Дәріс 15. Деректер базасы, SQL және Таратылған есептеулер

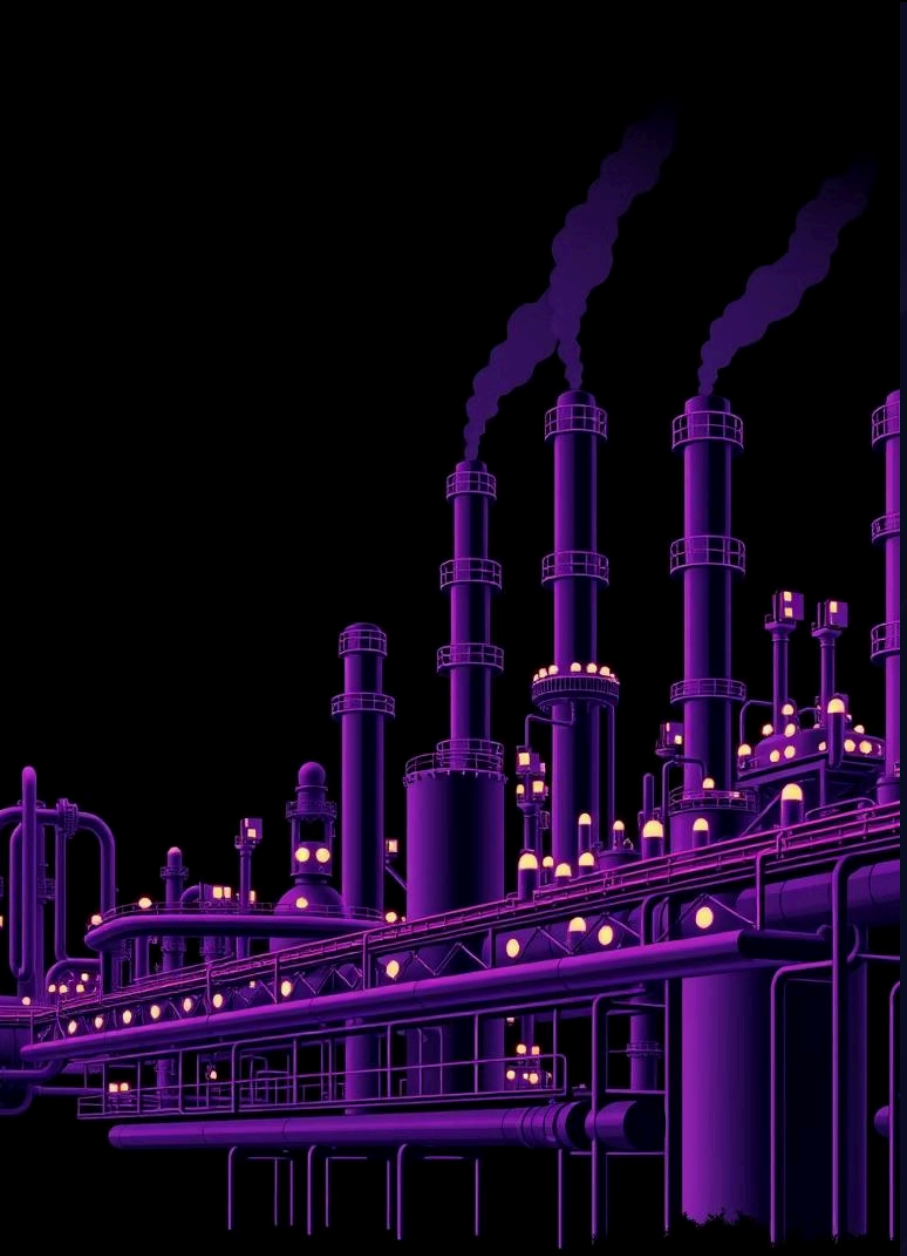
Оқытушы: Сарсембаева Т.С.

Дәрістің мақсаты

Деректерді сақтаудың негізгі "қоймасы" ретінде Реляциялық деректер базасы (RDBMS) екенін түсіндіруден басталады. Содан кейін, RDBMS-пен "сөйлесуге" арналған стандартты тіл – SQL негіздерін меңгерту көзделеді, оның ішінде DDL (CREATE TABLE), DML (INSERT/UPDATE) және DQL (SELECT) командалары қарастырылады. Әрі қарай, күрделі SQL сұраныстары (GROUP BY, JOIN) және оларды оңтайландыру (Индекстер) әдістері талданады. Сондай-ақ, дәстүрлі RDBMS "Big Data" (Үлкен деректер) көлемін өңдеуге қиналған кезде NoSQL тұжырымдамасына неліктен қажеттілік туғаны көрсетіледі. Соңында, үлкен деректерді бірнеше компьютерде параллель өңдеуге арналған Таратылған есептеулер (MapReduce) тұжырымдамасы зерттеледі.

