

<http://www.dcs.fmph.uniba.sk/~plachetk/TEACHING/DB1>

Tomáš Plachetka

Fakulta matematiky, fyziky a informatiky,
Univerzita Komenského, Bratislava

Zima 2025–2026

Organizácia kurzu: prednáška, konzultácie

Prednáška: doc. Dr. T. Plachetka

Pondelok 12:20–14:00, F1.328

Tento kurz je dvojsemestrálny, na tento predmet nadväzuje (v magisterskom štúdiu) predmet **Databázy**

Konzultácie:

M.262, Katedra informatiky

Čas odporúčam dohodnúť osobne (keď komunikujete cez email, navrhnite konkrétny čas aj nejaké alternatívy)

V skúškovom období konzultácie nie sú

Organizácia kurzu: cvičenia

Cvičenia: doc. RNDr. J. Mazák PhD.

Utorok 14:00, F.sklenik

Štvrtok 14:00, F1.328

Cvičenia začínajú krátkou hodnotenou "rozcvičkou" (ca. 5 minútovým písomným testom)

~~Na web stránke prednášky~~

~~<http://www.dcs.fmph.uniba.sk/~plachetk/TEACHING/DB1>~~

~~nájdete link na [systém organizácie cvičení](#). Platí rozvrh v tomto systéme. Nájdite svoje zaradenie do skupiny a ak treba, zmeňte ho~~



Úvod do databázových systémov cvičenia

Login

Študent

- Zmeniť e-mail
- Zmeniť heslo

Menu

- Hlavná stránka
- Rozvrh podľa mena
- Rozvrh podľa skupín
- Zmeny rozvrhu, obsadenosť skupín

Obsadenie skupín podľa predmetov

DB cvičenie

Názov	Čas	Miestnosť	Voľných/Celkom /Fronta	Stav	Zmeniť stav?
C1	Stv 12:20	F1.108	1/23/0	Sem nechcem	Zmeniť
C2	Stv 14:00	F1.108	4/23/0	Tu som	

Praktikum z DB

Názov	Čas	Miestnosť	Voľných/Celkom /Fronta	Stav	Zmeniť stav?
P1	Úto 12:20	M.217	3/22/0	Tu som	
P2	Str 18:10	M.217	0/22/0	Sem nechcem	Zmeniť

Tip: Zmenu robte len v nutnom prípade, inak môžete prísť o miesto v pôvodnej skupine

Databázové praktikum:

RNDr. M. Rjaško PhD.

Pondelok 16:30, I.H3

Výberový predmet ponúkaný a silne odporúčaný študentom kurzu
Úvod do databázových systémov.

Cieľ: získať **praktické zručnosti** pre prácu s databázovými systémami a súvisiacimi technológiami

Cvičenia

Hodnotia sa "**rozcvičky**" (možno pribudne niekoľko domácich úloh),
každé jednotlivé hodnotenie je z {0, 2, 5, 6}

$C = \max ($
 Σ najlepších 10 hodnotení rozcvičiek;
 Σ najlepších 9 hodnotení rozcvičiek a 1 najlepšieho
 hodnotenia z domácich úloh
 $)$

Nutná podmienka postupu: $C \geq 45\%$, t.j. **aspoň 27 bodov zo 60**

Písomný test v skúškovom období

Nutná podmienka postupu: $P \geq 45\%$, t.j. **aspoň 27 bodov zo 60**

Ústna skúška v skúškovom období

Organizácia: ročníkové projekty

Pondelok 22.9. o 18:10, F1

Jednorázové stretnutie venované organizácii ročníkových projektov
(pripomienku dostanete cez AIS email)

Dovtedy odporúčam pozrieť si projekty z uplynulých rokov na

<http://www.dcs.fmph.uniba.sk/~plachetk/TEACHING/RP/>

a premyslieť si, či máte vlastný nápad hodný ročného úsilia (a neskôr bakalárskej práce)

Čo je databáza (a čo je databázový systém)

Databáza je **kolekcia nejakých dát (údajov)**. Napríklad databáza kníh (alebo hudobných CD, študentov na študijnom oddelení, ...)

Dáta majú štruktúru. Aj dve veľmi rôzne knihy majú **spoločné atribúty** (autor, názov, vydavateľ, rok vydania, ISBN atď.), **líšia sa len v hodnotách atribútov**

Idea: Vyrobiť **univerzálny systém**, ktorý vie **pracovať s ľubovoľnou databázou** (pevnou štruktúrou)

Databázové systémy (DBMS) implementujú to, čo je v rôznych databázových aplikáciách spoločné

Príklady použitia databázových aplikácií

- Banky, poisťovne
- Akciové a iné burzy (eBay)
- Knižnice
- Rezervačné systémy (letecké spoločnosti, železnice, hotely, cestovné kancelárie, požičovne áut, ...)
- Supermarkety
- Firemná agenda (účtovníctvo, sklady, adresáre, ...)
- Fyzika (astronómia, meteorológia, geológia, geografia, ...)
- Štatistika
- ...

Spoločné charakteristiky DB aplikácií

- **Dáta majú štruktúru**, ktorá sa zriedka mení
- **Objem dát je veľký**
- Dáta nie sú uložené na užívateľovom počítači, sú prístupné cez počítačovú sieť (**klient-server**)
- Vyžaduje sa **vysoký stupeň bezpečnosti**
- **Dotazy sú zložité** („Koľko televízorov, ktoré našej firme dodávajú iba európski dodávatelia, sme predali vlani v Terchovej?“)
- Množstvo užívateľov **pristupuje k dátam súčasne**. Niektorí **dáta nielen čítajú, ale aj menia**
- **Vyžaduje sa vysoká odolnosť voči poruchám** (aj v prípade akejkoľvek havárie na strane klienta či servera musí databáza ostať v konzistentnom stave)
- Prístup koncových užívateľov k dátam musí byť jednoduchý (**API, GUI**)

„Definícia“ databázového systému (DBMS)

DBMS (Database Management System):

Systém, ktorý poskytuje pohodlné, bezpečné, mnohoužívateľské prostredie pre efektívnu manipuláciu s veľkým objemom perzistentných dát (aj v prípade neočakávaných porúch)

Cieľ: dosiahnuť **nezávislosť aplikačných programov od dát**
Napríklad, výber programovacieho jazyka pre tvorbu aplikácie nemá závisieť od internej reprezentácie dát

?–1960

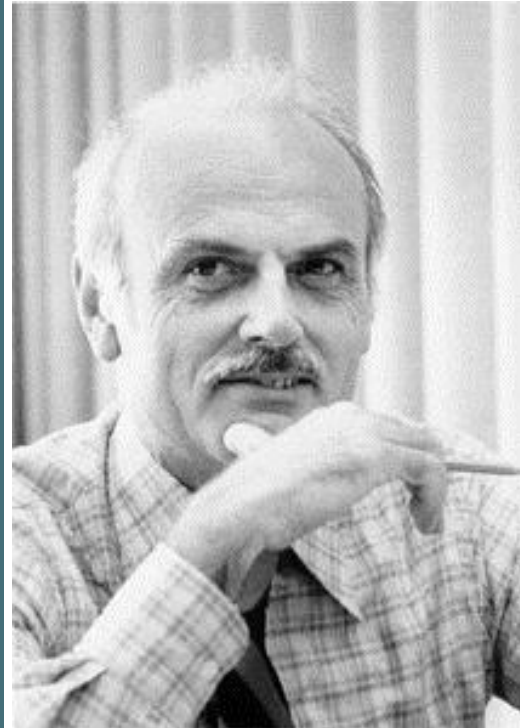
- Jednoúčelové, navzájom nekompatibilné aplikácie (banky, knižnice, letecké spoločnosti, byrokracia veľkých firiem)
- **Práca s dátami „zadrôtovaná“ do štruktúr zvoleného programovacieho jazyka**, množstvo formátov súborov, súborových systémov, počítačov, pamäťových médií, operačných systémov, ...

