BIG DATA 3

Technológie spracovania veľkých dát

Peter Bednár, Martin Sarnovský

Distribuované súborové systémy a databázy

- Súborové systémy
- Relačné databázy
- NoSQL databázy
  - Stĺpcovo orientované databázy
  - Úložiská kľúč-hodnota
  - Dokumentové databázy
  - Grafové databázy

Súborové systémy (1)

- Základnou jednotkou pre uchovanie dát je pomenovaný súbor
- Súborový systém nepredpisuje formát dát, súbor môže obsahovať ľubovoľné dáta (štruktúrované, alebo neštruktúrované), veľkosť súboru nie je pevne daná
- Efektívny prístup k dátam v rámci jedného súboru je ponechaný na aplikácii, súborový systém sa stará len o čo najrýchlejší zápis alebo čítanie dát z pamäťového média
- Základné operácie: vytvorenie nového súboru, zápis/čítanie dát do/z súboru, zmazanie súboru
  - Niektoré systémy podporujú len sekvenčné čítanie/zápis zo začiatku/konca súboru, niektoré aj prístup k ľubovoľnej pozícii v súbore

Súborové systémy (2)

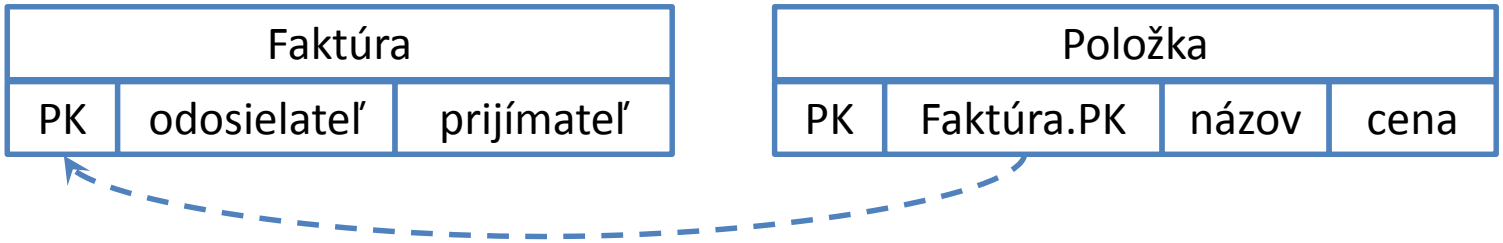
- Súbory sú organizované v hierarchickej adresárovej štruktúre
- Existuje jeden koreňový adresár, ktorý môže obsahovať pod-adresáre, ktoré môžu obsahovať pod-pod-adresáre atď.
- Úplný jedinečný identifikátor súboru je zložený z jeho názvu a cesty v adresárovej štruktúre
- Okrem názvu a obsahu spravuje súborový systém aj ďalšie metadáta o súbore resp. adresári, ako napr. meno používateľa ktorý súbor vytvoril/naposledy modifikoval, dátum a čas vytvorenia/poslednej modifikácie, prístupové práva a pod.

Relačné databázy (1)

- Relačné databázy sú najpoužívateľnejšou technológiou na uchovávanie štruktúrovaných dát
- Dáta sú uložené v relačnej štruktúre, ktorá má formu tabuľky
- Popisované objekty a vzťahy medzi nimi musia byť znormované do relačnej štruktúry:
  - Riadky tabuľky zodpovedajú záznamom, ktoré uchovávajú dáta o jednotlivých objektoch
  - Stĺpce zodpovedajú atribútom objektov

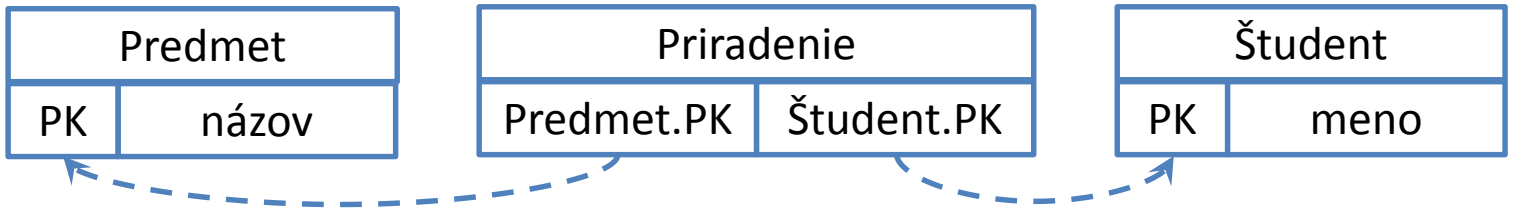
Relačné databázy (2)

- Aby sa mohli záznamy navzájom medzi tabuľkami odkazovať, záznamom v tabuľke je priradený jedinečný primárny kľúč (samostatný stĺpec)
- Relácia 1:N (napr. jedna faktúra obsahuje viacero položiek)



