Databázové systémy

doc. Ing. Marcela Hallová, PhD.

Ústav účtovníctva a informatiky

2. poschodie, č.d. 216

email: marcela.hallova @uniag.sk

Podmienky absolvovania predmetu

Zápočtový test praktický – **30** bodov

Zápočtový test teoretický – **20** bodov

Semestrálny projekt – **20** bodov

Praktická skúška – **30** bodov

Semestrálny projekt – vytvorenie vlastnej databázy z akejkoľvek oblasti, využitie nadobudnutých znalostí.

IBM SkillsBuild platforma

https://skillsbuild.org/
Odporúčaný kurz:
Data Fundamentals (Earn a credential!)
– možnosť náhrady seminárnej práce za 20 bodov.

Link pre prihlásenie:
https://sb-auth.skillsbuild.org/signup?ngo-id=0407

IBM SkillsBuild

Databáza, DBMS, dátový model

Databáza je organizovaná kolekcia údajov.

Databázový systém, resp. systém riadenia bázy dát (DBMS – Database Management System), je softvér, ktorý umožňuje údaje ukladať, vyhľadávať, meniť, chrániť, zdieľať a obnovovať.

Dátový model určuje, akým spôsobom sú dáta logicky organizované a aké vzťahy medzi nimi vieme vyjadriť.

Pojem	Význam	Príklad
Databáza	Konkrétne uložené dáta a ich štruktúra	evidencia študentov, predmetov a známok
DBMS	Softvér riadiaci databázu	PostgreSQL, Oracle Database, MongoDB
Dátový model	Logický spôsob reprezentácie údajov	relačný, dokumentový, grafový
Dotazovací jazyk/API	Prostriedok komunikácie s DBMS	SQL, Cypher, MongoDB Query API

Pred databázami: súborovo orientované spracovanie

studenti.dat
predmety.dat
znamky.dat
ucitelia.dat

- Prvé informačné systémy ukladali dáta priamo do súborov.
- Každá aplikácia poznala ich konkrétny formát a sama implementovala logiku načítania, vyhľadávania a aktualizácie.
- Takýto prístup funguje pri malých a izolovaných úlohách, no pri rastúcom objeme údajov prináša zásadné problémy.

Pred databázami: súbory

TechShop začína jednoducho – tri CSV súbory.

```
customers.csv
1;Peter
Novák;peter@email.sk;Nitr
a
2;Jana
Malá;jana@email.sk;Bratis
lava
```

```
products.csv
101;Notebook
Lenovo;899;12
102;Logitech
Mouse;39;45
```

```
orders.csv
5001;1;101;1
5002;2;103;2
```

Čo musí riešiť aplikácia sama?

- vyhľadávanie naprieč súbormi
- súbežný zápis viacerých používateľov
- konzistenciu a obnovu po chybe
- zmenu formátu a väzbu programu na dáta

Čo sa stane, ak dvaja zákazníci naraz kúpia posledný kus?

Pred databázami: súborovo orientované spracovanie

- Silná závislosť programu od fyzickej štruktúry súborov.
- Duplicitné dáta v rôznych aplikáciách.
- Ťažšie udržiavanie konzistencie.
- Komplikovaný súbežný prístup viacerých používateľov.
- Náročné zabezpečenie, auditovanie, zálohovanie a obnova.
- Potreba programovať nové procedúry pri každom novom type otázky nad dátami.

Hierarchické a sieťové modely

Hierarchické modely

- V 60. rokoch sa rozšírili hierarchické databázy.
- Dáta organizovali do stromovej štruktúry rodič–potomok.
- Takýto model bol efektívny pri úlohách, ktoré prirodzene kopírovali hierarchiu, no bol ťažko použitešný pri zložitejších vzťahoch.