1960–1970

- Návrhy **jednotných DBMS** (Database Management Systems):
 - hierarchický (stromový) dátový model, *à la* albumy-pesničky
 - sieťový (grafový) dátový model → štandard **CODASYL**, Committee on Data Systems and Languages
- Problém s CODASYL: ako automaticky prechádzať (prehľadávať) veľký graf? → potreba „programovacieho jazyka vyššej úrovne“

1970–dnes

- Edgar F. Codd: A Relational Model of Data for Large Shared Data Banks (IBM, 1970). Idea 1: každá dátová štruktúra sa dá reprezentovať **reláciou** (tabuľkou). Idea 2: s malou, vhodne zvolenou množinou **tabuľkových operácií** sa dá „urobiť všetko“. Toto sa dá presne formalizovať a matematicky dokázať → **SQL**



Edgar F. „Ted“ Codd, 1923–2003, British Computer Scientist, born in Portland, Dorset

- Studied maths and chemistry at Oxford
- Was a pilot in the Royal Air Force during WWII
- Joined IBM in New York as a mathematical programmer in 1948
- Earned a doctorate in CS from the University of Michigan in Ann Arbor, then joined IBM research in San Jose

1986–dnes

- SQL, Standard Query Language, ISO a ANSI štandard od 1986, niekoľkokrát aktualizovaný (rozšírený). Významná aktualizácia bola v 1999
 - **relačný dátový model**
 - príkazy na vytvorenie typov atribútov, vytvorenie relácií atď. (DDL, Data Definition Language)
 - kľúčový príkaz DML (Data Manipulation Language) je **SELECT**, ktorým sa formulujú dotazy
 - transakčné príkazy (BEGIN / COMMIT / ROLLBACK)
 - ...

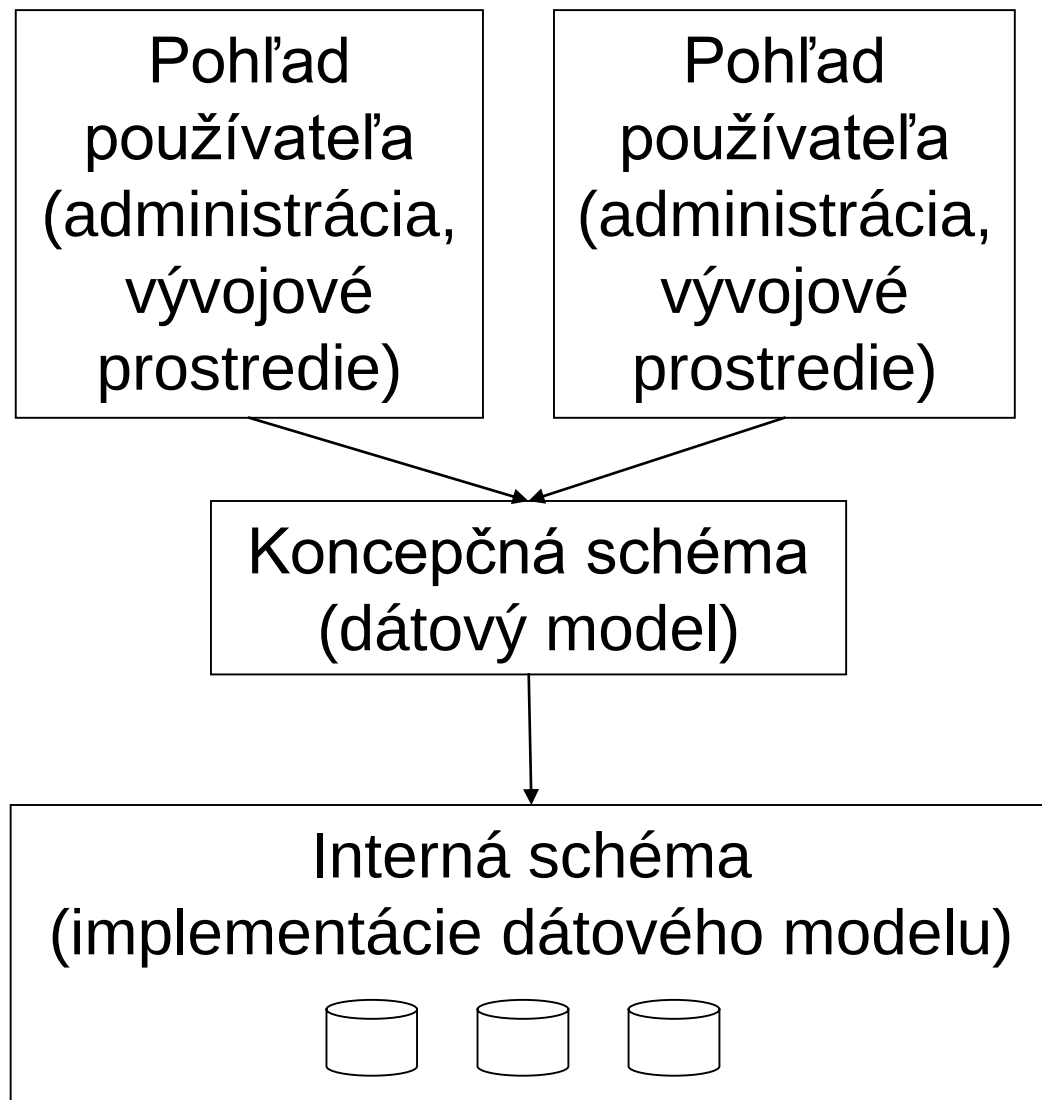
Praktické systémy nadržia tempo s "teoretickým" výskumom, ani so špecifikáciou SQL. Napr. rekurgia v SQL sa začala implementovať len nedávno

Ciel': nezávislosť aplikácií od internej reprezentácie dát

Tvorba a používanie konkrétnych aplikácií
v rôznych programovacích jazykoch, prístup k častiam dát, bezpečnosť, ...)

Dátový model
(napr. relačný)

Rôzne implementácie dátového modelu,
uloženie dát na perzistentných médiách, ...



