшірмесінЖасау	Access басқа дерекқорына ағымдағы дерекқордың кез-келген объектісінің (жаңа немесе ескі атымен) көшірмесін жасайды
ОбъектіЖою	Объектті (кестені, сұратуды, нысанды, есепті, макросты, модулді) жояды
ФорматтаШығару	Объектті Windows қосымшаларының форматтарында шығарады
АтынӨзгерту	Ағымдағы ДҚ-да объекттің атын өзгертеді
ОбъектіЖіберу	Объекттің мазмұнын файлға шығарады және оны электронды поштаға — белгіленген адресатқа жібереді
ДерекқордыТүрлендіру	Басқа ДҚ-дағы объекттерді экспорттайды немесе импорттайды (dBASE, Paradox, FoxPro)
МәтіндіТүрлендіру	Деректерді экспорттау немесе импорттау кезінде қолданылады. Ақпаратты мәтіндік файлдарға жібереді немесе мәтіндік файлдардағы деректерді түрлендіреді
ЭлектрондыКестені Түрлендіру	Электронды кестелер үшін баламалы



7.26-сурет. Пернелі нысанның үлгісі



7.27-сурет. Перне қасиеттерін белгілеудің (өзгертудің) мәнмәтіндік мәзірі

Макросты орындау пернеге басумен байланысты пайдаланушы интерфейсінің нысанын жасап шығару кезінде екі тәсілді қолдануға болады:

• макросты жасау және оны нысанның тиісті пернесімен байланыстыру;

• нысанда перне жасау және оның қасиеттерін сипаттау кезінде тиісті макросты жасап шығару.

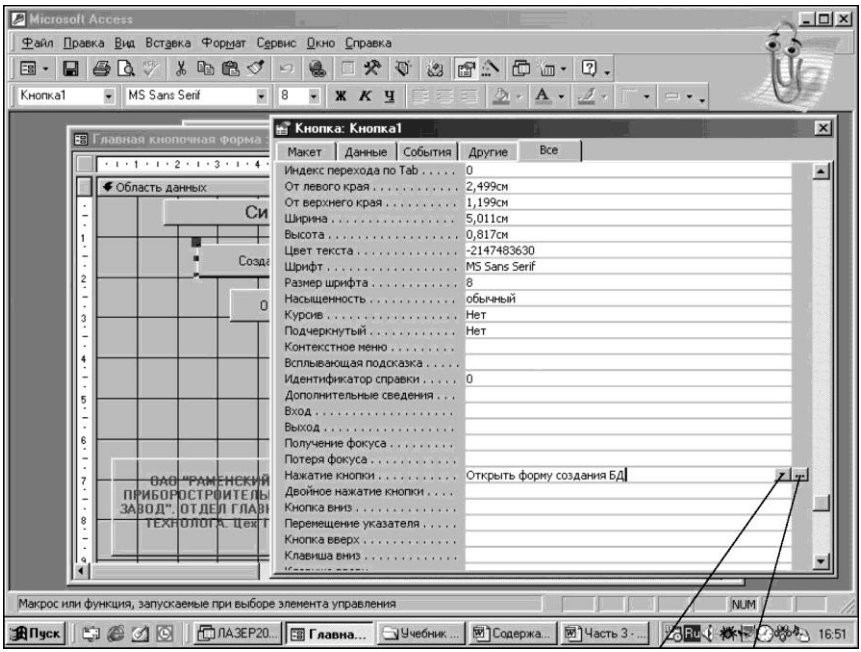
Кез-келген жағдайда алдын ала пернелі нысанды жасау керек. 7.26 суретте технологиялық үрдістерді автоматты жобалау жүйесінің пернелі нысаны көрсетілген.

Кез-келген пернеге басу кезінде тиісті сұхбат нысаны ашылу керек. Осы мысалда басқарушы макростарды жасау әрекеттерінің ретін қарастырайық. Сонымен, *Дерекқорды жасау* пернесі басқан кезде «Дерекқорды жасау» деп аталатын келесі сұхбат нысанын ашатын макрос орындалу керек.

Ол үшін келесі әрекеттерді орындау керек:

• *Құрастырушы* режимінде пернелі нысанды ашу керек;

• Макросты байланыстыру қажет болатын пернені ерекшелеу керек (қарастырылып жатқан мысалда бұл *Дерекқорды жасау пернесі);*



Бар макростарды таңдау пернесі

Макростарды, тіркестер мен бағдарламаларды құратын перне

7.28-сурет. Перне қасиеттерін белгілеудің (өзгертудің) мәнмәтіндік мәзірі

Ашылатын нысанның аты

• мәнмәтіндік мәзірді белсендіру (тінтуірдің оң жақ пернесі);

• ашылған мәнмәтіндік мәзірден *Қасиеттер* командасын таңдау (7.27 - сур);

• перне қасиеттерінің сипаттамасымен ашылған терезеде (7.28 - сур.) *Пернеге басу* немесе *Пернеге екі рет басу* қасиетін таңдау керек;

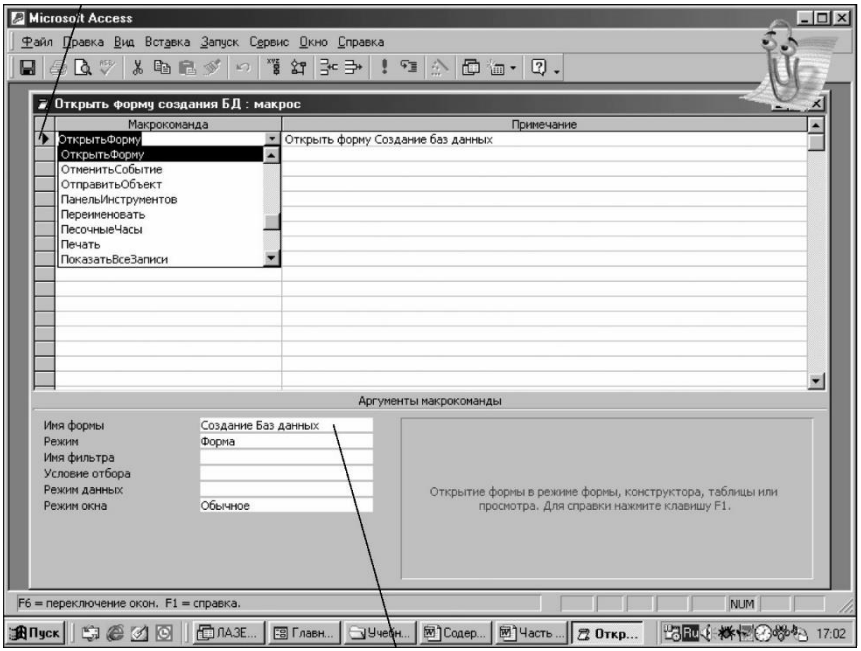
• осы мысалда *Пернеге басу* қасиетін таңдап, макростарды құрушы перне бойынша басу керек. Нәтижесінде макростың қасиеттерін сипаттау керек болатын терезе ашылады (7.29-сур.).

Дерекқормен кез-келген әрекетті орындауға арналған перне баламалы тәсілмен жасалады.

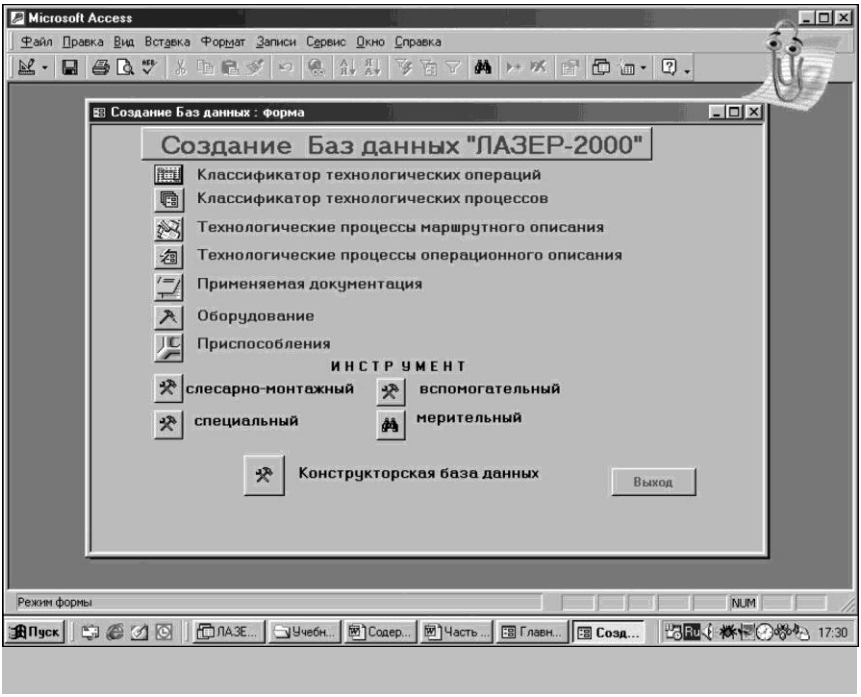
Қасиеттерді сипаттау терезесі ACCESS ортасындағы объектті-бағдарлы бағдарламалау негізі болып табылады. 7.30-суретте *Дерекқорды жасау* пернесіне басу кезінде ашылатын нысан ұсынылған. Осы нысанда пернелерді рәсімдеудің басқа үлгілері келтірілген.

Орындалған әрекеттер нәтижесінде перне үшін макрос жасалған және пернеге басу кезінде макрос командасы (немесе командалары) орындалады.

Макрокоманданы таңдау аясы

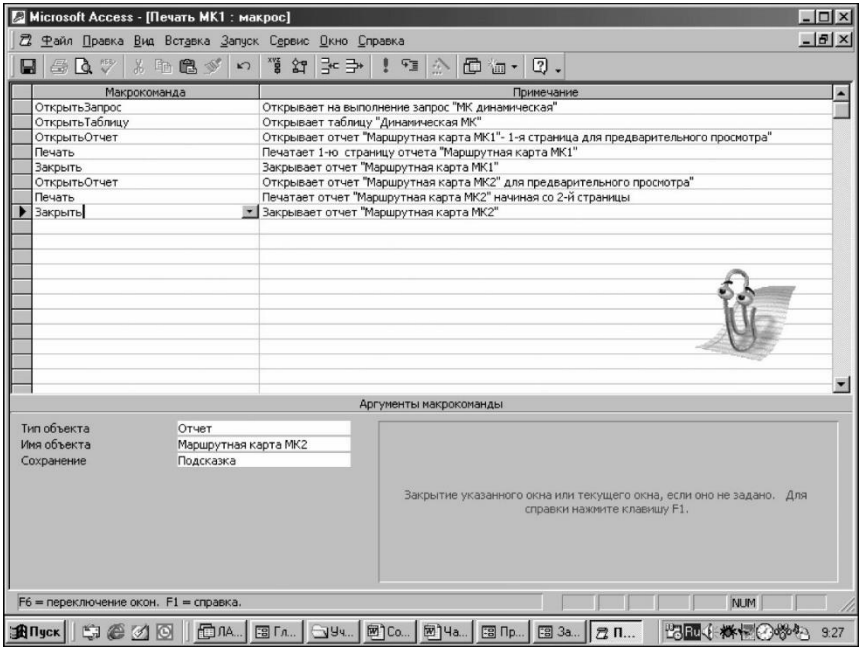


7.29-сурет. *Макростарды құрушы* терезесі



7.30- сур. Пернелі нысанның үлгісі

7.31- сур. Басқару пернелерімен деректерді енгізу нысанының үлгісі



7.32-сур.МК басып шығару пернесіне арналған макростың мазмұны

Бұл міндетті басқа реттілікпен де шешуге болады:

- пернелі нысанды жасап шығару;
- *Макростар Құрастырушысын* пайдалана отырып, әрбір перне үшін макростыжасап шығару.Бұл ретте әрбір макросқа ат тағайындау керек;
- *Құрастырушы режимінде нысанды ашу*;
- перне үшін *Қасиеттерін* терезесін ашу;
- тиісті жолда тізімнен макростың қажетті атын таңдау керек.

7.31 - суретте басқару пернелерімен ДҚ кестесінен ақпаратты енгізу нысанының үлгісі, ал 7.32-суретте - *МК басып шығару* пернесіне басқан кезде орындалатын макрокомандалардың ретінен тұратын макрос көрсетілген (бұйымды дайындаудың технологиялық үрдісінің маршрутты картаның мазмұнын басып шығару).

7.4. Пайдаланушы мәзірін жасап шығару

Ақпараттық жүйе пайдаланушының жұмыс істеуі үшін қолайлы болу үшін, тиімді деректер моделін (кестелер мен сұраныстардың құрамы мен өзара әрекеттесуі) жасаудан басқа қолайлы “достық” пайдаланушы интерфейсін жасап шығару керек.

Пайдаланушы интерфейсін жасап шығару құрал-саймандар панелін дәлдеумен, пайдаланушы мәзірін жасаумен, сұхбат нысандарын жасап шығарумен байланысты.

Құрал-саймандар панелін дәлдеу. Құрал-саймандар панелін, мәзір жолдарын және мәнмәтіндік мәзірді жасау және дәлдеу үшін, сонымен қатар түрі мен жұмысына әсер ететін қасиеттерді орнату үшін *Дәлдеу* сұхбат терезесі пайдаланылады. Оны ашу үшін *Түрі* мәзірінде *Құрал-саймандар панелі* және *Дәлдеу* қосалқы командасын таңдау керек.

Ашық дерекқорға арналған арнайы құрал-саймандар панелін келесі реттілікпен жасаңыз.

1. *Түрі мәзірінде Құрал-саймандар панелі командасын, содан кейін Дәлдеу қосалқы командасын таңдаңыз;*
2. *Құрал-саймандар панелі* қосымша бетінде *Жасау* пернесіне басыңыз.
3. *Құрал-саймандар панелі* аясында қажетті атты енгізіп, *ОК* пернесіне басыңыз.
4. *Құрал-саймандар панелі* қосымша бетінде *Қасиеттер* пернесіне басыңыз.
5. Қажетті қасиеттерін орнатып, *Жабу* пернесіне басыңыз.

Дәлдеу сұхбат терезесінен кейін жаңа құрал-саймандар панелі шығады.

Құрал-саймандар панелін жасауды аяқтау үшін келесі әрекеттерді орындаңыз.

1. *Дәлдеу* сұхбат терезесінен пернелерді қосыңыз.
2. Басқа құрал-саймандар панелінен перненің орнын ауыстырыңыз немесе көшірмесін жасаңыз.

Ескертпе: 1. Құрал-саймандар панеліне пайдаланушы мәзірін қосуға болады. 2. Бар макростарды іске қосатын пернелері бар құрал-саймандар панелі автоматты жасалады. Арнайы құрал-саймандар панелін нысанға немесе есепке қосуға болады.

Пайдаланушы мәзірін жасау. Арнайы мәнмәтіндік мәзірді келесі реттілікпен жасаңыз.

1. *Түрі мәзірінде Құрал-саймандар панелі командасын және Дәлдеу қосалқы командасын таңдаңыз*.
2. *Құрал-саймандар панелі* қосымша бетінде *Жасау* пернесіне басыңыз.
3. *Құрал-саймандар панелі* аясында қажетті атты енгізіп, *ОК* пернесіне басыңыз.
4. *Құрал-саймандар панелі* қосымша бетінде *Қасиеттер* пернесіне басыңыз.
5. “Тип” тізімі бар аясында “Мәнмәтіндік мәзір” тармағын таңдаңыз.
6. *Дәлдеу* жалаушасын орнатыңыз немесе алып тастаңыз және *Жабу* пернесіне басыңыз.

Мәнмәтіндік мәзір құрал-саймандар панеліне қосылады.

а

б

7.33-сур. Төменде шығатын мәзірі бар бет басы - нысанының үлгісі:
а — мәзір жабық; б — мәзір ашық

Сұхбат нысандарын жасап шығару. Пайдаланушы интерфейсі сұхбат нысандарының реттілігі болып табылатынын анық. Сәйкесінше, сұхбат нысандарын құрудан бұрын пайдаланушы интерфейсінің "сценарийін" жасау жұмысы болу керек.

Алдымен жасап шығарылған жүйенің негізгі тағайындалуын көрсету қажет болатын, бет басы *нысандары* жасалу керек. Осы нысандарда пайдаланушы мәзірін де ұйымдастыруға болады. Олар төменде шығатын немесе пернелі мәзір болуы мүмкін.

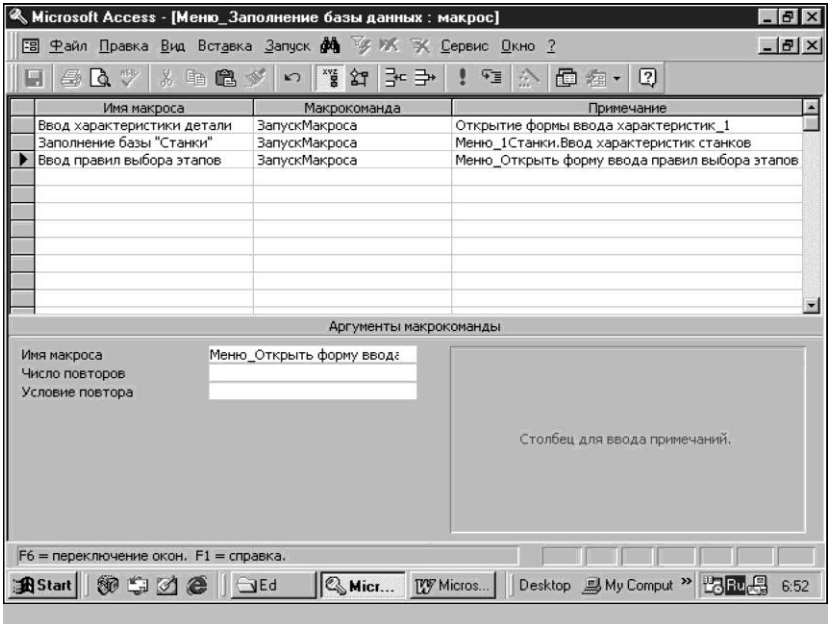
Төменде шығатын мәзірі бар бет басы — нысанның үлгісі (7.33-сур).

Төменде шығатын мәзірді макростардың белгілі бір реттілі түрінде жасауға болады.

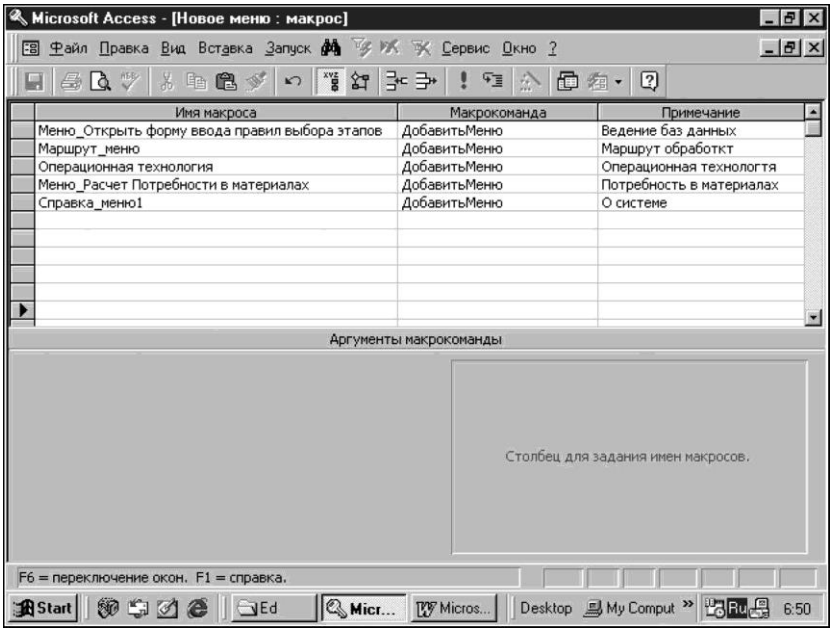
7.34-суретте Access жүйесінің дәстүрлі мәзірін алмастырған "Жаңа мәзір" макросы көрсетілген. Осы жаңа мәзір мына тармақтардан тұрады: "Дерекқорды жүргізу", «Өңдеу маршруты», «Операциялық технология», «Материалдар қажеттілігі» және «Жүйе туралы».

"Жаңа мәзір" тармағының әрбіреуі үшін келесі деңгейлі қосалқы мәзірдің құрамын орнататын келесі макрос жасап шығарылады.

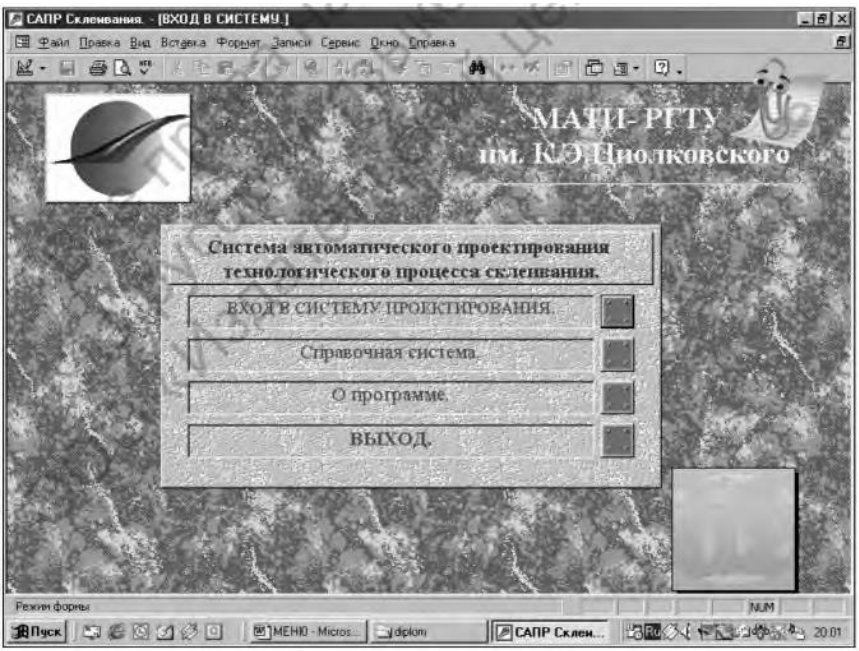
7.35-суретте "Дерекқорды жүргізу" тармағы үшін қосалқы мәзір құрамын белгілейтін, "Мәзір_Дерекқорды толтыру" макросы көрсетілген. Бұл макрос әрбіреуі тиісті макросты іске қосатын үш макрокомандадан тұрады.



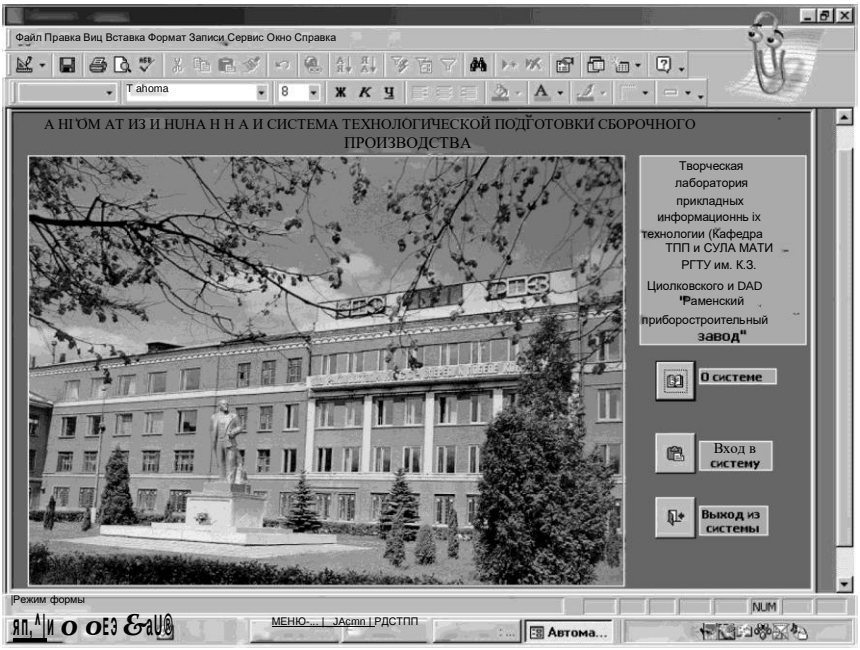
7.34-сур.Жаңа мәзірді құру макросының мәтінің үлгісі



7.35-сур. Мәнмәтіндік мәзірді жасайтын макрос мәтінінің үлгісі



7.36-сур. Пернелі мәзір нысанының үлгісі



7.37-сур. Фотосуреті бар пернелі мәзір нысанының үлгісі

Әрбір орындалуға іске қосылатын макростың тағайындалуы тиісті деректер енгізу нысанының ашылуында. Осы тәсілмен көп деңгейлі мәзірді жобалауға болады.

Пайдаланушы үшін мәзір жасаудың осындай тәсілінен басқа командалық пернелер түріндегі мәзірді қалыптастыру тәсілі барынша таралған, оларға баса отырып, пайдаланушы интерфейс сценарийі бойынша одан арғы жолды таңдайды.

7.36-суретте пернелі мәзір нысанының үлгісі көрсетілген.

Келтірілген үлгіден көрінетіндей, нысанның аясында нысанды безендіре алатын суреттер қоюға болады.

7.37-суретте фотосуреті бар пернелі мәзір нысанының үлгісі көрсетілген.

«Нысанмен жұмыс істеу жағымды болатындай жасау керек», - деген ниет білдіруге болады.

Бақылау сұрақтары

1. Ақпаратпен жұмыс істеудің қандай рәсімдері деректерді кестеге енгізу нысандарын қамтамасыз етеді?
2. Нысан түрлерін және оларды ACCESS жүйесінде конструкциялау тәсілдерін атаңыз.
3. *Нысандар құрастырушы* қандай блоктан тұрады?

4. Нысандар құрастырушының элемент панельдері қандай басқару командаларын (пернелерін) құрайды және олардың тағайындалуы қандай?

5. “Көпшілікке біреу” қатынасымен байланысты, деректерді кестеге енгізудің құрамдас нысандарын қандай реттілікпен жасап шығару керек?

6. “Пайдаланушы интерфейсі” ұғымы нені білдіреді?

7. Объектті-бағдарлап бағдарламалау әдісінің негізі не?

8. Пернелі нысан дегеніміз не? Пернелі нысандар қандай мақсатта жасап шығарылады?

9. Макрос және модуль дегеніміз не және дерекқорды басқарудың қандай міндеттері үшін жасап шығарылады?

10. Деректерді іріктеу шарттарын енгізу нысандарын сұранысты жобалау реттілігі қандай?

11. *Құрастырушы арқылы тіркес құру кезіндегі әрекеттер реті қандай?*

12. Төмендегі жазбадағы «Forms», «Кафедраның жүктемесі», «Тізімі бар ая0» сөздері нені білдіреді:

[Forms]! [Кафедраның жүктемесі]! [Тізімі бар ая]?

13. Деректерді енгізу нысанында жазбаларды қарау үшін қандай пернелер бар?

14. Құрамдас нысандарды қолдана отырып, байланысты кестелерге деректер енгізу реттілігі қандай?

15. Деректерді енгізу нысанында бірінші немесе соңғы жазбаны қарау үшін қандай әрекеттер ретін орындау керек?

16. Жаңа жазба енгізу үшін қандай әрекеттер ретін орындау керек?

17. Деректер енгізу нысанында кез-келген жазбаны қарау үшін қандай әрекеттер ретін орындау керек?

18. Кез-келген жазба аясында деректерді редакциялау және одан әрі енгізу үшін қандай әрекеттер ретін орындау керек?

19. Дерекқордағы есептер не үшін (қандай мақсатта) жасап шығарылады?

20. Microsoft Access-те қандай есеп жасау тәсілдері бар?

21. Деректер топтастырылған есеп қандай мақсатта жасап шығарылады және не болып табылады?

22. Қорытынды есеп қандай мақсатта жасап шығарылады және не болып табылады?

8 тарау

VISUAL BASIC FOR APPLICATIONS ОРТАСЫНДА БАСҚАРУШЫ БАҒДАРЛАМАЛАРДЫ ЖАСАУ

8.1. Visual Basic for Applications жалпы сипаттамалары

Visual Basic for Applications (VBA) бағдарламалау тілі Microsoft Office құралдарын жасап шығару ортасы ретінде пайдалана отырып, нақты пайдаланушының әрекеттерін автоматтандырудан бастап және толық ауқымды қосымшаларды жасап шығарумен аяқтай отырып, бағдарламалаудың кез-келген міндеттерін шешу мүмкіндігін беретін яMicrosoft Office барлық қосымшалары үшін ортақ құрал-сайман болып табылады.

Visual Basic for Applications бағдарламалаудың объектті-бағдарлы тілі болып табылатындықтан, осы қосалқы бөлімшеде Access-те пайдаланылуы мүмкін объектті модельдер сипатталған.

Access қосымшаларында негізгі жұмыс— бұл деректермен жұмыс, сондықтан біз кейбір деректерді басқару кітапханаларына сипаттама береміз:

- DAO (Data Access Objects);
- ADO (ActiveX Data Objects);
- JRO (JET AND REPLICATION OBJECTS).

Visual Basic forApplications тілінің келесі компоненттерін қарастырайық:

- рәсімдер мен қызметтер;
- ауыспалылар, константалар және деректер түрі;
- ауыспалылар мен рәсімдердің әрекет ету саласы;
- басқарушы конструкциялар - тармақталуы және циклдер;
- циклдер мен рәсімдерден шығу;
- модульдер.

8.2. Рәсімдер мен қызметтер

VBA-дағы бағдарламаның негізгі компоненттері Sub және EndSub операторлары немесе Function және EndFunction операторлары арасында жасалған бағдарламалық кодтың фрагменттері болып табылатын, рәсімдер мен қызметтер болып табылады, мысалы:

Sub<Рәсімнің аты> (<аргумент 1>, <аргумент 2>, ...) <оператор 1>
<оператор 2>
EndSub

немесе

Function<Қызметтің аты> (<аргумент 1>, <аргумент 2>, ...)
<оператор 1>
<оператор 2>

<Қызметтің аты> = <қайтарылатынМән>
EndFunction

Қызметтің рәсімнен ерекшелігі, оның атауы да ауыспалы болып табылады және мәнді қызметті шақыру нүктесіне қайтару үшін пайдаланылады.

Жазылған рәсімді немесе қызметті орындау үшін оны шақыру керек. Аргументтер тізімі бос болмайтын рәсімді тек басқа рәсімнен немесе қызметтен шақыруға болады. Бұл ретте агрументтердің іс-жүзіндегі мәндерінің тізімі бар атын VBA операторларының бірі ретінде белгілеу керек. Қызметті жекелеген VBA операторы көмегімен ғана емес, сондай-ақ агрументтердің іс-жүзіндегі мәндерінің тізімі бар атын тікелей формулаға немесе VBA негізіндегі бағдарламадағы тіркеске орналастырып, мысалы, Access-тің есептелетін сұраныс, нысандар мен есептердің өрістегіндегі формулаға орналастыра отырып шақыруға болады. Аргументтер тізімі бос рәсім (командалық макрос) басқа рәсім немесе қызметтен емес, сондай-ақ жылдам шақыру пернелерінің комбинациясының көмегімен, ашылатын мәзір командалары немесе құрал-саймандар панелінің пернелері көмегімен шақырылуы мүмкін. Сондай-ақ осы рәсімді әртүрлі оқиғаның орындалуымен байланыстыруға болады (мысалы, нысанды немесе есепті ашумен, нысандағы перне бойынша тінтуір шертпесімен, нысандарды басқару элементтеріне әсерімен, атап айтқанда, ActiveX басқару элементтеріне). Мұндай рәсімдер оқиғаларды өңдеу рәсімдері деп аталады. Аргументтердің берілуін талап ететін қызметтер немесе рәсімдерді мұндай тәсілмен шақыру мүмкін емес.

Егер шақырылатын рәсімнің аты бірегей болса және шақыратын рәсім орналасқан модульде орналасса, онда оны шақыру үшін атын көрсету , содан кейін жақшаға алмай аргументтердің іс-жүзіндегі мәндер тізімін белгілеу жеткілікті. Рәсімді шақырудың басқа тәсілі Call операторын пайдалануда. Алдымен Call операторы, содан кейін рәсімнің аты, содан кейін параметрлер тізімі жүреді, бұл жағдайда міндетті түрде жақшаға алу керек.

Қызметті рәсім сияқты шақыру болады, алайда жиі қызметті шақырудың басқа, ерекше тәсілі пайдаланылады - жақшаға алынған, тағайындау операторының оң жағындағы параметрлер тізімі бар аты пайдаланылады.

Екі аргументті (константа және тіркесті) бере отырып, CrossRC атты рәсімді шақыру үлгілері:

```
CrossRC 7, i + 2
```

немесе

```
Call CrossRC (7, i + 2)
```

Ал мында екі қызметті (Left және Mid) шақыру және олардан тіркес ретінде қайтарылатын мәнді пайдалану үлгісі:

```
yStr = Left (y, 1) &Mid (y, 2, 1)
```

Рәсімге немесе қызметке ауыспалыларды берудің екі түрлі тәсіліне рұқсат етіледі: сілтеме бойынша және мәні бойынша. Егер ауыспалы *сілтеме* бойынша берілсе, рәсімге немесе қызметке осы ауыспалының жадында мекен-жай беріледі. Шақырылатын рәсім нақты параметрдің мәнін өзгертуі мүмкін. Егер нақты параметр *мәні* бойынша берілсе, онда шақырылатын рәсім немесе қызмет осы параметр ретінде пайдаланылатын ауыспалының өзін емес, нақты параметрдің мәнін ғана алады. Нақты параметрдің алынған мәнінің барлық өзгерістері (егер шақырылатын рәсіммен орындалса) нақты параметрдің ауыспалысының мәніне әсер етпейді.

Параметрлерді рәсімге немесе қызметке беру тәсілі оның аргументтерін (формальды параметрлерін) сипаттау кезінде көрсетіледі. Аргументтің атына дейін жіберу тәсілін айқын сипаттаушы болуы мүмкін (ByRef сілтеме бойынша жібереді, ал ByVal — мәні бойынша жібереді). Егер параметрді жіберу тәсілі айқын көрсетілмесе, онда өздігінен сілтеме бойынша жіберу деп түсінуге болады.

Мысалмен түсіндірейік. Мына екі рәсімнің сипаттамасы болсын — Main және Examplel.

```
Sub Main ()
a = 10 b =
20 c = 30
Call Example1 (a, b, c,)
Call MsgBox (a)
Call MsgBox (b)
Call MsgBox (c)
End Sub
Sub Example1 (x, ByVal, ByRef) x
= x + 1 y = y + 1 z = z + 1 Call
MsgBox (x)
Call MsgBox (y)
Call MsgBox (z)
End Sub
```

Examplel қосалқы рәсімі формальды аргументтер ретінде әртүрлі сипатталған ауыспалыларды пайдаланады. Одан әрі осы рәсімнің барысында әрбіреуі бір бірлікке артады, содан кейін олардың мәндері MsgBox қызметі арқылы экранға шығарылады.

Main негізгі рәсімі a, b, c аыспалыларының мәндерін орнатады, содан кейін оларды нақты аргументтер ретінде Examplel рәсіміне жібереді. Бұл ретте бірінші аргумент сілтеме бойынша (өздігінен әрекет етеді), екіншісі - мәні бойынша, үшіншісі - сілтеме бойынша жіберіледі.

Examplel рәсімінен қайтқаннан кейін негізгі рәсім де экранға аргумент ретінде жіберілген үш ауыспалының мәнін шығарады.

Барлығы экранға алты мән шығарылады: алдымен 11, 21 және 31 сандары (барлық алынған мәндер 1-ге арттырылған және Example1 рәсімімен шығарылады); содан кейін 11, 20 және 31 сандары (бұл мәндер Main рәсімімен шығарылады, сілтеме бойынша жіберілген ауыспалылар артты, ал мәні бойынша жіберілген ауыспалы - артқан жоқ).

Бағдарлама бір немесе бірнеше модульде орналасуы мүмкін көптеген рәсім мен қызметтен тұрады (әдетте тұрады). Модульдер жобаларға топтастырылады; бұл ретте бір жобада жалпы модульдерді немесе рәсімдерді пайдаланатын бірнеше әртүрлі бағдарлама орналасуы мүмкін.

Бір модульде орналасқан әрбір рәсімнің аты бірегей болу керек; бұл ретте жобада бірнеше әртүрлі модуль болуы мүмкін. Әдетте бір жобада рәсімдердің тек бірегей аттарын пайдалану ұсынылады, бірақ ерекшеліктер де бар. Егер жобада аты бір бірнеше әртүрлі рәсім болса, онда атын нақтылау үшін рәсімді шақыру кезінде мына синтаксисті пайдаланған жөн:

< МодульдіңАты>.<РәсімніңАты>

Егер бұл ретте модульдің аты бірнеше сөзден тұрса, онда осы атты шаршы жақшаларға алу керек. Мысалы, егер модуль “Графикалық рәсімдер” деп, ал рәсім - “Крестик” деп аталса, онда шақыру төмендегідей болуы мүмкін:

[Графикалық рәсімдер].Крестик

Сондай-ақ басқа жобаларда орналасқан рәсімдерді де пайдалануға болады. Бұл ретте атты нақтылаудың тағы бір деңгейі қажет болуы мүмкін

< ЖобаныңАты>.<МодульдіңАты>.<РәсімніңАты>

8.3. Ауыспалылар, константалар және деректер түрі

Бағдарламалаудың басқа тілдеріндегідей, VBA-да ауыспалы мәндерді сақтау үшін, параметрлерді жіберу және есептеулерді жүргізу үшін ауыспалылар пайдаланылады. VBA-да ауыспалыларды сипаттау және пайдаланудың негізгі ерекшеліктеріне қысқаша тоқталайық:

Әдетте, ауыспалыны пайдаланудан бұрын, ол жарияланады, яғни алдымен ауыспалылардың қандай аттары бағдарламада пайдаланылатыны көрсетіледі; бұл ретте сақтау үшін осы ауыспалы пайдаланылатын деректердің түрлері жарияланады.

VBA-да, Basic-тің дағдылы тілінде сияқты төмендегі синтаксиспен Dim операторы пайдаланылады:

Dim <АуыспалыныңАты>[As <ДеректерТүрі>]

VBA-да ауыспалыларды атаудың келесі ережелері әрекет етеді:

- аты 255 символдан ұзын болмайды;
- аты артынан әріптер, цифрлар немесе астынан сызу символдары жүруі мүмкін әріптен басталу керек;
- атын жазған кезде бос орындар, тыныс белгілері немесе арнайы символдар болмау керек;
- ауыспалының атының соңында деректер түрін көрсететін, алты арнайы символдың біреуі қосылуы мүмкін: !, #, $, %, &, @. Бұл символдар ауыспалы атының бір бөлігі болып табылмайды. Егер бағдарламада бір уақытта string1$ және stringl аттары пайдаланылса, онда олар бір жол ауыспалысына сілтейді;
- деректер түрін анықтау символы әртүрлі ауыспалының бір атын пайдалануға немесе деректердің түрі бір уақытта сипаттауға және осы деректер түріне сәйкес келмейтін арнайы символды пайдалануға болмайды;
- ауыспалылардың аты ретінде VBA өзекті сөздерін және стандартты объект аттарын пайдалануға болмайды. Сондықтан ауыспалылардың атын бас әріптен емес, кіші әріптен бастау ұсынылады;
- VBA-да өзекті сөздерді және стандартты объекттердің атын енгізу кезінде бірінші әрібі автоматты бас әріпке айналады;
- ауыспалыларды белгілеу кезінде олардың атында латын алфавитін ғана емес, сондай-ақ кириллицаны пайдалануға болады.

Көптеген бағдарламалау тілдерін, мысалы Pascal-да ауыспалылар міндетті түрде хабарландырылу керек және осы хабарландырулар ауыспалылардың жадын резервтеу кезінде компилятормен пайдаланылады. Сол уақытта VBA-да - әдетті Basic тілінде хабарландырылмаған ауыспалыларды пайдалануға болады.

Бағдарламалау тілдеріндегі қиын байқалатын, сипатталмаған ауыспалылардың қолданылуына рұқсат ететін қателіктердің ең қауіпті көздерінің бірі ауыспалы аттарын жазу барысындағы жаңылысу болып табылады. Мұндай жаңылыстар транслятормен бұдан бұрын пайдаланылғаннан ерекше, тағы бір жаңа ауыспалының пайда болуы ретінде түсіндірілді және қате деп қабылданбайды.

VBA-да “соломон шешімі” қабылданған - осы проблеманы шешу бағдарламашының өзіне ұсынылған. Ол үшін Option Explicit операторы бар. Модульдің сипаттамасы осы оператордан басталу керек. Бұл жағдайда VBA осы модульде ауыспалылардың міндетті хабарландырылуын талап етеді және жарияланбаған ауыспалы кездескен сайын қателік туралы хабарламаны генерациялайды.

8.1-кестеде пайдаланылатын VBA деректер түрінің тізбесі келтірілген.

Ауыспалыларды сипаттау кезінде деректер түрі пайдаланылмайды. Мұндай жағдайда ауыспалы түрі ауыспалы атының соңғы символымен анықталуы мүмкін: @, #, %, &, !, $ (сәйкесінше Currency, Double, Integer, Long, Single немесе String). Мысалы, «$» символы жол деректерінің түрін анықтау символы болып табылса, онда text$ атты

8.1-кесте

VBA деректер түрі

Деректер түрі	Сипаттамасы
ARRAY	Ауыспалылар массиві. Массивтің нақты элементіне сілтеу үшін индекс пайдаланылады. Қажетті жад массив көлеміне тәуелді.
BOOLEAN	Екі логикалық мәннің бірін қабылдайды: True(Шындық) немесе *False*(Жалған) . Қажетті жад — 2 байт
BYTE	Таңбасыз сан — 0 бастап 255 дейін. Қажетті жад — 1 байт
CURRENCY	Жуықтау қателіктеріне жол бермеу қажет болғанда, үтірден кейінгі бекітілген ондық таңба санымен ақша есебін жүргізу үшін пайдаланылады. Ықтимал мәндер диапазоны: -922 337 203 685 477,5808 бастап 92 2 3 3 7 203 685 477,5807 дейін. Қажетті жад — 8 байт. Өздігінен түрін анықтау үшін «@» символы пайдаланылады.
DATE	Күндерді сақтау үшін пайдаланылады. Ықтимал мәндер диапазоны: 0100 ж. 1 қаңтардан 9999 ж. 31 желтоқсанға дейінҚажетті жад — 8 байт

Деректер түрі	Сипаттамасы
DOUBLE	Нақтылығы қосарланған тұрақсыз нүктесімен санды мәндер. Теріс сандар үшін ықтимал мәндер диапазоны: -1,79769313486232E308 бастап -4,94065645841247E-324 дейін. Оң сандар үшін ықтимал мәндер диапазоны: 4,94065645841247E-24 бастап 1,79769313486232E308 дейін. Қажетті жад — 8 байт. Өздігінен түрін анықтау үшін «#» символы пайдаланылады.
INTEGER	Қысқа бүтін санды мәндер. Ықтимал мәндер диапазоны: -32 768 бастап 32 767 дейін. Қажетті жад — 2 байт. Өздігінен түрін анықтау үшін «%» символы пайдаланылады.
LONG	Ұзын бүтін санды мәндер. Ықтимал мәндер диапазоны: -2 147 483 648 бастап 2 147 483 647 дейін. Қажетті жад — 4 байт. Өздігінен түрін анықтау үшін «&» символы пайдаланылады.
OBJECT	Объекттерге сілтеме сақтау үшін ғана пайдаланылады. Қажетті жад — 4 байт
SINGLE	Нақтылығы әдетті тұрақсыз нүктемен санды мәндер. Теріс сандар үшін ықтимал мәндер диапазоны: -3,402823E38 бастап -1,401298E-45 дейін. Теріс сандар үшін ықтимал мәндер диапазоны: 1,401298E -45 бастап 3,402823E38 дейін. Қажетті жад — 4 байт. Өздігінен түрін анықтау үшін «!» символы пайдаланылады.
STRING	Жол мәндерін сақтау үшін пайдаланылады. Жолдың ұзындығы — 0 бастап 64 Кбайт дейін. Қажетті жад — 1 символға 1 байт. Өздігінен түрін анықтау үшін «$» символы пайдаланылады.
VARIANT	Әртүрлі деректер түрін сақтау үшін пайдаланылуы мүмкін. Қажетті жад — 16 байт плюс жол мәндерінің әрбір символына 1 байт. Өздігінен түрін анықтау символы жоқ.
ПАЙДАЛАНУШЫ АНЫҚТАЙТЫН	Пайдаланушы анықтайтын деректер түрі. Бөлінетін жадтың тағайындалуы мен көлемі түрін анықтауға тәуелді. Деректердің құрылымын сипаттау үшін пайдаланылады. Әртүрлі деректер түрінің көптеген әртүрлі мәнін сақтау мүмкіндігін береді.

автоматты түрде “символдар жолы” типті ауыспалы болып табылады. Егер соңғы символ жоғарыда аталған символдың ешқайсысы болмаса және түрі айқын көрсетілмесе, онда мұндай жағдайда ауыспалы кез-келген типті деректерді сақтау мүмкіндігін беретін Variant деректер түрін өздігінен тағайындайды.

Бір рәсімде ауыспалының соңында бір-бірінен тек түрін анықтайтын арнайы символмен ерекшеленетін ауыспалыны пайдалануға болмайды. Мысалы, бір уақытта var$ және var % ауыспалыларын пайдалануға болмайды. «AS <АуыспалыТүрі>» сипаттағыш көмегімен аттың соңында түрін анықтау символын құрайтын ауыспалыны айқын жариялауға болмайды (егер мұндай анықтау түрін анықтау символының дағдылы қолданылуына қайшы болмаса да). Мысалы, келесі анықтаманың кез-келгенін енгізуге тырысып, қателік туралы хабарлама аласыз:

```
Dim var1$ As String
Dim var2% As Integer
```

Рәсімнің немесе қызметтің аргументтерінің деректер түрін анықтау үшін рәсімнің немесе қызметтің тікелей бас жолында деректер түрін сипаттау пайдаланылады. Мысалы, рәсімнің келесі бас жолы оның параметрлерін жол типті ауыспалылар ретінде сипаттайды:

```
Sub SplitStr(strl As String, str2 As String, str3 As String).
```

Қызметпен қайтарылатын мәндер деректер түрін анықтау қызметтің бас жолын аяқтайды, мысалы,

```
Function FindSplitSpace (strl As String) As Integer
```

қызметпен қайтарылатын мәнді қысқа типті ауыспалы ретінде сипаттайды.

Ат берілген константалардың пайдаланылуын қарастырайық. Оларды сипаттау үшін Dim ауыспалыларын сипаттау операторына ұқсас Const операторы пайдаланылады. Осы операторды синтаксисі:

```
Const < КонстантаныңАты>(As <ДеректерТүрі >] = <тіркес>
```

мұндағы<тіркес> — бұл константа ретінде пайдаланылуы қажет, мәнді қайтаратын кез-келген мән немесе формула.

Мысалы, келесі оператор бүтін санды константаны анықтайды maxLen:

```
ConstmaxLen% = 30
```

Пайдаланушы сипаттайтын константалардан басқа алдын ала сипаттаусыз бағдарламалар мәтінінде пайдаланатын алдын ала айқындалған кіріктірілген константалар бар.

Кіріктірілген константаларды атау кезінде бұл константаның қандай қосымша объектілеріне жататынын анықтау мүмкіндігін беретін стандартты келісім пайдаланылады. Мысалы, Access объекттеріне жататын кіріктірілген константалардың аттары, «ac» префиксінен басталады; Excel объекттеріне жататын— «xl» префиксінен; Word объекттеріне жататын — «wd» префиксінен; VBA объекттеріне жататын— «vb» префиксінен басталады.

Мысалы,

DoCmd. OpenForm «Orders», acNormal, stLinkCriteria

командасында кіріктірілген Access acNormal константасы пайдаланылады.

Объекттерге сілтемелер. Visual Basic-те әдетті ауыспалылардан басқаобъектке сілтеме болып табылатын ауыспалылар жиі пайдаланылады. Объекттерге сілтеме үшін ауыспалыларды пайдалану бағдарлама мәтінін қысқартып және жеңілдетіп қана қоймай, оның жұмысын айтарлықтай жылдамдату мүмкіндігін береді.

Объекттің ауыспалысын пайдалану әдетті ауыспалыларды пайдаланудан біршама ерекшеленеді. Бұл жағдайда мұндай ауыспалыны жариялап қана қоймай, пайдаланудан бұрын Set арнайы оператор көмегімен тиісті объектті тағайындау керек.

Мұндай жариялым мен тағайындаудың синтаксисі:

Dim<АуыспалыныңАты>AsObject
Set<АуыспалыныңАты> = <ОбъектКеСілтеме>

Кейде осындай ауыспалыны жариялау кезінде объекттің нақты түрін көрсету қолайлы. Бұл ретте Office объекттер моделінен кез-келген нақты объектті пайдалануға болады. Мысалы:

DIM MYBASE AS DATABASE
Set MyBase = DBEngine.Workspaces(0).Databases(0)

Осындай жариялау және тағайындаудан кейін ағымдағы ашық дерекқорға жүгіну үшін MyBase ауыспалысын пайдалануға болады. Мұндай сілтеме жылдам өңделеді және ауыспалыларды көптеген нақтылау (нүктелер) операторын пайдаланатын, күрделі иерархиялық сілтемелердің орнына объекттерге тікелей сілтемелер үшін пайдаланатын бағдарлама жылдам жұмыс істейді.

Массивтер. Массив — бұл бір уақытта бірнеше бірдей мән сақталатын ауыспалы. Массивті формальды анықтау - бұл бір типті индексацияланған ауыспалылардың жиынтығы.

Массивтің пайдаланылатын индекстерінің саны әртүрлі болуы мүмкін. Жиі бір немесе екі индексі бар массивтер пайдаланылады. VBA-да 60 индекс пайдалануға болады. Массивтің индекстер саны туралы массивтің көлемдігі туралы ретінде айтады. Бір индексі бар массивтербір өлшемді, екі индексі бар - екі өлшемді және т.б. деп аталады.

Массивті пайдаланудан бұрын міндетті түрде Dim операторының көмегімен жариялап, массивте сақталатын мәндер түрін көрсету керек. Массивтегі барлық мәндер бір деректер түріне жату керек. Массивті жариялау кезінде Variant түрін пайдалана отырып, осы шектеуден өтіп кетуге болады. Мұндай жағдайда массив элементтері әртүрлі типті деректердің мәндерін қабылдай алады.

Массивті жариялау операторының синтаксисі:

Dim <МассивтіңАты> (<көлем1>, <көлем2>, ...) As <ДеректерТүрі>

Жақшада көрсетілген шамалар массив көлемдерін белгілейді - индекстер санын және әрбір нақты индекстің барынша рұқсат етілетін мәнін белгілейді. Бұл ретте массив элементтерін индекстеу өздігінен нөлден басталады. Сонымен, жариялым

DIM ARRAY (9) AS INTEGER

бүкіл типтің ауыспалысы болып табылатын 10 элементтің бір өлшемді массивін айқындайды, ал жариялым

DIM ARRAY (4, 9) AS VARIANT

әмбебап Variant типтің ауыспалылары болып табылатын елу (5 х 10) элементінен ек өлшемді массивті айқындайды.

Массивті жариялау кезінде индекстің жоғарғы шегін ғана емес, сондай-ақ төменгі шегін көрсетуге болады, яғни нақты индекс массивінің өзгеру диапазонын айқын белгілеуге болады, бұл ретте төменгі шегі кез-келген бүтін сан болуы мүмкін. Осындай анықтаманың синтаксисі:

Dim <МассивтіңАты> (<мин1> To <макс1>,) As <ДеректерТүрі>

Келтірілген мысалдарда тіркелген көлемнің массивтері туралы айтылды, олардың элементтерінің саны массивті сипаттау кезінде Dim операторында айқын көрсетілген. Мұндай массивтер статикалық деп аталады. VBA-да көлемдері сипаттау кезінде тіркелмейтін, динамикалық массивтерді де пайдалануға болады. Динамикалық массивтің көлемін анықтау тікелей бағдарламаны орындау кезінде жасалуы мүмкін.

Динамикалық массивті анықтау кезінде Dim операторында массивтің атынан кейін бос жақшалар және ауыспалы түрінің сипаттамасы тұру керек. Индекстер саны мен оларды өзгерту диапазоны белгіленбейді. Массивті пайдаланудан бұрын ReDim операторын енгізу керек, ол динамикалық массив индекстерінің көлемдігін және өзгеру диапазондарын белгілейді.

Динамикалық массивтің көлемдерін анықтау және жариялау синтаксисі:

Dim <МассивтіңАты> () As <ДеректерТүрі>
ReDim <МассивтіңАты> (<көлем1>, <көлем2>, ...)

Динамикалық массивті жариялау және көлемдерін анықтау мысалын, содан кейін осы массивтің одан арғы көлемінің өзгеру мысалын келтірейік:

```
Dim dArray () As Variant
ReDim dArray (1, 2)
dArray (0, 0) = 2
dArray (0, 1) = 3
k = dArray (0, 0) + dArray (0, 1)
ReDim dArray (k)
dArray (0) = (Жол1)
```

Осы мысалда dArray массиві алдымен алты элементтен тұратын екі өлшемді массив ретінде айқындалады, содан кейін бір өлшемді массив ретінде қайта айқындалады, бұл ретте индекстің жоғарғы шегі k ауыспалының мәнімен белгіленеді.

8.4. Ауыспалылар мен рәсімдердің әрекет ету аясы

VBA-дағы бардық рәсімдердің, қызметтердің, ауыспалылар мен константалардың өз әрекет ету аясы бар. Бұл барлығының бағдарламалық кодтың белгіленген жерінде - сипатталған жерінде пайдаланылуы мүмкін екенін білдіреді. Мысалы, егер А ауыспалы Procl рәсімінің деңгейінде Dim операторының көмегімен сипатталған болса, онда дәл осы рәсім оның әрекет ету аясы болып табылады. Егер басқа Proc2 рәсімі болса, онда осы ауыспалыны жариялымсыз пайдалануға болмайды.

Ауыспалылардың әрекет ету аясы. Ауыспалының бағдарламаның қай жерінде және қалай сипатталғаны, жадқа қаншалықты ұзақ "өмір сүретіні" және тағайындалған мәнін сақтайтыны оның әрекет ету аясы айқындайды. Ауыспалылардың әрекет ету аясын айқындау кезінде үш түрлі деңгей бар: рәсім деңгейі, модуль деңгейі және жоба деңгейі.

Рәсім деңгейіндегі ауыспалыны айқындау үшін, оның сипаттамасы осы рәсімнің деңгейіне орналасады.

Модуль деңгейіндегі рәсімді айқындау үшін және оны осы модульдің барлық рәсімінде ортақ пайдалану үшін ең қолжетімді ету үшін оның сипаттамасын модульдерді жариялау секциясына - қандай да бір рәсімдердің немесе қызметтердің мәтінінің алдына орналастырған жөн. Бұл ретте әрекет ету аясының айқын сипаттамасы пайдаланылуы мүмкін. Бұл жағдайда «Dim» өзекті сөзінің орнына, «Private» өзекті сөзі пайдаланылады.

Жоба деңгейіндегі ауыспалыны сипаттау үшін, оның сипаттамасын жоба модульдерінің біріндегі жариялым секциясына орналастыру керек; бұл ретте міндетті түрде «Public» өзекті сөзі пайдаланылу керек.

Осындай тәсілмен сипатталған ауыспалылар жобаның кез-келген модулінде пайдаланылуы мүмкін.

Жоғарыда айтылғанның барлығы константалар мен массивтердің әрекет ету аясының сипаттамасы мен анықтамасына жатады.

Ауыспалылар үшін оларды сипаттайтын, деңгейін өзгертпейтін, бірақ рәсім деңгейінде сипатталған ауыспалының мәнін жұмысы аяқталғаннан кейін де сақтау мүмкіндігін беретін тағы бір тәсіл бар.

Ол үшін статикалық ауыспалы ретінде айқындалатын Static ауыспалыны пайдаланған жөн. Осындай ауыспалы жадта бөлінген орынды және сипатталған әрі пайдаланылған рәсім аяқталғаннан кейін де өз мәнін сақтайды.

Бұған қарамастан статикалық ауыспалы басқа рәсімдерде пайдаланыла алмайды. Оның әрекет ету аясы емес, тек өмір сүру уақыты ғана өзгереді. Егер статикалық ауыспалы сипатталған рәсім қайта шақырылса, онда бұл ауыспалы бұдан бұрынғы шақыру кезінде рәсімнің жұмысы аяқталу сәтінде ие болған мәнін сақтайды. Әдетті (статикалық емес) ауыспалылар қайтадан инициалданады және рәсімге кіру кезінде бос мәндер алады.

Рәсімдер мен қызметтердің әрекет ету өрістері. Рәсімдер мен қызметтердің әрекет ету аясының тек екі деңгейі бар: модуль деңгейі және жоба деңгейі. Өздігінен жоба деңгейі пайдаланылады. Рәсім немесе қызмет жобасында кез-келген басқа рәсім немесе қызмет шақырылуы мүмкі. Жоба деңгейіндегі рәсімдер мен қызметтерді сипаттау кезінде міндетті емес «Public» өзекті сөзі пайдаланылуы мүмкін. Осы сөздің болуы немесе болмауы рәсімге әсер етпейді.

Егер тек модуль деңгейінде пайдаланылатын рәсімді сипаттау қажет болса, онда ол үшін «Private» өзекті сөзі қолданылады. Алайда рәсімнің осындай сипаттамасы рәсім үшін әрекет ету аясын қысқартады және жеке-дара пайдаланылуына тыйым салады. Оны тек басқа рәсімнен шақыруға болады.

Рәсімдерді немесе қызметтерді сипаттау кезінде «Static» өзекті сөзі пайдаланылуы мүмкін. Ол рәсімнің әрекет ету аясына әсер етпейді, бірақ осы рәсімнің немесе қызметтің ішінде сипатталған барлық ауыспалыға әсер етеді. Бұл жағдайда барлық жергілікті ауыспалы Static мәртебесіне ие болады және рәсім аяқталғаннан кейін жадта сақталады; қайтадан шақыру кезінде бұрынғы мәндері сақталады.