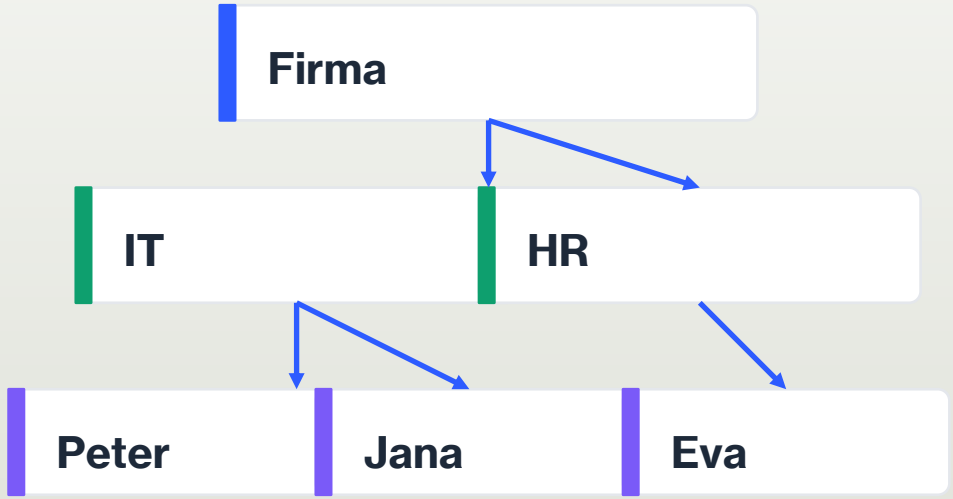
Sieťové modely

- Umožnil záznamom zložitejšie väzby a viacero ciest medzi objektmi.
- Programátor však často musel poznať navigačné cesty v dátach.
- Aplikácia teda bola stále výrazne previazaná so spôsobom, akým boli vzťahy fyzicky alebo logicky reprezentované.

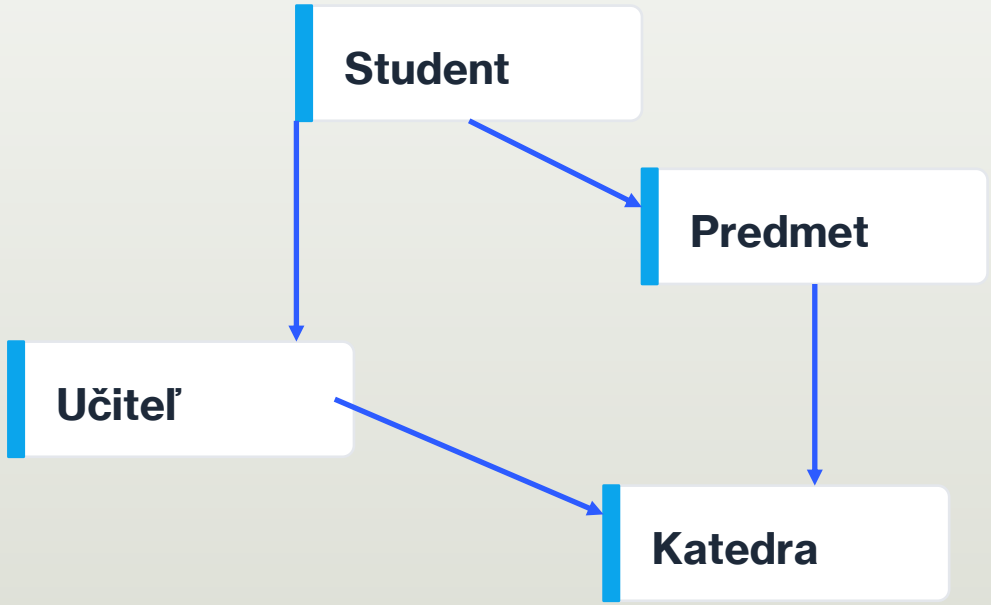
Hierarchický a sieťový model

Prvé DBMS riešili správu dát, ale navigácia bola stále silno viazaná na štruktúru.

Hierarchický model



Sieťový model



Problém: aplikácia často musí poznať cestu, ktorou sa k dátam dostane.

Coddov relačný model: zásadný obrat (1970)

- Relačný model navrhol reprezentovať údaje pomocou relácií a manipulovať s nimi deklaratívnejším spôsobom, ktorý používateľa oddeľuje od detailov fyzického uloženia.
- Coddova práca sa stala teoretickým základom relačných databáz.
- Kľúčový posun nebol iba v tom, že údaje „vyzerajú ako tabuľky“.
- Podstatná bola dátová nezávislosť: používateľ alebo aplikácia môže formulovať otázku nad logickou štruktúrou dát bez explicitného programovania fyzickej navigácie po uložených záznamoch.

