

ИНФОРМАТИКА МАМАНДАРЫН ДАЯРЛАУДЫ ЖЕТІЛДІРУДЕ ҚОЛДАНЫЛАТЫН ЭЛЕКТРОНДЫҚ ОҚУ- ӘДІСТЕМЕЛІК ҚҰРАЛ

Меруерт Серікқызы МӘЛІБЕКОВА

Л.Н.Гумилев атындағы Еуразия ұлттық университетінің
педагогика ғылымдарының докторы

Ғалымжан Есенгелдыұлы ӨТЕҒҰЛОВ

Л.Н.Гумилев атындағы Еуразия ұлттық университетінің ізденушісі

Л.Н.Гумилев атындағы Еуразия ұлттық университетінің математика және ақпараттық технологиялар факультетінің «Информатика», «Ақпараттық жүйелер», «Есептеуіш техника және программалық камсыздандыру» мамандықтарының студенттеріне арналып жасалған ақпараттық – дидактикалық кешен білім мазмұнын арттыру мақсатында жасалған электронды оқу-әдістемелік кешен (сурет 1).

Ақпараттық қызметтердің кеңеюіне байланысты интеллектуалды, көлемді бағдарламалық кешендер пайда бола бастады. Көптеген ұйымдар мен жеке қолданушылар бағдарламалық жабдықтары мен деректерін желіде тиімді ұйымдастырудың жолдарын жасады, яғни ақпарат алмастыру мен оны жіберу үшін желілерде оңтайлы ұйымдастыру құрастырылды, жалпы бағдарламалық жабдықтар мен деректерді файлдық серверлерде орналастыра бастады. Суретте



көрсетілгендей кешен информатика мамандығының оқу жоспарын қарай қазақ-орыс тілдерінде пәндерді қамтып жасалған Тақырыбымызға қарай аталған кешеннен мысал ретінде алсақ, клиент сервер технологиясына арналған пәннің маңыздылығын келтіруге болады.

Деректер қорын басқару жүйесінің файлдық серверлерінің көмегімен көптеген пайдаланушылар бір деректер қорымен жұмыс істей алатын болғанын білеміз. Бағдарламаларда сұрауларды өңдеу қолданушының ЭЕМ-да орындалады, егер де қарапайым сұранысты файлдық серверден алу керек болса, бүтін файлды компьютерге көшіру керек болды, ол желінің жұмысының артып кетуіне, конфликтілі жағдайларға әкелді. Мұндай кемшіліктерді жою үшін клиент-сервер технологиясы ұсынылды, оны қазіргі таңда бірден-бір сұраныстар ұйымдастырушы SQL тілі орындайды және ол туралы кешеннің мазмұнынан толық, сапалы материалдар алуға болады.

Бөлінетін файлдар негізіндегі архитектура дербес компьютерді локальді желіге файлдық сервер арқылы біріктіре бастағанда белгілі бола бастады. Бұл архитектура өткен ғасырдың 80-жылдары компьютерлерді желіге жаппай біріктіру кезінде пайда болды. Мұндай архитектураның мәні - әрбір дербес компьютерде файлдық серверде тұрған, жалпы файлдарды қолданатын, қосымша жүктеледі. Әрбір клиент өзінің жеке деректері сияқты ортақ деректермен жұмыс істейді. Бірақ, мұндай көпқолданушы жүйесінің кемшіліктері бар: бірнеше қолданушы интенсивті жұмыс істегенде желідегі трафик өседі; локальді желінің өткізу мүмкіншілігінің шектеулі екендігін ескерсек, жүйенің реакциясының уақыты көп болады; одан кейін желінің басқа жұмысының жылдамдығы азаяды [1].

Деректер қорын ұйымдастырудың файл-сервер технологиясының келесі бір кемшіліктері:

- деректер қорының тұтастығын сақтау мен қолдаудағы қиындықтар;

- бір қолданушы деректермен жұмыс істеп отырғанда немесе оны өзгерткен жағдайда олар басқа қолданушылардың деректер қорларының локальді көшірмелеріне, бірінші қолданушы файлды жаппайынша бірден бармайды, сондықтан кез келген қолданушы уақытша бұрынғы ақпаратты ғана көре алады;

- деректер қорын пайдалануда қолданушылардың жұмыстарын параллель жүргізу қажет болады, бір қолданушы жұмыс істеп жатқан

жазбаларға, бағандарға басқа қолданушылар үшін блокировка қою керек, яғни уақытша немесе мүлдем тыйым салу керектігі туады;

- деректер қорымен жұмыс істеуде қолданушылар иерархиясын құру қажеттігі, деректерді алуда бакылау жасауды ұйымдастыру, қолданушыны идентификациялау сияқты процедураларды ұйымдастыру қажеттігі туды;

- аппарат істен шыға қалса (көбінесе тоқ көзінің аяқ астынан өшуі), деректердің дұрыстығын тексеру қиындайды. Мұндай жағдайда аппаратты жөндеу мен қайта қалпына келтіру жүйе администраторының жұмысы болады.