Келесі мысалды қарастырайық.

```
Public A1 As String
Private A2 As Integer
Dim A3 As Single
Sub Proc1()
```

```
    Dim A4 As Integer Static
    A5 As Integer A1 =
    "Мәтіндік жол"
    A2 = 2 A3 =
    3.14 A4 =
    A4 + 4 A5
    =A5 + 5
    MsgBox A4
    MsgBox A5
End Sub
Sub Proc2()
    Proc1
    MsgBox
    A1
    MsgBox
    A2
    MsgBox
    A3
    MsgBox
    A4
    MsgBox
    A5 Proc1
End Sub
```

Осы мысалда ауыспалы A1 бүкіл жоба деңгейінде айқындалған («Public» өзекті сөзі пайдаланылды), ал ауыспалы A2 және A3 модуль деңгейінде, ал ауыспалы A4 - Procl рәсімінің деңгейінде ғана, ауыспалы A5 Procl рәсімінің деңгейінде айқындалғанымен, статикалық ауыспалы ретінде сипатталған.

Рroc2 рәсімін шақырған кезде, осы рәсімнен A1, A2, A3, A4 және A5 барлық бес ауыспалыға мәндерді тағайындайтын, содан кейін A4 және A5 ауыспалылардың ағымдағы мәндерін сұхбат терезеге шығаратын *(MsgBox),* Procl рәсімі шақырылады.

Осы рәсім аяқталғаннан кейін РГОС2 рәсімінен барлық бес ауыспалының ағымдағы мәндері шығарылады. Бұл ретте келесі жағдай болады. Ауыспалы A1...A3 өз мәндерін сақтайды, себебі олар модуль деңгейінде сипатталған, ал ауыспалы A4 және A5 мәндері 0 болады (бос мәндер), себебі осы ауыспалылардың әрекет еті аясы тек Procl рәсімі болып табылады.

Содан кейін Procl рәсімі қайта шақырылады және қайтадан ауыспалы A4 және A5 мәндерін өзгертіп, экранға шығара бастайды. Бұл ретте ауыспалы A4 қайтадан 4 мәніне ие болады, себебі осы ауыспалы үшін рәсімді жаңа шақырған кезде жаңадан жад бөлінеді және ол бос мәнмен инициалданады. A4-ке қарағанда статикалық ауыспалы ретінде сипатталған ауыспалы A5 осы рәсімді бұдан бұрын шақырған кездегі мәнін сақтайды және нәтижесінде қайтадан шақыру кезінде оның мәні 10-ға тең болады.

8.5. Басқарушы конструкциялар - тармақтар және циклдер

Бағдарламалаудың барлық тілдеріндегідей, VBA-да бағдарламаны орындау тәртібін өзгерту мүмкіндігін беретін, әртүрлі басқарушы конструкциялар бар. Егер басқарушы конструкциялар пайдаланылмаса, онда ең біріншісінен бастап ең соңғысымен аяқтай отырып, бағдарламалау тілінің операторының ретімен орындалуы іске асырылады. Кейбір ең қарапайым жағдайларда жеткілікті болғанымен, кейде кейбір операторлардың орындалуын өткізе отырып немесе оларды көп рет қайталай отырып, белгілі бір шарттарды орындау кезінде операторлардың орындалу тәртібін өзгерту керек.

Тәжірибе көрсеткендей, ақпаратты өңдеудің кез-келген алгоритмдерін іске асыру үшін басарудың екі түрлі конструкциясының болуы жеткілікті: тармақтар және циклдер.

Тармақтар. Тармақтардың басқарушы конструкциялары кейбір жағдайды тексеру мүмкіндігін береді, содан кейін осы тексеру нәтижелеріне тәуелді қандай да бір операторды орындау мүмкіндігін береді. Тармақтарды ұйымдастыру үшін VBA-да If тармақтар операторының әртүрлі нысандары және Select Case таңдау операторы пайдаланылады.

If операторының қарапайым қысқа нысаны бір жағдайды тексеру үшін пайдаланылады, содан кейін тексеру нәтижесіне тәуелді не орындау үшін, не бір немесе бірнеше операторды өткізу үшін пайдаланылады. If тармағының операторының қысқаша нысаны бір жолы және блок нысанына ие болуы мүмкін. Бір жолған If қысқаша нысаны төмендегідей жазылуы мүмкін:

If <шарт>Then <оператор>

Блок нысанында қысқаша тармақ төмендегідей болады:

```
If<шарт>Then
<оператор1>


<операторN>
EndIf
```

Шарт ретінде *True*(Шындық) немесе *False*(Жалған) мәнін қайтаратын логикалық тіркесті немесе кез-келген арифметикалық тіркесті пайдалануға болады. Егер арифметикалық тіркес пайдаланылса, онда осы тіркестің нөлдік мәні логикалық *False мәніне, ал кез-келген нөлдік емес мәні True мәніне баламалы. Шарт False мәнін қайтарған жағдайда, оператор немесе қысқаша тармақтар операторының денесін құрайтын және* «Then» және «End If» өзекті сөздерімен жасалған операторлар блогы орындалмайды.

Ескертпе.Қысқаша тармақталу операторын бір жолға жазған кезде «End If» өзекті сөздері пайдаланылмайды.

If операторының толық нысаны екі түрлі операторлар блогы болған жағдайда пайдаланылады және шартты тексеру нәтижелері бойынша олардың біреуін орындау керек. Осындай If нысаны бір жолға жазыла алмайды және жазбаның блокты нысанына ие:

If<шарт>Then<Опера
торлар Блогы1>
ELSE
<ОператорларБлогы2>
EndIf

Егер шарт шындық болса, онда «Then» және «Else» өзекті сөздерінің арасына алынған бірінші операторлар блогы орындалады; әйтпесе - «Else» және «EndIf» сөздерінің арасына алынған екінші блок орындалады.

Кейде бірнеше әртүрлі шартты тексеру негізінде баламалы әрекеттердің біртұтас тобынан бір әрекетті таңдауға тура келеді. Ол үшін If...Then...Else If тармақталу операторларының тізбегін пайдалануға болады:

If <шарт1>Then
<ОператорларБлогы1>
Else If <шарт2>Then
<ОператорларБлогы2>
Else If <шарт3>Then
<ОператорларБлогы3>

Else If <шартN> Then
<ОператорларБлогыN>
Else
<ОператорларБлогы_Else>
End If

If...Then...Else If операторларының осындай тізбектерінің иілімдігі жоғары және көптеген мәселелерді шешу мүмкіндігін береді, алайда егер бірнеше мүмкіндіктің біреуін таңдау әрқашан бір тіркестің әртүрлі мәніне негізделген болса, ол үшін арнайы Select Case таңдау операторын пайдалану қолайлы, оның синтаксисі төмендегідей:

Select Case <тексерілетінТіркес>
Case <мәндерТізімі1>
<операторларБлогы 1>
Case <мәндерТізімі2>
<операторларБлогы2>

Case
<мәндерТізіміЗ><оператор

Case Else
<операторларБлогы_Еке>
End Select

Тексерілетін тіркес Select Case операторының жұмысы басталған кезде есептеледі. Осы тіркес кез-келген типті мәнді қайтара алады.

Тіркестер тізімі үтірмен бөлінген, бір немесе бірнешетіркес болып табылады. Операторды орындау кезінде осы тізімнің ең болмағанда бір элементі тексерілетін тіркеске сәйкес келуі тексеріледі. Тіркестер тізімінің осы элементтерінің мынадай нысаны болуы мүмкін:

<тіркес>

(бұл жағдайда тексерілетін тіркес мәнінің белгіленген тіркеске сәйкес келуі тексеріледі);

<тіркес1> To <тіркес2>

(бұл жағдайда тексерілетін тіркес мәнінің көрсетілген мәндер диапазонында орналасуы тексеріледі);

If <логикалықОператор><тіркес>

(бұл жағдайда тексерілетін тіркес логикалық оператор көмегімен көрсетілген мәнмен салыстырылады; мысалы, егер тексерілетін мән 20-дан кем болса, If >= 20 шарты орындалған болып есептеледі).

Егер тізімдегі элементтің ең болмағанда біреуі тексерілетін тіркеске сәйкес келсе, онда тиісті операторлар тобы орындалады және осымен Select Case операторының орындалуы аяқталады, ал қалған тіркес тізімдері тексерілмейді. Егер барлық осы тізімнің бірде бір элементі тексерілетін тіркестің мәніне сәйкес келмесе, онда болған жағдайда Else тобының операторы орындалады.

Циклдер. VBA-да екі топқа бөлуге болатын, циклдерді ұйымдастыру құралдарының жеткілікті жинағы бар:

- Do...Loop шартымен циклдер;
- For.Next атаумен циклдер.

Do...Loop типті циклдер цикл денесін құрайтын, операторлар блогының орындалуы қанша рет қайталануы қажет белгісіз болған жағдайда пайдаланылады. Мұндай цикл белгілі бір шарт орындалғанға дейін өз жұмысын жалғастырады. Do...Loop циклдерінің төрт түрі бар, олар тексерілетін шарт түрімен және осы тексерудің орындалу уақытымен ерекшеленеді. Осы төрт цикл конструкциясының синтаксисі 8.2-кестеде келтірілген.

Кесте 8.2. Цикл конструкциясының синтаксисі

Конструкция	Сипаттамасы
DoWhile<шарт><операторларБлогы>Loop	Осы конструкцияда циклді орындау шартының мәні операторлар орындалғанға дейін тексеріледі
Do<операторларБлогы>LoopWhile<шарт>	Осы конструкцияда циклді орындау шартының мәні операторлар орындалғаннан кейін тексеріледі
DoUntil<шарт><операторларБлогы>Loop	Осы конструкцияда оператор орындалғанға дейін белгіленетін, циклдің жұмысын аяқтау шартының мәні көрсетіледі
Do <операторларБлогы> LoopUntil<шарт>	Осы конструкцияда оператор орындалғаннан кейін белгіленетін, циклдің жұмысын аяқтау шартының мәні көрсетіледі

Сондай-ақ For...Next атап шығуымен цикл операторының екі түрі бар. Массивті өңдеу кезінде, сонымен қатар кейбір операторлар тобының орындалуын белгіленген сан рет қайталау қажет болғанда, есептеуішпен For...Next циклі пайдаланылады. Do...Loop циклдеріне қарағанда циклдің осы түрі есептеуіш деп аталатын арнайы ауыспалыны пайдаланады, оның мәні цикл денесін белгіленген шамаға орындаған сайын артады немесе азаяды. Осы ауыспалының мәні белгіленген мәнге жеткенде, циклдің орындалуы аяқталады. Осы циклдің синтаксисі (тік жақшаға міндетті емес элементтер алынған):

```
For <есептеуіш> = <бастапқыМәні>To <соңғыМәні>[Step <өсіру>]
    <операторларБлогы>
Next <есептеуіш>
```

Келтірілгенге бірнеше ескерту келтірейік: цикл конструкциясының фрагментіне:

• <өсіру> оң және теріс сан болуы мүмкін. Егер теріс өсіруді пайдалансақ, онда цикл денесі ең болмағанда бір рет орындалу үшін соңғы мән бастапқы мәнге тең немесе кем болу керек;

• For.Next цикл жұмысы аяқталғаннан кейін есептеуіш ретінде пайдаланылған ауыспалы, егер өсіру оң болса, соңғы мәннен жоғары мәнге ие болады, ал егер өсіру теріс болса, соңғы мәннен кем мәнге ие болады;

• егер бастапқы және соңғы мән сәйкес келсе, онда цикл денесі тек бір рет орындалады.

For...Next циклінің тағы бір түрін қарастырайық, ол объекттерді өңдеу кезінде VBA-да жиі пайдаланылады, біртекті объекттердің массивін немесе тегін құрайды.

For Each.Next циклінің осы түрінде есептегіш жоқ, ал цикл денесі әрбір объекттің массиві немесе тегі үшін орындалады. Осындай циклдің синтаксисі:

```
For Each <элемент>In <жиынтық>
<операторларБлогы>
Next [<элемент>]
```

мұндағы<элемент> — объекттер тегінің элементтеріне сілтеу үшін пайдаланылатын ауыспалы; <жиынтық> — бұл массивтің немесе тегінің аты.

Ұқсас циклдің пайдаланылу мысалын келтірейік. Келесі рәсім ағымдағы ашық дерекқордың барлық кестелерінің барлық өрістерінің тізімін басып шығаруға арналған:

```
Public Sub EnumerateAllFields ()
Dim MyBase As Database
Dim tdf As TableDef, fid As Field
Set MyBase = DBEngine.Workspaces(0).Databases(0)
For Each tdf In MyBase.TableDefs
Debug.Print "Кесте "& tdf.Name For
Each fid In tdf. Fields Debug. Print
"Поле: "& fid. Name Next fid NEXT
TDF Set MyBase = Nothing End Sub
```

Осы мысалда Dim операторлары келесі ауыспалыларды жариялады:

- MyBase — «Дерекқор» объектісі;
- tdf — дерекқор кестесі;
- fid — кесте аясы.

Set операторы MyBase ауыспалыға ағымдағы ашық дерекқор атының мәнін тағайындайды. Одан әрі кестенің әрбір анықтамасы үшін кестенің аты басып шығарылады, содан кейін осындай типті цикл барлық аяның атын басып шығарады.

8.6. Циклдер мен рәсімдерден шығу

Әдетте рәсімнің орындалуы соңғы оператордан кейін аяқталады, ал циклдің орындалуы - жұмысын аяқтау шартына қол жеткізгеннен кейін аяқталады.

Алайда жиі кейбір рәсім операторларын немесе циклдің қайталануын орындамай, рәсімнің немесе циклдің орындалуын мерзіміне дейін тоқтату қажет болады.

Мысалы, егер рәсімді орындау кезінде жұмыстың жалғасуын мәнсіз ететін қателік туындаса, онда дереу рәсімнен шығу командасын орындауға болады.

Басқа мысал: егер For...Next циклі массивтегі қажетті мәнді іздеу үшін пайдаланылса, онда массивтің қажетті элементі табылғаннан кейін массив элементтерін одан әрі қараудың мәні жоқ. Exit операторының біреуі арқылы басқарушы конструкциядан мерзімінен бұрын шығуға болады. Do.Loop циклдерінен мерзімінен бұрын шығу үшін Exit Do операторы, ал For циклдерінен шығу үшін - Exit For операторы пайдаланылады. Рәсімдер мен қызметтерден мерзімінен бұрын шығу үшін Exit sub және Exit Function тиісті операторлары пайдаланылады.

Exit операторының пайдаланылуы орынды болғанымен, оны аса қажет болғанда ғана пайдаланған жөн. Осы операторды шамадан тыс жиі қолданылуы жазылған мәтіннің түсіндірілуін және оның ретке келтірілуін күрделендіреді.

Келесі мысалдарды қарастырайық:

```
ub = Ubound (dArrey)
fFound = False
For I =LBound (dArrey) to ub If
dArrey(i) = searchValue Then fFound
= True Exit For NEXT
```

Exitоператорының пайдаланылуына жол бермеуге болады. Бұл келесі мысалда көрсетілген:

```
i = LBound (dArrey) ub
= Ubound (dArrey)
fFound = False Do
    If dArrey(i) = searchValue Then fFound = True i = i
    + 1
Loop Until (i > ub) or fFound
```

8.7. Модульдер

Модуль— бұл бір бағдарламалық бірлікке жиналған қосымшаларға арналған VisualBasic тіліндегі сипаттамалар мен рәсімдер жинағы. Модульдің екі негізгі түрі бар: класс модулі және стандартты модульдер. Модульдегі әрбір рәсім Function рәсім-қызметі немесе Sub рәсім-қосалқы бағдарламасы болуы мүмкін.

Класс модульдері. Класс модульдері белгілі бір нысанмен немесе есеппен байланысқан бағдарламалар болып табылады. Класс модульдері нысандағы немесе есептегі оқиғаға жауап ретінде іске қосылатын, оқиғаларды өңдеу рәсімдерін құрайды. Оқиғаларды өңдеу рәсімдері мысалы, пернеге басу сияқты нысанның немесе есептің тәртібін және олардың оқиғаға жауап беру тәртібін басқару үшін пайдаланылады.

Нысан немесе есеп үшін оқиғаны өңдеудің бірінші рәсімін құру кезінде автоматты түрде онымен байланысты нысанның немесе есептің модулі жасалады. Нысанға немесе есепке арналған модульді қарау үшін

Құрастырушы режиміндегі құрал-саймандар панеліндегі Бағдарлама пернесіне басу жеткілікті.

Нысандар мен есептердің модульдерінің рәсімдерінде стандартты модульдерге қосылған рәсімдерді шақыру болуы мүмкін.

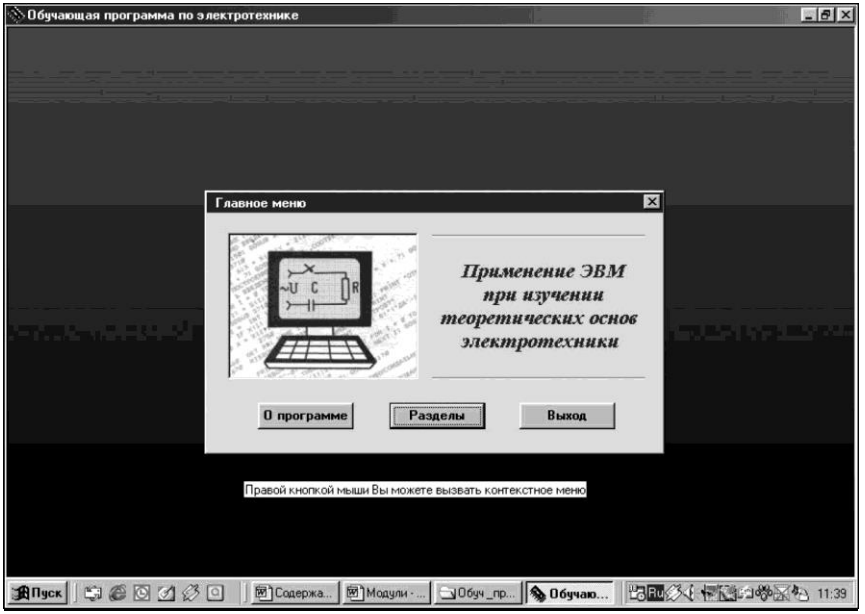
Стандартты модульдер. Стандартты модульдер ешқандай объектке байланысты емес жалпы рәсімдерді, сонымен қатар дерекқордың кез-келген терезесінен іске қосылуы мүмкін рәсімдерді құрайды.

Стандартты модульдер ғаламдық ауыспалыларды жариялау үшін жасап шығарылуы мүмкін. Егер модульдің рәсімдерінде қосымшаның нақты объекттеріне (нысандар, есептер, басқару элементтері) сілтемелер көрсетілмесе, онда осындай модуль Access басқа қосымшаларында қолданылуы мүмкін.

8.1, *а*- суретте басты нысан, 8.1-суретте, *б* — электртехниканың теориялық негіздері бойынша оқу құралының электронды нұсқасында оқыту бағдарламасымен жұмыс істеу режимдерінің пернелі нысаны көрсетілген.

Төменде 8.1-суретте көрсетілген, оқу құралының осы және басқа нысандарымен жұмысын басқаратын модульдің мәтіні келтірілген.

```
'Осы модуль жалпы мақсатты әртүрлі рәсімді сақтайды
'Жасап шығарушылар: Фуфаев Э.В. және Фуфаева Л.И.
Option Explicit
'Төменде нысандар арасындағы жағдай параметрлерінің берілуі
үшін жауап беретін ауыспалылар келтірілген'.
Public ShowMax As Boolean
Private Default As Boolean
Private FrmBackColor As Long
'Рәсім барлық нысандардың бағдарламада инициалдануын, экран
орталығында позициялануын іске асырады.
Public Sub InitForm(TForm As Form, TPicture As PictureBox, TFrame
As Frame, TButton As CommandButton, TSSubItem As Menu) TForm.Left
= 0 TForm.Top = 0
TForm.Width = Screen.Width '12000
TForm.Height = Screen.Height ' - 430 '8570
```



а

б

8.1-сур. Нысандар:
а — басты; *б* — оқыту бағдарламасымен жұмыс істеу режимдерін таңдаудың пернелі нысаны

```
TPicture.Left = 0 TPicture.Top = 0
TPicture.Width = Screen.Width ' - 15 * 6 TPicture.Height =
Screen.Height ' - 15 * 25 TPicture.BackColor = FrmBackColor If
ShowMax Then TForm.WindowState = 0 If Not ShowMax Then
TForm.WindowState = 2 If Default Then Lines TPicture
TButton.Left = (TPicture.Width - TFrame.Width * 15)/30
TButton.Top = (TPicture.Height - TFrame.Height * 15 - 750)/30
TFrame.Left = TButton.Left + 2
TFrame.Top = TButton.Top + 2
If Default Then TSSubItem.Checked = True
If Not Default Then TSSubItem.Checked = False
End Sub
'Түстің өтуін пайдалана отырып, басты нысанның фонын салу
рәсімі.
Public Sub Lines(Picture As PictureBox)
Dim i, J, Stp, col As Integer J = 0 col = 255 Stp = Step
For i = 0 To (Picture.Height\15)
J = J + 1
If (J = Stp) And (col <> 1) Then col = col - 1 J = 0 End If
Picture.Line (Picture.Left, i)-(Picture.Left + Picture.Width, i), RGB(0, 0,
col)
Next End Sub
'Фон түсінің өзгеру қадамын есептейтін қосалқы рәсім
Public Function Step() As Integer Dim i, Col_ As Integer i = 0
Col_ = 0
Do Until Col_ > (Screen.Height\15)
Col_ = Col_ + 255
i = i + 1
Loop
If Col_ - (Screen.Height \ 15) > 100 Then i = i - 1 Step = i End Function
'Тізілімнен бағдарлама параметрлерін оқу
```

```
    Public Sub GetFromRegister()
    Dim Ret As Long
    Ret = GetSetting ("Book", "Init", "BackColor",
vbApplicationWorkspace)
"7303023)
    FrmBackColor = Ret
    Ret = GetSetting("Book", "Init", "Default", 1)
    If Ret = "0" Then Default = False
    If Ret = "1" Then Default = True
    Ret  =  GetSetting("Book",  "Init",
    "Size", 2)
    If Ret = "0" Then ShowMax = True
    If Ret = "2" Then ShowMax = False
    End Sub
    Бағдарлама параметрлерін реестрде сақтау
    PublicSubSavetoRegister()
     SaveSetting          "Init    "BackColor", FrmBackColor
     "Book",  If  Default   ",
     SaveSetting          "Init    "Default", "1"
     "Book", Else         ",
     SaveSetting          "Init    "Default", "0"
     "Book", End If       ",
     If ShowMax Then
     SaveSetting          "Init    "Size", "0"
     "Book", Else         ",
     SaveSetting          "Init    "Size", "2"
      End If End Sub
      'Басты нысанның мәнмәтіндік мәзірінің бірінші тармағының
 таңдауын өңдеу
      Public Sub Item1Clk(ColorDialog As CommonDialog, TPicture As
 PictureBox, TItem As Menu)
      On Error GoTo ErrHandler ColorDialog.Flags = cdlCCRGBInit
      ColorDialog.ShowColor FrmBackColor = ColorDialog.Color
      TPicture.BackColor = FrmBackColor TItem.Checked = False Default =
      False Exit Sub
      ErrHandler: Exit Sub End Sub
      'Басты нысанның мәнмәтіндік мәзірінің екінші тармағының
 таңдауын өңдеу
      Public Sub Item2Clk(TPicture As PictureBox, TItem As Menu)
      If TItem.Checked Then Default = False TItem.Checked = False
```

```
TPicture.BackColor = TPicture.BackColor
Else
Default = True
TItem.Checked = True
Lines TPicture End If
End Sub
```

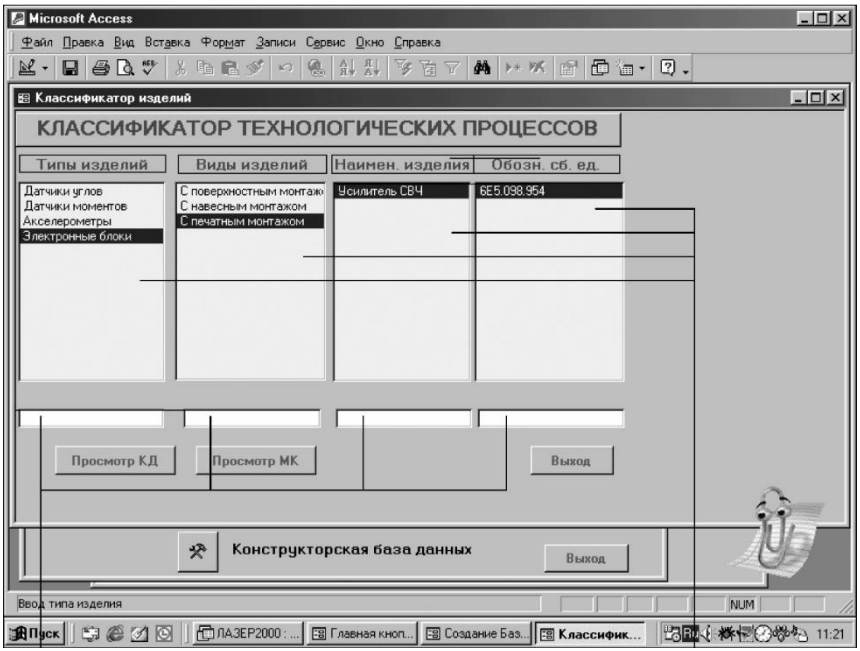
Сонымен, біз VBA бағдарламалау тілінің негізгі ұғымдарымен таныстық.

Access жүйесінде деректер жұмысты автоматтандыру үшін қолданудың кейбір тәсілдерін қарастырайық.

8.2-суретте бұйымның кәсіпорнында дайындалатын технологиялық үрдістерді жіктегішті жасау нысаны ұсынылған.

Жіктегіш құрама бірліктің белгіленуін жылдам іздеуге арналған (құрастыру сызбасының №).

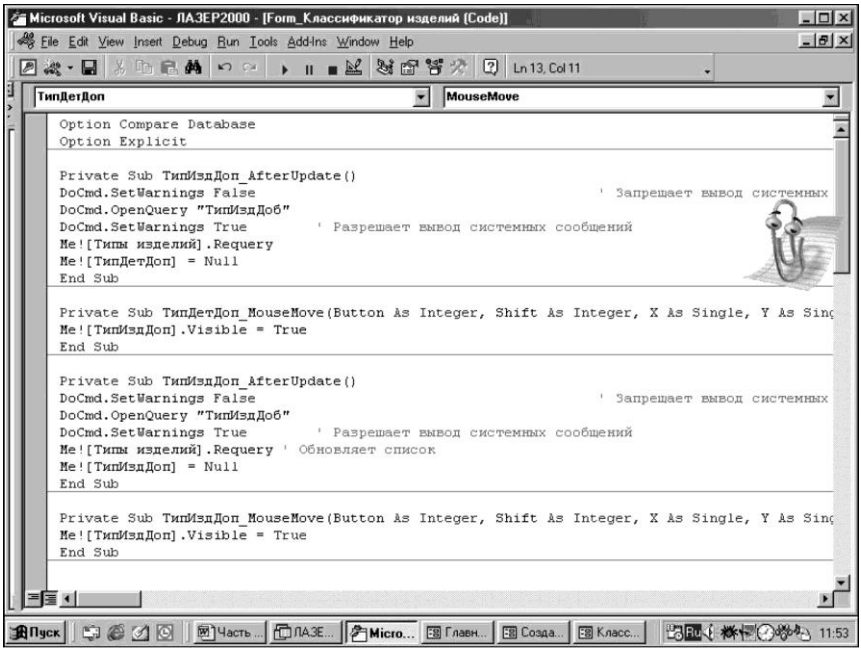
Осы жіктегіш “көбіне біреу” қатынастарымен байланысқан, төрт кесте (тізім) түрінде ұсынылуы мүмкін, төрт деңгейлі иерархиялық жүйе болып табылады: • бірінші кестеде бұйым түрлерінің атаулары бар;



Тізімге енгізу өрістері

Жіктеу топтарының байланысқан тізімдері

8.2-сурет. *Технологиялық үрдістерді жіктегіш* нысаны



8.3-сур. Бағдарлама фрагментімен *Бағдарламаларды құрушы* терезесі

- екінші кестеде әрбір түріне арналған бұйым түрлерінің атаулары;
- үшінші кестеде бұйымдардың атаулары, осы мысалда - нақты түріне жататын құрама бірліктердің атаулары;
- төртінші кестеде құрама бірлік сызбасының белгіленуі (нөмірі).

Жіктегішті толтыру алгоритмі мына әрекеттерден тұрады:

- меңзерді бірінші тізімнің деректерді енгізу аясына орналастыру (бұйым түрлері);
- түрінің атауын енгізу;
- бұйымның түрін ерекшелеу және меңгерді екінші тізімнің деректер енгізу аясына орналастыру — *Бұйым түрлері*;
- келесі тізімдерді толтыру үшін көрсетілген әрекеттерді қайталау керек.

Осы рәсімдерді орындау үшін VISUALBASIC тіліндегі бағдарламаны қарастырайық.

Бағдарламалар *Бағдарламаларды құрушы* терезесіндегі макростарға және тіркестерге ұқсас жасап шығарылады.

Бағдарлама мәтінінің фрагментімен *Бағдарламаларды құрушы* терезесі 8.3-суретте келтірілген. Бағдарламаны жасап шығару реті:

- *Құрастырушы режимінде нысанды ашу*;

- объектті (аяны) ерекшелеу;
- *Объекттің қасиеттері* терезесін шақыру.

Қасиеттер терезесінде тиісті қасиеттің жолында *Бағдарламаларды құрушыны* шақыру керек. Нәтижесінде бастапқы тіл операторларымен *Бағдарламаларды құрушы* шығады:

Private Sub — жаңа қосалқы бағдарламаны жариялайтын нұсқаулық. (Бағдарлама мәтіні)

End Sub — қосалқы бағдарламаны жабатын нұсқаулық.

Төменде *Технологиялық үрдістерді жіктегіш* нысанын толтыру кезінде орындалатын қосалқы бағдарламалардың мәтіндері келтірілген (8.2-сур.).

```
Option Compare Database
Option Explicit
Private Sub ҚосБұйымТүрі_AfterUpdateO' Бұйымдар түрі тізімін толықтыру бағдарламасының басы
DoCmd.SetWarnings False ' Жүйелік хабарламаларды шығаруға тыйым салады
DoCmd.OpenQuery "ҚосБұйымТүрі"
DoCmd.SetWarnings True ' Жүйелік хабарламаларды шығаруға рұқсат етеді
Me^^mi бұйымдар].Requery
Me![ҚосБұйымТүрі] = Null End
Sub

Private Sub ҚосБұйымТүрі_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Me![ҚосБұйымТүрі].Visible =
True End Sub

Private Sub ҚосБұйымТүрі_AfterUpdateO
DoCmd.SetWarnings False ' Жүйелік хабарламаларды шығаруға тыйым салады
DoCmd.OpenQuery "ҚосБұйымТүрі"
DoCmd.SetWarnings True ' Жүйелік хабарламаларды шығаруға рұқсат етеді
Me^^mi бұйым].Requery ' Тізім жаңартылады
Me![ҚосБұйымТүрі] = Null End Sub

Private Sub ҚосБұйымТүрі_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Me![ҚосБұйымТүрі].Visible =
True End Sub

Private Sub ҚосБұйымТүрі_AfterUpdateO
```

```
    DoCmd.SetWarnings False ' Жүйелік хабарламаларды шығаруға
тыйым салады
    DoCmd.OpenQuery "ҚосБұйымТүрі"
    DoCmd.SetWarnings True ' Жүйелік хабарламаларды шығаруға
рұқсат етеді
    Me![БұйымТүрлері].Requery ' Тізімді
    ЖаңартадыMe![ҚосБұйымТүрі] = Null End Sub
    Private Sub МидИздДоп_MouseMove(Button As Integer, Shift As
Integer, X As Single, Y As Single)
    Me![ҚосБұйымТүрі].Visible = True End Sub

    Private Sub БұйымАтауы_AfterUpdateO
    Me![БұйымАтауы].Requery
    End Sub

    Private Sub БұйымАтауы_MouseMove(Button As Integer, Shift As
Integer, X As Single, Y As Single)
    Me![БұйымАтауы].SetFocus End Sub

    Private Sub БұйымТүрлері_AfterUpdateO
    Me![БұйымТүрлері].Requery Me![БұйымАтауы].Requery
    Me![БұйымБелг].Requery End Sub

    Private Sub БұйымТүрлері_MouseMove(Button As Integer, Shift As
Integer, X As Single, Y As Single)
    Me![^nbi бұйым].SetFocus End Sub

    Private Sub БұйымБелгMouseMove(Button As Integer, Shift As
Integer, X As Single, Y As Single)
    Me![БұйымБелг].SetFocus End Sub

    Private Sub ҚосБұйымТүрі_AfterUpdateO
    DoCmd.SetWarnings False ' Жүйелік хабарламаларды шығаруға
тыйым салады
    DoCmd.OpenQuery "ҚосБұйымТүрі"
    DoCmd.SetWarnings True ' Жүйелік хабарламаларды шығаруға
рұсқат етеді
    Me![БұйымТүрлері].Requery ' ТізімдіЖаңартады
    Me![ҚосБұйымТүрі] = Null End Sub

    Private Sub ҚосБұйымТүрі_MouseMove(Button As Integer, Shift As
Integer, X As Single, Y As Single)
```

Me![ҚосБұйымТүрі]^шЫе = True End Sub

Private Sub БұйымТүрлеріAfterUpdateO Me![БұйымАтауы].Requery Me![БұйымБелг].Requery End Sub

Private Sub БұйымТүрлеріMouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Me![БұйымТүрлеріЭД-SetFocus End Sub

Бақылау сұрақтары

1. Макростар қандай кластарға бөлінеді?
2. Нысандағы пернеге “басу” кезіндегі әрекеттерді орындауға арналған макростарды жасау тәсілдерін атаңыз.
3. Макрос *құрастырушын* пайдалана отырып макрос жасау кезіндегі әрекеттердің реттілігі қандай?
4. Ашық дерекқор үшін арнайы құрал-саймандар панелін жасау кезіндегі әрекеттердің реттілігін қалыптастырыңыз.
5. Белсенді дерекқор үшін арнайы мәнмәтіндік мәзірді жасау кезінде әрекеттердің қандай реттілігі қажет?
6. Пернелі нысандар мен бас бет-нысандары не үшін?
7. VISUALBASIC тілінде жазылған модуль - қосалқы бағдарламалар қандай жағдайда жасап шығарылады?
8. Access анықтамалық жүйесін пайдалана отырып, бағдарлама мәтінінің келтірілген фрагменттеріндегі тіркестердің тағайындалуын өз бетімен талдаңыз:

DoCmd.OpenQuery"ҚосБұйымТүрі"
Me![ҚосБұйымТүрі] = Null
Me![БұйымБелг].SetFocus

9. SQL операторларын орындау үрдісі қандай кезеңдерден тұрады?
10. Сақталатын рәсімдер қандай қызметті атқарады және SQLServer-де қандай бағдарламалау тілінде жазылады?
11. Триггерді жазу үшін қандай команда қолданылады?
12. Сұраныстарды оңтайландыру үрдісі нені білдіреді?
13. WITHENCRIPTING параметрі нені білдіреді?
14. Триггерлер опереторларының құрамын шектейтін ережелерді атаныз.

9 тарау

КІРІКТІРІЛГЕН SQL ТІЛІ

9.1. Кіріктірілген SQL тілінің тағайындалуы және ерекшеліктері

8 тарауда біз VBA ортасында басқарушы бағдарламаларды жасап шығару мүмкіндіктері мен технологиясын баяндадық, ал осы тарауда SQL тілінің құралдарымен дерекқорды басқарудың кейбір мүмкіндіктерін қарастырамыз.

SQLтілі (5 тарауды қараңыз) дерекқорға қолжетімдікті ұйымдастыруға арналған. Бұл ретте ДҚ-ға қолжетімдіктің екі режимде жүзеге асырылу мүмкіндігі болжалады: интерактивті режимде және қолданбалы бағдарламаларды (қосымшаларды) орындау режимінде.

Осы екіжақтылықтың арқасында SQL келесі оң қасиеттерге ие:

• сұраныстардың интерактивті тілінің барлық мүмкіндіктері қолданбалы бағдарламалауда қолжетімді;

• интерактивті режимде одан әрі жұмыс істеп тұрған қосымшаларға орнатылуға дайын болатын, ақпаратты өңдеудің негізгі алгоритмдерін ретке келтіруге болады.

SQL, бағдарламалау тілі болғанымен, ерекше бағдарлығынан әмбебап бағдарламалау тілдерінің көптеген мүмкіндіктеріне ие болмайды. Онда циклдерді ұйымдастыратын, ішкі ауыспалыларды жариялау және пайдалану, кейбір шарттардың талдауын ұйымдастыру мүмкіндігін және шарттың орындалуына тәуелді бағдарлама барысын өзгерту мүмкіндігін беретін дәстүрлі операторлар жоқ. Жалпы SQL тек дерекқорды басқаруға арналған қосалқы тіл деп атауға болады. Қосымшаларды, шынайы бағдарламаларды жасау үшін SQL тілінің операторлары кіріктірілетін басқа негізгі бағдарламалау тілдерін пайдалану керек.

Бағдарламалаудың негізгі тілдері мына тілдер болуы мүмкін: C, COBOL, PL/1, Pascal. Қолданбалы бағдарламаларда бағдарламалау тілдерін қолданудың екі тәсілі бар:

• кіріктірілген SQL. Осындай тәсілмен SQLоператорлары негізгі тілде тікелей бағдарламаның бастапқы мәтініне кіріктіріледі. SQL операторлары кіріктірілген бағдарламаларды компиляциялау кезінде бастапқы мәтінді орындаушы бағдарламаға түрлендіретін, SQL арнайы препроцессор пайдаланылады;

• интерфейсті бағдарламалау тілдері (API — application program interface). Осы тәсілді пайдалану кезінде қолданбалы бағдарлама қызметтерді шақыра отырып, арнайы қызметтерді қолдану арқылы ДҚБЖ-мен өзара әрекеттеседі.

SQL операторларын орындау үрдісі шартты бес кезеңге бөлінуі мүмкін (9.1-сур.).

Бірінші кезеңде SQL операторына синтаксистік талдау жүргізіледі. Осы кезеңде синтаксис ережелеріне сәйкес SQL операторының жазбаларының дұрыстығы тексеріледі.

Екінші кезеңде SQL операторының параметрлерінің: қатынас аттарының, деректер аясы аттарының, аталған объекттермен жұмыс істеу бойынша пайдаланушының артықшылықтарының дұрыстығы тексеріледі.

Үшінші кезеңде сұрату оңтайландырылады. ДҚБЖ біртұтас сұранысты минималды операциялар қатарына бөледі және сұранысты орындауға уақытша шығындар тұрғысынан олардың орындалу реттілігін оңтайландырады. Осы кезеңде сұранысты орындаудың бірнеше жоспары құрылады және олардың ішінен бір - ДҚ-дың осы жағдайы үшін оңтайлысы таңдалады.

Төртінші кезеңде ДҚБЖ үшінші кезеңде дайындалған, сауалнаманың оңтайлы жоспарының екілік нұсқасын генерациялайды. ДҚБЖ сұраныстын орындаудың екілік жоспары іс-жүзінде бағдарламаның объектті кодының баламасы болып табылады.

Бесінші кезеңде ДҚБЖ сұраны орындай отырып, жасап шығарылған жоспарды іске асырады.

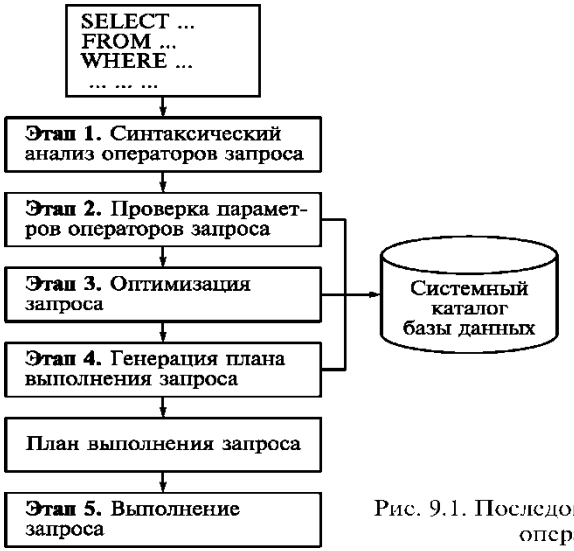


Рис. 9.1. Последовательность выполнения операторов SQL

Аталған кезеңдер ДҚ-ға жүгіну саны бойынша және оларды орындау үшін қажетті процессорлық уақыты бойынша ерекшеленеді.

Синтаксистік талдау өте жылдам жүргізіледі, ДҚ-дың жүйелік каталогтарына жүгінуді талап етпейді.

Семантикалық талдау метадеректер қорымен, яғни жүйелік ДҚ каталогтарымен жұмыс істеуді талап етеді, сондықтан осы кезеңді орындау кезінде жүйелік каталогқа жүгінеді.

Сұраныс жоспарын оңтайландыруға байланысты кезең жүйелік каталогпен ғана емес, сұраныста пайдаланылатын барлық қатынастардың ағымдағы жағдайын, олардың сыртқы жадтың беттерінде және сегменттерінде физикалық орналасуын сипаттайтын ДҚ туралы статистикалық ақпаратпен жұмыс істеуін талап етеді. Аталған себептер салдарынан оңтайландыру кезең сұранысты орындау үрдісіндегі еңбекті көп қажет ететін және ұзақ. Алайда егер оңтайландыру кезеңін жүргізбесек, онда оңтайландырылмаған сұранысты орындау уақыты оңтайландырылған сұрныс уақытын бірнеше есе арттыруы мүмкін. Сұранысты оңтайландыруға жұмсалған уақыт оңтайландырылмаған сұратуды орындауға шығындарды орнын толтырады.

SQL операторларын орындау кезеңдері интерактивті режимде және қосымшаның ішінде бірдей.

Әдетте, әртүрлі бағдарламалық жүйелерде (ДҚБЖ) жасап шығарушылар SQL тілінің меншік модификацияларын кіріктіреді.

Кіріктірілген SQL-дің өз ерекшеліктері бар. Алайда SQL операторларын негізгі бағдарламалау тілімен біріктіру кезінде мына қағидалар сақталу керек:

• SQL операторлары бағдарламалаудың бастапқы тілінде тікелей бағдарлама мәтініне енгізіледі. Бастапқы бағдарлама SQL операторларын компиляциялайтын, SQL препроцессордың кіреберісіне келіп түседі;

• кіріктірілген SQL операторлары бағдарламалаудың негізгі тілінің ауыспалыларына сілтеуі мүмкін;

• кіріктірілген SQL операторлары бағдарламалаудың негізгі тілінің ауыспалылары көмегімен SQL-сұратулардың нәтижелерін алады;

• ДҚ қатынастарының атрибуттарына (NULL) белгісіз мәндерді тағайындау үшін арнайы қызметтер пайдаланылады;

• сұраныс нәтижелерінің жол бойынша өңделуін қамтамасыз ету үшін кіріктірілген SQL-ге интерактивті SQL-де болмайтын бірнеше жаңа оператор қосылады.

Деректерді манипуляциялау операторлары бағдарламалық SQL кодына кіріктіру үшін өзгертуді қажет етпейді. Алайда іздеу операторы (SELECT) өзгертуді талап етті.

SELECT стандартты операторы қалыптастырылған сұраныс шарттарына сәйкес, деректер жинағыш қайтарады. Интерактивті SQL-де осы алынған деректер жинағы пайдаланушы консоліне шығарылады және ол алынған нәтижелерді қарай алады.

SELECT кіріктірілген операторы бағдарламалаудың негізгі тілдерімен келісілетін деректердің құрылымдарын жасау керек.

Кіріктірілген SQL-де сұраныстар екі түрге бөлінеді:

• күтілетін нәтижелер деректердің бір жолына сәйкес келетін бір жолдық сұраныстар. Осы жол бірнеше бағанның мәнін құрауы мүмкін;

• нәтижесі біртұтас жолдар жинағын алу болып табылатын көп жолды сұраныстар. Бұл ретте қосымша барлық алынған жолды өңдеу мүмкіндігіне ие болу керек. Яғни, алынған жолдар жинағын қарайтын және өңдейтін механизм болу керек.

Сұраныстың бірінші түрі - кіріктірілген SQL бар бір жолды сұраныс — төмендегідей болатын SQL операторының модификациясын шақырды:

SELECT [{ALL | DISTINCT}] <қайтарылатын бағандар тізімі>
INTO <негізгі тілдің ауыспалылар тізімі>
FROM <бастапқы кестелер тізімі>
[WHERE <қосу және іздеу шарттары>]

Кіріктірілген SELECT операторына негізгі тілдің ауыспалылар тізімін құрайтын жаңа INTO операторы қосылды. Дәл осы ауыспалыларға бір жолдық сұратудың нәтижелері орналастырылады, сондықтан негізгі тілдің ауыспалылар тізімі тәртібі бойынша, түрі бойынша және деректер көлемі бойынша қайтарылатын бағандар тізімімен келісілу керек. Бағдарламалаудың кез-келген тілі ережелері бойынша барлық негізгі ауыспалылар алдын ала қолданбалы бағдарламада сипатталған. Мысалы, егер “Кітапхана” ДҚ-да READERS («Оқырмандар») кестесі болса, онда біз нақты бір оқырман туралы мәлімет ала аламыз.

CREATE TABLE READERS
(READER_JD Smallint (4) PRIMARY KEY.
FIRST_NAME char (30) NOT NULL.
LAST_NAME char (30) NOT NULL.
ADRES char (50).
HOME_PHON char (12).
WORK_PHON char (12).
BIRTH_DAY date CHECK (DateDiff (year. GetDate () DIRTH_DAY)>= 17
):

Ол үшін негізгі ауыспалыларды сипаттаймыз. Transact SQL тілін пайдалана отырып, MS SQL SERVER 7.0 үшін мысалды қарастырайық. Жергілікті ауыспалыларды сипаттау кезінде Transact SQL тілінде арнайы «@» символы пайдаланылады. Transact SQL-дегі комментарийлер/* комментарий */ жұп символдарға алынған.

```
DECLARE @reader_id INT
DECLARE @FIRS_NAME Char (30), @LAST_NAME Char (50). @ADRES Char (50)
DECLARE @HOME_PHON Char(12).l3WORK_PHON Char(12) /* оқырман билетінің бірегей нөмірін белгілейік */
SET @READER_ID = 4
/* енді сұратуды орындап, алынған мәліметтерді бұдан бұрын белгіленген ауыспалыларға орналастырамыз */
SELECT READERS.FIRST_NAME. READERS.IAST_NAME. READERS.ADRES.
READERS.HOME_PHON. READERS.WORK_PHON
INTO @FIRS_NAME. @LAST_NAME. @ADRES. @HOME_PHON. @WORK_PHON
FROM READERS
WHERE READERS.READER_ID = @READER_ID
```

Осы қарапайым мысалда біз ауыспалылардың аттарын READERS кестесінің баған аттарындай жасадық, бірақ бұл міндетті емес. Алайда транслятор осы объекттерді ерекшелей алады, сондықтан Transact SQL диалектісінде жергілікті ауыспалыларды арнайы «@» символына алу қабылданған. Мысалда біз өрістердің білікті аттарын, кесте атына алынған өрістер аттарын пайдаландық. Алайда бұл міндетті емес, себебі сұраныс тек бір кестенің деректерін таңдайды.

Осы мысалда негізгі ауыспалылар әртүрлі рөлді атқарады. @READER_ID жергілікті ауыспалы сұранысқа қатысты кіреберіс болып табылады. Оған 4 мәні тағайындалған. Сұраныста бұл мән деректерді сүзу үшін пайдаланылады, сондықтан осы ауыспалы WHERE шартында пайдаланылады.

Қалған негізгі ауыспалылар шығаберіс ауыспалылар рөлін атқарады. Онда ДҚБЖ-ге ДҚ-дан алынған READERS қатынасының тиісті өрістерінің мәндерін орналастыра отырып, сұранысты орындау нәтижелерін орналастырады.

9.2. Көп жолды сұраныстарды өңдеу меңзерлері - операторлары

Одан күрделірек - көп жолды сұраныстарды қарастырамыз. Көп жолды сұраныстарды іске асыру үшін меңзер немесе жазбалар жинағын көрсеткіш ұғымы енгізіледі.

Меңзермен жұмыс істеу үшін бірнеше жаңа SQL операторы қосылады:

DECLARE CURSOR орындалатын сұранысты айқындайды, меңзер атын белгілейді және сұраныс нәтижелерін белгіленген меңзермен байланыстырады. Бұл оператор сұраныс үшін орындаушы болып табылмайды, ол тек келешек көптеген жазбалар құрылымын айқындайды және оны меңзердің бірегей атымен байланыстырады.

Бұл оператор бағдарламалау тілдеріндегі деректерді сипаттау операторына ұқсас;

OPEN ДҚБЖ-ге сипатталған сұранысты орындау, белгіленген сұранысқа сәйкес келетін виртуалды жолдар жинағын жасау командасын береді. OPEN операторы нәтижелер жолының виртуалды жинағының бірінші жолының алтына жазбаларға нұсқаушыны (меңзерді) орнатады;

FETCH жазбаға нұсқаушыны келесі позицияға жылжытады. Көптеген ДҚБЖ-де бұл оператор орын ауыстырудың аса кең қызметтерін іске асырады. Ол нұсқаушыны туынды жазбаға, ілгері және кейін жылжыту мүмкіндігін береді, абсолютті және салыстырмалы адресаттауға жол береді, меңзерді динамикалық кестенің бірінші немесе соңғы жазбасына орнату мүмкіндігін береді;

CLOSE меңзерді жабады және динамикалық кесте жазбаларына қолжетімдікті тоқтатады. Ол меңзер мен негізгі сұранысты орындау нәтижесі арасындағы байланысты іс-жүзінде жояды. Алайда коммерциялық ДҚБЖ-да CLOSE операторы виртуалды жазбалар жинағын жоюды білдіре бермейді.

Стандарт меңзерді айқындау операторының келесі синтаксисін айқындайды:

DECLARE <меңзер_аты>CURSOR FOR
<меңгер_сипаттізімі>Меңзердің сипаттізімі>::= <SELECT сұраныстың_тіркесі>

Меңзердің аты — бұл бағдарламалаудың негізгі тіліндегі қол жетімді сәйкестендіргіш.

Меңзерді жариялауда негізгі ауыспалылар пайдаланылуы мүмкін. Алайда OPEN операторын орындау кезінде негізгі сұраныстағы мәндерді сүзу шарттарына байланысты, кіреберіс ауыспалылар ретінде пайдаланылатын барлық негізгі ауыспалылардың мәндері белгілену керек екенін есте сақтаған жөн.

Кітапхананың барлық қарызға алушылардың тізімін құрайтын меңзерді айқындаймыз. Қарыз алушылар деп қолында тапсыру мерзімі өтіп кеткен, кемінде бір кітап бар оқырмандарды атайық.

```
DECLARE Debtor_reader_cursor CURSOR FOR
SELECT READERS. FIRST_NAME. READERS. LAST_NAME,
READERS.ADRES.
READERS.HOME_PHON. READERS.WORK_PHON. BOOKS.
TITLE
FROM READERS. BOOKS.EXEMPLAR
WHERE READERS.READER_ID = EXEMPLAR. READER_ID AND
BOOKS.ISBN = EXEMPLARE. ISBN AND
EXEMPLAR. DATA_JUT > Getdate()
ORDER BY READERS. FIRST NAME
```

Меңзерді айқындау кезінде біз қайтадан Transact SQL Getdate() қызметін пайдаландық, ол ағымдағы күннің мәнін қайтарады. Сонымен, белгілі бір меңзер кітапханаға уақытында қайтармаған, кітап атаулары көрсетілген, қарызға алушылар тізімін құрайтын жолдар жинағын жасайды.

SQL2 Transact SQL стандартына сәйкес меңзердің кеңейтілген анықтамасын құрайды.

DECLARE <меңзердің_аты>[INSENSITIVE] [SCROLL] CURSOR FOR<SELECT ТАҢДАУ ОПЕРАТОРЫ>
[FOR {READ ONLY | UPDATE [OF <баған_аты1> [....n]]}]

INSENSITIVE (сезімтал емес) параметрі меңзерді ашқаннан кейін басқа пайдаланушылар жасалған бастапқы кестелердегі барлық өзгерістер көрінбейтін, айқындалатын меңзерге сәйкес келетін жолдар жинағының режимін айқындайды. Осындай деректер жинағы басқа пайдаланушылармен бастапқы кестелерде жүргізілуі мүмкін барлық өзгерістерге сезімтал болмайды, меңзердің осы түрі ДҚ-мен "дереу жабысуына" сәйкес келеді.

ДҚБЖ осындай меңзерді аса жылдам әрі үнемді өңдей алады. Сондықтан егер ДҚ жағдайын белгілі бір нақты уақытқа қарастыру және өңдеу шынымен қажет болса, онда "сезімтал емес меңзерді" жасаудың мәні бар.

SCROLL өзекті сөзі FETCH операторында меңзер бойынша (FIRST, LAST, PRIOR, NEXT, RELATIVE, ABSOLUTE) кез-келген режимге орналастыру қолжетімді болатынын айқындайды.

Егер SCROLL өзекті сөзі көрсетілмесе, онда тек стандартты ілгері жылжыту қолжетімді болып есептеледі: FETCH операторындағы NEXT сипаттізімі.

Егер READ ONLY (тек оқуға арналған) сипаттізімі көрсетілсе, онда бастапқы кестелердің өзгерістері мен жаңартулары осы меңзерді пайдалана отырып орындалмайды. Осындай сипаттізімі бар меңзер өңдеу барысында ең жылдам болуы мүмкін. Алайда READ ONLY сипаттізімін арнайы көрсетпесеңіз, онда ДҚБЖ негізгі кестелермен модификация операцияларына рұқсат етілген деп есептейді, бұл жағдайда ДҚ тұтастығын қамтамасыз ету үшін жүйе меңзермен операцияларды баяу өңдейді.

UPDATE [OF <баған аты 1> [,...<баған атын>]] параметрі пайдалану кезінде біздің меңзермен жұмысымыз барысында өзгерістерге рұқсат етілетін, бағандар тізбесін белгілейміз. Осындай шектеу ДҚБЖ жұмысын жеңілдетеді және жылдамдатады. Егер бұл параметр көрсетілмесе, онда меңзердің барлық бағандарын өзгертуге болады деп болжалады. Біздің мысалға оралайық. Егер бізге қарыз алушылар туралы мәліметтерді беретін ДҚ дереу жабысуы қажет болса, онда меңзермен жұмысты жылдамдату мүмкіндігін беретін барлық параметрлерді қолданамыз.

Онда меңзерді сипаттау операторы келесі түрде болады:

```
DECLARE Debtor_reader_cursor INSENSITIVE CURSOR FOR
SELECT READERS.FIRST_NAME. READERS.LAST_NAME. READERS.ADRES. READERS.HOME_PHON. READERS.WORK_ PHON. BOOKS.TITLE
FROM READERS.BOOKS.EXEMPLAR
WHERE READERS.READER_ID = EXEMPLAR.READER_ID AND
BOOKS.ISBN = EXEMPLARE.ISBN AND EXEMPLAR.DATA_OUT
> Getdate ()
ORDER BY READERS.FIRST_NAME
FOR READ ONLY
```

Меңзерді сипаттау кезінде меңзермен байланысты жолдардың негізгі жинағын жасау үшін пайдаланылатын SELECT операторының түріне шектеу жоқ. SELECT операторында топтастырулар және кіріктірілген қосалқы сұраныстар, есептелетін өрістер пайдаланылуы мүмкін.

Кеңейтілген FETCH операторы келесі синтаксиске ие: FETCH
[NEXT | PRIOR | FIRST | LAST |
ABSOLUTE {n | <ауыспалының_аты>}
| RELATIVE{n|<ауыспалының_аты>}]
FROM
<меңзердің аты>INTO <негізгі ауыспалылар тізімі>

МұндаNEXT параметрі меңзермен байланысты негізгі жолдар жинағындағы ағымдағыдан кейінгі келесі жолдың таңдалуын белгілейді. PRIOR параметрі ағымдағыға қатысты бұдан бұрынғы жолдың жылжытылуын белгілейді. FIRST параметрі жинақтың бірінші жолына, ал LAST параметрі жинақтың соңғы жолына жылжытылуын белгілейті.

Бұдан басқа, кеңейтілген жылжыту операторында бірден белгіленген жолған орналастыруға рұқсат етіледі; бұл ретте ABSOLUTE параметрімен белгілей отырып, абсолютті адресаттау және RELATIVE параметрімен белгілей отырып, салыстырмалы адресаттауға рұқсат етіледі. Салыстырмалы адресаттау кезінде оң сан нұсқарды ағымдағы жазбадан төмен, теріс сан ағымдағы жазбадан жоғары жылжытады.

Алайда кеңейтілген FETCH операторын қолдану үшін SQL2 стандартына сәйкес меңзердің сипаттамасы міндетті түрде SCROLL өзекті сөзін құрауы керек. Кейде мұндай меңзерлер бұралатын деп аталады. Стандартқа осы меңзерлер жақын арада ғана енді, сондықтан коммерциялық ДҚБЖ-де ұқсас меңзерлермен жұмыс істейтін операторлар жиі қатты ерекшеленеді.

Бүгінгі күннің реалийлері коммерциялық ДҚБЖ жеткізушілерін SQL соңғы стандартын аса қатаң қадағалауға мәжбүрлейді. Техникалық құжаттамада FETCH операторының синтаксисінің екі нұсқасын кездестіруге болады: біреуі, стандартқа сәйкес келеді және екіншісі меңзермен жұмыс істеу үшін тек ДҚБЖ деректерімен ұсынылатын қосымша мүмкіндіктермен стандартты кеңейтеді.