1970: Coddov relačný model

Zásadná zmena: pracujeme s logickými reláciami a deklaratívnymi dotazmi.

STUDENT

ID	Meno	Mesto
1	Peter	Nitra
2	Jana	Bratislava
3	Adam	Košice

PREDMET

ID	Názov
10	Databázy
11	Programovanie

```
SELECT meno
FROM studenti
WHERE mesto = 'Nitra';
```

Kľúčová myšlienka

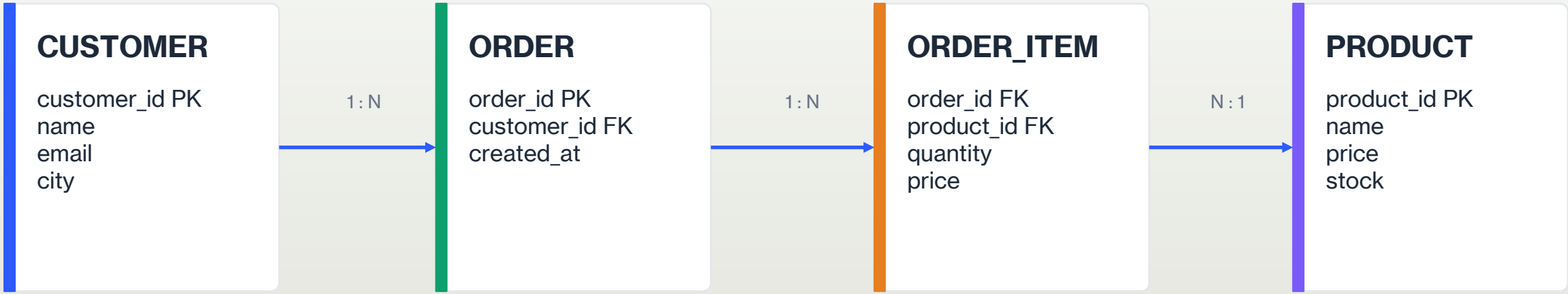
Používateľ opisuje, ČO chce. DBMS rozhoduje, AKO sa k výsledku dostane.

System R, SQL a optimalizácia dotazov

```
SELECT meno,
priezvisko
FROM student
WHERE rocnik =
2
ORDER BY
priezvisko;
```

- V roku 1973 začalo IBM pracovať na projekte System R, ktorý mal ukázať, že relačný model možno implementovať ako praktický databázový systém.
- System R sa stal významným testovacím prostredím pre SQL a pre vývoj automatickej optimalizácie dotazov.
- Donald Chamberlin a Raymond Boyce vytvorili jazyk SEQUEL, neskôr známy ako SQL.
- SQL je deklaratívny jazyk: používateľ prevažne opisuje, aký výsledok chce.

TechShop prechádza na relačnú databázu




```
SELECT c.name, o.order_id, p.name, oi.quantity
FROM customer c
JOIN orders o ON o.customer_id = c.customer_id
JOIN order_item oi ON oi.order_id = o.order_id
JOIN product p ON p.product_id = oi.product_id;
```

Rozmach relačných databáz

- Od konca 70. a počas 80. rokov sa relačné systémy komercializovali a postupne sa stali dominantným modelom pre podnikové spracovanie dát.
- Dôležitou vetvou vývoja bol akademický projekt POSTGRES na University of California, Berkeley,
- Implementácia POSTGRES sa začala v roku 1986. Z tohto projektu sa neskôr vyvinul PostgreSQL, dnešný objektovo-relačný open-source DBMS.
- Medzi známe relačné databázové systémy dnes patria PostgreSQL, MySQL, MariaDB, Oracle Database, Microsoft SQL Server, IBM Db2 a ďalšie.