Dátový model je matematická notácia dát a manipulácie s dátami (algebra)

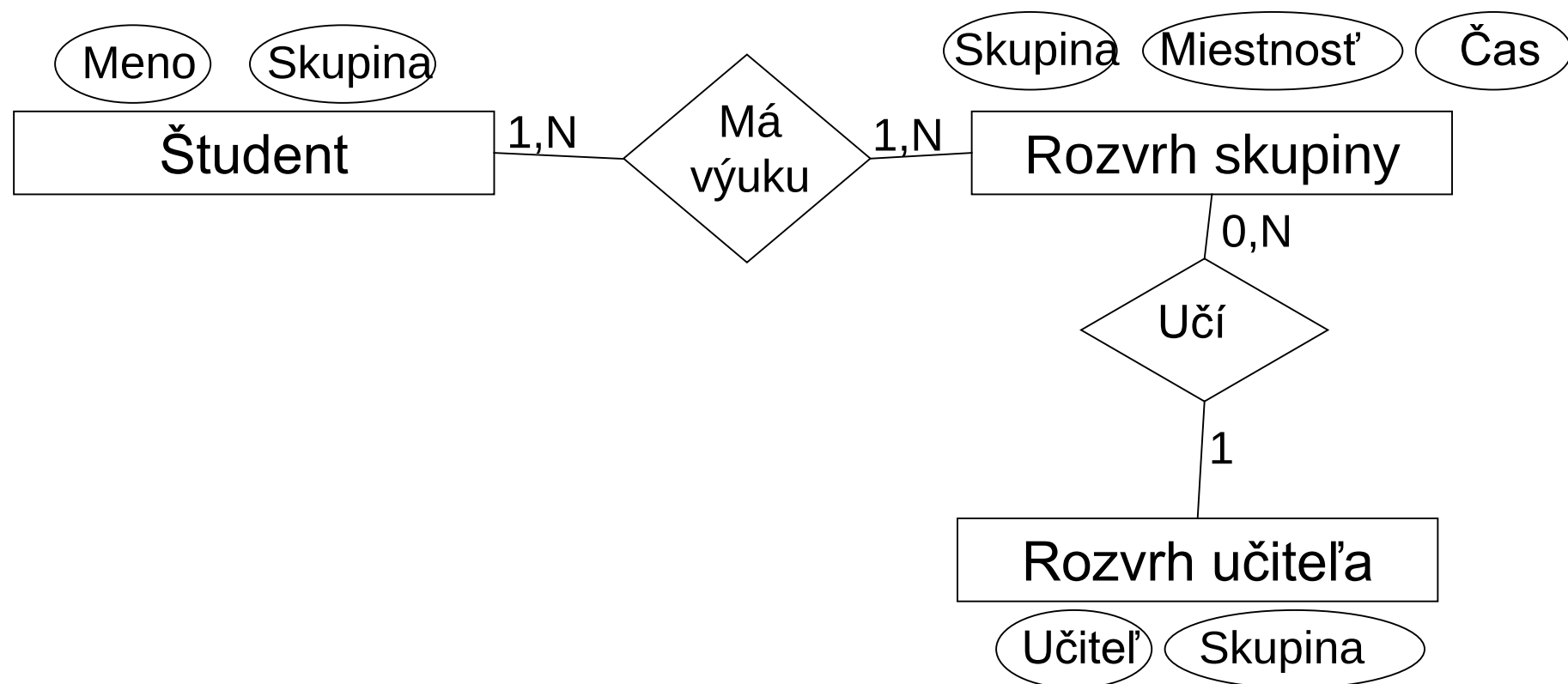
Používané dátové modely:

- **relačný**
- entitno-relačný
- navigačný (XML)
- objektový

Entitno-relačný dátový model

Entitno relačný model popisuje entity, atribúty entít a vzťahy medzi entitami. **Nemá však žiadne operácie** (je to len formálne presná notácia)

Príklad ER diagramu:



XML sa používa na popis štruktúrovaných textov. **Základnými konceptami sú [strom](#) (linearizovaný do textu správnym uzátvorkovaním tagov) a [odkaz](#) (link).** DTD (Document Type Definition) je jazyk, v ktorom sa popisuje gramatika tagov (t.j. syntakticky správne vnorenia tagov) pre daný typ dokumentu

XPath je jednoduchý jazyk na koncepcnej úrovni, jadrom sú operácie pre "manuálne" traverzovanie stromu:

- home
- up, down
- first_sibling, next_sibling, prev_sibling
- first_down, next_down, prev_down

XQuery je vyšší dotazovací jazyk na aplikačnej úrovni, ktorý používa XPath výrazy ako základ: FLWOR+HTML
(FLWOR je skratka pre *for*, *let*, *where*, *order by*, *return*)

Relačný dátový model

Jediným konceptom je **relácia** (tabuľka). Toto vymyslel Codd zhruba v 1970

Relačné operácie:

- zjednotenie, prienik, rozdiel
- kartézsky súčin
- selekcia
- projekcia
- ...

DBMS ako napr. Oracle, MySQL, MS SQL Server, IBM DB2, Postgres, Sybase/SAP atď. používajú relačný dátový model, preto sa im hovorí relačné DBMS

História databázových systémov (na Zemi)