Relačné databázy (3)

- Relácia N:M (napr. jeden študent si môže zvoliť viacero predmetov a jeden predmet môže byť zvolený viacerými študentmi)
  - Vyžaduje samostatnú tabuľku, ktorá bude obsahovať odkazy na primárne kľúče oboch entít



Databázová schéma

- Tabuľky majú pevnú štruktúru, ktorú definuje databázová schéma
  - Dátové typy atribútov - čísla, Boolovské hodnoty, reťazce s pevnou a premenlivou dĺžkou, dátum a čas, atď.
- Schéma ďalej predpisuje ďalšie podmienky pre dodržanie konzistentnosti dát (napr. či daný stĺpec môže obsahovať nevyplnené údaje - NULL hodnota, alebo či majú byť všetky hodnoty v stĺpci jedinečné - UNIQUE a pod.)

Structured Query Language - SQL

- Pre prácu s dátami v relačných databázach sa používa štandardný jazyk SQL
- Umožňuje:
  - Definovanie a modifikáciu databázovej schémy
  - Vkladanie, modifikáciu a dopytovanie dát
    - Filtrovanie podľa podmienky
    - Usporiadanie podľa hodnôt atribútov (ORDER BY)
    - Zoskupovanie dát (GROUP BY)
    - Prepájanie dát z rôznych tabuliek (JOIN)
- SQL je deklaratívny jazyk, tzn. programátor predpisuje požadovanú operáciu, nie ako sa má vykonať (to závisí od implementácie databázového servera)

Fyzický prístup k dátam

- Aby bolo možné efektívne pristupovať k dátam, riadky tabuľky sú rozdelené na stránky ktoré sa zapisujú a čítajú z pamäťového média naraz
- Stránky majú pevnú veľkosť a vďaka pevnej štruktúre záznamov je možné priamo adresovať jednotlivé záznamy v stránke
- Stránky a záznamy v nich sú usporiadané podľa primárneho kľúča, tzn. databázový server dokáže efektívne vyhľadať záznam pre daný primárny kľúč s čo najmenším počtom stránok prečítaným do operačnej pamäte

Indexovanie

- Aby bolo možné efektívne filtrovať dáta aj podľa ďalších atribútov, pre tabuľku a atribút je možné vytvoriť index
- Index obsahuje usporiadaný zoznam hodnôt atribútu a primárny kľúč záznamu v ktorom sa hodnota vyskytla, napr.

Položka		
PK	názov	cena
1	jablká	1,5
2	hrušky	1,5
3	čerešne	1,8
4	zemiaky	0,8
5	broskyne	2,0
6	pomaranče	2,3

Položka_cena	
cena	PK
0,8	4
1,5	1
1,5	2
1,8	3
2,0	5
2,3	6

Transakcie

- K dátam môže pristupovať naraz viacero klientov, ktoré môžu dáta čítať alebo meniť v rovnakom čase
- Niektoré operácie vyžadujú aby boli naraz zmenené údaje vo viacerých záznamoch, tieto operácie musia byť spojené do jednej transakcie
- Napr. ak máme tabuľku účtov s ich aktuálnym stavom:
  1. Začiatok transakcie
  2. Odpočítaj peniaze od prvého účtu – operácia 1
  3. Pripočítaj peniaze k druhému účtu – operácia 2
  4. Ak sa nevyskytla chyba, prijmi obidve zmeny, inak zamietni obidve zmeny

ACID

- Transakcie musia spĺňať vlastnosti ACID
- Atomickosť (Atomicity) – buď sa úspešne vykonajú všetky operácie, ktoré sú súčasťou transakcie, alebo žiadna
- Konzistentnosť (Consistency) – pri ukončení transakcie (či už prijatím alebo zamietnutím zmien) ostanú dáta v konzistentnej podobe
- Izolácia (Isolation) – ak bude naraz prebiehať viacero transakcií od viacerých klientov, výsledok by mal byť rovnaký ako keby prebiehali sekvenčne
  - tzn. ak by najprv začala transakcia A, potom počas jej vykonávania začala transakcia B, nezávisle na tom v akom poradí by skončili, dáta by mali byť v rovnakom stave ako by najprv prebehla celá transakcia A a potom celá transakcia B
- Trvalosť (Durability) – po ukončení transakcie sú všetky zmeny trvalo uložené (aj v prípade následnej chyby na servery)

Príklad spájania dát z viacerých tabuliek (1)

Zamestnanec			
ID	meno	plat	oddelenieID
1	Ema	2300	1
2	Anna	800	3
3	Jano	950	2
4	Juraj	1100	2

