Үлкен деректер (Big Data) ұғымы

Сайлауқызы Ж. – *КжЖИ кафедрасының қауымдастырылған профессоры*
Қазкен Г. - *КжЖИ кафедрасының оқытушысы*

1. Big Data ұғымы

Үлкен деректер (Big Data) – бұл **классикалық әдістермен өңдеуге болмайтын, көлемі үлкен, әртүрлі және үздіксіз жаңартылатын деректер жиынтығы**. Оған мәтін, сурет, видео, аудио, сенсорлық деректер, транзакциялар және әлеуметтік желідегі ақпарат кіреді.

Үлкен деректерді өңдеу үшін арнайы **технологиялар мен құралдар** қажет, олар деректерді жинау, сақтау, өңдеу және талдауға мүмкіндік береді.

Big Data сипаттамаларының негізі ретіндегі 5V-модель

Big Data – бұл көлемі үлкен, әртүрлі, үздіксіз жаңартылатын және сенімді деректер жиынтығы, ал 5V моделі осы деректердің көлемін (Volume), жылдамдығын (Velocity), түрлілігін (Variety), сенімділігін (Veracity) және олардың бизнеске қосатын пайдасын (Value) сипаттайды.

Volume: Big Data дәуіріндегі деректер көлемі

Volume (көлем) – ұйымдарды «селдей» басып өтетін мәліметтердің молдығын сипаттайды. Бұрын компаниялар деректерді өз серверлерінде сақтап, ішкі ресурстарын пайдаланды.

Velocity

Үлкен деректердің көлемі артқан сайын, олардың ағым жылдамдығы да маңызды болады. Velocity – деректердің үздіксіз ағынын, оларды жинау, өңдеу және талдау жылдамдығын сипаттайды.

Үздіксіз деректер ағымымен жұмыс жасау арқылы ұйымдар:

- құнды ақпаратты нақты уақытта ала алады,
- шешім қабылдауды жылдамдатады,
- нарықтағы өзгерістерге дер кезінде жауап береді,
- операциялық процестердің тиімділігін арттырады.

Variety (Түрлілігі)

Үлкен деректер әртүрлі формада болады: **құрылымдалған, жартылай құрылымдалған және құрылымдалмаған**.
Барлық деректер көздерін тиімді талдау (big data analytics) ұйымдарға:
•өз ортасын және компанияны жақсы түсінуге,
•шешім қабылдауды оңтайландыруға,
•клиент қажеттіліктерін қанағаттандыруға,
•шығындарды азайтуға,
•процестерді оңтайландыруға,
•жаңа табыс көздерін табуға мүмкіндік береді.

Сенімділігі (Veracity)

Бұл сипаттама екі аспектіні қамтиды:
1.Статистикалық сенімділік – деректердің дәлдігі мен тұрақтылығы.
2.Деректердің сенімділігі – деректердің шығу көзі, жинау және өңдеу әдістері, инфрақұрылым мен қолданылған компьютерлік құралдарға байланысты.
Veracity үлкен деректердің **сенімді, нақты және рұқсатсыз өзгерістерден қорғалғанын** қамтамасыз етеді. Бұл ақпаратты дұрыс талдауға және дұрыс шешім қабылдауға мүмкіндік береді.

Value (құндылық) – Үлкен деректердің ең соңғы қасиеті – олардың шешім қабылдау, бизнес процестер немесе аналитика үшін қосатын құндылығы мен пайдалы болуы.Деректерді пайдалы ету үшін оны білімге айналдыру қажет. Мұны орындау үшін:деректерді талдау (data mining),болжамдық аналитика (predictive analytics),мәтінді талдау (text mining) сияқты технологиялар қолданылады.Бұл үш негізгі бизнес мақсатқа жетуге көмектеседі:шығындарды азайту,жылдам және тиімді шешім қабылдау,жаңа өнімдер мен қызметтерді жобалау.

.

1.2 Деректер түрлері мен көздері

Сайлауқызы Ж. – *КжЖИ кафедрасының қауымдастырылған профессоры*
Қазкен Г. - *КжЖИ кафедрасының оқытушысы*

Кіріспе

- Қазіргі заманда деректер – кез келген ақпараттық жүйенің, ұйымның және цифрлық экономиканың негізгі ресурсы болып табылады. Деректерді дұрыс жинау, сақтау және өңдеу арқылы ғана тиімді шешім қабылдауға болады. Сондықтан деректердің түрлерін және олардың қайдан алынатынын түсіну Big Data технологияларын меңгерудің негізі болып табылады.
- Ауызша түсіндіру үшін:Бұл тақырыпта біз деректердің қандай болатынын, олардың қайдан пайда болатынын және неге оларды әртүрлі тәсілмен сақтау керек екенін қарастырамыз

Деректер (Data)

Деректер (Data) – белгілі бір объект, құбылыс немесе процесс туралы тіркелген мәліметтер жиынтығы. Олар сандар, мәтін, сурет, бейне, дыбыс немесе өлшеу нәтижелері түрінде болуы мүмкін.