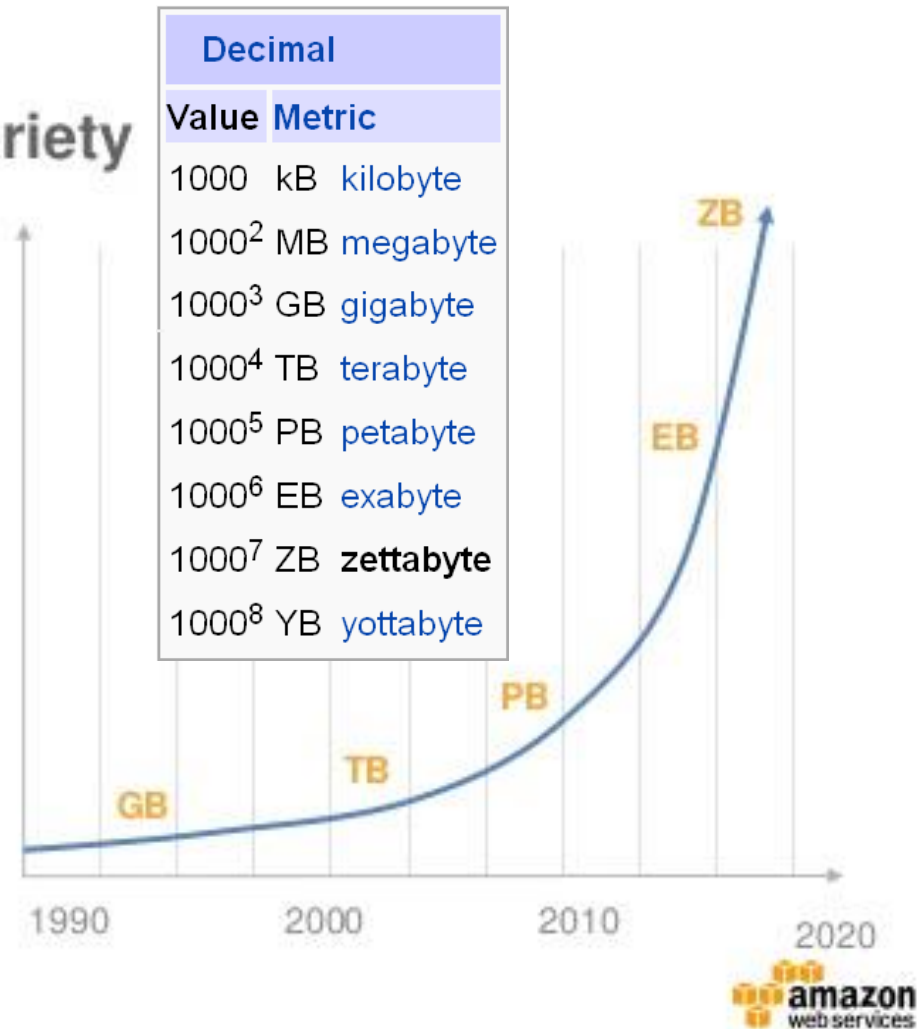
1990–dnes

CORBA CGI DB2 C++ V\$ Tables JDBI
PHP ABAP XML
z/OS C# SQL Derby Ruby ZPARMs
COBOL XML EJB Oracle Sybase ONDB
Unix Linux SQL server ODBC JSON Informix
JDBC VTAM DBEaver db4o DNS JODB
Perl VB DB2 connect Windows Java
PostgreSQL Python MQ HTTP MongoDB
Mckoi Scala TCP/IP ASP HTML PHP
Java Applet JDAL NoSQL
.NET MySQL DBPool Firebird

Nárast celkového objemu dát (na Zemi)

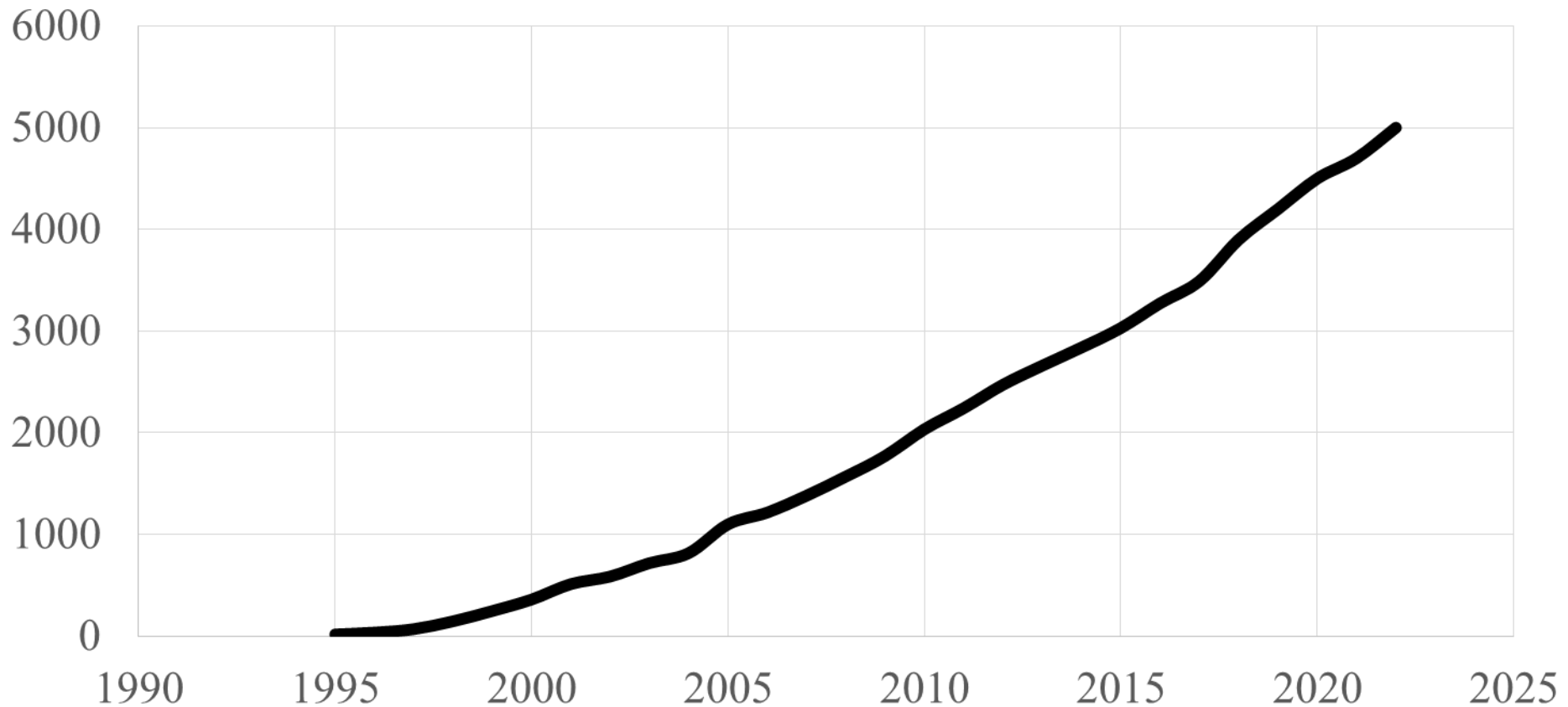
Data Volume, Velocity, & Variety

- 4.4 zettabytes (ZB) of data exists in the digital universe today
 - 1 ZB = 1 billion terabytes
- 450 billion transaction per day by 2020
- More unstructured data than structured data