Oddelenie	
ID	názov
1	Vedenie
2	Výroba
3	Učtáreň

```
SELECT * FROM Zamestnanec, Oddelenie WHERE
    Zamestnanec.plat > 1000 AND
    Zamestnanec.oddelenieID = Oddelenie.ID -- implicitný JOIN
```

Príklad spájania dát z viacerých tabuliek (2)

1. Ak existuje index pre Zamestnanec.plat, efektívne vieme získať ID zamestnancov s platom nad 1000 euro [1, 4], inak by sme museli načítať všetky záznamy zamestnancov do pamäti a porovnať ich platy
2. Podľa primárneho kľúča Zemestnanec.ID načítame záznamy zamestnancov 1 a 4 a zistíme ID ich oddelení
3. Podľa primárneho kľúča Oddelenie.ID načítame záznamy oddelení 1 a 2 a zistíme ich názov

- Pri zložitých vzťahoch musia byť relácie rozdelené do viacerých tabuliek – spájanie tabuliek však vyžaduje zložité operácie a častý prístup na disk

Rekurzívne dopyty

Tweet		
ID	text	retweet
1	Ahoj!	NULL
2	Ahoj!	1
3	mám sa fajn	NULL
4	Ahoj!	2

Ako môžeme zistiť všetky Tweety, ktoré boli rekurzívne preposlané z daného Tweetu?

Relačné databázy - zhrnutie

- Sú vhodné ak:
  - Vyžadujeme štandardné technológie, ktoré sú bežne dostupné a rozšírené
  - Požadujeme podporu ACID transakcií
- Nie sú vhodné ak:
  - Dáta obsahujú zložité/rekurzívne relácie, ktoré po znormovaní do relačnej formy vyžadujú viacnásobné spájanie tabuliek
  - Sú dáta heterogénne a často sa mení ich dátová schéma

NoSQL databázy

- Označujú skupinu rôznorodých technológií, ktoré nevyžadujú aby boli dáta modelované do relačnej štruktúry a uložené v dátových tabuľkách
- NoSQL neznamená, že databáza nepodporuje jazyk SQL!
- Často sú navrhnuté ako distribuované úložiská pre väčšiu škálovateľnosť a robustnosť voči chybám
  - Avšak na úkor striktnej konzistencie dát

NoSQL databázy – prehľad

- Stĺpcovo-orientované databázy
- Úložiska kľúč-hodnota
- Dokumentové databázy
- Grafové databázy

Stĺpcovo-orientované databázy (1)

- Dáta majú tabuľkovú formu, ale na rozdiel od relačných databáz sú hodnoty uložené po stĺpcoch
- Atribúty môžu odkazovať na pole hodnôt alebo vnorený objekt, ktorý nie je potrebné modelovať samostatnými tabuľkami
- Hodnoty sú usporiadané a odkazujú sa na primárny kľúč objektu podobne ako pri relačných indexoch
- Keďže atribút obsahuje hodnoty iba jedného typu, záznamy je možné efektívne skomprimovať
  - Viac dát je možné uchovávať priamo v operačnej pamäti, skracuje sa čas na zápis/čítanie z disku

Stĺpcovo-orientované databázy (2)

- Nie je potrebné vyhradiť miesto pre chýbajúce hodnoty
- Zefektívňuje sa spájanie viacerých tabuliek
- Menej efektívne je, ak je potrebné do výsledku zahrnúť viacero atribútov

Adresár		
ID	meno	tel. čísla
1	Ema	[]
2	Anna	[+12]
3	Jano	[+23]
4	Juraj	[+25, +26]
5	Peter	[+13]

Adresár.meno	
Anna	2
Ema	1
Jano	3
Juraj	4
Peter	5

Adresár.tel.čísla	
+12	2
+13	5
+23	3
+25	4
+26	4

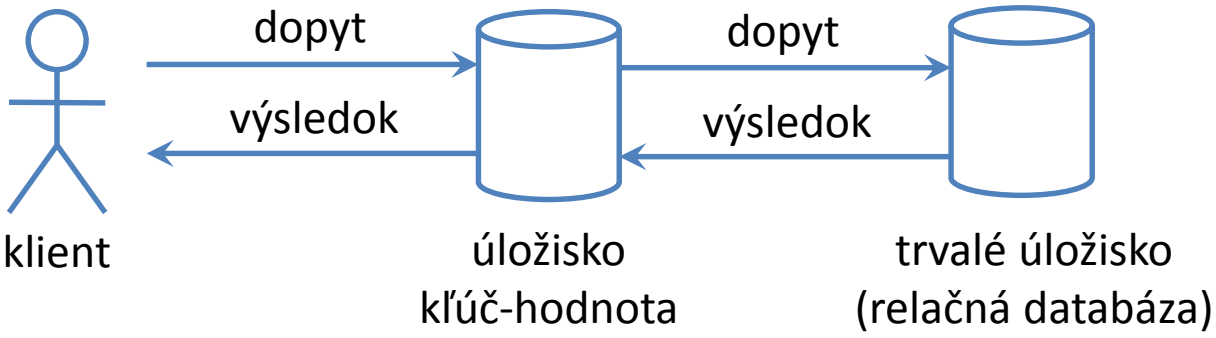
Úložiska kľúč-hodnota (1)

