Databázové systémy

Úvod do predmetu

Obsah prednášky

- Organizácia predmetu
- Database vs Database Management System
- Dátové modely
  - Relačný dátový model
- Relačná Algebra

Kontakt

- rastislav.bencel@stuba.sk
  - v štandardnom režime kancelária 5.30 (v súčasnosti home office)
- Možnosť kontaktu
  - E-mail
  - Slack

Cvičiaci

- Ing. Ján Balažia, PhD.
- Ing. Martin Binder
- Ing. Jakub Findura
- Ing. Viktor Lančarič
- Ing. Vladimír Kunštár

Prednášky

- Rastislav Bencel a Ján Balážia
- Prednášky budú nahrávané a dostupne na google drive
- Odporúčaná literatúra
  - https://www.db-book.com/db7/index.html

Cvičenia

- Formou konzultácií
  - Riešenie problémov vzniknutých počas vypracovaní zadaní
  - Pripravené úlohy, ktoré môžete riešiť a konzultovať
- K dispozícii príklady z riešenej problematiky v rámci prednášky
  - Možnosť konzultácie

Zadania

- 5 zadaní spolu za **50** bodov
- Odovzdania - viď harmonogram (2., 4., 6., 9., 12. týždeň)
- Všetky časti nadväzujú na seba (primárne na prvú časť zadania)
- Potrebné viesť si svoj privátny github repozitár v rámci github classroom
  - Pozvánka v priebehu prvého týždňa
- Odovzdanie – podľa pokynov v rámci daného zadania.
- Zadania budú kontrolované na kopírovanie

Podvádzanie

- Všetky zadania je potrebné vypracovať samostatne a v prípade zistenia akéhokoľvek nečestného konania búde študent hodnotený známkou **FX**

Podmienky absolvovania

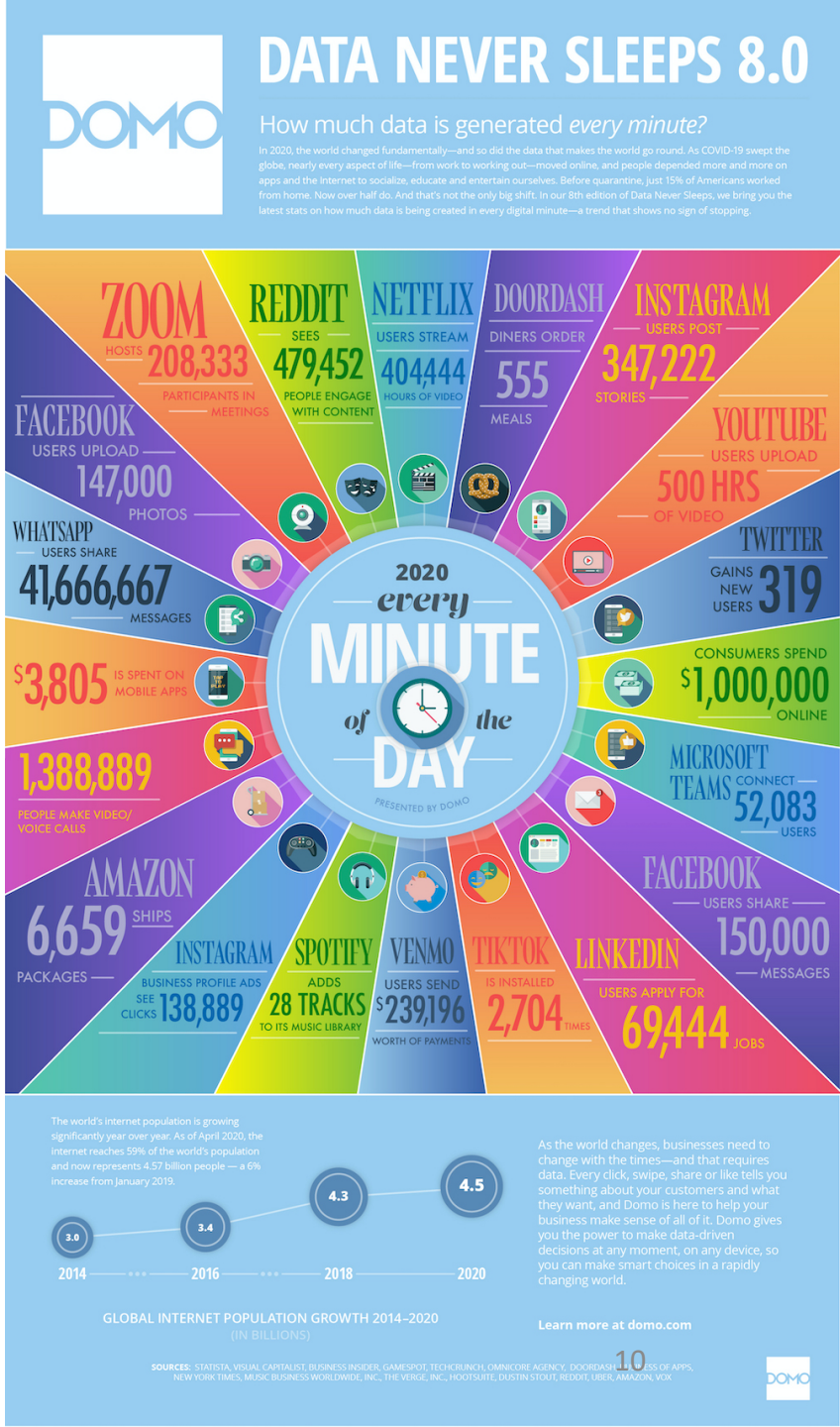
- Za každé zadanie je potrebné získať nenulový počet bodov a spolu za semester minimálne 25 bodov (50% z 50b za semester)
- Zo záverečného testu získať minimálne polovicu bodov z 50 bodov. (25b)
- Záverečná známka je udelená podľa štandardnej stupnice

Prečo databázy?

- Čo je databáza?
  - Je zbierka/kolekcia dát, ktoré obsahujú informácie

- Veľké množstvo dát, ktoré je potrebné niekde uchovať
  - Databáza ≠ DBMS (Database Managment System)

- Databázy predstavujú jeden z hlavných komponentov väčšiny aplikácií



Realizácia databázy

- Príklad: vytvorte databázu, ktorá bude slúžiť na evidovanie štatistík o hráčoch
  - Je potrebné uchovávať
    - Informácie o hráčoch
    - O sezónach, za ktoré tímy hrali daný hráči

- **Ako môžeme realizovať danú aplikáciu?**

Realizácia databázy - súbor

- Možnosť uloženia dát v súbore – jednotlivé informácie oddelené “delimiterom“ napr. čiarka v prípade .csv súboru
  - Pre každú entitu osobitný súbor
  - Aplikácia musí uskutočňovať zakaždým parsovanie pri potrebe načítaní/úprave údajov

Realizácia databázy - súbor

Player (name, country, position)

”Joe Sakic”, “Canada”, ”C”
”Peter Forsberg”, “Sweden”, ”C”
“Patric Roy”, ”Canada”, “G”

Season (year, name, team, number)

”2001”,”Joe Sakic”, “Colorado Avalance”, ”19”
”2001”,”Peter Forsberg”, “Colorado Avalance”, ”21”
“2005”,”Peter Forsberg”, ” Philadelphia Flyers”, “21”

Realizácia databázy - súbor

- Príklad: Z akej krajiny pochádza Peter Forsberg?