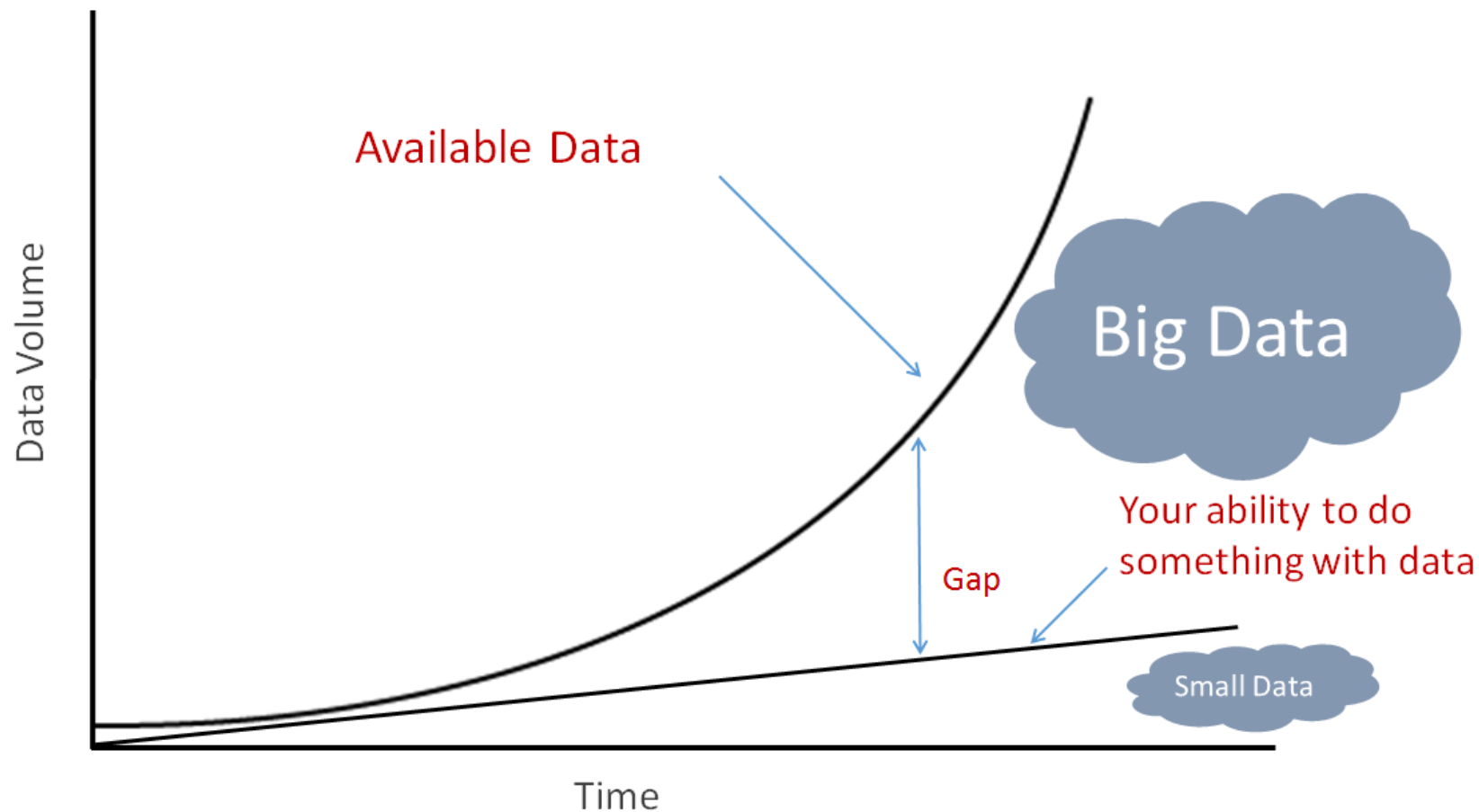


Nárast počtu používateľov Internetu (na Zemi)

Number of Internet Users in the World
in millions



Data Explosion

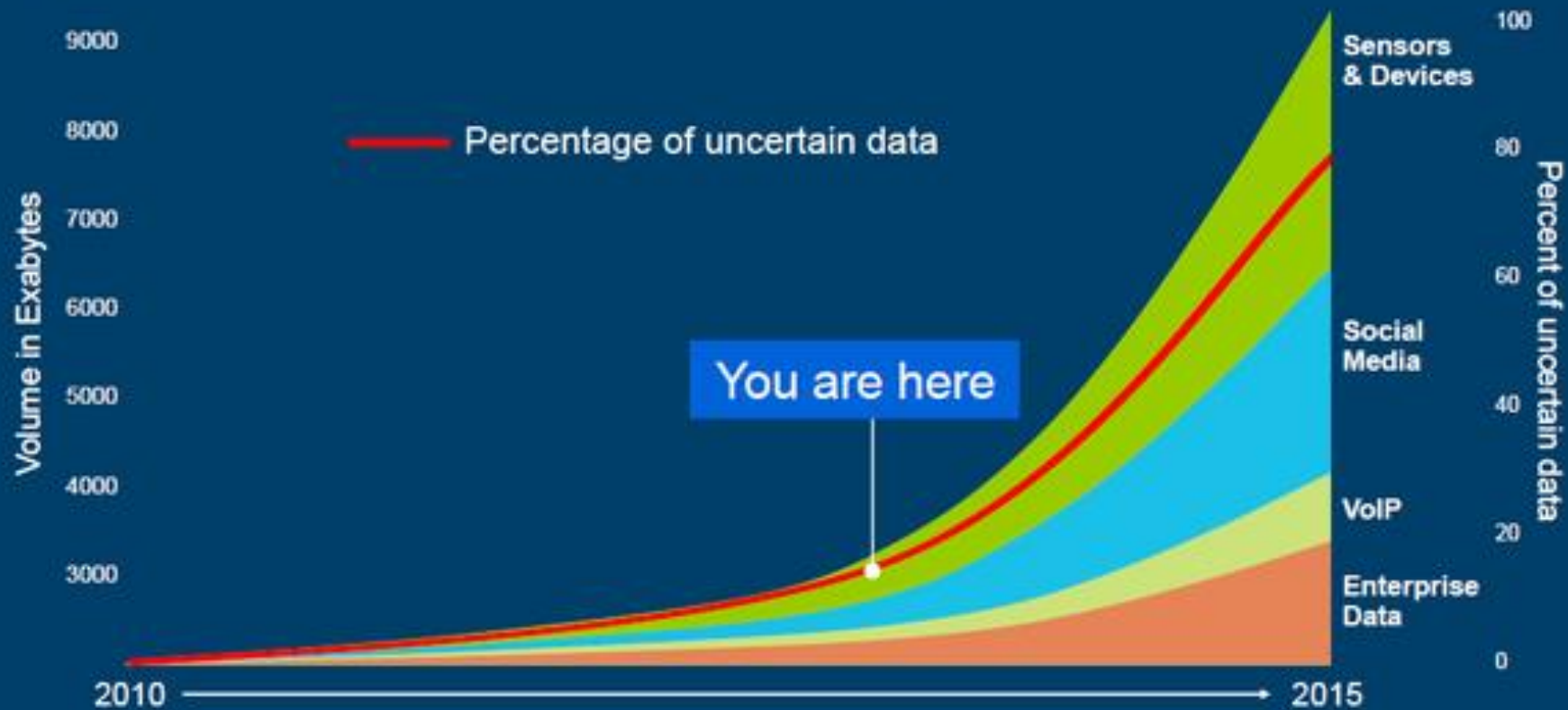


Distribúcia dát (na Zemi)