Егер сіз ДҚ басқа платформаға орын ауыстыру қажет деп болжасаңыз, ал бұны әрқашан көздеу керек, онда стандартты мүмкіндіктерді пайдаланған жөн. Бұл жағдайда қосымша аса платформалы-тәуелсіз болады және басқа ДҚБЖ-ға орын ауыстыру жеңіл болады.

9.3. Меңзерді жабу операторы

Меңзерді жабу операторының синтаксисі қарапайым, ол мынадай болады:

CLOSE<меңзердің_аты>

Меңзерді жабу операторы меңзерді ашу операторымен жасалған уақытша кестені жабады және қолданбалы бағдарламаның осы объектке қолжетімдігін тоқтатады. Жабу операторының бірден-бір параметрі меңзердің аты болып табылады.

Меңзерді жабу операторы меңзерді ашу операторынан кейін кез-келген сәтте орындалуы мүмкін.

Кейбір коммерциялық ДҚБЖ-де меңзерді жабу операторынан басқа меңзерді деактивациялау (жою) операторы пайдаланылады. Мысалы, MS SQL Server 7.0-де меңзерді жабу операторымен қатар

DEALLOCATE <меңзердің_аты> операторы пайдаланылады.

Мұнда меңзерді жабу операторы меңзерге байланысты деректер жинағын жоймайды, тек оған қолжетімдікті жабады және бұдан бұрын осы меңзермен байланысты болған барлық бұғаттауларды босатады.

DEALLOCATE SQL Server операторын орындау кезінде DECLARE меңзерді сипаттау командасымен пайдаланылатын бөлінетін жадты босатады. Осы команданы орындағаннан кейін осы меңзер үшін OPEN командасын орындау мүмкін емес.

9.4. Меңзерді пайдалана отырып деректерді жою және жаңарту

Қолданбалы бағдарламаларда меңзерлер жиі деректерді ретімен қарау үшін пайдаланылады. Егер меңзер топтастыру операциясымен байланысты болмаса, онда іс жүзінде меңзердің әрбір жолы қатаң түрде бастапқы кестенің бір жолына ғана сәйкес келеді және осы жағдайда меңзерді деректерді жедел түзету үшін пайдалану қолайлы.

Стандартта меңзерге байланысты деректерді түрлендіру операциялары анықталған. Меңзердің ағымдағы көрсеткішіне байланысты жолды жою операциясының синтаксисі мынадай:

DELETE FROM <кестенің аты> WHERE CURRENT OF <меңзердің аты>

Егер операторда көрсетілген меңзер ашық және кейбір жолған орнатылған болса және меңзер өзгертілетін кестені анықтаса, онда меңзердің ағымдағы жолы жойылады, ал ол келесі жолдың алдында позицияланады. DELETE операторының FROM бөлімінде көрсетілген кесте меңзер сипаттізімінің FROM ең сыртқы бөлімінде көрсетілген кесте болу керек.

Егер меңзердің келесі жолын оқу қажет болса, онда FETCH NEXT операторын қайтадан орындау керек.

Меңзер деректерді түрлендіру үшін пайдаланылуы мүмкін. Позициялық түрлендіру операциясының синтаксисі мынадай:

UPDATE <кестенің_аты> SET <бағанның_аты>= (<мәні> | NULL} [{,<бағанның_аты>= {<мәні> | NULL}}...]
WERE CURRENT OF <меңзердің_аты>

Позициялық жаңартудың бір операторы меңзердің ағымдағы позициясына сәйкес келетін, кесте жолдарының бағандарының бірнеше мәні ауыстырылуы мүмкін. Түрлендіру операциясын орындағаннан кейін меңзер позициясы өзгермейді.

Жоюдың (DELETE) және түрлендірудің (UPDATE) позициялық операторларын қолдану үшін меңзер SQL1 стандартына сәйкес келесі талаптарға сәйкес келу керек:

• меңзерге байланысты сұраныс деректерді бір бастапқы кестеден оқу керек, яғни (DECLARE CURSOR) меңзерді айқындауға байланысты SELECT сұраныстың FROM ұсынысында тек бір кесте белгілену керек;

• сұраныста ORDER BY ретке келтіру параметрі болмауы мүмкін. Меңзер жолдары мен бастапқы кестенің өзара сәйкестігі сақталу үшін меңзер ретке келтірілген деректер жинағын сәйкестендіру керек;

• сұраныста DISTINCT өзекті сөзі болмауы керек;

• сұраныста топтастыру операциясы болмауы керек, яғни онда GROUP BY немесе HAVING ұсынысы болмауы керек;

• позициялық жою немесе жаңарту операцияларын қолданғысы келетін пайдаланушы негізгі кестеде осы операцияларды орындауға тиісті құқыққа ие болу керек.

Жаңарту операциялары үшін меңзерді пайдалану ДҚБЖ тарапынан ұқсас меңзермен жұмысты айтарлықтай күрделендіреді, сондықтан позициялық түрлендіруге байланысты операциялар тек оқу үшін пайдаланылатын меңзермен операцияларға қарағанда аса баяу орындалады. Дәл сондықтан меңзерді анықтау операторында егер түрлендіру операциясы үшін осы меңзерді пайдаланғыңыз келмесе, READ ONLY сөйлемін көрсету міндетті. Егер бастапқы дәлдеу бойынша қосымша нұсқаулар болмаса, ДҚБЖ түрлендіру мүмкіндігімен меңзерді жасап шығарады.

Меңзерлер — қосымшалардың бизнес-логикасын қалыптастыруға арналған қолайлы құрал, алайда түрлендіру мүмкіндігі бар меңзерді ашсаңыз, онда ДҚБЖ сіздің меңзеріңізге кірген негізгі кестенің барлық жолдарын бұғаттайды және осылай басқа пайдаланушылардың осы кестемен жұмысы бұғатталады.

Қажетті бұғаттау санын азайту үшін, интерактивті бағдарламалар жұмыс істеген кезде келесі ережелерді сақтаған жөн:

- транзакцияларды барынша қысқа ету керек;
- әрбір сұраныстан кейін және бағдарлама жасаған өзгертулерден кейін барынша жылдам COMMIT аяқтау операторын орындау керек;
- пайдаланушымен қарқынды өзара әрекеттесу жүзеге асырылатын немесе деректердің жолдары көп қаралатын бағдарламалардан қашқақтау керек;
- егер мүмкін болса, онда айналатын меңзерлерді (SCROLL) қолданбаған жөн. Себебі олар ашық меңзерге байланысты барлық іріктеу жолдарының бұғатталуын талап етеді;
- қарапайым реттік меңзерді пайдалану сізбен параллельді жұмыс істейтін және сол кестелерді пайдаланатын басқа пайдаланушылардың бұғатталуын азайтатын, FETCH операциясы орындалғаннан кейін жүйеге ағымдағы жолды бұғаттаудан шығару мүмкіндігін береді;
- егер мүмкін болса, меңзерді READ ONLY деп айқындаңыз.

Біз клиент-сервер модельдерін қарастырған кезде, дерекқор серверлерінің дамыған модельдерінде клиенттік қосымшаның бизнес-логикасының үлкен бөлігі клиентте емес, дәл осы серверде орындалады. Ол үшін сақталатын рәсімдер деп аталатын және ДҚ-да кестелер мен өзге негізгі объекттер ретінде сақталатын, арнайы объекттер пайдаланылады.

Осыған орай, қосымшаларда пайдаланылуы мүмкін меңзерлер әдетте сервер меңзерлері мен клиент меңзерлері деп бөлінеді.

Сервердің меңзері серверде құрылады және орындалады; оған байланысты деректер клиенттің компьютеріне жолданбайды. Сервердің меңзерлері әдетте сақталатын рәсімдер немесе триггерлерде айқындалады.

Клиенттің меңзерлері — бұл клиентте орындалатын қолданбалы бағдарламаларда айқындалатын меңзерлер. Осы меңзерге байланысты жолдар жинағы клиенттің компьютеріне жіберіліп, сонда өңделеді. Егер меңзермен үлкен деректер жинағы байланысты болса, онда меңзерге байланысты жолдар жинағын жіберу операциясы желі мен клиенттік компьютердің айтарлықтай уақыты мен ресурстарын алуы мүмкін.

Әрине, сервердің меңзерлері аса үнемді және жылдам орындалады. Сондықтан меңзерлерді пайдалануға байланысты соңғы ұсыныс клиент меңзерінің орнына сервер меңзерін жиірек пайдаланатындай, қосымшаңыздың жұмыс істеу логикасын трансформациялау керек.

9.5. Сақталатын рәсімдер

ДҚ-мен жұмыс істейтін қосымшалар тұрғысынан сақталатын рәсімдер (Stored Procedure) — бұл серверде орындалатын қосалқы бағдарламалар. ДҚ-ға қатысты - бұл ДҚ-да жасалатын және сақталатын объекттер. Олар клиенттік қосымшалардан шақырылуы мүмкін. Бұл ретте бір рәсім клиенттік қосымшаның кез-келген мөлшерінде пайдаланылуы мүмкін, бұл қолданбалы бағдарламалық жасақтаманы жасап шығаруға еңбек шығынын айтарлықтай үнемдеу және кодты қайтадан пайдалану стратегиясын тиімді қолдану мүмкіндігін береді. Кез-келген рәсімдер сияқты бағдарламалаудың стандартты тілдерінде сақталатын рәсімдер кіреберіс және шығаберіс параметрлерге ие болуы мүмкін немесе мүлде болмауы мүмкін.

Сақталатын рәсімдер пайдаланушы қосымшаларымен ғана емес, сондай-ақ триггерлермен белсендірілуі мүмкін.

Сақталатын рәсімдер бағдарламалаудың арнайы кіріктірілген тілінде жазылады. Олар кез-келген SQL операторын құрауы мүмкін, сонымен қатар бағдарламалаудың рәсімге бағдарланған тілдерінің баламалы операторларына ұқсас, бағдарламаларды орындау барысын басқаратын операторлар жинағын құрайды. Коммерциялық ДҚБЖ-де сақталатын рәсімдердің мәтіндерін жазу үшін бағдарламалаудың меншік тілдері пайдаланылады. Сонымен, ДҚБЖ Oracle-да ол үшін PL/SQL тілі, ал MS SQL Server-де Transact SQL тілі пайдаланылады. Oracle-дің соңғы нұсқаларында сақталатын рәсімдерді жазу үшін Java тілін пайдалану жарияланған.

Сақталатын рәсімдер ДҚ объекттері болып табылады. Әрбір сақталатын рәсім алғаш орындау кезінде компиляцияланады. Компиляция барысында рәсімдерді орындаудың оңтайлы жоспары құрылады. Рәсімнің сипаттамасы оны орындау жоспарымен бірге ДҚ жүйелік кестелерінде сақталады.

Сақталатын рәсімді құру үшін SQL CREATE PROCEDURE операторы қолданылады.

Сақталатын рәсімді өздігінен тек ДҚ иесі ғана және сақталатын рәсімді құрушы орындай алады. Алайда сақталатын рәсімнің иесі оны іске қосу ережелерін басқа пайдаланушыларға табыстай алады.

Сақталатын рәсімнің аты жазылатын бағдарламалау тілінде сәйкестендіруші болып табылады және осы тілдегі сәйкестендірушілерге қойылатын барлық талаптарға сәйкес келу керек.

MS SQL Server-де сақталатын рәсім келесі оператормен жасалады:

```
CREATE PROCEDURE <рәсімнің_аты> [;<нұсқа>]
{(@параметр1деректер_түрі}
[VARYING] [=<өздігінен_мәні>] [OUTPUT]] [,.параметр^]
[ WITH
{RECOMPILE | ENCRYPTION | RECOMPILE, ENCRYPTION}]
[FOR REPLICATION]
AS РӘСІМНІҢ ДЕНЕСІ
```

Мұндағы міндетті емес VARYING өзекті сөзі бұдан бұрын айқындалған параметр үшін өздігінен белгіленген мәнді айқындайды.

RECOMPILE өзекті сөзі құрылатын сақталатын рәсімнің компиляция режимін айқындайды. Егер RECOMPILE өзекті сөзі белгіленсе, онда рәсім орындауға шақырылған сайын қайта компиляцияланады. Бұл рәсімнің орындалуын баяулатуы мүмкін. Алайда егер осы сақталатын рәсіммен өңделетін деректер алғаш шақыру кезінде жасалған бұдан бұрынғы орындау жоспары одан арғы шақырулар кезінде мүлде тиімсіз болатындай қарқынды болса, онда осы рәсімді құру кезінде осы параметрді қолданған жөн.

ENCRYPTION өзекті сөзі сақталатын рәсімнің бастапқы мәтіні ДҚ-да сақталмайтын режимді айқындайды. Мұндай режим сақталатын рәсімдер болып табылатын зияткерлік өнімге авторлық құқықты сақтау үшін қолданылады. Жасап шығарушы жасап шығарылған рәсімдердің бастапқы мәтіндері тапсырыс берушіде жұмыс істейтін ДҚ әкімшісіне қолжетімді болғанын қаламаса, осындай режим жиі қолданылады.

Алайда сақталатын рәсімнің атынан басқа барлық қалған параметрлер міндетті емес. Рәсімдер рәсім немесе қызмет-рәсімдер болуы мүмкін. Бұл ұғымдар мұнда жоғары деңгейлі бағдарламалау тіліндегідей дәстүрлі түсіндіріледі. Сақталатын қызмет-рәсім рәсімнің атын айқындайтын ауыспалыға тағайындалатын мәнді қайтарады.

Рәсім мәнді айқын түрінде қайтармайды, алайда онда осы параметрдің шыға беріс болып табылатынын айқындайтын OUTPUT өзекті сөзі пайдаланылуы мүмкін.

Қарапайым сақталатын рәсімнің бірнеше мысалын қарастырайық.

/* параметрлер кітапханасындағы нақты кітап данасының болуын тексеретін рәсім:

@ISBN — кітаптың шифрі — рәсім дана санына тең параметрді қайтарады. Егер нөл қайтып келсе, онда бұл кітапханадағы осы кітаптың бос данасы жоқ екенін білдіреді. */ CREATE PROCEDURE COUNT_EX ((@ISBN varchar (12))

AS

/* ішкі ауыспалының анықтаймыз */

DECLARE (@TEK_COUNT int

/* тиісті операторды орындаймыз SELECT

Қазіргі уақытта оқырмандардың қолындағы емес, кітапханадағы даналарды ғана санаймыз */

SELECT @TEK_COUNT = select count (*)

FROM EXEMPLAR

WHERE ISBN = @ISBN

AND READER_@ID Is NULL AND EXIST = True /* 0 — кітапханадағы осы кітаптың бірде бір бос данасының жоқтығын білдіреді */

RETURN @TEK_COUNT

Сақталатын рәсім бірнеше тәсілмен шақырылуы мүмкін. Қарапайым тәсіл - операторды пайдалану:

EXEC <рәсімнің аты><кіреберіс параметрдің мәні1>... <шығаберіс параметрге арналған ауыспалының аты 1>...

Бұл ретте барлық кіреберіс және шығаберіс параметрлер міндетті түрде рәсімде айқындалған тәртіппен белгіленуі керек.

Мысалы, егер ISBN 966-7393-0809 бар «Oracle8. Пайдаланушы энциклопедиясы» кітаптар данасының санын табу керек болса, онда бұдан бұрын жасалған сақталатын рәсімді шақыру мәтіні төмендегідей болу керек:

/*Екі ауыспалы айқындалды: осы кітаптың @Ntek данасы кітапханада бар; @ISBN — кітаптың халықаралық шифрі */ declare @Ntek int

DECLARE @ISBN VARCHAR(14)

/* Ауыспалының мәнін тағайындаймыз @ISBN */

SELECT @ISBN = '966-7393-08-09'

/* @Ntek ауыспалыға COUNT_EX сақталатын рәсімді орындау нәтижелерін тағайындаймыз*/

EXEC @Ntek =COUNT_EX:2 @ISBN

Егер сақталатын рәсімнің бірнеше нұсқасы айқындалса, онда шақыру кезінде орындауға арналған нақты нұсқаның нөмірін көрсете аласыз. Мысалы, COUNT_EX рәсімнің 2 нұсқасында осы рәсімді орындайтын соңғы операторды түрі төмендегідей

EXEC @Ntek =COUNT_EX:2 @ISBN

Алайда, егер рәсімде кіреберіс параметрлердің мәндері өздігінен айқындалса, онда рәсімді іске қосу кезінде барлық параметрдің мәні көрсетілмеуі мүмкін. Бұл жағдайда рәсімді шақыру операторы келесі түрде жазылуы мүмкін:

EXEC <рәсімнің аты><параметрдің аты1>=<параметрдің мәні 1>... <параметрдің атыN>=<параметрдің мәніN>...

Мысалы, белгілі бір баспа белгілі бір жылы басып шығарған кітап санын есептейтін рәсімді құрамыз. Рәсімді құру кезінде басып шығарылған жылы ретінде өздігінен ағымдағы жылдың мәнін белгілейміз.

CREATE PROCEDURE COUNT_BOOKS (@YEARIZD int = Year (GetDate()).

@PUBLICH varchar (20))

/* белгілі бір жылы басып шығарылған, нақты баспаның кітап санын есептеу рәсімі. Параметрлері: @YEARIZD int басып шығарылған жылы және

@PUBLICH баспаның аты */

AS

DECLARE @TEK_Count int

SELECT @TEK_COUNT = SELECT COUNT (ISBN)

FROM BOOKS

WHERE YEARIZD =@YEARIZD AND PUBLICH = @PUBLICH

SELECT операторымен орындаумен бірге оны орындау нәтижелері бұдан бұрын айқындалған @lLK_Count ауыспалыға тағайындалады*/

/* рәсімнің жұмыс істеу нәтижесін қалыптастыру кезінде біз кітапханада белгіленген жылы шығарылған кейбір баспаның бірде бір кітабы болмауы мүмкін екенін есепке алуымыз керек. Бұл жағдайда SELECT сұранысын орындау нәтижесінің мәні белгісіз болады, бірақ санды мәндерді талдаған жөн. Сондықтан біз қайтарылатын мән ретінде nl,n2,...,nm мәндер тізімінен алғашқы нақты мәнді, яғни NULL тең болмайтын мәнді қайтаратын, Transact SQL COALESCE (nl,n2,...,nm) тілінің арнайы кіріктірілген қызметінің жұмыс нәтижелерін пайдаланамыз*/

RETURN COALESCE (@TEK_Count, ())

Енді осы рәсімді шақырамыз, ол үшін рәсімді орындау нәтижелерін орналастыруға болатын ауыспалыны дайындаймыз:

declare @N int

@N ауыспалыда ағымдағы жылы «АКАДЕМИЯ» баспасы басып шығарған, кітапханада бар кітап санын аламыз. Біз барлық параметрлерді белгілей отырып, осы рәсімге қайта орала аламыз:

EXEC @N = COUNT_BOOKS @PUBLICH = 'АКАДЕМИЯ', @YEARIZD = 2004

Сонда 2014 ж. «АКАДЕМИЯ» баспасымен басып шығарылған және біздің кітапханамызда бар кітап санын аламыз.

Егер біз параметрлерді аттары бойынша белгілейтін болсақ, онда рәсімді жасау кезінде сипатталған тәртіппен белгілеу міндетті емес.

Әрбір сақталатын рәсім ДҚ объектісі болып табылады. Жүйелік каталогта оның бірегей аты мен бірегей ішкі нөмірі бар. Біз сақталатын рәсімнің мәтінін өзгерту кезінде алдымен осы рәсімді ДҚ-да сақталатын объект ретінде жоюымыз керек және содан кейін ғана оның орнына жаңасын жазуымыз керек. Сақталатын рәсімді жою кезінде сол уақытта оның барлық нұсқасы жойылатынын атап өткен жөн.

Ескі рәсімді жою және оны жаңасына ауыстыру үрдісін автоматтандыру үшін, сақталатын рәсім мәтінінің басында жүйелік каталогта осындай аты бар “сақталатын рәсім” типті объекттің болуын тексеруге болады және осы объекттің сипаттамасы болған жағдайда оны жүйелік каталогтан жою керек. Бұл жағдайда сақталатын рәсімнің мәтіні арнайы тексеру операторымен ескертіледі және мысалға, келесі түрде болуы мүмкін:

/* жүйелік каталогта осындай аты және ДҚ иесі жасаған типті объекттің болуын тексеру */

IF exists (select * from sysobjects where id = object_id ('dbo.NEW_BOOKS') and systant & Oxf = 4)

/* егер объект болса, онда оны алдымен жүйелік каталогтан жоямыз*/

drop procedure dbo.NEW_BOOKS

GO

CREATE PROCEDURE NEW_BOOKS (@ISBN varchar (12).@TITL varchar (255),@AUTOR varchar (30).@COAUTOR varchar (30).@

YEARIZD int. @PAGES INT,@NUM_EXEMPL INT)

/* осы кітаптың дана саны көрсетілген жаңа кітапты енгізу рәсімі

```
Параметрлер
@ISBN          varchar (12) — кітап шифрі
@TITL          varchar(255) — атауы
@AUTOR         varchar(30) — автор
@COAUTOR       varchar(30) — бірлескен автор
@YEARIZD       int          — басып шығарылған жылы
@PAGES         int          — бет саны
@NUM_EXEMPL    int          — дана саны */
AS
/*қалған кіріске алынбаған, яғни инвентарлы нөмірлер тағайындалмаған кітап данасының саны сақталатын ауыспалыны сипаттаймыз */
/DECLARE @TEK int
/ *кітап туралы деректерді BOOKS кестесіне енгіземіз*/
INSERT INTO BOOKS VALUES @ISBN. @TITL. @AUTOR. @COAUTOR.@YEARIZD@PAGES)
/* қалған енгізілуі қажет дана санының ағымдағы есептегіш мәнін тағайындау /
SELECT @TEK = @NUM_EXEMPL
/* осы кітаптың жаңа данасын енгізу үшін циклді ұйымдастырамыз */ WHILE @TEK>0 /* қалған дана саны нөлден артық */

BEGIN
/* кітап данасының инвентарлы нөмірі үшін IDENTITY қасиетін белгілегендіктен, инвентарлы нөмірді енгізу қажет емес. СУБД бұдан бұрынғыға бір бірлікті қосып, оны автоматты түрде есептейді, INSERT енгізу операторын орындау кезінде енгізеді.
Кітапханадағы дананың барын айқындайтын ая (EXIST), — логикалық ая; біз онда кітапханадағы бар кітап данасына сәйкес келетін TRUE мәнін енгіземіз */
INSERT into EXEMPLAR (ISBN.DATA_IN. DATA_OUT. EXIST)
VALUES (@ISBN.GetDate ().GetDate ()).TRUE)
/* қалған дана санын есептегіштің ағымдағы мәнін өзгерту */
SELECT @TEK = @TEK - 1
END /* кітап данасы туралы деректерді енгізу циклінің соңы */

GO
```

Егер біз дананың инвентарлы нөмірі ретін инкрементті аяны пайдаланбасақ, кітапханада сақталатын соңғы кітап данасының нөмірін бір бірлікке арттыра отырып, инвентарлы нөмірді өзіміз тағайындау алушы едік. Кітапханада бар дана санын есептеуге болатын еді, бірақ біз кейбіреулерін жойып алар едік. Сонда жаңа дананың нөмірі әлдеқайда пайдаланылуы мүмкін және біз деректерді енгізе алмаймыз, жүйе бастапқы кілттің бірегейлігін бұзу мүмкіндігін бермейді.

Бұл жағдайда рәсімнің мәтіні төмендегідей болады:

```
/* жүйелік каталогта ДҚ иесі жасап шығарған, осындай аты мен түрі бар объекттің болуын тексеру */
if exists (select * from sysobjects where id = object_id('dbo. NEW_BOOKS') and sysstat & Oxf = 4)
/* егер объект болса, онда алдымен оны жүйелік каталогтан жоямыз*/
drop procedure dbo. NEW_BOOKS
CREATE PROCEDURE NEW_BOOKS (@ISBN varchar (12).@TITL varchar (255),@AUTOR varchar (30).@CoAuTOR varchar (30).@ YEARIZD int. @PAGES INT,@NUM_EXEMPL INT)
/* осы кітаптың дана саны көрсетілген жаңа кітапты енгізу рәсімі Параметрлер

@ISBN          varchar (12)    кітаптың шифрі
@TITL          varchar(255)    атауы
@AUTOR         varchar(30)     автор
@COAUTOR       varchar(30)     бірлескен автор
@YEARIZD       int             шығарылған жылы
@PAGES         int             бет саны
@NUM EXEMPL    int             дана саны */
 */
DECLARE @TEK int
DECLARE @INV int
INSERT INTO BOOKS VALUES (@ISBN.@TITL.@AUTOR. @COAUTOR.@YEARIZD.@PAGES)
/* қалған енгізілуі қажет дана санының ағымдағы есептегіш мәнін тағайындау */
SELECT @TEK = @NUM EXEMPL
/* кітапханадағы инвентарлы нөмірдің ең жоғарғы мәнін айқындау */
SELECT @INV = SELECT MAX (ID_XEMPLAR)
FROM EXEMPLAR
/* осы кітаптың жаңа данасын енгізу үшін циклді ұйымдастырамыз */
WHILE @TEK>0
/* қалған дана саны нөлден артық */ BEGIN

insert into EXEMPLAR (ID_XEMPLAR.ISBN.DATA_IN. DATA_OUT.EXIST)
VALUES ((@INV.(@ISBN.GETDATE (). GETDATE (), TRUE) /* есептегіш пен инвентарлы нөмірді есептегіштің ағымдағы мәндерін өзгерту */
```

```
SELECT @TEK = @TEK - 1 SELECT @INV = @INV + 1
END
/* кітап данасы туралы деректерді енгізу циклінің соңы */
GO
```

Сақталатын рәсімдер басқасын шақыруы мүмкін. Белгілі бір оқырманның оқу билетінің нөмірін қайтаратын сақталатын рәсімді жасап шығарамыз:

```
if exists (select * from sysobjects where id = object fd ('dbo.
CK_READER') AND sysstat & Oxf = 4)
/* егер объект болса, алдымен оны жүйелік каталогтан жоямыз */
drop procedure dbo.CK_READER
/* Егер оқырман болса, рәсім оқу билетінің нөмірін қайтарады, және болмаса - 0. Параметрлер ретінде тегі мен туған күнін жібереміз */
CREATE PROCEDURE CK_READER (@FIRST_NAME varchar
(30). @BIRTH_DAY varchar (12))
AS
/*оқу билетінің нөмірі сақталатын, ауыспалыны сипаттаймыз */
DECLARE @NUM_READER INT /* оқырманның болуын
анықтаймыз */
SELECT @NUM_READER = STLECT NUM_READER FROM
READERS
WHERE FIRST_NAME = @ FIRSTJAME AND convert (varchar (8).
BIRTH_DAY,4) = @BIRTH_DAY
RETURN COALESCE ((@NUM_READER.0)
```

Біз мұнда da.ta.Time деректер түрін varchar(8) деректер түріне түрлендіру қызметін пайдаландық. Салыстыру операциясын орындау кезінде деректер түрін келісу үшін жасау керек. Шынымен, @BIRTH_DAY кіреберіс ауыспалының (varchar) символ түрі бар, ал BIRTH_DAY дерекқор аясының SmallDateTime түрі бар.

Сақталатын рәсімдер бірнеше шығаберіс параметрдің болуын болжайды. Ол үшін әрбір шығаберіс параметр өз деректер түрін белгілегеннен кейін OUTPUT өзекті сөзіне ие болу керек. Бірнеше шығаберіс параметрі бар сақталатын рәсім мысалын қарастырайық.

Жаңа оқырманды енгізу рәсімін жасаймыз; бұл ретте рәсімнің ішінде жаңа оқу билетінің нөмірін тағайындамау үшін, осы оқырманның картотекамызда болуын тексереміз. Рәсімнің шығаберіс параметрлері оқу билетінің нөмірі болады - кітапханамызда осындай сипаттамалары бар оқырманның бұдан бұрын жазылу белгісі, ал егер жазылған болса, қанша кітап алғанын көре аламыз.

/* осы рәсімнің ДҚ-да болуын тексеру */ if exists (select * from sysobjects where id = object_id (N'[dbo].[NEW_READER]') and OBJECTPROPERTY (id, N' Is Procedure')=1)

drop procedure [dbo].[NEW_READER]

GO

/* енгізілетін параметрлердің белгіленген мәндері бар оқырманның барын тексеру рәсімі

Егер мұндай оқырман болмаса, рәсім оқу билетінің жаңа нөмірін қайтарады, кері жағдайда - ескі нөмірді және оқырман қарызға алған кітап санын хабарлайды */

CREATE PROCEDURE NEW_READER (@NAME_READER varchar (30), (@ADRES varchar (40). @HOOM_PHONE char (9). (@WORK_PHONE char (9), @BIRTH_DAY varchar (8).

@NUM_READER int OUTPUT.

/* оқу билетінің нөмірін айқындайтын шығаберіс параметр */

@Y_N int OUTPUT.

/* оқырманның бұдан бұрын кітапханаға жазылғанын айқындайтын шығаберіс параметр */

@COUNT_BOOKS int OUTPUT

/* оқырман қолындағы кітап санын айқындайтын шығаберіс параметр */)

AS

/* егер оқырман кітапханаға жазылған болса, оқу билетінің нөмірі сақталатын ауыспалы */ DECLARE @N_R int '* оқырманның болуын анықтау */

EXEC @N_R = CK_READER @NAME_READER.@BIRTH_DAY IF @N_R= 0 Or @N_R Is Null

* егер белгіленген сипаттамаларға сәйкес келетін оқырман болмаса, яғни @N_R ауыспалыға нөл мәні тағайындалса немесе оның мәні айқындалмаса, онда жаңа оқырман үшін оқу билетінің жаңа нөмірін тағайындауға өтеміз*/

BEGIN

/* оқу билетінің нөмірін инкрементті ая ретінде айқындағандықтан, енгізу операторында көрсетпейміз, жүйе жаңа оқырманға кезекті нөмірді өзі тағайындайды */

INSERT INTO RADER NAME_READER.ADRES. HOOM_PHONE,WORK_PHONE. BIRTH_DAY)

VALUES (@NAME_READER.@ADRES.@HOOM_PHONE.

(@WORK_PHONE, Convert (smalldatetime. @BIRTH_DAY.4))

/* INSERT операторында @BIRTH_DAY символды ауыспалыны BIRTH_DAY (туған күні) аясы үшін айқындалған smalldatetime деректер түріне түрлендіруіміз керек. Оны Transact SQL Convert тілінің кіріктірілген қызметі арқылы түрлендіреміз*/

/* енді оқу билетінің тағайындалған нөмірін айқындаймыз */

SELECT @NUM_READER = NUM_READER

FROM READER

WHERE NAME_READER = @NAME_READER Convert (varchar (8).BIRTH_DAY.4) = @BIRTH_DAY /* мұнда қайтадан типті түрлендіру қызметін пайдаланамыз, алайда бұл жағдайда BIRTH_DAY аясын smalldatetime түрінен @BIRTH_DAY кіреберіс параметрі белгіленген, varchar (8). түріне түрлендіру керек */

SELECT @Y_N =0

/* @Y_N шығаберіс параметріне 0 (нөл) мәнін тағайындаймыз, бұл осы оқырманның бұдан бұрын кітапханамызға жазылмағанын білдіреді */

SELECT @COUNT_BOOKS = 0

/* оқырманның қолындағы кітап санын сақтайтын шығаберіс параметрге нөл мәнін тағайындаймыз */

RETURN 1

END

ELSE

/* егер @N_R ауыспалының мәні нөлге тең болмаса, онда белгіленген сипаттамаларға сәйкес келетін оқырманның біздің кітапханамызға бұдан бұрын жазылғанын білдіреді */

BEGIN

/* оқу билетінің нөмірі табылған оқырманның қолындағы кітап санын анықтау */

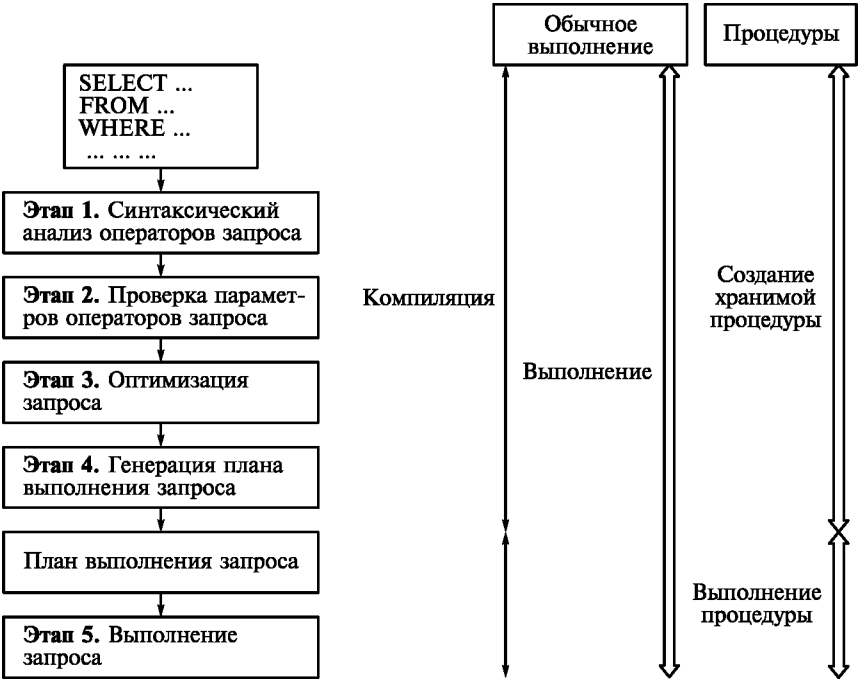
SELECT @COUNT_BOOKS = COUNT(INV_NUMBER)

FROM EXEMPLAR WHERE NUM_READER = @N_R SELECT @COUNT_BOOKS = COALESCE (@COUNT_ BOOKS.0)

/* @COUNT_BOOKS шығаберіс параметрге біздің оқырманымыздың қолындағы кітап санына тең мәнді тағайындаймыз; егер бұдан бұрынғы @COUNT_BOOKS сұратуында белгісіз мән тағайындалса, онда параметрлері ретінде белгіленген, мәндер тізімінен алғашқы белгілі мәнді қайтаратын COALESCE(@COUNT_ BOOKS.0) кіріктірілген қызметін пайдалана отырып, оны нөлге ауыстырамыз*/

SELECT @Y_N = 1

/* @Y_N шығаберіс параметріне 1 мәнін тағайындаймыз, бұл осы оқырманның біздің кітапханаға бұдан бұрын жазылғанын білдіреді */



9.2-сур. SQL операторларын клиентте орындау үрдісі және сақталатын рәсімді орындау үрдісі

Сақталатын рәсімдер де клиент-сервер сәулетімен желіде жұмыс істеу кезінде жылдам әрекет етуін арттыруда өзекті рөлді атқарады.

9.2-суретте SQL операторларын “клиентте” орындау үрдісі және сақталатын рәсімді орындау үрдісі ұсынылған.

Бұл жағдайда клиент серверге тек сақталатын рәсімді іске қосу командасын орындау үшін ғана жүгінеді. Сақталатын рәсімнің өзі серверде орындалады. Желі бойынша жіберілетін ақпарат көлемі қысқартылады.

9.6. Триггерлер

Триггер — бұл SQL Server тиісті кестелерді түрлендіру операцияларын орындау кезінде шақыратын, сақталатын рәсімнің арнайы түрі. Триггер байланысты болатын операцияны орындау кезінде автоматты түрде активтендіріледі. Триггерлер бір кестенің үстінде түрлендірудің бір немесе бірнеше операциясымен байланысады.

Әртүрлі коммерциялық ДҚБЖ-де әртүрлі триггер қарастырылады. Сонымен MS 5QL Server-де триггерлер тек пост сүзгілер ретінде ғана, яғни оқиға өткеннен кейін орындалатын триггерлер айқындалған.

ДҚБЖ Oracle-да триггердің екі түрі айқындалған:

• түрлендіру операциясын іске асырудан бұрын іске қосылуы мүмкін триггерлер (олар BEFORE-триггерлер деп аталады);

• MS SQL Server триггерлеріне баламалы, тиісті түрлендіруді орындағаннан кейін белсендірілетін триггерлер (олар AFTER-триггерлер деп аталады).

Триггерлер ДҚ-дың семантикалық тұтастығын сақтау үшін тиімді пайдаланылуы мүмкін, алайда олардың басымдылығы кестелерді сипаттау деңгейінде және кестелер арасындағы байланыстар деңгейінде белгіленетін шектеу - ережелер (constraints) басымдылығынан кем. Әрқашан триггерлерді жазу кезінде осыны есте сақтау керек. Байланыстар бойынша біртұтастық ережелерін бұзған кезде (DRI — Declarative Referential Integrity) триггер ешқашан іске қосылмауы мүмкін.

Триггерлерді жасау үшін арнайы команда пайдаланылады:

CREATE TRIGGER <триггердің_аты>
ON <кестенің_аты>
FOR {[INSERT][, UPDATE] [, DELETE]}
[WITH ENCRIPTING]
AS
SQL-операторлар (бағдарламаның мәтіні)

Триггердің аты ДҚБЖ бағдарламалаудың кіріктірілген тіліндегі сәйкестендіруші болып табылады және тиісті талаптарға сәйкес келу керек.

FOR параметрінде триггерді іске қосатын, бір немесе бірнеше түрлендіру операциясы белгіленеді.

WITH ENCRIPTING параметрінің мәні сақталатын рәсімдердің мәнімен бірдей. Ол триггердің бастапқы мәтінін жасырады.

Триггер операторларының құрамын шектейтін келесі ережелер бар:

• триггердің денесінде жаңа ДҚ объекттерін құру операцияларын пайдалануға болмайды;

• триггерде ДҚ объектілерінің барлық түрлері үшін DROP объекттерін жою командасын пайдалануға болмайды;

• триггердің денесінде

• ДҚ-дың ALTER TABLE, ALTER DATABASE объекттерін өзгерту командаларын пайдалануға болмайды;

• ДҚ объекттеріне белгіленген қолжетімдік құқықтарын өзгертуге, яғни GRAND немесе REVOKE командаларын орындауға болмайды;

• триггер ұсыну үшін жасалмайды (VIEW);

• триггер сақталатын рәсімдерге қарағанда ешқандай мәнді қайтара алмайды, ол сервермен автоматты түрде іске қосылады және өз бетімен бірден бір клиентпен байланыса алмайды.

Бақылау сұрақтары

1. SQL тілінің құралдарымен ДҚ-қа қолжетімдік қандай екі режимде жүзеге асырылуы мүмкін?
2. Әмбебап бағдарламалау тілдеріне тән қандай операторлар SQL тілінде жоқ?
3. SQL операторларының сұраныстарды орындау үрдісі шартты түрде қандай кезеңдерге бөлінуі мүмкін?
4. Кіріктірілген SQL-дегі сұраныстар қандай екі түрге бөлінеді?
5. INTO операторының мақсаты қандай?
6. «Меңзер» ұғымы нені білдіреді?
7. Қолданбалы бағдарламаларда меңзер не үшін пайдаланылады?
8. Мына операторлар нені білдіреді:
   - DECLARE CURSOR;
   - OPEN;
   - FETCH;
   - CLOSE?
9. «Сақталатын рәсім» (Stored Procedure) ұғымы нені білдіреді?
10. Сақталатын рәсім мәтіндерін жазу үшін коммерциялық ДҚБЖ-де бағдарламалаудың қандай тілдері пайдаланылады?
11. Триггер сақталатын рәсімнен немен ерекшеленеді?

III БӨЛІМ

БӨЛІНГЕН ДЕРЕККОРЛАРДЫ БАСҚАРУ ЖҮЙЕЛЕРІ

10 тарау

ДЕРЕКТЕРДІ БӨЛІП ӨҢДЕУ

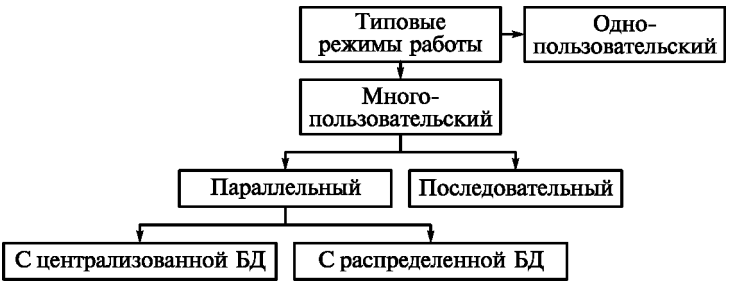
10.1. Негізгі ұғымдар

ДҚБЖ желіде орналаспаған дербес компьютерде орналасқан кезде, ДҚ әрқашан монополиялық режимде пайдаланылады. Онымен бірнеше пайдаланушы жұмыс істесе де, олар тек ретімен жұмыс істей алады.

Алайда, жергілікті дерекқорларды қолдану тәжірибесі көрсеткендей, көптеген жағдайда құрамындағы ақпарат көп пайдаланушылық сипатқа ие, сондықтан пайдаланушылардың дерекқорлармен бір уақытта жұмыс істеу мүмкіндігін қамтамасыз ететін ДҚБЖ жасап шығару қажеттігі туындайды. Бұған қоса, барлық қазіргі заманға сай кәсіпорындар ақпараттық жасақтау саласындағы өз саясатын CALS-технологияларының қағидаттары негізінде құрады (4 тарауды қараңыз).

Әртүрлі пайдаланушының ақпаратқа бір уақытта қолжетімдік мүмкіндігін қамтамасыз ететін дерекқорларды басқару жүйелері *бөлінген дерекқорларды басқару жүйелері* деп атайды. Жалпы жағдайда ДҚ пайдалану режимдері 10.1-суретте келтірілген түріне ие.

Бөлінген дерекқорларды басқару жүйелерінде қолданылатын негізгі ұғымдарды қарастырайық.



10.1-сур. Дерекқормен жұмыс істеу режимдері

ДҚ пайдаланушы — дерекқорға жүгінетін бағдарлама немесе адам.

Сұраныс— пайдаланушының ДҚ-ға ақпаратты енгізу, алу немесе өзгерту мақсатымен ДҚ-ға жүгіну үрдісі.

Транзакция — ДҚ-ды бір қарама-қайшы емес жағдайдан екінші қарама-қайшы емес жағдайға аударатын, ДҚ-дағы деректерді түрлендіру операцияларының реті.

ДҚ логикалық құрылымы — физикалық тәуелсіз деңгейде ДҚ анықтау; ДҚ концептуалды моделіне жуығырақ.

ДҚ топологиясы немесе бөлінген ДҚ құрылымы — дерекқордың физикалық ұйымын желіде тарату схемасы.

Жергілікті дербестік жергілікті ДҚ ақпараты және оған байланысты деректерді анықтау жергілікті иесіне жатады және өзі басқарады.

Қашықтықтан сұраныс жасау — модем байланысын пайдалана отырып орындалатын сұраныс.

Қашықтықтан транзакцияны іске асыру мүмкіндігі — бір қашықтықтан басқарылатын тораптағы, көптеген SQL-сұраныстан тұратын бір транзакцияны өңдеу.

Бөлінген транзакцияға қолдау көрсету желінің бірнеше торабында (қашықтықтан басқарылатын немесе жергілікті) орындалатын, бірнеше SQL сұраныстан тұратын транзакциялардың өңделуіне рұқсат етіледі, алайда бұл жағдайда әрбір сұрату тек бір торапта өңделеді.

Бөлінген сұраныс — өңдеу кезінде желінің әртүрлі торабында орналасқан, ДҚ-дағы деректер пайдаланылатын сұраныс.

Бөлінген деректерді өңдеу жүйелері мультибағдарламалық операциялық жүйелерде құрылған және орталық ЭЕМ сыртқы жадының құрылғыларында ДҚ орталықтандырылып сақталуын және терминалды көп пайдаланушыға қолжетімді режимін пайдаланған ДҚ-дың алғашқы нұсқасына байланысты. Бұл ретте пайдаланушы терминалдарының меншік ресурстары, яғни деректерді сақтау және өңдеу үшін пайдаланылуы мүмкін процессорлары мен жадтары болған жоқ. Көп пайдаланушы режимінде жұмыс істейтін алғашқы толығымен релициялық жүйе IBM фирмасының ДҚБЖ SYSTEM R болды. Онда SQL деректерін манипуляциялау тілі және шамамен барлық коммерциялық ДҚБЖ-де базистік болып табылатын, деректерді бөліп өңдеу кезінде қолданылатын синхрондаудың негізгі қағидалары ретінде іске асырылған.

10.2. Бөлінген дерекқорлар технологиясындағы клиент-сервер модельдері

Клиент-сервер есептегіш модель 1990 жылдары ашық жүйелердің пайда болуына байланысты. “Клиент-сервер” термині ақпаратты өңдеудің екі үрдісінен құралған, бағдарламалық жасақтама сәулетіне жатқан: клиенттік және серверлік.

Клиенттік үрдіс кейбір қызметтерді сұрастырды, ал серверлік үрдіс олардың орындалуын қамтамасыз етті. Бұл ретте бір серверлік үрдіс көптеген клиенттік үрдіске қызмет көрсетеді деп болжалады. Дерекқорды басқарудың осы моделінің аппаратпен іске асырылуы кәсіпорынның жергілікті есептегіш желілерінің құрылуына байланысты болатынын есепке ала отырып, ақпаратты өңдеу үрдісінің осылай ұйымдастырылуын клиент-сервер сәулеті деп атайды.

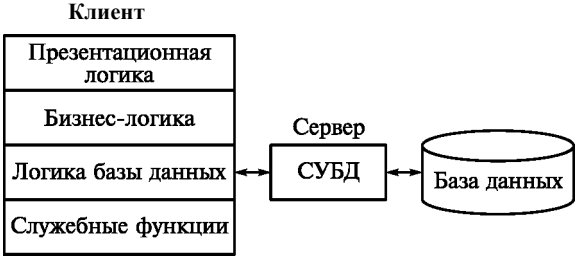
Дерекқорды басқару технологиясына қолданбалы клиент-сервер технологиясының негізгі қағидасы стандартты интерактивті қосымша қызметтерін табиғаты әртүрлі бес топқа бөлуде:

- деректерді енгізу және көрсету қызметтері (Presentation Logic);
- қосымшаның міндеттерін шешудің негізгі алгоритмдерін айқындайтын қолданбалы қызметтер (Business Logic);
- қосымшаның ішіндегі деректерді өңдеу қызметтері (Database Logic);
- ақпараттық ресурстарды басқару қызметтері (Database Manager System);
- алғашқы төрт топ қызметтерінің арасындағы байланыстар рөлін атқаратын қызметтік функциялар.

Клиент-сервер сәулетінде дерекқормен жұмыс істейтін типтік қосымшаның құрылымы 10.2-суретте келтірілген.

Презентациялық логика (Presentation Logic) қосымшаның бір бөлігі ретінде пайдаланушы қосымшаның жұмысын өз экранында көре алушылығымен айқындалады. Бұған пайдаланушы қосымшаның жұмыс істеу барысында көретін немесе толтыратын барлық интерфейстік экранды нысандар жатады. Осы бөлікке пайдаланушының экранына кейбір аралық міндеттердің шешімі немесе анықтамалық ақпарат ретінде шығатынның барлығы жатады. Сондықтан презентациялық логиканың негізгі міндеттері мыналар болып табылады:

- экран суреттерін қалыптастыру;
- ақпараттағы экран нысандарын оқу және жазу;
- экранды басқару;



10.2-сур. Дерекқормен жұмыс істейтін үлгісі қосымшаның құрылымы

• тінтуірдің қозғалыстарын өңдеу және пернетақтаның пернелеріне басу. *Бизнес-логика,* немесе *қосымшалардың логикасы* (Business processing Logic), — бұл қосымшаның нақты міндеттерінің шешім алгоритмдерін айқындайтын қосымша кодының бір бөлігі. Әдетте бұл код C, C++, Visual-Basic және басқалары сияқты әртүрлі бағдарламалау тілдерін пайдалана отырып жазылады.

Деректерді өңдеу логикасы (Data manipulation Logic) — бұл тікелей қосымша ішіндегі деректерді өңдеуге байланысты қосымша кодының бір бөлігі.-

10.1- кесте

Клиент—сервер модельдерінде қосымша компоненттерінің қызметтерін бөлу

Бөліну модельдері	Қосымша компоненттері						Пайдалануы
	Ұсыну логикасының қызметі		Бизнес логика қызметтері		Деректерді басқару қызметі		
Бөлінген көрініс (DP)		DP	RP	DBL	RDM	DDM	
							Клиент
							Сервер
Қашықтықтан көрсету (RP)							
							Клиент
							Сервер
Бөлінген бизнес-логика (DBL)							
							Клиент
Деректерді қашықтықтан басқару (RDM)							
							Клиент
							Сервер
Деректерді бөліп басқару (DDM)							
							Клиент
							Сервер
Бірлескен DBL және DDM							
							Клиент
							Сервер

Ескертпе. Қосымшаның тиісті компоненттері фонмен ерекшеленген.

Деректерді ДҚБЖ басқарады. Деректерге қолжетімдікті қамтамасыз ету үшін SQL тілі пайдаланылады.

Деректерді басқару процессоры (Database Manager System Processing) — бұл ДҚБЖ болып табылады. ДҚБЖ қызметтері қосымшаның бизнес-логикасынан жасырын болу керек, алайда қосымшаның сәулетін қарастыру үшін оларды қосымшаның бөлек бөлігіне ерекшелеу керек.

Орталықтандырылған сәулетте (Host-based processing) қосымшаның осы бөліктері бір ортада орналасады және бір атқарушы бағдарламаның ішінде құрамдастырылады.

Қайта орталықтандырылған сәулетте бұл міндеттер серверлік және клиенттік үрдістер арасында әртүрлі бөлінуі мүмкін. Бөліну сипатына тәуелді бөлінудің келесі модельдерін ерекшелеуге болады (10.1-кесте):

- бөлінген презентация (DP — Distribution Presentation);
- қашықтықтан презентациялау (RP — Remote Presentation);
- бөлінген бизнес-логика (RBL — Remote business logic);
- деректерді бөліп басқару (DDM — Distributed data management);
- деректерді қашықтықтан басқару (RDM — Remote data management).

Осы шартты жіктеме жекелеген міндеттердің серверлік және клиенттік үрдістер арасында қалай бөліне алатынын көрсетеді. Осы жіктемеде қашықтықтан бизнес-логиканы іске асыру мүмкіндігі жоқ. Өз бетімен толық жойыла алмайды, тек бір-бірімен өзара әрекеттесуі мүмкін әртүрлі үрдістер арасында бөлінуі мүмкін.

10.3. Екі деңгейлі модель

Екі деңгейлі модель іс-жүзінде жоғарыда көрсетілген бас қызметтің екі платформада орындалатын екі үрдіс арасында бөліну нәтижесі болып табылады: клиентте және серверде. Ешбір модель таза күйінде болмайды, алайда әрбір екі деңгейлі модельдің тән ерекшеліктерін қарастырайық: деректерді қашықтықтан басқару модельдері және деректерге қашықтықтан қолжетімдік модельдері.

Деректерді қашықтықтан басқару моделі. Сондай-ақ файлдық сервер моделі деп аталады (FS — File Server). Осы модельде презентациялық логика және бизнес-логика клиенттік бөлігінде орналасады. Серверде деректері бар файлдар орналасады және файлдарға қолжетімдік сақталады. Ақпараттық ресурстарды басқару қызметтері осы модельдің клиенттік бөлігінде орналасқан.

Рис. 10.3. Модель файлового сервера

Сурет 10.3. Файылдық сервер моделі

Осы модель қызметтерінің бөлінуі 10.3-суретте ұсынылған.

Осы модельде дерекқордың файлдары серверде сақталады, клиент серверге файлдық командалармен жүгінеді, ал барлық ақпараттық ресурстарды басқару механизмі, яғни метадеректер қоры клиентте орналасады.

Осы модельдің артықшылығы қосымшаның екі өзара әрекеттесетін үрдіске бөлінуінде. Бұл ретте сервер (серверлік үрдіс) сұраныстармен жүгінетін көптеген клиенттерге қызмет көрсете алады. ДҚБЖ осы модельде клиенттік компьютерде орналасқан.

Клиенттік сұранысты орындау алгоритмі мынадай:

1. Сұраныс ЯМД командаларында қалыптастырылады.
2. ДҚБЖ осы сұранысты файлдық командалар реттілігіне аударады.
3. Әрбір файлдық команда клиенттің компьютеріне ақпарат блогын көшіреді, ал ДҚБЖ алынған ақпаратты талдайды; егер алынған блокта сұранысқа жауап болмаса, онда келесі ақпарат блогын көшіру туралы шешім қабылданады және т.б.
4. Ақпаратты серверден клиент компьютеріне көшіру клиенттің сұранысына жауап алынғанға дейін жүзеге асырылады.

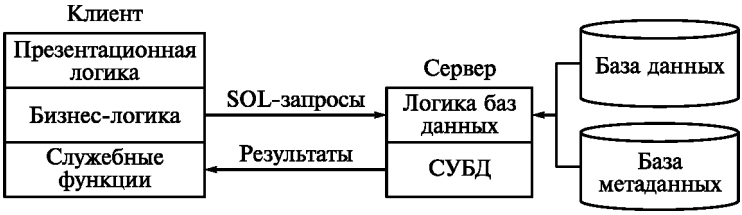
Бұл модельдің мынадай кемшіліктері бар:

• желі бойынша қосымшаға қажетті көптеген блоктар мен файлдардың жіберілуіне байланысты жоғары желілік трафик;

• тек файлдық деректермен айқындалатын, деректерді манипуляциялау операцияларының тар спектрі;

• деректерге қолжетімдік қауіпсіздігінің барабар құралдарының жоқтығы (тек файлдық жүйе деңгейінде қорғау).

Деректерге қашықтықтан қолжетімдік моделі. Қашықтықтан қолжетімдік моделінде (RDA — Remote Data Access) дерекқор серверде сақталады. Серверде ДҚБЖ өзегі де орналасқан. Клиенттің компьютерінде қосымшаның презентациялық логикасы және бизнес-логика орналасқан.



10.4-сур. Деректерге қашықтықтан қолжетімдік моделінің құрылымы

Клиент серверге SQL тіліндегі сұратулармен жүгінеді. Қашықтықтан қолжетімдік моделінің құрылымы 10.4-суретте келтірілген.

Осы модельдің артықшылықтары мынада:

• көрсетілім компоненті мен қолданбалы компонентті клиенттік компьютерге көшіру операциялық жүйеде орындалатын үрдістердің жалпы санын азайта отырып, ДҚ серверін айтарлықтай босатады;

• ДҚ сервері оған тән емес қызметтерден босатылады; процессор немесе сервердің процессорлары сұраныстар мен транзакциялардың деректерін өңдеу операцияларымен толық босатылады;

• желіге жүктеме бірден азаяды, себебі ол бойынша клиенттерден серверге файлдық терминологияға енгізу-шығаруға сұраныстар емес, SQL-ге сұраныстар жіберіледі, ал олардың көлемі айтарлықтай кем. Клиент сұраныстарға жауап ретінде файл блоктарын емес, сұранысқа сәйкес келетін деректерді алады.

RDA-модельдің негізгі артықшылығы — клиент-сервер интерфейсін бірегейлендіру (қосымша-клиент пен сервердің қарым-қатынасындағы стандарт SQL тілі болып табылады).

Бұл модельдің мынадай кемшіліктері бар:

• қосымшаның клиенттің бөлігіндегі қарқынды жұмыс істеу кезінде SQL тіліндегі сұраныс желіні жүктемеден айтарлықтай босата алады;

• себебі осы модельде клиентте презентациялық логика және бизнес-логика орналасады, онда әртүрлі қосымшада баламалы қызметтерді қайталау кезінде тиісті бизнес-логиканың коды әрбір клиенттік қосымша үшін қайталанады. Бұл қосымшаның артық қайталануын тудырады;

• осы модельде сервер пассивті рөл атқарады, сондықтан ақпараттық ресурстарды басқару қызметтері клиентте орындалу керек.

10.4. Дерекқор серверінің моделі

Қашықтықтан басқару моделінің кемшіліктерін жою үшін, мына шарттар қадағалануы керек.

1. ДҚ тұрақты түрде деректермен ғана емес, сондай-ақ объектті деректер арасындағы байланыстармен айқындалатын, пәндік саланың

ағымдағы жағдайын көрсету керек, яғни ДҚ-да сақталатын деректер тұрақты түрде қарама-қайшы болмау керек.

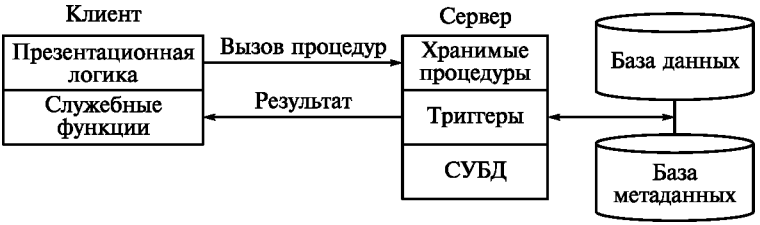
2. ДҚ пәндік саланың кейбір ережелерін, қызмет ететін (business rules) заңдарды көрсету керек. Мысалы, егер қоймада белгілі бір номенклатура бөлшектерінің кейбір жеткілікті қоры (сақтандыру қоры) болғанда ғана зауыт қалыпты жұмыс істей алады; қоймада бөлшекті дайындауға арналған материал жеткілікті болған жағдайда ғана, бөлшек өндіріске жіберіле алады және т.б.

3. ДҚ жағдайын тұрақты бақылап отыру керек, барлық өзгерістерді қадағалап, оларға барабар әсер ету керек. Мысалы, кейбір өлшенетін параметр ауыспалы мәнге жеткен кезде белгілі бір аппаратура сөндірілу керек; тауар қоры рұқсат етілетін нормадан төмен азайған кезде нақты жеткізушіге тиісті тауарды жеткізуге өтінім қалыптастырылу керек және т.б.

4. ДҚ-дағы кейбір жағдайдың пайда болуы қолданбалы міндеттің орындалу барысына анық және жедел әсер ету керек.