Player (name, country, position)

”Joe Sakic”, “Canada”, ”C”
”Peter Forsberg”, “Sweden”, ”C”
“Patric Roy”, ”Canada”, “G”

```
for line in file:
    record = parse(line)
    if record[0] == “Peter Forsberg“:
      print record[1]
```

Aké nedostatky môže mať takéto riešenie?

Realizácia databázy - súbor

- Integrita dát
  - Ako zabezpečíme, že meno hráča je rovnaké pre každý jeho záznam v dátach o sezóne?
  - Čo ak je prepísaný rok sezóny neplatným stringom napr. “dvetisícdesať“

Realizácia databázy - súbor

- Implementácia
  - Ako nájsť efektívne dáta?
  - Čo ak chceme vytvoriť novú aplikáciu, ktorá ma využívať rovnakú databázu
  - Čo v prípade, že dva procesy sa pokúšajú zapísať dáta do toho istého súboru v rovnakom čase ?

Realizácia databázy - súbor

- Durabillity (trvalosť)
  - Čo v prípade, že dôjde k výpadku počas zapisovania dát do súboru?
  - Čo ak chceme aby daná databáza bola replikovaná na viacerých strojoch pre zvýšenie dostupnosti ?

DBMS - Database Management System

- Predstavuje softvér, ktorý umožňuje aplikáciám pristupovať, ukladať a analyzovať informácie uložené v databáze
- Všeobecný účel DBMS je umožniť definovanie, vytváranie, modifikovanie a administrovanie DB

Prvotné DBMS

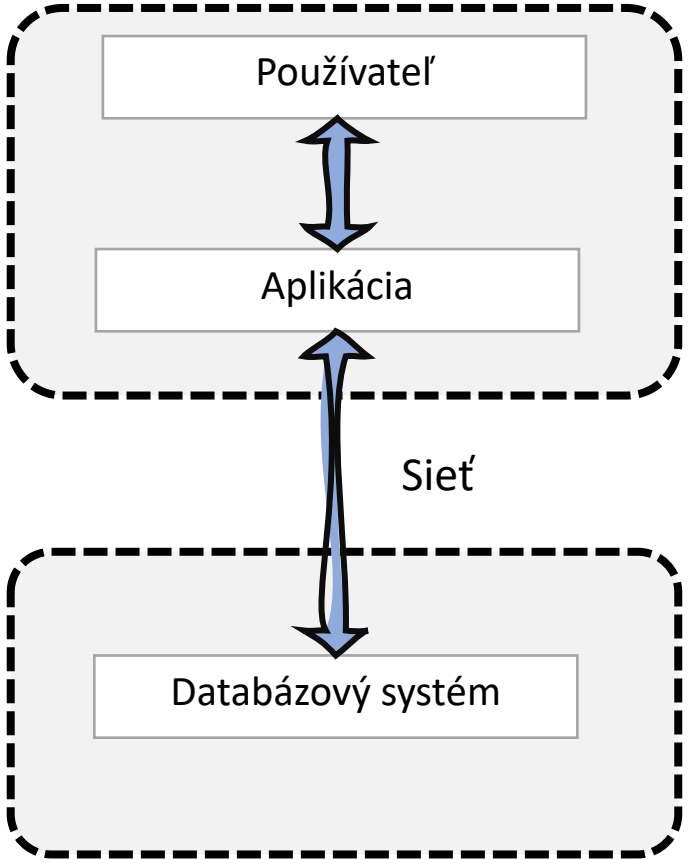
- Databázové aplikácie bolo náročné implementovať a udržiavať
- Silné prepojenie medzi logickou a fyzickou vrstvou
- Potreba znalosti aké dopyty budete uskutočňovať predtým ako implementujete danú aplikáciu

Typy DBMS

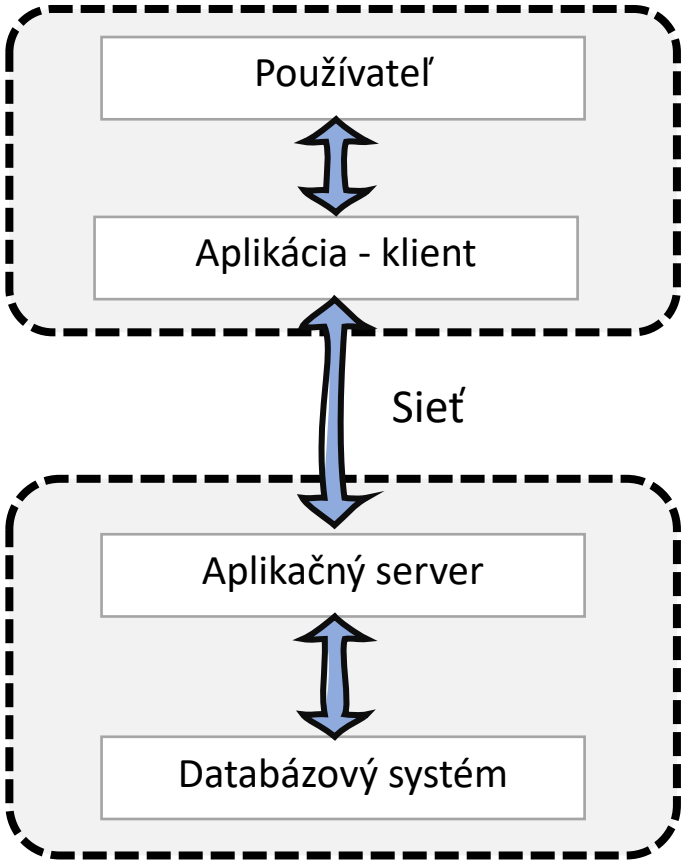
- Z pohľadu zaťaženia (Workload)
  - On-line Transaction Processing (OLTP)
    - Rýchle operácie, ktoré zakaždým pracujú nad malým množstvom dát
  - On-line Analytical Processing (OLAP)
    - Komplexnejšie operácie, ktoré čítajú naraz veľké množstvo dát za účelom rátania rôznych agregácií nad dátami