Nová vlna nerelačných databáz

- S rozmachom webových služieb, vyhľadávačov, sociálnych sietí, mobilných aplikácií a cloudovej infraštruktúry sa objavili štruktúry s enormným objemom dát, vysokou rýchlosťou zápisu, globálnou distribúciou a rýchlo sa meniacimi dátovými štruktúrami.
- Vznik nového pojmu – **NoSQL**.
- No SQL nie je jedna databáza ani jeden presne definovaný model.
- Je to zastrešujúce označenie rôznych nerelačných databázových rodín.

Výraz „NoSQL“ je najlepšie chápať ako rodinu alternatívnych dátových modelov a architektúr, nie ako tvrdenie „SQL sa nesmie používať“.

Hlavné rodiny NoSQL databáz - dokumentové databázy

- Dokumentové databázy ukladajú záznamy ako dokumenty.
- V dokumentovej databáze môžeme všetky údaje, ktoré patria k jednej veci, uložiť spolu v jednom dokumente.
- Dokumentová databáza má dátový model, len býva flexibilnejší a jeho zmena môže byť menej nákladná.
- Typické využitie: produktové katalógy, content management, profily používateľov, aplikačné dáta s prirodzene vnorenou štruktúrou.

```
{
  "customer_id": 125,
  "name": "Peter Novak",
  "address": {"city": "Nitra", "country": "Slovakia"},
  "orders": [{"product": "Notebook", "price": 1200}]
}
```

TechShop: dokumentový katalóg

Produkty majú veľmi odlišné vlastnosti. Dokumentový model môže byť prirodzenejší.

```
{
  "_id": 101,
  "name": "Lenovo ThinkPad",
  "category": "notebook",
  "specifications": {
    "cpu": "Intel Core i7",
    "ram": "16 GB",
    "ssd": "512 GB"
  }
}
```

```
{
  "_id": 501,
  "name": "TechShop T-shirt",
  "category": "clothing",
  "specifications": {
    "size": ["S", "M", "L"],
    "color": "black",
    "material": "cotton"
  }
}
```

Hlavné rodiny NoSQL databáz - key-value databázy

- Key-value databáza = kľúč → hodnota
- Každý údaj má:
  - kľúč (key) – podľa neho údaj nájdeme,
  - hodnotu (value) – samotný uložený obsah.
- Key-value databáza ukladá údaje ako dvojice „kľúč → hodnota“, podobne ako slovník, v ktorom podľa jedinečného kľúča veľmi rýchlo nájdeme požadovanú hodnotu.
- Typické využitie: nákupné košíky, počítadlá, fronty, real-time stav

KĽÚČ		HODNOTA
cart:125	→	Notebook, 2 kusy
cart:126	→	Myš, 1 kus
cart:127	→	Monitor, 2 kusy

Hlavné rodiny NoSQL databáz - Wide-column / column-family databázy

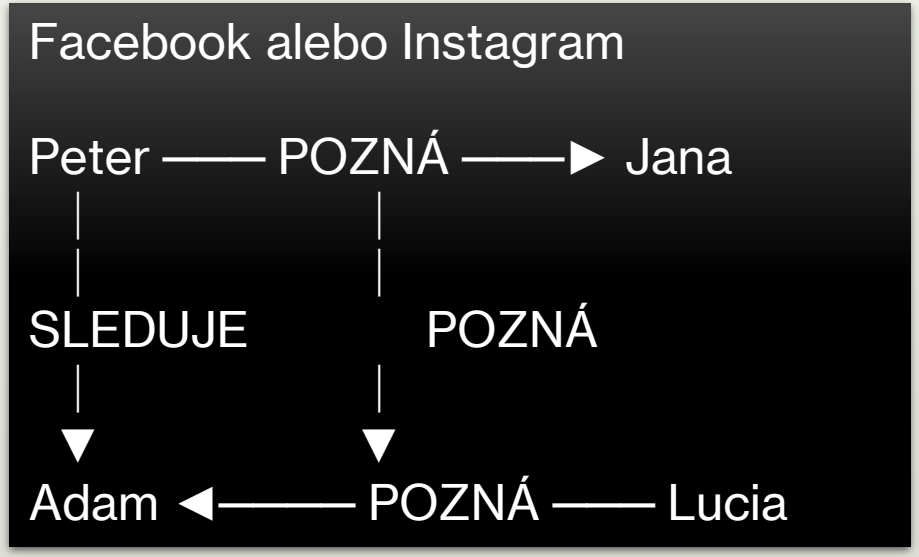
- Wide-column databázy organizujú dáta okolo riadkových kľúčov a rodín stĺpcov, pričom jednotlivé riadky nemusia mať identickú sadu stĺpcov.
- Wide-column databázy organizujú dáta okolo riadkových kľúčov a rodín stĺpcov, pričom jednotlivé riadky nemusia mať identickú sadu stĺpcov.
- Typické využitie: masívne distribuované datasety, časové rady, udalosti, štruktúry s vysokým objemom zápisov

```
user:101
 ├──── meno  → Peter
 ├──── email → peter@mail.sk
 └──── mesto → Nitra
```

```
user:102
 ├──── meno                → Jana
 ├──── mesto               → Bratislava
 ├──── posledné_prihlásenie → 12.9.2026
 └──── počet_objednávok    → 37
```