5. ДҚБЖ маңызды міндеттерінің бір деректер түрін бақылау болып табылады. Қазіргі сәтте ДҚБЖ синтаксистік тек стандартты-рұқсат етілетін деректер түрін, яғни SQL бір бөлігі болып табылатын, деректерді сипаттайтын тілде - DDL (data definition language)-де айқындалған тілде анықталу керек. Алайда шынайы пәндік салаларда семантикалық құрамдас болып табылатын деректер бар, мысалы, объекттердің координаталары немесе өлшем бірліктері.

Көптеген қазіргі заманғы ДҚБЖ осындай модельді қолдайды: Informix, Ingres, Sybase, Oracle, MS SQL Server. Осы модельдің негізі SQL-серверін бағдарламалау құралы ретінде сақталатын рәсімдер механизмі, ақпараттық қойманың ағымдағы жағдайын қадағалау механизмі ретінде триггерлер механизмі және домен құрылымын қолдау механизмі деп аталатын, деректердің пайдаланушы түріне шектеу механизмі болып табылады. Дерекқордың белсенді серверінің моделі 10.5-суретте көрсетілген.



10.5-сур. Дерекқордың белсенді серверінің моделі

Осы модельде бизнес-логика клиент пен сервер арасында бөлінген. Серверде бизнес-логика сақталатын рәсімдер - ДҚ-да сақталатын және тікелей ДҚБЖ-мен басқарылатын арнайы бағдарламалық модульдер ретінде іске асырылған. Клиенттік қосымша сақталатын рәсімдерді іске қосу командасымен серверге жүгінеді, ал сервер бұл рәсімді орындайды және ДҚ-да көзделген барлық өзгерістерді тіркейді. Сервер клиентке экранға шығару үшін немесе бизнес-логиканың бір бөлігін орындау үшін қажет болатын, орындалған сұраныстың деректерін клиентке қайтарады. Бұл ретте клиент пен сервер арасындағы ақпарат алмасу трафигі азаяды.

Дерекқордың сервер моделіндегі орталықтандырылған бақылау триггер механизмін пайдалана отырып орындалады. Триггерлер де ДҚ бір бөлігі болып табылады.

“Триггер” термині электроникадан алынған және ДҚ жағдайына байланысты арнайы оқиғаларды қадағалау механизмін семантикалық нақты сипаттайды. Триггер ДҚ-дағы белгілі бір оқиға туындаған кезде іске қосылатын, тумблер болып табылады. ДҚБЖ өзегі ДҚ-да құрылған және сипатталған триггерлерді шақыратын барлық оқиғаларға мониторинг жүргізеді және тиісті оқиға туындаған кезде сервер тиісті триггерді іске қосады. Әрбір триггер дерекқорда орындалатын бағдарлама болып табылады. Триггерлер сақталатын рәсімдерді шақыра алады.

Триггерлерді пайдалану механизмі бір триггер іске қосылған кезде басқа триггерлердің іске қосылуын туындататын оқиғалардың туындау мүмкіндігін болжайды.

Сервер осы модельде белсенді болып табылады, себебі клиент қана емес, сондай-ақ сервер триггерлердің механизмін пайдалана отырып ДҚ-дағы деректерді өңдеу бастамашысы болуы мүмкін.

Сақталатын рәсімдер және триггерлер ДҚ сөздігінде сақталады. Олар бірнеше клиентпен пайдаланылуы мүмкін, бұл әртүрлі клиенттік қосымшадағы деректердің өңделу алгоритмінің қайталануын азайтады.

Осы модельдің кемшілігі сервердің өте үлкен жүктемесі болып табылады, себебі ол көптеген клиентке қызмет көрсетеді және мына қызметтерді орындайды:

- сипатталған триггерлерге байланысты оқиғаларды мониторингтейді;
- триггерлерге байланысты оқиғалар пайда болған кезде триггерлердің автоматты іске қосылуын қамтамасыз етеді;
- әрбір триггердің ішкі бағдарламасының орындалуын қамтамасыз етеді;
- пайдаланушылардың сұранысы бойынша сақталатын рәсімдерді іске қосады;
- триггерлерден сақталатын рәсімдерді іске қосады;
- клиентке қажетті деректерді қайтарады;

• ДҚБЖ барлық қызметтерін қамтамасыз етеді (деректерге қолжетімдік, ДҚ-дағы деректердің біртұтастығын бақылау және сақтау, қолжетімдікті бақылау, барлық пайдаланушының бірыңғай ДҚ-мен дұрыс параллельді жұмыс істеуін қамтамасыз ету).

Егер серверге қосымшаның бизнес-логикасының үлкен бөлігін көшірсек, онда осы модельде клиенттерге қойылатын талаптар азаяды. Кейде мұндай модельді жіңішке клиенті бар модель деп атайды. Бұдан бұрын қарастырылған модельдер қалың клиенті бар модель деп атайды.

Серверді жүктеп түсіру үшін үш деңгейлі модель - қосымша серверінің моделі ұсынылған.

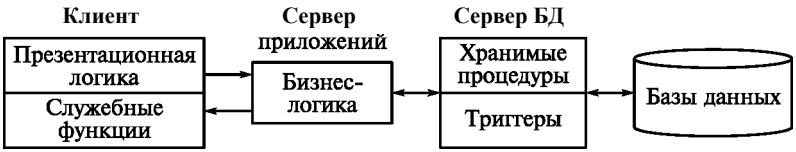
10.5. Қосымша серверінің моделі

Бұл модель екі деңгейлі модельдің кеңеюі болып табылады. Онда клиент пен сервер арасындағы қосымша аралық деңгей енгізіледі. Үш деңгейлі модельдің сәулеті 10.6-суретте келтірілген. Осы аралық деңгей бір немесе бірнеше қосымша серверін құрайды.

Осы модельдегі қосымша компоненттері үш орындаушы арасында бөлінеді: клиент, сервер, дерекқор сервері.

Клиент графикалық пайдаланушы интерфейсін, жергілікті редакторларды қоса алғанда ұсыну логикасын қамтамасыз етеді;клиент компьютер-клиентте орналасқан, жергілікті ДҚ-ға жүгінуін құрауы мүмкін, клиент қосымшасының жергілікті кодын іске қоса алады. Клиент жергілікті немесе ғаламдық желіге қолжетімдікті қамтамасыз ететін, қосымша бөлігінің front-end коммуникациялық қызметтерін атқарады. Клиент пен сервер арасындағы өзара әрекеттесуді қосымша іске асыру бөлінген транзакцияларды басқаруды құрауы мүмкін, бұл клиенттің бөлінген транзакциялар менеджерінің клиенті болатын жағдайларға сәйкес келеді.

Қосымша серверлері сәулеттің жаңа аралық деңгейін құрайды. Қосымша серверлері клиенттің қызметтерін өзара әрекеттесетін жұмыс топтарының бір бөліктері ретінде қолдайды, желілік домендік операциялық орталық сақтайды, бизнес-логиканың аса ортақ ережелерін сақтайды және орындайды, деректер бар каталогтарды қолдайды, хабарламамен алмасуды және әсіресе бөлінген транзакцияларға сұраныстың сақталуын қамтамасыз етеді.



10.7-сур. Қосымшалар серверінің моделі

Осы модельдегі дерекқор серверлері тек ДҚБЖ қызметтерімен айналысады: ДҚ құру және жүргізу қызметтерін қамтамасыз етеді, реляциялық ДҚ біртұтастығын сақтайды, деректер қоймасының қызметін қамтамасыз етеді (warehouse services). Бұдан басқа, ДҚ резервтік көшірмелерін жасау және істен шыққан кейін ДҚ қалпына келтіру, транзакциялардың орындалуын басқару және ескірген (мирасқор болған) қосымшаларға (legacy аррИсайоп) қолдау көрсету қызметтері жүктеледі.

Бұл модель екі деңгейлі модельдерге қарағанда аса үлкен иілімдікке ие. Клиенттер OLAP-қосымшаларының (On-line analytical processing) саласына жататын, дерекқорда күрделі аналитикалық есептеулер жасайтын жағдайда, қосымша серверінің моделінің артықшылықтары аса айқын. Осы модельде клиенттің бизнес-логикасының үлкен бөлігі нақты ДҚБЖ-де іске асырылған, кіріктірілген SQL мүмкіндіктерінен оқшауланған және C, C++, Cobol сияқты бағдарламалау тілдерінде орындалуы мүмкін. Бұл жүйенің төзе алушылығын, оның ауқымдылығын арттырады.

10.6. Дерекқор серверлерінің модельдері

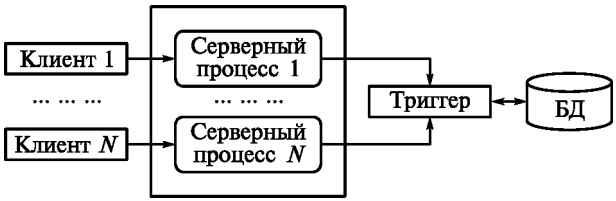
Алғашқы ДҚБЖ құру кезеңінде клиент-сервер технологиясы жаңа ғана пайда бола бастаған. Сондықтан бастапқыда жүйелер сәулетінде “клиент” және “сервер” сияқты процессорлардың өзара әрекеттесуін ұйымдастырудың барабар механизмі болмаған. Қазіргі заманғы ДҚБЖ-де ол іс-жүзінде негізін қалаушы болып табылады және оның іске асырылу тиімділігіне жалпы жүйенің жұмыс істеу тиімділігі тәуелді.

Ұқсас механизмдерді ұйымдастыру түрлерінің эволюциясын қарастырайық. Негізінен бұл механизм серверлік үрдістерді іске асыру құрылымымен айқындалады және жиі *дерекқор серверінің сәулеті* деп аталады.

Бастапқыда, атап өткендей, деректерді басқару (сервер қызметі) және пайдаланушымен өзара әрекеттесу бір бағдарламада біріктірілген модель болған. Оны ДҚ серверлерін дамытудың нөлдік кезеңі деп атауға болады.

Содан кейін деректерді басқару қызметтері сервер деген дербес топқа бөлініп алынған, алайда пайдаланушының сервермен өзара әрекеттесу моделі дерекқор кестелерінің арасындағы “бірге бір” байланыстар құрылымына сәйкес келді (10.7-сур.), яғни сервер тек бір пайдаланушының (сервердің) сұраныстарына қызмет көрсетті, ал бірнеше клиентке қызмет көрсету үшін баламалы сервер санын іске қосу керек.

Серверді жекелеген бағдарламаға ерекшелеу революциялық қадам болды, ол атап айтқанда, желі бойынша өзара әрекеттестіре отырып, серверді бір машинаға, ал пайдаланушысы бар бағдарламалық интерфейсті басқа машинаға орналастыру мүмкіндігін берді.



10.7-сур. “Бірге бір” моделіндегі клиенттік және серверлік үрдістердің өзара әрекеттесуі

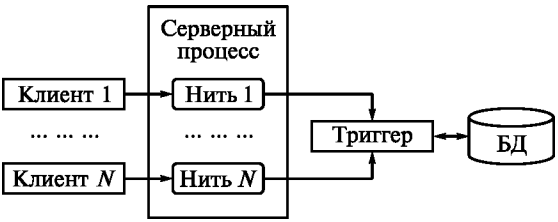
Алайда көптеген пайдаланушыға қызмет көрсету үшін сервердің үлкен көлемін іске қосу қажеттігі осындай жүйенің мүмкіндіктерін қатты шектеді.

Клиенттің көп санына қызмет көрсету үшін серверде бір уақытта жұмыс істейтін серверлік процессордың көп саны іске қосылу керек болатын, ал бұл ЭЕМ ресурстарына қойылатын талаптарды арттырады.

Бұдан басқа, осы модельдің әрбір серверлік үрдісі тәуелсіз ретінде іске қосылған, сондықтан егер бір клиент басқа серверлік процессормен басқа клиент үшін орындалған сұранысты қалыптастырса, онда сұраныс қайтадан орындалады. Мұндай модельде серверлік үрдістердің өзара әрекеттесуін қамтамасыз ету өте қиын. Бұл модель ең қарапайым дәне бірінші болып пайда болған.

“Бірге бір” ақпараттық моделінде туындайтын мәселелер көп клиенттен сұраныстарды өңдеуге қабілетті, сервері жеке бөлінген жүйелер сәулетінде шешіледі. Сервер ғана деректерді басқару монополиясына ие және бір уақытта бірнеше клиентпен өзара әрекеттеседі (10.8-сур.). Логикалық әрбір клиент сервермен сұрату жіберілетін жекелеген жіппен (thread) немесе ағынмен байланысқан. Мұндай сәулет *көп ағынды бір сервер* деген атқа ие болады (multi-threaded). Ол көп пайдаланушы жұмыс істеген кезде туындайтын (trashing) операциялық жүйеге жүктемені азайту мүмкіндігін береді.

Бұдан басқа, көптеген клиенттің бір сервермен өзара әрекеттесу мүмкіндігі бөлінетін объекттерді (ашық файлдардан бастап жүйелік каталогтағы деректермен аяқтай отырып) толық шамада пайдалану мүмкіндігін береді, бұл жадтың тұтынушылығын және операциялық



10.8-сур. Көп ағынды бір серверлі сәулет

жүйе үрдістерінің жалпы санын айтарлықтай азайтады. Мысалы, “бірге бір” моделі бар жүйе 100 пайдаланушы үшін ДҚБЖ үрдістерінің 100 көшірмесін жасайды, сол уақытта көп ағынды сәулеті бар жүйеге ол үшін тек бір серверлік үрдіс қажет болады.

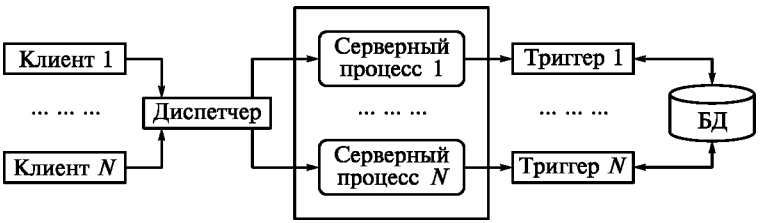
Алайда мұндай шешімнің өз кемшіліктері бар. Себебі серверлік үрдіс тек бір процессорда орындалуы мүмкін, мультипроцессорлы платформалар үшін ДҚБЖ қолданылуына табиғи шектеу туындайды. Егер компьютердің, мысалы, төрт процессоры болса, онда бір сервері бар ДҚБЖ қалған үшеуіне жүктеме салмай, тек біреуін пайдаланады.

Кейбір жүйелерде бұл мәселе аралық диспетчерді енгізумен шешіледі. Ұқсас сәулет *виртуалды сервердің сәулеті* деп аталады (virtual server) (10.9-сур.).

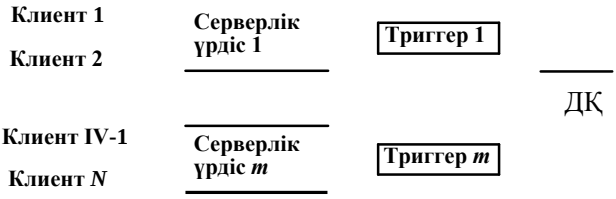
Осы сәулетте клиенттер шынайы серверге емес, серверге сұратуларды диспетчерлеу қызметін атқаратын, диспетчер деп аталатын аралық буынға қосылады. Бұл жағдайда көп процессорлы платформаны пайдалануға шектеу жоқ. Сервер саны жүйедегі процессор санымен келісілуі мүмкін.

Алайда осы сәулет те кемшіліксіз емес, себебі мұнда жүйеге клиент пен сервер арасында орналастырылатын жаңа қабат қосылады, бұл серверлерді жүктеу (load balancing) теңгеріміне қолдау көрсету ресурстардың тұтынылуын арттырады және клиент-сервер өзара әрекеттестігін басқару мүмкіндігін шектейді. Біріншіде, нақты клиенттен сұранысты нақты серверге жолдау мүмкін емес; екіншіден, серверлер тең құқылы болады, себебі сұраныстарға қызмет көрсету үшін басымдылықтар орнату мүмкіндігі жоқ.

Клиент пен сервер арасындағы осындау өзара әрекеттестікті ұйымдастыру бірнеше кассир терезесі және арнайы банк қызметкері, әрбір келушіні (клиентті) бос кассирге (өзекті серверге) жолдайтын зал әкімшілігі болатын, банк баламасы деп қарастырылуы мүмкін.



10.9-сур. Виртуалды сервер сәулеті



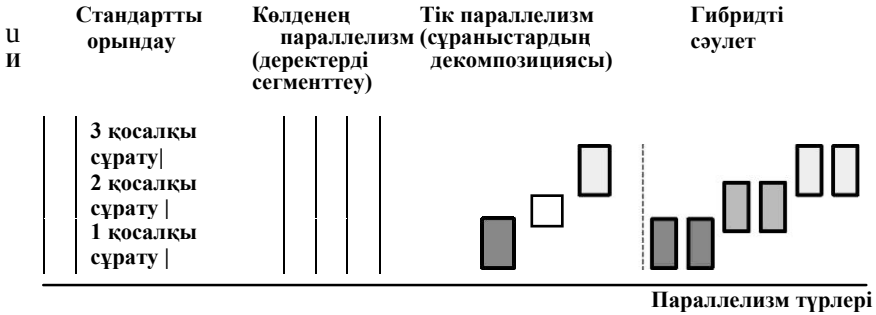
10.10-сур. Көп ағынды мультисерверлі сәулет

Барлық келішу тең құқылы (басымдылығы тең) болғанда, жүйе қалыпты жұмыс істейді, алайда арнайы терезеде қызмет көрсетілуі қажет, басымдығы артық келуші келген кезде проблемалар туындайды. Клиенттердің басымдылығын есепке алу транзакцияларды жедел өңдеу жүйелерінде әсіресе маңызды, алайда диспетчері бар жүйелер сәулеті дәл осы мүмкіндікті бере алмайды.

Мультипроцессорлы платформалар үшін ДҚБЖ проблемаларын уақытында шешу дерекқордың бірнеше серверін және әртүрлі процессорда іске қосу мүмкіндігінде. Бұл ретте әрбір сервер көп ағынды болу керек. Егер осы екі шарт орындалса, онда 10.10-суретте келтірілген бірнеше сервері бар көп ағынды сәулет туралы сөз қылу негізі бар.

Осындай сәулетті *көп жіпті мультисерверлі сәулет* деп те атайды. Осы сәулет бір пайдаланушылық сұраныстың бірнеше серверлік үрдіспен орындалуының параллельденуін қамтамасыз етеді.

Бұл жағдайда пайдаланушы сұратуы параллельді орындалуы мүмкін бірқатар қосалқы сұраныстарға бөлінеді, ал олардың орындалу нәтижелері сұранысты орындаудың ортақ нәтижесіне біріктіріледі. Сонда сұраныстардың жылдам орындалуын қамтамасыз ету үшін олардың қосалқы сұраныстары жекелеген серверлік үрдістерге жіберілуі мүмкін, содан кейін алынған нәтижелер жалпы нәтижеге біріктірілу керек (10.11-сур).



10.11-сур. Көп жіпті мультисерверлі сәулет

Осы жағдайда серверлік үрдістер бұдан бұрын қарастырылған тәуелсіз үрдістер болып табылмайды. Осы серверлік үрдістерді *жіптер* (treads) деп атау қабылданған. Пайдаланушылардың көптеген сұраныстарының жіптерін басқару ДҚБЖ-ден қосымша шығындарды талап етеді, алайда деректер қоймасында ақпаратты жылдам өңдеу кезінде осындай тәсілдің келешегі бар.

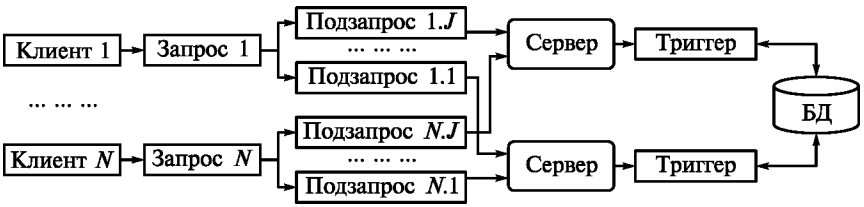
10.7. Параллелизм түрлері

Сұраныстарды параллелдеудің мынадай бағдарламалық-аппараттық тәсілдері қарастырылады: көлденең, тік және аралас параллелизм.

Көлденең параллелизм. Бұл параллелизм ДҚ-да сақталатын ақпарат бірнеше физикалық сақтау құрылғыларына - бірнеше дискке бөлінетін жағдайда туындайды. Бұл ретте бір қатынастың ақпараты көлденеңінен бөліктерге бөлінеді. Параллелизмнің осы түрін кейде деректерді параллелдеу немесе сегментация деп атайды. Параллельдікке бірдей операцияны орындау арқылы, мысалы, әртүрлі физикалық сақталатын деректерді сүзумен қолжеткізуге болады. Бұл операциялар әртүрлі үрдіспен параллельді орындалуы мүмкін - олар тәуелсіз. Біртұтас сұранысты орындау нәтижесі жекелеген операцияларды орындау нәтижелерінен құралады.

Деректерді тиісті сегменттеу кезінде осындай сұранысты орындау уақыты осы сұранысты дәстүрлі тәсілдермен бір үрдіспен орындау уақытына қарағанда кем.

Тік параллелизм. Бұл параллелизмге пайдаланушының сұратуын құрайтын операцияларды конвейермен орындау арқылы қол жеткізуге болады. Бұл тәсіл ДҚБЖ өзегімен реляциялық операцияларды орындау моделін күрделендіруді талап етеді. Ол ДҚБЖ өзегі функционалды компоненттеріне негізделе отырып, сұранысты декомпозициялай алатынын болжайды; бұл ретте бірқатар қосалқы сұраныс сұранысты орындаудың жекелеген қадамдары арасындағы байланысты азайта отырып, параллельді орындалуы мүмкін.



10.12-сур. Тік параллелизм кезінде сұраныс жасау сұлбасы

10.13-сур. Аралас параллелизм кезінде сұранысты орындау сұлбасы

Мысалы, реляциялық алгебра операцияларының келесі реттілігін қарастырайық:

1. R5 = R1 [A,C].
2. R6 =R2 [A,B,D].
3. R7 = R5[A > 128].
4. R8 = R5[A]R6.

Онда бірінші және үшінші операцияны біріктіріп, екінші операциямен параллельді орындауға болады, содан кейін нәтижелермен соңғы - төртінші операцияны орындауға болады.

Ұқсас сұратуды орындаудың жалпы уақыты, әрине, төрт операциядан тұратын реттілікті орындаудың дәстүрлі тәсіліне қарағанда айтарлықтай кем болады (10.12-сур.).

Аралас параллелизм. Бұл параллелизм көлденеңінен және тік параллелизмнің гибриді болып табылады (10.13-сур.).

Параллелизмнің барлық түрі қосымшаларда қолданылады, онда олар өте үлкен деректер көлемімен күрделі сұраныстарды орындау уақытын айтарлықтай қысқарту мүмкіндігін береді.

Бақылау сұрақтары

1. Мына ұғымдарға анықтама беріңіз:
   - ДҚ топологиясы немесе бөлінген ДҚ құрылымы;
   - жергілікті дербестік;
   - қашықтықтан сұраныс жасау;
   - төлінген транзакцияға қолдау көрсету;
   - презентациялық логика;
   - бизнес-логика.

2. Қандай екі деңгейлі модельдерді білесіз? Олардың артықшылықтары мен кемшіліктерін атаңыз.

3. Дерекқор ұйымдарының келесі сәулетінің сипаттамаларын атаңыз:
   - көп ағынды бір серверлі сәулет;
   - виртуалды сервері бар сәулет;
   - көп жіпті мультисерверлі сәулет.

4. Сұранысты не үшін параллельдейді және параллелизмнің қандай түрлерін білесіз?

SQL SERVER 2000 ЖЕЛІЛІК ДЕРЕККОР

11.1. SQL Server 2000 компоненттері

SQL Server 2000 ірі кәсіпорындар ауқымындағы дерекқорларды басқарудың қазіргі заманғы жүйелерінің бірі болып табылады.

SQL Server 2000-мен кәсіби жұмыс істеу үшін оның қызмет ету қағидаттарын түсіну керек, қандай да бір жағдайда қай компонентті пайдалану қажет екенін білу керек. SQL Server 2000 компоненттерін (қызметтерін), олардың тағайындалуын және пайдалану әдістерін қарастырайық.

Windows NT немесе Windows 2000 операциялық жүйенің басшылығында жұмыс істейтін көптеген серверлік өнімдер сияқты Microsoft SQL Server 2000 операциялық жүйе қызметтерінің жинағы түрінде іске асырылған, олардың әрбіреуі өз бетімен іске қосылады және белгілі бір міндеттер тобы үшін жауап береді.

SQL Server қызметтер тізімін келтірейік:

- MSSQLServer;
- SQLServerAgent;
- Microsoft Search;
- Microsoft Distributed Transaction Coordinator.

SQL Server 2000 жекелеген қызметтер түрінде іске асыру ДҚБЖ-ге операциялық жүйенің бөліктері ретінде жұмыс істеу, меншік қолжетімдік құқықтарына ие болу және осы сәтте компьютерде жұмыс істеп отырған пайдаланушыға тәуелсіз болу мүмкіндігін береді. Windows 95/ 98 операциялық жүйесі осы қызметтердің жұмысына қолдау көрсетпейді, сондықтан SQL Server 2000 осы операциялық жүйе басшылығымен жұмыс істеуі үшін автоматты түрде қызметтер эмуляциясы орындалады. SQL Server 2000 әрбір қызметін толығырақ қарастырайық.

MSSQLServer қызметі. MSSQLServer қызметі ДҚБЖ өзегі болып табылады және барлық негізгі операцияны орындайды. MSSQLServer қызметінің міндеттеріне пайдаланушыларды тіркеу, олардың қолжетімдік құқықтарын тіркеу, қосылысты орнату, пайдаланушылардың дерекқорға жүгінуіне қызмет көрсету, сақталатын рәсімдерді орындау, дерекқор файлдарымен жұмыс істеу және тағы басқасы.

MSSQLServer қызметінің функцияларына жүйелік ресурстардың пайдаланылуын бақылау кіреді. MSSQLServer қызметі жүйеден мерзімді бос ресурстардың бары туралы сұрайды және олар жеткілікті болғанда қосымша жад немесе процессордың уақыт бөледі. Алынған ресурстар барлық қосылған пайдаланушы арасында бөлінеді, сөйтіп

сұратуды өңдеудің ең жоғарғы өнімділігіне қол жеткізіледі. Көп процессорлы жүйені пайдалану кезінде пайдаланушылардың “ауыр” сұратуларын барлық қолжетімді процессорлар арасында параллелдеу орындалады.

Барлық қалған қызметтері ДҚБЖ иілімдігі мен функционалдығын қосатын, MSSQLServer қызметінің кеңеюі ретінде қарастыруға болады. MSSQLServer қызметі әрқашан бірінші болып іске қосылады. Тек сәтті басталғаннан кейін ғана басқа қызметтер іске қосылуы және өз жұмысын бастай алуы мүмкін.

SQLServerAgent қызметі. SQLServerAgent қызметі ең алдымен СУБД әкімшілігін автоматтандыруға арналған. Осы қызметтің міндетіне тапсырмаларды автоматты іске қосу және операторларға сервер жұмысындағы істен шығу туралы хабарлау кіреді. SQLServerAgent қызметінің көмегімен белгілі бір уақытта әртүрлі міндетті іске қосуға болады, бұл әкімшіні кертартпа жұмыстың көп бөлігін жоюы мүмкін. Мысалы, әкімші резервтік көшірме жасау және пайдаланушының кем белсендігі кезінде дерекқордағы ақпараттың біртұтастығын тексеру операцияларының автоматты түрде орындалуын жоспарлауы мүмкін. Бұл ретте әкімшіге операцияларды орындау барысын бақылау қажет болмайды.

SQLServerAgent қызметі орындайтын операциялардың көп бөлігі MSSQLServer қызметімен орындалатын жүйелік сақталатын рәсімдер түрінде іске асырылған. SQLServerAgent қызметінің жұмысында үш типті объект қолданылады:

- Jobs (тапсырмалар);
- Operators (операторлар);
- Alerts (оқиғалар).

Міндеттерді автоматты іске қосу кестесін қоса алғанда, барлық осы объект туралы ақпарат жүйелік Msdb дерекқорында сақталады. SQLServerAgent әрбір басталған сайын осы дерекқордың мазмұнын талдайды. Егер қызметті іске қосу кезінде кешіктірілген міндеттер жиналып қалса немесе конфигурацияланған оқиға туындаса, онда қызмет тиісті әрекеттерді орындайды.

Тапсырмаларды, операторларды және оқиғаларды басқару үшін әртүрлі әдісті пайдалануға болады. Ең қолайлысы Enterprise Manager утилитасының графикалық интерфейсін пайдалану. Екінші тәсіл жүйелік сақталатын рәсімдерді және Transact-SQL командаларын шақыруда болып табылады. Үшінші тәсіл SQL-DMO интерфейсіне жүгінуді болжайды. Соңғы жағдайда тапсырмалармен, операторлармен және оқиғалармен жұмыс істеу интерфейсін қамтамасыз ететін, өзінің меншік қосымшаларын жазуға болады.

SQLServerAgent қызметінің білікті жұмыс істеу тәсілі дерекқорды сүйемелдеуге шығындарды, атап айтқанда, операторлар мен әкімшілер санын азайту арқылы қысқарта алады. SQLServerAgent қызметін қолданудан экономикалық әсер кәсіпорын көлеміне пропорционалды: кәсіпорын үлкен болған сайын, SQL Server 2000 мүмкіндіктерінен пайда соғұрлым үлкен.

Jobs объекттері. Осындай типті объекттер автоматты түрде орындалуы қажет міндеттерді сипаттайды. Әрбір міндет үшін іске қосудың (schedule) бір немесе бірнеше кестесі көрсетіледі. Бұдан басқа, тапсырма талап бойынша (on demand), яғни қолмен орындалуы мүмкін. Әрбір тапсырма бір немесе одан артық қадамнан (step) тұрады. Қадам ретінде мыналар болуы мүмкін:

- Transact-SQL командасы немесе сұранысы;
- репликацияның қосалқы жүйесін басқару командалары;
- Windows командалық жол мен қосымшаның утилиттері;
- VB Script немесе JavaScript тілдерінде жазылған скрипттер және т.б.

SQLServerAgent қызметі тапсырмалардың дұрыс орындалуын бақылау, көп қадамды тапсырмаларды жасау мүмкіндігіне ие. Қадамдар өзара белгілі бір ережелермен байланысты болуы мүмкін. Мысалы, егер дерекқордың біртұтастығын тексеру сәтті аяқталса, онда қызмет деректердің резервтік көшірмесін жасайды; кері жағдайда, сервер әкімшіге электронды пошта бойынша немесе пейджерге тиісті хабарландыру жібере алады. Қызмет міндеттердің белгілі бір уақытта және сервердің аз жүктелген уақытында орындалуын қамтамасыз ете отырып, іске қосу уақытын икемді басқару мүмкіндігін береді.

Operators объекттері. Осындай типті объекттер сервердің жұмыс күйінде сақталуы үшін жауап беретін, қызметкер-операторларды сипаттайды. Шағын ұйымдарда әдетте оператор мен әкімшінің рөлін бір адам біріктіреді. Үлкен кәсіпорындар мен корпорацияларда әкімші мен операторлардың рөлдері жиі бірнеше адамға бөлінген. Әкімші тек жауапкершілікті жұмысты орындайды, мысалы дерекқорларды жоспарлау, құру және өзгерту. Оператор жиі дерекқорлардың резервтік көшірмесін жасау, пайдаланушыларды қосу, деректердің біртұтастығын бақылау және т.б. кертартпа жұмыспен айналысады. Егер ұйым үлкен болса, онда мамандандырылған операторларды пайдалануға болады. Мысалы, операторлардың біреуі резервтік көшірме жасау операцияларын орындау үшін жауапты, екіншісі деректердің біртұтастығын бақылайды және т.б. Операторлардың әрбіреуі қызметіне қатысты хабарлама алу керек. Резервтік көшірме жасау операторы қажетсіз бұғаттау мәселелерін шеше бастағаны қажет емес.

SQL Server 2000 өз жұмысының параметрлерін қадағалайды және кемшіліктерді байқаған кезде, мысалы, дискте бос кеңістік болмаса, операторға хабарлай алады. Ол үшін SQLServerAgent қызметі пайдаланылады.

Алдымен операторларды конфигурациялау керек және оқиғаларды көрсету керек, соңғысы туындаған кезде операторларға хабарландыру жіберіледі. SQLServerAgent қызметі операторларға хабарлау үшін белгілі бір хабарламаларды электронды пошта бойынша жібере алады немесе тікелей оператордың пейджеріне жібере алады. Бұдан басқа, операторға хабарлау үшін net *send командасын шақыруға рұқсат етіледі, оның көмегімен жергілікті желі бойынша хабарлама жіберуге болады. Кімде-кім операторға сервердің жұмысындағы проблема туралы хабарлайды деген сеніммен, команданың іске қосылуын хабарламаны желінің барлық пайдаланушысы ала алатындай дәлдеуге болады. Бірақ жиі* net send командасы хабарламаны нақты пайдаланушыға жіберу үшін қызмет етеді.

Alerts объекттері. Alerts типті объекттер SQL Server 2000 әсер етуі қажет оқиғаларды сипаттайды. Сервер сипатталған оқиға пайда болған кезде SQLServerAgent қызметінің көмегімен бір немесе бірнеше операторға сервердің жұмысындағы кемшіліктердің байқалғаны туралы хабарландыру жібереді. SQL Server 2000 оқиғалары сервер жұмысының шамамен барлық аспекттерін қамтиды, бұл SQL Server 2000 жұмысын тиімді бақылау мүмкіндігін береді. Операторларға сервердің жұмыс істеу параметрлері туралы білу үшін, әрқашан сервердің жанында болу міндетті емес. Істен шығу анықталған кезде оператор ғимаратта болмауы мүмкін, бірақ ол пейджерге хабарландыру алып, қажетті әрекеттерді, соның ішінде қашықтықтан қажетті шаралар қабылдай алады. Сипатталған тәсіл үлкен ұйымдардағы дерекқорды сүйемелдеуге шығындарды азайту мүмкіндігін береді. Кәсіпорынның әрбір сервері үшін дербес оператордың қажеттігі жоқ.

Microsoft Search (MSSearch) қызметі. Full-Text Search деп аталатын Microsoft Search қызметі дерекқордың кестелерінде символды ақпаратты іздеу үшін пайдаланылады. Microsoft Search қызметі толық мәтіндік іздеуді орындау мүмкіндігін береді. Толық мәтіндік іздеу технологиясы нақты сөздер мен сөз тіркестерді ғана емес, сондай-ақ мағынасы мен жазылуы ұқсас сөздерді қалпына келтіру мүмкіндігін береді. Толық мәтіндік іздеуді іске асыру үшін толық мәтіндік каталогтар (full-text catalog) және толық мәтіндік индекстер бар (full-text index). Толық мәтіндік каталогтар мен индекстердің деректері негізгі деректерден жеке арнайы файлдарда сақталады. Осы файлдармен жұмыс істеу бойынша барлық әрекеттерді MSSearch қызметі жүзеге асырады. MSSQLServer және MSSearch қызметтері арасындағы байланыс арнайы жеткізуші арқылы жүзеге асырылады (full-text provider).

MSSearch қызметі мерзімді дерекқор кестелерінің мазмұнын талдайды және толық мәтіндік каталогтар мен индекстерді жаңартады (repopulation).

Толық мәтіндік индексті қайтадан жасау қажет болса, онда индексті қайта құру (rebuild) керек. Осындай тәсілдің нәтижесі толық мәтіндік іздеудің деректерін негізгі деректерден жеке басқару мүмкіндігі болып табылады. Әкімші осы толық мәтіндік индекстердің жаңартылу интервалдарын дәлдеу керек. Бұдан басқа, толық мәтіндік іздеудің файлдарының резервтік көшірмесін жасау және қалпына келтіру операцияларын негізгі деректерден жеке орындау керек.

Microsoft Distributed Transaction Coordinator (MSDTC) қызметі. SQL Server 2000 пайдаланушыларға бір уақытта бірнеше дереккөзбен жұмыс істеу мүмкіндігін береді. Пайдаланушылар бір сұратуда бір немесе әртүрлі серверде сақталатын, әртүрлі дерекқорға жүгінуі мүмкін. Бұдан басқа, пайдаланушылар SQL Server 2000 серверлеріне ғана емес, сондай-ақ OLE DB технологиясымен жұмыс істейтін кез-келген дерекөздерге жүгінуі мүмкін. Осы технология Oracle, FoxPro, MS Access сияқты реляциялық дереккөздерге ғана емес, сондай-ақ MS Excel кітабының мәтіндік файлдары сияқты реляциялық емес дереккөздерге және басқа қосымшаларға жүгіну мүмкіндігін береді.

Бір транзакциядан көптеген дереккөзге жүгіну үшін SQL Server 2000 бөлінген транзакцияларды (distributed transaction) пайдаланады. Бөлінген транзакцияларды басқару үшін бөлінген транзакциялардың үйлестірушісі болады (Distributed Transaction Coordinator). SQL Server 2000-де бөлінген транзакцияларды үйлестіруші MSDTC қызметі түрінде іске асырылған. Бұл қызмет бөлінген транзакцияларды орындай бастау қажет болатын жағдайларды автоматты түрде қадағалайды. Кейбір жағдайларда пайдаланушы оның транзакциясы бөлінген ретінде орындалып жатқанын сезбеуі мүмкін. MSDTC қызметі пайдаланушыдан бөлінген транзакцияларды өңдеу бойынша барлық әрекеттерді жасырады. Бөлінген транзакциялар әрбір дерекөзде бөлінген транзакциялар үйлестірушісімен ашылатын, көптеген жергілікті транзакция ретінде іске асырылады. MSDTC қызметі бөлінген транзакцияның барлық қатысушыларына деректердің біртұтастығы қамтамасыз етілетіндей барлық транзакцияларды синхрондайды. Бұған арнайы екі фазалы өзгерістер хаттамасын (2PC, two-phase commit protocol) пайдалана отырып қол жеткізуге болады.

11.2. SQL Server 2000 жүйелік дерекқорлары

SQL Server 2000 өз жұмысында бірнеше жүйелік дерекқорды пайдаланады. Осы дерекқорлар SQL Server 2000 орнату кезінде автоматты түрде құрылады және жойылмау керек. Серверді дәлдеу бойынша бүкіл ақпарат осы дерекқорларда сақталады. Оларды бүкіл жүйелік және пайдаланушылық ақпарат сақталатын, Windows операциялық жүйесінің тізілімімен салыстыруға болады. Тізілімді жою немесе зақымдау жүйенің бұзылуына әкеледі. SQL Server 2000 жүйелік дерекқорларында да баламалы жағдай байқалады. Жүйелік дерекқорлардың тізімін келтірейік:

- Master;

- Tempdb;
- Msdb.

Пайдаланушылар операциялық жүйенің тізілімімен жұмыс істеу кезінде жиі арнайы құрал-саймандарды, мысалы басқару панелінің утилиттерін таңдайды. Тізіліммен тікелей жұмыс істеу ұсынылмайды, себебі шағын өзгерістердің өзі жүйенің жұмыс істеу қабілетін айтарлықтай зақымдай алады. Жүйелік дерекқорлармен жұмыс істеу кезінде де осылай жасау керек. *Select, Insert, Update, Delete командаларының көмегімен тікелей жұмыс істеу ұсынылмайды. Пайдаланушылар Select командасы арқылы деректерді тек оқи алады. Жүйелік кестелердегі деректерді өзгерту үшін* SQL Server 2000-де жүйелік сақталатын рәсімдер жинағы бар, олардың көмегімен серверді әкімшілеу бойынша шамамен кез-келген әрекетті орындауға болады.

Master дерекқоры. Осы жүйелік дерекқор SQL Server 2000 басты дерекқоры болып табылады. Ол Windows операциялық жүйесінің тізілімі қызметін атқарады. Қалған жүйелік дерекқорлардың екінші кезектегі мағынасы бар және оларды қосалқы деп есептеуге болады. Master дерекқорында пайдаланушылық дерекқор серверіндегі сервердің конфигурация параметрлері туралы, серверге қолжетімдігі бар пайдаланушылар туралы бүкіл жүйелік ақпарат және басқа жүйелік ақпарат сақталады.

Master дерекқоры өздігінен SQL Server 2000 орнату каталогының Data каталогында құрылады. Дерекқор екі файлдан тұрады:

- Master.mdf — деректерді құрайтын, дерекқордың негізгі файлы. Осы файлдың көлемі орнатқаннан кейін 8 Мбайт құрайды;
- Master.ldf — транзакциялар журналын сақтауға арналған дерекқор файлы. Осы файлдың көлемі орнатқаннан кейін 1 Мбайт құрайды.

Model дерекқоры. Осы жүйелік дерекқор жаңа дерекқор құру үшін шаблон болып табылады. SQL Server 2000-де жаңа дерекқорды құру технологиясы Model дерекқорын сервер көрсетілген орынға көшіретіндей және тиісті түрде атын өзгертетіндей құрылған. Егер дерекқорды құру кезінде атынан басқа ешқандай параметр көрсетілмесе, онда жаңа дерекқор Model дерекқорының толық көшірмесі болып табылады. Егер құрылатын дерекқордың көлемі мен файлдар құрамы айқын көрсетілсе, онда көшірмесі жасалған дерекқор тиісті түрде өзгертіледі. Алайда кез-келген жағдайда негізі ретінде Model дерекқоры пайдаланылады.

Model дерекқорының параметрлерін өзгерте отырып, өздігінен құрылатын дерекқордың параметрлерін басқаруға болады. Бұдан басқа, Model дерекқорын дерекқордың мазмұны мен қасиеттеріне корпоративтік стандарт ретінде пайдалануға болады, яғни әкімші Model дерекқорында кестелер жинағын және әрбір дерекқорда болу қажет сақталатын рәсімдерді құра алады.

SQL Server 2000 орнатқаннан кейін Model дерекқорының көлемі 1,5 Мбайт болып орнатылған. Model дерекқоры Data каталогында орналасқан және әрбіреуінің көлемі 0,75 Мбайт болатын екі файлдан тұрады:

- Model.mdf — деректерді құрайтын, дерекқордың негізгі файлы;
- Model.ldf — транзакциялар журналын сақтау үшін пайдаланылатын дерекқор файлы.

Tempdb дерекқоры. Tempdb дерекқоры (толық атауы — Temporary DataBase) пайдаланушылармен жұмыс істеу сеансы кезінде құрылатын, барлық уақытша объекттерді сақтауға арналған. Егер кестелер немесе көріністер сияқты тұрақты объекттер пайдаланушы дерекқорында жасалса, онда уақытша объекттер Tempdb дерекқорында туындайды. Tempdb дерекқорына қолжетімдік автоматты түрде барлық пайдаланушыда болады және әкімші осы дерекқорға қолжетімдік ұсыну үшін ешқандай әрекет қабылдамау керек.

Tempdb дерекқорының айрықша ерекшелігі сервер ақталған сайын жойылуы болып табылады. Әрине, пайдаланушылар жасаған барлық уақытша объекттер де жойылады. SQL Server 2000 келесі іске қосу кезінде Tempdb дерекқоры қайтадан құрылады. Tempdb дерекқорын құру кезінде, пайдаланушылық дерекқорлары үшін сияқты, негіз ретінде Model дерекқоры қолданылады, бұл ретте оның барлық қасиеттері қалады. Әкімші Model дерекқорының параметрлерін өзгерте отырып, осыны есепке алу керек. Осы дерекқордың параметрлерін дұрыс конфигурацияламау барлық пайдаланушының жұмысына жағымсыз әсер етуі мүмкін. Бұдан басқа, Tempdb дерекқорының параметрлерін жоспарлау кезінде дисктегі бос кеңістікке қойылатын талаптарды есепке алу керек. Барлық дерекқор үшін сияқты Tempdb үшін дерекқор файлдарының автоматты өсу мүмкіндігі сақталады. Пайдаланушылар Tempdb дерекқорының ресурстарына қарқынды жүгінген кезде өседі. Осы дерекқордың бастапқы көлемі мен өсу қадамын дұрыс таңдау керек. Осы параметрлерді дұрыс конфигурацияламаса, жүйенің өнімділігі айтарлықтай азаюы мүмкін.

Tempdb дерекқоры SQL Server 2000 орнату каталогының Data каталогында орналасатын екі файлдан тұрады:

- Tempdb.mdf — уақытша объекттерді құрайтын, дерекқордың негізгі файлы. Осы файлдың көлемі орнатқаннан кейін 8 Мбайт құрайды;
- Tempdb.ldf — транзакциялар журналы. Осы файлдың көлемі орнатқаннан кейін 0,5 Мбайт құрайды.

Msdb дерекқоры. Msdb жүйелік дерекқоры SQL Server 2000 әкімшілендірілуін және басқарылуын автоматтандыруға қатысты болатын SQLServerAgent қызметі пайдаланатын ақпаратты және операторлар мен оқиғалар туралы ақпаратты орналастыруға арналған.

11.3. SQL Server 2000 құрал-саймандары

Әдетте әкімшілеу құрал-саймандары SQL Server 2000 орнату кезінде орнатылады. Бұған қарамастан олар жекелеп қосылуы мүмкін. SQL Server 2000 құрал-саймандарын қарастырайық.

Enterprise Manager. Бұл құрал-сайман әртүрлі міндеті орындау үшін негізгі болып табылады:

- Қауіпсіздік жүйесін басқару;
- Дерекқор мен оның объекттерін құру;
- Резервтік көшірмелерді жасау және қалпына келтіру;
- Репликацияның қосалқы жүйесін конфигурациялау;
- SQL Server 2000 қызметтерінің жұмыс параметрлерін басқару;
- Автоматтандыру қосалқы жүйесін басқару;
- Қызметтердің жұмысын іске қосу және тоқтату;
- Байланысқан және қашықтықтағы серверлерді конфигурациялау;
- DTS пакеттерін құру, басқару және орындау.

Келтірілген тізіммен Enterprise Manager барлық қолданылу саласы таусылмайды және кеңеюі мүмкін. Алайда осы құрал-сайманның маңыздығын түсіну үшін аталған міндеттер жеткілікті.

SQL Server 2000 әкімшілік міндеттерінің үлкен бөлігі келесі әдістермен орындалуы мүмкін:

- Transact-SQL құралдарын пайдалану;
- Enterprise Manager графикалық интерфейс арқылы;
- шеберлер арқылы (wizards).

Әдістерді атау тәртібі олармен жұмыс істеу күрделілігін азайтуға сәйкес келеді. Ең күрделісі Transact-SQL құралдарымен міндеттерді орындау болып табылады, себебі бұл командалар мен сақталатын рәсімдердің синтаксисін білуді талап етеді. Transact-SQL құралдарын пайдалану пайдаланушыға жүйелік деректерге тікелей қолжетімдікті қамтамасыз етеді.

SQL Server Service Manager. SQL Server Service Manager утилитасының жалғыз міндеті пайдаланушыға қолайлы іске қосу механизмін ұсыну, SQL Server 2000 қызметтерін тоқтату және тоқтата тұру. Бұдан басқа, ол операциялық жүйені жүктеу кезінде қандай да бір қызметті автоматты іске қосуға тыйым салу немесе рұқсат ету мүмкіндігін береді.

Service Manager утилитасы SQL Server 2000 орнату кезінде орнатылады және өздігінен операциялық жүйені жүктеу кезінде автоматты түрде іске қосылады.

SQL Server Profiler. SQL Server Profiler утилитасы — бұл әкімші SQL Server 2000 қандай да бір аспекттерін бақылай алатын графикалық құрал-сайман.

Осы утилитаның жұмыс істеу негізінде Performance утилитасының жұмыс істеу негізіндегі сияқты қағида жатыр. Пайдаланушылық сұраныстарды, сақталатын рәсімдерді, Transact-SQL командаларын орындау кезінде, серверге қосылу және ажыратылу кезінде, сондай-ақ көптеген басқа әрекеттер кезінде SQL Server 2000 өзегі жүйелік кестелерде операцияның орындалу барысы туралы әртүрлі ақпаратты сақтайды. Бұл ақпарат арнайы сақталатын рәсімдер арқылы алынуы мүмкін. SQL Server Profiler утилитасы осы сақталатын рәсімдерді қажетті ақпарат алу үшін пайдаланады. Содан кейін алынған деректер графикалық интерфейс арқылы қолайлы түрде ұсынылады. Алайда пайдаланушылар тікелей сақталатын рәсімдерге жүгіне отырып, SQL Server 2000 үрдістері туралы ақпараттарды ала алады. Осы сақталатын рәсімдер негізінде SQL Server 2000-дің қажетті нысанда жұмыс істеуі туралы ақпаратты көрсететін меншік қосымшаны жазуға болады.

QL Server 2000 жұмысының мониторингі оқиғаларды (events) қадағалауға негізделеді. Оқиға SQL Server 2000 өзегімен генерацияланады және бақылауға болатын ең аз жұмыс көлемі болып табылады. Әрбір оқиға параметрлерін және қандай да бір ақпараттың мәнін сипаттайтын, оқиғалардың (event classes) қандай да бір санатына жатады. SQL Server Profiler оқиғасы мен оқиғалар класы арасындағы айырмашылықты жақсы түсіну үшін Performance Monitor объекттерімен және объект даналарымен баламаны жүргізейік. SQL Server Profiler оқиғалар класы Performance Monitor объектісі сияқты абстрактілі сипаттама болып табылады, ал оқиғаның өзі (объекттің данасы) қандай да бір объекттің жұмысы туралы ақпарат болып табылады.

SQL Server оқиғалар класының саны үлкен. Олармен жұмыс істеуді жеңілдету үшін олар 12 санатқа (category) бөлінген.

Query Analyzer. Осы құрал-сайман сұратуларды орындауға және олардың орындалуын талдауға арналған. Query Analyzer пайдаланылу жиілігі мен маңыздығы бойынша Enterprise Manager-мен салыстырылады.

Ерекшеліктерінің бірі сақталатын рәсімдердің орындалу жолы тарту мүмкіндігі болып табылады. Пайдаланушылар жол тарту кезінде (break points) тоқтау нүктелерін пайдалана алады, сонымен қатар рәсім командаларының қадаммен орындалуын жүзеге асыра алады.

Query Analyzer көмегімен сұратуларды және сақталатын рәсімдерді орындаудан басқа сұратудың орындалу өнімділігін бағалауға болады. Ол үшін сұратудың орындалуының бағалау (estimated) немесе нәтиже беретін жоспарын (execution plan) көрсетуге рұқсат еткен жөн, оны *Query мәзірі арқылы орындауға, онда Display Estimated Execution Plan* және *Display Execution Plan тармақтарын таңдау арқылы орындауға болады.*

Сұранысты орындаудың бағалау жоспары сервердің сұраныстың жекелеген қадамдарын орындауға шығындар туралы болжамдары негізінде қалыптастырылады.

Сұранысты орындаудың нәтиже беретін жоспары сұранысты орындағаннан кейін генерацияланады және істің шынайы жағдайын көрсетеді. Мінсіз жағдайда бағалау және нәтиже беретін орындау жоспарларының мәні бірдей болады. Алайда көп пайдаланушылық жүйелермен жұмыс істеу кезінде сұранысты шынайы орындау күтілгеннен артық уақытты алуы мүмкін.

Жиі процессордың басқа пайдаланушылардың сұраныстарын орындаумен немесе басқа транзакциялармен сұранысты орындау үшін қажетті ресурстарының бұғатталуымен айналысуынан болады.

Upgrade Wizard. Upgrade Wizard шебері SQL Server 6.5 бастап SQL Server 2000 дейін дерекқорларды жаңартуға арналған.

Жаңарту барысында SQL Server 2000-ге деректер, сонымен қатар сақталатын рәсімдер, триггерлер, ережелер, бастапқы дәлдеулер, біртұтастығын шектеу, көріністерді қоса алғанда жаңартылатын дерекқор объекттерінің барлық жинағы көшіріледі. Бұдан басқа, дерекқор объекттеріне қолжетімдік құқықтары орнатылған дерекқорларды пайдаланушылар және т.б. көшірілген болып табылады. Бұдан басқа, жаңарту барысында репликация қосалқы жүйесінің барлық дәлдеулері көшіріледі.

Import and Export Data. Осы құрал-сайман екі дереккөз арасындағы ақпараттың көшірмесін жасайтын, DTS пакетін құруға арналған деректерді импорттау (экспорттау) шебері болып табылады. Шебердің айрықша белгісі деректердің көшірмесін жасау үрдісін конфигурациялау қарапайымдылығы болып табылады.

Шебердің пайдалану кемшіліктеріне екіден астам дереккөзді өңдеу , сонымен қатар күрделі түрленуді және бұдан бұрынғы қатынастарды анықтау мүмкінсіздігі жатады. Бұдан басқа, DTS мүмкіндіктерінің үлкен бөлігі, мысалы, электронды пошта бойынша хабарлама жіберу қолжетімсіз болады. Шеберді пайдаланудың күмәнсіз артықшылығы қарапайым міндеттердің жеңіл шешілуі болып табылады. Егер дерекқор кестелеріне MS Excel файлынан ақпаратты жүктеу қажет болса, онда шебердің мүмкіндіктері жеткілікті болады. Сөйтіп, тәжірибесіз пайдаланушылардың өзі ақпарат алмасудың негізгі операцияларын орындай алады.

Client Network Utility және Server Network Utility. SQL Server 2000 желімен жұмыс істеуі үшін желілік хаттаманың болуы жеткіліксіз. Клиенттер серверде және клиентте сервермен қосылысты орната алу үшін, арнайы желілік кітапханаларды (Network Library) қосу керек. Осы кітапханалар динамикалық қосылатын кітапханалар түрінде іске асырылған (DLL, dynamic link library) және операциялық жүйеге қосылады.

Кітапхана хаттаманың базалық мүмкіндіктерін кеңейтеді және IPC механизмдері пайдаланылатын, клиент пен сервер арасында деректермен алмасу бойынша әртүрлі желілік операцияларды орындайтын дәлдеу болып табылады.

Кітапханаларды SQL Server 2000 орнату кезінде және кейінірек орнатуға болады. Егер орнатқаннан кейін кітапхананы қосу немесе жою қажет болса, онда ол үшін SQL Server 2000 бірге орнатылатын Server Network Utility утилитасын пайдалану керек.

Осы кітапхананың көмегімен сервердің желілік параметрлері конфигурацияланады, яғни желілік кітапханалар көрсетіледі, олардың көмегімен пайдаланушылар серверге жүгіне алады. Алайда клиенттің тарапынан желілік кітапханалар болу керек және сервермен жұмыс істеу үшін конфигурациялану керек. Клиентті конфигурациялау SQL Server әкімшілеу құрал-саймандарын орнату кезінде қосылатын, Client Network Utility утилитасы арқылы орындалады.

Конфигурацияланған параметрлер Enterprise Manager, Query Analyzer және басқа әкімшілеу құралдарының жұмысы үшін пайдаланылады. Клиент пен сервердің өзара әрекеттесуі сәтті болатынына кепілдік беру үшін, клиент серверде қолдау көрсетілетін ең болмағанда бір кітапхананы пайдалануын, сонымен қатар қажет болған жағдайда тиісті түрде оның қасиеттерін көрсетуін қамтамасыз ету керек.

Командалық жолдың утилиттары. Графикалық интерфейсі бар әлдеқайда қарастырылған утилиттардан басқа SQL Server 2000-да әртүрлі міндеттерді орындауға болатын командалық жол утилиттарының жинағы бар. Утилиттардың кейбіреулері серверде автоматты түрде пайдаланылады және утилитаға қарағанда, SQL Server 2000 өзегінің бір бөлігі болып табылады.

Шеберлер. Ертерек айтылғандай, көптеген міндеттер Enterprise Manager графикалық интерфейсін және арнайы шеберлер арқылы (wizards) Transact-SQL құралдарының көмегімен орындалуы мүмкін. Шеберлер әкімшілік міндеттерді орындаудың аса қарапайым тәсілі болып табылады. Шеберлердің кемшіліктері барынша шектелген мүмкіндіктер болып табылады.

Алайда кейбір шеберлерге қолданылмайды. Репликация қосалқы жүйесін конфигурациялау шеберлері барынша күрделі үрдіс болып табылады. Мысалы, Enterprise Manager құралдарымен жариялым жасау мүмкін емес. Тиісті шеберді пайдалану керек. Әрине, әрқашан Transact-SQL құралдарын пайдалануға болады. Бірақ кейде күрделігі мен еңбекті көп қажет ететіндігі соншалықты, ең жақсы шешім шеберді пайдалану болып табылады.

Бақылау сұрақтары

1. SQL Serve келесі қызметтерінің негізгі тағайындалуын атаңызг:
   - MSSQLServer;
   - SOLServerAgent;
   - Microsoft Search (MSSearch);
   - Microsoft Distributed Transaction Coordinator (MSDTC).
2. SQL Server жүйелік дерекқорлардың негізгі тағайындалуын атаңыз:
   - Master;
   - Model;
   - Tempdb;
   - Msdb.
3. SQL Server 2000 қандай құрал-саймандарын білесіз?

ORACLE БӨЛІНГЕН ДЕРЕККОРЛАРДЫ БАСҚАРУ ЖҮЙЕСІ

12.1. Қысқаша құру тарихы

1977 ж. Л.Эллисон, Б.Майнер және Э.Оуэтс Relational Software Incorporated (RSI) фирмасын ұйымдастырды, ол (БДҚБЖ) Oracle *бөлінген дерекқорларды басқару жүйесінің* бастамасы болды. Б. Майнер және Д. Оуэтс С және SQL тілдерін пайдалана отырып, бөлінген дерекқорларды басқару жүйесін жасап шығаруды шешті. 1979 ж. бағдарламалық жүйелер нарығына БДҚБЖ Oracle 2 ұсынылды, ол ОС RSX-11 операциялық жүйесінің басшылығында жұмыс істеді.

1983 ж. Oracle 3 жүйесінің жаңа нұсқасы пайда болды. Жасап шығарушылар SQL тіліне өзгерістер енгізді, бұл жүйенің өнімділігін арттыру мүмкіндігін берді. Oracle 2 қарағанда үшінші нұсқа толығымен С тілінде жазылған. Осы сәттен бастап RSI фирмасы өзінің атауын Oracle Corporation деп өзгертті.