Databázové aplikácie - architektúra

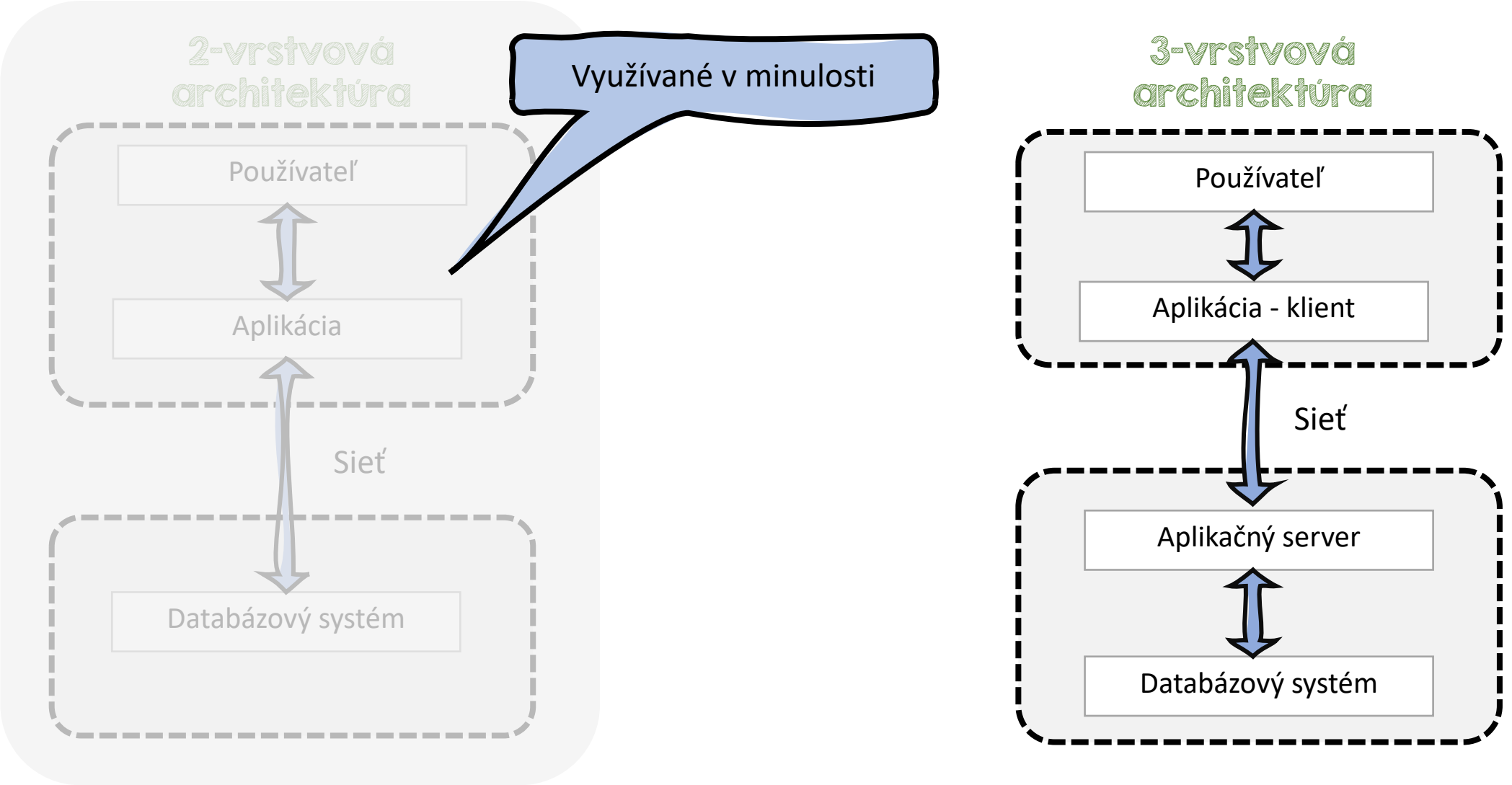
2-vrstvová architektúra



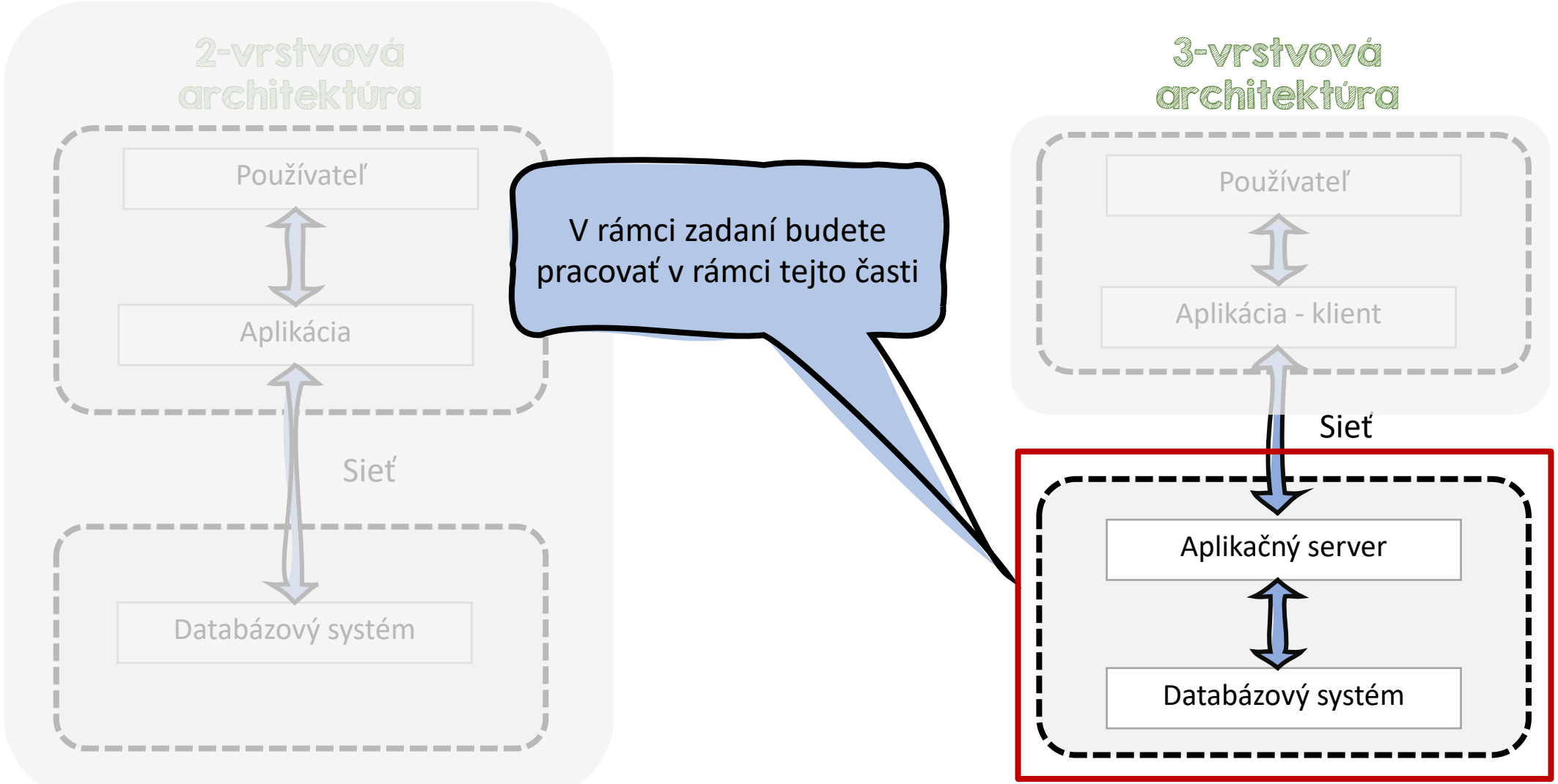
3-vrstvová architektúra



Databázové aplikácie - architektúra



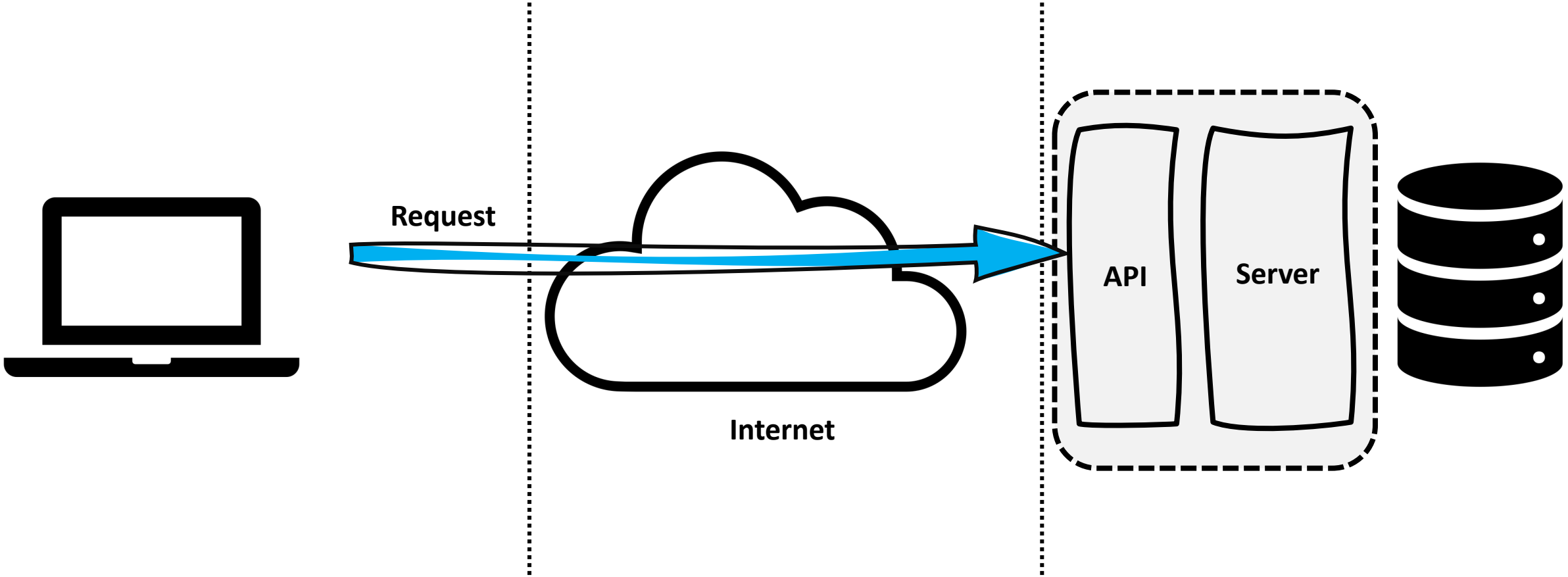
Databázové aplikácie - architektúra



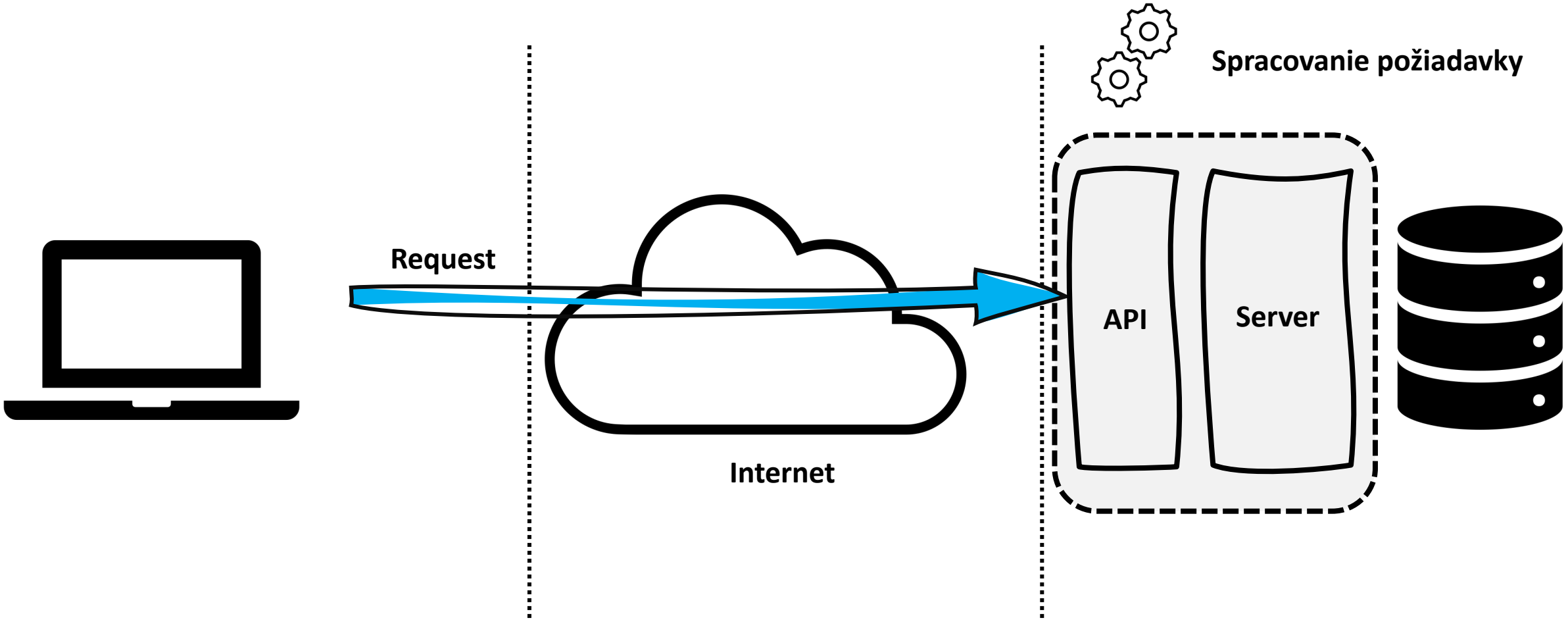
Server

- Zadanie URL do prehliadača môže mať dva významy
  - Predstavuje cestu k súboru na serveri
  - Predsavuje API end point na serveri
    - Podľa parametrov v URL obslúži server danú požiadavku a podľa požiadavky vygeneruje odpoveď pre požiadavku.
    - Generovanie odpovede môže obsahovať ďalších viacero procesov napr. Získanie informácií z databázy.

