Databázové systémy

Ján Mazák

FMFI UK Bratislava

Tento predmet

- zameraný najmä na praktické aspekty práce s relačnými DBMS
- dotazy
- navrhovanie databázových schém v relačnom modeli
- transakcie
- efektivita (význam indexov, vkladanie veľkých objemov dát)

Tento predmet

- zameraný najmä na praktické aspekty práce s relačnými DBMS
- dotazy
- navrhovanie databázových schém v relačnom modeli
- transakcie
- efektivita (význam indexov, vkladanie veľkých objemov dát)
- ale aj prehľad o teoretických aspektoch (vyjadrovacia sila dotazovacích jazykov, zložitosť vybraných algoritmov)

Hodnotenie

- účasť na prednáškach ani cvičeniach nie je povinná
- prednášky neraz zahŕňajú aj „teoretické precvičenie“

Hodnotenie

- účasť na prednáškach ani cvičeniach nie je povinná
- prednášky neraz zahŕňajú aj „teoretické precvičenie“
- na známku E treba 60% bodov zo všetkého
- cvičenia sú praktické, vhodné na samostatnú či skupinovú prácu (aspoň 10 x 2 b, treba min. 12)
- tri domáce úlohy (3 x 15 b, treba min. 27)
- písomka z dotazov (10 b, treba min. 6)
- ústna skúška (25 b, treba min. 15)

Využívanie LLM

Vyskúšajte prácu s ChatGPT či iným LLM:

- nechajte si nejakú tému či detail vysvetliť
- nechajte ho klásť vám otázky k učivu
- rozpory medzi tvrdeniami LLM a inými materiálmi možno diskutovať na prednáške

Využívanie LLM

Vyskúšajte prácu s ChatGPT či iným LLM:

- nechajte si nejakú tému či detail vysvetliť
- nechajte ho klásť vám otázky k učivu
- rozpory medzi tvrdeniami LLM a inými materiálmi možno diskutovať na prednáške
- môžete si nechať kontrolovať dotazy, príp. vyžiadať si alternatívne riešenia a nechať LLM posúdiť ich výhody/nevýhody

Čo je databáza?

- kolekcia údajov
- so štruktúrou

Čo je databáza?

- kolekcia údajov
- so štruktúrou

DBMS (database management system), voľne tiež databáza

- obsahuje veci spoločné pre jednotlivé databázy
- pracuje nad dátami abstraktne, používateľ (tvorca konkrétnej databázy) definuje všetky vzťahy medzi dátami (napr. konzistentnosť)

Požiadavky na DBMS

- trvalé uchovanie (persistency)
- bezpečnosť (na úrovni hw, sw, používateľov)
- paralelizmus (veľa používateľov zároveň)
- pohodlnosť (vysokoúrovňové deklaratívne jazyky)
- efektívnosť (tisíce dotazov za sekundu)

Požiadavky na DBMS

- trvalé uchovanie (persistency)
- bezpečnosť (na úrovni hw, sw, používateľov)
- paralelizmus (veľa používateľov zároveň)
- pohodlnosť (vysokoúrovňové deklaratívne jazyky)
- efektívnosť (tisíce dotazov za sekundu)

Aplikácie zapisujúce extrémne množstvo dát neraz nevyužívajú DBMS, lebo by ich to spomalilo.

Čo všetko súvisí s DBMS?

- fyzické umiestnenie dát (HDD vs. SSD, súborový systém)
- sieťové pripojenia (klienti, distribuovanosť databázy)
- paralelizmus (veľa používateľov robiacich operácie nad tými istými dátami)
- optimalizácia dotazov (algoritmická zložitosť)

Multitier architecture

- 1-tier: používateľ/aplikačný program pracuje priamo s db (napr. androidová aplikácia s SQLite)
- 2-tier: používateľ k db pristupuje cez sieť, ale priamo

Multitier architecture

- 1-tier: používateľ/aplikačný program pracuje priamo s db (napr. androidová aplikácia s SQLite)
- 2-tier: používateľ k db pristupuje cez sieť, ale priamo
- 3-tier: bežné webové aplikácie: frontend a backend, používateľ nekomunikuje priamo s db a všetka komunikácia ide cez API backendu
  (možno napr. filtrovať dáta podľa prihláseného užívateľa nad rámec databázového dotazu)

Dátový model

- relačný (tabuľky)
- entitno-relačný
- objektový
- hierarchický (strom)
- dokumentový (XML, json)
- graf
- key-value store

SQL — relačný model
NoSQL — key-value store, graph, document...