IBM Investor Briefing



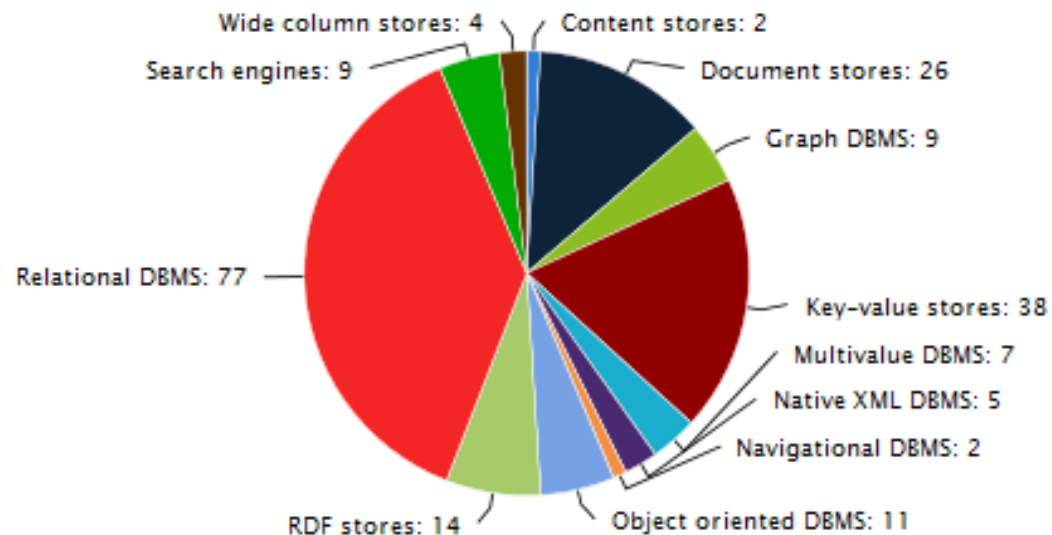
Big Data: This is just the beginning



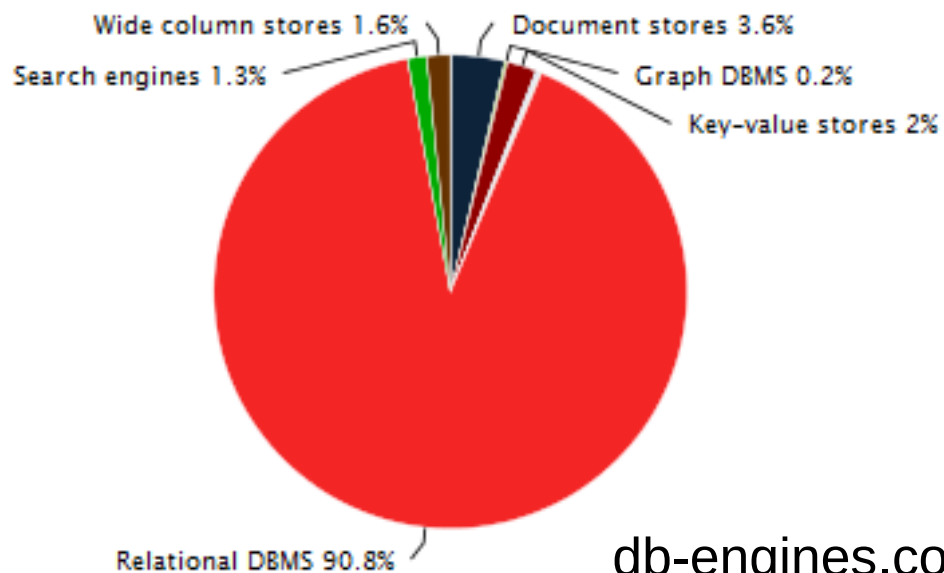
© 2013 International Business Machines Corporation

4

Podiel systémov na organizáciu dát (na Zemi)



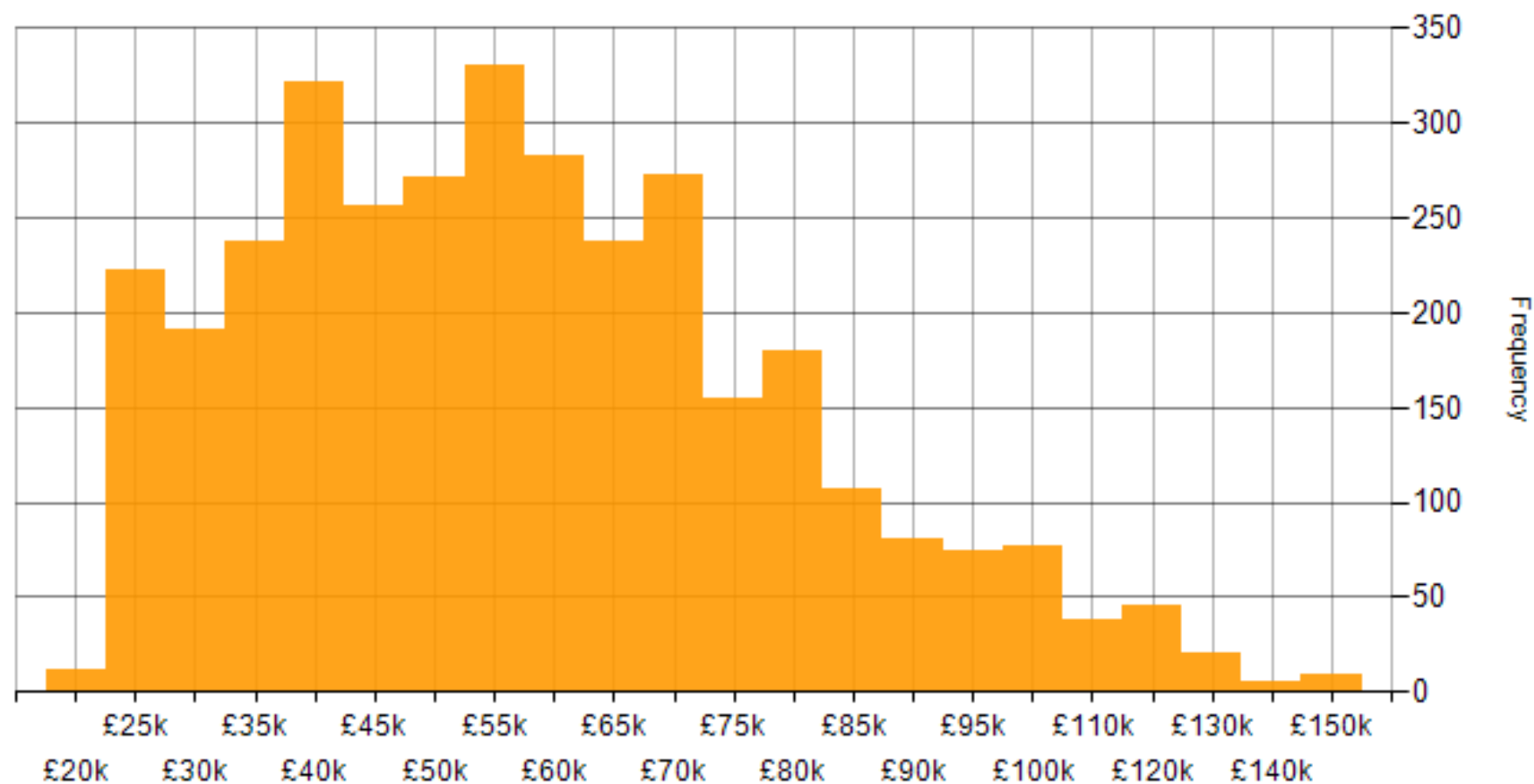
počet implementácií
(počet rôznych systémov)



popularita

db-engines.com

Platy v IT, SQL (v UK)



www.itjobswatch.co.uk

Relačný kalkul (logika 1. rádu)

	Relačný kalkul
Konštanta	$a, b, \text{jozo}, \dots, 1, 2, 3, \dots$
Premenná	$X, Y, Z, \dots$
Zložený term (funkčný symbol)	$f(t_1, t_2, \dots, t_n)$
Atomická formula (predikát)	$p(t_1, t_2, \dots, t_n)$
Konjunkcia	$p(\dots) \wedge q(\dots)$
Disjunkcia	$p(\dots) \vee q(\dots)$
Implikácia	$p(\dots) \Rightarrow q(\dots)$
Negácia	$\neg p(\dots)$
Existenčný kvantifikátor	$\exists X p(\dots, X, \dots)$
Všeobecný kvantifikátor	$\forall X p(\dots, X, \dots)$
Zátvorky	$(,)$

Čo hovorí táto formula (v prirodzenom jazyku)?