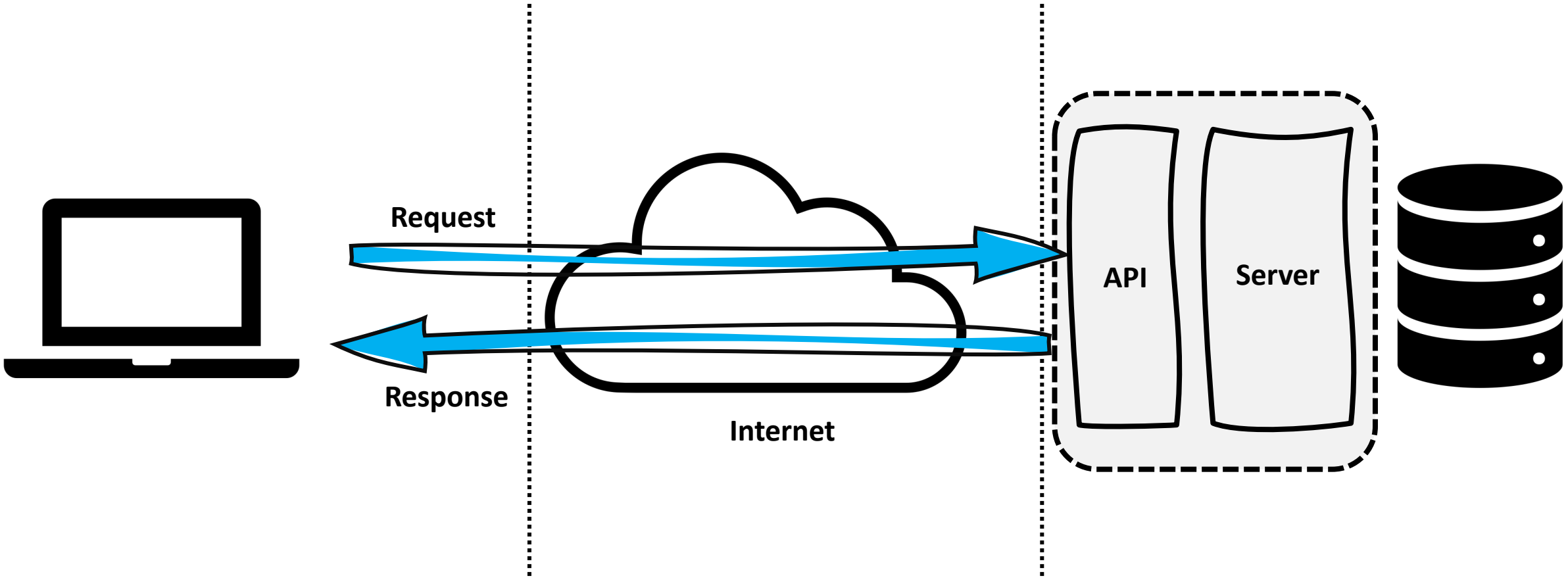
Komunikácia so serverom



Komunikácia so serverom



Komunikácia so serverom



Rest API

- **HTTP** metódy
  - **GET** – požiadavka o dáta. Nemodifikuje dáta na serveri.
  - **POST** – vytvorenie dát
  - **PUT** – aktualizovanie dát. Posiela všetky informácie aj tie, ktoré nie je nutné modifikovať
  - **PATCH** – aktualizovanie dát. Posiela iba informácie, ktoré je potrebné modifikovať
  - **DELETE** – vymazanie dát

- Rôzne stránky obsahujú takéto API pre 3rd party aplikácie, pre možnosti poskytnutia dodatočnej funkcionality

Priklad API

- https://docs.google.com/**document**/**d**/**1z-VlIT5B5q0kv2xu0GMPk7unqFiRhx4tmv56Oana1LA**
- **document** – hovorí serveru, že chceme dokument (word)
- **d/1z-VlIT5B5q0kv2xu0GMPk7unqFiRhx4tmv56Oana1LA** - hovorí serveru, že chceme dokument („d“), ktorý ma id **1z-VlIT5B5q0kv2xu0GMPk7unqFiRhx4tmv56Oana1LA**

JSON - JavaScript Object Notation

- Predstavuje štandard pre výmenu dát v čitateľnej forme pre človeka
- Serializovanie (Serialize) – zmena objektu do stringu, ktorý vieme následne deserializovať
- Deserializovanie (Deserialize) – zmena serializovaného stringu na objekt

- Váš aplikačný server v rámci zadania bude vracať dáta v tomto formáte

JSON - JavaScript Object Notation

- ukážka

Dátový model

- Predstavuje súbor princípov pre popis dát uchovávaných v databáze
- Schéma je samotný popis dát v rámci daného dátového modelu

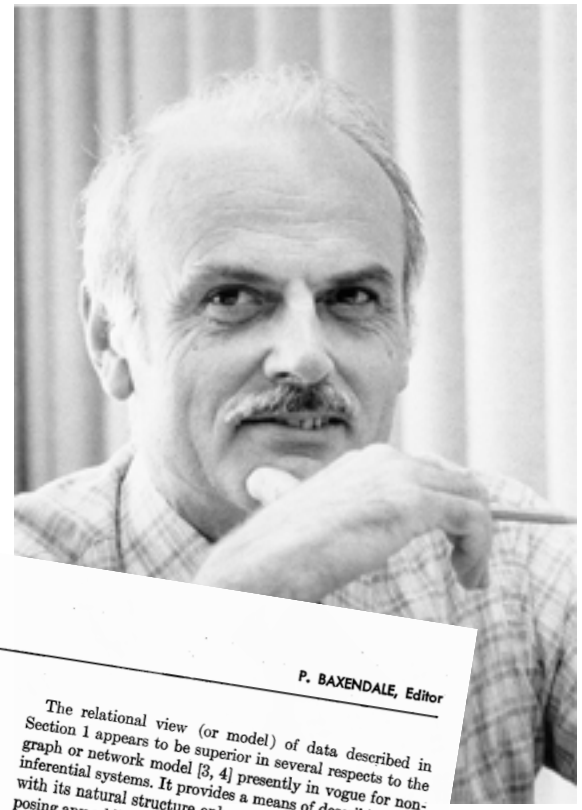
Dátový model

- Rôzne dátové modely
  - Relačný — Najviac používané
  - Key/Value
  - Graph
  - Document
  - Column family — NoSQL
  - Array-matrix — Strojové učenie (machine learning)
  - Hierarchical
  - Network — Historické (nechcete využívať)

Relačná dátový model

- Edgar Frank Codd

- https://dl.acm.org/doi/10.1145/362384.362685
- https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf



Information Retrieval P. BAXENDALE, Editor

A Relational Model of Data for Large Shared Data Banks

E. F. CODD
IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

Existing noninferential, formatted data systems provide users with tree-structured files or slightly more general network models of the data. In Section 1, inadequacies of these models are discussed. A model based on n-ary relations, a normal form for data base relations, and the

The relational view (or model) of data described in Section 1 appears to be superior in several respects to the graph or network model [3, 4] presently in vogue for non-inferential systems. It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation and organization of data on the other.

A further advantage of the relational view is that it forms a sound basis for treating derivability, redundancy, and consistency of relations—these are discussed in Section 2. The network model, on the other hand, has spawned a number of confusions, not the least of which is mistaking the derivation of connections for the derivation of relations (see remarks in Section 2 on the "connection trap").

Finally, the relational view permits a clearer evaluation of the scope and logical limitations of present formatted data systems, and also the relative merits (from the standpoint) of competing

Relačný dátový model

- **Relational model**
- Predstavuje neusporiadanú množinu
- Hlavné pojmy
  - Relácia/**Tabuľka** z angl. Relation
    - Každá obsahuje unikátne meno
  - Primárny kľúč
- Tvoria ju tabuľky

- Databázová schéma vs inštancia databázy

Relačný dátový model - primárny kľúč

- Primárny kľúč jednoznačne identifikuje záznam danej tabuľky
- Niektoré DBMS vedia automatický generovať interný primárny kľúč v prípade, že ho nedefinujete.

- DBMS obsahujú automatické generovanie integer hodnoty
  - Štandard SQL:2003 - Sequence
  - MySQL – Auto_increment
  - PostgreSQL - serial

id	name	country	year
	J.R.R. Tolkien	England	1892
	Andrzej Sapkowski	Poland	1948
	Andrzej Sapkowski	Czech Republic	1957

Relačný dátový model - primárny kľúč

- Primárny kľúč jednoznačne identifikuje záznam danej tabuľky
- Niektoré DBMS vedia automatický generovať interný primárny kľúč v prípade, že ho nedefinujete.