Фирма автоматтандырылған жүйелерді “интеллектуалдануын” арттыру және деректерді аналитикалық өңдеу мүмкіндіктерін кеңейту бағытында өзінің БДҚБЖ әрқашан жетілдіреді.

12.2. Негізгі ұғымдар мен терминология

Баяндалғаннан ерекшеленетін, Oracle жүйесінің терминдері мен ұғымдарын қарастырамыз.

Блок (Block) — БДҚБЖ Oracle-да деректерді сақтаудың ең кішкентай бірлігі. Тақырыпша ақпаратын және блоктың өзін құрайды (деректер немесе PL/SQL-код). Блок көлемі 2 бастап 16 дейін Кбайт құрайды.

Буфер — деректерді сақтау үшін пайдаланылатын, жедел жадтың көлемі. Буфер пайдаланылуы болжалатын немесе жақын арада пайдаланылған деректерді құрайды. Көп жағдайда буфер қатты дискте сақталатын деректердің көшірмесін құрайды. Буфердегі деректер өзгертілуі және дискке жазылуы мүмкін немесе уақытша сақтау үшін буферге орналастырылуы мүмкін. Oracle-ға қолданбалы буферлер жақын арада жүгінген деректер блогын құрайды. Буферлердің жиынтығы *деректер буферлерінің кэшін* құрайды. Сондай-ақ буферде одан әрі дискке жазылатын өзгерістер журналының уақытша жазбалары сақталады (өзгерістер журналының буфері).

Лас буфер (*dirty buffer)* — құрамы өзгерген буфер. DBWR мерзімді лас буферлерді қатты дискке лақтырады.

Сипаттамалардың динамикалық кестелері (Dynamic Performance Tables) — бұл Oracle данасын іске қосу кезінде автоматты түрде құрылатын және осы дананың сипаттамаларын сақтау үшін пайдаланылатын кестелер. Олар қосылыстар, енгізу (шығару), ортаның параметрлерінің бастапқы мәндерін және т.б. туралы ақпаратты құрайды.

Сұраныс — бұл "тек оқуға арналған" транзакция. Сұраныс *Select* командасы арқылы генерацияланады. Қарапайым транзакция мен сұрату арасындағы айырмашылық сұраныс кезінде деректер өзгермейтіндігінде.

Индекс — деректерді жылдам әрі тиімді алу мүмкіндігін беретін құрылым (қандай да бір кітаптың мазмұны қажетті бөлімді тауып беретін сияқты). Индекс бір немесе бірнеше бағанға жарияланады. Кестеге қолжетімдік индекстелген баған (бағандар) бойынша жасалады.

Кластер — физикалық біреу болып сақталатын және ортақ бағандары бар кестелер жинағы. Егер ортақ бағандары бар, екі және одан артық кестелерге сұратулар жиі өңделсе, кластерлерді пайдалану тиімді. Кестелерге кластерлі кестенің бір бөлігі болған жағдайда да жекелеп жүгінуге болады.

Бәсекелесу (concurrency) — бұл бағдарламаның бір уақытта бірнеше қызмет атқару қабілеті. Oracle-ға қолданбалы *бәсекелесу* — бұл көп пайдаланушы үшін деректерге бір уақытта қолжетімдігінің болуы.

Бақылау нүктесі (Checkpoint) — барлық өзгертілген деректердің (жадтағы деректер блогы) дискке жазылуына әкелетін операция. Бұл істен шығудан кейін дерекқорды жылда қалпына келтіру мәселесіндегі өзекті фактор.

Деректер буферінің кэші — деректерге жылдам қолжетімдікке арналған жад саласы. Аппараттық қамтамасыз ету тұрғысынан — бұл негізгі жадтан жылдамырақ болатын шағын жад көлемі (жедел жадқа қолданбалы). Жадтың осы көлемі деректерді немесе нұсқаулықтарды орталық процессорға (ОП) жүктеу үшін қажетті уақытты азайту үшін ОП өз бетімен кіріктірілген кэшті құрайды.

Сұлба объекттері — бұл дерекқорды құрайтын абстракция (логикалық құрылым). Сұлба объекттері индекстерден, кластерлерден, пакеттерден, реттіліктерден, сақталатын рәсімдерден, синонимдерден, кестелерден, көрсетілімдерден және т.б. тұрады.

Реттілік, реттіліктер генераторы - деректер буферінің кэшінде сақталатын, цифрлардың реттілігін құру үшін пайдаланылады.

Көрсетілім (түрі) — бұл бір немесе одан артық кестеден деректерді қарауға арналған жиектеме, терезе. Түрі ешқандай деректерді сақтамайды, тек оларды көрсетеді. Түрлерімен ешбір шектеусіз кестелермен сияқты операцияларды (сұратуларды құру, жаңарту, жою) жасауға болады. Кестеден қажетті деректердің бір бөлігін немесе бірнеше кестеден тұратын деректер жинағын алу арқылы дерекқорда сақталатын деректердің пайдаланушымен қабылдануын жеңілдету үшін жиі көрсетілімдер пайдаланылады. Бұдан басқа, көрсетілімдер пайдаланушылардың кейбір деректерге қолжетімдігін шектеу үшін пайдаланылуы мүмкін.

Бағдарламалық блок — БДҚБЖ Oracle қатысты осы пакетті, сақталатын рәсімдерді немесе рәсімдердің реттілігін сипаттау үшін пайдаланылатын бағдарлама.

Рәсім — белгілі бір міндетті орындайтын SQL немесе PL/SQL-командаларының жинағы. Рәсімнің кіреберіс параметрлері болуы мүмкін, бірақ шығаберіс параметрлері жоқ.

Деректер сөздігі (Data Dictionary) — ДҚ туралы ақпаратты сақтау үшін пайдаланылатын кестелер жинағы.

Сұлба (Schema) — ДҚ объектілерінің топтамасы.

Кесте — Oracle ДҚ деректерін сақтаудың негізгі бірлігі. Кесте атаудан, жолдар мен бағандардан тұрады. Әрбір бағанның аты және деректер түрі бар. Кестелер кесте кеңістіктерінде сақталады, бұл ретте жиі бір кесте кеңістігінде бірнеше кесте орналасады.

Транзакция — әрекеттер реттілігінің логикалық аяқталған фрагменті (тіркеумен немесе кейін шегінумен аяқталған, бір немесе одан артық SQL-командасы).

Триггер — Insert, Update немесе *Delete командаларын орындау кезінде автоматты түрде іске қосылатын рәсімдерді жасау мүмкіндігін беретін механизм.*

Узкое место (Bottleneck) — жүйенің өнімділігін немесе тиімділігін шектейтін компоненттер.

Қызмет — бұл белгілі бір міндетті іске асыратын SQL немесе PL/SQL-командаларының жиынтығы. Қызметтің рәсімнен ерекшелігі ауыспалының қандай да бір мәнін қайтаратындығында (рәсім ештеңе қайтармайды). Қызметті құру желі бойынша берілетін нұсқаулықтар санын азайту мүмкіндігін береді.

Сақталатын рәсім — бұл деректер сөздігінде сақталатын SQL-сұрату. Сақталатын рәсімдер сұратуларды тиімді пайдалану үшін жасап шығарылады. Сақталатын рәсімдерді пайдалану кезінде БДҚБЖ желілік трафигін азайтуға болады және сөйтіп өнімділігін арттыруға болады.

Таза буфер (clean buffer) — бұл құрамы өзгертілмеген буфер. Осы буфер өзгертілмегендіктен, DBWR үрдісіне оны қатты дискке жазу қажеттігі жоқ.

DBWR (*DataBase WRiter)* — негізгі міндеті дерекқордың өзгерістерін физикалық қатты дискке жазу болып табылатын үрдіс.

DDL (Data Definition Language) — деректерді сипаттау тілі. Осы тілдің командалары дерекқор объекттерін құру, өзгерту және жоюға арналған. Oracle жүйесінде DDL командалары дерекқорды әкімшілеуге байланысты. Әрбір DDL-команданы орындауға дейін және кейін Oracle жүйесі міндетті түрде барлық ағымдағы транзакцияларды тіркейді (ақпараттың жоғалуына жол бермеу үшін).

DML (Data Manipulation Language) — деректерді манипуляциялау тілі. Осы тілдің командалары сұратуларды құру және схеманың бар объектілерінің деректерін пайдалану мүмкіндігін береді. DDL-ға қарағанда әрбір командадан кейін транзакция тіркелмейді. DML-дің мынадай командалары бар: *Delete, Insert, Select жәнеUpdate; Explain* plan-командасы; және *Lock* table-командасы.

SGA (System Global Area) — деректерді және Oracle данасының басқарушы ақпаратын сақтау үшін пайдаланылатын жадтың бөлінетін саласы. SGA Oracle данасын іске қосу кезінде жадта орналасады және жұмыс аяқталғаннан кейін босатылады. SGA деректер буфері, өзгерістер журналының буфері және бөлінетін пул (shared pool) құрайды.

12.3. Oracle конфигурациялары

Конфигурацияның көп түрі бар. Олардың кейбіреулерін қарастырайық.

Веб-сервер — статикалық және динамикалық вед-беттермен жұмыс істеуге арналған. Бұл беттер өте қарапайым және дерекқордан генерацияланатын кешенді болуы мүмкін. Oracle веб-сервері, әдетте, коммерциялық веб-қосымшалар үшін пайдаланылады. Осындай қосымшалар сатып алушыларға тауардың суреттері және бейне иллюстрациялары бар каталогтарды қарау мүмкіндігін береді. Сатып алушы ұнаған тауарды сатып ала алады. Oracle веб-серверінің тән белгілері әдетте пайдаланушылардың айтарлықтай санына қолдау көрсетеді және деректердің үлкен көлемін құрайды. Веб-сервердің өнімділігі жедел жад көлеміне тәуелді.

Видео-сервер — бейне ақпартты өңдеуге арналған. Бейне сервердің тән ерекшелігі бейне ағынның үлкен көлеміне қолдау көрсету үшін өткізу жолы кең болу керек. Бейне сервер енгізудің (шығарудың) үлкен жүктемесімен жұмыс істей алу керек, себебі құрылғылардан оқу кезінде бірден үлкен деректер блогы жүктеледі.

Ақпараттық дүңгіршек (Data Mart) — бұл деректер қоймасының кішірейтілген нұсқасы (Data Warehouse), әдетте, тар шеңберде мамандандырылған міндеттерді шешуге бағытталған. Ол жүз гигабайттан кем жадты талап ететін ақпаратты сақтайды және өңдейді.

DSS тән белгілері — бұл әртүрлі кестеде және әртүрлі дерекқорда сақталатын, үлкен ақпарат көлемін өңдеуге байланысты көптеген сұратуларды орындау.

Деректер қоймасы (Data Warehouse) — бұл OLTP және DSS-жүйелерінің жұмыс нәтижелерін сақтайтын ірі ауқымды жүйе. Бұл жүйе жадтың жүздеген гигабайтын алатын ақпараттың сақталуын және өңделуін қамтамасыз ету керек.

DSS (Decision Support System) — бұл деректерді зияткерлік талдау барысында пайдаланылатын шешімдердің қабылдануын қолдау жүйелері.

OLAP (On-line Analytical Processing) — шынайы уақыт ауқымындағы ақпаратты аналитикалық өңдеу жүйесі. Әдетте, OLAP жүйелерін пайдаланушылар — бұл деректермен ғаламдық деңгейде жұмыс істейтін қаржылық аналитикалар немесе маркетингтік қызметкерлер құрамы.

OLTP (On-line Transaction Processing) — бұл транзакцияларды жедел өңдеу жүйесі.

OLTP-жүйелерінің тән белгілері көп пайдаланушылық дерекқормен жұмыс істейтін пайдаланушылардың көп бөлігінің жұмысын қамтамасыз етуінде. Пайдаланушылар өз сұратуларына деректердің қайтарылуын күтетіндіктен, жүйе барлық клиенттің сұратуларына жылдам жауап берілуін қамтамасыз ету керек. OLTP-жүйелер осыған арналған. OLTP-жүйелерінің жұмысы ақпаратты өңдеудің тиісті үрдістерінің жоғары жылдамдығын қамтамасыз ету керек.

12.4. Пайдаланушылар түрі

Пайдаланушы түрлері және олардың міндеттері Oracle конфигурациясына және корпоративтік дерекқордың нақты ұйымына тәуелді ерекшеленуі мүмкін. Ірі жүйелерде, мысалы, дерекқор әкімшісінің міндеттері бірнеше маман арасында бөлінуі мүмкін, ал шағын жүйелерде бір адам бір уақытта бірнеше пайдаланушы түрінің қызметтерін орындауы мүмкін.

Дерекқорды басқарудың барлық жүйелеріне тән, пайдаланушылардың мынадай негізгі түрлерін ерекшелеуге болады:

- дерекқор әкімшілері;
- деректерді қорғау бойынша әкімшілер;
- қосымшаны жасап шығарушылар;
- қосымшаның әкімшілері;
- дерекқорды пайдаланушылар;
- желі әкімшілері.

DataBase Administrator (DBA) — бұл дерекқордың жұмысын басқаратын маман. Әдетте DBA міндеттері екі санатқа бөлінеді: негізгі және қосымша.

DBA *негізгі міндеттері* мына міндеттерден тұрады.

• Жаңа бағдарламалық жасақтаманы орнату. DBA осы міндеті Oracle жаңа нұсқаларын, қосымшаларды және ДҚБЖ әкімшілеуге жататын өзге бағдарламалық жасақтаманы орнатуда. Бұл міндет сондай-ақ орнатылатын бағдарламаларды жұмыс ортасына енгізуден бұрын міндетті түрде тестіленуін көздейді.

• Бағдарламалық және аппараттық қамтамасыз етудің конфигурациясы. Көп жағдайда бағдарламалық және аппараттық қамтамасыз етуді дәлдеуге қолжетімдікке тек жүйелік әкімші ие, сондықтан DBA бағдарламаларды орнату керек, бағдарламалық және аппараттық қамтамасыз етуді тек жүйелік әкімшімен бірге конфигурациялау керек.

• Қауіпсіздікті қамтамасыз ету. Бұл DBA негізгі міндеттерінің бірі. Қауіпсіздікті басқару және әкімшілеу пайдаланушыларды қосу және жою, квоталарды басқару, аудит және қауіпсіздік мәселелерін шешуді құрайды.

• Өнімділігін дәлдеу. DBA тұрақты түрде жүйенің өнімділігін тексеру керек, ал қажет болғанда оны қайта дәлдеу керек. Жақсы дәлденген жүйенің өзі тұрақты тексеріліп отыруды және мерзімді қайта дәлдеуді талап етеді. Кейде жүйенің параметрлерін өзгерту жеткілікті, кейде индекстерді өзгерту, кестенің құрылымын қайта құру жеткілікті.

• Жүйенің резервтік көшірмесін жасау және қалпына келтіру. Мүмкін DBA басты міндеттерінің бірі — жүйеде деректерді тұрақты сақтау болуы мүмкін. Тиімді жасау үшін резервтік көшірме жасау рәсімін және деректерді қалпына келтіру стратегиясын жасап шығару керек. Деректердің резервтік көшірмесін жасау және қалпына келтірудің өңделген сұлбасын мерзімді тестілеу маңызды.

• Тұрақты (жоспарлы) қызмет көрсету рәсімі. Пайдаланушылардың жұмысын бұзбау үшін ДҚБЖ-ге ерте немесе демалыс күндері қызмет көрсету керек. Қызмет көрсетуге мыналар кіреді: жүйені мұрағаттау, тестілеу және дәлдеу. Жүйеге жоспарлы қызмет көрсету үшін әкімші ДҚБЖ қызмет көрсету күнтізбесін жасап, клиенттерге хабардар ету керек.

• Ақаулықтарды оқшауландыру және жүйені істен шыққаннан кейін қалпына келтіру. DBA міндеттеріне ДҚБЖ істен шыққан жағдайда жұмысқа қабілеттігін қалпына келтіру кіреді. Жүйенің істен шығуы пайдаланушылардың өз деректеріне қолжетімдігін жоғалтуына әкелетіндіктен, DBA жүйенің жұмысын барынша жылдам қалпына келтіру керек.

Жақсы дайындалған DBA жүйені істен шыққаннан кейінгі қалпына келтіру жоспарын қарастырып, құрауы керек.

DBA қосымша міндеттері жекелеген клиенттерге көмек көрсету болып табылады және мынадай әкімшілеу міндеттерін құрауы мүмкін.

• Деректерді талдау. DBA деректерді талдау керек және жекелеген жасап шығарушыларға немесе пайдаланушыларға деректерді сақтау тиімділігін немесе өнімділігін жақсарту бойынша ұсыныстар беру керек.

• ДҚ жасап шығару (алдын ала). DBA ДҚ құрылымын жасап шығарудың бастапқы дәрежесіне қатыса алады. DBA жүйені ішінен білетіндіктен, ол жасап шығарушылар командасына әлеуетті мәселелерге көрсете алады және бағдарламалардың өнімділігін арттыруға көмектесе алады.

• Жасап шығарушыларға сақталатын SQL- рәсімдер бойынша консультация жүргізу. DBA жасап шығарушылар мен пайдаланушылар үшін консультант болуға дайын болу керек. DBA SQL-код мәселелерін шешуге және сақталатын рәсімдерді жасап шығаруға (жазуға) жиі қатыстырылады.

• Өндірістік стандарттар мен келісімдерді аттары бойынша жасап шығару. Бұл міндет қосымшалар разрядына жатқызылғанымен, ол басқарудың негізгі ұйымдастырушылық мәселесінің бірін атқарады. Қосымшаларды жасап шығаруда және ашуда бірнеше әртүрлі топ қатыса алатынын есепке ала отырып, DBA белсене қатысу керек, қосымшалар осы стандарттарға сәйкес келу үшін, өндірістік стандарттар мен келісімдердің аты бойынша жасап шығарылуында басты рөлді атқару керек.

• Ортаны құжаттау. DBA жабдықтың конфигурациясын, бағдарламалық қамтамасыз ету мен ДҚБЖ жаңарту және өзгертуді қоса алғанда, ДҚБЖ ортасының әрбір аспектін, сонымен қатар жүйе мен оның параметрлерінің өзгеруіне байланысты барлық мәселелерді құжаттау керек. DBA қажет болған жағдайда құжаттама бойынша жүйені толық қалпына келтіре алу керек.

• Жүйенің жүктемесін және қажетті жад көлемін жоспарлау. Пайдаланушылардың артқан тұтынушылығын қанағаттандыру үшін қосымша серверлерді, қосымша диск және жедел жадты алу қажеттігі DBA жұмысының ажырамас бөлігі болып табылады. Аппараттық құралдардың күтілетін қажеттігін болжай отырып, әкімші кәсіпорынның ақпараттық жүйесінің жұмыс сенімділігін қамтамасыз етеді.

12.6. Деректерді сақтаудың физикалық сәулеті

БДҚБЖ Oracle сақталатын ақпараттың үлкен көлемдеріне бір уақытта қолжеткізуге арналған (терабайттар). БДҚБЖ екі құрамдас бөліктен тұрады: дерекқор (ақпарат) және дана (жүйені нақты іске асыру).

Дерекқор

Дерекқор жүйеде сақталатын физикалық файлдардан және логикалық бөліктерден (мысалы, ДҚ сұлбасы) тұрады. Физикалық файлдар - бұл дискте сақталатын файлдар, ал логикалық файлдар физикалық деңгейдің компоненттері болып табылады.

Сонымен, Oracle дерекқорлары екі деңгейден тұрады: физикалық және логикалық.

Физикалық деңгей. Файлдың үш санатын құрайды: деректер файлдары, операция журналдарының файлдары (redo log files), басқарушы файлдар.

Деректер файлдары. Осы файлдарда ДҚ-да бар ақпарат сақталады. Бұл бір файл және жүздеген файл болуы мүмкін. Бір кестедегі ақпарат бірнеше дерек файлдарына бөлінуі мүмкін (ал бірнеше кесте дерек файлдарының кеңістігін өзара бөлуі мүмкін). Кестелерді бірнеше дерек файлдары бойынша бөлу жүйенің өнімділігін айтарлықтай арттыра алады. Дерек файлдарының саны Maxdatafiles параметрімен шектелген.

Операция журналдарының файлдары. Жүйе істен шыққан жағдайда қалпына келтіру үрдісі үшін қажетті ақпаратты құрайтын және дерекқорда болған барлық өзгерістерді сақтайтын файлдар. Операциялар журналының көмегімен жасалған, бірақ жүйенің істен шығуына дейін тіркелмеген өзгерістер қалпына келтіріледі. Операция журналдарының файлдары аппараттық істен шығудан жақсы қорғалу керек (бағдарламалық және аппараттық деңгейде). Егер операция журналының ақпараты жоғалмаса, онда жүйені қалпына келтіру мүмкін емес.

Басқарушы файлдар. Басқарушы файлдар дерек файлдарының және операция журналдарының файлдарының орналасуын қоса алғанда, Oracle данасын іске қосу үшін қажетті ақпаратты құрайды. Басқарушы файлдар жақсы қорғалу керек.

Логикалық деңгей. Логикалық деңгей мына элементтерді құрайды:

- кесте кеңістіктері;
- ДҚ сұлбасы.

Кестелі кеңістіктер. Дерекқор кестелі кеңістіктер деп аталатын, бір және одан артық логикалық бөліктен тұруы мүмкін. Кестелі кеңістіктер деректерді өзара логикалық топтастыру үшін пайдаланылады. Мысалы, бухгалтерлік деректер үшін бір кесте кеңістігін айқындауға, ал басқасын - қоймалық деректер үшін айқындауға болады. Топтарды кесте кеңістіктері бойынша сегменттеу осы топтардың әкімшілендірілуін жеңілдетеді. Әрбір кесте кеңістігі бір немесе одан артық деректер файлынан тұрады. Бір кестелі кеңістік үшін бірнеше дерек файлдарын пайдалана отырып

оларды әртүрлі дискке бөлуге болады, сөйтіп енгізу (шығару) жылдамдығын және сәйкесінше жүйенің өнімділігін арттыруға болады.

ДҚ сұлбасы. БДҚБЖ Oracle-да диск кеңістігін бақылау арнайы логикалық құрылымдарды - дерекқор сұлбаларын пайдалана отырып жүзеге асырылады. Осы құрылымдар деректер блогынан, экстенттерден, сегменттерден тұрады.

Деректер блогы — бұл Oracle ДҚ-да деректерді сақтаудың ең кішкентай бірлігі. Деректер блогы өзі туралы ақпаратты және деректерді құрайды. Олар физикалық дискте сақталады. Деректер блогы көбінесе 2Кб (2048 байт) алады, жүйенің жұмыс істеу тиімділігін арттыру үшін бұл мәнді өзгертуге болады.

Экстент деректер блогынан тұрады. Экстенттер сегменттердің құрылыс блоктары болып табылады және сол уақытта деректер блогынан тұрады. Экстенттер пайдаланылмайтын (бос) қойма кеңістігін азайту үшін пайдаланылады. Кесте кеңістіктеріндегі деректер санын арттыру шамасына қарай экстенттер өсуі мүмкін деректерді сақтау үшін пайдаланылады. Сөйтіп, бірнеше кесте кеңістігі осы кестелі кеңістіктердің бөлімдерін алдын ала анықтамай өзара қойма кеңістігін бөле алады.

Кестелі кеңістікті құру кезінде экстенттердің анықтамаларының аз санын, сонымен қатар әлдеқайда анықталған экстенттерді толтыру кезінде қосылатын экстенттер санын көрсетуге болады. Осындай бөліну ДҚ қоймасының бүкіл кеңістігін бақылау мүмкіндігін береді.

Сегмент, өз кезегінде белгілі бір деректер түрін құрайтын экстенттер жиынтығынан тұрады. Oracle ДҚ сегменттің төрт түрін пайдаланады:

- Деректер сегменті — пайдаланушы деректерін сақтайды;
- Индексті сегмент — индекстерді құрайды;
- Кейін шегіну сегменті — ДҚ бұдан бұрынғы жағдайына оралу кезінде пайдаланылатын кейін шегіну ақпаратын құрайды;
- Уақытша (аралық) сегмент — SQL-тіркесті орындау үшін қосымша жұмыс кеңістігі қажет болғанда құрылады. Бұл сегменттер SQL-командалары орындалғаннан кейін бірден жойылады. Аралық сегменттер ДҚ-мен әртүрлі операцияларда, мысалы сұрыптауда пайдаланылады.

Дана

Дана бөлінетін жад пен үрдістерден тұратын деректерге қолжетімдіктің нақты тәсілі болып табылады.

Бөлінетін жад. Oracle бөлінетін жадты (shared memory) деректер мен индекстерді кэштеу, бағдарламалық кодты сақтау сияқты әртүрлі мақсаттарда пайдаланады.

Бөлінетін жад бірнеше бөлікке (немесе жад құрылымына) бөлінеді. Oracle жадының негізгі құрылымдары:

- жүйелік ғаламдық сала;
- бағдарламалық ғаламдық сала.

Жүйелік ғаламдық сала (SGA — System Global Area). Бұл бөлінетін жад саласы, оны Oracle деректерді және Oracle нақты бір данасының ақпаратын басқару үшін пайдаланады. SGA Oracle данасын іске қосу кезінде жадта орналасады және тоқтау кезінде жадты босатады. Oracle-дің әрбір іске қосылған данасы меншік SGA ие.

SGA-дағы ақпарат мына компоненттерден тұрады (олардың әрбіреуі дананы іске қосу кезінде жадта құрылады): ДҚ буферлерінің кэші, өзгерістер журналының буфері, бөлінетін пул.

ДҚ буферлерінің кэші соңғы ашық деректер блогын сақтайды. Бұл блоктар өзгерген, бірақ дискке әлі жазылмаған деректерді құрауы мүмкін (лас блоктар); өзгертілмеген немесе өзгертілгеннен кейін дискке жазылған деректерді құрауы мүмкін (таза блоктар). ДҚ буферлерінің кэші соңғы пайдаланылатын блоктардың алгоритмі негізіндегі деректер блогын сақтайтындықтан, аса белсенді пайдаланылатын блоктар әрқашан жадта қалады (дисктің енгізілуін (шығарылуын) азайтады және жүйенің өнімділігін арттырады).

Өзгерістер журналының буфері ДҚ өзгерістері туралы деректерді сақтайды. Өзгерістер журналының буфері мүмкіндігінше өзгерістер журналының файлына жылдам әрі тиімді жазылады. (Өзгерістер журналы жүйе істен шыққан жағдайда ДҚБЖ Oracle данасын қалпына келтіру үшін пайдаланылады.)

Бөлінетін пул — кітапхана кэшіндегі бөлінетін SQL-салалар және деректер сөздігінің ішкі ақпараты сияқты, бөлінетін жад құрылымдары сақталатын SGA саласы. Бөлінетін пул кітапхана кэшінен және деректер сөздігінің кэшінен тұрады.

Кітапханалық кэш бөлінетін SQL-тіркестерді сақтау үшін пайдаланылады. Мұнда әрбір бірегей SQL-тіркес үшін кэштелетін жолдарды талдау ағашы және орындау жоспары құрылады (яғни кітапханалық кэште сақталады). Егер бірнеше қосымша бірдей SQL-тіркесті жіберсе, онда жұмысты жылдамдату үшін бөлінетін SQL-сала пайдаланылады (әлдеқайда талданған жолдар мен дайын орындау жоспары пайдаланылатындықтан, уақыт үнемделеді).

Деректер сөздігінің кэші Oracle ДҚ-на сілтеме ретінде пайдаланылатын кестелер мен көрсетілімдер жинағын құрайды. Мұнда ДҚ-дың логикалық және физикалық құрылымы туралы ақпарат сақталады. Деректердің сөздігі мына ақпаратты құрайды:

• Пайдаланушылық ақпарат (мысалы, пайдаланушылық артықшылықтар);

• ДҚ кестелері үшін айқындалған біртұтастығын шектеу;

• ДҚ барлық бағандары мен кестелерінің аттары және деректер түрі;

• Деректер сұлбасының объекттері пайдаланатын және анықтаған жад көлемі туралы ақпарат.

Жоғары өнімділігін қамтамасыз ету үшін деректер сөздігінің кэшіне жеткілікті жад көлемін орнату керек.

Бағдарламалық ғаламдық сала (PGA — Program Global Area). Бұл деректер мен Oracle серверлік үрдістері туралы басқарушы ақпарат сақталатын жад саласы. PGA көлемі мен мазмұны Oracle орнату кезінде белгіленетін опциялармен айқындалады. Бұл сала мына компоненттерден тұрады:

• стек кеңістігі — бұл сеанстардың ауыспалыларын, сеанстардың массивтерін және т.б. сақтайтын жад;

• сеанстың ақпараты — егер Oracle мультижіпті режимде жұмыс істемесе, онда сеанстың ақпараты PGA-да сақталады. Әйтпесе сеанстың ақпараты SGA-да сақталады;

• жекеше SQL-саласы— байланысқан ауыспалылар және шынайы уақыт буферлері сақталатын PGA бір бөлігі.

Үрдіс. Үрдіс (немесе жіп) — бұл пайдаланушы үшін білдірмей орындалуы мүмкін, бағдарламалық кодты орындау механизмі. Бұдан басқа, бірнеше үрдіс бір уақытта жұмыс істеуі мүмкін. Әртүрлі операциялық жүйеде және әртүрлі платформада олар әртүрлі аталуы мүмкін (үрдістер, жіптер, домендер және т.б.), бірақ мәні бірдей. БДҚБЖ Oracle үрдістің екі түрімен жұмыс істейді: пайдаланушы үрдістері және Oracle үрдістері, сондай-ақ фон немесе көлеңке деп аталады. Кейбір операциялық жүйелерде (Windows NT сияқты) үрдістер шынымен жіптер болып табылады, бірақ ұғымдарды шатастырмау үшін оларды үрдістер деп атаймыз.

Пайдаланушы үрдістер. Пайдаланушы (клиенттік) үрдістер — бұл БДҚБЖ-мен пайдаланушылық қосылыстар. Пайдаланушы үрдісі енгізуді басқарады және Oracle бағдарламалық интерфейсі арқылы Oracle серверлік үрдістермен өзара әрекеттеседі. Пайдаланушылық үрдіс пайдаланушыға ақпарат беру үшін пайдаланылады және қажет болған жағдайда аса қолайлы нысанда ұсынады.

Oracle үрдістері. Oracle үрдістері пайдаланушы үрдістеріне арналған қызметтерді орындайды. Бұл үрдістер екі топқа бөлінуі мүмкін: серверлік үрдістер (белсенді үрдістер үшін қызметтерді орындайды) және фон үрдістері (жалпы БДҚБЖ қызметтерін орындайды).

Серверлік үрдістер (көлеңкелі) пайдаланушы сұратуларын орындай отырып, пайдаланушы және Oracle үрдістері арасында өзара әрекеттеседі. Мысалы, егер пайдаланушылық үрдіс SGA-да жоқ деректер бөлігін сұратса, онда көлеңке үрдісі ДҚ-дан SGA-ға деректер блогының оқылуы мүмкін жауапты болады.

Пайдаланушылық және көлеңке үрдісі арасында “бірге бір” байланысы туындайды, алайда бір көлеңке үрдісі жүйе ресурстарын үнемдей отырып, бір уақытта бірнеше пайдаланушылық үрдіспен өзара әрекеттесе алады (мультижіпті сервердің конфигурациясы).

Фон үрдістері БДҚБЖ Oracle әртүрлі міндеттерін орындау үшін пайдаланылады. Бұл міндеттер Oracle данасымен өзара әрекеттесуден лас блоктарды дискке жазуға дейін түрленеді. Oracle тоғыз фон үрдісі бар:

• DBWR (DataBase WRiter) лас блоктардың ДҚ блок буферлерінен дискке жазылуы үшін жауапты. Транзакция деректер блогындағы ақпаратты өзгертетін кезде, осы деректер блогы дискке баяу жазылуы міндетті емес. Сәйкесінше, DBWR деректерді дискке барлық өзгерістерді жекелеп жазуға қарағанда аса тиімді тәсілмен жаза алады. DBWR әдетте оқу үшін қажетті болатын кезде жазады. Жақын арада пайдаланылған деректер де жазылады. Асинхронда кіреберісі (шығаберісі) бар жүйелер үшін DBWR бір үрдіс жеткілікті. Қалған жүйелер үшін DBWR бірнеше үрдісін құра отырып, өнімділігін арттыруға болады;

• LGWR (LoG WRiter) журнал буферіндегі деректерді өзгерістер журналына жазады;

• CKPT (ChecK PoinT) DBWR үрдістеріне бақылау нүктесін орындау және барлық дерек файлдары мен басқарушы файлдарды жаңарту қажеттігі туралы сигнал береді. Бақылау нүктесі - бұл ДҚ-ның барлық өзгертілген буферлері дискке жазылатын оқиға. CKPT — бұл міндетті үрдіс емес. Егер CKPT іске қосылмаса, онда оның жұмысын LGWR қабылдайды;

• PMON (Process MONitor) қалған үрдістерді сақтау үшін және алдын ала бүлінгендерді қайта іске қосу үшін пайдаланылады. Сонымен қатар PMON буферлердің пайдаланылмайтын салаларын тазартады және бос болмауы мүмкін ресурстарды босатады. Барлық қатып қалған үрдістер мен диспетчерлерді қайта іске қосу үшін жауапты;

SMON (System MONitor) іске қосу кезінде дананы қалпына келтіреді. Бұл уақытша сегменттерді тазалайды және аяқталмаған транзакцияларды қалпына келтіреді, сонымен қатар ДҚ дефрагменттейді;

RECO (RECOvery) бөлінген ДҚ-дағы аяқталмаған транзакцияларды тазартады. Даулы транзакцияларды тіркейді немесе кейін шегінеді;

ARCH (ARCHiver) толтыру кезінде өзгерістер журналының файлдарының көшірмесін жасайды. Егер БДҚБЖ *Archivelog режимінде жұмыс істесе,* ARCH активті. Егер жүйе осы режимде жұмыс істемесе, онда жүйені істен шыққаннан кейін қалпына келтіру мүмкін болмайтын жағдайлар болуы мүмкін. Кейбір жағдайда *Noarchivelog режимінде де жұмыс істеуге болады*;

LCKn (Parallel Server LoCK) — сервер параллельді режимде жұмыс істеу кезінде 10 үрдіске дейін (мұндағы n — 0-ден 9-ға дейінгі мән) пайдаланылуы мүмкін. Данааралық бұғаттау қызметін атқарады;

Dnnn (Dispatcher) — сервер мультижіпті режимде жұмыс істеген кезде өзара байланыстың әрбір хаттамасы үшін жауапты болатын, ең болмағанда бір диспетчерлік үрдіс болады. Диспетчерлік үрдістер пайдаланушылық және бөлінетін серверлік үрдістер арасындағы өзара әрекеттесуді ұйымдастырады.

12.7. Транзакциялар. Жұмыс істеу қағидалары

Транзакция — бұл (commiting) тіркеумен немесе (rollbacking) кейін шегінумен аяқталған, бір немесе одан артық SQL-команда. *Тіркеу* дегеніміз барлық өзгерістерді қабылдау және сақтау. *Кейін шегіну* - бұл соңғы өзгерістердің күшін жою, яғни ДҚ бұдан бұрынғы жағдайына қайта оралу рәсімі.

Oracle жүйесінің қалай жұмыс істейтінін түсіну үшін қарапайым транзакцияның жұмыс істеу үлгісін қадамы бойынша қарастырайық.

1. Қосымша пайдаланушылық енгізуді өңдейді және SQL*Net арқылы сервермен қосылады.

2. Сервер қосылуға сұранысты қабылдайды және серверлік үрдісті құрады.

3. Пайдаланушы SQL-командасын (немесе командалар жиынтығын) орындайды. Біздің үлгіде пайдаланушы кестенің жолындағы деректерді өзгертеді деп есептейік.

4. Серверлік үрдіс сәйкес SQL-командалары бар SQL-саласы бар ма жоқ па екенін - бөлінетін пулды қарастырады. Егер ол баламалы бөлінетін SQL-саланы тапса, онда серверлік үрдіс пайдаланушының деректерге қолжетімдік құқығын тексереді. Құқығы бар делік, онда серверлік үрдіс бөлінетін SQL-саласын пайдалана отырып, командаларды орындайды. Алайда егер бөлінетін SQL-саласы табылмаса, онда жаңасына жад бөлінеді, содан кейін бөлшектеу және SQL-командаларын орындау жүзеге асырылады.

5. Серверлік үрдіс SGA-дағы деректерді іздейді (егер болса) немесе оларды SGA-дағы деректер файлынан оқиды.

6. Серверлік үрдіс SGA-дағы деректерді өзгертеді. Серверлік үрдіс деректер файлындағы деректерді тек оқи алады. Кейін DBWR үрдісі өзгертілген деректер блогын тұрақты қоймаға (қатты диск, магнитті таспа) жазады.

7. Пайдаланушы *Commit* (тіркеу) немесе *Rollback* (кейін шегіну) командасын орындайды. *Commit* транзакцияны аяқтайды, ал *Rollback* өзгерістердің күшін жояды. Егер транзакция тіркелген болса, онда LGWR үрдісі оны дереу түрде өзгерістер журналының файлына жазады.

8. Егер транзакция сәтті аяқталса, онда клиенттік үрдіске аяқтау коды жіберіледі. Егер қандай да бір істен шығу болса, онда қателік туралы хабарлама қайтарылады.

Ескертпе. Өзгерістер журналының файлына (redo log file) жазу аяқталғанға дейін, транзакция тіркелген болып есептелмейді. Бұл механизм істен шыққан кезде тіркелген транзакцияның қалпына келтірілуіне себепші болады.

12.8. Oracle қызметтерін шолу

БДҚБЖ Oracle жұмыс істеу кезінде деректердің біртұтастығын қамтамасыз ету, істен шыққаннан кейін дерекқорды қалпына келтіру, қателіктерді ұстап қалу және т.б. сияқты міндеттердің орындалуын қамтамасыз ету керек. Бұл мынадай қызметтермен жүзеге асырылады: бақылау нүктелерін құру (checkpointing), журналдау және мұрағаттау.

Бақылау нүктелерін құру. Бақылау нүктесін құру сигналы DBWR немесе LGWR үрдісінен келіп түседі. Бірақ бақылау нүктесі дегеніміз не және ол не үшін қажет?

Деректер блогының барлық өзгертулері блок буферлерінде болатындықтан, жадтағы деректердің өзгеруі дисктегі осы блоктарда көрсетіле бермейді.

Кэштеу үрдісі соңғы пайдаланылған блок алгоритмі бойынша жүргізіледі, сондықтан тұрақты өзгертілетін буфер соңғы пайдаланылған болып белгіленіп, DBWR үрдісі оны дискке жазады. Бақылау нүктесі осы буферлер дискке жазылу үшін қолданылады. Барлық лас буферлер міндетті түрде дискте сақталу керек.

Бақылау нүктесі екі режимде орындалуы мүмкін: қалыпты бақылау нүктесі және жылдам бақылау нүктесі.

Қалыпты бақылау нүктесі режимінде лас буферлер реттілікпен DBWR үрдісімен жазылады. Бұл режимде бақылау нүктесі аса ұзақ орындалады, бірақ жылдам бақылау нүктесіне қарағанда аз жүйелік ресурстарды қозғайды.

Жылдам бақылау нүктесі режимінде DBWR біруақытта бірнеше буферді жазады. Осындай бақылау нүктесі өте жылдам орындалады. Ол енгізу (шығару) жұмысы тұрғысынан аса тиімді, бірақ сол уақытта жүйенің өнімділігін айтарлықтай азайтады.

Бақылау нүктелерін жиі орындау істен шығу жағдайында жүйені қалпына келтіру үшін қажетті уақыттың артуына себепші болады. Бақылау нүктесі өзгерістер журналы ауыстырылған кезде автоматты түрде орындалады.

Журналдау және мұрағаттау. Өзгерістер журналы (redo log) Oracle ДҚ барлық өзгерістерін жазады. Өзгерістер журналын құру мақсаты жүйенің істен шығу жағдайында және деректер файлы жоғалған жағдайда ДҚ шұғыл қалпына келтіру мүмкіндігі болып табылады. Бұдан бұрын жасалған резервтік көшірмелерден деректердің файлдарын қалпына келтіргеннен кейін, өзгерістер журналының файлдары (соның ішінде журналдың мұрағаттық файлдары) барлық соңғы транзакцияны қайталай алады.

Сөйтіп, деректер файлдары толығымен қалпына келтіріледі.

Өзгерістер журналының файлы толығымен толтырылған кезде, журнал ауыстырылады және LGWR үрдісі жаңа файлды бастайды. Журнал ауыстырылған кезде ARCH үрдісі толтырылған файлды журналдау файлдарының мұрағатына жазады. Мұрағаттау аяқталған кезде, өзгерістер журналының файлы қолжетімді болып белгіленеді. Өзгерістер журналының мұрағаттық файлдары сенімді сақталу керек, себебі олар жүйені қалпына келтіру үшін қажет болуы мүмкін.

12.9. Oracle-дегі триггерлер

Триггерлерді, сақталатын рәсімдерді және жай ғана скрипттерді құру үшін (Oracle-де олар атаусыз блоктар деп аталады) Oracle-дің өз тілі құрылған. Бұл тіл PL/SQL (Program Language SQL) деп аталады.

Әрбір кесте үшін 12 триггер құруға болады. Триггердің шаблоны мынадай:

```
CREATE TRIGGER [name] (триггерді шақыру оқиғасы)...
(триггерді шектеу міндетті емес)
BEGIN
(триггердің әрекет етуі)
END;
```

Триггерді анықтау кезінде қанша рет орындалу керектігін көрсетуге болады: әрбір өзгертілетін жол үшін (row trigger) немесе қанша жол өзгертілетініне тәуелсіз (statement trigger) барлық орындалатын тіркес үшін бір рет.

Row trigger — триггердің жиі пайдаланылатын түрі. Әрбір жол үшін бір рет орындалады. Мысалы, егер SQL-тіркес UPDATE кестедегі бірнеше жолды жаңартса, онда триггер UPDATE тіркесімен өзгертілетін әрбір жол үшін шақырылады. Егер тіркес бірде бір жолға әсер етпесе, онда триггер шақырылмайды.

Statement trigger — бірде бір жол өзгертілмесе де, кестедегі өзгертілген жолдар санына тәуелсіз шақырылады. Мысалы, егер DELETE тіркесі кестеден бірнеше жолды жойса, онда DELETE тіркесі деңгейінің триггері кестеден қанша жол жойылатынына тәуелсіз бір рет қана шақырылады.

Триггердің айқындау кезінде тіркеске дейін немесе кейін - (trigger timing) триггердің орындалу сәтін көрсету керек. BEFORE және AFTER тіркес триггерлеріне және жол триггерлеріне қолданбалы.

INSTEAD-OF (орнына) — Oracle қолданатын триггердің тағы бір түрі.

INSTEAD-OF триггерлері тек Oracle8i редакциясында қолжетімді. Олар көп кестелі және объектті көріністерде пайдаланылуы мүмкін. Олар басқа триггерлерге қарағанда DML-тіркестері (INSERT, UPDATE және DELETE) орындаудың орнына қолданылады.

Көріністі INSERT, UPDATE және DELETE көмегімен қалыпты кесте ретінде түрлендіруге болады және тиісті өзгеріс үшін INSTEAD-OF триггер іске қосылады. INSTEAD-OF триггерлер әрбір өзгертілетін жол үшін активтендіріледі.

INSTEAD-OF триггерлерінің мүмкіндіктерін көрсету үшін әрбір бөлімнің менеджерлерін атап өтетін триггерді (view manager_info) жасаймыз:

```
CREATE VIEW manager_info AS
SELECT d.deptno, d.deptname, e.empno, e.empname
FROM emp e, dept d
WHERE e.empno = d.manager_num;
```

Енді көріністегі ендірмені өңдейтін INSTEAD-OF триггерін айқындаймыз. view manager_info-ге ендірме *dept кестенің manager_num бағанын жаңарту операциясына трансляциялануы мүмкін.*

Триггерде бөлім менеджері көрсеткен бөлімде ең болмағанда бір жұмысшы жұмыс істеу керектігін көрсететін шектеуді айқындауға болады.

```
CREATE TRIGGER manager_info_insert
INSTEAD OF INSERT ON manager_info
--жаңа менеджер туралы ақпарат
REFERENCING NEW AS n
FOR EACH ROW
DECLARE
empCount NUMBER;
BEGIN
/* Бірінші тексеру бөлім жұмысшыларының саны бірден артық екенін растау үшін орындалады */
SELECT COUNT(*) INTO empCount
FROM emp e
WHERE e.deptno =:n.deptno;
/* Егер жұмысшы жеткілікті болса, онда оны менеджер қылу керек*/
IF empCount >= 1 THEN
UPDATE dept d SET
manager_num = :n.empno
WHERE d.deptno = :n.deptno;
END IF;
END;
```

1. Триггер дегеніміз не?
2. Транзакция дегеніміз не?
3. Oracle ДҚ физикалық деңгейі файлдардың қандай санаттарын құрайды?
4. Деректер сөздігі қандай кестелі кеңістікте сақталады?
5. Сегмент, экстент және деректер блогы арасындағы айырмашылық қандай?
6. Oracle данасының негізгі элементтері қандай?

IV БӨЛІМ

ПОСТРЕЛЯЦИЯЛЫҚ ДЕРЕКТЕР ҚОРЫ

13 Тарау

КЕҢЕЙТІЛГЕН РЕЛЯЦИЯЛЫҚ МОДЕЛЬГЕ БЕЙІМДЕЛУ

13.1. Реляциялық деректер қорын жетілдірудің басты бағыттары

Постреляциялық деректер қоры деп аталатын басқару жүйелерін зерттеу және әзірлеу саласындағы басты бағыттарды қарастырайық.

ДҚБЖ-нің келесі буынын дамыту саласындағы басты үш бағытты атап өтуге болады.

Бірінші бағыт ДҚБЖ-нің ағымдағы реляциялық басқару және ұйымдастыру технологияларын барынша қолдана отырып, сыртқы жадты басқару жүйелерін әрі қарай жетілдіруге негізделген.

Екінші бағыт интерфейстері стандартталған модульдер жиынтығы түріндегі генераторларды басқару жүйесін әзірлеуді қамтиды.

Үшінші бағытты, негізінде, алғашқы екі бағыттың синтезі десек те болады. ДҚБЖ қандай да бір ережелер жүйесі мен сол ережелерге сәйкес жасалатын модуль-әрекеттердің жиынтығының интерпретаторы болып табылады. Мұндай жүйелер үшін ережелерді қалыптастырушы арнайы тілдер әзірленеді.

ДҚБЖ-нің келесі буынын реляциялық жүйелердің тікелей ұрпақтары десе де болады. Соған қарамастан, үшінші буын жүйелерінің түрлі бағыттарын жеке-жеке қарастырып өткен абзал, себебі олардың әрқайсысының сипаттамалары әр түрлі.

Реляциялық деректер моделінің басты ережелерінің бірі – қатынастарды қалыптандыру талабы. Бұл ДҚ-ның құрылымын ұғынықты етуге, әрі компьютер жадының едәуір ресурстарын үнемдеуге мүмкіндік береді. Бұл ретте, ақпараттың жинақылығын қамтамасыз ету үшін, соған сәйкес кестелер арасындағы байланыстар қолданылады.

Дегенмен, реляциялық ДҚБЖ бизнес саласындағы басқару міндеттері үшін ғана емес, сондай-ақ өнеркәсіптік өндіріс саласында да (CALS-технологиялар) қолданыла бастады. Реляциялық деректер қорын пайдалану – технологиялық үдерістерді жобалаудың автоматтандырылған жүйесін әзірлеуге, эксперттік жүйелерді басқаруға

және өзге де басқару міндеттері мен өндірістің техникалық дайындығын әзірлеуге өте тиімді

Әдетте, мұндай жүйелер құрылымы күрделі объектілерді – логикалық байланысқан қандай да бір кестелер жиынтығын пайдаланады, олардың ақпараттарын талдау үшін, оның үстіне тиімді техникалық шешімдерді пайдалану үшін күрделі сұраныстар жүргізуге тура келеді.

Реляциялық ДҚ-ны пайдаланудың аталмыш міндеттеріне сәйкес, оларды дамытудың басты бағыты пайда болды. Бұл бағыттың мәні деректер қорын басқару жүйелерінде күрделі объектілердің қалыптасуына сүйенеді, яғни мұндай объектілер ДҚ-ның бастапқы кестелерімен қатар, соған сәйкес сұраныстарды да қамтиды.

Сонымен қатар, мұндай объектілердің логикалық және физикалық көрінісінің арасында нақты шекара сақталады. Әсіресе, әр күрделі объектіні (кез келген күрделілік деңгейіндегі) біртұтас қалпында деректер қорының бір бөлігінен екінші бөлігіне немесе тіпті басқа деректер қорына тасымалдауға немесе көшіруге болады.

Бұл зерттеу саласының ауқымы өте кең, онда деректер моделін, оның құрылымын, сұраныстар мен басқару транзакцияларының, журнализацияның тілдерін әзірлеу мәселелері қозғалады.

Бұл – ДҚБЖ әзірлеудің жаңа бағыты, ол реляциялық модельге сүйенсе де, онда қатынастарды қалыптандырудың толық талаптары қамтылмайтын кездер де болады.

Әр түрлі мәселелерді, әсіресе CALS-технологиялары бағытындағы мәселелерді шешуге реляциялық тәсілді қолдану саласы кеңейген сайын, ДҚ кестелерін қалыптандыру принципін қолдану және әр түрлі сұраныстарды орындау кезіндегі жекелеген транзакциялар мен рәсімдерді жасау – деректер қорын ұйымдастырушы бірыңғайланған сұлбаның барлық артықшылықтарын «жоққа шығаратыны» анықталды.

Нормаланбаған реляциялық деректер модельдерінде объект (жүйенің бастапқы элементі) ретінде тұрақты қарапайым деректер жиыны мен қатынастарының кортеждері (жазбаларды) мен массивтерінің (тұрақты индекстелген деректер жиыны) жасалуы мен сақталуына рұқсат етіледі. Сондай-ақ, мұндай тіркемелер шексіз болуы мүмкін.

Мұндай күрделі объектілер құрылатын деректер қорын басқаратын жүйелерді *объектілі-бағытталған деректер қоры (ОБДҚ)* деп атайды.

Мұндай жүйелердің аталмыш объектілерді әзірлеп, басқаруға қажетті бағдарламалау тілдерін қамтуы керектігі анық. Сонымен қатар, олар әр түрлі деректерді, оның ішінде пайдаланушылар құратын деректерді өңдеу керек.

1995 жылы Sun Microsystems компаниясы жаңа өнімнің – интерпретаторлар тобының Java атты тілінің шығарылғанын мәлімдеді.

Java тілі Си++ тілінің кеңейтілген қосалқы жиыны болып есептеледі. Бұл тілдің өзгешелігі оның операторлар бойынша интерпретацияланатынында (Basic тілінің стилінде), ал Java тілінде жазылған бағдамаларының қауіпсіздігі кепілдендірілген (мәселен, кез келген бағдарламаны орындау барысында интерпретаторға зақым келмейді). Ол үшін, көбіне, тіл рәсімдерінің ішінен сілтемелер үстіндегі математикалық әрекеттер алынып тасталады. Сондай-ақ, Java құрамында абстрактілі деректер түрін анықтайтын озық құралдары бар, қуатты объектілі-бағытталған тіл болып табылады. Sun Microsystems компаниясы Java тілін "ғаламдық тор" (World Wide Web) – Интернеттің қызмет көрсету мүмкіндіктерін кеңейту мақсатында дамытуда. Ондағы басты идея – WWW серверінен клиенттерге деректер немесе деректерді өңдеу нәтижелері емес, Java тілінде бағдарламаланатын және клиенттің тарапында орындалатын объектілердің орындалу әдістері берілетіндігін қамтиды.

13.2. Қосымшаларға бағытталған деректер қоры жүйелерінің генерациясы

ДҚБЖ дамуы барысында аталмыш бағыттың пайда болуы – кез келген қосымшада қолдануға жеткілікті әрі артық болатын деректер қорын басқаратын бірегей жүйені жасап шығару мүмкін еместігімен анықталады. Мәселен, өндіріс пен кәсіпкерліктегі практикалық міндеттерді шешу үшін қолданылып жүрген қазіргі ДҚБЖ-лердің қолданысына үңілетін болсақ, көп жағдайда жүйе мүмкіншіліктерінің 30%-ы ғана қолданылатынына көзіміз жетеді. Соған қарамастан, кең таралған жағдайларда қолдануға есептелген қосымша ДҚБЖ-сінің бар ауыртпалығын қосымшаның өзі көтеретіндігі рас.

Сондықтан, аяқталмаған бірегей ДҚБЖ өндіргеннен гөрі, нақты бір қосымшаға (немесе қосымшалар класына) бағытталған деректер қорының жүйесін жинауға мүмкіндік беретін Compiler compiler) компиляторларын жасап шығарудың болашағы әлдеқайда зор секілді.

Осылайша, жол жүру билеттерін резервте сақтау жүйелеріндегі сұраныстар, әдетте, өте қарапайым болады (мәселен: "№645 рейсіне кезекті орынды беру"), сондықтан сұраныстарды ауқымды түрде жетілдірудің қажеті жоқ. Алайда деректер қорында сақталатын ақпараттың кейде шиеленісіп кететіндігі соншалық (бір орынға екі немесе одан да көп билет болатын жағдайлар кімнің басынан өтпеді?), деректер қорының жаңартуларын синхрондау және оны кез келген ақау немесе істен шығудан кейін қалпына келтірудің кепілдігі аса маңызды.

Статистикалық жүйелерде сұраныстар күрделі болуы мүмкін (мәселен: "Ресей аумағында тұратын және тіркелген балаларының саны үштен аспайтын бойдақ ер адамдардың санын шығару"), сондықтан мұндай жағдайлар сұраныстарды жетілдірудің озық құралдарын қолдану қажеттілігін туғызады.

Қозғалып отырған жайт статистика жайлы болғандықтан, мұнда транзакцияларды қатаң түрде топтамалау және әр ақау мен істен шығудан кейін оларды нақты бұрынғы қалпына келтіруді сақтау талап етілмейді. (Әңгіме статистикалық ақпарат туралы болғандықтан, оның бірнеше бетін жоғалту әдетте аса көлемді шығын болып есептелмейді.)

Сондықтан, мүмкіншіліктері қосымшаның қажеттіліктеріне тиісті деңгейде сәйкес келетін деректер қорының жүйесін генерациялау машығын дамытқан жөн. Қазіргі таңда коммерциялық нарықта мұндай генерациялау жүйелері жоқ (мәселен, нақты бір қосымшаны жасау барысында Oracle жүйесінің сервері таңдалған болса, жүйенің қандай да бір ерекшеліктерінен бас тарта алмайсың).

13.3. Ережелермен басқарылатын сұраныстарды оңтайландыру

Реляциялық ДҚБЖ-дегі сұраныстарды оңтайландыру дегеніміз, әдетте, сұраныстың бастапқы көрінісі бойынша оны түрлендіру арқылы сұранысты орындаудың оңтайлы рәсімдік жоспарын жасап шығаратын сұранысты өңдеу тәсілі дегенді білдіреді.

Сұраныстың бастапқы көрінісінің сәйкес түрлендірулері ДҚБЖ-нің арнайы компоненті – оптимизатордың (оңтайландырғыш) көмегімен орындалады, және ол өндіретін сұранысты орындау жоспарының оңтайлылығы субъективті сипатқа ие, себебі әзірлеуші оңтайлылық критерийін оптимизаторға салып қойған.

Сұраныстар оптимизаторларына қатысты басты мәселе – оларды бағдарламалаудың қабылданған технологиясының жоқтығына келіп тіреледі. Әдетте, оптимизатор – компилятордың өзге компоненттерімен тығыс байланысқан, салыстырмалы түрде дербес рәсімдер жиынтығы болып табылады. Осы себепті, оңтайландыру стратегияларын өзгерту немесе оларды сапалы түрде кеңейту қиындықтар туғызады (жалпы оңтайландыру мен сұраныстарды оңтайландыру эмпирикалық тәртіп болғандықтан, мұны жасауға тура келеді, ал жақсы эмпирикалық алгоритмдер уақыт өте келе пайда бола бастайды).

Бұл мәселені компиляторлар өндірісінің дәстүрлі технологияларының шекарасынан шықпайтын ымыралы шешімдердің көмегімен шешеді. Жалпы алғанда, олардың барлығы компиляторлар құрылымын автоматтандыруды қамтамасыз ететін қандай да бір құралдарды қолдануға байланысты болады. Мәселен, ондай құралдар DB2, Oracle, Informix жүйелерінде болады.

13.4. Динамикалық ақпарат пен темпоралды сұранысты қолдау

Дәстүрлі реляциялық ДҚ пән саласы моделінің бір сәттік кескінін сақтайды. Қандай да бір объектінің t уақыт моментіндегі қандай да бір өзгерісі сол объект күйінің одан бұрынғы уақыт моментіне қайта

алмауына алып келеді. Шын мәнінде, озық ДҚБЖ-лерінің көптеген түрлерінде объектінің бұрынғы күйі өзгерістер журналында сақталады, бірақ қолданушыларға оларды қолдануға рұқсат жоқ.

Әрине, сақталатын қатынастарға айқын уақыт белгісін енгізіп, оның мәнін қосымшалар деңгейінде сақтап тұруға болады. Көп жағдайда солай жасалады. Осылайша, SQL стандартында деректердің арнайы типтері –*date* және *time* пайда болды. Алайда, мұндай тәсілдің де кемшіліктері болады: ДҚБЖ уақыттық қатынас өрісінің мағынасын білмейді де, оның мәндерінің дұрыстығын қадағалай алмайды. Осыған байланысты, қосымша сақтаудың артық көлемдері пайда болады (деректер объектісінің бұрынғы күйі басты ДҚ-да да, өзгерістер журналында да сақталады).