Маңызды ұғым:

- **Дерек** – бастапқы, өңделмеген мәлімет.
- **Ақпарат** – өңделген, мағынасы бар дерек.

Мысал: «85» – дерек. «Студент емтиханнан 85 балл алды» – ақпарат.

«Қазіргі жүйелердегі деректерді үш негізгі категорияға бөлуге болады: құрылымдалған, жартылай құрылымдалған және құрылымдалмаған деректер.

Құрылымдалған деректер – қатты схемаға ие деректер. Олар кестелерде көрсетіледі, әр жол – жеке жазба, ал бағандар өрістерді анықтайды. Мысалдар: банк транзакциялары, клиенттер туралы ақпарат, ERP жүйесіндегі деректер. Артықшылығы – сақтау, іздеу және талдау оңай. Бірақ масштабтау немесе әртүрлі деректер көздерімен жұмыс істеу қиын.

Жартылай құрылымдалған деректер – ішінара белгіленген құрылымы бар деректер. Көбіне JSON немесе XML форматында болады, өзгермелі құрылымды ақпаратты сақтауға мүмкіндік береді. Мысалдар: веб-сервер логтары, API жауаптары. Артықшылығы – икемділік, аналитикалық жүйелерге оңай интеграцияланады. Кемшілігі – құрылымдалған деректерге қарағанда талдауы күрделірек.

Құрылымдалмаған деректер – айқын схемасы жоқ деректер. Бұл – бейнелер, аудио, видео, мәтіндік құжаттар және әлеуметтік желіден алынған деректер. Артықшылығы – кез келген ақпаратты сақтау мүмкіндігі. Кемшілігі – дәстүрлі әдістермен талдау, іздеу және сақтау қиын.

Деректер әртүрлі көздерден келеді:

Ішкі көздер – ұйым ішіндегі деректер, мысалы, CRM немесе ERP жүйелері, клиенттер мен транзакциялар туралы ақпарат.

Сыртқы көздер – сыртқы ортадан алынған деректер: әлеуметтік желілер, ашық деректер, веб-ресурстар, ақылы аналитикалық қызметтер.

Ағынды көздер – үздіксіз келетін деректер, мысалы, IoT сенсорлары, сервер логтары немесе мониторинг жүйелері.

Мультимедиа көздері – бейне, аудио және сурет түрінде келетін деректер, камералардан, қолданушылар қосымшаларынан және сенсорлық құрылғылардан алынған.»

«Дерек көзін түсіну – интеграция, сақтау және талдау технологияларын дұрыс таңдауға мүмкіндік береді.»

«Үлкен деректермен жұмыс кезінде кездесетін негізгі мәселелер:

- **Деректер көлемі**: дәстүрлі ДҚБЖ-де сақтау және өңдеу қиын.
- **Форматтардың әртүрлілігі**: мәтін, кестелер, суреттер, видео әрқайсысына әртүрлі технология қажет.
- **Келу жылдамдығы**: ағындық деректерді нақты уақытта өңдеу керек.
- **Сенімділік және дәлдік**: ақпараттың дұрыстығы мен дәлдігін қамтамасыз ету маңызды.»

Қазіргі цифрлық қоғамда деректер – стратегиялық ресурс болып табылады. Олардың түрлері мен көздерін дұрыс анықтау ақпаратты сақтау, өңдеу және талдау тәсілін таңдауға тікелей әсер етеді.

Құрылымдалған деректер дәстүрлі реляциялық дерекқорларда тиімді сақталса, жартылай және құрылымдалмаған деректер икемді архитектураны қажет етеді. Осы себепті қазіргі таңда **NoSQL дерекқорлары мен Big Data платформалары кеңінен қолданылады.**

Деректер көздерінің көптігі (ішкі жүйелер, сыртқы сервистер, сенсорлар, әлеуметтік желілер) ақпаратты біріктіру мен талдауды күрделендіреді, бірақ сонымен қатар терең аналитика жүргізуге мүмкіндік береді.

Big Data технологиялары:

- ✓ үлкен көлемдегі деректерді өңдеуге,
- ✓ әртүрлі форматтағы ақпаратпен жұмыс істеуге,
- ✓ деректерді нақты уақыт режимінде талдауға,
- ✓ басқарушылық шешімдердің сапасын арттыруға мүмкіндік береді.