:Негізгі сұрақтар

- SQL дегеніміз не және оның негізгі DDL, DML, DQL командалары қандай?
- SQL-де JOIN және GROUP BY не үшін қолданылады?
- Индекс (Index) деректер базасының өнімділігін қалай жақсартады?
- NoSQL деректер базаларының RDBMS-тен негізгі айырмашылығы неде ("көлденең масштабтау")?
- MapReduce тұжырымдамасы дегеніміз не? "Map" және "Reduce" фазалары қандай қызмет атқарады?
- Apache Spark-тың дәстүрлі MapReduce-тан (Hadoop) негізгі артықшылығы неде?



Деректерді қайда сақтаймыз?

"Деректер – жаңа мұнай" десек, онда деректер базасы – бұл "мұнай қоймасы", ал SQL – сол қоймадан "мұнайды сорып алатын" құбыр желісі.

Алайда, деректер көлемі (Volume), жылдамдығы (Velocity) және әртүрлілігі (Variety) өскен сайын ("Big Data" - Үлкен деректер), дәстүрлі қоймалар жеткіліксіз бола бастады. Бұл бізді NoSQL және MapReduce сияқты таратылған есептеулер тұжырымдамаларына алып келді. Бүгінгі дәрісте біз деректерді сақтау мен өңдеудің осы негізгі технологияларын қарастырамыз.

Реляциялық деректер базасы және SQL негіздері

Реляциялық деректер базасы (RDBMS) – бұл деректерді Excel кестелеріне ұқсас, қатаң құрылымдалған кестелерде (tables) сақтайтын жүйе. Әрбір кесте жолдардан (rows) және бағандардан (columns) тұрады. Кестелер бір-бірімен кілттер (keys) арқылы байланысады.

SQL (Structured Query Language) – осы реляциялық базалармен "сөйлесуге" арналған стандартты тіл. Ол бірнеше негізгі бөліктен тұрады:

DDL (Data Definition Language) – Деректерді анықтау тілі

Бұл командалар деректер базасының құрылымын (схемасын) анықтайды.

- CREATE TABLE: Жаңа кесте құру.

```
CREATE TABLE Students (
StudentID INT PRIMARY KEY, \-- Бірегей кілт
FirstName VARCHAR(100),
LastName VARCHAR(100),
Course INT
);
```

DML (Data Manipulation Language) – Деректерді басқару тілі

Бұл командалар кестедегі деректерді қосу, өзгерту немесе жою үшін қолданылады.

- INSERT: Жаңа жол қосу.

```
INSERT INTO Students (StudentID, FirstName, LastName, Course)
VALUES (1, 'Ахмет', 'Байтұрсынов', 3);
```

- UPDATE: Бар жолды жаңарту.

```
UPDATE Students
SET Course = 4
WHERE StudentID = 1;
```

- DELETE: Жолды жою.

```
DELETE FROM Students
WHERE StudentID = 1;
```

DQL (Data Query Language) – Деректерді сұрау тілі

Бұл – деректерді алуға арналған ең маңызды бөлік.

- SELECT: Деректерді таңдау.

```
SELECT FirstName, LastName
FROM Students
WHERE Course > 2; \-- 2-курстан жоғары студенттерді таңдау
```

Күрделі сұраныстар: **GROUP BY, ORDER BY, JOIN**

- ORDER BY: Нәтижелерді сұрыптау.

```
SELECT * FROM Students
ORDER BY LastName ASC; \-- Тегі бойынша өсу ретімен
```

- GROUP BY: Деректерді топтастыру және агрегаттық функцияларды (COUNT(), SUM(), AVG()) қолдану.

```
\-- Әр курста қанша студент бар екенін санау
SELECT Course, COUNT(StudentID)
FROM Students
GROUP BY Course;
```

- JOIN: Екі немесе одан да көп кестені олардың ортақ кілттері арқылы біріктіру. Бұл – SQL-дің ең қуатты мүмкіндігі.

```
\-- Студенттердің аттарын және олардың пәндерінің аттарын біріктіру
SELECT S.FirstName, C.CourseName
FROM Students AS S
JOIN Grades AS G ON S.StudentID = G.StudentID
JOIN Courses AS C ON G.CourseID = C.CourseID;
```

Ішкі сұраулар **(Subqueries)** және Индекстер **(Indexes)**

- Ішкі сұрау (Subquery): Бұл – басқа сұраныстың ішінде орналасқан сұраныс.

```
\-- Орташа бағадан жоғары баға алған студенттерді табу
SELECT FirstName, LastName
FROM Students
WHERE StudentID IN (
SELECT StudentID
FROM Grades
WHERE Grade > (SELECT AVG(Grade) FROM Grades)
);
```

Индекстер (Indexes): Индекс – бұл деректер базасының өнімділігін арттыруға арналған арнайы құрылым.

