

Databázové systémy

Ján Mazák

FMFI UK Bratislava

Tento predmet

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

Tento predmet

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

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

- ▶ účasť na prednáškach ani cvičeniach nie je povinná
- ▶ prednášky niekedy 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)

- ▶ niektorí z vás pravidelne používajú SQL, iní sotva začínajú; nemá význam, aby všetci riešili naraz to isté
- ▶ v zadaniach cvičení preskakujte nebodované úlohy, ktoré zjavne viete riešiť
- ▶ AI vie vyriešiť cca všetky zadané úlohy, keď však prácu spraví za vás, takmer nič sa nenaučíte
- ▶ predmet zachytáva veľa rôznych vecí, aby ste mali prehľad, nepotrebuje však vedieť detaily (prehľad má hodnotu aj v dobe AI — nie je ľahké primäť LLM, aby vám povedal o niečom, čo netušíte, že existuje)
- ▶ občas sa dotkneme spracovania dát aj mimo databáz (napr. ako triediť dáta, ktoré sa nezmestia do RAM)

Riešenia cvičení a DÚ pred odovzdaním:

- ▶ diskutujte so spolužiakmi a AI na všeobecnej úrovni, nechajte si poradiť, ale primárne riešte úlohy sami
- ▶ riešenia pred odovzdaním nechajte skontrolovať aspoň dvom AI modelom (ChatGPT, Claude, Gemini...)
- ▶ opravte chyby, nechajte si navrhnúť zlepšenia a alternatívy
- ▶ AI dáva užitočný feedback, ale nie je celkom spoľahlivý; starostlivo preverujte, čo AI tvrdí, môžete konzultovať so spolužiakmi

- ▶ rozpory medzi tvrdeniami AI a inými materiálmi možno diskutovať na prednáške
- ▶ v niektorých prezentáciách otázky na zamyslenie, diskutujte o nich s AI a navzájom

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 klasický 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 vykonávajúcich 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 DQL a DML)
- ▶ s jednoduchou nadstavbou (prepared statement atď.)
- ▶ ORM (object-relational mapper) — pracuje sa s objektmi, DQL a DML sa automaticky generujú v pozadí

Dotazovacie jazyky

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

- ▶ 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.

$$\left\{ A \mid (\exists I \exists M \text{ vypil}(I, A, M)) \wedge \neg \left[\exists P (\exists A_2 \text{ lubi}(P, A_2)) \wedge \neg \text{lubu}(P, A) \right] \right. \\ \left. \wedge \neg \left[\exists K \left((\exists A_2 \text{ capuje}(K, A_2)) \wedge \neg \text{capuje}(K, A) \right) \right] \right\}$$

$\text{nelubenyNiekym}(A) \leftarrow \text{vypil}(_, A, _), \text{lubi}(P, _), \neg \text{lubi}(P, A).$

$\text{necapovanyNiekde}(A) \leftarrow \text{vypil}(_, A, _), \text{capuje}(K, _), \neg \text{capuje}(K, A).$

$\text{answer}(A) \leftarrow \text{vypil}(_, A, _), \neg \text{nelubenyNiekym}(A), \neg \text{necapovanyNiekde}(A).$

$\text{nelubenyNiekym} = (\pi_A(\text{vypil}) \times \pi_P(\text{lubi})) \triangleright \text{lubi}$

$\text{necapovanyNiekde} = (\pi_A(\text{vypil}) \times \pi_K(\text{capuje})) \triangleright \text{capuje}$

$\text{answer} = ((\pi_A(\text{vypil}) \triangleright \text{nelubenyNiekym}) \triangleright \text{necapovanyNiekde})$

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))
```

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

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

Prečo toľké zaostávanie za štandardom SQL?

- ▶ DBMS prioritizujú funkcionality podľa požiadaviek svojich zákazníkov
- ▶ nehody: skôr, ako sa unifikovaná verzia podpory pre JSON dostala do štandardu (cca 2016), rôzne DBMS to implementovali po svojom (druhá sada operátorov pre JSON nemá pridanú hodnotu a deprecation pre pôvodnú sadu vyvolá nevôľu zákazníkov)
- ▶ pre niektoré časti štandardu nie je známy žiaden v praxi efektívny spôsob výpočtu (alebo by vyžadoval masívnu zmenu vnútorného spôsobu fungovania DBMS)
- ▶ štandard je veľmi komplikovaný a fungovanie databázy tiež (paralelizmus, prelínanie funkcionalít), pridávanie nových vecí tak znamená veľké úsilie neraz s malou pridanou hodnotou a značným rizikom (spomalenie či bugy pre klientov, čo nič nové nepotrebujú)

Prečo toľké zaostávanie za štandardom SQL?

Príklady funkcionalít s nízkou podporou:

- ▶ temporal tables (evidencia histórie pre jednotlivé riadky, nie iba posledný stav)
- ▶ pattern matching zahŕňajúci postupnosti riadkov (napr. time-series data analysis)
- ▶ polymorphic table functions (výsledkom je tabuľka, ktorej stĺpce závisia od obsahu db)

Theorem (CAP theorem, proved in 2002)

Neexistuje systém, ktorý má zároveň tieto 3 vlastnosti:

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

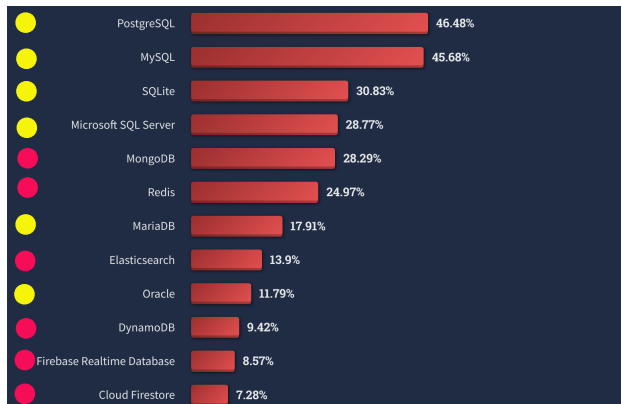
CP: MongoDB

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

- ▶ 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>

- ▶ <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>

Na zamyslenie

- ▶ Pozrite si syntax prvorádových formúl (link: kap. 1.3). Zdefinujte vhodné predikáty a zapíšte formuly zodpovedajúce tvrdeniam zo slajdu o predaji chleba v potravinách.
- ▶ Dokážte, že si vo formulách vystačíme so symbolmi \exists , \neg , \wedge (t.j. netreba ani všeobecný kvantifikátor, ani iné logické spojky; napr. Datalog iné symboly ani nemá).

Úlohy: relačný kalkul

Databáza: *osoba*(Meno), *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