Қазіргі цифрлық қоғамда деректер – стратегиялық ресурс болып табылады. Олардың түрлері мен көздерін дұрыс анықтау ақпаратты сақтау, өңдеу және талдау тәсілін таңдауға тікелей әсер етеді.

Құрылымдалған деректер дәстүрлі реляциялық дерекқорларда тиімді сақталса, жартылай және құрылымдалмаған деректер икемді архитектураны қажет етеді. Осы себепті қазіргі таңда **NoSQL дерекқорлары мен Big Data платформалары кеңінен қолданылады.**

Деректер көздерінің көптігі (ішкі жүйелер, сыртқы сервистер, сенсорлар, әлеуметтік желілер) ақпаратты біріктіру мен талдауды күрделендіреді, бірақ сонымен қатар терең аналитика жүргізуге мүмкіндік береді.

Big Data технологиялары:

- ✓ үлкен көлемдегі деректерді өңдеуге,
- ✓ әртүрлі форматтағы ақпаратпен жұмыс істеуге,
- ✓ деректерді нақты уақыт режимінде талдауға,
- ✓ басқарушылық шешімдердің сапасын арттыруға мүмкіндік береді.

1.3 NoSQL деректер қорларына кіріспе

NoSQL пайда болу алғышарттары

Қазіргі уақытта ақпарат көлемі жылдам өсіп келеді. Интернет, мобильді қосымшалар, әлеуметтік желілер, сенсорлар мен бұлттық сервистер үлкен көлемде деректер өндіреді.

Осыған байланысты:

- деректер көлемі артты;
- деректер әртүрлі форматта пайда болды;
- ақпаратты өңдеу жылдамдығына талап күшейді.

Осы мәселелерді шешу үшін **NoSQL дерекқорлары** пайда болды.

NoSQL (ағылш. *Not Only SQL* – «тек SQL емес») – бұл реляциялық емес деректер қорлары, үлкен көлемдегі құрылымдалмаған ақпаратты тиімді өңдеуге арналған.

NoSQL пайда болу алғышарттары

Қазіргі уақытта ақпарат көлемі жылдам өсіп келеді. Интернет, мобильді қосымшалар, әлеуметтік желілер, сенсорлар мен бұлттық сервистер үлкен көлемде деректер өндіреді.

Осыған байланысты:

- деректер көлемі артты;
- деректер әртүрлі форматта пайда болды;
- ақпаратты өңдеу жылдамдығына талап күшейді.

Осы мәселелерді шешу үшін **NoSQL дерекқорлары** пайда болды.

NoSQL (ағылш. *Not Only SQL* – «тек SQL емес») – бұл реляциялық емес деректер қорлары, үлкен көлемдегі құрылымдалмаған ақпаратты тиімді өңдеуге арналған.

NoSQL деректер қорларының негізгі ерекшеліктері:

NoSQL жүйелеріне тән қасиеттер:

- қатаң схеманың болмауы;
- деректер құрылымының икемділігі;
- горизонтальды масштабтау;
- үлкен көлемдегі деректермен жұмыс істеу;
- жоғары жылдамдық.

NoSQL деректер қорларының жұмыс принциптері

Көптеген NoSQL базалары **JSON** форматында деректерді сақтайды – бұл «атауы–мәні» жұптарынан тұратын ашық формат.

NoSQL базалары өз жұмыс ортасына бейімделген, бірақ жалпы ерекшеліктері бар: горизонталды масштабталу, операциялық және транзакциялық деректерді өңдеуге ыңғайлылық.

Жоғары өнімділік пен қарапайымдылық үшін NoSQL базалары оптимизацияланған.

NoSQL деректер қорларының жұмыс принциптері

Көптеген NoSQL базалары **JSON** форматында деректерді сақтайды – бұл «атауы–мәні» жұптарынан тұратын ашық формат.

NoSQL базалары өз жұмыс ортасына бейімделген, бірақ жалпы ерекшеліктері бар: горизонталды масштабталу, операциялық және транзакциялық деректерді өңдеуге ыңғайлылық.

Жоғары өнімділік пен қарапайымдылық үшін NoSQL базалары оптимизацияланған.

NoSQL дерекқорларының негізгі түрлері

NoSQL дерекқорлары бірнеше модельге бөлінеді:

1. **Кілт–мән (Key–Value)**
2. **Құжатқа бағытталған (Document-oriented)**
3. **Бағаналық (Column-family)**
4. **Графтық (Graph databases)**

Әр модель белгілі бір типтегі деректермен тиімді жұмыс істеуге арналған.