Темпоралды ДҚ деп аталатын зерттеу және әзірлеу саласының жекелеген бағыты да бар. Бұл салада деректерді модельдеу, сұраныс тілдері, сыртқы жадтағы деректерді ұйымдастыру және т.б. мәселелер зерттеледі. Темпоралды жүйелердің басты тезисі t1 уақыт моментінде жасалып, *t2* уақыт моментінде жойылған кез келген деректер объектісі үшін, ДҚ-да оның [t1,t2] уақыт интервалындағы барлық күйлері сақтаулы (сондай-ақ, оларды қолданушылар да қолдана алады).

Темпоралды ДҚБЖ прототиптерінің құрылыстары мен зерттеулері, әдетте, реляциялық жүйе үстіндегі қондырма бөлім түріндегі реляциялық ДҚБЖ негізінде жүзеге асырылады.

Бақылау сұрақтары

1. Реляциялық деректер қорын жетілдірудің басты бағыттарын атаңыз.
3. Деректер қоры жүйесін генерациялау әдісінің мәні неде?
4. Сұраныстарды оңтайландыру тәсілдерін атаңыз.
5. Темпоралды сұраныстар қандай міндеттер үшін қолданылады?

14 Тарау

ОБЪЕКТІЛІ-БАҒЫТТАЛҒАН ДҚБЖ

14.1. Объектілі-бағытталған тәсіл туралы жалпы түсініктер

Объектілі-бағытталған деректер қоры бағыты 1980жылдардың ортасында пайда болды. Бұл бағыттың ең қарқынды дамыған көрінісі соңғы жылдары айқын байқалуда.

ОБДҚ бағытының пайда болуы, ең алдымен, тәжірибе жүзіндегі қажеттіліктерден – алдыңғы ДҚ жүйелерінің технологиялары айтарлықтай қанағаттанарлық деңгейде болмаған күрделі ақпараттық қолданбалы жүйелерді жасап шығару қажеттілігімен анықталады.

Әрине, ОБДҚ жоқ жерден пайда болды дей алмаймыз. Оған сәйкес базисті ДҚ саласындағы бұрынғы жұмыстармен қатар, бұрыннан дамып келе жатқан объектілі-бағытталған бағдарламалау тілдері мен абстрактілі деректер типіндегі бағдарламалалау тілдері құрады.

ДҚ саласындағы бұрынғы жұмыстармен байланысы туралы айтар болсақ, біздің ойымызша, ОБДҚ саласындағы жұмыстарға реляциялық ДҚБЖ зерттеулері мен солардан кейінгі (хронологиялық тұрғыда), күрделі объектілерді басқара алатын ДҚ тобы ең ауқымды ықпал тигізеді. Сонымен қатар, ОБДҚ идеялары мен тұжырымдамаларына және бүкіл объектілі-бағытталған тәсілдерге ең айрықша әсер тигізген деректерді семантикалық модельдеуге деген көзқарас болды. Сондай-ақ, ОБДҚ-мен қатар дамып келе жатқан дедуктивті және белсенді ДҚ бағыттарының да әсері мол.

Көп жағдайда, объектілі-бағытталған тәсіл төмендегідей тұжырымдамаларға негізделеді:

- объект және объект идентификаторының;
- атрибуттар мен әдістердің;
- кластардың;
- кластар иерархиясы мен мұрагерлігінің.

Объектілі-бағытталған тілдер мен жүйелерде шынайы өмірдің кез келген болмысы объект түрінде бейнеленеді. Жасалу барысында кез келген объект жүйе генерациялайтын бірегей идентификаторға ие болады, ол объектімен үнемі байланыста болады, және объект күйі өзгергенде, ол өзгеріске ұшырамайды.

Әр объектінің өз күйі мен қылығы болады. Объект күйі дегеніміз – оның атрибуттарының мәндерінің жиынтығы. Объектінің қылығы – объект күйіне сүйеніп әрекет ететін әдістердің жиынтығы (бағдарламалық код).

Объект атрибутының мәні – ол да қандай да бір объект немесе объектілер жиынтығы. Объектінің күйі мен қылығы объект ішінде капсулданған; объектілер хабарламалар алмасу және сәйкес әдістерді орындау арқылы өзара әрекеттеседі.

Атрибуттары мен әдістері бірдей объектілер жиынтығы объектілер класын құрайды. Объект тек бір класқа тиесілі болуы керек (егер мұрагерлік мүмкіндіктер қарастырылмаса). Объект-үлгілерінде атрибуттары (бүтін, тармақтар және т.б.) жоқ қарапайым, алдын ала анықталған кластардың болуы да ықтимал. Басқа класс объектілері үшін төлсипат мәндері ретінде қызмет ете алатын класс осы атрибуттың *домені* деп аталады.

Қазіргі бар кластың негізінде жаңа класты қалыптастыру – мұрагерлікке рұқсат етіледі. Бұл жағдайда, бар кластың (суперкласс) тармағындағ ыкласс деп аталатын жаңа класс суперкластың барлық атрибуттары мен әдістерін иеленеді. Сондай-ақ, класс тармағында қосымша атрибуттар мен әдістерді анықтауға болады. Қарапайым және көптік мұрагерлік жағдайлары кездеседі. Бірінші жағдайда, тек қана бір суперкласс негізінде класс тармағы анықталуы мүмкін, ал екінші жағдайда класс тармақтары бірнешеу болуы мүмкін. Егер тіл немесе жүйе кластардың жеке мұрагерлігін қолдаса, кластар жиынтығы ағаш тәріздес иерархияны құрайды. Көптік мұрагерлік сақталған жағдайда, кластар кластық тор деп аталатын түбірімен бағдарланған кескінге қосылады. Класс объектісі осы кластың кез-келген суперкласына жатады деп есептеледі.

Объектілі-бағытталған тәсілдің соңғы идеяларының бірі – суперкласс атрибуттары мен әдістерін класс тармағында (шамадан тыс жүктеу әдістерін) қайта анықтау мүмкіндігі. Бұл мүмкіндік икемділікті арттырады, бірақ қосымша мәселені тудырады: объектілі-бағытталған бағдарламаны құрастырған кезде, оның класы (жалпы жағдайда, суперкласс) белгілі болса да, объектінің әдістерінің құрылымы мен коды белгісіз болуы мүмкін. Бұл мәселені шешу үшін, кешіктіріп байланыстыру әдісі деп аталатын, яғни хабарламаны жіберу кезінде объектіні іске асыру туралы мәліметтерді тану арқылы программаны орындаудың интерпретациялау тәсілі қолданылады. Класс тармақтарын анықтау тәсіліне кейбір шектеулерді енгізу арқылы, интерпретацияны қолданбай, тиімді жүзеге асыруға болады.

Осы негізгі ұғымдардың жиынтығымен кластардың мұрагерлік мүмкіндіктері мен соларға сәйкес мәселелерді назарға алмаса, объектілі-бағытталған тәсіл абстрактілі (немесе ерікті) деректер түрлерімен бағдарламалау тілдері тәсілдеріне ұқсас болып келеді.

Басқа жағынан қарасақ, егер объектілердің қылық аспектілерін жалпылайтын болсақ, объектілі-бағытталған тәсіл деректерді

семантикалық модельдеу тәсіліне (тіпті терминология бойынша) жақын деуге де болады. Семантикалық модельдерге негізделген басты абстракциялар объектілі-бағытталған көзқараста жанама түрде пайдаланылады. Атрибуттар мәндеріне өзге объектілер жатуы мүмкін күрделі объектілерді құру – агрегация абстракциясына негізделеді. Топтау абстракциясы – объект кластарын қалыптастырудың негізі болып табылады. Мамандандыру (қорыту) абстракциясы иерархияның немесе кластар торының құрылысына негізделген.

ОБДҚ-ның объектілі-бағытталған тәсілге қол жеткізуге мүмкіндік беретін ең басты жаңа артықшылығы – объектілердің қылық аспектісі болып табылады. Дәстүрлі ұйымдастыруға негізделген ДҚ-ға сүйенетін қолданбалы ақпараттық жүйелерде (семантикалық деректер модельдеріне негізделген түрлеріне дейін), қылық және құрылымдық бөліктер арасында айтарлықтай алшақтық байқалды. Жүйенің құрылымдық бөлігі ДҚ-ның бүкіл аппараттарының қолдауымен жұмыс істеді, оны модельдеуге, верификациялауға болатын, ал қылық бөлігі, әдетте, оқшау түрде құрылатын. Атап айтқанда, осы құрылымдық (статикалық) және қылық (динамикалық) бөліктерді бірге модельдеу және олардың үйлесімділігін қамтамасыз ету үшін қажетті ресми аппараттар мен жүйелік қолдауларболмады. ОБДҚ ортасында қолданбалы жүйені жобалау, әзірлеу және техникалық қызмет көрсету құрылымдық және қылықтық аспектілер интеграцияланатын процесс болып табылады. Әрине, бұл объектілерді анықтауға және оларға негізделген қолданбалы жүйені жасауға мүмкіндік беретін арнайы тілдерді талап етеді.

ДҚ-ны ұйымдастыру және басқару үшін объектілі-бағытталған тәсілді қолданудың ерекшелігі классикалық тұжырымдамалардың нақтылана түсіндірілуін және оларды кеңейтуді талап етті. Бұл сыртқы жадқа объектілерді ұзақ уақыт сақтауға, объектілердің ассоциативті қолжетімділігіне, мәліметтер мультиқолжетімділік жағдайында ОБДҚ-ның үйлесімді қалпын сақтауды және тағы да басқа мүмкіндіктерді қамтамасыз ету қажеттілігімен анықталды. Дәстүрлі парадигмада кездеспейтін, бірақ ОБДҚ-да талап етілетін үш аспектіні атап өту керек.

Бірінші аспект класты анықтау барысында (спецификалық шектеулер, шегерім ережелері және т.б.) мағлұматтарды сипаттау құралдарына мұқтаждыққа келіп тіреледі.

Екінші аспект — әр түрлі кластардың объектілері арасындағы семантикалық байланыстардың барлық түрлерін анықтау механизмінің қажеттілігіне негізделеді. Шын мәнінде, бұл деректерді семантикалық модельдеу құралдарының ОБДҚ-да толықтай таратылуын талап етеді. Ұқсастыру абстракциясын пайдалану қажеттілігі ОБДҚ-ның автоматтандырылған жобалау және инженерия саласында пайдаланылуына байланысты болып табылады.

Үшінші аспект класс ұғымын қайта қарастырумен байланысты. ОБДҚ контексінде класты осы түрдегі объектілер жиынтығы ретінде қарау, яғни, сонымен бірге объектілердің типі мен кластардың тұжырымдамаларын қолдау ыңғайлы.

14.2. Объектілі-бағытталған деректердің модельдері

Алғашқы нысандандырылған және көпшілік мақұлдаған деректер моделі Кодд реляциялық моделі болды. Алдыңғы модельдерде болғандай, бұл модельде де келесідей үш аспект бөлінді: құрылымдық, тұтас және манипуляциялық. Реляциялық модельдегі деректер құрылымдары жазық қалыпқа келтірілген қатынастарға негізделеді, тұтастық шектеулер бірінші реттік логиканы қолдану арқылы анықталады, деректерді өңдеу реляциялық алгебра немесе оған теңестірілген есептеме негізінде жүзеге асырылады. Көп жағдайларда реляциялық деректер моделінің сәтті болуы – оның реляциялық алгебраның қатаң математикалық аппараттарына және теориялар жиынтығына негізделгендігіне байланысты.

Деректерді объектілі-бағытталған тәсілмен модельдеудің басты мәселелері жалпы объектілі-бағытталған деректер моделі негізделетін арнайы математикалық аппараттың жоқ болуымен байланысты. Объектілердегі деректерді басқару әдістерін әзірлеу, дәстүрлі емес міндеттерді бағдарламалау процесі секілді, «бағдарламалау өнері» болып қала береді.

Әдістер *жалпыға ортақ* (басқа кластардың объектілерінен қолжетімді) немесе *жеке* (тек осы класта ғана қолжетімді) болуы мүмкін.

Мәселен, объектілі-бағытталған деректер қорын басқару жүйесі дегеніміз – бұл бағдарламалау жүйесімен ДҚБЖ-нің бірігуін білдіреді және ол объектілі-бағытталған деректер моделіне негізделген.

ОБДҚ-ның негізгі мақсаты – бірыңғай ақпараттық кеңістікті құру қажеттілігіне байланысты.

Бұл ортада жобаның құрылымдық және қылық бөліктері арасында ешқандай қайшылық болмауы керек және сыртқы жадтағы күрделі деректер құрылымдарын тиімді басқару керек.

Қосымшаны жасаған кезде дәстүрлі реляциялық жүйелерден айырмашылығы, мұнда скалярлық мәндермен жұмыс істеуге бағытталған процедуралық бағдарламалау тілін және жиынтықтармен жұмыс істеуге бағытталған декларативті сұраныс тілін қолдануға тиіспіз, ОБДҚ тілдік ортасы – табиғи түрде қамтылатын объектілі-бағытталған бағдарламалау жүйесі, ұзақ мерзімді объектілермен жұмыс істеу құралы.

Бағдарламалау тілінде дерекқорлармен жұмыс істеу құралдарын табиғи түрде қосу ұзақ мерзімді (сыртқы ДҚ-да сақталатын) объектілермен жасалатын жұмыс бағдарламада болған уақытша объектілермен жұмыс істейтін синтаксистік құрылымдардың (семантикасы бірдей) негізінде жасалуы керек дегенді білдіреді.

ОБДҚ-ның бұл аспектісі деректер қорын бағдарламалау тілдеріне жақын бағыт болып табылады. ОБДҚ және ДҚ бағдарламалау тілдері өздерінің көптеген сипаттары бойынша тек терминология тұрғысынан ғана ерекшеленеді; басты айырмашылық – ОБДҚ тілдеріндегі кластық мұрагерлік тәсілінің сақталуы.

ОБДҚ тілдік ортасының тағы бір аспектісі интерактивті түрде қолданылуы мүмкін сұраныс тілдеріне деген қажеттілік. Егер ОБДҚ бағдарламалау тілдерінде сыртқы ДҚ объектілеріне қатынасу негізінен навигациялық тұрпатта болса, онда сұраныс тілдері декларативті стиль үшін ыңғайлы болады. Өздеріңіз білетіндей, декларативті сұраныс тілдері бағдарламалау тілдеріне қарағанда аз дамыған.

14.3. Объектілі-бағытталған деректер қорының бағдарламалау тілдері

Қазіргі таңда, нөлден бастап жаңадан ойластырылған ОБДҚ бағдарламалау тілі жоқ. Мұндай тілді құрудың нақты тәсілі ретінде қолданыстағы объектілі-бағытталған тілді қолданылу (қажетті кеңейтумен) қарастырылды.

Бағдарламалауға объектілі-бағытталған және функционалдық тәсілдерді енгізуді жүзеге асырған алғашқы тілдердің бірі – Лисп тілі (Common Lisp) болды.

Неғұрлым тиімді іске асыру қажеттілігі объектілі-бағытталған тілдің негізі ретінде BASIC (BASIC) және C ++ танымал бағдарламалау тілдерін қолдануға итермеледі.

Объектілі-бағытталған ДҚБЖ-сінде ОБДҚ бағдарламалау тілін (немесе тілдер тобын) ғана емес, сонымен қатар дамыған сұрау тілін қолдау қажеттілігін қазіргі таңда барлық дерлік әзірлеушілер мойындап отыр. Жүйе интерактивті режимде түпкі пайдаланушысына тікелей қол жеткізуге болатын жеңіл оқу интерфейсін қолдауы керек.

Объектілі-бағытталған дерекқор жүйелерімен интерактивті интерфейстерді ұйымдастырудағы ең кең таралған тәсіл аралаушыларды қолдануға негізделеді. Бұл жағдайда соңғы интерфейс әдетте графикалық болып табылады. Экран ОБДҚ сұлбасын (немесе ішкі сұлбаны) көрсетеді және пайдаланушы навигациялар арқылы объектілерді көре алады. Кейбір зерттеушілер бұл жағдайда объектілерді инкапсуляциялау принципін елемеуге және объектілердің интерьерін пайдаланушыға ұсынуға болады деп санайды.

Көптеген ОБДҚ жүйелерінде мұндай интерфейс бар, бірақ навигациялық сұраныстар тілі дегеніміз – тіпті реляциялық жүйелердің сұраныс тілдерімен салыстырғанда, артқа жасалған қадамдай қабылданатыны анық.

Навигациялық емес сұраныс тілдері. Қазіргі уақытта мұндай тілдерді дамытудың үш тәсілі бар.

Бірінші тәсіл– бұл реляциялық жүйелердің сұраныс тілдерін кеңейту. SQL тіліне жақын синтаксиспен ең көп таралған тілдер. Бұл, әрине, осы тілдің кеңінен таралуы мен көпшіліктің мақұлдағанымен байланысты.

Екінші тәсіл толық логикалық объектілі-бағытталған есептеу тіліне негізделген.

Үшінші тәсіл декларативті және объектілі-бағытталған бағдарламалау әдістерін қолдануға негізделген.

Сұраныс тілін әзірлеу барысында қолданылатын тәсілге қарамастан, әзірлеушілер объектіліө-бағытталған тәсілдің дәстүрлі арнасына сай келмейтін бір тұжырымдамалық мәселеге тап болады.

ОБДҚ тұжырымдамасына сүйенсек, сұранысты қалыптастырудың негізін ОБДҚ-да бірдей типтегі объектілер жиынтығын бейнелейтін класс құруы керек. Алайда сұраныс нәтижесі қандай болуы мүмкін? Объектілі-бағытталған тәсілдің басты ұғымдарының жиынтығы осы мәселенің мәніне сәйкес келетін тұжырымдаманы қамтымайды. Әдетте, бұл мәселені шешу үшін, объектілер жиынтығының базалық тұжырымдалар жиынтығын кеңейтіп, сұраныстың нәтижесі объектілердің белгілі бір жиынтығы – класс үлгілері болады деген тұжырымға келеді. Бұл өте шектеулі тәсіл, себебі сұраныс тілінің құрамында реляциялық байланыс операторына ұқсас құралдардың болу мүмкіндігі автоматты түрде шығарылып қалады.

Сұранысты оңтайландыру мәселелері. ОБДҚ жүйесінде сұранысты оңтайландырудың негізгі мақсаты сыртқы ОБДҚ жадына қол жеткізу арқылы оңтайлы сұранысты орындау жоспарын жасау болып табылады.

Сұранысты оңтайландыру реляциялық дерекқорлар контексінде жақсы зерттеліп, әзірленген. Сұранысты процедуралық емес деңгейде синтаксистік және семантикалық оңтайландырудың белгілі әдістері, қарапайым реляциялық операцияларды орындау алгоритмдері, сұраныс жоспарының құнын бағалау әдістері бар.

Әрине, объектілер жазықтық қатынастарынан гөрі әлдеқайда күрделі құрылымға ие болуы мүмкін, бірақ бұл айырмашылық аса маңызды емес. ОБДҚ сұраныстарын оңтайландырудағы негізгі қиындық, іріктеу шарттары объектілердің (әдістерінің) сыртқы атрибуттарында пайда болған кезде болады, ал нақты оңтайландыру (яғни оңтайлы жоспар әзірлеу үшін) үшін ішкі атрибуттарда анықталған шарттар (күй айнымалылары) қажет.

Осыған ұқсас жағдай реляциялық ДҚБЖ-лерінде де ДҚ көрінісіндегі сұранысты оңтайландыру кезінде кездеседі. Бұл жағдайда, шарттар тағы да сыртқы атрибуттар (көрініс атрибуттары) терминдерінде құрылады және сұранысты оңтайландыру үшін бұл шарттар сақталған қарым-қатынас атрибуттарында анықталған шарттарға айналуы керек. Осындай алдын-ала оңтайландырудың кең тараған тәсілі – қажетті түрлендірулерді қамтамасыз ететін (SQL тілін қолданған жағдайдың бәрінде болмаса да) көріністерді алмастыру болып табылады.

ОБДҚ жүйелерінде бұл жағдай екі түрлі жайттың әсерінен қиындайды.

Біріншіден, әдістер әдетте кейбір процедуралық бағдарламалау тілдерінде бағдарламаланады да, олардың параметрлері болуы мүмкін, яғни жалпы жағдайда көрініс атрибуттарын анықтау кезіндегідей әдіс денесі тек арифметикалық өрнектен бөлек, сонымен қатар, басқа объектілердің тармақтарын, функционалдық шақыруларын және әдістерін қамтитын параметрленген бағдарламаны қамтиды.

Екіншіден, әдісті нақты іске асыру, тіпті объектінің құрылымы сұранысты компиляциялау уақытында белгісіз болуы мүмкін.

Мәселені жеңілдетудің бір жолы – объектілердің ішкі атрибуттарының кейбірін (оңтайландыру үшін ең маңыздыларын) көру мүмкіндігін ашу. Бұл тұрғыда көру мүмкіндігін тек сұраныс компиляторы үшін ғана ашу да жеткілікті болатын еді, яғни мұндай айнымалыларды класс тармақтарында қайта анықтауға тыйым салынатын еді. Пайдаланушы тұрғысынан, мұндай атрибуттар сәйкес түрдің мәнін қайтаратын параметрлер жоқ әдістерге ұқсайды. Дегенмен, бұл сұранысты оңтайландырудың қажеттіліктерін ескере отырып, объектілерді қатаң инкапсуляциялауды (қосымшаны іске асыруға сыни тәуелділіктен сақтау үшін) және ОБДҚ сұлбасын мұқият жобалау мүмкіндігін қамтамасыз еткен абзал болар еді.

Объектілердің бір класына сынамаларды іріктеу шарттарын алдын-ала оңтайландыру тәсілі төмендегідей болуы мүмкін.

1. Шарттың логикалық формуласын конъюнктивті қалыпты формаға айналдыру. Біз нақты бір ҚҚФ-ны таңдау әдісіне тоқталмаймыз, бірақ, әрине, жақсы ҚҚФ таңдалуы керек (мысалы, атомарлық конъюнкттер саны ең көп).

2. Денені компиляциялау уақытында белгілі әдістерді қамтитын кез-келген конъюнкт үшін, әдістер шақыруларын олардың параметрлері қойылған денелерімен ауыстыру. (Бұл параметрлерде басқа объектілердің функцияларына немесе әдістеріне шақырулар жоқ деп болжайық.)

3. Осындай кез-келген конъюнкт үшін барлық жеңілдетулерді жасау, яғни, статикалық есептелуі мүмкін барлық деректерді есептеу. Жалпы алғанда, бұл тапсырма өте қиын болса да, ОБДҚ-ны дұрыс ойластыратын болса, әдістер қатарына оңай жүзеге асырылатын қарапайым әдістер қосылуы мүмкін, оларда шарттарды орнату өте қарапайым түрде жүреді. Мұндай шарттар өте тиімді оңтайландырылатын болады.

4. Егер де айнымалы мәндер мен тұрақты мәндерге негізделген қарапайым салыстыру предикаттары конъюнкттар пайда болса, осы конъюнкттарды оңтайлы сұранысты оңтайлы орындау жоспарын жасау үшін пайдалану. Егер мұндай конъюнкттарды алу мүмкіндігі болмаса, объектілер класын «фильтрлеудің» жалғыз әдісі – әр объектінің логикалық өрнегін толықтай есептел шығару арқылы (мүмкін жеңілдетілген) ретімен тексеріп шығу болып табылады.

Оңтайландыру мүмкіндіктері бағдарламалау әдістеріне, белгілі бір сұраныс тілінің ерекшелігіне және ОБДҚ сұлбасының қаншалықты жақсы жобаланғанына байланысты бағдарламалау тілінің ерекшеліктеріне байланысты болады. Атап айтқанда, қолданылатын бағдарлама тілі объектілерді бағдарламалау әдістерінің ең тәртіпті бағдарламалау стилін ынталандыруы керек. Сұраныс тілі қолданушылардың мүмкіндіктерін (атап айтқанда, сұраныс шарттарында қолданылатын әдістердің параметрлері бойынша) шектеуі керек. ОБДҚ сұлбалары кластарда қарапайым әдістерді қамтуы керек, олар класс тармақтарында кездеспейді және олар қолжетімділік әдістерін ұйымдастырудың негізін құрайтын күй айнымалыларына негізделген.

Бұл шектеулер қолданбалы бағдарламаның ОБДҚ-ны жүзеге асыру ерекшеліктеріне тәуелді болып қалуына әкелмейді, себебі объектілер толығымен инкапсуланған күйде қалады. Сұраныс шарттарында қарапайым әдістерді қолдану іске асыру талаптары бойынша емес, объектілердің семантикасы арқылы ынталандырылуы керек.

Бақылау сұрақтары

1. Объектілі-бағытталған тәсілдің деректер қорын құрудағы принциптерін атаңыз.
2. Объектілі-бағытталған деректердің қандай модельдерін білесіз?
3. Объектілі-бағытталған деректер қорын әзірлеу үшін қандай

ОБЪЕКТІЛІ-БАҒЫТТАЛҒАН ДҚБЖ CACHE

15.1. ДҚБЖқұрылымы

Cache ДҚБЖ постреляциялық объектілі-бағытталған ДҚБЖ-ға жатады. «Постреляциялық ДҚЖБ» термині жаңа буынның ДҚБЖ-сіне жатады дегенді білдіреді. Бұл жерде деректердің біркелкі архитектурасы мен объектілі-бағытталған технологияны толықтай қорғау сияқты технологиялық жаңалықтардың қанша қатары болса, соншалықты уақыттың бірнеше аспектілері айтылып тұр (ДҚБЖ өзінің негізгі реляциялық бәсекелестерінен кейін барып пайда болды).

Аталып өткен объектілі-бағытталған деректер қорының жобалау принциптеріне сәйкес Cache жүйесінің келесі мүмкіндіктері бар:

- тек деректер ғана емес, оларды өңдейтін әдістері де сақталатын ДҚ объект-элементтерінің болуы;
- бұл жүйеде мультимедиалық деректерді өңдеуге болады және қолданушыларға кез-келген күрделілік дәрежесіндегі өздерінің деректер құрылымын жасауға мүмкіндік береді;
- ОБДҚ абстракцияның жоғары деңгейіндегі жұмыстарды жібереді.

ДҚБЖ жүйесінің басты ерекшелігі деректердің бірыңғай архитектурасы арқылы жүзеге асатын деректерді сақтау қабілетінің әдеттегі көрсеткішіне тәуелді болмауы. Аталған архитектураның шеңберінде транзакцияның өңдеуіне бағытталған деректер қорының көп өлшемді құрылымында үздіксіз байқалып отыратын объект пен кестелердің біркелкі сипаттамалары бар. Объектілердің класы анықталғаннан кейін автоматты түрде осы класстың SQL форматындағы реляциялық сипаттамасы туындайды. Дәл осындай жолмен деректер Сөздігіне деректерді анықтайтын тіл (ДАТ) яғни реляциялық деректер қоры форматындағы сипаттамасы түскеннен кейін автоматты түрде деректердің реляциялық және объектік сипаттамасы туындайды және сол арқылы объект форматындағы жол қалыптасады. Осы тұста барлық сипаттамалар келісілген түрде жүзеге асады және барлық операциялар түзетулер бойынша тек бір ғана деректің сипаттамасымен өткізіледі. Бұл өңдеудің уақытын қысқартуға және есептелген ресурстарды үнемдеуге мүмкіндік береді. Қосымшалар айтарлықтай тез жұмыс істейді. 15.1 кестесінде ДҚБЖ архитектурасының негізгі элементтері көрсетілген:

- жүйе жұмыс істейтін платформалар;

Кесте 15.1. Cache постреляциялық ДҚБЖ-сінің архитектурасы (Сәулеті)

Direct	Objects	SQL	WEB
Cache Object Script			
MD	Objects		SQL
MDS			
Платформалар			

- деректердің көп өлшемді сервері;
- деректерге апарар жолдың үш мүмкіндігі;

- Cache' ObjectScript бизнес-логикасының сипаттама тілі;
- жобалау құралдарына арналған интерфейстер және қосымшалардың өңдеулері және Cache' Server Pages Web-технологиясы.

Cache — көп платформалы жүйе. Cache келесі операциялық жүйелерді қолдайды: ОС Windows-тың барлық гаммаларын, Linux, Unix және Open VMS негізгі орындауларын. Unix-тің жаңа орындауларын қолдау жоспарланып жатыр. Itanium жаңа платформасына баса назар аударылуда.

Жалпы деректер Cache-те деректердің (MDS) көпөлшемді серверінің басқаруымен сақталып отырады. Cache негізінде деректердің қаншалықты көп мөлшерде қолданылатынын көрсететін және оларды сақтауға мүмкіндік беретін деректердің транзакционды көпөлшемді моделі жатыр. Деректердің көпөлшемді сервері оларды тек екі өлшемді кестеде ғана сақтауға мүмкіндік беретін реляциялық ДҚБЖ-ның қойған көптеген шектеулерін жоя алады. Бәрімізге белгілі ДҚ реляциялық моделі деректердің күрделі құрылымымен жұмыс жасағанда қажетті болып келетін көптеген кестелерден тұрады. Бұл, өз кезегінде күрделі транзакцияларды орындауда айтарлықтай қиындықтар туғызып, қажетсіз информациялардың сақталуына әкеліп соғады. Cache барлық деректерді көп өлшемді кесілген массивтер түрінде сақтайды.

Деректердің бірегей транзакционды көп өлшемді моделі реляциялық ДҚБЖ-ға лайықты кейбір проблемаларды болдырмауға жол беріп, сол деректерді сақтау деңгейіне дейін оңтайландырады.

Cache деректерінің көп өлшемді сервері үлкен және өте үлкен деректер қорының (жүздеген гигабайт, терабайттар) жүйесіндегі транзакцияларды өңдеуге және бір мезетте жұмыс істейтін көп мөлшердегі қолданушыларға арналған. Cache деректердің көп өлшемді сервері артық деректер мен кестелерді сақтаудан бас тарта отырып, жоғары өндірушілікті алуға мүмкіндік береді.

Cache деректер транзакционды моделі деректерді сақтау деңгейіне дейін оңтайландырып, деректердің объекті моделі мен күрделі түрлерін

қолдауға мүмкіндік береді. Осы аталған барлық мүмкіндіктер күрделі жүйелерді жасауда атарлықтай жеңілдік туғызады.

Cache-те деректердің біркелкі архитектурасының концепциясы жүзеге асқан, яғни сол баяғы Cache деректер көп өлшемді серверінің басқаруымен сақталған деректерге апарар үш мүмкіндік бар: тікелей, реляциялық және объектілі.

Деректерге апарар тікелей жол (Cache Direct Access) жоғары өндірушілік пен программисттер тарапынан толық бақылаумен қамтамасыз етеді. Қосымшаны өңдеушілер тікелей сақтау құрылымымен жұмыс жасау мүмкіндігіне ие болады. Бұл жолды қолдану өңдеушілер квалификациясына белгілі бір талаптар қояды, қосымша деректерін сақтауды оңтайландыруға және деректерді өңдейтін өте тез алгоритмді жасауға жол береді

Деректерге апарар реляциялық жол (Cache SQL) орнатылған SQL тілін қолдана отырып реляциялық қосымшаларды максималды түрде өндірушілікті қамтамасыз етеді. Cache SQL SQL 92 стандартына сай келеді. Бұдан басқа өңдеуші триггерлердің әртүрлі түрлері мен сақталатын рәсімдерін қолдана алады.

Деректерге апаратын объектілі және тікелей жолдарды қолданбай-ақ Cache-тегі қосымшалар деректердің көп өлшемді серверінің жоғары өндірушілігінің есебінде тез жұмыс жасайды.

Деректерге апарар объектілі жол (Cache Objects) Java, Visual C++, VB және басқа ActiveX-біріккен өңдеу құралдары, PowerBuilder және Delphi сияқты жобалаулардың объектілі-бағытталған тілдерін қолдану барысында жүзеге асады. Бұл үшін Cache-те еншілік (оның ішінде көптік), инкапсуляция және полиморфизмдер толықтай қамтылған деректер қорын басқаратын объекті модель жүзеге асқан. Өңдеушілер информациялық жүйені жасау барысында деректер (класс өзгешелігі) мен класс тәлімі (класс әдістері) сақталған объектілер класының жиынтығы түріндегі пәндік саланы модельдей отырып, өңдеулерге деген объектілі-бағытталған дүниелер ала алады.

Cache, деректердің объекті моделін қолдай отырып, пәндік аумақтарды жобалау кезінде обекті-бағытталған тәсілдерді табиғи түрде қолдануға қалай жол берсе өңдеу (Java, C++, Delphi, VB) құралдарымен қосымшаларды жүзеге асыруда да солай жол береді.

Объектілер класы анықтала салысымен, Cache автоматты түрде SQL қолдана отырып қарауға мүмкіндік беретін осы деректердің реляциялық сипаттамаларын түрлендіреді.

Дәл осы жолмен деректерді Сөздікке (деректердің реляциялық қорын сипаттау) импорттау барысында Cache автоматты түрде объектілерге сияқты деректерге де апарар жолды аша отырып деректердің объекті және реляциялық сипаттамаларын түрлендіреді. Осы сәтте деректердің барлық сипаттамалары келісілген түрде жүргізіледі, барлық операциялар түзетулер бойынша тек бір ғана деректің үлгісі арқылы өткізіледі.

Бұдан басқа бағдарламаушы сол деректерге тікелей жол арқылы шыға алады.

Cache — жобалау құралдарына және қосымша өңдеуге көп интерфейстер қолдау табатын ашық жүйе.

Cache іс жүзінде кеңінен таралған Web-серверлері бар барлық атаулы деген платформаларда жұмыс істейді. Осы тұста қосымшаның бір платформадан екінші платформаға толық ауысуы қарастырылған. Деректер қорын өңдеудегі заманауи объектілі-бағытталған технологиялардың осындай артықшыларына қарамастан өңдеушілерді реляциялық технологиядан объектілі-бағытталғанға ауысуға шешім қабылдауда біраз ойландыратын бірнеше бөгеттері де бар. Негізгі бөгет болып реляциялық ДҚБЖ-не сүйенетін өңдеулердің айтарлықтай үлкен көлемі болып табылады. Деректерді өңдеудің жаңа технологиясына ауысу барысында көп мәселелерді «нөлден» бастау қажет, сондықтан осындай ауысудың мақсаттылығы туралы сұрақ туындайды.

Бұдан бөлек постреляциялық ДҚБЖ қатарындағы объекті технологияның құрылымды болып келетін SQL сияқты дамыған және стандартталған тілі жоқ.

Cache жүйесінде аталған мәселелер реляциялық технологиядан объекті технологияға ауысу мүмкіндігімен қамтамасыз ететін рәсімдер негізінде шешілді.

15.2. Cache және WWW-технологиялар

Өңдеудің және қолға алынған CALS-технологиясының енгізген шарттарында Web-сервистер Cache ортасы ұсынатын негізгі мүмкіндіктердің бірі болып келеді.

Бәріміз білетіндей WWW жұмысының негізгі принципі барлық құжаттарды HTML бірыңғай форматында жасау, ал әртүрлі компьютерлік платформаларда қызмет істейтін сатып алушылардың қосымша браузерлері осы құжаттардың көбін бірдей етіп көрсетеді десе де болады. Бірақ шешімдерді құжаттарды өңдеудің статистикалық принциптері көрсетеді. Оларға динамика беру үшін сатып алушы жаққа да және сервер жаққа да қызмет ететін бағдарламалаудың әртүрлі құралдарын қолдану керек болды.

Егер өңдеуші сатып алушы жағында қызмет ететін скрипттік тілдерді таңдаса, онда олардың қызметтік аумағы көрсетілген құжатпен шектеліп қалады, ал кіретін деректер ретінде қолданушы әрекеті, немесе оның компьютерінде сақталған ақпарат қолданылады. Бұл шектеу скрипттік тілдердің құжат жүктелген және скрипттер қабылданған WWW-серверден белгілі бір қосымша ақпарат алмағанымен байланысты.

Егер сайтта деректерді басқарудың біршама күрделірек жолдарын қолдану талап етілсе, онда сервердің өзінде жұмыс істеген, орындалған модульдерді қолдану керек болады.

Тек осындай жолмен ғана деректер қорына кіруге жол ашуға және соңыра қашықтағы қолданушыға жіберілетін алынған ақпараттар негізінде парақшаларды қалыптастыруға болады. Тек серверде ғана жұмыс сеансын және электрондық коммерцияның барлық қосымшаларында қажет болып табылатын қолданушылар идентификациясын қолдауды қамтамасыз ету мүмкін болды.

Әрине, Web-жобалау технологияы аз болмады. Осы мақсат үшін C++ сияқты классикалық тілдер, Web-те арнайы бағытталған ASP сияқты технологиялар қолданылды. Дегенмен осылардың барлығының бір ортақ сипаты болды, ол – құжаттардың соңғы қорытындысы HTML форматында шығарылды.

Компьютерлік индустриядағы үдерістің жалпы жылдамдығын есепке алсақ, онда ғаламтор желісінде өте ұзақ қолданылған, көп өмір сүретін HTML жақын арада XML жаңа тілімен жаңартылуы мүмкін екенін мойындауымызға болады. Оның үстіне CALS негізіндегі халықаралық стандарттар құжаттарды форматтаудың осы тіліне бағытталып келеді.

Web-сервирстер және жекеленген сатып алушылар жүйесі тек өте күрделі функционалдылығы бар ресурстар үшін ғана қажет.

Жалпы жағдайда Web-сервистер информацияны XML тілінде қабылдайды және жібереді. Бұл тіл, әрине, ашық стандарт болып есептеледі. Сонымен қатар XML де алдыңғы HTML сияқты платформаға және операциялық жүйеге тәуелді емес. Осы екі фактордың үйлесімділігі өндеушілерге әртүрлі компьютерлік платформаларда пайдаланылатын сатып алушылардың әртүрлі қосымшаларын еркін түрде жасап шығаруларына мүмкіндік береді, яғни бір сервиске бірнеше сатып алушылардың қосымшалары сәйкес келуі мүмкін.

Серверден сатып алушыға және керісінше жіберілетін XML-құжаттар, Web- сервисінің идеяларына тартымдылық туғызатын, барлық брандмауэрлерде ашылатын HTTP протоколы бойынша жіберіледі.

Web-сервистер өздігінен құжатталатын сервис екенін ескере кету керек, яғни кез-келген сатып алушының қосымшасы сервис құрылымы туралы, оның функционалдылығы туралы ақпарат және Web-сервиспен қолданылатын шақыру функцияларының ережесін ала алады.

Web-сервистер барлық ақпаратты үш түрлі жолмен жібереді және қабылдай алады: HTTP стандартты хаттамасының GET және POST әдістерін қолдана отырып немесе XML-дан қалыптасқан SOAP тілінің көмегімен. Алғашқы екі нұсқа да стандартты браузерді сатып алушының қосымшасы ретінде қолдануға мүмкіндік береді, бірақ оның ең тиімді нұсқа емес екенін мойындау қажет.

Біріншіден браузердің көмегімен әжептәуір күрделі сервистің барлық функцияларын шақырту ұйымдастыру өте қиын. Өңдеуші сервиске кіруге арналған Web-парақшалар жасап шығару үшін оның құрылымымен ертерек танысып, зерттеуі керек. Сондықтан бұндай жағдайда біз өздігінен құжатталатын қабілетімізден айырыламыз. Екіншіден ақпараттар қорытындысы бәрібір де браузер адекватты түрде көрсете алмайтын «таза» XML форматында шығатынын ескеруіміз қажет. Сондықтан біршама салмақты функционалдылығы бар SOAP тілін қолданған ыңғайлы.

15.3. al Basic.NET — қосымшаларды құрастыру ортасы

Visual Basic.NET — Windows-қосымшаларды тез өңдейтін, объектілі-бағытталған концепциялар қорында толық функционалды профессионалды жобаларды жасап шығаратын заманауи визуалды орта. Visual Basic.NET-те қолданушы интерфейсін құрастыру тінтуірмен жасалатын жұмыс болып келеді. Кейде жоба шеберімен бірге жасаған кодтың бірнеше жолын ғана қосу керек болады. Visual Basic.NET — бұл біршама қарапайым, қатаң тұрпатталған тіл. Оның көмегімен объектілі-бағытталған конструкцияны жеңіл құрауға, графикалық қолданушылық интерфейс (GUI) жобалауға, деректер қорымен жұмыс істеуге болады.

Сондай-ақ Cache-пен біріктірілген Visual Basic.NET қамтамасыз етеді:

- бағдарламаның қолданушылық интерфейсін жасап шығару қарапайымдылығын;
- Web-сервистермен жұмыс істеу мүмкіндіктерін;
- жұмысты ғаламтор арқылы қоса отырып сатып алушы-серверлік қосымшаларды жасап шығаруды;
- SOAP-протоколының қолдауын.

15.4. SOAP — деректерді жіберудің көпплатформалы протоколы

SOAP-протокол — бұл алыстағы объектілердің әдістерін шақыра алатын, қосымшаларға арналған ережелер жиынтығы Бұл алыстағы объектілердің нақты қайда екендігі – басқа каталогта, корпоративтік интражелілерде немесе SOAP қолданатын сатып алушылар бағдарламаларына арналған ғаламтор желісінде, ол мүлдем маңызды емес. SOAP XML тілінің негізінде жасалған. Сатып алушы мен сервер арасындағы ақпарат алмасу SOAP ережесі бойынша жазылған бөлек XML-құжат болып табылады.

Бәрімізге белгілі, әрбір XML-құжаттың логикалық құрылымы оның DTD-блогымен анықталады.

SOAP үшін алдын-ала барлық мүмкін деген тег және деректердің түрі анықталып қояды, сондықтан SOAP-құжаттар DTD-блокқа тәуелді емес. Осындай серверлерді біріздендіру арқылы сатып алушылар белгісіз тегпен келген құжаттардың синтаксистік талдауы сияқты біршама күрделі рәсімдерден босайды.

SOAP хаттамасы — бұл хатқа бағытталып, жаһандық желі бойынша объектіні алыстағы шақыртуға негізделген әлсіз байланыстағы механизм.

Оның жұмыс принциптерін қарастырайық. URL біле тұра сатып алушы SDL (Service Description Language) сервисін сипаттау тілінде серверден ол туралы ақпарат сұрайды (келтірілген әдістер, параметрлер және т.б.). Жауап ретінде сатып алушы керекті ақпараттары бар және оған қандай әдістер ұсынылып оларлы қалай қолдану керек екендігін көрсететін ақпараты бар SDL-файл алады.

Әдістер HTTP-сұрақтар мен жауаптардың көмегімен ұсынылады. Сұранысқа үш бөлімнен тұратын XML-мәтін енгізіледі:

- хаттың қандай мазмұнда екенін анықтайтын SOAP envelope (пакет);
- хат тақырыпаттарын анықтайтын SOAP атауы;
- сұраныс және оған берілген жауап туралы ақпаратты қамтитын SOAP денесі.

Сұраныстың өңдеу қорытындыларынан немесе қатенің кодынан тұратын жауап та XML түрде келеді.

Cache жаңа нұсқасы — Cache 5 — платформасы барлары үшін SOAP хаттамасын жүзеге асырады. Осы орайда сырттағы компаниялардың аралық қосымшаларын қолдану міндеттелмейді, барлық қажетті функциялар Cache ДҚ жүйелік класы түрінде жүзеге асқан.

Cache SOAP функциялардың кең спектріне келесілер жатады:

- алшақтағы жүйелерге ұсынылған Web-әдістері бар, кластарды анықтау арқылы Web-қызмет жасап шығару. Осы тұста сатып алушы мен сервердің арасындағы трафикті қысқартуға бағытталған өндірушіліктің өсуіне әкеліп соғатын барлық әдістер ДҚ аумағына үздіксіз шақыртылып отырады;
- қолжетімді әдістердің автоматты түрде жасалуы мен каталогтың жарық көруі (WSDL) .

Web-қызметтер мен каталог баспаларын жасап шығару үшін Cache туралы ешқандай да қосымша білімінің қажеті жоқ. Қызметтерді Cache объектік моделінің негізінде жүзеге асырады. Осы тұста қызметтерді жасап шығару үшін алдымен %SOAP.WebService жүйелік класының ізін жалғаған класын анықтау керек. Сервердегі баспаларға жататын әдістер WebMethod параметрлерімен сипатталады. Әдістің логикасын жүзеге асыратын код Cache класының әдісінің коды сияқты

Cache Object Script, Cache Basic немесе Cache SQL қолдану арқылы жүзеге асуы мүмкін. Деректердің күрделі түрін (кірістірілетін объекттер, топтамалар, қарым-қатынастар және т.б.) жарияланған әдістің аргументтері және мәні ретінде қолдануға болады. Сонда Cache серверге қажетті сұраныс түскен кезде автоматты түрде деректердің күрделі түрлерінің қажетті объектілерін жасап шығарады.

Web-қызметке бір әдіспен мысал келтірейік:

```
Class MyApp.StockService Extends %SOAP.WebService
{
Parameter SERVICENAME = "MyStockService";
Parameter LOCATION = "http://localhost/csp/user";
Parameter NAMESPACE = "http://tempuri.org";
/// Әдіс ертеңге өз бағасын қайтарады
ClassMethod Forecast (StockName As %String) As %Integer [WebMethod]
{
// кездейсоқ сандар генераторын қолданамыз Set price = $Random(l000)
Quit price
```

Сонымен қатар, Cache автоматты түрде жасалған қызметтің дұрыс жұмысын тексеру үшін қолдануға болатын әдістердің тесттік CSP-парақшаларын және каталогтың CSP-парақшасын түрлендіреді. Осы тұста CSP- парақшалар қарапайым браузерлерге қарауға болатын әдеттегі Web-қосымшалар түрінде ұсынылады.

Cache SOAP Server келесі негізде жұмыс жасайды:

- әр Web Service-ке %S SOAP көмегімен ұлғаятын жаңа Cache-класс жасалады.WebService$;
- бұл класстың көмегімен Web Service әдісіне сай келетін бір немесе бірнеше әдістер анықталады. Олардың әрқайсысы «WebMethod» кілт сөзі анықтамасына қосылған кезде WebMethod ретінде анықталуы мүмкін. Сұранысты жариялап және «WebMethod» кілт сөзін анықтамасына қосу арқылы объектілердің жиымын қайтаратын Web-әдістерді анықтауға болады;
- Web Service класы құрастырылады, және Cache компиляторы автоматты түрде SOAP-сервисінің құрылымын сипаттайтын каталогты ақпаратқа кіріктіріп, әрбір Web-әдіске арнап SOAP-интерфейс құрайды. Web-әдіске арналған SOAP-интерфейс Cache XML-технологияны қолдана отырып SOAP сұранысын қайта жасайтын түрленген класс болып есептеледі;
- соңына Cache автоматты түрде әрбір Web-сервиске арнап WSDL-құжат жасап шығарады. WSDL-құжат бар әдістердің тізімін, сигнатураны және олар SOAP-сатып алушы көмегімен қалай шақыртылатынын көрсететін XML-құжат болып табылады.

WSDL-құжат CSP қолданатын Web (HTTP)-сервер көмегімен жасалады. WSDL-құжат динамикалық түрде жасалған және автоматты түрде Web Service-класының кез-келген өзгерісін көрсететін болады;

- Soap-сатып алушы өз кезегінде Cache- серверге сұраныс жіберетін WSDL-құжатты сұрай отырып бар Web-сервисті ашады. Бұл ақпаратты WSDL-құжатта қолдана отырып, SOAP-сатып алушы XML-хата жасап оны SOAP-серверге жіберу (HTTP) арқылы белгілі бір әдісті жүзеге асырады;
- Cache SOAP-сервер Cache (CSP) HTTP Gateway көмегімен SOAP шақыруын алады. Сервер хаттың орауын шешіп, оның дұрыстығын тексереді және белгілі бір Web-әдіс шақырады. Cache Web-әдісті шақырар алдында SOAP-сервер Cache-тің барлық анықтамасына сай келетін кірме параметрлерді айырбастайды;
- Web-әдіс өзінің кодын орындап жауап қайтарады. Бұл жауап тармақтық константа жағынан өте қарапайым болуы мүмкін немесе XML-объект не массив болуы да мүмкін.

Cache SOAP хаттамасын қолдана отырып, Web-сервис шақыратын SOAP client— класының пайда болу мүмкіндігін қамтамасыз етеді.

Cache —SOAP қолданушысы келесі жолмен жұмыс жасайды:

- сіз шақырғыңыз келетін әрбір Web-service үшін (SOAP байланысты әдістер жиынтығы), Cache кітапханасындағы %SOAP.WebClient-тен келген жаңа Cache класының жаңа анықтамасын жасап шығару керек;
- SOAP қолданушылар класы Web-service әдістеріне сәйкес келетін бір немесе бірнеше класс әдістерінен тұрады. Осы әдістердің әрқайсысы «WebMethod» кілт сөзі анықтамасына қосылған кезде WebMethod ретінде анықталған;
- SOAP сатып алушы класын құрастырғанда Cache класс трансляторы автоматты түрде каталогтағы ақпаратты, SOAP құрылымының анықтамасын жеткізіп, әрбір Web-service-ке арнап SOAP тұтынушы интерфейсін жасап шығарады;
- SOAP тұтынушы интерфейсі Web-service үшін — объектті XML форматына ауыстыру технологиясын қолдана отырып SOAP сұранысын қайта жасауды жүзеге асыратын түрленген класс;
- SOAP Client WSDL-құжат сұранысының көмегімен өз кезегінде бұны Cache серверінен сұрайтын Web-service тауып алады. Бұл ақпаратты WSDL-құжатында қолдана отырып, SOAP Client белгілі бір әдіс шақырады. XML-хат жасай отырып оны (HTTP арқылы) SOAP серверіне жібереді;
- SOAP сервері SOAP сұранысын алады және хаттың орауын шешіп, дұрыстығын тексеріп және көрсетілген операцияны шақырады.

Бақылау сұрақтары

1. Cache ДҚБЖ қағида тұрғысында реляциялық ДҚБЖ-нен немен ерекшеленеді?
2. Деректерге апарар тікелей жол жүйесі қалай іске асады?
3. Ғаламтор желісінде Cache-те ақпаратты жіберетін қандай протоколдар қолдайды?
4. Visual Basic.NET. қосымшасын өңдеу ортасының ерекшеліктерін атаңыз?

16 тарау

ЕРЕЖЕЛЕРГЕ НЕГІЗДЕЛГЕН ДЕРЕККОР ЖҮЙЕЛЕРІ

16.1. Дерекқор жүйесі

Дерекқорда ақпараттың үш түрі сақталады.

1. Қолданушы деректерінің құрылымын сипаттайтын ақпарат (дерекқор сұлбасының құрылымдық бөлігінің сипаттамасы). Реляциялық деректер қорына қатысты мұндай ақпарат жүйелік қатынастар каталогтарында сақталады және негізінен базалық қатынастардың атауларын, олардың атрибуттарының атаулары мен деректер түрлерін қамтиды.

2. Қолданушы анықтаған қарым-қатынастарда сақталатын қолданушы деректерінің жиынтықтары.

3. Дерекқор тұтастығының шектеулерін, дерекқордың триггерлерін және ұсынылатын (виртуалды) қатынастарды анықтайтын ережелер.

Реляциялық жүйелерде ереже сонымен қатар жүйелік каталог кестелерінде сақталады, бірақ тегіс кестелер бұл мақсат үшін қолайлы бола бермейді.

Бірінші және екінші түрлердің ақпараттары дерекқорда модельленген нақты әлемнің объектілерін (болмыстарды) анық сипаттайды. Басқаша айтқанда, бұл қолданушылар дерекқорда сақтауға ұсынатын нақты деректер. Дерекқордың бұл бөлігі экстенсионалды деп аталады.

Үшінші түр ақпараты қолданушылар беретін әр түрлі операцияларды орындау кезінде ДҚБЖ-ні басқару қызметін атқарады.

Тұтастық шектеулері дерекқорды жаңарту әрекеттерінің орындалуын бұғаттауы мүмкін, айрықша жағдайлар туындағанда триггерлер соған сәйкес айрықша іс-әрекеттерді орындайды, көріністердің анықтамалары ұсынылатын кестелерді қолдану барысында олардың анық немесе жанама материализациясын тудырады. Деректер қорының бұл бөлігін интенсионалды деп атайды; оның құрамында тікелей фактілер емес, пән саласының семантикасын сипаттайтын ақпарат болады.

Байқайтын болсақ, реляциялық деректер қорындағы ең маңыздысы - экстенсионалды бөлік, ал интенсионалды бөлігі негізінен қосалқы рөл атқарады.

Ережелерге негізделген дерекқор жүйелерінде бұл екі бөлік әдетте тең болады.

16.2. Белсенді дерекқорлар

Егер де ДҚ ДҚБЖ-сіне қатысты қолданушы айқын көрсететін іс-әрекеттерді ғана емес, сондай-ақ ДҚ-ның өзіне енгізілген ережелерге сәйкес қосымша әрекеттерді де орындайтын болса, ол белсенді деп аталады.

Бұл идеяның негізі SQL тілінің құрамында қамтылды. Шынында да, бақылаушы жүйе қосымша әрекеттерді жасауы тиіс ДҚ-ға ереже енгізу сияқты, триггер немесе шартты әсерді анықтау дегеніміз не? Жалғыз жетіспеушілік, шын мәнінде, триггерлер белгілі жүйелерде толық іске асырылмаған. Және бұл кездейсоқтық емес, өйткені мұндай құрылғыны ДҚБЖ-да жүзеге асыру өте күрделі, қымбат және толық түсінілмейді.

Жауаптары әлі де белгісіз сұраныстардың ішінде төмендегілер кездеседі:

Тікелей қолданушы әрекетінен туындаған көмекші әрекеттердің жиынтығын тиімді анықтаудың жолы қандай?

Әрекет-шарт-әрекет -... тізбегіндегі циклдарды қалай тануға болады және осындай циклдар орын алғанда не істеу керек?

Қосымша шартты әрекеттер қандай транзакция аясында орындалады және туындаған үстеме шығыстарды қай қолданушының бюджетіне жатқызу керек?

Көптеген мәселелер SQL триггерлерін іске асырудың салыстырмалы қарапайым жағдайында да шешілмеген және тапсырма әлдеқайда кеңейіп келеді. ДҚБЖ-сінің бір бөлігі ретінде, шарттар мен әрекеттер ДҚ мазмұны немесе онымен тікелей қолданушы әрекеттерімен шектелмейтін жалпы өндірістік жүйе ретінде ұсынылады. Шарт күннің уақытын қамтуы мүмкін және әрекет сырттай орындалуы мүмкін, мысалы, оператор экранындағы ақпараттың шығуы. Белсенді ДҚ-дағы қазіргі заманғы барлық дерлік жұмыстар осындай өндірістік жүйені тиімді енгізу мәселесіне байланысты.

Іс жүзіндегі мақсаттар үшін ДҚБЖ-да құрылғының триггерлерін іске асыру әлдеқайда маңызды. Стандартты SQL3 жобасы шартты әсерлерді анықтайтын тіл құралдарының болуын қамтамасыз етеді. Оларды іске асыру белсенді ДҚ үшін бірінші практикалық қадам болады (тиісті коммерциялық енгізулер пайда болды).

16.3. Дедуктивті дерекқорлар

Дедуктивті ДҚ екі бөліктен құралады: фактілерді қамтитын экстенционалды, және экстенционалды бөлік пен қолданушы сұранысы негізінде жаңа фактілерді логикалық түрде шығаратын ережелерді қамтитын интенционалды бөлік.

Бұл жалпы анықтамамен SQL-ге бағытталған реляциялық ДҚБЖ-сін дедуктивтік жүйелерге жатқызуға болады. Шындығында, ДҚ-дың интенционалды бөлігінен басқа релациялық ДҚ сұлбасында анықталған көріністер жоқ. Қолданыстағы жаңа фактілерді алу үшін қандай нақты механизм қолданылғаны маңызды емес. SQL жағдайында, көріністің анықтамасының негізгі элементі - SQL таңдамалы таңдау операторы, ол өте табиғи, өйткені үлгілеу операторының нәтижесі жасалатын кесте болып табылады. Қажетті кеңейтілімділік де ұсынылады, өйткені ұсыныстар тек базалық кестелерден ғана емес, сонымен қатар көріністерден де анықтала алады.

Нақты дедуктивті ДҚБЖ-нің реляциялық ДҚБЖ-сінен басты айырмашылығы – ДҚ-ның интенционалды бөлігінің ережелерінде де, қолданушылар сұраныстарында да рекурсия болуы мүмкін. Рекурсияның болғаны әрдайым жақсы ма деген мәселені тереңірек талқылауға болады. Бір жағынан, рекурсивтік ережелер мен сұраныстарды анықтай білу мүмкіндігі дедуктивті дерекқорлардағы мәселерлерді шешуге мүмкіндік береді, себебі олар реляциялық жүйелерде үлкен мәселелерді тудырады (мысалы, күрделі бөліктерді қарабайыр компоненттерге бөлшектеу мәселесі). Екінші жағынан, рекурсия жасау мүмкіндігі дедуктивті ДҚБЖ-нің жүзеге асырылуын күрделендіріп, көп жағдайда оны шешілмейтін мәселеге айналдырады.

Біз дедуктивті жүйелердегі нақты мәселелерді, қоланылатын шектеулер мен пайдаланылатын әдістерді егжей-тегжейлі қарастырмаймыз. Тек, әдетте сұраныс тілдері мен ДҚ интенционалды бөлігінің анықтамалары логикалық болатындығын атап өтеміз (сондықтан, дедуктивті ДҚ логикалық деп аталады). Дедуктивті ДҚ мен білім қорлары арасында тікелей байланыс бар (ДҚ интенционалды бөлігін білім қоры (БҚ) ретінде қарастыруға да болады). Сонымен қатар, бұл екі болмыс арасындағы шекара нақтылауаса қиын; кем дегенде, бұл мәселе бойынша әлі күнге дейін жалпы қалыптасқан жоқ.

Реляциялық ДҚ дедуктивтінің туа біткен жеке жағдайы дегеннен басқа, реляциялық ДҚБЖ мен дедуктивті ДҚ арасындағы байланыс қандай? Негізгі байланыс – дедуктивті ДҚБЖ-ні жүзеге асыру үшін әдетте реляциялық жүйенің қолданылуы. Мұндай жүйе дедуктивті ДҚБЖ деңгейінен келіп түсетін фактілерді қорғаушы және сұраныстарды орындаушы рөлін атқарады. Реляциялық басқару жүйелерін пайдалану жаһандық сұранысты оңтайландыру міндетін күрт жаңартады.