Мысалы: Кітаптың соңындағы алфавиттік көрсеткіш (index). Егер сізге "SQL" туралы ақпарат керек болса, сіз бүкіл кітапты парақтамайсыз, тек индекске қарап, "S" әрпінен "SQL" сөзін тауып, оның қай бетте екенін бірден білесіз.

Сұраныстарды оңтайландыру **(Query Optimization)**

Миллиондаған жолдары бар кестелермен жұмыс істегенде, бірдей нәтиже беретін екі түрлі сұраныстың біреуі 1 секундта, екіншісі 1 сағатта орындалуы мүмкін.

EXPLAIN командасы: Көптеген ДББЖ (Деректер Базасын Басқару Жүйелері) сұраныстың "орындалу жоспарын" (execution plan) көрсету үшін EXPLAIN командасын ұсынады. Ол сұраныстың индекстерді қолданып жатқанын, кестелерді қалай біріктіріп жатқанын көрсетеді.

SELECT * қолданбау (тек қажетті бағандарды таңдау), WHERE шарттарын индекстелген бағандарға қолдану, JOIN операцияларын оңтайлы реттеу.

SQL-дан тыс: NoSQL мәліметтер базасы

XXI ғасырдың басында деректердің көлемі, жылдамдығы және әртүрлілігі (мысалы, Facebook-тегі суреттер, X-тегі твиттер, IoT құрылғыларынан келетін логтар) соншалықты өсті, дәстүрлі RDBMS бұл "Big Data" (Үлкен деректер) ағынын өңдеуге қинала бастады.

NoSQL ("Not Only SQL") – бұл реляциялық емес, қатаң схемасы жоқ және көлденең масштабталатын (horizontal scaling) деректер базаларының жалпы атауы.

- Көлденең масштабтау: Бір үлкен, қымбат серверді (vertical scaling) жақсарта бергенше, көптеген арзан, стандартты серверлерді (кластер) қосу арқылы жүйенің қуатын арттыру.

NoSQL негізгі түрлері:

Құжаттық (Document)

Деректерді JSON/BSON форматтағы икемді құжаттарда сақтайды. (Мысал: MongoDB)

Кілт-Мән (Key-Value)

Өте жылдам, қарапайым сақтау. (Мысал: Redis, Amazon DynamoDB)

Бағандық (Column-family)

Үлкен деректерді оқуға және жазуға оңтайландырылған. (Мысал: Apache Cassandra, Google Bigtable)

Графтық (Graph)

Байланыстарды (қабырғаларды) және нысандарды (түйіндерді) сақтауға арналған. (Мысал: Neo4j)

Таратылған есептеу: **MapReduce** тұжырымдамасы

Егер бізде 100 Терабайт (ТБ) дерек болса, оны бір компьютерге сыйдыру немесе бір компьютерде өңдеу мүмкін емес. Таратылған есептеу – бұл үлкен есепті көптеген кішігірім бөліктерге бөліп, оларды ондаған, жүздеген немесе мыңдаған компьютерлерден (кластер) тұратын желіде параллель түрде орындау процесі.

MapReduce – бұл Google тарапынан ұсынылған, осындай таратылған есептеулерді орындауға арналған қарапайым, бірақ өте қуатты бағдарламалау моделі. (Оның ең танымал ашық-кодтық имплементациясы – Apache Hadoop).

MapReduce процесі екі негізгі фазадан тұрады (ортасында "Shuffle" фазасы бар):

1. MAP (Бейнелеу) фазасы: Бастапқы деректерді өңдеп, оны (кілт, мән) – (key, value) жұптарына айналдыру. Кластердегі әрбір компьютер (node) деректердің тек өз бөлігін алады да, осы "Map" функциясын орындайды.
2. SHUFFLE (Араластыру) фазасы: "Map" фазасынан шыққан барлық (key, value) жұптарын жинап, оларды кілт бойынша топтастыру. Бұл фазаны жүйенің өзі автоматты түрде орындайды.
3. REDUCE (Кішірейту) фазасы: Әрбір бірегей кілт (key) үшін топтастырылған мәндер тізімін (\[value1, value2, ...\]) алып, оларды бір нәтижеге дейін агрегациялау (кішірейту).