NoSQL-дың артықшылықтары мен кемшіліктері

Артықшылықтары:

- Икемді деректер құрылымы.
- Таралған инфрақұрылымда тиімді жұмыс.
- Арзан технология.
- Жоғары өткізу қабілеттілігі.

Кемшіліктері:

- Нақтылы басқаруды қажет етеді.
- Күрделі сұрауларға шектеулі функционал.
- Деректерді сәйкестендіру қиындықтары.
- Күрделі сценарийлерде өнімділік төмендеуі мүмкін.

NoSQL дерекқорлары заманауи ақпараттық жүйелердің даму талаптарына сай қалыптасқан тиімді шешім болып табылады. Олар үлкен көлемдегі деректермен жұмыс істеуге, жүйелердің масштабталуын қамтамасыз етуге және жоғары жүктемелер жағдайында тұрақты қызмет етуге мүмкіндік береді.

Бұл технологиялар әсіресе динамикалық ортада, яғни деректер үнемі өзгеріп отыратын және жылдам өңдеуді қажет ететін жүйелерде маңызды рөл атқарады. NoSQL шешімдерін қолдану ұйымдарға икемді архитектура құруға, ресурстарды тиімді пайдалануға және ақпараттық жүйелердің өнімділігін арттыруға жағдай жасайды.

Осылайша, NoSQL дерекқорларын түсіну – заманауи бағдарламалау, Big Data және аналитикалық жүйелерді меңгерудің маңызды қадамы болып табылады.

NoSQL дерекқорлары заманауи ақпараттық жүйелердің даму талаптарына сай қалыптасқан тиімді шешім болып табылады. Олар үлкен көлемдегі деректермен жұмыс істеуге, жүйелердің масштабталуын қамтамасыз етуге және жоғары жүктемелер жағдайында тұрақты қызмет етуге мүмкіндік береді.

Бұл технологиялар әсіресе динамикалық ортада, яғни деректер үнемі өзгеріп отыратын және жылдам өңдеуді қажет ететін жүйелерде маңызды рөл атқарады. NoSQL шешімдерін қолдану ұйымдарға икемді архитектура құруға, ресурстарды тиімді пайдалануға және ақпараттық жүйелердің өнімділігін арттыруға жағдай жасайды.

Осылайша, NoSQL дерекқорларын түсіну – заманауи бағдарламалау, Big Data және аналитикалық жүйелерді меңгерудің маңызды қадамы болып табылады.

NoSQL дерекқорлары заманауи ақпараттық жүйелердің даму талаптарына сай қалыптасқан тиімді шешім болып табылады. Олар үлкен көлемдегі деректермен жұмыс істеуге, жүйелердің масштабталуын қамтамасыз етуге және жоғары жүктемелер жағдайында тұрақты қызмет етуге мүмкіндік береді.

Бұл технологиялар әсіресе динамикалық ортада, яғни деректер үнемі өзгеріп отыратын және жылдам өңдеуді қажет ететін жүйелерде маңызды рөл атқарады. NoSQL шешімдерін қолдану ұйымдарға икемді архитектура құруға, ресурстарды тиімді пайдалануға және ақпараттық жүйелердің өнімділігін арттыруға жағдай жасайды.

Осылайша, NoSQL дерекқорларын түсіну – заманауи бағдарламалау, Big Data және аналитикалық жүйелерді меңгерудің маңызды қадамы болып табылады.

1.4 NoSQL дерекқорлары

Соңғы жылдары деректер көлемінің қарқынды өсуі байқалуда, ал дәстүрлі реляциялық дерекқорлар мұндай көлем мен әртүрліліктегі деректерді өңдеуде әрдайым тиімді бола бермейді. Осыған байланысты **жоғары масштабталу**, **жоғары өнімділік** және **икемділік** қасиеттеріне ие реляциялық емес (NoSQL) дерекқорлар пайда болды.

Бұл дәрісте реляциялық емес дерекқорлардың негізгі санаттары қарастырылады: иерархиялық, желілік, құжатқа бағытталған, кілт–мән, бағандық және графтық дерекқорлар.

Иерархиялық дерекқорлар

Иерархиялық дерекқорлар деректерді **ағаш тәрізді құрылым** түрінде ұйымдастырады. Әрбір түйін (node) бірнеше ұрпаққа ие бола алады, бірақ тек бір ғана ата-ана түйіні болады.

Бұл модель **иерархиялық құрылымдарды** (мысалы, файлдық жүйелерді) модельдеу үшін тиімді.