- Ide o úložiská s najjednoduchším modelom, ktorý rozlišuje iba primárny kľúč objektu a asociované dáta
- Zvyčajne nie je definovaná vnútorná štruktúra dát - interpretácia asociovanej hodnoty je na klientovi
- Veľmi efektívne operácie vyhľadávania a zmeny hodnoty podľa kľúča
  - Okrem usporiadania sa využíva aj hešovanie
- Nie je podporované vyhľadávanie podľa ďalších atribútov
- Nie je možné vrátiť iba časť uložených dát (dá sa prečítať iba celá asociovaná hodnota)
- Niektoré databázy podporujú uloženie viacerých hodnôt pre jeden kľúč (prístup k nim je však len sekvenčný)

Úložiska kľúč-hodnota (2)

- Často podporujú konfiguráciu, kde dáta sú primárne uložené v operačnej pamäti s voliteľnou možnosťou ich trvalo uložiť na disk
- Často sa používajú ako vyrovnávacia pamäť a podporujú:
  - Dočasné záznamy - automaticky sa odstránia po uplynutí časového intervalu od ich pridania
  - Prioritnú frontu - ak sa dosiahne stanovený počet záznamov, odstránia sa najstaršie záznamy

Vyrovnávacia pamäť pre dopyty



Ako kľúč sa použije dopyt pre ktorý sa ako hodnota uloží výsledok. Úložisko kľúč-hodnota uchováva výsledky iba v operačnej pamäti. Ak klient požiada o nový dopyt, výsledok sa získa z relačnej databázy a dočasne sa uloží do úložiska kľúč-hodnota.

Úložiská kľúč-hodnota - zhrnutie

- Sú vhodné ak:
  - Potrebujeme rýchlo spracovať často aktualizované dáta
  - Potrebujeme veľmi rýchli prístup, pričom postačuje, že dáta sú dostupné iba dočasne
  - Schéma dát sa môže meniť
- Nie sú vhodné ak:
  - Je potrebné vyhľadávať záznamy podľa hodnôt
  - Je potrebné spájať záznamy podľa viacerých kľúčov (táto operácia musí byť implementovaná na klientovi)

Dokumentové databázy (1)

- Základnou jednotkou pre ukladanie dát je dokument s priradeným jedinečným identifikátorom
- Dokumenty môžu mať zložitú štruktúru, atribúty dokumentu môžu mať dátovú hodnotu (reťazec, číslo, dátum, a pod.), pole hodnôt, alebo vnorené objekty
- Databáza je tvorená kolekciami, ktoré zoskupujú dokumenty, kolekcia však môže obsahovať dokumenty rôznych typov – dopredu nie je definovaná žiadna schéma
- Nie sú podporované odkazy medzi dokumentami – atribút dokumentu môže obsahovať ID iného dokumentu, ale dereferencovanie ID a prístup k odkazovanému dokumentu je na klientovi

Dokumentové databázy (2)

- Zvyčajne poskytujú rozhranie implementované ako REST služba – komunikuje sa priamo pomocou HTTP protokolu
- Na výmenu dát (a často aj na zápis dopytu) sa používajú štruktúrované textové dokumenty ako napr. JSON, alebo XML

JSON – JavaScript Object Notation

- Hodnota môže byť reťazec, celé číslo, desatinné číslo a Boolovské premenné true a false, a null, pole hodnôt alebo vnorený objekt
  - Ďalšie dátové typy musia byť zakódované ako jedna z podporovaných hodnôt, napr. dátum a čas ako reťazec podľa ISO normy: 2016-02-20T18:05:48+00:00
- Pomocou zápisu JSON Schema je možné popísať požadovanú štruktúru JSON dokumentov, avšak dokumentové databázy zvyčajne validáciu štruktúry nepodporujú

JSON – príklad dokumentu

```
{
        "firstName": "John",
        "lastName": "Smith",
        "isAlive": true,
        "age": 25,
        "address": {
                "streetAddress": "21 2nd Street",
                "city": "New York",
                "state": "NY",
                "postalCode": "10021-3100”
        },
        "phoneNumbers": [
                { "type": "home", "number": "212 555-1234” },
                { "type": "office", "number": "646 555-4567” }
        ],
        "children": [],
        "spouse": null
}
```