Hlavné rodiny NoSQL databáz - grafové databázy

- Grafové databázy modelujú dáta pomocou uzlov, hrán/vzťahov a vlastností.
- Výhodou je prirodzená práca s problémami, pri ktorých sú vzťahy rovnako alebo viac dôležité než samotné entity.

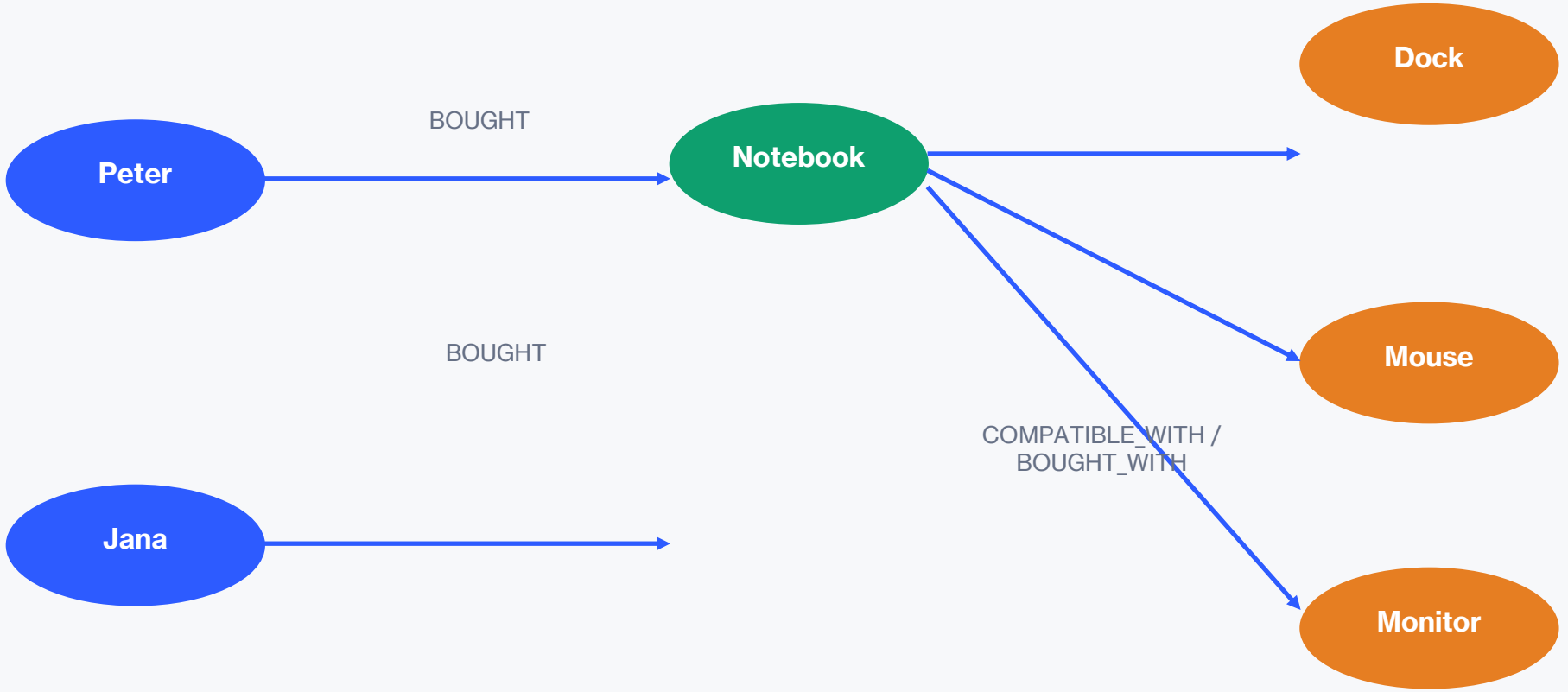


Máme tu dve základné veci:
- uzly (nodes) → Peter, Jana, Adam, Lucia,
- vzťahy (edges) → POZNÁ, SLEDUJE.

Pri relačnej databáze sa často pýtame:
„Aké údaje máme?“
Pri grafovej databáze sa často pýtame:
„Ako spolu tieto údaje súvisia?“

TechShop: keď sú najdôležitejšie vzťahy

Odporúčania a kompatibilita produktov sa prirodzene dajú modelovať grafom.



Grafový model

NODE

EDGE

PROPERTY

Vzťah je prvotriedny prvok modelu.

Relačné databázy (SQL)	Nerelačné databázy (NoSQL)
Dáta najmä v tabuľkách	Dáta v rôznych formách
Riadky a stĺpce	Dokumenty, key-value, grafy, wide-column
Zvyčajne pevnejšia schéma	Často flexibilnejšia schéma
Vzťahy pomocou kľúčov a JOIN	Vzťahy závisia od typu databázy
Silná podpora transakcií a integrity	Vlastnosti sa líšia podľa konkrétneho systému
Typicky používajú SQL	Používajú rôzne jazyky a API