Мысал: Сөз жиілігін санау **(Word Count)**

Бұл MapReduce-ті түсіндіруге арналған "Hello, World!" мысалы.

Міндет: 100 ТБ мәтіндік файлдардағы әрбір сөздің қанша рет кездесетінін санау.

Кіріс деректер (2 бөлікке бөлінген):

- Input 1: "hello world hello"
- Input 2: "hello mapreduce world"

1. MAP фазасы (екі компьютерде параллель):

- Worker 1 (Input 1):

```
map("hello") -> (hello, 1)
map("world") -> (world, 1)
map("hello") -> (hello, 1)
```

- Worker 2 (Input 2):

```
map("hello") -> (hello, 1)
map("mapreduce") -> (mapreduce, 1)
map("world") -> (world, 1)
```

2. SHUFFLE фазасы (Жүйелік):

```
(hello, [1, 1, 1])
(world, [1, 1])
(mapreduce, [1])
```

3. REDUCE фазасы (үш компьютерде параллель):

```
Reducer 1: reduce(hello, [1, 1, 1]) -> (hello, 3)
Reducer 2: reduce(world, [1, 1]) -> (world, 2)
Reducer 3: reduce(mapreduce, [1]) -> (mapreduce, 1)
```

Нәтиже: (hello, 3), (world, 2), (mapreduce, 1)

MapReduce қолданбалары

- Жаңалықтар арналарын талдау: Миллиондаған жаңалықтар мақалаларынан трендтегі тақырыптарды (Topic Modeling, LDA) немесе жиі кездесетін тіркестерді (n-грамм) табу.

Map: Әр мақаланы оқып, (n-грамм, 1) жұптарын шығару.

Reduce: Әрбір n-грамм үшін '1' сандарын қосу.

- Матрицаларды көбейту: Екі өте үлкен матрицаны (A * B = C) таратылған түрде көбейту.

Map: Матрицаның бөліктерін (блоктарын) әртүрлі компьютерлерге жіберіп, аралық көбейтінділерді есептеу.

Reduce: Аралық көбейтінділерді дұрыс кілттер бойынша қосып, C матрицасының соңғы ұяшықтарын алу.

Әрі қарай зерттеу үшін

MapReduce (Hadoop) Big Data саласында төңкеріс жасағанымен, ол кейбір міндеттерде (әсіресе итеративті алгоритмдерде, мысалы, машиналық оқыту) баяу жұмыс істеді, себебі ол әр операциядан кейін нәтижені дискіге жазып отырды. Сол үшін осы альтернативаларды пайдалануға болады:

Apache Spark

MapReduce-тің заманауи, әлдеқайда жылдам баламасы. Ол есептеулерді дискіде емес, жедел жадта (In-Memory) орындайды, бұл оны машиналық оқыту (MLlib) және SQL-сұраныстары (Spark SQL) үшін өте тиімді етеді.

Ағынды деректерді өңдеу (Stream Processing)

Деректерді "пакет" (batch) түрінде емес (MapReduce сияқты), нақты уақыт режимінде (real-time) өңдеу. (Мысал: Apache Kafka, Apache Flink).

Бұлтты деректер қоймалары (Cloud Data Warehouses)

Инфрақұрылымды басқаруды қажет етпейтін, ауқымды аналитикалық SQL-сұраныстарға арналған қызметтер. (Мысал: Google BigQuery, Amazon Redshift, Snowflake).

ҚОРЫТЫНДЫ

Курстың басты мақсаты – тек құралдарды ғана емес, сонымен қатар деректермен "ойлау" мәдениетін сіңіру болды. Біз "Неліктен?" деген сұрақтан бастадық. Біз деректердің жай ғана сандар жиынтығы емес, заманауи әлемдегі жаңа "стратегиялық актив", тіпті "жаңа мұнай" екенін түсіндік. Осы курстың негізгі міндеті – сол шикі "мұнайды" өңдеп, оны нақты білімге, құнды инсайтқа және дұрыс шешімге айналдыруды үйрену болды. Бұл – деректерді зерттеушінің (Data Scientist) негізгі міндеті .