- DBMS obsahujú automatické generovanie integer hodnoty
  - Štandard SQL:2003 - Sequence
  - MySQL – Auto_increment
  - PostgreSQL - serial

id	name	country	year
123	J.R.R. Tolkien	England	1892
124	Andrzej Sapkowski	Poland	1948
125	Andrzej Sapkowski	Czech Republic	1957

Relačný dátový model - cudzie kľúče

- Mapuje atribút z jednej tabuľky do na atribút inej tabuľky
- Vytvorené prepojenia/vzťahu medzi tabuľkami

Relačný dátový model - cudzie kľúče

author (id, name, country, year)

id	name	country	year
50	J.R.R. Tolkien	England	1892
51	Andrzej Sapkowski	Poland	1948
52	Avi Silberschatz	null	null
53	Henry F. Korth	null	null
54	S. Sudarshan	null	null

book (id, title, year)

id	name	language	year
1	The Lord of the Rings	english	1954
2	The Witcher: Sword of Destiny	polish	1992
3	Database System Concepts: Seventh Edition	english	2019

Relačný dátový model - cudzie kľúče

author_book(author_id, book_id)

author_id	book_id

author (id, name, country,year)

id	name	country	year
50	J.R.R. Tolkien	England	1892
51	Andrzej Sapkowski	Poland	1948
52	Avi Silberschatz	null	null
53	Henry F. Korth	null	null
54	S. Sudarshan	null	null

book (id, title, year)

id	name	language	year
1	The Lord of the Rings	english	1954
2	The Witcher: Sword of Destiny	polish	1992
3	Database System Concepts: Seventh Edition	english	2019

Relačný dátový model - cudzie kľúče

author_book(author_id, book_id)

author_id	book_id
50	1
51	2
52	3
53	3
54	3

author (id, name, country, year)

id	name	country	year
50	J.R.R. Tolkien	England	1892
51	Andrzej Sapkowski	Poland	1948
52	Avi Silberschatz	null	null
53	Henry F. Korth	null	null
54	S. Sudarshan	null	null

book (id, title, year)

id	name	language	year
1	The Lord of the Rings	english	1954
2	The Witcher: Sword of Destiny	polish	1992
3	Database System Concepts: Seventh Edition	english	2019

Relačný dátový model - cudzie kľúče

author_book(author_id, book_id)

author_id	book_id
50	1
51	2
52	3
53	3
54	3

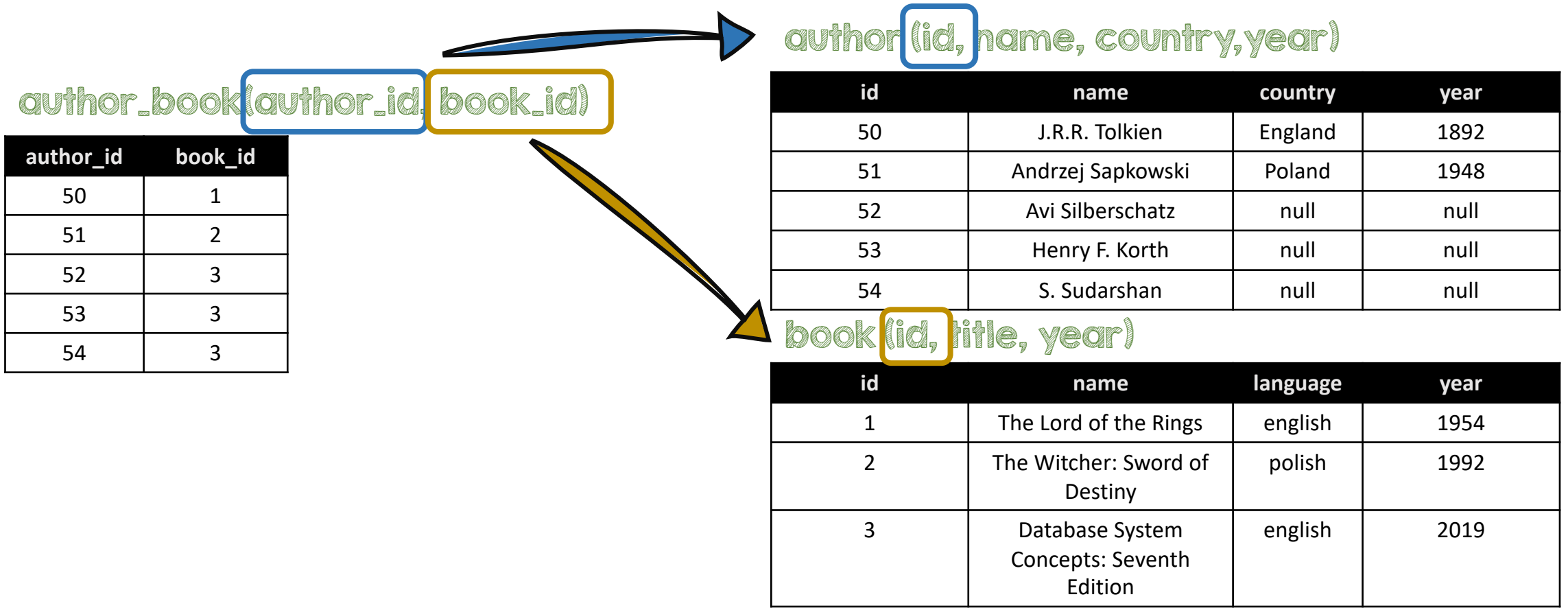
author (id, name, country, year)

id	name	country	year
50	J.R.R. Tolkien	England	1892
51	Andrzej Sapkowski	Poland	1948
52	Avi Silberschatz	null	null
53	Henry F. Korth	null	null
54	S. Sudarshan	null	null

book (id, title, year)

id	name	language	year
1	The Lord of the Rings	english	1954
2	The Witcher: Sword of Destiny	polish	1992
3	Database System Concepts: Seventh Edition	english	2019

Relačný dátový model - cudzie kľúče



author_id	book_id
50	1
51	2
52	3
53	3
54	3

id	name	country	year
50	J.R.R. Tolkien	England	1892
51	Andrzej Sapkowski	Poland	1948
52	Avi Silberschatz	null	null
53	Henry F. Korth	null	null
54	S. Sudarshan	null	null

id	name	language	year
1	The Lord of the Rings	english	1954
2	The Witcher: Sword of Destiny	polish	1992
3	Database System Concepts: Seventh Edition	english	2019

Data Manipulation Language (DML)

- Definuje metódy pre ukladanie a získavanie informácií z databázy
- **Relačná algebra**
  - Procedurálny prístup
  - Dopyt do DB definuje iba plán ako sa má DBMS nájsť požadované dáta
- **Relačný kalkulus**
  - Neprocedurálny prístup
  - Dopyt hovorí iba aké dáta chceme získať a nie ako sa dajú nájsť
- **SQL**
  - Stavia na relačnej algebre a relačnom kalkule

Relačná algebra

- Formálny základ pre operácie nad reláciami
  - Využíva sa napr. v oblasti optimalizácie exekučných plánov SQL dopytov
  - V prípade relačného modelu sú to tabuľky
  - Základ teórie implementácie SQL
- Sadá operácií nad reláciami umožňujúcich vyjadrovať dopyty
  - Dopyt (výraz) nad množinou relácií ako výsledok produkuje opäť reláciu (komponovateľnosť)

Relácia v relačnej algebre

- Zodpovedá tabuľke v SQL, ale bez duplikátov
  - SQL tabuľka je multimnožina – povoľuje duplikáty
  - N-tica relácie je vlastne riadok tabuľky

Príklad odstránenia duplikátov

- Ak urobíme výber iba určitých stĺpcov (Produkt, Cena) v rámci RA

Relácia

Produkt	Predajňa	Cena
Lopta	Karlovka	20
Hokejka	Karlovka	40
Lopta	Petržalka	20
Hokejka	Petžalka	50