Dopytovanie v dokumentových databázach

- Dokumentové databázy podporujú zložité dopyty agregujúce dáta
- Priamo podporujú fazetové vyhľadávanie
  - Používateľ môže interaktívne definovať rôzne ohraničenia a rozdeliť dokumenty do skupín podľa ich vlastností, systém automaticky prepočíta ktoré dokumenty spĺňajú ohraničenia a koľko je zahrnutých do každej skupiny
- Dopytovacie jazyky zvyčajne nie sú čisto deklaratívne: klient môže definovať dopyt ako postupnosť zreťazených operácií nad dátami

Základné operácie pre dopytovanie (1)

- Filtrovanie
  - Z množiny dokumentov sa vyberú do ďalšieho spracovania iba dokumenty, ktorých hodnoty atribútov spĺňajú zadanú podmienku
  - Dokumentové databázy podporujú základné operátory pre porovnanie hodnôt atribútov ako napr. =, <>, <, > ale aj zložitejšie testy ako napr. testovanie či bod s geografickou polohou patrí do zadanej oblasti, alebo jeho vzdialenosť od zadaného bodu je v danom intervale
  - V testoch sa možno odkazovať na prvky vnorených polí alebo objektov

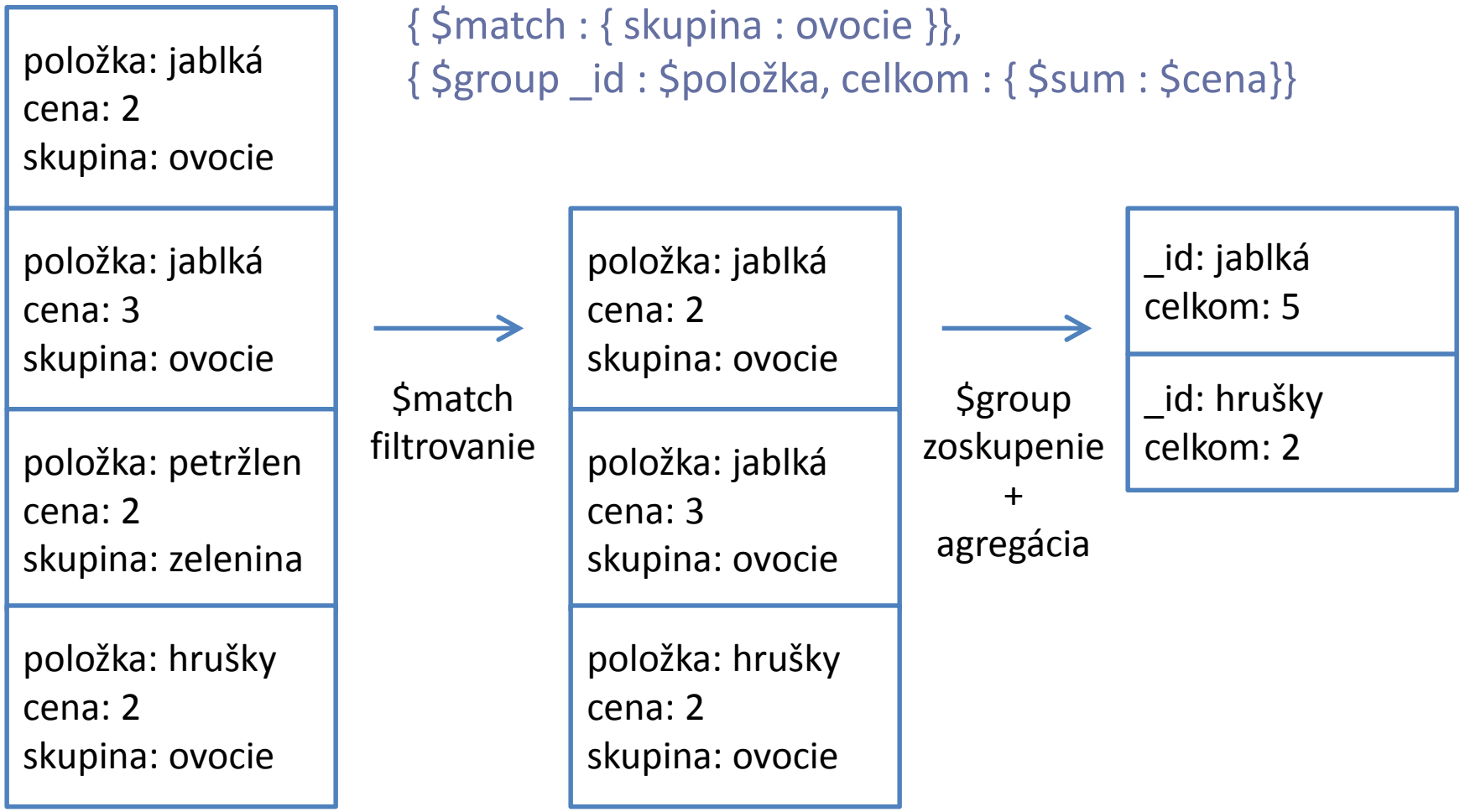
Základné operácie pre dopytovanie (2)