$\forall X (\text{osoba}(X) \Rightarrow (\exists Y \exists Z \text{rodic}(X, Y) \wedge \text{rodic}(X, Z)))$

„Každá osoba má aspoň jedného rodiča“

A čo hovorí táto formula (v prirodzenom jazyku)?

$\neg (\exists X \text{osoba}(X) \wedge \neg (\exists Y \text{rodic}(X, Y)))$

„Každá osoba má aspoň jedného rodiča“

Tentokrát sa podarilo ukázať (snád), že tieto dve formuly „vyjadrujú to isté“ (sú ekvivalentné)

Dá sa pre ľubovoľné dve formuly rozhodnúť či „vyjadrujú to isté“?

$\forall X (\text{osoba}(X) \Rightarrow (\exists Y \text{ rodic}(X, Y)))$

„Každá osoba má aspoň jedného rodiča“

$\text{osoba}(\cdot)$ a $\text{rodic}(\cdot, \cdot)$ sú predikáty. Pre konkrétne argumenty vrátia TRUE resp. FALSE. Všetky hodnoty, pre ktoré vracajú TRUE, uložíme do tabuliek (relácií)

Príklad naplnenia tabuliek:

osoba
adam
alica
eva

rodic	
alica	eva
alica	adam
adam	jozef

Čo je výsledkom horeuvedenej formuly pre túto databázu?

Dotazy v relačnom kalkule

$\{X: \text{osoba}(X) \wedge \exists Y \text{ rodic}(X, Y)\}$

„Osoby, ktoré majú aspoň jedného rodiča“

Obvykle nás zaujíma nie to, či nejaká formula platí, ale **pre ktoré hodnoty** voľných premenných platí

osoba
adam
alica
eva

rodic	
alica	eva
alica	adam
adam	jozef

Čo je výsledkom horeuvedenej formuly pre túto databázu?

Dotazy: ČO a AKO vypočítat'

assets			
Description	Producer	Model	Value
Microcomputer, 64kB RAM	Amstrad	CPC464	199
Microcomputer, 128kB RAM	Amstrad	CPC6128	299
Disk interface	Amstrad	DDI	149
Printer	Amstrad	DMP2000	159
Executive briefcase	Antler	AT0109	42
Wire paper clip	British Steel	BWC	1
Answering machine	British Telecom	BT2836	185
Photocopier	Canon	PC10	650
Dictation machine	Philips	DM510	190
Coffee maker	Philips	HD5349	30

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

V Pascale či **v C** vyjadrujeme **ako vypočítat'** výsledok. Ak 1 až N sú čísla riadkov tabuľky (poľa) *assets*, tak výsledkom je množina riadkov, ktoré vypíše tento program:

```
for i = 1 to N
if assets[i].Producer = 'Amstrad' and assets[i].Value < 200 then
    print(assets[i].Model, assets[i].Description)
```

Ešte pred napísaním tohto programu nám muselo byť jasné, **čo chceme vypočítat'**. Napríklad, museli sme si vyjasniť, že „výrobok“ znamená dvojicu [Model, Description]. A museli sme pochopiť, že dotaz znamená „Vyber usporiadané dvojice [Model, Description] z takých riadkov tabuľky *assets*, v ktorých Producer = 'Amstrad' a zároveň Value < 200.“

Dotazy: ČO a AKO vypočítat

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

Relačný kalkul umožňuje vyjadriť **čo treba vypočítat**: {[Model, Description]:

$\exists \text{Value assets(Description, 'Amstrad', Model, Value) } \wedge \text{Value} < 200\}$

Datalog je programovací jazyk, ktorý je "rozumným" zúžením relačného kalkulu. Aj v Datalogu vyjadrujeme (iba) **čo treba vypočítat**:

answer(Model, Description) ←

assets(Description, 'Amstrad', Model, Value), Value < 200.

Syntax Datalogu je veľmi jednoduchá (ešte jednoduchšia než syntax relačného kalkulu), vyjadrovacia sila veľká

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

Aj v **SQL** vyjadrujeme (iba) **čo treba vypočítat'**:

```
select Model, Description
```

```
from assets
```

```
where Producer = 'Amstrad' and Value < 200;
```

Výsledkom dotazu (akéhokoľvek) je opäť relácia (tabuľka)

Syntax SQL je bohatá, čo možno spohodľňuje prácu programátorom.

Komplikuje však prácu tvorcom kompilátorov a optimalizačných algoritmov
(a komplikuje porozumenie dotazom)

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

K „baroknej“ syntaxi SQL. Nasledujúce dotazy vyjadrujú to isté (malichernosti zanedbáme):

```
select Model, Description from assets  
where Producer = 'Amstrad' and Value < 200;
```

```
select Model, Description from assets where Producer = 'Amstrad'  
intersect
```

```
select Model, Description from assets where Value < 200;
```

```
select Model, Description from assets  
where Producer = 'Amstrad' and [Model, Description] in  
(select Model, Description from assets where Value < 200);
```

Ktorý z horeuvedených spôsobov je výpočtovo najefektívnejší?

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

V **relačnej algebre** vyjadrujeme (iba) **ako vypočítat'** výslednú reláciu:

$\text{answer} = \pi_{\text{Model, Description}} (\sigma_{\text{Producer} = \text{'Amstrad'} \wedge \text{Value} < 200} (\text{assets}))$

Relačná algebra má niekoľko (málo) operátorov, ktoré zo vstupnej tabuľky resp. tabuliek vypočítajú výslednú tabuľku, napr.

π je **projekcia**. $\pi_{\text{Description, Model}} (\text{assets})$ z relácie *assets* vyberie stĺpce Description a Model

σ je **selekcia**. $\sigma_{\text{Producer} = \text{'Amstrad'} \wedge \text{Value} < 200} (\text{assets})$ z relácie *assets* vyberie riadky, ktoré spĺňajú uvedenú podmienku

$\times$ je **kartézsky súčin** dvoch relácií

$\cap$, $\cup$, - sú **množinové operácie** (relácie musia byť „kompatibilné“)

S týmto sa dá „urobiť všetko“. (Takmer.)

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktoré výrobky od Amstrad sú lacnejšie než £200?