Аппараттың істен шығуында келесі бір проблемалар бар – жабылмаған транзакциялардың проблемасы болды.

Ақпараттық жүйеде кейбір іс-әрекеттер топпен орындалуы керек. Мысалы, бір кестеден деректі екінші кестеге ауыстыру кезінде екі операция орындалады: 1) деректі алу; 2) екінші кестеге қосу амалы. Егер аппарат осы екі амал ортасында істен шықса, онда деректер жоғалып кетеді. Осындай кемшіліктерді ескере келе, файл-сервер архитектурасының негізіндегі қосымшаларды өзгерту қажеттігі туды. Мысалы, желілік операциялық жүйелерде транзакцияларды қадағалау, қосымшамен көпқолданушылық режимінде бір кезде жұмыс істеу үшін блокировка қою. Қосымшалардың басқа ортаға (UNIX немесе басқа версияға) өту мүмкіншілік болмады. Бұл механизмді әр түрлі қосымшалар арасында сәйкестендіру керек болды, мысалы есеп құру, деректер енгізу, деректер мәндерін администрациялау т.б. Бұл жүйенің икемсіздігін және ауырлығына әсер етті. Бұл архитектура ақпараттық жүйелер үшін, аз қолданушылар үшін, аз көлемді деректер және арзан деп есептелетін ақпарат үшін қолданыла береді, ал үлкен көлемді ақпарат, көп қолданушылар үшін проблема туғызуы мүмкін.

Кезінде "терминалды" немесе "хост-терминал" архитектурасы қолданылды. Файлдық сервер архитектурасында деректер қолданушыға "жақын" болса, терминалды архитектурада қолданушының жұмыс орнында (терминалда) ақпараттың тек физикалық бейнесі болады және енгізуі орындалады, ал қосымшаның барлық логикасы, барлық деректер орталық компьютерде (хоста) сақталады. Мұндай архитектура үлкен компьютерлердің (мейнфреймдер) және 70-жылдар соны мен 80-жылдардың басында болды. Терминалды ақпараттық жүйені администрациялау және қадағалау қымбатқа түсті, ал компьютерлер арзан болды.

Ақпараттық жүйелердің логикалық жетістігі “клиент-сервер архитектурасы болды. Клиент-сервер архитектурасы бағдарламалардың өзара байланысын сипаттауға негізделген.

Файл-сервер архитектурасында деректерді өңдейтін бағдарламалар мен деректердің өздерін келесі компьютерге “алып бару” керек болды. Терминалды архитектурада қолданушы мен өңдейтін бағдарламаның алшақта жатуы немесе жекеленіп қалуы орын алды.

Клиент-сервер архитектурасы бұл кемшіліктерді жояды бағдарламалық компоненттердің тұтас жүйесін құрастырады. Атындағы байланысты белгілі бір ресурсты қажет ететін клиенттік процесс бар сонымен бірге сол ресурстармен қамтамасыз ететін серверлік процесс бар. Олардың бір компьютерде болуы міндетті емес.

Деректер қоры контекстінде клиент жұмыс станциясы ретінде қолданушы интерфейсі және қосымшалардың логикасын басқарады. Клиент қолданушыдан сұранысты алады, SQL немесе басқа тілде синтаксисін тексереді және сұранысты генерациялайды. Содан кейін серверге хабар жібереді, жауап күтеді және қолданушы үшін алынған деректерді құрастырады. Сервер сұраныстарды қабылдап алады және өңдейді, содан кейін қайтадан алынған нәтижені клиентке жібереді. Мұндай өңдеу деректердің тұтастығын және деректерді толықтырады. Бұдан басқа параллель басқару және қалпына келтіру жүріп отырады.

Клиент–сервер архитектурасы арқылы:

- көптеген қолданыста жүрген деректер қорына кең қолданыс табуға болады;
- аппараттық қамтамасыздандырудың бағасы азаяды, деректерді сақтау мен басқаруда үлкен қуатты компьютер тек сервер үшін ғана қажет болады;
- коммуникациялық шығын азаяды, қосымшалар операциялардың бір бөлігін клиент компьютерлерде орындайды және деректер қорына сұраныс орындағанда ғана желімен жібереді, одан желі бойынша жіберілетін деректер көлемі азаяды;
- деректердің “қарсылықтары” азаяды; сервер деректер тұтастығын өз бетінше басқарады, себебі сонда ғана шектеулер анықталады және тексеріледі әр қосымша өз тексеруін орындамайды;
- клиент–сервер архитектурасында ашық жүйелер бейнеленеді [2].