Relačný dátový model

Dáta v tabuľkách; stĺpce sú atribúty, riadky záznamy.

Name	Deptno
John	10
Thomas	20
Joe	40

Deptno	Dept. name
10	Accounting
20	PR
30	Development

Záznamy z rôznych tabuliek sa prepájajú operáciou JOIN.

Databázové jazyky

- Data Query Language (DQL) — dotazy
- Data Definition Language (DDL) — definovanie štruktúry db
- Data Manipulation Language (DML) — vkladanie, mazanie a úprava dát

Databázové jazyky

- Data Query Language (DQL) — dotazy
- Data Definition Language (DDL) — definovanie štruktúry db
- Data Manipulation Language (DML) — vkladanie, mazanie a úprava dát

Prístup k dátam z programovacích jazykov:

- priamo (cez DML)
- s jednoduchou nadstavbou (prepared statement atď.)
- ORM (object-relational mapper) — pracuje sa s objektmi, DML sa automaticky generuje v pozadí

Dotazovacie jazyky

- chceme, aby aplikácie boli nezávislé od reprezentácie dát
- optimalizácia na úrovni DBMS, nie u klienta

Dotazovacie jazyky

- chceme, aby aplikácie boli nezávislé od reprezentácie dát
- optimalizácia na úrovni DBMS, nie u klienta
- deklaračné jazyky (len čo chceme, nie ako to vypočítať): relačný kalkul (prvorádové matematické formuly), SQL, Datalog

Dotazovacie jazyky

- chceme, aby aplikácie boli nezávislé od reprezentácie dát
- optimalizácia na úrovni DBMS, nie u klienta
- deklaračné jazyky (len čo chceme, nie ako to vypočítať): relačný kalkul (prvorádové matematické formuly), SQL, Datalog
- interné db jazyky (procedurálne zachytávajú postup výpočtu): relačná algebra, fyzické operátory

Nevhodnosť prirodzeného jazyka

Ľudia neraz veci kvantifikujú nesprávne alebo neúplne (zvlášť pozor na ľudí pôsobiacich v odvetví, kde nepoznáte konvencie). Ako interpretovať tvrdenie „chlieb predávajú v potravinách“?

Nevhodnosť prirodzeného jazyka

Ľudia neraz veci kvantifikujú nesprávne alebo neúplne (zvlášť pozor na ľudí pôsobiacich v odvetví, kde nepoznáte konvencie). Ako interpretovať tvrdenie „chlieb predávajú v potravinách“?

- každý chlieb predávajú len v potravinách a nikde inde
- existuje druh chleba, ktorý predávajú len v miestnej predajni potravín
- existuje druh chleba, ktorý predávajú v každých potravinách
- existujú potraviny, ktoré predávajú aspoň jeden chlieb
- každé potraviny predávajú aspoň jeden chlieb
- ...

Ukážka relač. kalkulu, datalogu a relačnej algebry

Alkoholy, ktoré ľúbi každý pijan (ktorý niečo ľúbi), čapujú ich všade (kde niečo čapujú) a niekto ich už pil.

$$\Big\{A \mid (\exists I\, \exists M\, \mathsf{vypil}(I, A, M)) \wedge \neg\Big[\exists P(\exists A_2\, \mathsf{lubi}(P, A_2)) \wedge \neg\, \mathsf{lubi}(P, A))\Big)\Big] \\ \wedge \neg\Big[\exists K\Big((\exists A_2\, \mathsf{capuje}(K, A_2)) \wedge \neg\, \mathsf{capuje}(K, A)\Big)\Big]\Big\}$$

$$\begin{aligned} \mathsf{nelubenyNiekym}(A) &\leftarrow \mathsf{vypil}(\_, A, \_), \mathsf{lubi}(P, \_), \neg\, \mathsf{lubi}(P, A). \\ \mathsf{necapovanyNiekde}(A) &\leftarrow \mathsf{vypil}(\_, A, \_), \mathsf{capuje}(K, \_), \neg\, \mathsf{capuje}(K, A). \\ \mathsf{answer}(A) &\leftarrow \mathsf{vypil}(\_, A, \_), \neg\, \mathsf{nelubenyNiekym}(A), \neg\, \mathsf{necapovanyNiekde}(A). \end{aligned}$$