- Projekcia
  - Každý dokument z množiny sa transformuje na nový dokument resp. dokumenty, ktoré obsahujú atribúty odvodené z atribútov pôvodného dokumentu
  - Hodnoty môžu byť skopírované alebo transformované zvolenou funkciou (napr. čísla môžu byť zaokrúhlené, alebo sa vyextrahuje deň v týždni z dátumu) resp. nová hodnota môže vzniknúť kombináciou viacerých pôvodných hodnôt (napr. je možné vypočítať cenu s DPH z atribútov cena a DPH položky)

Základné operácie pre dopytovanie (3)

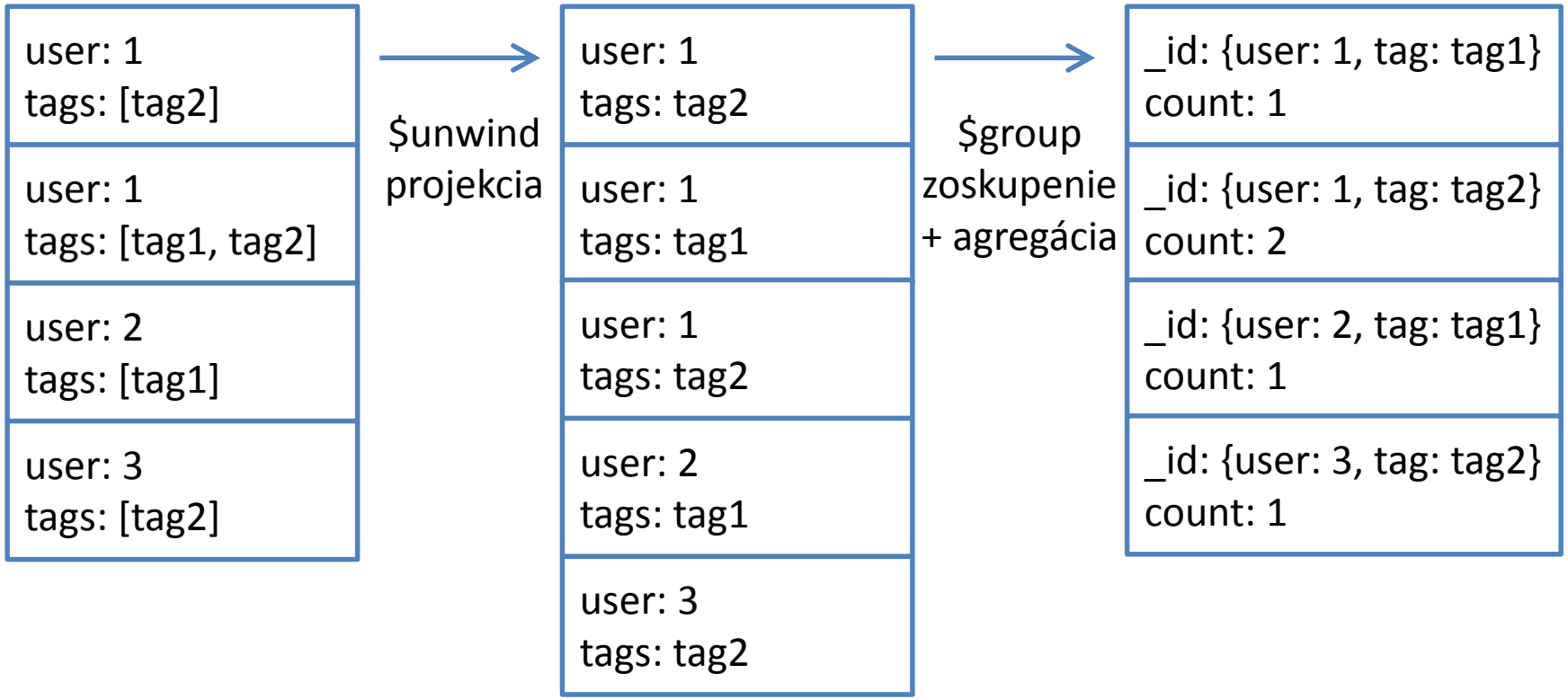
- Zoskupovanie
  - Množina dokumentov sa rozdelí na podmnožiny podľa hodnoty zadaného atribútu (pre diskrétne hodnoty alebo pre intervaly spojitých atribútov)
- Agregácia
  - Z množiny dokumentov sa vypočítajú agregované hodnoty podľa zvolenej funkcie (počet dokumentov v množine, min, max, sum, avg hodnôt zvolených atribútov)
  - Najčastejšie sa používa s operáciou zoskupenia, kedy sú vo výsledku zahrnuté agregácie pre každú podmnožinu
- Usporiadanie
  - Množina dokumentov sa usporiada podľa zvoleného kritéria (hodnoty atribútu zostupne/vzostupne alebo napr. podľa geografickej vzdialenosti od zadaného bodu)