Сонымен бірге клиент-сервер архитектурасының келесі артықшылықтарын да ажыратуға болады:

- бүкіл деректер қоры емес, тек қажетті ақпаратты желіде жіберу, яғни желінің трафигін төмендету;
- деректер қорына қажет жадыны бөлу, қажет жағдайда жадыны бірқалыпты өсіріп отыруды тиімді ұйымдастыру;
- деректер қауіпсіздігін сақтау, файлдарының «жасырын» түрде сақталуы, тек белгілі бір бағандар тобымен ғана жұмыс істеу құқығын беру;
- уақытша виртуальды объектілер құру;
- файлдарды индекстеуде оңтайлы жолдарын таңдау;
- көлемі аз бағдарламалық код арқылы үлкен деректер массивімен жұмыс істеу;
- деректер тұтастығын сілтемелер арқылы қолдау;
- триггер, транзакциялар, тосқауылдар құру мүмкіншілігі;
- қолданушылардың жеке топтары үшін ақпаратты алу құқығын орнату мүмкіншілігін қамтамасыз ету;
- деректер қорымен иерархиясына байланысты қолданушылардың бір уақытта жұмыс істеу мүмкіншілігі.

Бір деректер қорымен бірнеше қолданушы бір мезгілде жұмыс істегісі келсе, экранда көре беру мүмкіншілігі бар, ал егер бір (немесе бірнеше) қолданушы деректерді өзгерткісі келсе, проблема туатын жағдайлар болады. Сондықтан да көпқолданушы режимінде деректерге жетуде блокировка механизмі қойылады, ол біреу оқып отырған, бірақ әлі өңделмеген деректерді өзгертуге тыйым салады, кейде біреулер оқыған мәндерді (әлі өңделмеген) тығып қояды. Мұндай тосқауыл қою механизмі барлық қолданушылар үшін ортақ. Ол механизм:

Біріншіден, бағдарлама қолданушымен интерфейсті қолдайтын функциялардан және дискідегі деректермен жұмыс істейтін функциялардан тұрады. Егер бірнеше қолданушы бұл бағдарламаны бір мезгілде жіберсе, әрқайсысы үшін зердеге осы екі функция да жүктеледі. Бұл функцияларды барлықтары қолданатындықтан, оларды бағдарламадан бағдарлама-координаторға көшіріп қою ресурстарды үнемдейді.

Екіншіден, деректерді біреу бұзғысы келсе, қателер пайда болса бағдарламашы деректердің тұтастығы, функцияларға көңіл бөлуі дұрыс жүріп отыруына сенімді болуы керек. Сондықтан да деректерге жетуді жүзеге асыру құралдарын деректерге жақын жерге «тығып қою» керек. Оны бағдарлама-координатор орындайды.

Үшіншіден, жаңа қолданбалы бағдарламалар жасаған кезде мысалы басқа платформа немесе басқа тілде жазғанда жоғарыдағы деректерге жету функцияларын қайта тексеріп, қайтадан орындап жүзеге асыру керек. Бұл қателер кетуі мүмкін деген сөз, бағасы өседі бүкіл жүйенің жұмысын өзгертуге әкеледі [1, Б.17]. Сонымен біз интерфейспен жұмыс істейтін бір бағдарлама орнына екі байланысқан бағдарламаны – қолданушымен жұмыс істейтін (бағдарлама-клиент) және деректермен жұмыс істеу бағдарламасы (бағдарлама-сервер) яғни клиент-сервер архитектурасында жұмыс істейтін жүйені қарастырдық.

Деректер қорының бағдарлама-сервері деректерге жетудің негізгісі болып есептеледі. Сондықтан да функцияның негізгі бөлігі серверге жатады. Деректер қорының бағдарлама-сервері орындайтын негізгі функциялар:

1) деректерді алу үшін және модификациялау үшін клиенттің сұрауларын орындайды;

2) бірнеше қолданушының деректермен жұмыс істеуін қамтамасыз етеді;

3) қолданушыларды идентификациялауды қамтамасыз етеді және әр қолданушыны әртүрлі деректермен жұмыс істеуіне шектеу қояды;

4) аппараттық және бағдарламалық істен шыққанда деректердің тұтастығын қамтамасыз етеді;

5) деректерді санкцияланбаған қолданудан сақтайды;

6) аппараттық жүйелерді әкімшілеуде көмекші құралдарды пайдалануға мүмкіндік болады.

Аталған ойлар реляциялық деректер қорының математикалық қатынастар теориясына негізделетінін, яғни қуатты теориялық фундаментке сүйенетінін негіздейді. Сонымен бірге қатынастар теориясы мен қатынастарға қолданылатын теориялық-жиындық операцияларға негізделген.

Қолданған әдебиеттер тізімі:

1. Грачев А.Ю. INFORMIX. Введение в СУБД. – М.: ДИАЛОГ МИФИ, 2000. – 272 с.
2. Полякова Л.Н. Основы SQL. – М.: Интернет-университет информационных технологий. – 366 с.