Relácia po výbere stĺpcov

Produkt	Cena
Lopta	20
Hokejka	40
Hokejka	50

- Bol odstránený jeden záznam “lopta“

Relačná algebra - operácie

- **Selection** (Selekcia) - výber riadkov
  - Označenie - σ
- **Projection** (Projekcia)
  - Výber stlpcov
  - Označenie - π
- **Množinové operácie**
  - Union (prienik),
  - Intersection (zjednotenie),
  - Difference (rozdiel)
- **Product** – označovaný tiež ako cross join
  - Označenie: $\times$
- **Join**
  - Natural join - $\bowtie$
  - Theta join - $\bowtie_\theta$

Dopytovací výraz - názov relácie

- Predstavuje najjednoduchší platný dopytovací výraz v RA (Relačná algebra)
- Výsledkom je kopia danej relácie
- Príklad: Relácia Person

id	firstname	surname
1	Jozef	Mrkvička
2	Anton	Hruška

Selection - selekcia

- Umožňuje vybrať podmnožinu relácie na základe výberovej podmienky
  - **$\sigma_{<výberová\_podmienka>}(<relácia>)$**
  - unárna operácia
- Selekcia vyberá iba riadky z pôvodnej relácie, ktoré spĺňajú výberovú podmienku *<výberová_podmienka>*
- Štruktúra výslednej relácie je rovnaká ako relácia(*<relácia>*), ktorá vstupovala ako parameter selekcie
  - príklad: ak štruktúra bola (meno, priezvisko, vek) tak nová relácia má štruktúru (meno, priezvisko, vek)
- selekciu môžeme prirovnať k WHERE klauzule v rámci SQL

Selection - priklad

- Relácia osoba

Id	Meno	Priezvisko	Všp
1	Jozef	Mrkvička	1.5
2	Anton	Hruška	2.3
3	Ferdinand	Mrkvička	3.5

- $\sigma_{id=3}(osoba)$
- $\sigma_{všp<3}(osoba)$

Selection - priklad

- Relácia osoba

Id	Meno	Priezvisko	Výp
1	Jozef	Mrkvička	1.5
2	Anton	Hruška	2.3
3	Ferdinand	Mrkvička	3.5

- $\sigma_{id=3}(osoba)$

Id	Meno	Priezvisko	Výp
3	Ferdinand	Mrkvička	3.5

- $\sigma_{výp<3}(osoba)$

Selection - priklad

- Relácia osoba

Id	Meno	Priezvisko	Všp
1	Jozef	Mrkvička	1.5
2	Anton	Hruška	2.3
3	Ferdinand	Mrkvička	3.5

- $\sigma_{id=3}(osoba)$

Id	Meno	Priezvisko	Všp
3	Ferdinand	Mrkvička	3.5

- $\sigma_{všp<3}(osoba)$

Id	Meno	Priezvisko	Všp
1	Jozef	Mrkvička	1.5
2	Anton	Hruška	2.3

Selection - vlastnosti

- Komutatívnosť

  $\sigma_{<\alpha>(}\sigma_{<\beta>(}R)) = \sigma_{<\beta>}(\sigma_{<\alpha>(}R))$

  - z hľadiska výsledku nezáleží na poradí vyhodnotenia

- Možnosť prepísania vnorených selekcií na jednu

  $\sigma_{<\alpha>}(\sigma_{<\beta>}(R)) = \sigma_{<\alpha \text{ AND } \beta>}(R)$

Projection - projekcia

- Umožňuje vybrať podmnožinu atribútov (stĺpcov) relácie
  - $\pi_{<a1, \ldots ,an>}(<relácia>)$
  - unárna operácia
- Výsledok je nová relácia, ktorá obsahuje iba atribúty, ktoré sa nachádzali v rámci projekcie (*<a1,...,an>*)
- Výsledná relácia ma novú štruktúru definovanú vymenovanými stĺpcami v rámci danej projekcie
- projekciu môžeme prirovnať k vymenovaniu stĺpcov v SELECT časti SQL dopyu

Projection - priklad

- relácia osoba

Id	Meno	Priezvisko	Všp
1	Jozef	Mrkvička	1.5
2	Anton	Hruška	2.3
3	Ferdinand	Mrkvička	3.5

- $\pi_{id,priezvisko}(osoba)$

Projection - priklad

- relácia osoba

Id	Meno	Priezvisko	Všp
1	Jozef	Mrkvička	1.5
2	Anton	Hruška	2.3
3	Ferdinand	Mrkvička	3.5

- $\pi_{id,priezvisko}(osoba)$

Id	Priezvisko
1	Mrkvička
2	Hruška
3	Mrkvička

Projection - vlastnosti

- Ak zoznám atribútov obsahuje iba nekľúčové atribúty (resp. nie unikatné), výsledok projekcie môže obsahovať duplicitné záznamy, ktoré sú však odstránené
  - pozri slide s príkladom odstránenia duplikátov
- Nie je komutatívna
  - $\pi_a(\pi_{a,b,c}(R)) != \pi_{a,b,c}(\pi_a(R))$
  - pravá časť nie je vykonateľná - nie je možné vybrať stĺpec(atribút) ***b,c*** , ktoré už nie sú obsiahnuté v rámci výsledku $\pi_a(R)$ – tu je výstup relácia, ktorá obsahuje iba atribút ***a***
- Prepis viacerých vnorených projekcií
  - $\pi_a(\pi_{a,b,c}(R)) = \pi_a(R)$
  - stĺpce ***b,c*** nie sú potrebné realizovať, pretože z nich je následne vybraný iba atribút ***a***, preto je možne nahradiť dve projekcie jednou

Product - cross join

- Umožňuje získať kombinácie všetkých n-tíc z ľavej relácie so všetkými n-ticami z pravej relácie
  - <relácia1> × <relácia2>
  - binárna operácia
- Výsledná relácia vzniká kopírovaním všetkých dvojíc riadkov *relácie1* a *relácie2*
- Výsledná relácia má štruktúru definovanú ako zreťazenie atribútov prvého operandu a atribútov druhého
- Je potrebné si dať pozor na spoločné atribúty (spoločný názov atribútu napr. id), keďže vo výslednej relácií musia byť rozlíšiteľne – nemôžu obsahovať rovnaký názov, nevieme potom rozlíšiť z kade pochádzajú
  - riešenie – vyhýbať sa spoločným atribútom/názvom alebo vo výslednej relácií použiť označenie ktoré ich rozlíši napr. relácia1.id a relácia2.id
- Karteziánsky súčin je v SQL vedený ako CROSS JOIN

Cross join - priklad

reziser

id	Meno	Priezvisko
1	Peter	Jackson
2	Christopher	Nolan

film

id	Film	Id_reziser
1	Pán Prsteňov	1
2	King Kong	1
3	Dark Knight	2

reziser × film

Cross join - priklad

reziser

id	Meno	Priezvisko
1	Peter	Jackson
2	Christopher	Nolan

film

id	Film	Id_reziser
1	Pán Prsteňov	1
2	King Kong	1
3	Dark Knight	2

reziser × film

reziser.id	Meno	Priezvisko	film.id	Film	Id_reziser
1	Peter	Jackson	1	Pán Prsteňov	1
1	Peter	Jackson	2	King Kong	1
1	Peter	Jackson	3	Dark Knight	2
2	Christopher	Nolan	1	Pán Prsteňov	1
2	Christopher	Nolan	2	King Kong	1
2	Christopher	Nolan	3	Dark Knight	2

Cross join + selekcia - priklad

- $\sigma_{reziser.id = film.id\_reziser}$ (osoba × dom_zviera)