Príklad 1 v MongoDB



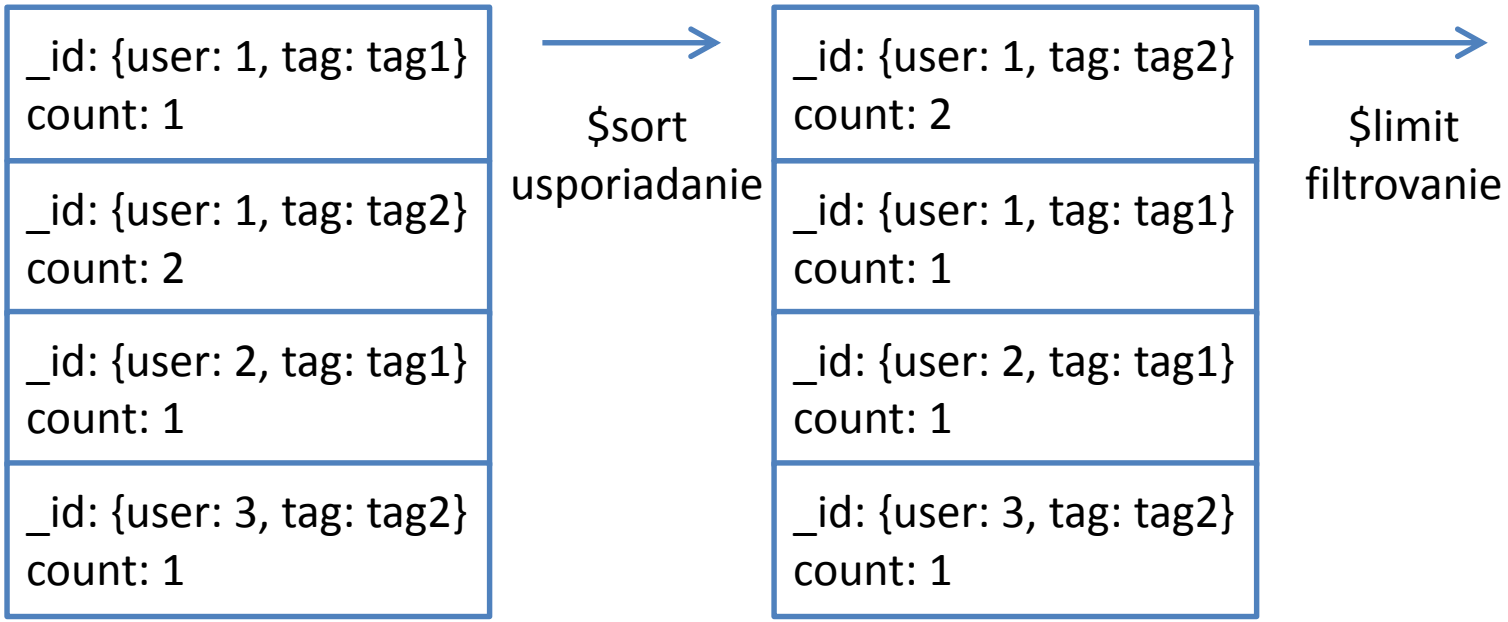
Príklad 2 v MongoDB (1)

{ $unwind: $tags}, -- rozbalíme pole tagov
{ $group _id : {user: $user, tag: $tags}, count : { $sum : 1 }} -- zoskupenie podľa viacerých atribútov



Príklad 2 v MongoDB (2)

{ $sort: {$count : -1}}, -- usporiadanie podľa počtu zostupne
{ $limit: 1} -- vrátime iba najfrekventovanejšiu dvojicu