PostgreSQL, MySQL, Oracle	MongoDB, Redis, Cassandra, Neo4j
Vhodné napr. pre objednávky, platby, účtovníctvo	Vhodné napr. pre katalógy, veľké distribuované dáta, grafy

SQL a NoSQL nie sú súperi. Každý prístup je vhodný na iný typ problému a v jednej aplikácii môžeme používať oba.

Ako vybrať vhodný databázový model

Dôležitá otázka: **Aké dáta máme a čo s nimi potrebujeme robiť?**

Ak potrebujeme...	**Vhodný model**	**Príklad**
Presne štruktúrované dáta a vzťahy	Relačný	PostgreSQL, MySQL
Flexibilné údaje	Dokumentový	MongoDB
Veľmi rýchlo nájsť hodnotu podľa kľúča	Key-value	Redis
Spracovať obrovské množstvo distribuovaných dát	Wide-column	Cassandra
Pracovať hlavne so vzťahmi medzi objektmi	Grafový	Neo4j

TechShop: celá evolúcia na jednom mieste

Súbory
CSV

jednoduchosť

RDBMS
PostgreSQL

integrita + transakcie

Document
MongoDB

heterogénny katalóg

Key-value
Redis

košíky + sessions

Graph
Neo4j

vzťahy + odporúčania

Každý krok rieši nový problém. Predchádzajúca technológia nemusí zmiznúť.

Súčasné trendy: cloud, distributed SQL, multimodel...