Артықшылықтары:

- Құрылымы қарапайым
- Иерархиялық деректермен жұмыс істеуге ыңғайлы

Кемшіліктері:

- Құрылымның икемсіздігі
- Күрделі байланыстарды көрсету қиын

Желілік дерекқорлар

Желілік дерекқорлар деректерді граф құрылымы түрінде ұсынады, мұнда түйіндердің бірнеше ата-анасы болуы мүмкін. Бұл иерархиялық модельге қарағанда деректер арасындағы байланысты икемдірек көрсетеді.

Бұл дерекқорлар 1960–1970 жылдары кеңінен қолданылған, алайда қазіргі таңда олардың қолданылуы айтарлықтай азайған.

Артықшылықтары:

- Күрделі байланыстарды көрсету мүмкіндігі
- Иерархиялық модельге қарағанда икемдірек

Кемшіліктері:

- Жобалау және қолдау күрделі
- Қазіргі заманғы жүйелерде сирек қолданылады

Құжатқа бағытталған дерекқорлар

Құжатқа бағытталған дерекқорлар деректерді **құжаттар** түрінде (көбіне JSON немесе XML форматында) сақтайды, индекстейді және іздейді. Құжаттар иерархиялық немесе жазық (flat) құрылымда болуы мүмкін және дербес ақпараттық бірлік болып саналады.

Бұл дерекқорлар **күрделі құрылымды деректермен** жұмыс істеуге және икемді сұраныстар құруға мүмкіндік береді.

Артықшылықтары:

- Схеманың икемділігі
- Күрделі және жартылай құрылымданған деректерді сақтау

Қолдану салалары:

- Веб-қосымшалар
- API және микросервистер
- Контентті басқару жүйелері

Кілт–мән дерекқорлары

Кілт–мән дерекқорлары деректерді **ассоциативті массив** түрінде сақтайды, мұнда әрбір мән бірегей кілтпен байланыстырылады. Бұл дерекқорлар қарапайым, өте жылдам және оңай масштабталады.

Алайда олар **күрделі сұраныстарды** және аналитикалық операцияларды қолдамайды. Сондықтан көбіне негізгі дерекқор ретінде емес, **көмекші жүйе** ретінде қолданылады.

Қолдану салалары:

- Кэштеу
- Сессияларды сақтау
- Таратылған жүйелердегі өзара әрекет

Бағандық дерекқорлар

Бағандық дерекқорлар деректерді **жолдар емес, бағандар бойынша** сақтайды. Бұл үлкен көлемдегі құрылымданған деректерді тиімді сақтауға және өңдеуге мүмкіндік береді.

Олар көбінесе **Big Data аналитикасында**, ақпараттық панельдерде, метрикалар мен логтарды өңдеуде қолданылады.

Артықшылықтары:

- Үлкен деректермен жоғары өнімділік
- Аналитикалық сұраныстарға тиімді

Графтық дерекқорлар

Графтық дерекқорлар – NoSQL дерекқорларының бір түрі; олар деректерді түйіндер (мәндер/сущностер) және қырлар (олардың арасындағы байланыстар) түрінде сақтайды және байланыстардың өзі деректердің жеке элементтерінен маңыздырақ болатын күрделі өзара тәуелділіктерді талдауға өте қолайлы, өйткені жоғары деңгейде байланысқан деректермен жұмыс істегенде реляциялық ДҚ қол жеткізе алмайтын икемділік пен өнімділікті ұсынады.

Бұл модель байланыстарға негізделген деректерді талдау үшін өте қолайлы.

Қолдану салалары:

- Әлеуметтік желілер
- Ұсыным жүйелері
- Биоинформатика
- Логистика және маршруттау

Реляциялық емес (NoSQL) дерекқорлар деректер схемасын қатаң сақтауды талап етпей, деректерді сақтау мен өңдеудің кең мүмкіндіктерін ұсынады. Әрбір NoSQL дерекқорының санатының **артықшылықтары мен шектеулері** бар, сондықтан оларды таңдау нақты жобаның талаптары мен деректердің сипаттамаларына байланысты жүзеге асырылуы тиіс.