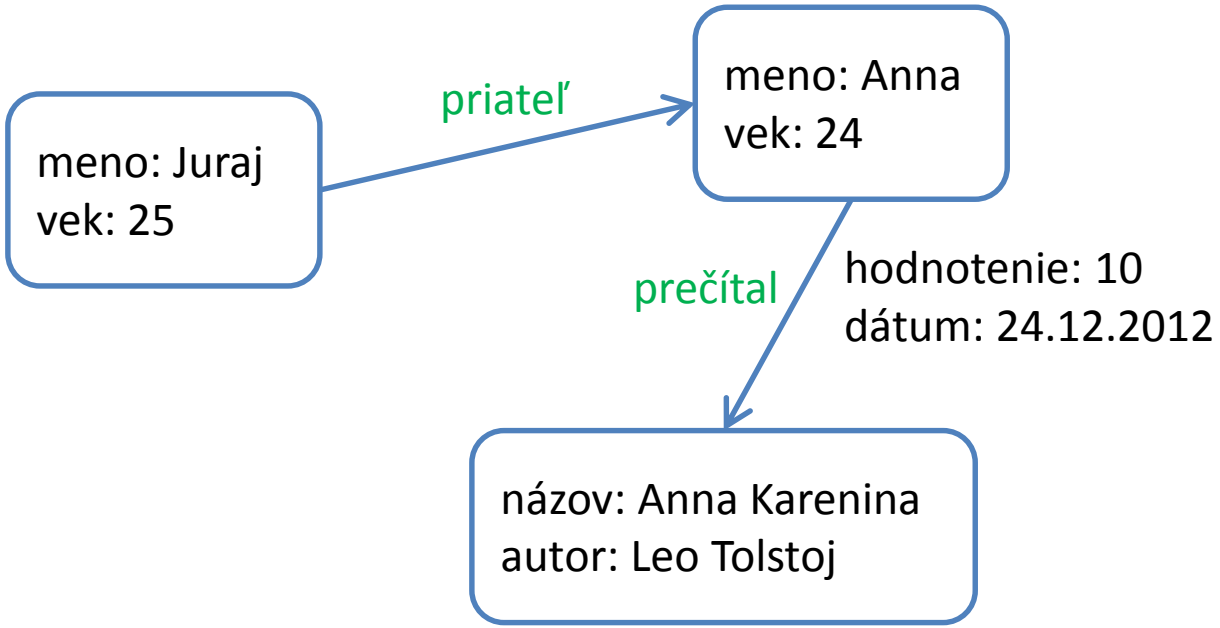


Dokumentové databázy - zhrnutie

- Sú vhodné ak:
  - Máme heterogénne dáta, ktorých schéma sa často mení
  - Máme komplexnú štruktúru objektov s rôznymi dátovými typmi atribútov, vnorenými objektmi a poľami
  - Uprednostňujeme rýchle čítanie dát pred striktnou konzistentnosťou
- Nie sú vhodné ak:
  - Dáta obsahujú prepojenia medzi dokumentami
  - Požadujeme atomické operácie medzi viacerými dokumentami
  - Požadujeme častú zmenu atribútov, hlavne vnorených objektov

Grafové databázy (1)

- Dáta sú uložené v podobe grafov zložených z:
  - Uzlov ktoré reprezentujú entity a
  - Orientovaných hrán, ktoré reprezentujú relácie medzi entitami
- Uzly a hrany môžu mať priradené dátové vlastnosti a typ



Grafové databázy (2)

- Medzi grafové databázy môžeme zaradiť aj RDF úložiská
- RDF – Resource Description Framework – model dát pre aplikácie sémantického webu a linkovaných dát
- Dáta sú uložené ako množina tripletov <subjekt, predikát, objekt>:

```
<osoba1>,   <meno>,     "Juraj"     -- dátové vlastnosti
<osoba1>,   <priateľ>,  <osoba2>    -- relácie
<osoba2>,   <meno>,     "Anna"
<osoba2>,   <prečítal>, <kniha1>
<kniha1>,   <názov>,    "Anna Karenina"
```

Dopytovanie v grafových databázach (1)

- Pre dopytovanie v grafových databázach je charakteristické rekurzívne prechádzanie medzi uzlami cez hrany, pričom je možné definovať podmienky pre uzly alebo hrany na prejdenej ceste

- Zjednodušené príklady v jazyku Cypher (Neo4j):

  Vráť Jurajových priateľov:

  ```
  MATCH ({meno: Juraj})  -[:priateľ]-> (osoba)
  RETURN osoba
  ```

  Vráť knihy, ktoré prečítali Jurajovi priatelia:

  ```
  MATCH ({meno: Juraj})  -[:priateľ]-> () -[:prečítal]-> (kniha)
  RETURN kniha
  ```

Dopytovanie v grafových databázach (2)

- Je možné definovať minimálny a maximálny počet prechodov cez daný typ hrany, pričom maximálny počet môže byť aj neobmedzený (nie je ohraničená hĺbka rekurzie)

  (uzol1) - [:označenie min.počet:max.počet]-> (uzol2)

- Vráť osoby, ktorí sú rekurzívne priatelia Jurajových priateľov:

  MATCH ({meno: Juraj}) -[:priateľ 2:*]-> (osoba)

  RETURN osoba

Grafové databázy - zhrnutie

- Sú vhodné ak:
  - Máme dáta s veľkým počtom relácií, ktoré je prirodzené modelovať ako grafy (sociálne siete, odporúčacie systémy, hierarchie a pod.)
  - Požadujeme dopyty s veľkou alebo neobmedzenou hĺbkou rekurzie
- Nie sú vhodné ak:
  - Uchovávame binárne dáta
  - Zapisujeme naraz veľký objem dát