$$\begin{aligned} \mathsf{nelubenyNiekym} &= (\pi_A(\mathsf{vypil}) \times \pi_P(\mathsf{lubi})) \rhd \mathsf{lubi} \\ \mathsf{necapovanyNiekde} &= (\pi_A(\mathsf{vypil}) \times \pi_K(\mathsf{capuje})) \rhd \mathsf{capuje} \\ \mathsf{answer} &= ((\pi_A(\mathsf{vypil}) \rhd \mathsf{nelubenyNiekym}) \rhd \mathsf{necapovanyNiekde} \end{aligned}$$

Ukážka SQL

Alkoholy, ktoré ľúbi každý pijan (ktorý niečo ľúbi), čapujú ich všade (kde niečo čapujú) a niekto ich už pil.

```
SELECT DISTINCT v.A
FROM vypil v
WHERE NOT EXISTS (SELECT 1 FROM lubi l
                  WHERE NOT EXISTS (SELECT 1 FROM lubi l2
                                    WHERE l2.P = l.P AND l2.A = v.A))
    AND NOT EXISTS (SELECT 1 FROM capuje c
                    WHERE NOT EXISTS (SELECT 1 FROM capuje c2
                                      WHERE c2.K = c.K AND c2.A = v.A))
```

História

- 1970: relačný model
- 1986: prvý štandard SQL
- od cca 2010: NoSQL, Spark...

História

- 1970: relačný model
- 1986: prvý štandard SQL
- od cca 2010: NoSQL, Spark...
- reálne systémy nedržia tempo s teóriou ani SQL štandardmi, napr. rekurzia je v najväčších DBMS implementovaná asi 10 rokov, kým štandard je z 1999
- a naopak k NoSQL systémom chýba univerzálna formálna teória (sú rôznorodé)

NoSQL databázy

Theorem (CAP theorem, proved in 2002)

Neexistuje systém, ktorý má zároveň tieto 3 vlastnosti:

- *consistency*
- *availability*
- *partition tolerance*

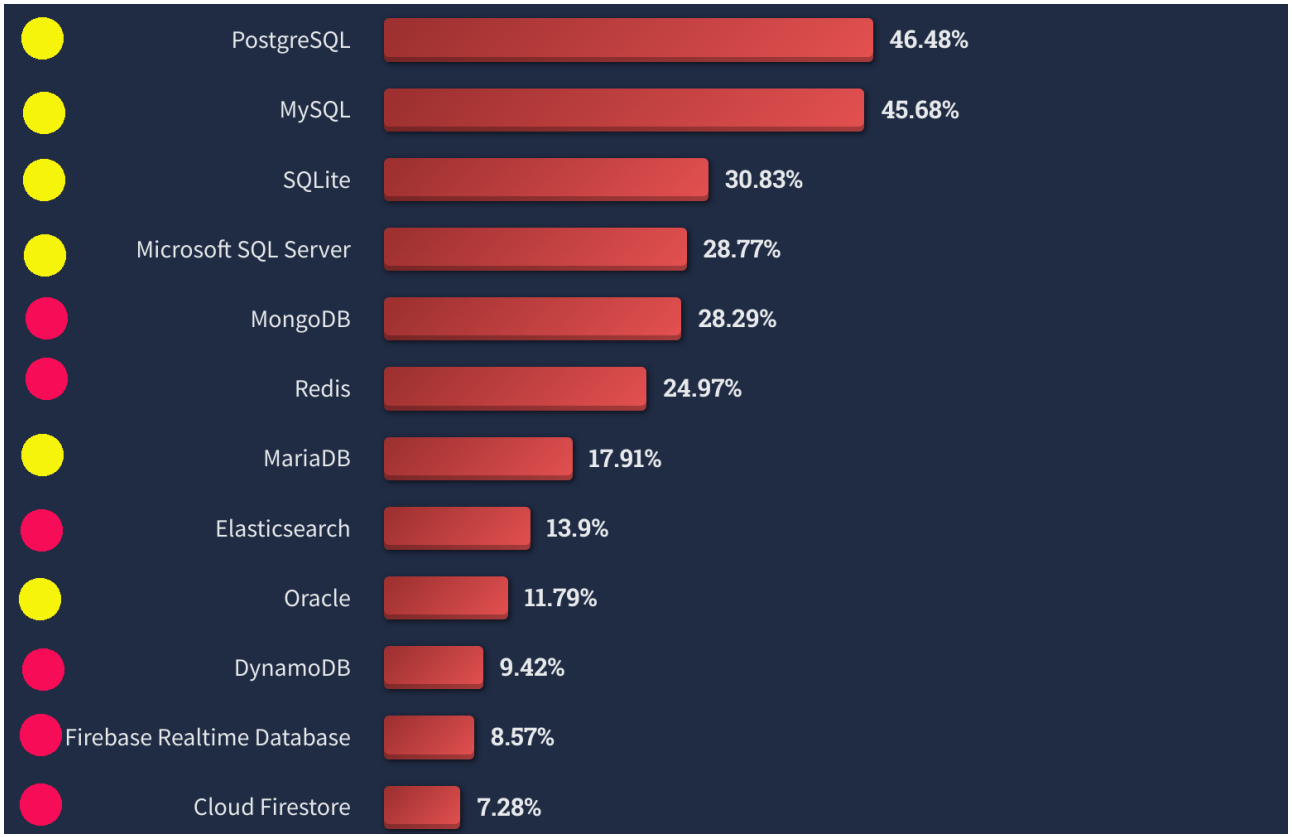
CP: MongoDB

AP: Cassandra

CA: partitions cannot be avoided, but PostgreSQL (somewhat)

Viac: `https://www.ibm.com/topics/cap-theorem`

Databases used by professionals (2022)



`https://survey.stackoverflow.co/2022/`

yellow — relational databases (SQL)
others — not relational, NoSQL

DBMS na tomto predmete

PostgreSQL

- výborný súlad s SQL štandardom
- zrozumiteľná a podrobná dokumentácia
- dobrý výkon, škálovateľnosť, široké využitie v praxi

SQLite

- celá databáza v jedinom súbore
- ľahko prenosná, zálohovateľná a administrovateľná
- nenáročná na pamäť i procesor
- chýbajú mnohé veci bežné v iných DBMS

DBMS pre dátovú vedu

- aj niektoré nie-relačné databázy podporujú jazyk SQL (napr. Apache Spark, Hadoop)
- nie-relačné databázy môžu zvýšiť výkon v špeciálnych prípadoch, inak všetko vieme spraviť aj s relačnou db
- limity tabuľky v PostgreSQL: 32 TB dát, miliardy riadkov, 1600 stĺpcov
- miesto komplikovanej distribuovanej db zvážiť zjednodušenie dátového modelu, aby bol zvládnuteľný v PostgreSQL (aspoň pre účely analýzy dát)

SQLite fun facts

- viac ako 10^{12} existujúcich SQLite databáz
- open source, ale nie open contribution
  — len 3 developeri
- kód v C, cca 160 000 riadkov
- 600 riadkov testov na každý riadok vlastného kódu
- 100% pokrytie možných vetiev v kóde
- aké situácie testujú? inšpirujte sa:
  `https://www.sqlite.org/testing.html`

Literatúra

- http://www.noucamp.org/cp2/dbt/DBIntroduction.pdf
- https://www.db-book.com/slides-dir/PDF-dir/ch1.pdf
- http://infolab.stanford.edu/~ullman/fcdb/ch1.pdf
- https://www.ibm.com/topics/cap-theorem

Úlohy: relačný kalkul

Databáza: *osoba*(A), *pozna*(Kto, Koho)

- osoby, ktoré poznajú sysľa
- osoby, ktoré nepoznajú nikoho (žiadne iné osoby)
- osoby, ktoré majú aspoň dvoch známych (osoby)
- osoby, ktoré pozná presne jedna osoba
- osoby, ktoré poznajú iba Jožka
- osoby, ktoré poznajú všetkých známych svojich známych
- osoby, ktoré majú všetky vzťahy symetrické