Cross join + selekcia - priklad

- $\sigma_{reziser.id = film.id\_reziser}$ (osoba × dom_zviera)

reziser.id	Meno	Priezvisko	film.id	Film	id_reziser
1	Peter	Jackson	1	Pán Prsteňov	1
1	Peter	Jackson	2	King Kong	1
2	Christopher	Nolan	3	Dark Knight	2

Natural join

- Pri vytváraní kombinácií si vynucuje rovnosť všetkých atribútov s rovnakým menom (jeden z dvojice atribútov eliminuje)
  - <relácia1> ⋈ <relácia2>
  - binárna operácia
- nepridáva vyjadrovaciu silu RA (je ho možné prepísať pomocou CROSS JOIN)
  - je ho možne vyjadriť pomocou premenovania, theta-joinu a projekcie.
- výsledkom je relácia, kde sú skopírované všetky dvojice riadkov relácie1 a relácie2. Atribúty s rovnakým meno sú testované na rovnosť a sú kopírované iba raz.
- v SQL je to tiež NATURAL JOIN

Natural join - priklad

reziser

id_reziser	Meno	Priezvisko
1	Peter	Jackson
2	Christopher	Nolan

film

id	Film	id_reziser
1	Pán Prsteňov	1
2	King Kong	1
3	Dark Knight	2

reziser ⋈ film

Natural join - priklad

reziser

id_reziser	Meno	Priezvisko
1	Peter	Jackson
2	Christopher	Nolan

film

id	Film	id_reziser
1	Pán Prsteňov	1
2	King Kong	1
3	Dark Knight	2

reziser ⋈ film

id_reziser	Meno	Priezvisko	Id	Film
1	Peter	Jackson	1	Pán Prsteňov
1	Peter	Jackson	2	King Kong
2	Christopher	Nolan	3	Dark Knight

Theta join

- Umožňuje vytvárať kombinácie n-tíc z ľavej relácie s n-ticami z pravej pre ktoré platí podmienka
  - $<relácia1> \bowtie_\theta <relácia2> \equiv \sigma_\theta(<relácia1> \times <relácia2>)$
- Výsledok relácie je skopírovanie všetkých dvojíc riadkov z relácie1 a relácie2, ktoré spĺňajú podmienku θ
  - rovnako ako v prípade karteziánskeho súčinu je vhodné sa vyhnúť spoločným atribútom. V prípade použitia je však nutné používať prefixy mien relácií pre jednotlivé atribúty
- Tak ako prirodzený join (natural join), ani theta join nepridáva relačnej algebre vyjadrovaciu silu
  - ako je vyššie uvedené, tak je možné ho prepísať ako $\sigma_\theta(<relácia1> \times <relácia2>)$ – selekcia aplikovaná na karteziánsky súčin
- Obdoba INNER JOIN .... ON

Typy joinov

- Theta (θ) podmienka môže byť ľubovoľná podmienka, vtedy sa jedná o theta join
  - podmienka ako v selekcii
- Ak je podmienka na test rovnosti, zvykne sa hovoriť o equijoin-e
- Natural join je špecifický prípad equijoin-u

Premenovanie - ρ

- predstavuje štandardnú substitúciu v matematike. Premenovať možno mena atribútov ale aj samotné meno relácie
  - $\rho_{R(a1, \ldots, an)}(<relácia>)$
  - môže obsahovať viacero zápisov ako vyjadriť dané premenovanie – malo by byť však jasné, ktoré meno je pôvodne a ktoré nové.
- Výsledok premenovania je kópia relácie, ktorá sa môže volať inak alebo niektoré/niektorý z jej atribútov sa volajú inak.

Množinové operácie

- Štandardné množinové operácie
- Relácie musia byť kompatibilné
  - musia mať ten istý typ, t.j musia mať rovnakú relačnú schému
    - relácie musia mať rovnaký počet atribútov a každá dvojica zodpovedajúcich atribútov musí mať rovnakú doménu (obor hodnôt)

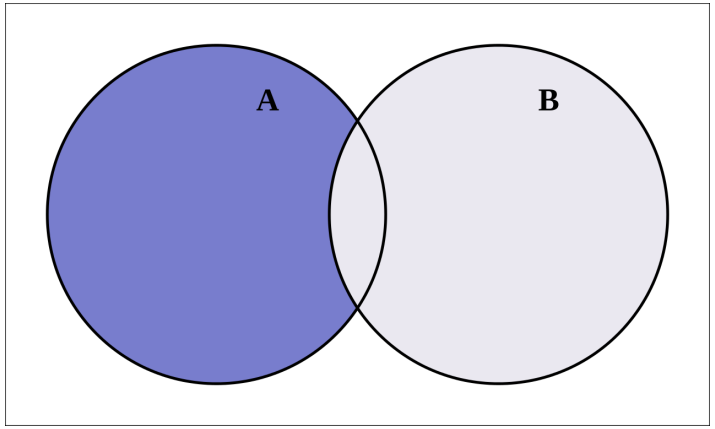
- Zjednotenie (UNION v SQL)
- Rozdiel (MINUS v SQL)
- Prienik (INTERSECT v SQL)

Prienik (INTERSECT v SQL)

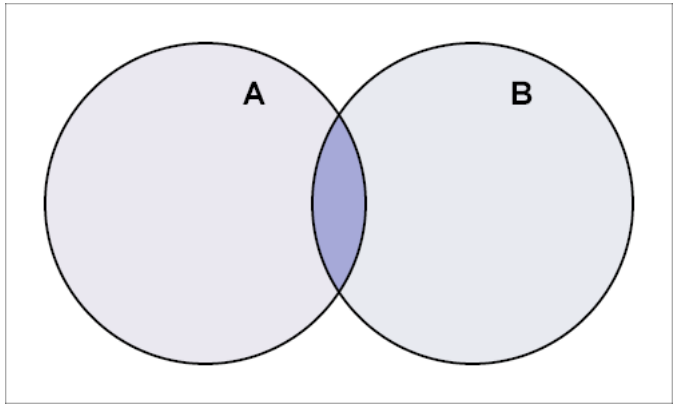
- $\pi_{meno}(osoba) \cap \pi_{meno}(dom\_zviera)$

- Nepridáva RA vyjadrovaciu silu
  - $A \cap B \equiv A - (A - B)$

- Vysvetlenie:
  - A– B je znázornené na obrázku 1 (horný obrázok)
  - Následne keď od A odrátame vyznačenú časť z horného obrázka dostaneme vyznačenú časť na obrázku 2 (dolný obrázok), čo predstavuje prienik



Obrázok 1



Obrázok 2

Zdroj: Wikipedia - Množina

- Relačná algebra nám umožňuje definovať kroky ako sa má daný dopyt rátať
  - $\sigma_{b\_id=1000}(R \bowtie S)$ vs. $(R \bowtie (\sigma_{b\_id=1000}(S))$
- Lepšie je zostať na High-level úrovni a nechať rozhodnúť DBMS, čo je výhodnejšie
  - Získať spojené riadky R a S, kde b_id = 1000

Relačný model - dopyty

```
for line in file:
   record = parse(line)
   if record[0] == “Peter Forsberg“:
     print record[1]
```

```
SELECT country FROM player
WHERE name = “Peter Forsberg“
```

Zhodnotenie

- Databázy sú všadeprítomné
- Relačný model umožňuje uskutočňovanie high-level dopytov
- Relačná algebra definuje možnosti spracovávania dopytov v rámci relačnej databázy

Ďalšia prednáška

- SQL

Otázky ?