Бұл күрделі міндетті шешу үшін әмбебап әрі қуатты құрал – Python тілі таңдалды. Бірақ есіңізде болсын, Python – бұл тек құрал; деректер ғылымының нағыз тілі – математика. Сондықтан іргетасты қалаудан бастадық:

- Сызықтық алгебра арқылы деректерді векторлар мен матрицалар түрінде құрылымдауды;
- Статистика арқылы деректердің "орталығын" (орташа, медиана) және "шашырауын" (вариация, корреляция) түсінуді;
- Ықтималдық теориясы мен Байес теоремасы арқылы белгісіздікті басқаруды меңгердік.

Іргетасты қалап болған соң, біз деректер ғалымының нақты жұмыс процесіне көштік. Бұл жұмыстың 70-80%-ы көзге көрінбейтін, бірақ ең маңызды дайындық екенін есте сақтаңыз. Біз деректерді әртүрлі көздерден жинауды (CSV, JSON, API) және оларды "тазалауды" үйрендік: жоғалған мәндерді толтыру, қайталауларды жою және модельге дайындау үшін масштабтау . Содан кейін, Matplotlib және Seaborn арқылы осы "құрғақ" сандарды түсінікті графиктерге айналдырып, деректерді "көруді" үйрендік.

Курстың өзегі, әрине, Машиналық оқыту (ML) болды . Біз жай ғана алгоритмдерді жаттап қоймай, олардың "жүрегін" – Градиенттік түсуді (Gradient Descent) түсінуге тырыстық. Бұл алгоритмнің қателік функциясын азайту үшін итеративті түрде "төмен қарай жүретінін" көрдік. Осы іргелі біліммен біз ML арсеналын жинадық:

- Сандарды болжау үшін Регрессияны (қарапайым және көптік);
- Категорияларды болжау үшін Классификацияны (Логистикалық регрессия, KNN, Аңғал Байес, Шешім ағаштары және Кездейсоқ ормандар);
- Жасырын топтарды табу үшін Кластерлеуді (K-Means, DBSCAN);
- Осылардың барлығының шыңы болып табылатын Терең оқытуды (Нейрондық желілер).

Біз бұл құралдардың жай ғана теория емес, нақты мәселелерді қалай шешетінін көрдік: мәтіндерді түсіну (NLP), әлеуметтік желілердегі ықпалды адамдарды анықтау (SNA, PageRank) және Netflix-тің сізге не ұнайтынын болжауы (Ұсыныс жүйелері).

Соңында, біз деректер бір компьютерге сыймайтын деңгейге ("Big Data") жеткенде не істеу керектігін түсіндік. Біз деректерді SQL арқылы сұрауды, NoSQL-да икемді сақтауды және MapReduce арқылы есептеулерді мыңдаған компьютерге тарату тұжырымдамасын меңгердік.

Қорытындылай келе, сіздер бұл курста тек теориялық іргетас қана емес, сонымен қатар Pandas, Scikit-learn және визуализация сияқты нақты, тәжірибелік құралдар жиынтығын алдыңыздар.

Сіздерде деректерді талдау сапарыңызды бастау үшін барлық негізгі білім бар.

Дәрісті меңгеруге арналған бақылау сұрақтары:

- SQL дегеніміз не және DDL, DML, DQL арасында қандай айырмашылық бар?
- Екі кестені (мысалы, Students және Courses) біріктіру үшін SQL-де қандай команда қолданылады?
- GROUP BY командасы COUNT() немесе AVG() сияқты агрегаттық функциялармен қалай жұмыс істейді?
- Деректер базасындағы Индекс (Index) не үшін қажет? Кітап мысалы арқылы түсіндіріңіз .
- NoSQL деректер базаларының дәстүрлі RDBMS-тен (SQL) екі негізгі артықшылығы неде?
- MapReduce моделінің "Map" және "Reduce" фазаларының негізгі міндеттері қандай?
- "Word Count" (Сөз санау) мысалында "Map" фазасы қандай (кілт, мән) жұбын шығарады?

Әдебиеттер тізімі:

- Silberschatz, A., Korth, H. F., & Sudarshan, S. (2019). Database System Concepts (7th ed.). McGraw-Hill.
- Date, C. J. (2003). An Introduction to Database Systems (8th ed.). Addison-Wesley.
- Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI '04). **https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf**
- Sadalage, P. J., & Fowler, M. (2012). NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Addison-Wesley.
- McKinney, W. (2022). Python for Data Analysis (3rd ed.). O'Reilly Media. **https://wesmckinney.com/book/**
- Géron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly Media. **https://github.com/ageron/handson-ml3**
- Karau, H., Konwinski, A., Wendell, P., & Zaharia, M. (2015). Learning Spark: Lightning-Fast Big Data Analysis. O'Reilly Media.