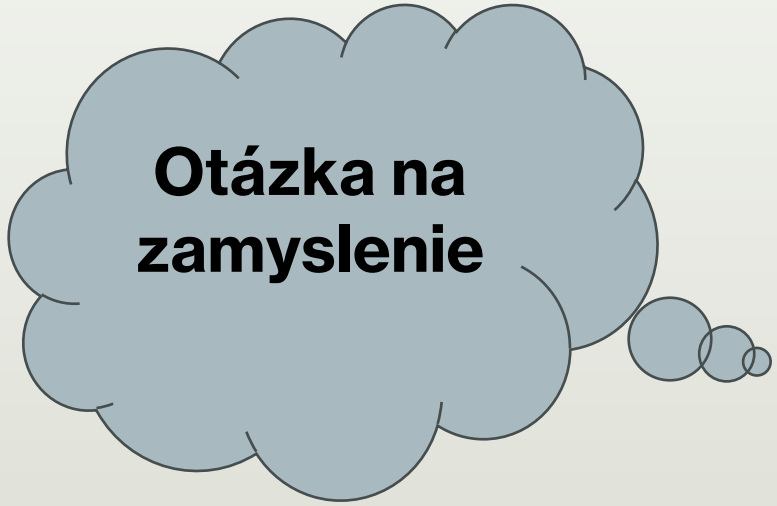
- **Cloud-managed databázy:** poskytovateľ automatizuje časť prevádzky, záloh, patchovania a škálovania.
- **Distributed SQL / NewSQL:** systémy sa snažia kombinovať SQL a relačný model s horizontálnou distribúciou a silnými transakčnými garanciami.
- **Multi-model databázy:** jeden produkt podporuje viac reprezentácií alebo vyhľadávacích mechanizmov, napr. relačné dáta, grafové či vektorové operácie.
- **In-memory spracovanie:** pracovná množina dát sa drží v pamäti s cieľom nízkej latencie.
- **Real-time analytics:** snaha zmenšovať hranicu medzi transakčným a analytickým spracovaním.

Zhrnutie

- História databázových systémov nie je jednoduchý rad náhrad, v ktorom nová technológia zruší predchádzajúcu.
- Ide o postupné rozširovanie priestoru možných riešení.
- Súborové spracovanie ukázalo potrebu centrálnej správy dát.
- Hierarchické a sieťové systémy priniesli organizovanejšie modely, no ostávali navigačne zložité.
- Coddov relačný model oddelil logický pohľad od fyzickej navigácie a spolu so SQL a optimalizátormi vytvoril základ moderných podnikových databáz.
- NoSQL databázy vznikli ako odpoveď na nové dátové modely a distribuované workloady, nie ako univerzálne „lepšie SQL“.
- Dokumentové, key-value, wide-column a grafové databázy riešia odlišné typy problémov.
- Dnešná prax preto často kombinuje viac technológií a hranice medzi kategóriami sa postupne prelínajú.


HISTÓRIA DATABÁZOVÝCH SYSTÉMOV
Nie je to koniec cesty.
Je to rozširovanie možností.
1. SÚBOROVÉ SPRACOVANIE
(60. – 70. roky)
Dáta v samostatných súboroch
Duplicitné dáta
Ťažká správa a zdieľanie
KDE SME BOLI
2. HIERARCHICKÉ a SIEŤOVÉ SYSTÉMY
(70. – 80. roky)
Organizovanejšie modely
Ale stále zložité na navigáciu
3. RELAČNÝ MODEL
(70. roky – dodnes)
Coddov model
Oddelenie logického pohľadu od fyzickej navigácie
SQL
Optimalizátory
Základ moderných podnikových databáz
4. NoSQL
(2000+)
Odpoveď na nové dátové modely a distribuované workloady.
Dokumentové
Key-value
Wide-column
Grafové
Rôzne databázy riešia odlišné typy problémov, nie sú univerzálne „lepšie SQL“.
5. SÚČASNOSŤ
(ČASTO VIAC TECHNOLÓGIÍ)
Kombinácia viacerých technológií
Rôzne úlohy, rôzne databázy
Hranice medzi kategóriami sa prelínajú
Viac možností = lepšie riešenia
ŠIRŠIE MOŽNOSTI
RÔZNE CESTY. JEDEN CIEĽ.
LEPŠIE RIEŠENIA.
NEJDE O TO, KTORÁ DATABÁZA JE NAJLEPŠIA,
ALE O TO, KTORÁ NAJLEPŠIE RIEŠI NÁŠ PROBLÉM.
DÁTA SPOJUJÚ SVETY
VIAC DÁT.
VIAC NÁPADOV.
LEPŠIA BUDÚCNOSŤ.
DATABÁZY TVORIA SVET OKOLO NÁS.

Ďakujem za pozornosť!



Predstavte si, že máte navrhnúť databázové riešenie pre novú sociálnu sieť.
Aký typ databázy (alebo databáz) by ste použili a prečo?

Zamyslite sa nad tým, aké dáta by sociálna sieť uchovávala, aké vzťahy medzi nimi existujú a či by na všetko stačil jeden databázový systém.