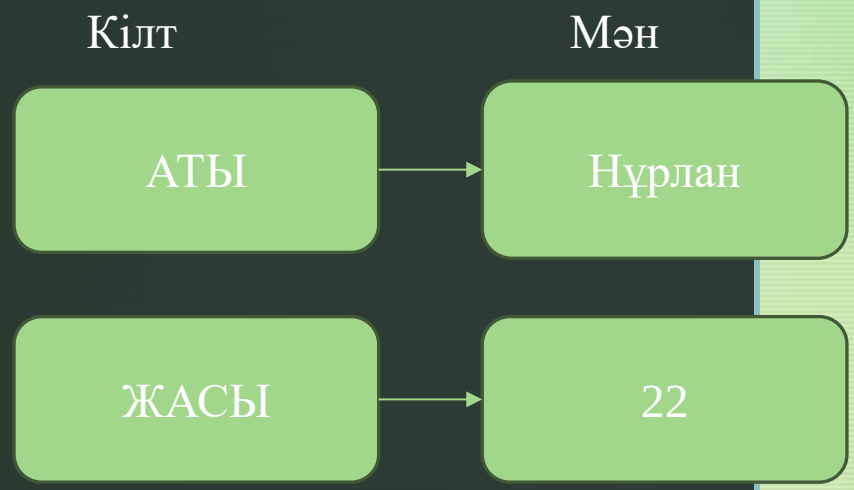
Реляциялық емес (NoSQL) дерекқорлар деректер схемасын қатаң сақтауды талап етпей, деректерді сақтау мен өңдеудің кең мүмкіндіктерін ұсынады. Әрбір NoSQL дерекқорының санатының **артықшылықтары мен шектеулері** бар, сондықтан оларды таңдау нақты жобаның талаптары мен деректердің сипаттамаларына байланысты жүзеге асырылуы тиіс.

1.5 Реляциялық емес дерекқорлардағы деректер модельдері

Бұл бөлімде реляциялық емес дерекқорларда қолданылатын төрт негізгі деректер моделі қарастырылады: кілт–мән моделі, бағандық модель, құжатқа бағытталған модель және графтық модель. Сондай-ақ осы модельдердің артықшылықтары мен шектеулері талқыланып, әрқайсысын іске асыру мысалдары келтіріледі.

Кілт–мән деректер моделі

- Кілт–мән моделі — реляциялық емес дерекқорлардағы **ең қарапайым деректер модельдерінің бірі**. Бұл модельде деректер **бірегей кілт** және соған сәйкес **мән** жұбы ретінде ұсынылады. Кілт жазбаны бірмәнді анықтау үшін қолданылады, ал мән кез келген деректер жиыны немесе объект болуы мүмкін.
- Бұл тәсіл бағдарламалауда кеңінен қолданылатын **массивтер, тізімдер немесе сөздіктер (dictionary)** ұғымымен тікелей сәйкес келеді.



Құрылу принципі

Әрбір жазба кілт–мән жұбы ретінде сақталады.

- **Кілт** – бірегей идентификатор
- **Мән** – жол, сан, тізім немесе күрделі құрылым (мысалы, JSON)

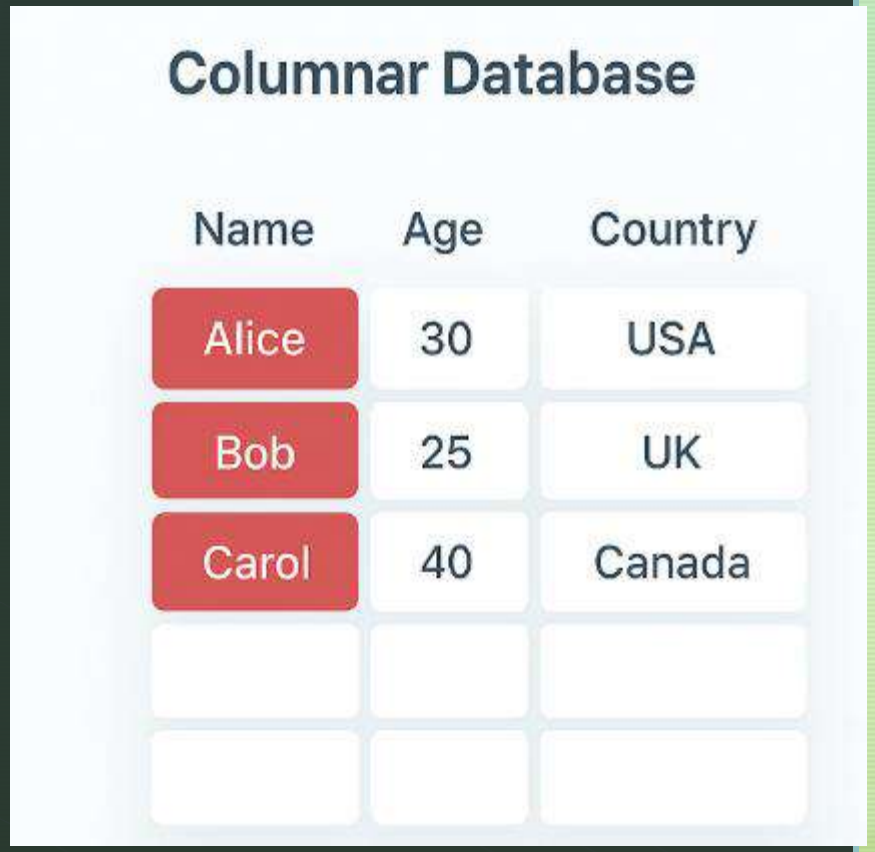
Кейбір жүйелерде мәндер хеш-кестелер немесе JSON түрінде берілуі мүмкін, бұл оларды құжатқа бағытталған модельдерге жақындатады, бірақ икемділігі төмен болады.