Реляциялық ДҚБЖ-сін әдеттегідей қолданғанда, сұраныстар өңдеуге бір-бірден түседі, сондықтан оларды ғалами (сұранысаралық) оңтайландыруға ешқандай себеп жоқ. Ал дедуктивті ДҚБЖ болса, қолданушының бір сұранысты орындау кезінде сұраныстар жинағын бірге оңтайландырылатын реляциялық ДҚБЖ-сіне генерациялайды.

Әрине, дедуктивті ДҚ ережелерінің жиынтығы ауқымды болып, оларды жедел жадқа орналастыру мүмкін болмаса, олардың сақталуын реттеу мәселесі туындап, оларды сыртқы жадта қолдану қиындайды. Сонымен қатар, мұндай жағдайда реляциялық жүйе қолданылуы мүмкін, бірақ ол аса тиімді емес. Ол үшін күрделі деректер құрылымдары және басқа да іріктеу шарттары қажет. Бұл мәселені шешуге арналған тек жекелеген әрекеттер белгілі, алайда жалпы шешім әлі де табылған жоқ.

Бақылау сұрақтары

1. Ережелерге негізделген дерекқор кестелерінің құрылымдары дәстүрлі – реляциялық ДҚ-дан несімен ерекшеленеді?
2. Белсенді және дедуктивті дерекқорлардың басты сипаттамаларын атаңыз.

V БӨЛІМ

ӨНДІРІС ПЕН КӘСІПКЕРЛІКТЕ ДҚБЖ ҚОЛДАНЫЛУЫНЫҢ ТӘЖІРИБЕЛІК МЫСАЛДАРЫ

17 Тарау

ӨНІМНІҢ ӨМІРШЕҢДІК КЕЗЕҢІН БАСҚАРУ ЖҮЙЕЛЕРІ

17.1. Кәсіпорынның интеграцияланған ақпараттық ортасы

4.1 бөлімінде біз заманауи өндіріс және кәсіпкерлік жағдайындағы ақпараттық жүйелерді дамытудың жаңа бағыты – CALS-технологиясы жайлы және, соған байланысты кәсіпорындарда бірнеше қолданушылық, үлестірілген ДҚ-ларды ұйымдастыруға өту қажеттілігін қарастырдық. Енді бірыңғай ақпараттық кеңістікті кез-келген кәсіпорында құру бойынша жұмыстарды ұйымдастырудың негізгі бағыттарын қарастырайық.

Осы жұмыс саласының тұжырымдамалық үлгісінен байқағанымыздай, CALS-технологияларының мәйегі, осы негізде құрылған автоматтандырылған жүйелердің интеграцияланған ақпараттық ортасы болып табылады.

ИАО идеясы CALS-технологиялары пайда болғанға дейін ұзақ уақыт бұрын ғылыми айналымға енгізілген. 1983 жылдың өзінде, жапон ғалымы N.Okino былай деп пайымдады: материалдық объектілер және солармен қатар жүретін жобалау процестері, технологиялық дайындық пен басқарудың адамның өзге іс-әрекеттері мен қызметінен ерекшеленетіні соншалық, оларға айрықша бағдарламалық-әдістемелік, математикалық және ақпараттық қамсыздандыру архитектурасыжауап беруі керек деп санады. Оның пікірінше, іргелі өндіріс жүйесін және басқа да қосымшаларда ақпаратты өңдеу арасындағы айырмашылық, компьютерлік технологияда негізінен екі жағдайғакеліп тіреледі.

1. Өндіріс және ондағы барлық процестер физикалық әлемге тиесілі және компьютердегі процестер ақпарат әлеміне жатады. Демек, өндіріс мәселелерін ақпаратқа айналдыру, сондай-ақ ақпараттық әлемнен физикалық түрде кері өту қажеттілігі туындайды. Бұл дұрыс модельдеу мәселесі, яғни, физикалық және ақпараттық кеңістік арасында хат алмасуды (мүмкін болса, жеке-жеке) құруды білдіреді.

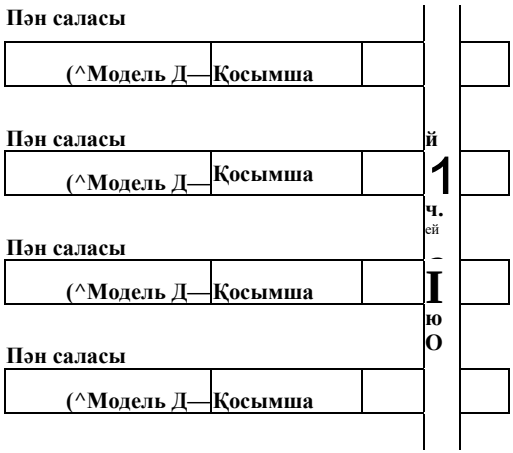
Есептеу мәселелерін шешу үшін дәстүрлі математикалық қамтамасыз етуді (МҚ) құрған кезде, мәселені шешудің жалғыз математикалық моделі даму орталықтарына енгізілді, ол қолданба интерфейсі арқылы әр түрлі қолдану салаларына бейімделеді (17.1 сур.).

Өндірістік мәселелерді шешу барысында бұл тәсіл іс жүзінде іске асырылмайды, себебі олардың күрделілігі мен алуан түрлілігіне байланысты біртұтас модель құрылмайды.

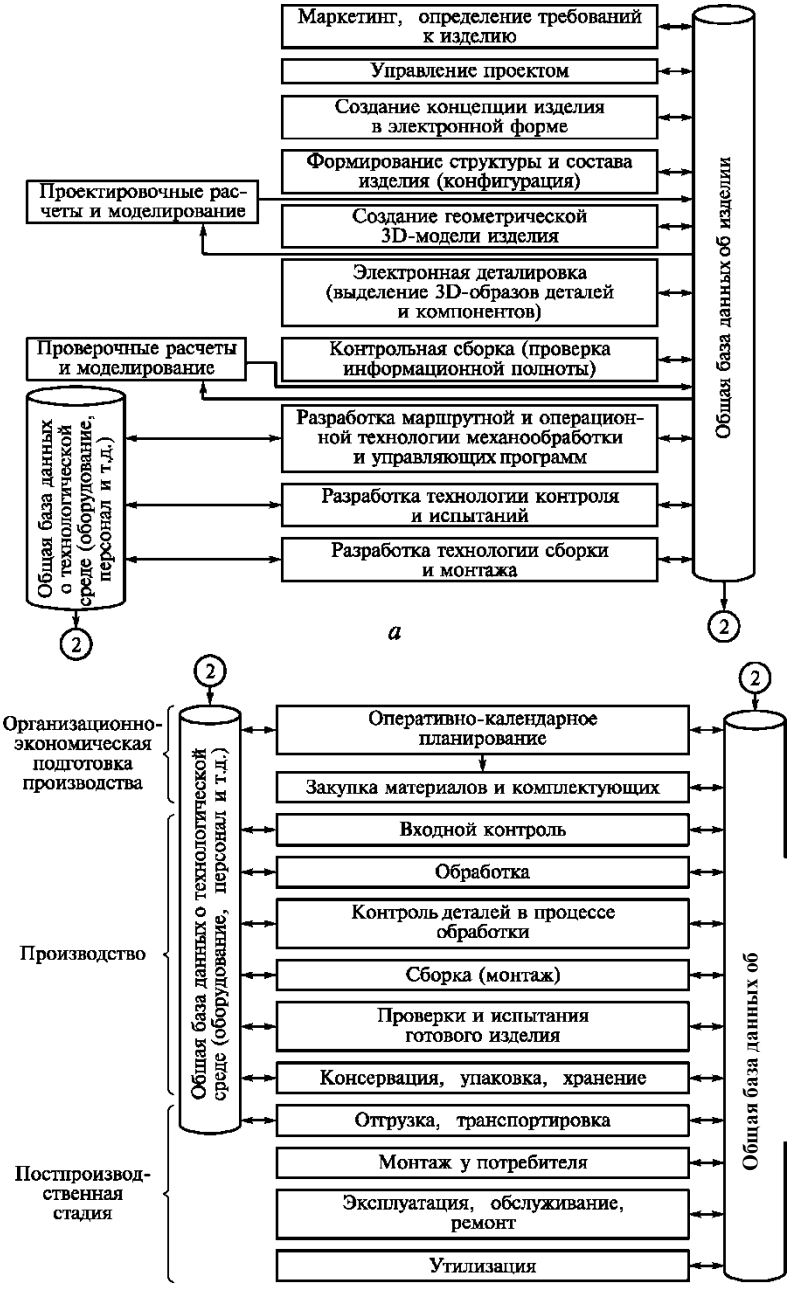
Егер Н. Окино зерттеген өндіріс мәселелеріне қосымша, өнімдерді жеткізу, пайдалану, жөндеу және жөндеу мәселелері қосылса болса, яғни өнімнің өміршеңдіккезеңінің (ӨК) өндірістен кейінгі кезеңдері қамтылса, жағдай одан сайын күрделене түседі.

2. Дәстүрлі тәсілдің жоғарыда көрсетілген кемшіліктерін (17.1 сур. қараңыз) есепке ала отырып, бірыңғай дерекқорын ұйымдастыру стратегиясын шегеріп, мәні 17.2 суретінде көрсетілген стратегияға көшу ұсынылады.

Мұнда жүйе мәйегінің рөлін модель емес, жалпы (интеграцияланған) дерекқоры (ЖДҚ) атқарады, оған бағдарламалық қосымшалар формасында жүзеге асырылған әр түрлі мәселелік-бағытталған модельдер жатқызылады. ЖДҚ-да ақпараттық әлемде физикалық әлемнің, объектілердің, материалдардың, өнімдердің, процестер және технологиялардың мәнін, құжаттардың түрлі, қаржы ресурстарын, кадр бөлімі мен өндіруші кәсіпорынның жабдықтарын, тапсырыс берушінің өнімді пайдалануынажасалатынжағдай, қызмет көрсету және жөндеу қызметтері және т.б. бейнелейтін ақпараттық объектілер қамтылады.



17.1- сур. Жалпы дерекқорларды жергілікті дерекқорлар негізінде ұйымдастыру стратегиясы



17.2 сур. Жалпы дерекқорларды бірнеше қолданушылық *(а)* және үлестірілген *(б)* дерекқорлар негізінде ұйымдастыру стратегиясы

Бір жағынан, Н. Окино объективті бағдарлы көзқарастың пайда болуын күтіп, объективті операциялық дуализм негізінде ақпараттық әлемде болып жатқан барлық нәрселерді қарауды ұсынды.

Н. Окино әзірлеген идеялардың мәні мынадай:

Физикалық әлемнің кез-келген субъектісі деректердің белгілі бір жиынтығы болып табылатын ақпараттық объектіге сәйкес келеді. Жеке тұлғаның кез-келген түрін пайдалану, оны басқа ұйымға (немесе сол субъектіге, бірақ параметрлердің әртүрлі мәндерінде) қайта өңдеу – өңдеу, өндіру, өлшеу, жобалау – ақпараттық әлемдегі операциялар (топ, бағдарлама және т.б.) арқылы ұсынылған.

Ақпараттық технологияларды одан әрі дамыту объектілі-бағытталғантәсілдің пайда болуына әкелді, бұл кәсіпорында орын алған көптеген процестерді виртуалды ақпараттық кеңістікте аударуға мүмкіндік бере отырып, CALS-технологияларын қолдануға байланысты мәселелерді тудырды.

Жоғарыда айтылғандар, әсіресе, әртүрлі ақпарат пен кең көлемді ақпаратпен айналысуға тура келетін барлық деңгейдегі өндіріс пен кәсіпкерлікті басқару процестеріне әртүрлі типтегі және мақсаттардағы техникалық құжаттама жасалатын өндірісті жобалау және технологиялық дайындау процестеріне қатысты. Бүгінде бұл процестер негізінен интеграцияланған ақпараттық ортада ақпараттық объектілерді құру, трансформациялау, тасымалдау және сақтау бойынша операциялардан тұрады.

17.2. Кәсіпорынның интеграцияланған ақпараттық ортасының құрылымы мен құрамы

ИАО – бұл өндірістік қызметтің барысында өнімнің өміршеңдік кезеңінің қатысушылары – кәсіпорынның барлық бөлімшелері мен қызметтерінде пайдаланылатын барлық ақпаратты қамтитын деректер қоймасы. Бұл қойма сыртқы және ішкі сілтемелерді көрсететін күрделі құрылымға ие. ИАО кем дегенде екі негізгі дерекқорды қамтуы керек:

- Бұйым (бұйымдар) туралы жалпы деректер қоры (БЖДҚ);
- Кәсіпорын туралы жалпы деректер қоры (КЖДҚ).

17.2 суретінен ИАО құрылымы өнімнің бүкіл өмірлік циклінде орын алатын процестермен өзара әрекеттесетінің, бұл процестерде ИАО-да қамтылған ақпарат пайдаланылғанын көруге болады.

Жоғарыда аталған қосымшалар бірыңғай дерекқорға жүгінеді, онда қажетті ақпараттық объектілерді табады, процестер кезінде жасалатын нысандар ИАО-ға сақтау және кейіннен басқа процестерде пайдалану үшін қайтарылады.

Өнімнің жалпы деректер қорымен процестер оның өміршеңдік кезеңінің барлық кезеңдерінде біріктіріледі. Кәсіпорынның жалпы деректер қоры өндірістің технологиялық және ұйымдастырушылық экономикалық дайындығына және нақты өндіріспен (дайын өнімдерді жөнелту және тасымалдау процестерін қоса алғанда) байланысты.

Жаңа бұйымды жасағанда және оны АЖЖ-сінің (CAE/CAD/CAM) конструкторлық және технологиялық құралдарының көмегімен технологиялық тұрғыда өндіріске даярлау кезінде ИАО-да бұйымның құрылымын, оның құрамы мен барлық құрамдас бөліктерін: бөлшектер, түйіндер, агрегаттар, құрамдас бөлшектер, материалдар және т.б. сипаттайтын ақпараттық объектілер құрылады. Әр ақпараттық объект физикалық объект ерекшеліктерін сипаттайтын атрибуттарды қамтиды, оларға келесілер жатады: техникалық талаптар мен шарттар, геометриялық (өлшемдік) параметрлік, массагабариттік көрсеткіштер, төзімділік, беріктік және ресурс сипаттамалары және бұйым мен оның компоненттерінің өзге де сипаттамалары.

Жалпы деректер қорындағы ақпараттық объектілер кәсіпорын шығаратын барлық өнімдер үшін өндірістік процестің барлық кезеңдерінде қажетті техникалық құжаттаманы беруге және сақтауға қажетті ақпаратты еркін форматта қамтиды. Әрбір ақпараттық объект бірегей кодпен анықталады және онымен әрекеттерді орындау үшін дерекқордан еркін алынуы мүмкін. Өнімнің жалпы мәліметтер қоры ақпараттық қызметтер мен қолдау қызметтерін ұсынады:

- өнімге тапсырыс берушілер (иелері);
- өндірушілердің (құрастырушылардың), технологтардың, өндіруші кәсіпорынның басқарушы және өндірістік қызметкерлерінің;
- тапсырыс беруші мен мамандандырылған қызметтердің пайдалану және жөндеу қызметкерлерінің.

БЖДҚ құрамына енетін АО құрамы туралы толығырақ 17.3 суретінде көрсетілген, оған сәйкес БЖДҚ құрамындағы үш бөлімді шартты түрде бөліп шығаруға болады:

- нормативтік-анықтамалық;
- ұзақ мерзімді;
- өзекті.

БЖДҚ *нормативтік-анықтамалық бөлімінің* мазмұны жаңартылып, жаңа және қабылданған нормативтік құжаттар қабылданады.

Ұзақ мерзімді бөлімде компанияның өз тәжірибесін жинақтайтын деректерді қамтитын АО сақталуы керек.

БЖДҚ ұзақ мерзімді бөлімі жаңа техникалық шешімдер ретінде қабылданады және оларды болашақта пайдалану үшін жарамды болып табылады.

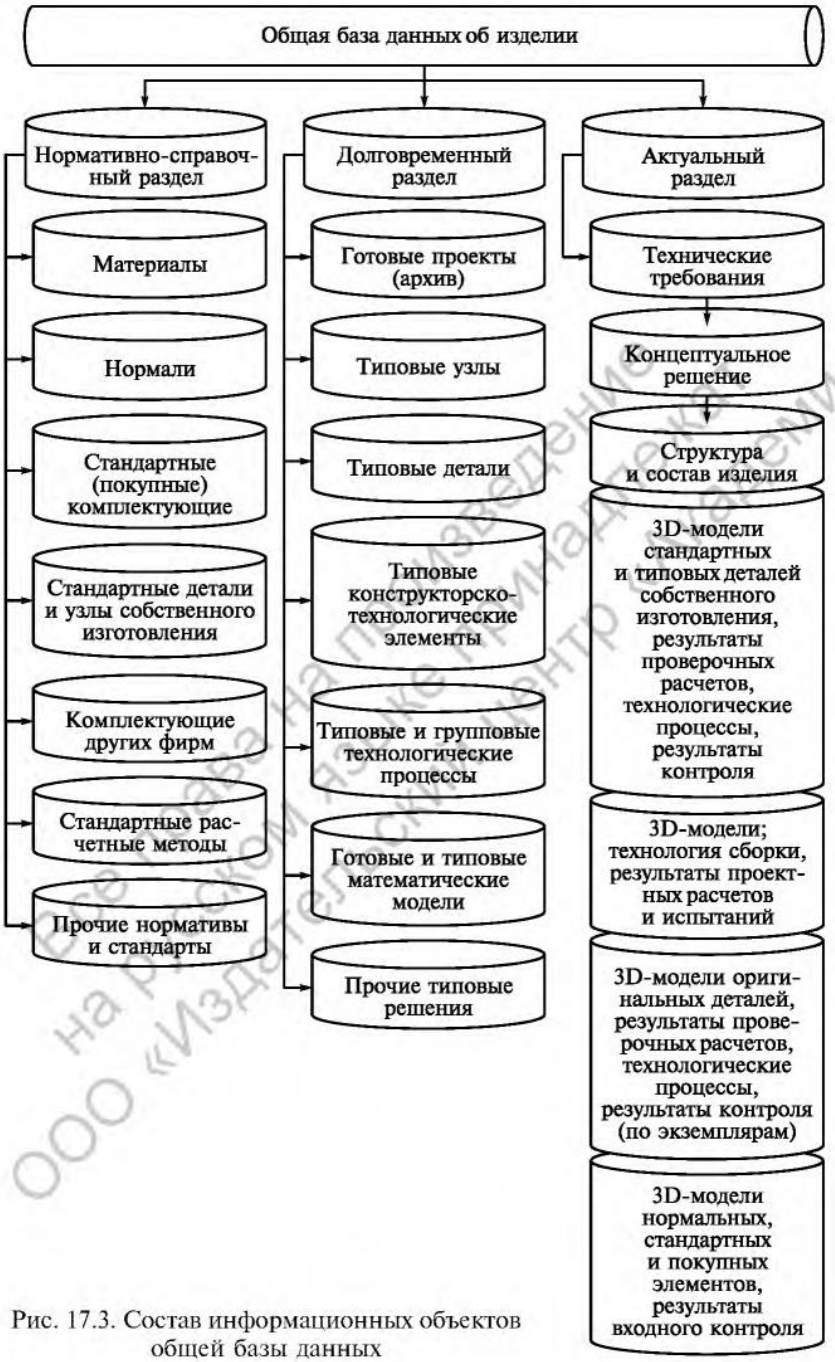


Рис. 17.3. Состав информационных объектов общей базы данных

17.3 сур. Жалпы дерекқорының ақпараттық объектілерінің құрамы

Өзекті бөлімде (мүмкін, көлемі бойынша ең үлкен және ең күрделі құрылым), өміршеңдікциклдің түрлі кезеңдеріндегі өнімдер туралы деректерді қамтитын АО сақталуы керек.

Бұл бөлімнің құрылымы өте ұқсас және қосымша ішкі бөлімдерге (классификация деңгейлері) ауытқуларды қоса алғанда, өңдеу мен нақтылауды талап етеді.

Жоғарыда айтылғандай, өнімге қатысты (тікелей немесе жанама) ақпараттық объектілерден басқа, кәсіпорын туралы ақпарат ақпараттық жүйеге кіреді: өндіріс және басқару құрылымы туралы, технологиялық және қосалқы жабдықтар, қызметкерлер, қаржы және т.б. Бұл деректердің бүкіл жиынтығы бірнеше секциядан тұрады: экономика және қаржы, кәсіпорынның сыртқы байланыстары, кәсіпорынның өндірістік және технологиялық ортасы, сапа жүйесі.

Экономика және қаржы бөлімінде ақпаратты қамтитын ақпараттық объектілер сақталуы керек:

- кәсіпорын бағасының конъюнктурасы, оның ішінде бағалары мен олардың қарқыны;
- кәсіпорынның қаржы ресурстарының жағдайы;
- қор және қаржы нарықтарындағы жағдайы (компания акцияларының бағасы, акциялардың индексі, пайыздық мөлшерлемелер, айырбастау бағамы және т.б.);
- нақты және болжамды тапсырыс кітапшасы;
- сондай-ақ басқа қаржылық, экономикалық және бухгалтерлік ақпарат.

Кәсіпорынның сыртқы байланыстарына арналған бөлімде АО нақты және ықтимал жеткізушілер мен тұтынушылар (тапсырыс берушілер) туралы ақпаратты қамтитын АО сақталуы тиіс. Бұл бөлім маркетингтік зерттеулер барысында құрылады және пайдаланылады.

Кәсіпорынның өндірістік-технологиялық ортасына арналған бөлімде төмендегідей мәліметтерді қамтитын ақпараттық объектілер сақталуы керек:

- кәсіпорынның өндірістік құрылымы туралы;
- технологиялық, қосалқы және бақылау жабдықтары;
- кәсіпорынның көлік-қойма жүйесі;
- кәсіпорынның энергиямен жарақтандырылуы;
- кадрлар;
- сондай-ақ кәсіпорын туралы басқа да деректер.

Сапа жүйесіне арналған бөлімде төмендегідей мәліметтер сақталуы керек:

- кәсіпорындағы сапа жүйесінің құрылымы туралы;
- кәсіпорындағы ағымдағы сапа стандарттары туралы;
- халықаралық және ресейлік сапа стандарттары туралы;
- сапа саласындағы қызметтік сипаттама туралы;
- сондай-ақ сапа жүйесі туралы басқа да ақпараттар.

ИАО-дан кәсіпорынның жұмыс істеуі үшін қажетті көптеген құжаттар алынуы мүмкін. Құжаттар электронды және дәстүрлі, қағаз түрінде ұсынылуы мүмкін.

CAE/ CAD/CAM жүйелерін электронды түрде жобалағанда, БАҚ 10303-203 стандарты талаптарына сай, конфигурацияны басқарудың электрондық құралдары қолданылуы керек.

17.3. Кәсіпорынның интеграцияланған ақпараттық ортасын басқару

Кәсіпорынның біріктірілген ақпараттық ортасын басқару ИАО-дағы барлық процестер басқарылатын, яғни уәкілетті тұлғалардың (әкімшілердің) және тиісті бағдарламалық қамтамасыз етудің әсеріне ұшырайтындығын болжайды. Осындай құралдардың жиынтығы әдетте деректер қорын басқару жүйесі деп аталады. ДҚБЖ функциясы мыналарды қамтиды:

- ақпаратты дерекқорға енгізу;
- резервтік көшірумен бірге ақпаратты сақтау;
- деректерді жаңарту (өзектілігін жоғалтқан деректердің орнына жаңа деректерді енгізу);
- деректердің сенімділігі мен тұтастығын қамтамасыз ету;
- әртүрлі негізде деректерді іздеу;
- есептер жасау;
- пайдаланушының деректерге қатынау құқықтарын белгілеу (өзгерту) және оперативті түрде тексеру және т.б.

ИАО-ның дәстүрлі деректер қорынан айырмашылығы, өнімдердің өміршеңдік кезеңінің барлық мүдделі қатысушылары арасында деректерді жинақтауды, сақтауды және беруді қамтамасыз ететін арнайы инфрақұрылымды құруды талап етеді. Мұндай инфрақұрылым жоғарыда аталған міндеттерді шешуге мүмкіндік беретін бағдарламалық және аппараттық кешен сипатында болуы керек.

Бірыңғай (және жалғыз) өндіріс орнында орналасқан дәстүрлі кәсіпорынның аясында осындай инфрақұрылым жергілікті компьютерлік желі және тиісті жүйе мен қолданбалы бағдарламалық қамтамасыз ету негізінде құрылады.

Географиялық бөлінген өндірістік құрылымы бар кәсіпорындар үшін, әсіресе, виртуалды кәсіпорындар үшін бұл мәселе аса маңызды.

Телекоммуникация құралдарының және жүйелерінің қазіргі жай-күйін талдау виртуалды кәсіпорын инфрақұрылымының негізін, сондай-ақ географиялық жағынан таралған құрылымы бар кәсіпорын деректерінің TCP / IP хаттамасының көмегімен берілетін ғаламдық Интернет желісі бола алатынын айтуға толық негіз бар.

Интернеттің сыртқы қарапайымдылығы мен қолжетімділігіне қарамастан, оны құрылымдау құралы ретінде қолдану бірқатар нақты мәселелерге тіреледі.

Бірінші мәселе, БАҚ технологиясына сәйкес ақпарат алмасудың барлық қатысушыларының деректерін тиімді жинақтау, сақтау және пайдалану үшін, дерекқор логикалық түрде портал ретінде Интернет технологиясында жиі аталатын объект бойынша логикалық түрде локализациялануы керек. Басқаша айтқанда, кәсіпорынның, виртуалды кәсіпорынның немесе кәсіпорынның ақпараттық қызметі үшін арнайы Интернет түйінін жасау керек.

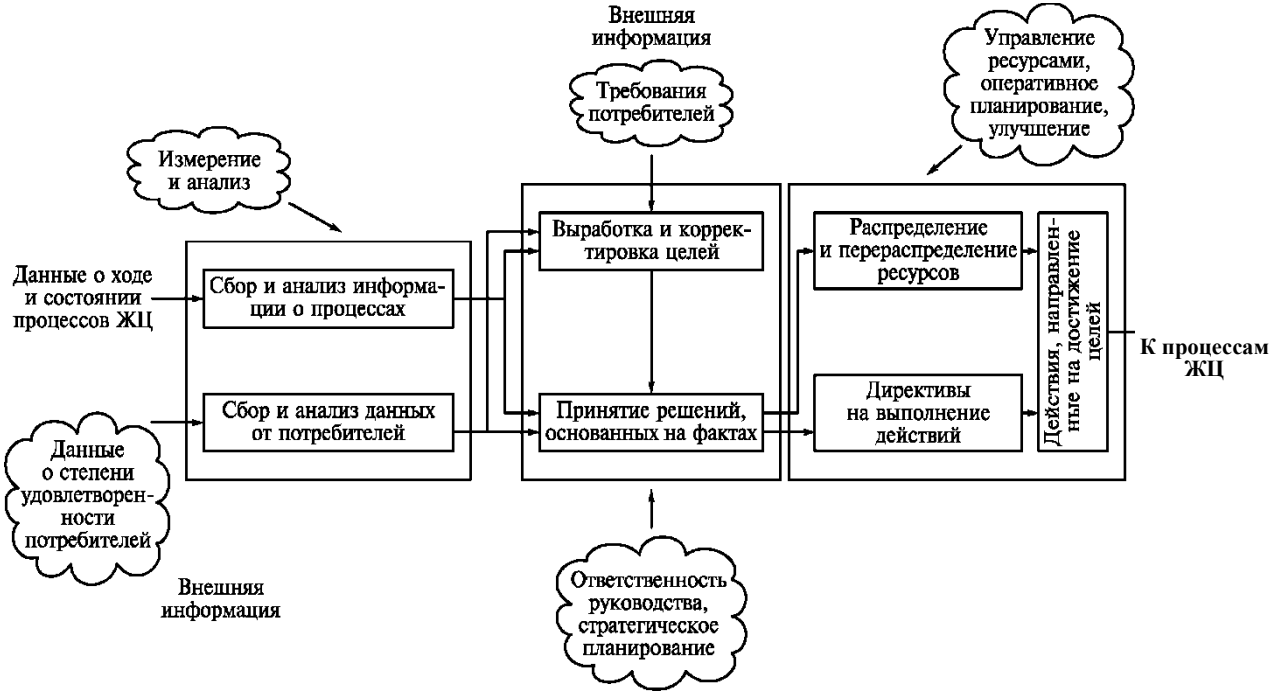
Екінші мәселе бұл түйін және сәйкесінше ақпарат алмасу қатысушылары бұл алмасуға өзге бөгде тұлғалар мен ұйымдардың араласуынан қорғалуы керек, тіпті ол тұлғалар мен ұйымдардың жаман ойы мен қастық мүдделері болмаса да.

Үшінші мәселе – бұл ақпараттың дұшпандық мақсаттарда (мемлекеттік және (немесе) коммерциялық құпияны құрайтын мәліметтерді ұрлау мақсатында) қатысушылар мен тұлғалардың жіберген деректерінің тұтастығын және (немесе) сенімділігін бұзу мақсатында пайдалану құқығынсыз жеке тұлғалардың және ұйымдардың ақпарат алмасуын қорғауға негізделеді.

17.4. Сапаны басқару

Өнімнің сапасын басқару жүйесі кәсіпорынның басқару қызметінің элементі болып табылады. ISO 9000: 2000 халықаралық стандартына сәйкес сапа менеджменті (СБ) өнімнің автоматтандырылған өңдеуіне және өнімнің өміршеңдік кезеңдерінің барлығында сапаны қамтамасыз ету рәсімдерінің құжаттамасына, осы процестерді, деректер мен құжаттаманы автоматтандырылған басқаруды қолдайтын ақпараттық жүйеге негізделуі керек. Бұл тұрғыда сапа менеджменті кәсіпорынының АБЖ-сінің ажырамас бөлігі болып табылады және ол CALS-технологияларына жатқызылуы мүмкін. Бұл сапа менеджменті жүйесі туралы ақпарат CALS-технологияларының стандарттарымен реттелетін форматтарда ұсынылуы керек және кәсіпорнының ИАО-сына кіретін ақпараттық объектілердің жиынтығынан тұруы керек дегенді білдіреді.

Сапаны басқару жүйесінің кеңейтілген құрылымы 17.4 суретінде көрсетілген, онда жүйенің бақылау объектісімен (өнімнің ӨК процестерін), сондай-ақ қарастырылып отырған жүйеге сыртқы ортамен байланысын көрсетеді.



17.4 сур. Сапаны басқару жүйесінің құрылымы

Құрылымдағы фактілерге негізделген мақсаттар мен шешімдерді әзірлеу және түзету блоктары ИАО 9000: 2000 стандарты бойынша басқару мен жоспарлау жауапкершілігі (осы тұрғыда стратегиялық) деп аталатын стандартқа ортақ болып табылады. Тұтынушылардан деректерді жинау және талдау, процестер туралы ақпаратты блокта өлшеу және талдау ретінде стандартта айтылған процестер бейнеленеді. Шешімдерді орындау (қорларды бөлу және қайта бөлу, мақсаттарға қол жеткізуге бағытталған іс-әрекеттер мен әрекеттер жөніндегі нұсқаулықтар) байланысты блоктар тобы ресурстарды басқару, жоспарлау (осы тұрғыда, операциялық) және жетілдіру деп аталатын барлық сатынықамтиды.

Жаңа бұйымды конструкторлық және технологиялық АЖЖ (CAD/CAM) құралдарымен жасау және өндіріс технологияларын даярлау барысында аталып өткендей, ИАО-да да АО құрылады, олар бұйымның құрылымын, оның құрамы мен барлық компоненттерін: бөлшектер, түйіндер, агрегаттар, құрамдас бөліктер, материалдар және т.б. бейнейлейді. әр АО объектінің нақты ерекшеліктерін сипаттайтын сипаттамалар (атрибуттар) жиынтығы болып табылады. Мұндай сипаттамаларға нақты объектінің қанағаттандыруы тиіс техникалық талаптар мен техникалық шарттар жатады. Бұйым туралы ақпараттан бөлек ИАО-да кәсіпорынның өндірістік ортасы жөніндегі ақпарат та қамтылады.

17.5. Жұмыс ағынын басқару

«Жұмыс ағыны» ұғымы – БАҚ (бұйымдарды ақпараттық қолдау) саласындағыжәне жалпы алғанда қазіргі заманғы басқару тәжірибесіндегі негізгі тұжырымдамалардың бірі. Бұл ұғым кәсіпорынның бизнес-процестерін формализациялау мен басқару тәсілдерін, сондай-ақ осы тәсілдерді іске асыратын бағдарламалық қамтамасыз етуді біріктіреді. Жұмыс технологиясын енгізудің танымалдылығы мен алуан түрлілігі 1990-шы жылдардың ортасында пайда болды. Жұмыс процесі жүйелеріне қойылатын талаптарды реттейтін стандарттарға және оларды жүзеге асыру құралдарына қатысты Wf MC (Workflow Management Coalition) халықаралық ассоциациясының (WfMC) құрылуына алып келді.

WfMC сөздігіне сүйенсек, бизнес-процесс – бизнес мақсатын немесе кәсіпорынның саяси мақсатын бірлесіп жүзеге асыратын бір немесе бірнеше өзара байланысты рәсімдер немесе функциялар (әдетте функционалдық рөлдер мен қарым-қатынастарды сипаттайтын ұйымдық шеңберде) болып табылады.

Ұйым ішіндегі немесе бірнеше әр түрлі ұйымдар ішіндегі бөлімшелердің бизнес-процесі, мысалы, клиент-жеткізуші қатынас жүйесінде. Бизнес-процесс қатысушылар арасында ресми және бейресми қарым-қатынастарды қамтуы мүмкін; оның ұзақтығы әр түрлі өзгеруі мүмкін.

Жұмыс ағыны – бұл тапсырмаларды қолмен немесе механикаландыру (автоматтандыру) арқылы өңделетін қызметкерлермен алынған жұмыс тапсырмаларын жүйелі түрде және берілген бизнес-процесте анықталған ережелерге сәйкес жинау. Бизнес-процесс – оның ережелері мен технологиялары бойынша жұмыс істейтін конвейердіңбір түрі деуге болады және жұмыс процесі (тапсырмалары) бұл конвейердің қозғалысындағы өнімдердің (түйіндер, бөліктер) ағынымен ұқсас.

Бизнес-процесс жұмыс ағыны мен осы ағынның элементтеріне (міндеттеріне), осы функцияларды жүзеге асыратын адамдарға және жабдықтарға, сондай-ақ осы функциялардың реттілігін реттейтін ережелерге сәйкес келетін функцияларды біріктіреді. Технологиялық жұмыс бәрін автоматтандыруға арналған.

WfMC сөздігінен екі анықтаманы келтірейік.

1. *Жұмыс ағыны*– процедуралық ережелердің жиынтығына сәйкес бір қатысушыдан екіншісіне қажетті әрекеттерді орындау үшін құжаттар, ақпарат немесе тапсырмалар берілетін бизнес-үрдістің толық немесе ішінара автоматтандырылуы.

2. *Жұмыс үрдісін басқару жүйесі*–бұл ағынның (бизнес-үрдістің) сипаттайтын, оны жасайтын және үрдістің сипаттамасын түсіндіре алатын, қатысушылары өзара әрекеттесетін бағдарламалық жасақтама арқылы басқаратын және қажет болған жағдайда тиісті бағдарламалық жасақтама мен құралдарды шақыратын жүйе.

Мұндай жүйе функцияны емес, үрдісті автоматтандырады. Жұмыс ағынын басқару жүйелерінің және онымен байланысты бағдарламалық қамсыздандырудың пайда болуы ақпараттық технологиялар нарығының жаңа кәсіпорындарды басқару принциптерін енгізуінің реакциясы болып табылады. Бүгінде бұл қағидалар функционалдық бағдардан (Адам Смит ойлап шығарған 1776 жылы және екі ғасырдан астам уақыт жұмыс істейді) үрдісті бағдарлау бағытында дамиды.

Бұрынғы барлық дерлік шешімдер (көбінесе жергілікті ДҚБЖ технологиясында енгізілген) үрдістен гөрі кейбір операцияларды және функцияларды автоматтандыруға мүмкіндік берді. Осы шешімдердің бір бөлігі ретінде қызметкерлер өз компьютерлерінде (немесе терминалдарында) отырғанда, деректер қорымен және өзара қарым-қатынаста ақпарат алмасып, деректерді, ақпараттарды, құжаттарды және форма туралы есептерді ала алады.

Сонымен қатар, қызметкерлердің іс-әрекеттерінің жүйелілігі және олардың өзара әрекеттесу ережелері нұсқаулықтармен жақсы айқындалады, ал жоғары билік органдары оларды жүзеге асырудың дұрыстығына және мерзіміне бақылау жасайды. Ал ақпараттық жүйе мұны ешқандай жолмен қолдамайды.

Үрдістің тәсілі кәсіпорындардың басшылығын үрдістің қатысушылары арасындағы өзара әрекеттесу қағидаларына назар аударуға мәжбүр болады, өйткені олардың немқұрайлылығы мен сенімсіздігі негізсіз шығындар мен шығындардың негізгі көзі болып табылады. Жекелеген функцияларды (операцияларды), құжат жолдарын орындауды, үрдістің әр түрлі кезеңдерінде қызметкерлерді жұмысқа орналастыру тәртібін және уақытын автоматты түрде қадағалау қажеттілігі жұмыс үрдісін басқарудың әртүрлі жүйелерін жасауға және БАҚ тұжырымдамасының негізгі инварианттарының бірі ретінде осы тұжырымдаманы бекітуге әкелді.

Бақылау сұрақтары

1. «Интеграцияланған ақпараттық орта» термині нені білдіреді?
2. «Ақпараттық объект» термині нені білдіреді?
3. Бұйым туралы жалпы деректер қорында қандай ақпарат қамтылуы керек?
4. Кәсіпорынның жалпы дерекқор құрамында қандай ақпарат болуы керек?
5. Сапаны басқару жүйесі қандай міндеттерді және қандай стандарттарға сәйкес шешеді?
6. «Жұмыс ағынын басқару» және «бизнес-үрдістер» ұғымдары арасында қандай байланыс бар?

8 Тарау

АВТОМАТТАНДЫРЫЛҒАН ЖОБАЛАУ ЖҮЙЕЛЕРІНДЕГІ ДЕРЕКТЕР ҚОРЫ

18.1. Автоматтандырылған жобалаудың конструкторлық жүйелеріндегі деректер қоры

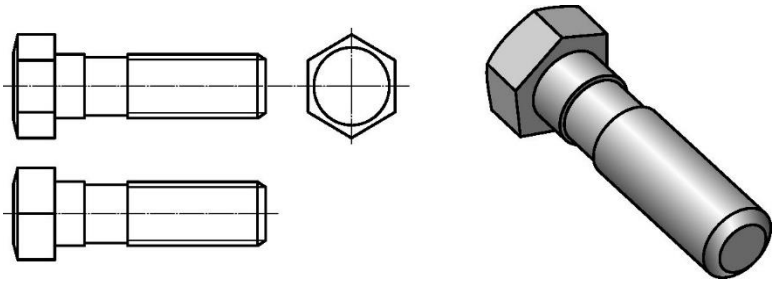
Тұрмыстық T-FLEX CAD немесе американдық Solid Works сияқты қазіргі заманғы жобалау жүйелері параметрлік сызу жүйелерін құрайды. Жаңа өнімге арналған сызбаларды жобалаудың осы әдісінің мәні әдеттегі өнімдердің сызбаларын қамтитын жобалау деректер қорын құруға негізделеді, оның барлық параметрлері айнымалыларды пайдалана отырып немесе формулалар арқылы есептелуі мүмкін. Бұл параметрлер дерекқорға (кестелер) енгізіледі.

Бұл жағдайда жаңа конструкцияның жобалануы келесі ретпен жүзеге асырылады.

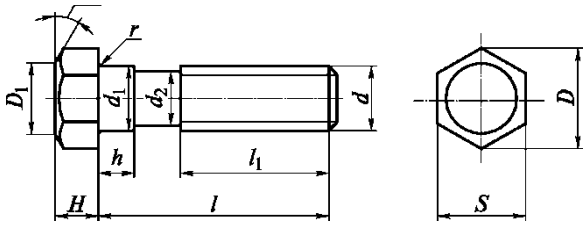
1. Конструкторлық дерекқордан өнім - аналог таңдалады.
2. Жаңа өнімнің параметрлері параметрлік деректер қорының кестесіне енгізіледі.
3. Енгізілген параметрлерге негізделген жүйе жаңа сызбаны қалыптастырады, оны конструкторлық деректер қорында сақтайды және оның параметрлері - параметрлік деректер базасында сақталады

Параметрлік деректер қорын жасау процесін төмендегідей әрекеттерге бөлуге болады:

1) деректер қорының кестесінің құрылымын қалыптастыру;
2) айнымалыларды, соның ішінде деректер қорына негізделген айнымалыларды құру;
3) сурет салу және жаңа деректерді енгізу параметрлерін таңдағанда пайдаланушы интерфейсін жасау.



18.1 сур. МЕМСТ 7795 – 70 сәйкес бұрандама



18.2 сур. Бұрандаманың өлшемдерін анықтайтын параметрлер

Дерекқорды құру келесі мысалда қарастырылады. Ол МЕМСТ 7795-70 сәйкес болуы керек (18.1 сур.).

18.2, 18.3 суреттерінде бұрандама тәрізді бөлікті құрастыруға арналған параметрлік деректер қорын қалыптастыру мысалдары көрсетілген.

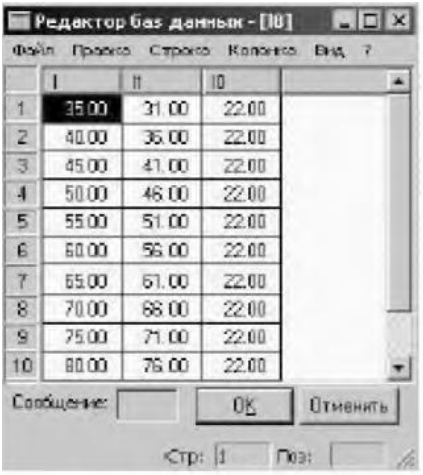
Бұл бұрандаманың параметрлері дерекқордан диаметрі мен ұзындығына байланысты таңдалуы керек. Бұрандаманың бірнеше конструкциясы бар, оған сәйкес сызбалар өзгертіліп отыру керек. Бұрандама басқа суреттердегі фрагменттер ретінде пайдаланылады.

Дерекқорды құрған кезде, мәндер таңдалатын ішкі дерекқорларды жасау қажет. Алдымен, қалғандары қандай параметрлерге сәйкес таңдалатынын шешуге тура келеді. 18.2 суретінде бұрандамаөлшемін анықтайтын параметрлер көрсетілген.

Біздің жағдайда бұл параметрлердің бірі –бұрандаманың диаметрі. Мәселен, бірінші баған – бұрандама диаметрі, содан кейін қалғандары (ұзындығын қоспағанда). Дерекқор бағандарының атаулары (және дерекқордың өзі) кейінірек, айнымалы мәндерді жасағанда, әр бағанға жауап беретін параметрді оңай еске түсіру үшін таңдалуы керек, бірақ баған атауларын тым ұзақ етіп жасамағаны дұрыс.

Деректер қорының редакторы								
	d	d1	h	S	H	D	r	I
1	6	6	3	10	4,20	10,90	0,25	2,00
2	8	8	4	12	5,50	13,10	0,40	2,80
3	10	10	5	14	7,00	15,30	0,40	3,50
4	12	12	6	17	8,00	18,70	0,60	4,00
5	16	16	8	22	10,00	24,30	0,60	5,00
6	20	20	10	27	13,00	29,90	0,80	6,50
7	24	24	12	32	15,00	35,00	0,80	7,50
8	30	30	15	41	19,00	45,20	1,00	9,50
9	36	36	18	50	23,00	55,40	1,00	11,50
10	42	42	21	60	26,00	66,40	1,20	13,00
11	48	48	24	70	30,00	77,70	1,60	15,00

18.3 сур. Бұрандама параметрлерінің кестесі көрсетілген фрагмент



18.4 сур. Бір параметр кестесінің фрагменті

Берілген параметр мемлекеттік стандартта қалай аталса, бағандарды да солай атаған абзал: бұранда диаметрі — *d,* кілт астындағы өлшем— *S*және т.б.

18.3 суретінде бұрандама параметрлері кестесінің фрагменті бейнеленген.

Біздің жағдайда бұрандама ұзындығы ең жақын стандартты ұзындыққа дейін дөңгелектенуі керек. Әрбір диаметр үшін ұзындығы бар жиынтығын ескеру қажет. Мысалы, диаметрі 6 мм болатын бұрандама ұзындығы 22 мм болуы мүмкін, диаметрі 12 мм болатын бұрандаманың ең аз ұзындығы 32 мм болады. Қай бұрандама ұзындығының таңдалғанына байланысты басқа екі параметр таңдалады: 11 және I0. Диаметрінің ұзындығы үшін бөлек дерекқор құрған абзал.

Қор атауларын I6, I 18 және т.б. деп қою керек. I дегеніміз – ұзындық туралы деректер қоры, ал нөмір – сол үшін арналған диаметр.

18.4 суретінде осындай кестенің фрагменті көрсетілген.

Бұл мысалда біз деректер қоры заманауи CAD-жүйелерінің ажырамас бөлігі болып табылатынын көрсеттік.

18.2. Технологиялық жобалау жүйелеріндегі деректер қоры

Access ДҚБЖ-сін электрлік сымдарды балқыту және балқыту технологиялық үрдістерінің автоматтандырылған жобалау жүйесін жасау үшін пайдаланудың нақты үлгісін қарастырайық. Бұл АЖЖ К.Э. Циолковский атындағы РГТУ – ТПП және СУЛА МАТИ кафедрасының студенттерінің дипломдық жобасы ретінде әзірленген. АЖЖ-ны дамытудың алғашқы технологиялық құралдары – бір аспап жасау зауытының стандарттары.

Қарастырылған АЖЖ келесі мәселелерді шешуге арналған:

• Қалайылау және дәнекерлеу операцияларын орындауда барысындағы қауіпсіздік техникасы жөніндегі талаптарды әзірлеу;

• Қалайылау және дәнекерлеу әдістеріне қарай технологиялық үдерістердің құрамы мен ретін жобалау;

• Дәнекерленген қосылыстарда пайда болған ақаулардың себептерін анықтау.

Көріп отырғанымыздай, аталған АЖЖ сараптамалық жүйе элементтерін қамтиды.

Жобалау әдістемесі келесідей. Технологиялық үрдісті жүргізудің қолданыстағы тәсілдеріне сәйкес кәсіпорында қолданылатын технологиялық үрдістің (ТҮ) түрлерін (нұсқаларын) белгілейтін классификатор әзірленуде, олардың әрқайсысы белгілі бір сипаттамалар жиынтығымен сипатталады.

ТҮ барлық нұсқалары үшін технологиялық ауысудың құрамы мен дәйектілігі белгіленеді, олардың мазмұны тиісті кестеде жазылады. ТҮ-нің әрбір нұсқасы осы кестеде келтірілген үрдістердің белгілі бір құрамына сәйкес келеді.

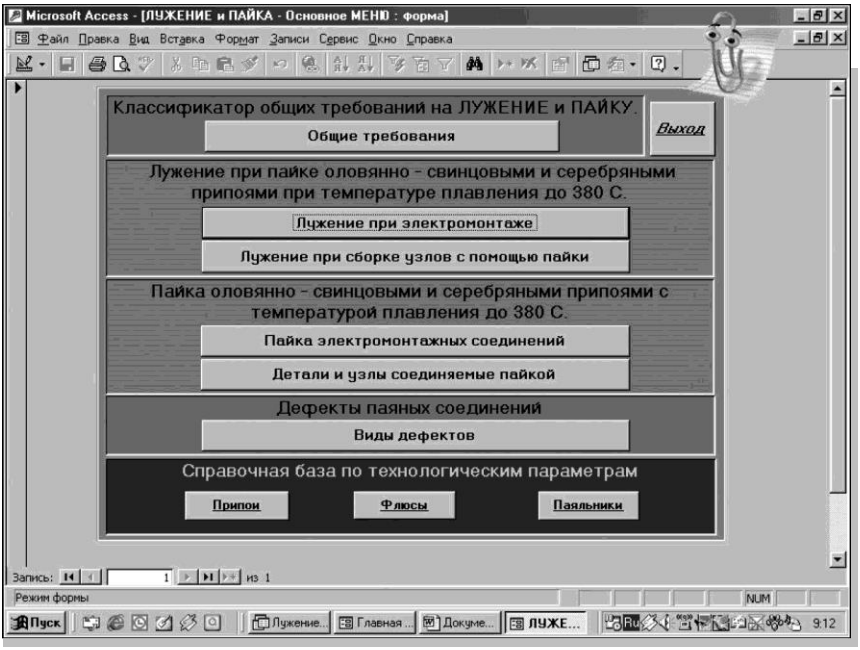
Автоматтандырылған жобалау үрдісі классификатордан ТҮ нұсқасын таңдауды азайтады. Содан кейін жүйе кестеден тиісті технологиялық ауысуды таңдайды да, оны операциялық диаграмма түрінде жасайды.

АЖЖ негізінде құрылған интерфейсті мысал ретінде келтірейік.

18.5 суретінде келесі әрекеттерді таңдайтын батырмалар формасында жасалған негізгі бет бейнеленген.

Қалайылау және дәнекерлеу батырмасын басқанда, келесі диалог формасы ашылады (18.6 сур.) да, одан әрі қарай төмендегілерге өтуге болады:

18.5 сур. АЖЖ негізгі беті



18.6 сур. Жүйемен жұмыс істеу режимін таңдау формасы

• қалайылау және дәнекерлеудің технологиялық операцияларын орындау үшін жалпы талаптарды әзірлеуге;

• электр жұмыстары кезінде немесе басқа да монтаждау жұмыстары кезінде тазалау жұмыстарына технологиялық операцияларды жобалау;

• электрмонтаждық қосылыстар немесе бөлшектер мен түйіндерді дәнекерлеуге арналған технологиялық операцияларды жобалау;

• дәнекерленген буындардың ақауларын анықтау.

18.6 суретінен байқасақ, жүйеде кәсіпорында пайдаланылатын беттерге, ағындарға және дәнекерлеу жабдығына (дәнекерлеу үтігі) қатысты анықтамалық мәліметтер алуға болады.

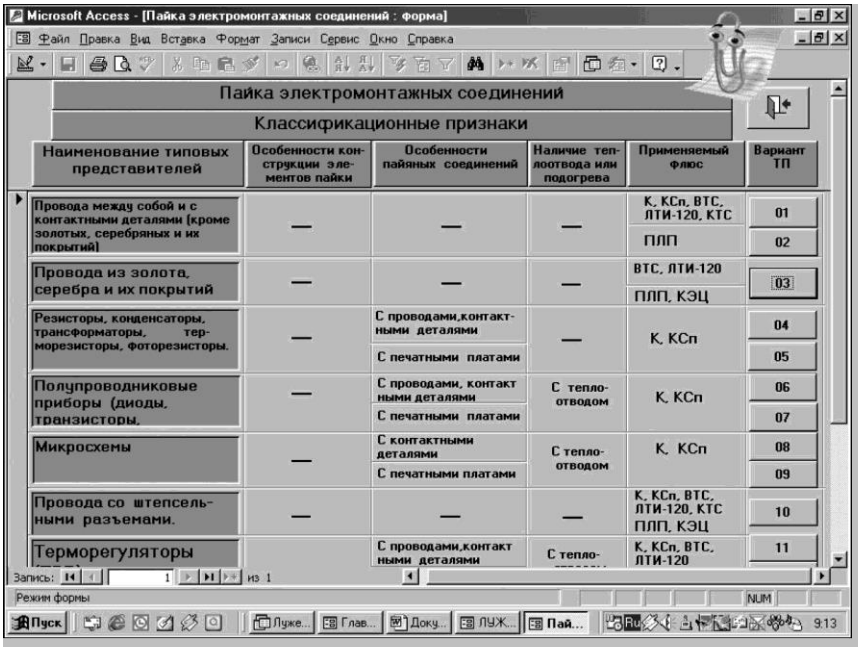
Жүйе жұмысының кейбір фрагменттерін ұсынамыз.

Электрмонтаждық қосылыстарды дәнекерлеу батырмасы басылғанда (18.6сур. қараңыз), ТҮ параметрлерін және олардың сипаттамаларын қамтитын жіктеуіш (18.7 сур.) ашылады.

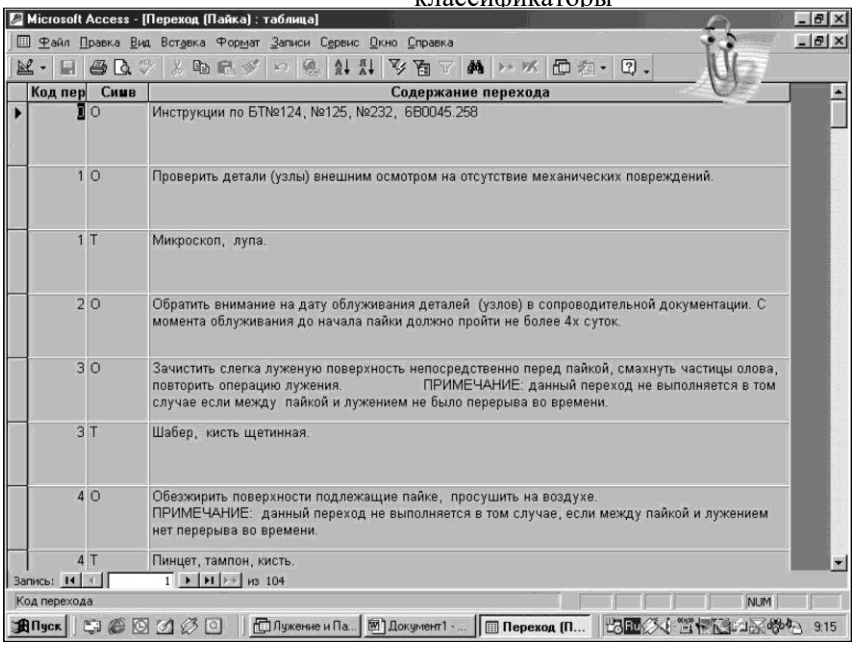
Қажетті опцияны таңдап, тиісті батырманы басқанда сұрау орындалады, оған 18.8 суретінде бейнеленген қажетті технологиялық ауысулар таңдалады және олар реттілікпен ұйымдастырылады.