Pozor, **rôzne „ekvivalentné zápisy dotazu“ v relačnej algebre znamenajú rôzne výpočty!**

Napríklad,

$\pi_{\text{Description, Model}} (\sigma_{\text{Producer} = \text{'Amstrad'} \wedge \text{Value} < 200} (\text{assets}))$

môže byť výpočtovo **oveľa efektívnejšie** než

$\pi_{\text{Description, Model}} (\sigma_{\text{Producer} = \text{'Amstrad'}} (\text{assets})) \cap \pi_{\text{Description, Model}} (\sigma_{\text{Value} < 200} (\text{assets}))$

Oba tieto dotazy počítajú rovnaký výsledok (pre akékoľvek naplnenie relácie assets), ale iným spôsobom. Keď operátory π , σ , $\cap$ implementujete trebárs ako funkcie v C, tak si uvedomíte rozdiel medzi tými dvomi výpočtami

Dotazy: ČO a AKO vypočítat'

assets			
Description	Producer	Model	Value
Microcomputer, 64kB RAM	Amstrad	CPC464	199
Microcomputer, 128kB RAM	Amstrad	CPC6128	299
Disk interface	Amstrad	DDI	149
Printer	Amstrad	DMP2000	159
Executive briefcase	Antler	AT0109	42
Wire paper clip	British Steel	BWC	1
Answering machine	British Telecom	BT2836	185
Photocopier	Canon	PC10	650
Dictation machine	Philips	DM510	190
Coffee maker	Philips	HD5349	30

assets(Description, Producer, Model, Value)

Ktorí výrobcovia dodávajú každý svoj výrobok za menej než £200?

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktorí výrobcovia dodávajú každý svoj výrobok za menej než £200?

Relačný kalkuľ:

$\{P: \forall D \forall M \forall V (\text{assets}(D, P, M, V) \Rightarrow V < 200)\}$

Vyjadruje táto formula to, čo treba?

Nie. Ak za P dosadíme takmer čokoľvek (napr. π), tak tá formula bude splnená

Správne je:

$\{P: (\exists D \exists M \exists V \text{ assets}(D, P, M, V)) \wedge$
 $(\forall D2 \forall M2 \forall V2 (\text{assets}(D2, P, M2, V2) \Rightarrow V2 < 200))\}$

Alebo, ekvivalentne:

$\{P: (\exists D \exists M \exists V \text{ assets}(D, P, M, V)) \wedge$
 $\neg (\exists D2 \exists M2 \exists V2 \text{ assets}(D2, P, M2, V2) \wedge V2 \geq 200)\}$

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktorí výrobcovia dodávajú každý svoj výrobok za menej než £200?

{P: ($\exists D \exists M \exists V$ assets(D, P, M, V)) $\wedge$

$\neg (\exists D2 \exists M2 \exists V2$ assets(D2, P, M2, V2) $\wedge V2 \geq 200)$ }

Datalog:

delivers_some_expensive_product(P) $\leftarrow$ assets(_, P, _, V), $V \geq 200$.

answer(P) $\leftarrow$ assets(_, P, _, _), $\neg$ delivers_some_expensive_product(P).

SQL:

select a.Producer

from assets a

where not exists

(select * from assets a2

where a.Producer = a2.Producer and a2.Value ≥ 200);

Dotazy: ČO a AKO vypočítat'

assets(Description, Producer, Model, Value)

Ktorí výrobcovia dodávajú každý svoj výrobok za menej než £200?

{P: ($\exists D \exists M \exists V$ assets(D, P, M, V)) $\wedge$

$\neg (\exists D2 \exists M2 \exists V2$ assets(D2, P, M2, V2) $\wedge$ V2 $\geq$ 200)}

Relačná algebra:

delivers_some_expensive_product = $\pi_{\text{Producer}} (\sigma_{\text{Value} \geq 200} (\text{assets}))$

answer = $\pi_{\text{Producer}} (\text{assets}) - \text{delivers_some_expensive_product}$

Dotazy (queries): zhrnutie

Ľubovoľný „rozumný“ dotaz sa dá vyjadriť v jazyku matematickej logiky. Avšak matematická logika (1. rádu) umožňuje vyjadrovať aj dotazy, ktoré sa nedajú vypočítať—a už vôbec nie jednotným spôsobom

Datalog je zúžením logiky 1. rádu. Pre tzv. bezpečné Datalogové programy je garantované, že **výsledok ľubovoľného dotazu sa dá vypočítať jednotným spôsobom**; t.j. dá sa implementovať stroj, ktorý to robí (prekladá programy z Datalogu do relačnej algebry)

Ľubovoľný bezpečný Datalogový dotaz sa dá automaticky preložiť do SQL.
Použitie tohto algoritmu nevyžaduje žiadnu znalosť SQL

SQL dotazy v takejto forme sú vhodné pre optimalizáciu
implementovanú v praktických systémoch

- **H. Garcia-Molina, J.D. Ullman, J. Widom:** Database Systems, The Complete Book, Prentice Hall 2003
- **M. Kifer, P.A. Bernstein, P.M. Lewis:** Database Systems: An Application-Oriented Approach, Addison-Wesley 2006
- **R. Elmasri, S.B. Navathe:** Fundamentals of Database Systems, Addison-Wesley 2006 (online)
- **S. Abiteboul, R. Hull, V. Vianu:** Foundations of Databases (online)
- **J.D. Ullman, J. Widom:** A First Course in Database Systems, Prentice Hall 1997; <http://www-db.stanford.edu/~ullman/fcdb.html>
- C.J. Date: An Introduction to Database Systems, Addison-Wesley 2003
- ...

Stále aktuálna kniha o transakciách:

P.A. Bernstein, V. Hadzilacos, N. Goodman: Concurrency Control and Recovery in Database Systems, Addison-Wesley 1987, <http://research.microsoft.com/pubs/ccontrol>

Online materiály zo Stanford University:

<https://lagunita.stanford.edu/courses/DB/2014/SelfPaced/about>

Videozáznamy krátkych prednášok:

www.youtube.com (**J. Widom: Database Management**)

Československí autori: **J. Pokorný, A. Scheber, T. Delikát**

- **Časopisy:**
 - ACM TODS (Transactions on Database Systems)
 - ACM SIGMOD (Management of Data)
- **Konference:**
 - PODS: Principles of Database Systems
 - VLDB: International Conference on Very Large Data Bases
 - EDBT: International Conference on Extended Database Technology
 - FODO: International Conference on Foundations of Data Organization
 - SIAM Data Mining Conference
 - Datakon, Datasem

Centrálne postavenie databáz v informatike



Vývoj a aplikácie databázových systémov súvisia s mnohými oblasťami informatiky a matematiky

„Some of today’s most beautiful theoretical results in mathematical logic are in *finite model theory*, an area derived directly from database theory“

(T. Griffin, University of Cambridge, UK)

Obsah tohto predmetu

- Matematické základy relačného modelu (relácie, domény, atribúty, ...)
- **Dotazovacie jazyky** (relačný kalkul, Datalog, SQL, relačná algebra)
- Fyzická organizácia dát (realizácia výpočtu dotazov, indexy, ...)
- Modelovanie reality (návrh databáz)
- **Teória návrhu relačných databáz** (funkčné závislosti, kľúče, normálne formy, ...)
- **Transakcie** (operácie, rozvrhy, obnoviteľnosť, sériovateľnosť, ...)

Letný semester (predmet Databázy): sémantika dotazov, optimalizácia dotazov, distribuované databázy, ...

Cvičenia sa venujú modro označeným témam