Бағандық (Column-Oriented) деректер моделі

- Бағандық модель – реляциялық емес дерекқорлардағы **кең таралған модельдердің бірі**. Бұл модельде деректер **бағандар бойынша** ұйымдастырылады және сақталады.
- **Құрылу принципі**
- Деректер жеке бағандарға бөлінеді. Әрбір баған бір типтегі мәндер жиынын сақтайды, ал жазбалар әртүрлі бағандар санынан тұруы мүмкін.

Бағандық (Column-Oriented) деректер моделі

Бағандық модель – реляциялық емес дерекқорлардағы **кең** таралған модельдердің бірі. Бұл модельде деректер **бағандар бойынша** ұйымдастырылады және сақталады.



Құрылу принципі

Деректер жеке бағандарға бөлінеді. Әрбір баған бір типтегі мәндер жиынын сақтайды, ал жазбалар әртүрлі бағандар санынан тұруы мүмкін.

Құжатқа бағытталған деректер моделі

Құжатқа бағытталған модель — **ең икемді деректер модельдерінің бірі**. Мұнда деректер JSON форматына ұқсас **құжаттар** түрінде сақталады.

Құрылу принципі

Деректер **құжаттар коллекциясы** ретінде ұйымдастырылады. Әр құжат:

- өрістер жиынынан тұрады
- өрістер саны мен типі әртүрлі болуы мүмкін

Қолдану мысалы:

Тапсырмаларды басқару жүйесі. Әр тапсырма жеке құжат ретінде сақталады және атауы, сипаттамасы, мерзімі сияқты өрістерді қамтиды.

Графтық деректер моделі

Графтық модель – **байланыстарға негізделген деректермен жұмыс істеуге арналған ең қуатты модельдердің бірі**.

Бұл модельде:

- **Түйіндер (nodes)** – объектілер
- **Қабырғалар (edges)** – объектілер арасындағы байланыстар

Құрылу принципі

- Әр түйін өз атрибуттарына ие, ал қабырғалар бағытталған болуы және қосымша сипаттамаларды қамтуы мүмкін.

Қолдану мысалы:

- Әлеуметтік желі. Пайдаланушылар – түйіндер, ал достық немесе жазылу – қабырғалар.

Деректер модельдерін салыстыру

Деректер моделі	Құрылу принципі	Артықшылықтары	Шектеулері
Кілт–мән	Кілт–мән жұбы	Қарапайым, икемді, жылдам	Күрделі сұраныстар жоқ
Бағандық	Бағандар мен өрістер	Икемді, өнімді	Сұраныстар күрделі
Құжатқа бағытталған	Құжаттар, өрістер	Икемді, күрделі сұраныстар	Деректердің артықтығы
Графтық	Түйіндер мен қабырғалар	Қуатты, байланыстарды талдау	Өңдеу қиындығы

Реляциялық емес дерекқорлардағы деректер модельдері деректерді ұйымдастыру мен сақтаудың жаңа тәсілдерін ұсынады және белгілі бір қолданбалар үшін анағұрлым қолайлы болуы мүмкін. Модельді таңдау қолданбаның талаптары мен деректердің сипаттамаларына байланысты жүзеге асырылады.

1.6 Реляциялық емес деректқорлар үшін сұраныс тілі

Бұл бөлімде реляциялық емес дерекқорлармен жұмыс істеуге арналған **сұраныс тілдерінің негізгі қағидаттары** қарастырылады.

Реляциялық емес дерекқорлар – бұл дәстүрлі реляциялық модельді қолданбайтын, оның орнына құжаттар, графтар, бағандар немесе кілт–мән сияқты басқа деректер модельдерін пайдаланатын дерекқорлар.

Сұраныс тілінің негізгі қағидаттары

NoSQL үшін сұраныс тілінің негізгі қағидаттары:

- Икемділік және масштабталу
- Әртүрлі деректер модельдерін қолдау
- Жоғары өнімділікті қолдау

Графтық деректер моделі үшін сұраныс тілі

Графтық деректер моделі үшін (мысалы, Neo4j) сұраныс тілі граф құрылымдарындағы деректерді іздеу, сүзу және талдау операцияларын орындауға арналған. Мұндай сұраныстар көбінесе Cypher тіліне негізделеді.