18.9 суретінде *Құрастырушы* режиміндегі сәйкес сұрау түрі көрсетілген. Осы сұраныстың мәтінін SQL-де жазайық:

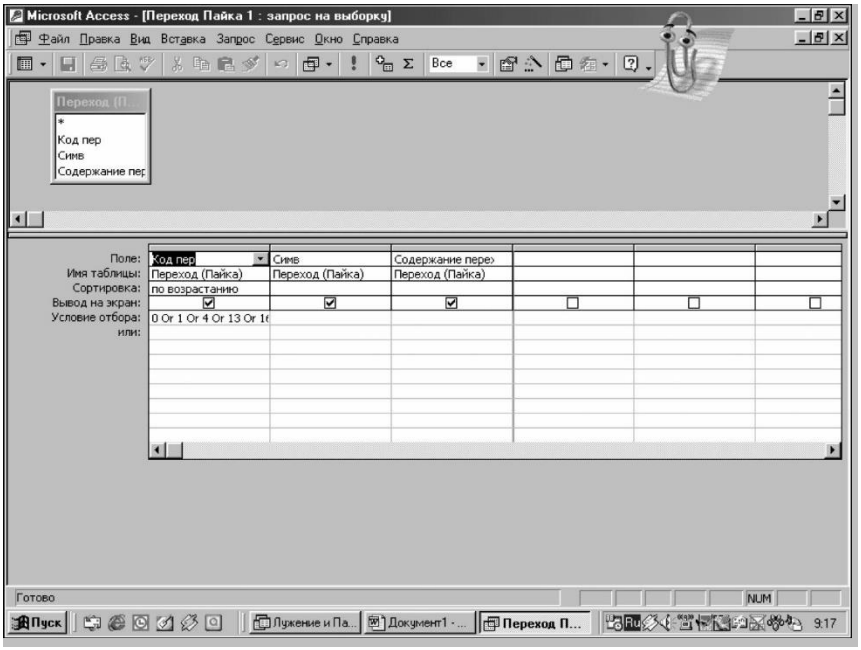
SELECT DISTINCTROW [Ауысу (Дәнекерлеу)].[Ауысу коды], [Ауысу (Дәнекерлеу)].Симв, [Ауысу (Дәнекерлеу)].[Ауысу мазмұны]



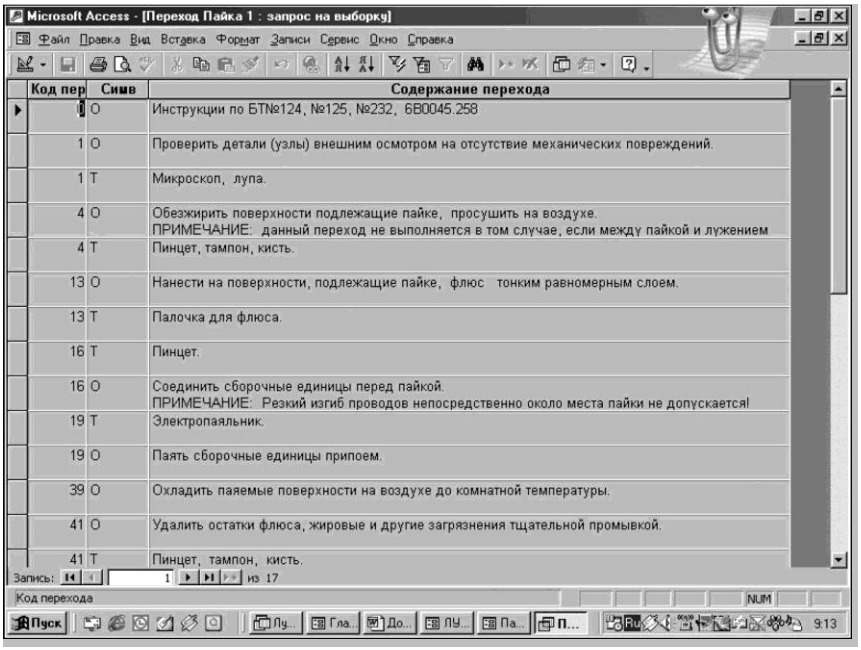
18.7 сур. Электрмонтаждық қосылыстарды дәнекерлеу процестерінің классификаторы



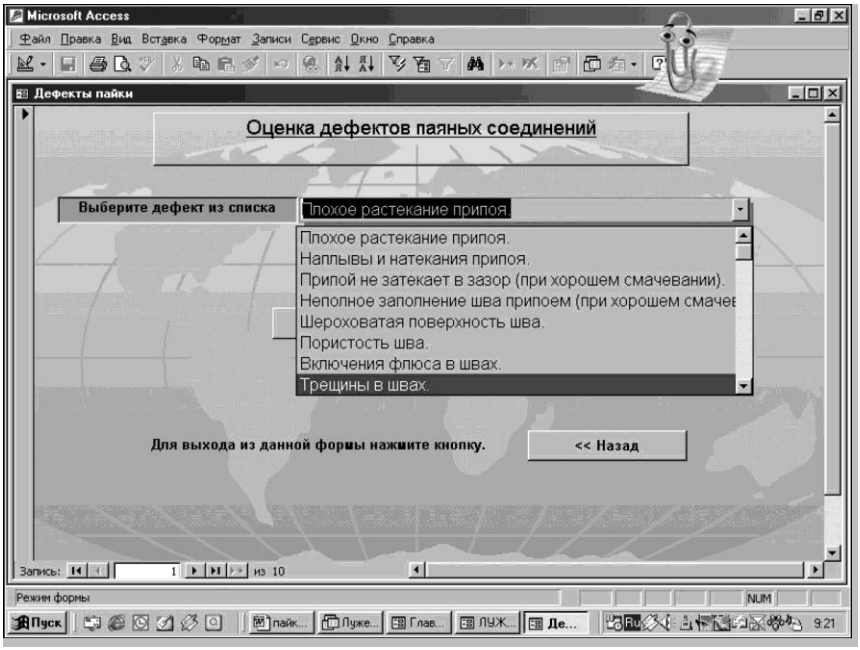
18.8 сур. Типтік технологиялық ауысуылар мазмұнының кестесі



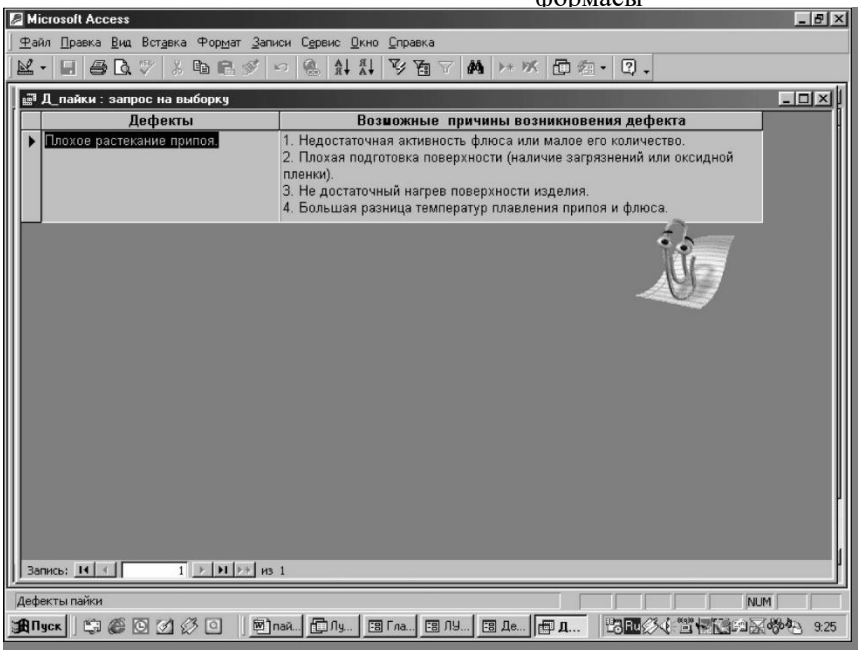
18.9 сур. *Құрастырушы* режиміндегі сұраныстың түрі



18.10- сур. Сұраныстың орындалу нәтижесі



18.11 сур. Дәнекерленген қосылыста пайда болған ақаудың түрін таңдау формасы



18.12 сур. Дәнекерленген қосылыста пайда болған ақаудың ықтимал себептері

FROM[Ауысу (Дәнекерлеу)]

WHERE ((([Ауысу (Дәнекерлеу)].[ауысу коды]) = 0 Or ([Ауысу (Дәнекерлеу)].[ауысу коды]) = 1 Or ([Ауысу (Дәнекерлеу)].[ауысу коды]) = 4 Or ([Ауысу (Дәнекерлеу)].[ауысу коды]) = 13 Or ([Ауысу (Дәнекерлеу)].[ауысу коды]) = = 16 Or ([Ауысу (Дәнекерлеу)].[ауысу коды])=19 Or ([Ауысу (Дәнекерлеу)]. [ауысу коды]) = 39 Or([Ауысу (Дәнекерлеу)].[ауысу коды]) = 41 Or([Ауысу (Дәнекерлеу)].[ауысу коды]) = 46 Or([Ауысу (Дәнекерлеу)].[ауысу коды]) = 48))

ORDERBY[Ауысу (Дәнекерлеу)].[ауысу коды];

Or шартымен байланыстырылған ауысулар кестесінен деректерді таңдау шартын қарастырайық:

[Ауысу (Дәнекерлеу)].[ауысу коды])=0 Or([Ауысу (Дәнекерлеу)].[ауысу коды]) = 1 ...

OR логикалық операторына қолдану арқылы таңдау шартын қолдану, жазбалардың мазмұнын берілген шарттар жинағына сай реттейді.

Сұраныстың орындалуы нәтижесінде, динамикалық кесте ашылады (18.10 сур.), ол таңдалған ТҮ нұсқасына ғана арналған ауысым реті мен мазмұнын құрайды.

18.11, 18.12 суреттерінде дәнекерленген қосылыста пайда болған ақаудың себептерін анықтау фрагменті көрсетілген.

18.3. Сараптамалық компьютерлік жүйелер

Кез келген кәсіпорынның негізгі міндеті технологиялық даярлық пен өндірісті басқару шығындарын азайту болып табылады. Маңызды мәселелердің бірі – жаңа өнімдерді өндіруге жұмсалатын еңбек және материалдық шығындарды жылдам бағалау. Бұл мәселені екі жолмен шешуге болады:

1) жоғары білікті мамандарды тарту;

2) технологиялық деректер қоры негізінде компьютер сараптамалық жүйелерді құру.

Қазіргі таңда көптеген кәсіпорындар менеджмент саласында білікті мамандардың жетіспеушілік мәселесіне тап болады, сондықтан басқару шешімдерін қабылдау саласындағы сараптамалық жүйелерді дамыту басым міндеттердің бірі болып табылады.

Сараптамалық жүйелерді дамыту үшін, Microsoft Access сияқты дерекқорды басқарудың тиімді жүйесін пайдаланады.

Жаңа өнімдерді өндіру бөліктерінің күтілетін шығындарын бағалауға арналған сараптамалық жүйені қарастырайық.

Ұсынылған сараптамалық жүйе төмендегідей болжамдарға негізделген.

1. Кәсіпорында негізгі өндірістің барлық деректемелері технологиялық классификаторға сәйкес аспап жасау және машина жасау (ТКД) бөлшектеріне сәйкес технологиялық коды бар.

2. Кәсіпорында технологиялық деректерді өңдеудің еңбекке жарамдылығы туралы деректер бар құрылымдық технологиялық үдерістің әрбір бөлшегіне арналған технологиялық деректер қоры бар.

Осы ережелерге сүйене отырып, жаңа бөліктердің күтілетін еңбекке ақы төлеу болжамын келесі іс-әрекеттерге дейін қысқартады.

1. Бөліктің технологиялық кодын тағайындау.

2. Дерекқордан ұқсастықтарды – ұқсас технологиялық коды бар мәліметтерді табу.

3. Дерекқорда бар осы технологиялық кодтың барлық деректемелерінің орташа күрделілігін есептеу.

4. Жаңа бөлшекке орташа есептелген еңбексыйымдылығын тағайындау.

Нәтижелер көрсеткендей, сараптамалық бағалаудың қателігі технологиялық классификацияның негізі 20% -дан аспайды, бұл белгілі бір кәсіпорын жағдайында жаңа өнімді жасаудың орындылығы туралы шешім қабылдау үшін әбден қанағаттанарлық.

Сараптамалық жүйені дамытудағы міндеттердің бірі – технологиялық классификаторға сәйкес бөлшектердің автоматтандырылған кодтау жүйесін дамыту.

Машина жасау және аспап жасау бөлшектерінің технологиялық классификаторы сонау 1981—1985 жылдары КСРО Мемлекеттік жоспарлау комитеті мен ҒТМК бұйрығында бекітілген ЕСКК және ЕСУД жөніндегі Бағдарламалық жұмыстарға сәйкес құрылған.

ЕСТПП іске асыру шеңберінде өндірісті дайындауда индустриядағы бөлшектердің технологиялық классификаторын кеңінен қолдану компьютерлік техниканы пайдалану және өндірістік техникалық құралдарды пайдалану арқылы өндірістік мәселелерді шешудің жоғары тиімділігін көрсетті.

Бөлшектердің технологиялық классификациясы өндірістің және оны басқарудың жүйесінде техникалық және экономикалық ақпараттың одақтық классификаторларымен бірге қолданылады. Салалардағы оны енгізу тәжірибесі келесі негізгі міндеттерді шешу үшін алғышарттар жасаған:

• олардың құрылымдық және технологиялық сипаттамалары туралы мәліметтердің номенклатурасын талдау;

• компьютерлік техниканы пайдалана отырып, стандартты және топтық технологиялық үдерістерді әзірлеу үшін жобалау және технологиялық ұқсастық үшін бөлшектерді топтастыру;

• өндірістік бірліктер (бөлікшелер, цехтар, фабрикалар) филиалдарының мамандануы;

• бөлшектерді өндірудің сериялығы мен шоғырлануын арттыру;

• бөлшектер мен оларды өндірудің технологиялық үдерістерін біріктіру және стандарттау;

• технологиялық жабдықтардың түрлерін ұтымды таңдау;

• бұрын дамыған типтік немесе топтық технологиялық процестерді тақырыптық іздеу және қарыз алу;

• бөлшектерді жобалауды және оларды өндірудің технологиялық процестерін автоматтандыру.

ТБЖ-ның негізгі мақсаты еңбексыйымдылығын төмендету және өндірісті технологиялық дайындау кезеңін қысқарту болып табылады.

Қазіргі уақытта индустрияда 40 және 50 класты «Жалпы машина жасау қосымшасы туралы мәліметтер» технологиялық классификаторы бөлшектерді өндірудің заманауи шарттарына толығымен сәйкес келмейді, өйткені ол негізінен жалпы машина жасау сипатындағы бөлшектердің шектеулі ауқымына дейін ғана жетеді.

Машинажасау және аспап жасау салаларының қазіргі технологиялық жіктеушісі оның құрылысының негізгі қағидаларын ескере отырып, негізгі және қосалқы салалардың барлық салаларының бөлшектерін қамтиды.

Бұл КҚБЖ классификаторының (71, 72, 73, 74, 75, 76 кластары) бөлшектері кластарының логикалық жалғасы және қосымшасы болып табылады.

КҚБЖ классификаторы МЕМСТ 2.201 - 80 «КҚБЖ. Өнімдер мен конструкторлық құжаттарды белгілеу» бірыңғай жіктелуі идентификацияланбаған, машинажасау мен аспап жасау өнімдерінің конструкциялық құжаттары жүйесінің ақпараттық бөлімі ретінде жасалды. Бөлімдердің кластары оңтайлы жағдайлар құрайды:

• автоматтандырылған басқару жүйелері үшін бірыңғай ақпараттық тілді құру және осындай ұқсастықтың дамуын болдырмау үшін мәліметтер мен олардың конструкторлық құжаттарын тақырыптық іздеуді жеңілдету;

• біріздендіру және стандарттау объектілері мен бағыттарын анықтау;

• басқа ұйымдардың әзірлеген, жобалау, өндіру, пайдалану, жөндеу кезіндегі әр түрлі кәсіпорындар мен ұйымдардың жобалау құжаттамасын оны қайта тіркеусіз пайдалануын қамтамасыз ету;

• жобалау және басқару саласында компьютерлік техниканы кеңінен енгізу;

• компьютерлік техниканы қолдана отырып өндірісті технологиялық дайындау мәселелерін шешу үшін технологиялық кодтармен бірге бөліктердің кластары бойынша бөліктер кодтарының қолданылуы.

Бұл классификаторды пайдалану CALS-технологиялық кәсіпорындарында құрудың және енгізудің негізгі әдістерінің бірі болуы керек.

Технологиялық классификатор бөлшектердің технологиялық кодтарын белгілейді:

- құю арқылы шығарылатын;
- соғу және көлемді қалыптау арқылы дайындалатын;
- табақты қалыптау арқылы дайындалатын;
- кесу арқылы өңделетін;
- термиялық өңделетін;
- полимерлік материалдар мен резеңкеден жасалған бұйымдар;
- қаптамалы;
- электрфизикалық-химиялық әдістермен өңделген;
- ұнтақты металлургиясында өндірілетін.

Бөлшектердің технологиялық жіктелуінің негізін – бұл әртүрлі жіктеу критерийлері бойынша топқа бөлінген фасеталық әдіс құрайды.

Технологиялық классификациялау негізгі және қосалқы өндірістерді машина жасау және аспап жасау бөлшектеріне жүргізіледі.

Классификациялау сипаттамалары бөлшектердің маңызды технологиялық сипаттамаларын пайдаланады, олар құрылымдық ерекшеліктерімен бірге олардың технологиялық ұқсастығын анықтайды. Жіктеу кестелері – технологиялық классификацияның негізгі ерекшеліктері мен оның өндіріс технологиялық әдісімен бөлшектердің түрін сипаттайтын сипаттамалар.

Мәліметтер әріптік-цифрлық кодпен кодталады. Бөліктердің технологиялық кодының құрылымында әр атрибут үшін белгілі бір санат (позиция) және белгілер саны белгіленеді. Кодтық белгілерді құру жүйесі компьютерлік техниканы қолдана отырып, оңтайлы бөлшектерден тұратын топтарды қалыптастыруды қамтамасыз етеді.

Конструкторлық-технологиялық кодының құрылымы өндіріс мәселелерін шешу үшін түрлі код комбинацияларында ақпаратты өңдеуді қамтамасыз етеді және шешілетін міндеттердің сипатына байланысты кодының бөлшектерін және олардың комбинацияларын пайдалануға мүмкіндік береді.

Технологиялық классификатор – бұл жекелеген сипаттамалардың құрамдас бөлшектері мен жіктеу кестелері түріндегі код белгілерінің жалпы сипаттамаларының атауларының жүйеленген тізімі. Классификатор формасы классификатордың басқа позицияларын өзгертпестен және оны қайта шығарусыз, оның мазмұнын жедел өзгертуге мүмкіндік береді.

Бөлшектің технологиялық коды 14 таңбадан тұрады. Бұл кодтың белгісі екі бөліктен тұрады: негізгі сипаттамалардың жіктеу топтарының кодының белгісі (тұрақты бөлік) – алты таңба; өндіріс әдісі бойынша бөлшектің объектісін сипаттайтын сипаттамалардың классификациялық топтарының кодының белгісі (айнымалы бөлік) - сегіз таңба (18.13 сур.).

1 2 3 4 5 6 X X X **7 8 9 10 11 12 13 14**
X X X x x x x x x x x

Негізгі сипаттамалардың
классификациялық тобының коды

Өндірістің технологиялық әдісінде егжей-тегжейлі сипатталатын негізгі белгілердің классификациялық топтарының коды

18.13 сур. Әзірлеу әдісі бойынша бөлшектің жалпы кодының құрылымы

Бөлшектердің технологиялық жіктелуінің келесі негізгі ерекшеліктері қабылданған:

- өлшемді сипаттамасы;
- материалдық топ;
- дайындаудың технологиялық әдісі бойынша бөлшек түрі.

Құрамында кодының құрылымы мен ұзындығынегізгі технологиялық ерекшеліктері бар классификациялық топтарының кодтары 18.14 суретінде көрсетілген.

Технологиялық әдіспен бөлшек түрі келесі ерекшеліктермен сипатталады:

- бастапқы дайындаманың түрі;
- кедір-бұдырлық параметрі;
- квалитет;
- технологиялық талаптар сипаттамалары;
- қосымша сипаттамасы;
- жаппай сипаттамасы және т.б.

Бөлшек түрін сипаттайтын сипаттамалардың классификациялық топтары үшін код таңбасының құрылымы тиісінше классификатордың әрбір бөлімінде берілген.

Конструктивті сипаттар мен технологиялық кодтың классификациялық топтар кодынан құралатын бөлшектің конструкторлық-технологиялық кодының құрылымы 18.15 суретінде көрсетілген түрде болады.

Технологиялық классификацияның жоғарыда аталған жүйесіне сәйкес жоғарыда аталған барлық деректердің автоматтандырылған кодтау жүйесі жасалды.

1 2 3 **4 5** **6**
XXX XX X

Өлшемдік сипаттама

Материал тобы

әзірлеудіңтехнологиялық әдісіне қарай бөлшектің түрі

Конструктивті сипаттардың классификациялық топтарының коды (КҚБЖ классификаторы бойынша классификациялық сипаттама)

XX XXX X

Класс
Подкласс
Группа
Подгруппа
Вид _____

Бөлшектің технологиялық коды

X X X X X X X X X X X X X

Бөлшек түрін сипаттайтын классификациялық сипаттар тобының коды
Технологиялық әзірлеу әдісі бойынша басты сипаттардың классификциялық топтарының коды

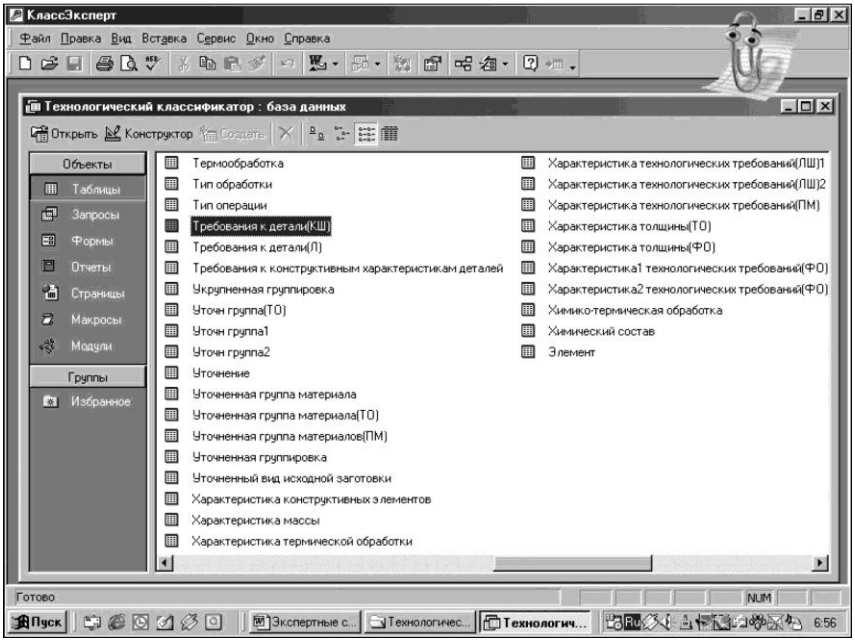
а *б*

18.15 сур. Бөлшектің конструктивтік-технологиялық кодының құрылымы:

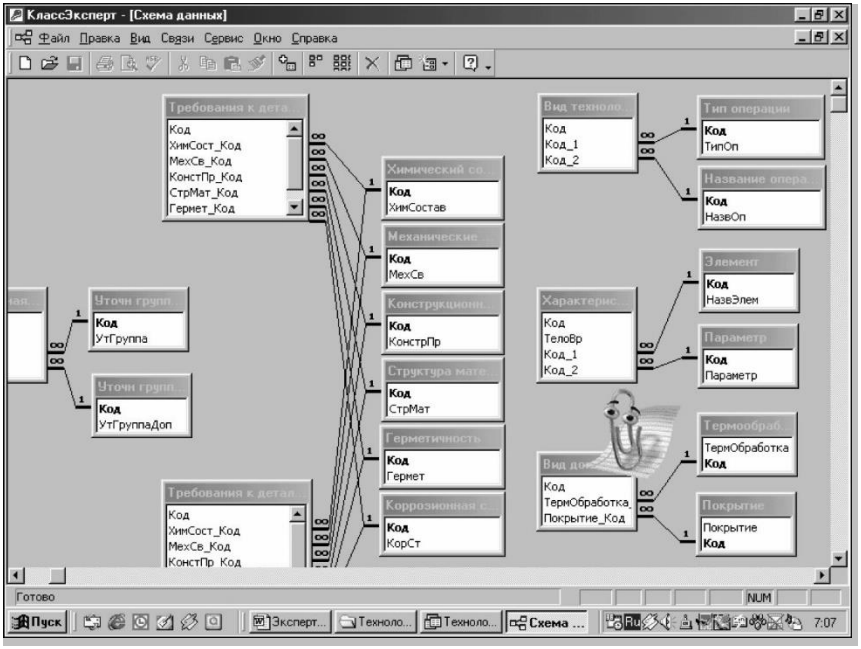
Жүйе бастапқы кестелерден тұрады - бөлшектің әрбір сипаттамасына тиісті технологиялық кодын тағайындайтын білім негіздері.

18.16 суретінде бастапқы кестелер құрамының фрагменті— білім қоры көрсетілген.

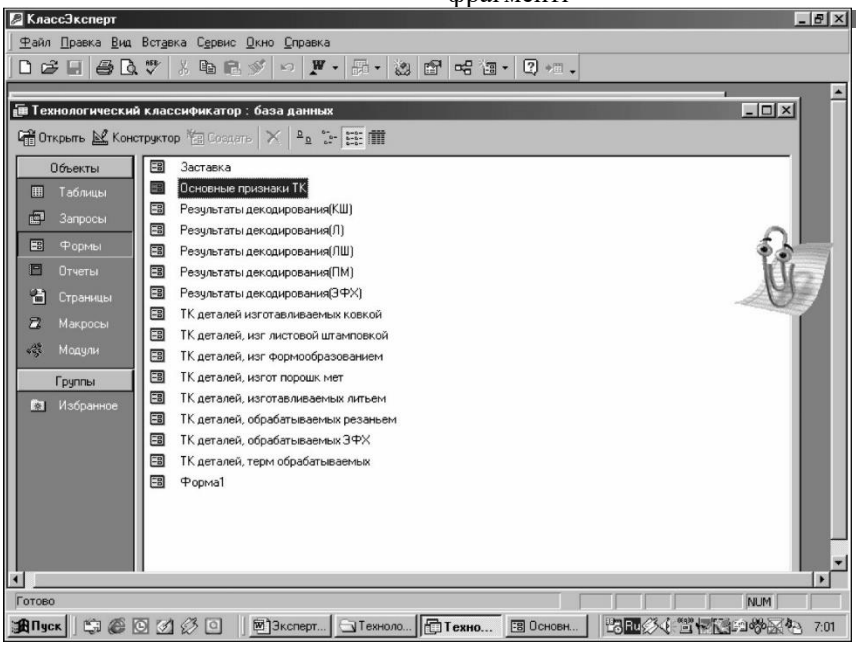
18.17 суретінде құрастырылған кестелер арасындағы байланыстар сұлбасының фрагменті көрсетілген.



18.16 сур. Сараптамалық жүйенің бастапқы кестесінің (білім кестесі) мазмұны



18.17 сур. Кестелер арасындағы байланысты көрсететін сұлбаның фрагменті



18.18 сур. Сараптамалық жүйенің диалогтік формаларының мазмұны

Берілген кестелер деректеріне сәйкес және жіктеу жүйесіне сәйкес бөлшектердің сипаттамаларын енгізудің тиісті формалары әзірленді.

18.18 суретінде сараптамалық жүйенің диалогтық формаларының құрамы көрсетілген.

18.19 суретінде бөлшектердің технологиялық классификациясының негізгі ерекшеліктерін кодтаудың объектісі көрсетілген.

18.20 суретінде беттерді кесу әдістерімен дайындалған бөлшектердің технологиялық сипаттамаларын кодтауға арналған объектіні көрсетеді.

18.21 суретінде сараптамалық жүйенің сұраныстар құрамы ұсынылған.

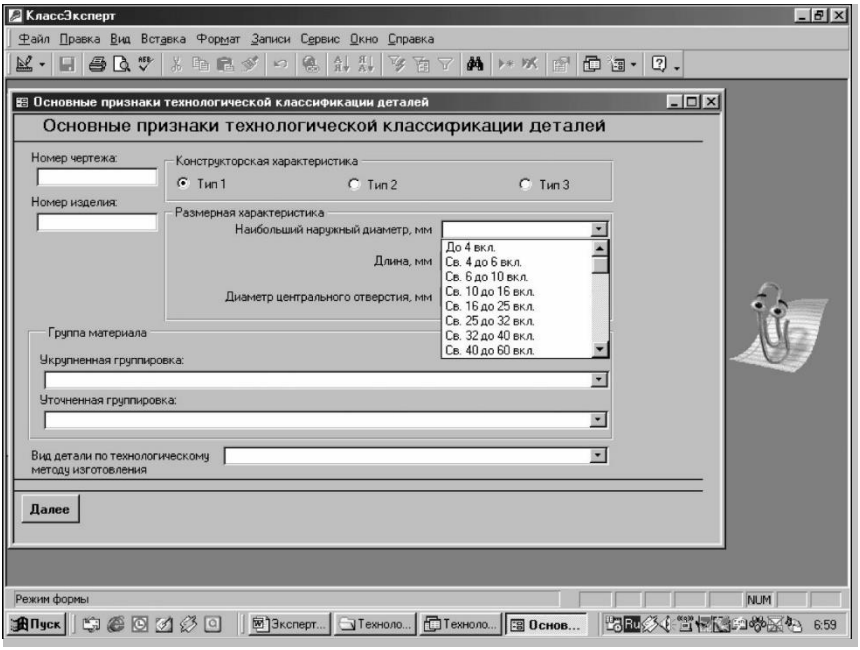
18.21 суретінен байқайтынымыздай, құрамында кодтау және декодтау сұраныстары бар.

Кодтау туралы сұраныстың нәтижесі – енгізілген сипаттамаға сәйкес бөлшекке арналған технологиялық кодты генерациялау.

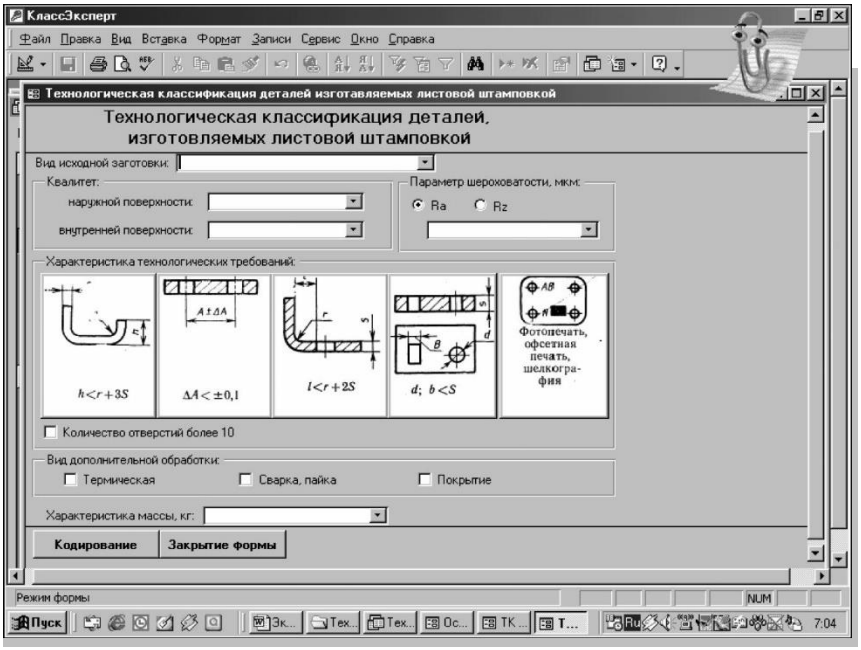
Декодтауға арналған сұраныстың нәтижесі кері кодын шешеді – енгізілген бөлшек кодына сәйкес оның технологиялық сипаттамалары орнатылған.

Кодтау сұраныстарының бір мәтінін мысал ретінде келтірейік:

INSERTINTO[Негізгі кесте] (ҚұрастыруТұрақтыКодҚалыпКод, СызбаНөмірі, БұйымНөмірі)



18.19 сур. Бөлшектерді классификациялаудың басты технологиялық сипаттарын кодтау формасы



18.20 сур. Табақты қалыптау әдістерімен жасалатын бөлшектердің технологиялық сипаттамаларын кодтау формасы

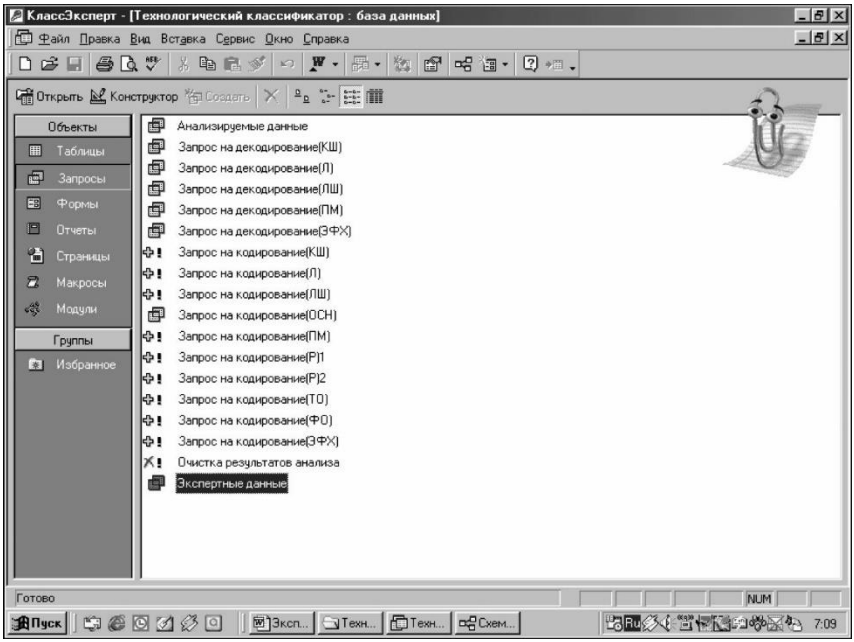
SELECTSELECT[Сұраныстыкодтау(ОСН)].Құрастыру, [Сұраныстыкодтау(ОСН)].ТұрақтыКод, [Бастапқы әзірлеменің түрі].[Код] & [Квалитет].[Код] & [Квалитет_1].[Код] & [Кедір-бұдырлық параметрі].[Код] & [Дәлдік деңгейлері].[Код] & [Қосымша өңдеудің түрі(Р)].[Код] & [Масса сипаттамасы].[Код] ASҚалып- Код, [Сұраныстыкодтау(ОСН)]. СызбаНөмірі, [Сұраныстыкодтау(ОСН)].БұйымНөмірі

FROM[Бастапқы әзірлеменің түрі], [Дәлдік деңгейлері], [Қосымша өңдеудің түрі(Р)], [Масса сипаттамасы], Квалитет, Квалитет ASКвалитет_1, [Кедір-бұдырлық параметрі], [Сұраныстыкодтау(ОСН)]

WHERE((([Бастапқы әзірлеменің түрі].Код)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өрістізіммен0]) AND((Квалитет.Код)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өріс тізіммен2]) AND((Квалитет_1.Код)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өрістізіммен 4]) AND(([Кедір-бұдырлық параметрі].Код)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өріс тізіммен 6]) AND(([Дәлдік деңгейлері].Код)= = [Кесіп өңделетін бөлшектердің Forms^^.]![Өрістізіммен 20]) AND(([Масса сипаттамасы].Код)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өрістізіммен 10])AND(([Бастапқы

әзірлеменің түрі].КодВидДетТМ)="4" AND(([Қосымша өңдеудің түріТКН(Р)].noKpbrrae_K^a)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өрістізіммен 40]) AND(([Қосымша өңдеудің түрі(Р)].ТермӨңдеу_Код)=[Forms]![Кесіп өңделетін бөлшектердің ТК-сы]![Өрістізіммен 38]));

Сараптамалық жүйе жұмысы ұқсас конструкторлық және технологиялық кодпен толық аналогты іздеуге негізделген. Бұл функция Visual Basic бағдарламасымен орындалады. Сонымен қатар аналогты көп сатылы іздестіру жүргізілуде, өйткені кәсіпорын шығаратын өнімнің ауқымы үлкен болса да, конструкторлық-технологиялық кодының 100% сәйкес келетін бөлігін табуға болады. Сондықтан толық аналог табылмаса, онда екінші іздеу жүргізіледі. Бұл жағдайда іріктеу критерийлері азаяды, яғни. жүйе конструкторлық және технологиялық кодекстің барлық ұстанымдарын салыстырмайды, бірақ ең маңыздысы ғана. Конструкторлық және технологиялық кодын таңдау жұмыстың технологиялық әдісімен (технологиялық кодының тұрақты бөлігінің алтыншы позициясы) бөлшек түріне негізделеді. Бұл технологиялық кодтың айнымалы бөлігінің айтарлықтай ерекшеленуімен байланыстырылады.



18.21 сур. Сынақ жүйесінің сұраныс құрамы

Осылайша, осы жүйе деректерді интеллектуалды талдауды жүзеге асыруға мүмкіндік береді, өйткені бір топқа тағайындалған бөлшектердің еңбексыйымдылығының таралуын бағалау мүмкін болады. Осылайша екі маңызды міндеттер шешіледі:

• жаңа бөлшектің еңбексыйымдылығын бағалау кезінде қатені болжауға мүмкіндік береді;

• бөлшектердің жіктелуіндегі қателерді, сондай-ақ егжей-тегжейлі мәліметтердің күрделілігін анықтаудағы қателерді анықтауға мүмкіндік береді.

Өндірістік бөлшектердің еңбексыйымдылығын сараптауға арналған алгоритмнің блок-сұлбасы 18.22. суретінде көрсетілген.

Төменде келтірілген кестедегі деректерді талдауға арналған VBA программасының мәтіні және жаңа түсініктемелерге жұмысты сараптамалық бағалау үшін қажетті түсініктемелер бар:

```
Option Compare Database
Option Explicit

Private Sub Батырма0_СИскО On
Error GoTo Err_Батырма1_Qick
Dim dbs As Database, Rst1 As Recordset, Rst2 As Recordset, Rst3 As
Recordset, Fj As Boolean, stDocName As String stDocName = "талдау
нәтижелерін тазалау" DoCmd.OpenQuery stDocName, acNormal, acEdit '
"Талдау нәтижелері" кестесін тазалау'
Set dbs = CurrentDb ' Ағымдағы дерекқорды білдіретін Database
типті объектілі айнымалыны қайтарады.'
Set Rst1 = dbs.OpenRecordset("Талданатын деректер") ' "Талданатын
деректер" кестесінің деректеріне RST1 атымен кіретін жаңа Recordset
объектісін жасайдыда, оны Recordsets тобына қосады.'
Set Rst2 = dbs.OpenRecordset("Талдау нәтижелері") ' "Талдау
нәтижелері" кестесінің деректеріне RST2 атымен кіретін жаңа
Recordset объектісін жасайдыда, оны Recordsets тобына қосады.'

Set Rst3 = dbs.OpenRecordset("Сараптамалық деректер") '
"Сараптамалық деректер" кестесінің деректеріне RST3 атымен кіретін
жаңа Recordset объектісін жасайдыда, оны Recordsets тобына қосады.'
' Сыртқы циклдің басталуы. Циклде "Талданатын деректер"
кестесінің деректері ретімен таңдалады
' және "Сараптамалық деректер" кестесінің деректерімен
салыстырылады. DoUntilRstl.EOF
Fj = False'Fj айнымалысы іздеу нәтижелерін көрсетеді
Rst3.MoveFirst' Әр іздеу циклының алдында указатель записей таблицы
"Сараптамалық деректер" кестесінің жазбаларын бағыттауыш бірінші
жазбаға келтіріледі DoUntilRst3.EOF' Ішкі цикл. Циклде
"Сараптмаалық деректер" кестесінің жазбалары ретімен таңдалады.
If (Rst1!Құрастыру LikeRst3!Құрастыру) And(Rst1!ТұрақтыКод
LikeRst3!ТұрақтыКод) And^П!ҚалыпКод LikeRst3!ҚалыпКод) Then
```

Fj = True' Егер кестелердегі жазбалар сәйкес келсе, табысты іздеу таңбасы орнатылады.'

Rst2.AddNew' "Талдау нәтижелері кестесіне жаңа жазба қосылады"'

Rst2!Талдау = ^Ч!Құрастыру) & "." & ^Ч!ТұрақтыКод) & "." & ^ШҚалыпКод) 'Талданатын бөлшектің технологиялық коды туралы ақпарат бар,'

Rst2!Эксперт = ^3!Констр) & "." & ^3!ТұрақтыКод) & "." & (Rst3!ҚалыпКод) 'Талданатын бөлшек пен аналог сызбаларының нөмірлері мен аналог коды'

Rst2!СызбаНомеріАн = Rst1! СызбаНомері Rst2!

СызбаНомеріЭксп = Rst3! СызбаНомеріа

Rst2!Пайыз = "100 %"

Rst2!Еңбексыйымдылығы = Rst3! Еңбексыйымдылығы 'Аналог бөлшектің әзірленуінің еңбексыйымдылығы.'

Rst2.Update

End If

Rst3.MoveNext

Loop

If Fj = False Then 'Егер бұрынғы іздеу циклінде аналогтың бөлшегі табылмаса, онда екінші іздеу орындалады. Бұл жағдайда технологиялық кодтың кейбір позициялары ескерілмейді'
Rst3.MoveFirst 'Ішкіцикл. Циклде "Сараптамалық деректер" кестесінің жазбалары ретімен таңдалады'

Do Until Rst3.EOF 'ІздеумаскасыGetStringJ функциясына құрылады. Бұл жағдайда бөлшекті дайындаудың технологиялық әдісі ескеріледі.'

If (КйВҚұрастыру = К5£3!Құрастыру) And (((КйРТұрақтыКод) &(Rstl! Қалып- Код)) Like GetStringJ((Rst3!ТұрақтыКод) & (Rst3!ҚалыпКод))) Then Fj = True 'Кестелердегі жазбалар сәйкес келсе, табысты іздеу орнатылады..'

Rst2.AddNew '"Талдау нәтижелері кестесіне жаңа жазба қосылады"'

Rst2!Талдау = (Rst1!Құрастыру) & "." & (Rst1!ТұрақтыКод) & "." & (Rst1!ҚалыпКод) 'Талданатын бөлшектің технологиялық коды туралы ақпарат бар.,'

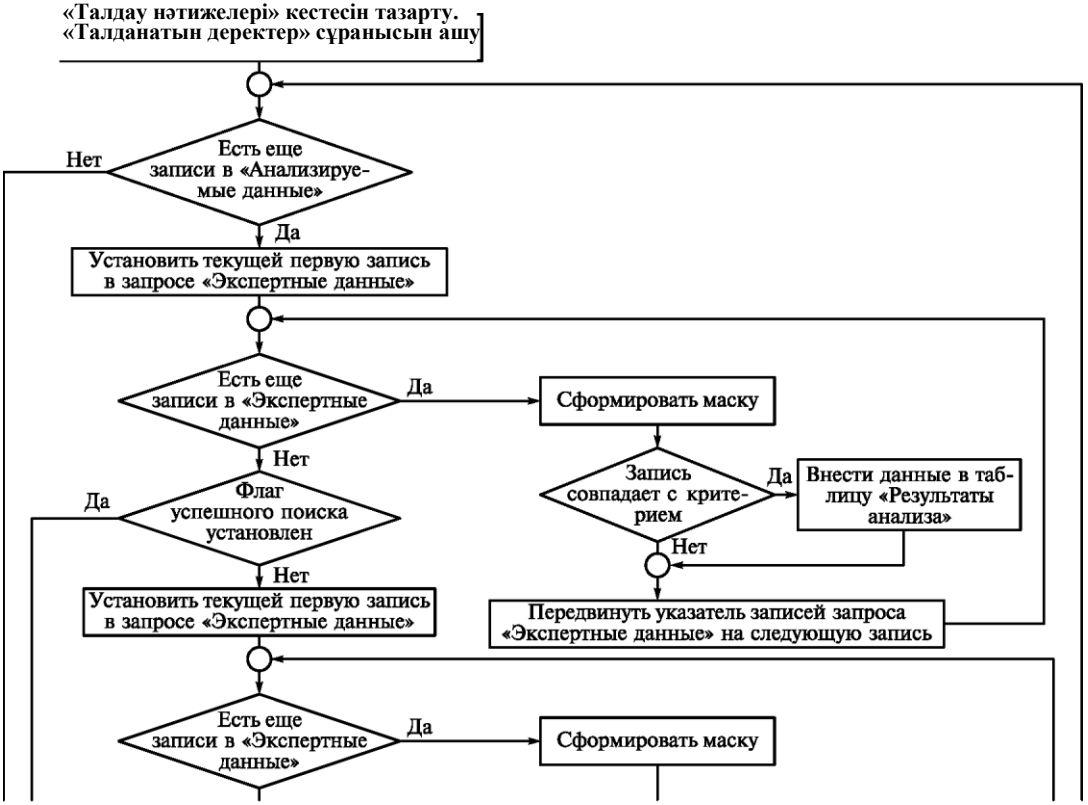
Rst2!Эксперт = (Rst3!Құрастыру) & "." & (Rst3!ТұрақтыКод) & "." & (Rst3!ҚалыпКод) 'Талданатын бөлшек пен аналог сызбаларының нөмірлері мен аналог коды'

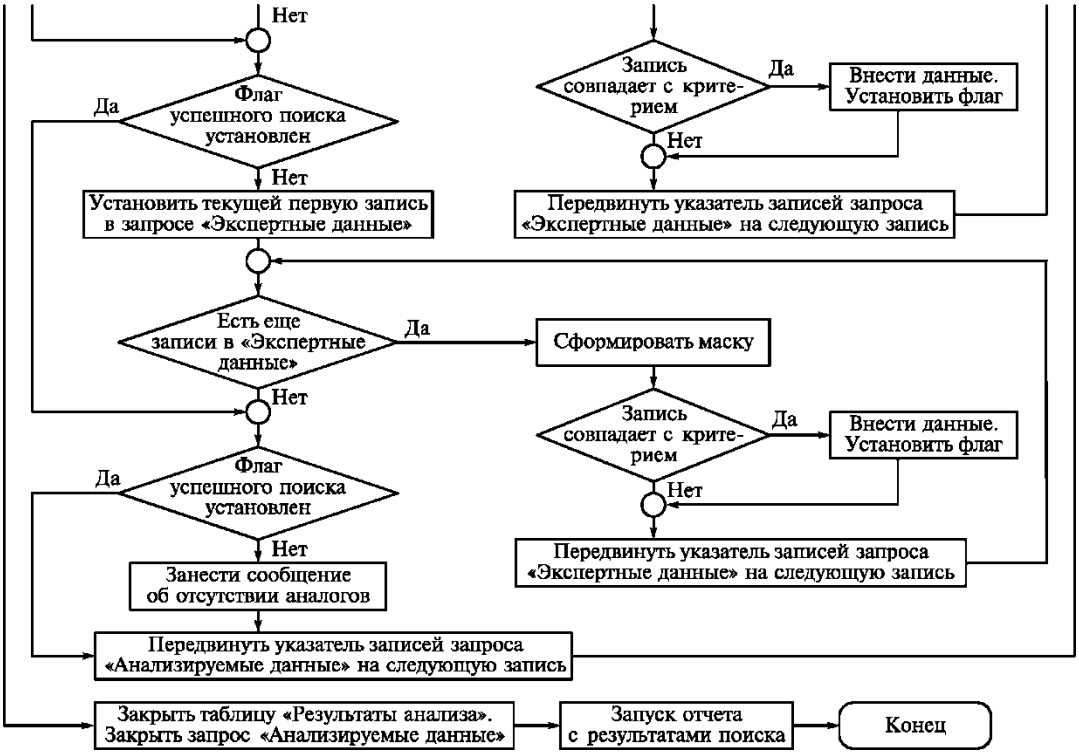
Rst2!Пайыз = "80 %"

Rst2!СызбаНөміріАн = Rst1!СызбаНөміріRst2!СызбаНөміріЭксп= Rst3!СызбаНөмірі Rst2!Еңбексыйымдылығы = Rst3!Еңбексыйымдылығы 'Аналог бөлшектің әзірленуінің еңбексыйымдылығы.'

Rst2.UpdateEndIf

Басы


«Талдау нәтижелері» кестесін тазарту.
«Талданатын деректер» сұранысын ашу
Есть еще записи в «Анализируемые данные»
Нет
Да
Установить текущей первую запись в запросе «Экспертные данные»
Есть еще записи в «Экспертные данные»
Да
Сформировать маску
Нет
Запись совпадает с критерием
Да
Внести данные в таблицу «Результаты анализа»
Нет
Флаг успешного поиска установлен
Да
Нет
Установить текущей первую запись в запросе «Экспертные данные»
Передвинуть указатель записей запроса «Экспертные данные» на следующую запись
Есть еще записи в «Экспертные данные»
Да
Сформировать маску



18.22 сур. Бөлшектерді жасаудың еңбексыйымдылығына сараптамалық бағалау жүргізу алгоритмінің блок- сұлбасы

```
Rst3.MoveNext
Loop
End If
If Fj = False Then
Rst3.MoveFirst Do Until
Rst3.EOF
If (Rst1!КоНСхр = Rst3!КоНСТр) And ((^және!ТұрақтыКод) &
(Rst1! Қалып-Код)) Like GetStringS((Rst3!ПостКод) & (Rst3! Қалып-
Код))) Then Fj = True Rst2.AddNew
Rst2!АНann3Нр = (Rst1!Құрастыру) & "." &^және!ТұрақтыКод) &
"." &^Ч!ҚалыпКод)
Rst2!3КсперТ = (Rst3!Құрастыру) & "." &^3!ТұрақтыКод) & "." &
(Rst3!ҚалыпКод)
Rst2!Пайыз = "60 %"
Rst2! СызбаНөмірiАн = Rst1!СызбаНөмірі Rst2!СызбаНөмірiЭксп =
Rst3!СызбаНөмірі Rst2!Еңбексыйымдылығы =
Rst3!Еңбексыйымдылығы Rst2.Update End If
Rst3.MoveNext
Loop
End If
If Fj = False Then Rst3.MoveFirst Do Until Rst3.EOF
If^және!Құрастыру= Rst3!Құрастыру) And((^және!ТұрақтыКод) &
(Rstl! ҚалыпКод)) Like GetStringK((Rst3!ТұрақтыКод) & (Rst3! Қалып-
Код))) Then Fj = True Rst2.AddNew
Rst2!АНann3Нр = ^және!Құрастыру) & "." &^және!ТұрақтыКод) &
"." &^Ч!ҚалыпКод)
Rst2!3КсперТ = ^3!Құрастыру) & "." &^3!ТұрақтыКод) & "." &
(Rst3!ҚалыпКод)
Rst2!Пайыз = "50 %"
Rst2!СызбаНөмірiАн= Rst1! СызбаНөмірі
Rst2!СызбаНөмірiЭксп= Rst3!СызбаНөмірі
Rst2!Еңбексыйымдылығы = Rst3!Еңбексыйымдылығы
Rst2.UpdateEndIf
Rst3.MoveNext
Loop
EndIf
```

IfFj = FalseThen'Егер де бөлшектің аналогы табылмаса, онда "Талдау нәтижелері" кестесіне'

Rst2.AddNew'Берілген бөлшектің аналогы жоқ деген мәліметті енгіземіз'

Rst2!Талдау = ^Н!Құрастыру) & "." & ^Н!ТұрақтыКод) & "." & ^Н!ҚалыпКод)

Rst2!Эксперт = "табылмады"

Rst2!СызбаНөміріАн = Rst1!СызбаНөмірі

Rst2!Пайыз = "0 %"

Rst2.Update

End If

Rstl.MoveNext

Loop

Rst1.Close "'Талданатын деректер" кестесінің мәліметтеріне жол берілетін ашық объектіні жабады.'

Rst2.Close "Талдау нәтижелері" кестесінің мәліметтеріне жол берілетін ашық объектіні жабады.'

Rst3.Close "'Сараптамалық деректер" кестесінің мәліметтеріне жол берілетін ашық объектіні жабады.'

stDocName = "Талдау нәтижелері"

DoCmd.OpenReport stDocName, acPreview "'Талдау нәтижелері" кестесінің негізінде есептеме құрады'

Ехй_Батырма1_СИск:

Exit Sub

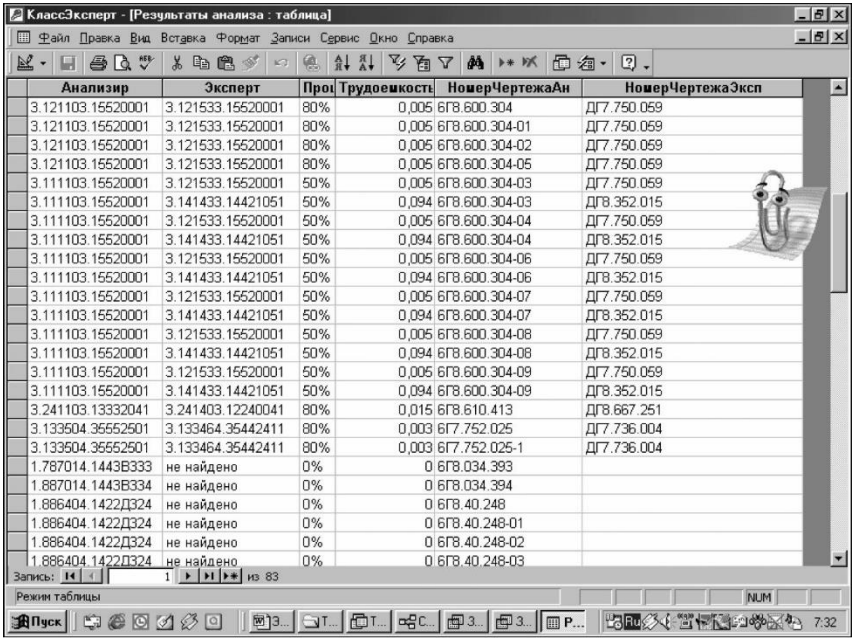
Public Function GetStringJ(Expert As String) As String GetStringJ = Mid(Expert, 1, 3) & "??" & Mid(Expert, 6, 2) & "???????" End Function

Public Function GetStringS(Expert As String) As String GetStringS = Mid(Expert, 1, 2) & "???" &Mid(Expert, 6, 2) & "???????" End Function

Public Function GetStringK(Expert As String) As String GetStringK = Mid(Expert, 1, 1) & "?" & Mid(Expert, 1, 1) & "??" & Mid(Expert, 6, 2) & "???????"

End Function

18.23 суретінде жаңа бөлшектерді дайындаудың еңбексыйымдылығын сараптамалық бағалау нәтижелерінің кестесінің фрагменті көрсетілген. Осы кестеден бірнеше мәліметтердің дәлдігі 80% дәлдікпен, 50% дәлдікпен бірнеше бөлшектердің және кейбір мәліметтердің аналогтары табылмаған болжанатын еңбексыйымдылығын бағалауы керек.


КлассЭксперт - [Результаты анализа : таблица]

Файл Правка Вид Вставка Формат Записи Сервис Окно Справка

Анализир	Эксперт	Прог	Трудоемкость	НомерЧертежаАн	НомерЧертежаЭксп
3.121103.15520001	3.121533.15520001	80%	0,005	6Г8.600.304	ДГ7.750.059
3.121103.15520001	3.121533.15520001	80%	0,005	6Г8.600.304-01	ДГ7.750.059
3.121103.15520001	3.121533.15520001	80%	0,005	6Г8.600.304-02	ДГ7.750.059
3.121103.15520001	3.121533.15520001	80%	0,005	6Г8.600.304-05	ДГ7.750.059
3.111103.15520001	3.121533.15520001	50%	0,005	6Г8.600.304-03	ДГ7.750.059
3.111103.15520001	3.141433.14421051	50%	0,094	6Г8.600.304-03	ДГ8.352.015
3.111103.15520001	3.121533.15520001	50%	0,005	6Г8.600.304-04	ДГ7.750.059
3.111103.15520001	3.141433.14421051	50%	0,094	6Г8.600.304-04	ДГ8.352.015
3.111103.15520001	3.121533.15520001	50%	0,005	6Г8.600.304-06	ДГ7.750.059
3.111103.15520001	3.141433.14421051	50%	0,094	6Г8.600.304-06	ДГ8.352.015
3.111103.15520001	3.121533.15520001	50%	0,005	6Г8.600.304-07	ДГ7.750.059
3.111103.15520001	3.141433.14421051	50%	0,094	6Г8.600.304-07	ДГ8.352.015
3.111103.15520001	3.121533.15520001	50%	0,005	6Г8.600.304-08	ДГ7.750.059
3.111103.15520001	3.141433.14421051	50%	0,094	6Г8.600.304-08	ДГ8.352.015
3.111103.15520001	3.121533.15520001	50%	0,005	6Г8.600.304-09	ДГ7.750.059
3.111103.15520001	3.141433.14421051	50%	0,094	6Г8.600.304-09	ДГ8.352.015
3.241103.13332041	3.241403.12240041	80%	0,015	6Г8.610.413	ДГ8.667.251
3.133504.35552501	3.133464.35442411	80%	0,003	6Г7.752.025	ДГ7.736.004
3.133504.35552501	3.133464.35442411	80%	0,003	6Г7.752.025-1	ДГ7.736.004
1.787014.1443В333	не найдено	0%	0	6Г8.034.393	
1.887014.1443В334	не найдено	0%	0	6Г8.034.394	
1.886404.1422Д324	не найдено	0%	0	6Г8.40.248	
1.886404.1422Д324	не найдено	0%	0	6Г8.40.248-01	
1.886404.1422Д324	не найдено	0%	0	6Г8.40.248-02	
1.886404.1422Д324	не найдено	0%	0	6Г8.40.248-03	

Запись: 1 из 83

Режим таблицы NUM

Пуск 7:32

18.23 сур. Жаңадан әзірленген бөдшектердің еңбексыйымдылығын сараптамалық бағалау нәтижелері көрсетілген кестенің фрагменті

Бұл жүйе Мәскеудің екінші аспап жасау зауытында сыналды. Зауыт өнімдерінің біреуі аналогтық үшін қабылданған бірқатар эксперименттер жүргізілді. Оның әрқайсысы мәліметтер қорына кодталған және енгізілген.

Басқа өнім бағаланған өнім ретінде қолданылды. Тәжірибе барысында талданатын өнімді өндірудің еңбексыйымдылығын бағалау және оны өндірудің еңбексыйымдылығының болжамының дұрыстығын бағалау міндетті болды.

Нәтижелер әзірленген сараптамалық жүйе өндірістік бөлшектердің еңбексыйымдылығын кемінде ± 20% дәлдікпен болжауға мүмкіндік беретінін көрсетті.

Бақылау сұрақтары

1. Сызбаларды параметрлік жобалаудың мәні неде?
2. Параметрлік сызбалау жүйелеріндегі деректер қорының қолданылу мақсаты неде?
4. ДҚБЖ қолдану арқылы сараптамалық жүйелерді жасаудың қандай салаларын айта аласыз?

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1. *Богтс У., Боггс. М.*UML және RationalRose: Пер. ағылш. — М.: Лорри, 2000.
2. *Бусленко Н.П., Калашников В. В., Коваленко И.Н.* Лекции по теории больших систем. — М.: Сов. радио, 1973.
3. *Буч Г., Рамбо* Д., *Джекобсон А.* Язык UML. Руководство пользователя: Ағылш. ауд. — М.: ДМК, 2000.
4. *Вейскас Д.* Эффективная работа с MicrosoftAccess2: Ағылш. ауд. — СПб.: Питер, 1995.
5. *Горев* А., *Ахаян* Р., *Макашаринов С.* Эффективная работа с СУБД. — СПб.: Питер, 1997.
6. *Карпова Т. С.* Базы данных: модели, разработка, реализация. — СПб.: Питер, 2001.
7. *Кирстен В., Иринге*р *М.* СУБД Cache— объектно-ориентированная разработка приложений. — СПб.: Питер, 2001.
8. *Китов А. И.* Программирование информационно-логических задач. — М.: Сов. радио, 1967.
9. *Маклаков С. В.*CASE-средства разработки информационных систем. — М.: Диалог — МИФИ, 2000.
10. Тихомиров Ю. MicrosoftSQLStrver7.0. — СПб.: BNV— Санкт-Петербург, 1999.
11. Толковый словарь по вычислительным системам / Под ред. В. Иллингуорта и др.; Ағылш. ауд.. А. К. Белоцкого и др.; Ред. Е. К. Масловского. — М.: Машиностроение, 1990.
12. *Фуфаев Э.*В., *Фуфаева Л.И.* Пакеты прикладных программ. — М.: Издательский центр «Академия», 2004.
13. *Харитонова* И., *Михеева В.*MicrosoftAccess2000. — СПб.: БХВ — Петербург, 2001.
14. *Хорафас* Д., *Легг С.* Конструкторские базы данных / Ағылш. ауд. Д. Ф. Миронова. — М.: Машиностроение, 1990.
15. *Шапошников И.*Web-сервисы Microsoft.Net. — СПб.: БХВ — Петербург, 2002.

МАЗМҰНЫ