Neo4j үшін сұраныс мысалы:

```
MATCH (n:Person)-[r:KNOWS]->(m:Person)

WHERE n.name = "John"

RETURN m.name
```

Бағандық деректер моделі үшін сұраныс тілі

Бағандық деректер моделі үшін (мысалы, Apache Cassandra) сұраныс тілі кесте бағандарынан деректерді оқу және жазу операцияларын орындауға мүмкіндік береді. Мұндай сұраныстар әдетте SQL тіліне ұқсас болады.

Cassandra үшін сұраныс мысалы:

- SELECT * FROM table WHERE column1 = "value"

Құжаттық деректер моделі үшін сұраныс тілі

Құжаттық деректер моделі үшін (мысалы, MongoDB) сұраныс тілі әдетте JavaScript тіліне негізделеді. Мұндай сұраныстар құжаттардың ішіндегі деректерді іздеу, сүзу, сұрыптау және жаңарту операцияларын орындауға мүмкіндік береді.

MongoDB үшін сұраныс мысалы:

- db.collection.find({ age: { $gt: 25 } }).sort({ name: 1 })

Кілт–мән деректер моделі үшін сұраныс тілі

Кілт–мән деректер моделі үшін (мысалы, Redis) сұраныс тілі берілген кілт бойынша деректерді оқу және жазу операцияларын орындауға арналған. Мұндай сұраныстар Redis протоколына негізделеді.

Redis үшін сұраныс мысалы:

- GET key

Деректерді оқу және жазу операциялары

Деректерді оқу:

- **GET** — кілт–мән дерекқорында кілт бойынша мәнді алу
- **SELECT** — бағандық және құжаттық дерекқорларда деректерді таңдау

Деректерді жазу:

- **SET** — кілт–мән дерекқорында мән орнату
- **INSERT** — құжаттық дерекқорға жаңа құжат қосу
- **CREATE COLUMN FAMILY** — бағандық дерекқорда бағандар тобын құру

MongoDB сұраныс тілі (MQL) синтаксисінің негіздері

MongoDB Query Language (MQL) – MongoDB дерекқорында деректерді іздеу және өңдеу үшін қолданылатын сұраныс тілі. Оның синтаксисі JavaScript тіліне ұқсас.

Жұмысты бастау үшін алдымен дерекқор контексті көрсетіледі:

- use database_name

MongoDB-дегі негізгі сақтау бірлігі – **құжат**. Құжат атрибуттар жиынынан тұрады және әр атрибут «атау : мән» жұбы ретінде беріледі. Құжаттар **коллекцияларда** сақталады. Бір коллекциядағы құжаттар бірдей атрибуттарға ие болмауы мүмкін.

Әр құжатта бірегей **_id** атрибуты болады. Егер ол көрсетілмесе, MongoDB оны автоматты түрде 24-разрядты оналтылық сан ретінде жасайды. Құжаттар JSON (JavaScript Object Notation) форматында беріледі.

Қорытындылай келе, реляциялық емес (NoSQL) дерекқорлар үшін сұраныс тілі бір ғана «әмбебап SQL» формасына сыймайды, себебі NoSQL әлемі модельдер бойынша әртүрлі: құжаттық, кілт–мән (key–value), бағандық (column-family), графтық. Сондықтан сұраныстар да дерекқордың табиғатына бейімделеді: құжаттық жүйелерде JSON-құрылымдармен жұмыс істейтін **find / filter** және **aggregation pipeline** сияқты тәсілдер кең таралған; key–value қоймаларында негізгі операциялар **get/put** принципіне сүйенеді; бағандық жүйелерде үлкен көлемді деректерді жылдам оқу/жазуға бағытталған командалар қолданылады; ал графтық дерекқорларда байланыстарды «жол» ретінде талдайтын арнайы тілдер тиімді.

Қорытындылай келе, реляциялық емес (NoSQL) дерекқорлар үшін сұраныс тілі бір ғана «әмбебап SQL» формасына сыймайды, себебі NoSQL әлемі модельдер бойынша әртүрлі: құжаттық, кілт–мән (key–value), бағандық (column-family), графтық. Сондықтан сұраныстар да дерекқордың табиғатына бейімделеді: құжаттық жүйелерде JSON-құрылымдармен жұмыс істейтін **find / filter** және **aggregation pipeline** сияқты тәсілдер кең таралған; key–value қоймаларында негізгі операциялар **get/put** принципіне сүйенеді; бағандық жүйелерде үлкен көлемді деректерді жылдам оқу/жазуға бағытталған командалар қолданылады; ал графтық дерекқорларда байланыстарды «жол» ретінде талдайтын арнайы тілдер тиімді.